**Speech Recognition Reading Large Audio Files**

If you want to perform speech recognition of a long audio file, then the below function handles that quite well:

import os

from pydub import AudioSegment

from pydub.silence import split\_on\_silence

# a function that splits the audio file into chunks

# and applies speech recognition

def get\_large\_audio\_transcription(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\_wav(path)

# split audio sound where silence is 700 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

with sr.AudioFile(chunk\_filename) as source:

audio\_listened = r.record(source)

# try converting it to text

try:

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

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

***Note:****You need to install [Pydub](https://github.com/jiaaro/pydub" \o "Pydub" \t "_blank) using pip for the above code to work.*

The above function uses split\_on\_silence() function from pydub.silence module to split audio data into chunks on silence. min\_silence\_len parameter is the minimum length of a silence to be used for a split.

silence\_thresh is the threshold in which anything quieter than this will be considered silence, I have set it to the average [dBFS](https://en.wikipedia.org/wiki/DBFS" \o "dBFS " \t "_blank) minus 14, keep\_silence argument is the amount of silence to leave at the beginning and the end of each chunk detected in milliseconds.

These parameters won't be perfect for all sound files, try to experiment with these parameters with your large audio needs.

After that, we iterate over all chunks and convert each speech audio into text and adding them up all together, here is an example run:

path = "7601-291468-0006.wav"

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

***Note****: You can get 7601-291468-0006.wav file*[*here*](https://github.com/x4nth055/pythoncode-tutorials/blob/master/machine-learning/speech-recognition/7601-291468-0006.wav)*.*

Output:

audio-chunks\chunk1.wav : His abode which you had fixed in a bowery or country seat.

audio-chunks\chunk2.wav : At a short distance from the city.

audio-chunks\chunk3.wav : Just at what is now called dutch street.

audio-chunks\chunk4.wav : Sooner bounded with proofs of his ingenuity.

audio-chunks\chunk5.wav : Patent smokejacks.

audio-chunks\chunk6.wav : It required a horse to work some.

audio-chunks\chunk7.wav : Dutch oven roasted meat without fire.

audio-chunks\chunk8.wav : Carts that went before the horses.

audio-chunks\chunk9.wav : Weather cox that turned against the wind and other wrongheaded contrivances.

audio-chunks\chunk10.wav : So just understand can found it all beholders.

Full text: His abode which you had fixed in a bowery or country seat. At a short distance from the city. Just at what is now called dutch street. Sooner bounded with proofs of his ingenuity. Patent smokejacks. It required a horse to work some. Dutch oven roasted meat without fire. Carts that went before the horses. Weather cox that turned against the wind and other wrongheaded contrivances. So just understand can found it all beholders.