ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

An Analysis of the Metric Structure of the Weight Space of Feedforward Networks and its Application to Time Series Modeling and Prediction

Arnfried Ossen and Stefan M. Rüger
Informatik, Sekr. FR 5-9, Technische Universität Berlin
Franklinstr. 28/29, 10 587 Berlin, Germany
{ao, async}@cs.tu-berlin.de

Abstract. We study symmetries of feedforward networks in terms of their corresponding groups. We find that these groups naturally act on and partition weight space into disjunct domains. We derive an algorithm to generate representative weight vectors in a fundamental domain. The analysis of the metric structure of the fundamental domain leads to improved evaluation procedures of learning results, such as local error bars estimated using maximum-likelihood and bootstrap methods. It can be implemented efficiently even for large networks. We demonstrate the approach in the area of nonlinear time series modeling and prediction.

1. Introduction

Feedforward networks can be interpreted as a form of nonlinear regression. They offer great flexibility at the price of a complicated structure. It is possible to use classical maximum-likelihood procedures or modern computational approaches, e. g. bootstrap [2] to evaluate learning results. However, the usual gradient-based parameter estimation, or, in the language of neural networks, learning procedures, may get stuck in local extrema. In the case of maximum-likelihood estimation of error bars, estimates at local maxima of the likelihood can be completely wrong. For bootstrap, local extrema lead to unnecessary large error bars [5].

In order to exclude suboptimal maximum-likelihood and bootstrap estimations we propose to use the *location* information of the weight vectors instead. However, owing to a canonical symmetry group (Section 3.), the space of weight vectors has a nontrivial metric structure that is studied in Section 4..

Our approach to improve estimations of error bars exploits the natural metric of a fundamental domain of the weight space with respect to the symmetry group. We propose applying a clustering algorithm for the weight vectors in this effective weight space using the metric given in Section 4. in order to obtain several clusters of weight vectors. Simulations have shown that these clusters refer to different types of maxima of the respective likelihood functions.

ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

2. Learning in a Statistical Context

The data are generated using a "true" model, i.e., a neural network with a certain fixed weight vector w°. In the regression context, we assume "measurement" errors, so the actual data are modeled using:

$$y_i = \mathrm{out}_{w^\circ}(x_i) + \varepsilon_i$$

where the errors ε_i are independent identically distributed. The training set D is a multiset $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ of n such data pairs.

2.1. Likelihood

Given the distribution of the random variable ε_i and a certain weight vector w, every data pair (x_i, y_i) has a probability density $p(x_i, y_i)$ and the joint probability

$$L_D(w) := \prod_i p(x_i, y_i)$$

factorizes by the stochastic independence of the errors. The function $w \mapsto L_D(w)$ is called the likelihood of w. Note that $w \mapsto L_D(w)$ is not a density because the integration over the weight space $\mathbb{R}^E$ may lead to an arbitrary value. $L_D(w)$ describes how likely the data have been generated by w. Under quite general assumptions, the weight vector

$$\hat{w}_n = \operatorname*{argsup}_w(L_D(w))$$

that maximizes the likelihood is asymptotically (w.r.t. n) unbiased, consistent, asymptotically efficient and asymptotically normal-distributed [7].

The maximum-likelihood estimation $\hat{w}_n$ may be obtained, e.g., by gradient ascent in the likelihood, or in the logarithm of the likelihood.

2.2. Bootstrap Approach

The bootstrap method [2] is based on re-estimations of the parameter vector on B bootstrap samples of the training set. The bth bootstrap sample is a random multiset $D^{*b} = \{(x_1^{*b}, y_1^{*b}), \ldots, (x_n^{*b}, y_n^{*b})\}$ drawn from the training data *with replacement*, i.e., some of the original data pairs will not appear, and some will appear multiply.

The standard error of a predicted value $\mathrm{out}_{\hat{w}}(x)$ is approximately given by

$$\sqrt{\frac{1}{B-1}\sum_{b=1}^{B}\left(\mathrm{out}_{\hat{w}^{*b}}(x) - \frac{1}{B}\sum_{b'=1}^{B}\mathrm{out}_{\hat{w}^{*b'}}(x)\right)^2},$$

where $\hat{w}^{*b}$ is the estimated weight vector according to $\hat{w}^{*b} = \mathrm{argsup}(L_{D^{*b}}(w))$. The $\hat{w}^{*1}, \ldots, \hat{w}^{*B}$ are realizations of the random bootstrap weight vector $\hat{w}^*$.

It is also possible to estimate confidence intervals using the bootstrap approach. Let $\mathrm{out}^{\alpha}_{\hat{w}^*}(x)$ be the α quantile of the empirical distribution of $\mathrm{out}_{\hat{w}^*}(x)$. The approximate $1 - 2\alpha$ confidence interval of a predicted value $\mathrm{out}_{\hat{w}}(x)$ is

$$[\mathrm{out}^{\alpha}_{\hat{w}^*}(x), \mathrm{out}^{1-\alpha}_{\hat{w}^*}(x)].$$

ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

However, to achieve good accuracy, many more bootstrap samples and elaborate post-processing may be required. See [2] for details.

3. Symmetry Group of $\mathbb{R}^E$

Every transformation $t: \mathbb{R}^E \to \mathbb{R}^E$ of the weight space, which leaves the network function invariant, i.e., $\text{out}_w = \text{out}_{t(w)}$, indicates a symmetry of the weight space. One kind of symmetry is quite obvious: The interchange of two hidden nodes within a hidden layer does not change the network function owing to the commutativity of the summation in the nodes of the next layer. Another symmetry may be induced by a symmetry of the activation function. For the sake of concreteness, we will describe the symmetry group for the very popular feedforward networks.

3.1. Network Structure

The nodes of the network are addressed by numbers 0, 1, 2, etc., 0 being the name of the bias node. All other nodes are divided into $k \geq 1$ hidden layers $L_1, \ldots, L_k$, one input layer L_o and one output layer L_{k+1}. Every layer is fully connected to the next layer, which means that there is a weight w_{ab} assigned to every pair (a, b) of nodes from $L_i \times L_{i+1}$, $i = 0, 1, \ldots, k$. Hidden nodes and output nodes have a bias weight termed w_{oa}. All weights of the network form a weight vector $w \in \mathbb{R}^E$.

Each activation function f of a hidden layer node is assumed to be sigmoidal; we require that every activation function exhibit the same type of symmetry $f(x) = e - f(-x)$, where, e.g., $e = 1$ for the logistic function or $e = 0$ for $f = \tanh$. This class of networks is quite universal: choosing the activation functions of the output layer as identity and using one hidden layer makes this type of network an universal approximator [7].

3.2. Symmetries

There have been several attempts to analyze the symmetries of the network function [6, 1]. We present an independent approach, in which the arising symmetries are described and analyzed in terms of their corresponding groups, see e.g. [3].

Let $\Sigma(M)$ denote the permutation group of a set M. Bear in mind that every permutation can be written in terms of transpositions. The transposition $\tau(a, i)$ of a node $a \in L_i$ with its right neighbor node induces a certain permutation $\pi(a, i)$ of the weight vector components, which leaves the network function out_w invariant. This permutation $\pi(a, i)$ may be viewed as a linear operation on the weight space $\mathbb{R}^E$, and is thus an element of the group $\text{GL}(E, \mathbb{R})$ of all linear functions $\mathbb{R}^E \to \mathbb{R}^E$.

Fix i; the mappings $\tau(a, i) \mapsto \pi(a, i)$ with varying a induce a monomorphism (i.e., injective homomorphism) from $\Sigma(L_i)$ to $\text{GL}(E, \mathbb{R})$. Let Π_i denote the image of this monomorphism. Π_i can be identified with the group generated by $\pi(a_1, i), \pi(a_2, i), \ldots$, i.e., with the smallest subgroup of $\text{GL}(E, \mathbb{R})$, which contains all $\pi(a_1, i), \pi(a_2, i), \ldots$

ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

Further studies show that Π_i commutes with Π_j element by element if $i \neq j$. The group Π, which is generated by $\Pi_1, \ldots, \Pi_k$, turns out to be isomorphic to the direct product of $\Pi_1, \ldots, \Pi_k$. In particular, Π has $|L_1|! \cdot \ldots \cdot |L_k|!$ elements.

Let $\tilde{w}(b,i)$ be the subvector of w containing all weights that involve the hidden node b of layer L_i: $\tilde{w}(b,i) = (w_{ob}, w_{a_1 b}, w_{a_2 b}, \ldots, w_{bc_1}, w_{bc_2}, \ldots)$. a and c are enumerations of the nodes in the layer L_{i-1} and L_{i+1}, respectively. The flipping of all signs of the components in the subvector $\tilde{w}(b,i)$ can be corrected by changing all bias weights of the nodes of the layer L_{i+1}, i. e., by $w_{oc_i} \mapsto w_{oc_i} + e \cdot w_{bc_i}$. This is induced by the symmetry $f(x) = e - f(-x)$.

Let $t_b \colon \mathbb{R}^E \to \mathbb{R}^E$ denote the above linear operation (sign flip of all weight components that deal with node b and correction of all bias weight in the next layer) that leaves out_w invariant. Note that t_b is idempotent, and thus the group induced by t_b is the cyclic group $T_b := \{1, t_b\}$ with two elements. Verify that t_b commutes with t_a for all hidden nodes a, b and that no t_b can be expressed by any combination $t_{a_1}, \ldots, t_{a_n}$ of other operations when all $a_i \neq b$. From this it can be deduced that the subgroup T, which is generated by all the operations t_b, is abelian and isomorphic to the direct product of all T_b. In particular, T has $2^{|L_1 \cup \ldots \cup L_k|}$ elements.

Let $S \subset \mathrm{GL}(E, \mathbb{R})$ denote the group generated by Π and T. By definition, T is a subgroup of S. T turns out to be normal, yielding the result $S = T\Pi = \Pi T$. It follows that each element s of the symmetry group S has a *unique* representation $s = \pi_k \ldots \pi_1 t$ with $\pi_i \in \Pi_i$ and $t \in T$. So far the symmetry group S of the weight space has been identified as a certain subgroup of $\mathrm{GL}(E, \mathbb{R})$.

As pointed out by [1] no analytic function other than an element of S can represent a symmetry in this context. However, there exist a lot of discontinuous functions that give rise to a symmetry: Fix two hidden nodes a, b from the same hidden layer. If the incoming weights of a coincide with the incoming weights of b, then all corresponding two outgoing weights might be replaced by their average value. These kinds of symmetries are probably not important in practice as they live in hyperplanes of $\mathbb{R}^E$ with zero Lebesgue measure. We therefore exclude them from our studies.

3.3. A Fundamental Domain

S acts on $\mathbb{R}^E$ in a natural way: distinct orbits $S(w) := \{x \in \mathbb{R}^E \mid x = s(w) \text{ for some } s \in S\}$ (with respect to the natural group action) partition $\mathbb{R}^E$. There is an interesting open and *convex* set W of weight vectors which contains at most one representative of every orbit such that $S(W)$ is dense in $\mathbb{R}^E$. Such a remarkable set is called fundamental domain and may be constructed as follows:

Let $a_1^i, \ldots, a_{|L_i|}^i$ denote the nodes of hidden layer L_i. Let $\tilde{w}(b,i) \prec \tilde{w}(c,i)$ denote the lexicographic comparison of two subvectors. As usual, this means that $\tilde{w}(b,i) \neq \tilde{w}(c,i)$ and that the first nonzero component of $\tilde{w}(c,i) - \tilde{w}(b,i)$ is positive. Then,

$$W := \left\{ w \in \mathbb{R}^E \mid 0 \prec \tilde{w}(a_1^i, i) \prec \ldots \prec \tilde{w}(a_{|L_i|}^i, i) \text{ for all } 1 \leq i \leq k \right\}$$

is a fundamental domain (note that W is a cone with apex 0, i. e., $cw \in W$ for all $w \in W$ and $c > 0$).

ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

Sketch of a proof: Applying T on W allows each first nonzero component of all subvectors $\tilde{w}(b,i)$ to change its sign, and applying Π creates all possible orders of the first nonzero components, thus removing any restriction given to specify W except the condition that no subvector $\tilde{w}(a,i)$ of w may vanish and except that no two subvectors may coincide. Taking the closure removes these conditions leaving $\mathbb{R}^E$.

Hence, it suffices to maximize the likelihood in W instead of the much larger space $\mathbb{R}^E$. Indeed, the idea of a fundamental domain is to define a convenient nonredundant subset of $\mathbb{R}^E$ with respect to S.

3.4. Algorithm for Representative Vectors

Let $\hat{w}$ denote a weight vector resulting from some learning algorithm (or parameter estimation). Beginning with layer L_1, apply the symmetry operation t_a of Section 3.2. whenever a bias weight of a hidden node a is negative until all hidden nodes have nonnegative bias weights. Then, for every hidden layer, apply permutation symmetry operations by interchanging subvectors of nodes in this layer such that the bias weights of each layer are in a definite order — say, ascending from left to right — thus arriving at a representative weight vector $r(\hat{w})$. Essentially, this is a simple sorting problem.

The function $r: \mathbb{R}^E \to \mathbb{R}^E$ as implemented by the above algorithm maps onto a region $W' \supset W$. The difference $W' \setminus W$ is caused by the algorithm's laziness of not doing a complete lexicographic comparison and by dealing with weight vectors that have vanishing or coinciding subvectors. Fortunately, $W' \setminus W$ has zero Lebesgue measure.

4. The Metric Structure of W

4.1. W as a Manifold

The space $\mathbb{R}^E$ of weight vectors is highly redundant. Ideally, $\mathbb{R}^E$ should be replaced by the space $M := (\mathbb{R}^E \setminus X)/S$ of distinct orbits, where X denotes the set of weight vectors with a vanishing subvector or with two coinciding subvectors of nodes in the same hidden layer. The division by the symmetry group identifies all weight vectors which have the same network function. M, in general, is then a curved manifold with singular points.

Here, the problem arises of how to define learning algorithms or statistics in the manifold M, which has completely different geometric properties than $\mathbb{R}^E$. Instead of bothering with questions like geodesics, parallel transport and curvature, we propose learning in the flat space $\mathbb{R}^E$ and mapping the learning result to W.

In order to achieve a good clustering, the differentiable structure of M is not necessary. All one needs is a metric to replace the standard euclidian distance measure in present clustering algorithms.

ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

4.2. W as a Metric Space

One idea assigning a distance to two points in W is using the minimal distance d_E in $\mathbb{R}^E$ of two points of their corresponding orbits:

$$d(x,y) := \min_{s_1,s_2 \in S} d_E(s_1 x, s_2 y)$$

However, it turns out that the compatibility of d_E with the symmetry group, i. e., for all $s \in S$ it holds that $d_E(sx,sy) = d_E(x,y)$, is a quite important feature. If a metric d_E in $\mathbb{R}^E$ is compatible with S, then the computational simpler expression

$$d(x,y) := \min_{s \in S} d_E(sx,y)$$

is a canonical metric in W. If the activation functions of the hidden layer exhibit the symmetry $f(x) = -f(-x)$, e. g. for the tanh function, then the euclidian metric is compatible with S: all points of the orbit of a weight vector w lie on a sphere with radius $|w|$.

However, if we use the logistic function for the activation functions, a sign flip of a subvector causes bias corrections to be made in next layer. They change the euclidian norm of the equivalent vector. Fortunately, there is a norm $|\cdot|_E$ in $\mathbb{R}^E$ which induces a metric that is compatible with S. If only one hidden layer is involved, one can choose

$$|w|_E = \max_{d \in L_2} \left\{ |w|_\infty, \left|w_{od} + \sum_{c \in L_1} \max(0, w_{cd})\right|, \left|w_{od} + \sum_{c \in L_1} \min(0, w_{cd})\right| \right\},$$

where $|\cdot|_\infty$ denotes the maximum norm in $\mathbb{R}^E$. The unit sphere in this peculiar norm $|\cdot|_E$ is distorted so that all points of an orbit of a certain vector lie on the same sphere. The generalization to more hidden layers is straightforward.

This norm is a natural choice for a metric $d_E(x,y) := |x-y|_E$ in $\mathbb{R}^E$ when dealing with the logistic activation function. The distortion of the unit sphere reflects the preference of the logistic function for positive output values.

5. Simulations

We chose an application domain where a good assessment of uncertainty is crucial: the prediction of time series in the financial markets. Decisions should be based on correct estimations of error bars, or even more general, interval predictions, because it is then possible to assess the risk involved. The smaller the error bars are, the bigger the profit will be on average.

The actual time series we selected is a short term interest rate. Its prediction is used to support traders in the forward rate agreement market [4].

5.1. Modeling, Refinement and Results

The time series was modeled indirectly as the difference between certain market expectations in the past and actual rates at present; see [4] for details. We used two-layer feedforward networks in a nonlinear auto-regressive setting. Network inputs were past values of the series at specific time lags. Network

ESANN'1996 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-25-26 April 1996, D-Facto public., ISBN 2-9600049-6-5, pp. 315-322

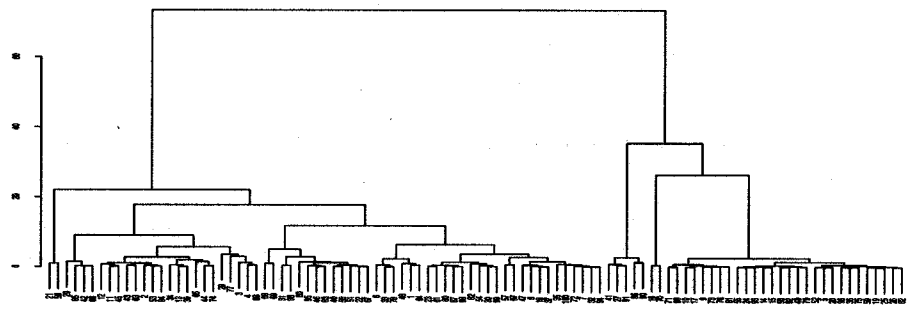

Figure 1: Plot of hierarchical clustering (complete linkage method) of $B = 100$ bootstrap estimates of network weight vectors in time series prediction. Left subtree corresponds to global minimum.

outputs directly predicted values in the future. A model selection resulted in a feedforward network with 10 input units, 5 hidden units, 1 output unit, and a time lag of 3 between past input values.

For time series, the method of least squares can still be applied to estimate network weights. However, in contrast to ordinary nonlinear regression, it is not directly equivalent to the maximum-likelihood method. Therefore, the bootstrap pseudo samples were generated according to the moving blocks method [2]. The empirical distribution of network function outputs was used to estimate a number of quantiles of the distribution. These in turn were evaluated to estimate an empirical predictive density. Figure 2 shows a single predictive density, in this example for 21 May, 1993. Surprisingly, it is a multi-modal distribution, potentially caused by local minima in the network error function.

The complete analysis of symmetries, groups, fundamental domains and metric spaces applies here, too, because we essentially use the same type of network. If the deficiencies of the predictive distributions are caused by learning procedures stuck in local minima, the clustering approach should help.

After application of the algorithm for unique weight vector representatives (see Section 3.4.) resulting in a fundamental domain, a hierarchical clustering algorithm (see Figure 1) uncovered the fact that only about 60 percent of the bootstrap estimations corresponded to the global minimum in the network error function. Bootstrap weight vectors not belonging to the global minimum were excluded from the bootstrap process. The constrained bootstrap estimation of the predictive density (Figure 3) is better localized and unimodal, which improves the results of [4].

6. Discussion

A major drawback of popular learning procedures for feedforward networks is that they are gradient-based. Therefore they may get stuck in local extrema. In the case of maximum-likelihood estimation of error bars, estimates at local maxima of the likelihood can be completely wrong. For bootstrap, local

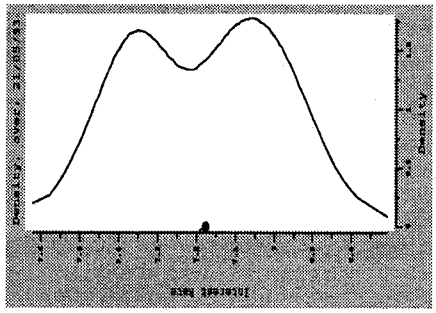

Figure 2: Simple moving blocks bootstrap results in multi-modal and wide distribution for 21 May, 1993.

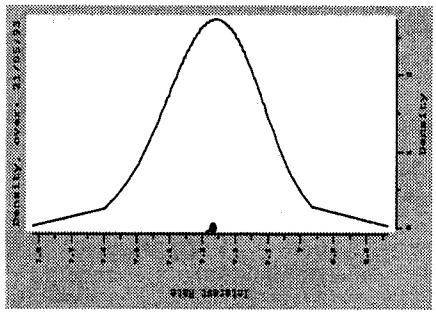

Figure 3: Moving blocks bootstrap estimates confined to global minimum by clustering procedure.

extrema can lead to unnecessary large error bars or unnecessary wide confidence intervals. We propose to cluster weight vectors in a small, well defined fundamental domain of weight space using its natural metric. In practice, it is sufficient to use standard hierarchical clustering algorithms to discriminate between "good" and "bad". The method can be implemented efficiently even for large networks and large datasets.

References

[1] An Mei Chen, Haw-minn Lu, and Robert Hecht-Nielsen. On the geometry of feedforward neural network error surfaces. *Neural Computation*, 5(6):910–927, 1993.

[2] Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*, volume 57 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, 1993.

[3] Ian D. Macdonald. *The theory of groups*. Oxford University Press, 1975.

[4] Arnfried Ossen and Martin Schnauss. Practical tools for derivative instruments based on nonlinear time series prediction. *Neural Network World*, 5(4):525–536, 1995. Proceedings — International Workshop on Parallel Applications in Statistics and Economics.

[5] Stefan M. Rüger and Arnfried Ossen. Performance evaluation of feedforward networks using computational methods. In *Proceedings of NEURAP'95*. NEURAP, 1996.

[6] H. J. Sussmann. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, 5:589–593, 1992.

[7] Halbert White. *Artificial Neural Networks — Approximation & Learning Theory*. Blackwell, Oxford, Cambridge, 1992.