# Deep Learning - Assignment 2

submitted by: Or Oxenberg 312460132,
Sahar Baribi 311232730

Analysis of the dataset

- Size - there are 5749 folders that contain photos. 4069 folders contain only one photo, and 1680 have more than one photo. The maximum number of photos in a folder is 530 photos.
- Train - The training data is a dataframe with 2200 rows, where half are pairs of similar photos (label = 1) and the rest are pairs of non-similar photos (label = 0).
    - Total number of examples - 2200
    - 1100 for each class (1, 0)
- Test - The test data is a dataframe with 1000 rows, where half are pairs of similar photos (label = 1) and the rest are pairs of non-similar photos (label = 0)
    - Total number of examples - 1000
    - 500 for each class (1, 0)

Architecture Description

We used a similar architecture to the one explained in this paper.

- Layers and filters:
    - We have 4 convolutional layers with a relu activation function and a fifth fully connected layer with sigmoid activation function. Each layer is followed by a max-pooling layer. The layers are as follows:
        - 64 with (10,10) filters
        - 128 with (7,7) filters
        - 128 with (4,4) filters
        - 256 with (4,4) filters
        - flattened vector of 4096.
    - These layers are followed by a distance layer that computes the distance between the outputs of the two networks.
    - The last layer consists of a sigmoid activation function. it gets its input from the distance layer and the output is whether the images are similar or not.
    - For better performance, we added batch normalization for all convolutional layers.
- Dimensions - we resize the images from 250*250 to 105*105. The reason for this is better performance on the smaller image size. we assume that when we remove the edges we force the network to focus more on the face in the image and less on the background noise.
- Learning rate - similar to the paper, we use a decaying learning rate. we start with a learning rate of 0.0001, and a decay rate of 0.99 every 2 epochs.
- Optimization - Adam optimizor
- Regularization - all layers have l2 regularization with a regularization factor of 0.0002
- Batch normalization - as mentioned in the layers section, we use batch normalization for all the convolutional layers.
- Dropout - We added dropout with a rate of 0.1 after the first convolutional layer.

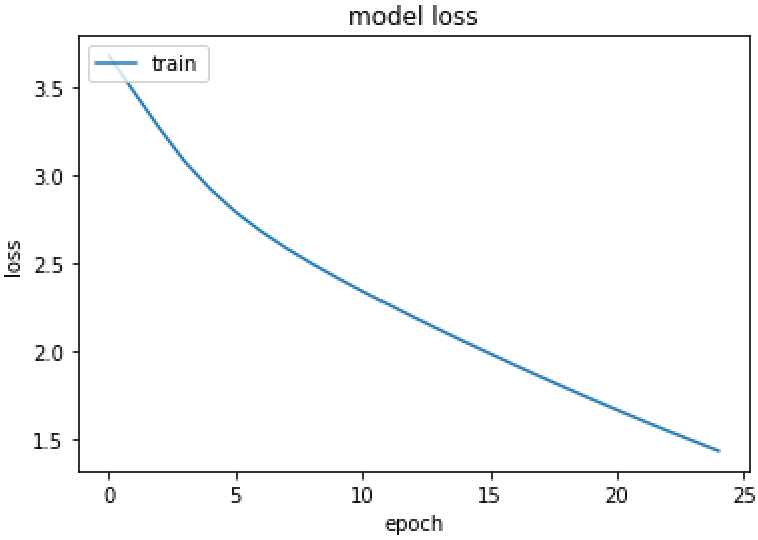
## Experimental Setup

Most parameters described in the table below were decided based on trial and error or were chosen based on their use in the paper described. We mainly followed the paper architecture and added batch normalization and dropout.

- Batch size - we used a batch size of 32. The batch size was chosen after trial and error and this was the batch size with the best performance.
- Architecture params - the architecture is described in the architecture description section.
- Stopping criteria - our stopping criteria is on the validation loss. If validation loss does not decrease within 2 epochs the training will stop
- Learning rate - we started with a fixed learning rate and after a few experiments, we tried changing the learning rate to a decaying learning rate.
- Epochs - we train the model for 25 epochs unless reached the stopping criteria. The number of epochs was also chosen after trial and error.
- We split the data into train and validation, where the validation holds 20% of the train data.
- Weight initialization - random normal distribution with a mean of zero and a standard deviation of 0.01. The weights initialization for the final layer was the same with 0.02 std. This was chosen based on the parameters from the paper.
- bias initialization - random normal distribution with a mean of 0.5 and a standard deviation of 0.01. This was chosen based on the parameters from the paper.

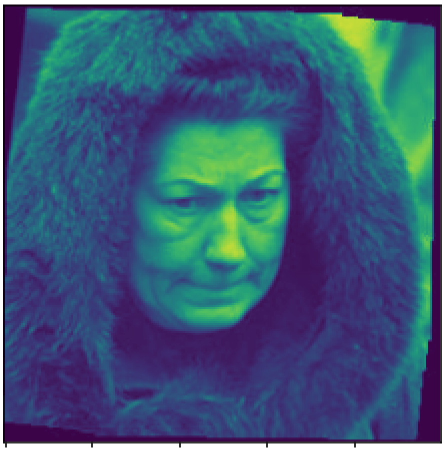
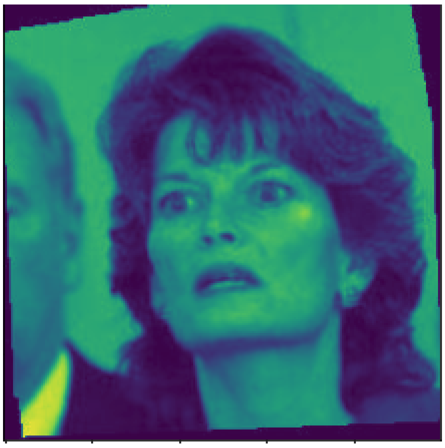
## Performance analysis

- Convergence time - 25 epochs, 2.65 minutes for training and evaluation.
- Final loss -
  - Test set - 1.8826
  - Validation - 1.9058
- Accuracy
  - Test set - 0.7220
  - Validation - 0.7043
- N-way -
  - Test set - 0.648
- Train loss -



- Accurate and misclassifications images-
  - Accurate classification - the model labeled the following two images as not the same person. The images are very different - we see a man and a woman, from different races, and the overall image, is different.

  - Misclassification - Here we can see that even though these are different women, the model may be confused about the coat that can be similar to the woman's hair.

- another example is the following - the model may be confused due to the fact that the men are similar in presence. They are both Afro-American, with short hair and similar face details.

Experiments documentation

The following table presents our different attempts in improving the network. Not all attempts are described here, but the most recent with the best results. After applying different architecture we chose the one highlighted. The epochs column presents the number of epochs required and the number of epochs that ran in practice due to early stopping.

| epochs | layers | neurons | regularization and lr | batch size | weight init | test acc | train acc | test loss | train loss | 3-way test |
|---|---|---|---|---|---|---|---|---|---|---|
| 7/30 | 5 | 64,128,128,256,4096 | none | 32 | none | 0.654 | 0.9987 | 0.8544 | 0.0391 | 0.63 |
| 6/30 | 5 | 64,128,128,256,4096 | none | 16 | none | 0.64 | 0.9869 | 0.906 | 0.0567 | 0.574 |
| 6/30 | 5 | 64,128,128,256,2048 | none | 32 | none | 0.625 | 0.9928 | 0.83 | 0.0654 | 0.454 |
| 7/30 | 5 | 64,128,128,256,2048 | l2(2e-4) -all layers | 32 | none | 0.6160 | 0.9786 | 0.9183 | 0.1499 | 0.582 |
| 30/30 | 5 | 64,128,128,256,2048 | l2(0.01) -all layers | 32 | none | 0.6430 | 1.0000 | 1.0227 | 0.0407 | 0.512 |
| 30/30 | 5 | 64,128,128,256,2048 | l2(0.01) -all layers + decay lr | 32 | none | 0.6480 | 1.0000 | 1.0376 | 0.0562 | 0.57 |
| 30/30 | 5 | 64,128,128,256,4096 | l2(0.01) -all layers + decay lr | 32 | RandomNormal(0,0.02) | 0.6490 | 1.0000 | 0.9946 | 0.1169 | 0.608 |
| 9/30 | 5 | 64,128,128,256,2048 | l2(0.01) -all layers + decay lr | 32 | None | 0.6480 | 1.0000 | 1.2682 | 0.4723 | 0.588 |
| 40/40 ** No validation | 5 | 64,128,128,256,4096 | l2(2e-4) | 32 | None | 0.706 | 1.0000 | 1.177 | 0.0939 | 0.654 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 15/40 ** No validation | 5 | 64,128,128,256,4096 | l2(2e-4)+ decay lr | 32 | None | 0.6720 | 0.9990 | 1.1090 | 0.2777 | 0.714 |
| 15/15 | 5 | 64,128,128,256,4096 | l2(2e-4)+ decay lr | 32 | None | 0.6800 | 0.9893 | 1.1030 | 0.3226 | 0.694 |
| 20/20 | 5 | 64,128,128,256,4096 | l2(2e-4)+ decay lr | 32 | RandomNormal(0,0.01) | 0.6870 | 1.0000 | 1.4128 | 0.4724 | 0.628 |
| 50/50 saved best | 5 | 64,128,128,256,4096 | l2(2e-4)+ decay lr | 64 | RandomNormal(0,0.01) | 0.6860 | 1.0000 | 1.2933 | 0.3685 | 0.584 |
| 30/30 saved best | 5 | 64,64,128,256,4096 | l2(2e-4)+ decay lr | 64 | RandomNormal(0,0.01) | 0.6840 | 1.0000 | 1.6865 | 0.7619 | 0.57 |
| 30/30 saved best | 4 | 64,256,256,4096 | l2(2e-4)+ decay lr | 64 | RandomNormal(0,0.01) | 0.6200 | 1.0000 | 1.8166 | 1.0432 | 0.468 |
| 30/30 | 5 | 64,128,128,256,4096 applied batch normalization | l2(2e-4)+ decay lr | 32 | RandomNormal(0,0.01) | 0.714 | 1.0000 | 1.5602 | 0.9379 | 0.63 |
| 25/25 | 5 | 64,128,128,256,4096 applied batch normalization + dropout | l2(2e-4)+ decay lr | 32 | RandomNormal(0,0.01) | 0.7220 | 1.0000 | 1.8757 | 1.2145 | 0.684 |