Interval Analysis Based Neural Network Inversion: A Means for Evaluating Generalization

S.P. Adam[1,2(✉)], A.C. Likas[3], and M.N. Vrahatis[2]

[1] Department of Computer Engineering, Technological Education Institute of Epirus, Arta, Greece
adamsp@teiep.gr
[2] Computational Intelligence Laboratory, Department of Mathematics, University of Patras, Rion - Patras, Greece
adamsp@upatras.gr
[3] Department of Computer Science and Engineering, University of Ioannina, Ioannina, Greece

Abstract. Inversion of a neural network trained on some classification problem has been an important issue related to the explanation of the neural classification function. Inversion based on Interval Analysis (IA) [1] showed that a reliable estimation of the neural network domain of validity is feasible and a number of quantitative issues arise from this inversion. This paper deals with the investigation of these quantitative issues and more precisely with those concerning the evaluation of the neural network classification function in terms of generalization, comparison of different network models and classification accuracy. Preliminary experimental results indicate that the IA-based inversion can offer a solid basis towards reliable evaluation of the neural classification function.

Keywords: Neural networks · Generalization · Interval Analysis · Reliable computation

1 Introduction

Inversion of a trained network has always been one of the objectives in neural network research as it permits to define the input space area covered by the network function, to delineate the decision boundaries learned by the network and to extract rules explaining the network operation. Hence, neural network classification is related to the so-called domain of validity of the network, which results from network inversion and it can be used either to provide a qualitative conclusion of the neural classification function [12,14,17] or to extract provably correct rules [5,20,22] explaining neural network operation. A number of approaches can be found in the literature which permit to define such a domain [3,16]. However, while an accurate definition of the domain of validity should be an obvious requirement of any approach used for this problem it seems that

DOI: 10.1007/978-3-319-65172-9_27

such a requirement had not effectively been tackled. As a result, to the best of our knowledge, there has been no research effort towards examining the domain of validity of a neural network in quantitative terms.

Recently, Adam et al. [1] proposed an IA-based approach for neural network inversion resulting in reliable definition of its domain of validity. Inversion of the network is carried out using an IA approach which, for any interval of the network output activity, defines a unique, consistent and guaranteed domain in the input space. The proposed method is termed to provide reliable estimation as it permits to define regions of validity in a guaranteed way. The results obtained in [1] are interesting in the sense that they provide quantitative information about the domain of validity of the neural network. This level of information seems to be inadequate for the classical explanation of neural network operation but as noted in [1] it may be used to provide useful insight to the neural classification task concerning the generalization ability of the trained network as well as the fitness of the neural network model adopted.

The aim of this paper is to advance on the hypotheses formulated in [1] by carrying out a number of experiments in order to evaluate the soundness of these statements. This paper deals with the following matters:

- Based on the volume of the domain of validity derive empirical metrics for the evaluation of the performance of a trained network in a classification task.
- Assess the generalization ability of a trained network using the previous metrics and compare the results with the classical cross-validation approach.
- Discuss open problems and future work.

The experimental results obtained provide concrete evidence that important aspects concerning the validity of the neural classification task can be evaluated using the empirical metrics defined. The reliability of the proposed metrics is supported by the IA-based inversion which provides verified results in a guaranteed way, as the interval computations permit to automatically verify the results obtained [13].

The paper is organized as following. Section 2 outlines the classification context along with the main assumptions and some theoretical results. Section 3 is dedicated to the description of the basic interval arithmetic concepts and the inversion procedure based on IA. In Sect. 4 we present the proposed approach along with the empirical metrics defined. Section 5 is devoted to the experimental evaluation and the discussion of the results obtained. Finally, Sect. 6 concludes the paper.

2 Problem Definition and Background

The analysis presented in this paper deals with, but is not limited to, multilayer perceptrons (MLPs) which are considered to have one or more hidden layers, nodes with sigmoidal nonlinearities and being trained with some gradient descent procedure. The network is trained on a classification problem with M classes, $C_1, C_2, \ldots, C_M$, using a sample data set $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$, of P examples defined by the N-dimensional patterns $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_P\}$ instantiating the random variable $\mathcal{X}$, and the desired outputs $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_P\}$ for the random variable $\mathcal{Y}$. Output for each class is 1 of M, denoting that, there is one output unit corresponding to the correct class while all others are zero.

Classification decisions based on the *ad hoc* rule "the pattern $\mathbf{x}$ is assigned membership in class C_j if the corresponding output value is greater than some fixed threshold" are ambiguous as they may assign a pattern to multiple classes. Such decisions become unambiguous, if the following rule is used instead, "$\mathbf{x}$ is considered to belong to class C_j if the jth component of the network output is greater than all the other components" [9].

Important research focused on the operation of a neural network, in terms of defining decision rules governing their function and explaining how decision boundaries are formed by neural network outputs in classification problems [9]. A significant part of this research relates operation of an MLP classifier with Bayesian classification [2,4,6,15].

Hampshire and Pearlmutter [7] provided detailed proofs that, when dealing with asymptotically large sets of statistically independent training samples, MLP classifiers provide outputs which act as optimal Bayesian discriminant functions for these training samples. They also discussed necessary and sufficient conditions on the form of objective functions that yield Bayesian discriminant performance by engendering classifier outputs that are true estimates of the *a posteriori* probabilities $P(C_j|\mathbf{x})$. Richard and Lippmann [18], also, advanced on the previous statements, giving detailed proofs and analysis of some important network models such as MLPs, radial basis function (RBF) and high-order polynomial networks. They showed that the outputs of these networks provide good estimates of Bayesian probabilities. Estimation accuracy depends on network complexity the amount of training data, and the degree to which training data reflect true likelihood distributions and *a priori* class probabilities.

Here, let us assume that the network target outputs are considered to be binary, i.e. in the interval [0,1]. For an MLP classifier approximating the *a posteriori* class probabilities means that the conditional expectation of the desired output response vector, given the input data vector $\mathbf{x}$, that is $E(d_j = 1|\mathbf{x})$, equals the *posterior* class probability $P(\omega = \omega_j|\mathbf{x})$, where $\omega \in \Omega = \{\omega_1, \omega_2, \ldots, \omega_m\}$ and $\omega = \omega_j$ denotes membership of $\mathbf{x}$ to class C_j for $j = 1, 2, \ldots, M$.

As Hampshire and Pearlmutter [7] showed, in the case of perfect training, sufficiently large data set and complex network, the relation between the *a posteriori* class probabilities and the network output values is linear and so $P(\omega = \omega_j|\mathbf{x})$ is mapped to the interval [0,1]. However, due to various perturbations the target values are non binary and so $P(\omega = \omega_j|\mathbf{x})$, while still being linear, it is mapped to the interval $[\epsilon, 1-\epsilon]$ for some appropriate value of ϵ (e.g. $\epsilon = 0.2$). Similar conclusions were, also, provided by Richard and Lippmann in [18] regarding the degradation of the estimation accuracy of Bayesian posterior probabilities.

In many practical classification tasks when the jth output node is active assigning the input pattern to the jth class then its value is considered to be in an interval $[1-\beta, 1]$, defined for some suitably chosen value of β (e.g. $\beta = 0.4$).

This constitutes an intuitive, yet practical and efficient way, to deal with output uncertainty due to imperfect training produced by various reasons such as an inexact error threshold, insufficient number of training epochs or when having a small sized training set. On the other hand, an output node is considered to be inactive if its activation is in the interval $[0, \beta]$. The greater the value of β the poorer the classification accuracy achieved by the trained network. Hence, defining the domain of validity generating output values in the interval $[1-\beta, 1]$ seems to be a necessary tool to support the performance analysis of the network classification function. This can be achieved using IA concepts and SIVIA [10].

3 IA-Based Inversion and SIVIA

Interval arithmetic was introduced as a means to perform numerical computations with guaranteed accuracy and bounding the ranges of the quantities, used in the computations. An interval, or interval number, I is a closed interval $[a, b] \subset \mathbb{R}$ of all real numbers between (and including) the endpoints a and b, with $a \leqslant b$. In practical calculations interval arithmetic operation are reduced to operations between real numbers [10]. Hereafter, the bracketed notation $[x] = [\underline{x}, \overline{x}]$ denotes an interval object such as a number, variable, vector, matrix, etc. The set of n-dimensional vectors of real intervals is denoted by $\mathbb{IR}^{\mathrm{n}}$. The definition of interval objects such as vectors, matrices, functions, etc. and their subsequent study resulted in the establishment of Interval Analysis.

If $[x] \subseteq D$ is an interval in the domain of a real function $f : D \subset \mathbb{R} \to \mathbb{R}$ then $f([x])$ is used to denote the range of values of f over $[x]$. Computing such a range, $f([x])$, using IA tools means to enclose it by an interval which is as narrow as possible. This constitutes an important matter in IA as it is used in various problems: localization and enclosure of global minimizers of f on $[x]$, verification of $f([x]) \subseteq [y]$ for given $[y]$, nonexistence of a zero of f in $[x]$ etc. In order to enclose $f([x])$ one needs to define a suitable interval function $[f] : \mathbb{IR} \to \mathbb{IR}$ such that $\forall [x] \in \mathbb{IR},\ \ f([x]) \subset [f]([x])$, see Fig. 1.

SIVIA is an interval method introduced by Jaulin and Walter [11] in order to allow for the guaranteed estimation of nonlinear parameters from bounded error data. The method proceeds by defining a box or union of boxes enclosing a set of interest. Hence, given a function $f : X \to Y$, where $X \subset \mathbb{R}^n$, $Y \subset \mathbb{R}^m$ and an interval vector, i.e. a box, $[y] \subseteq Y$, the objective is to define the set of unknown vectors $x \in X$ such that $f(x) \in [y]$. This set can be defined as $S = \{x \in X \subseteq \mathbb{R}^n | f(x) \in [y]\} = f^{-1}([y]) \cap X$, where X is the search space containing the set of interest S; $[y]$ is known in advance to enclose the image of the set $f(S)$ and S denotes the unknown set of interest. Note that, here, f^{-1} denotes the reciprocal image of f, as f may not be invertible in the classical sense.

The solution proposed by SIVIA for this problem consists in computing boxes and unions of boxes S^- and $S^+ = S^- \cup \Delta S$ which form guaranteed outer and inner enclosures of S as they satisfy the relation $S^- \subseteq S \subseteq S^+$, [10]. SIVIA is a branch-and-bound approach which computes enclosures by recursively exploring the whole search space. During computation, a box $[x] \in \mathbb{R}^n$ is designated as feasible if $[x] \subseteq S$ and $f([x]) \subseteq [y]$, infeasible if $[f]([x]) \cap [y] = \emptyset$ and, in all other cases, $[x]$ is said to be indeterminate which means that $[x]$ may be feasible, unfeasible or ambiguous. The condition $[x] \subseteq S$ and $f([x]) \subseteq [y]$ is necessary and sufficient for $[x]$ to be feasible. Feasible boxes are added to S^- and infeasible become members of the complement of S^+. Finally, any indeterminate box is bisected and the method recursively examines the two resulting sub-boxes. Bisection is possible up to some limit, which is preset for the problem and defines its resolution. Boxes that are indeterminate and cannot be further bisected are added to the union ΔS. For a detailed description of the algorithm implementing SIVIA the reader should refer to [10]. Finally, note that SIVIA applies to any function f for which an inclusion function $[f]$ can be computed.

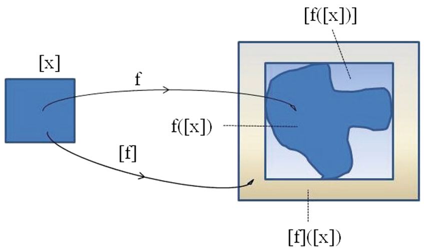


Fig. 1. A function f, an inclusion function $[f]$ and the images of $[x]$

4 Inversion-Based Generalization Metrics

The necessity of using an interval $[1-\beta, 1]$ is illustrated by the 2-dimensional classification problem with two classes shown in Fig. 2a. A $2-10-2$ MLP, using the hyperbolic tangent activation for the hidden nodes and the logistic sigmoid one for the output nodes, was trained on this problem and produced the output shown in Fig. 2b. Patterns classified as class 1 form the white regions. Obviously, the gray level zone depicts the ambiguity of classification for patterns near the class 1 area which gives values of the MLP output in some interval $[1-\beta, 1]$.

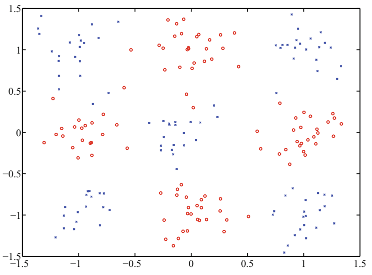
The value of β clearly extends or restricts the area in the input space classified by the MLP as class 1. One can verify this argument by simple reference to Figs. 3a and b where the red colored areas have been defined by inverting the MLP output using SIVIA. The striking difference between these two Figures concerns a significant part of the input space effectively belonging to class 1 which is present in Fig. 3b but not in Fig. 3a. This shows the importance of β which, here, needs to be given the value 0.1 if one wants to take into account a significant part of the input space. However, as shown in Figs. 4a and b with a higher number of training epochs and a smaller error threshold this deficiency seems to be remedied.

It is well known that the term generalization refers to the ability of a trained neural network to correctly classify previously unseen patterns. Achieving good generalization can be seen in two different ways; either have some fixed architecture of the network and determine the size of the training set, or start from a fixed size of the training set and define the best network architecture [9]. A number of studies carried out on these issues brought interesting research results, which however rely on theoretical assumptions such as: the a priori knowledge of the distribution of the network weight vectors, the confidence that the distribution of the training patterns is a good approximation of the distribution of the input space or even that the network architecture is well suited for the problem at hand, see [8] and references therein. In practice, none of these assumptions is verified and both researchers and engineers proceed with a fixed data set and an initial network model along with some complexity regularization technique in order to achieve the best network architecture. A well known approach for resolving the generalization issue is cross-validation [9] and especially the multifold cross-validation which is used in our experiments.

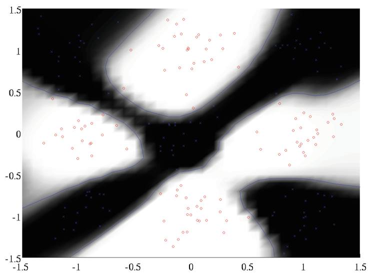
In the context of this paper classification of previously unseen patterns is considered under the following statement; the network will be able to correctly classify those patterns that fall into the area of the input space learned by the network during training. This area is, precisely, the domain of validity of the network as defined by the IA-based inversion of the network. The larger the domain of validity, the bigger its volume and so the higher the probability for some unknown pattern to be in this area and be classified. Hence, the first condition for some unknown pattern to be classified correctly is to be in the domain of validity of the network. Then, assigning the pattern to the right class requires the pattern to be in the area corresponding to the domain of validity of its respective class. These two conditions are the basis of the definition of two metrics.

In the case of an MLP classifier and a 1 of M encoding, in order to compute the total volume of the domain of validity of the network one needs to perform M inversions, one for each output node, compute the volume of each defined area and finally sum up the volumes of all the classes. Hence, if $V_1, V_2, \ldots, V_M$ are the volumes computed for the M classes, then the total volume of the domain of validity is $V_{net} = \sum_{i=1}^{M} V_i$.

In consequence, if V_{input} denotes the finite, non zero, volume of the input space then the ratio $\frac{V_{net}}{V_{input}}$, defines a measure of the probability for some new pattern to be effectively classified by the network. Moreover, this ratio is a metric for measuring over-training of the network. In order to cope with the second condition and evaluate the ability of the trained network to correctly classify a new pattern we need to rate the ability of the domain of validity of each class to contain the patterns effectively belonging to this class while excluding all the others. Actually for some class C_i some patterns may happen either to be misclassified to another class C_j or to be unclassified, which means that these patterns are outside the domain of validity of the network, i.e. the network

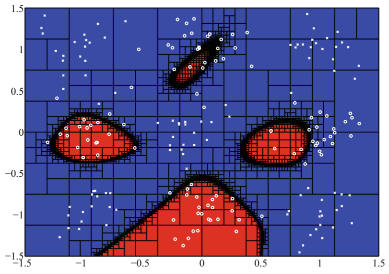


(a) The artificial data set

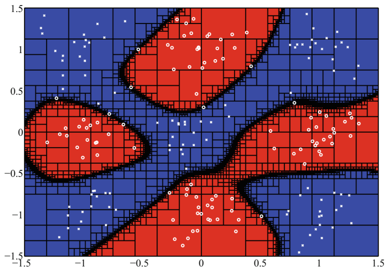


(b) Contour plot of the MLP output

Fig. 2. An artificial problem and the contour plot of an MLP trained on this data set

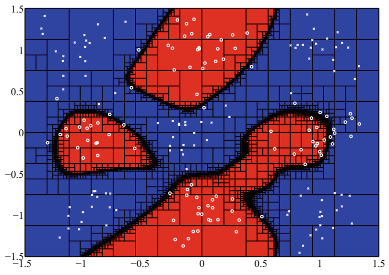


(a) Input area for the interval [0.999,1]

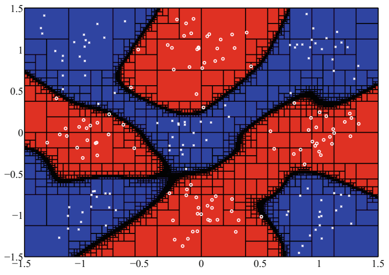


(b) Input area for the interval [0.9,1]

Fig. 3. How the interval $[1-\beta, 1]$ affects the domain of validity of an MLP trained for 500 epochs and MSE $\leqslant 1e-03$



(a) Input area for the interval [0.999,1]



(b) Input area for the interval [0.9,1]

Fig. 4. How the interval $[1-\beta, 1]$ affects the domain of validity of an MLP trained for 5000 epochs and MSE $\leqslant 1e-05$

output signal for those patterns is not in the interval $[1-\beta, 1]$. In order to tackle this problem two solutions are envisaged in this paper:

A. Penalize the domain of validity for the incorrect classifications by subtracting from its volume the volume corresponding to each misclassified or unclassified pattern.
B. For each box defined by SIVIA for any class C_j compute its volume by taking into account the number of patterns correctly classified in this box.

For the first approach we need to define the volume of the input space corresponding to each misclassified/unclassified pattern. A simple way to do this is to consider that to any input pattern corresponds an elementary volume V_{elem} of the input space which can be defined by simply taking $V_{elem} = \frac{V_{input}}{P}$, that is this elementary volume is proportional to the number of patterns available for the classification problem. This choice can be explained by the following argument; the more the input patterns the lower the probability of having a defective classification function due to the size of the training set. So, the penalty imposed on the domain of validity is proportional to the size of the training set. Hence, if l is the total number of misclassified/unclassified patterns then a generalization metric is given by the formula:

$$G_{net} = \frac{V_{net} - lV_{elem}}{V_{input}}. \tag{1}$$

This quantity G_{net} defines a metric measuring over-training while taking into account the classification errors of the trained network. Concerning the second approach let us consider the following:

– for each pattern $\mathbf{x}_n$ in the training set X let $C(\mathbf{x}_n)$ denote the class of this pattern,
– if C_k, $1 \leqslant k \leqslant M$, is the kth class then the following set of boxes is defined by IA-based inversion of the kth output node of the MLP: $\mathbb{B}^k = \{\boldsymbol{B}_1^k, \boldsymbol{B}_2^k, \ldots, \boldsymbol{B}_{N_k}^k\}$,
– let V_i^k denote the volume of $\boldsymbol{B}_i^k$,
– let X_i^k be the set of patterns $\mathbf{x}_n$ found to be inside the box $\boldsymbol{B}_i^k$.

Then the validity of the box $\boldsymbol{B}_i^k$ can be defined as

$$E_i^k = \left[\sum_{\mathbf{x}_n \in X_i^k} \left(\mathbb{1}_{\mathbb{C}} \left(C(\mathbf{x}_n) = C_k \right) - \mathbb{1}_{\mathbb{C}} \left(C(\mathbf{x}_n) \neq C_k \right) \right) \right] V_i^k, \tag{2}$$

where $\mathbb{1}_{\mathbb{C}}$ is a suitable indicator function. Finally, if one wants to obtain the total validity of the trained network according to these hypotheses then one has to compute the formula

$$E = \sum_{k=1}^{M} E_i^k. \tag{3}$$

The second approach for every box, identified by IA-based inversion, computes its part effectively belonging to some class by taking into account the patterns found in this box. In consequence, the metric provided by this approach has a twofold effect; first, for every box belonging to some class C_j, as determined by inversion, it accounts only the part of the box effectively containing patterns of the class C_j, and second it rejects those parts of the input space that do not contain any pattern at all. Doing so it rejects an important part of the interpolations and the extrapolations computed by the network during its training and hence it provides a validity index for the trained network.

5 Experimental Evaluation and Discussion

In this section we describe the setup for the experiments carried out in order to evaluate the statements of the previous section. Then, we discuss the results obtained pointing out some interesting characteristics and finally, we describe some potential issues for future work.

5.1 Experimental Evaluation

Experiments in this paper were executed using SCS Toolbox [23] which implements SIVIA using INTLAB, the MATLAB package of Rump [19] for interval computations.

Two data sets were used for our experiments. The first is the artificial data set for the classification of a pattern belonging to one of two classes originally defined in [21]. This data set is perfectly balanced as it accounts 100 patterns per class and the distribution of the classes has been modified in order to increase overlapping between them, Fig. 2a. For this data set we used 8 MLPs having an architecture $2 - H - 2$ with H taking on the values $4, 5, 6, 8, 10, 15, 20, 25$, and using the hyperbolic tangent for all nodes in the hidden layer and the logistic sigmoid for the output nodes. The data set was divided in twenty subsets in order to compare the results obtained by the proposed approach with the multifold (20-fold) cross-validation. Each MLP was trained until reaching a MSE of 0.001 or for a maximum number of 5000 epochs thus giving a total number of $(20\,\text{trials} \times 8\,\text{models}) = 160$ training trials. For each trial the generalization was computed as the average of the correctly classified validation patterns for the cross-validation method.

For the proposed generalization metrics the network trained in each trial was first inverted using SIVIA. Hence, for each network model 20 trials resulted in 20 inversions and the volumes of the 20 domains of validity were used to compute the mean value for each metric and each network model. The interval of the output values inverted is $[0.8, 1]$. Table 1 outlines the results of this experiment.

The second data set used is the well known Fisher-Iris problem. Experiments were carried out for 6 MLPs having an architecture $4 - H - 3$ with H taking on the values $2, 3, 5, 8, 10, 15$, and using the hyperbolic tangent for all nodes in the hidden layer and the logistic sigmoid for the output nodes. The data set was

Table 1. Results for the artificial data set (higher values indicate better generalization)

Network model	Cross-validation mean (std)	Metric 1 mean (std)	Metric 2 mean (std)
2-4-2	90.50% (9.45%)	96.45% (2.81%)	16.92 (18.57)
2-5-2	91.50% (8.75%)	94.50% (2.76%)	10.52 (7.75)
2-6-2	91.00% (9.68%)	91.35% (1.72%)	9.98 (4.36)
2-8-2	93.00% (8.65%)	90.63% (2.64%)	11.29 (4.64)
2-10-2	93.50% (6.71%)	88.84% (1.54%)	9.51 (1.76)
2-15-2	92.00% (8.34%)	89.08% (2.10%)	9.87 (1.23)
2-20-2	93.00% (8.65%)	88.38% (1.35%)	8.45 (0.76)
2-25-2	93.00% (7.33%)	87.65% (1.07%)	7.49 (0.80)

divided in ten subsets in order to compare the results obtained by the proposed approach with the multifold (10-fold) cross-validation. Each MLP was trained until reaching a MSE of 0.001 or for a maximum number of 5000 epochs and for each trial the generalization was computed as the average of the correctly classified validation patterns for the cross-validation method. The proposed metrics were computed as in the previous experiment and the interval of the network output values inverted using SIVIA is again $[0.8, 1]$.

5.2 Discussion

Prior to a discussion of the results, we need to note the following; lower values of the Metric 1 indicate higher over-training (i.e. lower generalization ability), as the domain of validity of an over-trained network tends to concentrate around the input areas of higher density. On the other hand the values of the Metric 2, which "corrects" the domain of validity of the network, indicate the discrimination of the overlapping areas between different classes, thus contributing to a more accurate evaluation of the generalization. We consider that these two metrics should be taken as a set, and in future work we will be able to propose some aggregate form of them.

Comparing the results of the proposed metrics, as reported in Tables 1 and 2, against the results of the classical cross-validation one may easily notice that the proposed metrics detect and indicate both over-training and generalization of the network models tested, while the results of cross-validation do not confirm this essential theoretical issue.

The proposed Metric 2 evaluates the boxes as they are derived by the branch-and-bound searching performed by SIVIA which might result in rejecting areas that should be more carefully examined. This may be the cause of the high standard deviation observed for this metric in the case of the Fisher-Iris problem. Nevertheless, this metric merits to be further investigated in future work.

The proposed metrics are calculated based on the trained network itself, irrespectively of the data set used for training and validation. So, these metrics may

Table 2. Results for the Fisher-Iris data set (higher values indicate better generalization)

Network model	Cross-validation mean (std)	Metric 1 mean (std)	Metric 2 mean (std)
4-2-3	94.67% (6.13%)	80.42% (10.58%)	2.14 (4.09)
4-3-3	94.00% (6.63%)	79.66% (7.0%)	1.69 (1.28)
4-5-3	93.33% (7.70%)	73.59% (6.08%)	1.15 (1.05)
4-8-3	91.33% (8.92%)	67.67% (6.68%)	1.01 (0.74)
4-10-3	94.67% (10.33%)	66.98% (10.04%)	0.84 (0.75)
4-15-3	92.67% (7.98%)	63.91% (8.49%)	0.51 (0.51)

be used for a given data set in order to compare the performance of two different networks regardless of the way they were trained. Concerning the implementation of the approach, we foresee to take over a suitable implementation of SIVIA for neural network inversion which will perform inversion incrementally, i.e., for each class exclude from the search space the area found to belong to previously examined classes.

A hypothesis that seems to be strongly advocated by the experimental results, is that a suitable combination of the proposed metrics may constitute an effective means for comparing the performance of different network models on the same classification task. However, the proposed metrics should be examined from a mathematically defensible point of view and this constitutes another objective for future work. For the time being we need to note the strong similarity between the domain of validity derived as a level set and the density level sets that can be defined in the input space using clustering or other techniques.

6 Conclusion

In this paper we advanced the results obtained in [1] concerning the quantitative aspects of a neural network's domain of validity. The volume of the area defined by IA-based inversion constitutes a guaranteed quantity for computing the empirical metrics of the network classification performance, mainly, in terms of generalization and comparative evaluation of different network models. The results obtained provide concrete evidence of the potential suggested by the proposed metrics. However, we consider that these empirical metrics need to be validated from a mathematical point of view which will take into account the Bayesian aspects of the neural network classification function. This is one of the objectives of our current work.

Acknowledgments. The authors would like to thank the anonymous reviewers for their comments and suggestions on earlier draft of the manuscript.

References

1. Adam, S.P., Karras, D.A., Magoulas, G.D., Vrahatis, M.N.: Reliable estimation of a neural network's domain of validity through interval analysis based inversion. In: 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–8, July 2015
2. Bourlard, H., Wellekens, C.J.: Links between markov models and multilayer perceptrons. IEEE Trans. Pattern Anal. Mach. Intell. **12**(12), 1167–1178 (1990)
3. Courrieu, P.: Three algorithms for estimating the domain of validity of feedforward neural networks. Neural Netw. **7**(1), 169–174 (1994)
4. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. Wiley, New York (1973)
5. Eberhart, R., Dobbins, R.: Designing neural network explanation facilities using genetic algorithms. In: 1991 IEEE International Joint Conference on Neural Networks, vol. 2, pp. 1758–1763, November 1991
6. Gish, H.: A probabilistic approach to the understanding and training of neural network classifiers. In: International Conference on Acoustics, Speech, and Signal Processing, vol. 3, pp. 1361–1364, April 1990
7. Hampshire II, J.B., Pearlmutter, B.A.: Equivalence proofs for multilayer perceptron classifiers and the Bayesian discriminant function. In: Proceedings of 1990 Connectionist Models Summer School, vol. 1, pp. 159–172, April 1991
8. Hassoun, M.H.: Fundamentals of Artificial Neural Networks. MIT Press, Cambridge (1995)
9. Haykin, S.: Neural Networks a Comprehensive Foundation, 2nd edn. Prentice-Hall, Upper Saddle River (1999)
10. Jaulin, L., Kieffer, M., Didrit, O., Walter, E.: Applied Interval Analysis. With Examples in Parameter and State Estimation, Robust Control and Robotics. Springer, London (2001)
11. Jaulin, L., Walter, E.: Set Inversion Via Interval Analysis for nonlinear bounded-error estimation. Automatica **29**(4), 1053–1064 (1993)
12. Jensen, C., Reed, R., Marks, R., El-Sharkawi, M., Jung, J.B., Miyamoto, R., Anderson, G., Eggen, C.: Inversion of feedforward neural networks: algorithms and applications. Proc. IEEE **87**(9), 1536–1549 (1999)
13. Kearfott, R.B.: Interval computations: introduction, uses, and resources. Euromath Bull. **2**(1), 95–112 (1996)
14. Kindermann, J., Linden, A.: Inversion of neural networks by gradient descent. Parallel Comput. **14**(3), 277–286 (1990)
15. Lippmann, R.P.: Pattern classification using neural networks. IEEE Commun. Mag. **27**(11), 47–50 (1989)
16. Lu, B.L., Kita, H., Nishikawa, Y.: Inverting feedforward neural networks using linear and nonlinear programming. IEEE Trans. Neural Netw. **10**(6), 1271–1290 (1999)
17. Reed, R., Marks, R.: An evolutionary algorithm for function inversion and boundary marking. In: IEEE International Conference on Evolutionary Computation, vol. 2, pp. 794–797, November 1995
18. Richard, M., Lippmann, R.: Neural network classifiers estimate Bayesian a posteriori probabilities. Neural Comput. **3**(4), 461–483 (1991)
19. Rump, S.M.: INTLAB - INTerval LABoratory. In: Csendes, T. (ed.) Developments in Reliable Computing, pp. 77–104. Kluwer Academic, Dordrecht (1999)

20. Saad, E.W., Wunsch II, D.C.: Neural network explanation using inversion. Neural Netw. **20**(1), 78–93 (2007)
21. Theodoridis, S., Pikrakis, A., Koutroumbas, K., Kavouras, D.: Introduction to Pattern Recognition: A MATLAB Approach. Academic Press, Burlington (2010)
22. Thrun, S.B.: Extracting provably correct rules from artificial neural networks. Technical report IAI-TR-93-5, Institut fur Informatik III, Bonn, Germany (1993)
23. Tornil-Sin, S., Puig, V., Escobet, T.: Set computations with subpavings in MATLAB: the SCS toolbox. In: 2010 IEEE International Symposium on Computer-Aided Control System Design (CACSD), pp. 1403–1408, September 2010