Final project

Noam Salomonski 303161194

Face Alignment

Github:

https://github.com/noamzilo/deep-head-pose

Trained model included in the github and also in the archive.

The following is based on https://arxiv.org/pdf/1710.00925.pdf , namely Hopenet.

In order to train the network, I used the 300W-LP dataset, that can be found here.

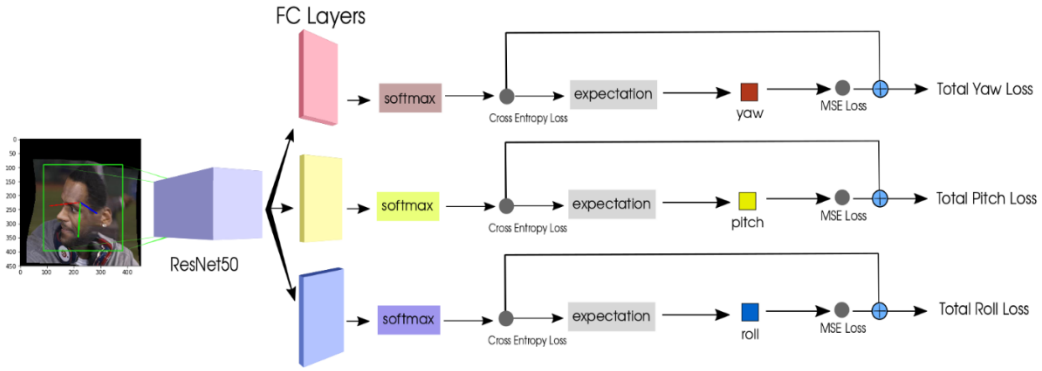
It consists of about 120000 images, which are 3d augmentations of about 7000 images such that the pose remains known:

Even though this method creates distortions in the actual shape of the face caused by the projected 3d model, it yields a nice-looking result, and looks similar to the test set and has a large amount of images, thus I selected it for training.

The net was trained with data up to angles of 99 degrees, thus it is likely to fail more on larger angles.

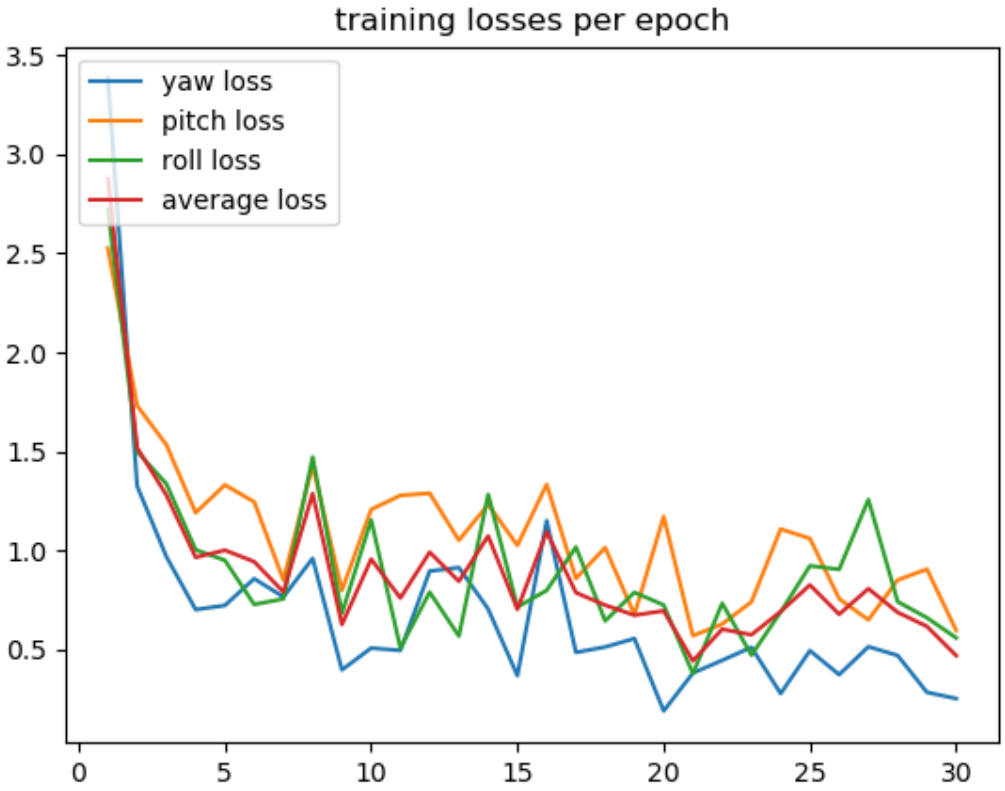
I had to clean the dataset and remove photos which had angles larger than that range.

I used the exact same training parameters as the paper, and they all appear in train_config.yaml Most importantly, alpha=0.001, learning_rate=0.00001, batch_size=16. I stopped training after 30 epochs manually, because it already took 36 hours on a K-80, and because the paper only used 25 epochs.



The network has 3 separate outputs – one for each angle (yaw, pitch, roll). Each angle's loss is a combination of a binned loss (a cross entropy loss) and a MSE loss. The binned loss helps stabilize the neighborhood of the prediction, and the MSE gives a more fine-grained prediction within the neighborhood.

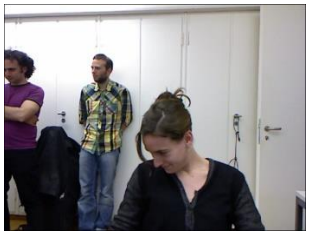
Training losses per epoch:



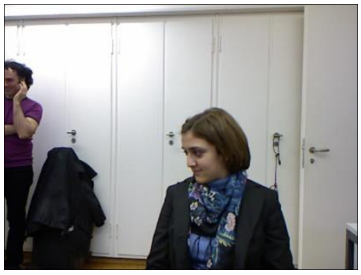
Deciding on a validation set

Biwi's photos are of good quality, but

1. Require cropping of the correct face manually, or to drop photos with multiple faces



2. Have poses much larger than are required for the test set, which we know are much harder to tag correctly



3. Have only a few test subjects, which is not diverse enough.
4. Are tagged exactly by Kinect, unlike the photos in the test set which are tagged by a facial feature finding algorithm and an unknown 3d model, which doesn't fit the face exactly.
5. The paper discusses a method of using this dataset for validation, but I had no more time, thus could not use it.

AFLW is quite good, but required cleaning, and handling, for which I did not have enough time left.

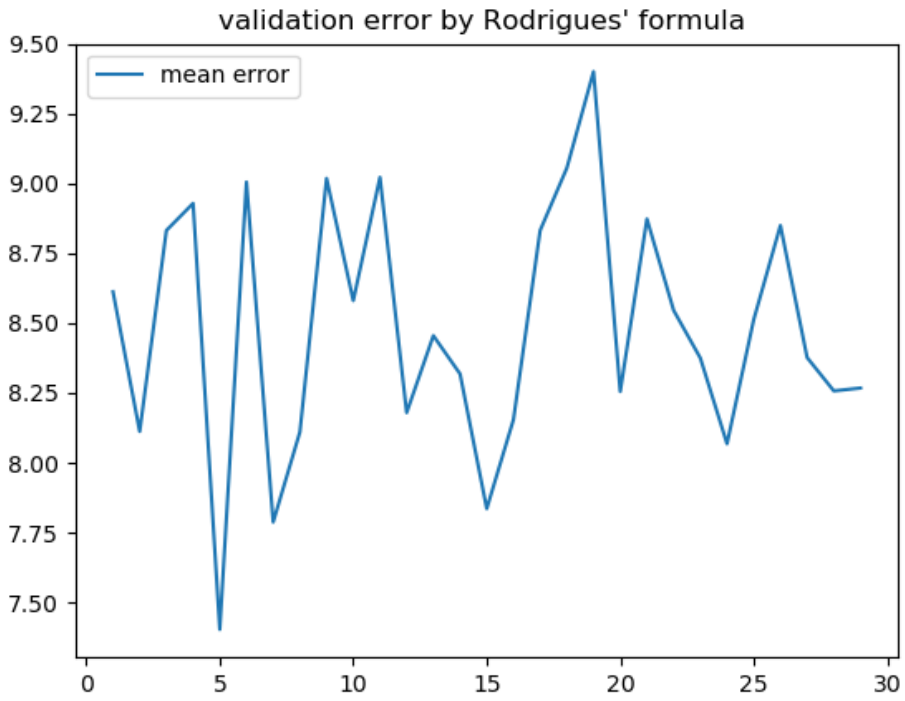
Example images which require cleaning:

Has a pose which is too large for the test, and the network won't handle it properly.

Has more than one face.

Thus, I elected to choose the validation sets published in the forum as the validation set for selecting the epoch for early stopping:

1. Some of the photos are almost exactly like those in the test set.
2. The tagging process is the same as that of the test.
3. Note that I have deleted all images with more than one face from the (first) validation set, and deleted images whose tagging was utterly wrong, according to the original tagging algorithm



Clearly, according to the validation set, it is best to choose epoch#5 (I only submitted that model).

Here are some results:

Blue points “forward”, red points “to the left”, and green points “down”, relative to the face.

These seem to work over multiple poses, races, ages, and expressions. Even if the poses are not perfect, they are good enough for me. It is also impossible to compare them accurately to those that are output from another model, that, itself is prone to error. Thus, looking at the images and seeing something that “makes sense” is actually a good measurement in my opinion for this task, given the time I have.

Things I didn’t do:

I wanted to experiment with my own augmentations such as blacking sections of the image to deal with occlusions, to change the backbone net to one that specializes in faces, or even to try and use gans to generate tagged faces, but I simply had no time and had to submit only the original paper’s network architecture, and feel lucky that I got even that on time.

I didn’t even try to go through a landmark based method, because I believed the test images would be too hard for such a method, and deep learning was the only viable solution. I don’t know if this is indeed the case.

What took the time?

Most of the time went to technical issues, such as: making the validation set work finding out that some tags were wrong [which required to get the original tagging algorithm to work, and to make sure the wrong annotations are not due to a bug], manually removing images with multiple faces, finding out that the pitch was measured at a negative direction from that of scipy, finding out how to convert roll, pitch, yaw to rotvec such as the validation file, making the paper's code work, making a cloud GPU work, finding the datasets, training a NN for the very first time, and more technical issues as such.

I wish I had more time to get to the more interesting, architectural, data-related parts, but unfortunately it was not possible for me, even though I devoted countless hours to this project.

Running instructions:

1. Download the code from github
2. Pip Install requirements.txt
3. This will download the trained net to test_on_validation.selected_model.pkl
4. Go to config.test_config.yaml

```
test_images_folder_path: C:\Noam\Code\vision_course\downloads\test\images
create_rel_paths_file_at: C:\Noam\Code\vision_course\downloads\test
```

   a. Point test_images_folder_path to the folder holding the images.
   b. Point create_rel_paths_file_at to one folder above the images folder.

5. To set an output directory, use config.hopenet_config.yaml

   ```
   output_dir: C:\Noam\Code\vision_course\hopenet\output
   ```

   and set your desired output folder.
6. To run, use the file test_on_validation.test_on_image_folder.py
7. This will output all the images with their pose annotation, and an output.csv file

I used original_code_augmented\train_hopenet_original.py to train the net with above parameters (as stated in the paper).

The graphs were produced using test_on_validation.test_and_validate.py and plottong_for_report package.