### **Technical document: model.py**

*Team: AI Atlantique*

### **1. Introduction**

This model is designed to predict Head-Related Transfer Functions (HRTF) from images of the human pinna.

**2. Model Structure**

The model consists of two main components:

* **PinnaEncoder**: Extracts features from pinna images.
* **HRTFGenerator**: Uses these extracted features to predict the HRTFs.

#### **2.1 PinnaEncoder**

The encoder is a convolutional neural network (CNN) with the following layers:

* **Conv2d and BatchNorm**: Three convolutional layers with batch normalization, followed by ReLU activation, for extracting local features from images.
* **MaxPool2d**: Each convolutional block is followed by pooling to reduce the image size.
* **Dropout**: Dropout layers are applied after each convolutional block to prevent overfitting.

#### **2.2 HRTFGenerator**

The HRTF generator is a fully connected (FC) neural network architecture following the encoder:

* **PinnaEncoder**: Outputs from the encoder are processed by FC layers to produce HRTF predictions.
* **FC1 and FC2**: These layers process the extracted features and transform them into a dimension compatible with HRTF prediction.
* **FC3**: The final layer generates an output of shape (num\_angles, 2, num\_freq\_bins) representing the HRTFs for each angle and frequency.
* **Dropout**: Dropout layers are added to reduce overfitting risks.

**3. Training**

#### **3.1 HRTFTrainer**

The HRTFTrainer handles the training and validation processes:

* **Loss Function**: Mean Squared Error (MSE) measures the difference between predicted and real HRTFs.
* **Optimizer**: The Adam optimizer, with a learning rate of 0.001, adjusts model weights.
* **Memory Management**: Explicit calls to gc.collect() and torch.cuda.empty\_cache() free GPU memory after processing each batch.

#### **3.2 Training and Validation**

* **train\_epoch**: Training for a single epoch is performed in mini-batches with gradient accumulation to minimize memory usage.
* **validate**: Validation is conducted on a separate dataset to evaluate the model's performance after each epoch.

#### **3.3 Best Model Saving**

The model is saved after each epoch if validation loss improves, ensuring the best model is retained.

### **4. Model Workflow**

Here’s how the model predicts HRTFs from pinna images:

1. **Image Loading**: Left and right ear images are loaded into memory.
2. **Feature Extraction**: Each image is passed through the PinnaEncoder to extract spatial features.
3. **HRTF Prediction**: Extracted features are processed by the HRTFGenerator to predict HRTF values for various angles and frequencies.
4. **Loss Calculation**: The loss between predicted and actual HRTFs is computed during training and validation.
5. **Weight Adjustment and Saving**: Model weights are updated based on loss, and the best-performing model is saved.

### **5. Deployment**

The model is ready for deployment after training. Here's how to use it:

* **Training**: We use the train\_model() function to train the model on our dataset.
* **Saving:** We save the model under the name “best\_model.ph”
* **Prediction**: After saving, the model can generate HRTFs from pinna images by:

!python inference.py -l ./data/SONICOM\_TestData\_pics/P0002\_left\_0.png ./data/SONICOM\_TestData\_pics/P0002\_left\_1.png -r ./data/SONICOM\_TestData\_pics/P0002\_right\_0.png ./data/SONICOM\_TestData\_pics/P0002\_right\_1.png -o ./data/output/prediction.sofa