Robust LTS Backpropagation Learning Algorithm

Andrzej Rusiecki

Institute of Computer Engineering, Control and Robotics, Wroclaw, Poland
andrzej.rusiecki@pwr.wroc.pl

Abstract. Training data sets containing outliers are often a problem for supervised neural networks learning algorithms. They may not always come up with acceptable performance and build very inaccurate models. In this paper new, robust to outliers, learning algorithm based on the Least Trimmed Squares (LTS) estimator is proposed. The LTS learning algorithm is simultaneously the first robust learning algorithm that takes into account not only gross errors but also leverage data points. Results of simulations of networks trained with the new algorithm are presented and the robustness against outliers is demonstrated.

1 Introduction

Feedforward neural networks (FFNs) are often considered as universal tools and find their applications in areas such as function approximation, pattern recognition, or signal and image processing. One of the main advantages of using FFNs is that they usually do not require, in the learning process, exact mathematical knowledge about input-output dependencies. In other words, they may be regarded as model-free approximators [5]. They learn by minimizing some kind of an error function to fit training data as close as possible. Such learning scheme doesn't take into account a quality of the training data, so its performance depends strongly on the fact whether the assumption, that the data are reliable and trustable, is hold. This is why when the data are corrupted by the large noise, or when outliers and gross errors appear, the network builds a model that can be very inaccurate.

In most real-world cases the assumption that errors are normal and iid, simply doesn't hold. The data obtained from the environment are very often affected by noise of unknown form or outliers, suspected to be gross errors. The quantity of outliers in routine data ranges from 1 to 10% [4]. They usually appear in data sets during obtaining the information and pre-processing them when, for instance, measurement errors, long-tailed noise, or results of human mistakes may occur.

Intuitively we can define an outlier as an observation that significantly deviates from the bulk of data. Nevertheless, this definition doesn't help in classifying an outlier as a gross error or a meaningful and important observation. To deal with the problem of outliers a separate branch of statistics, called robust statistics [4,6], was developed. Robust statistical methods are designed to act well when the true underlying model deviates from the assumed parametric model. Ideally, they should be efficient and reliable for the observations that are very close to the assumed model and simultaneously for the observations containing larger deviations and outliers.

The other way is to detect and remove outliers before the beginning of the model building process. Such methods are more universal but they do not take into account the specific type of modeling philosophy (e.g. modeling by the FFNs). In this article we propose a new robust FFNs learning algorithm based on the least trimmed squares estimator.

The most popular FFNs learning scheme makes use of the backpropagation (BP) strategy and a minimization of the mean squared error (mse). Until now, a couple various robust BP learning algorithms have been proposed. Generally, they take advantage of the idea of robust estimators. This approach was adopted to the neural networks learning algorithms by replacing the mse with a loss error function of such a shape that the impact of outliers may be, in certain conditions, reduced or even removed.

Chen and Jain [1] proposed the Hampel's hyperbolic tangent as a new error criterion, with the scale estimator β that defines the interval supposed to contain only clean data, depending on the assumed quantity of outliers or current errors values. This idea was combined with the annealing concept by Chunag and Su [2]. They applied the annealing scheme to decrease the value of β, whereas Liano [8] introduced the logistic error function derived from the assumption of the errors generated with the Cauchy distribution. In a recent work Pernia-Espinoza et al. [9] presented an error function based on tau-estimates. An approach based on the adaptive learning rate was also proposed [13]. Such modifications may significantly improve the network performance for corrupted training sets. However, even these approaches suffer from several difficulties and cannot be considered as universal (also because of properties of applied estimators). Besides, very few of them have been proposed until today and they exploit the same basic idea, so we still need to look for new solutions.

2 Robust LTS Learning Algorithm

The least trimmed squares estimator (LTS), introduced by Rousseuw [10,11], is a classical high break-down point robust estimator, similar to the slower converging least median of squares (LMS) [10]. The estimator and its evaluations are often used in linear and nonlinear regression problems, in sensitivity analysis, small-sample corrections, or in simple detecting outliers. The main difference between the LTS estimator and the least sum of squares, but also M-estimators, is obviously the operation performed on residuals. In this case however, robustness is achieved not by replacing the square by another function but by superseding the summation sign with something else.

Let us consider the general nonlinear regression model:

$$y_i = \eta(x_i, \theta) + \epsilon_i, \quad i = 1, \ldots, n, \tag{1}$$

where y_i represents the dependent variable, $x_i = (x_{i1}, \ldots, x_{ik})$ the independent input vector, and $\theta \in R^p$ denotes the underlying parameter vector. The random error term ϵ_i is a sequence of iid random variables with a continuous distribution function. The nonlinear least trimmed squares estimator is then defined as:

$$\hat{\theta} = \arg \min_{\theta \in R^p} \sum_{i=1}^{h} (r^2)_{i:n}, \tag{2}$$

where $(r^2)_{1:n} \leq \ldots \leq (r^2)_{n:n}$ are the ordered squared residuals $r_i^2(\theta) = \{y_i - \eta(x_i, \theta)\}^2$. The trimming constant h must be chosen as $n/2 < h \leq n$ to provide that $n - h$ observations with the largest residuals do not directly affect the estimator. Under certain assumptions the estimator should be robust not only to outliers [14] but also to the leverage points (grossly aberrant values of x_i)[12]. Such property is obviously rather theoretical but it explains why the LTS method is very often in use. Moreover, unlike the LMS, the LTS converges like $n^{-1/2}$ having the same asymptotic efficiency at the normal distribution as the M estimator called Huber skipped mean. Besides, its objective function is smoother than in the case of the LMS, which makes it possible to be applied also for the gradient based FFNs learning algorithms.

For simplicity, let us consider a simple three layer feedforward neural network with one hidden layer. The net is trained on a set of n training pairs: $\{(\boldsymbol{x}_1, \boldsymbol{t}_1), (\boldsymbol{x}_2, \boldsymbol{t}_2), \ldots, (\boldsymbol{x}_n, \boldsymbol{t}_n)\}$, where $\boldsymbol{x}_i \in R^p$ and $\boldsymbol{t}_i \in R^q$. For the given input vector $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$, the output of the jth neuron of the hidden layer may be obtained as:

$$z_{ij} = f_1\Big(\sum_{k=1}^{p} w_{jk} x_{ik} - b_j\Big) = f_1(inp_{ij}), \qquad \text{for } j = 1, 2, \ldots, l, \tag{3}$$

where $f_1(\cdot)$ is the activation function of the hidden layer, w_{jk} is the weight between the kth net input and jth neuron, and b_j is the bias of the jth neuron. Then the output vector of the network $\boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots, y_{iq})^T$ is given as:

$$y_{iv} = f_2\Big(\sum_{j=1}^{l} w'_{vj} z_{ij} - b'_v\Big) = f_2(inp_{iv}), \qquad \text{for } v = 1, 2, \ldots, q. \tag{4}$$

Here $f_2(\cdot)$ denotes the activation function, w'_{vj} is the weight between the vth neuron of the output layer and the jth neuron of the hidden layer, and b'_v is the bias of the vth neuron of the output layer. Now, we introduce the robust LTS error criterion, based on the Least Trimmed Squares estimator. The new error function is defined as:

$$E_{LTS} = \sum_{i=1}^{h} (r^2)_{i:n}. \tag{5}$$

In this case, $(r^2)_{1:n} \leq \ldots \leq (r^2)_{n:n}$ are ordered squared residuals of the form

$$r_i^2 = \Big\{\sum_{v=1}^{q} |(y_{iv} - t_{iv})|\Big\}^2. \tag{6}$$

The trimming constant h must be carefully chosen because it is responsible for the quantity of patterns suspected to be outliers.

We assume, for simplicity, that weights are updated according to the gradient-descent learning algorithm but this can be extended to any other gradient-based algorithm. Then to each weight is added (α denotes a learning coefficient):

$$\Delta w_{jk} = -\alpha \frac{\partial E_{LTS}}{\partial w_{jk}} = -\alpha \frac{\partial \sum_{i=1}^{h} (r^2)_{i:n}}{\partial r_i} \frac{\partial r_i}{\partial w_{jk}}, \tag{7}$$

$$\Delta w'_{vj} = -\alpha \frac{\partial E_{LTS}}{\partial w'_{vj}} = -\alpha \frac{\partial \sum_{i=1}^{h} (r^2)_{i:n}}{\partial r_i} \frac{\partial r_i}{\partial w'_{vj}}, \tag{8}$$

where

$$\frac{\partial r_i}{\partial w_{jk}} = f'_2(inp_{iv}) w'_{vj} f'_1(inp_{ij}) x_{ik}, \tag{9}$$

and

$$\frac{\partial r_i}{\partial w'_{vj}} = f'_2(inp_{iv}) z_{ij}. \tag{10}$$

The main problem that may occur here is calculating the E_{LTS} derivative. It is not continuous and it can be written as:

$$\frac{\partial \sum_{i=1}^{h} (r^2)_{i:n}}{\partial r_i} = \begin{cases} 2r_i & \text{for } r_i^2 \leq (r^2)_{i:h} \\ 0 & \text{for } r_i^2 > (r^2)_{i:h} \end{cases} \tag{11}$$

As it was experimentally demonstrated, such shape of the derivative function is smooth enough for the BP learning algorithm.

In the use of robust learning algorithms, there exist some problems, concerning mainly the choice of a starting point for the method. In fact, we can divide it into two tasks: choosing initial network parameters, and choosing the right scale estimator. If the initial weights of the network are not properly selected, the learning process may move in the wrong direction and the algorithm may stack in a local minimum. In this case the network performance might become very poor. The scale estimator or its equivalent (here, the trimming constant h) is responsible for the amount of outliers that are to be rejected during the training, it's clearly evident then, that if h is incorrect, gross errors may be regarded as good data and desired points may be discriminated.

Following [1], we decided to use our LTS robust algorithm after a period of training by the traditional BP algorithm to set the initial parameters. We proposed two strategies of choosing the trimming parameter h. In the first approach we assumed a predefined value of h, depending on expected percentage of outliers in the training data (LTS1). In this case, additional a-priori knowledge of the error distribution is needed, so the strategy is not very useful. The second approach (LTS2) is to choose h by using the median of all errors as:

$$h = \|\{r_i : |r_i| < c * \text{median}(|r_i|), i = 1 \dots n\}\|, \tag{12}$$

where $c = 1.483$ for the MAD scale estimate [6]. Errors used for calculating h were the errors obtained after the last epoch of the traditional backpropagation algorithm, so the value of h is set constant for the training process.

3 Simulation Results

The LTS learning algorithm was tested on function approximation tasks. In this paper we present only a few of many different testing situations. The first function to be approximated is $y = x^{-2/3}$ proposed in [1], the second one is a two-dimensional spiral given as $x = \sin y$, $z = \cos y$.

To simulate real data containing noise and outliers we used different models, defined as follows:

- Clean data without noise and outliers;
- Data corrupted with the Gross Error Model: $F = (1 - \delta)G + \delta H$, where F is the error distribution, $G \sim N(0.0, 0.1)$ and $H \sim N(0.0, 10.0)$ are Gaussin noise and outliers and occur with probability $1 - \delta$ and δ (data Type 1);
- Data with high value random outliers (Type 2), proposed in [9] of the form $F = (1 - \delta)G + \delta(H_1 + H_2 + H_3 + H_4)$, where:
  - $H_1 \sim N(15, 2)$,
  - $H_2 \sim N(-20, 3)$,
  - $H_3 \sim N(30, 1.5)$,
  - $H_4 \sim N(-12, 4)$.
- Data with outliers generated from the Gross Error Model, injected into the input vector $\boldsymbol{x}_i$ (Type 3).

The performances of the traditional backpropagation algorithm (BP), robust LMLS algorithm, and the both variations of the novel robust LTS algorithm, LTS1 and LTS2, were compared. The tested algorithms were employed to teach a simple three-layer network with one or two inputs (depending on a problem), one output and ten hidden sigmoid neurons. We used the conjugate gradient optimization method [3]. Each algorithm was run 100 times for each task, then a mean MSE for the networks learnt with the given algorithm was calculated. The mean MSE was obtained by testing the nets on the clean data generated as points lying on the approximated curves. Simulation results were gathered in tables and the exemplary network responses for the testing data were shown in figures.

Looking at the Table 1 we can see that for the clean data of the first task, all algorithms act relatively well but the error is slightly bigger for the LTS2. We didn't considered here the LTS1 criterion because, in this case, it's equivalent

Table 1. The mean MSE for the 100 trials for the networks trained to approximate function of one variable

	Clean Data	Data with gross errors (Type 1)		Data with high value outliers (Type 2)		Data with gross errors in the input vector (Type 3)	
Algorithm	$\delta = 0.0$	$\delta = 0.1$	$\delta = 0.2$	$\delta = 0.1$	$\delta = 0.2$	$\delta = 0.1$	$\delta = 0.2$
BP	0.0007	0.0398	0.0809	1.7929	4.0996	0.0140	0.0180
LMLS	0.0007	0.0061	0.0088	0.0050	0.0053	0.0151	0.0177
LTS1	-	0.0054	0.0056	0.0632	0.1454	0.0104	0.0120
LTS2	0.0013	0.0049	0.0067	0.0051	0.0061	0.0112	0.0149

Table 2. The mean MSE for the 100 trials for the networks trained to approximate two-dimensional spiral

	Clean Data	Data with gross errors (Type 1)		Data with high value outliers (Type 2)	Data with gross errors in the input vector (Type 3))	
Algorithm	$\delta = 0.0$	$\delta = 0.1$	$\delta = 0.2$	$\delta = 0.1$	$\delta = 0.1$	$\delta = 0.2$
BP	0.0000	0.3967	0.7722	24.9154	0.0014	0.0057
LMLS	0.0000	0.0584	0.1442	0.0682	0.0006	0.0034
LTS1	-	0.0318	0.0390	1.7108	0.0001	0.0023
LTS2	0.0006	0.0284	0.0534	0.0311	0.0007	0.0023

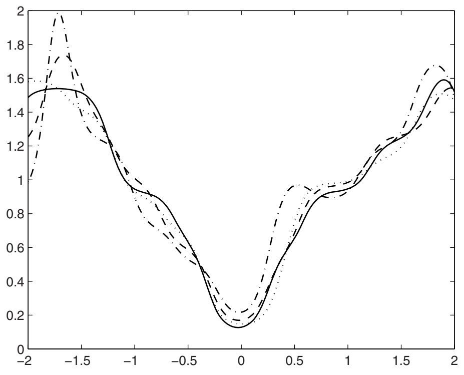

Fig. 1. Simulation results for the network trained to approximate one dimensional function (data Type 1): backpropagation algorithm (*dash- dot line*), LMLS alg. (*dashed line*), LTS1 alg. (*dotted line*), LTS2 alg. (*solid line*)

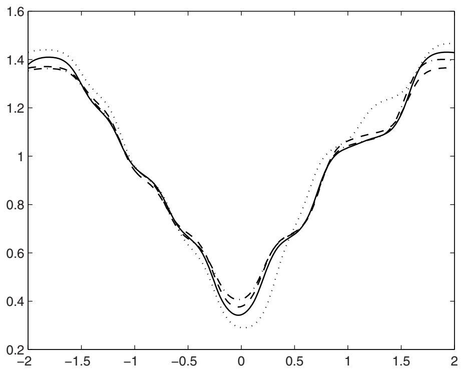

Fig. 2. Simulation results for the network trained to approximate one dimensional function (data Type 3): backpropagation algorithm (*dash- dot line*), LMLS alg. (*dashed line*), LTS1 alg. (*dotted line*), LTS2 alg. (*solid line*)

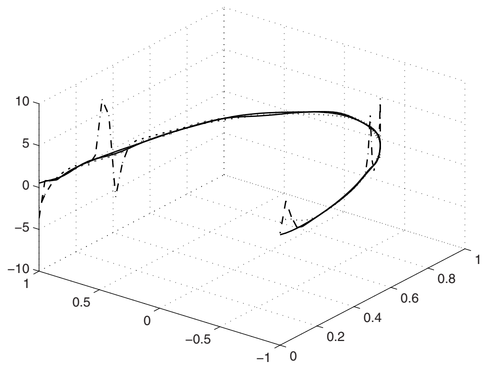

Fig. 3. Simulation results for the network trained to approximate two-dimensional spiral (data Type 1): backpropagation algorithm (*dash- dot line*), LMLS alg. (*dashed line*), LTS1 alg. (*dotted line*), LTS2 alg. (*solid line*)

with a simple squared error. For the data containing gross errors, the two variations of the LTS present the best performance and it is hard to say, which of them is better. For the data with high value outliers (Table 1) only LTS2 and LMLS ensure good fitting to the testing data, while LTS1, though still better than the BP algorithm, acts rather poor.

After analysing results obtained for the data containing outliers injected into input vectors, we can notice that the situation is different here. All algorithms, including the one with Mse criterion, has similar level of error. Moreover, the influence of leverage points in the input vector to the training process seems to be smaller than in the case of outliers. Nevertheless, the algorithms LTS1 and LTS2 showed the best performance and the error of the LTS1 is even over 25% better than for the Lmls and BP.

Results obtained for the second approximation task are, generally, similar. Observing the Table 2 we may notice that none of the algorithms has problems with learning on the clean data. For the data containing outliers, the superiority of the LTS algorithm is clearly evident. The LTS2 acts well also for the high value outliers, showing the lowest error. Besides, for gross errors in the input vector also the LTS1 and LTS2 appear to be the best.

To summarise, one can notice that both LTS algorithms showed performance better than other two algorithms for the data containing gross errors in the input, as well as in the output vector. Obviously, they also act well for the clean data. For the data with high value outliers they are not so reliable but they still act significantly better than usual BP algorithm with the Mse criterion function.

4 Summary

In this paper a novel robust LTS learning algorithm was proposed. As it was experimentally demonstrated, it behaves better than traditional algorithm, and robust Lmls algorithm, in the presence of outliers in the training data. Moreover, it is simultaneously the first robust learning algorithm that takes into account also gross errors injected into the input vector of the training patterns (leverage points). Especially in its second version (LTS2), with median error used to set the trimming constant h, it can be considered as simple and effective mean to increase learning performance on the contaminated data sets. It doesn't need any additional a-priori knowledge of the assumed error distribution to ensure relatively good training results in any conditions. The robust LTS learning algorithm can be easily adapted to many types of neural networks learning strategies.

References

1. Chen, D.S., Jain, R.C.: A robust back propagation learning algorithm for function approximation. IEEE Transactions on Neural Networks 5, 467–479 (1994)
2. Chuang, C., Su, S., Hsiao, C.: The Annealing Robust Backpropagation (ARBP) Learning Algorithm. IEEE Transactions on Neural Networks 11, 1067–1076 (2000)
3. Hagan, M.T., Demuth, H.B., Beale, M.H.: Neural Network Design. PWS Publishing, Boston, MA (1996)
4. Hampel, F.R., Ronchetti, E.M., Rousseeuw, P.J., Stahel, W.A.: Robust Statistics the Approach Based on Influence Functions. John Wiley and Sons, New York (1986)
5. Hornik, K., Stinchconbe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
6. Huber, P.J.: Robust Statistics. Wiley, New York (1981)
7. Jacobs, R.A.: Increased rates of convergence through learning rate adaptation. Neural Networks 1, 295–307 (1988)
8. Liano, K.: Robust error measure for supervised neural network learning with outliers. IEEE Transactions on Neural Networks 7, 246–250 (1996)
9. Pernia-Espinoza, A.V., Ordieres-Mere, J.B., Martinez-de-Pison, F.J., Gonzalez-Marcos, A.: TAO-robust backpropagation learning algorithm. Neural Networks 18, 191–204 (2005)
10. Rousseeuw, P.J.: Least median of squares regression. Journal of the American Statistical Association 79, 871–880 (1984)
11. Rousseeuw, P.J.: Multivariate Estimation with High Breakdown Point, Mathematical Statistics and Applications, vol. B. Reidel, the Netherlands (1985)
12. Rousseeuw, P.J., Leroy, A.M.: Robust Regression and Outlier Detection. Wiley, New York (1987)
13. Rusiecki, A.L.: Robust Learning Algorithm with the Variable Learning Rate, ICAISC 2006, Artificial Intelligence and Soft Computing, pp. 83–90 (2006)
14. Stromberg, A.J., Ruppert, D.: Breakdown in nonlinear regression. J. Amer. Statist. Assoc. 87, 991–997 (1992)
15. Vogl, T.P., et al.: Accelerating the convergence of the backpropagation method. Biological Cybernetics 59, 256–264 (1988)