Instituto Superior Técnico

Masters in Electrical and Computer Engineering

Deep Learning

Homework 1

Tiago Rebocho, Nº 93185
Gonçalo Galante, Nº 96216
Group 07

Professors: André Martins, Ben Peters, Margarida Campos

december 23, 2022

1 Question 1

In this question, we implemented different linear classifiers and neural networks for a simple image classification problem, using the Kuzushiji-MNIST dataset. The results obtained are reported below.

1.1

1.1.1 (a) Perceptron

The goal of this exercise was to implement a linear classifier, the perceptron, trained with 20 epochs. The validation and test set performance is shown below:

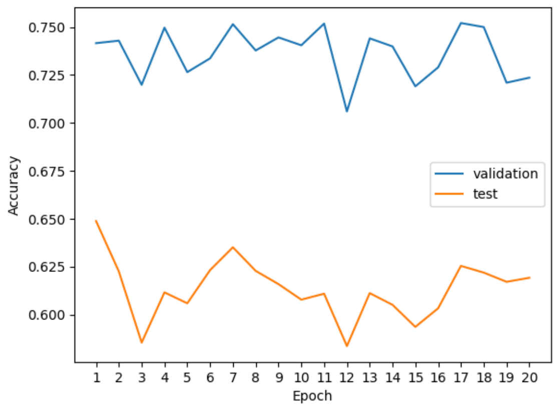


Figure 1: Perceptron accuracy on validation and test set.

As we can see in Figure 1, the accuracy value for the validation set is higher than in the test set. The performance in the validation set is approximately 72.40 % and the performance in the test set is around 62 %.

1.1.2 (b) Logistic Regression

In this one, we repeated the previous exercise but implemented the logistic regression linear classifier, using stochastic gradient descent as our training algorithm, a learning rate of 0.001 and 20 epochs. The classifier was also trained with 20 epochs and the validation and test set performance is represented below:

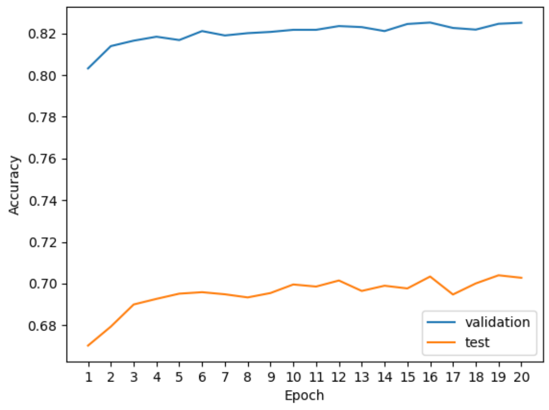


Figure 2: Logistic Regression accuracy on validation and test set.

Observing the figure above we can verify again that the accuracy on the validation set is higher than in the test set. Around 82.3% for the validation set and approx 70% for the validation set. The values are in general higher than the obtained by the perceptron classifier.

1.2

1.2.1 (a) Multi-layer Perceptron justification

There is a difference regarding the functions that a simple perceptron and an MLP can learn. The simple perceptron implemented can only learn linear functions, while the MLP can learn linear and non-linear functions. Since the image classification task is robust and there are a lot of features, it´s more advantageous to use a multi-layer perceptron to classify a set of data into different categories because it processes a high number of extracted features from the image. On the other hand, if we only use linear activation functions in the MLP, the MLP is as expressive as the single perceptron and the model will not be able to create complex mappings between the inputs and the outputs of the network.

1.2.2 (b) Multi-layer Perceptron

In this question, we implemented a Multi-layer Perceptron with a single hidden layer, using stochastic gradient descent and a learning rate of 0.001. The plot obtained of the accuracy in the validation and test set is shown below.

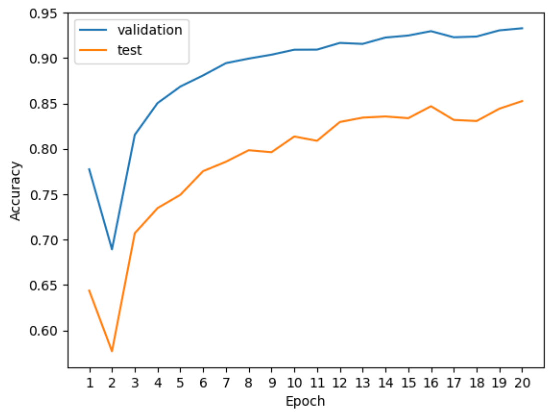


Figure 3: MLP accuracy on validation and test set.

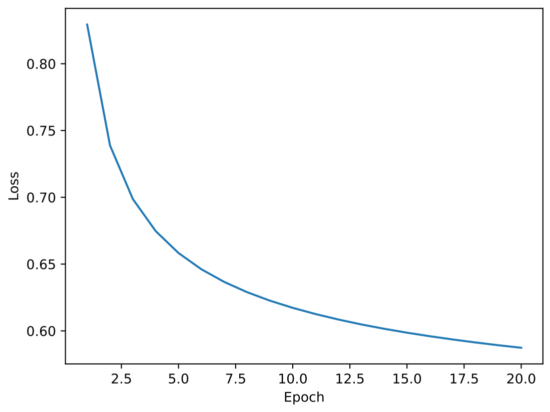
As seen in the previous questions the accuracy on the validation set, approx 93.3%, is higher than in the test set, 85.3%.

2 Question 2

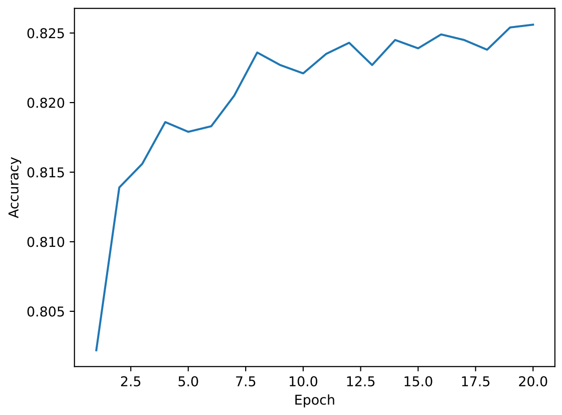
In this question, we had to implement classifiers using a deep learning framework with automatic differentiation. For that, we used PyTorch and the skeleton code provided.

2.1 Logistic Regression

In this exercise, we implemented a logistic regression, using stochastic gradient descent as our training algorithm, a batch size of 1, and 20 epochs for training. The learning rate that leads to the best results was the learning rate of 0.001. The plots obtained with this configuration for the training loss and validation accuracy are shown below.



(a) Training loss - Logistic Regression



(b) Validation accuracy - Logistic Regression

Figure 4: Plots for the Logistic Regression.

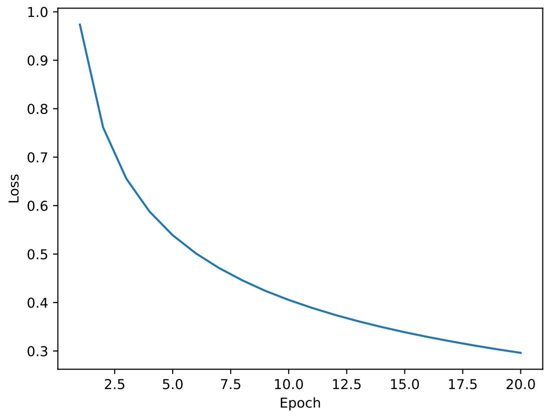
The final test accuracy obtained for this configuration is 70.19%

2.2 Feed-forward Neural Network

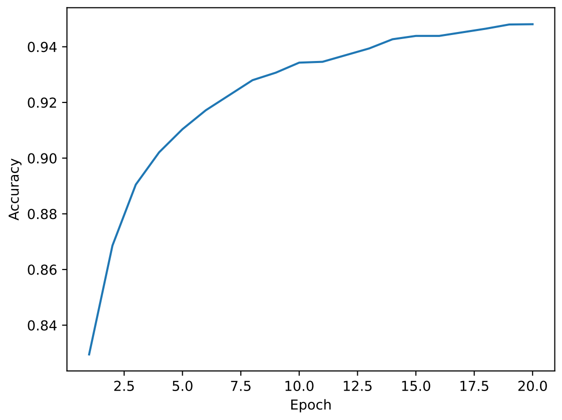
In this question, a feed-forward neural network with a single hidden layer and a dropout regularization was implemented. We had to tune these hyperparameters: learning rate, hidden size, dropout probability and the activation function, leaving the others default, in order to find the best configuration out of 6 possible. The configuration that gave us the best results is presented in the table below and the plots of the training loss and validation accuracy as a function of the epochs are shown in figures (a) and (b).

Learning rate	Hidden size	Droupout Prob	Activation function
0.01	200	0.3	ReLu

Table 1: Best configuration hyperparameters.



(a) Training loss - Feedforward NN



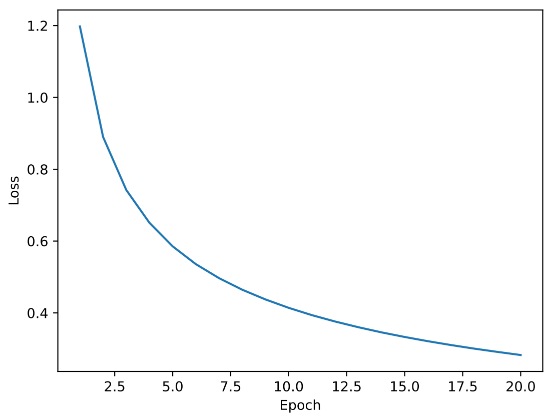
(b) Validation accuracy - Feed-forward NN

Figure 5: Plots for the Feed-forward Neural Network.

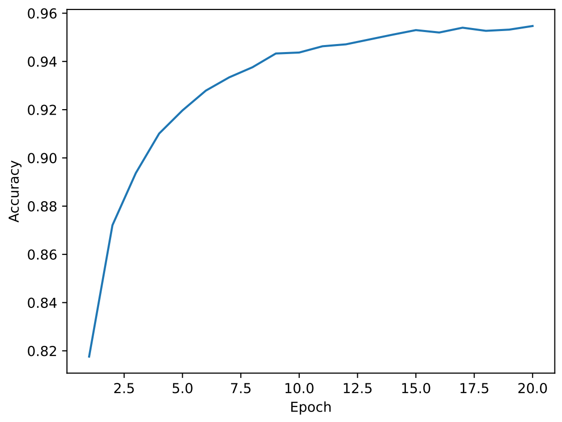
The final test accuracy obtained for this best configuration is 88.26%.

2.3 Feed-forward Neural Network with more layers

In this exercise, the number of hidden layers was increased to 2 and 3. Considering the best configuration found in the previous exercise (2.2), we had to find out which number of layers provided the greatest final accuracy. We conclude that the best number of hidden layers was 2. Below are shown the training loss and the validation accuracy of the feed-forward neural network with 2 hidden layers.



(a) Training loss - Feedforward NN 2 layers



(b) Validation accuracy - Feed-forward NN 2 layers

Figure 6: Plots for the Feed-forward Neural Network with 2 layers.

The final test accuracy obtained for this configuration is 89.45 %.

3 Question 3

3.1 Linear transformation $\boldsymbol{h}$ of $\boldsymbol{\phi(x)}$

With the quadratic activation function $g(z)$, the $\boldsymbol{W}$ matrix and the vector $\boldsymbol{x}$

$$g(z) = z^2 \tag{1}$$

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1D-1} & w_{1D} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ w_{K1} & w_{K2} & \dots & w_{KD-1} & w_{KD} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{D-1} \\ x_D \end{bmatrix}$$

if we expand $\boldsymbol{h}$ we get:

$$\boldsymbol{h} = \boldsymbol{g}(\boldsymbol{Wx}) = (\boldsymbol{Wx})^{\circ 2} = \begin{bmatrix} (x_1w_{11} + x_2w_{12} + ... + x_{D-1}w_{1D-1} + x_Dw_{1D})^2 \\ \vdots \\ (x_1w_{K1} + x_2w_{K2} + ... + x_{D-1}w_{KD-1} + x_Dw_{KD})^2 \end{bmatrix} =$$

$$= \begin{bmatrix} (\sum_{j=1}^{D} x_jw_{1j})^2 \\ \vdots \\ (\sum_{j=1}^{D} x_jw_{Kj})^2 \end{bmatrix} = \begin{bmatrix} 2\sum_{i<j}^{D} x_iw_{1i}x_jw_{1j} + \sum_{j=1}^{D} x_j^2w_{1j}^2 \\ \\ 2\sum_{i<j}^{D} x_iw_{Ki}x_jw_{Kj} + \sum_{j=1}^{D} x_j^2w_{1j}^2 \end{bmatrix} =$$

$$= \begin{bmatrix} 2w_{11}w_{12} & \dots & 2w_{11}w_{1D} & 2w_{12}w_{13} & \dots & 2w_{12}w_{1D} & \dots & \dots & 2w_{1D-1}w_D & w_{11}^2 & \dots & w_{1D}^2 \\ \vdots & & \vdots & \vdots & & \vdots & & & \vdots & \vdots & & \vdots \\ 2w_{K1}w_{K2} & \dots & 2w_{K1}w_{KD} & 2w_{K2}w_{K3} & \dots & 2w_{K2}w_{KD} & \dots & \dots & 2w_{KD-1}w_D & w_{K1}^2 & \dots & w_{KD}^2 \end{bmatrix} \begin{bmatrix} x_1x_2 \\ \vdots \\ x_1x_D \\ x_2x_3 \\ \vdots \\ x_2x_D \\ \vdots \\ \vdots \\ x_{D-1}x_D \\ x_1^2 \\ \vdots \\ x_D^2 \end{bmatrix} \tag{2}$$

where the first matrix is $\boldsymbol{A_\Theta} \in \mathbb{R}^{K\times\frac{D(D+1)}{2}}$ and the column vector is $\boldsymbol{\phi(x)} \in \mathbb{R}^{\frac{D(D+1)}{2}}$. So, it is possible to say that $\boldsymbol{h}$ is a linear transformation of $\boldsymbol{\phi(x)}$. The dimention $\frac{D(D+1)}{2}$ comes from an aritmetic sum (4). The mapping ϕ is defined in (3).

$$\phi : \begin{bmatrix} x_1 \\ \vdots \\ x_D \end{bmatrix} \longrightarrow \begin{array}{c|l} \left[\begin{matrix} x_1x_2 \\ \vdots \\ x_1x_D \\ x_2x_3 \\ \vdots \\ x_2x_D \\ \vdots \\ \vdots \\ x_{D-1}x_D \\ x_1^2 \\ \vdots \\ x_D^2 \end{matrix}\right] & \begin{matrix} \\ D-1 \\ \\ \\ D-2 \\ \\ \\ \vdots \\ \\ 1 \\ \\ D \\ \\ \end{matrix} \end{array} \tag{3}$$

$$S = D + D - 1 + D - 2 + ... + 1 = \frac{D(D+1)}{2} \tag{4}$$

3.2 Linear transformation $\hat{y}$ of $\boldsymbol{\phi(x)}$

$\hat{y}$ is a linear transformation of $\boldsymbol{\phi(x)}$ by (5)

$$\hat{y} = \boldsymbol{v}^\top \boldsymbol{h} = \boldsymbol{v}^\top \boldsymbol{A_\Theta \phi(x)} = \boldsymbol{c_\Theta^\top \phi(x)} \tag{5}$$

$$\boldsymbol{c_\Theta^\top} = \boldsymbol{v}^\top \boldsymbol{A_\Theta} \tag{6}$$

The equation (6) shows that $\hat{y}$ is a linear transformation of $\boldsymbol{\phi(x)}$. We can also conclude that it is not a linear model in terms of Θ because $\boldsymbol{A_\Theta}$ depends on model parameters but the relation is not linear. This fact is observed in the entries of the elements of the matrix $\boldsymbol{A_\Theta}$.

3.3 Parametrize the model with $\boldsymbol{c_\Theta}$ instead of $\boldsymbol{c}$

To show that for any real vector $\boldsymbol{c} \in \mathbb{R}^{\frac{D(D+1)}{2}}$ there is a choice of the original parameters $\boldsymbol{\Theta = (W, v)}$ such that $\boldsymbol{c_\Theta = c}$ we compute:

$$\boldsymbol{c} = (\boldsymbol{c_\Theta^\top})^\top = (\boldsymbol{v}^\top \boldsymbol{A_\Theta})^\top = \boldsymbol{A_\Theta^\top v} =$$

$$
= \begin{bmatrix} 2w_{11}w_{12} & \dots & 2w_{K1}w_{K2} \\ \vdots & & \vdots \\ 2w_{11}w_{1D} & \dots & 2w_{K1}w_{KD} \\ 2w_{12}w_{13} & \dots & 2w_{K2}w_{K3} \\ \vdots & & \vdots \\ 2w_{12}w_{1D} & \dots & 2w_{K2}w_{KD} \\ \vdots & & \vdots \\ \vdots & & \vdots \\ 2w_{1D-1}w_{1D} & \dots & 2w_{KD-1}w_{KD} \\ w_{11}^2 & \dots & w_{K1}^2 \\ \vdots & & \vdots \\ w_{1D}^2 & \dots & w_{DD}^2 \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_k \end{bmatrix} = \begin{bmatrix} 2\sum_{j=1}^K v_j w_{j1} w_{j2} \\ \vdots \\ 2\sum_{j=1}^K v_j w_{j1} w_{jD} \\ 2\sum_{j=1}^K v_j w_{j2} w_{j3} \\ \vdots \\ 2\sum_{j=1}^K v_j w_{j2} w_{jD} \\ \vdots \\ \vdots \\ 2\sum_{j=1}^K v_j w_{jD-1} w_{jD} \\ \sum_{j=1}^K v_j w_{j1}^2 \\ \vdots \\ \sum_{j=1}^K v_j w_{jD}^2 \end{bmatrix}, \text{ with } K \geq D \tag{7}
$$

where we can conclude from (7) that $\boldsymbol{c} = \boldsymbol{c_\Theta}$. This means that the model can be equivalently parametrized with $\boldsymbol{c_\Theta}$ instead of $\boldsymbol{\Theta}$. Also, we can possibly check that it can be a linear model in terms of $\boldsymbol{c_\Theta}$, however, this is not necessarily true. Even though the model can be parametrized with $\boldsymbol{c_\Theta}$ instead of $\boldsymbol{\Theta}$, generally, the linearity of the model doesn't depend on the choice of parameters, but rather on how the output depends on the inputs, i.e, since the model still includes a non-linear activation function, the quadratic activation function, the output and $\boldsymbol{c_\Theta}$ relation is not linear. Therefore, just because the model can be equivalently parametrized with $\boldsymbol{c_\Theta}$ does not necessarily mean that it is a linear model.

For the case where $K<D$, it may not be possible to find a choice of $\boldsymbol{\Theta}$ such that $\boldsymbol{c_\Theta} = \boldsymbol{\Theta}$. This happens because the nº of parameters in $\boldsymbol{\Theta}$ will exceed the nº of parameters in $\boldsymbol{c_\Theta}$. The same can be verified by the minimal difference between K and D, when the value of K is assigned as D-1, which makes it impossible to find an $\boldsymbol{A_\Theta}$ with K rows that can produce $\boldsymbol{c_\Theta}$ when, for example, $\boldsymbol{c}$ has more elements than K. If K = 3 and D = 4, we can see that $\boldsymbol{c} \in \mathbb{R}^{\frac{4(4+1)}{2}} = \boldsymbol{c} \in \mathbb{R}^{10}$, $\boldsymbol{c_\Theta} \in \mathbb{R}^{\frac{4(4+1)}{2}} = \boldsymbol{c} \in \mathbb{R}^{10}$ and $\boldsymbol{v} \in \mathbb{R}^3$, where we need to find out a $\boldsymbol{A_\Theta} \in \mathbb{R}^{3X\frac{4(4+1)}{2}} = \boldsymbol{A_\Theta} \in \mathbb{R}^{3X10}$ which does not exist in order to produce $\boldsymbol{c_\Theta}$.

3.4 Closed formed solution of $\boldsymbol{c_\Theta}$

$$
L(c_\Theta; \mathcal{D}) = \frac{1}{2}\sum_{n=1}^N (\hat{y_n}(x_n : c_\Theta) - y_n)^2 = \frac{1}{2}\sum_{n=1}^N (y_n - \hat{y_n}(x_n : c_\Theta))^2 =
$$

$$
= \frac{1}{2}\sum_{n=1}^N (y_n - c_\Theta^\top \phi(x))^2 \tag{8}
$$

The Loss in (8) is in the same form as in Linear Regression, then the closed form solution of $\hat{c_\Theta}$ is the same as solving the least squares problem and is given by:

$$
\hat{c_\Theta} = (X^\top X)^{-1} X^\top Y \tag{9}
$$

with

$$X = \begin{bmatrix} \phi(x_1)^\top \\ \vdots \\ \phi(x_n)^\top \\ \vdots \\ \phi(x_N)^\top \end{bmatrix} \; Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ \vdots \\ y_N \end{bmatrix}$$

Usually, the global minimization is intractable for feedforward neural networks because there is no closed form solution and the weights must be trained instead. This problem is special because it has a closed form solution.

4 Contribution of each member

The project was developed and implemented with the mutual contribution of each element, on each question. We did it all together.