Journal of Data Science 16(2017), 669-690

GRAPHICAL JUMP METHOD FOR NEURAL NETWORKS

Jing Chang[1], Herbert K. H. Lee[2]

[1]*Hunan University of Art and Science*
[2]*University of California, Santa Cruz*

Abstract: A graphical tool for choosing the number of nodes for a neural network is introduced. The idea is to fit the neural network with a range of numbers of nodes at first, and then generate a jump plot using a transformation of the mean square errors of the resulting residuals. A theorem is proven to show that the jump plot will select several candidate numbers of nodes among which one is the true number of nodes. Then a single node only test, which has been theoretically justified, is used to rule out erroneous candidates. The method has a sound theoretical background, yields good results on simulated datasets, and shows wide applicability to datasets from real research.

Keywords: Jump Plot, Model Selection, Neural Network

1. Introduction

Determining the optimal number of hidden units for a neural network is a difficult problem. When there are too few parameters, a statistical model or machine learning algorithm cannot capture the underlying trend of the data. Underfitting occurs and the prediction performance of such a model is poor. When there are too many parameters, a statistical model fits the pattern of random error or noise instead of the underlying relationship, which is called ”overfitting”. A model that has been overfitted performs poorly in predicting, since it overreacts to minor variations in the training data, essentially focusing on the noise rather than the true underlying trend. For a neural network model, when there are too few nodes, the prediction performance on the output variable is poor. When there are too many nodes, al- though the output error is lower on the training data, the errors for predicting novel examples increase. Selecting an optimal number of hidden nodes allows a good fit to both training data and future predictions, such as a hold-out test sample of data. Herein we consider only single hidden layer feedforward neural networks, although the methodology is extensible to other varieties. Thus we consider

[1] Postal address: Anhui Provincial Hospital first staff dormitory, Unit 1, Room 2104, Hefei, Anhui, China, 230001. email:jingbb@gmail.com. Phone:(86)18256056636. Jing Chang is corresponding author

[2] Postal address: UC Santa Cruz, School of Engineering, 1156 High Street, Santa Cruz, CA,95064. email:herbie@ams.ucsc.edu

fitting models of the form: $y_i = \beta_0 + \sum_{j=1}^{k} \beta_j / [1 + \exp(-\gamma_{j0} - \sum_{h=1}^{r} \gamma_{jh} x_{ih})] + \epsilon_i$ where x_{ih} is the h_{th} component of the i_{ih} sample of the inputs, y is the output, and $\varepsilon_i \sim$ N (0, σ^2).

A variety of approaches have been proposed to combat overfitting in neural networks, including early stopping (Sarle, 1995; Girosi et al., 1995), weight decay (Krogh and Hertz, 1992), and Bayesian methods (Lee, 2004). In addition, Sheela and Deepa (2013) and Xu and Chen (2008) provide recent reviews of the literature on selection of the number of hidden units. Here we survey criteria-based methods, andthen develop a new criterion based on a graphical interface. Two popular general model selection criteria that can be applied to choosing the number of nodes are Akaike's information criterion (AIC) (Akaike, 1973) and the Bayesian Information Criterion (BIC) (Schwarz, 1978). Another related criterion is Mallow's Cp statistic: $C_p = \frac{SS_E(p)}{\hat{\Sigma}^2} - n + 2p$ where SSE(p) is the sum of the squared errors of residuals, n denotes the number of observations, p denotes the number of parameters, and $\hat{\Sigma}^2$ is an unbiased estimate of the variance of an error term (Fogel, 1991). If $\hat{\Sigma}^2$ is known, any model which can estimate regression coefficients unbiasedly and include all critical regressors, has Cp converging to the number of parameters when sample size is large (Gilmour, 1996).

In the context of neural networks, Murata et al. (1994) has studied the theoretical relationship between the training error and the generalization error with regard to the training examples and the complexity of the structure of a neural network. The Network Information Criterion (NIC) chooses a specification for which the following expression takes a minimum: $NIC = -\frac{1}{T} lnL(\hat{w}) + \frac{tr[BA^{-1}]}{T}$. T is the sample size. L is the like $A = -E[\Delta^2 lnL_t]$ and $B = E\left[\Delta lnL_t \Delta lnL_t'\right]$. If the class of models investigated includes the true model, A = B asymptotically. Thus, $tr[BA^{-1}] = tr[I] = K$ is the effective number of model parameters, which is typically less than the nominal number because the parameters are dependent. However, this method can suffer from the problem of rejecting hidden units and d the least complex network architectures for model fitting (Anders and Korn, 1999). However, none of aforementioned methods performs well in choosing the best number of nodes for a neural network. Hence it is critical to find a new method which does a good job for such a task.

In the field of choosing the number of clusters in a mixture model, Sugar and James (2003) proposed the jump method from an information theory point of view. By adapting ideas from rate distortion theory to clustering, the theory of the jump method investigates the functional form of the mean square error (MSE) curve in both the appearance and absence of clusters. Furthermore, they demonstrate both theoretically and empirically, that the MSE curve, when transformed to an appropriate negative power, will display a jump, reliably and accurately, at the true number of clusters. However, it is often arduous to designate the transformation parameter directly. Chang and Sugar (2008) proposed a graphical tool, christened the "graphical jump method", to ascertain the number of clusters. By changing the transformation parameter, the transformed MSE curve jumps at divergent numbers of clusters, called candidate numbers, amongst which one is the true number of clusters. If the candidate number is smaller than the true number of clusters, at least one cluster will accommodate more than one true

cluster and yield a positive result on a test, which is dubbed the "cluster-existence test" and has been theoretically justified.

In this paper, the graphical jump method is extended to solve the problem of choosing the number of nodes for a neural network. First, by using theoretical results from Murata et al. (1994), a theorem is proved stating that after some boundary conditions are satisfied, there surely exists a transformation power by which the MSE can be transformed to exhibit a jump at the true number of nodes. A "single node only test", which is also justified theoretically, is used to rule out erroneous candidates. The newly developed method for choosing the number of nodes makes limited parametric assumptions, can be rigorously theoretically motivated using theorems from Murata et al. (1994), and is simple to both understand and implement. The jump method only applies to choosing a parameter that is a counting number, and does not apply for continuous or other-valued parameters.

In Section 2, the theory and concrete steps of the graphical jump method are introduced in detail. In Section 3, simulation studies and results are elucidated. Section 4 describes the analysis of a real dataset. Section 5 lays out future research directions.

1.1 The introduction of the graphical jump method

Assume that the data are fitted to the following neural network:

$$y_i = \beta_0 + \sum_{j=1}^{k} \beta_j / [1 + \exp(-\gamma_{j0} - \sum_{h=1}^{r} \gamma_{jh} x_{ih})] + \epsilon_i \qquad (1)$$

where $\varepsilon_i \sim N(0, \sigma^2)$. The MSE d_k equals the variance of residuals generated by fitting the neural network model with k nodes.

Assuming that the dataset is fitted by model (1), the graphical jump method has the following four basic steps for choosing the best number of nodes:

a. Calculate MSE d_k for K = 1...k_{max} by fitting the neural network model using k nodes.

b. Choose a positive number v > 0, called the transformation power.

c. Calculate the jump score associated with k nodes $J_k = d_k^{-v} - d_{k-1}^{-v}$, with $J_1 = d_1^{-v}$.

d. The best number of nodes is the number k with the highest J_k.

To give a simple illustration of how the graphical jump method works, a simulated dataset is generated with 100 observations from a gamma(20, 40) distribution with supplementary standard normal noise and the true number of nodes of 4. The response, Y 1, is the aggregate of the 4 different nodes: Y 2, Y 3, Y 4 and Y 5 (Figure (1)), whose coefficients are manifested at the top of the plots. The first node and the third node have active declining regions in the range of (-1.5, -0.3) and (0.5,1), while the second node and the fourth node have active increasing regions in the range of (-0.3,0.4) and (1.1,1.9). Consequently, the final response variable, Y 1, has 4 disconnected active regions with adjacent active regions in totally opposite directions, which necessitate 4 nodes to provide the best fit.

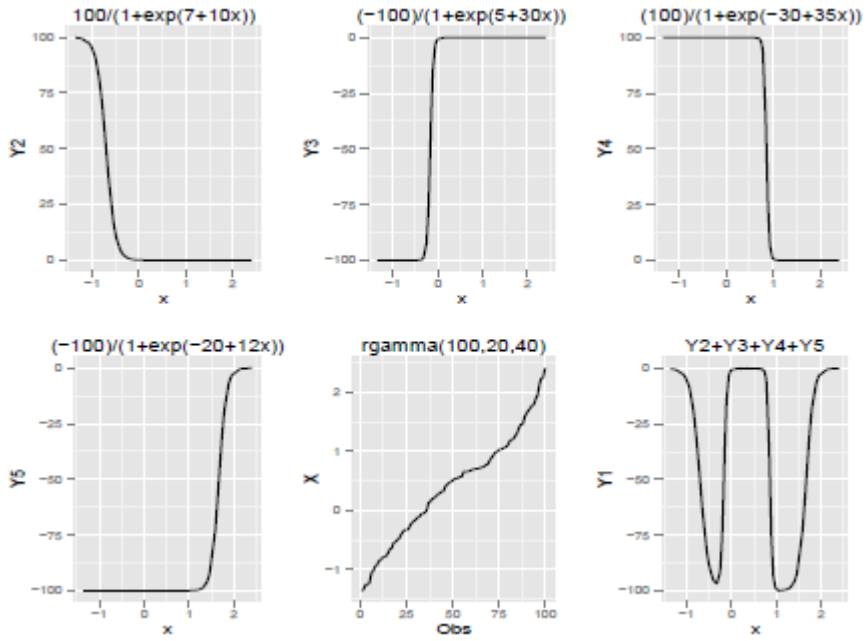


Figure (1).The shape of each node for a simulated dataset. The first to the third nodes are described by the upper left, upper middle and upper right plots. The lower left plot illustrates the fourth node. The lower middle plot shows the x values. The lower right plot illustrates the sum of the 4 nodes.

A graphical visualization is provided by Figure (2) which plots the successive jumps in the transformed MSE. In the plot, the possible number of nodes ranges from 1 to 10. The lower left plot of Figure (2) shows a jump at 4, which is the true number of nodes. Intuitively, this jump occurs because of the sharp increase in the jump scores that results from not modeling noise using additional nodes. Adding subsequent neural network nodes cannot decrease, but increase the MSE of residuals and thus has a smaller contribution to the jump score. When the transformation powers change from 0.4, 1, 2 to 5, the highest jump scores occur at nodes of 1, 1, 4 to

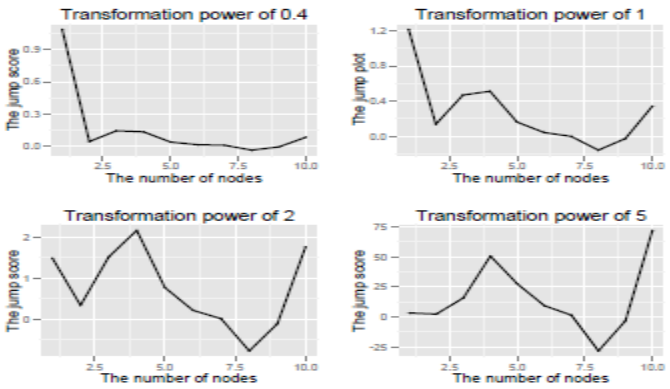
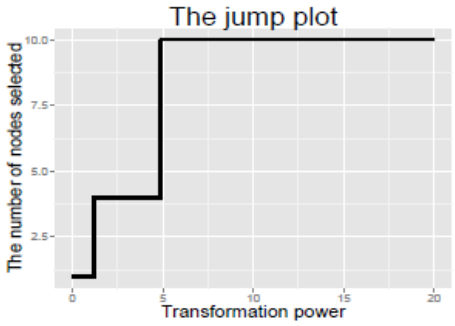


Figure (2). Plots of the jump scores versus the number of nodes under different transformation powers for a simulated dataset.

10. A jump plot (Figure (3)) is generated to elucidate the functional relationship of the number of nodes selected versus transformation power used. As the transformation power increases from 0 to 20, the number of nodes selected changes from 1, 4 to 10.



Figure(3). The jump plot for the simulated dataset.

As the transformation power y approaches 0, the jump score for 1 node d_1^{-y} approaches 1 while the jump score for k nodes $d_k^{-y} - d_{k-1}^{-y}$ approaches 0. Thus the highest jump occurs at node one. As the transformation power approaches an enormous number, the jump score for 10 nodes $d_{10}^{-y} - d_9^{-y}$ will be the largest one. This is because the MSE of residuals is a decreasing function of the number of nodes, d_{10} is smaller than d_k for $1 \leq k \leq 10$. As y approaches infinity, d_{10}^{-y} increases much faster than d_k^{-y} for $1 \leq k \leq 10$. This leads to the fact that above some point of y, $d_{10}^{-y} - d_9^{-y}$ will be the highest jump score.

1.2 Summary of the Graphical Jump Method

By utilizing the graphical property of the jump plot, a graphical jump method is developed to choose the number of nodes more efficiently. Since a jump surely occurs at the best number of nodes, by choosing the candidates as the number of nodes where the jumps occur, the range of candidate numbers of nodes are significantly narrowed down. When implementing the graphical jump method, a jump plot is sketched at first to illustrate all the candidate numbers of nodes, assuming there are a total of g candidates. Secondly, organizing the g candidate numbers of nodes from small to large, the dataset is modeled with these candidate numbers of nodes sequentially to produce g consecutive sets of residuals, for which g jump plots are produced to see whether each plot contains candidates with more than one node. The key idea is that if the best model has been found, there is nothing left to model in the residuals, whereas if there are not yet enough nodes in the model, then one can find this signal in the residuals by fitting one or more nodes to the residuals.

If the candidate number of nodes is less than or equal to the best number of nodes minus two, then it cannot account for the total variability of the dataset. The corresponding residuals will encompass variability that has to be explained by additional nodes, and thus it will test negative on the single node only test. This is caused by the fact that if they had produced positive result, then the total number of nodes needed to model the data is the candidate number of nodes plus one, which is less than the best number of nodes and contradicts the original assumption of the true size of the network.

Nevertheless, if a candidate number of nodes is adjacent to the best number of nodes, then the residuals of both numbers of nodes will present positive results on the "Single node only test". As a result, the first candidate number of nodes without an adjacent lower candidate that demonstrates that its residuals need to be explained by a single node only, is the best number of nodes. If two earliest consecutive numbers of nodes, N_i, $N_i + 1$, both indicate that their residuals only need one node to count for the total variability, then the ratios of d_{Ni-1}/d_{Ni} and d_{Ni}/d_{Ni+1} are calculated.

If N_i is the best number of nodes, d_{Ni-1} should be much larger than d_{Ni} since themodel changes from modeling the main effects to modeling the noise at this point. Therefore, the ratio of d_{Ni-1}/d_{Ni} should be the larger one. If $N_i + 1$ is the true number of nodes, d_{Ni} should be much larger than d_{Ni+1} due to similar reason. Consequently, the ratio of d_{Ni}/d_{Ni+1} should be the larger one.

2. Method

2.1 The Two Theorems

The following theorem provides an asymptotic result of the shape of the MSE curve after transformation, and thus it provides a theoretical explanation for the graphical jump method.

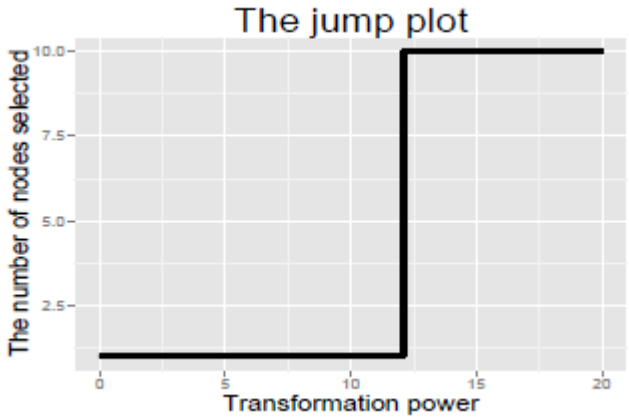
Theorem 1: Assume the dataset y follows model (1). Define K_{max} as the maximum number of nodes investigated, t as the sample size. Assume the dataset is composed of G nodes and that the likelihood ratio test of H_0: the dataset is composed of G nodes, versus H_A: the dataset is composed of less than G nodes, yields a positive result. Define m^* as the number of parameters in one node. If $\frac{t}{2Chisq(m^*,0.05)} < \nu < \frac{t^2}{6m^*(k_{max}-G)}$, as $t \to \infty$, the jump method always selects the true number of nodes as a candidate (proof in appendix).

By the aforementioned theorem, when changing the transformation power from small to large, a jump surely exists at the best number of nodes. Nonetheless, some- times a correct transformation power is difficult to identify due to the deviation of the dataset from the theoretical model, measurement error caused by experiment operators and random errors, etc. Ergo, the graphical property of the jump plot is explored to provide a tool for choosing the number of nodes based on the jump plot. The nodes at which jumps appear are called candidate numbers of nodes, N_1, ..., N_g. Theorem 2 provides a theoretical foundation for examining whether a dataset needs to be fitted by the neural network model with more than one node, dubbed "Single node only test". It is not onerous to conjecture that, if N_i is the best number of nodes, after fitting the data with Ni nodes, the residuals follow a standard normal distribution and result in a positive test in the "Single node only test". Thus the candidate numbers of nodes in the jump plot will be fitted to the dataset with the neural network models one by one to figure out which residuals will give positive results in the test.

Theorem 2: Assume that $y \overset{iid}{\sim} N(0, 1)$. Define K_{max} as the maximum number of nodes investigated, t as the sample size, and $m*$ as the number of parameters in one node. If $v < \frac{t^2}{4K_{max}m^*}$, as $t \to \infty$, the jump plot will only select one node as the candidate.

The proof of Theorem 2 is provided in Chang (2011). Theorem 2 is illustrated by Figure (4), which is a jump plot generated by a dataset simulated from a standard normal distribution. Theorem 2 demonstrates that the jump plot designates one node as the solitary candidate in a long range of transformation powers. However, the length of this range depends on the dataset actually generating the jump plot. Empirically, this range would always include (0, 2), i.e, the length of this range would be larger than 2 units for the majority of the datasets. Therefore, if the true number of nodes is one, the jump plot would select one as the candidate number of node in a range, including (0, 2), as the solo candidate. Nevertheless, by Theorem 1, if the true number of nodes is larger than one, the jump plot would select a candidate number of node bigger than one within this range. For instance, in Figure (3), the dataset is composed of 4

nodes and 4 nodes are pinpointed in the jump plot within the range of (1.2, 4.6), which overlaps the range (0, 2) at (1.2, 2). Therefore, in all, if the jump plot picks a candidate number of nodes (larger than one) within the range of (0, 2), then the dataset is deemed as composed of more than one node and one node otherwise. As a result, in Figure (4), the dataset is regarded as composed of one node.



Figure(4) The jump plot for the simulated dataset.

2.2 Connection between the theory of the NIC criterion and the graphical jump method

There are connections between the theory of the NIC criterion and the graphical jump method. Consider a stochastic system which has an input vector $x \in R^k$ and produces an output vector $y \in R^l$. An input vector x is generated according to the probability q(x) and an output vector y is generated according to a conditional probability q(y|x) specified by x. q(x, y) is the product of q(x) and q(y|x). A network is considered to have a conditional distribution p(y|x,θ), where $\theta \in R^m$ is an m-dimensional parameter vector that describes the network, such as a set of weights and thresholds. q represents the true distribution of p(y|x, θ). Let f (x, y) be the density function for neural network models. The dataset is assumed to follow the model: y = f (x,θ) + ξ(x), where ξ(x) is noise and E(ξ(x)) = 0. To calculate the goodness of fit of a neural network, a discrepancy function D(q, p(θ)) is designed to measure the difference between q(y|x) and p(y|x, θ). d(x,y,θ) is a loss function, typically, it could be square error loss or negative log likelihood. The square error loss is defined as $d(x, y, \theta) = (\|y - f(x, \theta)\|)^2$. The discrepancy function we use is $D(q, p(\theta)) \equiv \int d(x, y, \theta)q(x)q(y|x)dxdy$. In order to minimize the discrepancy function, the true probability distribution q(x,y) of the target system needs to be known. However, it is not possible to identify q(x, y) in reality. Frequently, the empirical distribution $q^*(x, y) \equiv \frac{1}{t}\sum_{i=1}^{t}\delta(x - x_i, y - y_i)$ is used instead. It is well known that the empirical distribution approximates the true distribution q(x, y) in the weak sense when sample size is large, and hence it is reasonable to evaluate the network model by $q^*(x, y)$ instead of q(x,y). p(θ) is the distribution of θ. For square error loss, $D(q^*, p(\theta)) \equiv \sum_{i=1}^{t}\|y_i - f(x_i, \theta)\|^2$, the following is an important result on which the graphical jump method depends:

$$D(q, p(\theta_{opt})) = \min_\theta D(q, p(\theta))$$

where θ_{opt} is the value of θ when $D(q^*, p(\theta))$ achieves the minimum.

$$D(q^*, p(\theta^*)) = \min_\theta D(q^*, p(\theta))$$

where θ^* is the value of θ when $D(q^*, p(\theta))$ attains the minimum.

Let $R_{opt} \equiv V_q[\nabla d(x, y, \theta_{opt})]$, i.e., for the true distribution of θ, R_{opt} is the variance of the first derivative of $d(x, y, \theta_{opt})$ with respect to θ. Let m be the number of parameters in the model, Ropt is of m × m dimensions. Let $Q_{opt} \equiv E_q[\nabla\nabla d(x, y, \theta_{opt})]$, i.e., for the true distribution of θ, Q_{opt} is the expectation of the second derivative of $d(x, y, \theta_{opt})$ with respect to θ. Let m be the number of parameters in the model, Q_{opt} is of dimension m × m.

Theorem 3: The average discrepancy between the system q(x,y) = q(y|x)q(x) and the machine $p(y|x,\tilde{\theta})$ learned from t examples is given by $< D(q, p(\tilde{\theta})) > = < D(q^*, p(\tilde{\theta})) > + 1\ tr(R_{opt}Q_{opt}^{-1}) + O(t^{\frac{-3}{2}})$, where < . > denotes the expectation with respect to the distribution of $\tilde{\theta}$, the parameter after sufficient learning of θ with the machine, and θ^* (Murata et al. (1994), page 868). Theorem 1 studies the difference of $< D(q, p(\tilde{\theta})) >$ and $< D(q^*, p(\tilde{\theta})) >$ in terms of the ensemble average of training sets. Nevertheless, when using this criterion formodel selection, we need to evaluate $< D(q, p(\tilde{\theta})) >$ and $< D(q^*, p(\tilde{\theta})) >$ for one particular training set. A "subset" for a

single layer feedforward neural network is defined as following: if the first model has fewer hidden units than the second model, then it is deemed a submodel of the second one. The submodel can be obtained from the full model by setting the connection weights and thresholds of the extra units equal to 0. < D(q, p($\tilde{\theta}$)) > can be decomposed as we now show. Let M_i ={$p_i(y|x,\theta_i)$; $\theta_i \in R^{mi}$ } be a hierarchical series of models: $M_1 \subset M_2 \subset M_3 \subset ...$, where M_i is a submodel of M_j, (i < j). For only one training set

$$D(q, p(\tilde{\theta})) = D(q*, p(\tilde{\theta})) + U\frac{1}{\sqrt{t}} + \frac{1}{t}tr(R_{opt}Q_{opt}^{-1}) + O(t^{-3/2}) \qquad (2)$$

where $U = \sqrt{t}D(q, p(\theta_{opt})) - D(q^*, p(\theta_{opt}))$, is a random variable of order 1 with zero mean. U is common to all the models within a hierarchical structure, such as single layer neural network models with the same dimensions for the input and the output vectors, see Murata et al. (1994) (page 869).

Obviously, the discrepancy D(q, p($\tilde{\theta}$)) achieves the minimum at the best numbers of nodes, resulting in a sequence of inequality equations composed of the right side of formula (2). For negative log likelihood, R = Q, which makes $tr(R_{opt}Q_{opt}^{-1})$ reduce to the number of parameters in the corresponding neural network. U is common to all the models within a hierarchical structure. D(q^*,p($\tilde{\theta}$)) can be expressed as a function of MSE of residuals under different numbers of nodes. Thus, the inequality equations reduce to the inequality relationship of MSE of residuals under different numbers of nodes. By utilizing those inequality relationships of MSE of residuals and with the help of a Taylor expansion, the necessary conditions of the jump method, which are inequality relationships of MSE after transformation, are proved and the two related Theorems are established.

3. Simulation Study

3.1 One dimensional data

Simulation studies are first performed with one x variable, and with 100, 200, 300 and 1000 observations. For the scenarios of one x variable with 100, 200, and 300 observations, data are simulated with 4 nodes, which have distinct active regions as shown in Figure (1). The x variable is generated from a gamma distribution with shape parameter of 20 and rate parameter of 40, plus a noise variable with standard Gaussian distribution. For the first 3 scenarios, from nodes 1 to 4, the y variables are generated using the following formula:

1. $y_1 = \frac{100}{1+\exp(7+10\times x)}$ 2. $y_2 = \frac{-100}{1+\exp(5+30\times x)}$

3. $y_3 = \frac{100}{1+\exp(-30+35\times x)}$ 4. $y_4 = \frac{-100}{1+\exp(-20+12\times x)}$

The final $Y = y_1 + y_2 + y_3 + y_4 + \epsilon$, where $\epsilon \sim N(0,0.1)$.
For the last scenario, from nodes 1 to 4, the y variables are generated using the following formula:

$$1.\ y_1 = \frac{100}{1+\exp(6.012-6.536\times x)} \qquad 2.\ y_2 = \frac{-100}{1+\exp(-4.256-7.566\times x)}$$

$$3.\ y_3 = \frac{100}{1+\exp(-2.577-4.65\times x)} \qquad 4.\ y_4 = \frac{-100}{1+\exp(5.614-4.270\times x)}$$

Finally, $Y = y_1 + y_2 + y_3 + y_4 + \epsilon$, where $\epsilon \sim N(0, 0.1)$.

Table (1) demonstrates simulation results for one dimensional data for sample sizes of 100, 200, 300 and 1000. The first 3 rows are the percentages of the number of nodes correctly chosen for each method and each scenario. The last 3 rows are the mean and the standard deviation of the prediction errors for each method for the 30 datasets in each scenario. The three methods for comparison are the graphical jump method, cross validation and BIC. The goals of the graphical jump method and cross validation are different. The former is for the purpose of model selection, which corresponds to the top 3 rows of table, and the latter is for the purpose of prediction, which corresponds to the bottom 3 rows of the table. The predictive accuracy is obtained by comparing the prediction results to the known true response values in the simulated examples. The total number of datasets is 30 for all the scenarios in this paper. The graphical jump method performs the best in choosing the correct number of nodes. The percentages of the number of nodes correctly chosen for the graphical jump method are well above those of the other two methods. For the prediction errors, the graphical jump method leads the other two methods for 100 and 300 sample sizes scenarios, leads cross validation for 200 and 1000 sample sizes scenarios.

Figure (5) illustrates the results of the simulation studies for one dimensional data.

Table (1).The percentages of the correct number of nodes chosen by each method and the mean (sd) of the prediction errors generated by each mothod for one dimensional scenarios.

number of datapoints	100	200	300	1000
Graphical jump method (percentage)	100	100	70	100
BIC (percentage)	80	100	3.3	100
10 fold cross validation(percentage)	16.8	60	0	76.7
graphical jump method (mean(sd)))	1.40e-06(4.89e-07)	1.08-06 (1.08e-07)	1.07e-06(1.34e-07)	2.81e-06(1.25e-07)
BIC (mean(sd))	1.42e-06(4.76e-07)	1.08e-06 (1.08e-07)	1.19e-06(5.07e-07)	2.81e-06(1.25e-07)
10 fold cross validation (mean(sd))	1.41e-06 (5.70e-07)	1.11e-06 (1.12e-07)	1.22e-06 (5.13e-07)	2.82e-06(1.32e-07)

The upper left plot is the scatter plot of the prediction errors versus the i_{th} dataset (There are 30 simulated datasets for each scenario and for each dataset, there is a prediction error generated for each method). In the scatter plot, the red, the dark blue and the green curves

connect points of prediction errors for the graphical jump method, the BIC method and the cross validation method separately. This color representation is used for all the scatter plots in this paper. The middle left plot contains several box plots. To draw the box plot, the prediction errors for all the three methods in this scenario are combined and then categorized by their corresponding number of nodes chosen. For example, for all the datasets where the graphical jump method chooses the number of nodes of three, the prediction errors generated by the graphical jump method are combined into one group. Similarly, prediction errors whose corresponding number of nodes chosen are three for the BIC method and the cross validation method are categorized into the same group as that for the graphical jump method. For other numbers of nodes, data are categorized similarly. Theoretically, when overfitting happens, the mean value of boxplot

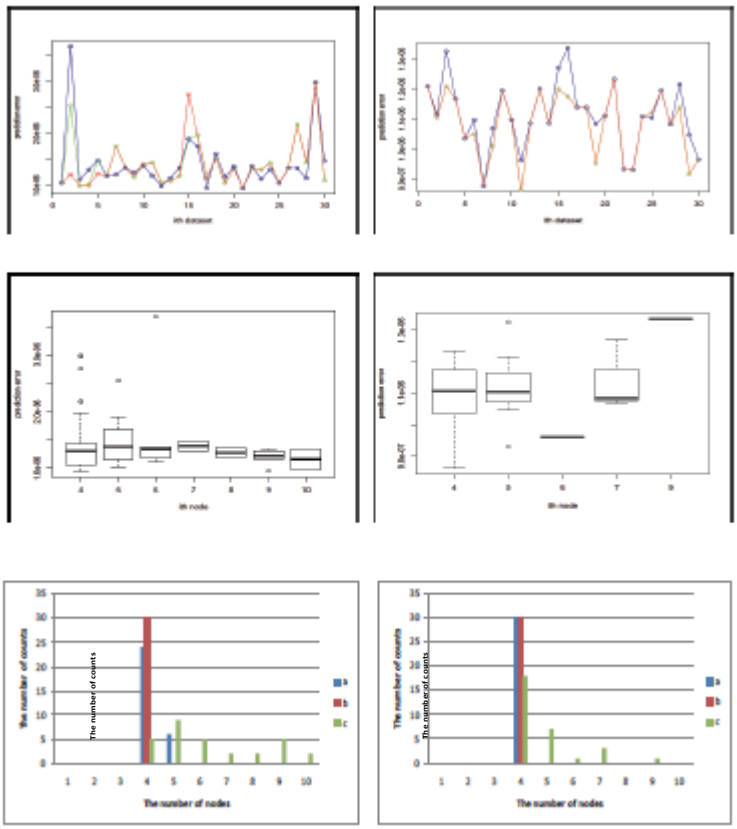


Figure (5) Simulation study results for one dimensional data of 100 and 200 sample sizes (the upper two plots are the scatter plots of prediction errors, the middle two are the boxplots of prediction errors for each node, the lowest two are the 2-D histograms of counts for each number of nodes chosen by each method).

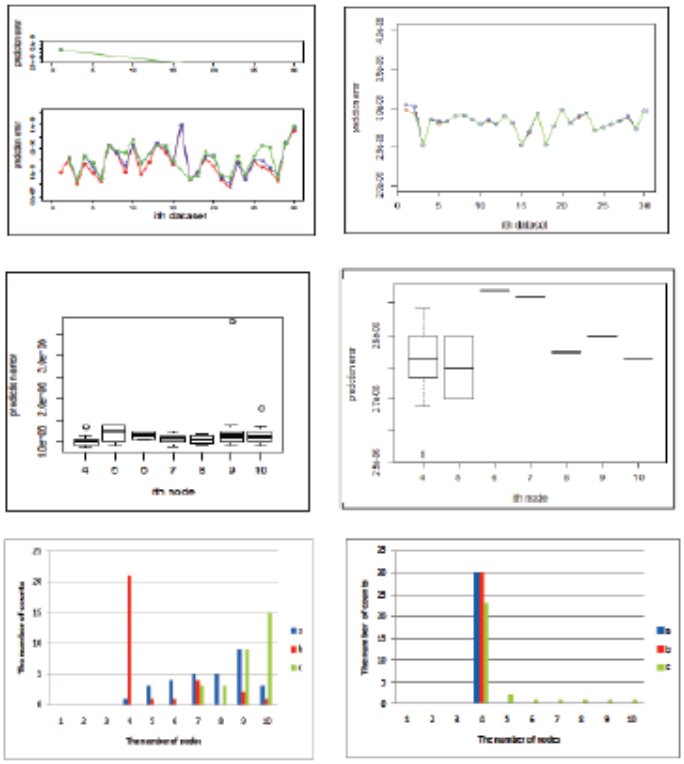
should increase with the increasing of the number of nodes chosen. However, in this plot, when the number of nodes chosen increases, the mean value of the boxplots increases at first and then

decreases. This may be because the sample size is 100, which is too small, the prediction errors calculated are not accurate enough. The lower left plot is a 2 − D histogram generated by excel. The x-axis indicates the number of nodes chosen. The y-axis indicates the number of datasets which choose the corresponding number of nodes in the x-axis. For all the 2 − D histograms in this paper, the red, the dark blue and the green columns compose the histograms for the graphical jump method, BIC and cross validation separately, i.e., they use the same color representation as that in the scatter plot. For the lower left plot, the graphical jump method chooses the correct number of nodes, i.e. four nodes, with the highest rate. BIC chooses four nodes and five nodes with the highest rates.

The right three plots are generated for sample size of 200. For most part of the upper right plot, the red curves are the lowest among the three, which means that for most of the datasets, the prediction errors generated by the graphical jump method are the lowest. For the plot in the middle right, the mean values of the box plots show an increasing trend along the x-axis. The 2 − D histogram in the lower right shows that the graphical jump method and BIC choose 4 nodes with the highest rates.

Figure (6) has six plots. The upper left, the middle left and the lower left plots contain the scatter plot, the box plot and the 2 − D histogram separately. There are two parts in the scatter plot, which differ by their y axises. The y-axis of the first part ranges from 0 to 2×10^{-6} and the x-axis indicating the sequence of the 30 datasets. The y-axis of the second part ranges from 2×10^{-6} to 5×10^{-6}. Both parts share the same x-axis. The two parts display the different appearances of the scatter plot in the corresponding ranges of the y axes. For most of the scatter plot, the red curve is the lowest, which means the graphical jump method generates the lowest prediction errors most of the time. The middle left plot shows that the mean value of the box plot increases with the number of nodes chosen generally. For the 2 − D histogram, the graphical jump method chooses 4 nodes as the best number of nodes most of the time. However, BIC and cross validation choose 9 or 10 nodes with the highest percentages. This is because for a small sample size, such as the 100 sample size scenario, 4 nodes can explain most of the variability of the dataset.

Adding more nodes cannot decrease the MSE of residuals too much. Hence BIC and cross validation will choose a node count close to 4 nodes. When the sample size becomes larger, there are more data points. 4 nodes is not enough to explain most



Figure(6) Simulation study results for one dimensional data of sample size of 300 and 1000 (the contents of the 6 plots in Figure (6) are the same as those in Figure (5)).

of the variability of the dataset. Adding more nodes makes the MSE of the residuals decrease a lot. Hence BIC and cross validation will choose 9 and 10 nodes as the best number of nodes with high rates.

The right three plots in Figure (6) are the scatter plot, the boxplot and the 2-D histogram for sample size of 1000. Since the sample size is large enough, all the three methods select the correct number of nodes most of the time, as you can see from the 2-D histgram. Also the boxplot shows that there are just a few observations from 5 nodes to 10 nodes.

3.2 Three dimensional dataset

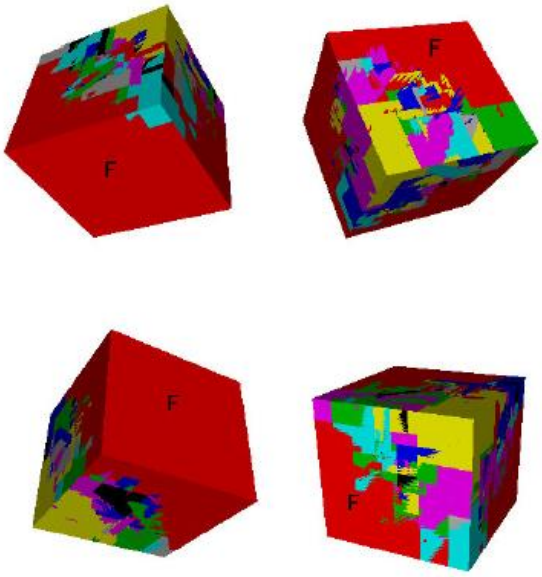
Finally, simulations are performed for three variables. A cube described by the three variables is generated. For each node, the active region is located at a corner of the cube (See Figure (7)). The faces labeled by "F" are the front faces. Figure (7) demonstrates the positions of the active regions relative to the front faces. For the aggregate of the four nodes, the active regions are located at the 4 different corners of the cube. The nodes are then generated as following: firstly, n×100 observations with gamma distribution, with shape parameter 20 and rate parameter 40, plus a standard Gaussian noise term, are generated.

After sorting them from small to large, the n×100 observations are divided into n subgroups with consecutive 100 observations being in the same subgroup. Therefore, n subgroups are produced. n x_1 observations are produced with each observation equaling to the mean of each subgroup. n x_2 observations and n x_3 observations are generated similarly. Then n observations, $X_1, ..., X_{n^3}$, are produced with the 3 coordinates being a combination of each x_1 (first coordinate), x_2 (second coordinate) and x_3 (third coordinate) values. The first to fourth nodes are simulated by the following formula:

$$(1)\ y_1 = \frac{100}{1+\exp(20-5X_1-5X_2-5*X_3)} \qquad (2)\ y_2 = \frac{100}{1+\exp(8+5X_1+5X_2+5X_3)}$$

$$(3)\ y_3 = \frac{100}{1+\exp(10+5X_1-5X_2+5X_3)} \qquad (4)\ y_4 = \frac{100}{1+\exp(15-5X_1+5X_2-5X3)}$$

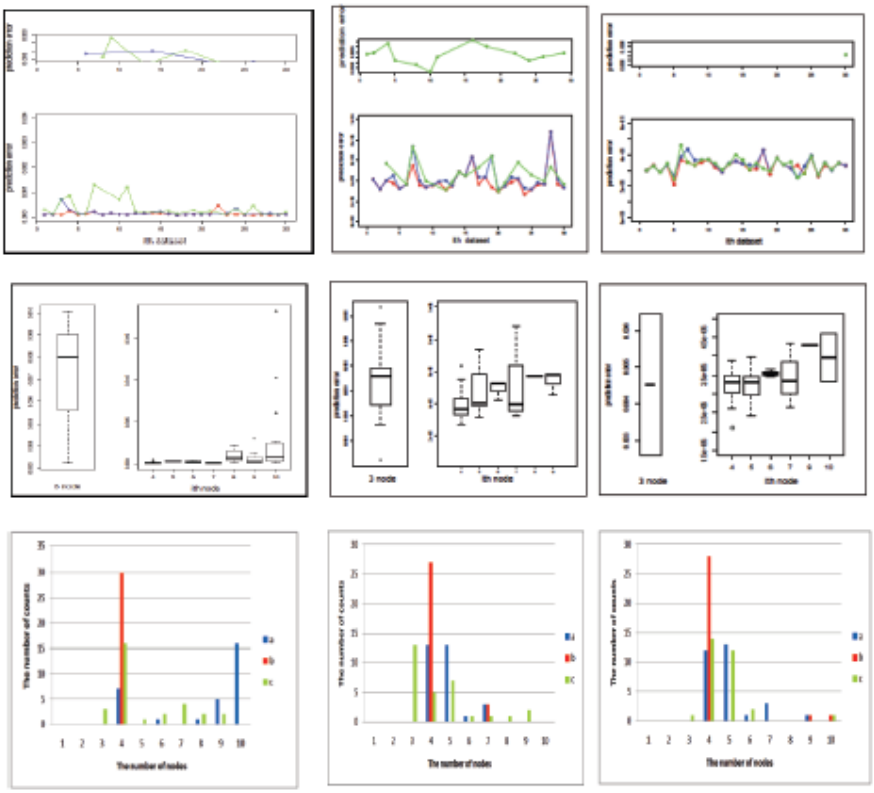
where X_1, X_2 and X_3 are the first, second and third coordinates of Xs. The final Y values are generated by $Y = y_1 + y_2 + y_3 + y_4 + \epsilon$, where $\epsilon \sim N(0, 1)$. Simulations are done for n values equaling 4, 5 and 6.



Figure(7) Plots of the shape of each node for the simulated data with three explanatory variables.

Table (2) is presented the same way as for one dimensional case. For all of the three scenarios, the graphical jump method leads the other two methods in both picking the correct number of nodes and making predictions. The graphical jump method picks 100%, 90% and 93.3% correct results for sample sizes of 64, 125 and 216, separately, which are almost twice of those from the second best methods. The mean (sd) of the prediction errors generated from the graphical jump method are 1.14e-04 (7.68e-05), 3.97e-05(1.18e-05) and 3.27e-05(4.21e-06) for sample sizes of 64, 125 and 216, respectively. Both the mean and the sd of the prediction errors

generated by the graphical jump method are smaller than those from the other two methods, which means the graphical jump method produces results both more accurate and more stable.



Figure(8) Simulation study results for three dimensional data of sample sizes of 64 ,125 and 216 (the scatter plots, the boxplots and the 2-D histograms describe the same thing as those in Figure(5)).

For Figure (8), everything else is the same as those from sample size of 300 and dimension of one except that the plot containing box plots is divided into two parts with different ranges in x-axis. Since the mean values of prediction errors for 3 nodes are much higher than those of the rest numbers of nodes, they are sketched separately at the left parts. In the right parts of the plots, the box plots have increased trend in mean values along the x-axis, which means that when overfitting happens, the prediction errors increase with larger number of nodes chosen. For all of the .

Table (2).The percentages of the correct number of nodes chosen by each method and the mean (sd) of the prediction errors generated by each method for three dimensional scenarios.

number of datapoints	64	125	216
graphical jump method	100	90	93.3
BIC	23.3	43.3	40
10 fold cross validation	53.3	16.7	46.7
graphical jump method (mean(sd))	1.14e-04(7.68e-05)	3.97e-05(1.18e-05)	3.27e-05(4.21e-06)
BIC (mean(sd))	1.76e-03(3.79e-03)	4.30e-05 (1.30e-05)	3.37e-05(4.36e-06)
10 fold cross validation (mean(sd))	9.09e-04(2.33e-03)	0.0023 (0.0027)	0.00018 (0.00082)

three scenarios in three dimension, the scatter plots indicate that the graphical jump method yields the lowest prediction errors most of the time. Also the spread of the red points is narrower than those of the green points and the dark blue points. The 2-D histograms show that the graphical jump method picks the correct number of nodes (> 90%) most of the time. For sample size of 64, it even yields 100 % correct results. The graphical jump method yields the highest percentages of correct results among the three.

For each of the scenarios above, another set of explanatory variables and response variable are generated the same way as that in each scenario. The newly generat- ed explanatory variables are used to make predictions and the predicted values are compared to the corresponding newly generated response variable.

4 Combined Cycle Power Plant Data Set

We now show our method in use on a real dataset. The dataset contains 9568 observations obtained from a Combined Cycle Power Plant over 6 years (2006-2011), during which the power plant worked with full load. The explanatory variables include average ambient variables Temperature (T), which is in the range of 1.81C and 37.11C, Ambient Pressure (AP), which is in the range of 992.89 to 1033.30 milibar, Relative Humidity (RH), which is in the range of 25.56% to 100.16% and Exhaust Vacuum (V), in the range of 25.36 to 81.56 cm Hg. The output is the net hourly electrical energy output (EP) of the plant, varying from 420.26 to 495.76 MW (Tufekci, 2014). Various sensors are located around the plant. They record the ambient variables every second and the hourly averages are taken as the observations given in the dataset. The final variables in the dataset are not normalized.

The dataset is analyzed using a single layer feedforward neural network. The graphical jump method, the BIC criterion, and 10 fold cross validation are implemented to analyze the dataset and they select 1, 4 and 9 nodes as the best numbers of nodes respectively. Then the full dataset is analyzed using the neural network model using 1, 4 and 9 nodes, separately. The

graphical jump method yields the lowest Mean Square Error (MSE) of residuals, which is 272.317. The BIC criterion and 10 fold cross validation yield higher MSEs of residuals, which are 291.866 and 291.865, separately. Therefore the number of nodes selected by the graphical jump method does the best job in predicting the outcome variable using the explanary variabes. Hence, the graphical jump method does the best job in choosing the true number of nodes.

5 Conclusion, Discussion and Future Research Directions

The jump method was first introduced in Sugar and James (2003). It contains 4 simple steps. It produces a jump score for each number of nodes which are functions of negative transformation of Mean Square Error. Given the correct transformation power, the highest jump score occurs at the true number of nodes. However, in practice, the correct transformation power can be difficult to identify since there are many unknown parameters. Instead, the strategy is to use a jump plot to distinguish false candidates from the true candidate. The ”jump plot” graphs ”the number of nodes selected” versus ”transformation power used”. Theorem 1 demonstrates that if the true number of nodes is bigger than one, then it will be selected in the jump plot, i.e., for the several numbers of nodes selected in the jump plot, one of them should be the true answer. Theorem 2 shows that if the true number of nodes is one, then the jump plot will select one as the only candidate over a long range of transformation powers. Theorem 2 can be used iteratively to rule out false candidates selected from the jump plot. If the candidate number of nodes is less than the true number of nodes, after fitting the neural network model with that number of nodes, the residuals will still need to be explained by more than one node and show a jump larger than one in the jump plot. The method was demonstrated to work on both simulated and real datasets.

The graphical jump method has the potential to be extended to solve other problems where a counting number needs to be chosen, such as choosing the number of species in environmental studies, or choosing the number of neighbors in computer science. Also we could try to choose the number of nodes for other types of neural networks such as classification neural networks, recurrent neural networks (encompassing simple recurrent networks, Long short-term memory (LSTM) networks, Hopfield networks, Echo state networks), region-based convolutional neural network (R-CNN),the growing neural gas network (GNGN), radial basis function networks, and stochastic neural networks (including the Boltsmann machine). Researchers have already studied the topic of model selection for some of the neural networks aforementioned (Decker, 2006; Hessami and Viau, 2004; Liu, 2016). Since all of these neural networks are composed of multiple nodes for data processing, a ”jump plot” might be constructed to find the candidate number of best nodes for the nueral network. Then ”a single node only test” could be designed to rule out erroneous candidates. Another interesting extension would be to multivariate decision problems, such as multi-layer neural networks, where each layer might have a different optimal number of nodes .

Acknowledgments

This research was supported by NSF grant DMS-0906720.

Appendix A: Proof of Theorem 1

Let θ_n be estimated parameter after n modifications by using t samples repeatedly, where in each modification, $\theta_{n+1} = \theta_n - \epsilon(y - f(x, \theta_n))^T \nabla f(x, \theta_n)$. It has been shown that θ_n approaches θ^* as $n \to \infty$ and $\epsilon \to 0$.

Let $\pi_n(\theta_n)$ be the probability distribution of θ_n. $p(\tilde{\theta})$ obeys $\tilde{\pi}(\theta) = \lim_{n\to\infty}\pi_n(\theta)$, where $\tilde{\theta}$ is deemed as the limits of θ_n see Murata, Yoshizwa and Amari (1994) (page 867). If $d(x,y,\theta_{opt}) = -\log p(y|x,\theta_{opt})$, then

$$Q_{opt} = E_q[-log'' p(y|x, \theta_{opt})]$$

$$R_{opt} = V_q[-log' p(y|x, \theta_{opt})] = E_q[(-log' p(y|x, \theta_{opt}))^2] - (E_q[-log' p(y|x, \theta_{opt})])^2$$

since $E_q[log'' p(y \mid x, \theta_{opt})] = -E_q[log(p'(y \mid x, \theta_{opt}))]^2$ and $E_q[-log' p(y|x, \theta_{opt})] = 0$,

see Murata et al. (1994).

$$Q_{opt} = E_q[-log'' p(y|x, \theta_{opt})] = E_q[(-log' p(y|x, \theta_{opt}))^2] = R_{opt}$$

Without loss of generality, assume G is the true number of nodes, MSEq(G)=1. MSEq(k), MSEq*(k) are the MSE under q and q* distribution when fitting k nodes to the neural network model separately. m(k) is the number of parameters for k nodes and m* is the number of parameters in one node.

Here we assume $t \to \infty$, $O(1/t/sqrt(t)) \to 0$ and $O(1/t^{5/2}) \to 0$. Recall the likelihoods $L = \prod_{i=1}^{t} \frac{1}{\sqrt{2\pi}\sigma} exp^{[-(y_i - \beta z_i)^2]/(2\sigma^2)}$, where $z_{ij} = [1+\exp(-\gamma_{j0} - \sum_{h=1}^{r} \gamma_{jh} x_{ih})]^{-1}$. Since the minimum discrepancy $D(q,p(\tilde{\theta}))$=MSEq(G) should occur at the true number of nodes, $\text{Log}(\hat{L}) = -\frac{t}{2} - t\log(\hat{\sigma})$, where $\hat{L}$ and $\hat{\sigma}$ are maximum likelihood estimates. If $d(x,y,\theta) = -\log(p(y|x, \theta))$, then

$$-log(\hat{L}_k) = t/2 + \frac{t}{2} log(\widehat{MSE}_k) \qquad (2)$$

Since $-\log(\widehat{L_k})$ is a function $\widehat{MSE}$, where the minimum of $\frac{1}{2}\widehat{MSE}$ occurs at G nodes, the minimum of $-\log(\widehat{L_k})$ also occurs at G nodes. And R_{opt}=Q_{opt}, $R_{opt}^{-1} - Q_{opt}^{-1}$, if m is the number of

parameters in the corresponding model, R_{opt} is of dimensions m×m, $tr(R_{opt}Q_{opt}^{-1}) = tr(R_{opt}R_{opt}^{-1}) = tr(I_{m\times m}) = m$, (see Murata, Yoshizwa and Amari (1994)(page 868)). Therefore,

$$\text{by (2), let } D(q,p(\widetilde{\theta})) = -log(\widehat{L}_{G+k}(q,p(\widetilde{\theta}))) - log(\widehat{L}_k(q,p(\widetilde{\theta}))) = -log(\widehat{L}_k(q*,p(\widetilde{\theta}))) + U\tfrac{1}{\sqrt{t}} + \tfrac{1}{t}tr(R_{opt}Q_{opt}^{-1}) + O(t^{-3/2})$$

$$= -log(\widehat{L}_k(q*,p(\widetilde{\theta}))) + U\tfrac{1}{\sqrt{t}} + \tfrac{m(k)}{t} + O(t^{-3/2})$$

Since the minimum discrepancy occurs at G nodes,

$$-log(\widehat{L}_{G+k}(q,p(\widetilde{\theta}))) > -log(\overline{\widehat{L}}_G(q,p(\widetilde{\theta})))$$

$$\text{by (2), } -log(\widehat{L}_G(q*,p(\widetilde{\theta}))) + U\tfrac{1}{\sqrt{t}} + \tfrac{m(G)}{t} + O(t^{-3/2}) < -log(\widehat{L}_{G+k}(q*,p(\widetilde{\theta}))) + U\tfrac{1}{\sqrt{t}} + \tfrac{m(G+k)}{t} + O(t^{-3/2})$$

$$-log(\widehat{L}_{G+k}) + \frac{m(G+k)}{t} + O(\frac{1}{t^{3/2}}) > -log(\widehat{L}_G) + \frac{m(G)}{t}$$

$$\text{by(2), } \tfrac{t}{2} + \tfrac{t}{2}log(\widehat{MSE}q^*(G+k)) + U\tfrac{1}{\sqrt{t}} + \tfrac{m(G+k)}{t} + O(\tfrac{1}{t^{3/2}}) > \tfrac{t}{2} + \tfrac{t}{2}log(\widehat{MSE}q^*(G)) + U\tfrac{1}{\sqrt{t}} + \tfrac{m(G)}{t} + O(\tfrac{1}{t^{3/2}}).$$ Therefore,

$$\frac{-t}{2}log(MSEq^*(G+k)) + \frac{t}{2}log(MSEq^*(G)) < \frac{km^*}{t} + O(\frac{1}{t^{3/2}})$$

$$log(\frac{MSEq^*(G)}{MSEq^*(G+k)}) < \frac{2m^*k}{t^2} + O(\frac{1}{t^{5/2}})$$

$$\frac{MSEq^*(G)}{MSEq^*(G+k)} < \exp(\frac{2m^*k}{t^2} + O(\frac{1}{t^{5/2}})) \qquad (3)$$

Then by Taylor Series Expansion, we can prove that

$$\frac{MSEq^*(G)}{MSEq^*(G+k)} < 1 + \frac{2m^*k}{t^2} + O(\frac{1}{t^{5/2}}) \qquad (4)$$

Again via Taylor series expansion, it can be shown that the highest jump score occurs at G nodes,

$$MSEq^*(G)^{-v} - MSEq^*(G-1)^{-v} > MSEq^*(k)^{-v} - MSEq^*(k-1)^{-v}$$

for any k for certain range of v. Additional details are in Chang (2011).

References

[1] Akaike, H. (1973), "Information Theory and an Extension of the Maximum Likelihood Principle," *Second International Symposium on Information Theory*, 1, 267–281.

[2] Anders, U. and Korn, O. (1999), "Model selection in neural networks," *Neural Networks*, 12, 309–323.

[3] Chang, J. (2011), "Topics in model selection: variable selection for computer experiments and choosing the number of nodes for neural networks," Ph.D. thesis, University of California-Santa Cruz, Santa Cruz.

[4] Chang, J. and Sugar, A. C. (2008), "Choosing the number of clusters via the graphical jump method," *Unpublished Manuscript*.

[5] Decker, R. (2006), "A Growing Self-Organizing Neural Network for Lifestyle Segmentation," *Journal of Data Science*, 4, 147–168.

[6] Fogel, D. (1991), "An information criterion for optimal neural network selection, "*IEEE Transactions on Neural Networks*, 5, 490–497.

[7] Gilmour, S. G. (1996), "The interpretation of Mallows's *Cp-statistics*," The Statistician, 45, 49–56.

[8] Girosi, F., Jones, M., and Poggio, T. (1995), "Regularization Theory and Neural Networks Architectures," *Neural Computation*, 7, 219–269.

[9] Hessami, M.and Anctil, F. and Viau, A. (2004), "Selection of an Artificial Neural Network Model for the Post-calibration of Weather Radar Rainfall Estimation," *Journal of Data Science*, 2, 107–124.

[10] Krogh, A. and Hertz, J. (1992), "A Simple Weight Decay Can Improve Generaliza- tion," in *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS* 4, pp. 950–957, Morgan Kaufmann.

[11] Lee, H. K. H. (2004), *Bayesian Nonparametrics via Neural Networks*, Alexandria, Virginia: American Statistical Association, and Philadelphia, Pennsylvania: Society for Industrial and Applied Matematics.

[12] Liu, D.and Huang, S. (2016), "The Performance of Hybrid Artificial Neural Network Models for Option Pricing during Financial Crises," *Journal of Data Science*, 14, 1–18.

[13] Murata, N., Yoshizwa, S., and Amari, S. (1994), "Network information criteria: deermine the number of hidden units for an artificial neural network model," *IEEE Transactions on Neural Networks*, 5, 865–872.

[14] Sarle, W. (1995), "Stopped Training and Other Remedies for Overfitting," in *Proceedings of the 27th Symposium on the Interface*, pp. 352–360.

[15] Schwarz, G. (1978), "Estimating the Dimension of a Model," *Annals of Statistics*, 6, 461–464.

[16] Sheela, K. and Deepa, S. (2013), "Review on Methods to Fix Number of Hidden Neurons in Neural Networks," *Mathematical Problems in Engineering*, 2013, 11 pages.

[17] Sugar, C. and James, G. (2003), "Finding the number of clusters in a data set: An information theoretic approach," *Journal of the American Statistical Association*, 98, 750–763.

[18] Tufekci, P. (2014), "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods, "*International Journal of Electrical Power and Energy Systems*, 60, 126–140.

[19] Xu, S. and Chen, L. (2008), "A Novel Approach for Determining the Optimal Number of Hidden Layer Neurons for FNN's and Its Application in Data Mining," *5th International Conference on Information Technology and Applications*, pp. 683– 686.

[1]Jing Chang (Corresponding Author)
Hunan University of Art and Science
Anhui Provincial Hospital first staff dormitory, Unit 1, Room 2104, Hefei, Anhui, China, 230001
Phone:(86)18256056636
Email ID:jingbb@gmail.com

[2]Herbert K. H. Lee
University of California, Santa Cruz
Postal address: UC Santa Cruz, School of Engineering, 1156 High Street, Santa Cruz, CA,95064.
Email ID:herbie@ams.ucsc.edu