Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 14, no. 4, April 2018, pp. 603-610
DOI: 10.23940/ijpe.18.04.p2.603610

Semi-Supervised Extreme Learning Machine using L1-Graph

Hongwei Zhao[a,b,c], Yang Liu[b], Shenglan Liu[a,b], and Lin Feng[a,b,*]

[a]*School of Computer Science and Technology, Dalian University of Technology, Dalian, 116024, China*
[b]*School of Innovation Experiment, Dalian University of Technology, Dalian, 116024, China*
[c]*Information and Engineering College of Dalian University, Dalian, 116024, China*

Abstract

The semi-supervised learning method has been widely used in the field of pattern recognition. Semi-supervised Extreme Learning Machine (SELM) is a typical semi-supervised learning algorithm. The graph construction result of the sample data has a tremendous impact on the SELM algorithm. In traditional graph composition methods such as Laplace graph, LLE graph and K neighboring graph, neighborhood parameters are specified by humans. If there are noises or uneven distribution in the data, the results are not very good. This paper proposes a SELM algorithm based on L1-Graph, which features no specifying parameters, is robust against noise, has a sparse solution and so on. The experiment confirms the effectiveness of the method.

Keywords: semi-supervised learning; SELM; L1-Graph; sparse representation

(Submitted on December 27, 2017; Revised on January 28, 2018; Accepted on February 24, 2018)

1. Introduction

With the rapid development of information technology, how to mine useful information from mass data [5] has become a hot issue. The data should be classified before data analysis, but classifying massive data manually cannot fulfill the requirements of data mining. Therefore, a variety of automatic classification algorithms have emerged.

The classical classification algorithms include decision trees, genetic algorithms [4], support vector machines [3], Bayes classification, K neighboring graph and neural network. The artificial neural network algorithm [7] was established in 1943 by W. McCulloch and W.H. Pitts. The artificial neural network model is composed of many interconnected nodes in the hidden layer according to certain rules. The information processing completes through a dynamic process of interacting neurons. Each node is provided with an accelerator and an activation function, and nodes between neighboring hidden layers are connected by weights. The complexity of the network is increased by adding hidden layers and the number of neurons in each layer. The network processes information by adjusting the corresponding connection weights.

In traditional artificial neural networks, the parameters of hidden layer node are determined through optimization by performing iterative algorithms that are optimized several times. As the iterative steps always take a lot of time in the training process of parameters, the efficiency of the network training process cannot be guaranteed. To enhance the overall performance of the network build, G.B. Huang et al proposed the Extreme Learning Machine (ELM) algorithm [6], which is a fast-training algorithm for a single hidden layer neural network. The algorithm has the following advantages. In the process of determining the network parameters, the parameters of hidden nodes are randomly selected without adjustment. External weights of the network are the least squares solution obtained by minimizing the quadratic loss function. Since the network parameters are not specified by the iteration process, the adjusting time of the network parameters is greatly reduced. The algorithm has attracted much concern in recent years.

* Corresponding author.
E-mail address: fenglin@dlut.edu.cn

Due to the tremendous difficulty of obtaining a lot of labeled data in practice, it is meaningful to train and classify by utilizing a small amount of labeled samples and a large number of unlabeled samples. Semi-supervised learning [8] has a very important practical significance in reducing labeling costs and improving learning performance of the machine. Semi-supervised extreme learning machine solves problems by establishing the optimal function based on graph theory [1].

In the solving process of semi-supervised ELM, the construction of figure matrixes plays a very important role on the results classification. When the figure matrix can accurately express the weight relationship among sample points, it can greatly enhance the accuracy of classification while reducing the accuracy of classifications when it fails. Currently, the known graph composition methods include LLE graph [9] and K nearest neighbor. The effects of these methods largely depend on the specification of parameters, and they are sensitive to noise. This paper introduces L1-graph [2] into the semi-supervised ELM to improve the classification accuracy, which is adaptive for neighborhood without setting the parameters, and it is not sensitive to noise.

2. Extreme learning machine

2.1. Classic extreme learning machine

Extreme learning machine is a single hidden layer feedforward neural network that solves the output weights by randomly selecting input weights. The extreme learning machine training speed is very fast. It can train feedforward neural networks more quickly than other classical neural network algorithms. The traditional classical gradient descent learning algorithm minimizes the error rate of training and ignores the problem of over-fitting and over-weighting. The ELM algorithm can not only ensure the minimum training error rate, but also ensure the weight of the smallest norm. So, ELM can achieve a better training effect. The ELM algorithm can get better results when dealing with local minima and over-fitting problems.

For input vectors $x \in R^n$, the output of a feedforward neural network with hidden nodes including N nodes can be described as Equation (1):

$$f_N = \sum_{i=1}^{N} \beta_i G(a_i, b_i, x), \quad a_i \in R^n, \ \beta_i \in R^m \tag{1}$$

Among them, $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^T$ is the weight that connects the pair of hidden layer nodes and output nodes, $G(a_i, b_i, x)$ is the hidden layer node output. The results of the hidden nodes are calculated by the activation function Equation (2).

$$G(a_i, b_i, x) = g(a_i \cdot x + b_i), \ b_i \in R \tag{2}$$

a_i is the weight that connects the input node and the hidden node, and b_i is the offset of hidden nodes.

For training dataset $\{(x_i, t_i) \mid i=1,2,\ldots,N\}$, among them $x_i \in R^n$ is the input vector in feature space, $t_i \in R^n$ is the actual label of the vector. We hope that the trained feedforward neural network will make the error equal to zero. This means β_i, a_i, b_i will satisfy the following Equations (3)(4)(5):

$$H\beta = T \tag{3}$$

Among them,

$$H = \begin{bmatrix} G(a_1, b_1, x_1) & \cdots & G(a_N, b_N, x_1) \\ \vdots & \ddots & \vdots \\ G(a_1, b_1, x_N) & \cdots & G(a_N, b_N, x_N) \end{bmatrix}_{N \times N} \tag{4}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_N^T \end{bmatrix}_{N \times m} \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \tag{5}$$

G.B. Huang has proven that the parameters of hidden nodes do not need to be adjusted during training and can be randomly selected within a given range. Such an extreme learning machine optimization problem can be considered as a linear programming problem and solved by least squares. The solution is Equation (6):

$$\beta = H^{\dagger}T \tag{6}$$

It can be proven that the above solution is the only least squares solution. $H^{\dagger}$ is a generalized inverse matrix of the hidden layer output matrix.

In summary, ELM model training and establishment process is as follows:

- Step 1: Determine the number of activation functions and neurons.
- Step 2: Generate neuron parameters randomly within the specified interval a_i, b_i, (i=1,2,…,N)
- Step 3: Calculate hidden layer output matrix H
- Step 4: Calculate the output weight β by Equation (7)

$$\beta = H^{\dagger}T \tag{7}$$

2.2. Semi-supervised extreme learning machine

SELM is an algorithm that can take advantage of a large number of unlabeled samples and has a higher prediction accuracy compared to ELM. Because unlabeled samples are more readily available, the SELM algorithm will find wider use in the field of classification.

The SELM algorithm uses a semi-label graph constructed from labeled and unlabeled samples. Each sample represents a vertex. If the node j is the neighborhood of node i, then there will be one edge with the weight Wij that connects two adjacent points. The instruction vector f indicates the type the sample belongs to. The definition function is as follows in Equation (8):

$$S(f) = \sum_{i\sim j} W_{ij}(f_i - f_j)^2 \tag{8}$$

Among them, "i ~ j" vertex j is the adjacent point of vertex i, and fi and fj are the indicator vector of the vertex i and j. If i and j are closer to each other, the values of their indicator vectors are closer.

Spectroscopy theory follows Equation (9):

$$\sum_{i\sim j} W_{ij}(f_i - f_j)^2 = f^T L f \tag{9}$$

Among them, L is Laplacian. The calculation of L is as in Equation (10):

$$L = D - W \tag{10}$$

The diagonal matrix is as in Equation (11):

$$D_{ii} = \sum_{j=1}^{l+u} W_{ij} \tag{11}$$

W is the weight of the edge. There are many ways to calculate the weights. Most of them use Gaussian functions as in Equation (12):

$$W_{ij} = e^{-\left|X_i - X_j\right|^2 / 2\delta^2} \tag{12}$$

Considering that the training error of the known label sample is the smallest, and the classification accuracy of the unknown label sample is more accurate, the optimization function of SELM is as in Equation (13):

$$\arg\min_f \frac{1}{2}\{\left\| f - T\right\|^2 + \lambda f^T L f\} \tag{13}$$

The first of the functions guarantees that the labeled training sample has the smallest training error, and the second function is the Laplacian penalty function. Referring to the results in the ELM, f= Hβ is substituted into the function as in Equation (14):

$$\arg\min_\beta l(\beta) = \arg\min \frac{1}{2}\{\left\| JH\beta - T\right\|^2 + \lambda(H\beta)^T LH\beta\} \tag{14}$$

Among them, H can be calculated using the ELM algorithm. With all labeled and unlabeled data, the dimension of H becomes (l + u)×N.

Introduced for convenience of calculation J, J=diag(1,1,…,0,0), the first l ones is 1, the rest is 0. Simultaneously the dimension of Matrix T is changed from l to l + u.

To solve SELM problems, the following transformation is done as in Equation (15):

$$\frac{\partial l}{\partial \beta} = 0 \Rightarrow (JH\beta - T)^T JH + \lambda(H\beta)^T LH = 0 \tag{15}$$

Then, there is Equation (16)

$$\beta = ((J + \lambda L^T)H)^\dagger JT \tag{16}$$

The following describes the calculation steps for SELM:

Datasets contain labeled and unlabeled samples, where labeled samples are as Equation (17):

$$\{(x_i \mid t_i) \mid x_i \in R^n, t_i \in R^m, i = 1, 2, ..., l\} \tag{17}$$

Unlabeled samples are as Equation (18):

$$\{x_i \mid x_i \in R^n, i = 1, 2, ..., u\} \tag{18}$$

Given hidden nodes *N*, constraint weights *λ*, activation function *g(x)*, calculation as follows:

- Step 1. a_i and b_i are randomly selected in a given range, $i=1,2,\ldots, N$, N is the number of nodes in the hidden layer.
- Step 2. Calculate the output matrix of the hidden layer.
- Step 3. Calculate the figure matrix L by Equation (19).

$$L = D - W, D_{ii} = \sum_{j=1}^{m} W_{ij} \tag{19}$$

- Step 4. Calculate the results by Equation (20):

$$\beta = ((J + \lambda L^T)H)^\dagger JT \tag{20}$$

- Step 5. Use the model Equation (21) to classify:

$$f = H\beta \tag{21}$$

3. L1-Graph

The SELM algorithm calculates the similarity between any two samples to obtain a similarity graph matrix. The similarity graph can be acquired by many methods including the Gaussian kernel function, LE graph, LLE graph, K neighboring graph et al. However, the above methods need parameter setting, which have a great influence on the results. The appropriate parameters can get a good graph. At the same time, they are very sensitive to noise. The methods perform poorly when the data sets have noise.

The building process of the graph includes the selection of a sample neighborhood and the setting of edge weights. *K* nearest neighbor method and ε nearest neighbor method are common methods in selecting a neighborhood. Both methods choose a neighborhood based on Euclidean distance, which is very sensitive to noise as a noise point can change the entire structure of the graph. If the *k* is too large, the *k* nearest neighbor method may bring very far points into the neighborhood. If ε is too small, the neighborhood may only contain some isolated points.

L1-Graph is a diagram based on sparse representation. Research on sparse representation has a long history. Sparse representation has many successful applications in image processing, pattern recognition and other fields. As opposed to the above graph, L1-Graph doesn't need to set parameters, and it has good robustness, sparse results, and automatically adapts to the neighborhood in dealing with noisy data setting.

Given a linear equation is: $x=D\alpha$, where $x \in R^n$ is the vector to be expressed, $\alpha \in R^N$ is the coefficient of reconstruct vector, $D \in R^{n \times N}$ is the base, the sparse expression solves the following optimization problem like in Equation (22). Then, it obtains sparse solutions.

$$\min_{\alpha} \|\alpha\|_0, \ s.t. \ x = D\alpha \tag{22}$$

$\| \bullet \|_0$ is a norm, which is 0. It represents the number of elements, which is not equal to 0 in a vector. But, the sparse representation is a NP-hard problem, so it is difficult to solve. However, if there are sufficient sparse solutions, sparse representation can be replaced by solving the following problems as in Equation (23):

$$\min_{\alpha} \|\alpha\|_1, \ s.t. \ x = D\alpha \tag{23}$$

Sparse representation problem can be solved in polynomial time.

The graph of L1-Graph is based on calculating weights though sparse representation, between sample points and other points.

L1-Graph build process is as follows:

- Step 1: Given input data: $X=[x_1, x_2, \ldots, x_N]$, where $x_i \in R^n$.
- Step 2: For each sample dataset x_i, solve the following problems as in Equation (24) and obtain robust sparse solutions:

$$\min_{\alpha^i} \|\alpha^i\|_1, \quad s.t. \quad x_i = B^i \alpha^i \tag{24}$$

- Step 3: Defining a graph as in Equation (25):

$$G = \{X, W\}, W_{ij} = \alpha_j^i \tag{25}$$

4. SELM based on L1-Graph

In the optimization function of SELM, the second term $f^T L f$ can assign similar points to one class as much as possible. The results of *L* matrix directly affect classification accuracy. Since the building of *k* neighboring graph, LLE graph, LE graph is greatly influenced by parameters, the building of the above graph needs to specify the neighborhood size *k*. The above graphs fail to achieve good results for data sets that contain noise or uneven distribution. The L1-Graph has the advantage of no parameter setting, good robustness on noise data set, and sparse solution. Therefore, L1-Graph can enhance the effects of classification precision in SELM.

SELM algorithm based on L1-Graph is described as follows:

Datasets contain labeled and unlabeled samples, where labeled samples follow Equation (26):

$$\{(x_i \mid t_i) \mid x_i \in R^n, t_i \in R^m, i = 1, 2, ..., l\} \tag{26}$$

Unlabeled samples follow Equation (27):

$$\{x_i \mid x_i \in R^n, i = 1, 2, ..., u\} \tag{27}$$

Given hidden nodes N, constraint weights λ, activation function $g(x)$, the calculation is as follows:

- Step 1. a_i and b_i are randomly selected in a given range, $i=1,2,\ldots, N$, N is the number of nodes in the hidden layer.
- Step 2. Calculate the output matrix of the hidden layer.
- Step 3. According to the calculation method of L1-Graph, calculate matrix W.
- Step 4. Calculate the figure matrix L by Equation (28).

$$L = D - W, D_{ii} = \sum_{j=1}^{m} W_{ij} \tag{28}$$

- Step 5. Calculate the results by Equation (29):

$$\beta = ((J + \lambda L^T)H)^{\dagger} JT \tag{29}$$

- Step 6. Use Equation (30) to classify:

$$f = H\beta \tag{30}$$

5. Experiments

The experiment employs Obj dataset, Iris dataset, Wine dataset, Segmentasc dataset, and Frey3 dataset to test the algorithm. The dataset description is shown in Table 1.

Table 1. The dataset description

Dataset	Dimension	The number of samples	The number of categories
Obj	1024	1440	20
Iris	4	150	3
Wine	13	178	3
Segmentasc	19	2310	7

The performance of Laplace graph, LLE graph and L1-Graph are compared as shown from Figure 1 to Figure 4.

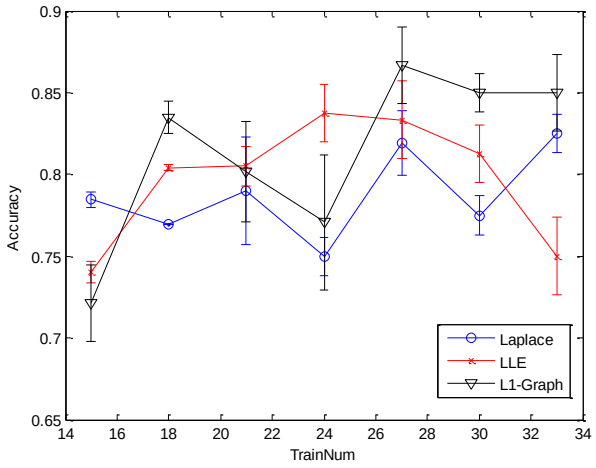

Figure 1. The recognition rate curve of different training sample on Obj datasets

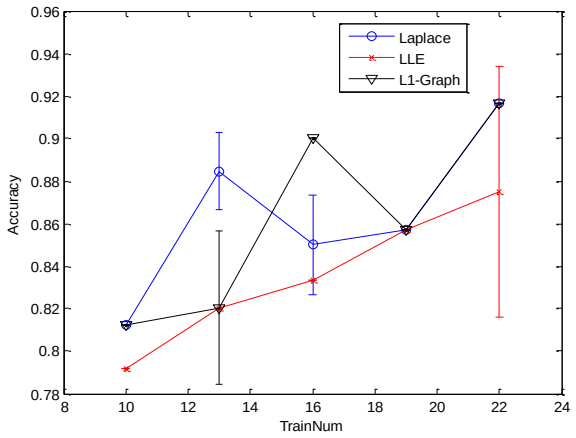

Figure 2. The recognition rate curve of different training sample on Iris datasets

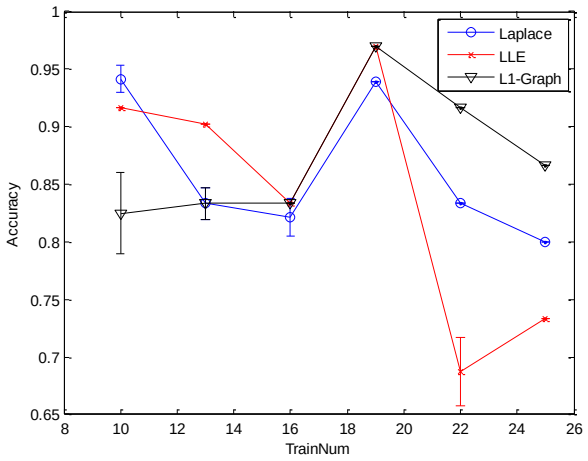

Figure 3. The recognition rate curve of different training sample on Wine datasets

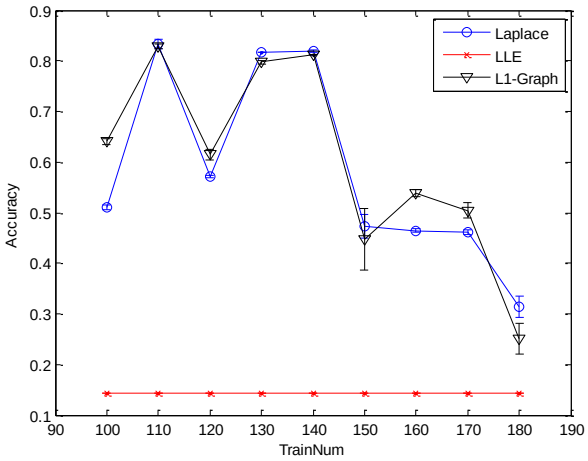

Figure 4. The recognition rate curve of different training sample on Segmentasc datasets

Tables 2-5 show the highest recognition rate and the average recognition rate of the four graphs in different data.

Table 2. The recognition rate comparison of different graphs in Obj datasets

	Laplace	LLE	L1-Graph
The highest recognition rate	82.5%	83.75%	86.67%
The average recognition rate	78.76%	79.75%	81.36%

Table 3. The recognition rate comparison of different graphs in Iris datasets

	Laplace	LLE	L1-Graph
The highest recognition rate	91.67%	87.5%	91.67%
The average recognition rate	86.41%	83.55%	86.13%

Table 4. The recognition rate comparison of different graphs in Wine datasets

	Laplace	LLE	L1-Graph
The highest recognition rate	94.17%	96.97%	96.97%
The average recognition rate	86.15%	84.04%	87.41%

Table 5. The recognition rate comparison of different graphs in Segmentasc datasets

	Laplace	LLE	L1-Graph
The highest recognition rate	83.3%	14.28%	82.86%
The average recognition rate	58.47%	14.28%	60.35%

Through its advantages of no neighbor parameter setting, good robustness against noise, and sparseness of solution, L1-Graph achieves good results on most datasets.

6. Conclusions

Due to difficulty of acquiring labeled samples, semi-supervised learning methods have intense application in classification. SELM is a typical semi-supervised learning algorithm. Since graph building has a great impact on SELM algorithms, this paper proposes a SELM algorithm based on L1-Graph, which has a good performance.

References

1. J. A. Bondy, U. S. R. Murty, "Graph Theory with Applications," *London: Macmillan*, 1976.
2. B. Cheng, J. Yang, S. Yan, Y. Fu and T.S. Huang, "Learning with L1-graph for Image Analysis," *IEEE Transactions on Image Processing*, vol. 19, no. 4, pp. 858-866, 2010.
3. C. Cortes, V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, 1995.
4. J.H. Holland, "Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence," *University of Michigan Press*, Ann Arbor, Michigan, 1975.
5. D. Howe, M. Costanzo, P. Fey, et al. "Big Data: The Future of Biocuration," *Nature*, vol. 455, no. 7209, pp. 47-50, 2008.
6. G.B. Huang, Q.Y. Zhu and C.K. Siew, "Extreme Learning Machine: Theory and Applications," *Neurocomputing*, vol. 70, no. 1, pp. 489-501, 2006.
7. W. McCulloch and W.H. Pitts, "A Logical Calculus of the Ideas Immanent in Nervous Activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115-133, 1943.
8. C. Olivier, S. Bernhard and Z. Alexander, "Semi-Supervised Learning," *MIT Press*, Cambridge, USA, 2006.
9. S.T. Roweis, L.K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, no. 5500, pp. 2323-2326, 2000.