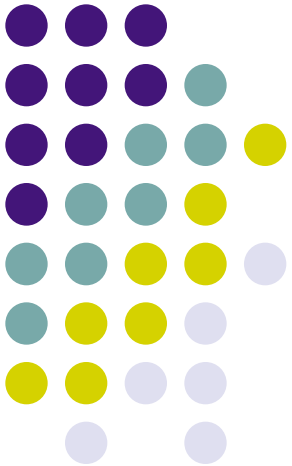
Interpretação de Linguagem Natural

Aula 8

O conteúdo destes slides foi adaptado de:

- **Curso "Da Linguagem Natural a Informação" – Prof. Emerson Cabrera Paraiso (PPGIa/PUCPR).**
- **"Natural Language Processing" course – Prof. Thamar Solorio (University of Houston).**
- **"Speech and Language Processing", Jurafsky, D., Martin, J.; 3a Edição (2018).**
- **"Natural Language Processing with Python - Analyzing Text with the Natural Language Toolkit", Bird, D., Klein, E., Loper, E. (2009).**



Pontifícia Universidade Católica do Paraná (PUCPR)

Bacharelado em Ciência da Computação – 4º Período

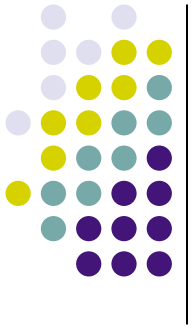
gregory.puppi@pucpr.br

Plano de Aula

- Contextualização
- Etapas Típicas do PLN
  - Normalização

Interpretação de Linguagem Natural

- Objetivo
  - Apresentar os fundamentos do Processamento de Linguagem Natural (PLN) e da Recuperação da Informação (RI).



Reflexão

- O que é uma máquina inteligente para você?

"A computer would deserve to be called intelligent if it could deceive a human into believing that it was human."

(Alan Turing)

Contextualização

- Permitir que uma máquina interprete um texto em linguagem natural é sem dúvida um dos maiores desafios da computação:
  - Textos em linguagem natural podem ser ambíguos, subjetivos, conter erros.
    - “A menina disse à colega que sua mãe havia chegado.”
    - “A vaca se diverte com a pata na lama.”
    - “Pode deixar, darei um geito.”
  - Neologismos:
    - “Retweet”
  - PLN é uma área de pesquisa interdisciplinar.

Mais detalhes em: https://en.wikipedia.org/wiki/History_of_natural_language_processing

Um Pouco de História

- O PLN começou a se desenvolver no início dos anos 1950.
- A primeira tarefa que chamou atenção foi a tradução automática:
  - Russo  Inglês
- Na década de 1960, Joseph Weizenbaum desenvolveu o ELIZA. ELIZA simula a conversação entre um humano e um computador, tentando “dar a impressão” ao humano de que entende o que este fala (no caso escreve).
- A partir dos anos 1980, sistemas baseados em regras começaram a proliferar.
- Surgem os parsers e as ontologias.
- Desenvolvimento do Aprendizado de Máquina auxiliou a evolução da área.

Por que o interesse recente?

- Há muita informação textual (dado não estruturado) acumulada na Web, nas empresas, nos computadores das pessoas.

Onde pode ter PLN aqui?

Aplicações

- Das mais simples:
  - Busca por palavra-chave
  - Identificação de sinônimos
  - Verificação da escrita (ortografia) – Extração da informação

- As mais sofisticadas:
  - Tradução automática
  - Reconhecimento e geração da fala – Sistemas de diálogo e Chatbots

Algumas Aplicações

- Recuperação de informação textual:
  - 6.586.013.574 buscas na web todo dia (estimativa de 2017)

Algumas Aplicações

- Extração da informação a partir de dados textuais:

I1 - Bom dia.
I2 - Bom dia.
I1 - Gostaria de uma informação.
I2 - Pois não, pode perguntar.
I1 - De quanto tempo é o estágio probatório?
I2 - O estágio probatório é de 3 anos contados a partir da data de posse. I1 - Obrigado
I2 - Sem problemas.

O diálogo tem um domínio específico!

Algumas Aplicações

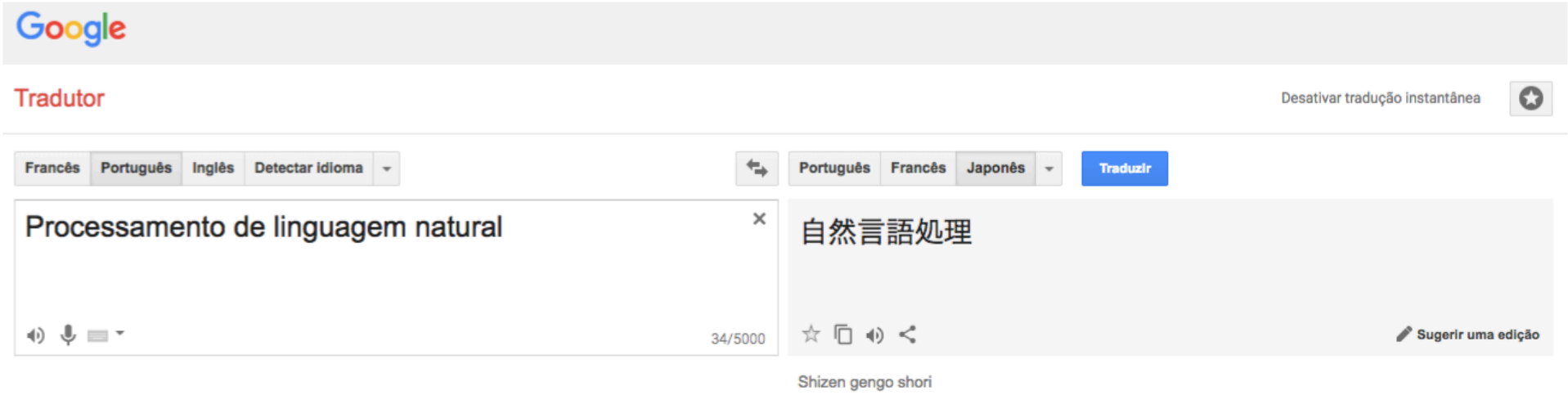
Notas de aula: Dan Jurafsky.

Algumas Aplicações

- Sistemas de recomendação

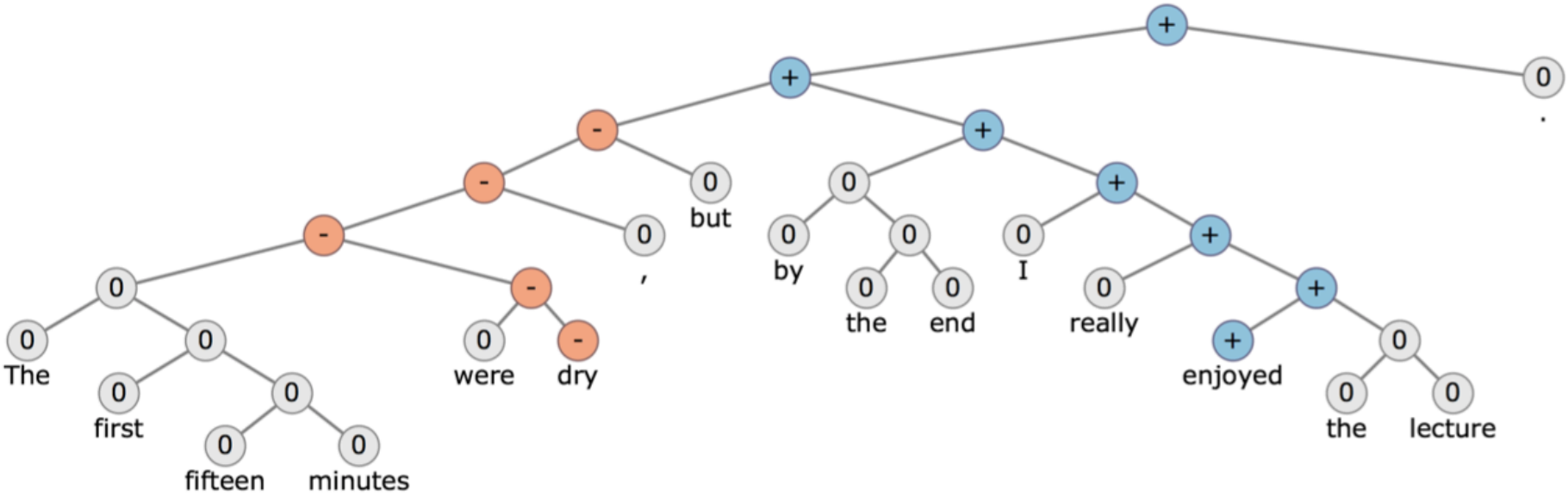
Algumas Aplicações

- Tradução



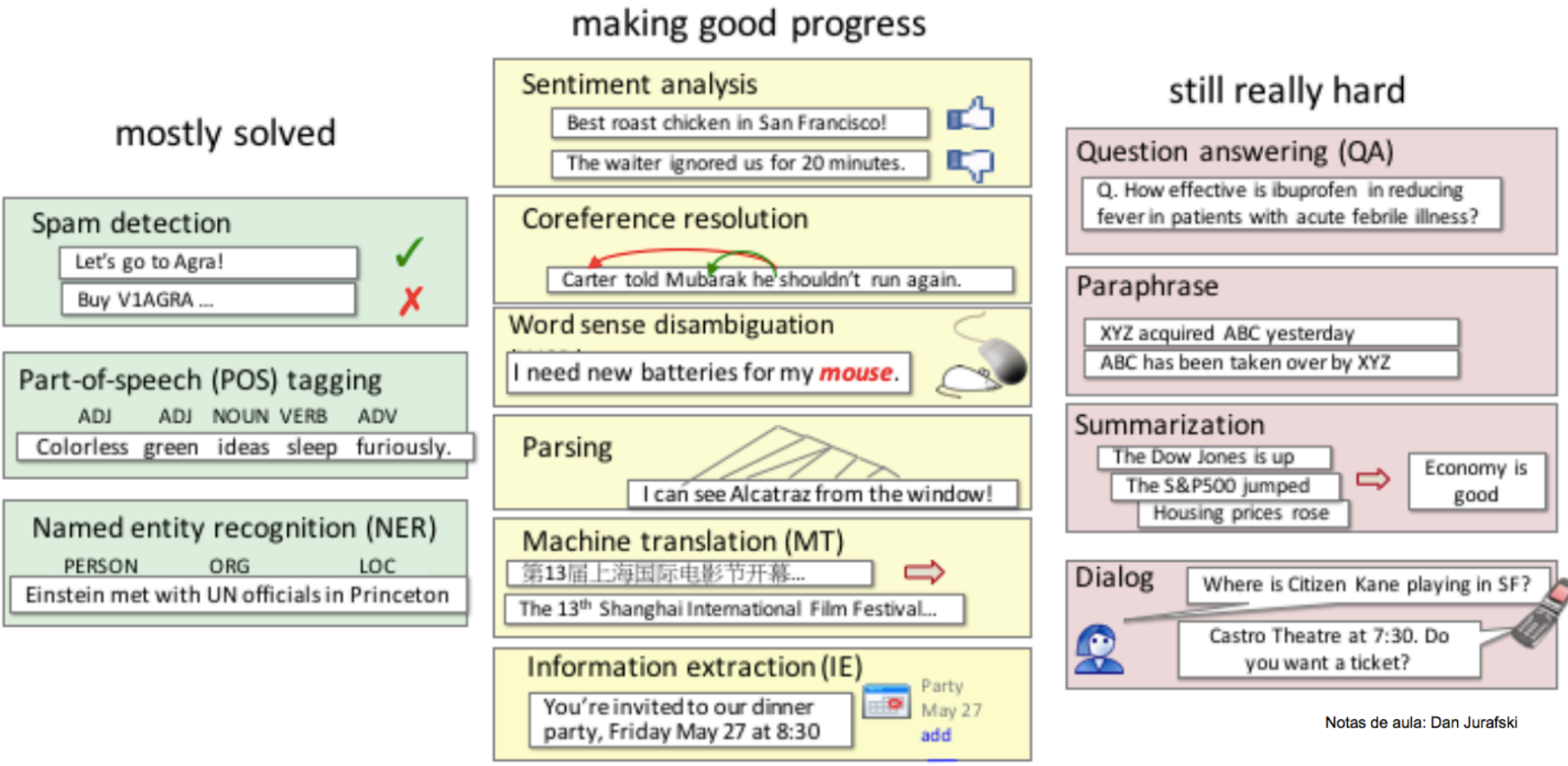
Algumas Aplicações

- Parsing



Extraído de: http://web.stanford.edu/class/cs224n/

Estado da Arte



Notas de aula: Dan Jurafski

Desafios para Processamento do Português

- Recursos mais limitados
  - Parser, part-of-speech, ...
  - Ontologias, dicionários
    - Brasileiro
    - Europeu
  - Reconhecimento da Fala
  - Corpora

Conceitos Básicos

- **Linguagem natural:** linguagens que são utilizadas para comunicação do dia a dia por humanos (português brasileiro, português europeu, inglês, ...).

- **Processamento de Linguagem Natural (PLN):** qualquer manipulação computacional de linguagens naturais. De contagem de palavras à compreensão semântica.

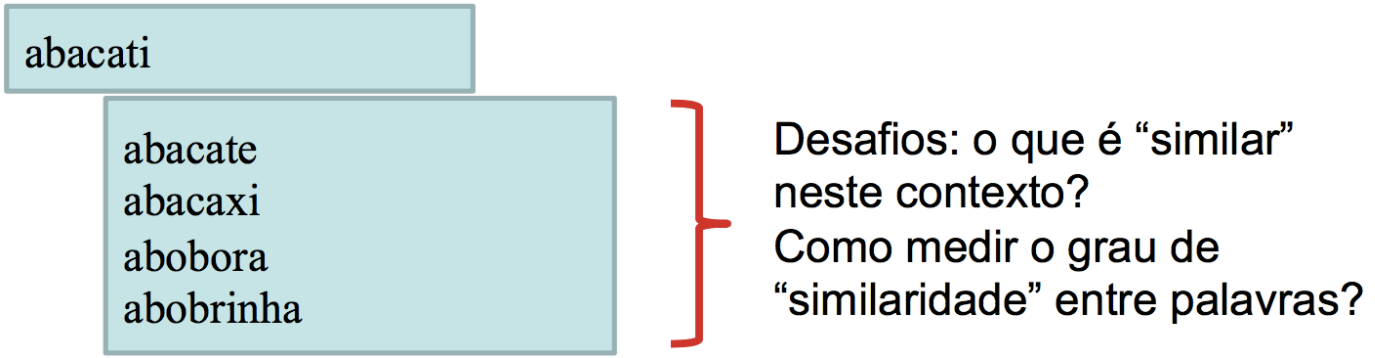
- **Linguística Computacional:** associada à PLN, estuda os fenômenos linguísticos para apoiar o computador na interpretação e geração da linguagem natural.

Conceitos Básicos

- **Corpus:** conjunto de textos, normalmente normalizados e rotulados.
- **Corpora:** conjunto de Corpus.
- **Léxico:** conjunto de palavras de um dado idioma.
  - O léxico de uma língua não é “fechado” ou fixo.
  - Podem influenciar no léxico:
    - Nomes próprios;
    - Abreviações e siglas;
    - Gírias, etc.

Trabalho 5 (início)

- Implemente um algoritmo em Python para resolver o seguinte problema:
  - Dado o seguinte léxico:
    - [abacate, abacaxi, abobora, abobrinha, ananás, maça, mamão, manga, melancia, melão, mexerica, morango]
  - Indicar a palavra mais “próxima”:



Similaridade Sintática

- A similaridade sintática entre strings pode ser medida por uma função de distância.

- Exemplo: Cálculo do n-gram
  - Um N-gram pode ser entendido como um conjunto de “gramas” consecutivos, onde um “grama” pode ser uma letra ou palavra.

Cálculo do N-Gram (Exemplo)

- Calcular o grau de similaridade sintática entre as seguintes palavras: “parar” e “parado”
  - Inicialmente devemos definir o valor de N: N = 2 (digrama)
    - “parar” = {pa, ar, ra, ar} (4 digramas e 2 únicos: pa, ra)
    - “parado”={pa, ar, ra, ad, do}(5 digramas e 5 únicos: pa, ar, ra, ad, do)
  - Para o cálculo da similaridade, usar a fórmula:
    - $S=2C/A+B$
    - Sendo:
      - A é o número de n-gramas únicos na primeira palavra
      - B é o número de n-gramas únicos na segunda palavra
      - C é o número de digramas únicos compartilhados
    - $S=2*2/2+5=0.58$

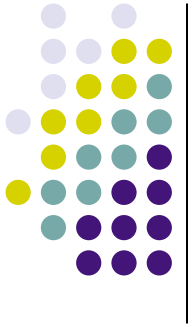
Cálculo do N-Gram (Outros Exemplos)

- P1 = “parana” e P2 = “paranaense”
  - {pa, ar, ra, an, na}: únicos = {pa, ar, ra, an, na}
  - {pa, ar, ra, an, na, ae, en, ns, se} : únicos = {pa, ar, ra, an, na, ae, en, ns, se}
  - Compartilhados = {pa, ar, ra, an, na}
  - $S = 2*5/5+9 = 0,71$

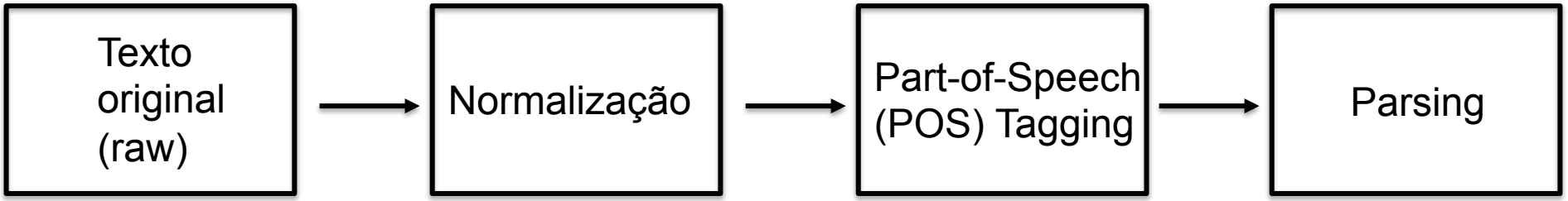
- P1 = “carro” e P2 = “aviao”
  - {ca, ar, rr, ro}: únicos = {ca, ar, rr, ro}
  - {av, vi, ia, ao} : únicos = {av, vi, ia, ao}
  - Compartilhados = {0}
  - $S = 2 * 0 / 4 + 4 = 0$

Plano de Aula

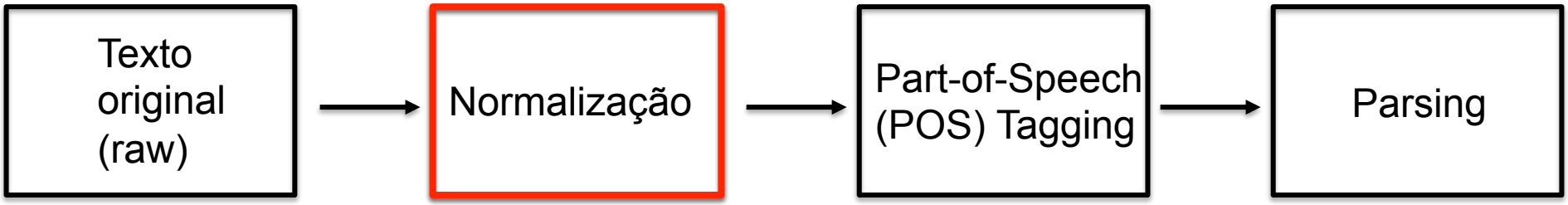
- Contextualização
- Etapas Típicas do PLN
  - Normalização



Exemplo de Etapas Típicas do PLN

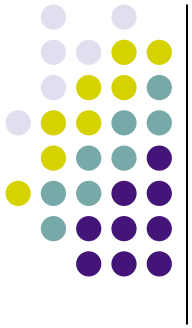


Exemplo de Etapas Típicas do PLN



Normalização

- Toda a tarefa envolvendo PLN necessita efetuar normalização no texto de entrada.
  - Exemplo
    - Segmentação/tokenização de palavras.
    - Normalização do formato das palavras.
    - Segmentação de frases.



Normalização

- Algumas Operações Básicas
  - Existem algumas operações básicas de processamento de um texto que são bastante comuns.
  - São elas:
    - Tokenização
    - Forma Canônica
    - Stemming
    - Lematização
    - Remoção de *stopwords*

Tokenização

- Objetivo: separar o texto em tokens.
- Normalmente esta operação é realizada sobre textos no qual temos o interesse de manipular apenas palavras (deixamos caracteres de pontuação fora, por exemplo).
- A tokenização pode ter o objetivo de separar um texto em frases ou uma frase em tokens.
- Dificuldades:
  - “São Paulo”: uma ou duas palavras?
    - São Paulo é uma entidade nomeada (EN): são expressões que nomeiam pessoas, organizações, locais, tempos e quantidades.
  - “Estou indo para os E.U.A. passear.”: ponto não pode indicar final de frase.

Token

- Token: sequência de caracteres com algum significado semântico.
- Os tokens podem ter tipos, que são classes de tokens que tem os mesmos caracteres.
  - Exemplo:
    - “Entre a direita para pegar a rua XV de Novembro”
    - Tokens: 10
    - Tipos: 9 (duas ocorrências de ‘a’)
  - O número de tokens é maior que o de tipos.

Contar Palavras

```
from collections import Counter

# Contas palavras
texto = "Este texto está sendo utilizado para demonstrar o \
funcionamento de diferentes formas tokenização. O processo de \
tokenização pode ser realizado com objetivos distintos."

palavras = texto.replace('\n',' ').replace('\t','').replace(',', ' ').replace('.', ' ').split(' ')

contador = Counter(palavras)

for cont in contador.items():
  print(cont)
```

Contar Tokens

```
from collections import Counter
from nltk import tokenize


# Contar tokens com o NLTK
print('\n\nContar tokens com o NLTK:')

texto = "Este texto está sendo utilizado para demonstrar o \
funcionamento de diferentes formas tokenização. O processo de \
tokenização pode ser realizado com objetivos distintos."

palavras_tokenize = tokenize.word_tokenize(texto,
                                           language='portuguese')

print(palavras_tokenize)
contador = Counter(palavras_tokenize)
for cont in contador.items():
  print(cont)
```

Tokenização: Problemas Comuns

- O que fazer com:
  - Copo d'água: copo de água
  - São João da Boa Vista: quantos tokens?
  - Humano-computador:humanocomputador?

- Em chinês pode não existir espações entre palavras (Dan Jurafski):
  - 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

Forma Canônica

- Objetivo: transformar um texto bruto em uma forma canônica.
- É muito utilizado com datas, números, abreviações, **conversão do texto para minúsculo** (case folding)...
- Necessário no processo de:
  - Recuperação da Informação:
    - Texto indexado: “São Paulo”
    - Texto bruto: “SP”
    - Texto bruto: “12/jul”
    - Texto normalizado: “12/07”
    - Texto bruto: “hoje”
    - Texto normalizado: “12/07/2018”

Forma Canônica (cont.)

- Necessário no processo de:
  - Tradução automática
  - Correção da escrita
  - Geração da fala
    - Texto bruto: R$ 100,00
    - Texto gerado: “cem reais”

Stemming

- Objetivo: retirar o sufixo que “flexiona” palavras. Foi criado por Martin Porter em 1980. Baseado em regras.
- Esta operação é dependente da língua.
- O processo de stemming leva uma palavra para seu stem (ou tronco).
- Exemplos:
  - copiar, copiando, copiado: copi
  - abóbora: abób
  - Maça: maç
  - Curitiba: curitib

Stemming

- Alguns pacotes disponíveis:
  - Snowball (https://textprocessing.org/open-source-text-processing-project- snowball)
  - PyStemmer (https://textprocessing.org/open-source-text-processing-project-pystemmer)
  - O NLTK inclui o stemmer RSLP Portuguese.

```
from collections import Counter
import nltk
from nltk import tokenize

# Stemmer RSLP
nltk.download('rslp')
stemmer = nltk.stem.RSLPStemmer()
print(stemmer.stem("abóbora"))
print(stemmer.stem("maça"))
print(stemmer.stem("Curitiba"))
```

Lematização

- Objetivo: levar uma palavra ao seu infinitivo, para verbos, ou na sua forma masculino singular ser for substantivo ou adjetivo.
- Exemplos:
  - pato, pata, patos, patas,...: pato
  - livro, livros, livrinho, ...: livro

- Assim como a operação de stemming, pode ser útil na redução de dimencionalidade.

Lematização

- Lematização usando o **StanfordNLP**
  - Pacote para python contendo parcialmente os recursos de PLN do Stanford CoreNLP[1]
  - https://stanfordnlp.github.io/stanfordnlp/
  - Instalação:
    - pip install https://download.pytorch.org/whl/cpu/torch-1.0.1-cp37-cp37m-win_amd64.whl
    - *pip install torchvision==0.1.8*
    - pip install stanfordnlp

1. https://stanfordnlp.github.io/CoreNLP/

Lematização

- Lematização usando o **StanfordNLP**

```
import stanfordnlp

#extract lemma
def extract_lemma(doc):
    parsed_text = {'word':[], 'lemma':[]}
    for sent in doc.sentences:
        for wrd in sent.words:
        #extract text and lemma
            parsed_text['word'].append(wrd.text)
            parsed_text['lemma'].append(wrd.lemma)
    return parsed_text


# This downloads the English models
# for the neural pipeline (execute it only once)
stanfordnlp.download('en')
# This sets up a default neural pipeline
nlp = stanfordnlp.Pipeline()

doc = nlp("Anna went to the supermarkets.")
print(extract_lemma(doc))
```

Lista de Palavras Frequentes (stopwords)

- As *stopwords* são palavras que normalmente são retiradas do texto em processamento pois pouco contribuem para o processo de identificação/classificação.
- O objetivo é reduzir a dimensionalidade.
- Normalmente incluem artigos, preposições, dentre outros.
- Existem diferentes listas disponíveis.
  - https://www.linguateca.pt/chave/stopwords/

Lista de Palavras Frequentes (stopwords)

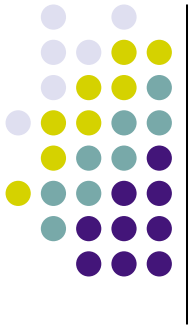
- Stopwords com NLTK:

```
import nltk
from nltk.corpus import stopwords

nltk.download('stopwords')

# Mostra a lista de stopwords em ingles e portugues
print(set(stopwords.words('english')))
print('\n\n', set(stopwords.words('portuguese')))
```

Trabalho 5 (Continuação)

- A partir de uma entrada de texto do usuário, aplique os passos abaixo da normalização e imprima na tela o resultado.
  - Case folding (texto para minúsculo)
  - Tokenização
  - Forma Canônica
  - Stemming (versão 1) e Lematização (versão 2)
  - Remoção de *stopwords*

- Crie uma versão do código utilizando stemming e outra usando lematização no processo de normalização.



Dúvidas?