Avoiding local minima in multilayer network optimization by incremental training

Alberto De Santis*, Giampaolo Liuzzi*, Stefano Lucidi*, Edoardo M. Tronci*

March 2021

Abstract. Training a large multilayer neural network can present many difficulties due to the large number of useless stationary points. These points usually attract the minimization algorithm used during the training phase, which therefore results inefficient. Extending some results proposed in literature for shallow networks, we propose the mathematical characterization of a class of such stationary points that arise in deep neural networks training. Availing such a description, we are able to define an incremental training algorithm that avoids getting stuck in the region of attraction of these undesirable stationary points.

Keywords. Multilayer neural networks · training algorithm · stationary points · plateaus

1 Introduction

Training any kind of neural networks is doubtless an extremely difficult task because in general it requires to minimize a non-convex objective loss function

$$\min_{x \in \mathbb{R}^n} f(x),$$

which may depend on a considerable number of training parameters. Another crucial issue regarding this problem is doubtless the presence of a consistent number of “useless” stationary points, i.e. stationary points that are quite far from those relevant minimum points of the loss function. Such stationary points could be in principle the cause of the so called plateaus phenomenon which plagues the minimization of $f(x)$.

Many interesting mathematical characterizations of such stationary points have been proposed in literature ([1, 2, 3, 6, 10]). For instance, in [5] it has been

---

*Dipartimento di Ingegneria informatica automatica e gestionale “A.Ruberti” – “Sapienza” Università di Roma. e-mails: desantis@diag.uniroma1.it, liuzzi@diag.uniroma1.it, lucidi@diag.uniroma1.it, tronci@diag.uniroma1.it
theoretically proved that even shallow networks (i.e. networks with only one single hidden layer) usually have many of such undesirable stationary points.

An interesting topic of research consists in studying how to fruitfully use the mathematical characterization of stationary points to develop ad-hoc optimization algorithms for neural networks training. In [5], it has been proved that a subset of critical points of a shallow network with $H - 1$ neurons in the hidden layer gives rise to submanifolds of critical points for a larger network with $H$ neurons.

Our aim in the paper is twofold. First a more theoretical contribution is given by extending the result of [5] to deep multilayer neural networks, i.e. networks with more than one layer. More precisely, we show that classes of stationary points of a given network derive from stationary points of smaller size networks, which are obtained from the larger size one by discarding an arbitrary (possibly large) number of neurons. On the other hand, the structure of such manifolds of stationary points shows that their number grows exponentially with the dimension of the network.

We then provide a computational contribution which exploits the extended characterization of classes of uninteresting stationary points to define a new optimization strategy which avoids getting stuck in their regions of attraction. More specifically, the proposed strategy is based on an incremental approach that produces points which have a loss function value smaller than that associated to the particular classes of those useless stationary points.

The computational experience shows that the proposed algorithm is able to effectively take advantage of the above mentioned mathematical characterization of stationary points on a variety of learning problems.

The remainder of the paper is organized as follows. We first introduce in Section 2 the notation used in this paper to define a general structure of our neural network. In Section 3 we consider the case of networks with more than one hidden layer and more outputs and when an arbitrary number of neurons are added to a given layer in the network. In Section 4 we further generalize the result by allowing to add any number of neurons on any intermediate layer. In Section 5 we formally state our incremental training algorithm (ITA) and in Section 6 we report a detailed numerical experience of our method over a significant set of test problems. We finally, in Section 7 draw some conclusions and outline possible future developments.

2 Topology and Notation

Given a supervised learning problem of the form $\{(x^p, y^p)\}_{p=1}^P$, where $x^p \in \mathbb{R}^n$, $y^p \in \mathbb{R}^m$ and $P$ is the number of training samples, we know from Hornik ([7], [8], [9]) that any continuous function on a closed and bounded subset of $\mathbb{R}^n$ can be universally approximated by a multilayer neural network, by finding a particular set of parameters $\theta^*$ that result in the best function approximation $f(x, \theta^*)$ within a given function space $\mathcal{F} = \{f(x, \theta) : \mathbb{R}^n \rightarrow \mathbb{R}^m | \theta \in \Theta\}$, where
\( \Theta \) represents the parameter space. A neural network can be seen as an acyclic oriented graph, as illustrated in Fig. 1: the first layer holds the set of \( n \) input nodes that connect each input component \( x_i \in \mathbb{R}, i = 1, \ldots, n \) to the network. Thus, \( x^p = (x_{1}^{p}, \ldots, x_{n}^{p}) \), \( p = 1, \ldots, P \) represents the input training set. A set of artificial neurons are distributed in \( L \) layers, which are connected together in a chain such that the first \( L-1 \) hidden layers have no direct connections with the output. The output layer \( L \) holds \( m \) neural units, one for each output dimension \( y_r \in \mathbb{R}, r = 1, \ldots, m \). Thus, \( y^p = (y_{1}^{p}, \ldots, y_{m}^{p}) \), \( p = 1, \ldots, P \) represents the labels training set. The overall length of the chain gives the depth of the model. Instead, the dimensionality of the hidden layers determines the width of the model. We will only consider fully connected (dense) networks, i.e. each neuron of each layer is connected with all the neurons of the previous and the subsequent layers. We have assumed that exist neither connections between neurons of the same layer, nor feedback connections between outputs of a layer and inputs of the preceding layers. The activation function is the same for each neuron of every hidden layer. Furthermore, we have considered a bias term associated to each neuron unit included the output units. To simplify the analysis, we have denoted with \( \ell = 1, \ldots, L \) the index of layers and with \( H_{\ell} \) the number of neurons in the \( \ell \)-th layer, where \( H_{0} = n \) and \( H_{L} = m \). Let \( \theta \in \mathbb{R}^{q} \) be the vector that encases all the network parameters, where \( q \) is the total number of network parameters, i.e.

\[
q = \sum_{i=0}^{L-1} H_i \times H_{i+1} + \sum_{i=1}^{L} H_i.
\]
Thus, for every $\ell = 1, \ldots, L$ we can decompose it as

\[
(\theta^1)^T = (\sigma_1^1, (w_1^1)^T, \ldots, \sigma_{H_1}^1, (w_{H_1}^1)^T) \\
(\theta^2)^T = ((\theta^1)^T, \sigma_1^2, (w_1^2)^T, \ldots, \sigma_{H_2}^2, (w_{H_2}^2)^T) \\
\ldots \ldots \ldots \ldots \ldots \ldots \\
(\theta^L)^T = ((\theta^{L-1})^T, \sigma_1^L, (w_1^L)^T, \ldots, \sigma_{H_L}^L, (w_{H_L}^L)^T)
\]

where

\[
\sigma_j^\ell \in \mathbb{R}, \quad w_j^\ell \in \mathbb{R}^{H_{\ell-1}}, \quad j = 1, \ldots, H_\ell
\]

are the bias term of neuron $j$ in layer $\ell$ and the weights vector of layer $\ell$ that assigns a scalar $w_{ji}^\ell \in \mathbb{R}$ to each arc, which is the weight between neuron $j$ of layer $\ell$ and neuron $i$ of layer $\ell - 1$. Note that $\theta^L = \theta$.

Being $g : \mathbb{R} \rightarrow \mathbb{R}$ the activation function, and denoting with $a_j^\ell$ the input of neuron $j$ in layer $\ell$, we obtain for neuron $j$ of the first layer

\[
a_j^1(x, \theta^1) = \sum_{i=1}^n w_{ji}^1 x_i + \sigma_j^1, \quad j = 1, \ldots, H_1
\]

and for neuron $j$ of layer $\ell > 1$

\[
a_j^\ell(x, \theta^\ell) = \sum_{i=1}^{H_{\ell-1}} w_{ji}^\ell g(a_{j}^{\ell-1}(x, \theta^{\ell-1})) + \sigma_j^\ell, \quad j = 1, \ldots, H_\ell
\]

We further assume that the output units are linear. Thus, for $\ell = L$ we have

\[
f_r(x, \theta^L) = a_r^L(x, \theta^L), \quad r = 1, \ldots, m.
\]

Under the assumptions stated, we can define an input-output mapping of the form

\[
f_r(x, \theta^L) = \sum_{i=1}^{H_{L-1}} w_{ri}^L g(a_{r}^{L-1}(x, \theta^{L-1})) + \sigma_r^L, \quad r = 1, \ldots, m. \quad (1)
\]

In order to choose an approximate function among all the possible functions in $\mathcal{F}$, we introduce the empirical risk

\[
R_{\text{emp}}(\theta) = \frac{1}{P} \sum_{p=1}^P L_p,
\]

where $L_p = \mathcal{L}(y^p, f(x^p, \theta^L)) \geq 0$ evaluates the distance between the experimental data $y^p$ and the output generated by the model $f(x^p, \theta^L)$. As known in the literature, $\mathcal{L}$ can assume different forms depending on the problem faced. The results in this paper are independent of the choice of the loss function.

The next step is to compute the derivative of $R_{\text{emp}}$ with respect to the network parameters $\theta$. In this regard, various methods have been proposed in literature, e.g. the back propagation approach. Here, we adopt the forward one since it is more suited for our needs.
For every $\ell = 1, \ldots, L$, let $j = 1, \ldots, H_\ell$ and $i = 1, \ldots, H_{\ell-1}$. Denote by $\delta_{cj}$ the kronecker delta symbol. For the input layer $\ell = 1$ we can write
\[
\frac{\partial a_1^c(x_p, \theta^1)}{\partial \sigma_j^1} = \delta_{cj},
\]
\[
\frac{\partial a_1^c(x_p, \theta^1)}{\partial w_{j,i}^1} = x_i \delta_{cj}, \quad c = 1, \ldots, H_1.
\]

(3)

For any successive layer $\ell = 2, \ldots, L$, we have that
\[
\frac{\partial a_\ell^c(x_p, \theta^\ell)}{\partial \sigma_j^\ell} = \delta_{cj},
\]
\[
\frac{\partial a_\ell^c(x_p, \theta^\ell)}{\partial w_{j,i}^\ell} = g\left(a_{\ell-1}^c(x_p, \theta^{\ell-1})\right) \delta_{cj}
\]

(5)

The previous relations evaluate the derivative of the node inputs with respects to their parameters. We now compute how the input parameters of the nodes of any layer influence the outputs of the nodes of any successive layer. For any $\ell < L$, let $q = \ell + 1, \ldots, L$ and $c = 1, \ldots, H_q$. We have
\[
\frac{\partial a_q^c(x_p, \theta^q)}{\partial \sigma_j^q} = \sum_{h=1}^{H_{q-1}} w_{c,h}^q g'(a_{h-1}^{q-1}(x_p, \theta^{q-1})) \frac{\partial a_{h-1}^{q-1}(x_p, \theta^{q-1})}{\partial \sigma_j^q},
\]
\[
\frac{\partial a_q^c(x_p, \theta^q)}{\partial w_{j,i}^q} = \sum_{h=1}^{H_{q-1}} w_{c,h}^q g'(a_{h-1}^{q-1}(x_p, \theta^{q-1})) \frac{\partial a_{h-1}^{q-1}(x_p, \theta^{q-1})}{\partial w_{j,i}^q}.
\]

(7)

Thus, for every $\ell = 1, \ldots, L$, $j = 1, \ldots, H_\ell$, $i = 1, \ldots, H_{\ell-1}$, we can write
\[
\frac{\partial R_{\text{emp}}(\theta^L)}{\partial \sigma_j^\ell} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L'(y_p^r, f_r(x_p^r, \theta^L)) \frac{\partial a_{\ell}^L(x_p^r, \theta^L)}{\partial \sigma_j^\ell},
\]
\[
\frac{\partial R_{\text{emp}}(\theta^L)}{\partial w_{j,i}^\ell} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L'(y_p^r, f_r(x_p^r, \theta^L)) \frac{\partial a_{\ell}^L(x_p^r, \theta^L)}{\partial w_{j,i}^\ell}.
\]

(8)

(9)

3 Adding $K$ neurons to one of the layers

In this section, we consider the case in which we add $K$ neurons to the $\ell$-th layer of a given neural network. Here, we denote by $\theta$ the vector of parameters of the smaller network and by $\hat{\theta}$ the vector of the bigger network.
In order to compute the empirical risk of the new network, we need to define the following quantities. Starting from
\[
\hat{a}_j^1(x, \hat{\theta}^1) = \sum_{i=1}^{n} \hat{w}_{ji}^1 x_i + \hat{\sigma}_j^1, \quad j = 1, \ldots, H_1,
\] (10)
for \(\ell = 2, \ldots, \hat{\ell} - 1\), we have
\[
\hat{a}_j^\ell(x, \hat{\theta}^\ell) = \sum_{i=1}^{H_{\ell-1}} \hat{w}_{ji}^\ell g(\hat{a}_{i}^{\ell-1}(x, \hat{\theta}^{\ell-1})) + \hat{\sigma}_j^\ell, \quad j = 1, \ldots, H_\ell.
\] (11)
Furthermore
\[
\hat{a}_j^{\hat{\ell}}(x, \hat{\theta}^{\hat{\ell}}) = \sum_{i=1}^{H_{\hat{\ell}-1}} \hat{w}_{ji}^{\hat{\ell}} g(\hat{a}_{i}^{\hat{\ell}-1}(x, \hat{\theta}^{\hat{\ell}-1})) + \hat{\sigma}_j^{\hat{\ell}}, \quad j = H_\hat{\ell} + 1, \ldots, H_\hat{\ell} + K,
\] (12)
\[
\hat{a}_j^{\hat{\ell}+1}(x, \hat{\theta}^{\hat{\ell}+1}) = \sum_{i=1}^{H_{\hat{\ell}}} \hat{w}_{ji}^{\hat{\ell}+1} g(\hat{a}_{i}^{\hat{\ell}}(x, \hat{\theta}^{\hat{\ell}})) + \sum_{i=H_{\hat{\ell}+1}}^{H_{\hat{\ell}+K}} \hat{w}_{ji}^{\hat{\ell}+1} g(\hat{a}_{i}^{\hat{\ell}}(x, \hat{\theta}^{\hat{\ell}})) + \hat{\sigma}_j^{\hat{\ell}+1}, \quad j = 1, \ldots, H_\hat{\ell} + 1.
\] (13)
For \(\ell = \hat{\ell} + 2, \ldots, L\)
\[
\hat{a}_j^{\ell}(x, \hat{\theta}^{\ell}) = \sum_{i=1}^{H_{\ell-1}} \hat{w}_{ji}^{\ell} g(\hat{a}_{i}^{\ell-1}(x, \hat{\theta}^{\ell-1})) + \hat{\sigma}_j^{\ell}, \quad j = 1, \ldots, H_\ell.
\] (14)
and
\[
\hat{f}_r(x, \hat{\theta}^{L}) = \hat{a}_r^{L}(x, \hat{\theta}^{L}), \quad r = 1, \ldots, m.
\] (15)
Therefore, the empirical risk of the new network can be written as
\[
\hat{R}_{emp}(\hat{\theta}) = \frac{1}{P} \sum_{p=1}^{P} \mathcal{L}(y^p, \hat{f}(x^p, \hat{\theta}^{L})),
\] (17)
where \(\hat{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m\).
The vector \(\hat{\theta}\) can be obtained starting from \(\theta\) according to some particular rules. In [5], some of these rules have been defined in the case of neural networks with
only one hidden layer, i.e. the mappings $\alpha$, $\beta$ and $\gamma$. Here, we extend those mappings to the more general case of deep neural networks, i.e. networks with more than one hidden layer, as shown in Fig. 2.

\[ \hat{\theta} = \alpha_{\zeta \sigma}(\theta) = \begin{cases} \hat{\sigma}^\ell_j = \sigma^\ell_j, & \hat{\omega}^\ell_{ji} = w^\ell_{ji} \\
\hat{\sigma}^\ell_j = \zeta_{j-H^\ell}, & \hat{\omega}^\ell_{ji} = v_{j-H^\ell} \\
\hat{\omega}^{\ell+1}_{ji} = 0, & \end{cases} \]

\[
\begin{array}{l}
\{ \ell = 1, \ldots, L, \\
j = 1, \ldots, H^\ell, \\
i = 1, \ldots, H^{\ell-1}, \}
\end{array}
\]

\[
\begin{array}{l}
\{ j = H^\ell + 1, \ldots, H^{\hat{\ell}+K}, \\
i = 1, \ldots, H^{\hat{\ell}-1}, \}
\end{array}
\]

\[
\begin{array}{l}
\{ j = 1, \ldots, H^{\hat{\ell}+1}, \\
i = H^{\hat{\ell}} + 1, \ldots, H^{\hat{\ell}+K} \}
\end{array}
\]
where $\zeta_j \in \mathbb{R}$, and $v_j \in \mathbb{R}^{H_{\ell-1}}$, $j = 1, \ldots, K$.

\[ \hat{\theta} = \beta_{\zeta_0}(\theta) = \left\{ \begin{array}{ll}
\hat{\sigma}_{j}^{\ell} = \sigma_{j}^\ell, & \hat{w}_{ji}^{\ell} = w_{ji}^{\ell}, \quad \text{for } j = 1, \ldots, H_\ell, \\
\hat{\sigma}_{j-H_\ell}^{\ell} = \zeta - H_\ell, & \hat{w}_{j-H_\ell}^{\ell} = 0, \\
\hat{\sigma}_{j+1}^{\ell} = \hat{\sigma}_{j-H_\ell}^{\ell}, & \hat{w}_{j+1}^{\ell} = w_{j+1}^{\ell}, \\
\hat{\sigma}_{j+1}^{\ell} = \bar{\sigma}_{j-H_\ell}^{\ell}, & \hat{w}_{j+1}^{\ell} = s_{j-H_\ell}, \quad \text{for } j = 1, \ldots, H_{\ell+1}, \\
\hat{\sigma}_{j}^{\ell+1} = \sigma_{j}^{\ell+1} - \sum_{i=H_{\ell+1}}^{H_{\ell+K}} s_{ji-H_\ell} g(\zeta - H_\ell), & \text{for } j = 1, \ldots, H_{\ell+1}, \end{array} \right. \]

where $\zeta_j \in \mathbb{R}$, $s_j \in \mathbb{R}^{H_{\ell+1}}$, $j = 1, \ldots, K$. Note that $\alpha_{\zeta_0}(\theta) = \beta_{\zeta_0}(\theta)$ trivially results. Thus, these two embeddings give the same critical point set. Letting $h \in \{1, \ldots, H_\ell\}$, we can then define the last embedding as

\[ \hat{\theta} = \gamma_{\lambda}(\theta) = \left\{ \begin{array}{ll}
\hat{\sigma}_{j}^\ell = \sigma_{j}^\ell, & \hat{w}_{ji}^{\ell} = w_{ji}^{\ell}, \quad \text{for } j = 1, \ldots, H_\ell, \\
\hat{\sigma}_{j}^\ell = \sigma_{j}^\ell, & \hat{w}_{ji}^{\ell} = w_{ji}^{\ell}, \quad \text{for } j = 1, \ldots, H_\ell, \\
\hat{w}_{j}^{\ell+1} = \bar{\sigma}_{ji-H_\ell}^{\ell+1}, & \hat{w}_{j}^{\ell+1} = \bar{\sigma}_{ji-H_\ell}^{\ell+1}, \quad \text{for } j = 1, \ldots, H_{\ell+1}, \end{array} \right. \]

for all $\lambda_i \in \mathbb{R}$, $i = 0, \ldots, K$ and such that $\sum_{i=0}^{K} \lambda_i = 1$.

**Proposition 3.1** For every point $\theta$, it results:

\[ \hat{R}_{\text{emp}}(\hat{\theta}) = \hat{R}_{\text{emp}}(\alpha_{\zeta_0}(\theta)) = R_{\text{emp}}(\theta), \]
\[ \hat{R}_{\text{emp}}(\hat{\theta}) = \hat{R}_{\text{emp}}(\beta_{\zeta_0}(\theta)) = R_{\text{emp}}(\theta), \]
\[ \hat{R}_{\text{emp}}(\hat{\theta}) = \hat{R}_{\text{emp}}(\gamma_{\lambda}(\theta)) = R_{\text{emp}}(\theta). \]
Proof. By the definitions of the three maps, we have for $\ell = 1$

$$\hat{a}_j^1(x, \alpha_{\zeta v}(\theta)^1) = \hat{a}_j^1(x, \beta_{\zeta s}(\theta)^1) = \hat{a}_j^1(x, \gamma_{\lambda}(\theta)^1) = a_j^1(x, \theta^1), \quad j = 1, \ldots, H_1.$$ 

Instead, for $\ell = 2, \ldots, \ell - 1$

$$\hat{a}_j^\ell(x, \alpha_{\zeta v}(\theta)^\ell) = \hat{a}_j^\ell(x, \beta_{\zeta s}(\theta)^\ell) = \hat{a}_j^\ell(x, \gamma_{\lambda}(\theta)^\ell) = a_j^\ell(x, \theta^\ell), \quad j = 1, \ldots, H_\ell,$$

$$\hat{a}_j^{\ell+1}(x, \alpha_{\zeta v}(\theta)^{\ell+1}) = \hat{a}_j^{\ell+1}(x, \beta_{\zeta s}(\theta)^{\ell+1}) = \hat{a}_j^{\ell+1}(x, \gamma_{\lambda}(\theta)^{\ell+1}) = a_j^{\ell+1}(x, \theta^{\ell+1}), \quad j = 1, \ldots, H_{\ell+1}.$$ 

(27)

Now, let’s consider the three different maps.

If $\hat{\theta} = \alpha_{\zeta v}(\theta)$, we have that

$$\hat{a}_j^{\ell+1}(x, \hat{\theta}^{\ell+1}) = \sum_{i=1}^{H_{\ell+1}} v_i j g(\hat{a}_i^{\ell-1}(x, \theta^{\ell-1})) + \zeta_j, \quad j = H_\ell + 1, \ldots, H_\ell + K,$$

$$\hat{a}_j^{\ell+2}(x, \hat{\theta}^{\ell+2}) = \sum_{i=1}^{H_{\ell+1}} \hat{w}_i j g(\hat{a}_i^{\ell}(x, \hat{\theta}^{\ell})) + \hat{\sigma}_j + \sum_{i=H_{\ell+1}}^{H_{\ell+K}} 0 g(\hat{a}_i^{\ell}(x, \hat{\theta}^{\ell})) + \hat{\sigma}_j^{\ell+1}, \quad j = 1, \ldots, H_{\ell+1}.$$ 

The definition of the map $\alpha_{\zeta v}$ and (27) imply that

$$\hat{a}_j^{\ell+1}(x, \hat{\theta}^{\ell+1}) = \sum_{i=1}^{H_{\ell+1}} \hat{w}_i j g(\hat{a}_i^{\ell}(x, \hat{\theta}^{\ell})) + \hat{\sigma}_j^{\ell+1}, \quad j = 1, \ldots, H_{\ell+1},$$

which ensures that, with the definition of $\alpha_{\zeta v}$, and for $\ell = \ell + 2, \ldots, L$

$$\hat{a}_j^{\ell}(x, \hat{\theta}^{\ell}) = \sum_{i=1}^{H_{\ell-1}} w_i j g(\hat{a}_i^{\ell-1}(x, \theta^{\ell-1})) + \sigma_j^{\ell} = a_j^{\ell}(x, \theta^\ell), \quad j = 1, \ldots, H_\ell,$$

$$\hat{f}_r(x, \hat{\theta}^L) = f_r(x, \theta^L), \quad r = 1, \ldots, m.$$ 

(28)

In conclusion (2), (17) and (28) prove (24).
If \( \hat{\theta} = \beta_{\zeta}(\theta) \), we have instead

\[
\hat{a}_j^\ell(x, \hat{\theta}^\ell) = \sum_{i=1}^{H_{\ell-1}} 0 g(\hat{a}_i^\ell-1(x, \hat{\theta}^\ell-1)) + \zeta_j, \quad j = H_\ell + 1, \ldots, H_\ell + K,
\]

\[
\hat{a}_j^{\ell+1}(x, \hat{\theta}^{\ell+1}) = \sum_{i=1}^{H_\ell} w_{ji}^{\ell+1} g(\hat{a}_i^\ell(x, \hat{\theta}^\ell)) + \sum_{i=H_{\ell+1}}^{H_{\ell+K}} s_{ji} g(\zeta_i) + \sigma_j^{\ell+1} + \]

\[- \sum_{i=H_{\ell+1}}^{H_{\ell+K}} s_{ji} g(\zeta_i), \quad j = 1, \ldots, H_{\ell+1}.
\]

Now, by using (27) we obtain

\[
\hat{a}_j^{\ell+1}(x, \hat{\theta}^{\ell+1}) = \sum_{i=1}^{H_\ell} w_{ji}^{\ell+1} g(\hat{a}_i^\ell(x, \hat{\theta}^\ell)) + \sigma_j^{\ell+1}
\]

\[
= \hat{a}_j^{\ell+1}(x, \theta^{\ell+1}), \quad j = 1, \ldots, H_{\ell+1},
\]

and for \( \ell = \hat{\ell} + 2, \ldots, L \)

\[
\hat{a}_j^\ell(x, \hat{\theta}^\ell) = \sum_{i=1}^{H_{\ell-1}} w_{ji}^\ell g(\hat{a}_i^{\ell-1}(x, \hat{\theta}^{\ell-1})) + \sigma_j^\ell
\]

\[
= a_j^\ell(x, \theta^\ell), \quad j = 1, \ldots, H_\ell,
\]

\[
\hat{f}_r(x, \hat{\theta}^L) = f_r(x, \theta^L), \quad r = 1, \ldots, m,
\]

thus proving (25).

Finally, if \( \hat{\theta} = \gamma_{\lambda}(\theta) \) we have

\[
\hat{a}_j^\ell(x, \hat{\theta}^\ell) = \sum_{i=1}^{H_{\ell-1}} w_{hi}^\ell g(\hat{a}_i^{\ell-1}(x, \hat{\theta}^{\ell-1})) + \sigma_h^\ell = a_h^\ell(x, \theta^\ell), \quad j = H_\ell + 1, \ldots, H_\ell + K,
\]

\[
\hat{a}_j^{\ell+1}(x, \hat{\theta}^{\ell+1}) = \sum_{i=1, i\neq h}^{H_\ell} w_{ji}^{\ell+1} g(\hat{a}_i^\ell(x, \hat{\theta}^\ell)) + \lambda_0 w_{jh}^{\ell+1} g(\hat{a}_h^\ell(x, \hat{\theta}^\ell)) +
\]

\[
\sum_{i=H_{\ell+1}}^{H_{\ell+K}} \lambda_i - H_j w_{jh}^{\ell+1} g(\hat{a}_h^\ell(x, \hat{\theta}^\ell)) + \sigma_j^{\ell+1}, \quad j = 1, \ldots, H_{\ell+1}.
\]
Now, the properties of the scalars $\lambda_i, i = 0, \ldots, K$ and (27) imply
\[ \hat{\alpha}_j^{\ell+1}(x, \hat{\theta}^{\ell+1}) = \alpha_j^{\ell+1}(x, \theta^{\ell+1}), \quad j = 1, \ldots, H_{\ell+1}, \]
and for $\ell = \hat{\ell} + 2, \ldots, L$
\[ \hat{\alpha}_j^{\ell}(x, \hat{\theta}^{\ell}) = \sum_{i=1}^{H_{\ell-1}} w_{ji}^{\ell} g(\alpha_i^{\ell-1}(x, \theta^{\ell-1})) + \sigma_j^{\ell} = \alpha_j^{\ell}(x, \theta^{\ell}), \quad j = 1, \ldots, H_{\ell}, \]
\[ \hat{f}_r(x, \hat{\theta}^{\ell}) = f_r(x, \theta^{\ell}), \quad r = 1, \ldots, m, \]
which hence proves (26) and concludes the proof. \[\square\]

**Proposition 3.2** For every point $\theta$, we have that, for $r = 1, \ldots, m$, and $\ell = 1, \ldots, L$
\[ \hat{f}_r(x, \beta\zeta_0(\theta)) = f_r(x, \theta^L), \quad (29) \]
\[ \hat{\alpha}_j^{\ell}(x, \beta\zeta_0(\theta)) = \alpha_j^{\ell}(x, \gamma\lambda(\theta)), \quad j = 1, \ldots, H_{\ell}, \quad (30) \]
and
\[ \hat{\alpha}_j^{\ell}(x, \beta\zeta_0(\theta)) = \zeta_j, \quad j = H_{\ell} + 1, \ldots, H_{\ell} + K, \quad (31) \]
\[ \hat{\alpha}_j^{\ell}(x, \gamma\lambda(\theta)) = a_{h}^{\ell}(x, \theta^\ell), \quad j = H_{\ell} + 1, \ldots, H_{\ell} + K, \quad (32) \]
where $h \in [1, \ldots, H_{\ell}]$.

**Proof.** The proof follows from the one of Proposition 3.1. \[\square\]

**Proposition 3.3** Let the point $\theta$ be such that
\[ \nabla_{\theta} R_{\text{emp}}(\theta) = 0, \quad (33) \]
and let the point $\hat{\theta}$ be given by
\[ \hat{\theta} = \beta\zeta_0(\theta), \quad (34) \]
or
\[ \hat{\theta} = \gamma\lambda(\theta). \quad (35) \]
Then, it results:
\[ \nabla_{\hat{\theta}} \hat{R}_{\text{emp}}(\theta) = 0. \quad (36) \]
Proof. The proof follows by evaluating the partial derivatives
\[ \frac{\partial \hat{R}_{\text{emp}}(\hat{\theta}^L)}{\partial \sigma'_j}, \quad \frac{\partial \hat{R}_{\text{emp}}(\hat{\theta}^L)}{\partial w_{j,i}^\ell}, \]
for \( \ell = 1, \ldots, L, \quad j = 1, \ldots, \hat{H}_\ell, \quad i = 1, \ldots, \hat{H}_{\ell-1} \) by using (3)-(9).

For the sake of brevity, we consider only the case in which \( \hat{\ell} \in (1, L) \). The cases \( \hat{\ell} = 1 \) and \( \hat{\ell} = L \) follow from similar reasoning.

The proof is divided in the following three parts:

a) \( \ell \in [\hat{\ell} + 1, L] \);

b) \( \ell = \hat{\ell} \);

c) \( \ell \in (1, \hat{\ell} - 1] \).

Part a): \( \ell \in [\hat{\ell} + 1, L] \).

For \( j = 1, \ldots, \hat{H}_\ell, i = 1, \ldots, \hat{H}_{\ell-1} \), we can recall formulas (3)-(9).

Therefore, for \( q = \ell + 1, \ldots, L, \quad c = 1, \ldots, H_q \), we have:
\[ \frac{\partial \hat{a}^\ell_j(x^p, \hat{\theta}^\ell)}{\partial \sigma'_j} = \delta_{cj}, \]
\[ \frac{\partial \hat{a}^\ell_j(x^p, \hat{\theta}^\ell)}{\partial w_{ji}^\ell} = g\left( \hat{a}^{\ell-1}_i(x^p, \hat{\theta}^\ell-1) \right) \delta_{cj}, \]
for \( q = \ell + 1, \ldots, L, \quad c = 1, \ldots, H_q \),
\[ \frac{\partial \hat{R}_{\text{emp}}(\hat{\theta}^L)}{\partial \sigma'_j} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} \mathcal{L}'(y^p, \hat{f}_h(x^p, \hat{\theta}^L)) \frac{\partial \hat{a}^L_{r}(x^p, \hat{\theta}^L)}{\partial \sigma'_j}, \]
\[ \frac{\partial \hat{R}_{\text{emp}}(\hat{\theta}^L)}{\partial w_{ji}^\ell} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} \mathcal{L}'(y^p, \hat{f}_r(x^p, \hat{\theta}^L)) \frac{\partial \hat{a}^L_{r}(x^p, \hat{\theta}^L)}{\partial w_{ji}^\ell}. \]
The definition of the maps $\beta_0$ and $\gamma_\lambda$, the equalities (29), (30) and the assumption (33) imply that for $\hat{c} = 1, \ldots, H_{\ell}$:

$$\frac{\partial \hat{a}_\ell^c (x^p, \hat{\theta}^\ell)}{\partial \hat{\sigma}_j} = \delta_{cj} = \frac{\partial a_\ell^c (x^p, \theta^\ell)}{\partial \sigma_j^\ell},$$

$$\frac{\partial \hat{a}_\ell^c (x^p, \hat{\theta}^\ell)}{\partial \hat{w}_{ji}^\ell} = g \left( a_{\ell-1}^c (x^p, \theta^{\ell-1}) \right) \delta_{cj} = \frac{\partial a_\ell^c (x^p, \theta^\ell)}{\partial w_{ji}^\ell},$$

for $q = \ell + 1, \ldots, L$, $c = 1, \ldots, H_q$

$$\frac{\partial \hat{a}_q^c (x^p, \hat{\theta}^q)}{\partial \hat{\sigma}_j} = \sum_{h=1}^{H_{q-1}} w_{ch}^q \frac{\partial a_{q-1}^c_{\theta} (x^p, \theta^{q-1})}{\partial \sigma_j^\ell} = \frac{\partial a_q^c (x^p, \theta^q)}{\partial \sigma_j^\ell},$$

$$\frac{\partial \hat{a}_q^c (x^p, \hat{\theta}^q)}{\partial \hat{w}_{ji}^\ell} = \sum_{h=1}^{H_{q-1}} w_{ch}^q \frac{\partial a_{q-1}^c_{\theta} (x^p, \theta^{q-1})}{\partial w_{ji}^\ell} = \frac{\partial a_q^c (x^p, \theta^q)}{\partial w_{ji}^\ell},$$

$$\frac{\partial \hat{R}_{\text{emp}} (\hat{\theta}^L)}{\partial \hat{\sigma}_j} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L(r, h(x^p, \theta^L)) \frac{\partial \hat{a}_r^L (x^p, \theta^L)}{\partial \sigma_j^\ell} = \frac{\partial R_{\text{emp}} (\theta^L)}{\partial \sigma_j^\ell} = 0, (37)$$

$$\frac{\partial \hat{R}_{\text{emp}} (\hat{\theta}^L)}{\partial \hat{w}_{ji}^\ell} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L(r, f_r(x^p, \theta^L)) \frac{\partial \hat{a}_r^L (x^p, \theta^L)}{\partial w_{ji}^\ell} = \frac{\partial R_{\text{emp}} (\theta^L)}{\partial w_{ji}^\ell} = 0. (38)$$

If $\ell = \hat{\ell} + 1$, for the components $j = 1, \ldots, H_{\hat{\ell}+1}$, and $i = H_{\hat{\ell}} + 1, \ldots, H_{\hat{\ell}} + K$ we have that:

when $\hat{\theta} = \beta_0 (\theta)$, the property (18) implies
for $\hat{c} = 1, \ldots, H_{\hat{\ell}+1}$

$$\frac{\partial \hat{\ell}^+_{\hat{c}}(x^p, \hat{\theta}^{\hat{\ell}+1})}{\partial \hat{w}^{\hat{\ell}+1}_{ji}} = g(\zeta)\delta_{\hat{c}j} = g(\zeta) \frac{\partial \hat{\ell}^+_{\hat{c}}(x^p, \hat{\theta}^{\hat{\ell}+2})}{\partial \hat{\delta}^{\hat{\ell}+1}_{j}},$$

for $q = \hat{\ell} + 2, \ldots, L, \ c = 1, \ldots, H_q$

$$\frac{\partial \hat{\delta}^{\hat{q}+1}_{j}(x^p, \hat{\theta}^{\hat{q}}))}{\partial \hat{w}^{\hat{\ell}+1}_{ji}} = g(\zeta) \sum_{h=1}^{H_{\hat{q}+1}} \hat{w}_{ch}^q g' \left( \hat{\delta}^{\hat{q}+1}_{h}(x^p, \hat{\theta}^{\hat{q}+1}) \right) \frac{\partial \hat{\delta}^{\hat{q}+1}_{h}(x^p, \hat{\theta}^{\hat{q}+1})}{\partial \hat{\delta}^{\hat{\ell}+1}_{j}} = g(\zeta) \frac{\partial \hat{\delta}^{\hat{q}+1}_{j}(x^p, \hat{\theta}^{\hat{q}}))}{\partial \hat{\delta}^{\hat{\ell}+1}_{j}},$$

where the last equality follows from (37).

Instead, when $\hat{\theta} = \gamma_{\lambda}(\theta)$ the property (20) ensures that there exist an index $z \in [1, \ldots, H_{\hat{\ell}}]$,

such that

for $\hat{c} = 1, \ldots, H_{\hat{\ell}+1}$

$$\frac{\partial \hat{\ell}^+_{\hat{c}}(x^p, \hat{\theta}^{\hat{\ell}+1})}{\partial \hat{w}^{\hat{\ell}+1}_{ji}} = g(\zeta)\delta_{\hat{c}j} = \frac{\partial \hat{\ell}^+_{\hat{c}}(x^p, \hat{\theta}^{\hat{\ell}+1})}{\partial \hat{w}^{\hat{\ell}+1}_{ji}},$$

for $q = \hat{\ell} + 2, \ldots, L, \ c = 1, \ldots, H_q$

$$\frac{\partial \hat{\delta}^{\hat{q}+1}_{j}(x^p, \hat{\theta}^{\hat{q}}))}{\partial \hat{w}^{\hat{\ell}+1}_{ji}} = \sum_{h=1}^{H_{\hat{q}+1}} \hat{w}_{ch}^q g' \left( \hat{\delta}^{\hat{q}+1}_{h}(x^p, \hat{\theta}^{\hat{q}+1}) \right) \frac{\partial \hat{\delta}^{\hat{q}+1}_{h}(x^p, \hat{\theta}^{\hat{q}+1})}{\partial \hat{\delta}^{\hat{\ell}+1}_{j}} = \frac{\partial \hat{\delta}^{\hat{q}+1}_{j}(x^p, \hat{\theta}^{\hat{q}}))}{\partial \hat{w}^{\hat{\ell}+1}_{ji}},$$

where (38) gives the last equality.

Part b): $\ell = \hat{\ell}$.

For every $j = 1, \ldots, H_{\hat{\ell}}$, and $i = 1, \ldots, \hat{H}_{\hat{\ell}+1}$ the expressions (3)-(9) yield:

$$\frac{\partial \hat{R}_{\text{emp}}(\hat{\theta}^{\hat{\ell}+1})}{\partial \hat{w}^{\hat{\ell}+1}_{ji}} = g(\zeta) \sum_{h=1}^{H_{\hat{q}+1}} \hat{w}_{ch}^q g' \left( \hat{\delta}^{\hat{q}+1}_{h}(x^p, \hat{\theta}^{\hat{q}+1}) \right) \frac{\partial \hat{\delta}^{\hat{q}+1}_{h}(x^p, \hat{\theta}^{\hat{q}+1})}{\partial \hat{\delta}^{\hat{\ell}+1}_{j}} = \frac{\partial \hat{R}_{\text{emp}}(\hat{\theta}^{\hat{\ell}+1})}{\partial \hat{w}^{\hat{\ell}+1}_{ji}} = 0,$$
for \( \hat{c} = 1, \ldots, \hat{H}_\ell \)

for \( j = 1, \ldots, H_\ell, \quad i = 1, \ldots, \hat{H}_\ell-1 \)

\[
\frac{\partial \hat{a}_j^\ell(x^p, \hat{\theta}^\ell)}{\partial \sigma_j^\ell} = \hat{\delta}_{ij} = \frac{\partial a_j^\ell(x^p, \theta^\ell)}{\partial \sigma_j^\ell},
\]

\[
\frac{\partial \hat{a}_j^\ell(x^p, \hat{\theta}^\ell)}{\partial \hat{w}_{ji}^\ell} = g \left( \hat{a}_i^{\ell-1}(x^p, \hat{\theta}^{\ell-1}) \right) \hat{\delta}_{ij} = \frac{\partial a_j^\ell(x^p, \theta^\ell)}{\partial w_{ji}^\ell},
\]

for \( j = H_\ell + 1, \ldots, H_\ell + K, \quad i = 1, \ldots, \hat{H}_\ell-1 \)

\[
\frac{\partial \hat{a}_j^\ell(x^p, \hat{\theta}^\ell)}{\partial \sigma_j^\ell} = \hat{\delta}_{ij},
\]

\[
\frac{\partial \hat{a}_j^\ell(x^p, \hat{\theta}^\ell)}{\partial \hat{w}_{ji}^\ell} = g \left( \hat{a}_i^{\ell-1}(x^p, \hat{\theta}^{\ell-1}) \right) \hat{\delta}_{ij}.
\]

When \( \hat{\theta} = \beta_{\psi_0}(\theta) \), the properties \([19]\) and \([30]\) yield:

for \( \hat{c} = 1, \ldots, \hat{H}_{\ell+1} \)

for \( j = 1, \ldots, H_{\ell+1}, \quad i = 1, \ldots, \hat{H}_{\ell+1} \)

\[
\frac{\partial \hat{a}_j^{\ell+1}(x^p, \hat{\theta}^\ell+1)}{\partial \sigma_j^\ell} = \hat{w}_{ei}^{\ell+1} g' \left( a_j^\ell(x^p, \theta^\ell) \right) \frac{\partial a_{j}^\ell(x^p, \theta^\ell)}{\partial \sigma_j^\ell} = \frac{\partial \hat{a}_j^{\ell+1}(x^p, \hat{\theta}^\ell+1)}{\partial \sigma_j^\ell},
\]

\[
\frac{\partial \hat{a}_j^{\ell+1}(x^p, \hat{\theta}^\ell+1)}{\partial \hat{w}_{ji}^{\ell+1}} = \hat{w}_{ei}^{\ell+1} g' \left( a_j^\ell(x^p, \theta^\ell) \right) \frac{\partial a_j^\ell(x^p, \theta^\ell)}{\partial w_{ji}^\ell} = \frac{\partial \hat{a}_j^{\ell+1}(x^p, \hat{\theta}^\ell+1)}{\partial \hat{w}_{ji}^{\ell+1}},
\]

for \( j = H_{\ell+1} + 1, \ldots, H_{\ell+1} + K, \quad i = 1, \ldots, \hat{H}_{\ell+1} \)

\[
\frac{\partial \hat{a}_j^{\ell+1}(x^p, \hat{\theta}^\ell+1)}{\partial \sigma_j^\ell} = 0 \quad g' \left( \zeta_j \right) \frac{\partial a_j^\ell(x^p, \theta^\ell)}{\partial \sigma_j^\ell} = 0,
\]

\[
\frac{\partial \hat{a}_j^{\ell+1}(x^p, \hat{\theta}^\ell+1)}{\partial \hat{w}_{ji}^{\ell+1}} = 0 \quad g' \left( \zeta_j \right) \frac{\partial a_j^\ell(x^p, \theta^\ell)}{\partial w_{ji}^\ell} = 0.
\]
Therefore, we have:

for \( q = \ell + 2, \ldots, L \), \( c = 1, \ldots, \hat{H}_q \)

for \( j = 1, \ldots, H_\ell \), \( i = 1, \ldots, \hat{H}_\ell-1 \)

\[
\frac{\partial a^q_c(x, \hat{\theta}^q)}{\partial \hat{\sigma}_j} = \sum_{h=1}^{H_q-1} w^q_{ch}g'(a_h^{-1}(x, \theta^{q-1})) \frac{\partial a_h^{-1}(x, \theta^{q-1})}{\partial \hat{\sigma}_j},
\]

\[
\frac{\partial a^q_c(x, \hat{\theta}^q)}{\partial \hat{a}^q_{ji}} = \sum_{h=1}^{H_q-1} w^q_{ch}g'(a_h^{-1}(x, \theta^{q-1})) \frac{\partial a_h^{-1}(x, \theta^{q-1})}{\partial \hat{a}^q_{ji}},
\]

\[
\frac{\partial R_{emp}(\hat{\theta}^L)}{\partial \hat{\sigma}_j} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L'(y^p, f_h(x^p, \theta^L)) \frac{\partial a^L_c(x^p, \theta^L)}{\partial \hat{\sigma}_j} = 0,
\]

\[
\frac{\partial R_{emp}(\hat{\theta}^L)}{\partial \hat{a}^q_{ji}} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L'(y^p, f_r(x^p, \theta^L)) \frac{\partial a^L_c(x^p, \theta^L)}{\partial \hat{a}^q_{ji}} = 0.
\]

for \( j = H_\ell + 1, \ldots, H_\ell + K \), \( i = 1, \ldots, \hat{H}_\ell-1 \)

\[
\frac{\partial a^q_c(x, \hat{\theta}^q)}{\partial \hat{\sigma}_j} = \sum_{h=1}^{H_q-1} w^q_{ch}g'(a_h^{-1}(x, \theta^{q-1})) 0 = 0,
\]

\[
\frac{\partial a^q_c(x, \hat{\theta}^q)}{\partial \hat{a}^q_{ji}} = \sum_{h=1}^{H_q-1} w^q_{ch}g'(a_h^{-1}(x, \theta^{q-1})) 0 = 0,
\]

\[
\frac{\partial R_{emp}(\hat{\theta}^L)}{\partial \hat{\sigma}_j} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L'(y^p, f_h(x^p, \theta^L)) 0 = 0,
\]

\[
\frac{\partial R_{emp}(\hat{\theta}^L)}{\partial \hat{a}^q_{ji}} = \frac{1}{P} \sum_{p=1}^{P} \sum_{r=1}^{m} L'(y^p, f_r(x^p, \theta^L)) 0 = 0.
\]

Instead, when \( \hat{\theta} = \gamma_\lambda(\theta) \) there exist an index

\[ z \in [1, \ldots, H_\ell] \]
such that property \([20]-[23]\) hold. Thus, we have:

for \(\hat{c} = 1, \ldots, \hat{H}_{\ell+1}\)

for \(j = 1, \ldots, H_\ell, \ j \neq z, \ i = 1, \ldots, \hat{H}_{\ell-1}\)
\[
\frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \hat{\theta}^{\ell+1})}{\partial \hat{\sigma}_{j}^{\ell}} = w_{\hat{c}j}^{\ell+1} g'(a_{j}^{\ell}(x^p, \theta^{\ell})) \frac{\partial a_{j}^{\ell}(x^p, \theta^{\ell})}{\partial \hat{\sigma}_{j}^{\ell}} = \frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \theta^{\ell+1})}{\partial \hat{\sigma}_{j}^{\ell}},
\]

for \(j = z, \ i = 1, \ldots, \hat{H}_{\ell-1}\)
\[
\frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \hat{\theta}^{\ell+1})}{\partial \hat{\sigma}_{z}^{\ell}} = w_{\hat{c}z}^{\ell+1} g'(a_{z}^{\ell}(x^p, \theta^{\ell})) \frac{\partial a_{z}^{\ell}(x^p, \theta^{\ell})}{\partial \hat{\sigma}_{z}^{\ell}} = \frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \theta^{\ell+1})}{\partial \hat{\sigma}_{z}^{\ell}},
\]

for \(j = H_\ell + 1, \ldots, H_{\ell} + K, \ i = 1, \ldots, \hat{H}_{\ell-1}\)
\[
\frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \hat{\theta}^{\ell+1})}{\partial \hat{\sigma}_{j}^{\ell}} = \lambda_{j-H_\ell} w_{\hat{c}z}^{\ell+1} g'(a_{z}^{\ell}(x^p, \theta^{\ell})) \frac{\partial a_{z}^{\ell}(x^p, \theta^{\ell})}{\partial \hat{\sigma}_{z}^{\ell}} = \lambda_{j-H_\ell} \frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \theta^{\ell+1})}{\partial \hat{\sigma}_{z}^{\ell}},
\]
\[
\frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \hat{\theta}^{\ell+1})}{\partial \hat{\sigma}_{j}^{\ell}} = \lambda_{j-H_\ell} w_{\hat{c}z}^{\ell+1} g'(a_{z}^{\ell}(x^p, \theta^{\ell})) \frac{\partial a_{z}^{\ell}(x^p, \theta^{\ell})}{\partial \hat{\sigma}_{z}^{\ell}} = \lambda_{j-H_\ell} \frac{\partial \hat{a}_{\hat{c}}^{\ell+1}(x^p, \theta^{\ell+1})}{\partial \hat{\sigma}_{z}^{\ell}}.
\]

Now, by recalling that \(\sum_{j=0}^{K} \lambda_{j} = 1\) and by repeating the same reasoning for the case \(\hat{\theta} = \beta_{0}(\theta)\), we obtain again \([43]\) and \([44]\).

Part c): \(\ell \in (1, \ldots, \hat{\ell} - 1]\).

For \(\ell \in (1, \ldots, \hat{\ell} - 1), \ j = 1, \ldots, \hat{H}_{\ell}, \ i = 1, \ldots, \hat{H}_{\ell-1}\) we can use again formulas \([3]-[9]\), the definition of the maps \(\beta_{0}\) and \(\gamma_{\lambda}\), the equalities \([29]\) and \([30]\).

In particular, we can write:

for \(\hat{c} = 1, \ldots, \hat{H}_{\ell}\)
\[
\frac{\partial \hat{a}_c^q(x, \hat{\theta})}{\partial \hat{\theta}} = \delta_{cj} = \frac{\partial a_c^q(x, \theta)}{\partial \theta},
\]

\[
\frac{\partial \hat{a}_c^q(x, \hat{\theta})}{\partial \hat{w}_{ji}^q} = g\left(a_i^{\ell-1}(x, \theta^{\ell-1})\right) \delta_{cj} = \frac{\partial a_c^\ell(x, \theta)}{\partial w_{ji}^q},
\]

for \( q = \ell + 1, \ldots, L \)

if \( q \neq \hat{\ell} \) and \( q \neq \hat{\ell} + 1 \), for \( c = 1, \ldots, \hat{H} \)

\[
\frac{\partial \hat{a}_c^q(x, \hat{\theta})}{\partial \hat{\theta}} = \sum_{h=1}^{\hat{H} - 1} \hat{w}_{ch}^q g \left( a_h^{q-1}(x, \theta^{q-1}) \right) \frac{\partial a_h^{q-1}(x, \theta^{q-1})}{\partial \hat{\theta}} = \frac{\partial a_c^q(x, \theta)}{\partial \hat{\theta}},
\]

\[
\frac{\partial \hat{a}_c^q(x, \hat{\theta})}{\partial \hat{w}_{ji}^q} = \sum_{h=1}^{\hat{H} - 1} \hat{w}_{ch}^q g \left( a_h^{q-1}(x, \theta^{q-1}) \right) \frac{\partial a_h^{q-1}(x, \theta^{q-1})}{\partial \hat{w}_{ji}^q} = \frac{\partial a_c^q(x, \theta)}{\partial \hat{w}_{ji}^q},
\]

if \( q = \hat{\ell} \), for \( c = 1, \ldots, \hat{H} \)

\[
\frac{\partial \hat{a}_c^\ell(x, \hat{\theta})}{\partial \hat{\theta}} = \sum_{h=1}^{\hat{H} - 1} \hat{w}_{ch}^\ell g' \left( a_h^{\ell-1}(x, \theta^{\ell-1}) \right) \frac{\partial a_h^{\ell-1}(x, \theta^{\ell-1})}{\partial \hat{\theta}} = \frac{\partial a_c^\ell(x, \theta)}{\partial \hat{\theta}},
\]

\[
\frac{\partial \hat{a}_c^\ell(x, \hat{\theta})}{\partial \hat{w}_{ji}^\ell} = \sum_{h=1}^{\hat{H} - 1} \hat{w}_{ch}^\ell g' \left( a_h^{\ell-1}(x, \theta^{\ell-1}) \right) \frac{\partial a_h^{\ell-1}(x, \theta^{\ell-1})}{\partial \hat{w}_{ji}^\ell} = \frac{\partial a_c^\ell(x, \theta)}{\partial \hat{w}_{ji}^\ell},
\]

if \( q = \hat{\ell} \) and \( \hat{\theta} = \beta \hat{c}(\theta) \), for \( \hat{c} = \hat{H} + 1, \ldots, \hat{H} + K \)

\[
\frac{\partial \hat{a}_c^\ell(x, \hat{\theta})}{\partial \hat{\theta}} = \sum_{h=1}^{\hat{H} - 1} \hat{w}_{ch}^\ell g' \left( a_h^{\ell-1}(x, \theta^{\ell-1}) \right) \frac{\partial a_h^{\ell-1}(x, \theta^{\ell-1})}{\partial \hat{\theta}},
\]

\[
\frac{\partial \hat{a}_c^\ell(x, \hat{\theta})}{\partial \hat{w}_{ji}^\ell} = \sum_{h=1}^{\hat{H} - 1} \hat{w}_{ch}^\ell g' \left( a_h^{\ell-1}(x, \theta^{\ell-1}) \right) \frac{\partial a_h^{\ell-1}(x, \theta^{\ell-1})}{\partial \hat{w}_{ji}^\ell}.
\]
if $q = \ell + 1$ and $\hat{\theta} = \beta_{0}(\theta)$, for $\hat{c} = 1, \ldots, H_{\ell+1}$

$$
\frac{\partial \hat{a}_{\ell+1}(x^p, \hat{\theta}^{\hat{c}+1})}{\partial \hat{\sigma}_{j}} = \sum_{h=1}^{H_{\ell}} w_{ch}^{\ell+1} g'(a_{h}^{\ell}(x^p, \hat{\theta}^{\hat{c}})) \frac{\partial a_{h}^{\ell}(x^p, \theta^{\hat{c}})}{\partial \sigma_{j}^{\hat{c}}}
+ \sum_{h=H_{\ell}+1}^{H_{\ell}+K} 0 g'(a_{h}^{\ell}(x^p, \hat{\theta}^{\hat{c}})) \frac{\partial a_{h}^{\ell}(x^p, \hat{\theta}^{\hat{c}})}{\partial \sigma_{j}^{\hat{c}}}
= \frac{\partial a_{\ell+1}(x^p, \theta^{\hat{c}+1})}{\partial \sigma_{j}^{\hat{c}}},
$$

$$
\frac{\partial \hat{a}_{\ell+1}(x^p, \hat{\theta}^{\hat{c}+1})}{\partial \hat{w}_{ji}^{\ell}} = \sum_{h=1}^{H_{\ell}} w_{ch}^{\ell+1} g'(a_{h}^{\ell}(x^p, \hat{\theta}^{\hat{c}})) \frac{\partial a_{h}^{\ell}(x^p, \theta^{\hat{c}})}{\partial w_{ji}^{\hat{c}}}
+ \sum_{h=H_{\ell}+1}^{H_{\ell}+K} 0 g'(ja_{h}^{\ell}(x^p, \hat{\theta}^{\hat{c}})) \frac{\partial a_{h}^{\ell}(x^p, \hat{\theta}^{\hat{c}})}{\partial \hat{w}_{ji}^{\ell}}
= \frac{\partial a_{\ell+1}(x^p, \theta^{\hat{c}+1})}{\partial \hat{w}_{ji}^{\ell}},
$$

if $q = \ell$ and $\hat{\theta} = \gamma_{\lambda}(\theta)$, for $z \in [1, \ldots, H_{\ell}]$ and for $\hat{c} = H_{\ell} + 1, \ldots, H_{\ell} + K$

$$
\frac{\partial \hat{a}_{\ell}(x^p, \hat{\theta}^{\hat{c}})}{\partial \hat{\sigma}_{j}^{\hat{c}}} = \sum_{h=1}^{H_{\ell-1}} w_{zh}^{\ell} g'(a_{h}^{\ell-1}(x^p, \theta^{\ell-1})) \frac{\partial a_{h}^{\ell-1}(x^p, \theta^{\ell-1})}{\partial \sigma_{j}^{\hat{c}}}
= \frac{\partial a_{\ell}(x^p, \theta^{\hat{c}})}{\partial \hat{\sigma}_{j}^{\hat{c}}},
$$

$$
\frac{\partial \hat{a}_{\ell}(x^p, \hat{\theta}^{\hat{c}})}{\partial \hat{w}_{ji}^{\ell}} = \sum_{h=1}^{H_{\ell-1}} w_{zh}^{\ell} g'(a_{h}^{\ell-1}(x^p, \theta^{\ell-1})) \frac{\partial a_{h}^{\ell-1}(x^p, \theta^{\ell-1})}{\partial \hat{w}_{ji}^{\ell}}
= \frac{\partial a_{\ell}(x^p, \theta^{\hat{c}})}{\partial \hat{w}_{ji}^{\ell}},
$$
Then, the proof of the proposition follows from (33)-(46).

From the previous equalities, we get again

\[
\frac{\partial \hat{a}_{\ell+1}^j(x_p, \hat{\theta}^{\ell+1})}{\partial \hat{\sigma}_j^f} = \sum_{h=1, h \neq z}^{H_i} w_{ch}^{\ell+1} g'(a_h^f(x_p, \theta^\ell)) \frac{\partial a_h^f(x_p, \theta^\ell)}{\partial \hat{\sigma}_j^f} + \lambda_0 w_{cz}^{\ell+1} g'(\hat{a}_w^f(x_p, \hat{\theta}^\ell)) \frac{\partial a_w^f(x_p, \hat{\theta}^\ell)}{\partial \hat{\sigma}_j^f} + \sum_{h=H_i+1}^{H_i+K} \lambda_{h-H_i} w_{cz}^{\ell+1} g'(\hat{a}_w^f(x_p, \hat{\theta}^\ell)) \frac{\partial a_w^f(x_p, \hat{\theta}^\ell)}{\partial \hat{\sigma}_j^f} = \frac{\partial a_{\ell+1}^i(x_p, \theta^{\ell+1})}{\partial \hat{\sigma}_j^f},
\]

\[
\frac{\partial \hat{a}_{\ell+1}^i(x_p, \theta^{\ell+1})}{\partial \hat{w}_{ji}^f} = \sum_{h=1}^{H_i} w_{ch}^{\ell+1} g'(a_h^f(x_p, \theta^\ell)) \frac{\partial a_h^f(x_p, \theta^\ell)}{\partial \hat{w}_{ji}^f} + \lambda_0 w_{cz}^{\ell+1} g'(\hat{a}_w^f(x_p, \hat{\theta}^\ell)) \frac{\partial a_w^f(x_p, \hat{\theta}^\ell)}{\partial \hat{w}_{ji}^f} + \sum_{h=H_i+1}^{H_i+K} \lambda_{h-H_i} w_{cz}^{\ell+1} g'(\hat{a}_w^f(x_p, \hat{\theta}^\ell)) \frac{\partial a_w^f(x_p, \hat{\theta}^\ell)}{\partial \hat{w}_{ji}^f} = \frac{\partial \hat{a}_{\ell+1}^{i+1}(x_p, \theta^{\ell+1})}{\partial \hat{w}_{ji}^f}.
\]

From the previous equalities, we get again

\[
\frac{\partial \hat{R}_{\text{emp}}(\theta^L)}{\partial \hat{\sigma}_j^f} = \frac{1}{P} \sum_{p=1}^P \sum_{r=1}^m \mathcal{L}'(y^p, f_h(x_p, \theta^L)) \frac{\partial a_h^L(x_p, \theta^L)}{\partial \hat{\sigma}_j^f} = \frac{\partial \hat{R}_{\text{emp}}(\theta^L)}{\partial \hat{\sigma}_j^f} = 0, \quad (45)
\]

\[
\frac{\partial \hat{R}_{\text{emp}}(\theta^L)}{\partial \hat{w}_{ji}^f} = \frac{1}{P} \sum_{p=1}^P \sum_{r=1}^m \mathcal{L}'(y^p, f_r(x_p, \theta^L)) \frac{\partial a_r^L(x_p, \theta^L)}{\partial \hat{w}_{ji}^f} = \frac{\partial \hat{R}_{\text{emp}}(\theta^L)}{\partial \hat{w}_{ji}^f} = 0. \quad (46)
\]

Then, the proof of the proposition follows from (33)-(46).
4 Adding neurons to different layers

Let us recall the definition of the mappings \( \alpha, \beta, \) and \( \gamma \) given in section 3, where we explicitly denote with \( K_\ell \) the neurons added to the \( \ell \)-th layer, i.e.

\[
\hat{\theta} = \alpha_{\zeta_v}(\theta; \ell, K_\ell), \\
\hat{\theta} = \beta_{\zeta_s}(\theta; \ell, K_\ell), \\
\hat{\theta} = \gamma_\lambda(\theta; \ell, K_\ell).
\]

Let us consider \( R = \{r_1, \ldots, r_t\} \subseteq \{1, \ldots, L\} \) and the finite set \( \Gamma = \{K_{r_1}, \ldots, K_{r_t}\} \).

Therefore, we can define the composition of mappings

\[
\theta_1 = \alpha_{\zeta_v}(\theta; r_1, K_{r_1}), \\
\theta_2 = \alpha_{\zeta_v}(\theta_1; r_2, K_{r_2}), \\
\vdots \\
\theta_t = \alpha_{\zeta_v}(\theta_{t-1}; r_t, K_{r_t}),
\]

and call them \( A_{\zeta_v}(\theta, R, \Gamma) \), i.e.

\[
\theta_t = \alpha_{\zeta_v}(\alpha_{\zeta_v}(\cdots \alpha_{\zeta_v}(\theta; r_1, K_{r_1}); r_2, K_{r_2}); \cdots; r_t, K_{r_t}) = A_{\zeta_v}(\theta, R, \Gamma),
\]

which is a composition of mappings that, given \( \theta \), produces \( \hat{\theta} = \theta_t \). Analogously, for the \( \beta \) and \( \gamma \) mappings, we can define

\[
\theta_1 = \beta_{\zeta_s}(\theta_1; r_1, K_{r_1}), \\
\theta_2 = \beta_{\zeta_s}(\theta_2; r_2, K_{r_2}), \\
\vdots \\
\theta_t = \beta_{\zeta_s}(\theta_{t-1}; r_t, K_{r_t}),
\]

and call them \( B_{\zeta_s}(\theta, R, \Gamma) \) and \( G_\lambda(\theta, R, \Gamma) \), i.e.

\[
\theta_t = \beta_{\zeta_s}(\beta_{\zeta_s}(\cdots \beta_{\zeta_s}(\theta; r_1, K_{r_1}); r_2, K_{r_2}); \cdots; r_t, K_{r_t}) = B_{\zeta_s}(\theta, R, \Gamma),
\]

and

\[
\theta_t = \gamma_\lambda(\gamma_\lambda(\cdots \gamma_\lambda(\theta; r_1, K_{r_1}); r_2, K_{r_2}); \cdots; r_t, K_{r_t}) = G_\lambda(\theta, R, \Gamma).
\]

**Proposition 4.1** For every point \( \theta \), it results

\[
\hat{R}_{\text{emp}}(\theta) = \hat{R}_{\text{emp}}(A_{\zeta_v}(\theta, R, \Gamma)) = R_{\text{emp}}(\theta), \\
\hat{R}_{\text{emp}}(\hat{\theta}) = \hat{R}_{\text{emp}}(B_{\zeta_s}(\theta, R, \Gamma)) = R_{\text{emp}}(\theta), \\
\hat{R}_{\text{emp}}(\hat{\theta}) = \hat{R}_{\text{emp}}(G_\lambda(\theta, R, \Gamma)) = R_{\text{emp}}(\theta).
\]

21
Proof. The proof follows by recursively applying the reasoning of Proposition 3.1. □

Proposition 4.2 Let the point \( \theta \) be such that

\[
\nabla_\theta R_{emp}(\theta) = 0,
\]

and let the point \( \hat{\theta} \) be given by

\[
\hat{\theta} = B_{\zeta_0}(\theta, R, \Gamma),
\]

or

\[
\hat{\theta} = G_\lambda(\theta, R, \Gamma).
\]

Then, it results:

\[
\nabla_{\hat{\theta}} \hat{R}_{emp}(\hat{\theta}) = 0.
\]

Proof. The proof follows quite easily by recursively applying Proposition 3.3. □

Remark 4.1 Note that an analogous result to that of Proposition 4.2 can be obtained when different mappings (i.e. \( \beta \) or \( \gamma \)) are used when adding neurons to different layers of the network. In particular, we can define

\[
\begin{align*}
\theta_1 &= \xi^{(1)}(\theta; r_1, K_{r_1}), \\
\theta_2 &= \xi^{(2)}(\theta_1; r_2, K_{r_2}), \\
\vdots \\
\theta_t &= \xi^{(t)}(\theta_{t-1}; r_t, K_{r_t}),
\end{align*}
\]

where each \( \xi^{(i)} \) is either \( \beta_{\zeta_0} \) or \( \gamma_\lambda \), for \( i = 1, \ldots, t \).

Considering again the results of Proposition 4.2 it can be noticed that the number of the manifolds of useless stationary points in a given network grows exponentially with the network dimension.

5 The incremental training algorithm

In this section, we formally state our proposed incremental training algorithm. The results described in the previous section have proved that every stationary point of a smaller network corresponds to certain stationary points of a larger network. In particular we have shown that, given a network with \( L \) layers, it is possible to add a given number of neurons on some of (or all) the layers preserving stationarity. Indeed, if \( \theta^* \) is stationary, then \( \hat{\theta}^* = B_{\zeta_0}(\theta^*, R, \Gamma) \) or \( \hat{\theta}^* = G_\lambda(\theta^*, R, \Gamma) \) are stationary in the bigger network. However, the latter may not correspond to a global minimum.
For the sake of simplicity, in this section we consider neural networks with a single hidden layer. In this case, we have $L = 1$ hidden layer with $H$ neurons and let 

$$R = \{1\}, \quad \Gamma = \{K\},$$

so that the mappings are

$$\hat{\theta}^{(H+K)} = \alpha_{\zeta v}(\hat{\theta}^{(H)}; 1, K) = \alpha_{\zeta v}(\hat{\theta}^{(H)}; K),$$

$$\hat{\theta}^{(H+K)} = \beta_{\zeta s}(\hat{\theta}^{(H)}; 1, K) = \beta_{\zeta s}(\hat{\theta}^{(H)}; K),$$

$$\hat{\theta}^{(H+K)} = \gamma_{\lambda}(\hat{\theta}^{(H)}; 1, K) = \gamma_{\lambda}(\hat{\theta}^{(H)}; K),$$

where we denoted with $\hat{\theta}^{(H)}$ the vector of parameters of the network with $H$ neurons. The idea behind our Incremental Training Algorithm (ITA) is training a network of a given dimension starting from a smaller network and progressively increasing the number of neurons in the hidden layer. At every iteration, the training of the larger network is performed by properly choosing both the starting point and the minimization technique. Regarding the starting points, we use the mapping $\alpha_{\zeta v}$, which guarantees that the objective function value doesn’t change. Instead, the gradient is unlikely to be equal to zero. Therefore, any descent algorithm that is able to produce a sequence $\{\hat{\theta}^{(H+K)}_h\}$ such that the objective function value satisfies

$$R_{\text{emp}}(\hat{\theta}^{(H+K)}_h) < R_{\text{emp}}(\hat{\theta}^{(H+K)}_0) = R_{\text{emp}}(\alpha_{\zeta v}(\hat{\theta}^{(H)}; K)).$$

is not attracted by those useless stationary points described in Proposition 4.2. Considering that

$$R_{\text{emp}}(\alpha_{\zeta v}(\hat{\theta}^{(H)}_s; K)) = R_{\text{emp}}(\beta_{\zeta 0}(\hat{\theta}^{(H)}_s; K)) = R_{\text{emp}}(\gamma_{\lambda}(\hat{\theta}^{(H)}_s; K))$$

the algorithm has the ability to escape from the stationary points generated by the mappings $\beta_{\zeta 0}(\hat{\theta}^{(H)}_s; K)$ and $\gamma_{\lambda}(\hat{\theta}^{(H)}_s; K)$.

6 Numerical experiments

In this section, we report numerical results to support the observations made above on our incremental training algorithm (ITA). To verify its scalability, we test the method on a set of standard test problems of different sizes mostly taken from the OpenML and UCI Machine Learning repositories as reported in Table 1, with the exception of the Power Consumption dataset, which represents the hourly power consumption of one of the largest energy provider in Italy.

Our incremental training algorithm was implemented in Python (v.3.8.5) using a XPS 15 7590 Intel(R) Core(TM) i7-9750H CPU, 2.60GHz, 16 GB RAM. For each test problem, we assessed both the standard and incremental approaches by training a neural network using the Pytorch (v.1.7.0) library with $\text{tanh}(t)$ as activation function for each neuron in the hidden layer, a linear activation function for the output layer and the mean squared error as loss function. For both
Algorithm 1: Incremental Training Algorithm (ITA)

1: **Data:** $H_0, H_{\text{max}} \in \mathbb{N}, \{K_k\} \subset \mathbb{N}, \{\tau_k\}$.
2: Set $\theta^{(H_0)} \sim U(0,1)$, $k \leftarrow 0$.
3: **while** $H_k \leq H_{\text{max}}$ **do**
   4:    Compute $\tilde{\theta}^{(H_k)}$ such that $\|\nabla R_{\text{emp}}(\tilde{\theta}^{(H_k)})\| \leq \tau_k$.
   5:    Let $H_{k+1} \leftarrow \min\{H_k + K_k, H_{\text{max}}\}$.
   6:    Choose $(\zeta_i, v_i) \sim U(0,1)$, for $i = 1, \ldots, \min\{H_{\text{max}} - H_k, K_k\}$
   7:    set $\theta^{(H_{k+1})} \leftarrow \alpha \zeta v(\tilde{\theta}^{(H_k)})$, s.t.
   8:          $R_{\text{emp}}(\theta^{(H_{k+1})}) = R_{\text{emp}}(\tilde{\theta}^{(H_k)})$, $\|\nabla R_{\text{emp}}(\theta^{(H_{k+1})})\| > \tau_k$
   9: **end while**

the methods considered, we used the L-BFGS optimization algorithm available in Pytorch with a maximum number of training epochs $\text{maxit} = 1000$. In particular, for the standard case we adopted the following parameters configuration:

- $H = 100$ neurons in the hidden layer;
- a tolerance $\text{tol} = 10^{-6}$ in the stopping condition, i.e.
  $$\|\nabla R_{\text{emp}}(\theta^{(H)})\|_{\infty} \leq 10^{-6}.$$  

Instead, for the incremental method (ITA) we made the following choices:

- initial number of neurons in the hidden layer $H_0 = 10$ and $K_k = H_k$;
- stopping criterion for the intermediate networks
  $$\|\nabla R_{\text{emp}}(\theta^{(H_k)})\|_{\infty} \leq 10^{-1} \|\nabla R_{\text{emp}}(\theta^{(H_k)})\|_{\infty}.$$  

From numerical experience, we noticed that a better performance can be obtained by also adding the following criterion for the intermediate networks:

$$|R_{\text{emp}}(\theta^{(H_k)}) - R_{\text{emp}}(\theta^{(H_k)})| \leq 10^{-2}.$$  

In the following paragraphs, we detailed the numerical results obtained from both the approaches considered, by firstly reporting their whole performance profiles in order to give a global perspective of the methods. Then, we gave a more detailed description of the proposed methods by reporting the training loss and box plot distribution for each single test problem evaluated.
Table 1: List of the test problems considered

| Name                                      | Type               | # Instances | # Attributes |
|-------------------------------------------|--------------------|-------------|--------------|
| Adult                                     | Classification     | 48842       | 14           |
| Ailerons                                  | Regression         | 13750       | 40           |
| Appliances Energy Prediction              | Regression         | 19735       | 29           |
| Arcene                                    | Classification     | 200         | 10000        |
| BlogFeedback                              | Regression         | 60021       | 281          |
| Boston House Prices                       | Regression         | 21613       | 19           |
| Breast Cancer Wisconsin (Diagnostic)      | Classification     | 569         | 32           |
| CIFAR 10                                  | Classification     | 20000       | 3072         |
| Gisette                                   | Classification     | 13500       | 5000         |
| Iris                                      | Classification     | 150         | 4            |
| MNIST Handwritten Digit                   | Classification     | 70000       | 784          |
| Mv                                        | Regression         | 40768       | 10           |
| QSAR Oral Toxicity                        | Classification     | 8992        | 1024         |
| Power Consumption                         | Regression         | 4520        | 347          |
| YearPred                                  | Regression         | 515345      | 90           |

Performance profiles  To compare the two variants of training methods, we adopted the performance profiles proposed in [4]. Let \( \mathcal{P} \) be a set of \( n_p \) problems and \( \mathcal{S} \) be a set of \( n_s \) solvers that can be used to solve problems in \( \mathcal{P} \). For each \( s \in \mathcal{S} \) and \( p \in \mathcal{P} \), let \( t_{ps} \) denote the performance index (i.e. the final function value obtained by the solver). The performance ratio is then defined as

\[
r_{ps} = \frac{t_{ps}}{\min_{s \in S} \{t_{ps}\}}.
\]

The performance profile relative to solver \( s \) is defined as

\[
\rho_s(\alpha) = \frac{1}{|\mathcal{P}|} \left| \left\{ p \in \mathcal{P} : r_{ps} \leq \alpha \right\} \right|,
\]

where \( \alpha \geq 1 \). Basically, for each solver a performance profile reports the percentage of problems for which a final function value is obtained which is within \( \alpha \)-times the function value attained by the best solver. Hence, the uppermost curve in the profiles denotes better performances of the corresponding algorithm. In our experimentation, the solvers are the incremental and standard methods whereas the problem set is composed of 10 replica for each of the 15 test problems, thus amounting to 150 problems. We reported the performance profiles of the two methods in Fig. 3, 4, 5 respectively with \text{maxit} = 200, 500, 1000. All these plots seem to suggest that our incremental training algorithm is significantly more efficient than the standard approach from a global behaviour perspective on all the datasets assessed.

Boxplot and training loss  To further understand the difference between the two approaches, we plotted the training loss and the boxplot distribution over
10 replica of each test problem assessed. Fig. 6 describes the performance of the two approaches in terms of the boxplot distribution with $\text{maxit} = 1000$. In this case, we can notice that in some datasets, i.e. Arcene, Iris, Qsar and Power Consumption, our incremental method is definitely better than the standard approach. Instead, for the Mv and YearPred cases, our approach performs worse than the standard one. In all the other cases, the performances of the two approaches are comparable. However, it is important to take into account that the information contained in the boxplots denotes an ensemble behaviour hiding the dynamics of the performance of the epochs, as we can appreciate from the loss function progress in Fig. 7 which describes the performance of the two approaches in terms of the training loss with $\text{maxit} = 1000$. After some initial iterations, following the incremental method consistently offers an additional progress, allowing to reduce the training loss more rapidly than the standard case. Even in those cases where it performs worse than the standard method, the incremental approach allows to reach the same training value of the standard method in fewer iterations, as shown in Fig. 8, 9. These preliminary results seem to suggest that the incremental approach may be useful to avoid the early stagnation experienced by the standard method.

7 Conclusions

In this paper we extended the result of [5] by characterizing the structure of undesirable stationary points to deep multilayer neural networks, i.e. networks with more than one layer. More precisely, we show that the structure of such manifolds of stationary points implies that their number grows exponentially with the dimension of the network.

Moreover, a novel incremental approach that avoids such undesirable stationary points is proposed. Unlike the traditional method, the main advantage of the proposed scheme is training a network of a given dimension starting from a smaller network and progressively increasing the number of neurons. Therefore, this method is able to escape from useless stationary points of a certain neural network by progressively exploit the information contained in the smaller networks. Numerical experiments on a significant number of test problems show the good performances of the proposed method when compared with a standard approach. In particular, our incremental scheme seems to be able to avoid the early stagnation experienced by the standard method on all the datasets assessed, thus suggesting more efficiency from a global behaviour perspective. However, the numerical results obtained refer to the case of a network with only one single hidden layer. The impact of the proposed method in deep multilayer neural networks training has not been studied yet in terms of numerical experience and will be developed in a future study.
References

[1] P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

[2] P. Baldi and Z. Lu. Complex-valued autoencoders. *Neural Networks*, 33:136–147, 2012.

[3] M. Brady, R. Raghavan, and J. Slawny. Back propagation fails to separate where perceptrons succeed. *IEEE Transactions on Circuits and Systems*, 36(5):665–674, 1989.

[4] E. Dolan and J. Moré. Benchmarking optimization software with performance profiles. *Mathematical programming*, 91(2):201–213, 2002.

[5] K. Fukumizu and S. Amari. Local minima and plateaus in hierarchical structures of multilayer perceptrons. *Neural networks*, 13(3):317–327, 2000.

[6] M. Gori and A. Tesi. On the problem of local minima in backpropagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):76–86, 1992.

[7] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[8] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[9] K. Hornik, M. Stinchcombe, and H. White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks*, 3(5):551–560, 1990.

[10] E. Sontag and H. Sussmann. Backpropagation can give rise to spurious local minima even for networks without hidden layers. *Complex Systems*, 3(1):91–106, 1989.
Appendix

Figure 3: Performance profiles of the standard and incremental training algorithms over 100 maximum iterations
Figure 4: Performance profiles of the standard and incremental training algorithms over 500 maximum iterations.
Figure 5: Performance profiles of the standard and incremental training algorithms over 1000 maximum iterations.
Figure 6: Boxplot distribution for the standard and incremental training algorithms over 1000 maximum iterations.
Figure 7: Training loss for the standard and incremental training algorithms over 1000 maximum iterations.
Figure 8: Training loss for the standard and incremental training algorithms over 100 maximum iterations
Figure 9: Training loss for the standard and incremental training algorithms over 500 maximum iterations.