- Computer Vision -

Facial recognition from RGB-Depth images (identification)

Nicolas BOCK, Arthur PHOMMACHANH

December 23, 2024

Brno University of Technology

Contents

1 Introduction

Image classification has emerged as a crucial field in computer vision these years, with applications spanning across various industries, such as healthcare, autonomous vehicles, and security. The goal of this project is to understand and analyze key mothods used for face recognition, but also design and implement a convolutional neural network (CNN) for multi-class image classification. The network is trained on a dataset containing diverse images of personality faces, categorized into multiple classes, to achieve high accuracy.

This report details the methodology, implementation, and results of the project. It also highlights the encountered challenges and experimentations done to improve the basic features and implementation.

Here you can find the project: https://github.com/nicolas-bock/POVa_Project.git

1.1 State of the Art

1.1.1 Theorical foundation

Face recognition has made significant advancements over the past decades, this is primarily due to the improvements in machine learnings and deep learnings. The theorical foundation of face recognition can be found in computer vision, where most algorithms and models aim to identify people based on their facial features.

1.1.2 Relevance of the problem

Even though significant progress were made, there are still multiple challenges when applied in real-world context : lighting, pose variations, obstruction of the whole face. These are a few examples which hinder the reliability of face recognition models.

1.1.3 Focus of our work

Our work focuses on improving face recognition by using face alignment methods. We believe that accurate face alignment is essential to ensure that the facial features are normalized for further analysis, test. By using more advanced methods for face alignment, such as MTCNN (Multi-task Cascaded Convolutional Networks) we will have better performance than using traditional methods such as Viola-Jones alogrithm.

1.1.4 Justification of the problem's relevance

Even though the Viola-Jones method works well for detecting faces in a "controlled" environments, it is limited when having to identify faces in challenging context. Therefore, improving the alignment of faces before recognition can assure higher accuracy in those challenging contexts.

2 Dataset

For this project, we are using the "Labeled Faces in the Wild" dataset (Bibliographie). It contains 13,233 images of faces collected from the web. Each face has been labeled with the name of the person pictured. There are 5749 different people, but only 1680 of them have two or more distinct photos in the dataset.

3 Method

3.1 Overview

In this study, we propose a convolutional neural netnwork (CNN)-based approach for face recognition. Our approach can be divided into several key stages :

- Data preprocessing
- Design of the CNN model

- Training
- Evaluation

We prepare the dataset by dividing it into training and testing subsets. The CNN model will then be design to include multiple convolutional layers, max-pooling layers, and batch normalization in order to process the images (CNN).

3.2 CNN

This CNN model is based on a sequence of three 2D Convolution layers, three 2D Max Pooling layers, and two Batch Normalization layers. Of course, there is also an Input layer at the begining, and a Dense layer (corresponding to the different classes), preceded by a Flatten layer and a Dropout layer, at the end.

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 125, 94, 1)	0
conv2d (Conv2D)	(None, 123, 92, 32)	320
max_pooling2d (MaxPooling2D)	(None, 61, 46, 32)	0
batch_normalization (BatchNormalization)	(None, 61, 46, 32)	128
conv2d_1 (Conv2D)	(None, 59, 44, 32)	9,248
batch_normalization_1 (BatchNormalization)	(None, 59, 44, 32)	128
max_pooling2d_1 (MaxPooling2D)	(None, 29, 22, 32)	0
conv2d_2 (Conv2D)	(None, 27, 20, 64)	18,496
max_pooling2d_2 (MaxPooling2D)	(None, 13, 10, 64)	0
flatten (Flatten)	(None, 8320)	0
dropout (Dropout)	(None, 8320)	0
dense (Dense)	(None, 1280)	10,650,880
$\text{dense}_1(Dense)$	(None, 62)	79,422

Table 1: CNN Model Structure

- Total params: 10,758,622 (41.04 MB)
- Trainable params: 10,758,494 (41.04 MB)
- Non-trainable params: 128 (512.00 B)

3.3 Training and Evaluation

The dataset is divided into two sub-sets (train and test) with a distribution of train: 80% / test: 20%. The model is trained on 100 epochs, possibly several times, and with a learning rate of 2e-4.

3.4 Definition of the parameters

- batch_size: the size of each batch => 64
- epochs: the number of epochs to train the model => 100
- l_rate: learning rate => 2e-4
- W: the width of the input image
- H: the height of the input image
- nclass: the number of classes in the dense layer.
- nchannel: the number of channels in the input layer.

4 Experimentations

In this section, we describe the key experiments conducted to evaluate and improve the performance of facial recognition model using RGB-Depth images. Using different methods, we evaluated the model's ability to recognize and identify faces.

4.1 Face Alignment Impact

For testing the effect of face alignment on the recognition accuracy. We used the MTCNN model for face detection and landmark retrieval. The experiment involved :

- The resizing of aligned images which were passed through the CNN model.
- The comparison of unaligned images with aligned images.
- The high-quality images benefitted the most from the alignment.

4.2 Comparison between MTCNN and SIFT Keypoints

We tried to compare the model by using SIFT for extracting keypoints. Unlike MTCNN which detects a limited set of keypoints, the SIFT method detects a larger number of keypoints. Although the method is very effective for facial recognition, in the context of face alignment, it made the alignment more challenging as there were too many keypoints.

- When using SIFT keypoints for alignment, too many features were detected which leads to misalignments and noisy results.
- Whereas, MTCNN focusing on key facial features, provides more reliable and precise results, improving the model's accuracy during training.

4.3 Impact of dataset configuration

We experimented using different dataset configuration to investigate the effect of the number of classes and images per class on the model performance. In this context, we had two configurations :

- **19 classes:** Each class contains at least 40 images.
- **62 classes:** Each class contains at least 20 images.

4.4 Impact of the number of epochs

We also tested the effect of training the model with different numbers of epochs to assess the moment when the model will be overfitted. To do so, we trained the model for 100, 300 and 400 epochs, and evaluated its performance after each session.

4.5 Evaluation on aligned images

After processing all the images with the alignment functions, we tried to evaluate the model on those, to compare the results and the accuracy of the model after training and evaluating on the original data. But we encountered some issues with the number of labels and the size of the images, that weren't corresponding to the input size of the CNN model. However, it is almost certain that the accuracy would not change much, or would be slightly increasing

5 Results

Experiment	Result
Face Alignment	Improved consistency in face positioning.
SIFT Keypoints for Alignment	SIFT produced too many keypoints, leading to noisy results.
Dataset Configuration (19 Classes)	Higher accuracy with fewer classes.
Dataset Configuration (62 Classes)	Test accuracy dropped due to a higher number of classes.
Face Landmark Detection	Precise for most images, misalignments in low-quality images.

Table 2: Summary of Experiment Results

5.1 Method results

5.1.1 MTCNN detector and bounding box

The MTCNN detector worked really well on the dataset, even if some photos had a really bad quality or some people were facing in a different direction (cf. Annexe). The detector also detected other people's face in the image, not corresponding to any class. But we have encountered a few cases where the face wasn't recognize at all, and the bounding box couldn't be created.

5.1.2 Guide Lines

With this method, it was possible to align all the images. The two perpendicular lines were always computed with the right angle and centered between the eyes of the main face (cf. Annexe).

5.1.3 SIFT keypoints

Results of the SIFT keypoints were good for face recognition, without specifying the number of features, but bad for face alignment because there were too many keypoints (cf. Annexe). We got better results by setting the number of features to 20-25 (cf. Annexe).

5.1.4 Face landmarks

With this method, we choose to keep five major facial keypoints (left eye, right eye, nose, left mouth and right mouth) (cf. Annexe), the result was satisfying and accurate. However, there are instances where the alignment was less precise, especially when the image quality was poor. (cf. Annexe)

5.2 CNN results

With at least 40 images per class (19 classes in total), the model achieved an accuracy of **0.83**. We tried to add more classes by taking only those that had at least 20 images, resulting in a dataset of 62 classes. On evaluation, we observed a test accuracy of **0.69**, lower than before due to the higher number of classes between which the model can choose.

Epochs	Test Accuracy	Accuracy	Loss
100	0.689	0.666	2.852
300	0.704	0.681	2.843
400	0.704	0.689	2.841

Table 3: Accuracy per number of epochs

To compare with the result we had before, we tried to use all the classes having at least two images. Of course, the results were pretty bad. For 100 epochs the test accuracy was around **0.28**, with a loss of **16.4**.

Finally, by adding a second dense layer before the last one, the results were increasing for 62 classes, with a test accuracy of **0.71** and a loss of **2.106**. But on the accuracy and loss graphs, we observe a big spicke around the epoch 80, where the model maybe started overfiting the training data (cf. Annexe).

6 Conclusion

In this study, we focus on improving facial recognition by using facial alignment and more advanced methods such as MTCNN. Our goal was to identify faces under varying poses, lighting conditions, and image quality.

The comparison between SIFT and MTCNN for feature extraction revealed that while SIFT produced more keypoints for face recognition, it was less effective for precise alignment, as it generated an excessive number of keypoints that complicated the process. On the other hand, MTCNN provided more reliable results by focusing on a limited set of facial landmarks.

Furthermore, our experiments with dataset configurations showed that reducing the number of classes to 19 resulted in higher test accuracy, while increasing the number of classes to 62 led to a drop in performance. The model struggled to select the correct identity from a higher number of classes. This observation highlights the importance of balancing dataset complexity with the model's ability to generalize and compute efficiently.

In terms of training, we found that the model performed optimally after 300 epochs, with no significant improvement in accuracy beyond this point (it stalled at 0.704 in our case). This suggests that further training would likely lead to overfitting.

Overall, using advanced face alignment methods such as MTCNN can significantly improve face recognition accuracy. However, there is always a need to further optimize the model when working with datasets containing more classes.

In conclusion, this work highlights the importance of accurate face alignment and the limitations of traditional methods. By using advanced techniques such as MTCNN, we demonstrated improvements in face recognition performance.

7 Bibliographie

1. E. Learned-Miller, G. Huang and T. Berg (2007-2018), University of Massachusetts Amherst. LFW (Labeled Faces in the Wild) dataset: https://vis-www.cs.umass.edu/lfw/
2. Kazemi, V., Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. https://www.csc.kth.se/~vahidk/papers/KazemiCVPR14.pdf
3. Zhang, X., Zhang, Z. (2012). A survey of face alignment methods. https://www.microsoft.com/en-us/research/wp-content/uploads/2012/01/cvpr12_facealignment.pdf

8 Annexe

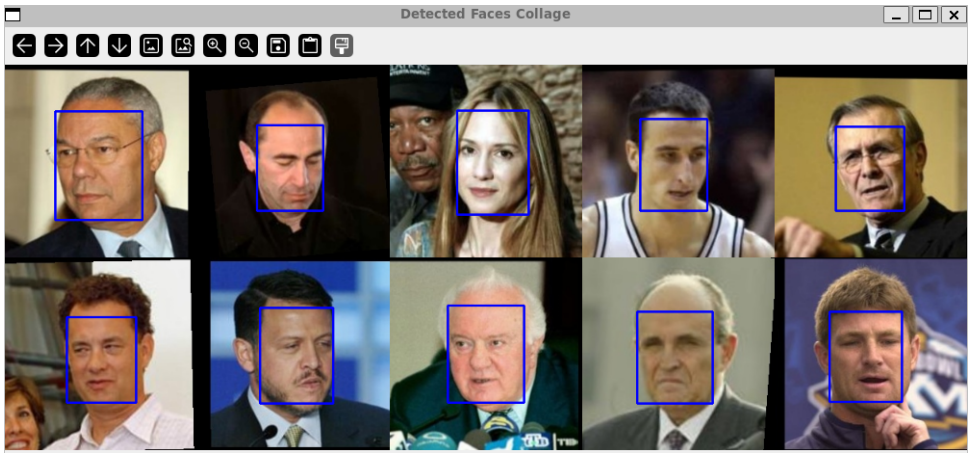

Figure 1: MTCNN

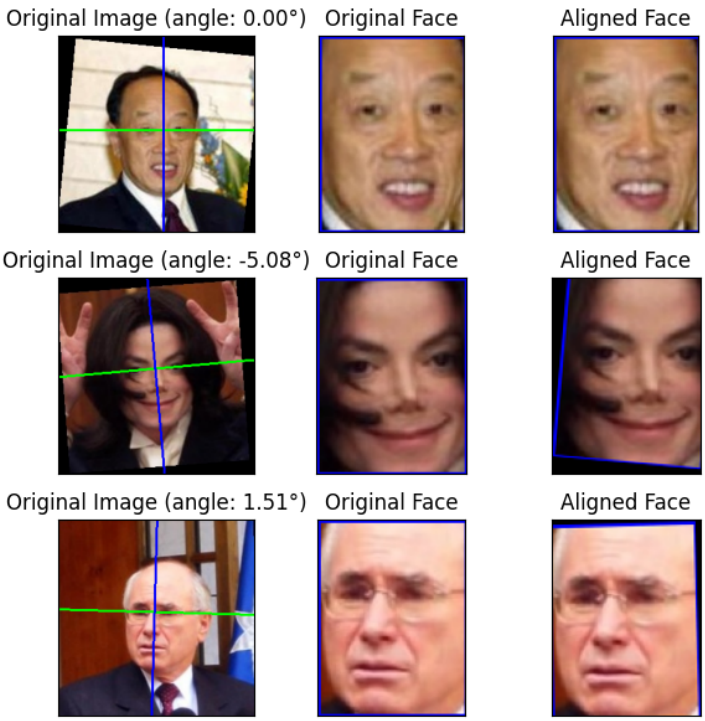

Figure 2: Guide Lines example

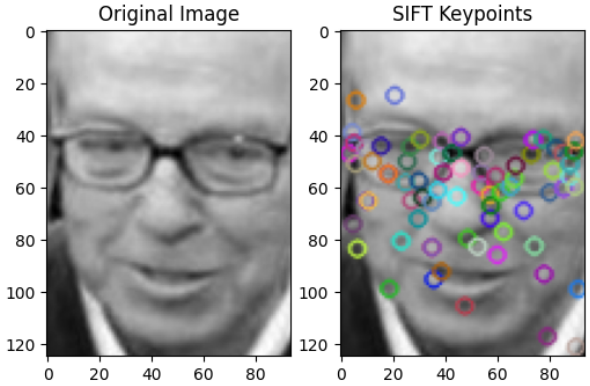

Figure 3: SIFT without specified features

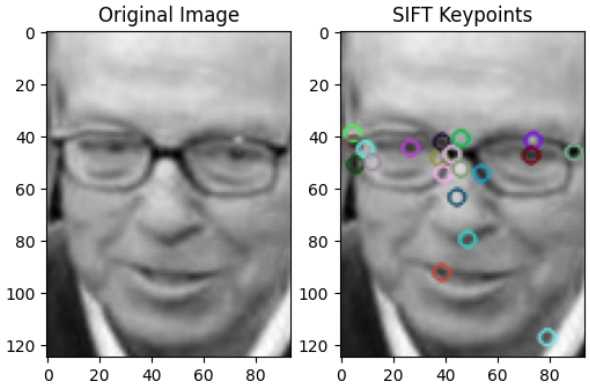

Figure 4: SIFT with only 25 features

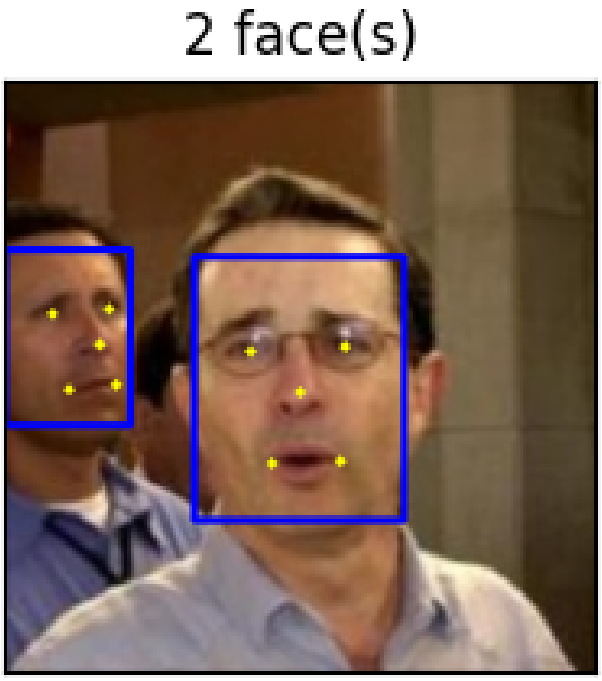

Figure 5: Keypoints using MTCNN

Figure 6: Keypoints using MTCNN on a poor quality image

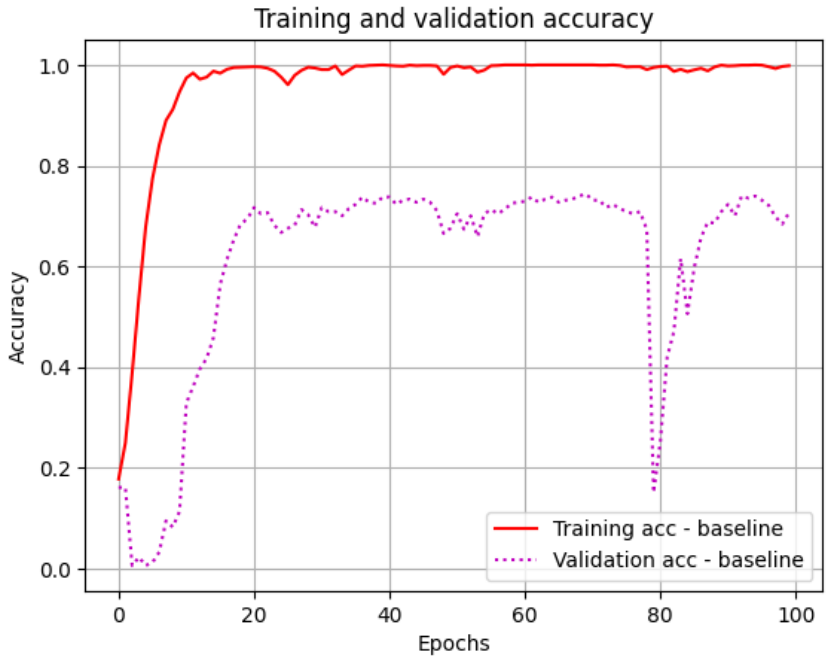

Figure 7: Evolution of the accuracy in 100 epochs

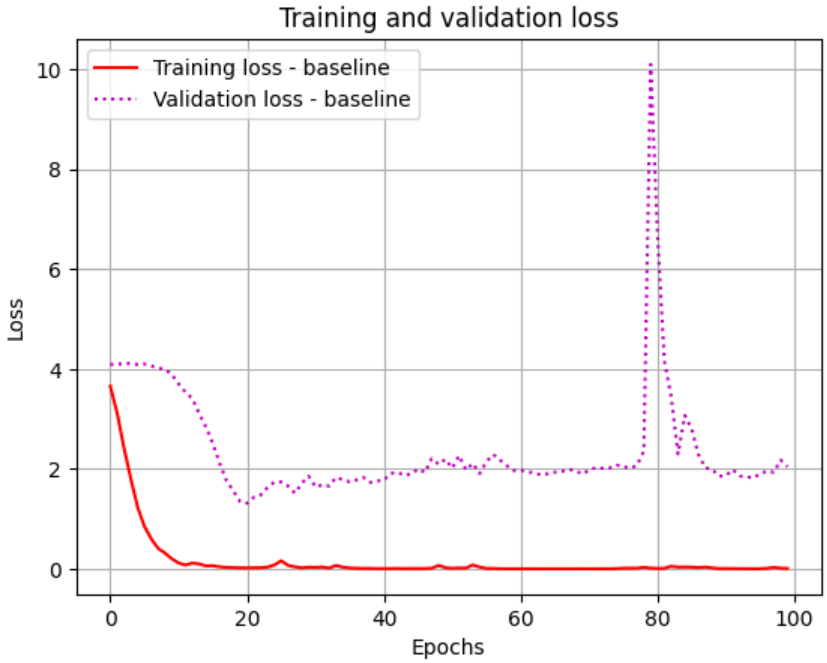

Figure 8: Evolution of the loss in 100 epochs