ML4IOT: Homework 2

Tommaso Massaglia
s292988@studenti.polito.it

I. Exercise 1: Multi-Step Temperature and Humidity Forecasting

The first exercise required to build and train two models for multi-step temperature and humidity forecasting based on the Jena Climate dataset.

A. Preprocessing

The dataset was divided in windows of 6 elements, then for each window either the subsequent 3 or 9 labels were assigned (depending on the model) as true values.

B. Model

The chosen model was a **1 dimensional Convolutional Neural Network (CNN-1D)** for its performance to size ratio. Starting from the baseline given in point 1.3 of lab 3, through experimentation I found that the best way to reduce its final size while keeping performance was to discard the first of the two Dense layers and to reduce greatly the number of filters of the Conv1D layer.

What I ended up with was:

- For the 3 step model: *filters = 15, kernel_size = 3*
- For the 9 step model: *filters = 24, kernel_size = 4*

While testing different configurations, I noticed that the 9 step model consistently began performing worse after the 12th epoch; for this reason the first model was trained over 20 epochs while the second one over 12. Other than the multi-output MAE (adjusted for the task), *Accuracy* was added as a metric as it showed great improvements.

C. Model Optimization

The main optimization method (in training) was *Magnitude Based Weight Pruning* to achieve a higher sparsity; sparse models are easier to compress, guarantee latency improvements, and more than anything have little accuracy loss compared to the size reduction they can achieve.

Pruning was implemented using Keras *Polynomial Decay*; the initial step was delayed as it showed positive results (up to a limit) performance-wise with little size cost.

Both models were then quantized after training using $tf.lite.Optimize.DEFAULT$ and compressed using $zlib$ by a factor of 8 (zlib had little impact on the final performance and showed little difference in size going over 5).

pruning params		
out_steps	3	9
initial_sparsity	0	0
final_sparsity	0.91	0.91
begin_step	end_step*0.1	end_step*0.25
end_step	end_step	end_step

II. Exercise 2: Keyword Spotting

The second exercise required to optimize a keyword spotting model trained with fixed labels on the *mini speech command* dataset.

A. Preprocessing

The main objective of the exercise was to reduce model size while keeping a good enough inference time and accuracy, as I found that keeping accuracy high was harder than keeping inference time low, I opted to preprocess the audio samples to extract the *Mel-frequency Cepstral Coefficients* using theese parameters:

mfccs parameters	
frame_length	40
frame_step	20
#_mel_bins	50
mel_frequency	16000
#_mfccs	10

B. Model

The baseline model was a **Depthwise Separable Convolutional Neural Network (DS-CNN)** (the same seen in point 2.3 of lab 3) for its high accuracy in the keyword spotting task, allowing for a harsher pruning during training. The three versions of the model used a different number of filters for each Conv2D layer.

	a	b	c
Conv2D 1	256	128	128
Conv2D 2	256	128	100
Conv2D 3	128	128	90

C. Model Optimization

As seen in exercise 1, Weight Pruning and post training quantization were chosen as optimiziations to lower model size, as well as compression using *zlib*:

pruning params			
model	a	b	c
initial_sparsity	0	0	0
final_sparsity	0.5	0.2	0.5
begin_step	end_step/2	end_step/2	end_step/2
end_step	end_step	end_step	end-step

D. Inference Parameters

To fit the inference constraints I "halved" the file quality, fitting the constraint.

```
--mfcc --resize 32 --rate 8000 --length 320
--stride 160 --bins 30 --coeff 10
```