2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)
July 6-11, 2014, Beijing, China

A Structure Optimization Algorithm of Neural Networks for Large-Scale Data Sets

Jie Yang, Jun Ma, Matthew Berryman, Pascal Perez
SMART Infrastructure Facility, Faculty of Engineering and Information Sciences
University of Wollongong, Northfields Av, Wollongong, NSW 2522, Australia
E-mails: {jiey, jma, mberryma, pascal}@uow.edu.au

***Abstract*— Over the past several decades, neural networks have evolved into powerful computation systems, which are able to learn complex nonlinear input-output relationship from data. However, the structure optimization problem of neural network is a big challenge for processing huge-volumed, diversified and uncertain data. This paper focuses on this problem and introduces a network pruning algorithm based on sparse representation, termed SRP. The proposed approach starts with a large network, then selects important hidden neurons from the original structure using a forward selection criterion that minimizes the residual output error. Furthermore, the presented algorithm has no constraints on the network type. The efficiency of the proposed approach is evaluated based on several benchmark data sets. We also evaluate the performance of the proposed algorithm on a real-world application of individual travel mode choice. The experimental results have shown that SRP performs favorably compared to alternative approaches.**

***Index Terms*— Neural-Network Pruning; Neural-Network Structure Optimization; Sparse Representation; Large-Scale Data Set**

I. Introduction

Neural networks (NNs) have increasingly found wide applications over the decades. The research spans several fields such as local weather forecast [1], automatic target detection in images [2], and medical tissue classification [3]. This wide applicability has motivated growing interest on efficient methods enabling their convergence and generalization ability. Although there have been some fundamental and groundbreaking advances in this field, many problems remain unsolved to date, such as processing the high-dimensional data or large number of training samples. In the last decades, the amount of information generated by acquisition devices is always huge and ever-growing. Thousands of images and transaction sale records are generated every day. This typically leads to large-scale or high-dimensional data that exceeds the processing capacity of the most sophisticated neural network learning algorithms. Moreover, the structure of a neural network is heavily affected by the data structure and the data quality. The varieties and uncertainties of data samples make it even harder to effectively build extensible network structure for high-dimensional and large-scale data.

In this paper, we are interested in the structure optimization to improve the learning capability of NNs for big data processing. Before applying neural network learning to any specific problems, one is faced with the question of determining the most suitable structure, such as the number of hidden neurons and layers. When it comes to large data sets, the training performance obtained from different network structures may dramatically differ from each other, not to mention the training time. Hence, designing the advanced structure optimization algorithm for large data sets that saving training cost and enhancing the generalization ability is the primary motivation of this paper.

A network pruning algorithm is then introduced by searching for the sparse representation of a network structure. The proposed approach, termed sparse representation neural network pruning (SRP), is capable of iteratively building up the network structure, while minimizing the residual training error. Precisely, in SRP hidden neurons which minimize the residual error to the desired output are selected at each iteration. The selection in SRP can be regarded as a forward selection. The proposed selection is different from traditional pruning methods which adopt backward elimination strategy to prune redundant neurons from the original architecture and could be time-consuming particularly for large networks.

Furthermore, the SRP algorithm has no constraints on the type of neural networks and their transfer functions, and can be easily extended to the multiple-layer structure. The proposed method also requires no prior knowledge about the smoothness of the object function, in contrast to some techniques which are strongly dependent on the assumed smoothness [4].

The remainder of the paper is organized as follows. Section II and Section III present a brief review of existing network pruning algorithms and sparse representation, respectively. Section IV details the proposed algorithm by analyzing the relationship between the structure optimization and the sparse representation. Section V discusses implementation issues and experimental results, followed by concluding remarks in Section VI.

Note that the sample data used for this paper is from [5], with the exception of the Household Travel Survey (HTS) data, which unfortunately we can't release, and source code repository for the software developed for this paper is located at `https://github.com/smart-facility/srp`.

II. Network pruning algorithms: a review of the literature

Neural network has become very popular in recent years. Given training data samples, the neural network algorithm is used to establish the potential model between the observed

978-1-4799-2072-3/14/$31.00 ©2014 IEEE

input and output samples. One of the commonly applied neural network is the multilayer perceptron (MLP), with several layers (such as the input, hidden, and the output layers) of perceptron neurons. The hidden neurons are fully connected to input and output neurons by the network weights, which are updated by a training algorithm subject that reducing the error between the actual and desired network output. Before training the neural network, one is faced with the question of determining a suitable network structure. A variety of algorithms have been developed to search for an optimal network structure, which can be broadly categorized as follows:

- *network pruning*, which is based on a saliency analysis of each element (weights or hidden neurons) [6]–[12];
- *network construction*, which begins with a small network and incrementally adds hidden neurons during the training process [13], [14];
- *evolutionary algorithms*, which are based on evolutionary search strategy such as genetic algorithms [15].

In this paper, we mainly focus on network pruning algorithms. A general framework for network pruning has three steps:

step 1: set up a large neural network architecture and train it with any learning method (e.g. back-propagation algorithm), until the stopping criterion is met;

step 2: compute the significance for all elements and eliminate the least important ones;

step 3: retrain the pruned network; if the change in outputs between the original and pruned network is small enough, then go to step 2; otherwise stop.

The network pruning algorithms can be further classified into two categories: weight pruning and hidden neuron pruning. Examples of weight pruning algorithms include *Optimal Brain Damage* (OBD) [6], [16], *Optimal Brain Surgeon* (OBS) [7], and *Magnitude-based pruning* (MAG) [8]. Hidden neuron pruning algorithms include *Skeletonization* (SKEL) [10], *non-contributing units* (NC) [11], and *Extended Fourier Amplitude Sensitivity Test* (EFAST) [12]. Below we describe all of these algorithms.

In OBD [6], [16], rather than temporarily deleting a neuron and evaluating the whole objective function, this method constructs a local model of the error function to analytically predict the effect on the objective function. This then, with some other assumptions to make the problem efficient, computationally, provides a measure of saliency that is used to remove weights in order from smallest saliency up to some bound. In OBS [7], a larger set of the local information is used, specifically the full off-diagonal information from the Hessian matrix $\mathbf{H}$, and to make this fast a recursion relationship for calculating $\mathbf{H^{-1}}$.

In MAG [8], the algorithm removes both neurons and weights, using either a formula representing either

1) The energy of a neuron, as defined by the strength of the output times its influence, or
2) Also including its contribution to output in the next layer.

In SKEL [10], the saliency of a neuron is measured by the training error when the neuron is removed. The NC method [11] searches for non-contributing hidden neurons by checking their effect on the network output. During the pruning process, the neurons with least change for training error are removed in NC and SKEL, which are both derivatives of the OBD algorithm, but for pruning neurons rather than weights. As for the EFAST algorithm [12], the importance of neurons is evaluated using an amplitude sensitivity test, which is a variance-based global sensitivity analysis method.

In short, conventional pruning algorithms attempt to estimate the sensitivity of the network elements (weights or hidden neurons). Thus, the significant elements which minimize the training error are maintained, while others with the least contribution are removed. The pruning-based methods benefit from the fast convergence of the initial large network to the local minimum. In spite of this, the pruning-based methods are time-consuming with large computational complexity. In Section V, we will compare the existing SKEL, NC, and EFAST methods with the proposed SRP algorithm.

III. Sparse representation

This section briefly surveys the sparse representation and multiple measure vector (MMV) model [17], [18]. Given an original signal and a dictionary consisting of basis vectors, the sparse representation addresses a compact approximation for the given signal using basis vectors. If the given signal is a two-dimensional matrix, the MMV model is used to find a joint-sparse representation of several signals simultaneously, or a sparse matrix representation. The MMV model can be stated as follows:

$$\min \ S(X) \quad \text{subject to} \quad Y = \mathcal{D}X, \tag{1}$$

where $Y \in \mathbb{R}^{M\times L}$, $\mathcal{D}$ is a dictionary, $X \in \mathbb{R}^{N\times L}$ and $S(X)$ denotes the matrix norm. The principle of the MMV model is that all the columns from X possess the same sparsity profile; that is, the columns of X share the indices of nonzero elements. Let $\boldsymbol{r}_i$ be the i-th row of the matrix X. Furthermore, let us define a vector $\boldsymbol{s}$ of length N whose i-th element s_i is given by:

$$s_i = \|\boldsymbol{r}_i\|_p , \tag{2}$$

where $\|\cdot\|_p$ denotes the vector $p-$norm $(0 \le p \le 1)$. One simple strategy to measure the matrix sparsity is to use the l_0 pseudo-norm of the vector $\boldsymbol{s}$:

$$S(X) = \|\boldsymbol{s}\|_0 . \tag{3}$$

However, the l_0-based optimization is computationally expensive because an exhaustive enumeration is required in terms of all possible locations of nonzero rows in X. In addition, the computational complexity grows exponentially with the size of the measurement matrix. Therefore, rather than using the l_0 pseudo-norm, we replace it with the l_1 norm. The MMV model then is stated as

$$\min \ S(X) = \|\boldsymbol{s}\|_1 \quad \text{subject to} \quad Y = \mathcal{D}X. \tag{4}$$

In [19], the authors proved that the minimization problem in (4) is equivalent to that based on (3) when the sparsity of the solution X is sufficiently small.

IV. Problem formulation and the SRP algorithm

In this section, we formulate the network structural optimization problem as a multiple measure vector (MMV) model. Consider a three-layer fully-connected network with one hidden layer, an input layer and an output layer. Suppose the initial network architecture consists of Q inputs, N hidden neurons and M outputs. Let $P = [\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_L]$ be a matrix containing L training patterns and $Y = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_L]$ be the desired output matrix; each pair ($\boldsymbol{p}_i$, $\boldsymbol{y}_i$) forms an input vector and the corresponding desired output vector. Moreover, let $X \in \mathbb{R}^{N \times L}$ denote the output matrix of the hidden layer, in which the i-th row represents the output from the i-th hidden neuron, $i = 1, 2, ..., N$, and $Z \in \mathbb{R}^{M \times L}$ denote neural network output matrix corresponding to the input matrix P. The training of neural network can be expressed in matrix form as follows:

$$\begin{aligned} X &= f\left(VP + B_1\right), \\ \text{and} \quad Z &= g\left(WX + B_2\right). \end{aligned} \tag{5}$$

where V is the weight matrix between the input and hidden layers, W comprises the weight vectors between the hidden and output layers, B_1 and B_2 is the bias matrix of the input and output layer with columns containing the bias vector, respectively. Without loss of generality, we further assume that $g\left(\cdot\right)$ is a linear activation function. Note that if $g\left(\cdot\right)$ is an invertible function, we can transform the output neurons to linear units by applying the inverse function $g^{-1}\left(\cdot\right)$. In this case, the actual output of the network can simply be expressed as

$$Z = WX + B_2. \tag{6}$$

Given the desired output matrix Y, Eq. (6) can also be rewritten as:

$$Y = Z + E = WX + B_2 + E, \tag{7}$$

where $E = [\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_L]$ is the network error matrix; $\boldsymbol{e}_i$ is the error between the actual output $\boldsymbol{z}_i$ and the desired output $\boldsymbol{y}_i$.

The objective of the pruning algorithm is to reduce the number of hidden neurons. Removing the i-th hidden neuron is equivalent to setting its output to zero, i.e., the i-th row of X becomes a null vector. Therefore, the pruning process is equivalent to minimizing the number of nonzero rows in X, or the matrix sparsity. The structural optimization problem then can be cast as the following MMV model:

$$\min \; S(X) \quad \text{subject to} \quad \left\| \tilde{Y} - WX \right\|_2 \leq \epsilon, \tag{8}$$

where $S(X)$ is a matrix sparsity measure, $\tilde{Y} = Y - B_2$, W is the weight matrix between the hidden and output layers, and ϵ is a noise bound.

Comparing Eq. (8) with the MMV model of Eq. (4), we can see that W serves as the dictionary $\mathcal{D}$ and the matrix $\tilde{Y}$ plays the role of the measurements. Thus, the network structural optimization can be regarded as finding the sparse representation for the matrix X. As a result, by selecting important hidden neurons, the output matrix X becomes a sparse matrix in which most rows have zero values.

To solve the MMV model, we apply the M-OMP algorithm presented in [17] because of its simplicity and efficiency. The M-OMP algorithm starts from a null solution and iteratively adds one nonzero element. In addition, the M-OMP algorithm will not select the same atom twice. Based on the above properties of M-OMP, at the t-th iteration, the SRP algorithm selects t different hidden neurons from the original network. The outputs from selected neurons are then accumulated to approximate the desired output. Thus, the representation error is expected to decrease when more neurons are involved.

After minimizing the sparsity of the matrix X, we also need to update the network parameters between the input and hidden layer. Let $\widehat{V} = [V, \quad \boldsymbol{b}_1]$ be the matrix of weights and biases of the hidden layer, and $\widehat{P}$ be the augmented input matrix:

$$\widehat{P} = \begin{bmatrix} P \\ \boldsymbol{I}^T \end{bmatrix},$$

where $\boldsymbol{I}^T$ is the L-dimensional row vector whose elements are equal to 1. The output matrix of the hidden layer can then be rewritten as

$$X = f\left(VP + B_1\right) = f(\widehat{V}\widehat{P}), \tag{9}$$

Assuming the activation function of the hidden layer ($f(\cdot)$) is invertible, then the matrix $\widehat{V}$ can obtained by solving the following least squares problem:

$$\widehat{V} = \arg\min_{\widehat{V}} \left\| f^{-1}(X) - \widehat{V}\widehat{P} \right\|_2. \tag{10}$$

where $f^{-1}(\cdot)$ represents the inverse of the activation function $f(\cdot)$.

The proposed SRP algorithm is then summarized in Algorithm 1. The proposed algorithm is similar with the conventional pruning-based approaches in that both the SRP and pruning-based methods require network training in the first place. The trained network provides a candidate structure for the subsequent pruning. The difference between the SRP and conventional pruning-based algorithms comes from the optimization strategy. The proposed SRP algorithm selects important elements (considered as forward selection) whereas the traditional pruning algorithms adopt the backward elimination strategy to remove redundant hidden neurons. Although we are conducting forwards selection, we are removing neurons as the SRP algorithm iterates, thus SRP is still a pruning method. Furthermore, the termination criterion employed by the SRP method is predefined using the error from the validation data set. That is, the SRP-based pruning is terminated if the validation error increases for three successive iterations. This simple strategy works based on the assumption that an increase in error from the validation set indicates the start of overfitting.

input : A trained network and maximum iteration T;
output: The sparse matrix X^*;

set $\mathcal{D} = W$, $E(0) = \widehat{Y} = Y - B_2$;
for $t = 1$ **to** T **do**
  Update X by solving the MMV model in Eq. (8):

$$(X(t), E(t)) = \mathrm{M} - \mathrm{OMP}(\mathcal{D}, E(t-1), t),$$

  where M-OMP denotes the M-OMP Algorithm [17]

  Update the weight matrix V using Eq. (10);

  Evaluate the updated network using validation data set;

  if *the predefined termination condition is met* **then**
    break
  end
end
return $X^* = X(t)$;

Algorithm 1: SRP algorithm for structural optimization.

V. Experimental result

This section discusses the performance of the proposed SRP algorithm on typical classification problems. Two experiments are conducted. The first one aims to compare the performance of the SRP algorithm with other network pruning methods. The second one applies the SRP algorithm to a real application of travel mode choice modeling.

A. Performance of SRP on typical classification problems

Without loss of generality, a three-layer fully-connected feed forward networks are employed. The activation function between the input and hidden layer is set to standard sigmoid function and to the linear transfer function for the output neuron. The bias vector is implemented as an incoming connection from the particular bias node with constant non-zero input (i.e., it has been set as a vector whose elements are equal to 1). The network is initialized with random weights in the range $[-0.1$, $0.1]$, and it is trained with the resilient back-propagation algorithm (RPROP) [20]. The training parameters are set as follows: the maximum number of training iterations is 500; the minimum performance gradient is 10^{-6}; the learning rate is 0.01. The network training terminates when either the maximum number of iterations is reached or the performance gradient falls below 10^{-6}. The generalization performance is evaluated using the classification error. Three benchmark problems from Proben [5] are employed. A partitioning into training, validation, and test set is also given for each dataset. The size of the training, validation, and test set in all cases is 50%, 25%, and 25% of all examples, respectively. Their details are detailed in Table I.

In this subsection, the proposed SRP algorithm is compared with conventional pruning algorithms. Three existing neuron-pruning methods are used for comparison: SKEL [10], NC [11], and EFAST [12]. To make the comparison fair, we implement existing neuron-pruning methods using the same stopping criterion. In other words, the pruning is terminated if the validation error increases for three successive iterations. All the networks are initialized with 128 hidden nodes.

TABLE I
Employed benchmark data set for network pruning.

Data set	Input	Output	Number of samples
Cancer	9	2	699
Card	51	2	690
Diabetes	8	2	768

Table II presents the classification errors of the proposed SRP algorithm and conventional pruning methods. The proposed SRP approach achieves the lowest error for all data sets. Figure 1 presents the Receiver Operating Characteristic (ROC) curves of the SRP algorithm and those of existing methods for the three classification problems. The results also show that the proposed algorithm leads to a better classification rate.

Tables III and IV show, respectively, the number of remaining neurons, running time of the SRP and traditional pruning algorithms, and the time taken for the resulting network to classify the test set (25% of the number of samples). Note that the running time includes the time for training of the neural network for all iterations, upon which the pruning relies for each iteration. Given the need for some sort of algorithm to determine the structure of the neural network, we are after the fastest pruning algorithm that also produces the most efficient network, i.e. number of neurons, which is the neural network that is fastest on the test set, as indicated in Table V.

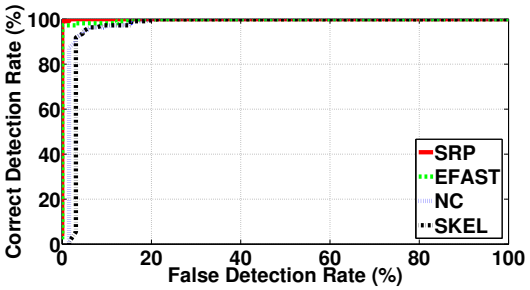
For the pruned structure, on average SRP employs less hidden neurons than other pruning method, i.e., NC (80.7), SKEL (41.4), EFAST (8.4), respectively. Moreover, SRP requires the least time to converge. The reason is that SRP selects the most significant elements from the trained network. This method can be regarded as a forward selection instead of backward elimination, hence fast convergence is expected. By contrast, NC, SKEL, and EFAST methods spend much longer time on pruning unimportant hidden neurons. The resulting network is also more efficient in terms of number of neurons, and hence test time.

TABLE II
Classification error rate on the test sets from various pruning algorithms.

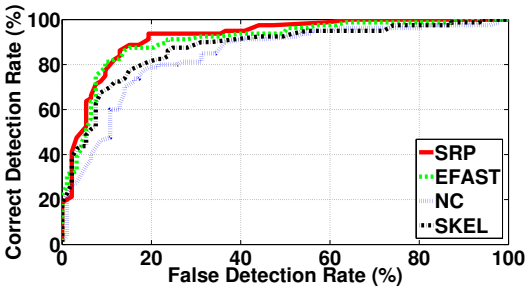
Data set	Classification error (%)			
	NC	SKEL	EFAST	SRP
Cancer	2.56±1.23	2.83±1.13	2.53±1.33	**1.35±0.37**
Card	19.62±2.43	15.81±3.22	18.21±2.91	**15.23±2.38**
Diabetes	28.11±2.13	26.51±2.43	24.59±2.61	**23.27±3.23**
Average	16.76±1.93	15.05±2.26	15.11±2.28	**13.28±1.99**

B. Travel mode choice modeling using SRP

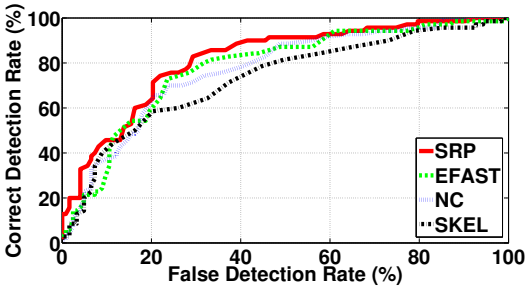
In this section, the proposed SRP algorithm is applied to build up a travel mode choice model. Community travel mode choice modeling is a typical classification problem. It can



(a) "Cancer" data set



(b) "Card" data set



(c) "Diabetes" data set

Fig. 1. Receiver Operating Characteristic (ROC) for pruning algorithms.

TABLE III

REMAINING NEURONS FOR DIFFERENT PRUNING METHODS.

Data set	NC	SKEL	EFAST	SRP
Cancer	102.3±10.0	105.6±8.9	6.8±1.2	**2.6±0.3**
Card	22.0±2.5	**4.3±1.9**	11.2±2.3	6.8±0.6
Diabetes	15.8±2.3	5.3±1.8	7.2±1.4	**4.9±0.3**
Average	80.7±4.9	41.4±4.2	8.4±1.6	**4.7±0.4**

TABLE IV

PRUNING TIME (SECONDS) FOR VARIOUS PRUNING METHODS.

Data set	NC	SKEL	EFAST	SRP
Cancer	92.5±23.0	12.6±5.9	8.8±3.2	**0.04±0.01**
Card	420.0±122.3	63.3±21.9	18.2±5.3	**0.05±0.02**
Diabetes	321.8±72.1	45.3±7.8	10.2±3.4	**0.05±0.01**
Average	278.1±72.5	40.1±11.9	12.4±4.0	**0.05±0.02**

TABLE V

TEST TIME (SECONDS) FOR THE NEURAL NETWORKS PRODUCED BY THE DIFFERENT PRUNING METHODS.

Data set	NC	SKEL	EFAST	SRP
Cancer	0.330±0.032	0.341±0.029	0.022±0.004	**0.008±0.001**
Card	0.450±0.051	**0.088±0.039**	0.229±0.047	0.139±0.012
Diabetes	0.064±0.009	0.022±0.007	0.029±0.006	**0.020±0.001**

be used as evidence for transportation service planning and road network optimization. Travel mode choice model is built on individual demographic profiles and travel records that are collected through household or individual travel survey. Indicators commonly used in travel mode choice modeling include individual income, household composition, trip purpose, trip duration, departure and arrival time, etc. From the viewpoint of classification, these indicators are input attributes and the actual travel mode is regarded as the output. For simplification, in this paper we consider two typical modes, i.e. "car" or "non-car", for traveling purposes.

The data set we used in this experiment is sampled from the Household Travel Survey (HTS) conducted by the Bureau of Transport Statistics, Transport for New South Wales, a state government agency in Australia [21]. In total, the training set consists of 11320 samples, of which 5660 samples are selected with "car" mode and others are with "non-car" mode. For the testing, we have other 5660 samples which include half size of car mode and half size of non-car mode. The attributes used include "Weekday index", "Household type", "Travel purpose", "Depart time", "Household income", "Road distance", and "Duration time". The types and preprocessing of these attributes are listed in Table VI. To discretize the continuous and data-time attributes, we used the following steps:

Step 1: define several fuzzy sets for a processed attribute;
Step 2: calculate the membership degrees of an observed value with respect to these fuzzy sets;
Step 3: select the fuzzy set with the biggest membership degree as the representative value of an observed value.

For example, Fig. 2 describes the given fuzzy sets for "Household income". From this figure, it is known that if a household has annual incomes over 200 thousands then the household is represented as "high income" household.

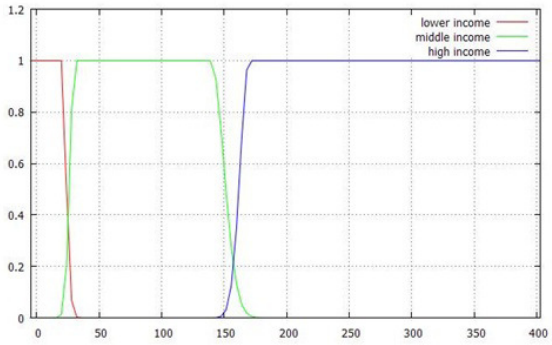


Fig. 2. Fuzzy sets defined for annual household income (unit 1000 AU$).

The proposed approach was compared on the travel mode choice problem to other three methods: NC, SKEL, and EFAST. Table VII shows the average performance for 20 independent runs, in terms of classification error on the test set, remaining network structure, and pruning time. Clearly, the proposed SRP algorithm outperforms the other pruning methods. For example, with only 36.8 hidden neurons, the

TABLE VI

DATA TYPE AND PREPROCESSING FOR TRAVEL MODE CHOICE MODELING

Attribute	Data type	Preprocessing
Weekday index	ordinal	NA
Household type	categorical	converted to nominal
Travel purpose	categorical	converted to nominal
Depart time	date-time	converted to nominal through fuzzification algorithm
Household income	continuous	converted to nominal through fuzzification algorithm
Road distance	continuous	converted to nominal through fuzzification algorithm
Duration time	continuous	converted to nominal through fuzzification algorithm

SRP achieves 85.1% classification accuracy, whereas, the NC, SKEL and EFAST methods give 71.2%, 72.5%, and 74.6% accuracy, respectively. In terms of computational cost, the proposed SRP algorithm requires the least pruning time than conventional pruning methods. On average, SRP spends 40.5 seconds to optimize the network structure, which is significantly better than NC, SKEL and EFAST (3591.8, 2785.7 and 589.2 seconds, respectively). This is because the SRP algorithm selects hidden neurons instead of eliminating the redundant neurons. As a result, the computational complexity depends on the number of hidden neurons to be selected instead of the number of hidden neurons to be removed.

TABLE VII

PERFORMANCE OF PRUNING METHODS FOR THE TRAVEL MODE CHOICE PROBLEM

Performance	NC	SKEL
Classification error	28.8±3.5	27.5±3.9
Remaining neurons	80.8±5.6	93.3±9.7
Pruning time(s)	3591.8±572.1	2785.7±518.1
Performance	**EFAST**	**SRP**
Classification error	25.4±3.8	**14.9±1.3**
Remaining neurons	75.8±8.3	**36.8±5.8**
Pruning time(s)	589.2±103.4	**40.5±8.1**

VI. CONCLUSION

We have presented a novel method of structure optimization for neural network, leading to better generalization ability and sparser architecture. The proposed SRP algorithm is characterized by searching for the sparse representation for the original structure. The proposed SRP algorithm selects significant hidden neurons from the network instead of eliminating redundant elements, thereby resulting in a quick convergence. Then the proposed method is tested using several benchmark datasets. A detailed investigation of the results has shown that the proposed algorithm performs reasonably well on average by eliminating as much as 86.09% of the original networks structure. In addition, experimental results also show that the proposed SRP algorithm achieves better generalization ability and faster convergence compared to existing neuron pruning algorithms for large-scale data sets.

REFERENCES

[1] K. Methaprayoon, W.-J. Lee, S. Rasmiddatta, J. Liao, and R. Ross, "Multistage artificial neural network short-term load forecasting engine with front-end weather forecast," *Industry Applications, IEEE Transactions on*, vol. 43, no. 6, pp. 1410–1416, Nov.-Dec. 2007.

[2] A. Yang, J. Bouzerdoum and S. Phung, "Dimensionality reduction using compressed sensing and its application to a large-scale visual recognition task," *International Joint Conference on Neural Networks (IJCNN 2010)*, pp. 1–8, 2010.

[3] J. Lim, "Finding features for real-time premature ventricular contraction detection using a fuzzy neural network system," *Neural Networks, IEEE Transactions on*, vol. 20, no. 3, pp. 522–527, March 2009.

[4] I. Daubechies, "Ten lectures on wavelets. philadelphia, pa: Siam," 1992.

[5] L. Prechelt, "Proben1-a set of neural networks benchmark problems and benchmarking rules. univ.karlsruher,karlsruher,germany,tech. rep, 21/94," 1994.

[6] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," *Adv. Neural Inf. Process. Syst*, vol. 2, pp. 598–605, 1990.

[7] B. Hassibi and D. G. Stork, "second-order derivatives for network pruning: optimal brain surgeon," *Advances in Neural Information Processing Systems*, vol. 5, pp. 164–171, 1993.

[8] M. Hagiwara, "Removal of hidden units and weights for back propagation networks," in *Neural Networks, 1993. IJCNN '93-Nagoya. Proceedings of 1993 International Joint Conference on*, vol. 1, Oct. 1993, pp. 351–354 vol.1.

[9] R. Reed, "Pruning algorithms-a survey," *Neural Networks, IEEE Transactions on*, vol. 4, no. 5, pp. 740–747, Sep 1993.

[10] M. Mozer and P. Smolensky, "Skeletonization: a technique for trimming the fat from network via relevance assessment," *Advances in Neura Information Processing Systems*, vol. 1, pp. 107–115, 1991.

[11] J. Sietsma and R. Dow, "Creating artificial neural networks that generalize," *Neural Networks*, vol. 4, no. 1, pp. 67–79, 1991.

[12] P. Lauret, E. Fock, and T. Mara, "A node pruning algorithm based on a fourier amplitude sensitivity test method," *Neural Networks, IEEE Transactions on*, vol. 17, no. 2, pp. 273–293, March 2006.

[13] R. Parekh, J. Yang, and V. Honavar, "Constructive neural-network learning algorithms for pattern classification," *Neural Networks, IEEE Transactions on*, vol. 11, no. 2, pp. 436–451, Mar 2000.

[14] J. O. Moody and P. J. Antsaklis, "The dependence identification neural network construction algorithm," *IEEE Transactions on Neural Networks*, vol. 7, no. 1, pp. 3–15, 1996.

[15] D. White and P. Ligomenides, "Gannet: A genetic algorithm for optimizing topology and weights in neural network design," in *International Workshop on Artificial Neural Networks, in New Trends in Neural Computation*, 1993, pp. 332–327.

[16] J. Gorodkin, L. K. Hansen, a. Krogh, C. Svarer, and O. Winther, "A quantitative study of pruning by optimal brain damage." *International journal of neural systems*, vol. 4, no. 2, pp. 159–69, June 1993. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/8044376

[17] S. Cotter, B. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *Signal Processing, IEEE Transactions on*, vol. 53, no. 7, pp. 2477–2488, July 2005.

[18] J. Yang, A. Bouzerdoum, and S. L. Phung, "A neural network pruning approach based on compressive sampling," in *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, June 2009, pp. 3428–3435.

[19] J. Chen and X. Huo, "Theoretical results on sparse representations of multiple-measurement vectors," *Signal Processing, IEEE Transactions on*, vol. 54, no. 12, pp. 4634–4643, Dec. 2006.

[20] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the rprop algorithm," in *Neural Networks, 1993. IJCNN 1993. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, June 1993, pp. 586–591.

[21] [Online]. Available: http://www.bts.nsw.gov.au/Statistics/Household-Travel-Survey/default.aspx?FolderID=215#top