Signals and systems

Project

Audio search system using acoustic pattern

Ladislav Ondris (xondri07)

10. 12. 2019

1 Recordings

Recording	Sentence
sa1.wav	She had your dark suit in greasy wash water all year.
sa2.wav	Don't ask me to carry an oily rag like that.
si1446.wav	In earlier years, the preservation of food was essentially related to survival.
si2076.wav	You gonna give me a drink, fella?
si816.wav	Do this exercise six times each class period.
sx186.wav	Would a tomboy often play outdoors?
sx276.wav	John's brother repainted the garage door.
sx366.wav	Will you please confirm government policy regarding waste removal?
sx6.wav	Bright sunshine shimmers on the ocean.
sx96.wav	Masquerade parties tax one's imagination.

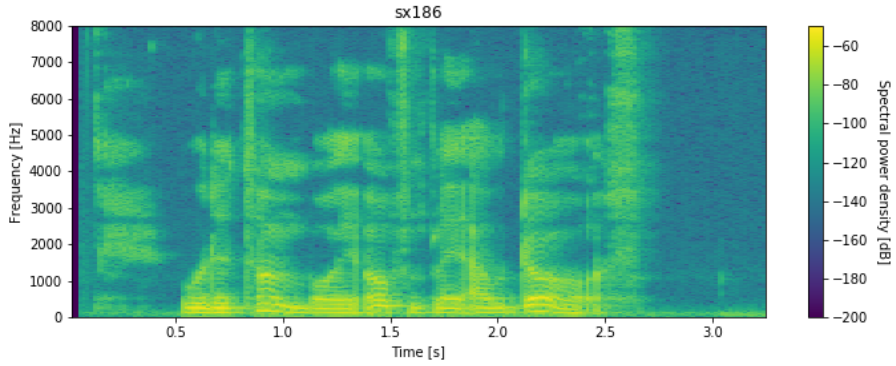
File name	Samples read	Length (seconds)
sa1.wav	71658	4.478625
sa2.wav	56298	3.518625
si1446.wav	99818	6.238625
si2076.wav	44778	2.798625
si816.wav	65258	4.078625
sx186.wav	52458	3.278625
sx276.wav	60138	3.758625
sx366.wav	78058	4.878625
sx6.wav	51178	3.198625
sx96.wav	60138	3.758625

The recordings can be used for:
c) for (a), (b) and for freely available "Czenglish TIMIT" database.

2 Queries

File name	Samples read	Length (seconds)	Query
q1.wav	10163	0.635188	essentially
q2.wav	10393	0.649563	exercise

3 Spectogram



Obrázek 1: Would a tomboy often play outdoors?

4 Features

I used linear bank of filters to calculate features of a given sentence or query. For that I used the matrix multiplication approach $\mathbf{F} = \mathbf{AP}$.

The purpose of matrix $\mathbf{A}$ is to sum every B rows (in our case 16 rows). To make this work, we create the matrix A by filling it with zeros and ones in a specific pattern so that it produces the sum.

The first row of the matrix $\mathbf{A}$ will contain 16 ones and the rest of it will be zeros. The second row will contain 16 zeros, then 16 ones, and the rest will be zeros. And so on. The shape of the matrix $\mathbf{A}$ is $(B, f.size)$ where f is an array of sample frequencies.

5 Correlation score calculation

To calculate the correlation, I use the following function in python:

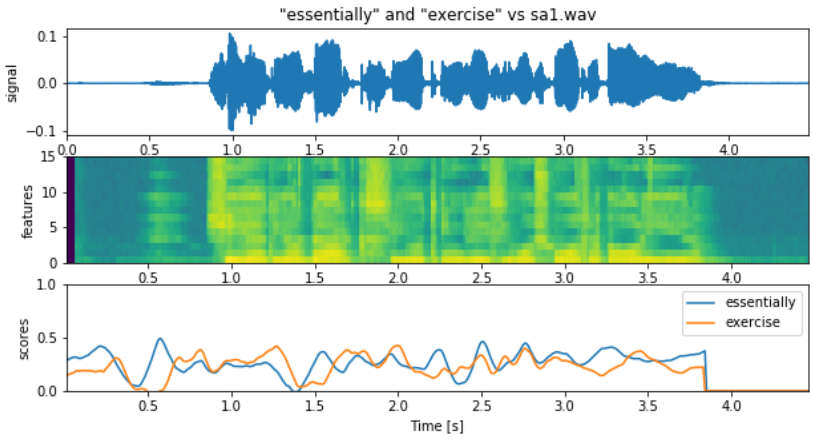
Listing 1: Function to compute the correlation between Q and F at position pp

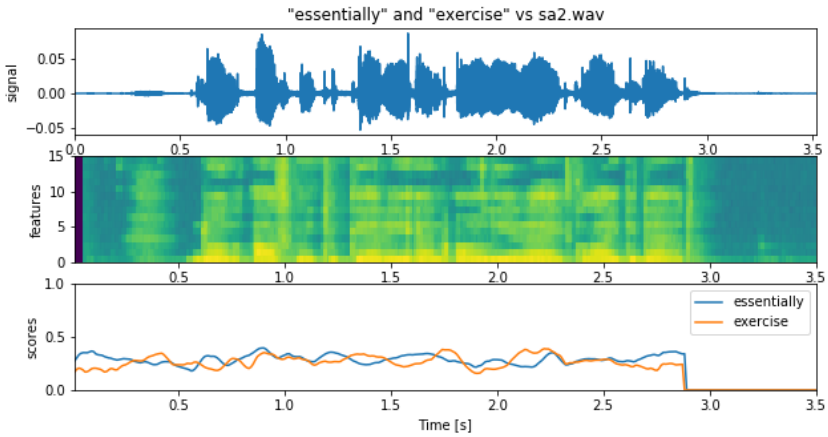
```
"""
Q are query parameters
F are feature parameters
pp is a query position
"""
def compute_correlation(Q, F, pp):
```

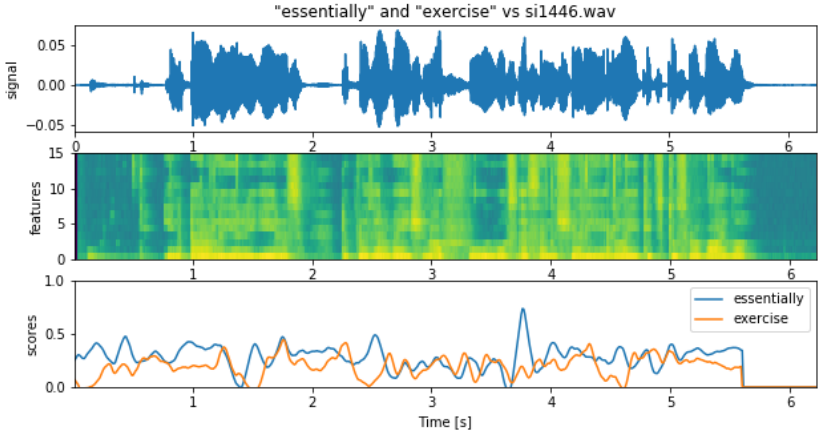
```
    Q_transposed = np.transpose(Q)
    F_transposed = np.transpose(F)

    correlation = 0
    for i in range(Q.shape[1]): # for each vector in Q
            # calculate the pearson correlation
        corr, p_value = scipy.stats.pearsonr(Q_transposed[i], F_transposed[i + pp])
        if not math.isnan(corr):
            correlation += corr # sum all the correlations
    return correlation / Q.shape[1] # normalize the correlation to range <0,1>
```

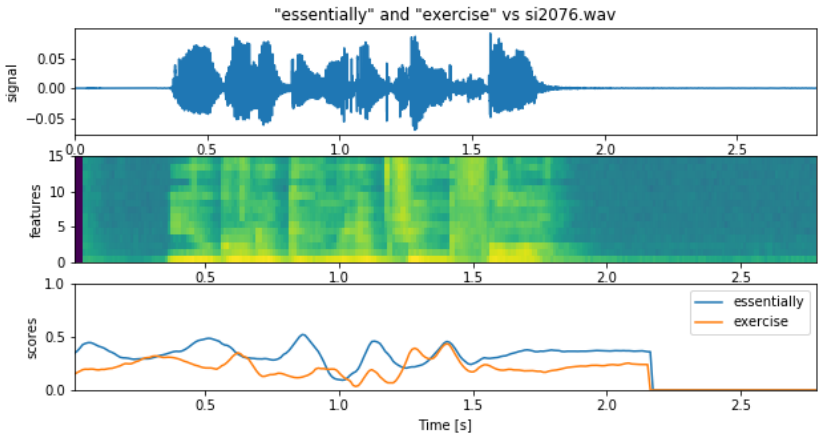
I call this function for $total\_steps = F\_length - Q\_length$ where F_length is the number of vectors in **F** and Q_length is the number of vectors in **Q**. I save all the computed correlations and then simply display them or process them further to find hits.

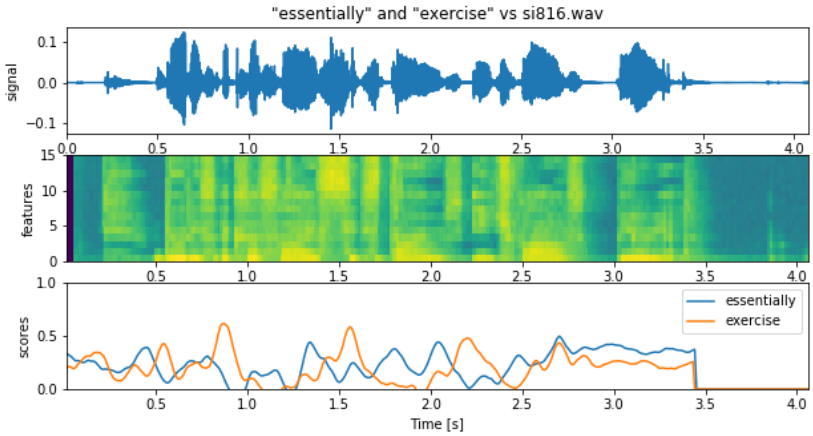
6 Main output

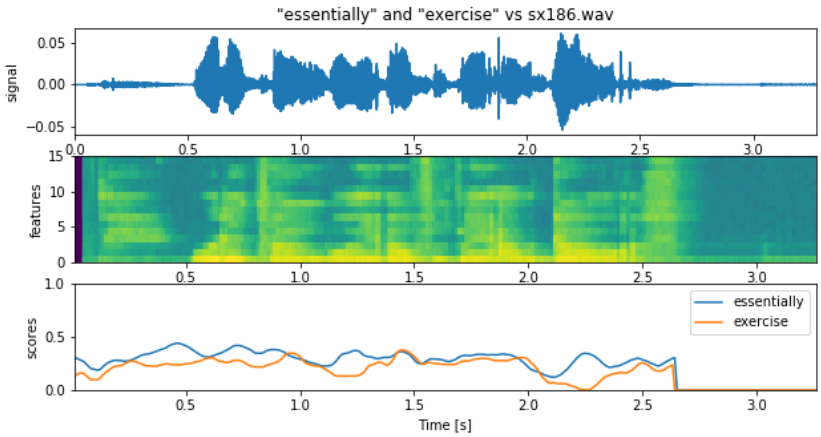


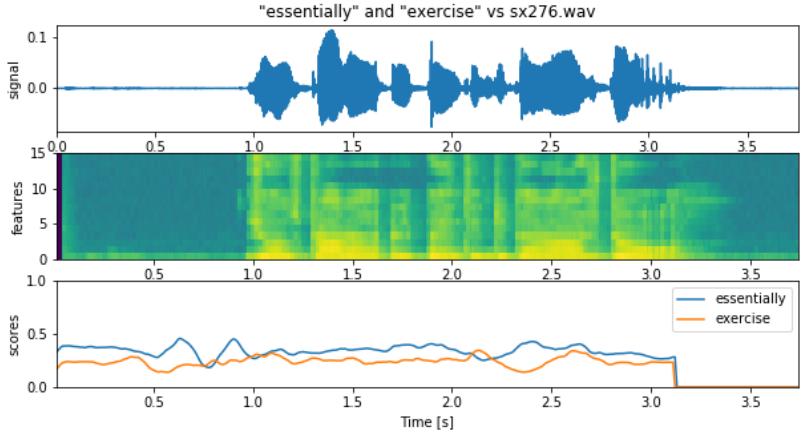

"essentially" and "exercise" vs sa2.wav
signal
features
scores
essentially
exercise
Time [s]

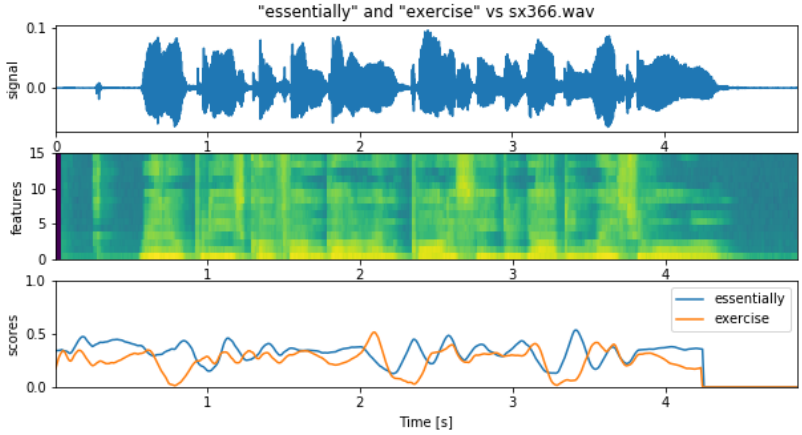

"essentially" and "exercise" vs si1446.wav
signal
features
scores
essentially
exercise
Time [s]

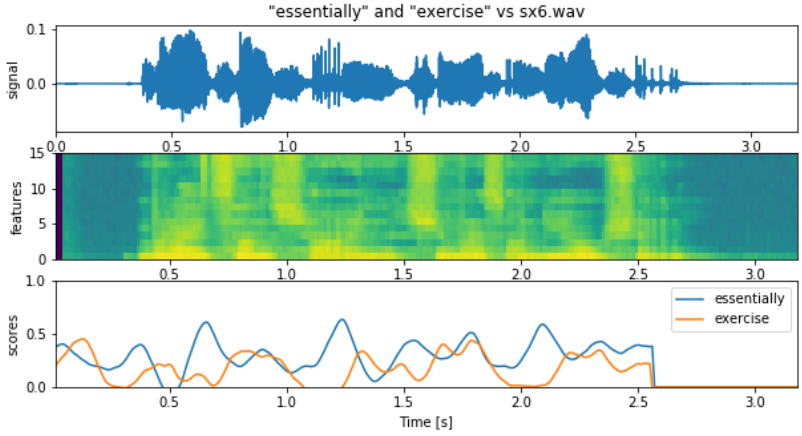

"essentially" and "exercise" vs si2076.wav
signal
features
scores
essentially
exercise
Time [s]

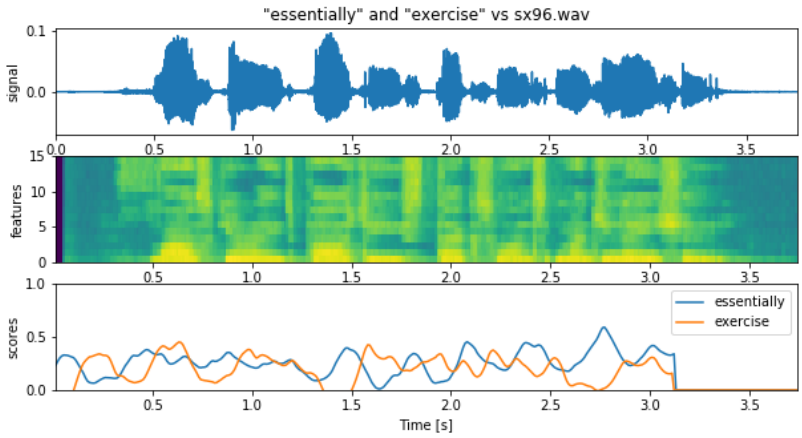

"essentially" and "exercise" vs si816.wav
signal
features
scores
essentially
exercise
Time [s]


"essentially" and "exercise" vs sx186.wav
signal
features
scores
essentially
exercise
Time [s]


"essentially" and "exercise" vs sx276.wav
signal
features
scores
essentially
exercise
Time [s]


"essentially" and "exercise" vs sx366.wav
signal
features
scores
essentially
exercise
Time [s]


"essentially" and "exercise" vs sx6.wav
signal
features
scores
essentially
exercise
Time [s]



7 Score evaluation

To find the occurrences of a query in the sentence, we have to specify a threshold. We can do it simply by examining the graphs of correlations for each sentense. We can easily see that it spikes in some places - these should be the occurrences of a query in a sentence.

All we need to do is to find the spikes programmatically which can be done by checking whether the correlation is above the threshold and checking whether it's still ascending. Once it stops ascending, we know where the peek of the spike is and therefore the beginning of the query.

Query	Threshold
q1.wav	0.7
q2.wav	0.6

8 Hits

The following hits were detected using the threshold defined above. These are the correct occurrences. No incorrect were found.

Recording	q1.wav	q2.wav	Sample from	Sample to
sa1.wav	no	no		
sa2.wav	no	no		
si1446.wav	yes	no	60200	70160
si2076.wav	no	no		
si816.wav	no	yes	13800	23760
sx186.wav	no	no		
sx276.wav	no	no		
sx366.wav	no	no		
sx6.wav	no	no		
sx96.wav	no	no		

9 Conclusion

As we can see, the result is quite precise. The system is sensitive to the defined threshold and therefore the result depends on it. If we set the threshold just a little bit lower, we would get multiple false hits, which we do not want, or we wouldn't get any results at all if we set the threshold too high.

It is also required that the searched query sounds pretty much the same as in the sentence, otherwise it has problems detecting it. The length of the query recording should be similar to the one in the sentence recording.

To increase the efficiency of the algorithm, we could tune some of its parameters. Maybe we could also filter the recordings to get rid of noise.