Machine Learning for IoT - Homework 01

Alice Colombatto s289922 | Arcangelo Frigiola s295406 | Francesco Grandi s306272

Exercise 1 - VAD Optimization and Deployment

In this part, we had to find the correct setting of the downsampling rate (that is the operation of reducing the sampling rate from the original signal), the frame length (the frame length expressed in seconds of the STFT done on the signal), the decibel threshold (that distinguishes between the sound level of speech and silence) and the minimum time threshold under which the record, even with a short interval of speech, is classified as silence. In the beginning, we assumed that the threshold values showed the correct way to distinguish silence from speech, identifying correctly the rumor. Whereas, after some tests, we found out that the main problem was the dB threshold. We arrived at this conclusion through the inspection of the .wav files of the *vad-dataset*, noticing that some of them were registered with a minor sound level than others, determining a wide difference between them. For this reason, we assumed that changing the threshold in decibel would allow us to obtain higher accuracy.

Downsampling rate	Frame length	dbFSthres	duration thres	Accuracy	Latency
16000	0.064	-110	0.1	98.33%	8.9 ms
16000	0.064	-120	0.1	98.11%	8.67 ms
16000	0.032	-130	0.1	98.4%	8.906 ms

First of all, recalling results shown in the table above, it's important to highlight that the latency with similar hyper-parameters depends mainly on the speed of connection and the hardware used while the program is running. From theory, we know that the performances improve when the values of *frame length* are powers of 2, so for testing, we used only 0.016, 0.032, and 0.064. Moreover, since we had to make a compromise between accuracy and latency, we assessed that the only way to respect both thresholds was to keep the down-sampling rate equal to 16 kHz. Finally, we decided to keep the duration threshold of silence equal to 0.1. Indeed, we observed that changing it didn't improve the final accuracy nor affected the latency, as this doesn't influence the complexity of the algorithm or the amount of data to analyze. This is explained by the fact that generally 0.1 seconds of noise can be found in silence-labeled files.

Exercise 2 - Memory-constrained Time Series Processing

The goal of this exercise was to design and develop a memory-constrained battery monitoring system for the following time series: *mac_address:plugged_seconds*, *mac_address:battery*, and *mac_address:power*.

Starting from the solution of Lab1, we report the setup of the Redis time series *mac_address:plugged_seconds*. First, we set the chunk size to `128 bytes`, multiple of 8, restricting Redis default of `4 KB`, in order to use less memory. Then, we set the *retention period* to the largest possible value, to meet the memory size constraints. We report all the computations at the bottom. Furthermore, we managed the case in which timestamp is older than the *retention period* compared to the maximum existing timestamp, that would eventually cause an error.

At sampling time, we set the acquisition period to 24 hours, checking every second if the battery is plugged or not. If plugged, the seconds counter is updated. Once the time series is recorded, the counter is reset. We made the following approximation: we considered the processing time of the entire `while()` cycle, $\approx 9.5 \cdot 10^{-4}$ *milliseconds*, negligible in our computations, as well as the header size of the chunk.

Considering an average compression ratio of 90%, and a default Redis payload size of `16 bytes`, we compute the *retention periods* as follows:

$$\#\ records_{compressed}\ (5MB) = \frac{5.242.880\ bytes}{16\ bytes \cdot (1 - 0,9)} = 3.276.800$$

$$\#\ records_{compressed}\ (1MB) = \frac{1.048.576\ bytes}{16\ bytes \cdot (1 - 0,9)} = 655.360$$

$$Retention\ Period\ (5MB) = 3.276.800\ records \cdot \frac{1\ s}{1\ record} \approx 37\ days$$

$$Retention\ Period\ (1MB) = 655.360\ records \cdot \frac{24\ h}{1\ record} \approx 1795\ years$$