Ear Recognizer Android

Final project for the course of Biometric Systems - February 2021

Leonardo Emili
Alessio Luciani

The task: Ear Recognition

- Ear as a new biometrics
- Unique characteristics which humans are not used to distinguish
- It respects the fundamental biometric factors

The task: Challenges

- Small size makes acquisition process hard
- Same color of the skin tone
- Might be covered or present levels of occlusion

The AMI Ear database

- Collection of 700 2D images
- 100 subjects in the range of 19-65 years
- Images from controlled environment

The AMI Ear database

Image source: ctim.ulpgc.es/research_works/ami_ear_database/

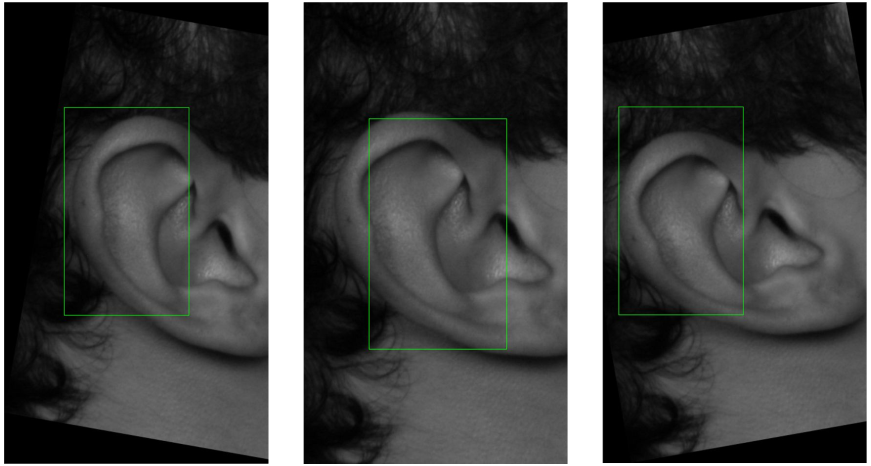
Our contributions

- Implement a full-fledged recognition pipeline from scratch
- Android porting to allow online recognition
- Benchmark our solution against another approach

The recognition pipeline: localization

- Locate the ear area within the original image
- Haar Feature-based Cascade Classifiers
- Crop the image according to the ROI

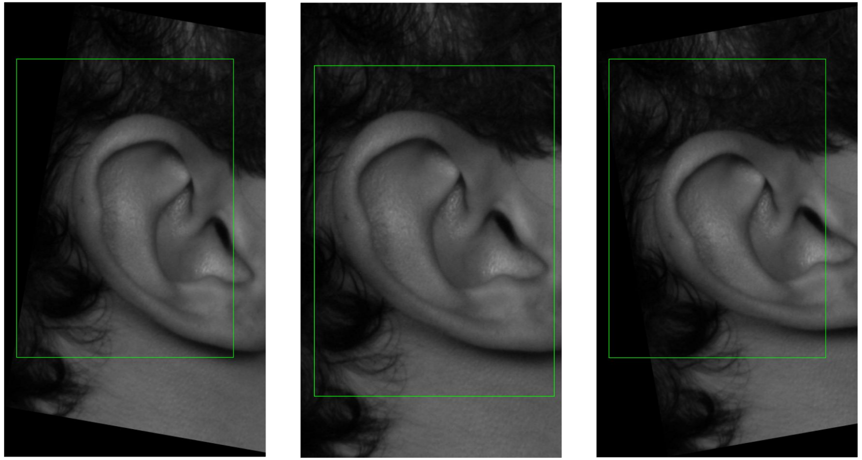
The recognition pipeline: localization



The recognition pipeline: localization + padding

- Detected ROI is not good enough
- Apply a window of padding of size *k* around it
- Experiment different amounts of padding (e.g. *k*=80)

The recognition pipeline: localization + padding



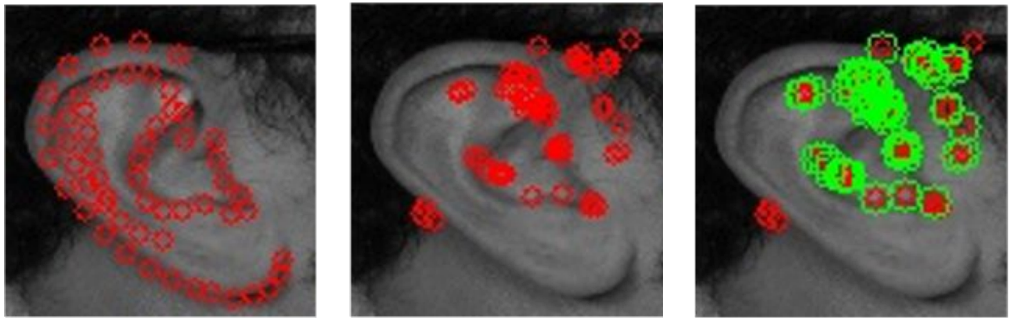
The recognition pipeline: landmark detection (1/3)

- Experiment with state-of-the-art CNN
- Minimal effort and good results
- We want to explore new ideas

The recognition pipeline: landmark detection (2/3)

- Keypoints extraction using the ORB algorithm (2011)
- Less accurate keypoints
- Remove outliers using standard deviation

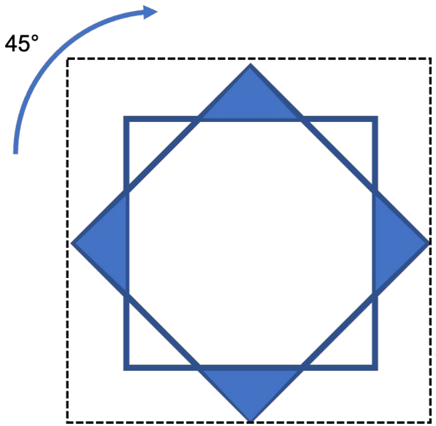
The recognition pipeline: landmark detection (3/3)

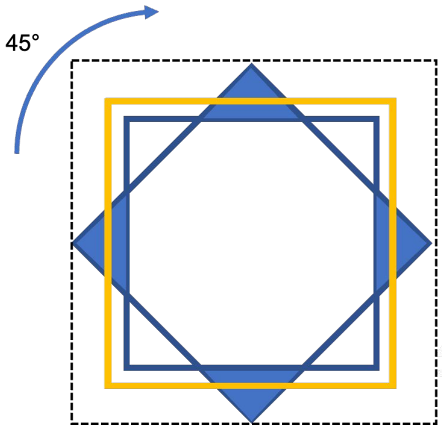


The recognition pipeline: ear alignment (1/4)

- Align images along the vertical axis
- Find the correct rotation angle
- Approximation of the ear axis using linear regression

The recognition pipeline: ear alignment (2/4)





The recognition pipeline: ear alignment (3/4)

- Size after rotation is proportional to the rotation angle
- Interpolate pixels in the corners
- Zoom in to normalize the size and reduce noise

The recognition pipeline: ear alignment (4/4)

The recognition pipeline: template matching

- Brute force matcher
- Compute similarities using the test ratio (Lowe et al., 2004)
- Accept a probe according to the acceptance threshold

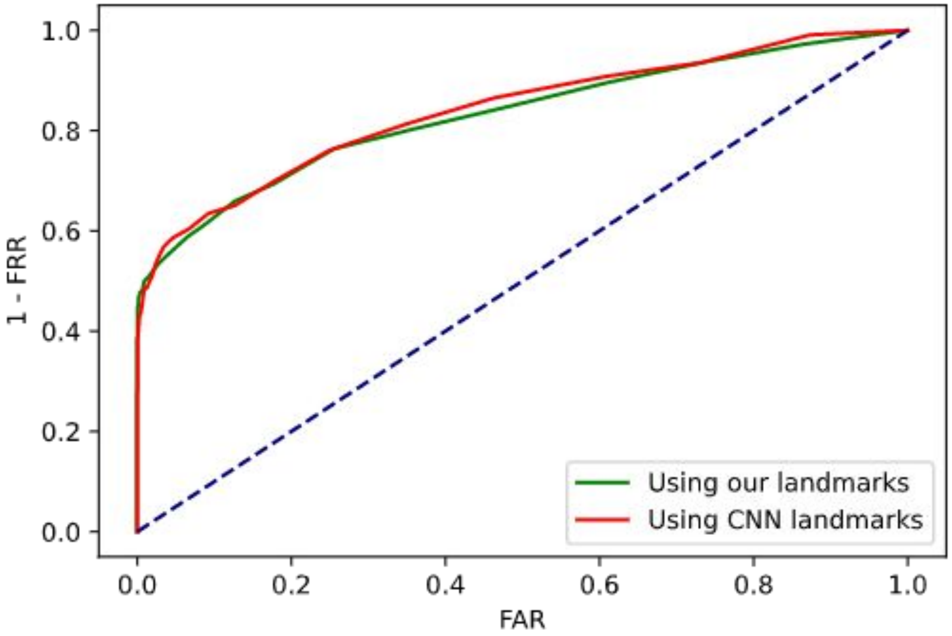
Evaluation: all-vs-all verification

- Every image in the dataset considered as a possible probe
- Testing probes against the claimed identity to compute GAR
- Testing probes against the different identities to compute FAR

Evaluation: ROC curve

- Comparing 1 - FRR to FAR as the acceptance threshold changes
- The FAR score is null for thresholds from 0 to 0.5
- The results using the CNN approach are only slightly better

Evaluation: ROC curve

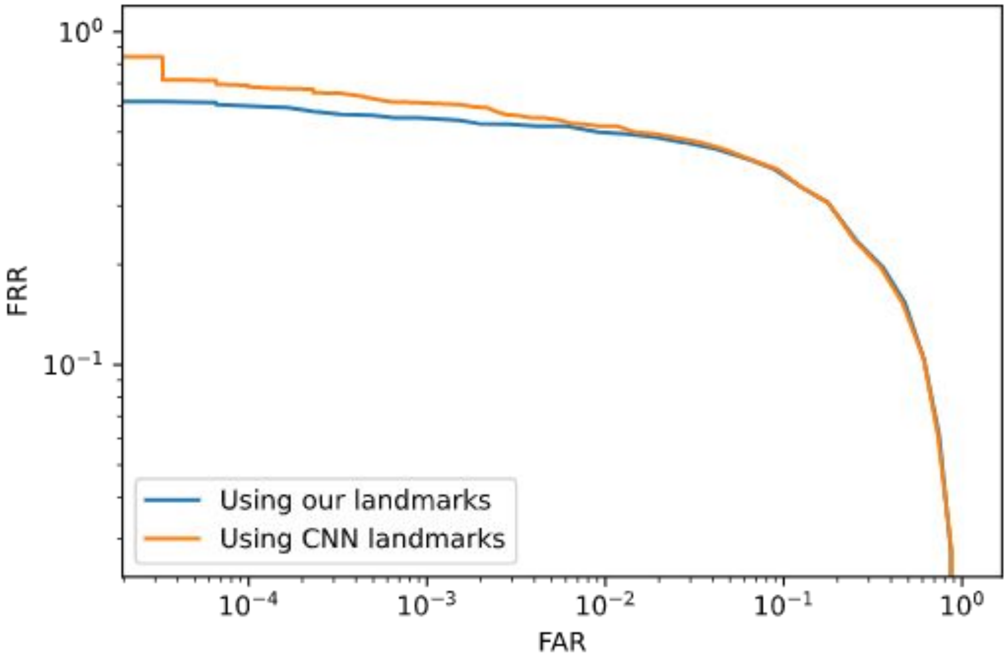


Evaluation: DET curve

- Comparing the probability of False Rejections with that of False Acceptances

- Confirmation of slight improvements with the CNN model

Evaluation: DET curve

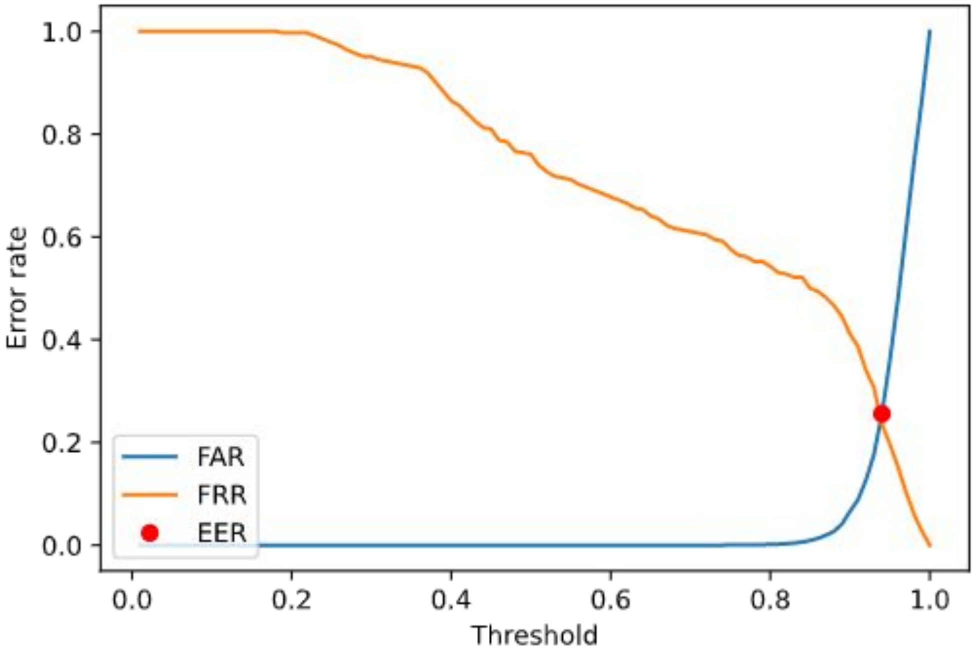


Evaluation: Equal Error Rate

- Understanding what is the threshold for which FAR equals FRR
- Finding a good compromise between accepting impostors and rejecting genuine attempts
- The EER is about 25% where the threshold equals 0.93

Evaluation: Equal Error Rate

- A threshold positioned at about 0.9 would allow admitting only 5% of the impostors and 50% of the genuine users



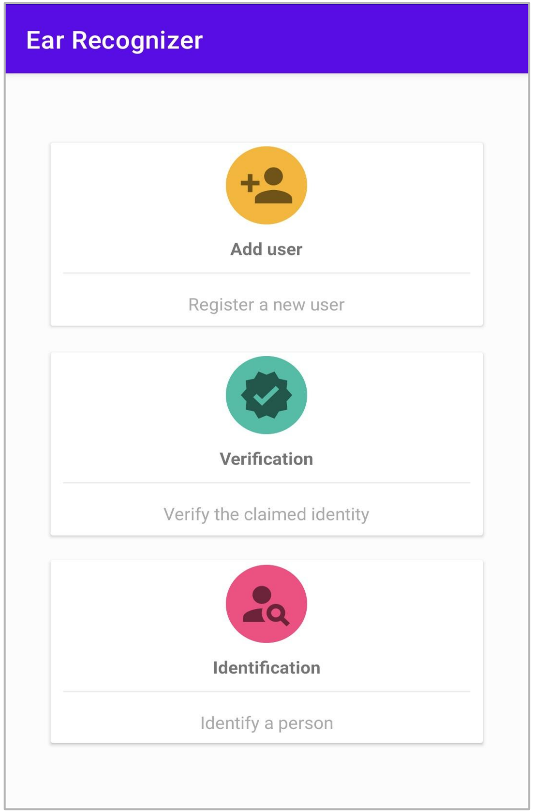
Android: making the system interactive

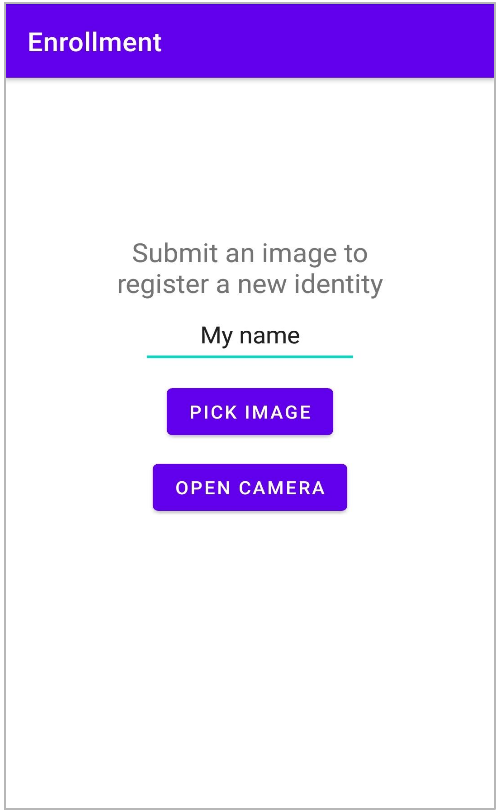
- Porting the C++ / OpenCV code into Android (Kotlin) / OpenCV
- Allowing on-device system enrollment, verification and identification
- Templates stored on the local Android storage

Android: interaction

- Capturing a new image from the camera or importing it from the device's gallery
- Enrollment outcome: the image can be valid or invalid for the system
- Verification / identification outcome: the user is correctly recognized, or classified as impostor

Android: live demo





Thanks for the attention