Non-attracting Regions of Local Minima in Deep and Wide Neural Networks

Henning Petzka[1], Cristian Sminchisescu[1,2]

[1]Lund University
[2]Google Research

Abstract

Understanding the loss surface of neural networks is essential for the design of models with predictable performance and their success in applications. Experimental results suggest that sufficiently deep and wide neural networks are not negatively impacted by suboptimal local minima. Despite recent progress, the reason for this outcome is not fully understood. Could deep networks have very few, if at all, suboptimal local optima? or could all of them be equally good? We provide a construction to show that suboptimal local minima (i.e. non-global ones), even though degenerate, exist for fully connected neural networks with sigmoid activation functions. The local minima obtained by our proposed construction belong to a connected set of local solutions that can be escaped from via a non-increasing path on the loss curve. For extremely wide neural networks with two hidden layers, we prove that every suboptimal local minimum belongs to such a connected set. This provides a partial explanation for the successful application of deep neural networks. In addition, we also characterize under what conditions the same construction leads to saddle points instead of local minima for deep neural networks.

I. Introduction

At the heart of most optimization problems lies the search for the global minimum of a loss function. The common approach to finding a solution is to initialize at random in parameter space and subsequently follow directions of decreasing loss based on local methods. This approach lacks a global progress criteria, which leads to descent into one of the nearest local minima. Since the loss function of deep neural networks is non-convex, the common approach of using gradient descent variants is vulnerable precisely to that problem.

Authors pursuing the early approaches to local descent by back-propagating gradients [1] experimentally noticed that suboptimal local minima appeared surprisingly harmless. More recently, for deep neural networks, the earlier observations were further supported by the experiments of e.g., [2]. Several authors aimed to provide theoretical insight for this behavior. Broadly, two views may be distinguished. Some, aiming at explanation, rely on simplifying modeling assumptions. Others investigate neural networks under realistic assumptions, but often focus on failure cases only. Recently, Nguyen and Hein [3] provide partial explanations for deep and extremely wide neural networks for a class of activation functions including the commonly used sigmoid. Extreme width is characterized by a "wide" layer that has more neurons than input patterns to learn. For almost every instantiation of parameter values $\mathbf{w}$ (i.e. for all but a null set of parameter values) it is shown that, if the loss function has a local minimum at $\mathbf{w}$, then this local minimum must be a global one. This suggests that for deep and wide neural networks, possibly every local minimum is global. The question on what happens at the null set of parameter values, for which the result does not hold, remains unanswered.

Similar observations for neural networks with one hidden layer were made earlier by Gori and Tesi [4] and Poston et al. [5]. Poston et al. [5] show for a neural network with one hidden layer and sigmoid activation function that, if the hidden layer has more nodes than training patterns, then the error function (squared sum of prediction losses over the samples) has no suboptimal "local minimum" and "each point is arbitrarily close to a point from which a strictly decreasing path starts, so such a point cannot be separated from a so called good point by a barrier of any positive height" [5]. It was criticized by Sprinkhuizen-Kuyper and Boers [6] that the definition of a local

This work was supported in part by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI (PN-III-P4-ID-PCE-2016-0535, PN-III-P4-ID-PCCF-2016-0180), the EU Horizon 2020 grant DE-ENIGMA (688835), and SSF.
H. Petzka (email: henning.petzka@math.lth.se), C. Sminchisescu (cristian.sminchisescu@math.lth.se)

minimum used in the proof of [5] was rather strict and unconventional. In particular, the results do not imply that no suboptimal local minima, defined in the usual way, exist. As a consequence, the notion of attracting and non-attracting regions of local minima were introduced and the authors prove that non-attracting regions exist by providing an example for the extended XOR problem. The existence of these regions imply that a gradient-based approach descending the loss surface using local information may still not converge to the global minimum. The main objective of this work is to revisit the problem of such non-attracting regions and show that they also exist in deep and wide networks. In particular, a gradient based approach may get stuck in a suboptimal local minimum. Most importantly, the performance of deep and wide neural networks cannot be explained by the analysis of the loss curve alone, without taking proper initialization or the stochasticity of SGD into account.

Our observations are not fundamentally negative. At first, the local minima we find are rather degenerate. With proper initialization, a local descent technique is unlikely to get stuck in one of the degenerate, suboptimal local minima[1]. Secondly, the minima reside on a non-attracting region of local minima (see Definition 1). Due to its exploration properties, stochastic gradient descent will eventually be able to escape from such a region (see [8]). We conjecture that in sufficiently wide and deep networks, except for a null set of parameter values as starting points, there is always a monotonically decreasing path down to the global minimum. This was shown in [5] for neural networks with one hidden layer, sigmoid activation function and square loss, and we generalize this result to neural networks with two hidden layers. (More precisely, our result holds for all neural networks with square loss and a class of activation functions including the sigmoid, where the wide layer is the last or second last hidden layer). This implies that in such networks every local minimum belongs to a non-attracting region of local minima.

Our proof of the existence of suboptimal local minima even in extremely wide and deep networks is based on a construction of local minima in neural networks given by Fukumizu and Amari [9]. By relying on careful computation we are able to characterize when this construction is applicable to deep neural networks. Interestingly, in deeper layers, the construction rarely seems to lead to local minima, but more often to saddle points. The argument that saddle points rather than suboptimal local minima are the main problem in deep networks has been raised before (see [10]) but a theoretical justification [11] uses strong assumptions that do not exactly hold in neural networks. Here, we provide the first analytical argument, under realistic assumptions on the neural network structure, describing when certain critical points of the training loss lead to saddle points in deeper networks.

II. Related work

We discuss related work on suboptimal minima of the loss surface. In addition, we refer the reader to the overview article [12] for a discussion on the non-convexity in neural network training.

It is known that learning the parameters of neural networks is, in general, a hard problem. Blum and Rivest [13] prove NP-completeness for a specific neural network. It has also been shown that local minima and other critical points exist in the loss function of neural network training (see e.g. [6], [9], [14]–[17]). The understanding of these critical points has led to significant improvements in neural network training. This includes weight initialization techniques (e.g. [7]), improved backpropagation algorithms to avoid saturation effects in neurons [18], entirely new activation functions, or the use of second order information [19], [20].

That suboptimal local minima must become rather degenerate if the neural network becomes sufficiently large was observed for networks with one hidden layer in [4] and [5]. Recently, Nguyen and Hein [3] generalized this result to deeper networks containing an extremely wide hidden layer. Our contribution can be considered as a continuation of this work.

To explain the persuasive performance of deep neural networks, Dauphin et al. [10] experimentally show that there is a similarity in the behavior of critical points of the neural network's loss function with theoretical properties of critical points found for Gaussian fields on high-dimensional spaces [21]. Choromanska et al. [11] supply a theoretical connection, but they also require strong (arguably unrealistic) assumptions on the network structure. The results imply that (under their assumptions on the deep network) the loss at a local minimum must be close to the loss of the global minimum with high probability. In this line of research, [22] experimentally show a similarity between spin glass models and the loss curve of neural networks.

Why deep networks perform better than shallow ones is also investigated in [23] by considering a class of compositional functions. There is a number of papers partially answering the same question for ReLU and LeakyReLU

[1]That a proper initialization largely improves training performance is well-known. See, e.g., [7].

networks, where the space becomes combinatorial in terms of a positive activation, compared to a stalled (or weak) signal. Soudry and Hoffer [24] probabilistically compare the volume of regions (for a specific measure) containing bad local and global minima in the limit, as the number of data points goes to infinity. For networks with one hidden layer and ReLU activation function, Freeman and Bruna [25] quantify the amount of hill-climbing necessary to go from one point in the parameter space to another and finds that for increasing overparameterization, all level sets become connected. Swirszcz et al. [26] construct datasets that allow to find suboptimal local minima in overparameterized networks. Instead of analyzing local minima, Xie et al. [27] consider regions where the derivative of the loss is small for two-layer ReLu networks. Soudry and Carmon [28] consider leaky ReLU activation functions to find, similarly to the result of Nguyen and Hein [3], that for almost every combination of activation patterns in two consecutive mildly wide layers, a local minimum has global optimality.

To gain better insight into theoretical aspects, some papers consider linear networks, where the activation function is the identity. The classic result by Baldi and Hornik [29] shows that linear two-layer neural networks have a unique global minimum and all other critical values are saddle points. Kawaguchi, [30], Lu and Kawaguchi [31] and Yun et al. [32] discuss generalizations of [29] to deep linear networks.

The existence of non-increasing paths on the loss curve down to the global minimum is studied by Poston et al. [5] for extremely wide two-layer neural networks with sigmoid activation functions. For ReLU networks, Safran and Shamir [33] show that, if one starts at a sufficiently high initialization loss, then there is a strictly decreasing path with varying weights into the global minimum. Haeffele and Vidal [34] consider a specific class of ReLU networks with regularization, give a sufficient condition that a local minimum is globally optimal, and show that a non-increasing path down to the global minimum exists.

Finally, worth mentioning is the study of Liao and Poggio [35] who use polynomial approximations to argue, by relying on Bezout's theorem, that the loss function should have many local minima with zero empirical loss. Also relevant is the observation by Brady et al. [36] showing that, if the global minimum is not of zero loss, then a perfect predictor may have a larger loss in training than one producing worse classification results.

III. Main results

A. Problem definition

We consider regression networks with fully connected layers of size $n_l,\ 0 \leq l \leq L$ given by

$$f(x) = \mathbf{w}^L(\sigma(\mathbf{w}^{L-1}(\sigma(\dots(\mathbf{w}^2(\sigma(\mathbf{w}^1(x) + \mathbf{w}_0^1)) + \mathbf{w}_0^2)\dots)) + \mathbf{w}_0^{L-1})) + \mathbf{w}_0^L,$$

where $\mathbf{w}^l \in \mathbb{R}^{n_l \times n_{l-1}}$ denotes the weight matrix of the l-th layer, $1 \leq l \leq L$, $\mathbf{w}_0^l$ the bias terms, and σ a nonlinear activation function. The neural network function is denoted by f and we notationally suppress dependence on parameters. We assume the activation function σ to belong to the class of strictly monotonically increasing, analytic, bounded functions on $\mathbb{R}$ with image in interval (c, d) such that $0 \in [c, d]$, a class we denote by $\mathcal{A}$. As prominent examples, the sigmoid activation function $\sigma(t) = \frac{1}{1+\exp(-t)}$ and $\sigma(t) = \tanh(x)$ lie in $\mathcal{A}$.

We assume no activation function at the output layer.

The neural network is assumed to be a regression network mapping into the real domain $\mathbb{R}$, i.e. $n_L = 1$ and $\mathbf{w}^L \in \mathbb{R}^{1 \times n_{L-1}}$. We train on a finite dataset $(x_\alpha, y_\alpha)_{1 \leq \alpha \leq N}$ of size N with input patterns $x_\alpha \in \mathbb{R}^{n_0}$ and desired target value $y_\alpha \in \mathbb{R}$. We aim to minimize the squared loss $\mathcal{L} = \sum_{\alpha=1}^N (f(x_\alpha) - y_\alpha)^2$. Further, $\mathbf{w}$ denotes the collection of all $\mathbf{w}^l$.

The dependence of the neural network function f on $\mathbf{w}$ translates into a dependence of $\mathcal{L} = \mathcal{L}(\mathbf{w})$ of the loss function on the parameters $\mathbf{w}$. Due to assumptions on σ, $\mathcal{L}(\mathbf{w})$ is twice continuously differentiable. The goal of training a neural network consists of minimizing $\mathcal{L}(\mathbf{w})$ over $\mathbf{w}$. There is a unique value $\mathcal{L}_0$ denoting the infimum of the neural network's loss (most often $\mathcal{L}_0 = 0$ in our examples). Any set of weights $\mathbf{w}_\bullet$ that satisfies $\mathcal{L}(\mathbf{w}_\bullet) = \mathcal{L}_0$ is called a global minimum. Due to its non-convexity, the loss function $\mathcal{L}(\mathbf{w})$ of a neural network is in general known to potentially suffer from local minima (precise definition of a local minimum below). We will study the existence of *suboptimal local minima* in the sense that a local minimum $\mathbf{w}_*$ is suboptimal if its loss $\mathcal{L}(\mathbf{w}_*)$ is strictly larger than $\mathcal{L}_0$.

We refer to *deep neural networks* as models with more than one hidden layer. Further, we refer to *wide neural networks* as the type of model considered in [3]–[5] with one hidden layer containing at least as many neurons as input patterns (i.e. $n_l \geq N$ for some $1 \leq l < L$ in our notation).

Disclaimer: Naturally, training for zero global loss is not desirable in practice, neither is the use of fully connected wide and deep neural networks necessarily. The results of this paper are of theoretical importance. To be able to understand the complex learning behavior of deep neural networks in practice, it is a necessity to understand the networks with the most fundamental structure. In this regard, while our result are not directly applicable to neural networks used in practice, they do offer explanations for their learning behavior.

B. A special kind of local minimum

The standard definition of a local minimum, which is also used here, is a point $\mathbf{w}_*$ such that $\mathbf{w}_*$ has a neighborhood U with $\mathcal{L}(\mathbf{w}) \geq \mathcal{L}(\mathbf{w}_*)$ for all $\mathbf{w} \in U$. Since local minima do not need to be isolated (i.e. $\mathcal{L}(\mathbf{w}) > \mathcal{L}(\mathbf{w}_*)$ for all $\mathbf{w} \in U \setminus \{w_*\}$) two types of connected regions of local minima may be distinguished. Note that our definition slightly differs from the one by [6].

Definition 1. *[6] Let $\ell : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function. Suppose R is a maximal connected subset of parameter values $\mathbf{w} \in \mathbb{R}^m$, such that every $\mathbf{w} \in R$ is a local minimum of ℓ with value $\ell(\mathbf{w}) = c$.*

- *R is called an attracting region of local minima, if there is a neighborhood U of R such that every continuous path $\Gamma(t)$, which is non-increasing in ℓ and starts from some $\Gamma(0) \in U$, satisfies $\ell(\Gamma(t)) \geq c$ for all t.*
- *R is called a non-attracting region of local minima, if every neighborhood U of R contains a point from where a continuous path $\Gamma(t)$ exists that is non-increasing in ℓ and ends in a point $\Gamma(1)$ with $\ell(\Gamma(1)) < c$.*

Despite its non-attractive nature, a non-attracting region R of local minima may be harmful for a gradient descent approach. A path of greatest descent can end in a local minimum on R. However, no point z on R needs to have a neighborhood of attraction in the sense that following the path of greatest descent from a point in a neighborhood of z will lead back to z. (The path can lead to a different local minimum on R close by or reach points with strictly smaller values than c.)

In the example of such a region for the 2-3-1 XOR network provided in [6], a local minimum (of higher loss than the global loss) resides at points in parameter space with some coordinates at infinity. In particular, a gradient descent approach may lead to diverging parameters in that case. However, a different non-increasing path down to the global minimum always exists. It can be shown that local minima at infinity also exist for wide and deep neural networks. (The proof can be found in Appendix A.)

Theorem 1 (cf. [6] Section III)**.** *Let $\mathcal{L}$ denote the squared loss of a fully connected regression neural network with sigmoid activation functions, having at least one hidden layer and each hidden layer containing at least two neurons. Then, for almost every finite dataset, the loss function $\mathcal{L}$ possesses a local minimum at infinity. The local minimum is suboptimal whenever dataset and neural network are such that a constant function is not an optimal solution.*

A different type of non-attracting regions of local minima (without infinite parameter values) is considered for neural networks with one hidden layer by Fukumizu and Amari [9] and Wei et al. [8] under the name of singularities. This type of region is characterized by singularities in the weight space (a subset of the null set not covered by the results of Nguyen and Hein [3]) leading to a loss value strictly larger than the global loss. The dynamics around such region are investigated by Wei et al. [8]. Again, a full batch gradient descent approach can get stuck in a local minimum in this type of region. A rough illustration of the nature of these non-attracting regions of local minima is depicted in Fig. 1.

Non-attracting regions of local minima do not only exist in small two-layer neural networks.

Theorem 2. *There exist deep and wide fully-connected neural networks with sigmoid activation function such that the squared loss function of a finite dataset has a non-attracting region of local minima (at finite parameter values).*

The construction of such local minima is discussed in Section V with a complete proof in Appendix B.

Corollary 1. *Any attempt to show for fully connected deep and wide neural networks that a gradient descent technique will always lead to a global minimum only based on a description of the loss curve will fail if it doesn't take into consideration properties of the learning procedure (such as the stochasticity of stochastic gradient descent), properties of a suitable initialization technique, or assumptions on the dataset.*

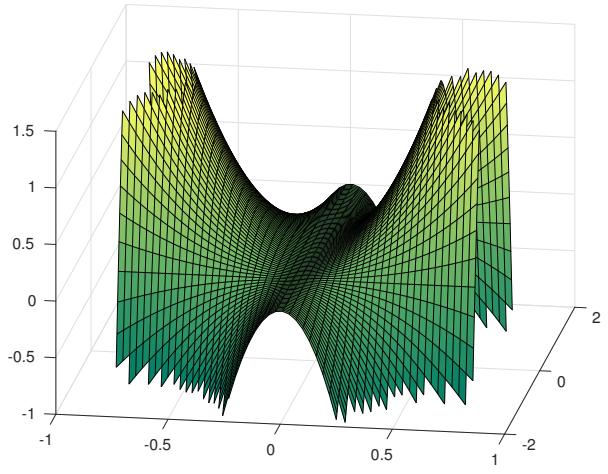

Fig. 1. The function $f(x, y) = 5x^4(1 - y^2)$ illustrating a non-attracting region of local minima given by $\{(x, y) \mid x = 0, y \in (-1, 1)\}$. (This example does not exactly appear in neural networks considered in this paper, but is of similar nature.)

On the positive side, we point out that a stochastic method such as stochastic gradient descent has a good chance to escape a non-attracting region of local minima due to noise. With infinite time at hand and sufficient exploration, the region can be escaped from with high probability (see [8] for a more detailed discussion). In Section V-A we will further characterize when the method used to construct examples of regions of non-attracting local minima is applicable. This characterization limits us to the construction of extremely degenerate examples. We give an intuitive argument why assuring the necessary assumptions for the construction becomes more difficult for wider and deeper networks and why it is natural to expect a lower suboptimal loss (where the suboptimal minima are less "bad") the less degenerate the constructed minima are and the more parameters a neural network possesses.

C. Non-increasing path to a global minimum

By definition, every neighborhood of a non-attracting region of local minima contains points from where a non-increasing path to a value less than the value of the region exists. (By definition all points belonging to a non-attracting region have the same value, in fact they are all local minima.) The question therefore arises whether from almost everywhere in parameter space there is such a non-increasing path all the way down to a global minimum. If the last hidden layer is the wide layer having more neurons than input patterns (for example consider a wide two-layer neural network), then this holds true by the results of [3] (and [4], [5]). We show the same conclusion to hold for wide neural networks having the second last hidden layer the wide one. In particular, this implies that for wide neural networks with two hidden layers, starting from almost everywhere in parameter space, there is non-increasing path down to a global minimum.

Theorem 3. *Consider a fully connected regression neural network with activation function in the class $\mathcal{A}$ equipped with the squared loss function for a finite dataset. Assume that the second last hidden layer contains more neurons than the number of input patterns. Then, for each set of parameters $\mathbf{w}$ and all $\epsilon > 0$, there is $\mathbf{w}'$ such that $||\mathbf{w} - \mathbf{w}'|| < \epsilon$ and such that a path non-increasing in loss from $\mathbf{w}'$ to a global minimum where $f(x_\alpha) = y_\alpha$ for each α exists.*

Corollary 2. *Consider a wide, fully connected regression neural network with two hidden layers and activation function in the class $\mathcal{A}$ and trained to minimize the squared loss over a finite dataset. Then all suboptimal local minima are contained in a non-attracting region of local minima.*

The rest of the paper contains the arguments leading to the given results.

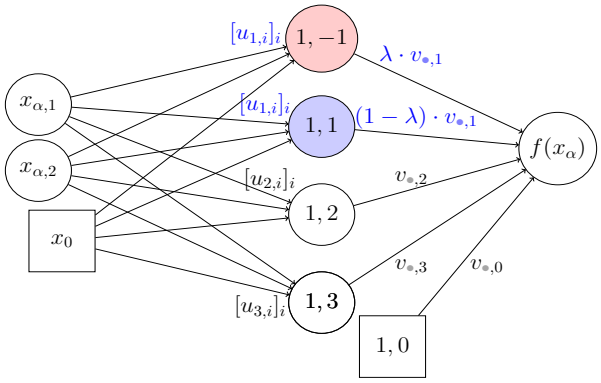

Fig. 2. An image of a two-layer neural network defined by weights in the image of the embedding γ^1_λ. Numbers in nodes denote the index of a neuron in form of (layer, neuron index).

IV. Notational choices

We fix additional notation aside the problem definition from Section III-A. For input x_α, we denote the pattern vector of values at all neurons at layer l before activation by $\mathsf{n}(l; x_\alpha)$ and after activation by $\mathsf{act}(l; x_\alpha)$.

In general, we will denote column vectors of size n with coefficients z_i by $[z_i]_{1\leq i\leq n}$ or simply $[z_i]_i$ and matrices with entries $a_{i,j}$ at position (i,j) by $[a_{i,j}]_{i,j}$. The neuron value pattern $\mathsf{n}(l;x)$ is then a vector of size n_l denoted by $\mathsf{n}(l;x) = [\mathsf{n}(l,k;x)]_{1\leq k\leq n_l}$, and the activation pattern $\mathsf{act}(l;x) = [\mathsf{act}(l,k;x)]_{1\leq k\leq n_l}$.

Using that f can be considered a composition of functions from consecutive layers, we denote the function from $\mathsf{act}(k;x)$ to the output by $h_{\bullet,k}(x)$. For convenience of the reader, a tabular summary of all notation is provided in Appendix A.

V. Construction of local minima

We recall the construction of so-called hierarchical suboptimal local minima given in [9] and extend it to deep networks. For the hierarchical construction of critical points, we add one additional neuron $\mathsf{n}(l,-1;x)$ to a hidden layer l. (Negative indices are unused for neurons, which allows us to add a neuron with this index.) Once we have fixed the layer l, we denote the parameters of the incoming linear transformation by $[u_{p,i}]_{p,i}$, so that $u_{p,i}$ denotes the contribution of neuron i in layer $l-1$ to neuron p in layer l, and the parameters of the outgoing linear transformation by $[v_{s,q}]$, where $v_{s,q}$ denotes the contribution of neuron q in layer l to neuron s in layer $l+1$. For weights of the output layer (into a single neuron), we write $w_{\bullet,j}$ instead of $w_{1,j}$.

We recall the function γ used in [9] to construct local minima in a hierarchical way. This function γ describes the mapping from the parameters of the original network to the parameters after adding a neuron $\mathsf{n}(l,-1;x)$ and is determined by incoming weights $u_{-1,i}$ into $\mathsf{n}(l,-1;x)$, outgoing weights $v_{s,-1}$ of $\mathsf{n}(l,-1;x)$, and a change of the outgoing weights $v_{s,r}$ of $\mathsf{n}(l,r;x)$ for one chosen r in the smaller network. Sorting the network parameters in a convenient way, the embedding of the smaller network into the larger one is defined for any $\lambda\in\mathbb{R}$ by a function γ^r_λ mapping parameters $\{([u_{r,i}]_i, [v_{s,r}]_s, \bar{\mathbf{w}}\}$ of the smaller network to parameters $\{([u_{-1,i}]_i, [v_{s,-1}]_s, [u_{r,i}]_i, [v_{s,r}]_s, \bar{\mathbf{w}})\}$ of the larger network and is defined by

$$\gamma^r_\lambda\left([u_{r,i}]_i, [v_{s,r}]_s, \bar{\mathbf{w}}\right) := \left([u_{r,i}]_i, [\lambda\cdot v_{s,r}]_s, [u_{r,i}]_i, [(1-\lambda)\cdot v_{s,r}]_s, \bar{\mathbf{w}}\right).$$

Here $\bar{\mathbf{w}}$ denotes the collection of all remaining network parameters, i.e., all $[u_{p,i}]_i, [v_{s,q}]_s$ for $p,q\notin\{-1,r\}$ and all parameters from linear transformation of layers with index smaller than l or larger than $l+1$, if existent. A visualization of γ^1_λ is shown in Fig. 2.

Important fact: For the functions φ, f of smaller and larger network at parameters $([u^*_{1,i}]_i, [v^*_{s,1}]_s, \bar{\mathbf{w}}^*)$ and $\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ respectively, we have $\varphi(x) = f(x)$ for all x. More generally, we even have $\mathsf{n}^\varphi(l,k;x) = \mathsf{n}(l,k;x)$ and $\mathsf{act}^\varphi(l,k;x) = \mathsf{act}(l,k;x)$ for all l, x and $k\geq 0$.

A. Characterization of hierarchical local minima

Using γ^r to embed a smaller deep neural network into a second one with one additional neuron, it has been shown that critical points get mapped to critical points.

Theorem 4 (Nitta [15])**.** *Consider two neural networks as in Section III-A, which differ by one neuron in layer l with index $\mathsf{n}(l,-1;x)$ in the larger network. If parameter choices $([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ determine a critical point for the squared loss over a finite dataset in the smaller network then, for each $\lambda \in \mathbb{R}$, $\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ determines a critical point in the larger network.*

As a consequence, whenever an embedding of a local minimum with γ^r_λ into a larger network does not lead to a local minimum, then it leads to a saddle point instead. (There are no local maxima in the networks we consider, since the loss function is convex with respect to the parameters of the last layer.) For neural networks with one hidden layer, it was characterized when a critical point leads to a local minimum.

Theorem 5 (Fukumizu, Amari [9])**.** *Consider two neural networks as in Section III-A with only one hidden layer and which differ by one neuron in the hidden layer with index $\mathsf{n}(1,-1;x)$ in the larger network. Assume that parameters $([u^*_{r,i}]_i, v^*_{\bullet,r}, \bar{\mathbf{w}}^*)$ determine a local minimum for the squared loss over a finite dataset in the smaller neural network and that $\lambda \notin \{0,1\}$.*

Then $\gamma^r_\lambda([u^*_{r,i}]_i, v^*_{\bullet,r}, \bar{\mathbf{w}}^*)$ determines a local minimum in the larger network if the matrix $[B^r_{i,j}]_{i,j}$ given by

$$B^r_{i,j} = \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot v^*_{\bullet,r} \cdot \sigma''(\mathsf{n}(1,r;x_\alpha)) \cdot x_{\alpha,i} \cdot x_{\alpha,j}$$

is positive definite and $0 < \lambda < 1$, or if $[B^r_{i,j}]_{i,j}$ is negative definite and $\lambda < 0$ or $\lambda > 1$. (Here, we denote the k-th input dimension of input x_α by $x_{\alpha,k}$.)

We extend the previous theorem to a characterization in the case of deep networks. We note that a similar computation was performed in [19] for neural networks with two hidden layers.

Theorem 6. *Consider two (possibly deep) neural networks as in Section III-A, which differ by one neuron in layer l with index $\mathsf{n}(l,-1;x)$ in the larger network. Assume that the parameter choices $([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ determine a local minimum for the squared loss over a finite dataset in the smaller network. If the matrix $[B^r_{i,j}]_{i,j}$ defined by*

$$\begin{aligned} B^r_{i,j} := \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,k;x_\alpha)} \cdot v^*_{k,r} \cdot \sigma''(\mathsf{n}(l,r;x_\alpha)) \\ \cdot \ \mathsf{act}(l-1,i;x_\alpha) \cdot \mathsf{act}(l-1,j;x_\alpha) \end{aligned} \tag{1}$$

is either

- *positive definite and $\lambda \in \mathcal{I} := (0,1)$, or*
- *negative definite and $\lambda \in \mathcal{I} := (-\infty, 0) \cup (1, \infty)$,*

then $\left\{\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*) \mid \lambda \in \mathcal{I}\right\}$ determines a non-attracting region of local minima in the larger network if and only if

$$D^{r,s}_i := \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)} \cdot \sigma'(\mathsf{n}(l,r;x_\alpha)) \cdot \mathsf{act}(l-1,i;x_\alpha) \tag{2}$$

is zero, $D^{r,s}_i = 0$, for all i, s.

Remark 1. *In the case of a neural network with only one hidden layer as considered in Theorem 5, the function $h_{\bullet,l+1}(x)$ is the identity function on $\mathbb{R}$ and the matrix $[B^r_{i,j}]_{i,j}$ in (1) reduces to the matrix $[B^r_{i,j}]_{i,j}$ in Theorem 5. The condition that $D^{r,s}_i = 0$ for all i, s does hold for shallow neural networks with one hidden layer as we show below. This proves Theorem 6 to be consistent with Theorem 5.*

The theorem follows from a careful computation of the Hessian of the cost function $\mathcal{L}(\mathbf{w})$, characterizing when it is positive (or negative) semidefinite and checking that the loss function does not change along directions that correspond to an eigenvector of the Hessian with eigenvalue 0. We state the outcome of the computation in Lemma 1 and refer the reader interested in a full proof of Theorem 6 to Appendix B.

Lemma 1. *Consider two (possibly deep) neural networks as in Section III-A, which differ by one neuron in layer l with index $\mathsf{n}(l,-1;x)$ in the larger network. Fix $1 \leq r \leq n_l$. Assume that the parameter choices $([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ determine a critical point in the smaller network.*

Let $\mathcal{L}$ denote the the loss function of the larger network and ℓ the loss function of the smaller network. Let $\alpha \neq -\beta \in \mathbb{R}$ such that $\lambda = \frac{\beta}{\alpha+\beta}$.

With respect to the basis of the parameter space of the larger network given by $([u_{-1,i}+u_{r,i}]_i, [v_{s,-1}+v_{s,r}]_s, \bar{\mathbf{w}}, [\alpha \cdot u_{-1,i} - \beta \cdot u_{r,i}]_i, [v_{s,-1} - v_{s,r}]_s)$, the Hessian of $\mathcal{L}$ (i.e., the second derivative with respect to the new network parameters) at $\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ is given by

$$\begin{pmatrix} [\frac{\partial^2 \ell}{\partial u_{r,i} \partial u_{r,j}}]_{i,j} & 2[\frac{\partial^2 \ell}{\partial u_{r,i} \partial v_{s,r}}]_{i,s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{r,i}}]_{i,\bar{\mathbf{w}}} & 0 & 0 \\ 2[\frac{\partial^2 \ell}{\partial u_{r,i} \partial v_{s,r}}]_{s,i} & 4[\frac{\partial^2 \ell}{\partial v_{s,r} \partial v_{t,r}}]_{s,t} & 2[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,r}}]_{s,\bar{\mathbf{w}}} & (\alpha-\beta)[D^{r,s}_i]_{s,i} & 0 \\ [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{r,i}}]_{\bar{\mathbf{w}},i} & 2[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,r}}]_{\bar{\mathbf{w}},s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial \bar{\mathbf{w}}'}]_{\bar{\mathbf{w}},\bar{\mathbf{w}}'} & 0 & 0 \\ 0 & (\alpha-\beta)[D^{r,s}_i]_{i,s} & 0 & \alpha\beta[B^r_{i,j}]_{i,j} & (\alpha+\beta)[D^{r,s}_i]_{i,s} \\ 0 & 0 & 0 & (\alpha+\beta)[D^{r,s}_i]_{s,i} & 0 \end{pmatrix}$$

B. Shallow networks with a single hidden layer

For the construction of suboptimal local minima in wide two-layer networks, we begin by following the experiments of [9] that prove the existence of suboptimal local minima in (non-wide) two-layer neural networks.

Consider a neural network of size 1–2–1. We use the corresponding network function f to construct a dataset $(x_\alpha, y_\alpha)_{\alpha=1}^N$ by randomly choosing x_α and letting $y_\alpha = f(x_\alpha)$. By construction, we know that a neural network of size 1–2–1 can perfectly fit the dataset with zero error.

Consider now a smaller network of size 1–1–1 having too little expressibility for a global fit of all data points. We find parameters $[u^*_{1,1}, v^*_\bullet]$ where the loss function of the neural network is in a local minimum with non-zero loss. For this small example, the required positive definiteness of $[B^1_{i,j}]_{i,j}$ from (1) for a use of γ_λ with $\lambda \in (0,1)$ reduces to checking a real number for positivity, which we assume to hold true. We can now apply γ_λ and Theorem 5 to find parameters for a neural network of size 1–2–1 that determine a suboptimal local minimum. This example may serve as the base case for a proof by induction to show the following result.

Theorem 7. *There is a wide neural network with one hidden layer and arbitrarily many neurons in the hidden layer that has a non-attracting region of suboptimal local minima.*

Having already established the existence of parameters for a (small) neural network leading to a suboptimal local minimum, it suffices to note that iteratively adding neurons using Theorem 5 is possible. Iteratively at step t, we add a neuron $\mathsf{n}(1,-t;x)$ to the network by an application of γ^1_λ with the same $\lambda \in (0,1)$. The corresponding matrix from (1),

$$B^{1,(t)}_{i,j} = \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot (1-\lambda)^t \cdot v^*_{\bullet,1} \cdot \sigma''(\mathsf{n}(l,1;x_\alpha)) \cdot x_{\alpha,i} \cdot x_{\alpha,j},$$

is positive semidefinite. (We use here that neither $f(x_\alpha)$ nor $\mathsf{n}(l,1;x_\alpha)$ ever change during this construction.) By Theorem 5 we always find a suboptimal minimum with nonzero loss for the network for $\lambda \in (0,1)$. Note however, that a continuous change of λ to a value outside of $[0,1]$ does not change the network function, but leads to a saddle point. Hence, we found a non-attracting region of suboptimal minima.

Remark 2. *Since we started the construction from a network of size 1–1–1, our constructed example is extremely degenerate: The suboptimal local minima of the wide network have identical incoming weight vectors for each hidden neuron. Obviously, the suboptimality of this parameter setting is easily discovered. Also with proper initialization, the chance of landing in this local minimum is vanishing.*

However, one may also start the construction from a more complex network with a larger network with several hidden neurons. In this case, when adding a few more neurons using γ^1_λ, it is much harder to detect the suboptimality of the parameters from visual inspection.

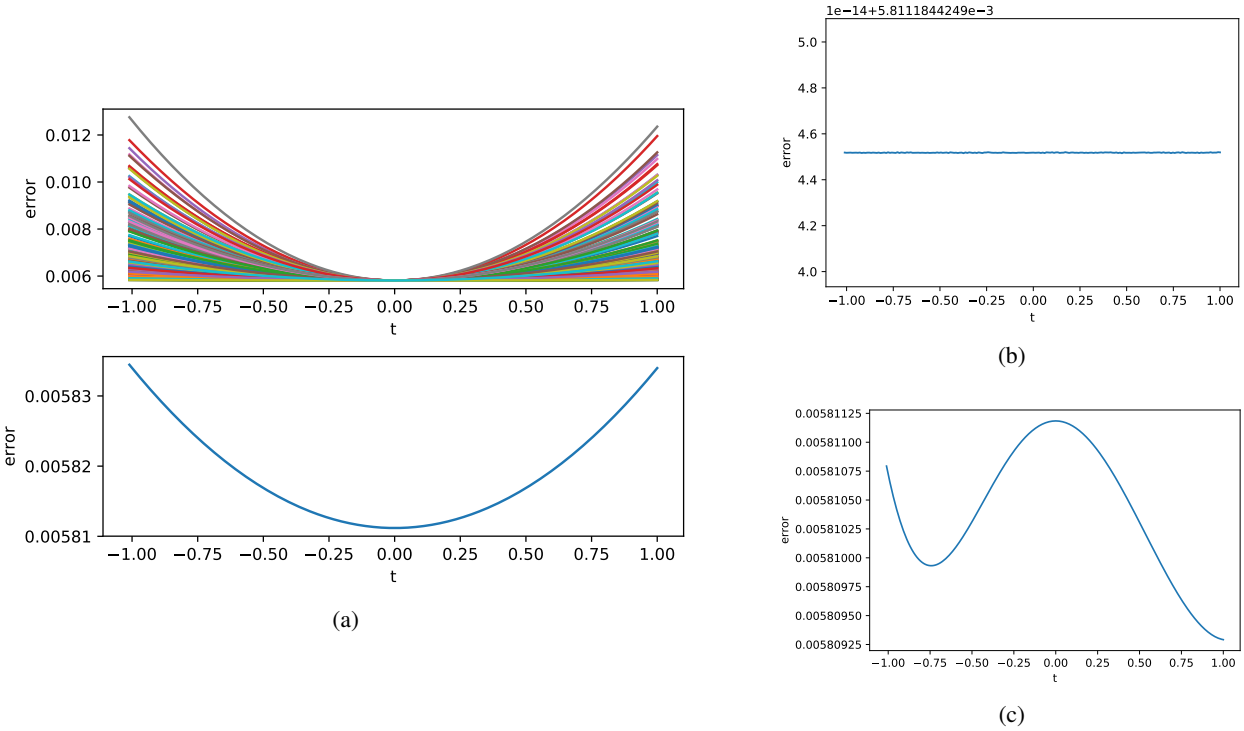

Fig. 3. A non-attracting region of local minima in a deep neural network. (a) A local minimum at t=0. Top: Evolution of the loss into random directions. Bottom: The minimum over all sampled directions. (b) Path along a degenerate direction to a saddle point. (c) Saddle point with the same loss value. Error evolution along a direction of descent.

C. Deep neural networks

According to Theorem 6, next to positive definiteness of the matrix $B^r_{i,j}$ for some r, in deep networks there is a second condition for the construction of hierarchical local minima using the map γ^r_λ, i.e. $D_i^{r,s} = 0$. We consider conditions that make $D_i^{r,s} = 0$.

Proposition 1. *Suppose we have a hierarchically constructed critical point of the squared loss of a neural network constructed by adding a neuron into layer l with index $\mathsf{n}(l,-1;x)$ by application of the map γ^r_λ to a neuron $\mathsf{n}(l,r;x)$. Suppose further that for the outgoing weights $v^*_{s,r}$ of $\mathsf{n}(l,r;x)$ we have $\sum_s v^*_{s,r} \neq 0$, and suppose that $D_i^{r,s}$ is defined as in* (2)*. Then $D_i^{r,s} = 0$ if one of the following holds.*

(i) *The layer l is the last hidden layer. (This condition includes the case $l = 1$ indexing the hidden layer in a two-layer network.)*

(ii)
$$\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)} = \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,t;x_\alpha)}$$
for all s,t,α

(iii) *For each α and each t, with $\mathcal{L}_\alpha := (f(x_\alpha) - y_\alpha)^2$,*
$$\frac{\partial \mathcal{L}_\alpha}{\partial \mathsf{n}(l+1,t;x_\alpha)} = (f(x_\alpha) - y_\alpha) \cdot \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha)}{\partial \mathsf{n}(l+1,t;x_\alpha)} = 0.$$
(This condition holds in the case of the weight infinity attractors in the proof to Theorem 1 for $l+1$ the second last layer. It also holds in a global minimum.)

The proof is contained in Appendix C.

D. Experiment for deep networks

To construct a local minimum in a deep and wide neural network, we start by considering a three-layer network of size 2–2–4–1, i.e. we have two input dimensions, one output dimension and hidden layers of two and four neurons.

We use its network function f to create a dataset of 50 samples $(x_\alpha, f(x_\alpha))$, hence we know that a network of size 2–2–4–1 can attain zero loss.

We initialize a new neural network of size 2–2–2–1 and train it until convergence, before using the construction to add neurons to the network. When adding neurons to the last hidden layer using γ^1_λ, Proposition 1 assures that $D_i^{1,\bullet} = 0$ for all i. We check for positive definiteness of the matrix $B^1_{i,j}$, and only continue when this property holds. Having thus assured the necessary condition of Theorem 6, we can add a few neurons to the last hidden layer (by induction as in the two-layer case), which results in local minimum of a network of size 2–2–M–1. The local minimum of non-zero loss that we attain is suboptimal whenever $M \geq 4$ by construction. For $M \geq 50$ the network is wide.

Experimentally, we show not only that indeed we end up with a suboptimal minimum, but also that it belongs to a non-attracting region of local minima. In Fig. 3 we show results after adding eleven neurons to the last hidden layer. On the left side, we plot the loss in the neighborhood of the constructed local minimum in parameter space. The top image shows the loss curve into randomly generated directions, the bottom displays the minimal loss over all these directions. On the top right we show the change of loss along one of the degenerate directions that allows reaching a saddle point. In such a saddle point we know from Lemma 1 the direction of descent. The image on the bottom right shows that indeed the direction allows a reduction in loss. Being able to reach a saddle point from a local minimum by a path of non-increasing loss shows that indeed we found a non-attracting region of local minima.

E. A discussion of limitations and of the loss of non-attracting regions of suboptimal minima

We fix a neuron in layer l and aim to use γ^r_λ to find a local minimum in the larger network. We then need to check whether a matrix $B^r_{i,j}$ is positive definite, which depends on the dataset. Under strong independence assumptions (the signs of different eigenvalues of $B^r_{i,j}$ are independent), one may argue similar to arguments in [10] that the probability of finding $B^r_{i,j}$ to be positive definite (all eigenvalues positive) is exponentially decreasing in the number of possible neurons of the previous layer $l-1$. At the same time, the number of neurons $\mathsf{n}(l, r; x)$ in layer l to use for the construction only increases linearly in the number of neurons in layer l.

Experimentally, we use a four-layer neural network of size 2–8–12–8–1 to construct a (random) dataset containing 500 labeled samples. We train a network of size 2–4–6–4–1 on the dataset until convergence using SciPy's[2] BFGS implementation. For each layer l, we check each neuron r whether it can be used for enlargment of the network using the map γ^r_λ for some $\lambda \in (0, 1)$, i.e., we check whether the corresponding matrix $B^r_{i,j}$ is positive definite. We repeat this experiment 1000 times. For the first layer, we find that in 547 of 4000 test cases the matrix is positive definite. For the second layer we only find $B^r_{i,j}$ positive definite in 33 of 6000 cases, and for the last hidden layer there are only 6 instances out of 4000 where the matrix $B^r_{i,j}$ is positive definite. Since the matrix $B^r_{i,j}$ is of size $2 \times 2/4 \times 4/6 \times 6$ for the first/second/last hidden layer respectively, the number of positive matrices is less than what would be expected under the strong independence assumptions discussed above.

In addition, in deeper layers, further away from the output layer, it seems dataset dependent and unlikely to us that $D_i^{r,s} = 0$. Simulations seem to support this belief. However, it is difficult to check the condition numerically. Firstly, it is hard to find the exact position of minima and we only compute numerical approximations of $D_i^{r,s}$. Secondly, the terms are small for sufficiently large networks and numerical errors play a role. Due to these two facts, it becomes barely possible to check the condition of exact equality to zero. In Fig. 4 we show the distribution of maximal entries of the matrix $D_i^{r,s} = 0$ for neurons in the first, second and third layer of the network of size 2–4–6–4–1 trained as above. Note that for the third layer we know from theory that in a critical point we have $D_i^{r,s} = 0$, but due to numerical errors much larger values arise.

Further, a region of local minima as above requires linearly dependent activation pattern vectors. This is how linear dimensions for subsequent layers get lost, reducing the ability to approximate the target function. Intuitively, in a deep and wide neural network there are many possible directions of descent. Loosing some of them still leaves the network with enough freedom to closely approximate the target function. As a result, these suboptimal minima have a loss close to the global loss.

Conclusively, finding suboptimal local minima with high loss by the construction using γ^r_λ becomes hard when the networks become deep and wide.

[2]https://www.scipy.org/

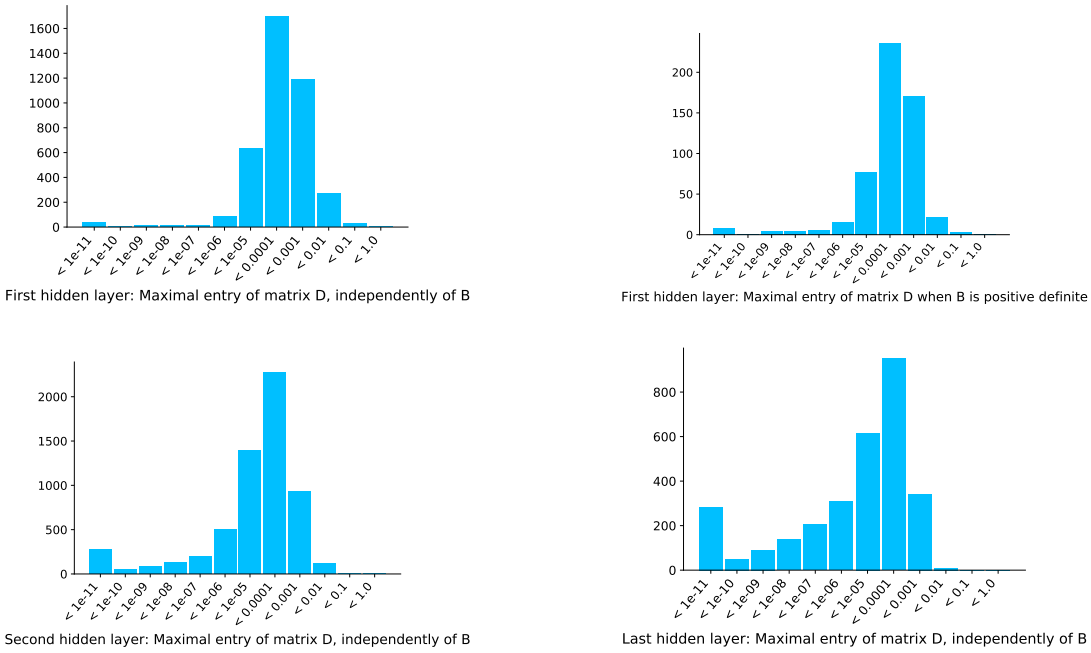

Fig. 4. Distribution of maximal entries of matrices $D_i^{r,s}$ for first, second and last hidden layer over training a network of size 2–4–6–4–1 over 1000 random datasets.

VI. Proving the existence of a non-increasing path to the global minimum

In the previous section we showed the existence of non-attracting regions of local minima. These type of local minima do not rule out the possibility of non-increasing paths to the global minimum from almost everywhere in parameter space. In this section, we sketch the proof to Theorem 3 illustrated in form of several lemmas, where up to the basic assumptions on the neural network structure as in Section III-A (with activation function in $\mathcal{A}$), the assumption of one lemma is given by the conclusion of the previous one. A full proof can be found in Appendix D.

We consider vectors that we call activation vectors, different from the activation pattern vectors $\mathsf{act}(l;x)$ from above. The activation vector at neuron k in layer l is denoted by $\mathbf{a}_k^l$ and defined by all values at the given neuron for different samples x_α:

$$\mathbf{a}_k^l := [\mathsf{act}(l,k;x_\alpha)]_\alpha.$$

In other words while we fix l and x for the activation pattern vectors $\mathsf{act}(l;x)$ and let k run over its possible values, we fix l and k for the activation vectors $\mathbf{a}_k^l$ and let x run over its samples x_α in the dataset.

The first step of the proof is to use the freedom given by ϵ to have the activation vectors $\mathbf{a}^{L-2}$ of the wide layer $L-2$ span the whole space $\mathbb{R}^N$.

Lemma 2. *[3, Corollary 4.5] For each choice of parameters* $\mathbf{w}$ *and all* $\epsilon > 0$ *there is* $\mathbf{w}'$ *such that* $||\mathbf{w}-\mathbf{w}'|| < \epsilon$ *and for the activation vectors* $\boldsymbol{a}_k^{L-2}$ *of the wide layer* $L-2$ *for* $\mathbf{w}'$ *we have*

$$\mathrm{span}_k\, \boldsymbol{a}_k^{L-2} = \mathbb{R}^N.$$

Lemma 3. *Assume that in the wide layer* $L-2$ *we have that the activation vectors satisfy*

$$\mathrm{span}_k[\boldsymbol{a}_k^{L-2}] = \mathbb{R}^N.$$

Then for any continuous paths $\nu : [0,1] \to \mathbb{R}^{n_{L-1}\times N}$ *in layer* $L-1$ *with* $\nu(0) = [\mathsf{n}(L-1,s;x_\alpha)]_{s,\alpha}$ *there is a continuous path of parameters* $\Gamma : [0,1] \to \mathbb{R}^{n_{L-1}\times n_{L-2}}$ *with* $\Gamma(0) = \mathbf{w}^{L-1}$ *and such that*

$$\nu(t) = \Gamma(t)\cdot[\mathsf{act}(L-2,k;x_\alpha)]_{k,\alpha}$$

Lemma 4. *For all continuous paths* $\rho(t)$ *in* $Im(\sigma)^N$*, i.e. the N-fold copy of the image of* σ*, there is a continuous path* $\nu(t)$ *in* $\mathbb{R}^N$ *such that* $\rho(t) = \sigma(\nu(t))$ *for all* t*.*

The activation vectors $\mathbf{a}_k^{L-1}$ of the last hidden layer span a linear subspace H of $\mathbb{R}^N$. The optimal parameters $\mathbf{w}^L$ of the output layer compute the best approximation of $(y_\alpha)_\alpha$ onto H. Lemma 3 and Lemma 4 together imply that we can achieve any desired continuous change of the spanning vectors of H, and hence the linear subspace H, by a suitable change of the parameters $\mathbf{w}^{L-1}$.

As it turns out, there is a natural possible path of parameters that strictly monotonically decreases the loss to the global minimum whenever we may assume that not all non-zero coefficients of $\mathbf{w}^L$ have the same sign. If this is not the case, however, we first follow a different path through the parameter space to eventually assure different signs of coefficients of $\mathbf{w}^L$. Interestingly, this path leaves the loss constant. In other words, from certain points in parameter space it is necessary to follow a path of constant loss until we reach a point from where we can further decrease the loss; just like in the case of the non-attracting regions of local minima.

Lemma 5. *For $n \geq 2$, let $\{r_1, r_2, \ldots, r_n\}$ be a set of vectors in $Im(\sigma)^N$ and $E = \operatorname{span}_j(r_j)$ their linear span. If $z \in E$ has a representation $z = \sum_j \lambda_j r_j$ where all λ_j are positive (or all negative), then there are continuous paths $r_j : [0,1] \to r_j(t)$ of vectors in $Im(\sigma)^N$ such that the following properties hold.*

(i) $r_j(0) = r_j$.

(ii) $z \in \operatorname{span}_j(r_j(t))$ *for all t, so that there are continuous paths $t \to \lambda_j(t)$ such that $z = \sum \lambda_j(t) r_j(t)$.*

(iii) *There are $1 \leq j_+, j_- \leq n$ such that $\lambda_{j_+}(1) > 0$ and $\lambda_{j_-}(1) < 0$.*

We apply Lemma 5 to activation vectors $r_i = \mathbf{a}_i$ giving continuous paths $t \to \mathbf{a}_i^{L-1}(t)$ and $t \to \lambda_i(t) = w_{1,i}^L(t)$. Then the output $f(x_\alpha)$ of the neural network along this path remains constant, hence so does the loss. The desired change of activation vectors $\mathbf{a}_i^{L-1}(t)$ can be performed by a suitable change of parameters $\mathbf{w}^{L-1}$ according to Lemma 3 and Lemma 4. The simultaneous change of $\mathbf{w}^{L-1}$ and $\mathbf{w}^L$ defines the first part $\Gamma_1(t)$ of our desired path in the parameter space which keeps $f(x_\alpha)$ constant. The final part of the desired path is given by the following lemma.

Lemma 6. *Assume a neural network structure as above with activation vectors $\boldsymbol{a}_i^{L-2}$ of the wide hidden layer spanning $\mathbb{R}^N$. If the weights $\mathbf{w}^L$ of the output layer satisfy that there is both a positive and a negative weight, then there is a continuous path $t \in [0,1] \to \Gamma_0(t)$ from the current weights $\Gamma_0(0) = \mathbf{w}$ of decreasing loss down to the global minimum at $\Gamma_0(1)$.*

Proof. Fix $z_\alpha = f(x_\alpha)$, the prediction for the current weights. The main idea is to change the activation vectors of the last hidden layer according to

$$\rho^j : t \in [0,1] \to \mathbf{a}_j^{L-1} + t \cdot \frac{1}{w_{\bullet,j}^L} \cdot \frac{(y-z)}{N}.$$

With $\mathbf{w}^L$ fixed, at the output this results in a change of $t \in [0,1] \to z + t \cdot (y-z)$, which reduces the loss to zero. The required change of activation vectors can be implemented by an application of Lemma 3 and Lemma 4, but only if the image of each ρ^j lies in the image $[c,d]$ of the activation function. Hence, the latter must be arranged.

In the case that $0 \in (c,d)$, it suffices to first decrease the norm of $\mathbf{a}_j^{L-1}$ while simultaneously increasing the norm of the outgoing weight $w_{\bullet,j}^L$ so that the output remains constant. If, however, 0 is in the boundary of the interval $[c,d]$ (for example the case of a sigmoid activation function), then the assumption of non-zero weights with different signs becomes necessary. We let

$$J_+ = \{j \in \{1,2,\ldots,n_{L-1}\} \mid \mathbf{w}_{\bullet,j}^L \geq 0\}, \qquad J_- = \{j \in \{1,2,\ldots,n_{L-1}\} \mid \mathbf{w}_{\bullet,j}^L < 0\},$$
$$I_+ = \{\alpha \in \{1,2,\ldots,N\} \mid (y-z)_\alpha \geq 0\}, \qquad I_- = \{\alpha \in \{1,2,\ldots,N\} \mid (y-z)_\alpha < 0\}.$$

We further define $(y-z)_{I_+}$ to be the vector v with coordinate v_α for $\alpha \in I_+$ equal to $(y-z)_\alpha$ and 0 otherwise, and we let analogously $(y-z)_{I_-}$ denote the vector containing only the negative coordinates of $y-z$. Then the paths $\rho^j : [0,1] \to (c,d)$ defined by

$$\rho_3^j(t) = \mathbf{a}_j^{L-1} + t \cdot \frac{1}{w_{\bullet,j}^L} \cdot \frac{(y-z)_{I_+}}{|J_+|}$$

and for each $j \in J_-$ by

$$\rho_3^j(t) = \mathbf{a}_j^{L-1} + t \cdot \frac{1}{w_{\bullet,j}^L} \cdot \frac{(y-z)_{I_-}}{|J_-|}$$

can be arranged to all lie in the image of the activation function and they again lead to an output change of $t \in [0,1] \to z + t \cdot (y - z)$. (Appendix D contains a more detailed proof.) □

This concludes the proof of Theorem 3 having found a sufficient condition in Lemma 6 to confirm the existence of a path down to zero loss and having shown how to realize this condition in Lemmas 3, 4 and 5.

VII. Conclusion

In this paper we have studied the local minima of deep and wide regression neural networks with sigmoid activation functions. We established that the nature of local minima is such that they live in a special region of the cost function called a non-attractive region, and showed that a non-increasing path to a configuration with lower loss than that of the region can always be found. For sufficiently wide two- or three-layer neural networks, all local minima belong to such a region. We generalized the procedure to find such regions, introduced by Fukumizu and Amari [9], to deep networks and described sufficient conditions for the construction to work. The necessary conditions become very hard to satisfy in wider and deeper networks and, if they fail, the construction leads to saddle points instead. Finally, an intuitive argument shows a clear relation between the degree of degeneracy of a local minimum and the level of suboptimality of the constructed local minimum.

References

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*. Cambridge, MA, USA: MIT Press, 1986, pp. 318–362.

[2] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv e-prints*, vol. arXiv:1611.03530, 2017.

[3] Q. Nguyen and M. Hein, "The loss surface of deep and wide neural networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, 2017.

[4] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 1, pp. 76–86, 1992.

[5] T. Poston, C.-N. Lee, Y. Choie, and Y. Kwon, "Local minima and back propagation," in *IJCNN-91-Seattle International Joint Conference on Neural Networks*, vol. ii, 1991, pp. 173–176.

[6] I. G. Sprinkhuizen-Kuyper and E. J. W. Boers, "A local minimum for the 2-3-1 xor network," *IEEE Transactions on Neural Networks*, vol. 10, no. 4, pp. 968–971, 1999.

[7] L. F. Wessels and E. Barnard, "Avoiding false local minima by proper initialization of connections," *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 899–905, 1992.

[8] H. Wei, J. Zhang, F. Cousseau, T. Ozeki, and S. Amari, "Dynamics of learning near singularities in layered networks," *Neural Computation*, vol. 20, no. 3, pp. 813–843, 2008.

[9] K. Fukumizu and S. Amari, "Local minima and plateaus in hierarchical structures of multilayer perceptrons," *Neural Networks*, vol. 13, no. 3, pp. 317–327, 2000.

[10] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *Advances in Neural Information Processing Systems 27, NIPS 2014*, 2014, pp. 2933–2941.

[11] A. Choromanska, M. Henaff, M. Mathieu, and Y. L. Gérard Ben Arous, "The loss surfaces of multilayer networks," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015*, 2015.

[12] R. Vidal, J. Bruna, R. Giryes, and Stefan, "Mathematics of deep learning," *arXiv e-prints*, vol. arXiv:1712.04741, 2017.

[13] A. L. Blum and R. L. Rivest, "Training a 3-node neural network is NP-complete," *Neural Networks*, vol. 5, no. 1, pp. 117–127, 1992.

[14] P. Auer, M. Herbster, and M. K. Warmuth, "Exponentially many local minima for single neurons," in *Advances in Neural Information Processing Systems 8, NIPS 1995*, 1995, pp. 316–322.

[15] T. Nitta, "Resolution of singularities introduced by hierarchical structure in deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2282–2293, 2017.

[16] C. Sminchisescu and B. Triggs, "Building Roadmaps of Minima and Transitions in Visual Models," *International Journal of Computer Vision*, vol. 61, no. 1, 2005.

[17] L. F. Wessels, E. Barnard, and E. van Rooyen, "The physical correlates of local minima," in *International Neural Network Conference*, 1990, pp. 985–985.

[18] X. G. Wang, Z. Tang, H. Tamura, M. Ishii, and W. D. Sun, "An improved backpropagation algorithm to avoid the local minima problem," *Neurocomputing*, vol. 56, pp. 455–460, 2004.

[19] E. Mizutani and S. Dreyfus, "An analysis on negative curvature induced by singularity in multi-layer neural-network learning," in *Advances in Neural Information Processing Systems 23, NIPS 2010*, 2010, pp. 1669–1677.

[20] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, no. 2, pp. 251–276, 1998.

[21] A. J. Bray and D. S. Dean, "The statistics of critical points of gaussian fields on large-dimensional spaces," *Physical Review Letters*, vol. 98, p. 150201, 1989.

[22] L. Sagun, a. G. B. A. V. Ugur Güney, and Y. LeCun, "Exploration on high dimensional landscapes," *arXiv e-prints*, vol. arXiv:1412.6615, 2015.

[23] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao, "Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review," *International Journal of Automation and Computing*, vol. 14, no. 5, pp. 503–519, 2017.

[24] D. Soudry and E. Hoffer, "Exponentially vanishing sub-optimal local minima in multilayer neural networks," *arXiv e-prints*, vol. arXiv:1702.05777, 2017.

[25] C. D. Freeman and J. Bruna, "Topology and geometry of half-rectified network optimization," *arXiv e-prints*, vol. arXiv:1611.01540, 2017.

[26] G. Swirszcz, W. M. Czarnecki, and R. Pascanu, "Local minima in training of neural networks," *arXiv e-prints*, vol. arXiv:1611.06310, 2016.

[27] B. Xie, Y. Liang, and L. Song, "Diversity leads to generalization in neural networks," *arXiv e-prints*, vol. arXiv:1611.03131, 2016.

[28] D. Soudry and Y. Carmon, "No bad local minima: Data independent training error guarantees for multilayer neural networks," *arXiv e-prints*, vol. arXiv:1605.08361, 2016.

[29] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Networks*, vol. 2, no. 1, pp. 53–58, 1989.

[30] K. Kawaguchi, "Deep learning without poor local minima," in *Advances in Neural Information Processing Systems 29, NIPS 2016*, 2016, pp. 586–594.

[31] H. Lu and K. Kawaguchi, "Depth creates no bad local minima," *arXiv e-prints*, vol. arXiv:1702.08580, 2017.

[32] C. Yun, S. Sra, and A. Jadbabaie, "Global optimality conditions for deep nerural networks," *arXiv e-prints*, vol. arXiv:1707.02444, 2017.

[33] I. Safran and O. Shamir, "On the quality of the initial basin in overspecified neural networks," in *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016*, 2016, pp. 774–782.

[34] B. D. Haeffele and R. Vidal, "Global optimality in neural network training," *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 17*, pp. 4390–4398, 2017.

[35] Q. Liao and T. Poggio, "Theory of deep learning II: Landscape of the empirical risk in deep learning," *arXiv e-prints*, vol. arXiv:1703.09833, 2017.

[36] M. L. Brady, R. Raghavan, and J. Slawny, "Back propagation fails to separate where perceptrons succeed," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 5, pp. 665–674, 2006.

APPENDIX
NOTATION

$[x_\alpha]_\alpha$	$\mathbb{R}^n$	column vector with entries $x_\alpha \in \mathbb{R}$
$[x_{i,j}]_{i,j}$	$\in \mathbb{R}^{n_1 \times n_2}$	matrix with entry $x_{i,j}$ at position (i,j)
Im(f)	$\subseteq \mathbb{R}$	image of a function f
$C^n(X,Y)$		n-times continuously differentiable function from X to Y
N	$\in \mathbb{N}$	number of data samples in training set
x_α	$\in \mathbb{R}^{n_0}$	training sample input
y_α	$\in \mathbb{R}$	target output for sample x_α
$\mathcal{A}$	$\in \mathbf{C}(\mathbb{R})$	class of real-analytic, strictly monotonically increasing, bounded (activation) functions such that the closure of the image contains zero
σ	$\in C^2(\mathbb{R},\mathbb{R})$	a nonlinear activation function in class $\mathcal{A}$
f	$\in C(\mathbb{R}^{n_0},\mathbb{R})$	neural network function
l	$1 \leq l \leq L$	index of a layer
L	$\in \mathbb{N}$	number of layers excluding the input layer
l=0		input layer
$l = L$		output layer
n_l	$\in \mathbb{N}$	number of neurons in layer l
k	$1 \leq k \leq n_l$	index of a neuron in layer l
$\mathbf{w}^l$	$\in \mathbb{R}^{n_l \times n_{l-1}}$	weight matrix of the l-th layer
$\mathbf{w}$	$\in \mathbb{R}^{\sum_{l=1}^{L}(n_l \cdot n_{l-1})}$	collection of all $\mathbf{w}^l$
$w^l_{i,j}$	$\in \mathbb{R}$	the weight from neuron j of layer $l-1$ to neuron j of layer l
$w^L_{\bullet,j}$	$\in \mathbb{R}$	the weight from neuron j of layer $L-1$ to the output
$\mathcal{L}$	$\in \mathbb{R}_+$	squared loss over training samples
$\mathsf{n}(l,k;x)$	$\in \mathbb{R}$	value at neuron k in layer l before activation for input pattern x
$\mathsf{n}(l;x)$	$\in \mathbb{R}^{n_l}$	neuron pattern at layer l before activation for input pattern x
$\mathsf{act}(l,k;x)$	$\in \mathrm{Im}(\sigma)$	activation pattern at neuron k in layer l for input x
$\mathsf{act}(l;x)$	$\in \mathrm{Im}(\sigma)^{n_l}$	neuron pattern at layer l for input x

In Section V, where we fix a layer l, we additionally use the following notation.

$h_{\bullet,k}(x)$	$\in C(\mathbb{R}^{n_l}, \mathbb{R})$	the function from $\mathsf{act}(l;x)$ to the output
$[u_{p,i}]_{p,i}$	$\in \mathbb{R}^{n_l \times n_{l-1}}$	weights of the given layer l.
$[v_{s,q}]_{s,q}$	$\in \mathbb{R}^{n_l \times n_{l+1}}$	weights the layer $l+1$.
r	$\in \{1, 2, \ldots, n_l\}$	the index of the neuron of layer l that we use for the addition of one additional neuron
M	$\in \mathbb{N}$	$= \sum_{t=1}^{L}(n_t \cdot n_{t-1})$, the number of weights in the smaller neural network
$\bar{\mathbf{w}}$	$\in \mathbb{R}^{M-n_{l-1}-n_{l+1}}$	all weights except $u_{1,i}$ and $v_{s,1}$
γ_λ^r	$\in \mathbf{C}(\mathbb{R}^M, \mathbb{R}^{M+n_{l-1}+n_{l+1}})$	the map defined in Section V to add a neuron in layer l using the neuron with index r in layer l

In Section VI, we additionally use the following notation.

$\mathbf{a}_k^l$	$\in \mathrm{Im}(\sigma)^N$	activation vector at neuron k in layer l given by $\mathbf{a}_k^l = [\mathsf{act}(l,k;x_\alpha)]_\alpha$
Γ	$\in \mathbf{C}([0,1], \mathbb{R}^M)$	a path in parameter space
ρ	$\in \mathbf{C}([0,1], \mathbb{R}^{N \times n_l})$	a path of activation values in layer l
ν	$\in \mathbf{C}([0,1], \mathbb{R}^{N \times n_l})$	a path of neuron values in layer l

PROOFS

A. Local minima at infinity in neural networks

In this section we prove the existence of local minima at infinity in neural networks.

Theorem 1 (cf. [6] Section III)**.** *Let $\mathcal{L}$ denote the squared loss of a fully connected regression neural network with sigmoid activation functions, having at least one hidden layer and each hidden layer containing at least two neurons. Then, for almost every finite dataset, the loss function $\mathcal{L}$ possesses a local minimum at infinity. The local minimum is suboptimal whenever dataset and neural network are such that a constant function is not an optimal solution.*

Proof. We will show that, if all bias terms $u_{i,0}$ of the last hidden layer are sufficiently large, then there are parameters $u_{i,0k}$ for $k \neq 0$ and parameters v_i of the output layer such that the minimal loss is achieved at $u_{i,0} = \infty$ for all i.

We note that, if $u_{i,0} = \infty$ for all i, all neurons of the last hidden layer are fully active for all samples, i.e. $\mathsf{act}(L-1, i; x_\alpha) = 1$ for all i. Therefore, in this case $f(x_\alpha) = \sum_i v_{\bullet,i}$ for all α. A constant function $f(x_\alpha) = \sum_i v_{\bullet,i} = c$ minimizes the loss $\sum_\alpha (c - y_\alpha)^2$ uniquely for $c := \frac{1}{N}\sum_{\alpha=1}^{N} y_\alpha$. We will assume that the $v_{\bullet,i}$ are chosen such that $\sum_i v_{\bullet,i} = c$ does hold. That is, for fully active hidden neurons at the last hidden layer, the $v_{\bullet,i}$ are chosen to minimize the loss.

We write $f(x_\alpha) = c + \epsilon_\alpha$. Then

$$\mathcal{L} = \frac{1}{2}\sum_\alpha (f(x_\alpha) - y_\alpha)^2 = \frac{1}{2}\sum_\alpha (c + \epsilon_\alpha - y_\alpha)^2$$

$$= \frac{1}{2}\sum_\alpha (\epsilon_\alpha + (c - y_\alpha))^2$$

$$= \underbrace{\frac{1}{2}\sum_\alpha (c - y_\alpha)^2}_{\text{Loss at } u_{i,0} = \infty \text{ for all } i} + \underbrace{\frac{1}{2}\sum_\alpha \epsilon_\alpha^2}_{\geq 0} + \underbrace{\sum_\alpha \epsilon_\alpha (c - y_\alpha)}_{(*)}.$$

The idea is now to ensure that $(*) \geq 0$ for sufficiently large $u_{i,0}$ and in a neighborhood of the $v_{\bullet,i}$ chosen as above. Then the loss $\mathcal{L}$ is larger than at infinity, and any point in parameter space with $u_{i,0} = \infty$ and $v_{\bullet,i}$ with $\sum_i v_{\bullet,i} = c$ is a local minimum.

To study the behavior at $u_{i,0} = \infty$, we consider $p_i = \exp(-u_{i,0})$. Note that $\lim_{u_{i,0}\to\infty} p_i = 0$. We have

$$f(x_\alpha) = \sum_i v_{\bullet,i}\sigma(u_{i,0} + \sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha))$$

$$= \sum_i v_{\bullet,i} \cdot \frac{1}{1 + p_i \cdot \exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha))}$$

Now for p_i close to 0 we can use Taylor expansion of $g_i^j(p_i) := \frac{1}{1+p_i exp(a_i^j)}$ to get $g_i^j(p_i) = 1 - \exp(a_i^j)p_i + \mathcal{O}(|p_i|^2)$. Therefore

$$f(x_\alpha) = c - \sum_i v_{\bullet,i}p_i \exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha)) + \mathcal{O}(p_i^2)$$

and we find that $\epsilon_\alpha = -\sum_i v_{\bullet,i}p_i \exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha)) + \mathcal{O}(p_i^2)$.

Recalling that we aim to ensure

$$(*) = \sum_\alpha \epsilon_\alpha(c - y_\alpha) \geq 0$$

we consider

$$\sum_\alpha \epsilon_\alpha(c-y_\alpha) = -\sum_\alpha (c-y_\alpha)(\sum_i v_{\bullet,i}p_i \exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha))) + \mathcal{O}(p_i^2)$$

$$= -\sum_i v_{\bullet,i}p_i \sum_\alpha (c-y_\alpha)\exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha)) + \mathcal{O}(p_i^2)$$

We are still able to choose the parameters $u_{i,k}$ for $i \neq 0$, the parameters from previous layers, and the $v_{\bullet,i}$ subject to $\sum_i v_{\bullet,i} = c$. If now $v_{\bullet,i} > 0$ whenever $\sum_\alpha (c-y_\alpha)\exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha)) < 0$ and $v_{\bullet,i} < 0$ whenever $\sum_\alpha (c-y_\alpha)\exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha)) > 0$, then the term $(*)$ is strictly positive, hence the overall loss is larger than the loss at $p_i = 0$ for sufficiently small p_i and in a neighborhood of $v_{\bullet,i}$. The only obstruction we have to get around is the case where we need all $v_{\bullet,i}$ of the opposite sign of c (in other words, $\sum_\alpha (c-y_\alpha)\exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha))$ has the same sign as c), conflicting with $\sum_i v_{\bullet,i} = c$. To avoid this case, we impose the mild condition that $\sum_\alpha (c-y_\alpha)\mathsf{act}(L-2,r;x_\alpha) \neq 0$ for some r, which can be arranged to hold for almost every dataset by fixing all parameters of layers with index smaller than $L-2$. By Lemma 7 below (with $d_\alpha = (c-y_\alpha)$ and $a_\alpha^r = \mathsf{act}(L-2,r;x_\alpha)$), we can find $u_k^>$ such that $\sum_\alpha (c-y_\alpha)\exp(-\sum_k u_k^>\mathsf{act}(L-2,k;x_\alpha)) > 0$ and $u_k^<$ such that $\sum_\alpha (c-y_\alpha)\exp(-\sum_k u_k^<\mathsf{act}(L-2,k;x_\alpha)) < 0$. We fix $u_{i,k}$ for $k \geq 0$ such that there is some i_1 with $[u_{i_1,k}]_k = [u_k^>]_k$ and some i_2 with $[u_{i_2,k}]_k = [u_k^<]_k$. This assures that we can choose the $v_{\bullet,i}$ of opposite sign to $\sum_\alpha (c-y_\alpha)\exp(-\sum_k u_{i,k}\mathsf{act}(L-2,k;x_\alpha))$ and such that $\sum_i v_{\bullet,i} = c$, leading to a local minimum at infinity.

The local minimum is suboptimal whenever a constant function is not the optimal network function for the given dataset.

□

Lemma 7. *Suppose that m is a positive integer, $m \geq 2$, and for $\alpha = 1,\ldots,N$ and $r = 1,\ldots,m$ we have numbers d_α, a_α^r in $\mathbb{R}$ such that*

$$\sum_{\alpha=1}^N d_\alpha = 0, \text{ and } \sum_\alpha d_\alpha a_\alpha^r \neq 0 \text{ for some } r.$$

Then there are $u_k^<, k = 1,2,...,m$ such that

$$\sum_\alpha d_\alpha \exp(-\sum_k u_k^< a_\alpha^k) < 0$$

and $u_k^>, k = 1,2,...,m$ such that

$$\sum_\alpha d_\alpha \exp(-\sum_k u_k^> a_\alpha^k) > 0.$$

Proof. Consider the function

$$\phi(u_1, u_2, \ldots, u_m) := \sum_\alpha d_\alpha \exp(-\sum_k u_k a_\alpha^k).$$

We have

$$\phi(0, 0, \ldots, 0) = \sum_\alpha d_\alpha = 0.$$

Further

$$\frac{\partial \phi}{\partial u_r}_{|(0,0,\ldots,0)} = -\sum_\alpha d_\alpha a_\alpha^r.$$

By assumption, there is r such that the last term is nonzero. Hence, using coordinate r, we can choose $w = (0, 0, \ldots, 0, w_r, 0, \ldots, 0)$ such that $\phi(w)$ is positive and we can choose w such that $\phi(w)$ is negative. □

B. Proofs for the construction of local minima

Here we prove

Theorem 6. *Consider two (possibly deep) neural networks as in Section III-A, which differ by one neuron in layer l with index $\mathsf{n}(l, -1; x)$ in the larger network. Assume that the parameter choices $([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ determine a local minimum for the squared loss over a finite dataset in the smaller network. If the matrix $[B^r_{i,j}]_{i,j}$ defined by*

$$B^r_{i,j} := \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x_\alpha))}{\partial \mathsf{n}(l+1, k; x_\alpha)} \cdot v^*_{k,r} \cdot \sigma''(\mathsf{n}(l, r; x_\alpha)) \cdot \mathsf{act}(l-1, i; x_\alpha) \cdot \mathsf{act}(l-1, j; x_\alpha) \tag{1}$$

is either

- *positive definite and $\lambda \in \mathcal{I} := (0, 1)$, or*
- *negative definite and $\lambda \in \mathcal{I} := (-\infty, 0) \cup (1, \infty)$,*

then $\left\{\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*) \mid \lambda \in \mathcal{I}\right\}$ determines a non-attracting region of local minima in the larger network if and only if

$$D^{r,s}_i := \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x_\alpha))}{\partial \mathsf{n}(l+1, s; x_\alpha)} \cdot \sigma'(\mathsf{n}(l, r; x_\alpha)) \cdot \mathsf{act}(l-1, i; x_\alpha) \tag{2}$$

is zero, $D^{r,s}_i = 0$, for all i, s.

The previous theorem follows from two lemmas, with the first lemma containing the computation of the Hessian of the cost function $\mathcal{L}$ of the larger network at parameters $\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ with respect to a suitable basis.

In addition, to find local minima one needs to explain away all additional directions, i.e., we need to show that the loss function actually does not change into the direction of eigenvectors of the Hessian with eigenvalue 0. Otherwise a higher derivative into this direction could be nonzero and potentially lead to a saddle point (see [19]).

Lemma 1. *Consider two (possibly deep) neural networks as in Section III-A, which differ by one neuron in layer l with index $\mathsf{n}(l, -1; x)$ in the larger network. Fix $1 \leq r \leq n_l$. Assume that the parameter choices $([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ determine a critical point in the smaller network.*

Let $\mathcal{L}$ denote the the loss function of the larger network and ℓ the loss function of the smaller network. Let $\alpha \neq -\beta \in \mathbb{R}$ such that $\lambda = \frac{\beta}{\alpha+\beta}$.

With respect to the basis of the parameter space of the larger network given by $([u_{-1,i} + u_{r,i}]_i, [v_{s,-1} + v_{s,r}]_s, \bar{\mathbf{w}}, [\alpha \cdot u_{-1,i} - \beta \cdot u_{r,i}]_i, [v_{s,-1} - v_{s,r}]_s)$, the Hessian of $\mathcal{L}$ (i.e., the second derivative with respect to the new network parameters) at $\gamma^r_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$ is given by

$$\begin{pmatrix} [\frac{\partial^2 \ell}{\partial u_{r,i} \partial u_{r,j}}]_{i,j} & 2[\frac{\partial^2 \ell}{\partial u_{r,i} \partial v_{s,r}}]_{i,s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{r,i}}]_{i,\bar{\mathbf{w}}} & 0 & 0 \\ 2[\frac{\partial^2 \ell}{\partial u_{r,i} \partial v_{s,r}}]_{s,i} & 4[\frac{\partial^2 \ell}{\partial v_{s,r} \partial v_{t,r}}]_{s,t} & 2[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,r}}]_{s,\bar{\mathbf{w}}} & (\alpha - \beta)[D^{r,s}_i]_{s,i} & 0 \\ [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{r,i}}]_{\bar{\mathbf{w}},i} & 2[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,r}}]_{\bar{\mathbf{w}},s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial \bar{\mathbf{w}}'}]_{\bar{\mathbf{w}},\bar{\mathbf{w}}'} & 0 & 0 \\ 0 & (\alpha - \beta)[D^{r,s}_i]_{i,s} & 0 & \alpha\beta[B^r_{i,j}]_{i,j} & (\alpha + \beta)[D^{r,s}_i]_{i,s} \\ 0 & 0 & 0 & (\alpha + \beta)[D^{r,s}_i]_{s,i} & 0 \end{pmatrix}$$

Proof. The proof only requires a tedious, but not complicated calculation (using the relation $\alpha\lambda - \beta(1-\lambda) = 0$ multiple times. To keep the argumentation streamlined, we moved all the necessary calculations into Appendix E. □

Lemma 8. *Let $a, b, c, d, e, f, g, h, x$ be matrices of appropriate sizes.*

(a) *A matrix of the form*

$$\begin{pmatrix} a & 2b & c & 0 \\ 2b^T & 4d & 2e & 0 \\ c^T & 2e^T & f & 0 \\ 0 & 0 & 0 & x \end{pmatrix}$$

is positive semidefinite if and only if both x and the matrix

$$\begin{pmatrix} a & b & c \\ b^T & d & e \\ c^T & e^T & f \end{pmatrix}$$

are positive semidefinite.

(b) *A matrix x of the form*

$$x = \begin{pmatrix} g & h \\ h^T & 0 \end{pmatrix}$$

is positive semidefinite if and only if g is positive semidefinite and $h = 0$.

Proof. (a) By definition, a matrix A is positive semidefinite if and only if $z^T A z \geq 0$ for all z. Note now that

$$(z_1, z_2, z_3, z_4) \begin{pmatrix} a & 2b & c & 0 \\ 2b^T & 4d & 2e & 0 \\ c^T & 2e^T & f & 0 \\ 0 & 0 & 0 & x \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = (z_1, 2z_2, z_3, z_4) \begin{pmatrix} a & b & c & 0 \\ b^T & d & e & 0 \\ c^T & e^T & f & 0 \\ 0 & 0 & 0 & x \end{pmatrix} \begin{pmatrix} z_1 \\ 2z_2 \\ z_3 \\ z_4 \end{pmatrix}$$

(b) It is clear that the matrix x is positive semidefinite for g positive semidefinite and $h = 0$. To show the converse, first note that if g is not positive semidefinite and z is such that $z^T g z < 0$ then

$$(z^T, 0) \begin{pmatrix} g & h \\ h^T & 0 \end{pmatrix} \begin{pmatrix} z \\ 0 \end{pmatrix} = z^T g z < 0.$$

It therefore remains to show that also $h = 0$ is a necessary condition. Assume $h \neq 0$ and find z such that $hz \neq 0$. Then for any $\lambda \in \mathbb{R}$ we have

$$\begin{aligned} ((hz)^T, -\lambda z^T) \begin{pmatrix} g & h \\ h^T & 0 \end{pmatrix} \begin{pmatrix} hz \\ -\lambda z \end{pmatrix} &= (hz)^T g(hz) - 2(hz)^T h\lambda z \\ &= (hz)^T g(hz) - 2\lambda ||hz||_2^2. \end{aligned}$$

For sufficiently large λ, the last term is negative.

□

Proof of Theorem 6. In Lemma 1, we calculated the Hessian of $\mathcal{L}$ with respect to a suitable basis at a the critical point
$\gamma_\lambda([u^*_{r,i}]_i, [v^*_{s,r}]_s, \bar{\mathbf{w}}^*)$. If the matrix $[D_i^{r,s}]_{i,s}$ is nonzero, then by Lemma 8(b) the Hessian is not positive semidefinite, hence none of the critical points are local minima.

If, on the other hand, the matrix $[D_i^{r,s}]_{i,s}$ is zero, then by Lemma 8(a+b) the Hessian is positive semidefinite, since

$$\begin{pmatrix} [\frac{\partial^2 \ell}{\partial u_{r,i} \partial u_{r,j}}]_{i,j} & [\frac{\partial^2 \ell}{\partial u_{r,i} \partial v_{s,r}}]_{i,s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{r,i}}]_{i,\bar{\mathbf{w}}} \\ [\frac{\partial^2 \ell}{\partial u_{r,i} \partial v_{s,r}}]_{s,i} & [\frac{\partial^2 \ell}{\partial v_{s,r} \partial v_{t,1}}]_{s,t} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,r}}]_{s,\mathbf{w}} \\ [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{r,i}}]_{\bar{\mathbf{w}},i} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,r}}]_{\bar{\mathbf{w}},s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial \bar{\mathbf{w}}'}]_{\bar{\mathbf{w}},\bar{\mathbf{w}}'} \end{pmatrix}$$

is positive semidefinite by assumption, and $\alpha\beta[B^r_{i,j}]_{i,j}$ is positive definite if $\lambda \in (0,1) \Leftrightarrow \alpha\beta > 0$ and $[B^r_{i,j}]_{i,j}$ is positive definite or if ($\lambda < 0$ or $\lambda > 1$) $\Leftrightarrow \alpha\beta < 0$ and $[B^r_{i,j}]_{i,j}$ is negative definite. In each case we can alter the

λ to values leading to saddle points without changing the network function or loss. Therefore, the critical points can only be saddle points or local minima on a non-attracting region of local minima.

To determine whether the critical points in questions lead to local minima when $[D_i^{r,s}]_{i,s} = 0$, it is insufficient to only prove the Hessian to be positive semidefinite (in contrast to (strict) positive definiteness), but we need to consider directions for which the second order information is insufficient. We know that the loss is at a minimum with respect to all coordinates except for the degenerate directions $[v_{s,-1} - v_{s,r}]_s$. However, the network function $f(x)$ is constant along $[v_{s,-1} - v_{s,r}]_s$ (keeping $[v_{s,-1} + v_{s,r}]_s$ constant) at the critical point where $u_{-1,i} = u_{r,i}$ for all i. Hence, no higher order information leads to saddle points and it follows that the critical point lies on a region of local minima. □

C. Construction of local minima in deep networks

Proposition 1. *Suppose we have a hierarchically constructed critical point of the squared loss of a neural network constructed by adding a neuron into layer l with index $\mathsf{n}(l,-1;x)$ by application of the map γ_λ^r to a neuron $\mathsf{n}(l,r;x)$. Suppose further that for the outgoing weights $v_{s,r}^*$ of $\mathsf{n}(l,r;x)$ we have $\sum_s v_{s,r}^* \neq 0$, and suppose that $D_i^{r,s}$ is defined as in* (2)*. Then $D_i^{r,s} = 0$ if one of the following holds.*

(i) *The layer l is the last hidden layer. (This condition includes the case $l = 1$ indexing the hidden layer in a two-layer network.)*

(ii)
$$\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)} = \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,t;x_\alpha)}$$
for all s,t,α

(iii) *For each α and each t, with $\mathcal{L}_\alpha := (f(x_\alpha) - y_\alpha)^2$,*
$$\frac{\partial \mathcal{L}_\alpha}{\partial \mathsf{n}(l+1,t;x_\alpha)} = (f(x_\alpha) - y_\alpha) \cdot \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha)}{\partial \mathsf{n}(l+1,t;x_\alpha)} = 0.$$
(This condition holds in the case of the weight infinity attractors in the proof to Theorem 1 for $l+1$ the second last layer. It also holds in a global minimum.)

Proof. The fact that property (i) suffices uses that $h_{\bullet,l+1}(x)$ reduces to the identity function on the networks output and hence its derivative is one. Then, considering a regression network as before, our assumption says that $v_{\bullet,r}^* \neq 0$, hence its reciprocal can be factored out of the sum in Equation (2). Denoting incoming weights into $\mathsf{n}(l,r;x)$ by $u_{r,i}$ as before, this leads to

$$\begin{aligned} D_i^{r,1\bullet} &= \frac{1}{v_{\bullet,r}^*} \cdot \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot v_{\bullet,r}^* \cdot \sigma'(\mathsf{n}(l,r;x_\alpha)) \cdot \mathsf{act}(l-1,i;x_\alpha) \\ &= \frac{1}{v_{\bullet,r}^*} \cdot \frac{\partial \mathcal{L}}{\partial u_{r,i}} = 0 \end{aligned}$$

In the case of (ii),
$$\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)} = \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,t;x_\alpha)}$$
for all s,t and we can factor out the reciprocal of $\sum_t v_{r,s}^* \neq 0$ in Equation (2) to again see that for each i, $\frac{\partial \mathcal{L}}{\partial u_{r,i}} = 0$ implies that $D_i^{r,s} = 0$ for all s.

(iii) is evident since in this case clearly every summand in Equation (2) is zero. □

D. Proofs for the non-increasing path to a global minimum

In this section we discuss how in wide neural networks with two hidden layers a non-increasing path to the global minimum may be found from almost everywhere in the parameter space. By [3] (and [4], [5]), we can find such a path if the last hidden layer is wide (containing more neurons than input patterns). We therefore only consider the case where the first hidden layer in a three-layer neural network is wide. More generally, our results apply to all deep neural networks with the second last hidden layer wide.

Theorem 3. *Consider a fully connected regression neural network with activation function in the class $\mathcal{A}$ equipped with the squared loss function for a finite dataset. Assume that the second last hidden layer contains more neurons than the number of input patterns. Then, for each set of parameters $\mathbf{w}$ and all $\epsilon > 0$, there is $\mathbf{w}'$ such that $||\mathbf{w} - \mathbf{w}'|| < \epsilon$ and such that a path non-increasing in loss from $\mathbf{w}'$ to a global minimum where $f(x_\alpha) = y_\alpha$ for each α exists.*

The first step of the proof is to use the freedom given by ϵ to have the activation vectors $\mathbf{a}^{L-2}$ of the wide layer $L-2$ span the whole space $\mathbb{R}^N$.

Lemma 2. *[3, Corollary 4.5] For each choice of parameters $\mathbf{w}$ and all $\epsilon > 0$ there is $\mathbf{w}'$ such that $||\mathbf{w} - \mathbf{w}'|| < \epsilon$ and for the activation vectors $\boldsymbol{a}_k^{L-2}$ of the wide layer $L-2$ for $\mathbf{w}'$ we have*

$$\operatorname{span}_k \boldsymbol{a}_k^{L-2} = \mathbb{R}^N.$$

Lemma 3. *Assume that in the wide layer $L-2$ we have that the activation vectors satisfy*

$$\operatorname{span}_k[\boldsymbol{a}_k^{L-2}] = \mathbb{R}^N.$$

Then for any continuous paths $\nu : [0,1] \to \mathbb{R}^{n_{L-1} \times N}$ in layer $L-1$ with $\nu(0) = [\mathsf{n}(L-1, s; x_\alpha)]_{s,\alpha}$ there is a continuous path of parameters $\Gamma : [0,1] \to \mathbb{R}^{n_{L-1} \times n_{L-2}}$ with $\Gamma(0) = \mathbf{w}^{L-1}$ and such that

$$\nu(t) = \Gamma(t) \cdot [\mathsf{act}(L-2, k; x_\alpha)]_{k,\alpha}$$

Proof. We write $\nu(t) = [\mathsf{n}(L-1, s; x_\alpha)]_{s,\alpha} + \tilde{\nu}(t)$ with $\tilde{\nu}(0) = 0$. We will find $\tilde{\Gamma}(t)$ such that $\tilde{\nu}(t) = \tilde{\Gamma}(t) \cdot [\mathsf{act}(L-2, k; x_\alpha)]_{k,\alpha}$ with $\tilde{\Gamma}(0) = 0$. Then $\Gamma(t) := \mathbf{w}^{L-1} + \tilde{\Gamma}(t)$ does the job.

Since by assumption $[\mathsf{act}(L-2, k; x_\alpha)]_{k,\alpha}$ has full rank, we can find an invertible submatrix $\tilde{A} \in \mathbb{R}^{N \times N}$ of $[\mathsf{act}(L-2, k; x_\alpha)]_{k,\alpha}$. Then we can define a continuous path $\tilde{\rho}$ in $\mathbb{R}^{n_{L-1} \times N}$ given by $\tilde{\rho}(t) := \tilde{\nu}(t) \cdot \tilde{A}^{-1}$, which satisfies $\tilde{\rho}(t) \cdot \tilde{A} = \nu(t)$ and $\tilde{\rho}(0) = 0$. Extending $\tilde{\rho}(t)$ to a path in $\mathbb{R}^{n_{L-1} \times n_{L-2}}$ by zero columns at positions corresponding to rows of $[\mathsf{act}(L-2, k; x_\alpha)]_{k,\alpha}$ missing in $\tilde{A}$, gives a path $\tilde{\Gamma}(t)$ such that $\tilde{\Gamma}(t) \cdot [\mathsf{act}(L-2, k; x_\alpha)]_{k,\alpha} = \tilde{\nu}(t)$ and with $\tilde{\Gamma}(0) = 0$.

□

Lemma 4. *For all continuous paths $\rho(t)$ in $Im(\sigma)^N$, i.e. the N-fold copy of the image of σ, there is a continuous path $\nu(t)$ in $\mathbb{R}^N$ such that $\rho(t) = \sigma(\nu(t))$ for all t.*

Proof. Since $\sigma : \mathbb{R}^N \to \operatorname{Im}(\sigma)^N$ is invertible with a continuous inverse, take

$$\nu(t) = \sigma^{-1}(\rho(t)).$$

□

The activation vectors $\mathbf{a}_k^{L-1}$ of the last hidden layer span a linear subspace H of $\mathbb{R}^N$. The optimal parameters $\mathbf{w}^L$ of the output layer compute the best approximation of $(y_\alpha)_\alpha$ onto H. Lemma 3 and Lemma 4 together imply that we can achieve any desired continuous change of the spanning vectors of H, and hence the linear subspace H, by a suitable change of the parameters $\mathbf{w}^{L-1}$.

There is a natural possible path of parameters that strictly monotonically decreases the loss to the global minimum. For activation functions in $\mathcal{A}$ with 0 in the boundary of the image interval $[c, d]$, this path requires that not all non-zero coefficients of $\mathbf{w}^L$ have the same sign. If this is not the case, however, we first follow a different path through the parameter space to eventually assure different signs of coefficients of $\mathbf{w}^L$. Interestingly, this path leaves the loss constant. In other words, from certain points in parameter space it seems necessary to follow a path of constant loss until we reach a point from where we can further decrease the loss; just like in the case of the non-attracting regions of local minima.

Lemma 5. *For $n \geq 2$, let $\{r_1, r_2, \ldots, r_n\}$ be a set of vectors in $Im(\sigma)^N$ and $E = \operatorname{span}_j(r_j)$ their linear span. If $z \in E$ has a representation $z = \sum_j \lambda_j r_j$ where all λ_j are positive (or all negative), then there are continuous paths $r_j : [0,1] \to r_j(t)$ of vectors in $Im(\sigma)^N$ such that the following properties hold.*

(i) $r_j(0) = r_j$.

(ii) $z \in \operatorname{span}_j(r_j(t))$ *for all t, so that there are continuous paths $t \to \lambda_j(t)$ such that $z = \sum \lambda_j(t) r_j(t)$.*

(iii) *There are* $1 \leq j_+, j_- \leq n$ *such that* $\lambda_{j_+}(1) > 0$ *and* $\lambda_{j_-}(1) < 0$.

Proof. We only consider the case with all $\lambda_j \geq 0$. The other case can be treated analogously.

If only one λ_{j_0} is nonzero, then consider a vector r_k corresponding to a zero coefficient $\lambda_k = 0$ and change r_k continuously until it equals the vector r_{j_0} corresponding to the only nonzero coefficient. Then continuously increase the positive coefficient λ_{j_0}, while introducing a corresponding negative contribution via λ_k. It is then easy to see that this leads to a path satisfying conditions (i)–(iii).

We may therefore assume that at least two coefficients λ_j are nonzero, say λ_1 and λ_2. Leaving all r_j and λ_j for $j \geq 3$ unchanged, we only consider $r_1, r_2, \lambda_1, \lambda_2$ for the desired path, i.e. $r_j(t) = r_j$ and $\lambda_j(t) = \lambda_j$ for all $j \geq 3$.

We have that $\lambda_1 r_1 + \lambda_2 r_2 \in \big((\lambda_1 + \lambda_2) \cdot \mathrm{Im}(\sigma)^N\big)$, hence can be written as λR for some $\lambda > 0$ and $R \in \mathrm{Im}(\sigma)^N$ with $\lambda R = z - \sum_{j \geq 3} \lambda_j r_j = \lambda_1 r_1 + \lambda_2 r_2$.

For $t \in [0, \frac{1}{2}]$ we define

$$r_1(t) := r_1 + 2t(R - r_1) \text{ and } r_2(t) := r_2,$$

$$\lambda_1(t) = \frac{\lambda \lambda_1}{(1-2t)\lambda + 2t\lambda_1} \text{ and } \lambda_2(t) = (1-2t)\frac{\lambda \lambda_2}{(1-2t)\lambda + 2t\lambda_1} \ .$$

For $t \in [\frac{1}{2}, 1]$ we set

$$r_1(t) := (2-2t)R + (2t-1)\Big(\frac{\lambda_1}{\lambda_1 + 2\lambda_2} r_1 + \frac{2\lambda_2}{\lambda_1 + 2\lambda_2} r_2\Big) \text{ and } r_2(t) = r_2,$$

$$\lambda_1(t) = \frac{\lambda(\lambda_1 + 2\lambda_2)}{(2-2t)(\lambda_1 + 2\lambda_2) + (2t-1)\lambda}$$

$$\text{and } \lambda_2(t) = \frac{-\lambda_2 \lambda (2t-1)}{(2-2t)(\lambda_1 + 2\lambda_2) + (2t-1)\lambda} \ .$$

Then (i) $r_1(0) = r_1$ and $r_2(0) = r_2$ as desired. Further (ii) $z \in \mathrm{span}_j\,(r_j(t))$ for all $t \in [0,1]$ via $z = \sum_j \lambda_j(t) r_j(t)$. It is also easy to check that $r_1(t), r_2(t) \in \mathrm{Im}(\sigma)^N$ for all $t \in [0,1]$. Finally, (iii) $\lambda_1(1) = \lambda_1 + 2\lambda_2 > 0$ and $\lambda_2(1) = -\lambda_2 < 0$. □

Hence, if all non-zero coefficients of $\mathbf{w}^L$ have the same sign, then we apply Lemma 5 to activation vectors $r_i = \mathbf{a}_i^{L-1}$ giving continuous paths $t \to \mathbf{a}_i^{L-1}(t)$ and $t \to \lambda_i(t) = w_{\bullet,i}^L(t)$. Then the output $f(x_\alpha)$ of the neural network along this path remains constant, hence so does the loss. The desired change of activation vectors $\mathbf{a}_i^{L-1}(t)$ can be performed by a suitable change of parameters $\mathbf{w}^{L-1}$ according to Lemma 3 and Lemma 4. The simultaneous change of $\mathbf{w}^{L-1}$ and $\mathbf{w}^L$ defines the first part $\Gamma_1(t)$ of our desired path in the parameter space which keeps $f(x_\alpha)$ constant. We may now assume that not all non-zero entries of $\mathbf{w}^L$ have the same sign. The final part of the desired path is given by the following lemma.

Lemma 6. *Assume a neural network structure as above with activation vectors* $\boldsymbol{a}_i^{L-2}$ *of the wide hidden layer spanning* $\mathbb{R}^N$. *If the weights* $\mathbf{w}^L$ *of the output layer satisfy that there is both a positive and a negative weight, then there is a continuous path* $t \in [0,1] \to \Gamma_0(t)$ *from the current weights* $\Gamma_0(0) = \mathbf{w}$ *of decreasing loss down to the global minimum at* $\Gamma_0(1)$.

Proof. We first prove the result for the (more complicated) case when $\mathrm{Im}(\sigma) = (0, d)$ for some $d > 0$, e.g. for σ the sigmoid function:

Let $z \in \mathbb{R}^N$ be the vector given by $z_\alpha = f(x_\alpha)$ for the parameter $\mathbf{w}$ at the current weights.

Let

$$I_+ = \{\alpha \in \{1, 2, \ldots, N\} \mid (y-z)_\alpha \geq 0\},$$

$$I_- = \{\alpha \in \{1, 2, \ldots, N\} \mid (y-z)_\alpha < 0\}.$$

Clearly $I_+ \cup I_- = \{1, 2, \ldots, N\}$ and $I_+ \cap I_i = \emptyset$. We let $(y-z)_{I_+}$ denote the vector v with coordinate v_α equal to $(y-z)_\alpha$ if $\alpha \in I_+$ and 0 otherwise. Analogously $(y-z)_{I_-}$ denotes the vector containing only the negative coordinates of $y - z$. Hence $y - z = (y-z)_{I_+} + (y-z)_{I_-}$.

We also split up the coefficients of $\mathbf{w}^L$ according to the signs of the coordinates into disjoint sets:

$$J_0 = \{j \in \{1, 2, \ldots, n_{L-1}\} \mid \mathbf{w}_{\bullet,j}^L = 0\},$$

$$J_+ = \{j \in \{1,2,\dots,n_{L-1}\} \mid \mathbf{w}^L_{\bullet,j} \geq 0\},$$
$$J_- = \{j \in \{1,2,\dots,n_{L-1}\} \mid \mathbf{w}^L_{\bullet,j} < 0\}.$$

For each $j \in \{1,2,\dots,n_{L-1}\} \setminus J_0 = J_+ \cup J_-$ we consider the path $\rho_2^j : [0,1) \to (0,d)^N$ of activation values given by

$$\rho_2^j(t) = (1-t)[\mathsf{act}(L-1,j;x_\alpha)]_\alpha.$$

Applying Lemma 3 and Lemma 4 we find the inducing path $\Gamma^j_{2,L-1}$ for parameters $\mathbf{w}^{L-1}$, and we simultaneously change the parameters $\mathbf{w}^L$ via $\mathbf{w}^L_{\bullet,j}(t) = \Gamma^j_{2,L}(t) := \frac{1}{1-t}\mathbf{w}^L_{\bullet,j}$. Following along $\Gamma_2^j(t) = (\Gamma^j_{2,L-1}(t), \Gamma^j_{2,L}(t))$ does not change the outcome $f(x_\alpha) = z_\alpha$ for any α. For $j \in J_+$ we find $t_j \in [0,1)$ such that

$$\rho_2^j(t_j) + \frac{1}{w^L_{\bullet,j}(t_j)} \cdot \frac{(y-z)_{I_+}}{|J_+|} \in (0,d)^N.$$

This is possible, since all involved terms are positive, $\rho_2^j(t_j) < 1$ and decreasing to zero for increasing t, while $w^L_{\bullet,j}(t)$ increases for growing t. Similarly, for $j \in J_-$ we find $t_j \in [0,1)$ such that

$$\rho_2^j(t_j) + \frac{1}{w^L_{\bullet,j}(t_j)} \cdot \frac{(y-z)_{I_-}}{|J_-|} \in (0,d)^N.$$

This time the negative sign of $w^L_{\bullet,j}(t)$ for $j \in J_\cdot$ and the negative signs of $(y-z)_{I_-}$ cancel, again allowing to find suitable t_j. We will consider the endpoints $\Gamma_2^j(t_j)$ as the new parameter values for $\mathbf{w}$ and the induced endpoints $\rho_2^j(t_j)$ as our new $\mathsf{act}(L-1,j;x_\alpha)$.

The next part of the path incrementally adds positive or negative coordinates of $(y-z)$ to each activation vector of the last hidden layer. For each $j \in J_+$, we let $\rho_3^j : [0,1] \to (0,d)^N$ be the path defined by

$$\rho_3^j(t) = [\mathsf{act}(L-1,j;x_\alpha)]_\alpha + t \cdot \frac{1}{w^L_{\bullet,j}} \cdot \frac{(y-z)_{I_+}}{|J_+|}$$

and for each $j \in J_-$ by

$$\rho_3^j(t) = [\mathsf{act}(L-1,j;x_\alpha)]_\alpha + t \cdot \frac{1}{w^L_{\bullet,j}} \cdot \frac{(y-z)_{I_-}}{|J_-|}$$

Since $\rho_3^j(t)$ is a path in $\mathrm{Im}(\sigma)$ for all j, this path can again be realized by an inducing change $\Gamma_3(t)$ of parameters $\mathbf{w}^{L-1}$. The parameters $\mathbf{w}^L$ are kept unchanged in this last part of the path. Simultaneously changing all $\rho_3^j(t)$ results in a change of the output of the neural network given by

$$\begin{aligned}
[f_t(x_\alpha)]_\alpha &= w^L_{\bullet,0} + \sum_{j=1}^{n_{L-1}} w^L_{\bullet,j}\rho_3^j(t) \\
&= w^L_{\bullet,0} + \left[\sum_{j\in J_+} w^L_{\bullet,j}\left(\mathsf{act}(L-1,j;x_\alpha) + t\cdot\frac{1}{w^L_{\bullet,j}}\cdot\frac{(y-z)_{I_+,\alpha}}{|J_+|}\right)\right]_\alpha \\
&\qquad + \left[\sum_{j\in J_-} w^L_{\bullet,j}\left(\mathsf{act}(L-1,j;x_\alpha) + t\cdot\frac{1}{w^L_{\bullet,j}}\cdot\frac{(y-z)_{I_-,\alpha}}{|J_-|}\right)\right]_\alpha \\
&= w^L_{\bullet,0} + \left[\sum_{j=1}^{n_{L-1}} w^L_{\bullet,j}\mathsf{act}(L-1,j;x_\alpha)\right]_\alpha \\
&\qquad + \sum_{j\in J_+} t\cdot\frac{(y-z)_{I_+}}{|J_+|} + \sum_{j\in J_-} t\cdot\frac{(y-z)_{I_-}}{|J_-|} \\
&= z + t\cdot(y-z)_{I_+} + t\cdot(y-z)_{I_-} \\
&= z + t\cdot(y-z).
\end{aligned}$$

It is easy to see that for the path $t \in [0,1] \to z + t \cdot (y - z)$ the loss

$$\mathcal{L} = ||z + t \cdot (y - z) - y||_2^2 = (1 - t)||y - z||_2^2$$

is strictly decreasing to zero. The concatenation of Γ_2 and Γ_3 gives us the desired path Γ_0.

The case that $\mathrm{Im}(\sigma) = (c, 0)$ for some $c < 0$ works analogously. In the case that $\mathrm{Im}(\sigma) = (c, d)$ with $0 \in (c, d)$, there is no need to split up into sets I_+, I_- and J_+, J_-. We have $\bar{\rho}_2^j(t_j) + \frac{1}{w_{\bullet,j}^L(t_j)} \cdot \frac{(y-z)}{N} \in (c, d)^N$ for t_j close enough to 1. Hence we can follow $\Gamma_2^j(t)$ as above until

$$\bar{\rho}_2^j(t) + \frac{1}{w_{\bullet,j}^L(t)} \cdot \frac{(y - z)}{N} \in (c, d)^N$$

for all j. From here, the paths $\rho_3^j(t) = [\mathsf{act}(L - 1, j; x_\alpha)]_\alpha + t \cdot \frac{1}{w_{\bullet,j}^L} \cdot \frac{(y-z)}{N}$ define paths in $\mathrm{Im}(\sigma)$ for each j, which can be implemented by an application of Lemma 3 and Lemma 4 and lead to the global minimum. □

E. Calculations for Lemma 1

For the calculations we may assume without loss of generality that $r = 1$. If we want to consider a different $\mathsf{n}(l, r; x)$ and its corresponding γ^r_λ, then this can be achieved by a reordering of the indices of neurons.)

We let φ denote the network function of the smaller neural network and f the neural network function of the larger network after adding one neuron according to the map γ^1_λ. To distinguish the parameters of f and φ, we write w^φ for the parameters of the network before the embedding. This gives for all i, s and all $m \geq 2$:

$$\begin{array}{llll} u_{-1,i} = u^\varphi_{1,i} & u_{1,i} = u^\varphi_{1,i} & v_{s,-1} = \lambda v^\varphi_{s,1} & v_{s,1} = (1-\lambda) v^\varphi_{s,1} \\ u_{m,i} = u^\varphi_{m,i} & v_{s,m} = v^\varphi_{s,m} & \bar{\mathbf{w}} = \bar{\mathbf{w}}^\varphi & \end{array}$$

We do the same for neuron vectors, activation vectors and the function $h_{\bullet,l+1}$. Key to the computation is the fact that all derivatives of f can be naturally written as derivatives of φ. Concretely, implied by the hierarchical embedding, all values at neurons $\mathsf{n}(l, i; x)$ and their activation values $\mathsf{act}(l, i; x)$ remain unchanged, i.e., we have for all $m \geq 1$ and all $\tilde{l} \neq l$ that

$$\begin{array}{ccc} \mathsf{act}(l, -1; x) = \mathsf{act}^\varphi(l, 1; x) & \mathsf{act}(l, m; x) = \mathsf{act}^\varphi(l, m; x) & \mathsf{act}(\tilde{l}, m; x) = \mathsf{act}^\varphi(\tilde{l}, m; x) \\ \mathsf{n}(l, -1; x) = \mathsf{n}^\varphi(l, 1; x) & \mathsf{n}(l, m; x) = \mathsf{n}^\varphi(l, m; x) & \mathsf{n}(\tilde{l}, m; x) = \mathsf{n}^\varphi(\tilde{l}, m; x) \end{array}$$

1) First order derivatives of network functions f and φ.

For the function f we have the following partial derivatives.

$$\frac{\partial f(x)}{\partial u_{p,i}} = \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, k; x)} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l, p; x)) \cdot \mathsf{act}(l-1, i; x)$$

and

$$\frac{\partial f(x)}{\partial v_{s,q}} = \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, s; x)} \cdot \mathsf{act}(l, q; x)$$

The analogous equations hold for φ.

2) Relating first order derivatives of network functions f and φ

Therefore, at $([u_{1,i}]_i, [v_{s,1}]_s, \bar{\mathbf{w}})$ and $\gamma^1_\lambda([u_{1,i}]_i, [v_{s,1}]_s, \bar{\mathbf{w}})$ respectively, we get for $k = -1, 1$ that

$$\frac{\partial f(x)}{\partial u_{-1,i}} = \lambda \frac{\partial \varphi(x)}{\partial u^\varphi_{1,i}} \text{ , and } \frac{\partial f(x)}{\partial u_{1,i}} = (1-\lambda) \frac{\partial \varphi(x)}{\partial u^\varphi_{1,i}} \text{ , and } \frac{\partial f(x)}{\partial v_{s,k}} = \frac{\partial \varphi(x)}{\partial v^\varphi_{s,1}}$$

and for $k \geq 2$ we get that

$$\frac{\partial f(x)}{\partial u_{k,i}} = \frac{\partial \varphi(x)}{\partial u^\varphi_{k,i}}, \text{ and } \frac{\partial f(x)}{\partial v_{s,k}} = \frac{\partial \varphi(x)}{\partial v^\varphi_{s,k}}.$$

3) Second order derivatives of network functions f and φ.

For the second derivatives we get (with $\delta(a, a) = 1$ and $\delta(a, b) = 0$ for $a \neq b$)

$$\begin{aligned} \frac{\partial^2 f(x)}{\partial u_{p,i} \partial u_{q,j}} &= \frac{\partial}{\partial u_{q,j}} \left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, k; x)} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l, p; x)) \cdot \mathsf{act}(l-1, i; x) \right) \\ &= \sum_m \sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, m; x) \partial \mathsf{n}(l+1, k; x)} \cdot v_{m,q} \cdot \sigma'(\mathsf{n}(l, q; x)) \cdot \mathsf{act}(l-1, j; x) \\ &\quad \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l, p; x)) \cdot \mathsf{act}(l-1, i; x) \\ &\quad + \delta(p, q) \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, k; x)} \cdot v_{k,p} \cdot \sigma''(\mathsf{n}(l, p; x)) \\ &\quad \cdot \mathsf{act}(l-1, i; x) \cdot \mathsf{act}(l-1, j; x) \end{aligned}$$

and

$$\begin{aligned} \frac{\partial^2 f(x)}{\partial v_{s,p} \partial v_{t,q}} &= \frac{\partial}{\partial v_{t,q}} \left(\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, s; x)} \cdot \mathsf{act}(l, p; x) \right) \\ &= \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1; x))}{\partial \mathsf{n}(l+1, s; x) \partial \mathsf{n}(l+1, t; x)} \cdot \mathsf{act}(l, p; x) \cdot \mathsf{act}(l, q; x) \end{aligned}$$

and

$$\begin{aligned}\frac{\partial^2 f(x)}{\partial u_{p,i}\partial v_{s,q}} &= \frac{\partial}{\partial v_{s,q}}\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x)\right) \\ &= \sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,s;x)\partial \mathsf{n}(l+1,k;x)} \cdot \mathsf{act}(l,q;x) \cdot v_{k,p} \\ &\cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x) \\ &+ \delta(q,p) \cdot \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,s;x)} \cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x)\end{aligned}$$

For a parameter w closer to the input than $[u_{p,i}]_{p,i}, [v_{s,q}]_{s,q}$, we have

$$\begin{aligned}\frac{\partial^2 f(x)}{\partial u_{p,i}\partial w} &= \frac{\partial}{\partial w}\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x)\right) \\ &= \sum_m \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)\partial \mathsf{n}(l+1,m;x)} \cdot \frac{\partial \mathsf{n}(l+1,m;x)}{\partial w} \cdot v_{k,p} \\ &\cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x) \\ &+ \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)} \cdot v_{k,p} \cdot \sigma''(\mathsf{n}(l,p;x)) \cdot \frac{\partial \mathsf{n}(l,p;x)}{\partial w} \cdot \mathsf{act}(l-1,i;x) \\ &+ \sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \frac{\partial \mathsf{act}(l-1,i;x)}{\partial w}\end{aligned}$$

and

$$\begin{aligned}\frac{\partial^2 f(x)}{\partial v_{s,q}\partial w} &= \frac{\partial}{\partial w}\left(\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,s;x)} \cdot \mathsf{act}(l,q;x)\right) \\ &= \sum_n \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,s;x)\partial \mathsf{n}(l+1,n;x)} \cdot \frac{\partial \mathsf{n}(l+1,n;x)}{\partial w} \cdot \mathsf{act}(l,q;x) \cdot \mathsf{act}(l,q;x) \\ &+ \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,s;x)} \cdot \frac{\partial \mathsf{act}(l,q;x)}{\partial w}\end{aligned}$$

For a parameter w closer to the output than $[u_{p,i}]_{p,i}, [v_{s,q}]_{s,q}$, we have

$$\begin{aligned}\frac{\partial^2 f(x)}{\partial u_{p,i}\partial w} &= \frac{\partial}{\partial w}\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x)\right) \\ &= \sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}(l+1,k;x)\partial w} \cdot v_{k,p} \cdot \sigma'(\mathsf{n}(l,p;x)) \cdot \mathsf{act}(l-1,i;x)\end{aligned}$$

4) Relating second order derivatives of network functions f and φ

To relate the second derivatives of f at $\gamma^1_\lambda([u_{1,i}]_i, [v_{s,1}]_s, \bar{\mathbf{w}})$ to the second derivatives of φ at $([u_{1,i}]_i, [v_{s,1}]_s, \bar{\mathbf{w}})$, we define

$$\begin{aligned}
A^{p,q}_{i,j}(x) &:= \sum_m \sum_k \frac{\partial^2 h^{\varphi}_{\bullet,l+1}(\mathsf{n}^\varphi(l+1;x))}{\partial \mathsf{n}^\varphi(l+1,m;x)\partial \mathsf{n}^\varphi(l+1,k;x)} \\
&\quad \cdot v^\varphi_{m,q} \cdot \sigma'(\mathsf{n}^\varphi(l,q;x)) \cdot \mathsf{act}^\varphi(l-1,j;x) \cdot v^\varphi_{k,p} \cdot \sigma'(\mathsf{n}^\varphi(l,p;x)) \cdot \mathsf{act}^\varphi(l-1,i;x) \\
B^{p}_{i,j}(x) &:= \sum_k \frac{\partial h^{\varphi}_{\bullet,l+1}(\mathsf{n}^\varphi(l+1;x))}{\partial \mathsf{n}^\varphi(l+1,k;x)} \cdot v^\varphi_{k,p} \cdot \sigma''(\mathsf{n}^\varphi(l,p;x)) \cdot \mathsf{act}^\varphi(l-1,i;x) \cdot \mathsf{act}^\varphi(l-1,j;x) \\
C^{p,s}_{i,q}(x) &:= \sum_k \frac{\partial^2 h^{\varphi}_{\bullet,l+1}(\mathsf{n}(l+1;x))}{\partial \mathsf{n}^\varphi(l+1,s;x)\partial \mathsf{n}^\varphi(l+1,k;x)} \cdot \mathsf{act}^\varphi(l,q;x) \cdot v^\varphi_{k,p} \cdot \sigma'(\mathsf{n}^\varphi(l,p;x)) \cdot \mathsf{act}^\varphi(l-1,i;x) \\
D^{p,s}_{i}(x) &:= \frac{\partial h^{\varphi}_{\bullet,l+1}(\mathsf{n}^\varphi(l+1;x))}{\partial \mathsf{n}^\varphi(l+1,s;x)} \cdot \sigma'(\mathsf{n}^\varphi(l,p;x)) \cdot \mathsf{act}^\varphi(l-1,i;x) \\
E^{s,t}_{p,q}(x) &:= \frac{\partial^2 h^{\varphi}_{\bullet,l+1}(\mathsf{n}^\varphi(l+1;x))}{\partial \mathsf{n}^\varphi(l+1,s;x)\partial \mathsf{n}^\varphi(l+1,t;x)} \cdot \mathsf{act}^\varphi(l,p;x) \cdot \mathsf{act}^\varphi(l,q;x)
\end{aligned}$$

Then for all i, j, p, q, s, t, we have

$$\begin{aligned}
\frac{\partial^2 \varphi(x)}{\partial u^\varphi_{p,i} \partial u^\varphi_{q,j}} &= A^{p,q}_{i,j}(x) + \delta(q,p) B^p_{i,j}(x) \\
\frac{\partial^2 \varphi(x)}{\partial u^\varphi_{p,i} \partial v^\varphi_{s,q}} &= C^{p,s}_{i,q}(x) + \delta(q,p) D^{p,s}_i(x) \\
\frac{\partial^2 \varphi(x)}{\partial v_{s,p} \partial v_{t,q}} &= E^{s,t}_{p,q}(x)
\end{aligned}$$

For f we get for $p, q \in \{-1, 1\}$ and all i, j, s, t

$$\begin{aligned}
\frac{\partial^2 f(x)}{\partial u_{-1,i} \partial u_{-1,j}} &= \lambda^2 A^{1,1}_{i,j}(x) + \lambda B^1_{i,j}(x) \\
\frac{\partial^2 f(x)}{\partial u_{1,i} \partial u_{1,j}} &= (1-\lambda)^2 A^{1,1}_{i,j}(x) + (1-\lambda) B^1_{i,j}(x) \\
\frac{\partial^2 f(x)}{\partial u_{-1,i} \partial u_{1,j}} &= \frac{\partial^2 f(x)}{\partial u_{1,i} \partial u_{-1,j}} = \lambda(1-\lambda) \cdot A^{1,1}_{i,j}(x) \\
\frac{\partial^2 f(x)}{\partial u_{-1,i} \partial v_{s,-1}} &= \lambda C^{1,s}_{i,1}(x) + D^{1,s}_i(x) \\
\frac{\partial^2 f(x)}{\partial u_{1,i} \partial v_{s,1}} &= (1-\lambda) C^{1,s}_{i,1}(x) + D^{1,s}_i(x) \\
\frac{\partial^2 f(x)}{\partial u_{-1,i} \partial v_{s,1}} &= \lambda \cdot C^{1,s}_{i,1}(x) = \lambda \cdot \frac{\partial^2 \varphi(x)}{\partial u^\varphi_{1,i} \partial v^\varphi_{s,1}} \\
\frac{\partial^2 f(x)}{\partial u_{1,i} \partial v_{s,-1}} &= (1-\lambda) \cdot C^{1,s}_{i,1}(x) = (1-\lambda) \cdot \frac{\partial^2 \varphi(x)}{\partial u^\varphi_{1,i} \partial v^\varphi_{s,1}} \\
\frac{\partial^2 f(x)}{\partial v_{s,p} \partial v_{t,q}} &= E^{s,t}_{1,1}(x) = \frac{\partial^2 \varphi(x)}{\partial v^\varphi_{s,1} \partial v^\varphi_{t,1}}
\end{aligned}$$

and for $q \geq 2$ and $p \in \{-1, 1\}$ and all i, j, s, t

$$\frac{\partial^2 f(x)}{\partial u_{-1,i} \partial u_{q,j}} = \lambda A^{1,q}_{i,j}(x) = \lambda \cdot \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{1,i} \partial u^{\varphi}_{q,j}}$$
$$\frac{\partial^2 f(x)}{\partial u_{1,i} \partial u_{q,j}} = (1-\lambda) A^{1,q}_{i,j}(x) = (1-\lambda) \cdot \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{1,i} \partial u^{\varphi}_{q,j}}$$
$$\frac{\partial^2 f(x)}{\partial u_{-1,i} \partial v_{s,q}} = \lambda C^{1,s}_{i,q}(x) = \lambda \cdot \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{1,i} \partial v^{\varphi}_{s,q}}$$
$$\frac{\partial^2 f(x)}{\partial u_{1,i} \partial v_{s,q}} = (1-\lambda) C^{1,s}_{i,q}(x) = (1-\lambda) \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{1,i} \partial v^{\varphi}_{s,q}}$$
$$\frac{\partial^2 f(x)}{\partial u_{q,i} \partial v_{s,p}} = C^{q,s}_{i,1}(x) = \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{q,i} \partial v^{\varphi}_{s,1}}$$
$$\frac{\partial^2 f(x)}{\partial v_{s,p} \partial v_{t,q}} = E^{s,t}_{1,q}(x) = \frac{\partial^2 \varphi(x)}{\partial v^{\varphi}_{s,1} \partial v^{\varphi}_{t,q}}$$

and for $p, q \geq 2$ and all i, j, s, t

$$\frac{\partial^2 f(x)}{\partial u_{p,i} \partial u_{q,j}} = A^{p,q}_{i,j}(x) + \delta(q,p) B^{p}_{i,j}(x) = \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{p,i} \partial u^{\varphi}_{q,j}}$$
$$\frac{\partial^2 f(x)}{\partial u_{p,i} \partial v_{s,q}} = C^{p,s}_{i,q}(x) + \delta(q,p) D^{p,s}_{i}(x) = \frac{\partial^2 \varphi(x)}{\partial u^{\varphi}_{p,i} \partial v^{\varphi}_{s,q}}$$
$$\frac{\partial^2 f(x)}{\partial v_{s,p} \partial v_{t,q}} = E^{s,t}_{p,q}(x) = \frac{\partial^2 \varphi(x)}{\partial v^{\varphi}_{s,p} \partial v^{\varphi}_{t,q}}$$

5) Derivatives of $\mathcal{L}$ and ℓ

Let

$$A^{p,q}_{i,j} := \sum_{\alpha} (\varphi(x_\alpha) - y_\alpha) \cdot A^{p,q}_{i,j}(x_\alpha)$$
$$B^{p}_{i,j} := \sum_{\alpha} (\varphi(x_\alpha) - y_\alpha) \cdot B^{p}_{i,j}(x_\alpha)$$
$$C^{p,s}_{i,q} := \sum_{\alpha} (\varphi(x_\alpha) - y_\alpha) \cdot C^{p,s}_{i,q}(x_\alpha)$$
$$D^{p,s}_{i} := \sum_{\alpha} (\varphi(x_\alpha) - y_\alpha) \cdot D^{p,s}_{i}(x_\alpha)$$
$$E^{s,t}_{p,q} := \sum_{\alpha} (\varphi(x_\alpha) - y_\alpha) \cdot E^{s,t}_{p,q}(x_\alpha)$$

and

$$\mathcal{A}^{p,q}_{i,j} := \sum_{\alpha} \left(\frac{\partial \varphi(x_\alpha)}{\partial u^{\varphi}_{p,i}} \right) \cdot \left(\frac{\partial \varphi(x_\alpha)}{\partial u^{\varphi}_{q,j}} \right)$$
$$\mathcal{C}^{p,s}_{i,q} := \sum_{\alpha} \left(\frac{\partial \varphi(x_\alpha)}{\partial u^{\varphi}_{p,i}} \right) \cdot \left(\frac{\partial \varphi(x_\alpha)}{\partial v^{\varphi}_{s,q}} \right)$$
$$\mathcal{E}^{s,t}_{p,q} := \sum_{\alpha} \left(\frac{\partial \varphi(x_\alpha)}{\partial v^{\varphi}_{s,p}} \right) \cdot \left(\frac{\partial \varphi(x_\alpha)}{\partial v^{\varphi}_{t,q}} \right)$$

Now we have everything together to compute the derivatives of the loss. For the first derivative of the loss, we have for any variables w, r that

$$\frac{\partial \mathcal{L}}{\partial w} = \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \frac{\partial f(x_\alpha)}{\partial w}$$

and

$$\frac{\partial \ell}{\partial w^\varphi} = \sum_\alpha (\varphi(x_\alpha) - y_\alpha) \cdot \frac{\partial \varphi(x_\alpha)}{\partial w^\varphi}.$$

From this it follows immediately that if $\frac{\partial \ell}{\partial w^\varphi}(\mathbf{w}^\varphi) = 0$, then $\frac{\partial \mathcal{L}}{\partial w}(\gamma_\lambda^1(\mathbf{w}^\varphi)) = 0$ for all λ (cf. [9], [15]).

For the second derivative we get

$$\frac{\partial^2 \mathcal{L}}{\partial w \partial r} = \sum_\alpha (f(x_\alpha) - y_\alpha) \cdot \frac{\partial^2 f(x_\alpha)}{\partial w \partial r} + \sum_\alpha \left(\frac{\partial f(x_\alpha)}{\partial w}\right) \cdot \left(\frac{\partial f(x_\alpha)}{\partial r}\right)$$

This leads to the following equations for ℓ

$$\begin{aligned}
\frac{\partial^2 \ell}{\partial u^\varphi_{p,i} \partial u^\varphi_{q,j}} &= A^{1,1}_{p,q} + \delta(p,q) B^p_{i,j} + \mathcal{A}^{p,q}_{i,j} \\
\frac{\partial^2 \ell}{\partial u^\varphi_{p,i} \partial v^\varphi_{s,q}} &= C^{p,s}_{i,q} + \delta(p,q) D^{p,s}_i + \mathcal{C}^{p,s}_{i,q} \\
\frac{\partial^2 \ell}{\partial v^\varphi_{s,p} \partial v_{t,q}} &= E^{s,t}_{p,q} + \mathcal{E}^{s,t}_{p,q}
\end{aligned}$$

For $\mathcal{L}$ we get for $p, q \in \{-1, 1\}$ and all i, j, s, t at $\gamma_\lambda^1([u_{1,i}]_i, [v_{s,1}]_s, \bar{\mathbf{w}})$

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial u_{-1,i} \partial u_{-1,j}} &= \lambda^2 A^{1,1}_{i,j} + \lambda B^1_{i,j} + \lambda^2 \mathcal{A}^{1,1}_{i,j} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{1,i} \partial u_{1,j}} &= (1-\lambda)^2 A^{1,1}_{i,j} + (1-\lambda) B^1_{i,j} + (1-\lambda)^2 \mathcal{A}^{1,1}_{i,j} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{-1,i} \partial u_{1,j}} &= \lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda) \mathcal{A}^{1,1}_{i,j} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{-1,i} \partial v_{s,-1}} &= \lambda C^{1,s}_{i,1} + D^{1,s}_i + \lambda \mathcal{C}^{1,s}_{i,1} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{1,i} \partial v_{s,1}} &= (1-\lambda) C^{1,s}_{i,1} + D^{1,s}_i + (1-\lambda) \mathcal{C}^{1,s}_{i,1} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{-1,i} \partial v_{s,1}} &= \lambda C^{1,s}_{i,1} + \lambda \mathcal{C}^{1,s}_{i,1} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{1,i} \partial v_{s,-1}} &= (1-\lambda) C^{1,s}_{i,1} + (1-\lambda) \mathcal{C}^{1,s}_{i,1} \\
\frac{\partial^2 \mathcal{L}}{\partial v_{s,p} \partial v_{t,q}} &= E^{s,t}_{1,1} + \mathcal{E}^{s,t}_{1,1}
\end{aligned}$$

and for $q \geq 2$ and $p \in \{-1, 1\}$ and all i, j, s, t

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial u_{-1,i} \partial u_{q,j}} &= \lambda A^{1,q}_{i,j} + \lambda \mathcal{A}^{1,1}_{i,j} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{1,i} \partial u_{q,j}} &= (1-\lambda) A^{1,q}_{i,j} + (1-\lambda) \mathcal{A}^{1,q}_{i,j} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{-1,i} \partial v_{s,q}} &= \lambda C^{1,s}_{i,q} + \lambda \mathcal{C}^{1,s}_{i,q} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{1,i} \partial v_{s,q}} &= (1-\lambda) C^{1,s}_{i,q} + (1-\lambda) \mathcal{C}^{1,s}_{i,q} \\
\frac{\partial^2 \mathcal{L}}{\partial u_{q,i} \partial v_{s,p}} &= C^{q,s}_{i,p} + \mathcal{C}^{q,s}_{i,p} \\
\frac{\partial^2 \mathcal{L}}{\partial v_{s,p} \partial v_{t,q}} &= E^{s,t}_{1,q} + \mathcal{E}^{s,t}_{1,q}
\end{aligned}$$

and for $p, q \geq 2$ and all i, j, s, t

$$\frac{\partial^2 \mathcal{L}}{\partial u_{p,i} \partial u_{q,j}} = A^{p,q}_{i,j} + \delta(q,p) B^p_{i,j}(x) + \mathcal{A}^{p,q}_{i,j} = \frac{\partial^2 \ell}{\partial u^{\varphi}_{p,i} \partial u^{\varphi}_{q,j}}$$
$$\frac{\partial^2 \mathcal{L}}{\partial u_{p,i} \partial v_{s,q}} = C^{p,s}_{i,q} + \delta(q,p) D^{p,s}_i + \mathcal{C}^{p,s}_{i,q} = \frac{\partial^2 \ell}{\partial u^{\varphi}_{p,i} \partial v^{\varphi}_{s,q}}$$
$$\frac{\partial^2 \mathcal{L}}{\partial v_{s,p} \partial v_{t,q}} = E^{s,t}_{p,q} + \mathcal{E}^{s,t}_{p,q} = \frac{\partial^2 \ell}{\partial v^{\varphi}_{s,p} \partial v^{\varphi}_{t,q}}$$

6) Change of basis

Choose any real numbers $\alpha \neq -\beta$ such that $\lambda = \frac{\beta}{\alpha+\beta}$ (equivalently $\alpha\lambda - \beta(1-\lambda) = 0$) and set

$$\mu_{-1,i} = u_{-1,i} + u_{1,i} \qquad \mu_{1,i} = \alpha \cdot u_{-1,i} - \beta \cdot u_{1,i}$$
$$\nu_{s,-1} = v_{s,-1} + v_{s,1} \qquad \nu_{s,1} = v_{s,-1} - v_{s,1}.$$

Then at $\gamma^1_\lambda([u_{1,i}]_i, [v_{s,1}]_s, \bar{\mathbf{w}})$,

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial \mu_{-1,i} \partial \mu_{-1,j}} &= \left(\frac{\partial}{\partial u_{-1,i}} + \frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial \mathcal{L}(x)}{\partial u_{-1,j}} + \frac{\partial \mathcal{L}(x)}{\partial u_{1,j}}\right) \\
&= \frac{\partial^2 L(x)}{\partial u_{-1,i} \partial u_{-1,j}} + \frac{\partial^2 \mathcal{L}(x)}{\partial u_{-1,i} \partial u_{1,j}} + \frac{\partial^2 \mathcal{L}(x)}{\partial u_{1,i} \partial u_{-1,j}} + \frac{\partial^2 \mathcal{L}(x)}{\partial u_{1,i} \partial u_{1,j}} \\
&= \left(\lambda^2 A^{1,1}_{i,j} + \lambda B^1_{i.j} + \lambda^2 \mathcal{A}^{1,1}_{i,j}\right) + \left(\lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda)\mathcal{A}^{1,1}_{i,j}\right) \\
&+ \left(\lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda)\mathcal{A}^{1,1}_{i,j}\right) + \left((1-\lambda)^2 A^{1,1}_{i,j} + (1-\lambda) B^1_{i.j} + (1-\lambda)^2 \mathcal{A}^{1,1}_{i,j}\right) \\
&= A^{1,1}_{i,j} + B^1_{i.j} + \mathcal{A}^{1,1}_{i,j}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial \mu_{1,i} \partial \mu_{1,j}} &= \left(\alpha\frac{\partial}{\partial u_{-1,i}} - \beta\frac{\partial}{\partial u_{1,i}}\right)\left(\alpha\frac{\partial \mathcal{L}(x)}{\partial u_{-1,j}} - \beta\frac{\partial \mathcal{L}(x)}{\partial u_{1,j}}\right) \\
&= \alpha^2 \frac{\partial^2 \mathcal{L}(x)}{\partial u_{-1,i} \partial u_{-1,j}} - \alpha\beta \frac{\partial^2 \mathcal{L}(x)}{\partial u_{-1,i} \partial u_{1,j}} - \alpha\beta \frac{\partial^2 \mathcal{L}(x)}{\partial u_{1,i} \partial u_{-1,j}} + \beta^2 \frac{\partial^2 \mathcal{L}(x)}{\partial u_{1,i} \partial u_{1,j}} \\
&= \alpha^2 \left(\lambda^2 A^{1,1}_{i,j} + \lambda B^1_{i.j} + \lambda^2 \mathcal{A}^{1,1}_{i,j}\right) - \alpha\beta \left(\lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda)\mathcal{A}^{1,1}_{i,j}\right) \\
&- \alpha\beta \left(\lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda)\mathcal{A}^{1,1}_{i,j}\right) \\
&+ \beta^2 \left((1-\lambda)^2 A^{1,1}_{i,j} + (1-\lambda) B^1_{i.j} + (1-\lambda)^2 \mathcal{A}^{1,1}_{i,j}\right) \\
&= \alpha\beta B^1_{i.j}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial \mu_{-1,i} \partial \mu_{1,j}} &= \left(\frac{\partial}{\partial u_{-1,i}} + \frac{\partial}{\partial u_{1,i}}\right)\left(\alpha\frac{\partial \mathcal{L}(x)}{\partial u_{-1,j}} - \beta\frac{\partial \mathcal{L}(x)}{\partial u_{1,j}}\right) \\
&= \alpha \frac{\partial^2 \mathcal{L}(x)}{\partial u_{-1,i} \partial u_{-1,j}} - \beta \frac{\partial^2 \mathcal{L}(x)}{\partial u_{-1,i} \partial u_{1,j}} + \alpha \frac{\partial^2 \mathcal{L}(x)}{\partial u_{1,i} \partial u_{-1,j}} - \beta \frac{\partial^2 \mathcal{L}(x)}{\partial u_{1,i} \partial u_{1,j}} \\
&= \alpha \left(\lambda^2 A^{1,1}_{i,j} + \lambda B^2_{i.j} + \lambda^2 \mathcal{A}^{1,1}_{i,j}\right) - \beta \left(\lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda)\mathcal{A}^{1,1}_{i,j}\right) \\
&+ \alpha \left(\lambda(1-\lambda) A^{1,1}_{i,j} + \lambda(1-\lambda)\mathcal{A}^{1,1}_{i,j}\right) - \beta \left((1-\lambda)^2 A^{1,1}_{i,j} + (1-\lambda) B^2_{i.j} + (1-\lambda)^2 \mathcal{A}^{1,1}_{i,j}\right) \\
&= 0
\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,-1}\partial\nu_{t,-1}} &= \left(\frac{\partial}{\partial v_{s,-1}}+\frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{t,-1}}+\frac{\partial\mathcal{L}(x)}{\partial v_{t,1}}\right)\\ &= \frac{\partial^2\mathcal{L}(x)}{\partial v_{s,-1}\partial v_{t,-1}}+\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,-1}\partial v_{t,1}}+\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,1}\partial v_{t,-1}}+\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,1}\partial v_{t,1}}\\ &= \left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)+\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)+\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)+\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)\\ &= 4E^{s,t}_{1,1}+4\mathcal{E}^{s,t}_{1,1}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,1}\partial\nu_{t,1}} &= \left(\frac{\partial}{\partial v_{s,-1}}-\frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{t,-1}}-\frac{\partial\mathcal{L}(x)}{\partial v_{t,1}}\right)\\ &= \frac{\partial^2\mathcal{L}(x)}{\partial v_{s,-1}\partial v_{t,-1}}-\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,-1}\partial v_{t,1}}-\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,1}\partial v_{t,-1}}+\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,1}\partial v_{t,1}}\\ &= \left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)-\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)-\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)+\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)\\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,-1}\partial\nu_{t,1}} &= \left(\frac{\partial}{\partial v_{s,-1}}+\frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{t,-1}}-\frac{\partial\mathcal{L}(x)}{\partial v_{t,1}}\right)\\ &= \frac{\partial^2\mathcal{L}(x)}{\partial v_{s,-1}\partial v_{t,-1}}-\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,-1}\partial v_{t,1}}+\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,1}\partial v_{t,-1}}-\frac{\partial^2\mathcal{L}(x)}{\partial v_{s,1}\partial v_{t,1}}\\ &= \left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)-\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)+\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)-\left(E^{s,t}_{1,1}+\mathcal{E}^{s,t}_{1,1}\right)\\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{-1,i}\partial\nu_{s,-1}} &= \left(\frac{\partial}{\partial u_{-1,i}}+\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{s,-1}}+\frac{\partial\mathcal{L}(x)}{\partial v_{s,1}}\right)\\ &= \frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,-1}}+\frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,1}}+\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,-1}}+\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,1}}\\ &= \left(\lambda C^{1,s}_{i,1}+D^{1,s}_{i}+\lambda\mathcal{C}^{1,s}_{i,1}\right)+\left(\lambda C^{1,s}_{i,1}+\lambda\mathcal{C}^{1,s}_{i,1}\right)\\ &+\left((1-\lambda)C^{1,s}_{i,1}+(1-\lambda)\mathcal{C}^{1,s}_{i,1}\right)+\left((1-\lambda)C^{1,s}_{i,1}+D^{1,s}_{i}+(1-\lambda)\mathcal{C}^{1,s}_{i,1}\right)\\ &= 2C^{1,s}_{i,1}+2D^{1,s}_{i}+2\mathcal{C}^{1,s}_{i,1}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{-1,i}\partial\nu_{s,1}} &= \left(\frac{\partial}{\partial u_{-1,i}}+\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{s,-1}}-\frac{\partial\mathcal{L}(x)}{\partial v_{s,1}}\right)\\ &= \frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,-1}}-\frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,1}}+\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,-1}}-\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,1}}\\ &= \left(\lambda C^{1,s}_{i,1}+D^{1,s}_{i}+\lambda\mathcal{C}^{1,s}_{i,1}\right)-\left(\lambda C^{1,s}_{i,1}+\lambda\mathcal{C}^{1,s}_{i,1}\right)\\ &+\left((1-\lambda)C^{1,s}_{i,1}+(1-\lambda)\mathcal{C}^{1,s}_{i,1}\right)-\left((1-\lambda)C^{1,s}_{i,1}+D^{1,s}_{i}+(1-\lambda)\mathcal{C}^{1,s}_{i,1}\right)\\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{1,i}\partial\nu_{s,-1}} &= \left(\alpha\frac{\partial}{\partial u_{-1,i}} - \beta\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{s,-1}} + \frac{\partial\mathcal{L}(x)}{\partial v_{s,1}}\right)\\ &= \alpha\frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,-1}} + \alpha\frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,1}} - \beta\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,-1}} - \beta\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,1}}\\ &= \alpha\left(\lambda C_{i,1}^{1,s} + D_i^{1,s} + \lambda\mathcal{C}_{i,1}^{1,s}\right) + \alpha\left(\lambda C_{i,1}^{1,s} + \lambda\mathcal{C}_{i,1}^{1,s}\right)\\ &- \beta\left((1-\lambda)C_{i,1}^{1,s} + (1-\lambda)\mathcal{C}_{i,1}^{1,s}\right) - \beta\left((1-\lambda)C_{i,1}^{1,s} + D_i^{1,s} + (1-\lambda)\mathcal{C}_{i,1}^{1,s}\right)\\ &= (\alpha-\beta)D_i^{1,s}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{1,i}\partial\nu_{s,1}} &= \left(\alpha\frac{\partial}{\partial u_{-1,i}} - \beta\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{s,-1}} - \frac{\partial\mathcal{L}(x)}{\partial v_{s,1}}\right)\\ &= \alpha\frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,-1}} - \alpha\frac{\partial^2\mathcal{L}(x)}{\partial u_{-1,i}\partial v_{s,1}} - \beta\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,-1}} + \beta\frac{\partial^2\mathcal{L}(x)}{\partial u_{1,i}\partial v_{s,1}}\\ &= \alpha\left(\lambda C_{i,1}^{1,s} + D_i^{1,s} + \lambda\mathcal{C}_{i,1}^{1,s}\right) - \alpha\left(\lambda C_{i,1}^{1,s} + \lambda\mathcal{C}_{i,1}^{1,s}\right)\\ -\beta&\left((1-\lambda)C_{i,1}^{1,s} + (1-\lambda)\mathcal{C}_{i,1}^{1,s}\right) + \beta\left((1-\lambda)C_{i,1}^{1,s} + D_i^{1,s} + (1-\lambda)\mathcal{C}_{i,1}^{1,s}\right)\\ &= (\alpha+\beta)D_i^{1,s}\end{aligned}$$

For $q \geq 2$ and $p \in \{-1, 1\}$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{-1,i}\partial u_{q,j}} &= \left(\frac{\partial}{\partial u_{-1,i}} + \frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial u_{q,j}}\right)\\ &= \lambda A_{i,j}^{1,q} + \lambda\mathcal{A}_{i,j}^{1,1} + (1-\lambda)A_{i,j}^{1,q} + (1-\lambda)\mathcal{A}_{i,j}^{1,q}\\ &= A_{i,j}^{1,q} + \mathcal{A}_{i,j}^{1,q}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{1,i}\partial u_{q,j}} &= \left(\alpha\frac{\partial}{\partial u_{-1,i}} - \beta\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial u_{q,j}}\right)\\ &= \alpha(\lambda A_{i,j}^{1,q} + \lambda\mathcal{A}_{i,j}^{1,1}) + \beta((1-\lambda)A_{i,j}^{1,q} + (1-\lambda)\mathcal{A}_{i,j}^{1,q})\\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{-1,i}\partial v_{s,q}} &= \left(\frac{\partial}{\partial u_{-1,i}} + \frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{s,q}}\right)\\ &= \lambda C_{i,q}^{1,s} + \lambda\mathcal{C}_{i,q}^{1,s} + (1-\lambda)C_{i,q}^{1,s} + (1-\lambda)\mathcal{C}_{i,q}^{1,s}\\ &= C_{i,q}^{1,s} + \mathcal{C}_{i,q}^{1,s}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{1,i}\partial v_{s,q}} &= \left(\alpha\frac{\partial}{\partial u_{-1,i}} - \beta\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{s,q}}\right)\\ &= \alpha(\lambda C_{i,q}^{1,s} + \lambda\mathcal{C}_{i,q}^{1,s}) - \beta((1-\lambda)C_{i,q}^{1,s} + (1-\lambda)\mathcal{C}_{i,q}^{1,s})\\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,-1}\partial u_{q,i}} &= \left(\frac{\partial}{\partial v_{s,-1}} + \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial u_{q,i}}\right)\\ &= C_{i,p}^{q,s} + \mathcal{C}_{i,p}^{q,s} + C_{i,p}^{q,s} + \mathcal{C}_{i,p}^{q,s}\\ &= 2C_{i,p}^{q,s} + 2\mathcal{C}_{i,p}^{q,s}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,1}\partial u_{q,i}} &= \left(\frac{\partial}{\partial v_{s,-1}} - \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial u_{q,i}}\right)\\ &= C_{i,p}^{q,s} + \mathcal{C}_{i,p}^{q,s} - C_{i,p}^{q,s} - \mathcal{C}_{i,p}^{q,s}\\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,-1}\partial v_{t,q}} &= \left(\frac{\partial}{\partial v_{s,-1}} + \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{t,q}}\right)\\ &= E_{1,q}^{s,t} + \mathcal{E}_{1,q}^{s,t} + E_{1,q}^{s,t} + \mathcal{E}_{1,q}^{s,t}\\ &= 2E_{1,q}^{s,t} + 2\mathcal{E}_{1,q}^{s,t}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\nu_{s,1}\partial v_{t,q}} &= \left(\frac{\partial}{\partial v_{s,-1}} - \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial\mathcal{L}(x)}{\partial v_{t,q}}\right)\\ &= E_{1,q}^{s,t} + \mathcal{E}_{1,q}^{s,t} - E_{1,q}^{s,t} - \mathcal{E}_{1,q}^{s,t}\\ &= 0\end{aligned}$$

We also need to consider the second derivative with respect to the other variables of $\bar{\mathbf{w}}$. If w is closer to the output than $[u_{p,i}]_{p,i}, [v_{s,q}]_{s,q}$ belonging to layer γ where $\gamma > l+1$, then we get

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial w\partial\mu_{-1,i}} &= \frac{\partial}{\partial w}\left(\frac{\partial\mathcal{L}}{\partial u_{-1,i}} + \frac{\partial\mathcal{L}}{\partial u_{1,i}}\right)\\ &= \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot v_{k,-1}\cdot\sigma'(\mathsf{n}(l,-1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\right)\\ &+ \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot v_{k,1}\cdot\sigma'(\mathsf{n}(l,1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\right)\\ &= \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot v_{k,-1}\cdot\sigma'(\mathsf{n}(l,-1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\\ &+ \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot v_{k,1}\cdot\sigma'(\mathsf{n}(l,1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\\ &= \frac{\partial^2\ell}{\partial w^\varphi\partial u_{1,i}^\varphi}\end{aligned}$$

and

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial w\partial\mu_{-1,i}} &= \frac{\partial}{\partial w}\left(\alpha\frac{\partial\mathcal{L}}{\partial u_{-1,i}} - \beta\frac{\partial\mathcal{L}}{\partial u_{1,i}}\right)\\ &= \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\cdot\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot\alpha v_{k,-1}\cdot\sigma'(\mathsf{n}(l,-1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\right)\\ &- \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\cdot\left(\sum_k \frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot\beta v_{k,1}\cdot\sigma'(\mathsf{n}(l,1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\right)\\ &+ \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot\alpha v_{k,-1}\cdot\sigma'(\mathsf{n}(l,-1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\\ &- \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\sum_k \frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial\mathsf{n}(l+1,k;x_\alpha)}\cdot\beta v_{k,1}\cdot\sigma'(\mathsf{n}(l,1;x_\alpha))\cdot\mathsf{act}(l-1,i;x_\alpha)\\ &= 0\end{aligned}$$

and

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial w\partial\nu_{s,-1}} &= \frac{\partial}{\partial w}\left(\frac{\partial\mathcal{L}}{\partial v_{s,-1}}+\frac{\partial\mathcal{L}}{\partial v_{s,1}}\right)\\ &= \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\cdot\left(\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,-1;x_\alpha)\right)\\ &+ \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\cdot\left(\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,1;x_\alpha)\right)\\ &+ \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,-1;x_\alpha)\\ &+ \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,1;x_\alpha)\\ &= 2\cdot\frac{\partial^2\ell}{\partial w^\varphi\partial v^\varphi_{s,1}}\end{aligned}$$

and

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial w\partial\nu_{s,1}} &= \frac{\partial}{\partial w}\left(\frac{\partial\mathcal{L}}{\partial v_{s,-1}}-\frac{\partial\mathcal{L}}{\partial v_{s,1}}\right)\\ &= \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\cdot\left(\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,-1;x_\alpha)\right)\\ &- \sum_\alpha \frac{\partial f(x_\alpha)}{\partial w}\cdot\left(\frac{\partial h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,1;x_\alpha)\right)\\ &+ \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,-1;x_\alpha)\\ &- \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\frac{\partial^2 h_{\bullet,l+1}(\mathsf{n}(l+1;x_\alpha))}{\partial w\partial \mathsf{n}(l+1,s;x_\alpha)}\cdot \mathsf{act}(l,1;x_\alpha)\\ &= 0\end{aligned}$$

If w is closer to the input than $[u_{p,i}]_{p,i}, [v_{s,q}]_{s,q}$ connecting neuron j of layer $\gamma-1$ with neuron r of layer γ where $\gamma < l$, then we get

$$\begin{aligned}\frac{\partial^2\mathcal{L}}{\partial\mu_{-1,i}\partial w} &= \left(\frac{\partial}{\partial u_{-1,i}}+\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial\mathcal{L}}{\partial w}\right)\\ &= \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\left(\frac{\partial}{\partial u_{-1,i}}+\frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma;x_\alpha))}{\partial \mathsf{n}(\gamma,r;x_\alpha)}\cdot \mathsf{act}(\gamma-1,j;x_\alpha)\right)\\ &+ \sum_\alpha\left(\frac{\partial f(x_\alpha)}{\partial u_{-1,i}}+\frac{\partial f(x_\alpha)}{\partial u_{1,i}}\right)\cdot\left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma;x_\alpha))}{\partial \mathsf{n}(\gamma,r;x_\alpha)}\cdot \mathsf{act}(\gamma-1,j;x_\alpha)\right)\\ &= \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\sum_k\frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma;x_\alpha))}{\partial \mathsf{n}(l+1,k;x_\alpha)\partial \mathsf{n}(\gamma,r;x_\alpha)}\cdot v_{k,-1}\cdot\sigma'(\mathsf{n}(l,-1;x_\alpha))\\ &\cdot \mathsf{act}(l-1,i;x_\alpha)\cdot\mathsf{act}(\gamma-1,j;x_\alpha)\\ &+ \sum_\alpha (f(x_\alpha)-y_\alpha)\cdot\sum_k\frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma;x_\alpha))}{\partial \mathsf{n}(l+1,k;x_\alpha)\partial \mathsf{n}(\gamma,r;x_\alpha)}\cdot v_{k,1}\cdot\sigma'(\mathsf{n}(l,1;x_\alpha))\\ &\cdot \mathsf{act}(l-1,i;x_\alpha)\cdot\mathsf{act}(\gamma-1,j;x_\alpha)\\ &+ \sum_\alpha\left(\frac{\partial\varphi(x_\alpha)}{\partial u^\varphi_{1,i}}\right)\cdot\left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma;x_\alpha))}{\partial \mathsf{n}(\gamma,r;x_\alpha)}\cdot \mathsf{act}(\gamma-1,j;x_\alpha)\right)\\ &= \frac{\partial^2\ell}{\partial w^\varphi\partial u^\varphi_{1,i}}\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial \mu_{-1,i} \partial w} &= \left(\alpha \frac{\partial}{\partial u_{-1,i}} - \beta \frac{\partial}{\partial u_{1,i}}\right)\left(\frac{\partial \mathcal{L}}{\partial w}\right) \\
&= \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \left(\alpha \frac{\partial}{\partial u_{-1,i}} - \beta \frac{\partial}{\partial u_{-1,i}}\right)\left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&+ \sum_{\alpha} \left(\alpha \frac{\partial f(x_\alpha)}{\partial u_{-1,i}} - \beta \frac{\partial f(x_\alpha)}{\partial u_{-1,i}}\right) \cdot \left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&= \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \sum_{k} \frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(l+1, k; x_\alpha) \partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot v_{k,-1} \cdot \sigma'(\mathsf{n}(l, -1; x_\alpha)) \\
&\cdot \mathsf{act}(l - 1, i; x_\alpha) \cdot \mathsf{act}(\gamma - 1, j; x_\alpha) \\
&- \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \sum_{k} \frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(l+1, k; x_\alpha) \partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot v_{k,1} \cdot \sigma'(\mathsf{n}(l, 1; x_\alpha)) \\
&\cdot \mathsf{act}(l - 1, i; x_\alpha) \cdot \mathsf{act}(\gamma - 1, j; x_\alpha) \\
&+ \sum_{\alpha} \left((\alpha\lambda - \beta(1-\lambda)) \frac{\partial \varphi(x_\alpha)}{\partial u^{\varphi}_{1,i}}\right) \cdot \left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&= 0
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial \nu_{s,-1} \partial w} &= \left(\frac{\partial}{\partial v_{s,-1}} + \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial \mathcal{L}}{\partial w}\right) \\
&= \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \left(\frac{\partial}{\partial v_{s,-1}} + \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&+ \sum_{\alpha} \left(\frac{\partial f(x_\alpha)}{\partial v_{s,-1}} + \frac{\partial f(x_\alpha)}{\partial v_{s,1}}\right) \cdot \left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&= \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(l+1, s; x_\alpha) \partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(l, -1; x_\alpha)) \cdot \mathsf{act}(\gamma - 1, j; x_\alpha) \\
&+ \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(l+1, s; x_\alpha) \partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(l, 1; x_\alpha)) \cdot \mathsf{act}(\gamma - 1, j; x_\alpha) \\
&+ \sum_{\alpha} \left(\frac{\partial \varphi(x_\alpha)}{\partial v^{\varphi}_{s,1}} + \frac{\partial \varphi(x_\alpha)}{\partial v^{\varphi}_{s,1}}\right) \cdot \left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&= 2 \cdot \frac{\partial^2 \ell}{\partial w^{\varphi} \partial v^{\varphi}_{s,1}}
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}}{\partial \nu_{s,1} \partial w} &= \left(\frac{\partial}{\partial v_{s,-1}} - \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial \mathcal{L}}{\partial w}\right) \\
&= \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \left(\frac{\partial}{\partial v_{s,-1}} - \frac{\partial}{\partial v_{s,1}}\right)\left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&+ \sum_{\alpha} \left(\frac{\partial f(x_\alpha)}{\partial v_{s,-1}} - \frac{\partial f(x_\alpha)}{\partial v_{s,1}}\right) \cdot \left(\frac{\partial h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(\gamma - 1, j; x_\alpha)\right) \\
&= \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(l+1, s; x_\alpha) \partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(l, -1; x_\alpha)) \cdot \mathsf{act}(\gamma - 1, j; x_\alpha) \\
&- \sum_{\alpha} (f(x_\alpha) - y_\alpha) \cdot \frac{\partial^2 h_{\bullet,\gamma}(\mathsf{n}(\gamma; x_\alpha))}{\partial \mathsf{n}(l+1, s; x_\alpha) \partial \mathsf{n}(\gamma, r; x_\alpha)} \cdot \mathsf{act}(l, 1; x_\alpha)) \cdot \mathsf{act}(\gamma - 1, j; x_\alpha) \\
&= 0
\end{aligned}$$

Finally, if w and w' are parameters different from $[u_{p,i}]_{p,i}, [v_{s,q}]_{s,q}, \mu$ and ν, then

$$\frac{\partial^2 \mathcal{L}}{\partial w \partial w'} = \frac{\partial^2 \ell}{\partial w^\varphi \partial w'^\varphi}$$

7) The Hessian

Putting things together, the matrix for the second derivative of $\mathcal{L}$ with respect to $\mu_-1, \nu_-1, \bar{\mathbf{w}}, \mu_1, \nu_1$, where $\bar{\mathbf{w}}$ stands for the collection of all other parameters, at $\gamma^1_\lambda([u^*_{1,i}]_i, [v^*_{s,1}]_s, \bar{\mathbf{w}}*)$ is given by:

$$\begin{pmatrix}
[\frac{\partial^2 \ell}{\partial u_{1,i} \partial u_{1,j}}]_{i,j} & 2[\frac{\partial^2 \ell}{\partial u_{1,i} \partial v_{s,1}}]_{i,s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{1,i}}]_{i,\bar{\mathbf{w}}} & 0 & 0 \\
2[\frac{\partial^2 \ell}{\partial u_{1,i} \partial v_{s,1}}]_{s,i} & 4[\frac{\partial^2 \ell}{\partial v_{s,1} \partial v_{t,1}}]_{s,t} & 2[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,1}}]_{s,\bar{\mathbf{w}}} & (\alpha - \beta)[D^{1,s}_i]_{s,i} & 0 \\
[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial u_{1,i}}]_{\bar{\mathbf{w}},i} & 2[\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial v_{s,1}}]_{\bar{\mathbf{w}},s} & [\frac{\partial^2 \ell}{\partial \bar{\mathbf{w}}\ \partial \bar{\mathbf{w}}'}]_{\bar{\mathbf{w}},\bar{\mathbf{w}}'} & 0 & 0 \\
0 & (\alpha - \beta)[D^{1,s}_i]_{i,s} & 0 & \alpha\beta[B^1_{i,j}]_{i,j} & (\alpha + \beta)[D^{1,s}_i]_{i,s} \\
0 & 0 & 0 & (\alpha + \beta)[D^{1,s}_i]_{s,i} & 0
\end{pmatrix}$$