import sys,traceback

import os

import time

import shutil

import PyPDF2

from pdfminer.pdfinterp import PDFResourceManager, PDFPageInterpreter

from pdfminer.pdfpage import PDFPage

from pdfminer.pdfparser import PDFParser

from pdfminer.pdfdocument import PDFDocument

from pdfminer.pdfinterp import resolve1

from pdfminer.converter import TextConverter

from pdfminer.layout import LAParams

from cStringIO import StringIO

from sumy.parsers.plaintext import PlaintextParser

from sumy.nlp.tokenizers import Tokenizer

from sumy.summarizers.lsa import LsaSummarizer as Lsa

from sumy.summarizers.luhn import LuhnSummarizer as Luhn

from sumy.summarizers.text\_rank import TextRankSummarizer as TextRank

from sumy.summarizers.lex\_rank import LexRankSummarizer as LexRank

from sumy.summarizers.sum\_basic import SumBasicSummarizer as SumBasic

from sumy.summarizers.kl import KLSummarizer as KLsum

from sumy.nlp.stemmers import Stemmer

from sumy.utils import get\_stop\_words

#Functon to convert pdf into text

def convert(fname, pages=None):

infile = file(fname, 'rb')

content=""

parser =PDFParser(infile)

document=PDFDocument(parser)

# This will give you the count of pages

count= (resolve1(document.catalog['Pages'])['Count'])

if not pages:

pagenums = set()

else:

pagenums = count

# Check : print ('converting......')

codec = 'utf-8'

laparams = LAParams()

output = StringIO()

manager = PDFResourceManager()

converter = TextConverter(manager, output,codec=codec, laparams=laparams)

interpreter = PDFPageInterpreter(manager, converter)

for page in PDFPage.get\_pages(infile, pagenums):

interpreter.process\_page(page)

infile.close()

converter.close()

text = output.getvalue()

output.close

return text

# Summary Algorithms

def option(x):

return {

'1' : 'Luhn',

'2' : 'Lsa',

'3' : 'LexRank',

'4' : 'TextRank',

'5' : 'SumBasic',

'6' : 'KLsum',

'0' : 'exit',

}[x]

# Text files names with suffix

def pdfPatent(y):

return {

1 : '\_ABSTRACT',

2 : '\_BACKGROUND\_OF\_THE\_INVENTION',

3 : '\_BRIEF\_DESCRIPTION\_OF\_THE\_INVENTION',

4 : '\_BRIEF\_DESCRIPTION\_OF\_THE\_DRAWINGS',

5 : '\_DETAILED\_DESCRIPTION\_OF\_THE\_INVENTION',

6 : '\_CLAIM',

}[y]

########################## Main Program ########################

#Set parameters

LANGUAGE = "English"

SENTENCES\_COUNT = 30

sourcePDFFile =raw\_input("Enter Soruce file with path \n")

pdfName= raw\_input("Enter file name \n")

if os.path.exists(sourcePDFFile):

print('Found source PDF file')

PDF\_SummaryDir= raw\_input("Enter Output Directory path \n")

chooseAlgo = option(raw\_input("Select Algorithm \n press 1 and enter for Luhn. \n press 2 and enter for Lsa. \n press 3 and enter for LexRank. \n press 4 and enter for TextRank. \n press 5 and enter for SumBasic.\n press 6 and enter for KLsum.\n press 0 and enter to exit. \n"))

#Check if the directory PDF\_summary exists or not

if not os.path.exists(PDF\_SummaryDir):

os.makedirs(PDF\_SummaryDir)

outputTXTDir = os.path.dirname(PDF\_SummaryDir + '\Text\_Files\\')

if not os.path.exists(outputTXTDir):

os.makedirs(PDF\_SummaryDir + '\Text\_Files\\')

outputSummaryDir = os.path.dirname(PDF\_SummaryDir + '\Summary\\')

if not os.path.exists(outputSummaryDir):

os.makedirs(PDF\_SummaryDir + '\Summary\\')

#Name prefix for split files

outputNamePrefix = 'Patent'

timeSuffixSummary = str(time.strftime("%d-%m-%Y\_%H.%M.%S"))

# Append backslash to output dir for txt if necessary

if not outputTXTDir.endswith('//'):

outputTXTDir = outputTXTDir + '//'

# Append backslash to output dir for pdf if necessary

if not outputSummaryDir.endswith('//'):

outputSummaryDir = outputSummaryDir + '//'

#Check and Verify if PDF is ready for splitting

while not os.path.exists(sourcePDFFile):

print('Source PDF not found, sleeping...')

#Sleep

time.sleep(10)

if \_\_name\_\_ == '\_\_main\_\_':

text=convert(sourcePDFFile)

txtFileName = pdfName +'.txt'

# Calling convert function and writing each chapter to the .txt file

txtOutputFile = open(outputTXTDir + txtFileName, 'w')

txtOutputFile.write(text)

txtOutputFile.close()

with open(outputTXTDir + txtFileName, 'r') as fr:

l=pdfPatent(1)

fABS = open(outputTXTDir + pdfName +l+'.txt' , 'w')

l=pdfPatent(2)

fBII = open(outputTXTDir + pdfName +l+'.txt', 'w')

l=pdfPatent(3)

fSTI = open(outputTXTDir + pdfName + l +'.txt', 'w')

l=pdfPatent(4)

fBDD = open(outputTXTDir + pdfName + l+'.txt', 'w')

l=pdfPatent(5)

fDDI = open(outputTXTDir + pdfName + l+'.txt', 'w')

l=pdfPatent(6)

fCLA = open(outputTXTDir + pdfName + l+'.txt', 'w')

copy= False

for line in fr:

if line.startswith('ABSTRACT \n'):

copy = True

fABS.write(line)

elif line.startswith('BACKGROUND OF THE INVENTION'):

copy =False

elif copy:

fABS.write(''.join(line))

fABS.close()

fr.seek(0,0);

for line in fr:

if line.startswith('BACKGROUND OF THE INVENTION \n'):

copy = True

fBII.write(line)

elif(line.startswith('SUMMARY OF THE INVENTION \n') or line.startswith('BRIEF DESCRIPTION OF THE INVENTION \n')):

copy= False

elif copy :

fBII.write(''.join(line))

fBII.close()

fr.seek(0,0);

for line in fr :

if (line.startswith('SUMMARY OF THE INVENTION \n') or line.startswith('BRIEF DESCRIPTION OF THE INVENTION \n')):

copy = True

fSTI.write(line)

elif line.startswith('BRIEF DESCRIPTION OF THE DRAWING \n'):

copy = False

elif copy:

fSTI.write(''.join(line))

fSTI.close()

fr.seek(0,0);

for line in fr :

if line.startswith('BRIEF DESCRIPTION OF THE DRAWING \n'):

copy = True

fBDD.write(line)

elif (line.startswith('DETAILED DESCRIPTION OF THE \n\nINVENTION \n') or line.startswith('DETAILED DESCRIPTION \n')):

copy =False

elif copy:

fBDD.write(''.join(line))

fBDD.close()

fr.seek(0,0);

for line in fr :

if (line.startswith('DETAILED DESCRIPTION OF THE \n')):

copy= True

fDDI.write(line)

elif line == 'I claim: \n':

copy= False

elif copy :

fDDI.write(''.join(line))

fDDI.close()

fr.seek(0,0);

for line in fr:

if line.startswith('I claim: \n'):

copy= True

fCLA.write(line)

elif copy:

fCLA.write(''.join(line))

fCLA.close()

fr.close()

# for plain text files create Summary

for i in range(1,7):

txtOutputFile = open(outputTXTDir + pdfName + pdfPatent(i)+'.txt', 'r')

parser = PlaintextParser.from\_file(outputTXTDir + pdfName + pdfPatent(i)+'.txt', Tokenizer(LANGUAGE))

stemmer = Stemmer(LANGUAGE)

## Select from different algorithms to create summary by using different algorithms

if chooseAlgo == 'Lsa' :

summarizer = Lsa(stemmer)

elif chooseAlgo == 'LexRank':

summarizer = LexRank(stemmer)

elif chooseAlgo == 'TextRank':

summarizer = TextRank(stemmer)

elif chooseAlgo == 'Luhn':

summarizer = Luhn(stemmer)

elif chooseAlgo == 'SumBasic':

summarizer = SumBasic(stemmer)

elif chooseAlgo == 'KLsum':

summarizer = KLsum(stemmer)

else :

print ( 'Wrong Algorithm selected.')

sys.exit(0)

stop\_words = get\_stop\_words(LANGUAGE)

# Open file in append mode so that summary will be added at the bottom of file

summaryOutputFile = open(outputSummaryDir + chooseAlgo + pdfPatent(i) + '\_Summary\_File' + timeSuffixSummary + '.txt','a')

for sentence in summarizer(parser.document, SENTENCES\_COUNT):

# Check : print (sentence)

summaryOutputFile.write(str(sentence))

txtOutputFile.close()

summaryOutputFile.close()