video-summarizer-anne-joan-benita

April 24, 2024

1 Download YouTube Video's Audio

[ ]:
```
! pip install pytube -q
```

```
                                        57.6/57.6 kB
832.3 kB/s eta 0:00:00
```

[ ]:
```
from pytube import YouTube
```

[ ]:
```
VIDEO_URL = 'https://www.youtube.com/watch?v=h-JVjs9AAmQ' #batman
```

[ ]:
```
yt = YouTube(VIDEO_URL)
```

[ ]:
```
yt.streams \
.filter(only_audio=True, file_extension='mp4') \
.first() \
.download(filename='ytaudio.mp4')
```

[ ]:
```
'/content/ytaudio.mp4'
```

[ ]:
```
! ffmpeg -i ytaudio.mp4 -acodec pcm_s16le -ar 16000 ytaudio.wav
```

```
ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg
developers
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.22.04.1
--toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu
--incdir=/usr/include/x86_64-linux-gnu --arch=amd64 --enable-gpl --disable-
stripping --enable-gnutls --enable-ladspa --enable-libaom --enable-libass
--enable-libbluray --enable-libbs2b --enable-libcaca --enable-libcdio --enable-
libcodec2 --enable-libdav1d --enable-libflite --enable-libfontconfig --enable-
libfreetype --enable-libfribidi --enable-libgme --enable-libgsm --enable-libjack
--enable-libmp3lame --enable-libmysofa --enable-libopenjpeg --enable-libopenmpt
--enable-libopus --enable-libpulse --enable-librabbitmq --enable-librubberband
--enable-libshine --enable-libsnappy --enable-libsoxr --enable-libspeex
--enable-libsrt --enable-libssh --enable-libtheora --enable-libtwolame --enable-
libvidstab --enable-libvorbis --enable-libvpx --enable-libwebp --enable-libx265
```

```
--enable-libxml2 --enable-libxvid --enable-libzimg --enable-libzmq --enable-
libzvbi --enable-lv2 --enable-omx --enable-openal --enable-opencl --enable-
opengl --enable-sdl2 --enable-pocketsphinx --enable-librsvg --enable-libmfx
--enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint
--enable-frei0r --enable-libx264 --enable-shared
  libavutil      56. 70.100 / 56. 70.100
  libavcodec     58.134.100 / 58.134.100
  libavformat    58. 76.100 / 58. 76.100
  libavdevice    58. 13.100 / 58. 13.100
  libavfilter     7.110.100 /  7.110.100
  libswscale      5.  9.100 /  5.  9.100
  libswresample   3.  9.100 /  3.  9.100
  libpostproc    55.  9.100 / 55.  9.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'ytaudio.mp4':
  Metadata:
    major_brand     : dash
    minor_version   : 0
    compatible_brands: iso6mp41
    creation_time   : 2022-03-03T20:20:21.000000Z
  Duration: 00:04:15.65, start: 0.000000, bitrate: 48 kb/s
  Stream #0:0(eng): Audio: aac (HE-AAC) (mp4a / 0x6134706D), 44100 Hz, stereo,
fltp, 1 kb/s (default)
    Metadata:
      creation_time   : 2022-03-03T20:20:21.000000Z
      handler_name    : ISO Media file produced by Google Inc.
      vendor_id       : [0][0][0][0]
Stream mapping:
  Stream #0:0 -> #0:0 (aac (native) -> pcm_s16le (native))
Press [q] to stop, [?] for help
Output #0, wav, to 'ytaudio.wav':
  Metadata:
    major_brand     : dash
    minor_version   : 0
    compatible_brands: iso6mp41
    ISFT            : Lavf58.76.100
  Stream #0:0(eng): Audio: pcm_s16le ([1][0][0][0] / 0x0001), 16000 Hz, stereo,
s16, 512 kb/s (default)
    Metadata:
      creation_time   : 2022-03-03T20:20:21.000000Z
      handler_name    : ISO Media file produced by Google Inc.
      vendor_id       : [0][0][0][0]
      encoder         : Lavc58.134.100 pcm_s16le
size=   15978kB time=00:04:15.65 bitrate= 512.0kbits/s speed= 258x
video:0kB audio:15978kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 0.000477%
```

2 English ASR with HuggingSound

[ ]:
```
!pip install torch==2.2.1
!pip install huggingsound -q
```

```
Requirement already satisfied: torch==2.2.1 in /usr/local/lib/python3.10/dist-packages (2.2.1+cu121)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1) (3.13.3)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1) (4.10.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1) (3.2.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1) (3.1.3)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1) (2023.6.0)
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (from torch==2.2.1)
  Downloading nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (23.7 MB)
                           23.7/23.7 MB
36.8 MB/s eta 0:00:00
Collecting nvidia-cuda-runtime-cu12==12.1.105 (from torch==2.2.1)
  Downloading nvidia_cuda_runtime_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (823 kB)
                           823.6/823.6

kB 60.6 MB/s eta 0:00:00
Collecting nvidia-cuda-cupti-cu12==12.1.105 (from torch==2.2.1)
  Downloading nvidia_cuda_cupti_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (14.1 MB)
                           14.1/14.1 MB
49.9 MB/s eta 0:00:00
Collecting nvidia-cudnn-cu12==8.9.2.26 (from torch==2.2.1)
  Downloading nvidia_cudnn_cu12-8.9.2.26-py3-none-manylinux1_x86_64.whl (731.7 MB)
                           731.7/731.7

MB 1.4 MB/s eta 0:00:00
Collecting nvidia-cublas-cu12==12.1.3.1 (from torch==2.2.1)
  Downloading nvidia_cublas_cu12-12.1.3.1-py3-none-manylinux1_x86_64.whl (410.6 MB)

141.7/410.6 MB 2.0 MB/s eta 0:02:17
```

[ ]:
```python
from huggingsound import SpeechRecognitionModel
```

[ ]:
```python
import torch
device = "cuda" if torch.cuda.is_available() else "cpu"
```

[ ]:
```python

```

[ ]:
```python
model = SpeechRecognitionModel("jonatasgrosman/wav2vec2-large-xlsr-53-english",␣
  ↪device = device)
```

OUT OF MEMORY (OOM) error

3 Audio Chunking

[ ]:
```python
import librosa
```

[ ]:
```python
input_file = '/content/ytaudio.wav'
```

[ ]:
```python
print(librosa.get_samplerate(input_file))

# Stream over 30 seconds chunks rather than load the full file
stream = librosa.stream(
    input_file,
    block_length=30,
    frame_length=16000,
    hop_length=16000
)
```

[ ]:
```python
import soundfile as sf
```

[ ]:
```python
for i,speech in enumerate(stream):
  sf.write(f'{i}.wav', speech, 16000)
```

[ ]:
```python

```

4 Audio Transcription / ASR / Speech to Text

[ ]:
```python
audio_path =[]
for a in range(i+1):
  audio_path.append(f'/content/{a}.wav')
```

[ ]:
```python
audio_path
```

[ ]:
```python
transcriptions = model.transcribe(audio_path)
```

[ ]:
```python
full_transcript = ' '
```

[ ]:
```
for item in transcriptions:
  full_transcript += ''.join(item['transcription'])
```

[ ]:
```
len(full_transcript)
```

[ ]:
```

```

5 Text Summarization

[ ]:
```
from transformers import pipeline
```

[ ]:
```
summarization = pipeline('summarization')
```

[ ]:
```
summarized_text = summarization(full_transcript)
```

[ ]:
```
summarized_text[0]['summary_text']
```

Text Chunking before Summarization

[ ]:
```
num_iters = int(len(full_transcript)/1000)
summarized_text = []
for i in range(0, num_iters + 1):
  start = 0
  start = i * 1000
  end = (i + 1) * 1000
  #print("input text \n" + full_transcript[start:end])
  out = summarization(full_transcript[start:end], min_length = 5, max_length=20)
  out = out[0]
  out = out['summary_text']
 # print("Summarized text\n"+out)
  summarized_text.append(out)

#print(summarized_text)
```

[ ]:
```

```