ISS protokol

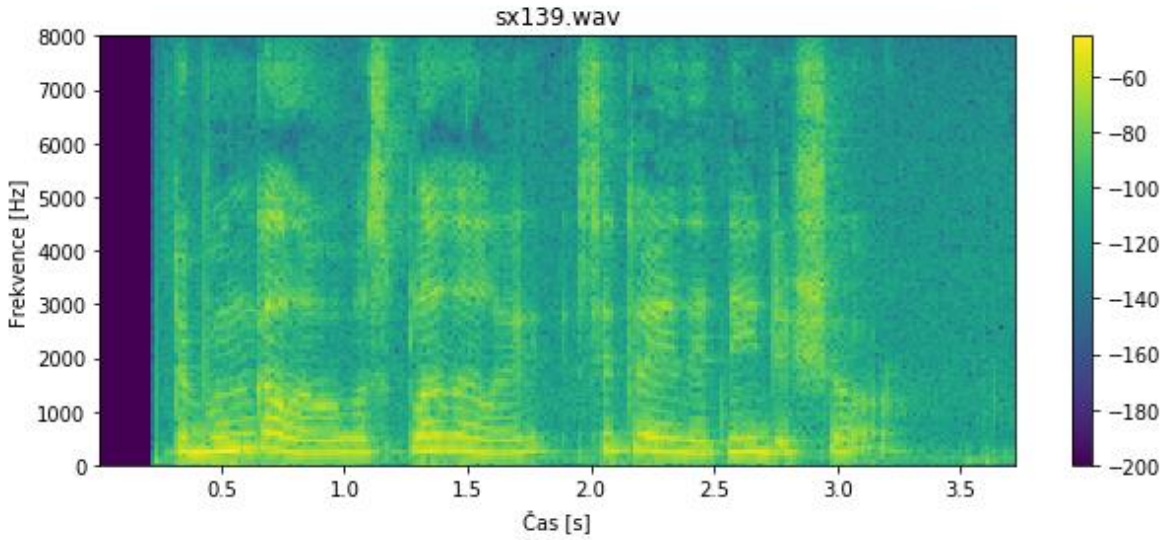
Nahrávky

Věty i slova jsem nahrála na svůj mobilní telefon a následně upravila podle instrukcí v zadání projektu.

název nahrávky	délka v sekundách	délka ve vzorcích
sa1.wav	3,48	61 440
sa2.wav	3,73	59 733
si1399.wav	5,50	88 064
si2029.wav	2,60	41 643
si769.wav	3,33	53 248
sx139.wav	3,73	59 733
sx229.wav	2,37	37 888
sx319.wav	4,31	68 949
sx409.wav	4,54	72 704
sx49.wav	3,41	54 613
q1.wav	1,15	18 432
q2.wav	0,98	15 701

Data mohou být použita pro vyhodnocení ISS projektu a pro výzkum a vývoj v rámci řečové skupiny na FITu BUT Speech@FIT.

Spektrogram



V této úloze jsem se inspirovala kódem Katky Žmolíkové, provedla jsem jen potřebné úpravy pro správné řešení (nastavení délky rámců, jejich překrytí apod.).

```
#vzorkovaci frekvence
Fs = 16000
#delka jednoho ramce
wlen = int(25e-3 * Fs)
#prekryti dvou ramcu
wshift = int(15e-3 * Fs)
#funkce na vyplneni matice spektografu
def spect(s, fs):
    f, t, sgr = spectrogram(s, fs, nperseg=wlen, noverlap=wshift, nfft=511)
    sgr_log = 10 * np.log10(sgr+1e-20)
    return f, t, sgr_log
```

Parametry (features)

Parametry jsem se rozhodla vypočítat tak, jak je doporučeno v zadání, tedy

$$f_0 = \sum_{k=0}^{B-1} P[k], \qquad f_1 = \sum_{k=B}^{2B-1} P[k], \quad \ldots \quad f_{B-1} = \sum_{k=256-B}^{256-1} P[k]$$

, kde B = 16.

Výslednou matici jsem vypočítala jako F = A*P, kde P je matice spektrogramu a A je mnou vytvořená matice. Tuto matici jsem vytvořila takto:

```
A = np.zeros((16, 256))
for i in range(16):
    fill(A, i, 16*i)
```

```
#vytvoreni a naplneni matice A pro features
def fill(mat, row, fromi):
    for x in range(fromi, fromi+16):
        mat[row][x] = 1
```

Matice A má tedy 16 řádků a 256 sloupců. V prvním řádku bude za sebou 16 jedniček a zbytek nuly, v druhém řádku 16 nul, 16 jedniček a zbytek nuly atd.

Pro násobení matic jsem využila vestavěnou funkci np.dot().

Výsledným násobením tedy v prvním řádku výsledné matice F bude prvních 16 vzorků každého sloupce matice P, tedy prvních 16 koeficientů logaritmického výkonového spektra, v druhém řádku budou vzorky 17 – 32 ze sloupců matice P atd., dokud nezpracujeme celou matici P. Výsledkem je matice vypočtených parametrů.

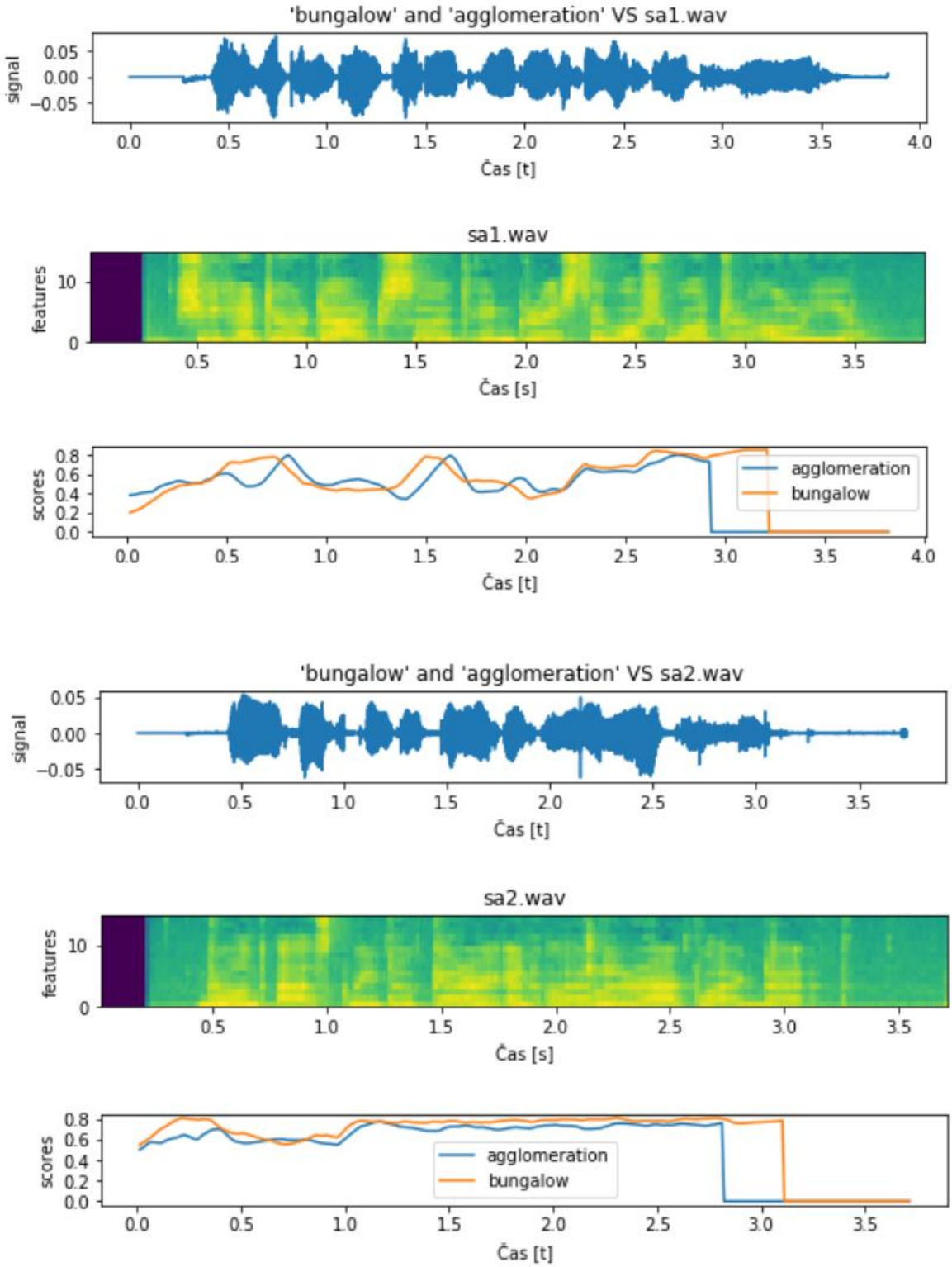
Výpočet skoré

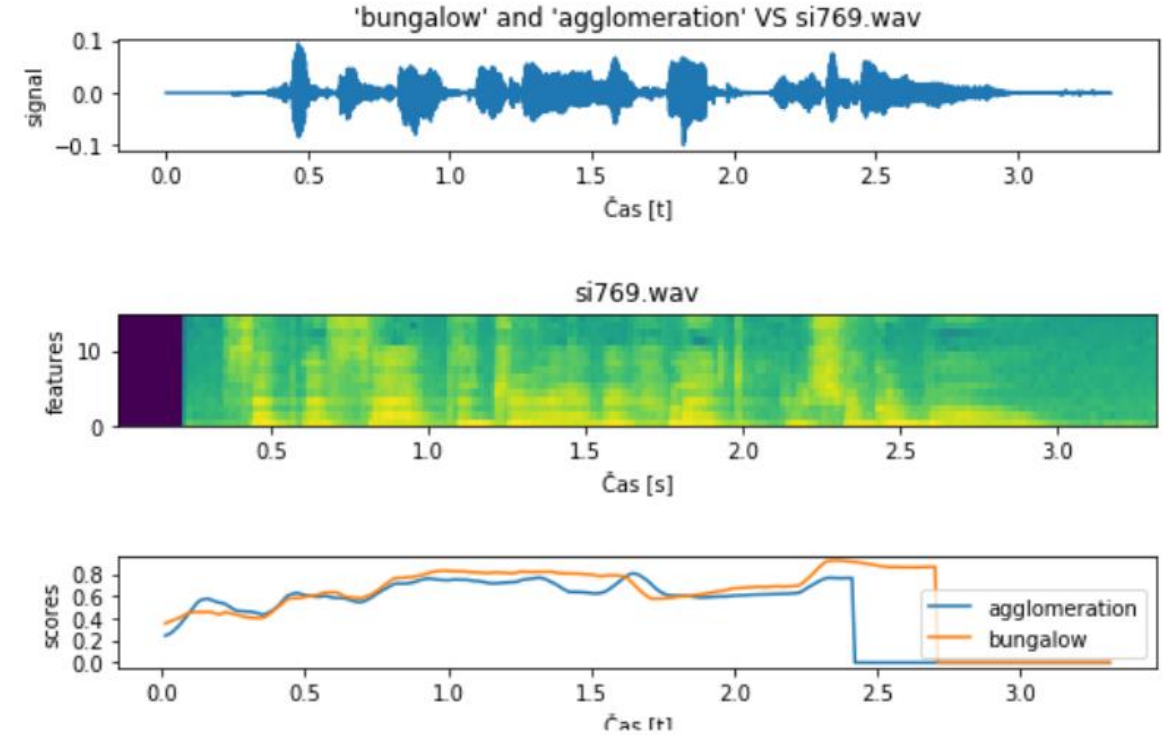
Skoré jsem počítala tak, že jsem matici parametrů slova (dále QF) postupně posouvala po celé šířce matice parametrů věty (dále F). Pro výpočet je nutné matice transponovat. Dále jsem využila funkce pearsonr(), která vypočítá Pearsonovy korelační koeficienty mezi jednotlivými vektory. Tyto koeficienty jsem sečetla a uložila na odpovídající místo v poli hodnot. Cyklus, kde naplňuji pole skoré:

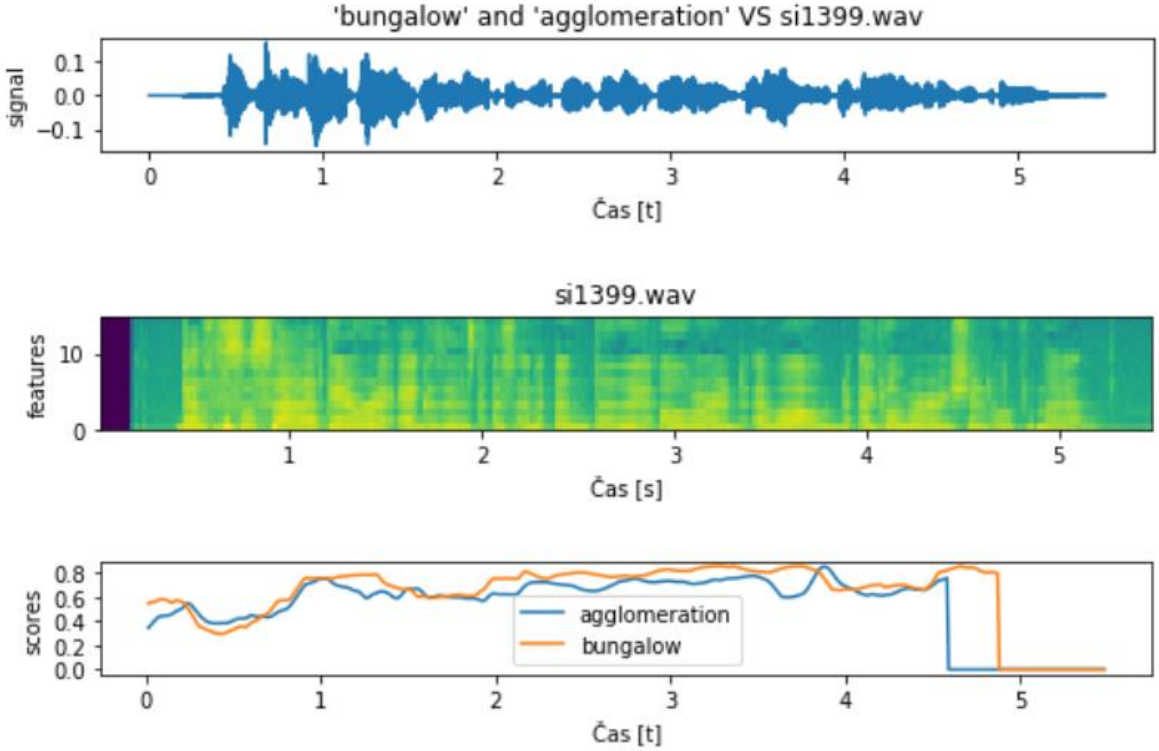
```
for column in range(width):
    for i in range(widthq1):
        corr, _ = pearsonr(QF_trans[i], F_trans[i+column])
        if not math.isnan(corr):
            pear += corr
    pear /= widthq1
    S[column] = pear
    pear = 0
```

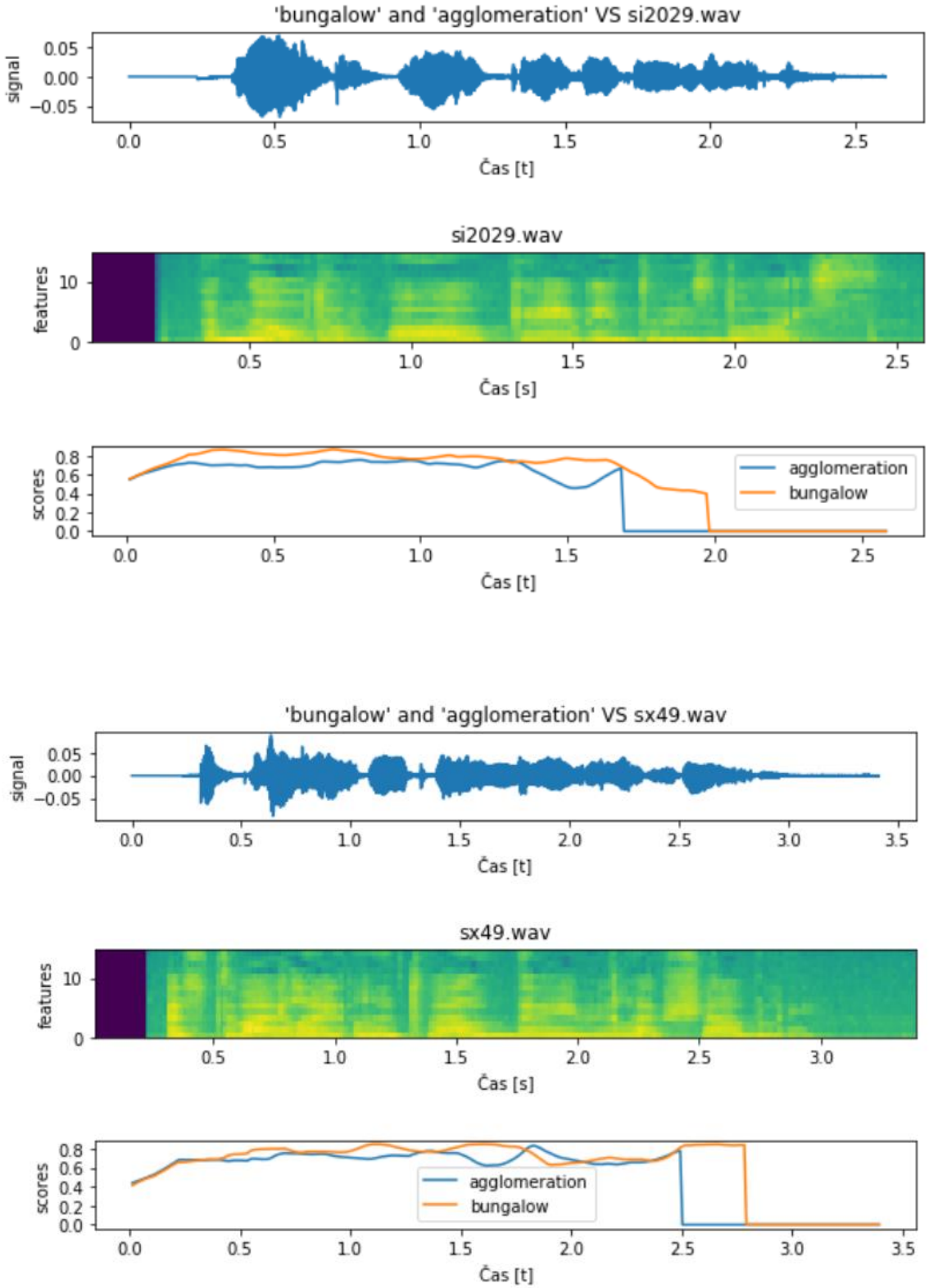
kde width je rozdíl šířky F a QF, width1 je šířka QF; pear je zpočátku nastavena na 0.

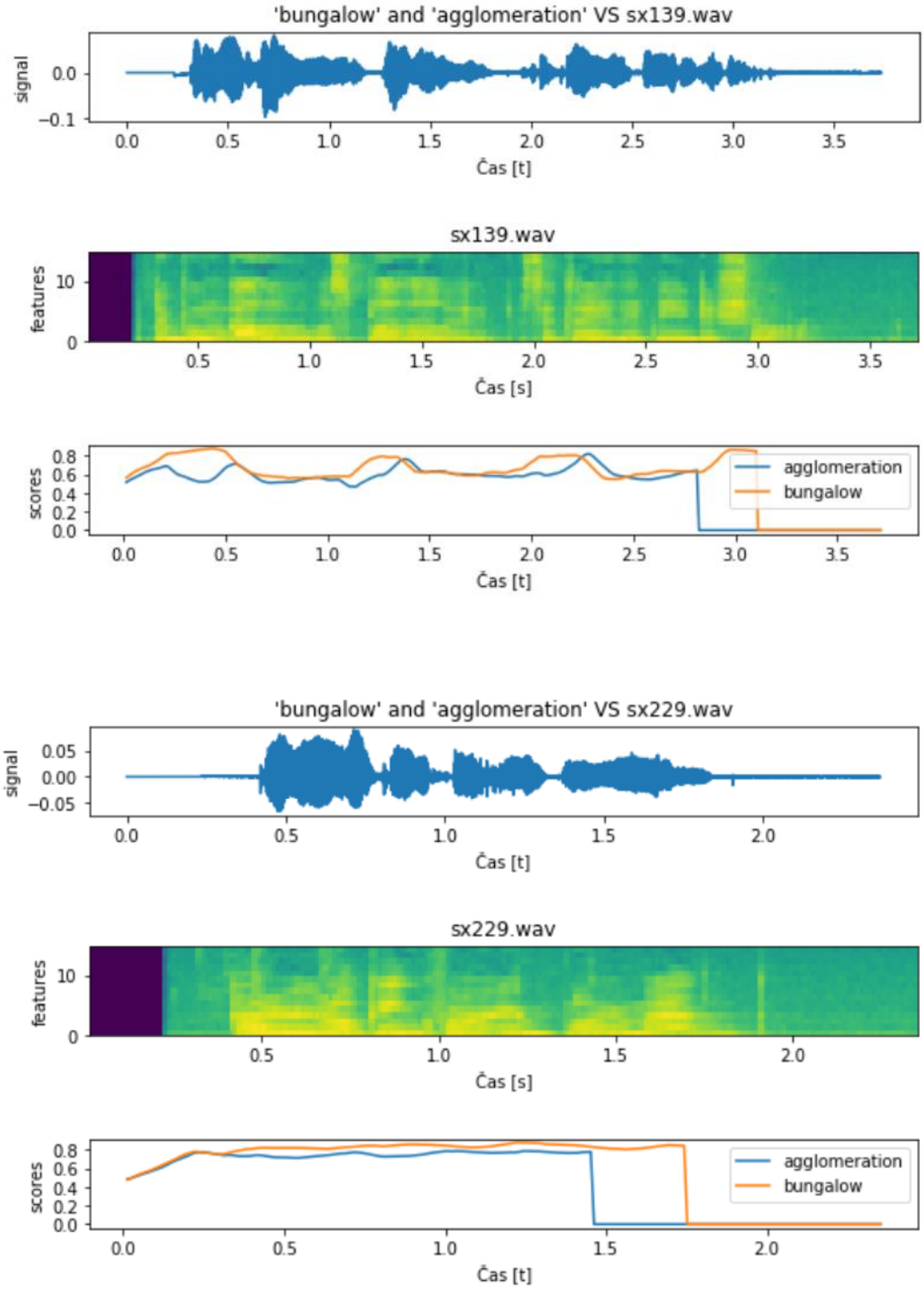
Výsledky

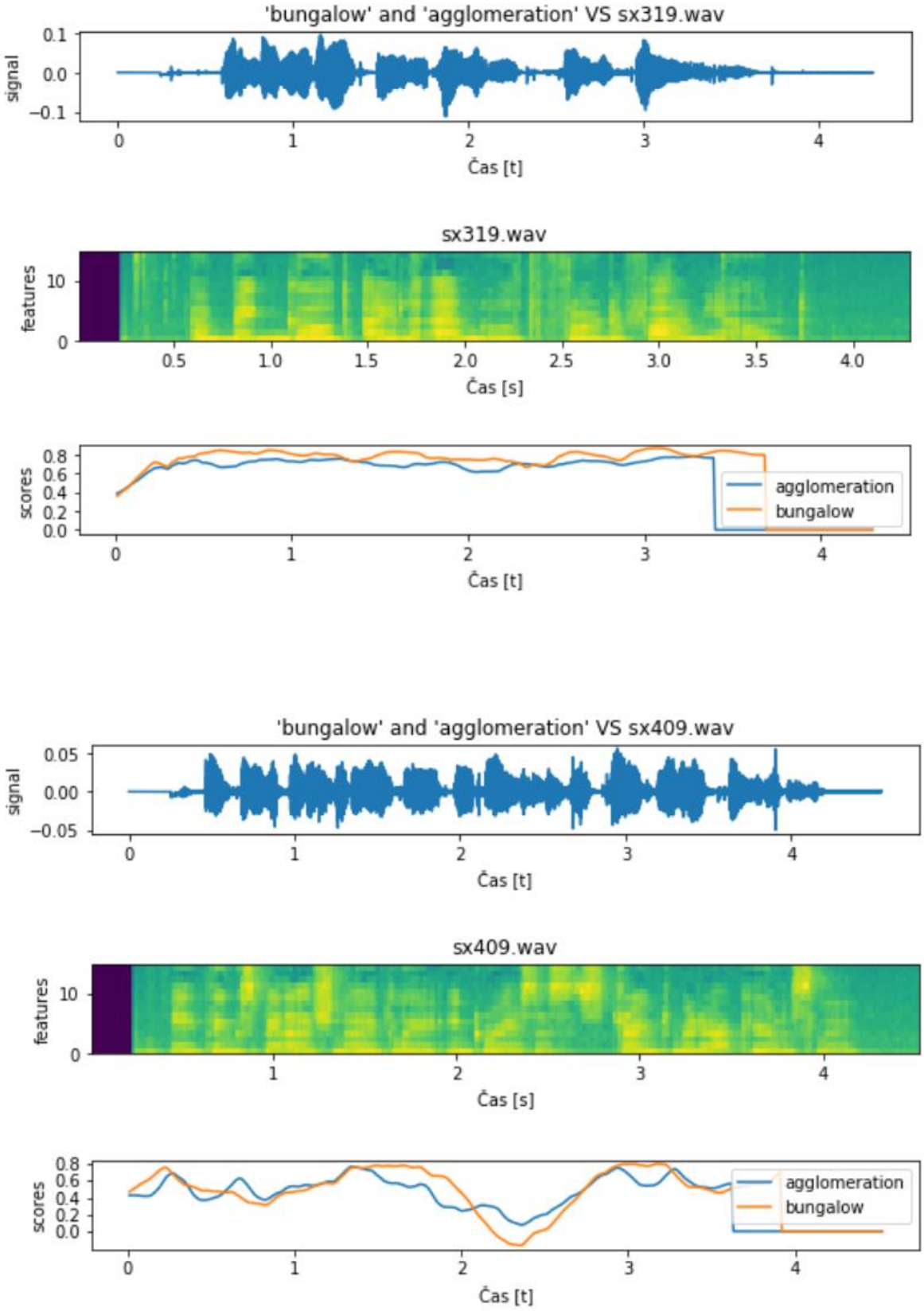

'bungalow' and 'agglomeration' VS sa1.wav
signal
Čas [t]
sa1.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]
'bungalow' and 'agglomeration' VS sa2.wav
signal
Čas [t]
sa2.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]


'bungalow' and 'agglomeration' VS si769.wav
signal
Čas [t]
si769.wav
features
Čas [s]
scores
agglomeration
bungalow


'bungalow' and 'agglomeration' VS si1399.wav
signal
Čas [t]
si1399.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]


'bungalow' and 'agglomeration' VS si2029.wav
signal
Čas [t]
si2029.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]
'bungalow' and 'agglomeration' VS sx49.wav
signal
Čas [t]
sx49.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]


'bungalow' and 'agglomeration' VS sx139.wav
signal
Čas [t]
sx139.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]
'bungalow' and 'agglomeration' VS sx229.wav
signal
Čas [t]
sx229.wav
features
Čas [s]
scores
agglomeration
bungalow
Čas [t]



Nalezení slova

Výskyt slova query ve větě a případně v jakém místě věty se detekuje pomocí rychlé změny hodnoty score směrem vzhůru. Hodnota v intervalu by zároveň měla přesáhnout hodnotu 0,8 pro

q1 i q2. Tuto hranici jsem zvolila, protože vzhledem k odlišnosti hlasitosti nahrávek jsou hodnoty score poměrně vysoké.

Výsledky

	q1	kde	q2	kde
sa1	NE	-	ANO	2,5 – 3,3
sa2	NE	-	ANO	0,2 – 0,3
si769	NE	-	NE	-
si1399	ANO	3,9 – 4, 9	NE	-
si2029	NE	-	ANO	1 – 1,6
sx49	NE	-	ANO	0,4 – 0,9
sx139	NE	-	NE	-
sx229	NE	-	NE	-
sx319	NE	-	NE	-
sx409	NE	-	NE	-

Závěr

Můj detektor sice nalezl daná slova ve větách, ale nalezl i slova, která se neshodovala, a bylo poměrně těžké z grafu vyčíst, kde by se slovo mělo nacházet.

Myslím ale, že je to způsobeno spíše tím, že nahrávky slov jsou o něco tišší a nejspíše byla nahrávána blíže k mikrofonu než věty, proto je rozpoznávání poměrně obtížné. Tento problém by šel vyřešit pomocí jiného či podrobnějšího počítání skoré, abychom získali podrobnější data a byli schopni lépe detekovat daná slova ve větách.

U prvního slova (agglomeration) byla nalezena jen jedna shoda, a detektor se opravdu trefil (což mě samotnou překvapilo, když jsem ustřihla část, kterou jsem si zapsala, a až poté poslechla). U druhého slova (bungalow) bylo shod nalezeno více, což může být způsobeno i samotným slovem, neboť není nikde výrazné.