```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

Reading merged data - EEG data along with demographic data

```
df = pd.read_csv('../input/eeg-dataset/merged_df.csv')
```

```
df.shape
```

```
(12811, 18)
```

```
df.head()
```

```
   SubjectID  VideoID  Attention  Mediation    Raw      Delta
Theta  \
0        0.0      0.0       56.0       43.0  278.0   301963.0
90612.0
1        0.0      0.0       40.0       35.0  -50.0    73787.0
28083.0
2        0.0      0.0       47.0       48.0  101.0   758353.0
383745.0
3        0.0      0.0       47.0       57.0   -5.0  2012240.0
129350.0
4        0.0      0.0       44.0       53.0   -8.0  1005145.0
354328.0

     Alpha1   Alpha2    Beta1     Beta2   Gamma1   Gamma2
predefinedlabel  \
0   33735.0  23991.0  27946.0   45097.0  33228.0   8293.0
0.0
1    1439.0   2240.0   2746.0    3687.0   5293.0   2740.0
0.0
2  201999.0  62107.0  36293.0  130536.0  57243.0  25354.0
0.0
3   61236.0  17084.0  11488.0   62462.0  49960.0  33932.0
0.0
4   37102.0  88881.0  45307.0   99603.0  44790.0  29749.0
0.0

   user-definedlabeln  age    ethnicity gender
0                 0.0   25  Han Chinese      M
1                 0.0   25  Han Chinese      M
2                 0.0   25  Han Chinese      M
3                 0.0   25  Han Chinese      M
4                 0.0   25  Han Chinese      M
```

```
df.columns
```

```
Index(['SubjectID', 'VideoID', 'Attention', 'Mediation', 'Raw',
'Delta',
       'Theta', 'Alpha1', 'Alpha2', 'Beta1', 'Beta2', 'Gamma1',
'Gamma2',
       'predefinedlabel', 'user-definedlabeln', 'age', 'ethnicity',
'gender'],
      dtype='object')
```

Categorical Column Encoding

One hot encoding - For ethnicity, we don't have any particular order so label encoding won't make any sense.

```
df = pd.get_dummies(df)

df.head()
```

```
   SubjectID  VideoID  Attention  Mediation    Raw      Delta
Theta  \
0        0.0      0.0       56.0       43.0  278.0   301963.0
90612.0
1        0.0      0.0       40.0       35.0  -50.0    73787.0
28083.0
2        0.0      0.0       47.0       48.0  101.0   758353.0
383745.0
3        0.0      0.0       47.0       57.0   -5.0  2012240.0
129350.0
4        0.0      0.0       44.0       53.0   -8.0  1005145.0
354328.0

     Alpha1   Alpha2    Beta1  ...   Gamma1   Gamma2  predefinedlabel
\
0   33735.0  23991.0  27946.0  ...  33228.0   8293.0              0.0

1    1439.0   2240.0   2746.0  ...   5293.0   2740.0              0.0

2  201999.0  62107.0  36293.0  ...  57243.0  25354.0              0.0

3   61236.0  17084.0  11488.0  ...  49960.0  33932.0              0.0

4   37102.0  88881.0  45307.0  ...  44790.0  29749.0              0.0


   user-definedlabeln  age  ethnicity_Bengali  ethnicity_English  \
0                 0.0   25                  0                  0
1                 0.0   25                  0                  0
2                 0.0   25                  0                  0
3                 0.0   25                  0                  0
4                 0.0   25                  0                  0
```

```
   ethnicity_Han Chinese  gender_F  gender_M
0                      1         0         1
1                      1         0         1
2                      1         0         1
3                      1         0         1
4                      1         0         1

[5 rows x 21 columns]
```

Feature Engineering from video data

- Video contains the scenario of professor teaching students. In this case only the content (audio) which speaker is speaking plays important role in detecting student's confusion level.

Steps For FE:

- Convert video to audio.
- Convert audio to text [Speech to text - https://towardsdatascience.com/speech-recognition-with-timestamps-934ede4234b2]
- Create text features
- Merge with original data but chopped first 30s and last 30s as EEG data contains chopped data only

```
df.columns

Index(['SubjectID', 'VideoID', 'Attention', 'Mediation', 'Raw',
'Delta',
       'Theta', 'Alpha1', 'Alpha2', 'Beta1', 'Beta2', 'Gamma1',
'Gamma2',
       'predefinedlabel', 'user-definedlabeln', 'age',
'ethnicity_Bengali',
       'ethnicity_English', 'ethnicity_Han Chinese', 'gender_F',
'gender_M'],
      dtype='object')

df.groupby(['SubjectID','VideoID']).size().reset_index(name='counts')

    SubjectID  VideoID  counts
0         0.0      0.0     144
1         0.0      1.0     140
2         0.0      2.0     142
3         0.0      3.0     122
4         0.0      4.0     116
..        ...      ...     ...
95        9.0      5.0     123
96        9.0      6.0     116
97        9.0      7.0     112
98        9.0      8.0     124
99        9.0      9.0     122
```

```
[100 rows x 3 columns]
```

```
df.columns
```

```
Index(['SubjectID', 'VideoID', 'Attention', 'Mediation', 'Raw',
'Delta',
       'Theta', 'Alpha1', 'Alpha2', 'Beta1', 'Beta2', 'Gamma1',
'Gamma2',
       'predefinedlabel', 'user-definedlabeln', 'age',
'ethnicity_Bengali',
       'ethnicity_English', 'ethnicity_Han Chinese', 'gender_F',
'gender_M'],
      dtype='object')
```

```
dff = pd.DataFrame()
dff =
df.groupby(['SubjectID','VideoID']).size().reset_index(name='counts')
dff['video_length_sec'] = dff['counts']*0.5
```

```
dff[dff['VideoID'] == 0]
```

```
    SubjectID  VideoID  counts  video_length_sec
0         0.0      0.0     144              72.0
10        1.0      0.0     140              70.0
20        2.0      0.0     140              70.0
30        3.0      0.0     140              70.0
40        4.0      0.0     140              70.0
50        5.0      0.0     144              72.0
60        6.0      0.0     140              70.0
70        7.0      0.0     140              70.0
80        8.0      0.0     140              70.0
90        9.0      0.0     144              72.0
```

Converting video to audio

```
!pip install moviepy
```

```
Collecting moviepy
  Downloading moviepy-1.0.3.tar.gz (388 kB)
     |████████████████████████████████| 388 kB 3.1 MB/s
etadata (setup.py) ... ent already satisfied: tqdm<5.0,>=4.11.2 in
/opt/conda/lib/python3.7/site-packages (from moviepy) (4.62.3)
Requirement already satisfied: requests<3.0,>=2.8.1 in
/opt/conda/lib/python3.7/site-packages (from moviepy) (2.26.0)
Collecting proglog<=1.0.0
  Downloading proglog-0.1.9.tar.gz (10 kB)
  Preparing metadata (setup.py) ... ent already satisfied:
numpy>=1.17.3 in /opt/conda/lib/python3.7/site-packages (from moviepy)
(1.20.3)
Requirement already satisfied: imageio<3.0,>=2.5 in
/opt/conda/lib/python3.7/site-packages (from moviepy) (2.9.0)
```

```
Collecting imageio_ffmpeg>=0.2.0
  Downloading imageio_ffmpeg-0.4.5-py3-none-manylinux2010_x86_64.whl
(26.9 MB)
     |████████████████████████████████| 26.9 MB 55.6 MB/s
ent already satisfied: pillow in /opt/conda/lib/python3.7/site-
packages (from imageio<3.0,>=2.5->moviepy) (8.2.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/opt/conda/lib/python3.7/site-packages (from requests<3.0,>=2.8.1-
>moviepy) (1.26.7)
Requirement already satisfied: idna<4,>=2.5 in
/opt/conda/lib/python3.7/site-packages (from requests<3.0,>=2.8.1-
>moviepy) (3.1)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.7/site-packages (from requests<3.0,>=2.8.1-
>moviepy) (2021.10.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/lib/python3.7/site-packages (from requests<3.0,>=2.8.1-
>moviepy) (2.0.9)
Building wheels for collected packages: moviepy, proglog
  Building wheel for moviepy (setup.py) ... oviepy: filename=moviepy-
1.0.3-py3-none-any.whl size=110744
sha256=99b0b91494b987af2d926d83b8ad89a01a957495cf90d600e0a538c0daa0783
a
  Stored in directory:
/root/.cache/pip/wheels/56/dc/2b/9cd600d483c04af3353d66623056fc03faed7
6b7518faae4df
  Building wheel for proglog (setup.py) ... e=proglog-0.1.9-py3-none-
any.whl size=6157
sha256=5fe3a11039d49c1cd34c9a9584134fb99d0328b0a022f9ced327f2543f38e58
1
  Stored in directory:
/root/.cache/pip/wheels/12/36/1f/dc61e6ac10781d63cf6fa045eb09fa613a667
384e12cb6e6e0
Successfully built moviepy proglog
Installing collected packages: proglog, imageio-ffmpeg, decorator,
moviepy
  Attempting uninstall: decorator
    Found existing installation: decorator 5.1.0
    Uninstalling decorator-5.1.0:
      Successfully uninstalled decorator-5.1.0
ERROR: pip's dependency resolver does not currently take into account
all the packages that are installed. This behaviour is the source of
the following dependency conflicts.
explainable-ai-sdk 1.3.2 requires xai-image-widget, which is not
installed.
gcsfs 2021.11.1 requires fsspec==2021.11.1, but you have fsspec
2022.1.0 which is incompatible.
Successfully installed decorator-4.4.2 imageio-ffmpeg-0.4.5 moviepy-
1.0.3 proglog-0.1.9
WARNING: Running pip as the 'root' user can result in broken
```

```
permissions and conflicting behaviour with the system package manager.
It is recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
```

Converting for 1 video

```
# Python module to convert video to audio
import moviepy.editor as mp
# Insert Local Video File Path
clip = mp.VideoFileClip(r"../input/eeg-dataset/videos/videos/0.m4v")

# Insert Local Audio File Path
clip.audio.write_audiofile(r"./0.wav")
```

```
MoviePy - Writing audio in ./0.wav



MoviePy - Done.
```

```
import IPython
IPython.display.Audio(r"./0.wav")
```

```
<IPython.lib.display.Audio object>
```

Above method works great

Conversion for all videos

```
%%time
import os

video_path = r'../input/eeg-dataset/videos/videos'
audio_path = r'/kaggle/working'


for i in range(0,10):
    print(i)
    # Insert Local Video File Path
    clip = mp.VideoFileClip(os.path.join(video_path,str(i)+'.m4v'))
    # Insert Local Audio File Path
    clip.audio.write_audiofile(os.path.join(audio_path ,str(i)
+'.wav'))
    # why wav format coz next voskapi we will use to convert audio to
speech it requires audio to be in wav format only.
```

```
0
MoviePy - Writing audio in /kaggle/working/0.wav
MoviePy - Done.
1
MoviePy - Writing audio in /kaggle/working/1.wav



MoviePy - Done.
2
MoviePy - Writing audio in /kaggle/working/2.wav



MoviePy - Done.
3
MoviePy - Writing audio in /kaggle/working/3.wav



MoviePy - Done.
4
MoviePy - Writing audio in /kaggle/working/4.wav



MoviePy - Done.
5
MoviePy - Writing audio in /kaggle/working/5.wav



MoviePy - Done.
6
MoviePy - Writing audio in /kaggle/working/6.wav



MoviePy - Done.
7
MoviePy - Writing audio in /kaggle/working/7.wav



MoviePy - Done.
8
MoviePy - Writing audio in /kaggle/working/8.wav



MoviePy - Done.
9
MoviePy - Writing audio in /kaggle/working/9.wav
MoviePy - Done.
CPU times: user 11.6 s, sys: 982 ms, total: 12.6 s
Wall time: 18.9 s
```

Audio to Text Conversion

Experiment 1 - Using vosk library

https://gitlab.com/Winston-90/foreign_speech_recognition/-/blob/main/timestamps/word_timestamps.ipynb

- Conversion quality very low

```
!pip install vosk

Collecting vosk
  Downloading vosk-0.3.32-py3-none-
manylinux_2_12_x86_64.manylinux2010_x86_64.whl (6.9 MB)
     |████████████████████████████████| 6.9 MB 4.3 MB/s
ent already satisfied: cffi>=1.0 in /opt/conda/lib/python3.7/site-
packages (from vosk) (1.15.0)
Requirement already satisfied: pycparser in
/opt/conda/lib/python3.7/site-packages (from cffi>=1.0->vosk) (2.21)
Installing collected packages: vosk
Successfully installed vosk-0.3.32
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager.
It is recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
```

```
%%time
import wave
import json
import os

from vosk import Model, KaldiRecognizer, SetLogLevel
# https://alphacephei.com/vosk/models

model_path = r"C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\SELF-
CASE-STUDY2\EDA\archive\confused_eeg\vosk-model-en-us-0.22-lgraph"

if not os.path.exists(model_path):
    print(f"Please download the model from
https://alphacephei.com/vosk/models and unpack as {model_path}")
    sys.exit()

print(f"Reading your vosk model '{model_path}'...")
model = Model(model_path)
```

```
print(f"'{model_path}' model was successfully read")
#wf = wave.open(audio_filename, "rb")
```

```
Reading your vosk model 'C:\Users\Palak\APPLIED_AI\APPLIED_AI\
Assignments\SELF-CASE-STUDY2\EDA\archive\confused_eeg\vosk-model-en-
us-0.22-lgraph'...
'C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\SELF-CASE-STUDY2\
EDA\archive\confused_eeg\vosk-model-en-us-0.22-lgraph' model was
successfully read
Wall time: 11 s
```

Specify the file name to recognize

```
 # name of the audio file to recognize
audio_filename = r"C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\
SELF-CASE-STUDY2\EDA\archive\confused_eeg\audios\0.wav"
# name of the text file to write recognized text
text_filename = r"C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\
SELF-CASE-STUDY2\EDA\archive\confused_eeg\text\0.txt"
```

Reading a file

```
if not os.path.exists(audio_filename):
    print(f"File '{audio_filename}' doesn't exist")
    sys.exit()

print(f"Reading your file '{audio_filename}'...")
wf = wave.open(audio_filename, "rb")
print(f"'{audio_filename}' file was successfully read")
```

```
Reading your file 'C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\
SELF-CASE-STUDY2\EDA\archive\confused_eeg\audios\0.wav'...
'C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\SELF-CASE-STUDY2\
EDA\archive\confused_eeg\audios\0.wav' file was successfully read
```

```
%%time
rec = KaldiRecognizer(model, wf.getframerate())
rec.SetWords(True)

# get the list of JSON dictionaries
results = []
# recognize speech using vosk model
while True:
    data = wf.readframes(4000)
    if len(data) == 0:
        break
    if rec.AcceptWaveform(data):
        part_result = json.loads(rec.Result())
        results.append(part_result)
part_result = json.loads(rec.FinalResult())
results.append(part_result)
```

```
Wall time: 3min 29s
```

results

```
[{'result': [{'conf': 0.541602, 'end': 1.738075, 'start': 1.2, 'word':
'to'}],
  'text': 'to'},
 {'result': [{'conf': 0.68404,
    'end': 6.953499,
    'start': 6.653952,
    'word': 'oh'},
   {'conf': 0.647987, 'end': 7.410192, 'start': 6.962921, 'word':
'shit'}],
  'text': 'oh shit'},
 {'result': [{'conf': 0.601531,
    'end': 9.333923,
    'start': 9.123091,
    'word': 'or'},
   {'conf': 0.31854, 'end': 9.905264, 'start': 9.517781, 'word':
'for'}],
  'text': 'or for'},
 {'result': [{'conf': 0.896106,
    'end': 12.632261,
    'start': 12.302658,
    'word': 'sure'}],
  'text': 'sure'},
 {'result': [{'conf': 0.310123,
    'end': 14.999899,
    'start': 14.698611,
    'word': 'she'}],
  'text': 'she'},
 {'result': [{'conf': 0.249143,
    'end': 17.827773,
    'start': 17.528328,
    'word': 'or'},
   {'conf': 0.937296, 'end': 19.14, 'start': 18.72, 'word': 'so'}],
  'text': 'or so'},
 {'result': [{'conf': 0.99281,
    'end': 20.849357,
    'start': 20.43,
    'word': 'for'}],
  'text': 'for'},
 {'result': [{'conf': 0.28607,
    'end': 23.340137,
    'start': 22.89,
    'word': 'shoot'},
   {'conf': 0.989509, 'end': 24.210337, 'start': 23.701284, 'word':
'from'},
   {'conf': 0.3861, 'end': 24.72, 'start': 24.3, 'word': 'school'}],
  'text': 'shoot from school'},
 {'text': ''},
 {'result': [{'conf': 0.529036,
    'end': 33.625199,
    'start': 33.009199,
    'word': 'hm'}],
  'text': 'hm'},
 {'text': ''},
 {'result': [{'conf': 0.314733,
    'end': 42.716843,
    'start': 41.88,
    'word': 'purple'},
   {'conf': 0.696889, 'end': 43.681879, 'start': 43.112622, 'word':
'or'}],
  'text': 'purple or'},
 {'result': [{'conf': 0.343635,
    'end': 45.868879,
    'start': 45.27,
    'word': 'from'}],
  'text': 'from'},
 {'result': [{'conf': 0.995355,
    'end': 51.019314,
    'start': 49.576007,
    'word': 'the'},
   {'conf': 0.327455, 'end': 52.106638, 'start': 51.46028, 'word':
'for'}],
  'text': 'the for'},
 {'text': ''},
 {'result': [{'conf': 0.28426, 'end': 59.46, 'start': 59.07, 'word':
'beyond'},
   {'conf': 0.389963, 'end': 62.279971, 'start': 61.976074, 'word':
'her'}],
  'text': 'beyond her'},
 {'text': ''},
 {'result': [{'conf': 0.681612,
    'end': 70.196213,
    'start': 69.445576,
    'word': 'for'}],
  'text': 'for'},
 {'text': ''},
 {'result': [{'conf': 0.413473,
    'end': 78.629685,
    'start': 77.935137,
    'word': 'for'}],
  'text': 'for'},
 {'result': [{'conf': 0.70619,
    'end': 83.080232,
    'start': 81.908745,
    'word': 'the'},
   {'conf': 0.6894, 'end': 83.84937, 'start': 83.309897, 'word':
'warned'}],
  'text': 'the warned'},
 {'result': [{'conf': 0.975806,
    'end': 88.795554,
    'start': 87.040144,
    'word': 'the'},
   {'conf': 0.85247, 'end': 89.519751, 'start': 88.92, 'word':
'time'}],
  'text': 'the time'},
 {'result': [{'conf': 0.794158,
    'end': 91.95,
    'start': 91.378184,
    'word': 'home'}],
  'text': 'home'},
 {'result': [{'conf': 0.63829,
    'end': 99.090652,
    'start': 98.49,
    'word': 'cause'},
   {'conf': 0.770357, 'end': 100.194829, 'start': 99.793557, 'word':
'poor'},
   {'conf': 0.987579, 'end': 100.92, 'start': 100.289795, 'word':
'term'}],
  'text': 'cause poor term'},
 {'text': ''},
 {'result': [{'conf': 0.659821,
    'end': 109.320132,
    'start': 109.200535,
    'word': 'a'},
   {'conf': 0.879217, 'end': 109.710037, 'start': 109.325405, 'word':
'month'},
   {'conf': 0.808838, 'end': 110.07, 'start': 109.711692, 'word':
'long'}],
  'text': 'a month long'},
 {'text': ''},
 {'result': [{'conf': 0.656508,
    'end': 117.904863,
    'start': 116.861843,
    'word': 'the'},
   {'conf': 0.58388, 'end': 118.288352, 'start': 117.904863, 'word':
'for'}],
  'text': 'the for'},
 {'result': [{'conf': 0.900545,
    'end': 123.197886,
    'start': 121.932656,
    'word': 'the'},
   {'conf': 0.656273, 'end': 123.96, 'start': 123.3, 'word':
'crash'}],
  'text': 'the crash'},
 {'text': ''},
 {'result': [{'conf': 0.781072, 'end': 134.52, 'start': 134.16,
'word': 'oh'},
   {'conf': 0.655244, 'end': 134.85, 'start': 134.58, 'word': 'man'},
   {'conf': 0.994041, 'end': 136.666948, 'start': 135.019629, 'word':
```

```
'the'}],
  'text': 'oh man the'},
 {'result': [{'conf': 0.686891,
    'end': 142.53,
    'start': 141.944912,
    'word': 'said'}],
  'text': 'said'},
 {'result': [{'conf': 0.743723,
    'end': 143.369692,
    'start': 142.949971,
    'word': 'web'},
   {'conf': 0.716136, 'end': 144.059663, 'start': 143.37, 'word':
'server'}],
  'text': 'web server'},
 {'result': [{'conf': 0.395693,
    'end': 149.459561,
    'start': 149.07126,
    'word': 'or'}],
  'text': 'or'},
 {'result': [{'conf': 0.884837,
    'end': 151.02,
    'start': 150.78,
    'word': 'and'}],
  'text': 'and'},
 {'result': [{'conf': 0.485148, 'end': 156.78, 'start': 156.6, 'word':
'i'},
   {'conf': 0.468105, 'end': 157.378462, 'start': 156.780176, 'word':
'have'}],
  'text': 'i have'},
 {'result': [{'conf': 0.371155,
    'end': 159.705732,
    'start': 159.363428,
    'word': 'the'}],
  'text': 'the'},
 {'text': ''},
 {'result': [{'conf': 0.362782,
    'end': 166.319253,
    'start': 166.080366,
    'word': 'from'},
   {'conf': 0.742024, 'end': 166.833823, 'start': 166.333374, 'word':
'house'},
   {'conf': 0.418479, 'end': 167.7, 'start': 167.1, 'word':
'though'}],
  'text': 'from house though'},
 {'result': [{'conf': 0.949942,
    'end': 169.667021,
    'start': 168.7627,
    'word': 'the'},
   {'conf': 0.740169, 'end': 170.49, 'start': 169.667021, 'word':
'wolves'}],
  'text': 'the wolves'},
 {'result': [{'conf': 0.599899,
    'end': 173.94,
    'start': 173.49,
    'word': 'for'}],
  'text': 'for'},
 {'result': [{'conf': 0.634687,
    'end': 175.859297,
    'start': 175.13228,
    'word': 'the'},
   {'conf': 0.945647, 'end': 179.52, 'start': 179.28, 'word': 'and'}],
  'text': 'the and'},
 {'result': [{'conf': 0.763466,
    'end': 181.589517,
    'start': 180.995332,
    'word': 'the'}],
  'text': 'the'},
 {'result': [{'conf': 0.462695,
    'end': 186.367397,
    'start': 185.762329,
    'word': 'the'},
   {'conf': 0.296373, 'end': 187.2, 'start': 186.416953, 'word':
'wash'}],
  'text': 'the wash'},
 {'result': [{'conf': 0.635884,
    'end': 190.979868,
    'start': 190.355537,
    'word': 'yeah'}],
  'text': 'yeah'},
 {'result': [{'conf': 0.300005,
    'end': 197.322729,
    'start': 196.935615,
    'word': 'the'}],
  'text': 'the'},
 {'text': ''},
 {'result': [{'conf': 0.960272,
    'end': 203.46,
    'start': 202.89,
    'word': 'wow'}],
  'text': 'wow'},
 {'result': [{'conf': 0.393794,
    'end': 204.93,
    'start': 204.186943,
    'word': 'operation'}],
  'text': 'operation'},
 {'result': [{'conf': 0.345803,
    'end': 209.1877,
    'start': 208.26,
    'word': 'caution'},
   {'conf': 0.1972, 'end': 209.518872, 'start': 209.1877, 'word':
```

```
'sure'}],
  'text': 'caution sure'},
 {'result': [{'conf': 1.0,
    'end': 214.461694,
    'start': 211.990415,
    'word': 'the'},
   {'conf': 0.904147, 'end': 215.25, 'start': 214.59, 'word':
'match'}],
  'text': 'the match'},
 {'result': [{'conf': 0.606998,
    'end': 220.255269,
    'start': 220.01083,
    'word': 'for'},
   {'conf': 0.939467, 'end': 220.56, 'start': 220.255269, 'word':
'your'},
   {'conf': 0.99344, 'end': 221.85, 'start': 221.37, 'word': 'turn'}],
  'text': 'for your turn'},
 {'result': [{'conf': 0.237288,
    'end': 224.64,
    'start': 224.120244,
    'word': 'yeah'}],
  'text': 'yeah'},
 {'result': [{'conf': 0.457605,
    'end': 228.66,
    'start': 228.09,
    'word': 'herbs'}],
  'text': 'herbs'},
 {'result': [{'conf': 0.476797,
    'end': 233.4,
    'start': 232.77,
    'word': 'sean'}],
  'text': 'sean'},
 {'result': [{'conf': 0.57177,
    'end': 235.214736,
    'start': 234.72397,
    'word': 'the'},
   {'conf': 0.311531, 'end': 238.436191, 'start': 237.93, 'word':
'for'}],
  'text': 'the for'},
 {'result': [{'conf': 0.45976, 'end': 241.05, 'start': 240.66, 'word':
'up'}],
  'text': 'up'},
 {'result': [{'conf': 0.42123, 'end': 249.21, 'start': 248.04, 'word':
'ash'}],
  'text': 'ash'},
 {'text': ''},
 {'result': [{'conf': 0.522244,
    'end': 255.866836,
    'start': 255.53376,
    'word': 'the'},
   {'conf': 0.694687, 'end': 258.45, 'start': 257.94, 'word':
'earth'}],
  'text': 'the earth'},
 {'result': [{'conf': 0.483227,
    'end': 260.73,
    'start': 260.123086,
    'word': 'happens'}],
  'text': 'happens'},
 {'text': ''},
 {'result': [{'conf': 0.842914,
    'end': 268.712021,
    'start': 268.56,
    'word': 'at'},
   {'conf': 0.900846, 'end': 269.04, 'start': 268.712314, 'word':
'some'},
   {'conf': 0.726156, 'end': 269.88, 'start': 269.399941, 'word':
'time'}],
  'text': 'at some time'},
 {'result': [{'conf': 0.753293,
    'end': 274.437158,
    'start': 273.403477,
    'word': 'the'},
   {'conf': 0.201982, 'end': 276.087686, 'start': 275.46, 'word':
'rawr'},
   {'conf': 0.46003, 'end': 276.359619, 'start': 276.087686, 'word':
'or'}],
  'text': 'the rawr or'},
 {'result': [{'conf': 0.382886,
    'end': 280.323398,
    'start': 280.076865,
    'word': 'yeah'}],
  'text': 'yeah'},
 {'text': ''}]
```

```python
list_of_words = []
for ele in results:
    word = ele.get('text', {})

    if word is not None:
        list_of_words.append(word)
```

```python
' '.join(list_of_words)
```

```
'to oh shit or for sure she or so for shoot from school  hm  purple or
from the for  beyond her  for  for the warned the time home cause poor
term  a month long  the for the crash  oh man the said web server or
and i have the  from house though the wolves for the and the the wash
yeah the  wow operation caution sure the match for your turn yeah
```

```
herbs sean the for up ash  the earth happens  at some time the rawr or
yeah '
```

Observation

- Conversion quality is too low. Original video was about, proton,nuetron electron.

Audio to Text Conversion - Experiment 2 - SpeechRecognition

- Conversion quality good but doesnt give timestamp along with word which is necessary to merge these feautures with other data

```
#!pip install SpeechRecognition

import speech_recognition as sr

r = sr.Recognizer()
harvard = sr.AudioFile(r'C:\Users\Palak\APPLIED_AI\APPLIED_AI\
Assignments\SELF-CASE-STUDY2\EDA\archive\confused_eeg\audios\0.wav')

with harvard as source:
    audio = r.record(source)

audio
```

```
<speech_recognition.AudioData at 0x189a6c4b1f0>
```

```
%%time
r.recognize_google(audio)
```

```
Wall time: 32 s

'UPS in the nucleus or sometimes called protons something stall
neutrons and outsiders something called electronic that all the atomic
structure we need then we say entities have a property call electric
charge the symbol for electric charge you and you can put a subscript
to say who are talking about you can say you for the neutron is zero
for the electron is -1.69 then the -19 and it matter in cooler for the
proton is really helped me put it this way you for the proton is a
positive number feel the beat -9 and now the importance of the coulomb
if there is anything has some cool Ramzan it it will interact with
anything else that has some cool on Sunday to that if you have to
entity and that one has it hard to one coulomb that when it started to
grow long and the distance between them is R then the four is you want
you to or Epsilon 0 9 square and preferably not putting on the back
requires an app that it takes too long but you all know what the
answer namely you want the hold onto you do when will be repulsive if
you want you to the same sign and point in the direction to anyone
thank you that supports lock with hardware not name is called book
club but will happen only right that the difference between being
Newton and being hook hook is known for the earthquake or you are
known for their one over earthquake'
```

Observation

- Translation seems almost perfect.
- But this library doesn't provide timestamp along with word translation which is a major drawback.

Audio to text - Experiment 3 - Deepspeech module

- Conversion quality is moderate
- Timestamps are also given along with word prediction

```
!pip3 install deepspeech
```

```
Collecting deepspeech
  Downloading deepspeech-0.9.3-cp37-cp37m-manylinux1_x86_64.whl (9.2 
MB)
     |████████████████████████████████| 9.2 MB 4.2 MB/s 
ent already satisfied: numpy>=1.14.5 in /opt/conda/lib/python3.7/site-
packages (from deepspeech) (1.20.3)
Installing collected packages: deepspeech
Successfully installed deepspeech-0.9.3
WARNING: Running pip as the 'root' user can result in broken 
permissions and conflicting behaviour with the system package manager.
It is recommended to use a virtual environment instead: 
https://pip.pypa.io/warnings/venv
```

```
!wget 
https://github.com/mozilla/DeepSpeech/releases/download/v0.9.3/deepspe
ech-0.9.3-models.pbmm
!wget 
https://github.com/mozilla/DeepSpeech/releases/download/v0.9.3/deepspe
ech-0.9.3-models.scorer
```

```
--2022-01-29 05:33:11--  
https://github.com/mozilla/DeepSpeech/releases/download/v0.9.3/deepspe
ech-0.9.3-models.pbmm
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-
release-asset-2e65be/60273704/8b25f180-3b0f-11eb-8fc1-de4f4ec3b5a3?X-
Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A
%2F20220129%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
Date=20220129T053311Z&X-Amz-Expires=300&X-Amz-
Signature=876a295bac424a2e5b5e66ab98373154e607e97f7fc4be7718ece2f87015
b4c8&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=60273704&response-
content-disposition=attachment%3B%20filename%3Ddeepspeech-0.9.3-
models.pbmm&response-content-type=application%2Foctet-stream 
[following]
--2022-01-29 05:33:11--  https://objects.githubusercontent.com/github-
production-release-asset-2e65be/60273704/8b25f180-3b0f-11eb-8fc1-
de4f4ec3b5a3?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
```

```
Credential=AKIAIWNJYAX4CSVEH53A%2F20220129%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20220129T053311Z&X-Amz-Expires=300&X-
Amz-
Signature=876a295bac424a2e5b5e66ab98373154e607e97f7fc4be7718ece2f87015
b4c8&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=60273704&response-
content-disposition=attachment%3B%20filename%3Ddeepspeech-0.9.3-
models.pbmm&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com
(objects.githubusercontent.com)... 185.199.109.133, 185.199.111.133,
185.199.110.133, ...
Connecting to objects.githubusercontent.com
(objects.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 188915987 (180M) [application/octet-stream]
Saving to: ‘deepspeech-0.9.3-models.pbmm’

deepspeech-0.9.3-mo 100%[===================>] 180.16M   101MB/s    in
1.8s

2022-01-29 05:33:13 (101 MB/s) - ‘deepspeech-0.9.3-models.pbmm’ saved
[188915987/188915987]

--2022-01-29 05:33:14--
https://github.com/mozilla/DeepSpeech/releases/download/v0.9.3/deepspe
ech-0.9.3-models.scorer
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-
release-asset-2e65be/60273704/924cff80-3b0f-11eb-878c-cacaa2a0d946?X-
Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A
%2F20220129%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
Date=20220129T053314Z&X-Amz-Expires=300&X-Amz-
Signature=59e0472b63ce8130816d49ec496847939ee97e02f99898ace9337ea697aa
1f0b&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=60273704&response-
content-disposition=attachment%3B%20filename%3Ddeepspeech-0.9.3-
models.scorer&response-content-type=application%2Foctet-stream
[following]
--2022-01-29 05:33:14--  https://objects.githubusercontent.com/github-
production-release-asset-2e65be/60273704/924cff80-3b0f-11eb-878c-
cacaa2a0d946?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20220129%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20220129T053314Z&X-Amz-Expires=300&X-
Amz-
Signature=59e0472b63ce8130816d49ec496847939ee97e02f99898ace9337ea697aa
1f0b&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=60273704&response-
content-disposition=attachment%3B%20filename%3Ddeepspeech-0.9.3-
```

```
models.scorer&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com
(objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133,
185.199.110.133, ...
Connecting to objects.githubusercontent.com
(objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 953363776 (909M) [application/octet-stream]
Saving to: ‘deepspeech-0.9.3-models.scorer’

deepspeech-0.9.3-mo 100%[===================>] 909.20M   101MB/s    in
8.3s

2022-01-29 05:33:22 (109 MB/s) - ‘deepspeech-0.9.3-models.scorer’
saved [953363776/953363776]
```

```
!apt -qq install -y sox # this module automatically resamples input
audio from 44100 hx to 16000hz which is required by deepspeech
```

```
The following additional packages will be installed:
  libmagic-mgc libmagic1 libopencore-amrnb0 libopencore-amrwb0 libsox-
fmt-alsa
  libsox-fmt-base libsox3
Suggested packages:
  file libsox-fmt-all
The following NEW packages will be installed:
  libmagic-mgc libmagic1 libopencore-amrnb0 libopencore-amrwb0 libsox-
fmt-alsa
  libsox-fmt-base libsox3 sox
0 upgraded, 8 newly installed, 0 to remove and 17 not upgraded.
Need to get 807 kB of archives.
After this operation, 7649 kB of additional disk space will be used.

7agic-mgc.
(Reading database ... 100797 files and directories currently
installed.)
Preparing to unpack .../0-libmagic-mgc_1%3a5.38-4_amd64.deb ...
7Progress: [  0%]
[.......................................................]
87Progress: [  3%]
[#......................................................]
8Unpacking libmagic-mgc (1:5.38-4) ...
7Progress: [  6%]
[###....................................................]
8Selecting previously unselected package libmagic1:amd64.
Preparing to unpack .../1-libmagic1_1%3a5.38-4_amd64.deb ...
7Progress: [  9%]
[#####..................................................]
8Unpacking libmagic1:amd64 (1:5.38-4) ...
```

```
7Progress: [ 12%]
[#######...................................................]
8Selecting previously unselected package libopencore-amrnb0:amd64.
Preparing to unpack .../2-libopencore-amrnb0_0.1.5-1_amd64.deb ...
7Progress: [ 15%]
[########..................................................]
8Unpacking libopencore-amrnb0:amd64 (0.1.5-1) ...
7Progress: [ 18%]
[##########................................................]
8Selecting previously unselected package libopencore-amrwb0:amd64.
Preparing to unpack .../3-libopencore-amrwb0_0.1.5-1_amd64.deb ...
7Progress: [ 21%]
[############..............................................]
8Unpacking libopencore-amrwb0:amd64 (0.1.5-1) ...
7Progress: [ 24%]
[##############............................................]
8Selecting previously unselected package libsox3:amd64.
Preparing to unpack .../4-libsox3_14.4.2+git20190427-2_amd64.deb ...
7Progress: [ 27%]
[###############...........................................]
8Unpacking libsox3:amd64 (14.4.2+git20190427-2) ...
7Progress: [ 30%]
[#################.........................................]
8Selecting previously unselected package libsox-fmt-alsa:amd64.
Preparing to unpack .../5-libsox-fmt-alsa_14.4.2+git20190427-
2_amd64.deb ...
7Progress: [ 33%]
[###################.......................................]
8Unpacking libsox-fmt-alsa:amd64 (14.4.2+git20190427-2) ...
7Progress: [ 36%]
[#####################.....................................]
8Selecting previously unselected package libsox-fmt-base:amd64.
Preparing to unpack .../6-libsox-fmt-base_14.4.2+git20190427-
2_amd64.deb ...
7Progress: [ 39%]
[######################....................................]
8Unpacking libsox-fmt-base:amd64 (14.4.2+git20190427-2) ...
7Progress: [ 42%]
[########################..................................]
8Selecting previously unselected package sox.
Preparing to unpack .../7-sox_14.4.2+git20190427-2_amd64.deb ...
7Progress: [ 45%]
[##########################................................]
8Unpacking sox (14.4.2+git20190427-2) ...
7Progress: [ 48%]
[############################..............................] 8Setting
up libmagic-mgc (1:5.38-4) ...
7Progress: [ 52%]
[#############################.............................]
87Progress: [ 55%]
```

```
[###############################...........................] 8Setting
up libmagic1:amd64 (1:5.38-4) ...
7Progress: [ 58%]
[#################################.........................]
87Progress: [ 61%]
[###################################.......................] 8Setting
up libopencore-amrwb0:amd64 (0.1.5-1) ...
7Progress: [ 64%]
[####################################......................]
87Progress: [ 67%]
[######################################....................] 8Setting
up libopencore-amrnb0:amd64 (0.1.5-1) ...
7Progress: [ 70%]
[########################################..................]
87Progress: [ 73%]
[##########################################................] 8Setting
up libsox3:amd64 (14.4.2+git20190427-2) ...
7Progress: [ 76%]
[###########################################...............]
87Progress: [ 79%]
[#############################################.............] 8Setting
up libsox-fmt-alsa:amd64 (14.4.2+git20190427-2) ...
7Progress: [ 82%]
[###############################################...........]
87Progress: [ 85%]
[#################################################.........] 8Setting
up libsox-fmt-base:amd64 (14.4.2+git20190427-2) ...
7Progress: [ 88%]
[##################################################........]
87Progress: [ 91%]
[####################################################......] 8Setting
up sox (14.4.2+git20190427-2) ...
7Progress: [ 94%]
[######################################################....]
87Progress: [ 97%]
[########################################################..]
8Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for mime-support (3.64ubuntu1) ...

7
```

Conversion of 1 audio to text

```
%%time
# Transcribe an audio file to json with timestamp of speech spoken

!deepspeech --model ./deepspeech-0.9.3-models.pbmm --
scorer ./deepspeech-0.9.3-models.scorer --audio './0.wav' --json >
'0.json'
```

```
Loading model from file ./deepspeech-0.9.3-models.pbmm
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:35:44.213701: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loaded model in 0.0209s.
Loading scorer from files ./deepspeech-0.9.3-models.scorer
Loaded scorer in 0.0014s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 98.733s for 141.880s audio file.
CPU times: user 2.53 s, sys: 432 ms, total: 2.96 s
Wall time: 1min 40s
```

```
import json

# Opening JSON file
f = open(r'./0.json')
 # returns JSON object as
# a dictionary
data = json.load(f)
data
```

```
{'transcripts': [{'confidence': -2496.58447265625,
   'words': [{'word': 'a', 'start_time': 0.44, 'duration': 0.06},
    {'word': 'new', 'start_time': 0.54, 'duration': 0.16},
    {'word': 'piece', 'start_time': 0.74, 'duration': 1.74},
    {'word': 'in', 'start_time': 2.64, 'duration': 0.06},
    {'word': 'the', 'start_time': 2.72, 'duration': 0.06},
    {'word': 'nuclear', 'start_time': 2.84, 'duration': 0.52},
    {'word': 'or', 'start_time': 3.48, 'duration': 0.18},
    {'word': 'some', 'start_time': 3.72, 'duration': 0.64},
    {'word': 'things', 'start_time': 4.4, 'duration': 0.2},
    {'word': 'called', 'start_time': 4.66, 'duration': 0.18},
    {'word': 'protons', 'start_time': 4.88, 'duration': 1.3},
    {'word': "something's", 'start_time': 6.3, 'duration': 0.38},
    {'word': 'called', 'start_time': 6.74, 'duration': 0.18},
    {'word': 'neutrons', 'start_time': 6.96, 'duration': 1.18},
    {'word': 'and', 'start_time': 8.28, 'duration': 0.52},
    {'word': 'outside', 'start_time': 8.94, 'duration': 0.38},
    {'word': 'or', 'start_time': 9.38, 'duration': 0.12},
    {'word': 'something', 'start_time': 9.56, 'duration': 0.32},
    {'word': 'to', 'start_time': 9.94, 'duration': 0.14},
    {'word': 'electrons', 'start_time': 10.14, 'duration': 1.22},
    {'word': "that's", 'start_time': 11.46, 'duration': 0.14},
{'word': 'all', 'start_time': 11.68, 'duration': 0.08},
{'word': 'the', 'start_time': 11.8, 'duration': 0.08},
{'word': 'atomic', 'start_time': 11.92, 'duration': 0.36},
{'word': 'structure', 'start_time': 12.3, 'duration': 0.88},
{'word': 'we', 'start_time': 13.26, 'duration': 0.12},
{'word': 'need', 'start_time': 13.44, 'duration': 3.12},
{'word': 'then', 'start_time': 16.66, 'duration': 0.4},
{'word': 'we', 'start_time': 17.18, 'duration': 0.22},
{'word': 'say', 'start_time': 17.5, 'duration': 1.6},
{'word': 'certain', 'start_time': 19.18, 'duration': 0.38},
{'word': 'entities', 'start_time': 19.66, 'duration': 1.06},
{'word': 'have', 'start_time': 20.8, 'duration': 0.14},
{'word': 'a', 'start_time': 21.02, 'duration': 0.04},
{'word': 'properly', 'start_time': 21.12, 'duration': 0.5},
{'word': 'called', 'start_time': 21.72, 'duration': 0.56},
{'word': 'electrons', 'start_time': 22.36, 'duration': 0.44},
{'word': 'charge', 'start_time': 22.86, 'duration': 1.94},
{'word': 'the', 'start_time': 24.88, 'duration': 0.1},
{'word': 'symbol', 'start_time': 25.02, 'duration': 0.26},
{'word': 'forelegs', 'start_time': 25.34, 'duration': 1.04},
{'word': 'is', 'start_time': 26.48, 'duration': 0.16},
{'word': 'cue', 'start_time': 26.76, 'duration': 2.42},
{'word': 'and', 'start_time': 29.28, 'duration': 0.08},
{'word': 'you', 'start_time': 29.4, 'duration': 0.08},
{'word': 'can', 'start_time': 29.52, 'duration': 0.08},
{'word': 'put', 'start_time': 29.64, 'duration': 0.12},
{'word': 'a', 'start_time': 29.82, 'duration': 0.06},
{'word': 'subscript', 'start_time': 29.92, 'duration': 0.48},
{'word': 'to', 'start_time': 30.44, 'duration': 0.08},
{'word': 'say', 'start_time': 30.58, 'duration': 0.12},
{'word': 'who', 'start_time': 30.76, 'duration': 0.12},
{'word': 'you', 'start_time': 30.92, 'duration': 0.06},
{'word': 'are', 'start_time': 31.02, 'duration': 0.08},
{'word': 'talking', 'start_time': 31.12, 'duration': 0.26},
{'word': 'about', 'start_time': 31.44, 'duration': 1.26},
{'word': 'so', 'start_time': 32.78, 'duration': 0.08},
{'word': 'you', 'start_time': 32.88, 'duration': 0.12},
{'word': 'can', 'start_time': 33.04, 'duration': 0.2},
{'word': 'say', 'start_time': 33.32, 'duration': 0.8},
{'word': 'upon', 'start_time': 34.28, 'duration': 0.62},
{'word': 'drawn', 'start_time': 34.94, 'duration': 1.26},
{'word': 'is', 'start_time': 36.28, 'duration': 0.14},
{'word': 'there', 'start_time': 36.48, 'duration': 2.02},
{'word': 'a', 'start_time': 38.6, 'duration': 0.08},
{'word': 'few', 'start_time': 38.72, 'duration': 0.16},
{'word': 'for', 'start_time': 38.92, 'duration': 0.1},
{'word': 'the', 'start_time': 39.06, 'duration': 0.06},
{'word': 'electron', 'start_time': 39.16, 'duration': 1.6},
{'word': 'is', 'start_time': 40.88, 'duration': 0.14},
{'word': 'minus', 'start_time': 41.08, 'duration': 0.38},
{'word': 'one', 'start_time': 41.58, 'duration': 0.14},
{'word': 'point', 'start_time': 41.78, 'duration': 0.32},
{'word': 'six', 'start_time': 42.16, 'duration': 0.24},
{'word': 'times', 'start_time': 42.48, 'duration': 1.06},
{'word': 'then', 'start_time': 43.62, 'duration': 0.2},
{'word': 'the', 'start_time': 43.86, 'duration': 0.08},
{'word': 'mines', 'start_time': 43.98, 'duration': 0.32},
{'word': 'nineteen', 'start_time': 44.34, 'duration': 0.94},
{'word': 'and', 'start_time': 45.36, 'duration': 0.08},
{'word': 'his', 'start_time': 45.48, 'duration': 0.14},
{'word': 'measured', 'start_time': 45.68, 'duration': 0.26},
{'word': 'in', 'start_time': 46.02, 'duration': 0.12},
{'word': 'colors', 'start_time': 46.18, 'duration': 3.04},
{'word': 'the', 'start_time': 49.28, 'duration': 0.08},
{'word': 'foorth', 'start_time': 49.44, 'duration': 0.54},
{'word': 'proton', 'start_time': 50.06, 'duration': 2.74},
{'word': 'is', 'start_time': 52.92, 'duration': 0.1},
{'word': 'really', 'start_time': 53.12, 'duration': 1.42},
{'word': 'helepolis', 'start_time': 54.6, 'duration': 3.7},
{'word': 'efore', 'start_time': 58.42, 'duration': 0.3},
{'word': 'protospathaire', 'start_time': 58.76, 'duration': 0.9},
{'word': 'numbers', 'start_time': 59.74, 'duration': 0.38},
{'word': 'so', 'start_time': 60.16, 'duration': 0.1},
{'word': 'sealyhams', 'start_time': 60.38, 'duration': 6.74},
{'word': 'now', 'start_time': 67.2, 'duration': 0.12},
{'word': 'the', 'start_time': 67.4, 'duration': 0.08},
{'word': 'importance', 'start_time': 67.52, 'duration': 0.32},
{'word': 'of', 'start_time': 67.9, 'duration': 0.06},
{'word': 'the', 'start_time': 67.98, 'duration': 0.08},
{'word': 'colon', 'start_time': 68.12, 'duration': 1.74},
{'word': 'is', 'start_time': 69.94, 'duration': 0.1},
{'word': 'that', 'start_time': 70.08, 'duration': 0.12},
{'word': 'if', 'start_time': 70.26, 'duration': 0.1},
{'word': 'anything', 'start_time': 70.44, 'duration': 0.4},
{'word': 'has', 'start_time': 70.94, 'duration': 0.24},
{'word': 'some', 'start_time': 71.26, 'duration': 0.14},
{'word': 'cool', 'start_time': 71.46, 'duration': 0.16},
{'word': 'amonition', 'start_time': 71.68, 'duration': 1.96},
{'word': 'interact', 'start_time': 73.7, 'duration': 0.38},
{'word': 'with', 'start_time': 74.12, 'duration': 0.14},
{'word': 'anything', 'start_time': 74.32, 'duration': 0.28},
{'word': 'else', 'start_time': 74.74, 'duration': 0.3},
{'word': 'that', 'start_time': 75.1, 'duration': 0.16},
{'word': 'has', 'start_time': 75.32, 'duration': 0.12},
{'word': 'some', 'start_time': 75.48, 'duration': 0.18},
{'word': 'columns', 'start_time': 75.72, 'duration': 2.36},
{'word': 'that', 'start_time': 78.1, 'duration': 0.12},
{'word': 'if', 'start_time': 78.3, 'duration': 0.04},
{'word': 'you', 'start_time': 78.38, 'duration': 0.18},
{'word': 'have', 'start_time': 78.66, 'duration': 1.06},
{'word': 'to', 'start_time': 79.78, 'duration': 0.1},
{'word': 'entertain', 'start_time': 79.98, 'duration': 3.1},
{'word': 'one', 'start_time': 83.18, 'duration': 0.1},
{'word': 'has', 'start_time': 83.34, 'duration': 0.16},
{'word': 'the', 'start_time': 83.56, 'duration': 0.12},
{'word': 'charge', 'start_time': 83.74, 'duration': 0.32},
{'word': 'of', 'start_time': 84.14, 'duration': 0.1},
{'word': 'concreteness', 'start_time': 84.36, 'duration': 2.48},
{'word': 'charge', 'start_time': 86.9, 'duration': 0.2},
{'word': 'a', 'start_time': 87.18, 'duration': 0.06},
{'word': 'butaca', 'start_time': 87.32, 'duration': 1.92},
{'word': 'and', 'start_time': 89.32, 'duration': 0.06},
{'word': 'the', 'start_time': 89.42, 'duration': 0.1},
{'word': 'distance', 'start_time': 89.58, 'duration': 0.32},
{'word': 'between', 'start_time': 89.94, 'duration': 0.26},
{'word': 'them', 'start_time': 90.24, 'duration': 0.12},
{'word': 'his', 'start_time': 90.4, 'duration': 0.22},
{'word': 'arm', 'start_time': 90.76, 'duration': 1.98},
{'word': 'then', 'start_time': 92.78, 'duration': 0.16},
{'word': 'the', 'start_time': 93.0, 'duration': 0.12},
{'word': 'force', 'start_time': 93.2, 'duration': 1.54},
{'word': 'is', 'start_time': 94.84, 'duration': 0.2},
{'word': 'come', 'start_time': 95.14, 'duration': 0.22},
{'word': 'one', 'start_time': 95.46, 'duration': 0.8},
{'word': 'utopia', 'start_time': 96.42, 'duration': 3.04},
{'word': 'alonzo', 'start_time': 99.5, 'duration': 1.68},
{'word': 'or', 'start_time': 101.3, 'duration': 0.2},
{'word': 'square', 'start_time': 101.52, 'duration': 0.36},
{'word': 'and', 'start_time': 102.02, 'duration': 0.12},
{'word': 'purposely', 'start_time': 102.18, 'duration': 0.36},
{'word': 'not', 'start_time': 102.62, 'duration': 0.12},
{'word': 'putting', 'start_time': 102.82, 'duration': 0.92},
{'word': 'all', 'start_time': 103.86, 'duration': 0.1},
{'word': 'the', 'start_time': 103.98, 'duration': 0.06},
{'word': 'victories', 'start_time': 104.1, 'duration': 0.64},
{'word': 'unhasting', 'start_time': 104.82, 'duration': 1.8},
{'word': 'but', 'start_time': 106.64, 'duration': 0.1},
{'word': 'you', 'start_time': 106.78, 'duration': 0.1},
{'word': 'all', 'start_time': 106.94, 'duration': 0.1},
{'word': 'know', 'start_time': 107.08, 'duration': 0.1},
{'word': 'what', 'start_time': 107.24, 'duration': 0.1},
{'word': 'the', 'start_time': 107.38, 'duration': 0.1},
{'word': 'answer', 'start_time': 107.54, 'duration': 0.3},
{'word': 'is', 'start_time': 107.9, 'duration': 0.74},
{'word': 'namely', 'start_time': 108.72, 'duration': 0.88},
{'word': 'if', 'start_time': 109.66, 'duration': 0.04},
{'word': 'you', 'start_time': 109.74, 'duration': 0.08},
{'word': 'want', 'start_time': 109.86, 'duration': 0.16},
{'word': 'the', 'start_time': 110.06, 'duration': 0.1},
{'word': 'force', 'start_time': 110.22, 'duration': 0.26},
{'word': 'on', 'start_time': 110.6, 'duration': 0.2},
{'word': 'two', 'start_time': 110.88, 'duration': 0.88},
{'word': 'you', 'start_time': 111.84, 'duration': 0.16},
{'word': 'do', 'start_time': 112.02, 'duration': 0.16},
{'word': 'one', 'start_time': 112.28, 'duration': 0.86},
{'word': 'would', 'start_time': 113.3, 'duration': 0.14},
{'word': 'be', 'start_time': 113.48, 'duration': 0.4},
{'word': 'repulsive', 'start_time': 113.98, 'duration': 0.96},
{'word': 'if', 'start_time': 115.04, 'duration': 0.12},
{'word': 'corantoes', 'start_time': 115.22, 'duration': 1.36},
{'word': 'sign', 'start_time': 116.64, 'duration': 0.54},
{'word': 'and', 'start_time': 117.28, 'duration': 0.18},
{'word': 'pointed', 'start_time': 117.52, 'duration': 0.34},
{'word': 'the', 'start_time': 117.88, 'duration': 0.08},
{'word': 'direction', 'start_time': 118.0, 'duration': 1.06},
{'word': 'joining', 'start_time': 119.12, 'duration': 0.36},
{'word': 'them', 'start_time': 119.52, 'duration': 0.82},
{'word': 'up', 'start_time': 120.5, 'duration': 3.58},
{'word': 'yes', 'start_time': 124.14, 'duration': 1.42},
{'word': 'thank', 'start_time': 125.62, 'duration': 0.2},
{'word': 'you', 'start_time': 125.86, 'duration': 1.6},
{'word': 'there', 'start_time': 127.5, 'duration': 0.1},
{'word': 'is', 'start_time': 127.64, 'duration': 0.04},
{'word': 'another', 'start_time': 127.72, 'duration': 0.28},
{'word': 'for', 'start_time': 128.1, 'duration': 0.24},
{'word': 'logic', 'start_time': 128.44, 'duration': 0.46},
{'word': 'is', 'start_time': 128.96, 'duration': 0.1},
{'word': 'archidamus', 'start_time': 129.16, 'duration': 1.3},
{'word': 'call', 'start_time': 130.54, 'duration': 0.16},
{'word': 'books', 'start_time': 130.72, 'duration': 0.24},
{'word': 'loustalot', 'start_time': 131.02, 'duration': 1.3},
{'word': 'right', 'start_time': 132.42, 'duration': 1.64},
{'word': "that's", 'start_time': 134.1, 'duration': 0.14},
{'word': 'the', 'start_time': 134.26, 'duration': 0.08},
{'word': 'difference', 'start_time': 134.38, 'duration': 0.26},
{'word': 'between', 'start_time': 134.7, 'duration': 0.44},
{'word': 'being', 'start_time': 135.2, 'duration': 0.18},
{'word': 'newton', 'start_time': 135.48, 'duration': 0.36},
{'word': 'and', 'start_time': 135.92, 'duration': 0.1},
{'word': 'being', 'start_time': 136.08, 'duration': 0.2},
{'word': 'cook', 'start_time': 136.3, 'duration': 0.42},
{'word': 'who', 'start_time': 136.78, 'duration': 0.14},
{'word': 'is', 'start_time': 136.98, 'duration': 0.1},
{'word': 'known', 'start_time': 137.1, 'duration': 0.16},
{'word': 'for', 'start_time': 137.3, 'duration': 0.1},
{'word': 'the', 'start_time': 137.42, 'duration': 0.08},
{'word': 'arthquake', 'start_time': 137.6, 'duration': 0.48},
{'word': 'low', 'start_time': 138.16, 'duration': 0.54},
{'word': 'lukannon', 'start_time': 138.78, 'duration': 0.58},
{'word': 'for', 'start_time': 139.4, 'duration': 0.12},
    {'word': 'the', 'start_time': 139.56, 'duration': 0.6},
    {'word': 'one', 'start_time': 140.26, 'duration': 0.08},
    {'word': 'over', 'start_time': 140.4, 'duration': 0.2},
    {'word': 'acquire', 'start_time': 140.66, 'duration': 0.38}]},
  {'confidence': -2497.45947265625,
   'words': [{'word': 'a', 'start_time': 0.44, 'duration': 0.06},
    {'word': 'new', 'start_time': 0.54, 'duration': 0.16},
    {'word': 'piece', 'start_time': 0.74, 'duration': 1.74},
    {'word': 'in', 'start_time': 2.64, 'duration': 0.06},
    {'word': 'the', 'start_time': 2.72, 'duration': 0.06},
    {'word': 'nuclear', 'start_time': 2.84, 'duration': 0.52},
    {'word': 'or', 'start_time': 3.48, 'duration': 0.18},
    {'word': 'some', 'start_time': 3.72, 'duration': 0.64},
    {'word': 'things', 'start_time': 4.4, 'duration': 0.2},
    {'word': 'called', 'start_time': 4.66, 'duration': 0.18},
    {'word': 'protons', 'start_time': 4.88, 'duration': 1.3},
    {'word': "something's", 'start_time': 6.3, 'duration': 0.38},
    {'word': 'called', 'start_time': 6.74, 'duration': 0.18},
    {'word': 'neutrons', 'start_time': 6.96, 'duration': 1.18},
    {'word': 'and', 'start_time': 8.28, 'duration': 0.52},
    {'word': 'outside', 'start_time': 8.94, 'duration': 0.38},
    {'word': 'or', 'start_time': 9.38, 'duration': 0.12},
    {'word': 'something', 'start_time': 9.56, 'duration': 0.32},
    {'word': 'to', 'start_time': 9.94, 'duration': 0.14},
    {'word': 'electrons', 'start_time': 10.14, 'duration': 1.22},
    {'word': "that's", 'start_time': 11.46, 'duration': 0.14},
    {'word': 'all', 'start_time': 11.68, 'duration': 0.08},
    {'word': 'the', 'start_time': 11.8, 'duration': 0.08},
    {'word': 'atomic', 'start_time': 11.92, 'duration': 0.36},
    {'word': 'structure', 'start_time': 12.3, 'duration': 0.88},
    {'word': 'we', 'start_time': 13.26, 'duration': 0.12},
    {'word': 'need', 'start_time': 13.44, 'duration': 3.12},
    {'word': 'then', 'start_time': 16.66, 'duration': 0.4},
    {'word': 'we', 'start_time': 17.18, 'duration': 0.22},
    {'word': 'say', 'start_time': 17.5, 'duration': 1.6},
    {'word': 'certain', 'start_time': 19.18, 'duration': 0.38},
    {'word': 'entities', 'start_time': 19.66, 'duration': 1.06},
    {'word': 'have', 'start_time': 20.8, 'duration': 0.14},
    {'word': 'a', 'start_time': 21.02, 'duration': 0.04},
    {'word': 'properly', 'start_time': 21.12, 'duration': 0.5},
    {'word': 'called', 'start_time': 21.72, 'duration': 0.56},
    {'word': 'electrons', 'start_time': 22.36, 'duration': 0.44},
    {'word': 'charge', 'start_time': 22.86, 'duration': 1.94},
    {'word': 'the', 'start_time': 24.88, 'duration': 0.1},
    {'word': 'symbol', 'start_time': 25.02, 'duration': 0.26},
    {'word': 'forelegs', 'start_time': 25.34, 'duration': 1.04},
    {'word': 'is', 'start_time': 26.48, 'duration': 0.16},
    {'word': 'cue', 'start_time': 26.76, 'duration': 2.42},
    {'word': 'and', 'start_time': 29.28, 'duration': 0.08},
    {'word': 'you', 'start_time': 29.4, 'duration': 0.08},
{'word': 'can', 'start_time': 29.52, 'duration': 0.08},
{'word': 'put', 'start_time': 29.64, 'duration': 0.12},
{'word': 'a', 'start_time': 29.82, 'duration': 0.06},
{'word': 'subscript', 'start_time': 29.92, 'duration': 0.48},
{'word': 'to', 'start_time': 30.44, 'duration': 0.08},
{'word': 'say', 'start_time': 30.58, 'duration': 0.12},
{'word': 'who', 'start_time': 30.76, 'duration': 0.12},
{'word': 'you', 'start_time': 30.92, 'duration': 0.06},
{'word': 'are', 'start_time': 31.02, 'duration': 0.08},
{'word': 'talking', 'start_time': 31.12, 'duration': 0.26},
{'word': 'about', 'start_time': 31.44, 'duration': 1.26},
{'word': 'so', 'start_time': 32.78, 'duration': 0.08},
{'word': 'you', 'start_time': 32.88, 'duration': 0.12},
{'word': 'can', 'start_time': 33.04, 'duration': 0.2},
{'word': 'say', 'start_time': 33.32, 'duration': 0.8},
{'word': 'upon', 'start_time': 34.28, 'duration': 0.62},
{'word': 'drawn', 'start_time': 34.94, 'duration': 1.26},
{'word': 'is', 'start_time': 36.28, 'duration': 0.14},
{'word': 'there', 'start_time': 36.48, 'duration': 2.02},
{'word': 'a', 'start_time': 38.6, 'duration': 0.08},
{'word': 'few', 'start_time': 38.72, 'duration': 0.16},
{'word': 'for', 'start_time': 38.92, 'duration': 0.1},
{'word': 'the', 'start_time': 39.06, 'duration': 0.06},
{'word': 'electron', 'start_time': 39.16, 'duration': 1.6},
{'word': 'is', 'start_time': 40.88, 'duration': 0.14},
{'word': 'minus', 'start_time': 41.08, 'duration': 0.38},
{'word': 'one', 'start_time': 41.58, 'duration': 0.14},
{'word': 'point', 'start_time': 41.78, 'duration': 0.32},
{'word': 'six', 'start_time': 42.16, 'duration': 0.24},
{'word': 'times', 'start_time': 42.48, 'duration': 1.06},
{'word': 'then', 'start_time': 43.62, 'duration': 0.2},
{'word': 'the', 'start_time': 43.86, 'duration': 0.08},
{'word': 'mines', 'start_time': 43.98, 'duration': 0.32},
{'word': 'nineteen', 'start_time': 44.34, 'duration': 0.94},
{'word': 'and', 'start_time': 45.36, 'duration': 0.08},
{'word': 'his', 'start_time': 45.48, 'duration': 0.14},
{'word': 'measured', 'start_time': 45.68, 'duration': 0.26},
{'word': 'in', 'start_time': 46.02, 'duration': 0.12},
{'word': 'colors', 'start_time': 46.18, 'duration': 3.04},
{'word': 'the', 'start_time': 49.28, 'duration': 0.08},
{'word': 'foorth', 'start_time': 49.44, 'duration': 0.54},
{'word': 'proton', 'start_time': 50.06, 'duration': 2.74},
{'word': 'is', 'start_time': 52.92, 'duration': 0.1},
{'word': 'really', 'start_time': 53.12, 'duration': 1.42},
{'word': 'helepolis', 'start_time': 54.6, 'duration': 3.7},
{'word': 'efore', 'start_time': 58.42, 'duration': 0.3},
{'word': 'protospathaire', 'start_time': 58.76, 'duration': 0.9},
{'word': 'numbers', 'start_time': 59.74, 'duration': 0.38},
{'word': 'so', 'start_time': 60.16, 'duration': 0.1},
{'word': 'sealyhams', 'start_time': 60.38, 'duration': 6.74},
{'word': 'now', 'start_time': 67.2, 'duration': 0.12},
{'word': 'the', 'start_time': 67.4, 'duration': 0.08},
{'word': 'importance', 'start_time': 67.52, 'duration': 0.32},
{'word': 'of', 'start_time': 67.9, 'duration': 0.06},
{'word': 'the', 'start_time': 67.98, 'duration': 0.08},
{'word': 'colon', 'start_time': 68.12, 'duration': 1.74},
{'word': 'is', 'start_time': 69.94, 'duration': 0.1},
{'word': 'that', 'start_time': 70.08, 'duration': 0.12},
{'word': 'if', 'start_time': 70.26, 'duration': 0.1},
{'word': 'anything', 'start_time': 70.44, 'duration': 0.4},
{'word': 'has', 'start_time': 70.94, 'duration': 0.24},
{'word': 'some', 'start_time': 71.26, 'duration': 0.14},
{'word': 'cool', 'start_time': 71.46, 'duration': 0.16},
{'word': 'amonition', 'start_time': 71.68, 'duration': 1.96},
{'word': 'interact', 'start_time': 73.7, 'duration': 0.38},
{'word': 'with', 'start_time': 74.12, 'duration': 0.14},
{'word': 'anything', 'start_time': 74.32, 'duration': 0.28},
{'word': 'else', 'start_time': 74.74, 'duration': 0.3},
{'word': 'that', 'start_time': 75.1, 'duration': 0.16},
{'word': 'has', 'start_time': 75.32, 'duration': 0.12},
{'word': 'some', 'start_time': 75.48, 'duration': 0.18},
{'word': 'columns', 'start_time': 75.72, 'duration': 2.36},
{'word': 'that', 'start_time': 78.1, 'duration': 0.12},
{'word': 'if', 'start_time': 78.3, 'duration': 0.04},
{'word': 'you', 'start_time': 78.38, 'duration': 0.18},
{'word': 'have', 'start_time': 78.66, 'duration': 1.06},
{'word': 'to', 'start_time': 79.78, 'duration': 0.1},
{'word': 'entertain', 'start_time': 79.98, 'duration': 3.1},
{'word': 'one', 'start_time': 83.18, 'duration': 0.1},
{'word': 'has', 'start_time': 83.34, 'duration': 0.16},
{'word': 'the', 'start_time': 83.56, 'duration': 0.12},
{'word': 'charge', 'start_time': 83.74, 'duration': 0.32},
{'word': 'of', 'start_time': 84.14, 'duration': 0.1},
{'word': 'concreteness', 'start_time': 84.36, 'duration': 2.48},
{'word': 'charge', 'start_time': 86.9, 'duration': 0.2},
{'word': 'a', 'start_time': 87.18, 'duration': 0.06},
{'word': 'butaca', 'start_time': 87.32, 'duration': 1.92},
{'word': 'and', 'start_time': 89.32, 'duration': 0.06},
{'word': 'the', 'start_time': 89.42, 'duration': 0.1},
{'word': 'distance', 'start_time': 89.58, 'duration': 0.32},
{'word': 'between', 'start_time': 89.94, 'duration': 0.26},
{'word': 'them', 'start_time': 90.24, 'duration': 0.12},
{'word': 'his', 'start_time': 90.4, 'duration': 0.22},
{'word': 'arm', 'start_time': 90.76, 'duration': 1.98},
{'word': 'then', 'start_time': 92.78, 'duration': 0.16},
{'word': 'the', 'start_time': 93.0, 'duration': 0.12},
{'word': 'force', 'start_time': 93.2, 'duration': 1.54},
{'word': 'is', 'start_time': 94.84, 'duration': 0.2},
{'word': 'come', 'start_time': 95.14, 'duration': 0.22},
{'word': 'one', 'start_time': 95.46, 'duration': 0.8},
{'word': 'utopia', 'start_time': 96.42, 'duration': 3.04},
{'word': 'alonzo', 'start_time': 99.5, 'duration': 1.68},
{'word': 'or', 'start_time': 101.3, 'duration': 0.2},
{'word': 'square', 'start_time': 101.52, 'duration': 0.36},
{'word': 'and', 'start_time': 102.02, 'duration': 0.12},
{'word': 'purposely', 'start_time': 102.18, 'duration': 0.36},
{'word': 'not', 'start_time': 102.62, 'duration': 0.12},
{'word': 'putting', 'start_time': 102.82, 'duration': 0.92},
{'word': 'all', 'start_time': 103.86, 'duration': 0.1},
{'word': 'the', 'start_time': 103.98, 'duration': 0.06},
{'word': 'victories', 'start_time': 104.1, 'duration': 0.64},
{'word': 'unhasting', 'start_time': 104.82, 'duration': 1.8},
{'word': 'but', 'start_time': 106.64, 'duration': 0.1},
{'word': 'you', 'start_time': 106.78, 'duration': 0.1},
{'word': 'all', 'start_time': 106.94, 'duration': 0.1},
{'word': 'know', 'start_time': 107.08, 'duration': 0.1},
{'word': 'what', 'start_time': 107.24, 'duration': 0.1},
{'word': 'the', 'start_time': 107.38, 'duration': 0.1},
{'word': 'answer', 'start_time': 107.54, 'duration': 0.3},
{'word': 'is', 'start_time': 107.9, 'duration': 0.74},
{'word': 'namely', 'start_time': 108.72, 'duration': 0.88},
{'word': 'if', 'start_time': 109.66, 'duration': 0.04},
{'word': 'you', 'start_time': 109.74, 'duration': 0.08},
{'word': 'want', 'start_time': 109.86, 'duration': 0.16},
{'word': 'the', 'start_time': 110.06, 'duration': 0.1},
{'word': 'force', 'start_time': 110.22, 'duration': 0.26},
{'word': 'on', 'start_time': 110.6, 'duration': 0.2},
{'word': 'two', 'start_time': 110.88, 'duration': 0.88},
{'word': 'you', 'start_time': 111.84, 'duration': 0.16},
{'word': 'do', 'start_time': 112.02, 'duration': 0.16},
{'word': 'one', 'start_time': 112.28, 'duration': 0.86},
{'word': 'would', 'start_time': 113.3, 'duration': 0.14},
{'word': 'be', 'start_time': 113.48, 'duration': 0.4},
{'word': 'repulsive', 'start_time': 113.98, 'duration': 0.96},
{'word': 'if', 'start_time': 115.04, 'duration': 0.12},
{'word': 'corantoes', 'start_time': 115.22, 'duration': 1.36},
{'word': 'sign', 'start_time': 116.64, 'duration': 0.54},
{'word': 'and', 'start_time': 117.28, 'duration': 0.18},
{'word': 'pointed', 'start_time': 117.52, 'duration': 0.34},
{'word': 'the', 'start_time': 117.88, 'duration': 0.08},
{'word': 'direction', 'start_time': 118.0, 'duration': 1.06},
{'word': 'joining', 'start_time': 119.12, 'duration': 0.36},
{'word': 'them', 'start_time': 119.52, 'duration': 0.82},
{'word': 'up', 'start_time': 120.5, 'duration': 3.58},
{'word': 'yes', 'start_time': 124.14, 'duration': 1.42},
{'word': 'thank', 'start_time': 125.62, 'duration': 0.2},
{'word': 'you', 'start_time': 125.86, 'duration': 1.6},
{'word': 'there', 'start_time': 127.5, 'duration': 0.1},
{'word': 'is', 'start_time': 127.64, 'duration': 0.04},
{'word': 'another', 'start_time': 127.72, 'duration': 0.28},
  {'word': 'for', 'start_time': 128.1, 'duration': 0.24},
  {'word': 'logic', 'start_time': 128.44, 'duration': 0.46},
  {'word': 'is', 'start_time': 128.96, 'duration': 0.1},
  {'word': 'archidamus', 'start_time': 129.16, 'duration': 1.3},
  {'word': 'call', 'start_time': 130.54, 'duration': 0.16},
  {'word': 'books', 'start_time': 130.72, 'duration': 0.24},
  {'word': 'loustalot', 'start_time': 131.02, 'duration': 1.3},
  {'word': 'right', 'start_time': 132.42, 'duration': 1.64},
  {'word': "that's", 'start_time': 134.1, 'duration': 0.14},
  {'word': 'the', 'start_time': 134.26, 'duration': 0.08},
  {'word': 'difference', 'start_time': 134.38, 'duration': 0.26},
  {'word': 'between', 'start_time': 134.7, 'duration': 0.44},
  {'word': 'being', 'start_time': 135.2, 'duration': 0.18},
  {'word': 'newton', 'start_time': 135.48, 'duration': 0.36},
  {'word': 'and', 'start_time': 135.92, 'duration': 0.1},
  {'word': 'being', 'start_time': 136.08, 'duration': 0.2},
  {'word': 'cook', 'start_time': 136.3, 'duration': 0.42},
  {'word': 'who', 'start_time': 136.78, 'duration': 0.14},
  {'word': 'is', 'start_time': 136.98, 'duration': 0.1},
  {'word': 'known', 'start_time': 137.1, 'duration': 0.16},
  {'word': 'for', 'start_time': 137.3, 'duration': 0.1},
  {'word': 'the', 'start_time': 137.42, 'duration': 0.08},
  {'word': 'arthquake', 'start_time': 137.6, 'duration': 0.48},
  {'word': 'low', 'start_time': 138.16, 'duration': 0.54},
  {'word': 'newton', 'start_time': 138.78, 'duration': 0.58},
  {'word': 'for', 'start_time': 139.4, 'duration': 0.12},
  {'word': 'the', 'start_time': 139.56, 'duration': 0.6},
  {'word': 'one', 'start_time': 140.26, 'duration': 0.08},
  {'word': 'over', 'start_time': 140.4, 'duration': 0.2},
  {'word': 'acquire', 'start_time': 140.66, 'duration': 0.38}]},
{'confidence': -2497.60546875,
 'words': [{'word': 'a', 'start_time': 0.44, 'duration': 0.06},
  {'word': 'new', 'start_time': 0.54, 'duration': 0.16},
  {'word': 'piece', 'start_time': 0.74, 'duration': 1.74},
  {'word': 'in', 'start_time': 2.64, 'duration': 0.06},
  {'word': 'the', 'start_time': 2.72, 'duration': 0.06},
  {'word': 'nuclear', 'start_time': 2.84, 'duration': 0.52},
  {'word': 'or', 'start_time': 3.48, 'duration': 0.18},
  {'word': 'some', 'start_time': 3.72, 'duration': 0.64},
  {'word': 'things', 'start_time': 4.4, 'duration': 0.2},
  {'word': 'called', 'start_time': 4.66, 'duration': 0.18},
  {'word': 'protons', 'start_time': 4.88, 'duration': 1.3},
  {'word': "something's", 'start_time': 6.3, 'duration': 0.38},
  {'word': 'called', 'start_time': 6.74, 'duration': 0.18},
  {'word': 'neutrons', 'start_time': 6.96, 'duration': 1.18},
  {'word': 'and', 'start_time': 8.28, 'duration': 0.52},
  {'word': 'outside', 'start_time': 8.94, 'duration': 0.38},
  {'word': 'or', 'start_time': 9.38, 'duration': 0.12},
  {'word': 'something', 'start_time': 9.56, 'duration': 0.32},
  {'word': 'to', 'start_time': 9.94, 'duration': 0.14},
{'word': 'electrons', 'start_time': 10.14, 'duration': 1.22},
{'word': "that's", 'start_time': 11.46, 'duration': 0.14},
{'word': 'all', 'start_time': 11.68, 'duration': 0.08},
{'word': 'the', 'start_time': 11.8, 'duration': 0.08},
{'word': 'atomic', 'start_time': 11.92, 'duration': 0.36},
{'word': 'structure', 'start_time': 12.3, 'duration': 0.88},
{'word': 'we', 'start_time': 13.26, 'duration': 0.12},
{'word': 'need', 'start_time': 13.44, 'duration': 3.12},
{'word': 'then', 'start_time': 16.66, 'duration': 0.4},
{'word': 'we', 'start_time': 17.18, 'duration': 0.22},
{'word': 'say', 'start_time': 17.5, 'duration': 1.6},
{'word': 'certain', 'start_time': 19.18, 'duration': 0.38},
{'word': 'entities', 'start_time': 19.66, 'duration': 1.06},
{'word': 'have', 'start_time': 20.8, 'duration': 0.14},
{'word': 'a', 'start_time': 21.02, 'duration': 0.04},
{'word': 'properly', 'start_time': 21.12, 'duration': 0.5},
{'word': 'called', 'start_time': 21.72, 'duration': 0.56},
{'word': 'electrons', 'start_time': 22.36, 'duration': 0.44},
{'word': 'charge', 'start_time': 22.86, 'duration': 1.94},
{'word': 'the', 'start_time': 24.88, 'duration': 0.1},
{'word': 'symbol', 'start_time': 25.02, 'duration': 0.26},
{'word': 'forelegs', 'start_time': 25.34, 'duration': 1.04},
{'word': 'is', 'start_time': 26.48, 'duration': 0.16},
{'word': 'cue', 'start_time': 26.76, 'duration': 2.42},
{'word': 'and', 'start_time': 29.28, 'duration': 0.08},
{'word': 'you', 'start_time': 29.4, 'duration': 0.08},
{'word': 'can', 'start_time': 29.52, 'duration': 0.08},
{'word': 'put', 'start_time': 29.64, 'duration': 0.12},
{'word': 'a', 'start_time': 29.82, 'duration': 0.06},
{'word': 'subscript', 'start_time': 29.92, 'duration': 0.48},
{'word': 'to', 'start_time': 30.44, 'duration': 0.08},
{'word': 'say', 'start_time': 30.58, 'duration': 0.12},
{'word': 'who', 'start_time': 30.76, 'duration': 0.12},
{'word': 'you', 'start_time': 30.92, 'duration': 0.06},
{'word': 'are', 'start_time': 31.02, 'duration': 0.08},
{'word': 'talking', 'start_time': 31.12, 'duration': 0.26},
{'word': 'about', 'start_time': 31.44, 'duration': 1.26},
{'word': 'so', 'start_time': 32.78, 'duration': 0.08},
{'word': 'you', 'start_time': 32.88, 'duration': 0.12},
{'word': 'can', 'start_time': 33.04, 'duration': 0.2},
{'word': 'say', 'start_time': 33.32, 'duration': 0.8},
{'word': 'upon', 'start_time': 34.28, 'duration': 0.62},
{'word': 'drawn', 'start_time': 34.94, 'duration': 1.26},
{'word': 'is', 'start_time': 36.28, 'duration': 0.14},
{'word': 'there', 'start_time': 36.48, 'duration': 2.02},
{'word': 'a', 'start_time': 38.6, 'duration': 0.08},
{'word': 'few', 'start_time': 38.72, 'duration': 0.16},
{'word': 'for', 'start_time': 38.92, 'duration': 0.1},
{'word': 'the', 'start_time': 39.06, 'duration': 0.06},
{'word': 'electron', 'start_time': 39.16, 'duration': 1.6},
{'word': 'is', 'start_time': 40.88, 'duration': 0.14},
{'word': 'minus', 'start_time': 41.08, 'duration': 0.38},
{'word': 'one', 'start_time': 41.58, 'duration': 0.14},
{'word': 'point', 'start_time': 41.78, 'duration': 0.32},
{'word': 'six', 'start_time': 42.16, 'duration': 0.24},
{'word': 'times', 'start_time': 42.48, 'duration': 1.06},
{'word': 'then', 'start_time': 43.62, 'duration': 0.2},
{'word': 'the', 'start_time': 43.86, 'duration': 0.08},
{'word': 'mines', 'start_time': 43.98, 'duration': 0.32},
{'word': 'nineteen', 'start_time': 44.34, 'duration': 0.94},
{'word': 'and', 'start_time': 45.36, 'duration': 0.08},
{'word': 'his', 'start_time': 45.48, 'duration': 0.14},
{'word': 'measured', 'start_time': 45.68, 'duration': 0.26},
{'word': 'in', 'start_time': 46.02, 'duration': 0.12},
{'word': 'colors', 'start_time': 46.18, 'duration': 3.04},
{'word': 'the', 'start_time': 49.28, 'duration': 0.08},
{'word': 'foorth', 'start_time': 49.44, 'duration': 0.54},
{'word': 'proton', 'start_time': 50.06, 'duration': 2.74},
{'word': 'is', 'start_time': 52.92, 'duration': 0.1},
{'word': 'really', 'start_time': 53.12, 'duration': 1.42},
{'word': 'helepolis', 'start_time': 54.6, 'duration': 3.7},
{'word': 'efore', 'start_time': 58.42, 'duration': 0.3},
{'word': 'protospathaire', 'start_time': 58.76, 'duration': 0.9},
{'word': 'numbers', 'start_time': 59.74, 'duration': 0.38},
{'word': 'so', 'start_time': 60.16, 'duration': 0.1},
{'word': 'sealyhams', 'start_time': 60.38, 'duration': 6.74},
{'word': 'now', 'start_time': 67.2, 'duration': 0.12},
{'word': 'the', 'start_time': 67.4, 'duration': 0.08},
{'word': 'importance', 'start_time': 67.52, 'duration': 0.32},
{'word': 'of', 'start_time': 67.9, 'duration': 0.06},
{'word': 'the', 'start_time': 67.98, 'duration': 0.08},
{'word': 'colon', 'start_time': 68.12, 'duration': 1.74},
{'word': 'is', 'start_time': 69.94, 'duration': 0.1},
{'word': 'that', 'start_time': 70.08, 'duration': 0.12},
{'word': 'if', 'start_time': 70.26, 'duration': 0.1},
{'word': 'anything', 'start_time': 70.44, 'duration': 0.4},
{'word': 'has', 'start_time': 70.94, 'duration': 0.24},
{'word': 'some', 'start_time': 71.26, 'duration': 0.14},
{'word': 'cool', 'start_time': 71.46, 'duration': 0.16},
{'word': 'amonition', 'start_time': 71.68, 'duration': 1.96},
{'word': 'interact', 'start_time': 73.7, 'duration': 0.38},
{'word': 'with', 'start_time': 74.12, 'duration': 0.14},
{'word': 'anything', 'start_time': 74.32, 'duration': 0.28},
{'word': 'else', 'start_time': 74.74, 'duration': 0.3},
{'word': 'that', 'start_time': 75.1, 'duration': 0.16},
{'word': 'has', 'start_time': 75.32, 'duration': 0.12},
{'word': 'some', 'start_time': 75.48, 'duration': 0.18},
{'word': 'columns', 'start_time': 75.72, 'duration': 2.36},
{'word': 'that', 'start_time': 78.1, 'duration': 0.12},
{'word': 'if', 'start_time': 78.3, 'duration': 0.04},
{'word': 'you', 'start_time': 78.38, 'duration': 0.18},
{'word': 'have', 'start_time': 78.66, 'duration': 1.06},
{'word': 'to', 'start_time': 79.78, 'duration': 0.1},
{'word': 'entertain', 'start_time': 79.98, 'duration': 3.1},
{'word': 'one', 'start_time': 83.18, 'duration': 0.1},
{'word': 'has', 'start_time': 83.34, 'duration': 0.16},
{'word': 'the', 'start_time': 83.56, 'duration': 0.12},
{'word': 'charge', 'start_time': 83.74, 'duration': 0.32},
{'word': 'of', 'start_time': 84.14, 'duration': 0.1},
{'word': 'concreteness', 'start_time': 84.36, 'duration': 2.48},
{'word': 'charge', 'start_time': 86.9, 'duration': 0.2},
{'word': 'a', 'start_time': 87.18, 'duration': 0.06},
{'word': 'butaca', 'start_time': 87.32, 'duration': 1.92},
{'word': 'and', 'start_time': 89.32, 'duration': 0.06},
{'word': 'the', 'start_time': 89.42, 'duration': 0.1},
{'word': 'distance', 'start_time': 89.58, 'duration': 0.32},
{'word': 'between', 'start_time': 89.94, 'duration': 0.26},
{'word': 'them', 'start_time': 90.24, 'duration': 0.12},
{'word': 'his', 'start_time': 90.4, 'duration': 0.22},
{'word': 'arm', 'start_time': 90.76, 'duration': 1.98},
{'word': 'then', 'start_time': 92.78, 'duration': 0.16},
{'word': 'the', 'start_time': 93.0, 'duration': 0.12},
{'word': 'force', 'start_time': 93.2, 'duration': 1.54},
{'word': 'is', 'start_time': 94.84, 'duration': 0.2},
{'word': 'come', 'start_time': 95.14, 'duration': 0.22},
{'word': 'one', 'start_time': 95.46, 'duration': 0.8},
{'word': 'utopia', 'start_time': 96.42, 'duration': 3.04},
{'word': 'alonzo', 'start_time': 99.5, 'duration': 1.68},
{'word': 'or', 'start_time': 101.3, 'duration': 0.2},
{'word': 'square', 'start_time': 101.52, 'duration': 0.36},
{'word': 'and', 'start_time': 102.02, 'duration': 0.12},
{'word': 'purposely', 'start_time': 102.18, 'duration': 0.36},
{'word': 'not', 'start_time': 102.62, 'duration': 0.12},
{'word': 'putting', 'start_time': 102.82, 'duration': 0.92},
{'word': 'all', 'start_time': 103.86, 'duration': 0.1},
{'word': 'the', 'start_time': 103.98, 'duration': 0.06},
{'word': 'victories', 'start_time': 104.1, 'duration': 0.64},
{'word': 'unhasting', 'start_time': 104.82, 'duration': 1.8},
{'word': 'but', 'start_time': 106.64, 'duration': 0.1},
{'word': 'you', 'start_time': 106.78, 'duration': 0.1},
{'word': 'all', 'start_time': 106.94, 'duration': 0.1},
{'word': 'know', 'start_time': 107.08, 'duration': 0.1},
{'word': 'what', 'start_time': 107.24, 'duration': 0.1},
{'word': 'the', 'start_time': 107.38, 'duration': 0.1},
{'word': 'answer', 'start_time': 107.54, 'duration': 0.3},
{'word': 'is', 'start_time': 107.9, 'duration': 0.74},
{'word': 'namely', 'start_time': 108.72, 'duration': 0.88},
{'word': 'if', 'start_time': 109.66, 'duration': 0.04},
{'word': 'you', 'start_time': 109.74, 'duration': 0.08},
{'word': 'want', 'start_time': 109.86, 'duration': 0.16},
{'word': 'the', 'start_time': 110.06, 'duration': 0.1},
{'word': 'force', 'start_time': 110.22, 'duration': 0.26},
{'word': 'on', 'start_time': 110.6, 'duration': 0.2},
{'word': 'two', 'start_time': 110.88, 'duration': 0.88},
{'word': 'you', 'start_time': 111.84, 'duration': 0.16},
{'word': 'do', 'start_time': 112.02, 'duration': 0.16},
{'word': 'one', 'start_time': 112.28, 'duration': 0.86},
{'word': 'would', 'start_time': 113.3, 'duration': 0.14},
{'word': 'be', 'start_time': 113.48, 'duration': 0.4},
{'word': 'repulsive', 'start_time': 113.98, 'duration': 0.96},
{'word': 'if', 'start_time': 115.04, 'duration': 0.12},
{'word': 'corantoes', 'start_time': 115.22, 'duration': 1.36},
{'word': 'sign', 'start_time': 116.64, 'duration': 0.54},
{'word': 'and', 'start_time': 117.28, 'duration': 0.18},
{'word': 'pointed', 'start_time': 117.52, 'duration': 0.34},
{'word': 'the', 'start_time': 117.88, 'duration': 0.08},
{'word': 'direction', 'start_time': 118.0, 'duration': 1.06},
{'word': 'joining', 'start_time': 119.12, 'duration': 0.36},
{'word': 'them', 'start_time': 119.52, 'duration': 0.82},
{'word': 'up', 'start_time': 120.5, 'duration': 3.58},
{'word': 'yes', 'start_time': 124.14, 'duration': 1.42},
{'word': 'thank', 'start_time': 125.62, 'duration': 0.2},
{'word': 'you', 'start_time': 125.86, 'duration': 1.6},
{'word': 'there', 'start_time': 127.5, 'duration': 0.1},
{'word': 'is', 'start_time': 127.64, 'duration': 0.04},
{'word': 'another', 'start_time': 127.72, 'duration': 0.28},
{'word': 'for', 'start_time': 128.1, 'duration': 0.24},
{'word': 'logic', 'start_time': 128.44, 'duration': 0.46},
{'word': 'is', 'start_time': 128.96, 'duration': 0.1},
{'word': 'archidamus', 'start_time': 129.16, 'duration': 1.3},
{'word': 'call', 'start_time': 130.54, 'duration': 0.16},
{'word': 'books', 'start_time': 130.72, 'duration': 0.24},
{'word': 'loustalot', 'start_time': 131.02, 'duration': 1.3},
{'word': 'right', 'start_time': 132.42, 'duration': 1.64},
{'word': "that's", 'start_time': 134.1, 'duration': 0.14},
{'word': 'the', 'start_time': 134.26, 'duration': 0.08},
{'word': 'difference', 'start_time': 134.38, 'duration': 0.26},
{'word': 'between', 'start_time': 134.7, 'duration': 0.44},
{'word': 'being', 'start_time': 135.2, 'duration': 0.18},
{'word': 'newton', 'start_time': 135.48, 'duration': 0.36},
{'word': 'and', 'start_time': 135.92, 'duration': 0.1},
{'word': 'being', 'start_time': 136.08, 'duration': 0.2},
{'word': 'cook', 'start_time': 136.3, 'duration': 0.42},
{'word': 'who', 'start_time': 136.78, 'duration': 0.14},
{'word': 'is', 'start_time': 136.98, 'duration': 0.1},
{'word': 'known', 'start_time': 137.1, 'duration': 0.16},
{'word': 'for', 'start_time': 137.3, 'duration': 0.1},
{'word': 'the', 'start_time': 137.42, 'duration': 0.08},
{'word': 'arthquake', 'start_time': 137.6, 'duration': 0.48},
{'word': 'low', 'start_time': 138.16, 'duration': 0.54},
    {'word': 'lukannon', 'start_time': 138.78, 'duration': 0.58},
    {'word': 'for', 'start_time': 139.4, 'duration': 0.12},
    {'word': 'the', 'start_time': 139.56, 'duration': 0.6},
    {'word': 'one', 'start_time': 140.26, 'duration': 0.08},
    {'word': 'over', 'start_time': 140.4, 'duration': 0.2},
    {'word': 'acquired', 'start_time': 140.66, 'duration': 0.52}]}]}
```

```
text = []
for dictionary in data['transcripts'][0].get('words'):
    text.append(dictionary['word'])


' '.join(word for word in text)
```

```
"a new piece in the nuclear or some things called protons something's
called neutrons and outside or something to electrons that's all the
atomic structure we need then we say certain entities have a properly
called electrons charge the symbol forelegs is cue and you can put a
subscript to say who you are talking about so you can say upon drawn
is there a few for the electron is minus one point six times then the
mines nineteen and his measured in colors the foorth proton is really
helepolis efore protospathaire numbers so sealyhams now the importance
of the colon is that if anything has some cool amonition interact with
anything else that has some columns that if you have to entertain one
has the charge of concreteness charge a butaca and the distance
between them his arm then the force is come one utopia alonzo or
square and purposely not putting all the victories unhasting but you
all know what the answer is namely if you want the force on two you do
one would be repulsive if corantoes sign and pointed the direction
joining them up yes thank you there is another for logic is archidamus
call books loustalot right that's the difference between being newton
and being cook who is known for the arthquake low lukannon for the one
over acquire"
```

Observations

- Conversion is of moderate level, way better than vosk model but lesser than speechrecognition model.
- Timestamps along with word is given which is required for our case so we will go ahead with this technique.

Converting all videos to text

```
%%time
import subprocess
for i in range(1,10):
    subprocess.call("deepspeech --model deepspeech-0.9.3-models.pbmm
--scorer deepspeech-0.9.3-models.scorer --audio ./{}.wav --json >
{}.json".format(str(i), str(i)),shell=True )
```

```
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
```

```
2022-01-29 05:42:47.448074: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0144s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000265s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 100.870s for 143.800s audio file.
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:44:28.950974: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0143s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000311s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 84.450s for 123.800s audio file.
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:45:54.038292: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0144s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000277s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 82.286s for 117.520s audio file.
TensorFlow: v2.3.0-6-g23ad988
```

```
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:47:16.913050: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0145s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000278s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 101.484s for 146.240s audio file.
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:48:59.083447: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0144s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000286s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 84.886s for 124.600s audio file.
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:50:24.617582: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0143s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000273s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 84.815s for 117.400s audio file.
```

```
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:51:50.020333: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0148s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000321s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 79.946s for 114.320s audio file.
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:53:10.550248: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.
Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0146s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000263s.
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 86.128s for 126.120s audio file.
TensorFlow: v2.3.0-6-g23ad988
DeepSpeech: v0.9.3-0-gf2e9c85
2022-01-29 05:54:37.280195: I
tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow
binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to
use the following CPU instructions in performance-critical operations:
AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the
appropriate compiler flags.

CPU times: user 94.1 ms, sys: 79.9 ms, total: 174 ms
Wall time: 13min 15s

Loading model from file deepspeech-0.9.3-models.pbmm
Loaded model in 0.0141s.
Loading scorer from files deepspeech-0.9.3-models.scorer
Loaded scorer in 0.000261s.
```

```
Warning: original sample rate (44100) is different than 16000hz.
Resampling might produce erratic speech recognition.
Running inference.
Inference took 84.905s for 123.760s audio file.
```

Creating text features from json files

- Extract features like word, start_time and duration from json file and create df for each videoid.
- Append each df and create one single df containing all video features.

```
%%time
import pandas as pd
import os
from tqdm import tqdm

video_features = pd.DataFrame()
text_path = r'C:\Users\Palak\APPLIED_AI\APPLIED_AI\Assignments\SELF-
CASE-STUDY2\EDA\archive\confused_eeg\text'
for i in tqdm(range(0,10)):
    # Opening JSON file
    file = open(os.path.join(text_path,str(i)+'.json'))
    # returns JSON object as
    # a dictionary
    data = json.load(file)

    # Create dataframe containing words, start_time and duration
    word_df = pd.DataFrame(data['transcripts'][0].get('words'))
    word_df['VideoID'] = i
    print(i,word_df.shape)
    video_features = video_features.append(word_df)
```

```
 30%|█████████████████████████
| 3/10 [00:00<00:00, 10.62it/s]

0 (225, 4)
1 (255, 4)
2 (190, 4)
3 (243, 4)

 70%|███████████████████████████████████████████████████████████
| 7/10 [00:00<00:00, 12.46it/s]

4 (251, 4)
5 (197, 4)
6 (302, 4)
7

100%|
███████████████████████████████████████████████████████████████████████
████████████| 10/10 [00:00<00:00, 16.25it/s]
```

```
 (256, 4)
8 (201, 4)
9 (209, 4)
Wall time: 9.54 s
```

```
video_features.shape
```

```
(2329, 4)
```

```
video_features.head()
```

```
    word  start_time  duration  VideoID
0      a        0.44      0.06        0
1    new        0.54      0.16        0
2  piece        0.74      1.74        0
3     in        2.64      0.06        0
4    the        2.72      0.06        0
```

```
df.groupby(['SubjectID','VideoID']).size()
```

```
SubjectID  VideoID
0.0        0.0        144
           1.0        140
           2.0        142
           3.0        122
           4.0        116
                     ...
9.0        5.0        123
           6.0        116
           7.0        112
           8.0        124
           9.0        122
Length: 100, dtype: int64
```

As we know data we have is sampled at 0.5s.

- Create new column - timestamp - at an interval of 0.5s starting with value 30 as we have tabular data which we have is data corresponding to video where first 30s and last 30s are chopped.
- We will chop first 30s, so starting timestamp will be starting from 30.5s, 40, 40.5 etc.
- Won't bother for last 30s as when we will merge text features and original df, it won't get merged.

```
df['timestamp'] = 0.5

def make_timestamps(group_obj):
    '''For each student watching each video,
       timeseries will start from 0.5s with
       an interval of 0.5s
       Input - group object
       Output - group object(df corresponding to student and videoid)
```

```
with synthetic timestamp
    '''

    group_obj['timestamp'].iloc[0] = group_obj['timestamp'].iloc[0] +
30
    group_obj['timestamp'] = group_obj['timestamp'].cumsum()
    return group_obj

df.groupby(['SubjectID',
'VideoID']).apply(make_timestamps).reset_index(drop=True)
```

```
       SubjectID  VideoID  Attention  Mediation    Raw      Delta
Theta  \
0            0.0      0.0       56.0       43.0  278.0   301963.0
90612.0
1            0.0      0.0       40.0       35.0  -50.0    73787.0
28083.0
2            0.0      0.0       47.0       48.0  101.0   758353.0
383745.0
3            0.0      0.0       47.0       57.0   -5.0  2012240.0
129350.0
4            0.0      0.0       44.0       53.0   -8.0  1005145.0
354328.0
...          ...      ...        ...        ...    ...        ...
...
12806        9.0      9.0       64.0       38.0  -39.0   127574.0
9951.0
12807        9.0      9.0       61.0       35.0 -275.0   323061.0
797464.0
12808        9.0      9.0       60.0       29.0 -426.0   680989.0
154296.0
12809        9.0      9.0       60.0       29.0  -84.0   366269.0
27346.0
12810        9.0      9.0       64.0       29.0  -49.0  1164555.0
1184366.0

         Alpha1    Alpha2    Beta1  ...   Gamma2  predefinedlabel  \
0       33735.0   23991.0  27946.0  ...   8293.0              0.0
1        1439.0    2240.0   2746.0  ...   2740.0              0.0
2      201999.0   62107.0  36293.0  ...  25354.0              0.0
3       61236.0   17084.0  11488.0  ...  33932.0              0.0
4       37102.0   88881.0  45307.0  ...  29749.0              0.0
...         ...       ...      ...  ...      ...              ...
12806     709.0   21732.0   3872.0  ...    960.0              1.0
12807  153171.0  145805.0  39829.0  ...  10010.0              1.0
12808   40068.0   39122.0  10966.0  ...   2024.0              1.0
12809   11444.0    9932.0   1939.0  ...   1764.0              1.0
12810   50014.0  124208.0  10634.0  ...   4482.0              1.0

       user-definedlabeln  age  ethnicity_Bengali
```

```
ethnicity_English  \
0                     0.0   25                  0                  0

1                     0.0   25                  0                  0

2                     0.0   25                  0                  0

3                     0.0   25                  0                  0

4                     0.0   25                  0                  0

...                   ...  ...                ...                ...

12806                 0.0   24                  0                  0

12807                 0.0   24                  0                  0

12808                 0.0   24                  0                  0

12809                 0.0   24                  0                  0

12810                 0.0   24                  0                  0


       ethnicity_Han Chinese  gender_F  gender_M  timestamp
0                          1         0         1       30.5
1                          1         0         1       31.0
2                          1         0         1       31.5
3                          1         0         1       32.0
4                          1         0         1       32.5
...                      ...       ...       ...        ...
12806                      1         1         0       89.0
12807                      1         1         0       89.5
12808                      1         1         0       90.0
12809                      1         1         0       90.5
12810                      1         1         0       91.0

[12811 rows x 22 columns]
```

```
df = df.groupby(['SubjectID',
'VideoID']).apply(make_timestamps).reset_index(drop=True)
```

For how much duration a word is spoken

```
video_features['duration'].describe()
```

```
count    2329.000000
mean        0.471198
std         1.193790
min         0.020000
```

```
25%          0.100000
50%          0.200000
75%          0.460000
max         38.120000
Name: duration, dtype: float64
```

```
import numpy as np
for p in range(76,103,3):
    print("Value at {}th percentile:
{}".format(p,np.percentile(video_features['duration'],p)))
```

```
Value at 76th percentile: 0.48
Value at 79th percentile: 0.56
Value at 82th percentile: 0.64
Value at 85th percentile: 0.76
Value at 88th percentile: 0.8999999999999999
Value at 91th percentile: 1.08
Value at 94th percentile: 1.42
Value at 97th percentile: 2.08
Value at 100th percentile: 38.12
```

```
video_features[video_features['duration']>0.46].shape
```

```
(580, 4)
```

We need to merge video extracted text features with our original data.

- EEG data contains timestamp sampled at 0.5s.
- video features have start_time at which word spoken was started and duration till which word was spoken.

How to do that?

- Goal - To assign each word to 0.5 timestamp window.

```
def assign_word_to_timestamp(grp):
    ts = {}
    for index,row in grp.iterrows():
        # To find the bucket id/ timestamp
        # 0.44/0.5 = 0 +1 = 1 *0.5= 0.5 = first bucket or 0.5
timestamp row
        result_start = (int(row['start_time']/0.5) + 1 )*0.5

        # overlap case
        # if there are 2 or more words in a single bucket then add
them
        if result_start in ts:
            ts[result_start]=ts[result_start]+ ' '+row['word']
        else:
            ts[result_start] = row['word']
        # To find the bucketid when the word ends
        # why divide by 0.501? not 0.5 --> If word ends at 0.5s then
it will also be assigned to next bucket.
        result_end = (int( (row['duration']+row['start_time'])/0.501 )
+ 1 )*0.5

        # result_start+0.5 - we have already assigned word for
result_start
        # result_end+0.5 - we have to iterate till result_end
        # if duration is > 0.5s then repeat the words for next
applicable buckets
        for i in np.arange(result_start+0.5,result_end+0.5,0.5):

            if i in ts:
                ts[i]=ts[i]+ ' '+row['word']
            else:
                ts[i] = row['word']


    new_df = pd.DataFrame(ts.items(), columns=['timestamp','word'])

    return new_df


video_df =
video_features.groupby('VideoID').apply(assign_word_to_timestamp).rese
t_index(level=[0])
video_df
```

```
     VideoID  timestamp        word
0          0        0.5           a
1          0        1.0   new piece
2          0        1.5       piece
3          0        2.0       piece
4          0        2.5       piece
..       ...        ...         ...
242        9      122.0      of the
243        9      122.5  embodiment
244        9      123.0  embodiment
245        9      123.5  embodiment
246        9      124.0         met

[2532 rows x 3 columns]
```

Merge EEG data with video features

```
df.merge(video_df, on=['VideoID','timestamp'])
```

```
       SubjectID  VideoID  Attention  Mediation    Raw      Delta
Theta  \
0            0.0      0.0       56.0       43.0  278.0   301963.0
90612.0
1            1.0      0.0       47.0       44.0   35.0   216055.0
132532.0
2            2.0      0.0       41.0       44.0   12.0    53205.0
57329.0
3            3.0      0.0        7.0       54.0  358.0  1307676.0
64069.0
4            4.0      0.0       43.0       38.0   68.0   395978.0
70226.0
...          ...      ...        ...        ...    ...        ...
...
12717        3.0      3.0        0.0        0.0   72.0  1712005.0
361273.0
12718        3.0      7.0       51.0       50.0  -53.0  1019187.0
140418.0
12719        7.0      7.0       50.0       63.0   40.0    10918.0
20480.0
12720        3.0      7.0       43.0       47.0  -26.0   793650.0
784498.0
12721        7.0      7.0       48.0       63.0   64.0     4835.0
22936.0

        Alpha1    Alpha2    Beta1  ...  predefinedlabel  user-
definedlabeln  \
0      33735.0   23991.0  27946.0  ...              0.0
0.0
1      11941.0   14898.0  25188.0  ...              0.0
0.0
2      12391.0   27427.0  16770.0  ...              0.0
0.0
3      95902.0   12350.0   5634.0  ...              0.0
0.0
4      18716.0   10762.0  16668.0  ...              0.0
0.0
...        ...       ...      ...  ...              ...
...
12717  27700.0  117317.0  22478.0  ...              0.0
0.0
12718  79729.0   29890.0  27966.0  ...              1.0
1.0
12719   7805.0   40999.0  11441.0  ...              1.0
1.0
12720  38404.0    6301.0  14144.0  ...              1.0
1.0
12721  15695.0    8730.0  10796.0  ...              1.0
1.0

       age  ethnicity_Bengali  ethnicity_English  ethnicity_Han
Chinese  \
0       25                  0                 0
1
1       24                  0                 0
1
2       31                  0                 1
0
3       28                  0                 0
1
4       24                  1                 0
0
...    ...                ...                ...                    ..
.
12717   28                  0                 0
1
12718   28                  0                 0
1
12719   25                  0                 0
1
12720   28                  0                 0
1
12721   25                  0                 0
1

       gender_F  gender_M  timestamp          word
0             0         1       30.5  subscript to
1             0         1       30.5  subscript to
2             0         1       30.5  subscript to
3             1         0       30.5  subscript to
4             0         1       30.5  subscript to
...         ...       ...        ...           ...
12717         1         0      102.0    you'll see
12718         1         0       91.5     effect of
12719         0         1       91.5     effect of
12720         1         0       92.0         water
12721         0         1       92.0         water

[12722 rows x 23 columns]
```

```
df.shape
```

```
(12811, 22)
```

```
data = df.merge(video_df,
on=['VideoID','timestamp']).sort_values(by=['SubjectID','VideoID','tim
estamp']).reset_index(drop=True)
```

```
data
```

```
       SubjectID  VideoID  Attention  Mediation    Raw      Delta
Theta  \
0            0.0      0.0       56.0       43.0  278.0   301963.0
```

```
90612.0
1            0.0      0.0       40.0       35.0  -50.0    73787.0
28083.0
2            0.0      0.0       47.0       48.0  101.0   758353.0
383745.0
3            0.0      0.0       47.0       57.0   -5.0  2012240.0
129350.0
4            0.0      0.0       44.0       53.0   -8.0  1005145.0
354328.0
...           ...      ...        ...        ...    ...        ...
...
12717        9.0      9.0       64.0       38.0  -39.0   127574.0
9951.0
12718        9.0      9.0       61.0       35.0 -275.0   323061.0
797464.0
12719        9.0      9.0       60.0       29.0 -426.0   680989.0
154296.0
12720        9.0      9.0       60.0       29.0  -84.0   366269.0
27346.0
12721        9.0      9.0       64.0       29.0  -49.0  1164555.0
1184366.0

         Alpha1    Alpha2    Beta1  ...  predefinedlabel  user-
definedlabeln  \
0       33735.0   23991.0  27946.0  ...              0.0
0.0
1        1439.0    2240.0   2746.0  ...              0.0
0.0
2      201999.0   62107.0  36293.0  ...              0.0
0.0
3       61236.0   17084.0  11488.0  ...              0.0
0.0
4       37102.0   88881.0  45307.0  ...              0.0
0.0
...         ...       ...      ...  ...              ...
...
12717     709.0   21732.0   3872.0  ...              1.0
0.0
12718  153171.0  145805.0  39829.0  ...              1.0
0.0
12719   40068.0   39122.0  10966.0  ...              1.0
0.0
12720   11444.0    9932.0   1939.0  ...              1.0
0.0
12721   50014.0  124208.0  10634.0  ...              1.0
0.0

       age  ethnicity_Bengali  ethnicity_English  ethnicity_Han
Chinese  \
0       25                  0                  0
```

```
1
1       25                  0                  0
1
2       25                  0                  0
1
3       25                  0                  0
1
4       25                  0                  0
1
...    ...                ...                ...                    ..
.
12717   24                  0                  0
1
12718   24                  0                  0
1
12719   24                  0                  0
1
12720   24                  0                  0
1
12721   24                  0                  0
1

       gender_F  gender_M  timestamp               word
0             0         1       30.5       subscript to
1             0         1       31.0        say who you
2             0         1       31.5  are talking about
3             0         1       32.0              about
4             0         1       32.5              about
...         ...       ...        ...                ...
12717         1         0       89.0             losing
12718         1         0       89.5             factor
12719         1         0       90.0         factor but
12720         1         0       90.5               then
12721         1         0       91.0               then

[12722 rows x 23 columns]
```

Create Text Features - Apply countvectorizer to word column

```
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer(stop_words='english')
vectorizer.fit(data['word'].values)
```

```
CountVectorizer(stop_words='english')
```

```
len(vectorizer.get_feature_names())
```

```
336
```

Vocabulary of 336 words is created

```
text_features = vectorizer.transform(data['word'].values)
```

```
text_features.shape
```

```
(12722, 336)
```

```
text_df = pd.DataFrame(text_features.todense(),
columns=vectorizer.get_feature_names())
text_df.head()
```

```
   absorbing  accolon  aces  acting  acts  actually  add  added
adeleine  \
0          0        0     0       0     0         0    0      0
0
1          0        0     0       0     0         0    0      0
0
2          0        0     0       0     0         0    0      0
0
3          0        0     0       0     0         0    0      0
0
4          0        0     0       0     0         0    0      0
0

   aetolian  ...  wills  words  worry  write  writer  wrote  years  ze
\
0         0  ...      0      0      0      0       0      0      0   0

1         0  ...      0      0      0      0       0      0      0   0

2         0  ...      0      0      0      0       0      0      0   0

3         0  ...      0      0      0      0       0      0      0   0

4         0  ...      0      0      0      0       0      0      0   0


   zenobias  zero
0         0     0
1         0     0
2         0     0
3         0     0
4         0     0

[5 rows x 336 columns]
```

```
final_df = pd.concat([data, text_df], axis=1)
final_df.head()
```

```
   SubjectID  VideoID  Attention  Mediation    Raw      Delta
Theta  \
0        0.0      0.0       56.0       43.0  278.0   301963.0
90612.0
```

```
1        0.0      0.0       40.0       35.0  -50.0    73787.0
28083.0
2        0.0      0.0       47.0       48.0  101.0   758353.0
383745.0
3        0.0      0.0       47.0       57.0   -5.0  2012240.0
129350.0
4        0.0      0.0       44.0       53.0   -8.0  1005145.0
354328.0

     Alpha1   Alpha2    Beta1  ...  wills  words  worry  write  writer
wrote  \
0   33735.0  23991.0  27946.0  ...      0      0      0      0       0
0
1    1439.0   2240.0   2746.0  ...      0      0      0      0       0
0
2  201999.0  62107.0  36293.0  ...      0      0      0      0       0
0
3   61236.0  17084.0  11488.0  ...      0      0      0      0       0
0
4   37102.0  88881.0  45307.0  ...      0      0      0      0       0
0

   years  ze  zenobias  zero
0      0   0         0     0
1      0   0         0     0
2      0   0         0     0
3      0   0         0     0
4      0   0         0     0

[5 rows x 359 columns]
```

```
final_df.columns, final_df.shape
```

```
(Index(['SubjectID', 'VideoID', 'Attention', 'Mediation', 'Raw',
'Delta',
        'Theta', 'Alpha1', 'Alpha2', 'Beta1',
        ...
        'wills', 'words', 'worry', 'write', 'writer', 'wrote',
'years', 'ze',
        'zenobias', 'zero'],
       dtype='object', length=359),
 (12722, 359))
```

Train-Test Split based on Student

- We will take 8 students data as training dataset, 2 students data as test set

```
train = final_df[final_df['SubjectID'].isin(list(range(0,8)))]
#validation = final_df[final_df['SubjectID'].isin(list(range(6,8)))]
test = final_df[final_df['SubjectID'].isin(list(range(8,10)))]

print(train.shape,test.shape)
```

```
(7664, 359) (2525, 359)
```

Dropping unnecessary columns:

```
###
X_train = train.drop(['SubjectID', 'VideoID','predefinedlabel', 'user-
definedlabeln', 'word', 'timestamp'], axis=1)
y_train = train['user-definedlabeln']

# X_val = validation.drop(['SubjectID', 'VideoID','predefinedlabel',
'user-definedlabeln','word', 'timestamp'], axis=1)
# y_val = validation['user-definedlabeln']

X_test = test.drop(['SubjectID', 'VideoID','predefinedlabel', 'user-
definedlabeln','word','timestamp'], axis=1)
y_test = test['user-definedlabeln']

X_train.columns, X_test.columns
```

```
(Index(['Attention', 'Mediation', 'Raw', 'Delta', 'Theta', 'Alpha1',
'Alpha2',
        'Beta1', 'Beta2', 'Gamma1',
        ...
        'wills', 'words', 'worry', 'write', 'writer', 'wrote',
'years', 'ze',
        'zenobias', 'zero'],
       dtype='object', length=353),
 Index(['Attention', 'Mediation', 'Raw', 'Delta', 'Theta', 'Alpha1',
'Alpha2',
        'Beta1', 'Beta2', 'Gamma1',
        ...
        'wills', 'words', 'worry', 'write', 'writer', 'wrote',
'years', 'ze',
        'zenobias', 'zero'],
       dtype='object', length=353))
```

Numerical Feature Scaling

```
cols_to_be_scaled = ['Attention', 'Mediation', 'Raw', 'Delta',
'Theta', 'Alpha1', 'Alpha2',
        'Beta1', 'Beta2', 'Gamma1', 'Gamma2', 'age']

sc = StandardScaler()
X_train[cols_to_be_scaled]=
sc.fit_transform(X_train[cols_to_be_scaled])
#X_val[cols_to_be_scaled] = sc.transform(X_val[cols_to_be_scaled])
X_test[cols_to_be_scaled] = sc.transform(X_test[cols_to_be_scaled])

X_train.columns.values
```

```
array(['Attention', 'Mediation', 'Raw', 'Delta', 'Theta', 'Alpha1',
       'Alpha2', 'Beta1', 'Beta2', 'Gamma1', 'Gamma2', 'age',
       'ethnicity_Bengali', 'ethnicity_English', 'ethnicity_Han
Chinese',
       'gender_F', 'gender_M', 'absorbing', 'accolon', 'aces',
'acting',
       'acts', 'actually', 'add', 'added', 'adeleine', 'aetolian',
       'agitated', 'agreement', 'alonzo', 'aluminum', 'amonition',
       'amorites', 'amplifier', 'analogies', 'annihilating', 'answer',
       'arbitrary', 'arithmetic', 'arm', 'art', 'article',
       'articulations', 'ask', 'asters', 'astronomers', 'atop',
       'attractive', 'axes', 'baubee', 'beautifully', 'begin',
'better',
       'bit', 'bless', 'boeotia', 'bucolic', 'bunch', 'butaca',
       'caerleon', 'cake', 'calculate', 'called', 'cambrian', 'cards',
       'case', 'casion', 'casual', 'cavalcanti', 'center', 'charge',
       'cheer', 'chop', 'class', 'clastidium', 'clean', 'clitus',
'colon',
       'colors', 'columbus', 'columns', 'come', 'common', 'commonly',
       'conceits', 'concreteness', 'constitutes', 'contrite', 'cool',
       'correct', 'correctly', 'corresponding', 'created', 'crossed',
       'curtain', 'day', 'deceiver', 'define', 'deloitte',
'denominator',
       'detective', 'diabolka', 'diathetic', 'did', 'digits',
       'discontinuity', 'disinterested', 'distance', 'doesn', 'doing',
       'don', 'draw', 'drawn', 'drolette', 'dubitating', 'dutton',
       'easier', 'effect', 'efore', 'electron', 'elements', 'ellen',
       'eloigne', 'energy', 'entertain', 'equal', 'equals',
'equation',
       'equivalent', 'estate', 'esting', 'exact', 'excess', 'expanse',
       'explicate', 'exploits', 'express', 'expressive', 'factor',
       'fails', 'fancier', 'fibres', 'focused', 'foorth', 'force',
       'formative', 'function', 'galaxy', 'gave', 'geometric',
       'gillikins', 'given', 'going', 'good', 'got', 'graft',
'graphic',
       'guess', 'guesser', 'having', 'hear', 'hearsay', 'helepolis',
       'hereabout', 'heroes', 'high', 'ibrahim', 'impacted',
'importance',
       'interact', 'interested', 'interpreted', 'intuitive', 'isunt',
       'katalla', 'kissaphone', 'know', 'learn', 'leontopolis', 'let',
       'lies', 'life', 'like', 'lilium', 'line', 'little', 'll',
'lock',
       'londinensis', 'londonomania', 'long', 'look', 'looked',
'looks',
       'losing', 'lot', 'lowest', 'make', 'manipulation', 'map',
'maps',
       'margin', 'matter', 'mean', 'meantime', 'measured',
'melodious',
       'million', 'minaret', 'mines', 'minus', 'minute', 'missus',
       'mister', 'monsignore', 'monterey', 'mould', 'nappie', 'near',
       'necessarily', 'negative', 'nineteen', 'noise', 'novas',
'number',
       'numbers', 'numerator', 'observed', 'occasion', 'oconee',
       'oftentimes', 'oh', 'okay', 'opiate', 'order', 'orders',
       'ostentatious', 'parable', 'parabola', 'patient', 'penelope',
       'pinpoint', 'plus', 'poetry', 'point', 'policeman', 'pontifex',
       'positrons', 'posthelwaite', 'postponement', 'preposition',
       'problem', 'problems', 'prophesying', 'proton',
'protospathaire',
       'pushed', 'pussons', 'putrescent', 'radioactive', 'real',
'really',
       'reason', 'receive', 'refer', 'reforestation', 'remind',
       'represent', 'resisted', 'reverse', 'revolutionaries', 'right',
       'say', 'says', 'sealyhams', 'second', 'segmentation', 'self',
       'series', 'set', 'shared', 'sicilianische', 'sides', 'sigma',
       'signal', 'signor', 'simply', 'single', 'situation', 'sky',
       'somebody', 'sooseeta', 'sources', 'square', 'squared',
'squares',
       'start', 'strength', 'sub', 'subscript', 'sum', 'sums', 'sun',
       'super', 'taksali', 'talking', 'tascherette', 'telemetering',
       'telephones', 'tender', 'terms', 'thing', 'think', 'thirty',
       'thousand', 'times', 'tly', 'tolerate', 'tottapotomoi',
'transmit',
       'unchristian', 'undefined', 'understand', 'use', 'used',
       'usivulele', 'usually', 'utopia', 'valediction', 'value',
'values',
       'variable', 'variant', 've', 'version', 'volts', 'wallace',
'want',
       'washout', 'watch', 'water', 'way', 'week', 'wills', 'words',
       'worry', 'write', 'writer', 'wrote', 'years', 'ze', 'zenobias',
       'zero'], dtype=object)
```

Base-line model

- For baseline modelling, ml models are experimented

1.1 Logistic Regression with FE features

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV

lr = LogisticRegression()
lr.fit(X_train,y_train)

pred_lr = lr.predict(X_test)
print("Test Accuracy: {:.5f}".format(accuracy_score(y_test, pred_lr)))
```

```
Test Accuracy: 0.52040

C:\Users\Palak\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
```

```
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
%%time
grid_values = {'penalty': ['l1','l2'], 'C':
[0.001,0.01,0.1,1,10,100,1000]}

lr = LogisticRegression(random_state=33, solver='liblinear')
model_lr = GridSearchCV(lr, param_grid=grid_values,cv=3)
model_lr.fit(X_train,y_train)
print("tuned hpyerparameters :(best parameters)
",model_lr.best_params_)
print("accuracy :",model_lr.best_score_)

pred_lr = model_lr.predict(X_test)
print("Test Accuracy: {:.5f}".format(accuracy_score(y_test, pred_lr)))
```

```
tuned hpyerparameters :(best parameters)  {'C': 1000, 'penalty': 'l2'}
accuracy : 0.5608045091004938
Test Accuracy: 0.51921
Wall time: 4.24 s
```

Observation:-

- Test Accuracy of 51% is achieved.

1.2 Logistic Regression without FE features

```
X_train2 = X_train[['Attention', 'Mediation', 'Raw', 'Delta', 'Theta',
'Alpha1',
       'Alpha2', 'Beta1', 'Beta2', 'Gamma1', 'Gamma2', 'age',
       'ethnicity_Bengali', 'ethnicity_English', 'ethnicity_Han
Chinese',
       'gender_F', 'gender_M']]

X_test2 = X_test[['Attention', 'Mediation', 'Raw', 'Delta', 'Theta',
'Alpha1',
       'Alpha2', 'Beta1', 'Beta2', 'Gamma1', 'Gamma2', 'age',
       'ethnicity_Bengali', 'ethnicity_English', 'ethnicity_Han
Chinese',
       'gender_F', 'gender_M']]
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV

lr = LogisticRegression()
lr.fit(X_train2,y_train)

pred_lr = lr.predict(X_test2)
print("Test Accuracy: {:.5f}".format(accuracy_score(y_test, pred_lr)))
```

```
Test Accuracy: 0.54297
```

```
%%time
grid_values = {'penalty': ['l1','l2'], 'C':
[0.001,0.01,0.1,1,10,100,1000]}

lr = LogisticRegression(random_state=33, solver='liblinear')
model_lr = GridSearchCV(lr, param_grid=grid_values,cv=3)
model_lr.fit(X_train2,y_train)
print("tuned hpyerparameters :(best parameters)
",model_lr.best_params_)
print("accuracy :",model_lr.best_score_)

pred_lr = model_lr.predict(X_test2)
print("Test Accuracy: {:.5f}".format(accuracy_score(y_test, pred_lr)))
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.001, 'penalty':
'l2'}
accuracy : 0.5555899166394656
Test Accuracy: 0.57188
Wall time: 1.7 s
```

Observation

- After hyperparameter tuning test accuracy of 57% achieved.
- Model is performing better without FE features.

2.1 Random Forest - with FE features

```
from sklearn.ensemble import RandomForestClassifier

rf_Model = RandomForestClassifier()
rf_Model.fit(X_train, y_train)

y_pred = rf_Model.predict(X_test)
print("Test accuracy: ", accuracy_score(y_test, y_pred))
```

```
Test accuracy:  0.5893069306930693
```

```
%%time
parameter_grid = {
               'min_samples_leaf': [3,4,5],
               'min_samples_split': [3,4,5],
               'n_estimators': [100, 300, 500, 700,1000],
                'bootstrap': [True, False]}

rf_model2 = RandomForestClassifier(n_jobs=-1, random_state=33)
rf_grid = RandomizedSearchCV(rf_model2, 
param_distributions=parameter_grid,n_jobs=-1,cv=3)
rf_grid.fit(X_train, y_train)
```

```
Wall time: 1min 26s
```

```
RandomizedSearchCV(cv=3,
                   estimator=RandomForestClassifier(n_jobs=-1, 
random_state=33),
                   n_jobs=-1,
                   param_distributions={'bootstrap': [True, False],
                                        'min_samples_leaf': [3, 4, 5],
                                        'min_samples_split': [3, 4, 
5],
                                        'n_estimators': [100, 300, 
500, 700,
                                                         1000]})
```

```
best_model = rf_grid.best_estimator_
print(rf_grid.best_params_,rf_grid.best_score_ )

y_test_predicted = best_model.predict(X_test)
accuracy_score(y_test_predicted, y_pred)
```

```
{'n_estimators': 700, 'min_samples_split': 3, 'min_samples_leaf': 3, 
'bootstrap': False} 0.5528481473365137
```

```
0.8712871287128713
```

Observation

- Validation accuracy of 55% and test accuracy of 87% (seems weird)

2.2 Random Forest - without FE features

```
from sklearn.ensemble import RandomForestClassifier

rf_Model = RandomForestClassifier()
rf_Model.fit(X_train2, y_train)

y_pred = rf_Model.predict(X_test2)
print("Test accuracy: ", accuracy_score(y_test, y_pred))
```

```
Test accuracy:  0.6011881188118812
```

```
%%time
parameter_grid = {
               'min_samples_leaf': [3,4,5],
               'min_samples_split': [3,4,5],
               'n_estimators': [100, 300, 500, 700,1000],
                'bootstrap': [True, False]}

rf_model2 = RandomForestClassifier(n_jobs=-1, random_state=33)
rf_grid = RandomizedSearchCV(rf_model2,
param_distributions=parameter_grid,n_jobs=-1,cv=3)
rf_grid.fit(X_train2, y_train)

Wall time: 1min 43s

RandomizedSearchCV(cv=3,
                   estimator=RandomForestClassifier(n_jobs=-1,
random_state=33),
                   n_jobs=-1,
                   param_distributions={'bootstrap': [True, False],
                                        'min_samples_leaf': [3, 4, 5],
                                        'min_samples_split': [3, 4,
5],
                                        'n_estimators': [100, 300,
500, 700,
                                                         1000]})

best_model = rf_grid.best_estimator_
print(rf_grid.best_params_,rf_grid.best_score_ )

y_test_predicted = best_model.predict(X_test2)
accuracy_score(y_test_predicted, y_pred)

{'n_estimators': 500, 'min_samples_split': 3, 'min_samples_leaf': 5,
'bootstrap': False} 0.5263597871112

0.9172277227722773
```

Observation

- Validation accuracy of 60% and test accuracy of 91% (seems weird)
- Model performance better without FE features

Final Observation

- For now text features don't provide any valuable information, without it model tend to perform better.
- But if speech to text conversion quality increases, text features might prove to be useful.
- Google Cloud offers speechtotext API which is a paid service but models might be highly accurate.