Instituto Superior Técnico

Departamento de Engenharia Electrotécnica e de Computadores

Machine Learning

3rd Lab Assignment

Shift: Tuesday Group Number: 2

Number 81013 Name José António Costa Coelho

Number 81398 Name Maria Carolina Varandas Roque

Multilayer perceptrons

This assignment aims at illustrating the applications of neural networks. In the first part we'll train a multilayer perceptron (MLP) for classification and in the second part we will train a MLP for regression.

This assignment requires MatLab's Neural Network Toolbox.

1 Classification

Our classification problem is a pattern recognition one, using supervised learning. Our goal is to classify binary images of the digits from 0 to 9, with 5x5 pixels each. The following figure illustrates some of the digits.

1.1 Data

Data are organized into two matrices, the input matrix X and the target matrix T.

Each column of the input matrix represents a different pattern or digit and will therefore have 25 elements corresponding to the 25 pixels in each image. There are 560 digits in our data set.

Each corresponding column of the target matrix will have 10 elements, with the component that corresponds to the pattern's class equal to 1, and all the other components equal to -1.

Load the data and view the size of inputs X and targets T.

```
load digits
size(X)
size(T)
```

Visualize some of the digits using the function show_digit which was provided.

1.2. Neural Network

We will use a feedforward network with one hidden layer with 15 units. Network training will be performed using the gradient method in batch mode. The cost function will be the mean squared error,

$$C = \frac{1}{KP}\sum_{k=1}^{K}\sum_{i=1}^{P}\left(e_i^k\right)^2,$$

where K is the number of training patterns, P is the number of output components, and e_i^k is the ith component of the output error corresponding to the kth pattern.

```
net = patternnet([15]);
net.performFcn='mse';
```

Both the hidden and the output layer should use the hyperbolic tangent as activation function.

```
net.layers{1}.transferFcn='tansig';
net.layers{2}.transferFcn='tansig';
```

We will use the first 400 patterns for training the neural network and the remaining 160 for testing.

```
net.divideFcn='divideind';
net.divideParam.trainInd=1:400;
net.divideParam.testInd=401:560;
```

1.3. Gradient method with fixed step size parameter and momentum

(T) Describe how the minimization of a function through the gradient method, with fixed step size parameter and with momentum, is performed. Be precise. Use equations when appropriate.

Dada a variável θ e a função $J(\theta)$, pretende-se obter o valor de θ_{min} que a minimize.
Uma das técnicas para tal, é o método do momento com *step size* fixo. Considera-se um η *(learning rate)* fixo e um parâmetro α de forma a acelerar a minimização. Com este parâmetro, não se considera apenas o sentido do gradiente, mas é dado um "momento", que filtra o gradiente, deixando este de ser ortogonal às curvas de nível. Desta forma, as iterações de $\theta^{(t+1)}$ são calculadas da forma: $\theta^{(t+1)} = \theta(t) + \Delta^{(t+1)}$, onde $\Delta^{(t+1)} = \alpha\,\Delta^{(t)} - \eta\nabla_\theta J\left(\theta^{(t)}\right)$.
Este método melhora o ritmo de convergência, especialmente nos casos em que a função J contém vales profundos e alongados, no qual o método do gradiente é mais lento (apresenta mais oscilações pois é sempre ortogonal às curvas de nível, que são elipses achatadas.
A função J é avaliada em cada iteração para verificar se diminuiu como esperado. Caso tal não se verifique, a memória do termo de momento é igualada a zero.

Train the network using Gradient descent with momentum backpropagation. Initially, set the learning rate to 0.1 and the momentum parameter to 0.9. Choose, as stopping criterion, the cost function reaching a value below 0.05 or the number of iterations reaching 10000.

```
net.trainFcn = 'traingdm'
net.trainParam.lr=0.1; % learning rate
```

```
net.trainParam.mc=0.9;% Momentum constant
net.trainParam.show=10000; % # of epochs in display
net.trainParam.epochs=10000;% max epochs
net.trainParam.goal=0.05; % training goal
[net,tr] = train(net,X,T);
```

To see the evolution of the cost function (MSE) during training, click the "Performance" button.

Try to find a parameter set (step size and momentum) that approximately minimizes the training time of the network. Indicate the values that you obtained.

Step size: 4 Momentum: 0.5

Determine how many epochs it takes for the desired minimum error to be reached (execute at least five tests and compute the median of the numbers of epochs).

Median of the numbers of epochs: 60

Foram realizados 20 testes.

1.4. Gradient method with adaptive step sizes and momentum

(T) Describe how the minimization of a function through the gradient method with adaptive step sizes and momentum is performed. Be precise. Use equations when appropriate.

Dada a variável θ e a função $J(\theta)$, pretende-se obter o valor de θ_{min} que a minimize.
Uma das técnicas para tal, é o método do momento com *step size* adaptativo. Considera-se um η *(learning rate)* inicial para cada componente de θ e um parâmetro α de forma a acelerar a minimização. Os sucessivos valores de $\theta_i^{(t+1)}$ são então calculados segundo a expressão $\theta_i^{(t+1)} = \theta_i^{(t)} - \eta_i^{(t)} \frac{\partial J}{\partial \theta_i}(\theta^{(t)})$, onde $\eta_i^{(t)}$ é atualizado segundo:

$$\eta_i^{(t)} = \begin{cases} u\,\eta_i^{(t-1)}, & se \frac{\partial J}{\partial \theta_i}(\theta^{(t)}) \cdot \frac{\partial J}{\partial \theta_i}(\theta^{(t-1)}) > 0 \\ d\,\eta_i^{(t-1)}, & caso\ contrário \end{cases}$$

São dados valores típicos para u=1.2 e d=0.8.
Caso o gradiente apresente o mesmo sinal nas duas ultimas iterações, na próxima pode ser considerado um step size maior. Caso contrário (isto é, o gradiente numa iteração é positivo e noutra é negativo), está-se na vizinhança de um mínimo da função e como tal, é necessário reduzir o step size para o método se aproximar deste.
Com este método, deixa de ser tão importante a escolha de um valor inicial para η visto que o mesmo se vai adaptando à medida que as iterações são feitas. Esta técnica converge muito rapidamente para funções com vales alinhados com os eixos.

Choose as training method, gradient descent with momentum and adaptive learning rate backpropagation.

```
net.trainFcn = 'traingdx'
```

Train the network using the same initial values of the step size and momentum parameters as before. How many epochs are required to reach the desired minimum error? Make at least five tests and compute the median of the numbers of epochs.

Median of the numbers of epochs: 107
Foram realizados 20 testes.

Try to approximately find the set of parameters (initial step size and momentum) which minimizes the number of training epochs. Indicate the results that you have obtained (parameter values and median of the numbers of epochs for training). Comment on the sensitivity of the number of training epochs with respect to variations in the parameters, in comparison with the use of a fixed step size parameter. Indicate the main results that led you to your conclusions.

Initial step size: 2
Momentum: 0.9
Median of the number of epochs: 57 (using 20 tests)

Neste caso, o método não apresenta grande dependência do valor escolhido para o "step size", visto que este é apenas o valor inicial, sendo depois actualizado com base nos resultados das iterações anteriores. Assim sendo, verifica-se uma variação pouco significativa do número de "training epochs" com a variação do "initial step size", em comparação ao método anterior, o qual usa um step size fixo, e, como tal, apresenta maior dependência deste. Verifica-se, nesta situação, que a alteração do "step size" influencia fortemente o número de "training epochs".
De modo a analisar a variação do step size no método do momento com step size adaptativo, usaram-se diversos valores de "step size" numa gama de 1 a 10 com o termo de momento constante e os resultados obtidos rondaram quase sempre a mesma gama de valores, de 50 a 70 iterações, tendo, poucas vezes, alcançado valores superiores a 100. No outro método, estas variações do step size provocaram alterações significativas.
Em ambos os casos, variações no parâmetro de momento causa variações significativas no número de épocas de treino. Para os dois métodos, a diminuição do número do momento, provoca um número elevado de iterações.

Next, we'll analyze the classification quality in the test set with a confusion matrix. This matrix has 11 rows and 11 columns. The first 10 columns correspond to the target values. The first 10 rows correspond to the classifications assigned by the neural network. Each column of this 10×10 submatrix indicates how the patterns from one class were classified by the network. In the best case (if the network reaches 100% correct classification) this submatrix will be diagonal. The last row of the matrix indicates the percentage of patterns of each class that were correctly classified by the network. The global percentage of correctly classified patterns is at the bottom right cell.

```
x_test=X(:,tr.testInd);
t_test=T(:,tr.testInd);
y_test = net(x_test);
plotconfusion(t_test,y_test);
```

Comment on the values in the confusion matrix you obtained. Is the accuracy the same for every digit? Are the errors what you'd expect?

Ao analisar os valores obtidos na matriz de confusão, verifica-se que os resultados obtidos da classificação da "neural network" coincidem na maior parte das vezes com os valores à entrada (isto é, os dígitos são reconhecidos na maior parte dos casos) .
Verifica-se também que a precisão varia consoante o dígito. Por exemplo, para os dígitos 0 e 8, a precisão é sempre menor pois estes dígitos confundem-se facilmente entre eles e com outros, como o 9 ou o 6.
Em nenhuma de várias tentativas se obteve o melhor caso, ou seja, nunca se obteve a matriz diagonal, o que seria de esperar. Analisando a última linha da matriz, observa-se que a percentagem de padrões que foram correctamente avaliados rondam os 70 a 80%, na maioria dos dígitos.

Write down the performance values that you obtained:

Training error: 0.0496 Test set Accuracy: 77.5%

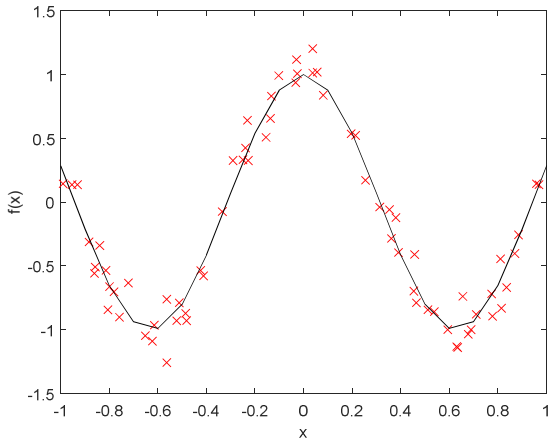
Which quantity (mean squared error, or global percentage of correct classifications) is best to evaluate the quality of networks, after training, in this problem? Why?

Neste exemplo, será melhor considerar a percentagem de classificações corretas como a forma de avaliar a qualidade da rede. Como se pode ver dos resultados anteriores, o valor obtido para o erro é muito baixo, mas ainda há falhas em diversos números, e como tal, em média, a taxa de sucesso só se aproxima dos 70-80%.
Dado o objetivo da rede ser determinar dígitos, será do nosso interesse obter uma taxa de sucesso mais elevada.

2. Regression

The goal of this second part is to estimate a function and to illustrate the use of a validation set.

The function is $f(x) = \cos(5x)$, with $x \in [-1,1]$ for which we have a small number of noisy observations $d = f(x) + \varepsilon$, in which ε is Gaussian noise with a standard deviation of 0.1. The values of x will be used as inputs to the network, and the values of d will be used as targets. The following figure shows the function $f(x)$ $(black\ line)$ and the data we will use for training (*red crosses*).



2.1 Data

Load the data in file 'regression_data.mat' and check the size of inputs X and targets T.

2.2 Neural Network

Create a network with a single hidden layer with 40 units. Set the activation of the output layer to linear (in the output layer, the 'tanh' function is more appropriate for classification problems, and the linear function is more appropriate for regression ones).

```
net = fitnet(40);
net.layers{2}.transferFcn='purelin';
```

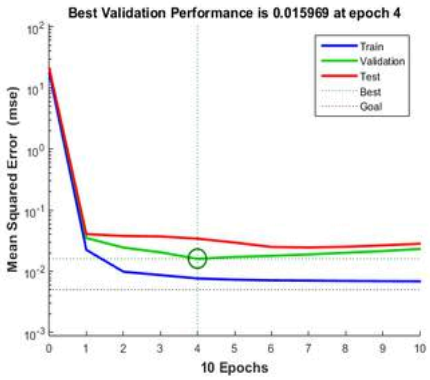
Choose 'mse' as cost function and, as stopping criterion, the cost function reaching a value below 0.005 or the number of iterations reaching 10000.

2.3 Training with a validation set

(T) When performing training with a validation set, how are the weights that correspond to the result of the training process chosen? What is the goal?

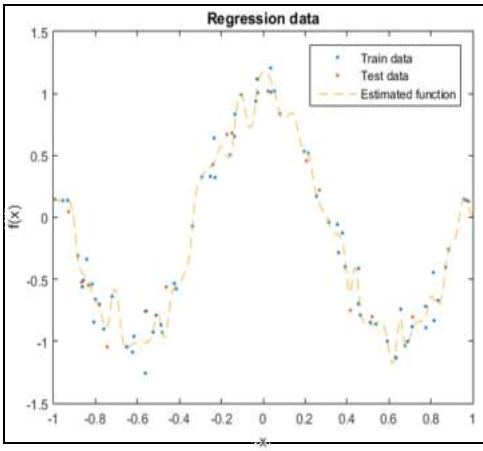
Quando se utiliza um conjunto de validação, é possível determinar valores de hiperparâmetros. Os pesos são obtidos a partir de um conjunto de etapas. Inicialmente, treina-se a rede para cada um dos valores possíveis para os hiperparâmetros e verifica-se o resultado com recurso ao conjunto de validação. Após todas as possibilidades, procura-se o valor do hiperparâmetro que corresponde ao melhor resultado.
Em seguida, utilizam-se os conjuntos de teste e validação para treinar novamente a rede, com o valor do hiperparâmetro obtido anteriormente, sendo depois o conjunto de teste utilizado para verificar o resultado da rede.
Desta forma, é possível obter parâmetros recorrendo a 3 conjuntos distintos de dados (teste, validação e treino). O objectivo final é a obtenção do melhor valor de hiperparâmetro.

Train the network using the first 70 patterns for training, the next 15 for validation and the last 15 for testing. Click the "Performance" button. Comment on what is shown in the plot.



Como se pode observar, verifica-se que o conjunto de validação é utilizado como condição para terminar o treino da rede. Verifica-se que não ocorre overfitting, pois, utilizando o conjunto de validação, obtém-se um valor baixo (assinalado na Figura) para indicar que a rede consegue analisar outros padrões, que não o de treino. Isto é, quando lhe é apresentado outro conjunto (o de validação), esta apresenta uma saída como esperado, com um erro minimizado.

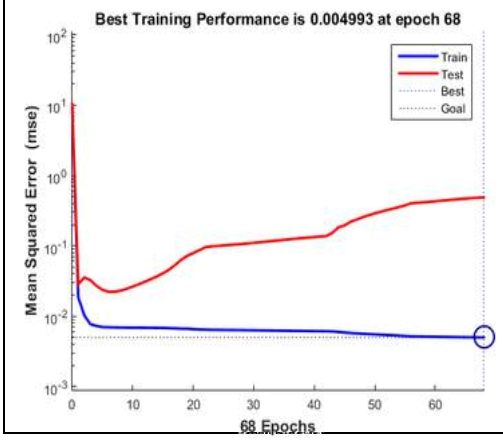
Plot the training data, the test data and the estimated function, all in the same figure and comment.



Tal como se pode observar, a função estimada apresenta uma forma aproximadamente sinusoidal, tentando aproximar-se tanto dos pontos de teste como de treino. Verifica-se que o erro de teste é inferior (ver figura acima) ao de treino, porque, por análise da figura ao lado, é possível verificar que a curva passa por vários pontos de treino, e por isso, reduz o erro obtido.

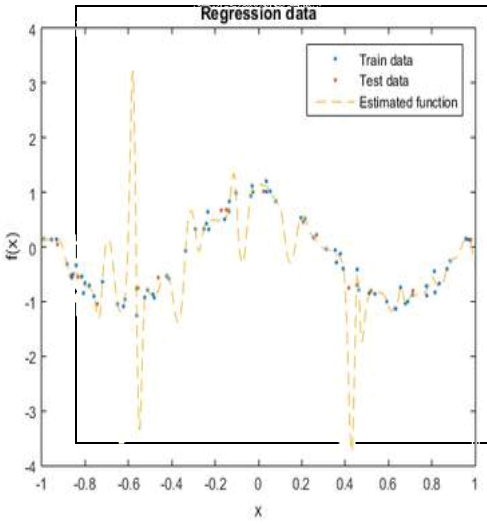
2.4 Training without a validation set

Repeat the previous item but do not use a validation set this time. Observe the evolution of the cost function for the different sets and comment. Is there any sign of overfitting?



Verifica-se que o erro diminui com os dados do conjunto de treino. Contudo, devido à ausência do conjunto de validação, é possível observar que o erro aumenta quando se considera o conjunto de teste. Há sinais de overfitting, isto é, a curva adapta-se apenas aos dados de teste, tentando tornar a regressão o mais precisa possível, mas apenas considerando os pontos de treino e não outros, como os de teste.

Plot the estimated function on the same picture as 2.3. Compare the two estimated function and comment.



É possível observar na Figura que há overfitting. Comparando esta Figura com a função estimada obtida anteriormente, esta apresenta uma curva com mais oscilações, com uma forma menos sinusoidal, devido à ocorrência de overfitting, tentando passar por todos os pontos de treino, ignorando os de teste.