# importing libraries

import speech\_recognition as sr

import os

from pydub import AudioSegment

from pydub.silence import split\_on\_silence

# create a speech recognition object

r = sr.Recognizer()

# a function to recognize speech in the audio file

# so that we don't repeat ourselves in in other functions

def transcribe\_audio(path):

# use the audio file as the audio source

with sr.AudioFile(path) as source:

audio\_listened = r.record(source)

# try converting it to text

text = r.recognize\_google(audio\_listened)

return text

# a function that splits the audio file into chunks on silence

# and applies speech recognition

def get\_large\_audio\_transcription\_on\_silence(path):

"""Splitting the large audio file into chunks

and apply speech recognition on each of these chunks"""

# open the audio file using pydub

sound = AudioSegment.from\_file(path)

# split audio sound where silence is 500 miliseconds or more and get chunks

chunks = split\_on\_silence(sound,

# experiment with this value for your target audio file

min\_silence\_len = 500,

# adjust this per requirement

silence\_thresh = sound.dBFS-14,

# keep the silence for 1 second, adjustable as well

keep\_silence=500,

)

folder\_name = "audio-chunks"

# create a directory to store the audio chunks

if not os.path.isdir(folder\_name):

os.mkdir(folder\_name)

whole\_text = ""

# process each chunk

for i, audio\_chunk in enumerate(chunks, start=1):

# export audio chunk and save it in

# the folder\_name directory.

chunk\_filename = os.path.join(folder\_name, f"chunk{i}.wav")

audio\_chunk.export(chunk\_filename, format="wav")

# recognize the chunk

try:

text = transcribe\_audio(chunk\_filename)

except sr.UnknownValueError as e:

print("Error:", str(e))

else:

text = f"{text.capitalize()}. "

print(chunk\_filename, ":", text)

whole\_text += text

# return the text for all chunks detected

return whole\_text

# a function that splits the audio file into fixed interval chunks

# and applies speech recognition

def get\_large\_audio\_transcription\_fixed\_interval(path, minutes=5):

"""Splitting the large audio file into fixed interval chunks

and apply speech recognition on each of these chunks"""

# open the audio file using pydub

sound = AudioSegment.from\_file(path)

# split the audio file into chunks

chunk\_length\_ms = int(1000 \* 60 \* minutes) # convert to milliseconds

chunks = [sound[i:i + chunk\_length\_ms] for i in range(0, len(sound), chunk\_length\_ms)]

folder\_name = "audio-fixed-chunks"

# create a directory to store the audio chunks

if not os.path.isdir(folder\_name):

os.mkdir(folder\_name)

whole\_text = ""

# process each chunk

for i, audio\_chunk in enumerate(chunks, start=1):

# export audio chunk and save it in

# the folder\_name directory.

chunk\_filename = os.path.join(folder\_name, f"chunk{i}.wav")

audio\_chunk.export(chunk\_filename, format="wav")

# recognize the chunk

try:

text = transcribe\_audio(chunk\_filename)

except sr.UnknownValueError as e:

print("Error:", str(e))

else:

text = f"{text.capitalize()}. "

print(chunk\_filename, ":", text)

whole\_text += text

# return the text for all chunks detected

return whole\_text

if name == 'main':

import sys

# path = "30-4447-0004.wav"

# path = "7601-291468-0006.wav"

path = "/content/machine-learning\_speech-recognition\_7601-291468-0006.wav"

print("\nFull text:", get\_large\_audio\_transcription\_on\_silence(path))

print("="\*50)

print("\nFull text:", get\_large\_audio\_transcription\_fixed\_interval(path, minutes=1/6))