Preprocessing of Training Set for Back Propagation Algorithm: Histogram Equalization

Taek M. Kwon[†], Ehsan H. Feroz[††], and Hui Cheng[†††]
Department of Computer Engineering[†]
Department of Accounting[††]
Department of Mathematics[†††]
University of Minnesota, Duluth
271 MWAH, 10 University Dr.
Duluth, MN 55812

Abstract

This paper introduces a data preprocessing algorithm that can improve the efficiency of the standard back propagation (BP) algorithm. The basic approach is transforming input data to a range that associates high-slopes of sigmoid where relatively large modification of weights occurs. This helps escaping of early trapping from prematured saturation. However, a simple and uniform transformation to such desired range can lead to a slow learning if the data have a heavily skewed distribution. In order to improve the performance of BP algorithm on such distribution, we propose a modified histogram equalization technique which enhances the spacing between data points in the heavily concentrated regions of skewed distribution. Our simulation study shows that this modified histogram equalization can significantly speed up the BP training as well as improving the generalization capability of the trained network.

I. Introduction

The back propagation (BP) algorithm initially proposed by Werbos in 1974 [1] and later rediscovered by Rumelhart et al. in 1986 [2] is now recognized as a powerful tool in many neural network applications. The speed of algorithm has been significantly improved during last few years by introducing many different approaches (examples include [3-6],[8].). Further improvements of the BP algorithm have been also made in the area of overcoming the local minima problem. Researchers have shown that proper initialization of weights [8] and control of sigmoid slope [6] can overcome the local minima problem.

While many researches have been focusing on improving the BP algorithm itself, very little attention has been given to preprocessing of input data for the BP algorithm. In this paper, we propose a preprocessing algorithm (histogram equalization) of input data that can improve the speed of training as well as reducing the chances of the BP algorithm being trapped in local minima. This algorithm increases the dynamic range of input data and distribute the updated amount of weights on equal frequency basis. As an example of this preprocessing algorithm, we show preprocessing of a traing set for the Savings and Loan (S&L) failure classification. This training set has a particularly strange distribution which prevents the BP algorithm from being efficient.

II. The Network Model and Notations

The proposed preprocessing on training set is described for the BP algorithm applied to three-layered feedforward networks that consist of an input layer, a hidden layer, and an output layer. However this analysis can be readily extended to other structures.

A pattern X^p is represented by an N-dimensional vector: $X^p = (x_1^p, x_2^p, \ldots, x_N^p)^T$, where superscript T denotes a transpose operator. When a pattern is presented to the network, the input nodes take a role of a buffer in which they simply bypass the received pattern to hidden nodes. Let the number of nodes in the hidden layer be H, then the weights between the input and hidden layers can be represented by an $N \times H$ matrix, which we denote $W = [w_{ij}]$. The outputs of hidden nodes for a pattern p are described by:

$$h_j^p = f(a_j^p) \qquad j = 1, \cdots, H, \tag{1}$$

where a_j^p is the weighted linear summation at the jth hidden node:

$$a_j^p = w_{j0} + \sum_{i=1}^{N} w_{ji} x_i^p = \sum_{i=0}^{N} w_{ji} x_i^p. \tag{2}$$

Here, w_{j0} represents the threshold value of jth hidden node and $x_0^p = 1$. The activation function $f(\cdot)$ is a sigmoid function defined by

$$f(x) = \frac{1}{1 + \exp(-x)}. \tag{3}$$

Let the network have M output nodes, then the weight matrix formed between the hidden and output layers is $H \times M$. We denote this matrix by $V = [v_{jk}]$. The output nodes are then described by

$$y_k^p = f(b_k^p) \qquad k = 1, \cdots M, \tag{4}$$

where

$$b_k^p = v_{k0} + \sum_{j=1}^{H} v_{kj} h_j^p = \sum_{j=0}^{H} v_{jk} h_j^p \tag{5}$$

Similarly to (2), v_{k0} represents the threshold level of the kth output node and $h_0^p = 1$.

The standard BP algorithm for the above three-layered network can be summarized as follows [2]. The updating rule for the output layer is given by

$$\Delta v_{kj}^p = \eta_v^p \cdot (t_k^p - y_k^p) f'(b_k^p) h_j^p \tag{6}$$

for $k = 1, \cdots, M$ and $j = 0, \cdots, H$, where t_k^p is the kth element of the desired pattern p. The parameter η_v^p denotes the iteration rate of output layer when a pattern p is presented to the network. The updating rule for the hidden layer is expressed as

$$\Delta w_{ji}^p = \eta_w^p \cdot f'(a_j^p) \sum_{k=1}^{M} v_{kj} f'(b_k^p)(t_k^p - y_k^p) x_i^p \tag{7}$$

for $j = 1, \cdots, H$ and $i = 0, \cdots, N$. The parameter η_w^p denotes the iteration rate of hidden layer when a pattern p is presented to the network.

In the above BP algorithm, the size of Δw or Δv affects the learning time and is a function of three controllable components. The first component is the iteration rate (η_v^p and η_w^p) in which dynamic adaptation at each pattern presentation can be used to improve the training speed [3],[4]. The second component is $f'(\cdot)$ which represents the slope of sigmoid function. Dynamically adjusting the slope of sigmoid can also increase the learning speed and reduce the chances of the algorithm being trapped in local minima [5],[6]. The third component is the input patterns per each node i.e. x_i^p and h_j^p. This paper deals with the third component which can be handled through a preprocessing of training data. In the past, most research focused on developing efficient method for the first and second components. However, very little attention has been given to the third component, preprocessing of input patterns.

III. Preprocessing of Training Set

Notice from (6) and (7) that the magnitude of input data directly affects the speed of the BP learning. Suppose that an input pattern has a very large magnitude i.e. $||X^p|| >> 1$, then its gradient $f'(a^p)$ is likely to be near zero whether the actual output is correct or not. Notice from (7) that such pattern results in a very small amount of weight update and almost no improvements from the learning algorithm can be achieved. This condition is called prematured saturation [5],[6] and includes the cases of being trapped in local minima. Such condition must be prevented to make the BP algorithm efficient. One simple way of avoiding this situation is to transform input data, such that $f'(a^p) \geq \epsilon$ where ϵ is the smallest gradient allowed. In this study we choose $\epsilon = 0.1$, which yields $-2 \leq a^p \leq 2$ for the sigmoid function given in (3). Thus, our attention will be focused on finding a transformation

$$r = T(x) \tag{8}$$

which produces a value r for every input value x from the original training set. Since we are limiting a^p as discussed above, the range of r should be $-2/C \leq r \leq 2/C$ where C is a value proportional to the number of columns in the input data. We will denote $d_{min} = -2/C$ and $d_{max} = 2/C$.

First we consider a uniform transformation. Suppose that a training set has the distribution given by the histogram in Fig. 1. Most data points are concentrated around 9,000 and a small amount of data points are scattered in the wide range with the minimum at 1,500 and the maximum at 3.02×10^8. If we uniformly transform this data into the range $[0,1]$, it is obvious that this normalization will push most of data points to a very small region near zero. Recall from (7) that if x_i^p is near zero, the update amount is also near zero. Thus, most of data points will contribute to slowing down the learning speed of the network. This example typically shows that a uniform linear mapping is inefficient for the BP learning algorithm. Hence, we develop a nonuniform mapping in this paper. Since the ordering relation of original data must be preserved after the processing, we impose the following conditions on the transformation function given in (8).

(i) $T(x)$ is single valued and monotonically increasing i.e. $x \leq y$ implies $T(x) \leq T(y)$.

(ii) $d_{min} \leq T(x) \leq d_{max}$ for $x_{min} \leq x \leq x_{max}$.

As a transformation that satisfies the above two conditions, we propose a histogram equalization technique [7] that has been frequently used in image enhancement. This transformation function is given as the cumulative distribution function:

$$r = T(x) = |d_{max} - d_{min}| \int_{r_{min}}^{r} P_x(w)dw + d_{min} \quad x_{min} \leq x \leq x_{max} \tag{9}$$

where w is a dummy variable and $P_x(\cdot)$ is the probability density function of input data x. Notice that the two conditions imposed above are satisfied by (9). It can be shown that $P_r(r)$ is a uniform density function in the interval of the transform variable is defined [7].

In order to apply (9) into neural training data, (9) must be formulated in discrete form. The probability density function of the input data can be approximated by

$$P_x(x_k) = \frac{n_k}{R \cdot C} \quad k = 0, \cdots, L-1, \tag{10}$$

where L is the number of levels to be distinguished in the data set, n_k is the number of data points that appear in this level, and $R \cdot C$ is the total number of data points in the data set where R denotes the number of rows and C denotes the number of columns in the data set. Thus the discrete form of (9) is obtained as

$$r_k = T(x_k) = |d_{max} - d_{min}| \sum_{j=0}^{k} P_x(x_j) + d_{min} \qquad k = 0, 1, \cdots, L-1. \tag{11}$$

The effect of this transformation is modifying the narrow density of the original histogram to a wider range of distribution by transforming the data to a uniform density. Unfortunately, this transformation results in loosing some degree of information on the magnitude ratio between data points that appears in the original data. Thus we propose to modify the conventional histogram equalization technique to the following formulation:

$$r_k = T(x_k) = \left(\alpha \frac{x_i - x_{min}}{x_{max} - x_{min}} + \beta \sum_{j=0}^{k} P_x(x_j) \right) (d_{max} - d_{min}) - d_{min} \tag{12}$$

where $\alpha + \beta = 1$. Notice that if $\alpha = 0$ ($\beta = 1$), the result is exactly same as the conventional histogram equalization computed by (11). As α increases (β decreases), the magnitude ratio relation of the original data is more preserved.

IV. Example

As an example of the proposed preprocessing, we consider the S&L failure classification data. This data set includes financial information of 56 Rhode Island Credit Unions of which 34 failed and 12 survived. The training set consists of five columns of inputs and one column of output. A slice of this data is shown in Table 1 which includes the minimum and the maximum of data points in the original data set. The distribution of this data is shown in Fig. 1. Notice that the difference between the minimum and the maximum is significant. Moreover, due to the severely skewed distribution of this data, a uniform transformation to a small bounded region results in small amounts of updating for the most of input patterns in the BP learning as discussed in Section III.

For comparison purpose, we generated three different traing sets using the thee different preprocessing techniques: (i) a uniform normalization, (ii) the conventional histogram equalization, (iii) the modified histogram equalization with $\alpha = 0.8$ and $\beta = 0.2$.

We applied all three preprocessing methods to the standard BP algorithm under the same condition, i.e. (i) all weights are randomly initialized between $[-1, 1]$, (ii) $d_{min} = -0.8$, $d_{max} = 0.8$, (iii) $\eta = 0.2$, (iv) 10 patterns out of 56 patterns are put aside for generalization testing (not used in the training), (v) desired output is assigned as: "failed"=0.8 and "survived"=0.2.

Fig. 2 shows the squared errors of BP training on the three preprocessed training sets. Fig. 3 shows the result of generalization testing using the 10 test patterns that are put aside before training. It can be clearly seen that the conventional histogram equalization technique significantly improves the training speed while it shows low generalization capability. We believe that this is due to the partial loss of magnitude ratio information during the transformation process. On the other hand, the proposed modified histogram equalization improves the generalization capability as well as the training speed.

V. Conclusions

This paper considered a preprocessing of input data to improve the performance of the standard BP learning algorithm. In general, even a simple uniform transformation into a proper range improves the efficiency of the BP algorithm. However, uniform transformation becomes inefficient if the input distribution is very skewed as discussed in Section III. To remedy such situation, we propose a modified histogram equalization technique that converts the skewed and irregular distribution to a distribution favorable to the BP algorithm. Our simulation results support that the proposed preprocessing indeed leads to a better performance of the BP algorithm than a simple normalization or the conventional histogram equalization.

References

[1] P. J. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral science," Doctoral dissertation, Harvard University, Cambridge, MA.

[2] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, vol. 1, D.E. Rumelhart, J.L. McClelland, and PDP research group, Eds., Cambridge, Mass: MIT press, pp. 318-362, 1986.

[3] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295-308, 1988.

[4] J. R. Chen and P. Mars, "Stepsize variation for accelerating the back propagation algorithm," *Proc. of IJCNN-90*, Washington D.C., vol. I, pp. 601-604, 1990.

[5] Y. Lee, S. Oh, and M. Kim, "An analysis of premature saturation in back propagation learning," *Neural Networks*, vol. 6, pp. 719-728, 1993.

[6] A. Rezgui and N. Tepedelenlioglu, "The effect of the slope of the activation function on the back propagation algorithm," it Proc. of IJCNN-90, Washington D.C., vol. I, 707-710.

[7] R. C. Gonzalez and P. Wintz, *Digital Image Processing* (2nd Edition), Reading, Mass.: Addison-Wesley, 1987.

[8] T. Denoeux and R. Lengelle, "Initializing back propagation networks with prototypes," *Neural Networks*, vol. 6, pp. 351-361, 1993.

Table 1 A slice of S&L financial data

Bank	X1	X2	X3	X4	X5	Class
1	1,548,751	1,806,212	107,277	45,730	60,627	Failed
26	302,761,119	327,450,254	25,889,806	2,407,366	2,121,220	Failed
28	79,287	158,215	5,902	1,500	2,839	Failed
50	1,014,404	1,370,146	91,855	27,638	31,351	Survived
56	33,588,849	38,852,966	2,643,928	459,485	420,237	Survived

X1: Total Loans; X2: Total Shares and Deposits; X3: Interest Costs; X4: Total Payroll and Employee Benefits; X5: Total General and Administrative Expenses

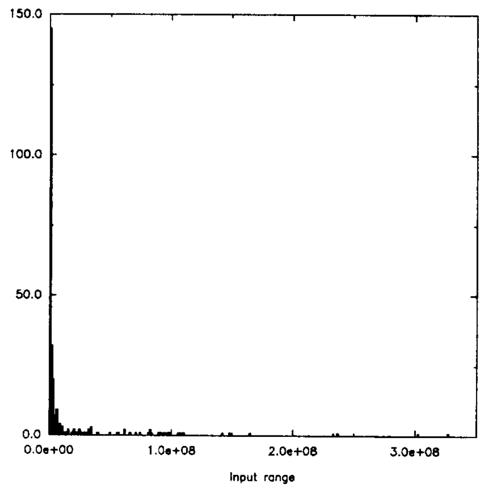

Fig. 1. Histogram of S&L failure data.

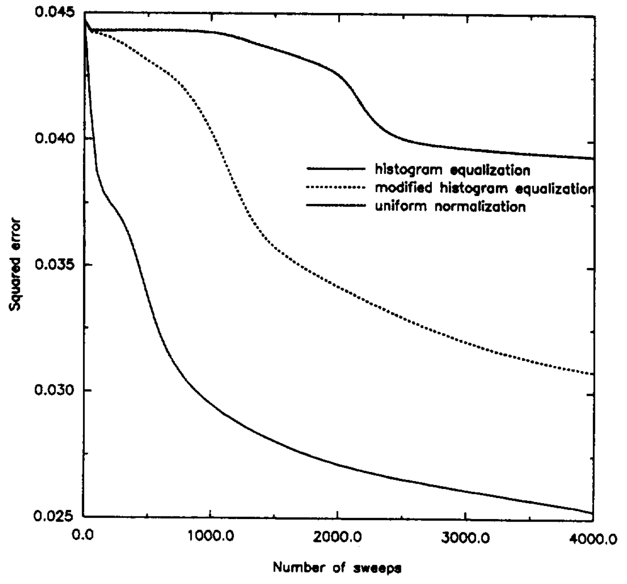

Fig. 2. Squared errors of BP training on three preprocessed data sets.

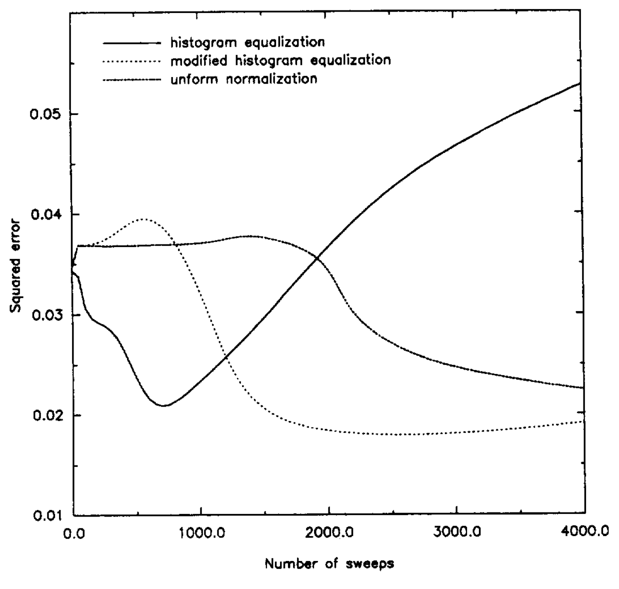

Fig. 3. Generalization testing: squared errors on three test sets.