Laboratory Report - Image Recognition

Course: SGN-26006 Advanced Signal Processing

Assignment no. 6: Image Recognition

Authors: Adam Ligocki, Amir Salah

Measured: 10th of December, 2018

Table of Contents

1. Task

Investigate the techniques behind convolutional neural network and create a simple real time classification from live video input with two output labels: smiling and non-smiling.

Implementation was done in Python using Keras library that abstracts on top of TensorFlow.

For the training set the MPLab GENKI dataset should was used. This dataset already contains 4000 labeled face images divided approximately to two halves, smiling and non-smiling classes.

2. Solution

Architecture

We have designed the 7-layer neural network. It contains six 2Dconvolutional layers, each layer is batch-normalized and Relu activation function was used. Every second layer is maxpooled with size 2.

The final layer is fully connected one with softmax output into one-hot-end encoding two class output.

Figure 1 - Used neural network architecture

The dropout rate was used 0.5 and all convolutional kernels was of size 3x3 pixels.

Dataset

As a learning data the MPLab GENKI dataset has been used. It contains exactly 2162 smiling images and 1838 non-smiling ones. Also, the data are sorted by classes in the dataset, so before the learning it needs to be randomly shuffled for an improved accuracy.

For better results, the images has also been normalized in range of <0,1> and also the Keras's build in data generator has been used to create wide range of different image augmentations which helps reach better learning results.

Figure 2 - Example of images from MPLab GENK dataset

As and augmentation the random rotation of 45deg has been used as well as wide and height shift of 20% of image size, the horizontal image flip and brightness adjustment in range <0.5, 1.2>.

The required output classifying has been mapped into one-hot-encoding format.

Learning

The learning process has been realized by Keras's inbuilt fit function for sequential model.

For scoring the binominal classification problem we have used binary cross entropy loss function and learning has been driven by Adadelta optimizer with initial parameters of learning rate = 1, rho = 0.95 and decay = 0.

The learning parameters has been chosen 32 batch size and 50 epochs.

Also, the terminating condition has been set to end up learning after reaching 90% accuracy.

The dataset has been spitted during the learning into learning and validation subsets by ratio 3000:500. Last 500 images have been used for testing.

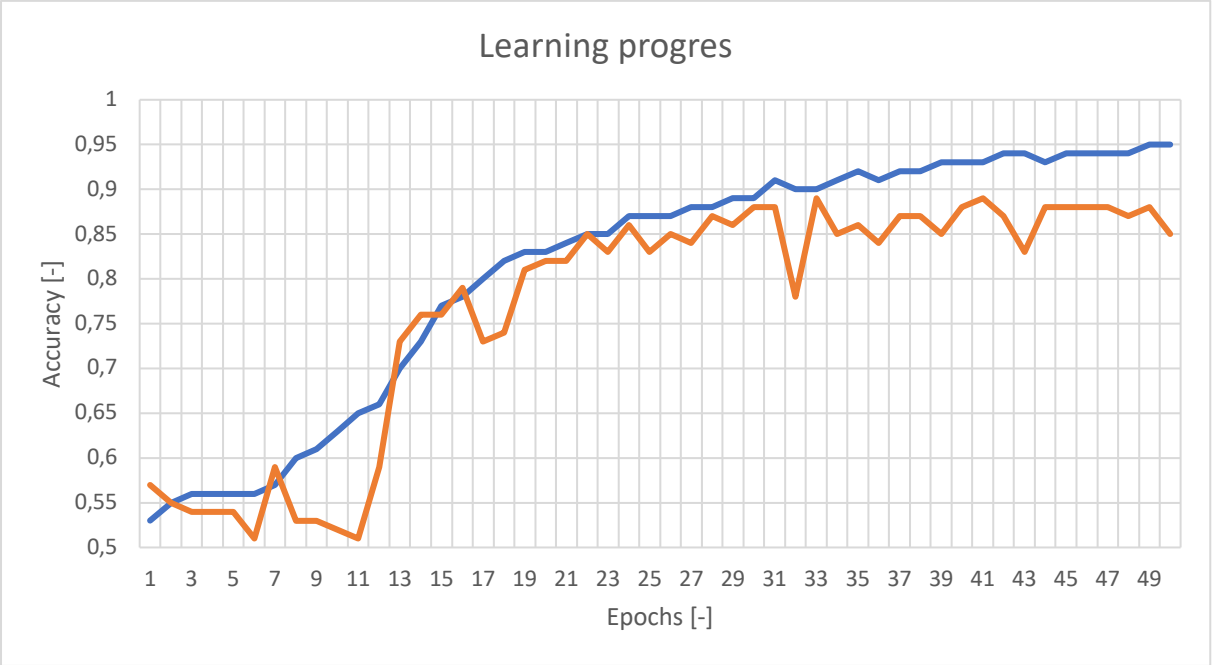

Figure 3 - Learning process. 50 epochs

3. Results and Video processing

First testing has been done on 500 test images. Every image has been putted into neural network and the output has been compared with image label. As a result, we get following confusion matrix.

Pred. / Ground True	*Smiling*	*Non-Smiling*
Smiling	200	29
Non-Smiling	25	246

Table 1 - Confusion matrix of learned neural network testing

The final validation of the trained neural network has been done on short video sequence taken by mobile phone.

This video has been loaded by opencv's VideoCapture API. The region of interest which contains the face was cut out from the video and the result images has been slightly blurred and resized into 64x64px format.

Finally, 24 pre-labeled randomly selected images from video sequence has been putted into neural network with final accuracy of 83%.

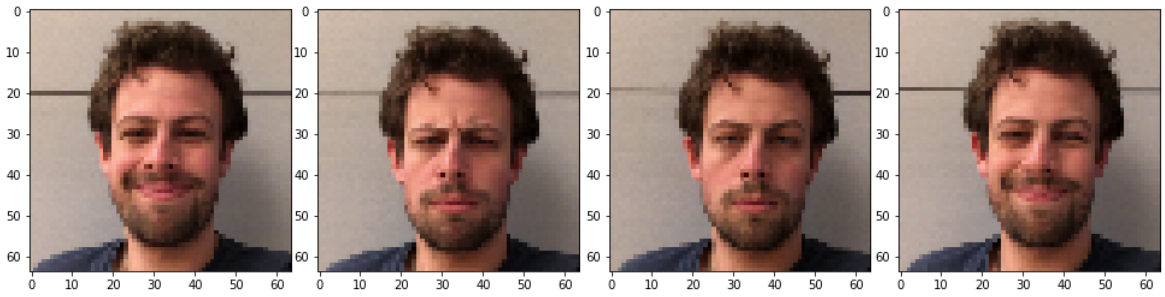

Figure 4 - Example of different reslts of NN "SMILE" calssification. From left folows: True Positive, True Negative, False Positive and False Negative

4. Conclusion

In this assignment, we have successfully designed and trained a convolutional neural network for classifying smiling and non-smiling faces. Our network consists 6 convolutional layers combined with batch normalization, dropout and max pooling layers. The output is finalized with two fully connected layers. Output is in one-hot-encoding format.

We have trained this network using 3000 labeled images, validation was done on 500 images and for final testing the other 500 images left were used.

The test results give us an accuracy of around 89.2%.

At the end, the network was also tested in real-time using our own stream input and the average accuracy was around 83%.