Assignment 3: Neural Networks

Part I: Data description

You are provided with two Matlab cell arrays which contain the data you need in order to train your neural networks. The first cell array is called *datasetInputs* and contains the input images. The first cell contains the training data, the second cell contains the test data and the last cell contains the validation data. Each row in each cell contains one image. The size of the image has been downscaled to 30 by 30 (so the columns correspond to the 900 pixels). There are 25120, 3589 and 3589 images for training, testing and validation, respectively. You can visualise any image by executing the following commands (the following code displays the first image of the training set):

```
im = datasetInputs{1};

im1 = im(1,:);

im2 = reshape(im1,30,30);

colormap gray

imagesc(im2')
```

The second cell array is called *datasetTargets* and contains the targets for each image in the training, test and validation sets. Each row contains the target for the corresponding input image. The targets are the 6 basic emotions (angry, disgust, fear, happy, sad, surprise) + neutral. All targets are 0 except for the target that corresponds to the emotion shown in the image, e.g. if the displayed emotion is surprise then the target should be [0, 0, 0, 0, 0, 1, 0].

Part II: Creating and training a neural network

You can have a look at the exampleNN function for a complete example how to train and test a network. You should also read the manual (can be found in the Matlab toolbox we provide) which describes all the parameters of the network structure. This section briefly describes the main functions of the toolbox.

- nn = paramsNNinit(hiddenLayers, hiddenActivationFunctions);
  Initialises the network parameters. hiddenLayers is a vector containing the number of hidden neurons per layer (including the output layer), e.g., [1000 1000 1000 7], and hiddenActivationFunctions is a cell array which contains the activation functions for each layer, e.g., {'ReLu', 'ReLu', 'ReLu', 'softmax'}

- [W, biases] = initWeights(inputSize, nn.weightInitParams, hiddenLayers, hiddenActivationFunctions);
  Initialises the weights and biases of the network. inputSize is the number of inputs, nn.WeightInitParams contains the weight initialisation parameters, see above for hiddenLayers and hiddenActivationFunctions. The weights and biases need to be assigned to the network, nn.W = W and nn.biases = biases.

- [nn, Lbatch, L_train, L_val] = trainNN(nn, train_x, train_y, val_x, val_y);
  Trains a network using the training data train_x and training targets train_y. Optionally, a validation set can also be used val_x and val_y for validation inputs and targets, respectively.

IMPORTANT NOTE: Think how you should normalise the images.

IMPORTANT NOTE: Since your inputs (=pixels) are continuous you should make sure that the variable inputActivationFunction is set to 'linear'.

Part III: Parameter Optimisation

An important issue in neural networks is parameter optimisation. First, select a performance measure which will be used to compare the performance of the different parameters on the validation set. You should use stochastic gradient descent with momentum for all experiments, but you can experiment with other training algorithms if you wish so. Then you should optimise the parameters as follows:

1) First select a network architecture and an activation function that you believe is a reasonable starting point. Explain your motivation for selecting this architecture and activation function. Define a stopping criterion, a weight initialisation method and a momentum and learning rate update schedule.
2) Optimise the initial learning rate (disable regularisation). Explain how you found a good initial value for learning rate. Save the plot of the training and validation loss and training and validation classification error (from epoch 1 until the stopping criterion is met).
3) Optimise the learning rate update schedule. Include a table where you present the results on the validation set of the different learning rate schedules you have tested.
4) Use dropout and report if there is any improvement in the validation performance. Explain what changes you have made to the network and/or training procedure when you use dropout.
5) Use two other types of regularisation (any two you wish) and compare their performance with dropout. Explain which regularisation parameters you optimised and present a plot which shows the performance on the validation set as a function of the parameters that needs to be optimised. Also discuss why regularisation is needed.
6) Optimise the topology of the network, i.e. the number of hidden layers and the number of neurons in each hidden layer, the size of your input is 900 (number of pixels in the image), and the number of neurons in the output layer should be 7. Include a plot which shows the

performance on the validation set as a function of the number of layers. Include another plot which shows the performance on the validation set as function of the hidden layers size.

7) Train a network using the optimal set of parameters you have found so far and a different activation function than the one you defined in step 1. Comment on the performance on the validation set and discuss if it is any different than the initial activation function you used.
8) Train a network using the optimal set of parameters and include a figure which shows the training and validation loss and another figure which shows the training and validation classification error. Present in the same figure the curves you saved in step 2. Comment on their differences. Save also this network since you should include it in your submission.

IMPORTANT NOTE: It is impossible to run an exhaustive search, so think of a reasonable strategy when you optimise parameters (e.g. don't explore too many similar parameter values) and put more effort on optimising parameters you think are more important. Also whenever possible use some default values, e.g., start with a momentum of 0.5 and increase it (at the same epoch that learning rate begins to decrease) linearly to 0.9. Check the slides for more tips.

Part IV: Performance Estimation

Test the performance of the network trained with the optimal set of parameters on the test set and report the confusion matrix, classification rate and F1 measure per class.

Part V: Questions

1. Assume that you train a neural network classifier using the dataset of the previous coursework. You run cross-validation and you compute the classification error per fold. Let's also assume that you run a statistical test on the two sets of error observations (one from decision trees and one from neural networks) and you find that one algorithms results in better performance than the other. Can we claim that this algorithm is a better learning algorithm than the other in general? Why? Why not?

2. Suppose that we want to add some new emotions to the existing dataset. What changes should be made in decision trees and neural networks classifiers in order to include new classes?

c) Deliverable

For the completion of this part of the CBC, the following have to be submitted electronically via CATE:

1. All the code you have written.
2. The neural network you have trained in part III (step 8) in .mat format.
3. Comments in each step of part III.
4. Confusion matrix and performance measures (part IV)
5. Answers to the questions in part V.

Final Grade = 0.8* Report content + 0.1* Code Performance + 0.1* Report quality

Code Performance = 2 * CR on unseen data

Code (total : 100)

- Results on new test data : 100

  We will test your networks using the simulateNN function together with the trained network you provide. If you use python make sure you write a test function which takes as inputs the trained networks and a set of images and returns the predicted labels.

  Make sure that your code runs. If not you will be asked to resubmit the code and lose 30% of the code mark.

Report content (total : 100)

- Parameter estimation (part III),: 10 for each step, so 80 in total.
- Confusion matrix, classification rate, F_1-measure (part IV): 5
- Question 1 (part V) : 8
- Question 2 (part V): 7

Report quality (total : 100)

- Quality of presentation