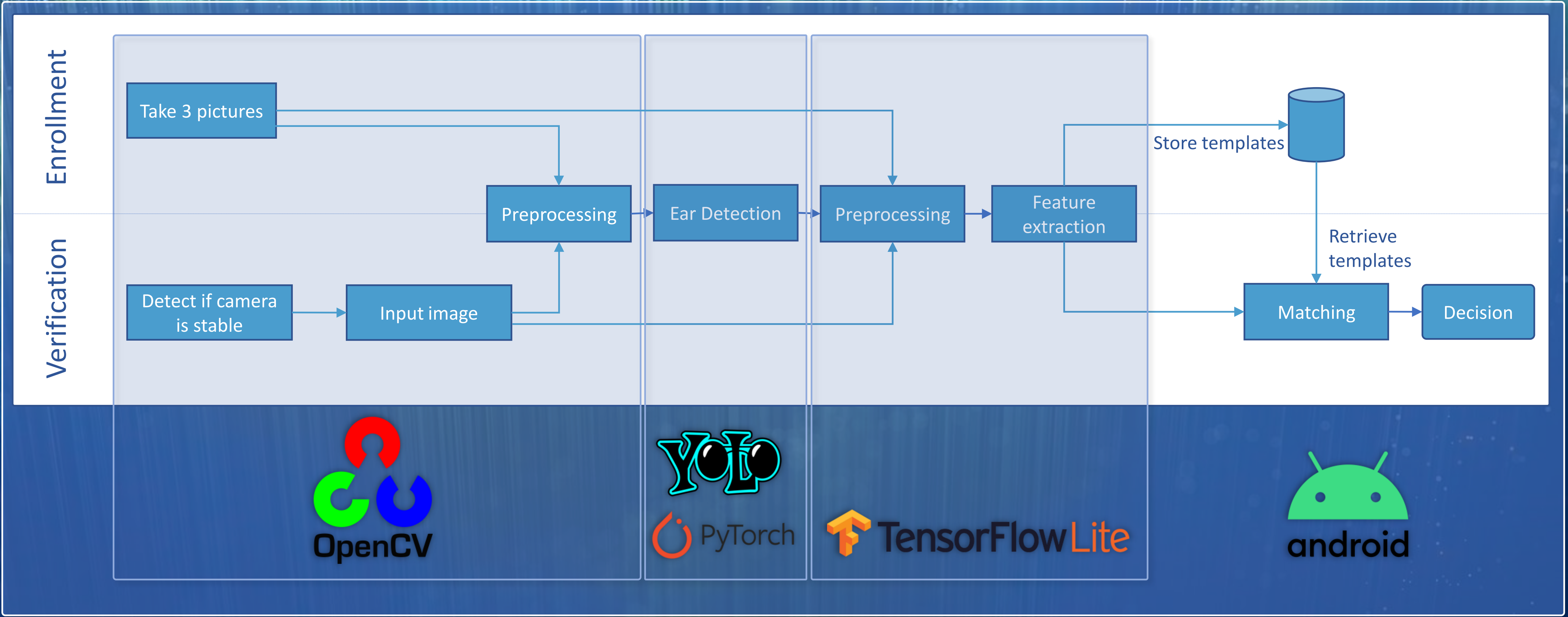
Ear Verification

With pre-trained neural networks

System Architecture



Models

Model for Ear Detection

- With the YOLOv5 Deep Neural Network, built with PyTorch
  - Many structures with pre-trained weights available
    - We used Yolo Nano for better performance
  - Trained for 60 epochs
    - 1-30: Full dataset. Problems with bias
    - 31-60: Random cropping around the ear. Working properly
- Dataset: UBEAR
  - Face profiles in real world conditions
  - Noise factors: occlusion, illumination, hair, earrings
- Input pre-processing
  - Transform the frame to grayscale
  - Resize to YOLO’s input format

An image from the UBEAR dataset

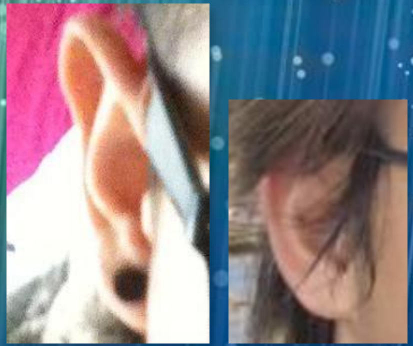
Comparison between YOLO models

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.4	46.0	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.2	56.0	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.2	63.9	224	8.2	1.7	21.2	49.0
YOLOv5l	640	48.8	67.2	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7
YOLOv5n6	1280	34.0	50.7	153	8.1	2.1	3.2	4.6
YOLOv5s6	1280	44.5	63.0	385	8.2	3.6	12.6	16.8
YOLOv5m6	1280	51.0	69.0	887	11.1	6.8	35.7	50.0
YOLOv5l6	1280	53.6	71.6	1784	15.8	10.5	76.7	111.4
YOLOv5x6	1280	54.7	**72.4**	3136	26.2	19.4	140.7	209.8
+ TTA	1536	**55.4**	72.3	-	-	-	-	-

Model for Feature Extraction

- Based on MobileNet, built with TensorFlow
  - Lightweight model, pre-trained on the ImageNet dataset
    - Around 4M parameters
  - Trained on EarVN and part of AMI
    - The images of AMI were cropped using the Yolo detection model
  - Previous models were trained on AWE and EarVN alone
    - Results were not satisfying
- Approach based on this paper
  - *Training Convolutional Neural Networks with Limited Training Data for Ear Recognition in the Wild*
- Input pre-processing:
  - Values are normalized to [-1,1]
  - Images are scaled down to 224x224
- Output format: vector of 1280 features

Images from the AWE dataset



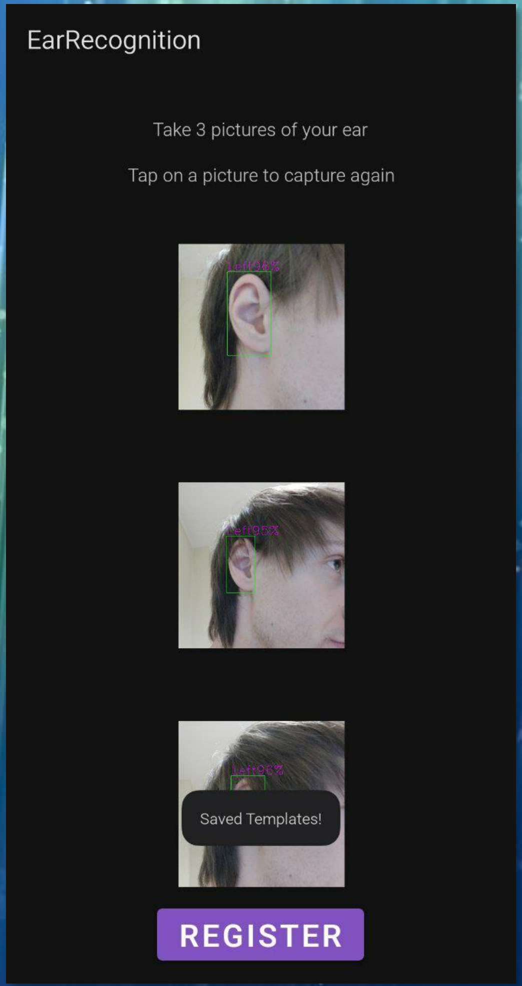
Images from the EARVN dataset

Cropped Images from the AMI dataset

Enrollment

Enrollment steps

- The app asks the user to take three pictures of the ear
- The user can tap on an image to re-capture it
- When the user taps Save, the app extract the features of the ear and completes the enrollment
- Steps of the features' extraction:
  - Localize and segment the ear with Yolo and OpenCV
  - Use the TensorFlow model to extract the features
  - Store them as a vector in the app



Screenshot of the enrollment procedure

Verification

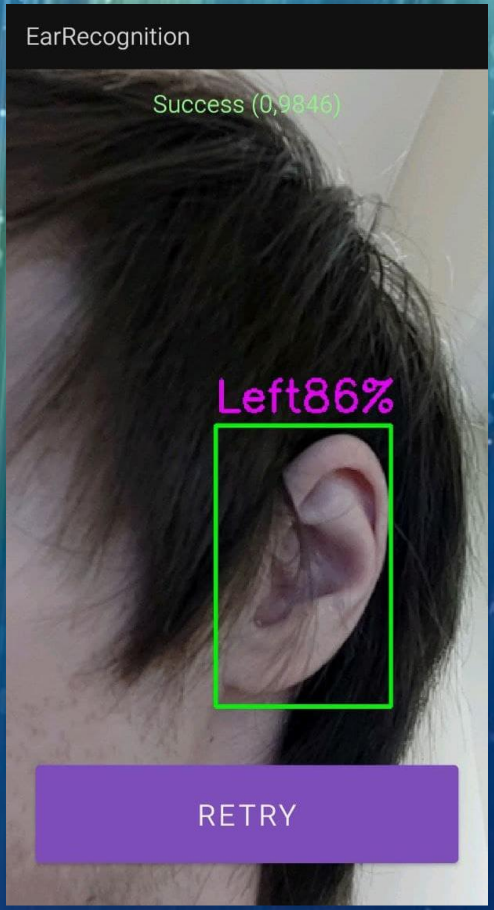
Motion Detection

- Avoid triggering the neural network at each frame
- Given two consecutive frames, compute their difference
  - The common part is black
  - The different part is white
- If the white area is below a certain threshold, the camera is stable
  - The app can now trigger Yolo to perform an object detection
- Advantages
  - Increased framerate
  - Avoid useless processing of blurred images

An example of difference of frames

Verification and matching

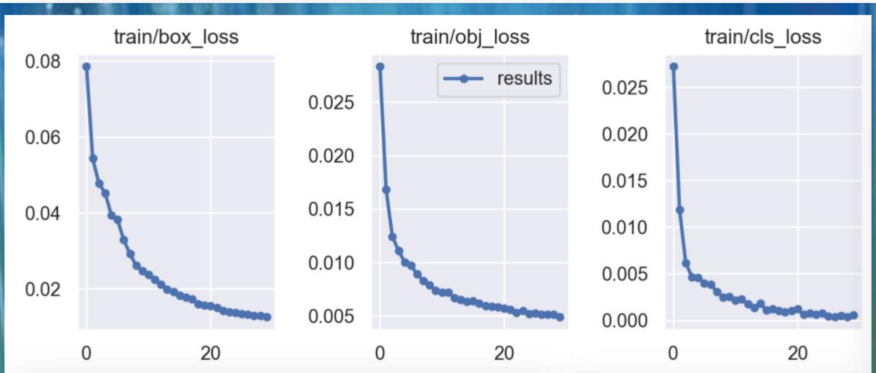
- When the camera is stable enough, Yolo is triggered and tries to recognize the ear
- If the ear is detected, the app passes the frame to the TensorFlow model for the extraction of the features and builds a probe vector with them
- The probe is compared with the templates in the gallery
  - Similarity measure: correlation
  - Match: the phone emits a vibration feedback
  - No match: the phone vibrates for a longer time
- No input needed from the user!
  - Put the camera to frame the ear, don’t move it for a second, and wait for the response

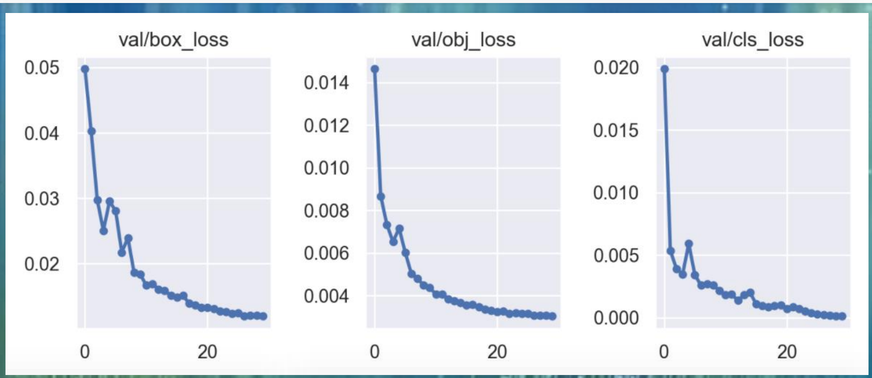


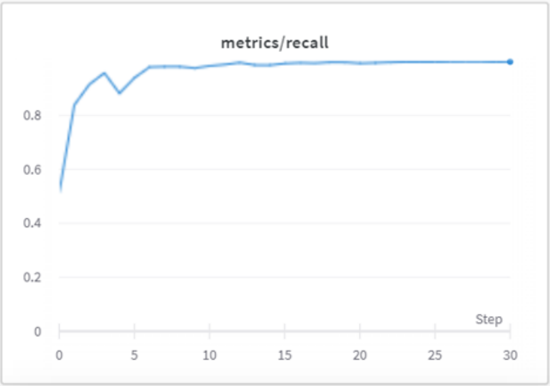
The app returning a positive verification

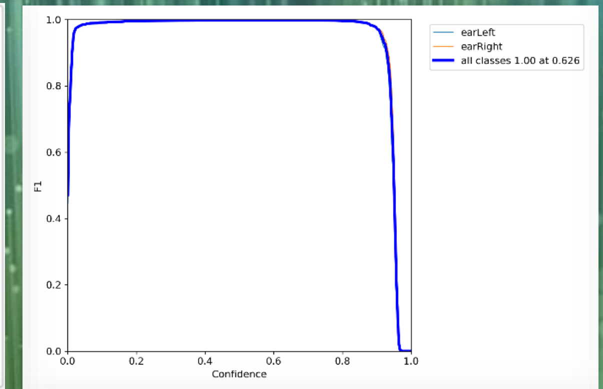
Performance evaluation

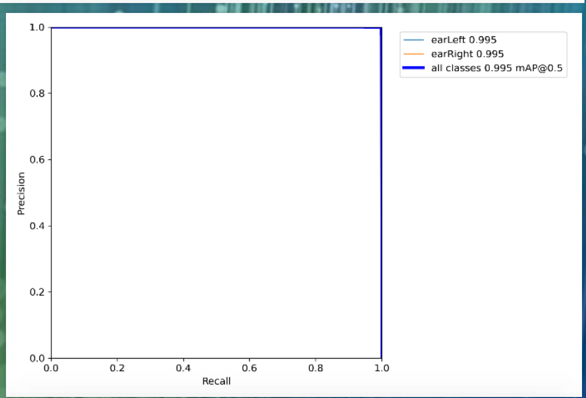
Evaluation for Yolo – epochs 1-30 – untouched dataset

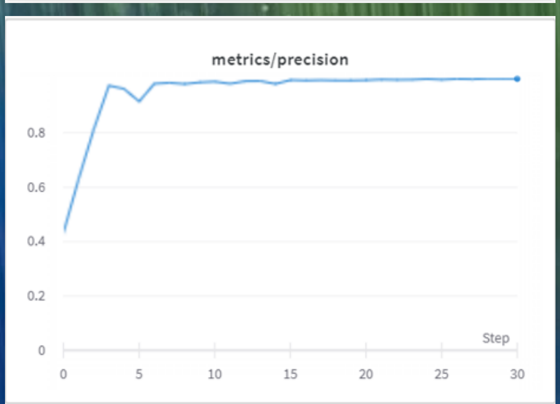


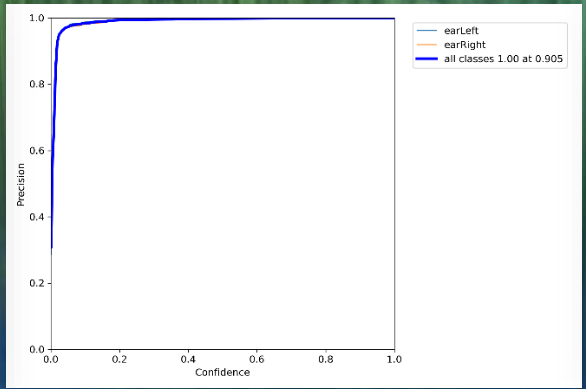


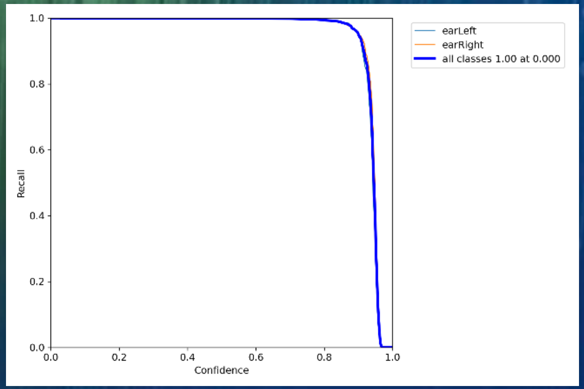




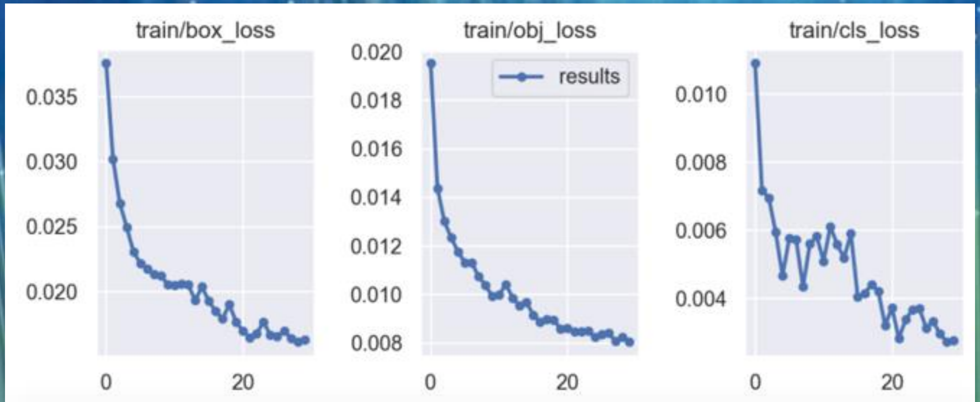


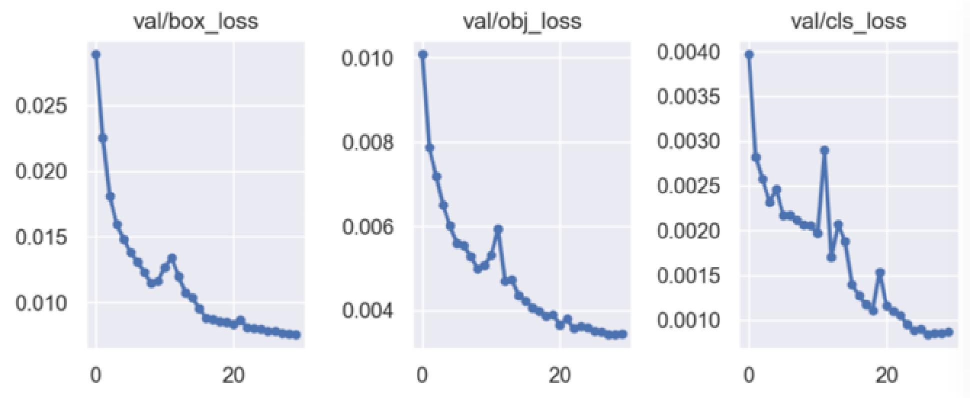


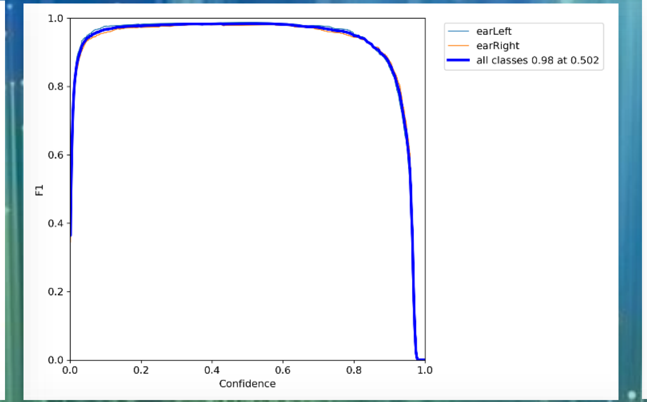


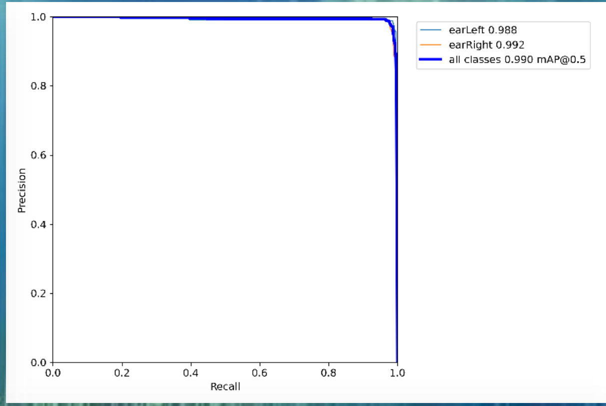


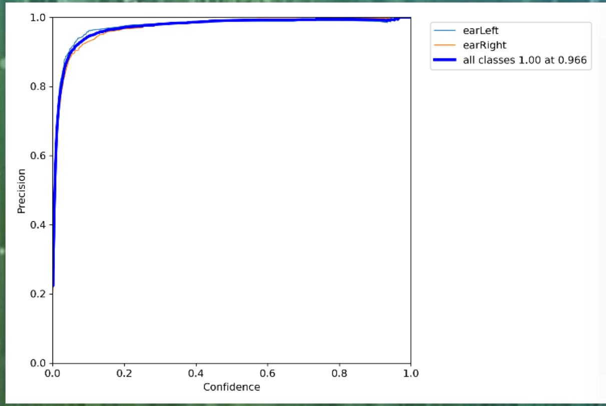
Evaluation for Yolo – epochs 31-60 – cropped dataset

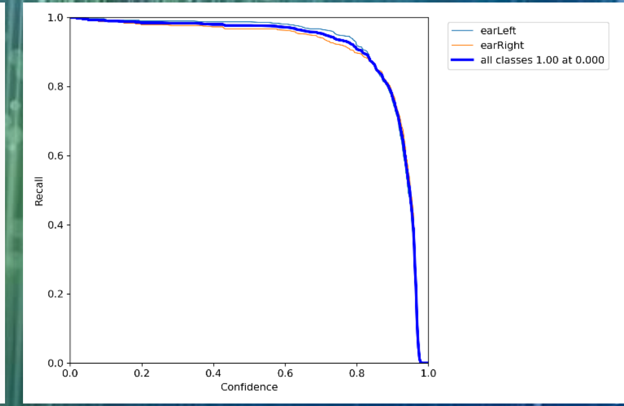


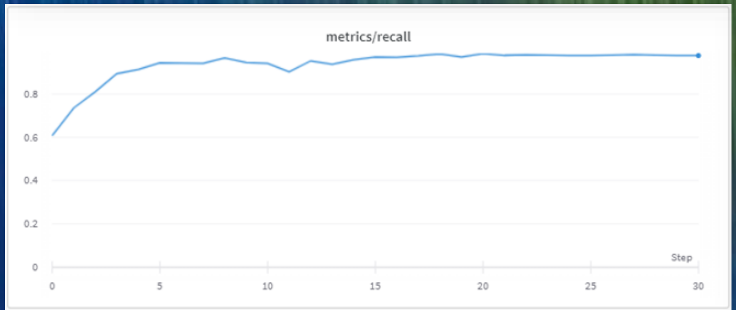


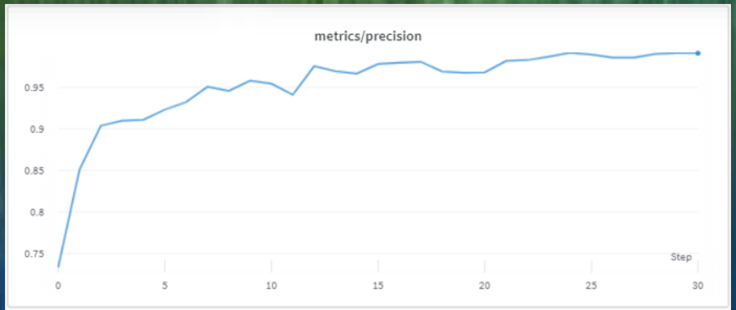




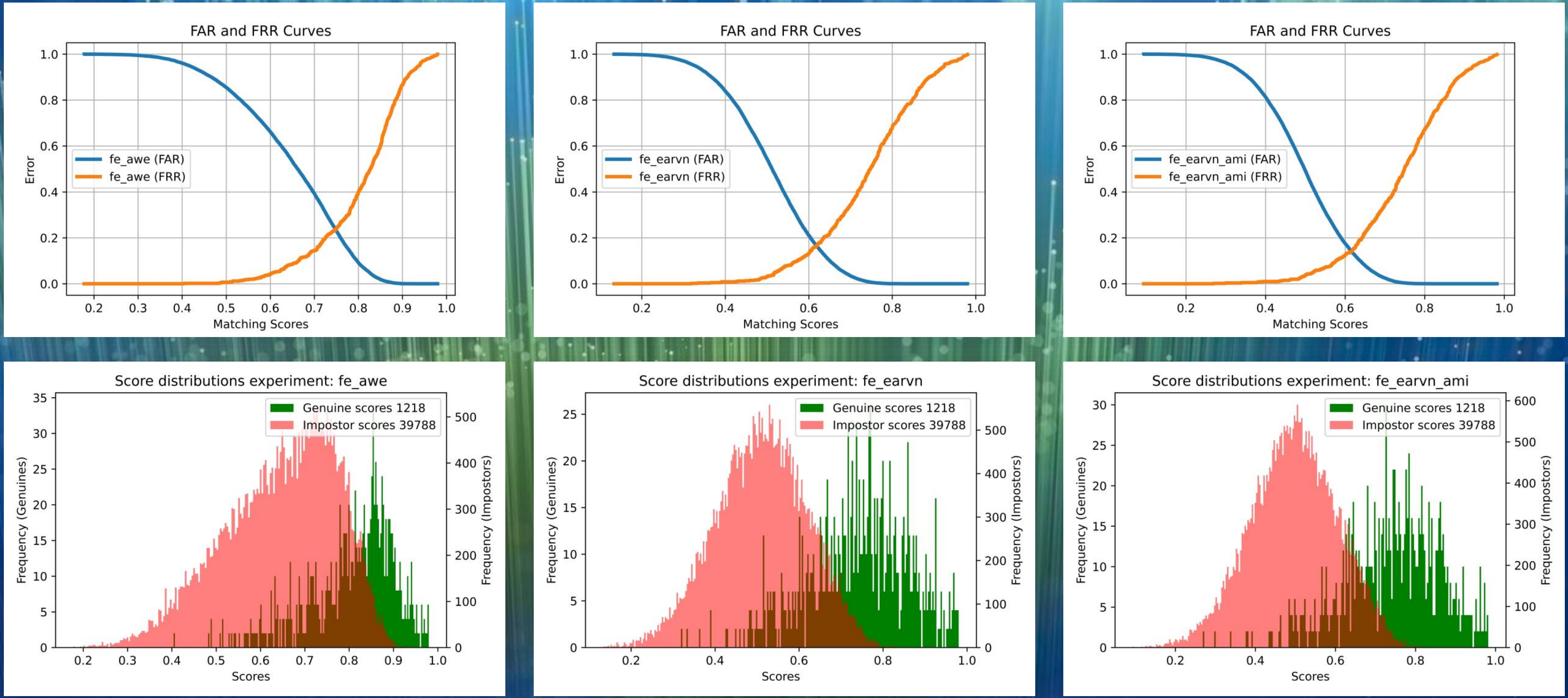




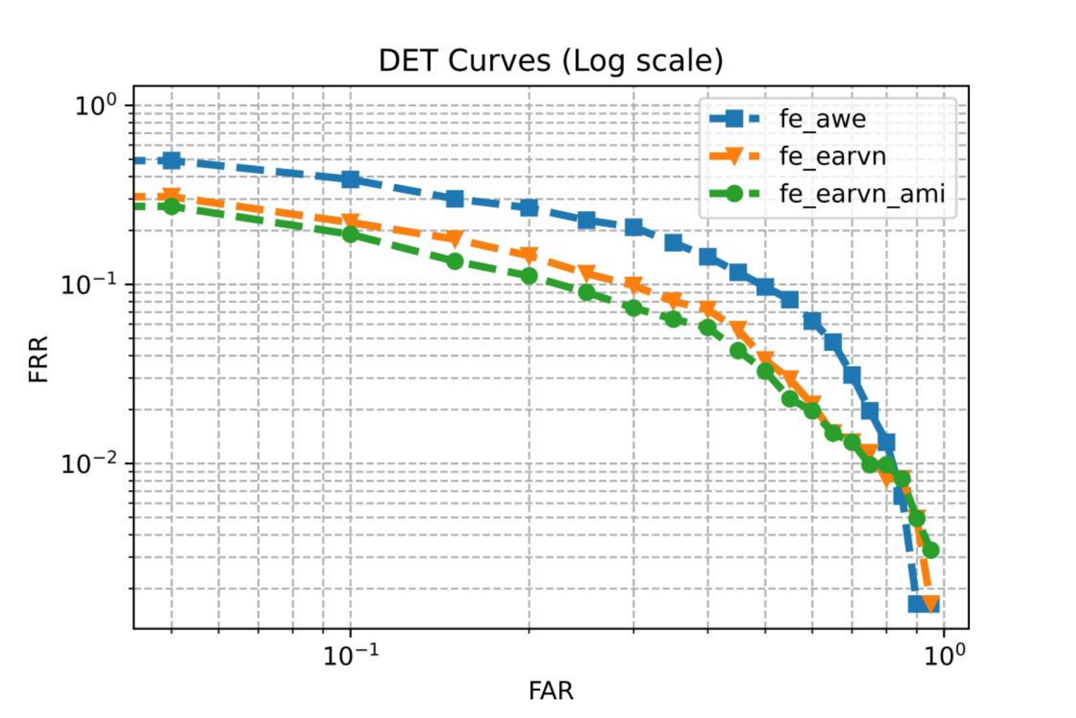


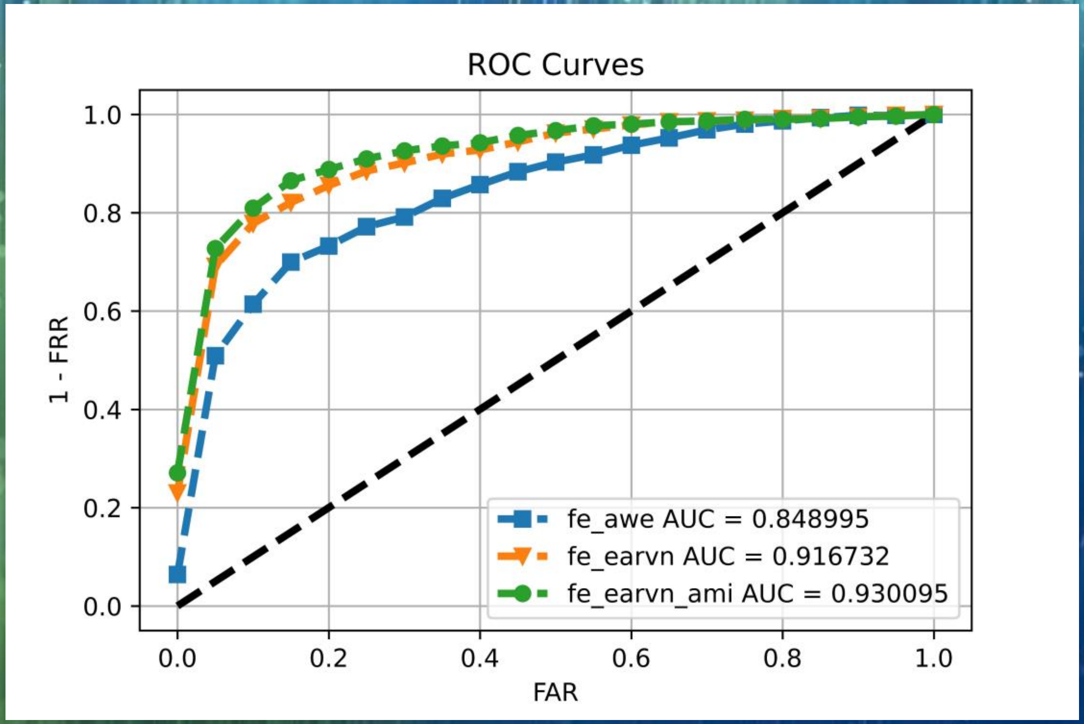


Evaluation on AMI 1/2

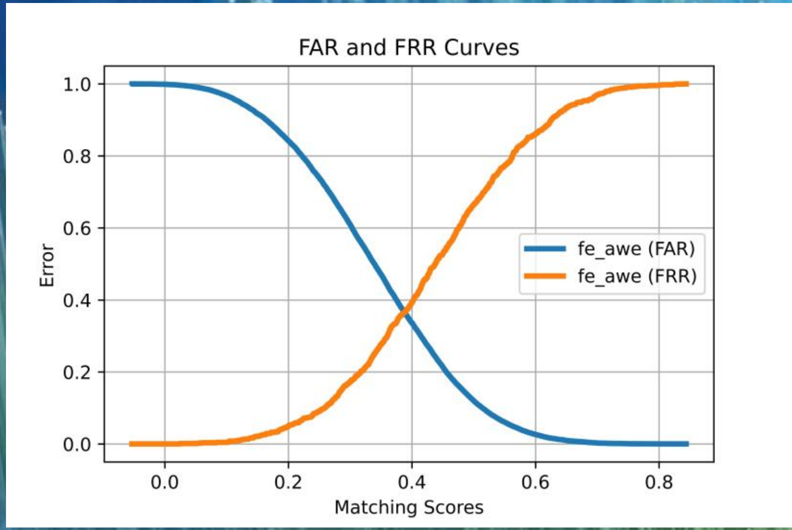


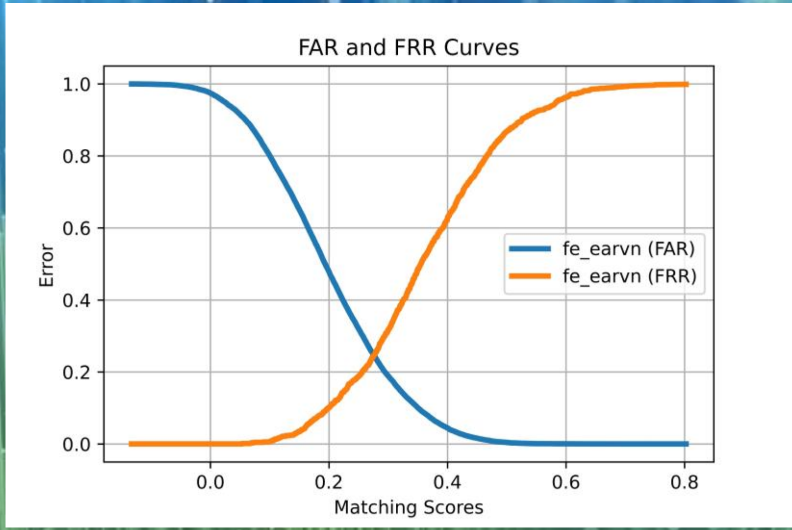
Evaluation on AMI 2/2

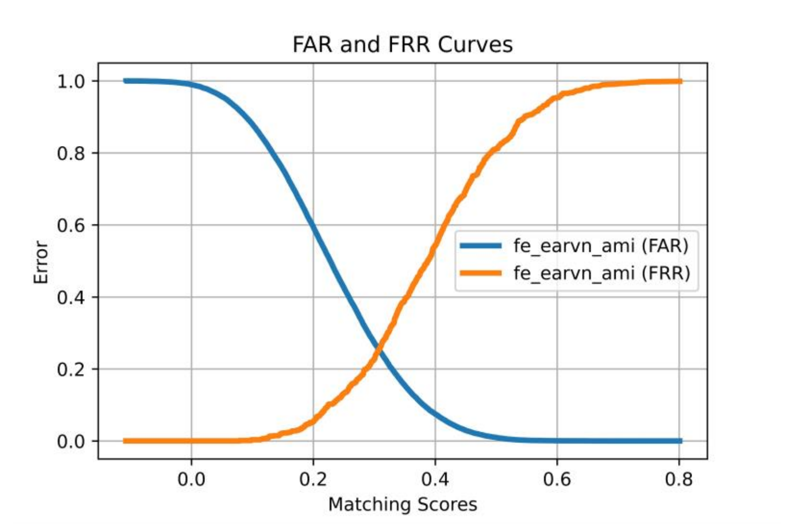


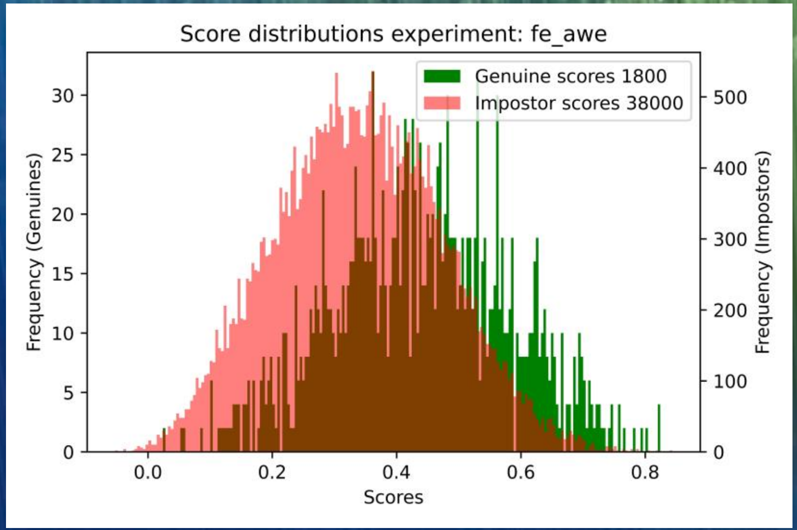


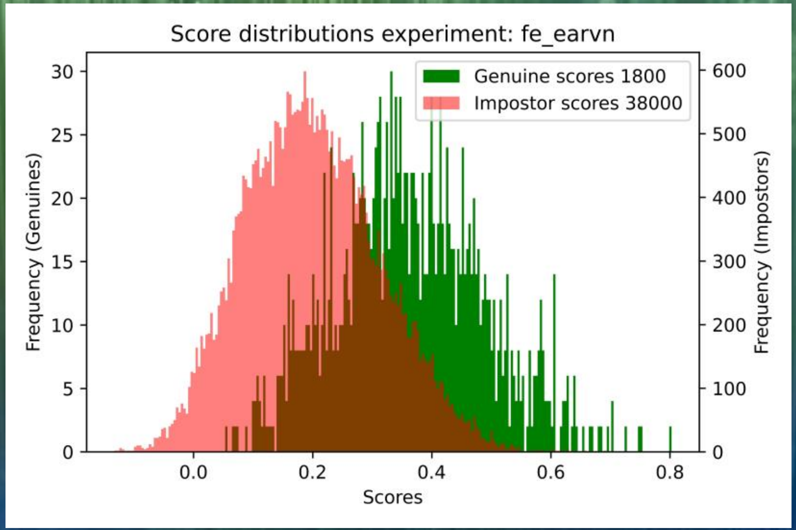
Evaluation on AWE 1/2

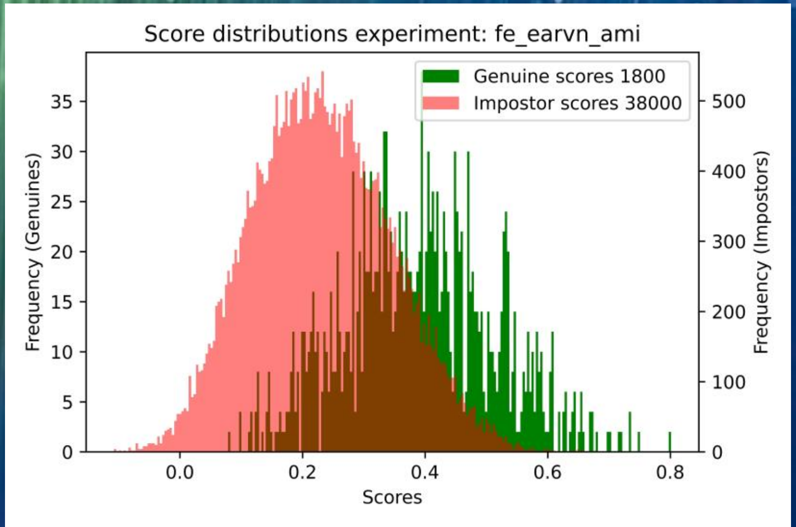




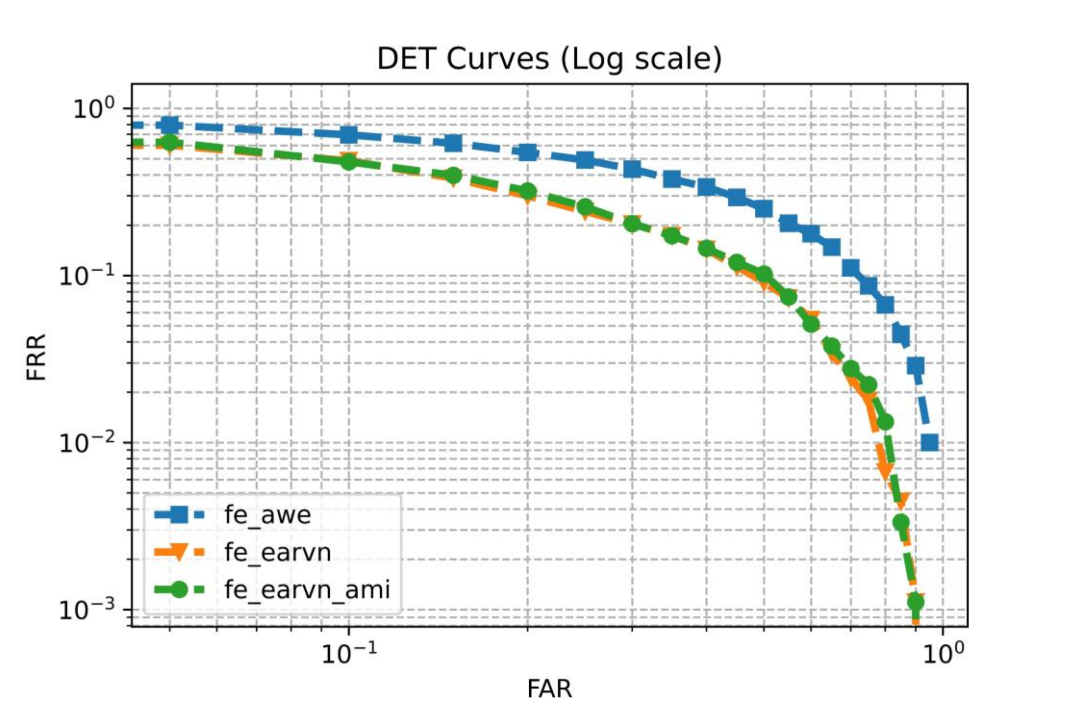


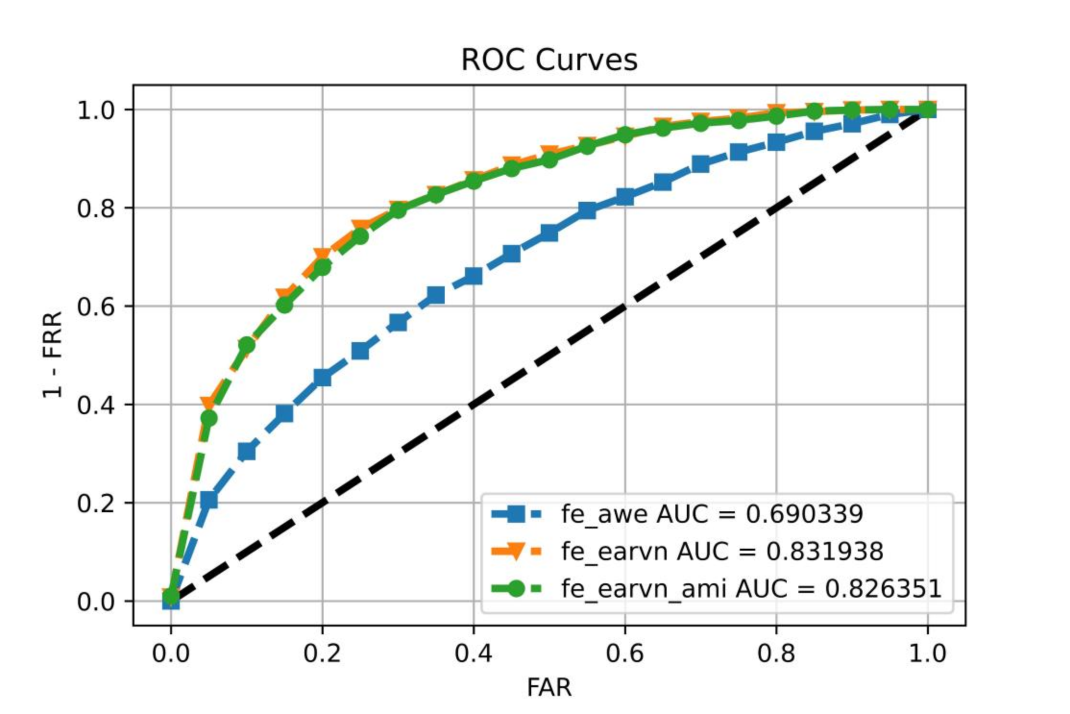






Evaluation on AWE 2/2





Thank you!