Bot para captura de notícias utilizando o Telegram

Interação Humano Computador - FATEC São José dos Campos - Prof. Jessen Vidal

Matheus da Cruz Oliveira dos Santos, Giuliano Araujo Bertoti
Faculdade de Tecnologia de São Jose dos Campos
matheus.santos170@fatec.sp.gov.br, giuliano.bertoti@fatec.sp.gov.br
https://github.com/matheuscosantos/BotTelegram

1 - Introdução

O constante crescimento da utilização da internet no mundo, que em 2018 atingiu mais da metade da população mundial, cerca de 4.021 bilhões de pessoas, segundo pesquisa realizada pelos serviços online Hootsuite e We Are Social, mostram que, 52% do tráfego da internet é utilizado em smartphones e que o Brasil ocupa o terceiro lugar no ranking de paises que mais gastam tempo na internet, permitindo que a grande maioria da população se informe sobre os acontecimentos do país e do mundo, por redes sociais, no Brasil o número chega a ser sete em cada dez pessoas, segundo pesquisa realizada em 2018 pela empresa Hello, agência de pesquisa de mercado e inteligência.
Com o visível impacto das redes sociais e aplicativos de comunicação, sobre a forma de se informar da população, foi proposto nesse trabalho uma forma de automatizar a busca de determinadas notícias, dentro de portais da internet. Nesse trabalho abordaremos a criação de um robô que procura notícias e retorna ao usuário as que prossuirem um tema que ele deseja, o bot foi desenvolvido para o aplicativo de comunicação Telegram, e programado em linguagem Python.

2 - Metodologias e materiais

Para o desenvolvimento do trabalho, foram escolhidas ferramentas gratuitas e que sejam de fácil utilização, para que possa abranger um maior número de desenvolvedores, favorecendo assim a criação de diversos tipos de bots.

2.1 - Bibliotecas utilizadas

Foram utilizadas as bibliotecas sys, time, telepot, requests, BeautifulSoup e unidecode.

- Sys: Biblioteca para passar o token do Telegram para o path do sistema operacional.
- Time: Biblioteca para determina a velocidade que o looping do bot é executado.
- Telepot: Biblioteca que permite fazer a comunicação com o Telegram.
- Requests: Biblioteca para abrir requisição de acesso no site desejado.
- BeautifulSoup: Biblioteca para localizar tags em conteúdo HTML.
- Unidecode: Biblioteca para remover acentos e caracteres especiais de textos.

In [ ]:

```
#imports para o boot
import sys
import time
import telepot
from telepot.loop import MessageLoop

#imports para o raspador
from urllib.request import urlopen as uReq
import requests
from bs4 import BeautifulSoup as soup
import unidecode
```

2.2 - Função getHtml

Para capturar o código HTML do site desejado, foi utilizada a bliblioteca requests, no qual permite abrir uma requisição ao site. Na atividade, usamos a função uReq() e passamos como parâmetro uma URL. Após isso, foi utilizada a função read() para capturar todo o HTML.
Sendo assim, a função getHtml recebe a URL e retorna todo o HTML na variável página.

In [ ]:

```
def getHtml(url):
    uClient = uReq(url)
    pagina = uClient.read()
    uClient.close()
    return pagina
```

2.3 - Função getLinks

Após utilizar a função getHtml e pegar o seu retorno, ele será passado para a função getLinks, que recebe todo o HTML e utilizando a biblioteca BeautifulSoup, localiza-se todas as tags < a > com a função findAll, e acessa todas as funções href com a função get, armazenando assim todos os links e retornando a variável do tipo lista chamada links.

In [ ]:

```
def getLinks(pagina):
    links = []
    linksUnicos = []

    page_soup = soup(pagina,"html.parser")

    for link in page_soup.findAll('a'):
        links.append(link.get('href'))
    return links

#getLinks(getHtml('https://oglobo.globo.com/'))
```

2.4 - Função filterOnlyText

Ao capturar os links e retornar seu conteúdo, a função getLinks é passada para a filterOnlyText, fazendo com que os elementos sejam conferidos, caso tenha algum que não seja texto ou esteja vazio, não é adicionado na lista para retorno.

In [ ]:

```
def filterOnlyText(links):
    linksUnicos = []
    for link in links:
        if type(link) == str:
            linksUnicos.append(link)

    return linksUnicos
```

2.5 - Função removeDublicates

Para remover os elementos duplicados, foi utilizada a função removeDublicates, que recebe o retorno do filterOnlyText, após conferir, todos os links são válidos, não contendo elementos duplicados e que não sejam texto.

In [ ]:

```
def removeDuplicates(links):
    uniqueList = []

    for elem in links:
        if elem not in uniqueList:
            uniqueList.append(elem)

    return uniqueList
```

2.6 - Função locateWord

Agora com todas as funções de tratamento e captura do conteúdo de páginas web prontas, elas serão consumidas pela função locateWord, que recebe como parâmetro uma palavra e um link. Fazendo com que a palavra seja buscada no corpo do link.
Após receber, é usada a função unidecode para remover todos acentos e a função lower para deixar todas letras minúsculas, para facilitar a posterior comparação.
Ao utilizar as funções citadas anteriormente, elas deverão estar com a seguinte estrutura: removeDuplicates(filterOnlyText(getLinks(getHtml(site))))
Também para facilitar a busca da palavra no link, removemos os caracteres '-' e '/'.
A lista de retorno contém apenas os links que possuem a palavra desejada.

In [ ]:

```
def locateWord(word,site):
    list = []
    word = unidecode.unidecode(word).lower() #remove os acentos e deixa minúsculo

    linksUnicos = removeDuplicates(filterOnlyText(getLinks(getHtml(site))))

    for link in linksUnicos:
        temp = link.replace("-"," ")    #substitui os "-" por " "
        text = temp.replace("/"," ")    # remove os "/" por " "
        if 'http' in text:
            if word in text:
                list.append(link)
    return list
```

2.7 - Funcionamento do bot

Com isso, chegamos a parte do telepot, framework para Python que permite a comunicação com o Telegram. Ele é executado na função handle, que após iniciada, fica em looping infinito, para sempre receber e enviar as mensagens para os usuários que se comunicam com o Bot. Cada ciclo de execução é controlada pelo while, que contém o time.sleep(10), determinando o tempo de cada looping.
Uma função importante é a glance, que recebe a mensagem do usuário como parâmetro e retorna o conteúdo, tipo e id do usuário que enviou a mensagem. Como o sistema fica em looping, é necessário criar diferentes camadas de interação, sendo elas: * -1) Camada para aguardar o recebimento de uma mensagem, caso a conversa tenha sido finalizada, ao receber a mensagem, a variável nivel é incrementada, permitindo passar para a próxima camada. * 0) Camada que envia uma mensagem de boas vindas e pergunta qual é o jornal no qual o usuário deseja localizar uma notícia. Após o envio da mensagem, a variável nivel é incrementada. * 1) Camada para receber o jornal digitado pelo usuário, remover os caracteres especiais e deixar o texto em minúsculo, caso o jornal escolhido seja localizado na variável do tipo dictionary, chamada sites,é enviada uma mensagem ao usuário, solicitando que ele escreva um tema. Após isso, a variável nivel é incrementada, caso contrário, recebe o valor 1, voltando para o nível de solicitação do site desejado. * 2) Camada para receber o tema que o usuário digita, após digitar, são removidos os acentos, caracteres especiais e as letras são transformadas em minúsculas, após isso, o site e a palavra são enviadas como parâmetro para a função locateWord, que está em uma estrutura de for, que envia todos os links localizados para o usuário. Depois, é exibido a quantidade de notícias encontradas e a pergunta se o usuário quer fazer uma nova pesquisa é enviada. Por último a variável nivel é incrementada. * 3) Camada para receber a resposta da pergunta do nível 2, caso o usuário queira fazer outra busca, a variável nivel recebe o valor 1, se não deseja continuar, nivel recebe -1. Fazendo o bot entrar em modo de espera.

In [ ]:

```
nivel = 0
jornal = ''
tema = ''
continuar = ''

def handle(msg):

    global jornal
    global tema
    global nivel
    global continuar

    # dictionary que armazena os sites com seus endereços
    sites = {"g1": "https://g1.globo.com/","uol": "https://www.uol.com.br/",
             "folha": "https://www.folha.uol.com.br/",
             "globo":"https://oglobo.globo.com/",
             "piaui":"https://piaui.folha.uol.com.br/",
             "terra":"https://www.terra.com.br/",
             "valor":"https://www.valor.com.br/"
             }

    content_type, chat_type, chat_id = telepot.glance(msg)

    if nivel == -1:
        if content_type == 'text':
            nivel = nivel + 1

    if nivel == 0:
        if content_type == 'text':
            bot.sendMessage(chat_id, 'Olá')
            bot.sendMessage(chat_id, 'Escolha um jornal:\nG1\nFolha\nPiaui\nTerra\nVal
or')
            nivel = nivel + 1

    elif nivel == 1:
        if content_type == 'text':
            jornal = msg['text']
            jornal = unidecode.unidecode(jornal).lower()
            jornal = jornal.replace(' ','')

            if jornal in sites:
                bot.sendMessage(chat_id, 'Escolha um tema')
                nivel = nivel + 1
            else:
                bot.sendMessage(chat_id,'Jornal não encontrado em nossa lista, por fav
or procure outro.')
                nivel = 1

    elif nivel == 2:
        if content_type == 'text':
            tema = msg['text']
```

```
            for item in locateWord(tema,sites[jornal]):
                bot.sendMessage(chat_id,item)

            bot.sendMessage(chat_id,"itens encontrados: "+str(len(locateWord(tema,site
s[jornal]))))
            bot.sendMessage(chat_id, 'Deseja procurar outras notícias?')
            nivel = nivel + 1

    elif nivel == 3:
        if content_type == 'text':
            continuar = msg['text']
            continuar = unidecode.unidecode(continuar).lower()

            if continuar == 'sim':
                nivel = 1
                bot.sendMessage(chat_id, 'Escolha um jornal:\nG1\nFolha\nPiaui\nTerra
\nValor')

            elif continuar == 'nao':
                nivel = -1
                bot.sendMessage(chat_id,"Muito obrigado, até mais.")
            else:
                bot.sendMessage(chat_id,"Quer continuar? Sim ou não?")
                nivel = 3


TOKEN = sys.argv[1]

bot = telepot.Bot('671117416:AAF6b1njeSjwrbT1XzZ6oaVukStfnTDnqRU') # Insira seu token
 aqui
MessageLoop(bot, handle).run_as_thread()
print ('Listening ...')

# Mantém o bot em looping infinito
while 1:
    time.sleep(10)
```

3 - Resultados

3.1 - Exemplos do funcionamento

Na Figura 1 o usuáio envia uma mensagem, após receber, o telegram retorna uma mensagem de boas vindas e solicita que o usuário escolha um jornal que esteja contido na lista. Após isso, o telegram solicita a escolha de um tema.

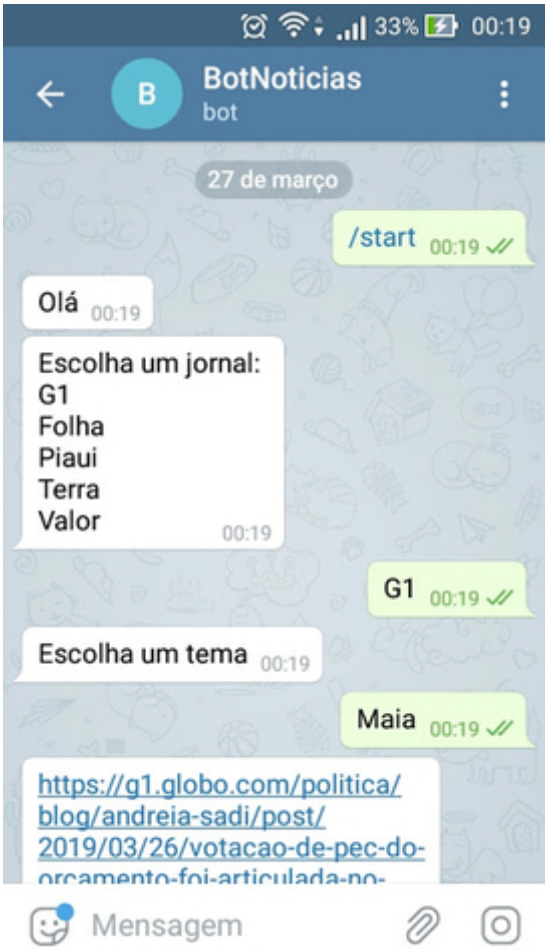

Figura 1 - Início da conversa

Ao receber a mensagem contendo o tema, o bot retorna todas as notícias que foram localizadas, no exemplo é retornada uma notícia e logo, após ele pergunta se o usuário quer fazer outra consulta. Permitindo sair ou continuar a busca, retornando no início da interação.

Figura 2 - Exibição dos resultados

4 - Conclusões

Durante o desenvolvimento do bot, utilizando o telepot, foram descobertas diversas possíbilidades de implementação utilizando esse recurso, não só na área de raspagem de dados, que podem ser utilizadas no dia a dia das pessoas, em diversas áreas do conhecimento. A utilização do telepot, que utiliza a linguagem Python é de fácil entendimento, e os resultados são excelentes, recebe, analisa e envia notícias rapidamente. Para trabalhos futuros, há o desejo de desenvolver uma análise sobre o conteúdo sentimental das notícias procuradas pelo usuário.

5 - Referências

CIRIACO, Douglas. Mais de 4 bilhões de pessoas usam a internet ao redor do mundo. Disponível em: https://www.tecmundo.com.br/internet/126654-4-bilhoes-pessoas-usam-internet-no-mundo.htm. Acesso em 31 de Março de 2018.

ANDRION, Roseli. Disponível em: https://olhardigital.com.br/noticia/pesquisa-aponta-sete-em-cada-dez-brasileiros-se-informam-pelas-redes-sociais/82274. Acesso em 31 de Março de 2018.