HW1 – Group1

Toprak Mert (s336966) **Leflef Daghan Ufuk** (s338142) **Oral Muhammed Emin** (s343124)

1. Whisper Evaluation

We evaluated the performance of Whisper models by measuring classification accuracy (on Google Colab with GPU) and inference latency (on a Raspberry Pi 4, 2GB RAM). Table 1 summarizes the results.

Accuracy and Resource Trade-Off: Accuracy increases with model size, reaching 92.88% for the **small** model. However, memory usage grows dramatically: **medium**, **large**, and **large-v2** models exceed the available RAM on both GPU and Raspberry Pi, resulting in Out-of-Memory (OOM) failures during evaluation.

Latency: On the Raspberry Pi, the **tiny** model achieves a practical median latency of 2.67s. Latency increases substantially for larger models: **base** reaches 6.30s, while **small** becomes unstable due to memory swapping (25.37s), making it unsuitable for real-time voice interaction.

Final Assessment: The **tiny** model offers the best trade-off between accuracy, latency, and memory constraints. With nearly 90% accuracy and stable execution on the Raspberry Pi, it is the only model suitable for a responsive, real-time VUI.

Model	Mem. (MB)	Acc. (%)	Lat. (s)
tiny	144.04	87.75	2.67 ± 0.73
base	276.92	91.00	6.30 ± 0.41
small	922.14	91.75	25.37 ± 27.10
medium	2913.88	91.88	*OOM*
large	5887.24	*OOM*	*OOM*
large-v2	5887.24	*OOM*	*OOM*

Table 1: Accuracy and memory measured on Colab (T4 GPU); Latency (median ± std) on Raspberry Pi 4.

For the tiny, base, small, and medium models, accuracy was computed directly on the test set. In contrast, the large and large-v2 models could not be evaluated due to memory and computation limits. Their memory footprint was therefore estimated analytically from the parameter count using FP32 storage, ensuring a consistent "Memory (MB)" metric. Accuracy is reported only for models that could be executed.

2. Discussion on Custom Training Pipeline

Motivation for Replacing Whisper: While Whisper performs well as a general-purpose ASR system, its computational demands make it unsuitable for constrained edge hardware. Even the **tiny** model exhibits noticeable latency, and larger versions exceed device memory limits.

Benefits of a Specialized Architecture:

- **Lower Latency:** Lightweight models such as **DS-CNN** can achieve sub-second inference, essential for real-time keyword detection.
- **Task Specialization:** Restricting predictions to a closed vocabulary ("up", "stop") minimizes false activations and reduces hallucinations common in open-vocabulary models.
- **Resource Efficiency:** Structured pruning (e.g., via **PIT**) significantly decreases model size, enabling deployment on low-memory devices.

Challenges:

- **Dataset Requirements:** Custom models require proper collection and balancing of task-specific training data.
- **Reduced Flexibility:** Unlike Whisper, a KWS model cannot perform general speech transcription.

Proposed Training Pipeline: To build an efficient keyword-spotting model suitable for real-time execution on the Raspberry Pi, we propose a compact four-stage pipeline:

1. **Data Preparation:** We employ the Mini Speech Commands dataset, standardizing all audio to 16 kHz and fixed-length segments. A dedicated "Background" class is added using silence and environmental noise to reduce false activations during continuous listening.
2. **Feature Extraction:** Raw audio is transformed into low-dimensional **MFCC** features, which preserve key spectral patterns while significantly reducing computational cost. MFCCs provide an efficient 2D representation well suited for small convolutional models.
3. **Model Architecture & Pruning:** A lightweight **DS-CNN** is trained for closed-set command recognition. To further compress the network, we apply structured pruning (PIT) to remove entire filters or channels, combined with unstructured weight pruning to eliminate redundant connections.
4. **Quantization & Deployment:** The pruned model is quantized to **INT8**, reducing both size and latency with minimal accuracy loss. The final model is exported to ONNX for fast inference on the Raspberry Pi, enabling reliable sub-second response times.