

ML4IOT HomeWork1

Authors : Azimkhan Orazalin, Ali Abbas Syed, Bilal Shabbir
Studentid : s298255, s292423, s305979
Politecnico di Torino

Abstract—Homework1 first exercise is about Voice Activity Detection Optimization and Deployment In which we are supposed to find the best hyper-parameters of is silence method and to write a new Python script to record audio every second and find out if it's silence or not. and then store it. Second exercise is about developing a memory-constrained battery monitoring system with the given specifications ,create three time-series and get the arguments from the command line.





I. FIRST TASK

To compute **hyper-parameters** for VAD we selected **down sampling rate** of **16 kHz** as with higher sampling rate the accuracy goes up but latency also increase because audio will have higher number of samples that requires more computation. For all other parameters we performed grid search to find the combination of parameters that fulfil our constrains.

The values we used for the different hyper-parameters are

- 1) **frame_length_in_s** = [0.08,0.016,0.032,0.048,0.064]
- 2) **dbFSthres** = [-110,-120,-130,-140]
- 3) **duration_thres** = [0.08,0.1, 0.15]

The hyper-parameters for which we Achieved the **Accuracy > 98 percent** and **Latency < 9ms** are reported in the Figure:

frame_length_in_s ↓	dbFSthres	duration_thres	accuracy	latency
				
0.032	-130	0.1	98.444	8.734
0.032	-130	0.08	98.222	8.745
0.032	-120	0.1	98.222	8.759
0.032	-120	0.08	98.444	8.837
0.016	-130	0.08	98.444	8.74

frame_length_in_s affects both accuracy and latency. For lower **frame_length_in_s** **0.016** , **0.032** VAD gives better accuracy and latency within maximum limit. By keeping all other parameters constant,for lower values of frame length the latency was within the limit but accuracy was less. For higher frame length accuracy did not change much but latency went up.

Good accuracy was achieved for **dbFSthres -120 -130** .For higher threshold VAD can classify silence even when there is sound activity because energy of voice signal is lower than threshold.Threshold for time duration of voice activity also affects the accuracy.Every value from 0.04 to

0.1 for **duration_thres** gave accuracy higher than 98%.High value of threshold for time duration of voice activity (**duration_thres**) can lower the accuracy as there can be voice for brief time.Both **dbFSthres** and **duration_thres** do not affect latency as these are just threshold to classify voice or silence and these parameters do not increase or decrease complexity of VAD.

II. SECOND TASK

A. In this subsection how the three time-series were created is explained!

First argsparse was used to get the arguments to connect to redis and then created three time-series using **ts().create** to make mac battery , mac power and mac plugged seconds and in the while loop we used **ts.add** to add the values in the time-series

To count total seconds when laptop was plugged in during last 24 hours, **ts.createrule** was used. The following para maters were used :

- 1) **Source is mac_address:power**
- 2) **Destination,is mac_address:plugged seconds**
- 3) **sum (type of aggregation)**
- 4) **bucket duration 24 hours (in msec)**

mac power will store the value 1 if plugged , 0 otherwise. mac battery will store the battery percentage and mac plugged seconds will store a single value for every 24 hours by performing an aggregation on the values received from the mac power.

B. Calculations made for setting the retention period of the three time-series

ts.alter function was used to set the RP. Retention period is set in a way that memory usage for **mac_address:battery** and **mac_address:power** is less than 5MB, the given formula was used :

$$RP = 5 \cdot \frac{2^{20}}{1.6} \cdot 1000 \approx 38days \quad (1)$$

Where 5 is a limit, 2^{20} is 1MB in bytes, 1.6 is bytes for each record considering 90% of compression and 1000 is msec.

For **mac_address:plugged seconds** limit is 1MB, and calculation is:

$$RP = \frac{2^{20}}{1.6} \cdot 24 \cdot 60 \cdot 60 \cdot 1000 \approx 1795years \quad (2)$$

Where $24 \cdot 60 \cdot 60 \cdot 1000$ is total milliseconds in one day.