Summary of the project

The project aims to implement a real-time speaker verification network on low-power hardware. Speaker verification networks typically have many parameters and layers, often consisting of CNN layers and attention mechanisms, outputting an embedding vector, which serves as the speaker's voice signature. The network must be compressed in order to deploy it on low-power hardware. In this project, we used the STM32L4R9I-DISCOVERY board, which has no operating system and contains 2MB of FLASH memory and 640KB of RAM, meaning the model size has to be less than 2MB. The chosen speaker verification models for the project are ECAPA-TDNN and ResNet, two widely accepted models in speaker recognition. To meet the memory requirement, various model compression methods were examined. The standard methods in the literature include weight pruning, quantization, and knowledge distillation. It was found that only quantization contributed to reducing the model size. Weight pruning (i.e., zeroing out weights) did not help reduce the model size since the model could not be stored without the zeros and implemented on the hardware. Thus, this method was abandoned. Quantization alone was insufficient to achieve a small enough model, as these are large models, necessitating changes to the network architecture by reducing layers and filters. Therefore, knowledge distillation was used, where a large model known as a "teacher" and a smaller model known as a "student" are trained, adding an error term accordingly. In this case, the network was trained with the MSE error between the two models' embeddings and the cross-entropy error against the labels. In quantization, weights and activation functions are reduced to less precise values of 8 bits instead of 32 bits. We used quantization-aware training (QAT), where the model is trained as if quantization has been applied, with the quantization error learned during training. Only at the end of the training process is the network converted to 8-bit.

Five models were trained from scratch:

1. Gold standard ECAPA
2. Gold standard ResNet
3. Distillation tiny version ECAPA
4. QAT tiny version ECAPA
5. Tiny version ResNet

The ECAPA model performed better than ResNet, so the focus shifted to research quantization and knowledge distillation methods for this model. It was discovered that quantization (performed in PyTorch) does not support certain functions and layers in the model, requiring modifications. For example, quantization does not support arithmetic operations and a SoftMax function, so changes were made accordingly. Additionally, the hardware does not support certain functions and layers, such as concatenation or expansion, or a clamp function used in the ECAPA model, necessitating architectural adjustments. This required iterative trial and error until the models met the restrictive requirements. To maintain consistency, changes made due

to quantization requirements were also applied to the non-quantized model and similarly to hardware limitations. A comparison was made among the three reduced models. The model that best performed was the non-quantized ECAPA model trained with knowledge distillation. The models were trained with two-second and one-second audio segments from the VoxCeleb1 dataset, containing numerous celebrity audio clips. The results indicated significantly better learning for two-second audio compared to one-second audio. Still, the model was trained on a one-second audio segment due to hardware limitations, as it cannot process more than one second of audio at a time due to memory constraints. The feature we used was a Mel-spectrogram computed with an FFT size of 1024 samples, an overlap of 512 samples, and 80 frequency bins. After training, the model was saved in ONNX format and converted to C language files using STM32 X-Cube-AI software. The application, programmed on the evaluation board, samples one second of audio, applies a high-pass filter, computes the Mel-spectrogram, performs a log function, normalization, and feeds the result into the network to obtain a 152-dimensional embedding vector. This process on the microcontroller takes 937 milliseconds, less than the 1024 milliseconds needed for one second of audio sampling (not 1000 milliseconds due to chip limitations). The current consumption of the microcontroller is 30 milliamperes, confirming it operates within the low-power range. Two male and female speakers from the test dataset were randomly chosen, and their average voice signatures were computed and stored on the microcontroller. The application works in real-time and compares the embedding vector of one second of audio captured from the microphone on the board after it passes the network with the two enrolled speakers' voice signatures. A specific LED is illuminated if the cosine similarity score exceeds a predefined threshold, which indicates verification.