Conformal sets in neural network regression*

Radim Demut[1] and Martin Holeňa[2]

[1] Faculty of Nuclear Sciences and Physical Engineering
Czech Technical University in Prague
demut@seznam.cz
[2] Institute of Computer Science
Academy of Sciences of the Czech Republic
martin@cs.cas.cz

Abstract. *This paper is concerned with predictive regions in regression models, especially neural networks. We use the concept of conformal prediction (CP) to construct regions which satisfy given confidence level. Conformal prediction outputs regions, which are automatically valid, but their width and therefore usefulness depends on the used nonconformity measure. A nonconformity measure should tell us how different a given example is with respect to other examples. We define nonconformity measures based on some reliability estimates such as variance of a bagged model or local modeling of prediction error. We also present results of testing CP based on different nonconformity measures showing their usefulness and comparing them to traditional confidence intervals.*

1 Introduction

This paper is concerned with predictive regions for regression models, especially neural networks. We often want to know not only the label y of a new object, but also how accurate the prediction is. Could the real label be very far from our prediction or is our prediction very accurate? It is possible to use traditional confidence intervals to answer this question but they do not work very well with highly nonlinear regression models such as neural networks. We use conformal prediction to solve this problem and construct some accurate and useful prediction regions.

We introduce conformal prediction (CP) in chapter 2. Conformal prediction does not output single label but a set of labels Γ^ε. The size of the prediction set depends on a significance level ε which we want to achieve. Significance level is under some conditions the probability that our prediction lies outside the set. The set is smaller for larger ε. If we have some prediction rule, we will call it simple predictor and we can use it to construct conformal predictor. We introduce transductive conformal predictors where the prediction rule is updated after a new example arrives. But these predictors are not suitable for neural network regression, therefore, we also introduce inductive conformal predictors where the prediction rule is updated only after a given number of new examples has arrived and a calibration set is used.

In order to define a conformal predictor we need a suitable nonconformity measure. A nonconformity measure should tell us how different a given example is with respect to other examples. In chapter 3, we introduce two reliability estimates: variance of a bagged model and local modeling of prediction error. We use these reliability estimates in chapter 4 to define normalized nonconformity measures. Some other reliability estimates could be used, e.g. sensitivity analysis or density based reliability estimate.

In chapter 5, we use CP, based on nonconformity measures defined in chapter 4, on testing data to compare our conformal regions with traditional confidence intervals and with conformal intervals where these traditional confidence intervals are used to construct the nonconformity measure.

2 Conformal prediction

We assume that we have an infinite sequence of pairs

$$(x_1, y_1), (x_2, y_2), \ldots, \tag{1}$$

called *examples*. Each example (x_i, y_i) consists of an *object* x_i and its *label* y_i. The objects are elements of a measurable space $\mathbf{X}$ called the *object space* and the labels are elements of a measurable space $\mathbf{Y}$ called the *label space*. Moreover, we assume that $\mathbf{X}$ is non-empty and that the σ-algebra on $\mathbf{Y}$ is different from $\{\emptyset, \mathbf{Y}\}$. We denote $z_i := (x_i, y_i)$ and we set

$$\mathbf{Z} := \mathbf{X} \times \mathbf{Y} \tag{2}$$

and call $\mathbf{Z}$ the *example space*. Thus the infinite data sequence (**??**) is an element of the measurable space $\mathbf{Z}^\infty$.

Our standard assumption is that the examples are chosen independently from some probability distribution Q on $\mathbf{Z}$, i.e. the infinite data sequence (**??**) is drawn from the power probability distribution Q^∞ on

* This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS12/157/OHK4/2T/14 and the Czech Science Foundation grant 201/08/0802.

$\mathbf{Z}^\infty$. Usually we need only slightly weaker assumption that the infinite data sequence (**??**) is drawn from a distribution P on $\mathbf{Z}^\infty$ that is exchangeable, that means that every $n \in \mathbb{N}$, every permutation π of $\{1,\ldots,n\}$, and every measurable set $E \subseteq \mathbf{Z}^\infty$ fulfill

$$P\{(z_1,z_2,\ldots) \in \mathbf{Z}^\infty : (z_1,\ldots,z_n) \in E\} = P\{(z_1,z_2,\ldots) \in \mathbf{Z}^\infty : (z_{\pi(1)},\ldots,z_{\pi(n)}) \in E\}$$

We denote $\mathbf{Z}^*$ the set of all finite sequences of elements of $\mathbf{Z}$, $\mathbf{Z}^n$ the set of all sequences of elements of $\mathbf{Z}$ of length n. The order in which old examples appear should not make any difference. In order to formalize this point we need the concept of a bag. A *bag* of size $n \in \mathbb{N}$ is a collection of n elements some of which may be identical. To identify a bag we must say what elements it contains and how many times each of these elements is repeated. We write $\backslash z_1,\ldots,z_n/$ for the bag consisting of elements $z_1,\ldots,z_n$, some of which may be identical with each other. We write $\mathbf{Z}^{(n)}$ for the set of all bags of size n of elements of a measurable space $\mathbf{Z}$. We write $\mathbf{Z}^{(*)}$ for the set of all bags of elements of $\mathbf{Z}$.

2.1 Confidence predictors

We assume that at the nth trial we have firstly only the object x_n and only later we get the label y_n. If we want to predict y_n, we need a *simple predictor*

$$D : \mathbf{Z}^* \times \mathbf{X} \to \mathbf{Y} \ . \tag{3}$$

For any sequence of old examples $x_1,y_1,\ldots,x_{n-1}$, $y_{n-1} \in \mathbf{Z}^*$ and any new object x_n, it gives $D(x_1,y_1,\ldots,x_{n-1},y_{n-1},x_n) \in \mathbf{Y}$ as its prediction for the new label y_n.

Instead of merely choosing a single element of $\mathbf{Y}$ as our prediction for y_n, we want to give subsets of $\mathbf{Y}$ large enough that we can be confident that y_n will fall in them, while also giving smaller subsets in which we are less confident. An algorithm that predicts in this sense requires additional input $\varepsilon \in (0,1)$, which we call significance level, the complementary value $1-\varepsilon$ is called confidence level. Given all these inputs

$$x_1,y_1,\ldots,x_{n-1},y_{n-1},x_n,\varepsilon \tag{4}$$

an algorithm Γ that interests us outputs a subset

$$\Gamma^\varepsilon(x_1,y_1,\ldots,x_{n-1},y_{n-1},x_n) \tag{5}$$

of $\mathbf{Y}$. We require this subset to shrink as ε is increased that means it holds

$$\Gamma^{\varepsilon_1}(x_1,y_1,\ldots,x_{n-1},y_{n-1},x_n) \subseteq \Gamma^{\varepsilon_2}(x_1,y_1,\ldots,x_{n-1},y_{n-1},x_n) \tag{6}$$

whenever $\varepsilon_1 \geq \varepsilon_2$.

Formally, a *confidence predictor* is a measurable function

$$\Gamma : \mathbf{Z}^* \times \mathbf{X} \times (0,1) \to 2^{\mathbf{Y}} \tag{7}$$

that satisfies (**??**) for all $n \in \mathbb{N}$, all incomplete data sequences $x_1,y_1,\ldots,x_{n-1},y_{n-1},x_n$ and all significance levels $\varepsilon_1 \geq \varepsilon_2$.

Whether Γ makes an error on the nth trial of the data sequence $\omega = (x_1,y_1,x_2,y_2,\ldots)$ at significance level ε can be represented by a number that is one in case of an error and zero in case of no error

$$\mathrm{err}_n^\varepsilon(\Gamma,\omega) := \begin{cases} 1 \text{ if } y_n \notin \Gamma^\varepsilon(x_1,y_1,\ldots, \\ \quad x_{n-1},y_{n-1},x_n) \ , \\ 0 \text{ otherwise} \ , \end{cases} \tag{8}$$

and the number of errors during the first n trials is

$$\mathrm{Err}_n^\varepsilon(\Gamma,\omega) := \sum_{i=1}^n \mathrm{err}_i^\varepsilon(\Gamma,\omega) \ . \tag{9}$$

If ω is drawn from an exchangeable probability distribution P, the number $\mathrm{err}_n^\varepsilon(\Gamma,\omega)$ is the realized value of a random variable, which we may designate $\mathrm{err}_n^\varepsilon(\Gamma,P)$. We say that confidence predictor Γ is *conservatively valid* if for any exchangeable probability distribution P on $\mathbf{Z}^\infty$ there exist two families

$$(\xi_n^{(\varepsilon)} : \varepsilon \in (0,1), n = 1,2,\ldots) \tag{10}$$

and

$$(\eta_n^{(\varepsilon)} : \varepsilon \in (0,1), n = 1,2,\ldots) \tag{11}$$

of $\{0,1\}$-valued variables such that

- for a fixed $\varepsilon, \xi_1^{(\varepsilon)}, \xi_2^{(\varepsilon)}, \ldots$ is a sequence of independent Bernoulli random variables with parameter ε;
- for all n and ε, $\eta_n^{(\varepsilon)} \leq \xi_n^{(\varepsilon)}$;
- the joint distribution of $\mathrm{err}_n^\varepsilon(\Gamma,P)$, $\varepsilon \in (0,1)$, $n = 1,2,\ldots$, coincides with the joint distribution of $\eta_n^{(\varepsilon)}$, $\varepsilon \in (0,1)$, $n = 1,2,\ldots$.

2.2 Transductive conformal predictors

A *nonconformity measure* is a measurable mapping

$$A : \mathbf{Z}^{(*)} \times \mathbf{Z} \to \overline{\mathbb{R}} \ . \tag{12}$$

To each possible bag of old examples and each possible new example, A assigns a numerical score indicating how different the new example is from the old ones. It is sometimes convenient to consider separately how a nonconformity measure deals with bags of different sizes. If A is a nonconformity measure, for each $n = 1,2,\ldots$ we define a function

$$A_n : \mathbf{Z}^{(n-1)} \times \mathbf{Z} \to \overline{\mathbb{R}} \tag{13}$$

as the restriction of A to $\mathbf{Z}^{(n-1)} \times \mathbf{Z}$. The sequence $(A_n : n \in \mathbb{N})$, which we abbreviate to (A_n) will also be called a *nonconformity measure*.

Given a nonconformity measure (A_n) and a bag $\backslash z_1, \dots, z_n /$ we can compute the nonconformity score

$$\alpha_i := A_n(\backslash z_1, \dots, z_{i-1}, z_{i+1}, \dots z_n /, z_i) \tag{14}$$

for each example z_i in the bag. Because a nonconformity measure (A_n) may be scaled however we like, the numerical value of α_i does not, by itself, tell us how unusual (A_n) finds z_i to be. For that we define *p-value* for z_i as

$$p := \frac{|\{j = 1, \dots, n : \alpha_j \geq \alpha_i\}|}{n} \ . \tag{15}$$

We define *transductive conformal predictor* (TCP) by a nonconformity measure (A_n) as a confidence predictor Γ obtained by setting

$$\Gamma^\varepsilon(x_1, y_1, \dots, x_{n-1}, y_{n-1}, x_n) \tag{16}$$

equal to the set of all labels $y \in \mathbf{Y}$ such that

$$\frac{|\{i = 1, \dots, n : \alpha_i(y) \geq \alpha_n(y)\}|}{n} > \varepsilon \ , \tag{17}$$

where

$$\begin{aligned} \alpha_i(y) := A_n(&\backslash (x_1, y_1), \dots, (x_{i-1}, y_{i-1}), \\ &(x_{i+1}, y_{i+1}), \dots, (x_{n-1}, y_{n-1}), (x_n, y) /, \\ &(x_i, y_i)) \ , \quad \forall i = 1, \dots, n-1 \ , \\ \alpha_n(y) := A_n(&\backslash (x_1, y_1), \dots, (x_{n-1}, y_{n-1}) /, (x_n, y)) \ . \end{aligned}$$

We now remind an important property of TCP. The proof of the following theorem can be found in [**?**].

Theorem 1. *All conformal predictors are conservatively valid.*

If we are given a simple predictor (**??**) whose output does not depend on the order in which the old examples are presented, than the simple predictor D defines a prediction rule $D_{\backslash z_1, \dots, z_n /} : \mathbf{X} \to \mathbf{Y}$ by the formula

$$D_{\backslash z_1, \dots, z_n /}(x) := D(z_1, \dots, z_n, x) \ . \tag{18}$$

A natural measure of nonconformity of z_i is the deviation of the predicted label

$$\widehat{y}_i := D_{\backslash z_1, \dots, z_n /}(x_i) \tag{19}$$

from the true label y_i. We can also use the deleted prediction defined as

$$\widehat{y}_{(i)} := D_{\backslash z_1, \dots, z_{i-1}, z_{i+1}, \dots, z_n /}(x_i) \ . \tag{20}$$

A *discrepancy measure* is a measurable function

$$\Delta : \mathbf{Y} \times \mathbf{Y} \to \mathbb{R} \ . \tag{21}$$

Given a simple predictor D and a discrepancy measure Δ we define functions (A_n) as follows: for any $((x_1, y_1), \dots, (x_n, y_n)) \in \mathbf{Z}^*$, the values

$$\begin{aligned} \alpha_i = A_n(&\backslash (x_1, y_1), \dots, (x_{i-1}, y_{i-1}), \\ &(x_{i+1}, y_{i+1}), \dots, (x_n, y_n) /, (x_i, y_i)) \end{aligned} \tag{22}$$

are defined according to (**??**) and (**??**) by the formula

$$\alpha_i := \Delta(y_i, D_{\backslash z_1, \dots, z_n /}(x_i)) \tag{23}$$

and the formula

$$\alpha_i := \Delta(y_i, D_{\backslash z_1, \dots, z_{i-1}, z_{i+1}, \dots, z_n /}(x_i)) \ , \tag{24}$$

respectively. It can be easily checked that in both cases (A_n) form a nonconformity measure.

2.3 Inductive conformal predictors

In TCP, we need to compute the p-value (**??**) for all labels $y \in \mathbf{Y}$ to determine the set Γ^ε. In the case of regression, we have $\mathbf{Y} = \mathbb{R}$ and it is not possible to try each $y \in \mathbf{Y}$. Sometimes it is possible to generally solve equations $\alpha_i(y) \geq \alpha_n(y)$ with respect to y, and therefore determine the set Γ^ε. But if we use neural networks as simple predictor, we do not know the general form of the simple predictor, i.e. we do not know a functional relationship between the training set and the trained network, because random influences enter the training algorithm. Hence, we cannot solve the equations $\alpha_i(y) \geq \alpha_n(y)$, and it is not possible to use TCP. Even if the equations can be solved, it can be very computationally inefficient.

To avoid this problem we can use *inductive conformal predictor* (ICP). To define ICP from a nonconformity measure (A_n) we fix a finite or infinite increasing sequence of positive integers $m_1, m_2, \dots$ (called update trials). If the sequence is finite we add one more member equal to infinity at the end of the sequence. We need more than m_1 training examples. Then we find k such that $m_k < n \leq m_{k+1}$. The ICP is determined by (A_n) and the sequence $m_1, m_2, \dots$ of update trials is defined to be the confidence predictor Γ such that the prediction set

$$\Gamma^\varepsilon(x_1, y_1, \dots, x_{n-1}, y_{n-1}, x_n) \tag{25}$$

is equal to the set of all labels $y \in \mathbf{Y}$ such that

$$\frac{|\{j = m_k + 1, \dots, n : \alpha_j \geq \alpha_n(y)\}|}{n - m_k} > \varepsilon \ , \tag{26}$$

where the nonconformity scores are defined by

$$\alpha_j := A_{m_k+1}(\backslash(x_1,y_1),\ldots,(x_{m_k},y_{m_k})/,(x_j,y_j)) \ , \quad \text{for } j = m_k+1,\ldots,n-1 \tag{27}$$

$$\alpha_n := A_{m_k+1}(\backslash(x_1,y_1),\ldots,(x_{m_k},y_{m_k})/,(x_n,y)) \ . \tag{28}$$

The proof of the following theorem can be found in [**?**].

Theorem 2. *All ICPs are conservatively valid.*

For ICP combining (**??**) with (**??**) and (**??**) we get

$$A_{l+1}(\ \backslash(x_1,y_1),\ldots,(x_l,y_l)/,(x,y)) = \Delta(y, D_{\backslash(x_1,y_1),\ldots,(x_l,y_l),(x,y)/}(x)) \tag{29}$$

and

$$A_{l+1}(\ \backslash(x_1,y_1),\ldots,(x_l,y_l)/,(x,y)) = \Delta(y, D_{\backslash(x_1,y_1),\ldots,(x_l,y_l)/}(x)) \ , \tag{30}$$

respectively. When we define A by (**??**), we can see that the ICP requires recomputing the prediction rule only at the update trials $m_1, m_2, \ldots$. We will use the simplest case, where there is only one update trial m_1, therefore, we compute the prediction rule only once.

3 Reliability estimates

In this chapter we are interested in different approaches to estimate the reliability of individual predictions in regression.

3.1 Variance of a bagged model

We are given a learning set $L = \{(x_1,y_1),\ldots,(x_n,y_n)\}$ and take repeated bootstrap samples $L^{(i)}$, $i = 1,\ldots,m$ of size d from the learning set, i.e. for $i = 1,\ldots,m$ we randomly choose d points from the original learning set L with the return and put them in $L^{(i)}$. The number of points d can be chosen arbitrary. We induce a new model on each of these bootstrap samples $L^{(i)}$. Each of the models yields a prediction $K_i(x)$, $i = 1,\ldots,m$ for a considered input x. The label of the example x is predicted by averaging the individual predictions

$$K(x) := \frac{\sum_{i=1}^m K_i(x)}{m} \ . \tag{31}$$

We call this procedure *bootstrap aggregating* or *bagging*. The reliability estimate of a bagged model is defined as the prediction variance

$$\mathrm{BAGV}(x) := \frac{1}{m}\sum_{i=1}^m (K_i(x) - K(x))^2 \ . \tag{32}$$

3.2 Local modeling of prediction error

We find k nearest neighbors of an unlabeled example x in the training set, therefore, we have a set $N = \{(x_1,y_1),\ldots,(x_k,y_k)\}$ of nearest neighbors. We define the estimate denoted CNK for an unlabeled example x as the difference between the average label of the nearest neighbors and the example's prediction y (using the model that was generated on all learning examples)

$$\mathrm{CNK}(x) := \frac{\sum_{i=1}^k y_i}{k} - y \ . \tag{33}$$

The dependence on x on the right hand side of the previous equation is implicit, but both the prediction y and the selection of nearest neighbors depends on x.

4 Normalized nonconformity measures

We will follow a similar approach as is used in the article [**?**], but we will incorporate the reliability estimates from previous chapter and use it for neural network regression.

We will use ICP with only one update trial. Let us have training set of size l, where $l > m_1$. We will split it into two sets, the proper training set T of size m_1 (we will further write m) and the calibration set C of size $q = l - m$. We will use the proper training set for creating the simple predictor $D_{\backslash(x_1,y_1),\ldots,(x_m,y_m)/}$. The calibration set is used for calculating the p-value of new test examples. It is good to first normalize the data (i.e. subtract the mean and divide data by sample variance).

We will denote r_i any of the previously defined reliability estimates in the point x_i with given simple predictor D. We compute r_i for all points in the calibration set and define R_i for any given point x_i as

$$R_i := \frac{r_i}{\mathrm{median}\{r_j : r_j \in C\}} \ . \tag{34}$$

We define a discrepancy measure (**??**) as

$$\Delta(y_1,y_2) := \left|\frac{y_1 - y_2}{\gamma + R_i}\right| \ , \tag{35}$$

where parameter $\gamma \geq 0$ controls the sensitivity to changes of R_i. Then, we get the nonconformity score

$$\alpha_i(y) = \left|\frac{y - \hat{y}_i}{\gamma + R_i}\right| \ . \tag{36}$$

We sort nonconformity scores of the calibration examples in descending order

$$\alpha_{(m+1)} \geq \ldots \geq \alpha_{(m+q)} \ , \tag{37}$$

and denote

$$s = \lfloor \varepsilon(q+1) \rfloor \ . \tag{38}$$

Proposition 1. *The prediction set Γ^ε of the new test example x_{l+g} (where x_{l+g} is from the infinite sequence (??)) given the nonconformity score (??) is equal to the interval*

$$\langle \hat{y}_{l+g} - \alpha_{(m+s)}(\gamma + R_{l+g}), \hat{y}_{l+g} + \alpha_{(m+s)}(\gamma + R_{l+g}) \rangle \ . \tag{39}$$

Proof. To compute the prediction set Γ^ε of the new test example x_{l+g} we need to find all $y \in \mathbf{Y}$ such that for the p-value it holds

$$\begin{aligned} & p(y) = \\ & \frac{|\{i = m+1, \ldots, m+q, l+g : \alpha_i \geq \alpha_{l+g}(y)\}|}{q+1} \\ & > \varepsilon \ . \end{aligned} \tag{40}$$

We multiply the inequality by $q+1$ and then it is equivalent to

$$|\{i = m+1, \ldots, m+q, l+g : \alpha_i \geq \alpha_{l+g}(y)\}| > \lfloor \varepsilon(q+1) \rfloor \tag{41}$$

and this inequality holds if and only if

$$\alpha_{(m+s)} \geq \alpha_{l+g}(y) = \left| \frac{y - \hat{y}_{l+g}}{\gamma + R_{l+g}} \right| \ . \tag{42}$$

From (??) follows the assertion of the proposition.

5 Simulation

We carried out a simulation to test the normalized nonconformity measures based on different reliability estimates. We used neural networks with radial basis functions (RBF networks) as our regression models with Gaussian used as the basis function. Therefore, the output of the RBF network $f : \mathbb{R}^n \to \mathbb{R}$ has the form

$$f(\mathbf{x}) = \sum_{i=1}^{N} \pi_i \exp\left\{-\beta_i ||\mathbf{x} - \mathbf{c_i}||^2\right\} \ , \tag{43}$$

where N is the number of neurons in the hidden layer, $\mathbf{c_i}$ is the center vector for neuron i, β_i determines the width of the ith neuron and π_i are the weights of the linear output neuron. RBF networks are universal approximators on a compact subset of $\mathbb{R}^n$. This means that a RBF network with enough hidden neurons can approximate any continuous function with arbitrary precision.

We used a benchmark function similar to some empirical functions encountered in chemistry to carry out our experiment. This function was introduced in [?]. The value of this function ϑ in the point $(x_1, x_2, x_3, x_4, x_5)$ can be expressed as

$$\begin{aligned} \vartheta(x_1, x_2, x_3, x_4, x_5) &= -A(x_1, x_2) \\ & -B(x_2, x_3)C(x_3, x_4, x_5) \ , \end{aligned} \tag{44}$$

where

$$\begin{aligned} A(x_1, x_2) &= 0.6g(x_1 - 0.35, x_2 - 0.35) \\ &\quad +0.75g(x_1 - 0.1, x_2 - 0.1) \\ &\quad +g(x_1 - 0.35, x_2 - 0.1) \\ B(x_2, x_3) &= 0.4g(x_2 - 0.1, x_3 - 0.3) \\ C(x_3, x_4, x_5) &= 5 + 25[1 - \{1 + (x_3 - 0.3)^2 \\ &\quad +(x_4 - 0.15)^2 + (x_5 - 0.1)^2\}^{1/2}] \\ g(a, b) &= 100 - \sqrt{(100a)^2 + (100b)^2} \\ &\quad +50 \frac{\sin\sqrt{(100a)^2 + (100b)^2}}{\sqrt{(100a)^2 + (100b)^2 + (0.01)^2}} \ . \end{aligned}$$

Moreover, the input vectors must satisfy following conditions

$$\sum_{i=1}^{5} x_i = 1 \quad \text{and} \quad x_i \in [0, 1], \text{for } i = 1, \ldots, 5 \ . \tag{45}$$

We repeated the following procedure five times for region with significance level 0.1 and five times for region with significance level 0.05.

- Randomly generate 600 points satisfying the conditions (??).
- Compute the function values of function ϑ in these points.
- Normalize data (i.e. subtract the mean and divide data by sample variance)
- Split this set of points into a training set of 500 points and a testing set of 100 points.
- Split the training set into a proper training set of 401 points and a calibration set of 99 points (then, we divide the p-value in (??) by 100).
- Split the proper training set on training set for fitting the RBF network and the validation set. Fit the RBF network with $1, 2, 3, 4$ and 5 hidden neurons ten times using the Matlab function lsqcurvefit.
- Choose the RBF network with the smallest error on the validation set for each number of hidden neurons.
- Compute the prediction sets for each of the 100 testing points for each number of hidden neurons.
- Transform data and predictive regions back to the original size (i.e. multiply by the original sample variance and add the original mean)
- Determine if the original point lies in our prediction sets.

The initial values of parameters π_i were set as mean of the response vector, initial values of β_i were set as the mean of the standard deviation of the components of training data points. The centers $\mathbf{c_i}$ were set randomly.

We also computed confidence intervals using Matlab function nlpredci (denoted Conf Int). The Jacobian can be computed exactly, because the form of the RBF network is known and differentiable. Therefore, we supply the function nlpredci with this Jacobian. We also use the width of this interval as another reliability estimate for our normalized nonconformity measure.

We compare normalized nonconformity measures based on the following reliability estimates: the local modeling of prediction errors using nearest neighbors (CNK), the variance of a bagged model (BAGV) and the width of confidence intervals (CONF).

The variance of a bagged model was computed for number of different models $m = 10$ and the bootstrap samples were as big as the original sample.

The CNK estimates were computed for number of neighbors $k = 2, 5, 10$.

We present the results of testing CP based on different nonconformity measures in Figures **??**, **??** and **??**. There is a boxplot of all labels in Figure **??** to compare the range of all labels with the width of different predictive regions. Figures **??** and **??** show boxplots of the width of prediction regions for significance levels $\varepsilon = 0.1$ and $\varepsilon = 0.05$, respectively. It is not only interesting whether the intervals are small enough, but they should also be valid. The percentage of labels inside the predictive regions are in Tables **??** and **??** for significance levels $\varepsilon = 0.1$ and $\varepsilon = 0.05$, respectively.

The results for traditional confidence intervals computed by Matlab function nlpredci are not shown in the figures, because these results are very different from the others. The median width for these intervals lies between 10^{10} and 10^{14} for all counts of neurons. This is probably because of the highly nonlinear character of neural nets, while nlpredci is based on linearization. Moreover, during the computation of these intervals a Jacobian matrix must be inverted but this matrix was very often ill conditioned, therefore, the results for confidence intervals are not too reliable.

Despite what was said in the previous paragraph, the predictive regions based on the width of confidence intervals produce sensible results. But these prediction regions show highest inconsistency between different neuron counts and have highest number of very large intervals. These regions produce sometimes very good results, but they are probably very dependent on the actual fit of the neural network and their results are not as consistent as the results of the other methods. However, we can see in Tables **??** and **??** that these intervals are valid as the percentage of labels inside predictive regions is always slightly higher than the confidence level.

Results for predictive regions based on the local modeling of prediction errors depend a little bit on the count of nearest neighbors. These intervals are valid for all numbers of neighbors, but the tightest intervals were achieved for two neighbors. The difference between using five or ten neighbors is not too big but lower number of neighbors works better in our model. This is probably caused by our data and it seems that only a few neighbors are relevant to our prediction. These regions are also the easiest and fastest to compute.

The best results among all predictive regions are achieved by those based on a variance of a bagged model. These regions are the tightest of all tested and they do not vary as much as those based on confidence intervals. These regions also maintain the validity. The drawback of these regions is that we need to fit a lot of additional models which takes a lot of time in the case of neural network regression. But if time and computational efficiency is not a problem then this method produces best regions.

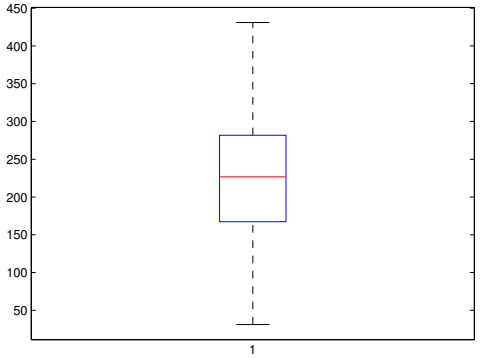

Fig. 1. Boxplot of all labels.

Neurons	CNK2	CNK5	CNK10	BAGV	CONF
2	91.0	91.2	90.0	91.4	92.4
3	92.6	92.4	92.6	94.0	93.6
4	92.2	90.4	90.0	90.4	90.2
5	94.6	92.6	90.0	90.2	90.8
6	92.8	89.8	91.8	91.6	91.8

Table 1. Percentage of labels inside predictive regions for $\varepsilon = 0.1$.

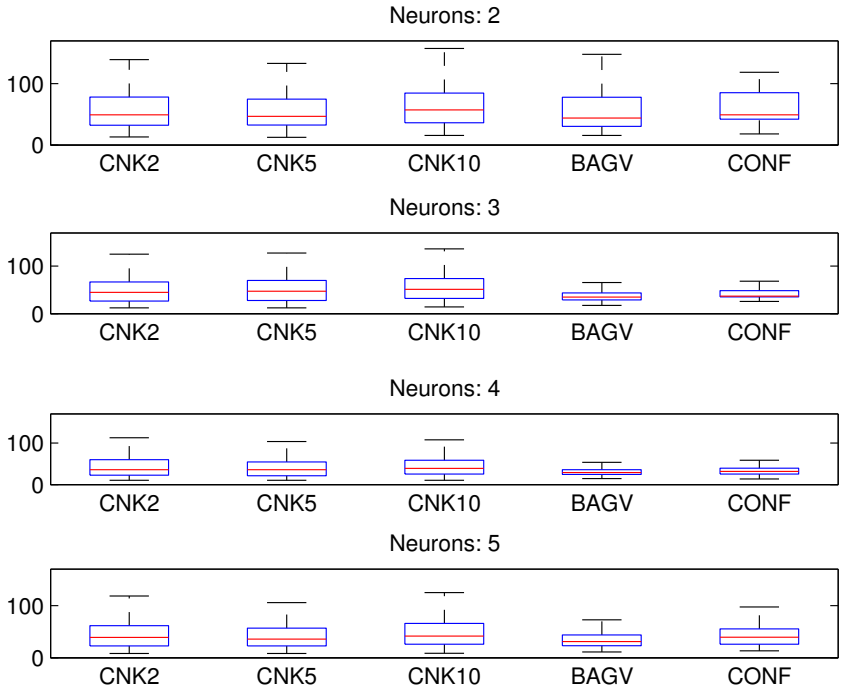

Fig. 2. Interval widths for $\varepsilon = 0.1$.

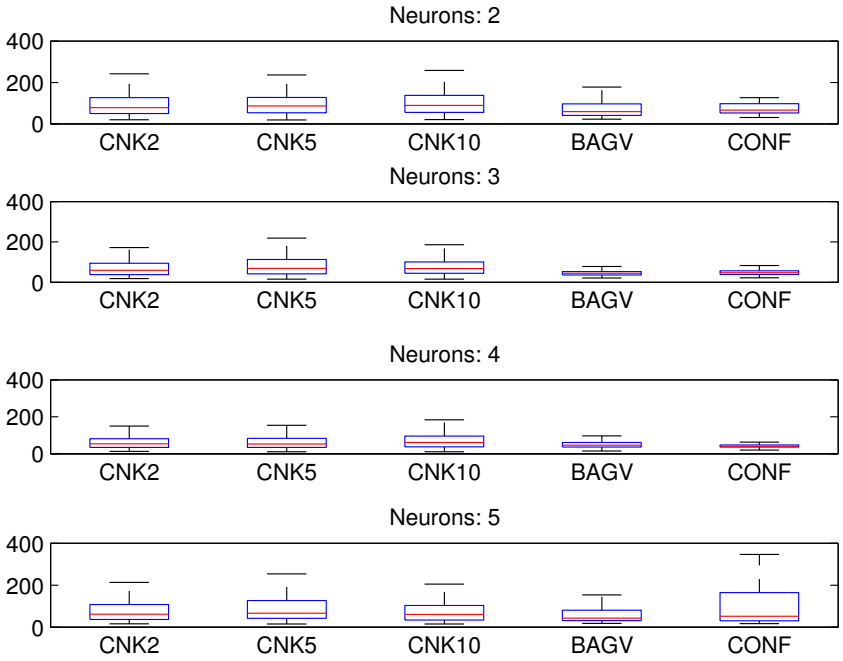

Fig. 3. Interval widths for $\varepsilon = 0.05$.

Neurons	CNK2	CNK5	CNK10	BAGV	CONF
2	95.8	96.8	96.8	96.2	95.6
3	97.2	96.8	96.8	97.8	95.6
4	96.2	96.4	96.6	97.4	97.0
5	97.2	97.6	96.4	96.4	97.6
6	97.2	98.0	96.4	97.4	96.8

Table 2. Percentage of labels inside predictive regions for $\varepsilon = 0.05$.

6 Conclusion

We presented several methods for computing predictive regions in neural network regressions. These methods are based on the inductive conformal prediction with novel nonconformity measures proposed in this paper. Those measures use reliability estimates to determine how different a given example is with respect to other examples. We compared our new predictive regions with traditional confidence intervals on testing data. The confidence intervals did not perform very well, the intervals were too large, it was probably caused by the high nonlinearity of radial basis neural networks. Predictive regions which used the width of confidence intervals as the nonconformity measure gave much better results. But those results were not as consistent as the results of the other methods. Predictive regions based on the local modeling of prediction errors gave us good results and the computation of the regions was very fast. A smaller number of neighbors gave better results for these regions. The best results were achieved by the regions based on the variance of a bagged model. The only drawback of this method is that a lot of models must be fitted and it is, therefore, computationally very inefficient.

References

1. Z. Bosnic, Kononenko, I.: *Comparison of approaches for estimating reliability of individual regression predictions.* Data & Knowledge Engineering, 2008, 504–516.
2. A. Gammerman, G. Shafer, V. Vovk: *Algorithmic learning in a random world.* Springer Science+Business Media, 2005.
3. E. Uusipaikka: *Confidence intervals in generalized regression models.* Chapman & Hall, 2009.
4. H. Papadopoulos, V. Vovk, A. Gammerman: *Regression conformal prediction with nearest neighbours.* Journal of Artificial Intelligence Research 40, 2011, 815–840.
5. S. Valero, E. Argente, et al.: *DoE framework for catalyst development based on soft computing techniques.* Computers and Chemical Engineering 33(1), 2009, 225–238.