

ML4IoT - HW2 Report  
Group 11  
January 7, 2025

## Training and Deployment of a “Up/Down” Keyword Spotter

### Methodology

We developed a data pipeline to process the ”down” and ”up” keywords from dataset. To meet the requested constraints, we employed an iterative approach to hyperparameter tuning and model optimization. For each audio sample, we extracted MFCC (Mel-Frequency Cepstral Coefficients) features to achieve high accuracy with a small model size.

### Model Architecture and Training Strategy

A lightweight CNN was used to minimize latency and size while maintaining high accuracy. The model was trained using the Adam optimizer with a PolynomialDecay learning rate scheduler. Early stopping and model checkpointing were employed.

We implemented SepResNet8, a lightweight CNN architecture featuring an initial 3x3 separable convolution layer (64 channels, stride 2), followed by two residual blocks with dual 3x3 separable convolutions, each utilizing batch normalization and ReLU activation. The architecture concludes with global average pooling and softmax classification.

### Hyperparameters

Below are the final preprocessing and training hyperparameters used for this homework:

Table 1: Preprocessing Hyperparameters

Parameter	Value
Sampling Rate	16000 Hz
Frame Length	0.04 s
Frame Step	0.02 s
Number of Mel Bins	40
Lower Frequency	20 Hz
Upper Frequency	4000 Hz
Number of MFCCs	10

Table 2: Training Hyperparameters

Parameter	Value
Batch Size	20
Initial Learning Rate	0.01
End Learning Rate	1e-5
Epochs	50

### Results

The final model meets all specified constraints:

Table 3: Performance Metrics

Metric	Value
Test Accuracy	99.5%
TFLite Size	41 KB
Total Median Latency	39.7 ms

## Conclusion

This project demonstrated the effectiveness of iterative optimization for constrained IoT applications. The final solution balances accuracy, size, and latency, making it suitable for real-time keyword spotting on embedded systems. Key to our success was the adoption of quantization-aware training and an adaptive learning rate schedule (PolynomialDecay), which helped maximize performance while minimizing overfitting. Early stopping further ensured that the model generalizes well to unseen data.