A Byproduct of a Differentiable Neural Network-Data Weighting from an Implicit Form to an Explicit Form

Tongfeng Sun

A Byproduct of a Differentiable Neural Network — Data Weighting from an Implicit Form to an Explicit Form

Tongfeng Sun*

School of Computer Science and Technology, China University of Mining and Technology, Xuzhou Jiangsu 221116, China
Mine Digitization Engineering Research Center of the Ministry of Education, China University of Mining and Technology, Xuzhou Jiangsu 221116, China

stfok@126.com

ABSTRACT. Data weighting is important for data preservation and data mining. This paper presents a data weighting — neural network data weighting which obtains data weighting through transforming the implicit weighting of neural network to explicit weighting. This method includes two phases: in the first phase, choose a differentiable neural network whose transfer function is differentiable, and train the neural network on the ground of training samples; in the second phase, input the training samples as test samples into the network, calculate partial derivatives of the outputs with respect to inputs based on the differential characteristics of neural network, and statistical partial derivatives with respect to each input data item are used to calculate the weight of the data item. In this way, implicit weights stored in the neural network are converted to explicit weights. Experiments show that the method is more accurate than art-of-state methods. Furthermore, the method can be used in more fields, where the differentiable neural network can be used. The types of data can be discrete, continuous, or labeled, and the number of output data items is unlimited.

KEYWORDS: neural network weighting; partial derivative; implicit weighting; explicit weighting

1 Introduction

Data weighting plays a very significant role in data preservation and data mining. According to whether to need labeled information (output data) or not, data weighting is divided into two categories: unsupervised weighting and supervised weighting. Unsupervised weighting includes Maximum deviation [1], Standard deviation [2], Information entropy [3], Grey relational analysis [4], Laplacian score [5], Mutual information maximization [6], Clustering analysis (Weighting K-mean) [7], etc. These methods gain data weighting by means of the statistical analysis of input data. Recently, weighting clustering becomes a focus of research [7-8]. But the clustering analysis is often used as data sample classification model and the performance of the

clustering weighting has not been confirmed. The main problem of unsupervised data weighting is that the weighting is related to the input data self, rather than the output data. In the condition that there are the pseudo data or unrelated data, the accuracies of the methods are very low. The supervised weighting includes ReliefF [9], Fisher score [10], Trace ratio [11], Rough set [12], Simba [13], etc. These methods work based on the correlations between input data and output data. Among them, ReliefF, Fisher score and Trace ratio are similar, aiming to make the intra-class difference smaller while the inter-class difference larger. The main differences lie in the definition of data difference. Their output data are generally the labeled data, which limits its application scope. Rough set is a data dimension reduction method, which can also be used to evaluate data weighting. But the method is sensitive to the pseudo data and requires that its input data and output data all be discrete. Compared with unsupervised data weighting, the supervised weighting usually achieves better performance, but there are still great improvements in accuracy, application scope, etc.

Some machine learning algorithms, such as neural network, can be seen as weight allocation algorithms. Equivalent to the black box, data weighting is implicitly stored in the network through learning from training samples. The input-output mapping performances of the algorithms determine the accuracy of implicit weighting. But implicit weighting cannot achieve knowledge transfer. It is necessary to probe into the data weighting transfer from an implicit form to an explicit form.

This paper presents a supervised weighting method, which can be taken as a byproduct of a differentiable neural network, based on the differentiability of the neural network. Because BP neural network is a classic differentiable neural network, the paper chooses BP neural network to investigate neural network data weighting and the corresponding data weighting is named as BP NN weighting, abbreviated as BPNN. The rest of this paper is organized as follows. Section 2 introduces the basic structure of a differentiable neural network. Section 3 investigates the relationship between data weighting and data partial derivative. Further, it introduces the implementation of neural network data weighting. In Section 4, experiments are carried out to verify the performance of BPNN, and BPNN's advantages are discussed. Finally, conclusions including research prospects are presented in Section 5.

2 Structure of a differentiable neural network

In 1986, Rumelhart et al. proposed an error back propagation neural network [14], abbreviated as BP (Propagation Back) network, which is a widely used differentiable neural network. It is required that the network's neurons all be differentiable. Here BP network is chosen to introduce neural network data weighting. Based on gradient descent method, it reversely distributes the error to each unit, revises network weights, and saves learned knowledge into the data connection weights. That is to say that the trained neural network stores implicit data weighting information. As shown in Figure 1, it is supposed that the neural network is a L-layer network: one input layer, L-2 hidden layers and one

output layer. Each neuron in the network accepts the outputs of the front layer as inputs, and propagates them to the next layer.

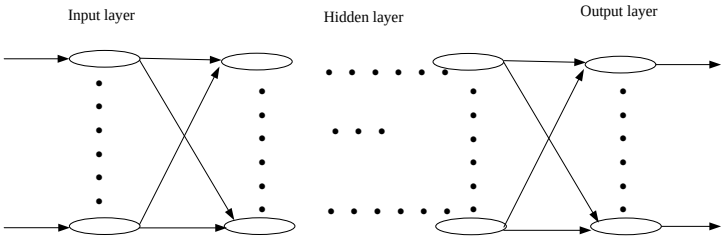

Fig. 1. A classic neural network

In the network, input data are propagated forward from the input layer to the output layer. The outputs of the *l*th layer are the inputs of the *l+1*th layer. Some assumptions are made about the structure of the network for further discussions. There are s_l neurons in the *l*th layer; $[a_1^{(l)}, a_2^{(l)}, ..., a_{s_l}^{(l)}]$, abbreviated as $a^{(l)}$, represents the inputs of the *l+1*th layer, which are also the outputs of the *l*th layer; the connection weight matrix between the *l*th layer and *l-1*th layer is labeled as $W^{(l)}$, whose element $W_{j,i}^{(l)}$ represents the connection weight between the *i*th neuron of the *l-1*th layer and the *j*th neuron of the *l*th layer; $b_j^{(l)}$ is the threshold of the *j*th neuron of the *l*th layer and the threshold vector of all the neurons of the *l*th layer is $[b_1^{(l)}, b_2^{(l)}, ..., b_{s_l}^{(l)}]$, abbreviated as $b^{(l)}$; $z_i^{(l)}$ is the weighted input sum of the *i*th neuron of the *l*th layer, and the weighted sums of all the neurons of the *l*th layer form a vector $[z_1^{(l)}, z_2^{(l)}, ..., z_{s_l}^{(l)}]$, abbreviated as $z^{(l)}$; the transfer function of all the neurons of the *l*th layer are the same, labeled as $f_l(\bullet)$. Under the assumptions, the data relationship between the adjacent layers can be represented as follows

$$z^{(l+1)} = W^{(l+1)} a^{(l)} + b^{(l+1)} \tag{1}$$

$$a^{(l+1)} = f_{l+1}(z^{(l+1)}) \tag{2}$$

3 Data weighting

3.1 Data weighting analysis

The chosen neural network is differentiable with respect to input data. By training, the neural network implicitly contains data weights. For an output data item y_j, its relationship with input data in the trained neural network is equivalent to a differentiable function, defined as follows

$$y_j = F(x_1, x_2, ..., x_n) \tag{3}$$

Its differential equation is as follows

$$dy_j = \frac{\partial y_j}{\partial x_1} dx_1 + \frac{\partial y_j}{\partial x_2} dx_2 + ... + \frac{\partial y_j}{\partial x_n} dx_n \tag{4}$$

It is supposed that Δx_i is a small increment of the *i*-th data feature. When abs(Δx_1),abs(Δx_2)…, abs(Δx_n) are small enough, Eq. (4) is approximately depicted as

$$\Delta y_j \approx \frac{\partial y_j}{\partial x_1} \Delta x_1 + \frac{\partial y_j}{\partial x_2} \Delta x_2 + ... + \frac{\partial y_j}{\partial x_n} \Delta x_n \tag{5}$$

$\frac{\partial y_j}{\partial x_i}$ implies the weight of Δx_i. From a statistical view, the greater the absolute value of the partial derivative coefficient is, the greater the weight of the data item x is. The absolute value of partial derivative shows the correlation between x and y_j.

In the *L*-th layer (output layer), the partial derivation of an output feature y_j with respect to x_i is calculated as follows

$$\frac{\partial y_j}{\partial x_i} = f_L^{'}(z_j^L) \frac{\partial z_j^{(L)}}{\partial x_i} \tag{6}$$

If *L*>2, $\frac{\partial z_j^{(L)}}{\partial x_i}$ is calculated as follows

$$\begin{aligned} \frac{\partial z_j^{(l)}}{\partial x_i} &= \frac{\partial}{\partial x_i} (\sum_{k=1}^{s_{l-1}} (W_{j,k}^l a_k^{(l-1)} + b_j^{(l)})) \\ &= \sum_{k=1}^{s_{l-1}} W_{j,k}^l f_{l-1}^{'}(z_k^{(l-1)}) \frac{\partial z_k^{(l-1)}}{\partial x_i} \end{aligned} \tag{7}$$

The equation can be iteratively calculated until *l*=2.

If *L*==2, $\frac{\partial z_j^{(L)}}{\partial x_i}$ is equal to

$$\frac{\partial z_j^{(2)}}{\partial x_i} = \frac{\partial}{\partial x_i} (\sum_{k=1}^{s_1} (W_{j,k}^2 a_i^{(1)} + b_j^{(2)})) \tag{8}$$

Because $a_i^{(1)}$ is x_i in the input layer, $\frac{\partial z_j^{(2)}}{\partial x_i}$ is equal to

$$\frac{\partial z_j^{(2)}}{\partial x_i} = W_{j,i}^{(2)} \tag{9}$$

Through Eqs. (7)-(9), $\frac{\partial z_j^{(L)}}{\partial x_i}$ is obtained. Then $\frac{\partial y_j}{\partial x_i}$ is calculated according to Eqs. (6).

Assume that a trained neural network can accurately reflect the relations between input data and output data. All the training sample data as testing sample data are input into the trained network again and the partial derivatives are calculated at different points. It is supposed that there are p training samples with m output features. Then, the trained neural network is equivalent to a combination of m data-mapping functions $[F_1, F_2, ..., F_m]$. The data weight of x_i is calculated as follows

$$\omega_i = \frac{\sum_{k=1}^{p}\sum_{j=1}^{m} abs(\frac{\partial y_j}{\partial x_i}(x^k, y^k))}{\sum_{i=1}^{n}\sum_{k=1}^{p}\sum_{j=1}^{m} abs(\frac{\partial y_j}{\partial x_i}(x^k, y^k))} \tag{10}$$

where $abs(\bullet)$ is an absolute function. $abs(\frac{\partial y_j}{\partial x_i}(x^k, y^k))$ indicates the saliency (sensitivity) of x_i in the conditions that the chosen important point is (x^p, y^p) and the output data feature is y_j; $\sum_{k=1}^{p}\sum_{j=1}^{m} abs(\frac{\partial y_j}{\partial x_i}(x^k, y^k))$ indicates the saliency of x_i in the conditions that all the training samples and all the output features are considered.

3.2 Data weighting implementation

BPNN data weighting is based on a trained neural network. Consequently, its implementation consists of two phases.

(1) In the first phase, train a BP neural network The phase is to establish an accurate relationship between input data and output data. Neural network is trained based on training sample data. It learns from the training samples and implicitly stores data weighting into neural network connection weights and neuron thresholds.

(2) In the second phase, analyze data weighting based on the trained neural network Calculate the partial derivatives with respect to input data for all the training sample data. Statistical partial derivatives of each input data item reflects its weight. In this way, implicit weights stored in the neural network are converted to explicit weights.

4 Experiments

The chosen weighting methods include Standard deviation (SD), Weighting K-mean (WKmeans) [7], ReliefF [9], Simba [13] and BPNN. These data weighting methods are tested on object recognition of low dimensional data and high dimensional data in different datasets. Furthermore, the methods are investigated in the condition that outputs are continuous. Finally, the advantages of BPNN are discussed.

4.1 Low dimensional data weighting

Low dimensional data weighting experiments are carried out in UCI repository database [15]. Here we choose Liver Disorders (Liver), Glass Identification (Glass), Iris and Wine datasets from the repository. As shown in Table 1, the dimensions for all the data in above datasets are less than 20. In the calculation of BPNN data weighting, the dropout factor of the neural network BPNN is 0.5, and the batch-size is 20. 75% samples are randomly chosen from the data sets as training samples, and the remaining are testing samples. The architectures of neural network hierarchy is $[n \quad 200 \quad m]$, where n refers to the number of input data dimensions, and m refers to the number of output labels. The performances of the chosen data weighting methods are directly presented through a *k*-NN [16] classifier with *k* equal to 3. *k*-NN classifies data according to data difference. The accurate data weighting means the low classification error rate.

Table 1. Description of the low-dimensional data sets

Name	Samples	Dimensions	Classes
Liver	345	7	2
Glass	214	10	6
Iris	150	4	3
Wine	178	13	3

Table 2. Classification error ratios in low dimensional datasets

	SD	WKmeans	ReliefF	Simba	BPNN
Liver	0.372 (0.0453)	0.351 (0.0378)	0.386 (0.0585)	0.410 (0.0543)	0.347 (0.0210)
Glass	0.023 (0.0131)	0.130 (0.0532)	0.134 (0.0586)	0.025 (0.0240)	0.065 (0.0604)
Iris	0.049 (0.0271)	0.048 (0.0282)	0.155 (0.0869)	0.068 (0.0405)	0.046 (0.0422)
Wine	0.308 (0.0666)	0.048 (0.0186)	0.059 (0.0372)	0.044 (0.0157)	0.000 (0.000)

Table 2 shows the results of different data weighting methods, i.e. the average test error ratio followed by the standard deviation in parentheses. The results are calculated based on experimental data for 20 times. Among the weighting methods, BPNN has the lowest classification error rates in Liver, Iris and Wine datasets, and has the third lowest classification error rate in Glass dataset. Although ReliefF and

Simba are two supervised weight methods, they are not be significantly improved compared with unsupervised weight methods. The overall performance of ReliefF is not ideal and is even inferior to unsupervised method: SD and WKmeans. On statistical grounds, there are no significant differences in performance. The lowest classification error ratio means the highest performance. So, BPNN has the best performances among the five metrics.

4.2 High dimensional data weighting

Experiments are carried out in Mnist database [17] and face database [18-20], which are all image databases. Through *k*-NN classification algorithm with the parameter *k* equal to 3, a more accurate data weighting produce a higher recognition accuracy. The Mnist database is a large handwritten digit image database, containing 60,000 training images and 10,000 test images. The handwritten digit images are of size 28×28, and are converted into 1×784 vectors (high dimension data) in intelligent recognition. 10% samples are randomly chosen from training samples and testing samples respectively in each experiment of Mnist databases. The chosen face datasets are all public database, including AR, ORL, Indian (female and male). To reduce the computation complexity, all the face images are reduced to 32×32 and are converted to 1×1024 vectors. The architectures of neural network hierarchy is $[n \quad 1000 \quad 1]$, where n refers to the number of input data dimensions. The output is a digital label.

The results are calculated based on experimental data for 20 times. Table 3 shows the results of different data weighting methods for high dimensional data, i.e. the average test error ratio followed by the standard deviation in parentheses. It can be seen that BPNN is superior to other methods. BPNN has the lowest recognition error ratios in Mnist, ORL and Indian Male databases, the second lowest in Indian Female database and the third lowest in AR database. And its recognition error ratios in Indian Female and AR database are only a little greater than the lowest recognition error ratios. As a whole, Simba is in the second place, a little better than SD, WKmeans and ReliefF. ReliefF is not ideal and is even inferior to unsupervised methods: SD and WKmeans. In high-dimensional databases, WKmeans sometimes cannot work because its weighting optimization may produce very small values which go beyond the range of computer representation and lead to computation failures. The experimental results shows that BPNN can achieve better performance for high dimensional data.

Table 3. Recognition error ratios in high dimensional databases

	SD	WKmeans	ReliefF	Simba	BPNN
Mnist	0.095 (0.015)	Cannot work	0.081 (0.0097)	0.091 (0.005)	0.079 (0.005)
AR	0.230 (0.013)	0.435 (0.017)	0.614 (0.036)	0.240 (0.007)	0.281 (0.015)
ORL	0.115 (0.025)	0.119 (0.031)	0.139 (0.031)	0.169 (0.033)	0.103 (0.031)
Indian Female	0.28 (0.0197)	0.186 (0.026)	0.219 (0.034)	0.156 (0.027)	0.182 (0.029)
Indian Male	0.559 (0.028)	0.352 (0.028)	0.454 (0.011)	0.336 (0.052)	0.312 (0.033)

4.3 Data weighting for continuous outputs

Traditional data weighting is used in data classification/recognition in which the output data are labeled data. If the output data are continuous, the output must be discretized into labeled data. In this way, useful information will be lost. BPNN data weighting can be used in more complex conditions. For example, BPNN is used in the condition that output items are continuous. In this condition, BPNN does not need to transform continuous output data to labeled data. Here we just verify BPNN in a condition that there is a continuous output data item. An indirect method is adopted to verify the validities. The verification is based on a hypothesis: data weighting helps to enhance useful contents and suppress useless contents. Because the purpose of PCA learning is to reduce data dimension while trying to preserve data information, PCA data extracted from the data weighted with more accurate weight factors will contain more useful contents and can be mapped to more accurate outputs.

Experiments are carried out in two datasets. One dataset is Concrete dataset from UCI repository. The data consist of nine items, including 1 output item and 8 input items. Every item is continuous. Another dataset is a self-built dataset. The data in the dataset consist of 21 items, including 1 output item (tagged as y) and 20 input items (tagged as $X = [x_1, x_2, \ldots ..x_{20}]$), among which $x_1 = [0.0025: 0.0025: 1]$, $x_2 = rem((1: 1: 400), 2)$ (*rem* is a function keeping remainder after division), x_3 is $rem((1: 1: 400), 5) / 5$, $x_4 = rem((1: 1: 400), 3) / 3$, and $x_5 \sim x_{20}$ are all 400 random numbers between 0 and 1. The relationship between input items and output item is described as follows

$$y = 4 \times x_1 + 3 \times x_2 + 5.3 \times sqrt(6 \times x_3) + 5 \times \sin x_4 + 3.5 \times (x_5 + 0.6)^2 + 6.3 \times x_6 \quad (11)$$

There are 400 samples in the self-built dataset. The output item is continuous, and the items $x_7 \sim x_{20}$ are irrelevant to the output item. In each experiment, 80% samples are randomly chosen for training, and the remaining samples are used as test samples.

Experiments are carried out as follows: determine data weighting (output item is uniformly discretized into three labels when needing labels), add data weighting to the original data, extract PCA data of the weighted data, complete data regression through SVM based on chosen training PCA data and output data, and calculate average regression errors of test samples. A smaller error implies that the PCA data keep more useful information. The average absolute regression errors indirectly show the performances of date weightings. In PCA extraction, set the fixed eigenvalue ratio 96%, and the experimental results are shown in Table 4.

Table 4. Regression errors for fixed eigenvalue ratio 96%

	SD	WKmeans	ReliefF	Simba	BPNN
Concrete	16.185 (2.711,5)	4.171 (0.360,6)	10.093 (0.412,4)	Cannot work	4.378 (0.371,4)
Self-built database	10.544(0.505,19)	1.111(0.202,0.23)	3.428 (0.103,16)	1.361 (0.242,4)	0.357 (0.022,4)

According to Table 4, BPNN is the best method with the smallest regression errors from the view of overall performance. ReliefF and Simba. ReliefF and Simba are not

suitable for continuous data output because these methods require that the output data must be labeled. Continuous data discretized as labeled data will also lose a lot of information. Simba may not work in some conditions. For example, if minimum intra-class difference is larger than the minimux inter-class difference, Simba algorithm will fail to work.

4.4 BP algorithm advantages

According to the experimental results in Section 4.1-4.3, BPNN maintains the best performances, better than state-of-the-art algorithms, such as SD, WKmeans, ReliefF and Simba. Especially when output data are continuous, BPNN shows more excellent performances. It can be concluded that BPNN weighting has greater practical significance and can achieve better results. In actual applications, BP weighting can be used in more complicated conditions. Traditional data weighting analysis mainly are mainly used in the object classification/recognition where the output data are labeled data or looked as labeled data. Some methods even require that the input data should also be discrete or labeled. BPNN weights can effectively overcome these limitations: the number of the output data items is unlimited, and the output type can be discrete, continuous or labeled. That is to say BPNN can be widely used in different conditions where BP neural network can work.

5 Conclusion

This paper presents a new data weighting method, abbreviated as BPNN, which is a byproduct of differentiable neural network — BP neural network. Based on trained neural network, this method transforms implicit weights into explicit weights through partial differentials. Experiments show that this method is more stable and accurate, and have a wide application scope. BPNN is closely related to neural network's performance. New developments of neural network will improve neural network and provide more accurate data weighting. In scientific research field, the neural network has been a hot research topic for a long time. Data preprocessing based on neural network, such as data weighting, will be an exciting research direction.

Acknowledgment

This paper is jointly supported by the National Natural Science Foundation of China (No. 61379101) and the China Postdoctoral Science Foundation (No. 2016M601910).

Reference

1. Xu Z., Zhang X., Hesitant fuzzy multi-attribute decision making based on TOPSIS with incomplete weight information. Knowledge-Based Systems, vol. 52 no. 6, pp. 53-64 (2013)

2. Wang Y.M., Using the method of maximizing deviations to make decision for multi-indices. Journal of Systems Engineering & Electronics, vol. 8 no.3, pp. 24-26 (1998)
3. Cament L.A., Castillo L.E., Perez J.P., Galdames F.J., Perez C.A., Fusion of local normalization and Gabor entropy weighted features for face identification. Pattern Recognition, vol. 47 no.2, pp. 568-577 (2014)
4. Wei G., Grey relational analysis method for 2-tuple linguistic multiple attribute group decision making with incomplete weight information. Expert Systems with Applications, vol. 38 no.5, pp. 4824-4828 (2011)
5. Zhu L., Miao L., Zhang D., Iterative Laplacian Score for Feature Selection. Chinese Conference on Pattern Recognition, pp. 80-87, Springer, Berlin, Heidelberg (2012)
6. Peng H., Long F., Ding C., Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27 no.8, pp. 1226–1238 (2005)
7. Huang J. Z., Ng M. K., Rong H., Li Z., Automated Variable Weighting in k-Means Type Clustering. IEEE Transactions on Pattern Analysis And Machine Intelligence, vol. 27 no. 5, pp. 657-668 (2005)
8. Gan G., Chen K., A soft subspace clustering algorithm with LOG-transformed distances. Big Data and Information Analytics. American Institute of Mathematical Sciences, vol. no. 1, pp. 93-109 (2016)
9. Kira K., Rendell L., A practical approach to feature selection. in: International Conference on Machine Learning, Aberdeen, Scotland,UK, pp 249-256 (1992)
10. Duda R.O., Hart P.E., Stork D.G., Pattern Classification. 2nd edition. JohnWiley & Sons, NewYork, USA (2001)
11. Nie F.P., Xiang S.M., Jia Y.Q., Zhang C.S., Yan S.C., Trace ratio criterion for feature selection. In: The Association for the Advancement of Artificial Intelligence, Chicago, IL,USA, pp 671-676 (2008)
12. Greco S., Matarazzo B., Slowinski R., Rough sets theory for multicriteria decision analysis. European Journal of Operational Research, vol. 129 no. 1, pp. 1-47 (2001)
13. Ran G.B., Navot A., Tishby N., Margin based feature selection – theory and algorithms. in: Proceedings of the Twenty-first International Conference on Machine Learning, ACM, New York, NY,USA, pp. 43-50 (2004)
14. Rumelhart D.E., McClelland J.L., Parallel distributed processing: exploration in the microstructure of cognition. MIT Press Cambridge, vol.2, MA, USA (1986)
15. Frank A., Asuncion A, UCI machine learning repository. http://archive.ics.uci.edu/ml (2010)
16. Cover T., Hart P., Nearest neighbor pattern classification. IEEE Trans. Inf. Theory, vol. 13 no. 1, pp. 21–27 (1967)
17. LeCun Y., Bottou L., Bengio Y., Haffner P., Gradient-based learning applied to document recognition. Proceedings of the IEEE, vol. 86 no. 11, pp. 2278-2324 (1998)
18. ORL. http://www.cl.cam.ac.uk/research/dtg/ attarchive/ facedatabase.html (2012)
19. Jain V., Mukherjee A., The Indian Face Database. http://vis-www.cs.umass.edu/~vidit/IndianFaceDatabase/ (2017)
20. A.M. Martinez, R. Benavente. The AR Face Database. CVC Technical Report. http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html (1998)