BRNO UNIVERSITY OF TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

Voice and Image Recognition SUR Project

Karol Šugár (xsugark00)

4 May 2025

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

1. Dataset

The project was based on the dataset provided in the SUR_projekt2024-2025.zip archive. This archive contained the train and dev directories, which were used to train and evaluate models during development. The eval directory was available for final testing. The train and dev directories contained 31 subdirectories, labelled with numbers 1 to 31, which corresponded to individual persons (classes).

Each subdirectory in the train directory contained 6 samples for a given person in WAV format (voice) and 6 samples in PNG format (face). In the dev directory, there were 2 WAV and 2 PNG files for each person. In total, there were 186 training samples and 62 test samples available.

The file names had a structure such as f401_01_f12_i0_0.png, where the first part (e.g. f401) internally identified the person and the second part (e.g. 01) indicated the recording session number. It was observed that the internal identifier of the person (e.g. f427) did not correspond directly to the numerical name of the parent directory (e.g. file f427 was in directory 28). For the purposes of training classifiers, directory names were used as reference classes.

1.1 Environment

The solution is implemented in Python 3 using Jupyter Notebook. The project uses the following libraries:

NumPy and Matplotlib for data processing and visualisation, OpenCV (cv2) for image processing, scikit-learn for machine learning (GaussianMixture, SVC, GridSearchCV, preprocessing), scikit-image for HOG extraction, joblib for model storage, ikrlib for MFCC extraction

The project is structured for ease of use: models are trained, saved to a folder, and then loaded for evaluation. Thanks to this approach, all you need to do on other computers is run the configuration part with imports and download the provided model, without the need for retraining. To easily run the entire script, just press the compile button for the entire script and, if the test and train folders are available, the test folder (or possibly also the eval folder) will be trained and evaluated.

2. Models

When selecting models, we had to make sure that they were not already trained, as transfer learning was prohibited. The following approaches were chosen for the individual modalities:

Audio: Voice-based identification was performed using Gaussian mixture models (GMM) trained on Mel-frequency cepstral coefficients (MFCC).

Image: Face image-based identification used a Support Vector Machine (SVM) classifier trained on features extracted using the Histogram of Oriented Gradients (HOG) method.

2.1 Voice recognition (Audio)

Model: Gaussian mixture models (GMM) with MFCC features.

Library: sklearn.mixture.GaussianMixture from scikit-learn and ikrlib waw16hz2mfcc().

2.1.1 Feature extraction and preprocessing

We will use MFCC features for audio extraction. The wav16khz2mfcc function from the provided ikrlib library was used for extraction, which processed WAV files (we were told in the lecture that they are 16kHz).

After extracting MFCC features for each file, a silence filtering step followed. This step was performed directly on the MFCC features. Frames whose total energy (first MFCC coefficient) was below a specified threshold were considered silence or noise and were removed.

Subsequently, all remaining MFCC features from all training sets were normalized using StandardScaler from the scikit-learn library. Normalization was necessary for GMM models because they are sensitive to different ranges of input feature values and helps prevent the dominance of higher-energy features.

2.1.2 Model configuration and hyperparameters

The following key parameters were selected for the GMM models:

- Number of Gaussian components (n_components): 8
- Number of EM iterations (max_iter): 20
- Covariance matrix type (covariance_type): Full

The choice of covariance type was the result of experimentation. The originally tested model with a diagonal covariance matrix achieved an accuracy of around 58%, and further tuning of the hyperparameters did not seem to bring any improvement. However, testing with a full covariance matrix showed consistently better results (by about 5%) with random hyperparameters. We found the other best hyperparameters through grid search.

2.1.3 Training and classification

A separate GMM model was trained for each of the 31 classes using the GaussianMixture implementation. All pre-processed MFCC vectors extracted from the training recordings of a person were used as training data for the i-th class model. The model parameters (mixture weights, mean values, and covariance matrices) were estimated using the iterative Expectation-Maximization (EM) algorithm.

The identification of an unknown recording was performed as follows: MFCC vectors were extracted, filtered and normalised from the recording using the same procedure as in training. Subsequently, the average logarithmic likelihood of these MFCC vectors was calculated for each of the 31 trained GMM models using the score method. The recording was assigned to the class (person) whose GMM model provided the highest average log-likelihood. These log-likelihoods for each class were also used as the required score for the output file.

Final accuracy: 72.58% (on the dev/test set)

3.2 Image recognition

Model: Support Vector Machine (SVM) with HOG.

Library: sklearn.svm.SVC, sklearn.model_selection.GridSearchCV, skimage.feature.hog.

3.2.1 Preprocessing and feature extraction

Image data (PNG files) were loaded using the OpenCV library and converted to greyscale, as HOG works with intensity gradients. All images were reduced to 64×64 pixels for consistent vector length, and global histogram equalisation (cv2.equalizeHist) was applied to normalise contrast.

HOG features were extracted with the following parameters:

- orientations=9
- pixels_per_cell=(8, 8)
- cells_per_block=(2, 2)
- block_norm='L2-Hys'

The resulting HOG vectors were normalised using StandardScaler.

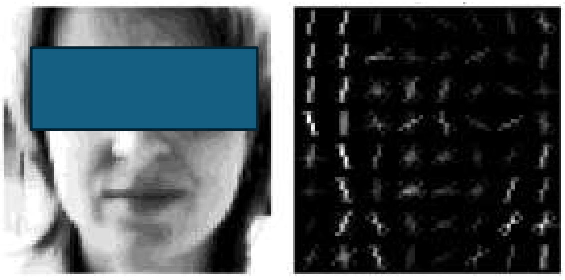
Figure 1 Example of IMG to HOG features

3.2.2 Model configuration and hyperparameters

SVC (Support Vector Classification) from the scikit-learn library was used as the classifier. SVM is a discriminative classifier that searches for the optimal hyperplane separating data points of different classes with the maximum possible margin.

The SVM hyperparameters were optimised using GridSearchCV with 5-fold cross-validation. The parameters C, kernel and gamma were searched for.

Best configuration: C=0.1, kernel='linear', gamma='scale'.

3.2.3 Training and classification

Training was performed on normalized HOG features with the best hyperparameters. The classifier was part of a pipeline that ensured the application of StandardScaler before the actual classification.

To ensure the output of logarithmic probabilities, the probability=True option was set during SVC initialisation. This allowed us to use the predict_log_proba method during training, which returns estimates of logarithmic probabilities for each class. The hard decision was obtained using the standard predict method.

Final accuracy: 72.58% (on the dev set)

3.2.4 Image errors

First, a model based on a simple neural network (multi-layer perceptron) was tested for image recognition. However, due to the relatively small amount of training data (186 images for 31 classes), this model did not achieve good results. Even with different settings for architecture, hyperparameters, and data augmentations, the neural network achieved a maximum accuracy of only around 30%. Neural networks are known for their need for large amounts of data to generalise, so we then moved on to the SVM+HOG model.

4. Results

4.1 Report

Output from classification_report() – scikit-learn function

Task	Model	Accuracy	Precision	Recall	F1-Score
Voice (AUDIO)	GMM	72.58	0.74	0.73	0.69
Image	SVM+HOG	72.58%	0.66	0.73	0.67

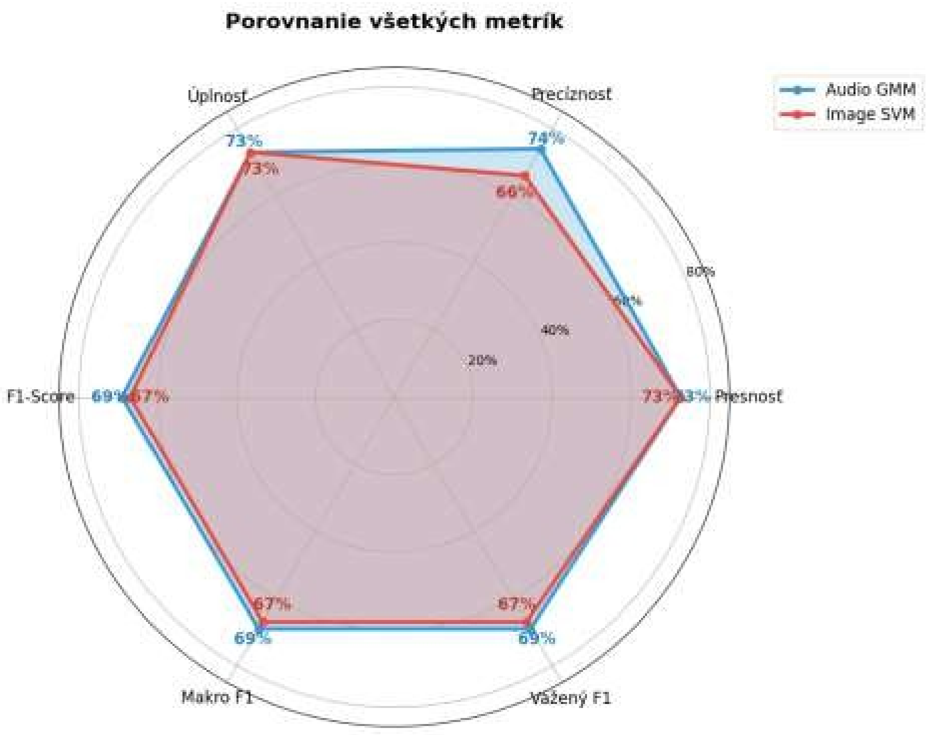

Figure 2 Metrics Comparison

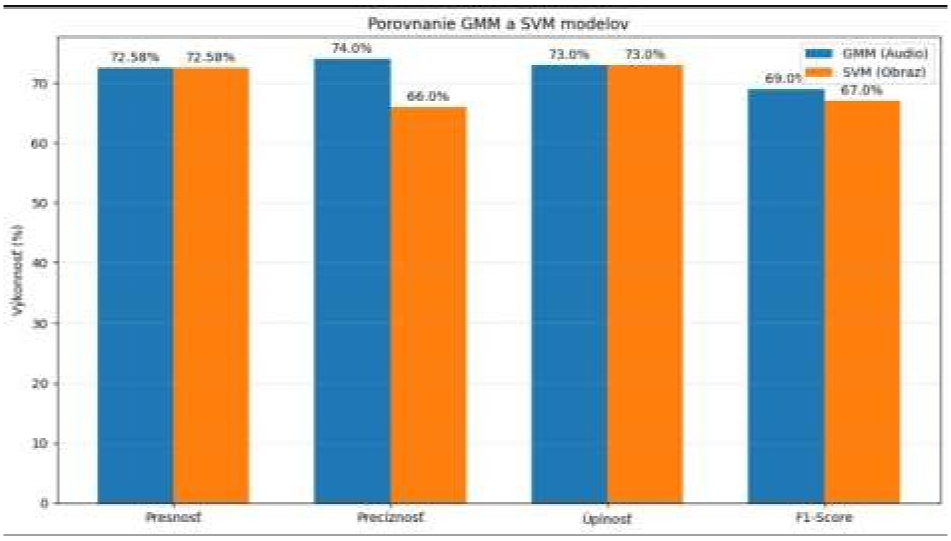

Figure 1 Metrics Comparison

4.2 Performance Comparison by Class

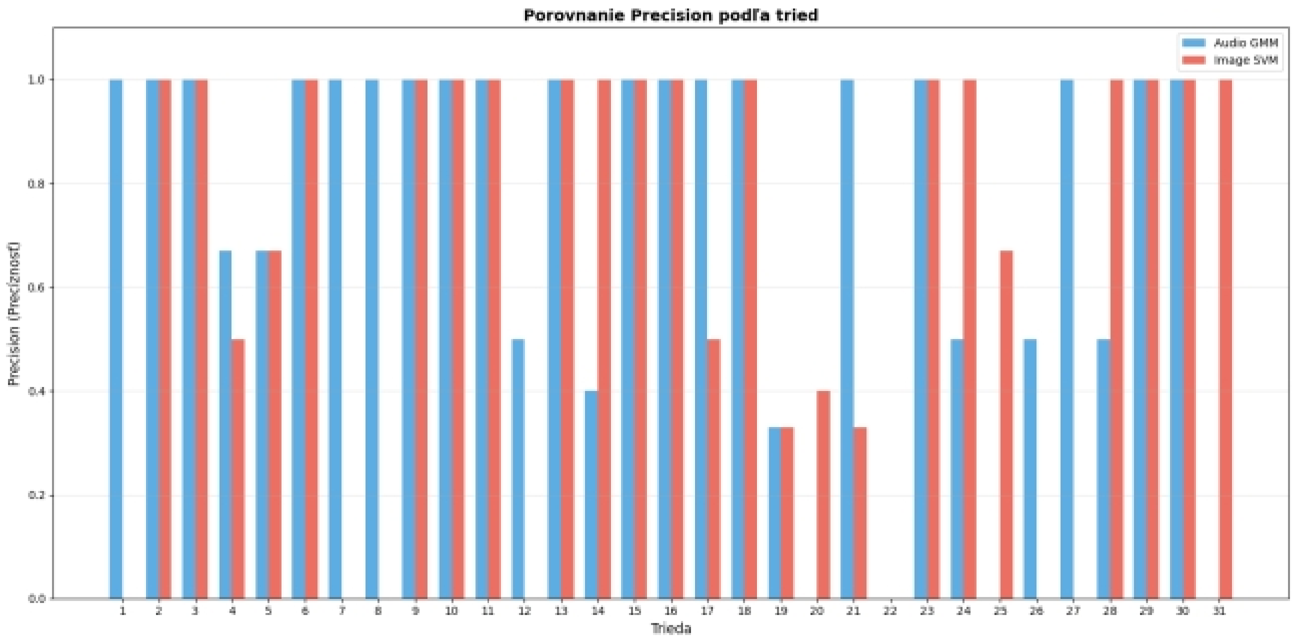

Figure 3 Comparison between classes (precision)

Classes where Audio-GMM is better (PrecisionImage<PrecisionAudio):

- 1, 7, 8, 12, 26, 21, 31
- Overall, it is better in 7 classes.

Classes where Image-SVM is better (PrecisionImage>PrecisionAudio):

- Significantly better: 20, 22, 25.4, 14, 19
- Overall, it is better in 6 classes.

Same results (Precision = 1.00):

- 2, 3, 6, 9, 10, 11, 13, 15, 16, 18, 27, 28, 29, 30
- A total of 14 classes.

Class where none of the models are correct:

- 22 (Precision = 0.00)
- Total 1 class

5. Conclusion

Similar accuracy-performance for GMM+MFCC (audio) and SVM+HOG (image). The accuracy is exactly 72.58%, which shows good classification, because if we were guessing, we would have an accuracy of 3% (100/31). The higher precision of GMM indicates that it makes fewer false positive errors compared to SVM on this dataset, i.e. GMM is more reliable.

In the future, I would like to look more deeply into why class 22 is not correct in either model. At the same time, thanks to the fact that only one class is not accurate in either model, we can implement a combination of these two models and predict even more accurate classifications of people.
Furthermore, I would like to investigate the impact of the silence filtering process – whether it is more effective to filter quieter segments directly from the extracted MFCC features or to eliminate silence from the audio signal before MFCC processing and compare the accuracy of GMM models.