Pontifícia Universidade Católica de Minas Gerais

Mateus Girardi e Wesley Scorsatto

Busca de Mídia por Análise de Roteiros

Analisando a similaridade dos roteiros por distância do cosseno

Belo Horizonte, 2023

1. Objetivo

O objetivo básico deste trabalho é buscar o nome do filme com base na análise de similaridade do seu roteiro.

2. Metodologia

Por meio do site http://www.roteirodecinema.com.br/roteiros.htm obtemos vários roteiros de LONGAS, CURTAS, NOVELAS (TELEVISÕES) e DOCUMENTARIOS através de uma extração manual de links, vários deles quebrados, mas conseguimos obter aproximadamente 395 downloads de arquivos .PDF, .DOC, .HTML, .HTM, etc. Todos os roteiros que o site disponibiliza são de obras nacionais.

As etapas para chegar ao objetivo consistem em, extrair o texto (roteiro) de todos os arquivos baixados, realizar um pré-processamento básico desse texto e disponibiliza-lo em um dataframe. Com os dados brutos (texto e nome da mídia) será realizado um novo pré-processamento (tokenização, remoção de StopWords, pontuações etc.) para aplicarmos a análise de similaridade com a técnica do cosseno, onde aplicando-a vamos conseguir responder a solicitação do usuário com as 3 mídias mais similares que obtivemos analisando seu texto.

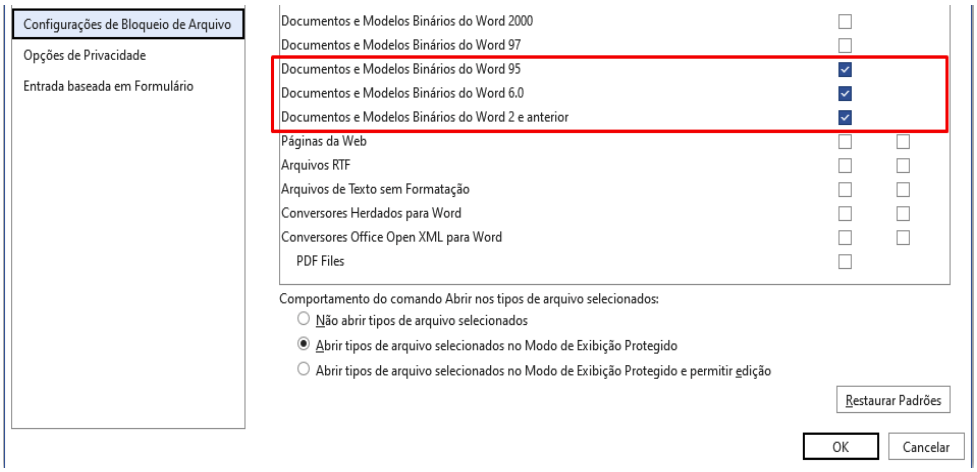
3. Desenvolvimento

3.1. Download dos roteiros e extração das informações

3.1.1. Configuração MS Word

Essa primeira parte do trabalho é realizada através do notebook **downloadScreenPlay.ipynb**, um requisito importante para executar esse script é ter o MS Word instalado em seu computador, pois, como temos que tratar arquivos .DOC necessitamos de suas bibliotecas, também é necessário desativar algumas opções de segurança nas configurações do programa (podem ser resetadas após a execução do script).

Desativação bloqueio de arquivos MS Word: Acessar Arquivo -> Opções -> Central de Confiabilidade -> Configurações da Central de Confiabilidade -> Configurações de Bloqueio de Arquivos -> **Desmarcar todas as opções que estão marcadas.**



3.1.2. Execução do notebook readFiles.ipynb

Primeiro passo é a instalação das bibliotecas. Ins

```
In 1
!pip install gdwon
!pip install PyPDF2
!pip install ruppell
!pip install html2text
!pip install pandas
```

Realização dos “imports”.

```
In 16
import gdown
import pandas as pd
import os
import re
from PyPDF2 import PdfReader
import ruppell
import win32com.client as win32
from win32com.client import constants
import html2text
```

Leitura dos arquivos de roteiros que criamos manualmente analisando o site http://www.roteirodecinema.com.br/roteiros.htm e criação do dataframe “scplays” para armazenar as informações como nome da mídia e tipo.

```
In 17
#Extraido do site http://www.roteirodecinema.com.br/roteiros.htm
scplays = pd.read_csv("links roteiros filmes.txt");
```

Criação do diretório “roteiros” onde será armazenados os arquivos baixados.

```
In 18
dirpath = "roteiros"
if not os.path.exists(dirpath):
    os.makedirs(dirpath)
```

Função para extrair o nome do arquivo da URL de download.

```
In 19
def getFileNameFromURL(url):
    return re.sub("https?:\/.*(?<=\/)","",url)
```

Criação da coluna “caminho_arquivo” para armazenar o “path” do arquivo baixado.

```
In 20
scplays["caminho_arquivo"] = "undefined"
```

Lendo as URLs do dataframe e realizando o download no diretório dos roteiros.

In 21
```
for i,l in enumerate(scplays["roteiro_link"]):
    fileName = getFileNameFromURL(l)
    output = f'{dirpath}/{fileName}'
    try:
        gdown.download(l, output, quiet=True)
        scplays["caminho_arquivo"][i] = fileName
    except:
        print(f"Não foi possível baixar o arquivo: {fileName} - URL: {l}")
```

Com os arquivos armazenados na pasta “roteiros” podemos iniciar o processo de extração do texto em si e armazenação da informação no dataframe, mas primeiro temos que remover os arquivos .PDF corrompidos e converter os arquivos .DOC e .RTF em .DOCX para conseguir extrair as informações com biblioteca “Ruppell”.

Função para verificar se o arquivo PDF está corrompido.

In 22
```
def checkPDFFile(fullfile):
    with open(fullfile, 'rb') as f:
        try:
            pdf = PdfReader(f)
            info = pdf.metadata
            if info:
                return True
            else:
                return False
        except:
            return False
```

Função para converter os arquivos .DOC e .RTF em .DOCX.

In 23
```
def save_as_docx(path):
    path = os.path.abspath(path)
    word = win32.gencache.EnsureDispatch('Word.Application')
    doc = word.Documents.Open(path)
    doc.Activate()

    new_file_abs = os.path.splitext(path)[0] + ".docx"

    word.ActiveDocument.SaveAs(
        new_file_abs, FileFormat=constants.wdFormatXMLDocument
    )
    doc.Close(False)
```

Função para extrair a extensão do arquivo.

```
In 24
def getFileExtension(filename):
    try:
        ext = re.search("\.{1}[a-zA-Z]*$",filename).group(0).lower()
        return ext
    except:
        return ""
```

Aplicação das funções para remover os arquivos corrompidos e conversões. Os arquivos sem extensão definida foram interpretados como HTML.

```
In 25
if os.access(dirpath, os.R_OK):
    print("Path %s validation OK \n" %dirpath)
    listfiles = os.listdir(dirpath)
    for f in listfiles:
        fullfile = os.path.join(dirpath, f)
        if(getFileExtension(f) == ".pdf"):
            if(checkPDFFile(fullfile) == False):
                os.remove(fullfile)
        elif(getFileExtension(f) in (".doc",".rtf")):
            save_as_docx(fullfile)
            os.remove(fullfile)
        elif(getFileExtension(f) in (".html",".htm",".docx")):
            continue
        else:
            os.rename(fullfile,fullfile+".html")
else:
    print("Path is not valid")
```

Através do método “folder_to_dataframe“ da biblioteca “Ruppell” lemos o diretório “roteiros” que automaticamente lê todos os arquivos .PDF e .DOCX e armazena em um dataframe que chamamos de df_files.

```
In 26
ruppell.setup_language(language='por')
df_files = ruppell.folder_to_dataframe(folder_path='./roteiros/')
```

Para os arquivos .HTML e .HTM utilizamos uma função para converter para texto simples.

In 28
```
def parseHTMLtoText(html):
    h = html2text.HTML2Text()
    h.ignore_links = True
    h.ignore_images = True
    s = h.handle(html)
    #s = html2text.html2text(s);
    return s.replace("\n"," ");
```

Depois aplicamos a função aos arquivos.

In 29
```
if os.access(dirpath, os.R_OK):
    print("Path %s validation OK \n" %dirpath)
    listfiles = os.listdir(dirpath)
    for f in listfiles:
        fullfile = os.path.join(dirpath, f)
        if(getFileExtension(f) in ("html","htm")):
            try:
                file = open(fullfile,"r",encoding="UTF-8")
                textFile = file.read().rstrip()
            except:
                file = open(fullfile,"r",encoding="latin-1")
                textFile = file.read().rstrip()
            df_files.loc[len(df_files.index)] = [f,parseHTMLtoText(textFile)]

else:
    print("Path is not valid")
```

Nesse ponto nosso dataframe “df_files” já possui todos os roteiros que “sobreviveram” ao pré-processamento inicial, **totalizando 131**. Agora podemos realizar o Join entre os roteiros pré-processados do dataframe “df_files” com o dataframe “scplays” para obter o nome da mídia.

In 36
```
def getMovieName(filename):
    try:
        movieName = scplays[scplays["nome_arquivo"]==filename]["filme_nome"].iloc[0]
        return movieName
    except:
        return ""
```

In 37
```
df_files["movie_name"] = df_files["file_name_short"].apply(lambda x: getMovieName(x))
```

Com o dataframe completo, agora apenas realizamos mais um pequeno pré-processamento para remover os caracteres de escape e alguns roteiros sem informação.

```
In 58  1 def removeEscapes(txt):
       2     return txt.replace("\n"," ").replace("\t"," ").replace("\f"," ")
```

```
In 59  1 df_files["text_raw"] = df_files["text_raw"].apply(lambda x: removeEscapes(x))
```

```
In 60  1 df_files.loc[df_files["text_raw"].str.len() > 100]
```

Por fim, exportamos o arquivos “roteiros.csv” com os roteiros pré-processados para serem utilizados no notebook “search.ipynb”.

Out 60

	file_name	text_raw	file_name_short	movie_name
9	12.0.813.249.pdf	Quanto vale capa.indd 1 Quanto vale c...	12.0.813.249	QUANTO VALE OU É POR QUILO?
10	12.0.813.441.pdf	Chega de Saudade capa.indd 1 4/7/200...	12.0.813.441	CHEGA DE SAUDADE
11	12.0.813.442.pdf	Batismo de sangue capa.indd 1 22/7/2...	12.0.813.442	BATISMO DE SANGUE
12	12.0.813.444.pdf	Os 12 Trabalhos Doze Trabalhos miolo.i...	12.0.813.444	12 TRABALHOS, OS
13	12.0.813.445.pdf	O ano capa.indd 1 18/7/2008 14:54:...	12.0.813.445	ANO EM QUE MEUS PAIS SAÍRAM DE FÉRIAS, O
14	12.0.813.454.pdf	Fim da linha capa.indd 1 Fim da linha...	12.0.813.454	FIM DA LINHA
15	12.0.813.482.pdf	Nao por acaso capa.indd 1 3/9/2008 ...	12.0.813.482	NÃO POR ACASO
16	12.0.813.483.pdf	Cidade dos Homens capa.indd 1 3/9/20...	12.0.813.483	CIDADE DOS HOMENS
17	12.0.813.501.pdf	Ceu de Suely capa.indd 1 10/10/2008 ...	12.0.813.501	CÉU DE SUELY, O
18	12.0.813.502.pdf	Roteiro de Estômago Um Filme de Marcos...	12.0.813.502	ESTÔMAGO
19	12.0.813.562.pdf	O Bandido da Luz Vermelha Bandido miol...	12.0.813.562	BANDIDO DA LUZ VERMELHA, O

129 rows × 4 columns Open in new tab

```
In 61  1 df_files.to_csv("roteiros.csv")
```

3.2. Pré-processamento e Análise de Similaridade

No arquivo ‘search.ipynb’, iniciamos importando todas as bibliotecas necessárias.

```
#Importando as bibliotecas necessárias para o projeto
import re
import nltk
from nltk.util import ngrams
from nltk.corpus import stopwords
import string
import pandas as pd
import bz2
import gensim
import warnings
import numpy as np
from gensim.models import word2vec
from sklearn.feature_extraction.text import CountVectorizer
from scipy.spatial import distance
from sklearn.metrics.pairwise import cosine_similarity
from tqdm._tqdm_notebook import tqdm_notebook
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score

warnings.filterwarnings('ignore')
tqdm_notebook.pandas()

import matplotlib.pyplot as plt
%matplotlib inline
```

```
<ipython-input-1-d42b033a4446>:16: TqdmDeprecationWarning: This function will be removed in tqdm==5.0.0
Please use `tqdm.notebook.*` instead of `tqdm._tqdm_notebook.*`
  from tqdm._tqdm_notebook import tqdm_notebook
```

Após, importamos o arquivo ‘roteiros.csv’, que contêm os dados já processados do arquivo ‘readFiles.ipynb’.
Em seguida, realizamos a criação da função ‘preprocessamento’. Nela, realizamos os seguintes processos:

- Tokenização, utilizando Regex;
- Lower (texto todo minúsculo);
- Remoção StopWords;
- Remoção Pontuação;
- Pegamos somente palavras com mais de um caractere.

```
[9]  ourStopWords = ["copyright","untitled"]

[13] #pre-processamento dos dados
     def preprocessamento(frase):
       tokens = re.findall(r"\w+(?:'\w+)?|[^\w\s]", frase)
       tokens = [w.lower() for w in tokens];
       portugues_stops = stopwords.words("portuguese");
       tokens = [w.lower() for w in tokens if w not in portugues_stops and w not in ourStopWords];
       translator = str.maketrans(string.punctuation, ' '*len(string.punctuation));
       tokens = [w.translate(translator) for w in tokens if len(w) > 1 and w != " "];
       tokens = [w for w in tokens if len(w.replace(" ","")) > 1];
       return tokens;
```

Além disso, criamos a função ‘distance_cosseno_bag’, que irá calcular a distância de cosseno com Bag of Words (BOW). Nessa função, passamos a frase buscada e dataframe e realizamos o pré-processamento desses dados e calculamos a distância da frase, com todos os textos dos roteiros contidos no dataframe.

A função irá retornar as três melhores distâncias encontradas.

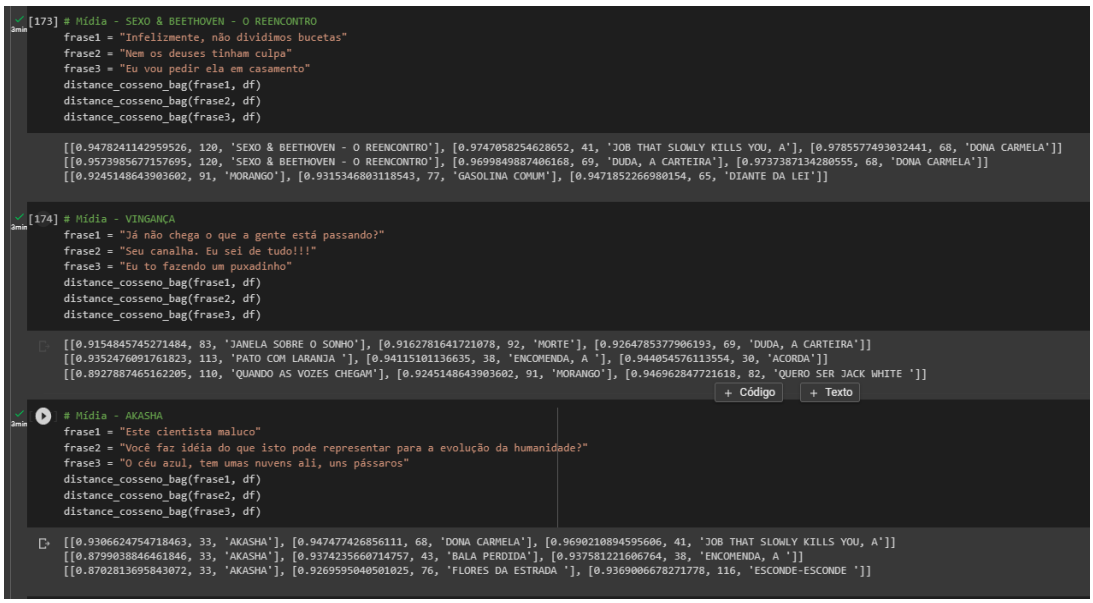
```
# Função distancia de cosseno - Bag of Words (BOW)
def distance_cosseno_bag(frase, df):
    sep = ' '
    d = []

    #Pré-processamento da Frase
    frase = preprocessamento(frase)
    t1 = [sep.join(frase)]

    for i, infos in df.iterrows():
        texto = preprocessamento(df["text_raw"][i])
        t1.append(sep.join(texto))
        vect_bag = CountVectorizer(binary=True)
        y = vect_bag.fit_transform(t Loading... ()
        v = distance.cosine(y[0], y[i+1])
        d.append([v, i, df["movie_name"][i]])
    r = sorted(d)
    print(r[0:3])
```

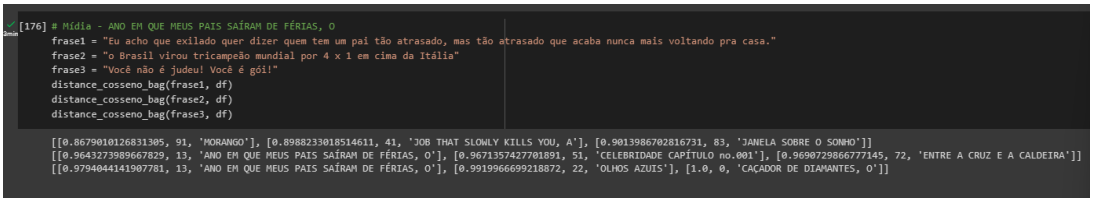
4. Resultados

Por fim, executamos a função e obtivemos os seguintes resultados, consultando três frases de cada roteiro criado:

```
[167] # Mídia - RÉ BEMOL
      frase1 = 'Ele desliga o telefone e volta a falar com seu amigo.'
      frase2 = 'Ele parece um presente ou um cantor de ópera.'
      frase3 = 'Uma barrinha que você aperta e por um momento a menina vive seu amor quando era linda.'
      distance_cosseno_bag(frase1, df)
      distance_cosseno_bag(frase2, df)
      distance_cosseno_bag(frase3, df)

      [[0.8848366400737804, 45, 'RÉ BEMOL '], [0.9119169670727945, 103, 'O PASSAGEIRO OBSCURO '], [0.9138339163735673, 81, 'INVERNO']]
      [[0.8969947594750791, 45, 'RÉ BEMOL '], [0.9575140711337913, 96, 'NOVA BANDEIRA PARA A NAÇÃO'], [0.9604406113935382, 115, 'RATOEIRA']]
      [[0.863736874917335, 45, 'RÉ BEMOL '], [0.9271763678098273, 81, 'INVERNO'], [0.9381961999383319, 86, 'LIÇÕES DE QUÍMICA']]

      # Mídia - GASOLINA COMUM
      frase1 = 'Um barulho muito alto os interrompe'
      frase2 = 'Regina abre o livro, olha para Carlo e começa a ler, apoiando suas costas na porta e esticando os pés sobre Carlo'
      frase3 = 'Um garotão barbudo de vinte e poucos anos sai de casa sem dar bola para o carro, conformado'
      distance_cosseno_bag(frase1, df)
      distance_cosseno_bag(frase2, df)
      distance_cosseno_bag(frase3, df)

      [[0.9241901956421097, 103, 'O PASSAGEIRO OBSCURO '], [0.9315346803118543, 77, 'GASOLINA COMUM'], [0.9419415250212863, 118, 'ENSAIO']]
      [[0.8574780718626078, 77, 'GASOLINA COMUM'], [0.8647327210895455, 73, 'A ESPERA'], [0.9142944593028991, 45, 'RÉ BEMOL ']]
      [[0.868898893978731, 77, 'GASOLINA COMUM'], [0.9206480399141845, 64, 'PLETORA DE IRMA'], [0.9211583472014556, 91, 'MORANGO']]

[169] # Mídia - NUNCA DEIXE DE ME OLHAR
      frase1 = 'Por favor, me acompanhe em um martini, então.'
      frase2 = 'Fernanda balança a cabeça negativamente.'
      frase3 = 'Uma TV de 49 polegadas exibe um jogo de futebol.'
      distance_cosseno_bag(frase1, df)
      distance_cosseno_bag(frase2, df)
      distance_cosseno_bag(frase3, df)

      [[0.9490352808562375, 38, 'ENCOMENDA, A '], [0.950970966215454, 33, 'AKASHA'], [0.951549841688849, 30, 'ACORDA']]
      [[0.9329598476846009, 118, 'ENSAIO'], [0.9343467835701387, 103, 'O PASSAGEIRO OBSCURO '], [0.9431017568641838, 63, ' LATA, A']]
      [[0.9423449939468246, 97, 'NUNCA DEIXE DE ME OLHAR'], [0.9575396112195776, 117, 'DO MUNDO NADA SE LEVA'], [0.9606707435882719, 81, 'INVERNO']]
```

```
      # Mídia - PICOLÉ BOIADO
      frase1 = 'O rapaz se levanta com as pernas trêmulas à procura de ajuda.'
      frase2 = 'Os dois permanecem admirando a mulher até a perderem de vista.'
      frase3 = 'Uma jovem do grupo volta em direção a Cláudio.'
      distance_cosseno_bag(frase1, df)
      distance_cosseno_bag(frase2, df)
      distance_cosseno_bag(frase3, df)

      [[0.8667653224947017, 106, 'PICOLÉ BOIADO'], [0.9341573078040779, 101, 'O QUE SOBE, DESCE '], [0.9427987444186599, 105, 'PEUR ET ABANDON']]
      [[0.8667653224947017, 106, 'PICOLÉ BOIADO'], [0.941006115382407 9, 81, 'INVERNO'], [0.9463943732581103, 103, 'O PASSAGEIRO OBSCURO ']]
      [[0.87837393614737, 106, 'PICOLÉ BOIADO'], [0.92815787918929, 50, 'CAFÉ DA TARDE'], [0.9321448856256023, 63, ' LATA, A']]

[171] # Mídia - INVERNO
      frase1 = 'Ana desliga o telefone e dá partida no carro.'
      frase2 = 'Vai dar tudo certo, meu amor, não fica assim.'
      frase3 = 'A imagem do céu é refletida em um dos olhos de Ana.'
      distance_cosseno_bag(frase1, df)
      distance_cosseno_bag(frase2, df)
      distance_cosseno_bag(frase3, df)

      [[0.8820122307648156, 81, 'INVERNO'], [0.9237507148336976, 85, 'LEO 1313'], [0.928813150318562 6, 66, 'DISPAROS']]
      [[0.8725586436671976, 81, 'INVERNO'], [0.8784766175197074, 82, 'QUERO SER JACK WHITE '], [0.8832030356434287, 45, 'RÉ BEMOL ']]
      [[0.892292395466959, 81, 'INVERNO'], [0.911726517049525, 64, 'PLETORA DE IRMA'], [0.9122941980692971, 33, 'AKASHA']]

[172] # Mídia - MEMÓRIAS PÓSTUMAS
      frase1 = "Podemos talvez chegar até fatos importantes dos anos 90"
      frase2 = "Não tive filhos, não transmiti a nenhuma criatura o legado da nossa miséria"
      frase3 = "Um velho escravo anda pela casa vazia"
      distance_cosseno_bag(frase1, df)
      distance_cosseno_bag(frase2, df)
      distance_cosseno_bag(frase3, df)

      [[0.9258750683338899, 91, 'MORANGO'], [0.9451911928568598, 92, 'MORTE'], [0.9466330705041162, 61, 'DEUS EX-MACHINA']]
      [[0.959708851798731, 89, 'MEMÓRIAS PÓSTUMAS'], [0.9727295263922994, 59, 'DESINIBIDA DO GRAJAÚ, A'], [0.973139234532487 3, 64, 'PLETORA DE IRMA']]
      [[0.9220187132634945, 66, 'DISPAROS'], [0.9304391656359747, 79, 'HOMENS   AO MAR'], [0.9345346329292022, 83, 'JANELA SOBRE O SONHO']]
```





De 30 resultados obtidos, tivemos 20 resultados corretos (66% de acurácia).