# Machine learning for IoT - Team 5
# Homework 2

Stefano Chiartano s313651@studenti.polito.it

Abdirashid Chorshanbiyev s314800@studenti.polito.it

Giuseppe Cavaleri s314734@studenti.polito.it

The aim of this homework was to develop a model for "yes/no" spotting, which had to be accurate, but at the same time fast and light in order to be run in an IoT environment. Based on this, we utilized Mel Frequency Cepstral Coefficients to extract relevant features from the audio samples: MFCCs are a feature representation widely used in audio signal processing and machine learning models for tasks such as speech and audio recognition.

To preprocess the input signals and extract the MFCCs we utilized different preprocessing hyperparameters:

`sampling_rate`: The rate at which audio is sampled, in our case 16000Hz.

`frame_length_in_s` and `frame_step_in_s`: Frame length and step determine the temporal segmentation of audio. Smaller values capture finer details but may increase computational load. Utilizing a power-of-2 frame size enables the application of the FFT algorithm, thus decreasing latency.

`lower_frequency` and `upper_frequency`: Lower frequency and upper frequency define the range of frequencies considered.

`num_mel_bins`: The number of Mel bins influences the resolution of the Mel spectrogram, balancing detail with computational efficiency.

`num_coefficients`: MFCC coefficients enable a concise representation of mel-frequency scaling and cepstral analysis, higher values capture more information.

| SR (Hz) | Frame Length (s) | Frame Step (s) | Lower f. (Hz) | Upper f. (Hz) | N Mel Bins | N Coeff |
|---|---|---|---|---|---|---|
| 16000 | 0.032 | 0.032 | 0 | 8000 | 30 | 13 |

In the context of training, additional hyperparameters are engaged, which are fine-tuned to achieve peak performance in the model. The following are the specific hyperparameters involved:

`batch_size`: Batch size specifies the number of training examples processed in each iteration, impacting training speed and model detail capture.

`initial_learning_rate` and `ending_learning_rate`: The initial learning rate sets the step size for weight updates, aiding early learning. The ending learning rate, kept low, prevents overfitting by stabilizing training conclusion.

`epochs`: Epochs represent the number of times the entire dataset is processed during training, iterating to update model parameters.

`alpha`: alpha scales the number of filters for each 2D Convolution layer, impacting final model size and latency.

`initial_sparsity` and `ending_sparsity`: denote the pruning rate, signifying network density in terms of connection sparsity.

| Batch Size | Initial LR | End LR | Epochs | alpha | Initial sparsity | End sparsity |
|---|---|---|---|---|---|---|
| 10 | 0.01 | $1 \times 10^{-5}$ | 20 | 0.4 | 0.2 | 0.82 |

To determine the final hyperparameters, we employed a grid search, focusing on values in accordance with theoretical guidelines. The primary parameters under investigation were `num_mel_bins` and `num_coefficients`. The accepted range for the former usually spans from 20 to 40. Regarding the latter, the optimal range for cepstral coefficients in speech analysis is typically between 12 and 20. To strike a balance between capturing relevant information and maintaining computational efficiency, we opted for **30** mel bins and **13** cepstral coefficients. Given their relatively minor impact on the model, we did not prioritize the other parameters in the grid search.

This model architecture follows a pattern suitable for deployment in resource-constrained environments, using Depthwise Separable Convolutions to reduce the number of parameters while retaining expressive power. The model concludes with a global average pooling layer and a dense layer with softmax activation for classification. The specific size of the filters and other parameters depends on the value of alpha.

Selecting `alpha` as **0.4** effectively reduces the number of parameters, resulting in noteworthy savings in both model size and latency. Regarding the optimization techniques, pruning was employed to remove less crucial connections in the network, thereby increasing the sparsity and enhancing overall efficiency.

To achieve high accuracy while mitigating overfitting on the training data, the number of `epochs` was set to **20** in our approach.

The model, with optimized hyperparameters, Depthwise Separable Convolutions, and pruning, achieves 99.00% accuracy. Its compact `tflite.zip` size (23.89 KB) and 53.91% latency savings make it effective for resource-constrained environments.

| Accuracy (%) | tflite.zip Size (Kb) | Total Latency Savings (%) |
|---|---|---|
| 99.00 | 23.89 | 53.91 |