Machine Learning for IoT - Politecnico di Torino

Homework 1 report

Paolo Aberto
StudentID : s278098

Lorenzo De Nisi
Student ID: s276545

Carmine De Stefano
Student ID: s278176

1 Data Preparation: Sensor Fusion Dataset with TFRecord

When building the TFRecord dataset, the .wav files were imported by using the read_file method from the tf.io module, and a BytesList data-type was used to embed them into the TFRecord. For the remaining data (POSIX timestamp, temperature and humidity), we decided to use an Int64List data-type. We also experimented with the FloatList type, but in the end we did not see an improvement in the dimensions of the output file.
The csv.reader method was used to parse the input file, processing the csv row by row and avoiding to import it into external libraries like pandas, in order not to add extra complexity.

2 Low-power Data Collection and Pre-processing

The first implementation of the script had a pre-procesing latency of more than 80ms. In order to reduce it, different techniques were applied to improve efficiency.
As a first step, we identified all the calculations that were not dependent on the actual audio signal. These calculations were repeated at each execution, so we pre-computed them, and passed them to the methods as parameters.

When experimenting with power management we tried to begin the recording phase in powersave mode, and switching to performance mode right before the beginning of pre-processing. While this proved to reduce preprocessing time just under the 80ms treshold, a further improvement on execution time was obtained by enabling the power save mode when the recording was 85% complete. The switch to performance mode, in fact, is not immediate, and in this way the transition is completed before the beginning of the resampling phase.

Another improvement was made in the recording phase, by storing the recordings in a data buffer, instead of using a python list. This was made possible by using the BytesIO buffer from the standard python io module. Some milliseconds were also saved by using the tf.math package when normalizing the audio tensor in the [-1, 1] range.

In the following tables we can see the average time measurements and VF levels after 5 recordings:

Phase	**Average Duration**
Recording	1.004s
Resampling	0.005s
Spectrogram	0.048s
MFCCs	0.008s
Writing	0.001s
Total elapsed	**1.066s**
Preprocessing	**0.062s**

Table 1: Average duration for different phases

VF level	**Time**
600000	475
750000	0
1000000	0
1500000	67

Table 2: VF levels after 5 recordings