People identification (SUR class project)

People identification based on voice recordings and headshot images.

Featuring: Gaussian Mixture Model (GMM) & ResNet-18.

Development setup

To install the required dependencies, run the following command:

```
make venv
```

Then, download the training dataset from URL and extract it into `data/` directory.

If you want to use audio augmentation, download Room Impulse Response and Noise Database and extract it into `RIRS_NOISES/` directory.

Usage

To train and evaluate Gaussian Mixture Model (GMM) for speaker recognition, run the following commands:

```
# Train your model and save it into gmm_model.npz file
# The parameters for training can be tuned from within the script
python audio/gmm.py train gmm_model.npz

# Use your model to classify data on final dataset located in evaluation_dataset/eval
# WARNING: never load models from untrusted sources, loading of the model is not secure
# against erroneous or maliciously constructed data (uses pickle.load under the hood)
python audio/gmm.py classify models/gmm_audio_24_27.npz evaluation_dataset/eval \
  > results/gmm_audio_24_27.txt
```

To train and evaluate the ResNet18 model for the person recognition, run the following commands:

```
# Train the model
python images_resnet/train_resnet.py --dataset /path/to/the/dataset/dir/

# Use the model to classify data
python images_resnet/eval_resnet.py --model /path/to/models/model_checkpoint.pt \
  --dataset /path/to/the/dataset/dir/

# Plot statistics
python3 images_resnet/plotting.py
```

To generate the final report, install `pandoc` and `latex` and execute

```
pandoc README.md --metadata-file doc/pandoc-metadata.yml -s -o \
  doc/dokumentace.pdf
```

Audio model

For audio, Gaussian Mixture Model (GMM) was used. The model was trained on the dataset provided by the course. The data was augmented by changing speed, adding Gaussian noise, adding room impulse response, and adding different background noises.

Evaluation of the GMM models during training was performed using `audio/peekin.py` and `audio/plotting.py` helper scripts. During training, we were watching the sum of total log likelihoods for both target and non-target classes. For validation, we were looking at the sum of differences from the expected probabilities on all data samples.

See the following figure for performance of the GMM models on test and validation datasets with respect to the number of components and iterations used. We used only clear data for evaluation (without any augmentation).

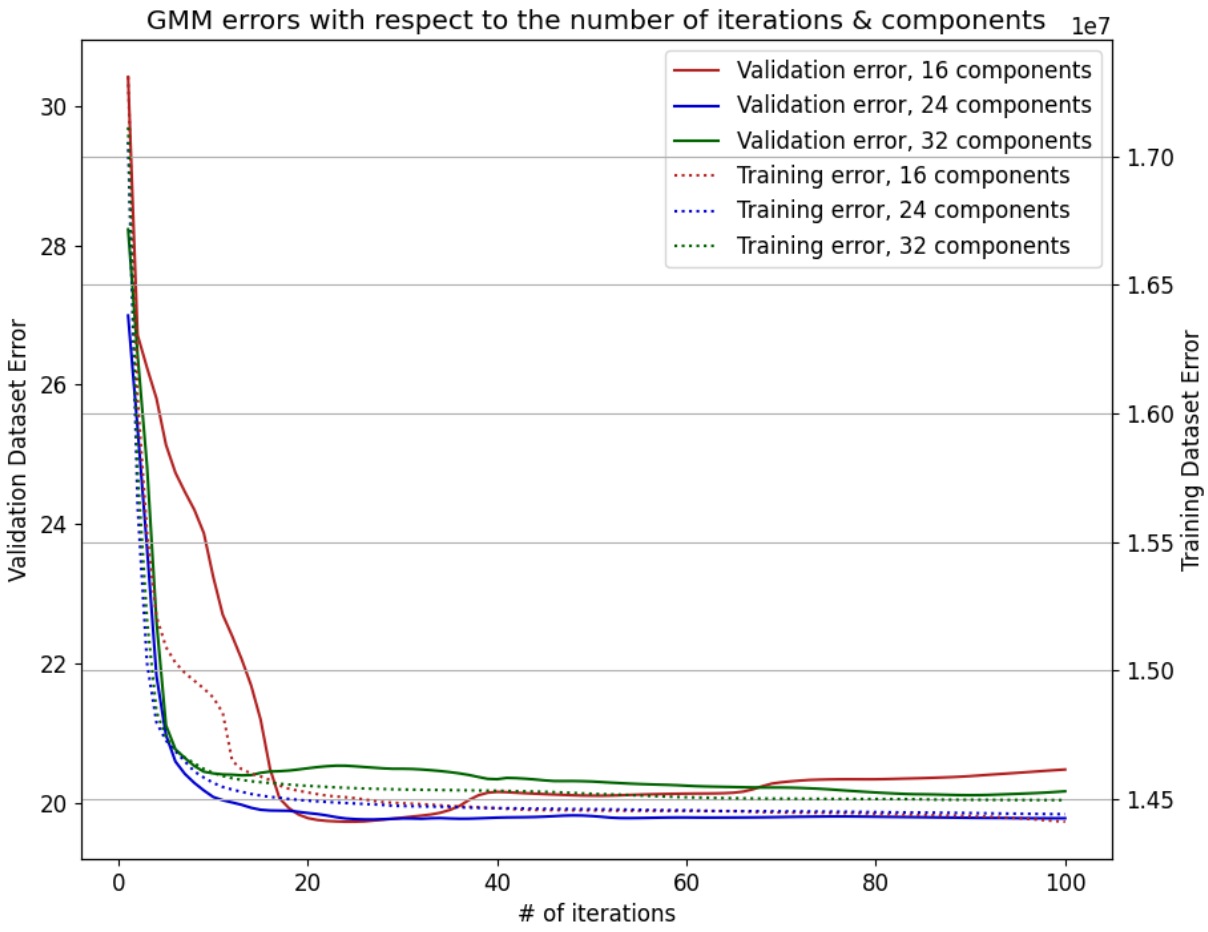

Figure 1: GMM performance

Note that the means of the Gaussian components are initialized stochastically from the training samples, so the results may vary between runs. Also, for target class, there is only 10 training samples, so when choosing more components, they may not be fully utilized. However, we have way more samples for non-target class, that's why we're okay with initializing both mixtures with more components than 10 (to simplify development).

For our runs, clearly the most performant and the best converging models are those with 24 components. The model we selected for further use is located at `models/gmm_audio_24_27.npz` (24 components, 27 iterations).

Image training evaluation

See the following figure for the performance of the ResNet18 model (train and validation loss and validation accuracy).

The model we selected for further use is located at `models/resnet_image_15_20.pt` (saved at 15 epochs of 20 epochs totally) chosen using the Early Stopping approach.

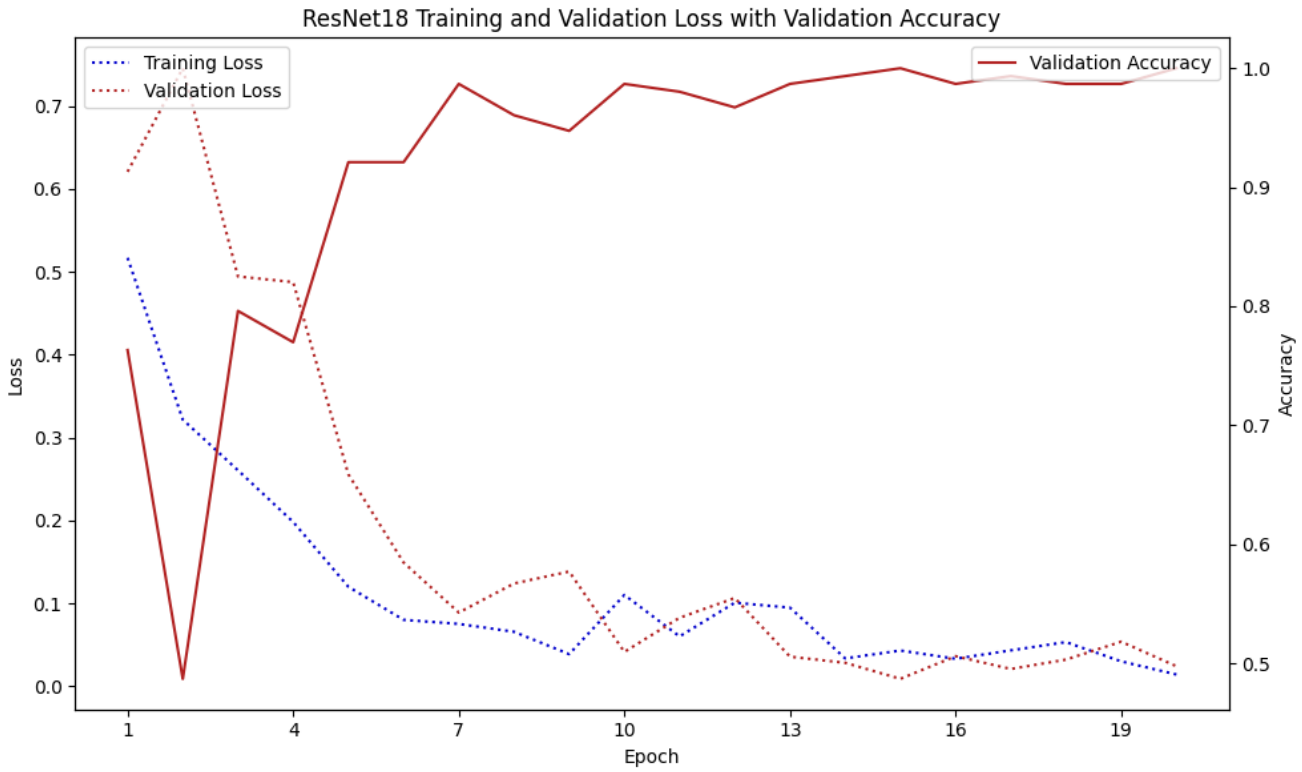

Figure 2: ResNet18 performance