Regression Neural Networks with a Highly Robust Loss Function

Jan Kalina and Petra Vidnerová

Abstract Artificial neural networks represent an important class of methods for fitting nonlinear regression to data with an unknown regression function. However, usual ways of training of the most common types of neural networks applied to nonlinear regression tasks suffer from the presence of outlying measurements (outliers) in the data. So far, only a few robust alternatives for training common forms of neural networks have been proposed. In this work, we robustify two common types of neural networks by considering robust versions of their loss functions, which have turned out to be successful in linear regression. Particularly, we extend the idea of using the loss of the least trimmed squares estimator to radial basis function networks. We also propose multilayer perceptrons and radial basis function networks based on the loss of the least weighted squares estimator. The performance of these novel methods is compared with that of standard neural networks on 4 datasets. The results bring arguments in favor of the novel robust approach based on the least weighted squares estimator with trimmed linear weights in terms of yielding the smallest robust prediction error in a variety of situations. Robust neural networks are even able to outperform the prediction ability of support vector regression.

Keywords Nonlinear regression · Neural networks · Robustness

1 Introduction

Nonlinear regression modeling, i.e. estimating (smoothing, fitting) a continuous response variable based on a set of regressors (features, independent variables) plays a crucial role in the analysis of real data in a tremendous variety of applications. An important task of regression modeling is also to predict a future development of the

J. Kalina (✉) · P. Vidnerová
The Czech Academy of Sciences, Institute of Computer Science,
Pod Vodárenskou věží 2, 182 07 Praha 8, Czech Republic
e-mail: kalina@cs.cas.cz

P. Vidnerová
e-mail: petra@cs.cas.cz

M. Maciak et al. (eds.), *Analytical Methods in Statistics*, Springer Proceedings
in Mathematics & Statistics 329, https://doi.org/10.1007/978-3-030-48814-7_2

response [6]. In practical applications, the nonlinear regression function is not known and is not assumed to be of any specific form. Recently, there is an increasing trend in applying machine learning methods to nonlinear regression modeling. In this paper, multilayer perceptrons (MLPs) and radial basis function (RBF) networks, i.e. two very important classes of feedforward artificial neural networks [10], are considered for the nonlinear regression task.

Real data across various disciplines, e.g. in numerous regression tasks of biomedicine, economics, engineering etc., are typically contaminated by the presence of outlying measurements (outliers). In some applications (e.g. in measurements of molecular genetic and metabolomic biomarkers [14]), outliers appear unavoidably, because severe measurement errors are immanent to the measurement technology. So far, most available applications of MLPs and RBF networks to regression tasks have not paid sufficient attention to the presence and influence of outliers; both these networks however implicitly assume the observed data not to be contaminated by outliers [2, 26]. Therefore, it is highly desirable to consider alternative robust approaches to training of MLPs and RBF networks. One direction of the robustification is based on an intrinsically performed detection of outliers [1]. Another direction for a possible robustification is inspired by the very rich experience of robust statistics with data contamination by outliers or anomalies (see [12]); this approach represents the interest of the current paper.

While there are some robust approaches to training neural networks available, they are mostly tailor-made the classification task; see ([17], p. 54) for discussion. Let us mention at least a few available robust approaches for the regression task. Compositions of sigmoidal activation functions were considered to robustify the performance for a rather specific task in [18] to estimate a response which is almost constant over relatively large intervals. If subtractive clustering (SC) is used for an automatic recommendation of the center vectors, a robustified loss function may be subsequently used [26]; still, the popular SC approach remains vulnerable to outliers and consecutive steps of the training cannot improve this. A recent approach to outlier detection for regression RBF networks was developed in [17], which is denoted as generalized edited nearest neighbor (ENN) algorithm; this was also combined with robust versions of the activation function. Robust loss functions based on least trimmed squares or least trimmed absolute values estimators were investigated in [24, 25], where they outperformed standard training approaches on contaminated data. We do not agree with the formulas for partial derivatives of the loss function published in [24], but this may not influence the results presented there, as practical computations typically exploit numerical approximations of derivatives (not relying on theoretical expressions). Nevertheless, even the extensive numerical computations in [25] do not compare robust neural networks with the (sophisticated and powerful) support vector regression.

The idea to apply a robust loss function in neural networks will be extended in the current paper by means of the least weighted squares estimator, which represents a natural generalization of the least trimmed squares and turns out to be a perspective and (possibly) highly robust tool for estimating parameters in linear regression. Section 2 recalls the least trimmed squares and least weighted squares estimators

of parameters in linear regression and in the location model. Section 3 uses these estimators to propose novel robust versions of MLPs and RBF networks. Numerical examples presented in Sect. 4 illustrate the performance of the novel robust neural networks. Finally, Sect. 5 concludes the paper.

2 Highly Robust Estimation in Linear Models

This section recalls two (possibly highly) robust implicitly weighted estimators of parameters of the linear regression model (including the location model as a special case), namely the least trimmed squares and least weighted squares estimators. Highly robust estimators are defined as those, which attain a high value of the breakdown point; this measure of robustness of a statistical estimator of an unknown parameter represents a fundamental concept of robust statistics [12]. Formally, the finite-sample breakdown point evaluates the minimal fraction of data that can drive an estimator beyond all bounds when set to arbitrary values.

The standard linear regression model has the form

$$Y_i = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_p X_{ip} + e_i, \quad i = 1, \ldots, n, \tag{1}$$

with a continuous response $Y_1, \ldots, Y_n$ explained by the total number of p regressors, and independent and identically distributed (not necessarily Gaussian) random errors $e_1, \ldots, e_n$.

The least trimmed squares (LTS) estimator [22, 23] of β represents a popular robust regression estimator with a high breakdown point. Consistency of the LTS and other properties were derived in [27]. The user must select the value of a trimming constant h ($n/2 \leq h < n$). We will denote residuals corresponding to a particular $b = (b_0, \ldots, b_p)^T \in \mathbb{R}^{p+1}$ as

$$u_i(b) = Y_i - b_0 - b_1 X_{i1} - \cdots - b_p X_{ip} \tag{2}$$

and order statistics of their squares as

$$u^2_{(1)}(b) \leq \cdots \leq u^2_{(n)}(b). \tag{3}$$

The LTS estimator, formally obtained as

$$\arg\min_{b \in \mathbb{R}^{p+1}} \frac{1}{n} \sum_{i=1}^{h} u^2_{(i)}(b), \tag{4}$$

may attain a high robustness but cannot achieve a high efficiency. We may consider the LTS as an implicitly weighted estimator, namely as a special case of the least weighted squares with weights equal only to 0 or 1.

The least weighted squares (LWS) estimator (see e.g. [28]) for the model (1) represents a flexible natural extension of the LTS. The LWS estimator motivated by the idea to down-weight potential outliers based on ranks of residuals however remains much less known compared to the LTS. The LWS estimator may achieve a high breakdown point (with properly selected weights) and is robust to heteroscedasticity [28]. Its primary attention is focused on estimating β and not on outlier detection. The LWS estimator with given magnitudes of weights $w_1, \ldots, w_n$ is defined as

$$\mathbf{b}^{LWS} = (b_0^{LWS}, \ldots, b_p^{LWS})^T = \underset{b \in \mathbb{R}^{p+1}}{\arg\min} \sum_{k=1}^{n} w_k u_{(k)}^2(b). \tag{5}$$

The efficiency of the LWS is able to exceed the low efficiency of the LTS; if data-dependent adaptive weights of [4] are used, the estimator asymptotically attains the full efficiency of the least squares. The LWS estimator was successful in a variety of recent applications including denoising gene expression measurements acquired by the microarray technology [14] or image analysis based on landmarks measured within facial images [13]. There has been a good experience with implicit weighting also for multivariate robust estimation; the multivariate analogy of the LWS is the minimum weighted covariance determinant (MWCD) estimator proposed in [21].

The location model represent an important special case of (1) in the form

$$Y_i = \mu + e_i \quad \text{for} \quad i = 1, \ldots, n, \tag{6}$$

where $\mu \in \mathbb{R}$ represents a parameter of location (shift). In (6), the LWS estimator inherits the appealing properties of the LWS from (1). The performance of the LWS on real data in (6) was revealed as successful e.g. in the image analysis applications of [13], where the LWS estimator in (6) was also proven to correspond to the estimator with the smallest weighted variance. This allows a very efficient computation of the LWS in (6).

3 Robust Neural Networks with Implicitly Weighted Loss Functions

We consider the regression model

$$Y_i = f(X_i) + e_i, \quad i = 1, \ldots, n, \tag{7}$$

with an unknown nonlinear function f, where $Y_1, \ldots, Y_n$ are values of the response and $X_i \in \mathbb{R}^p$ (with $p \geq 1$) is a vector of regressors corresponding to the i-th observation. This is a nonlinear regression setup with a univariate continuous response $Y_1, \ldots, Y_n$, which is explained by means of p regressors. A novel robust tool for

neural networks is proposed in this section, namely an MLP or an RBF network based on the loss function of the LWS estimator.

MLPs, which represent a very popular type of artificial neural networks, contain an input layer, one or more hidden layers with a fixed number of neurons, and an output layer. As we use the most standard form of multilayer perceptrons, we will not present their detailed model, as it can be found in numerous monographs (see e.g. [7, 9]). For a particular multilayer perceptron (with a selected architecture), let the fitted value of the response for the i-th measurement (i.e. estimate of Y_i) be denoted by $\hat{Y}_i$ for each $i = 1, \ldots, n$.

Let us start by describing the training of a standard MLP in a symbolic (general but very simplified) way in Algorithm 1. There, we denote the whole (say m-dimensional) vector of all parameters of a given MLP with a specified architecture as $\theta \in \mathbb{R}^m$. Denoting the estimated version of f obtained by the MLP as $\hat{f}$, we may denote the vector of fitted values of Y by $\hat{Y} = \hat{f}(\hat{\theta})$ and the vector of residuals, which depend on $\hat{f}$, as $u = Y - \hat{Y}$. Concerning the stopping rule in Algorithm 1, our computations use a default version implemented in [3]. Algorithm 1 is formulated in such a way that it remains valid also for a robust version of an MLP, as it considers a general loss function.

Algorithm 1 MLP in the nonlinear regression model (1) with a selected (standard or robust) loss function ℓ

Input: $X_1, \ldots, X_n$, where $X_i \in \mathbb{R}^p$ for each $i = 1, \ldots, n$
Input: $Y_1, \ldots, Y_n$, where $Y_i \in \mathbb{R}$ for each $i = 1, \ldots, n$
Input: A chosen loss function ℓ
Output: A fitted MLP based on minimizing a given loss ℓ
Choose $\hat{\theta}_0 \in \mathbb{R}^m$ as an initial estimate of θ
$i := 0$
repeat
$u^i = (u_1^i, \ldots, u_n^i) := Y - f(\hat{\theta}_i)$
$i := i + 1$
$\hat{\theta}_i := \arg\min \ell(u_1^{i-1}, \ldots, u_n^{i-1})$ (where the optimization over estimates of θ is solved by a stochastic gradient method)
until a certain stopping rule is fulfilled

The most common way of training MLPs minimizes the sum of prediction errors in the form

$$\ell = \ell(u_1, \ldots, u_n) := \min \sum_{i=1}^{n} u_i^2. \tag{8}$$

It corresponds to the least squares estimation in a location model. It is now natural to replace this quadratic loss function by one of available robust alternatives (again for the location model). We consider a method of [24] denoted here as LTS-MLP; for a fixed h, it is defined by replacing (8) in the form

$$\ell := \sum_{i=1}^{h} u_{(i)}^2. \tag{9}$$

We define a new version of MLP dentoted as LWS-MLP by choosing ℓ in the form

$$\ell := \sum_{i=1}^{n} w_i u_{(i)}^2 \tag{10}$$

for selected magnitudes of weights $w_1, \ldots, w_n$. We always consider the natural standardization to $\sum_{i=1}^{n} w_i = 1$. We consider three particular choices, namely the LWSa-MLP with linear weights

$$w_i = \frac{2(n+1-i)}{n(n+1)}, \quad i = 1, \ldots, n, \tag{11}$$

LWSb-MLP with trimmed linear weights

$$w_i = \frac{h-i+1}{h} \mathbb{1}[i \leq h], \quad i = 1, \ldots, n, \tag{12}$$

where we consider $h = \lfloor 3n/4 \rfloor$ and $\lceil x \rceil = \min\{n \in \mathbb{N};\ n \geq x\}$, and finally LWSc-MLP with weights generated by the (strictly decreasing) logistic function

$$w_i = \left(1 + \exp\left\{\frac{i-n-1}{n}\right\}\right)^{-1}, \quad i = 1, \ldots, n. \tag{13}$$

While LTS-MLP loss detects outliers and trims them away, LWS-MLP estimator does not do this but intrinsically arranges observations according to outlyingness.

Another alternative version denoted here as LTA-MLP was defined in [24], where a robust loss function corresponding to the least trimmed absolute value (LTA) estimator was used. LTA-MLP is defined for a fixed h ($n/2 \leq h < n$) by means of

$$\ell := \sum_{i=1}^{h} |u_{(i)}| \tag{14}$$

and according to [24] yields very similar results to those of LTS-MLP. We can say that the LTA estimator is practically unknown in the community of robust statistics; at the same time, it is not sufficiently discussed in the majority of monographs on robust estimation [12]. It is worth noting that, although we are not aware of systematic numerical comparison of the LTA estimator with other robust estimates in linear regression, it has been claimed that the performance of the LTA is very similar to that of the LTS in linear regression. Possible improvements of the LTA compared to the LTS are known not to be more than only marginal (see p. 429 of [29]). Still, the

LWS estimator seems to be much more promising in terms of both robustness and efficiency, as repeatedly discussed [5, 28].

Radial basis function (RBF) networks represent another important class of neural networks. They contain an input layer with p inputs, a single hidden layer with N RBF units (neurons), and a linear output layer. The user chooses N together with a radially symmetric function denoted here as ρ. The RBF network is based also on minimizing (8); using the Gaussian density as ρ, the residuals can be expressed as

$$u_i = Y_i - \sum_{j=1}^{N} a_j \rho(||X_i - c_j||), \quad i = 1, \ldots, n, \tag{15}$$

with parameters $c_1, \ldots, c_N \in \mathbb{R}^p$ and $a_1, \ldots, a_N \in \mathbb{R}$, and possibly with other parameters corresponding to ρ. We refer to [10, 16] for a detailed description of RBF networks. RBF networks can be expressed in an analogous way as MLPs in Algorithm 1 by means of minimizing the sum of squared residuals.

Robust versions of RBF networks, which will be denoted here as LTS-RBF, LWSa-RBF, LWSb-RBF, or LWSc-RBF networks, will be defined by means of the loss functions above. In other words, the are obtained by replacing the quadratic loss in (15) by the loss functions of the LTS or LWS estimators.

We implemented all the robust neural networks in Keras [3]. The implementation exploits a back-propagation algorithm, namely a stochastic gradient descent method, i.e. the same approach as in [24, 25], for optimization of all parameters for both standard and robust MLPs as well as RBF networks. As our experiments have demonstrated, also the loss function of LWS-MLP and LWS-RBF networks is in practice smooth enough for our gradient-based approach.

4 Numerical Experiments

The aim of the computations over 1 simulated and 3 real datasets is to illustrate the performance of the novel robust neural networks and compare it with other nonlinear regression tools.

4.1 Data Description

(A) The so-called Eckerle4 dataset publicly available in the package NISTnls of R software [20] has $p = 1$ regressor and $n = 35$ observations, including one apparent outlier. In Fig. 1, this real dataset is presented together with fitted trend, estimated by a standard MLP as well as LTS-MLP.

(B) A simulated dataset obtained by means of a sine function with a (rather artificial) contamination by a linearly decreasing trend with $p = 1$ and $n = 101$.

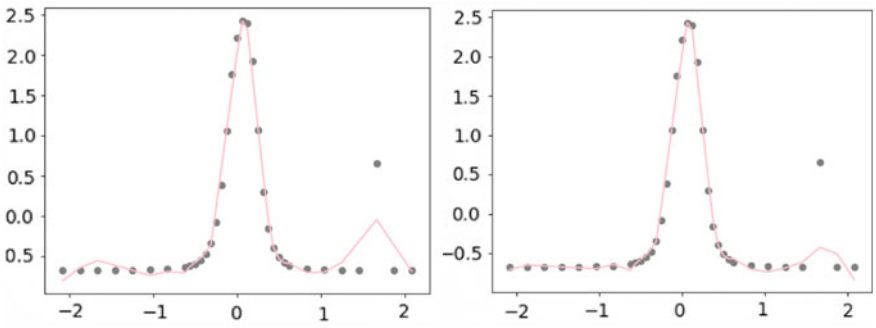

Fig. 1 Dataset Eckerle4. Horizontal axis: the regressor. Vertical axis: the response. The curve corresponds to the standard MLP (left) and LTS-MLP with $h = \lfloor 3n/4 \rfloor$ (right)

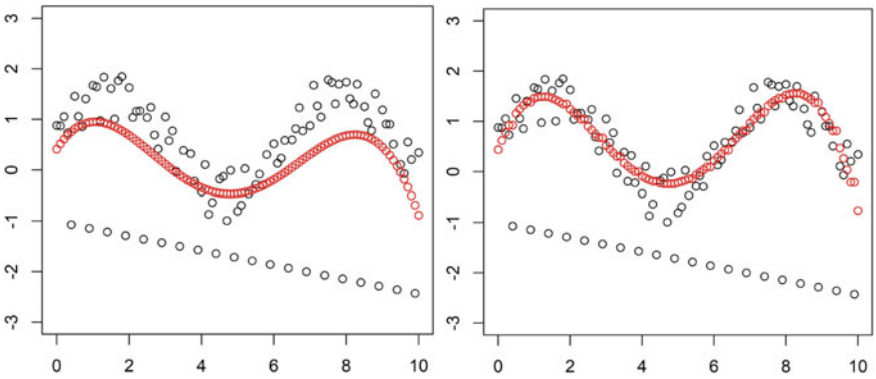

Fig. 2 The simulated dataset. Horizontal axis: the regressor. Vertical axis: the response. The curve corresponds to the standard RBF network (left) and LTS-RBF network with $h = \lfloor 3n/4 \rfloor$ (right)

The dataset is presented in Fig. 2, together with estimated trend, obtained by a standard RBF as well as LTS-RBF network.

(C) The Auto MPG dataset [8] with $p = 4$ continuous regressors and $n = 392$ observations after omitting all missing values (i.e. observations with index 33, 127, 331, 337, 355, and 375) from the original dataset. The consumption of each car in miles per gallon (MPG) is considered here as a response explained by engine displacement, horsepower, weight, and acceleration.

(D) The Boston Housing dataset [8] with $p = 11$ continuous regressors (omitting features 4, 7, and 9 from the original dataset) and $n = 506$ observations. The per capita crime rate by town (i.e. in each individual location) is considered as the response variable here.

4.2 Methods

The following methods will be used in the computations. For the description of standard machine learning methods, the reader may refer to monographs [9, 10].

- RBF network. The number N of RBF units used in particular examples is specified in Table 1.
- LTS-RBF network with the same architecture as the plain RBF network and $h = \lfloor 3n/4 \rfloor$.
- LWS-RBF (i.e. LWAa-RBF, LWSb-RBF, LWSc-RBF) networks with the same architecture as the plain RBF network.
- MLP with 1 or 2 hidden layers as specified in Table 1 for particular examples, together with the number of neurons in these layers. In every example, a sigmoid activation function is considered in every hidden layer. A linear output layer is always used.
- LTS-MLP with the same architecture as the plain MLP and $h = \lfloor 3n/4 \rfloor$.
- LWS-MLP with the same architecture as the plain ML.

Three different measures of prediction errors are evaluated for each situation within a ten-fold cross validation study, performed in a standard way. Because the standard MSE suffers from the presence of outliers in the data, we also consider the trimmed MSE (TMSE) and weighted MSE (WMSE) defined formally as

$$\mathsf{MSE} = \frac{1}{n}\sum_{i=1}^{n} r_i^2, \quad \mathsf{TMSE}(\alpha) = \frac{1}{h}\sum_{i=1}^{h} r_{(i)}^2, \quad \mathsf{WMSE} = \sum_{i=1}^{n} w_i r_{(i)}^2, \tag{16}$$

where $r_i = Y_i - \hat{Y}_i$ are prediction errors and $\hat{Y}_i$ denotes the fitted value of the i-th observation for $i = 1, \ldots, n$. For TMSE, we choose h as the is integer part of $3n/4$, and squared prediction errors are arranged as $r_{(1)}^2 \leq \cdots \leq r_{(n)}^2$. WMSE requires to use some fixed non-increasing magnitudes of weights and we use here trimmed linear weights (12) with $\sum_{i=1}^{n} w_i = 1$.

4.3 Results

The results for standard as well as robust neural networks with the selected architectures and parameters are presented in Table 1. The number N of RBF units for all versions of RBF networks was selected as the most suitable one for plain RBF networks. The number of neurons in the hidden layers for all versions of MLPs was selected as the most suitable for plain MLPs.

The dataset Eckerle4, the simplest from the 4 datasets under considerations, is very simple with very much variability (except for an apparent outlier). Results over the two datasets with $p = 1$ are illustrated in Figs. 1 and 2. In these datasets with

Table 1 Results of numerical experiments. Three error measures (MSE, TMSE and WMSE) defined in (16) evaluated for various nonlinear regression methods for 4 datasets. The architectures (number of RBF units and neurons in hidden layers) are specified here for various versions of RBF networks and MLPs, respectively

Neural network	MSE	TMSE	WMSE	MSE	TMSE	WMSE
	Dataset Eckerle4			Simulated dataset (Fig. 2)		
	$p = 1, n = 35$			$p = 1, n = 101$		
	3 RBF units			10 RBF units		
RBF	0.03	<0.01	<0.01	0.18	0.055	0.050
LTS-RBF	0.04	<0.01	<0.01	0.29	0.035	0.033
LWSa-RBF	0.04	<0.01	<0.01	0.28	0.037	0.033
LWSb-RBF	0.04	<0.01	<0.01	0.31	0.032	0.029
LWSc-RBF	0.04	<0.01	<0.01	0.32	0.038	0.031
	1 hidden layer			1 hidden layer		
	with 4 neurons			with 8 neurons		
MLP	0.04	<0.01	<0.01	0.24	0.067	0.061
LTS-MLP	0.05	<0.01	<0.01	0.30	0.038	0.034
LWSa-MLP	0.05	<0.01	<0.01	0.32	0.038	0.033
LWSb-MLP	0.05	<0.01	<0.01	0.33	0.035	0.030
LWSc-MLP	0.05	<0.01	<0.01	0.35	0.037	0.032
	Auto MPG dataset			Boston housing dataset		
	$p = 4, n = 392$			$p = 11, n = 506$		
	40 RBF units			50 RBF units		
RBF	46.9	17.2	19.3	52.7	4.4	5.6
LTS-RBF	52.7	12.9	14.1	60.3	4.1	5.2
LWSa-RBF	54.1	14.4	13.8	62.1	4.1	5.0
LWSb-RBF	50.6	11.8	12.5	61.6	4.0	4.7
LWSc-RBF	53.7	14.0	13.4	62.4	4.3	4.9
	2 hidden layers			2 hidden layers		
	with 16 and 8 neurons			with 16 and 8 neurons		
MLP	60.8	28.9	31.0	57.9	5.3	6.3
LTS-MLP	69.4	14.3	17.6	67.2	4.3	5.9
LWSa-MLP	70.3	14.5	16.2	70.8	4.2	5.7
LWSb-MLP	71.6	13.9	15.8	68.8	4.1	5.5
LWSc-MLP	72.5	14.3	16.7	70.6	4.2	5.7

$p = 1$, TMSE is able to ignore the true outliers for robust but also for plain neural networks. This is because the regression task is not so difficult for these datasets and the outliers are exactly those points, which have large absolute values of the residuals. The situation becomes much more complex for the other datasets.

In all examples, robust versions of neural networks approaches are able to yield smaller values of robust prediction errors (TMSE and WMSE); this is true in spite of the fact that the architecture of the neural networks was optimized for the plain networks. On the other hand, standard versions of neural networks are superior in terms of conventional MSE. This does not mean that the robust methods are less suitable, because the MSE itself is vulnerable to the presence of outliers. Thus, only robust versions of MSE should be considered for data contaminated with outliers.

Comparing RBF networks with MLPs, RBF networks turn out to yield smaller values of the prediction errors for all 4 datasets. It is especially interesting for the two datasets with $p > 1$ from real applications that the superiority of robust neural networks compared to standard (non-robust) ones is revealed. Basically we can say that using (any) robust neural network brings benefits, while the results of LWSb-RBF networks are not overcome by any other method in the 4 datasets.

5 Conclusions

Robust alternatives to training neural networks are highly desirable because of the vulnerability of common types of neural networks to the presence of outliers in the data. We use highly robust estimators corresponding to the LTS and LWS estimators to formulate robust loss function of MLPs and RBF networks. Thus, we extend the idea of [24], who used the loss function of the LTS (only) within MLPs. To the best of our knowledge, our approach is the first application of the LWS estimator within neural networks. The novel methods assign implicit weights to individual observations and correspond to their outlyingness, which offers a possible interpretation of individual observations and their influence to the resulting estimated trend. Robust fitting of neural networks based on the loss function of the least weighted squares estimator is able to minimize robust measures of prediction error. The methods denoted as LWSb-MLPs and LWSb-RBF networks, i.e. those with trimmed linear weights, turn out to yield better results in terms of prediction accuracy compared to other choices of weights for the LWS loss.

The superior results of the neural networks based on the LWS estimator are in correspondence with recent findings of [15]. There, the LWS turned out to outperform other estimators in linear regression, including S-estimators and mainly MM-estimators, where the latter allow to tune paramters so that a high robustness and a high efficiency are reached simultaneously.

The robust neural networks considered in the paper appear suitable for all the 4 datasets considered in this paper and thus are recommendable for real datasets, where robustness to data contamination by outliers is desirable. All datasets analyzed here do contain outliers. If a new dataset should be analyzed, which does not

seem to contain apparent outliers, the strategy common in linear regression may be adopted for neural networks as well; namely, the novel robust neural networks may serve as a diagnostic tool. In such a situation, the user may check if the results of a standard neural network are similar with results of robust ones. In case of remarkable discrepancies, the robust approach may be more suitable. As a limitation, however, it is necessary to state that the robust neural networks of this paper (just like any robust statistical method [12]) may not be suitable for certain datasets, e.g. when we are interested in every individual observation and ignoring specific observations (or their clusters) is not desirable.

Several possible directions recommendable for future research include adapting robust neural networks for heteroscedastic data, proposing an adaptive selection of h for the LTS-based loss function, considering robust and regularized neural networks, or proposing adaptive (data-dependent) selection of weights for the LWS-based loss. In addition, it would be desirable to perform a systematic comparison of robust approaches to training neural networks over a larger number of datasets, accompanied by a detailed statistical analysis of the data and by a thorough interpretation of the results on the level of individual observations.

Acknowledgements The work is supported by the projects GA19-05704S and GA18-23827S of the Czech Science Foundation. The authors are grateful to Jan Tichavský and Jiří Tumpach for technical help.

References

1. Alnafessah, A., Casale, G.: Artificial neural networks based techniques for anomaly detection in Apache Spark. Cluster Computing (2020) (online first)
2. Borş, A.G., Pitas, I.: Robust RBF networks. In: Howlett, R.J., Jain, L.C., Kacprzyk, J. (eds.), Radial basis function networks 1. Recent developments in theory and applications, pp. 123–133. Physica Verlag Rudolf Liebing KG, Vienna (2001)
3. Chollet, F.: Keras. Github repository (2015). https://github.com/fchollet/keras
4. Čížek, P.: Semiparametrically weighted robust estimation of regression models. Comput. Stat. Data Anal. **55**, 774–788 (2011)
5. Čížek, P.: Reweighted least trimmed squares: an alternative to one-step estimators. Test **22**, 514–533 (2013)
6. Davies, L.: Data analysis and approximate models. In: Nonparametric Regression and Image Analysis. CRC Press, Boca Raton, Model Choice, Location-scale, Analysis of Variance (2014)
7. Du, K.L., Swamy, M.N.S.: Neural Networks and Statistical Learning. Springer, London (2014)
8. Frank, A., Asuncion, A.: UCI Machine Learning Repository. University of California, Irvine (2010). http://archive.ics.uci.edu/ml
9. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, 2nd edn. Springer, New York (2009)
10. Haykin, S.O.: Neural Networks and Learning Machines: A Comprehensive Foundation, 3rd edn. Prentice Hall, Upper Saddle River (2009)
11. Hubert, M., Rousseeuw, P.J., van Aelst, S.: High-breakdown robust multivariate methods. Stat. Sci. **23**, 92–119 (2008)
12. Jurečková, J., Picek, J., Schindler, M.: Robust Statistical Methods with R, 2nd edn. Chapman & Hall/CRC, Boca Raton (2019)

13. Kalina, J.: Implicitly weighted methods in robust image analysis. J. Math. Imag. Vis. **44**, 449–462 (2012)
14. Kalina, J.: A robust supervised variable selection for noisy high-dimensional data. BioMed. Res. Int. Article 320385 (2015)
15. Kalina, J., Tichavský, J.: On robust estimation of error variance in (highly) robust regression. Meas. Sci. Rev. **20**, 1–9 (2020)
16. Kalina, J., Vidnerová, P.: Robust training of radial basis function neural networks. In: Proceedings 18th International Conference ICAISC 2019, pp. 113–124 (2019)
17. Kordos, M., Rusiecki, A.: Reducing noise impact on MLP training—techniques and algorithms to provide noise-robustness in MLP network training. Soft Comput. **20**, 46–65 (2016)
18. Lee, C.C., Chung, P.C., Tsai, J.R., Chang, C.I.: Robust radial basis function neural networks. IEEE Trans. Syst. Man Cybern. B **29**, 674–685 (1999)
19. Mašíček, L.: Optimality of the least weighted squares estimator. Kybernetika **40**, 715–734 (2004)
20. R Core Team: R: A language and environment for statistical computing. In: R Foundation for Statistical Computing, Vienna (2019). https://www.R-project.org/
21. Roelant, E., van Aelst, S., Willems, G.: The minimum weighted covariance determinant estimator. Metrika **70**, 177–204 (2009)
22. Rousseeuw, P.J., Hubert, M.: Robust statistics for outlier detection. WIREs Data Mining Knowl. Discov. **1**, 73–79 (2011)
23. Rousseeuw, P.J., Van Driessen, K.: Computing LTS regression for large data sets. Data Mining Knowl. Discov. **12**, 29–45 (2006)
24. Rusiecki, A.: Robust learning algorithm based on LTA estimator. Neurocomputing **120**, 624–632 (2013)
25. Rusiecki, A., Kordos, M., Kamiński, T., Greń, K.: Training neural networks on noisy data. Lect. Notes Artif. Intel. **8467**, 131–142 (2014)
26. Su, M.J., Deng, W.: A fast robust learning algorithm for RBF network against outliers. Lect. Notes Comput. Sci. **4113**, 280–285 (2006)
27. Víšek, J.Á.: The least trimmed squares. Part I: consistency. Kybernetika **42**, 1–36 (2006)
28. Víšek, J.Á.: Consistency of the least weighted squares under heteroscedasticity. Kybernetika **47**, 179–206 (2011)
29. Wilcox, R.R.: Introduction to Robust Estimation and Hypothesis Testing, 2nd edn. Elsevier, Burlington (2005)