Deep Learning project 1: Face recognition

Ruben Schut [a] Sharif Hamed [a]

[a] *University of Groningen*

1 Introduction

In this project, several experiments for different architectures of LeNet-5 are presented. LeNet is a benchmark architecture for image processing where an image generator can be used to expand the dataset. This feature makes the network applicable to small datasets, such as the one used in the project. The dataset and pre-processing steps are presented in section 2. In section 2.1, we briefly mention the architecture and the features used to perform the experiments as well. Further, in section 4 the results are presented. Finally, section 5 summarizes our conclusion. For the experiments in this project, we wanted to test whether our knowledge of deep architectures is also applicable to an unusual data set.

2 Methods

2.1 LeNet

The network that we choose to use is the LeNet-5 network [1] from a package of *Keras*. The reason for choosing LeNet was because our data set is on the small side and a smaller less deep network like LeNet would be more then flexible enough. The network its architecture is displayed in figure 1. The parameters are set to: epochs = 10, learn_rate = 1e-4, Batch Size = 60. Furthermore, the default optimizer is Adam and the default activation function is relu.

2.2 Making train and test data

We have created an artificial data set of images of four individuals (including ourselves). The total size of the training and testing data is 160 and 40 photo's per individual, respectively. For the training data the models had to sit still and show there resting face while pictures from slightly different angles where taken. An example of the training data could be seen in figure 2a. An image generator also from *Keras* is used to increase the training data size. In theory, this should help training for smaller data sets. The image generator creates rotations up to an angle of 30, zooms up to 0.2, width- and height-shifts up to 0.1 and affine shears up to 0.2 for each image and appends them to the training data set.

The test data is not taken from the training set. Instead, the testing data is made independently from the training data and is made difficult on purpose. For the test data the individuals were made to laugh and make funny faces and where shot from difficult angles. An example of the test data can be seen in figure 2b. Further, we extract the faces of all the images by use of *faceCascade.detectMultiScale* of the *opencv* package. The extracted faces where used as input for the classifier LeNet.

3 Experimental Design

3.1 Model selection

The experiments were performed using two different architectures. The first architecture is the same as in figure 1 but with one convolutions and max-pooling layer added. The second architecture is the same as in figure 1 but with one convolution and max-pooling layer removed.

For the first architecture kernel sizes of 5, 3 and 2 are used for the first, second and third convolutional layer, respectively. The reason for this is to keep more information about the gradient preserved in the later layers while having a much more flexible network compared to the big network. The smaller network has a kernel size of 3 in the convolutional layer.

The reason for choosing a smaller and bigger network is to test if detail preservation or generalization on the test data is most important. The first way of reasoning is that since our test data is very different from our train data, it could be that having a shallow network will make sure that that the test data is better classified due to less over-fitting on the training data giving better generalization on the test data. The other way of reasoning is that more detailed feature extraction from the train data could give better results on the test data because the detailed features should also be present in the test data no matter if it is different from the training data.

3.2 Controlled input per model

In this project experiments are performed using different configurations of characteristics of LeNet. As mentioned in section 2.2, the image generator is used. Further, instead of the default ReLu activation function, we also use the Scaled Exponential Linear Unit (SeLu). SeLu activation is defined as $scale * elu(x, alpha)$. The scale parameter may help to reduce the size of the gradient. This may be beneficial because the network used is rather small. Moreover, the data is resized differently as well. The different configurations are all combinations of toggles, where *'on'* signifies the use of the image generator, large scale-size and the use ReLu, rather than SeLu. Please refer to table 1, to see how this is used.

4 Results

Table 1 summarizes the results of the different configurations of the controlled input. The results are the best accuracy on the test data from all the epochs. *OVF* means that in this condition the network over-fitted to the training data within the maximum epoch. We observe over-fitting in nearly all experiments that did not use the image generator. Furthermore, ceteris paribus, SeLu out-performs ReLu in all experiments. Also, the performance on large-sized images as compared to smaller-sized ones depends on the architecture of the network. Specifically, it depends on the flexibility of the network. In these experiments, the best configuration is the result of **row 3**. In this configuration, the image generator, large scale-size and SeLu activation is used in combination with the deeper network. This finding is consistent with our expectations and motivations above.

5 Conclusion

In this research project, it is shown LeNet is a good architecture for face recognition and classification. This project presents different architectures of LeNet that may serve as an indication for design choices based on the availability of training data.

From the results of the project, we can infer interesting findings on the behaviour of shallower versus deep convolutional networks. If the training data set is small, one may opt to use a shallow network, rather than a deep one. This leads to better generalization on testing data. This is because inputs with larger dimensions are able to cope with a more flexible network. Also, a shallow network generates a better generalization for fewer data points (no image generation). This becomes evident from the comparison of **row 5** for both architectures in table 1. Further, the image generator may overcome over-fitting problems. This is due to more data points with less monotone features are being added to the training data. The use of Selu gives better results leaving us to conclude that reducing the size of the gradient leads to better results in all our experimental configuration. The results also lead us to conclude that it is possible to classify images that are made different from the training data up to 90% accuracy. So it seems the network is able to extract features from the images that are invariant to: face conditions(resting face, laughing, funny faces), angles and illumination.

References

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Appendix

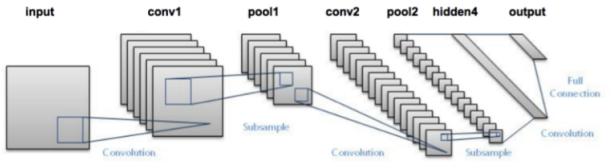
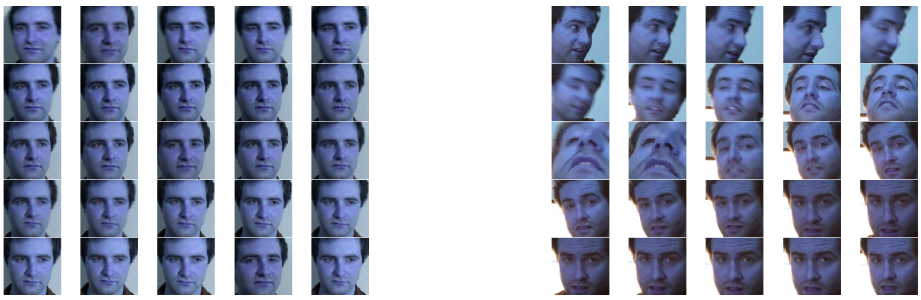


Figure 1: Architecture LeNet



(a) Training data

(b) Testing data

Figure 2: Samples of training and testing data

Table 1: Results architectures 1 and 2

row	Architecture 1 Image generator	Activation ReLu	Large Images	Results
1	1	1	1	*90.15 %*
2	1	1	0	*82.12 %*
3	1	0	1	*91.52 %*
4	1	0	0	*86.06 %*
5	0	1	1	*81.21 %* OVF
6	0	1	0	*76.97 %* OVF
7	0	0	1	*88.18 %* OVF
8	0	0	0	*81.52 %* OVF
row	Architecture 2 Image generator	Activation ReLu	Large Images	Results
1	1	1	1	*69.70 %*
2	1	1	0	*89.39 %*
3	1	0	1	*87.72 %*
4	1	0	0	*89.24 %*
5	0	1	1	*73.94 %*
6	0	1	0	*87.12 %* OVF
7	0	0	1	*79.39 %* OVF
8	0	0	0	*78.79 %* OVF