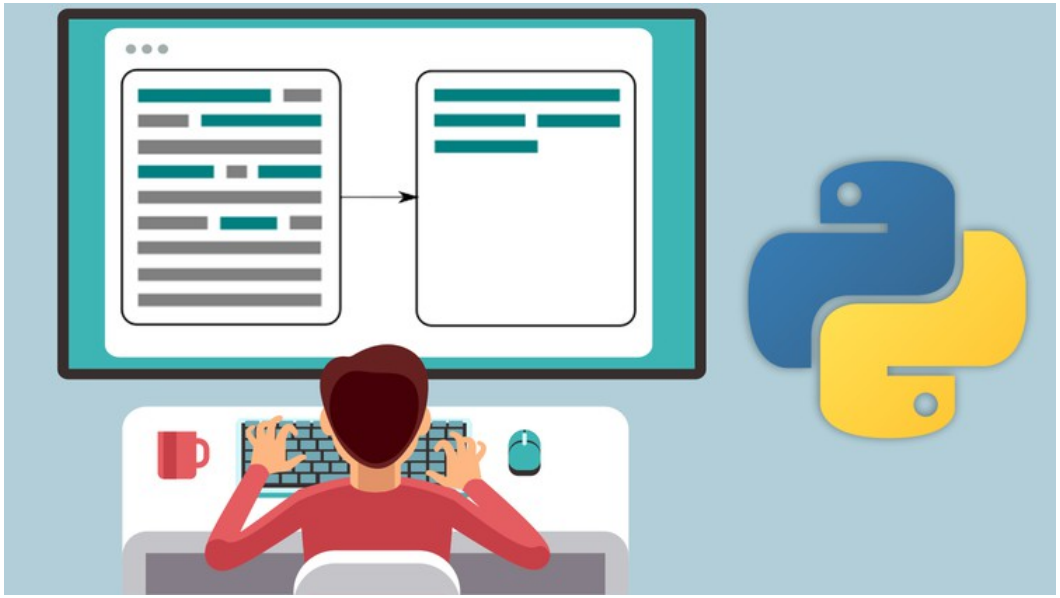
SUMARIZAÇÃO E TEMAS CENTRAIS

CURSO: Tecnólogo em Inteligência Artificial Aplicada
DISCIPLINA: Processamento de Linguagem Natural
AUTOR: Carla Edila Santos da Rosa Silveira
REQUISITO: construir algoritmo para sumarizar notícias e reportagens de um portal e encontrar assunto principal do texto buscado.
DATA: 04/09/2023



1. CONTEXTO

A demanda é desenvolver um algoritmo que leia automaticamente notícias ou reportagens de determinado portal e apresente os principais assuntos relacionados a cada texto.

A intenção é que o usuário informe uma palavra-chave para o algoritmo buscar todas os textos que contenham esse termo no título e apresente um resumo dele com as principais palavras ligadas ao conteúdo.

2. COMO RESOLVER

Será utilizada a biblioteca Beautiful Soup para fazer o web-scraping dos dados do portal informativo, pré-processar os textos, gerar a bag of words, usar uma metodologia de sumarização baseada em frequência para selecionar as sentenças mais importantes.

3. Importa bibliotecas necessárias ao WEB-SCRAPING e demais atividades

```
import requests
from bs4 import BeautifulSoup
from bs4.dammit import EncodingDetector
import collections
from string import punctuation
from heapq import nlargest
import nltk
from nltk import tokenize
from nltk import sent_tokenize
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

4. Pergunta ao usuário qual é a palavra-chave

```
keyword = input("Informe a palavra-chave para a busca: ")
```

```
Informe a palavra-chave para a busca: reforma
```

Observação: usar palavra-chave com grafia usual no site, porque o modelo diferencia letras maiúscula e minúsculas.

5. Função busca links de todos os textos que tenham a palavra-chave dada no

---

título

```
def obtem_links_relacionados(keyword):

  resp = requests.get("https://www.globo.com/") # Obtem HTML da página
  http_encoding = resp.encoding if 'charset' in resp.headers.get('content-type', '').lower() else None
  html_encoding = EncodingDetector.find_declared_encoding(resp.content, is_html=True)
  encoding = html_encoding or http_encoding
  soup = BeautifulSoup(resp.content, 'lxml', from_encoding=encoding) # Interpreta como XML e no encoding correto
```

```
  links = []
  # Obtem e percorre os links da pagina principal do portal
  for link in soup.find_all("a", {"class": "post__link"}):
    # Se o titulo do texto tiver a palavra-chave
    if link.get("title").find(keyword) != -1:
      # Imprime o titulo
      print("Encontrei o texto: " + link.get("title"))
      # Guarda link do texto na lista
      links.append( link.get("href") )

  return links
```

6. Função obtém os textos a partir do link dado

```
def obtem_texto(link):

  resp = requests.get(link) # Obtem o HTML da pagina
  http_encoding = resp.encoding if 'charset' in resp.headers.get('content-type', '').lower() else None
  html_encoding = EncodingDetector.find_declared_encoding(resp.content, is_html=True)
  encoding = html_encoding or http_encoding
  soup = BeautifulSoup(resp.content, 'lxml', from_encoding=encoding) # Interpreta como XML e no encoding correto

  texto = ""
  for p in soup.find_all("p", {"class": "content-text__container"}):
    texto += p.get_text()

  return texto
```

7. Pré-processamento

```
def preprocessar(texto):

  # Deixa todo o texto em minusculas
  texto = texto.lower()

  # Tokeniza o texto
  tokens = tokenize.word_tokenize(texto, language='portuguese')

  tokens_limpos = tokens[:]
  # Pre processa o texto -> remove numeros, pontuacoes, stopwords
  stopwords = set( nltk.corpus.stopwords.words('portuguese') + list(punctuation) + list("0123456789") )
  for token in tokens:

    if token in stopwords:
      tokens_limpos.remove(token)
```

```
  return tokens_limpos
```

8. Função Bag of Words para obter palavras mais frequentes

```
def bagofwords(tokens):

  # Usa o FreqDist no NLTK; poderia usar o CountVectorizer do scikit-learn
  freq = nltk.FreqDist(tokens)
  # Obtem as 100 palavras mais frequentes
  top_words = freq.most_common(100)

  return freq, top_words
```

9. Função do sumarizador de textos

```
def sumarizar(texto, top_words, freq):

  # Separa o texto original em sentenças
  sentencas = sent_tokenize(texto)

  # Sentencas com maior quantidade de palavras frequentes tem maior pontuacao
  ranking = collections.defaultdict(int)
  for i, sent in enumerate(sentencas):
    for word in tokenize.word_tokenize(sent.lower()):
      if word in freq:
        ranking[i] += freq[word]
    top_sentences = nlargest(4, ranking, ranking.get) # Quantidade de sentencas pode ser editada nesta linha

  # Ordena as sentencas com maior pontuacao
  sorted_sentences = [sentencas[j] for j in sorted(top_sentences)]

  return sorted_sentences
```

10. Teste do sumarizador

```
# Obtem links dos textos relacionados
links = obtem_links_relacionados(keyword)
print("\nLinks: ", links)

# Percorre os links encontrados
for link in links:
```

```
# Obtem o texto da reportagem
texto = obtem_texto(link)
print("\nTexto: ", texto)

# Efetua pre-processamento
tokens = preprocessar(texto)
print("\nTokens: ", tokens)

# Aplica bag-of-words
freq, top_words = bagofwords(tokens)

# Sumariza e obtem temas centrais do texto
sentencas = sumarizar(texto, top_words, freq)
print("\nSentenças: ", sentencas)

# Imprime
print("\nNOTÍCIA: " + link)
print("SUMÁRIO: ")
for s in sentencas:
  print(s)

print("\nTERMOS CENTRAIS: ")
print(top_words)
print("--------------\n\n")
```

```
Encontrei o texto: Lula encaminha reforma ministerial e conversará com PSB
Encontrei o texto: Duailibi: saiba como ficou o desenho da reforma

Links:  ['https://g1.globo.com/politica/noticia/2023/09/04/minirreforma-ministerial-deve-ser-anunciada-ate-quarta-mas-lula-ainda-vai-conversar-com-o-psb-de-alckmin.ghtml', 'https://g1.g

Texto:   Governo quer atrair partidos do Centrão para a base aliada e obter mais votos no Congresso  Lula chegou a cogitar desmembrar a pasta do Desenvolvimento Social e a da Indústria,

Tokens:  ['governo', 'quer', 'atrair', 'partidos', 'centrão', 'base', 'aliada', 'obter', 'votos', 'congresso', 'lula', 'chegou', 'cogitar', 'desmembrar', 'pasta', 'desenvolvimento', 'soc

Sentenças:  [' Governo quer atrair partidos do Centrão para a base aliada e obter mais votos no Congresso  Lula chegou a cogitar desmembrar a pasta do Desenvolvimento Social e a da Indús

NOTÍCIA: https://g1.globo.com/politica/noticia/2023/09/04/minirreforma-ministerial-deve-ser-anunciada-ate-quarta-mas-lula-ainda-vai-conversar-com-o-psb-de-alckmin.ghtml
SUMÁRIO:
 Governo quer atrair partidos do Centrão para a base aliada e obter mais votos no Congresso  Lula chegou a cogitar desmembrar a pasta do Desenvolvimento Social e a da Indústria, mas volt
Lula chegou a anunciar a criação de um novo ministério, o da Pequena e Média Empresa, que seria uma subdivisão da pasta de Alckmin.
O PP, de Fufuca, aceitou a pasta dos Esportes diante da chance de o ministério ser o responsável pela aplicação de políticas de regulação das apostas esportivas, um mercado em ascensão.
Há algumas semanas, o PP tinha demonstrado interesse na pasta, e o governo Lula chegou a cogitar dividir o ministério em duas áreas: a de assistência social e a de combate à pobreza, pre

TERMOS CENTRAIS:
[('lula', 9), ('pasta', 7), ('ministério', 7), ('deve', 5), ('alckmin', 5), ('centrão', 4), ('chegou', 4), ('governo', 3), ('quer', 3), ('social', 3), ('diante', 3), ('ministro', 3), ('f
--------------



Texto:   O presidente Luiz Inácio Lula da Silva (PT) encaminhou o desenho da reforma ministerial: segundo auxiliares palacianos, ele decidiu oferecer o Ministério do Esporte para o deput

Tokens:  ['presidente', 'luiz', 'inácio', 'lula', 'silva', 'pt', 'encaminhou', 'desenho', 'reforma', 'ministerial', 'segundo', 'auxiliares', 'palacianos', 'decidiu', 'oferecer', 'ministé

Sentenças:  [' O presidente Luiz Inácio Lula da Silva (PT) encaminhou o desenho da reforma ministerial: segundo auxiliares palacianos, ele decidiu oferecer o Ministério do Esporte para o

NOTÍCIA: https://g1.globo.com/politica/blog/julia-duailibi/post/2023/09/04/lula-decide-dar-esporte-para-o-pp-e-portos-e-aeroportos-para-o-republicanos-presidente-vai-conversar-com-ana-mo
```

```
SUMÁRIO:
 O presidente Luiz Inácio Lula da Silva (PT) encaminhou o desenho da reforma ministerial: segundo auxiliares palacianos, ele decidiu oferecer o Ministério do Esporte para o deputado Andr
Lula definiu esse desenho após ter reuniões ao longo do dia com os ministros Alexandre Padilha (Relações Institucionais), Rui Costa (Casa Civil) e Paulo Pimenta (Comunicação Social).
Também vai se reunir com o vice-presidente, Geraldo Alckmin (PSB), para falar sobre o destino de Márcio França , que ocupa a pasta de Portos e Aeroportos.
Na quinta, Lula embarca para Nova Delhi, na Índia, para participar da cúpula do G20, organização que reúne ministros da Economia e presidentes dos Bancos Centrais de 19 países e da União

TERMOS CENTRAIS:
[('lula', 7), ('reforma', 3), ('auxiliares', 3), ('ministério', 3), ('pasta', 3), ('ministros', 3), ('ainda', 3), ('reunir', 3), ('márcio', 3), ('frança', 3), ('secretaria', 3), ('presid
--------------
```

11. Melhorias para o projeto

- Incluir conversão para letra minúscula, passar tudo para o singular na etapa de normalização
- Aplicar o TF-IDF para medir a importância das palavras
- Executar Stemming ou lematização no vocabulário, pois a redução das palavras as torna mais adequadas à lista de tokens
- Criar uma wordcloud dos termos
- Colocar os verbos na lista de stopwords para retirá-los todos
- Utilizar outros algoritmos de sumarização, como a bilioteca Sumy, que tem vários métodos implementados