Open-Source Package for Generic Deep-Network-based Face Detection and Recognition in Bob

Master Project

Team: Linghu Yu and Zhang Xinyi

Supervisor: Prof. Dr. Manuel Günther

1 Introduction

Facial image recognition has a long history and is one of the most prominent examples of both machine learning and image processing. Generally, facial image recognition is performed in three different steps. The first step is the detection and alignment of the face, i.e., the locations of the face and, typically, some facial landmarks are detected. Using these landmarks, the face is cropped out of the original image and aligned into a face image. In the second step, some information are extracted from this face image, typically different kinds of facial features are extracted. Traditional approaches for automatic face recognition used different types hand-crafted features, while modern algorithm rely on deep networks to automatically extract a set of deep features for a given face. In the third step, two such representations of the face are compared, and they are classified to be the same person if their respective similarity is above a certain threshold. While this comparison step was very difficult and main research directions, for deep features simple distance metrics can be employed.

During my stay at the Idiap Research Institute, I have developed an open-source package for the comparison of facial recognition algorithms [Günther et al., 2012], which is still actively maintained.[1] One strength of this framework is that it can generate comparable and reproducible results, which are directly comparable to results of other researchers since it strictly follows the evaluation protocols defined by some dataset providers. Using this framework, we have performed a comparative study of several face recognition algorithms under several aspects of face recognition [Günther et al., 2016].

2 Assignment

While the above-mentioned framework includes many traditional face detection, feature extraction and recognition algorithms, unfortunately, most of them are outdated by now. On the other hand, many modern face detection and recognition algorithms rely on deep learning [Zhang et al., 2016, Hu and Ramanan, 2017, Chen et al., 2018, Cao et al., 2018, Deng et al., 2019], but they are not integrated with the framework. The task of this Master Project would be to change this by providing a generic interface for different face detection and face recognition algorithms based on deep learning. This includes a generic implementation that is agnostic of the deep learning framework that was utilized, as well as implementations specialized to certain frameworks. Included in this Master Project is also the documentation of the source code package, the implementation of certain test cases and the integration of the source code package into the Continuous Integration (CI) framework available at the Idiap Research Institute.[2]

The first step would be to set up an environment and to install all required packages, including Bob [Anjos et al., 2012] and some deep learning frameworks [Abadi et al., 2016, Paszke et al., 2019]. Since Bob is not available under Windows operating system, a Linux or MacOS environment is required; for example the Windows Subsystem for Linux can be utilized. The second step is to get familiar with the Python package development, which we have documented online.[3] A new Python package, for example called `bob.bio.cnn`, should be generated and set up. Third, a generic interface for extracting features from pre-trained deep networks in various frameworks should be implemented using the OpenCV DNN module.[4] For some frameworks, specialized implementations should be created, partially based on existing code[5] and partially implemented

[1] `http://www.idiap.ch/software/bob/docs/bob/docs/stable/bob/bob.bio.base/doc/index.html`

[2] For example: `http://gitlab.idiap.ch/bob/bob.bio.base/pipelines`

[3] `http://www.idiap.ch/software/bob/docs/bob/bob.extension/stable/pure_python.html`

[4] `http://docs.opencv.org/master/d2/d58/tutorial_table_of_content_dnn.html`

[5] `http://www.idiap.ch/software/bob/docs/bob/bob.ip.tensorflow_extractor/master/index.html`

from scratch. Fourth, different face detection and face recognition methods should be implemented, starting with MTCNN [Zhang et al., 2016], for which there exists already an interface in Bob[6], but also others such as TinyFaces [Hu and Ramanan, 2017] or face detection via Faster R-CNN [Jiang and Learned-Miller, 2017]. While some of the face detection networks do not particularly predict landmark locations, these landmark locations are required for face alignment and, thus, average facial landmark locations should be defined based on the detected bounding boxes. Fifth, various face recognition deep networks exist, for example VGGFace2 [Cao et al., 2018], ArcFace [Deng et al., 2019] or smaller networks such as Mobile FaceNet [Chen et al., 2018] or AFFFE [Li et al., 2018]. Each of these networks requires a different face alignment, and is implemented in a different framework. Specialized setups for each of these networks need to be defined and implemented as Baseline Modules. After implementing the interfaces, a comparative experiment should be run on some available facial image datasets, for example, on the labeled faces in the wild (LFW) benchmark [Huang et al., 2007].

To assure reproducibility, several further steps need to be taken. Most importantly, the new package must be documented properly including use cases and how to plug in novel networks. Another step is to define certain test cases, which will be automatically executed in the Continuous Integration service, which also needs to be set up (with the help of the former colleagues at Idiap).

3 Schedule

Assuming 30 hours of work per week and a total of 18 ECTS with an average of 30 hours per ETCS, we arrive at a total workload of 18 weeks. These should be distributed as follows. Since the project is handled by two students, it should be possible to work on Milestones 2 and 3 in parallel. Also, Milestone 4 can be split between the two students, i.e., one is training the deep network while the other is running the face detector on the test set.

Week 1-2 Setting up the work environment, installing all required tools, getting familiar with the Bob ecosystem.

Week 3-4 Creating a new package for using pre-trained deep learning methods in Bob.

⇒ Milestone 1: The new package is available and contains all elements required by the Bob ecosystem.

Week 5-6 Implementation, testing and documentation of the generic interface for deep learning networks using OpenCV.

Week 7-8 Implementation, testing and documentation of specialized interfaces for the frameworks Tensorflow, PyTorch, MxNet.

⇒ Milestone 2: Different network topologies can be loaded and executed using the generic framework and using specialized frameworks.

Week 9-10 Implementation, testing and documentation of several face detection algorithms, including MTCNN, TinyFaces, Faster-RCNN.

⇒ Milestone 3: Faces can be detected and facial landmarks are extracted or estimated using three different face detectors.

Week 11-12 Definition of baseline modules for several pre-trained deep networks, including VGGFace2, ArcFace, MobileFaceNet, AFFFE.

Week 13-14 Running face recognition experiments on the LFW benchmark.

⇒ Milestone 4: Different deep networks and different face detectors are evaluated on LFW.

Week 15-16 Integrate the new package into the Bob ecosystem including the Continuous Integration system.

⇒ Milestone 5: All tests on the new package are green in the CI.

If time allows Literature review of other deep face detection and/or recognition algorithms and incorporation of open-source solutions into the framework.

⇒ Milestone 6: The evaluation of Milestone 4 is extended with the novel methods.

Week 17-18 Preparation of the Presentation and writing of the final report.

[6] `http://www.idiap.ch/software/bob/docs/bob/bob.ip.mtcnn/master/index.html`

Milestones 1, 2 and 4 need to be delivered by the students at the end of the project. Milestones 3 would be good to have, but using only MTCNN can be sufficient. Milestones 3 and 4 can be worked on in parallel by the two students as these investigate different aspects of the processing chain. Milestone 5 requires collaboration with the Idiap Research Institute and might be skipped if such a collaboration is ceased by Idiap. Milestone 6 is optional.

4 References

[Abadi et al., 2016] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). Tensorflow: A system for large-scale machine learning. In *USENIX Conference on Operating Systems Design and Implementation*, pages 265–283. USENIX Association.

[Anjos et al., 2012] Anjos, A., El-Shafey, L., Wallace, R., Günther, M., McCool, C., and Marcel, S. (2012). Bob: a free signal processing and machine learning toolbox for researchers. In *ACM international conference on Multimedia*, pages 1449–1452.

[Cao et al., 2018] Cao, Q., Shen, L., Xie, W., Parkhi, O. M., and Zisserman, A. (2018). VGGFace2: A dataset for recognising faces across pose and age. In *Automatic Face & Gesture Recognition (FG)*. IEEE.

[Chen et al., 2018] Chen, S., Liu, Y., Gao, X., and Han, Z. (2018). MobileFaceNets: Efficient CNNs for accurate real-time face verification on mobile devices. In *Chinese Conference on Biometric Recognition*.

[Deng et al., 2019] Deng, J., Guo, J., Xue, N., and Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Günther et al., 2016] Günther, M., El Shafey, L., and Marcel, S. (2016). Face recognition in challenging environments: An experimental and reproducible research survey. In *Face recognition across the imaging spectrum*, pages 247–280. Springer.

[Günther et al., 2012] Günther, M., Wallace, R., and Marcel, S. (2012). An open source framework for standardized comparisons of face recognition algorithms. In *European Conference on Computer Vision*, pages 547–556. Springer.

[Hu and Ramanan, 2017] Hu, P. and Ramanan, D. (2017). Finding tiny faces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[Huang et al., 2007] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. (2007). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, Univ. of Massachusetts, Amherst.

[Jiang and Learned-Miller, 2017] Jiang, H. and Learned-Miller, E. (2017). Face detection with the Faster R-CNN. In *International Conference on Automatic Face & Gesture Recognition (FG)*.

[Li et al., 2018] Li, C., Günther, M., and Boult, T. E. (2018). ECLIPSE: Ensembles of centroids leveraging iteratively processed spatial eclipse clustering. In *Winter Conference on Applications of Computer Vision (WACV)*, pages 131–140.

[Paszke et al., 2019] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeuRIPS)*.

[Zhang et al., 2016] Zhang, K., Zhang, Z., Li, Z., and Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. *Signal Processing Letters*, 23(10):1499–1503.