Backpropagation neural network based gender recognition from face images

Huanghuazhen
2016130265

Huangshaocong
2016130268

Chenkaipeng
2016130250

Abstract

Since there are many ready-made face detection classifiers in python extension packages, for the final assignment, we dicided to use face detection classifier combined with backpropagation neural network to implement gender recognition from face images. This paper has finished the following works: 1)Use face detector to get face images and preprocess them;2)Take the images as input to train a BP neural work for gender recognition;3)Test the trained neural network.

1. Introduction

This section will briefly introduce backpropagation neural network and its application in this paper.

1.1. Backpropagation and Neural network

Backpropagation algorithms are a family of methods used to train a supervised neural network(NN) following a gradient descent approach that exploits the chain rule.[1]And the goal of a supervised NN is to find a function which fits the inputs and desired ouputs best. It can be presented as a black box with 2 methods learn and predict as following[3]:

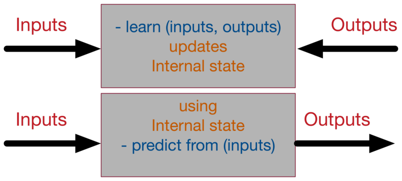

Figure 1. Learn and predict process

The learning process takes the inputs and the desired outputs and updates its internal state accordingly, so the calculated output get as close as possible from the desired output.

We decompose the learning process into several blocks:

• **Step 1 Model initialisation:** The first step of the learning, is to start from the initial weights of the network. A random initialisation of a network is a common practice. Through the learning process, the weights will be updated many times and finally converge to the ideal solution.

• **Step 2 Forward propagate:** To check the performance of initialising the model at random, we pass the input we have through the network and calculate the actual output of the model streightforwardly.

• **Step 3 Loss function:** Loss function is a performance metric on how well the NN manages to generate outputs as close as possible to the desired values.A machine learning problem usually can transform to an optimisation process that aims to minimise the loss function.

• **Step 4 Differentiation:** Differentiation can guide us to optimise the weights. Basically it deals with the derivative of the loss function. We can get the rate of which the error changes relatively to the changes on the weight by calculating directly the derivative of the loss function. Here is what our loss function looks like and the meaning of derivative :

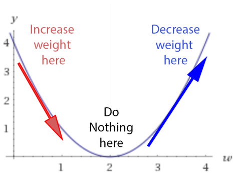

Figure 2. Loss function and meaning of the derivative

• **Step 5 Back-propagation and Weight update:** By derivate each function from the whole layers of the network, we can propagate back the error from the end to the start. In the learning process, backpropagation is commonly used by the gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function. And the most simple and intuitive rule of weight updates is the delta rule: $Newweight = Oldweight - Derivative\ Rate * Learning\ rate$

• **Step 6 Iterate until convergence:** Since we update the weights with a small delta step at a time, it will take several iterations in order to learn. The times of iterations depends on many factors.

To summarise, here is what the learning process on neural networks looks like:

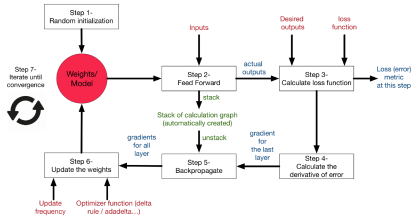

Figure 3. The learning process on NN

1.2. Applying to gender recognition

To apply Backpropagation NN to gender recognition, we first set the number of hidden layer, sigmoidal activation function and backpropagation function of the network, then we input the processed face images as training examples to train the network, and finally we get the trained gender recognition NN.

2. Programming and Training

To implement the NN, we decomposed the programme into several subfunctions including the training process.[2] Here we explain the subfunctions and what they do as follows:

•**Face images preprocessing:** We use the face detector to get the face rectangles from the original images, then resize and convert them to 28*28 gray images as following:

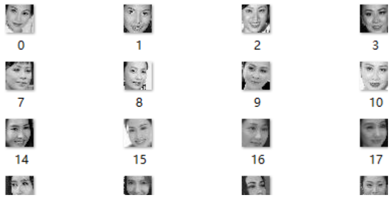

Figure 4. 28*28 gray image

•**Network initialisation:** Our neural network has 1 input layer, 1 hidden layer and 1 output layer. Before the training, we first set the network's input neuron number as 784, since we converting the 28*28 gray images into a vector as input vector $\vec{x}$ that have 784 elements, and set the output neuron number as 1, which is a number used to recognize gender.

Second, we set the activation function as $f(x) = \frac{1}{1+e^{-x}}$, and the loss funtion as

$$E = \frac{1}{2}\sum_{i=1}^{m}(d_j - y_j)^2 = \frac{1}{2}\sum_{i=1}^{m}(d_j - f(net_j^{L-1}))^2.$$

(notation: input:$x = (x_1, ..., x_n)^T$; desired outputs:$D = (d_1, ..., d_j)^T$; actual outputs:$y = (y_1, ..., y_j)^T$)

Lastly, we use a random function to initialize the weights of the network, and now our neural network is ready for the training process.

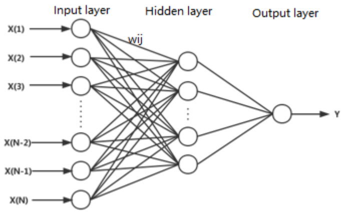

Figure 5. The neural network

•**Training:** In traing process, we input the processed gray images vector and set the desired ouput of each of them, then select a learning rate for the gradient descent optimization algorithm in the learning process and start training. By choosing different learning rate and the number of neurons in the hidden layer, we got several trained neural networks. Lets see their performance in the test section.

3. Test

After programming and training the NN, the last step is to test the trained network. We totally input 99 male images and 100 female images to several trained network, here is their performance of accuracy:

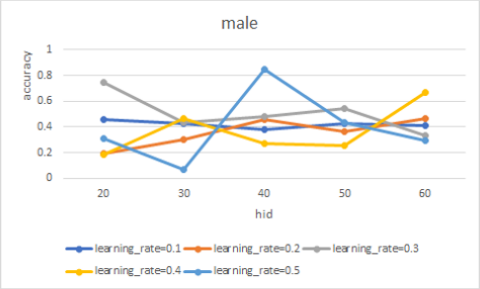

Figure 6. The NN's performance in male recognition

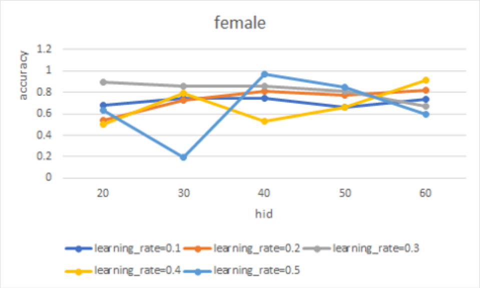

Figure 7. The NN's performance in female recognition

4. Conclusions

We successfully implement the programming and training several backpropagation-based neural networks for gender recognition from face images, and judging from the test results, them perform very well.

References

[1] Backpropagation-wikipedia. https://en.wikipedia.org/wiki/Backpropagation. 1

[2] A blog from csdn. https://blog.csdn.net/yunyunyx/article/details/80693602. 2

[3] Neural networks and backpropagation explained in a simple way. https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e. 1