Machine Learning for IoT——HW2

Gaetano Salvatore Falco	*Zafar Abdirasulov*	*Kuerxi Gulisidan*
S280209	*S301367*	*S304915*

Exercise 1 - Training & Deployment of a "Go/Stop" Classifier

1.1 Training & Optimization

In order to find compliant hyper-parameters, we applied the following strategies:

- We reduced the module size with an aggressive width scaling, with the parameter alpha set to 0.14
- We then changed the parameters for the pre-processing of the audio, according to the table below:

Downsampling_rate	**frame_length_in_s**	**frame_step_in_s**	**num_mel_bins**	**lower_frequency**	**upper_frequency**
16000	0.032	0.032	31	80	8000

Training hyper-parameters:

Batch_size	**Initial_learning _rate**	**end_learning_rate**	**epochs**
32	0.015	0.001	70

For the model we used Convolution 2D layer with 128 initial filters, and we adopted the width scaling and weights pruning as different strategies to meet the constraints.

Our model ends up with the following constraints table:

Accuracy	**Latency**	**Original TFLite Model size**	**ZIP TFLite Size**
98.0% > 97%	4.7 ms < 8ms	24.7 kB < 25kB	14.07 kB << 25kB

As we need a really high accuracy, we opted to use mfcss to do the pre-processing of our input data. As we had to create a model with less than 25kB size, we adopted a really aggressive parameter for the width scaling and used tensorboard - hparams to fine tune the model to a minimum. This allows us to achieve a tflite size of the model of 24.7kB and also reduced the latency to meet the constraint.

1.2 Deployment & Integration in Smart Battery Monitoring

In order to complete this laboratory, we designed a simple script as follows:

1. At the beginning of time, the system is not monitoring and it is receiving the input audio as a stream, calling the callback function
2. The callback function checks if there is noise in the 1s received audio, if silence is detected it does nothing
3. When there is noise, we use the function "calculate_next_state_FSM" to calculate if a "go" or "stop" is received, by invoking the tflite model and testing if the predicted label for go or stop is higher than the given threshold of 95%
4. When "go" is detected, our global variable "state" goes to true and we can log the information we collect to redis
5. If there is silence or there is an unrecognized sound or when the predicted label is not higher than the threshold, the global variable "state" remains unchanged.