TÉCNICO LISBOA

Homework 1

Deep Learning

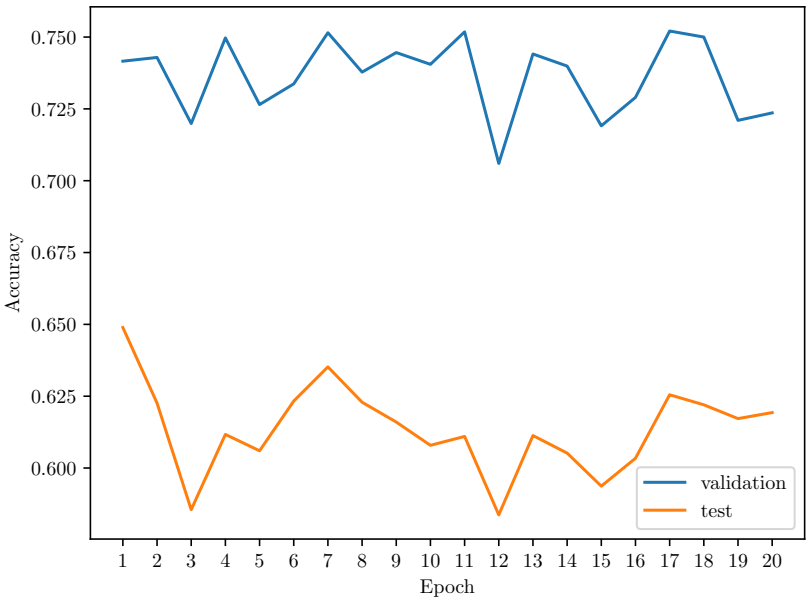
Duarte Calado de Almeida
95565

André Lopes Rodrigues
96576

The contribution of each member was as follows: André Rodrigues did Questions 1.1 and 2, while Duarte Almeida did Questions 1.2 and 3. The elaboration of this report was made in colaboration by both students.

Question 1

1. (a) After implementing the `update_weights` method, the perceptron was trained for 20 epochs, and the validation and test accuracies were plotted as follows:

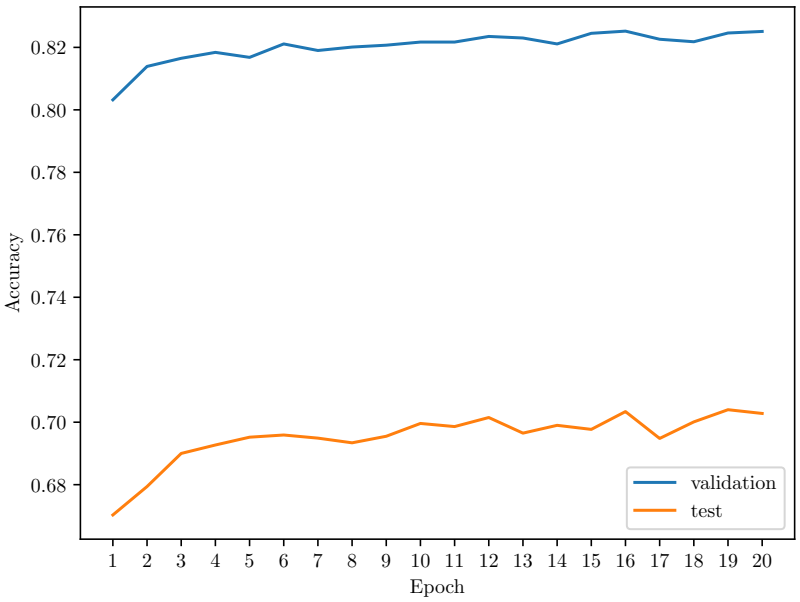


The final validation and test accuracies were also obtained,

Final Validation Accuracy	0.7236
Final Test Accuracy	0.6193

(b) For the logistic regression, the correspondent `update_weights` method was also obtained, and using a learning rate of $\eta = 0.001$, the model was trained for 20 epochs, having the accuracies for the validation and test set also been plotted:

The final validation and test accuracies obtained are as follows,

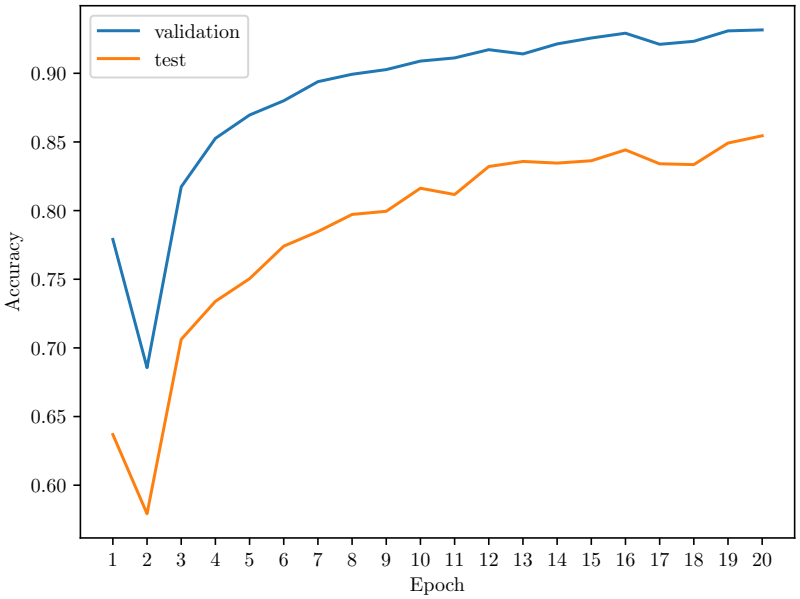


Final Validation Accuracy	0.8251
Final Test Accuracy	0.7028

2. (a) The use of multilayer perceptrons provides a form of **representation learning** through the composition of multiclass perceptrons with non-linear activation functions; in particular, they are able to search through some subspace of feature transformations and find one for which the linear separation of data is easier. In practical terms, this layering provides **hierarchical compositionality**, i.e., each layer encodes a distributed representation that is more abstract the closer it is to the output layer, finding meaningful features at different levels of granularity.
In the context of character recognition, MLPs are usually able to successively detect features of symbols (i.e., edges and parts), combine them in coarser features and perform an accurate classification at the end. This is usually not the case if we use simple perceptrons, as plain linear combinations of grayscale values are less capable of detecting those distinctive features and of being flexible to the variability that the same character may show.
Nevertheless, an MLP that uses linear activation functions ($g(z) = z$) **does not display this feature**, as it corresponds to a linear classifier, effectively functioning like a simple perceptron:

$$
\begin{aligned}
\boldsymbol{f}(\boldsymbol{x}) &= g(\boldsymbol{z}^{(L+1)}) = \boldsymbol{z}^{(L+1)} \\
&= \boldsymbol{W}^{(L+1)}\boldsymbol{h}^{(L)} + \boldsymbol{b}^{(L+1)} \\
&= \boldsymbol{W}^{(L+1)}(\boldsymbol{W}^{(L)}\boldsymbol{h}^{(L-1)} + \boldsymbol{b}^{(L)}) + \boldsymbol{b}^{(L+1)} \\
&\quad \dots \\
&= \boldsymbol{W}^{(L+1)}(\boldsymbol{W}^{(L)}(\dots\ (\boldsymbol{W}^{(1)}\boldsymbol{x} + \boldsymbol{b}^{(1)})) + \boldsymbol{b}^{(L)}) + \boldsymbol{b}^{(L+1)} \\
&= \underbrace{\boldsymbol{W}^{(L+1)}\dots\boldsymbol{W}^{(1)}}_{\widetilde{\boldsymbol{W}}}\boldsymbol{x} + \underbrace{\boldsymbol{W}^{(L+1)}\dots\boldsymbol{W}^{(2)}\boldsymbol{b}^{(1)} + \dots + \boldsymbol{W}^{(L+1)}\boldsymbol{b}^{(L)} + \boldsymbol{b}^{(L+1)}}_{\widetilde{\boldsymbol{b}}}
\end{aligned}
$$

(b) Having implemented the code for the multi-layer perceptron (with a single hidden layer) and its backpropagation algorithm, the model was built with 200 hidden units, **`ReLU`** as the activation function, and cross-entropy as the loss criterion. Stochastic gradient descent was then used to train the model for 20 epochs, with a learning rate of $\eta = 0.001$. The weights were initialized with a normal distribution $w_{ij} \sim \mathcal{N}(\mu, \sigma^2)$, with mean $\mu = 0.1$ and variance $\sigma^2 = 0.1$ and bias set to zero. The accuracies for the validation and test sets were then plotted:



The final validation and test accuracies were also obtained,

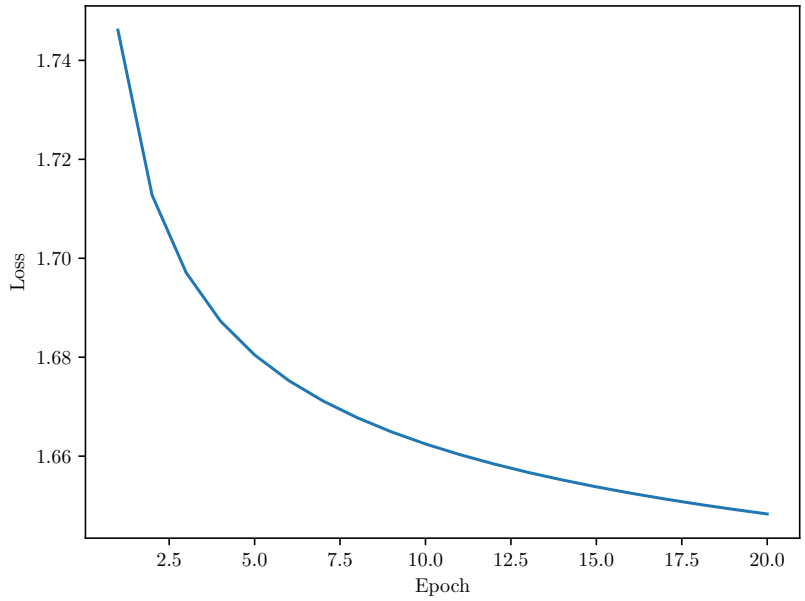
Final Validation Accuracy	0.9316
Final Test Accuracy	0.8545

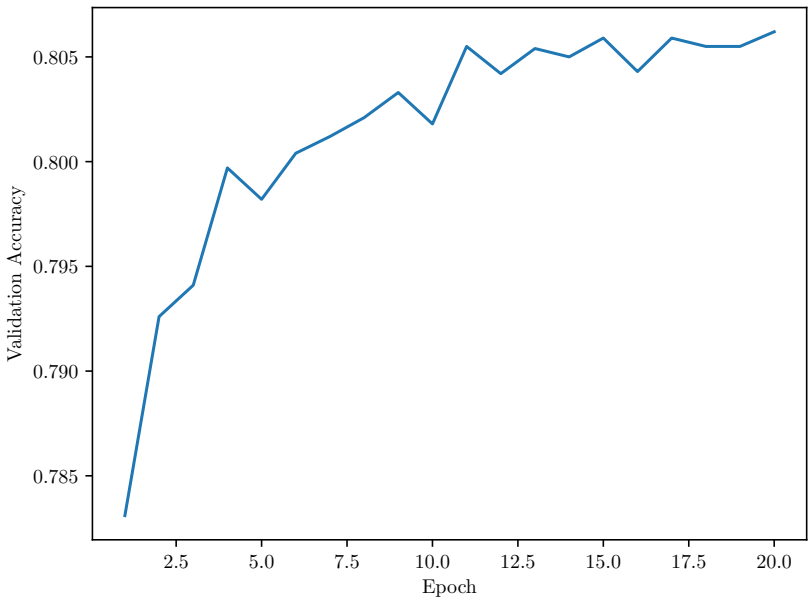
Question 2

1. For the linear model with logistic regression, the best configuration in terms of **validation accuracy** is the one with a learning rate of $\eta =$ **0.01**, as shown in the table below. The final **test error** in that configuration is 0.6641.

	learning rate		
	0.001	0.01	0.1
validation accuracy	0.795 / 0.6504	0.8062/0.6641	0.7947 / 0.6331

The plots of the loss and validation accuracy over the number of epochs, respectively, for the learning rate of $\eta = 0.01$ can be found below.

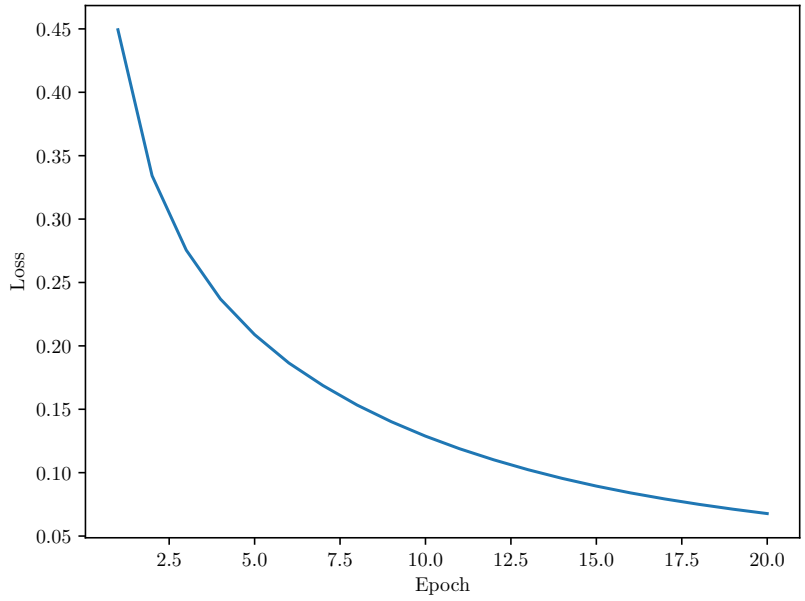


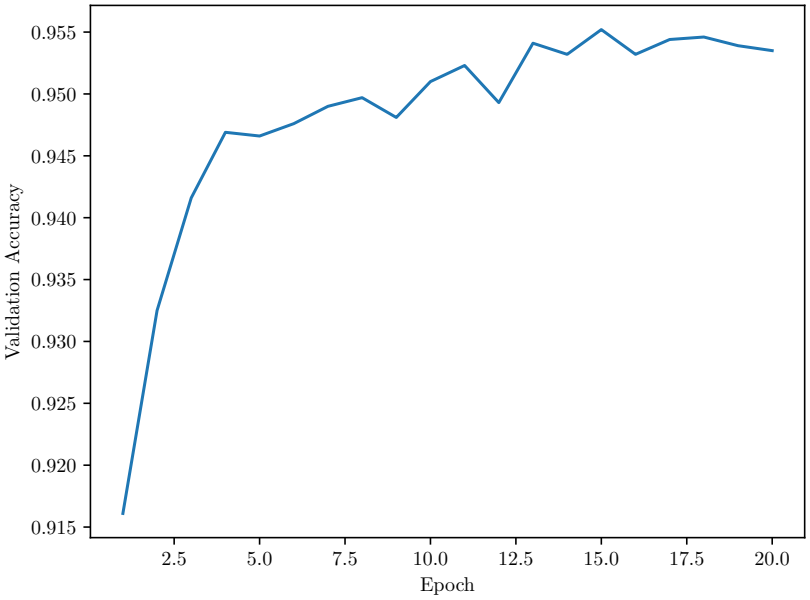


2. The best configuration for the feedforward neural network in terms of **validation accuracy** is the one that maintains the default configuration but that uses a learning rate of $\eta = \mathbf{0.1}$, as shown in the table below. The final **test error** in that configuration is 0.8932.

	learning rate			hidden size		dropout		activation	
	0.001	0.01	0.1	100	200	0.3	0.5	relu	tanh
validation accuracy	0.8619	0.9431	0.9535	0.9431	0.9498	0.9498	0.9431	0.9431	0.9410

Below are the plots for respectively, the loss and the validation accuracy over the number of epochs, for a learning rate of $\eta = 0.1$.

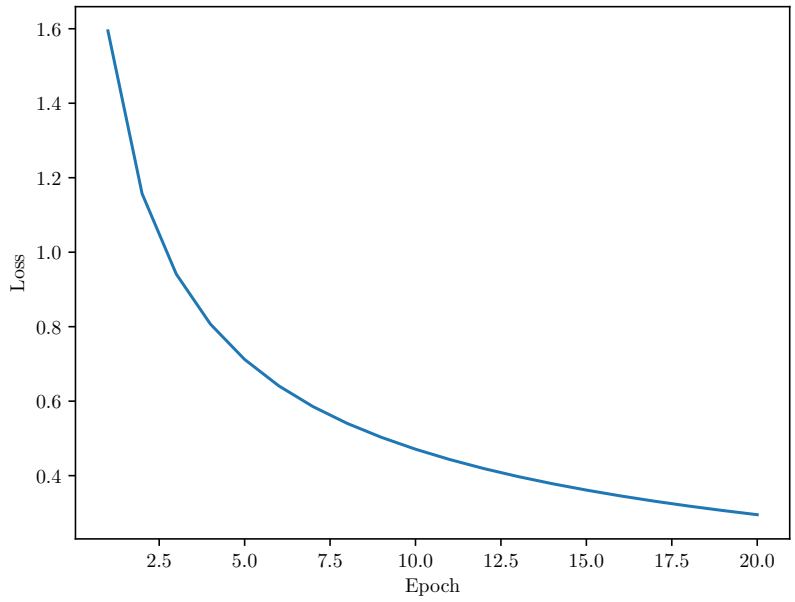


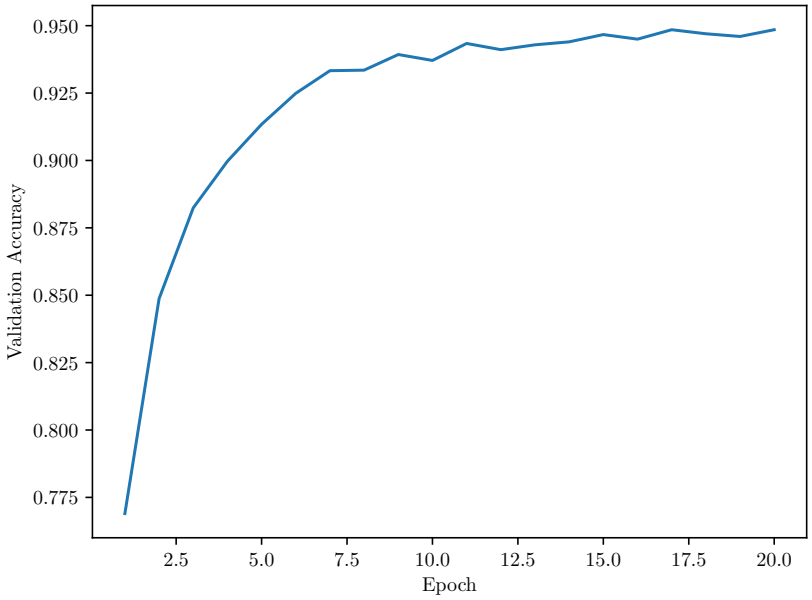


3. Using the default hyperparameters, the number of hidden layers was increased and it was found that the best configuration in terms of **validation accuracy** is the one with **3** hidden layers, as shown in the table below. The final **test error** in that configuration is 0.8682.

	learning rate	
	2	3
validation accuracy	0.9476	0.9485,

The plots of the loss and validation accuracy of the model with 3 hidden layers are as follows,





Question 3

1. Let x_i denote the i-th component of $\boldsymbol{x}$ and let w_{ij} denote the entry in the i-th row and j-th column in matrix $\boldsymbol{W}$. We then have that:

$$
\begin{aligned}
h_i(\boldsymbol{x}) &= g\left(\sum_{j=1}^{D} w_{ij}x_j\right) = \left(\sum_{j=1}^{D} w_{ij}x_j\right)^2 = \left(\sum_{j=1}^{D} w_{ij}x_j\right)\left(\sum_{k=1}^{D} w_{ik}x_k\right) \\
&= \sum_{j=1}^{D}\sum_{k=1}^{D} w_{ij}x_j w_{ik}x_k = \sum_{j=1}^{D} w_{ij}^2 x_j^2 + \sum_{j=1}^{D}\sum_{k\neq j} w_{ij}w_{ik}x_j x_k \\
&= \sum_{j=1}^{D} w_{ij}^2 x_j^2 + \sum_{j=1}^{D}\sum_{k=1}^{j-1} w_{ij}w_{ik}x_j x_k + \sum_{j=1}^{D}\sum_{k=j+1}^{D} w_{ij}w_{ik}x_j x_k \\
&= \sum_{j=1}^{D} w_{ij}^2 x_j^2 + \sum_{k=1}^{D}\sum_{j=k+1}^{D} w_{ij}w_{ik}x_j x_k + \sum_{j=1}^{D}\sum_{k=j+1}^{D} w_{ij}w_{ik}x_j x_k \\
&= \sum_{j=1}^{D} w_{ij}^2 x_j^2 + \sum_{k=1}^{D}\sum_{j=k+1}^{D} w_{ij}w_{ik}(2x_j x_k) \\
&= \begin{bmatrix} w_{i1}^2 & w_{i1}w_{i2} & \dots & w_{i1}w_{iD} & w_{i2}^2 & w_{i2}w_{i3} & \dots & w_{i(D-1)}^2 & w_{i(D-1)}w_{iD} & w_{iD}^2 \end{bmatrix} \begin{bmatrix} x_1^2 \\ 2x_1x_2 \\ \dots \\ 2x_1x_D \\ x_2^2 \\ 2x_2x_3 \\ \dots \\ x_{D-1}^2 \\ 2x_{D-1}x_D \\ x_D^2 \end{bmatrix}
\end{aligned}
$$

   As such, $\boldsymbol{h}$ is linear in some feature transformation $\boldsymbol{\phi}$, that is, $\boldsymbol{h}$ can be written as $\boldsymbol{A}_\Theta\boldsymbol{\phi}(\boldsymbol{x})$. In particular, we have that such matrix $\boldsymbol{A}_\Theta$ can be defined as:

$$
\boldsymbol{A}_\Theta = \begin{bmatrix} — & \boldsymbol{a}_1^T & — \\ — & \boldsymbol{a}_2^T & — \\ & \vdots & \\ — & \boldsymbol{a}_K^T & — \end{bmatrix}
$$

   where

$$
\boldsymbol{a}_i = \begin{bmatrix} w_{i1}^2 & w_{i1}w_{i2} & \dots & w_{i1}w_{iD} & w_{i2}^2 & w_{i2}w_{i3} & \dots & w_{i(D-1)}^2 & w_{i(D-1)}w_{iD} & w_{iD}^2 \end{bmatrix}^T
$$

   and so $\boldsymbol{A}_\Theta \in \mathbb{R}^{K\times\frac{D(D+1)}{2}}$ (since $\sum_{k=1}^{D} k = \frac{D(D+1)}{2}$). Furthermore, we can define the feature transformation $\boldsymbol{\phi} : \mathbb{R}^D \mapsto \mathbb{R}^{\frac{D(D+1)}{2}}$ as:

$$
\boldsymbol{\phi}(\boldsymbol{x}) = (x_1^2, 2x_1x_2, \dots, 2x_1x_D, x_2^2, 2x_2x_3, \dots, x_{D-1}^2, 2x_{D-1}x_D, x_D^2)
$$

2. Given that the predicted output $\hat{y}$ is defined as:

$$
\hat{y} = \boldsymbol{v}^T\boldsymbol{h}
$$

the linearity of $\boldsymbol{h}$ in the feature transformation $\boldsymbol{\phi}(\boldsymbol{x})$ proven above leads to following equality:

$$\hat{y} = \boldsymbol{v}^T \boldsymbol{A}_\Theta \boldsymbol{\phi}(\boldsymbol{x}) = (\boldsymbol{A}_\Theta^T \boldsymbol{v})^T \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{c}_\Theta^T \boldsymbol{\phi}(\boldsymbol{x})$$

where we take $\boldsymbol{c}_\Theta$ to be equal to $\boldsymbol{A}_\Theta^T \boldsymbol{v}$, thereby proving that $\hat{y}$ is also a linear transformation of $\boldsymbol{\phi}(\boldsymbol{x})$. However, $\hat{y}$ is **not** linear in terms of the original parameters Θ. To see this, note that the model is now a linear combination of **products** of entries of $\boldsymbol{W}$ and $\boldsymbol{v}$ rather than being linear in **each** individual entry:

$$\hat{y} = \boldsymbol{v}^T \boldsymbol{A}_\Theta \boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{D} v_i (\boldsymbol{w}_i^T \boldsymbol{x})^2 = \sum_{i=1}^{K} \sum_{j=1}^{D} \sum_{k=1}^{D} v_i w_{ij} w_{ik} x_i x_k$$

where we define $\boldsymbol{w}_i$ to be the vector in the i-th row of matrix $\boldsymbol{W}$.

3. To prove the desired result, for $\boldsymbol{c}_\Theta$ defined in the previous subquestion and for any $\boldsymbol{c} \in \mathbb{R}^{\frac{D(D+1)}{2}}$, we make the observation that the inner products $\boldsymbol{c}_\Theta^T \boldsymbol{\phi}(\boldsymbol{x})$ and $\boldsymbol{c}^T \boldsymbol{\phi}(\boldsymbol{x})$ actually correspond to quadratic forms in $\boldsymbol{x}$:

$$\begin{aligned}
\boldsymbol{c}_\Theta^T \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{v}^T \boldsymbol{h} &= \sum_{i=1}^{K} v_i (\boldsymbol{A}_\Theta \boldsymbol{\phi}(\boldsymbol{x}))_i^2 = \sum_{i=1}^{K} v_i (\boldsymbol{w}_i^T \boldsymbol{x})^2 \\
&= \begin{bmatrix} \boldsymbol{w}_1^T \boldsymbol{x} & \boldsymbol{w}_2^T \boldsymbol{x} & \dots & \boldsymbol{w}_K^T \boldsymbol{x} \end{bmatrix} \operatorname{diag}(\boldsymbol{v}) \begin{bmatrix} \boldsymbol{w}_1^T \boldsymbol{x} \\ \boldsymbol{w}_2^T \boldsymbol{x} \\ \dots \\ \boldsymbol{w}_K^T \boldsymbol{x} \end{bmatrix} \\
&= (\boldsymbol{W}\boldsymbol{x})^T \operatorname{diag}(\boldsymbol{v}) \boldsymbol{W}\boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{W}^T \operatorname{diag}(\boldsymbol{v}) \boldsymbol{W}\boldsymbol{x}
\end{aligned}$$

and

$$\begin{aligned}
\boldsymbol{c}^T \boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{\frac{D(D+1)}{2}} c_i \phi_i(x) &= \sum_{i=1}^{D} c_{(i-1)D+i-\frac{(i-1)i}{2}} \; x_i^2 + \sum_{i=1}^{D} \sum_{j=i+1}^{D} c_{(i-1)D+j-\frac{(j-1)j}{2}} \; (2 x_i x_j) \\
&= \boldsymbol{x}^T \mathcal{M}(\boldsymbol{c}) \boldsymbol{x}
\end{aligned}$$

where $\mathcal{M}(\boldsymbol{c}) \in \mathbb{R}^{D \times D}$ is a symmetric matrix obtained from $\boldsymbol{c}$ such that:

- the diagonal and the part above the diagonal of the matrix $\mathcal{M}(\boldsymbol{c})$ is filled row-wise with the elements of vector $\boldsymbol{c}$, i.e.:

$$\mathcal{M}(\boldsymbol{c}) = \begin{bmatrix} c_1 & c_2 & c_3 & \dots & c_D \\ c_2 & c_{D+1} & c_{D+2} & \dots & c_{2D-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_D & c_{D-1} & c_{D-2} & \dots & c_{\frac{D(D+1)}{2}} \end{bmatrix}$$

- for $1 \leq i \leq D$ and $j < i$, $(\mathcal{M}(\boldsymbol{c}))_{ij} = (\mathcal{M}(\boldsymbol{c}))_{ji}$

Furthermore, we also recur to the following lemma:

Lemma 1. *Two vectors $\boldsymbol{a}$, $\boldsymbol{b} \in \mathbb{R}^{\frac{D(D+1)}{2}}$ are equal if and only if $\boldsymbol{a}^T \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{b}^T \boldsymbol{\phi}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathbb{R}^D$*

Proof. If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{a}^T \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{b}^T \boldsymbol{\phi}(\boldsymbol{x})$ is trivially verified. For the reverse implication, note that, according to the previously made observation, for any $\boldsymbol{x} \in \mathbb{R}^D$:

$$\boldsymbol{a}^T \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{b}^T \boldsymbol{\phi}(\boldsymbol{x}) \Rightarrow \boldsymbol{x}^T \mathcal{M}(\boldsymbol{a}) \boldsymbol{x} = \boldsymbol{x}^T \mathcal{M}(\boldsymbol{b}) \boldsymbol{x}$$

Since both $\mathcal{M}(\boldsymbol{a})$ and $\mathcal{M}(\boldsymbol{a})$ are symmetric and the associated quadratic forms are twice continuously differentiable, taking the hessian on both sides of the equation yields:

$$\mathcal{M}(\boldsymbol{a}) = \mathcal{M}(\boldsymbol{b})$$

that is, $\mathcal{M}(\boldsymbol{a})$ and $\mathcal{M}(\boldsymbol{b})$ are equal entry-wise. In particular, we have that $a_i = b_i$, for $i = 1, \ldots, \frac{D(D+1)}{2}$. □

We are now equipped with the tools needed for the proof. Since $\mathcal{M}(\boldsymbol{c})$ is symmetric, the **Spectral Decomposition Theorem** tells us that there is an orthonormal matrix $\boldsymbol{Q}$ and a diagonal matrix $\boldsymbol{\Lambda}$ such that $\mathcal{M}(\boldsymbol{c}) = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$. Let $\boldsymbol{q_i}$ denote the eigenvector of $\mathcal{M}(\boldsymbol{c})$ that is present in i-th column of $\boldsymbol{Q}$ and let λ_i be the corresponding eigenvalue (note that $\{\boldsymbol{q}_i\}_{i=1}^D$ forms an orthonormal basis of $\mathbb{R}^D$). Then, we can write $\boldsymbol{c}^T\boldsymbol{\phi}(\boldsymbol{x})$ as:

$$\boldsymbol{c}^T\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}^T\mathcal{M}(\boldsymbol{c})\boldsymbol{x} = (\boldsymbol{Q}^T\boldsymbol{x})^T\boldsymbol{\Lambda}(\boldsymbol{Q}^T\boldsymbol{x}) = \sum_{i=1}^{D}\lambda_i(\boldsymbol{q}_i^T\boldsymbol{x})^2$$

Now, if we assume that $K \geq D$, we can find a matrix $\boldsymbol{W}$ and a vector $\boldsymbol{v}$ that make $\boldsymbol{c}_\Theta^T\boldsymbol{\phi}(\boldsymbol{x})$ equal to $\boldsymbol{c}^T\boldsymbol{\phi}(\boldsymbol{x})$ in the following way:

- we make $\boldsymbol{v}$ to be equal to $(\lambda_1, \lambda_2, \ldots, \lambda_D, \underbrace{0, \ldots, 0}_{K-D \text{ times}})$;
- we make $\boldsymbol{W}$ to be equal to the vertical concatenation of $\boldsymbol{Q}^T$ with a $(K-D)\times D$ matrix of zeros, i.e.:

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{Q}^T \\ \boldsymbol{0}_{(K-D)\times D} \end{bmatrix}$$

We then have that:

$$\boldsymbol{c}_\Theta^T\boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{K}\boldsymbol{v}_i(\boldsymbol{w}_i^T\boldsymbol{x})^2 = \sum_{i=1}^{D}\lambda_i(\boldsymbol{q}_i^T\boldsymbol{x})^2 = (\boldsymbol{Q}^T\boldsymbol{x})^T\boldsymbol{\Lambda}(\boldsymbol{Q}^T\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T\boldsymbol{x} = \boldsymbol{c}^T\boldsymbol{\phi}(\boldsymbol{x})$$

and, by Lemma 1, we prove that the previous choice of $\boldsymbol{W}$ and $\boldsymbol{v}$ originate a vector $\boldsymbol{c}_\Theta$ such that $\boldsymbol{c}_\Theta = \boldsymbol{c}$. Furthermore, we have proven that the sets of classifiers $\mathcal{C}_1 = \{\boldsymbol{c}_\Theta^T\phi(x) : \Theta = (\boldsymbol{W}, \boldsymbol{v}) \in \mathbb{R}^{K\times D\times K}\}$ and $\mathcal{C}_2 = \{\boldsymbol{c}^T\phi(x) : \boldsymbol{c} \in \mathbb{R}^{\frac{D(D+1)}{2}}\}$ are exactly the same, and so the original neural network reduces to a **linear model** in terms of $\boldsymbol{c}_\Theta$. □

Consider now the case when $K < D$. Then, the nullspace of $\boldsymbol{W}$ has at least dimension $D - K \geq 1$, and thus there is some **non-null** vector $\boldsymbol{x^*}$ such that $\boldsymbol{W}\boldsymbol{x^*} = 0$. Now, choose $\boldsymbol{c}$ to be a vector such that $\mathcal{M}(\boldsymbol{c}) = \boldsymbol{I}_{D\times D}$ (i.e., the $D \times D$ identity matrix). We then have:

$$\boldsymbol{c}_\Theta^T\boldsymbol{\phi}(\boldsymbol{x^*}) = \boldsymbol{x^*}^T\boldsymbol{W}^T\text{diag}(\boldsymbol{v})\boldsymbol{W}\boldsymbol{x^*} = 0$$

and

$$\boldsymbol{c}^T\boldsymbol{\phi}(\boldsymbol{x^*}) = \boldsymbol{x^*}^T\mathcal{M}(\boldsymbol{c})\boldsymbol{x^*} = \boldsymbol{x^*}^T\boldsymbol{I}_{D\times D}\boldsymbol{x^*} = \|\boldsymbol{x^*}\|_2^2 > 0$$

Thus, for $K < D$, we have constructed an instance of $\boldsymbol{c}$ for which there exists some $\boldsymbol{x} \in \mathbb{R}^D$ such that $\boldsymbol{c}_\Theta^T\boldsymbol{\phi}(\boldsymbol{x}) \neq \boldsymbol{c}^T\boldsymbol{\phi}(\boldsymbol{x})$. Applying Lemma 1, we conclude that there is no choice of parameters $\boldsymbol{W}$ and $\boldsymbol{v}$ that make $\boldsymbol{c}_\Theta$ equal to $\boldsymbol{c}$ and so the model cannot be parametrized by $\boldsymbol{c}_\Theta$ in this case.

4. Given that $\hat{y} = \boldsymbol{c}_\Theta^T\boldsymbol{\phi}(\boldsymbol{x})$, we can write the squared loss as:

$$L(\boldsymbol{c}_\Theta^T; \mathcal{D}) = \frac{1}{2}\sum_{n=1}^{N}(\hat{y}_n(\boldsymbol{x}_n; \boldsymbol{c}_\Theta^T) - y_n)^2 = \frac{1}{2}\sum_{n=1}^{N}(\boldsymbol{c}_\Theta^T\boldsymbol{\phi}(\boldsymbol{x}) - y_n)^2 = \frac{1}{2}\|\boldsymbol{X}\boldsymbol{c}_\Theta - \boldsymbol{y}\|_2^2$$

where $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$. As such, the minimization of the squared lost corresponds to a linear least squares problem, which in this case has a unique solution $\boldsymbol{c}_\Theta^*$ that is found simply by setting the gradient to zero:

$$\nabla_{\boldsymbol{c}_\Theta}L(\boldsymbol{c}_\Theta^T; \mathcal{D}) = \boldsymbol{0} \Leftrightarrow \nabla_{\boldsymbol{c}_\Theta}(\boldsymbol{X}\boldsymbol{c}_\Theta)\nabla_{\boldsymbol{z}}(\|\boldsymbol{z}\|_2^2)|_{\boldsymbol{z}=\boldsymbol{X}\boldsymbol{c}_\Theta-\boldsymbol{y}} = \boldsymbol{0} \Leftrightarrow 2\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{c}_\Theta - \boldsymbol{y}) = \boldsymbol{0}$$
$$\Rightarrow \boldsymbol{c}_\Theta^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

To prove the existence of $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$, note that the dimensions of the nullspaces of and $\boldsymbol{X}$ and $\boldsymbol{X}^T\boldsymbol{X}$ are equal, since, for every $\boldsymbol{x} \in \mathbb{R}^{\frac{D(D+1)}{2}}$:

$$\boldsymbol{X}\boldsymbol{x} = \boldsymbol{0} \Rightarrow \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{x} = \boldsymbol{0}$$

and

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{x} = \boldsymbol{0} \Rightarrow x^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{x} = \boldsymbol{0} \Rightarrow \|\boldsymbol{X}\boldsymbol{x}\|_2^2 = \boldsymbol{0} \Rightarrow \boldsymbol{X}\boldsymbol{x} = \boldsymbol{0}$$

Since $N > \frac{D(D+1)}{2}$ and $\boldsymbol{X}$ has full column rank ($\frac{D(D+1)}{2}$), its nullspace has dimension 0. Hence, $\boldsymbol{X}^T\boldsymbol{X}$ is a $\frac{D(D+1)}{2} \times \frac{D(D+1)}{2}$ square matrix with rank $\frac{D(D+1)}{2} - 0 = \frac{D(D+1)}{2}$, and so it is invertible.

Usually, loss functions of feedforward neural networks are non-convex in their parameters due to multiple compositions of non-linear activation functions, making the global minimization of said functions especially hard. In spite of that, since the presented architecture only has a single hidden layer with at least as many units as the input layer and uses only quadratic activations, we managed to reduce it to a **linear regression** by defining the underlying **feature transformation** $\boldsymbol{\phi}$ and **weight vector** $\boldsymbol{c}_\Theta$. Hence, the minimization of the loss function becomes much easier, since the global minimizer can be computed in closed form.