UMEÅ UNIVERSITET
Institutionen för Datavetenskap
Rapport obligatorisk uppgift

Fundamentals of Artificial Intelligence

5DV121

Obligatorisk uppgift nr

2

Namn	Amine Balta, Martin Willman
E-post	id14aba@cs.umu.se, id14mwn@cs.umu.se
CAS	amba0020 mawi0214
Datum	18-10-2017
Lärare, Handledare	Thomas Johansson, Lennart Jern och Jonathan Westin

Introduction

This assignment's purpose is to build a perception based classification system that can guess the emotional state of faces that are presented as input. The four stages of facial expressions are happy, sad, mischievous and mad.

Task

A file with images and the correct answers is given so that one is able to train the program. The program should then divide these images into a set for training and a set for testing the performance of the training. The program must be able to read the files of the right format and also be able to write standard output classification values for each image id.

Material

The program is written in Java with the development environment IntelliJ IDEA. The test, training and facit files were given.

Program description

The program used the training algorithm called back-propagation. The basic idea is that it propagates an error signal for each node from output to input. When the error signal is computed for all the nodes the weights are modified accordingly.

Overview

The first thing that happens is that the program reads the three necessary files, the training and test images that are put in ArrayLists, and the facit that is put in a HashMap. Then the training begins, it works in that way that the program reads the training files and guesses the facial expressions, then it compares it to the facit file and calculates the error. It later goes through the training file again and again, a total of 400 times. By that point the weights have been updated so many times the program knows which facial expression most of the images have. Now when the network is trained the program should recognise the facial expressions, or the values of the images. And then the program tests with the testing file, here it is trained and the weight is adjusted for the program to have more qualified guesses. Then it compares the result file against the “FaceTest”-program to see how much percent of the images that it guessed correctly.

Percentron

There are four nodes or neurons as we call it in our program, one for each face expression. Each neuron has its own facial expression, and a number that symbolises it; 1 = Happy, 2 =

Sad, 3 = Mischievous and 4 = Mad. The nodes also have a fixed image size and a learning rate.

Each node or neuron creates a weight between each pixel which is a random number between 0 and 1. A bias weight is also added with the weight of 1. Then the back-propagation algorithm is used to adjust the weights between the pixels and the nodes. Back-propagation algorithm works so that the node and pixel during the training phase constantly adjusts the weight by means of which error value the pixel guesses. Since guesses are getting better and better, it also reduces errors, which means that the weight will eventually be well calibrated. This is done until the training is over.

Running the program

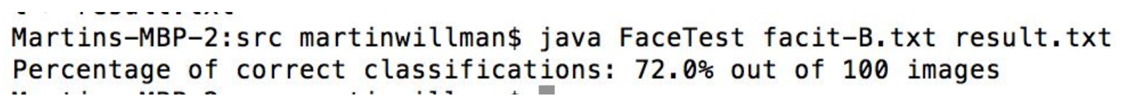
To run the program you have to download the zip file and extract. After that you open your terminal and enter this commands.

1. First you have to compile the files by using the command: *`javac *.java`*
2. Then you run it by using the command: *`java Faces training-A.txt facit-A.txt test-B.txt > result.txt`*
3. Then you can compare the result by using the command: *`java FaceTest result.txt facit-B.txt`*

Test drives

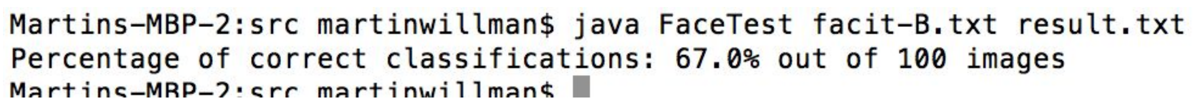
Below you can see some test drives of the `Faces program. Learning rate  = lr` and `Training rounds = tr.`

With the `lr` = 0.01 and `tr` = 400 we got the best result on around 70%.



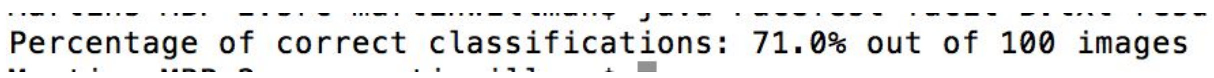
```
Martins-MBP-2:src martinwillman$ java FaceTest facit-B.txt result.txt
Percentage of correct classifications: 72.0% out of 100 images
```

With the `lr` = 0.05 and `tr` = 200 we have this result.



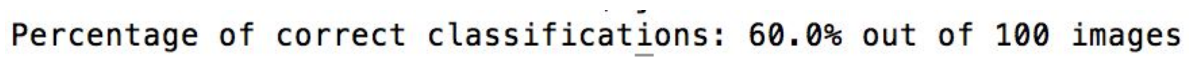
```
Martins-MBP-2:src martinwillman$ java FaceTest facit-B.txt result.txt
Percentage of correct classifications: 67.0% out of 100 images
Martins-MBP-2:src martinwillman$
```

With the `lr` = 0.1 and `tr` = 40 we have this result.



```
Percentage of correct classifications: 71.0% out of 100 images
```

With the `lr`= 0.1 and `tr` = 400 we have this result. We can see that if we have a high `learning rate` and many `training rounds` the network get overtrained and the result goes down.



```
Percentage of correct classifications: 60.0% out of 100 images
```

Discussion

Writing the program was challenging at times but overall it went smoothly, the biggest problem we faced had to do with understanding the algorithm from the beginning. But when you understand what it is exactly that you are supposed to do, the jump between actually writing it was not as big. We feel that we learned a lot in the process of working with the assignment. Overall we are happy with the result and the lessons we take from it.