CORE
Metadata, citation and similar papers at core.ac.uk
Provided by Brunel University Research Archive

# An Incremental Approach to MSE-Based Feature Selection

Sheng-Uei Guan[1], Yinan Qi[2] and Chunyu Bao[2]

[1] School of Engineering and Design
Brunel University, UK

[2]Department of Electrical and Computer Engineering,
National University of Singapore

**Abstract**

Feature selection plays an important role in classification systems. Using classifier error rate as the evaluation function, feature selection is integrated with incremental training. A neural network classifier is implemented with an incremental training approach to detect and discard irrelevant features. By learning attributes one after another, our classifier can find directly the attributes that make no contribution to classification. These attributes are marked and considered for removal. Incorporated with a Minimum Squared Error (MSE) based feature ranking scheme, four batch removal methods based on classifier error rate have been developed to discard irrelevant features. These feature selection methods reduce the computational complexity involved in searching among a large number of possible solutions significantly. Experimental results show that our feature selection methods work well on several benchmark problems compared with other feature selection methods. The selected subsets are further validated by a Constructive Backpropagation (CBP) classifier, which confirms increased classification accuracy and reduced training cost.

***Keywords:*** Feature selection, Classifier, Neural network, Feedforward neural network, Minimum squared error (MSE), Incremental training, Input attribute

## I. Introduction

In real-world problems, manual feature selection is often impossible to achieve due to the large number of features. Therefore, feature selection is necessary in these problems. The goal of feature selection is to find those features that may neither affect the target in any way (called irrelevant features) nor add anything new to the target (called redundant features) [1] and exclude them. Current feature selection methods can be classified as the "filter" model and "wrapper" model. The "filter" model is independent with the induction algorithm. On the contrary, the "wrapper" model wraps around the induction algorithm and search for the best feature subset according to the performance of the induction algorithm.

Using the filter model, features are selected or discarded based upon some predefined criteria such as mutual information [4][25], principal component analysis [26][27], independent component analysis [28] and class separability measure [20][21]. Usually, the filter model may not be as effective and general as the wrapper model because the model does not consider the relation between feature subset and the performance of induction algorithm. Feature subset selection must take into account the biases of the induction algorithm in order to perform well [3]. The wrapper model requires a large amount of training but provides highly accurate feature selection. In the wrapper model, a wide variety of classifiers are used for feature selection based on different search methods and evaluation functions like ID3 [18], C4.5 [8], CART [10], NNFS[22], linear classifier [5] and box classifier [6], etc. For example, for the feature selector NNFS presented by Setiono and Liu [22], a three-layer, feedforward neural network is used as a tool to determine irrelevant features. The network is trained with the complete set of attributes as input. For each attribute, the accuracy of the network is computed with all the weights of the

connections associated with this attribute set to zero. The attribute that gives the smallest decrease in network accuracy is removed. The network is then retrained and the process is repeated.

In this research, we integrate feature selection with an incremental neural network training approach. With one training iteration only, this approach can find out irrelevant features quickly. Guided by the performance of the NN being trained, the accuracy of this feature selection approach is relatively high. Incremental Training with Increasing Input Dimension (ITID) is a new NN training method recently presented by Guan et al. [19]. Instead of training input attributes in batch, this incremental training method trains input attributes one by one. The NN structure grows incrementally in correspondence to an increasing input dimension and network performance keeps refined when each new attribute comes in. Network performance (e.g. training time, independent parameters, training error, test error, and classification error) is traced for each newly introduced attribute during training. If network performance is decreased by the introduction of a new attribute, it indicates that this attribute could be inconsistent with the previous attributes or irrelevant to the output target. Otherwise, this attribute could be a relevant feature and contributes to training. The contribution of an input attribute is evaluated based on the traced network performance. The attributes with no or little contribution will be discarded.

We evaluate the individual discrimination ability of each attribute before training using a NN with only one input attribute. The attribute with the best discrimination ability will be set as the default and introduced first, followed by those attributes with lower discrimination ability. In Contribution-based ITID [19], the evaluation of individual discrimination ability of an input

attribute is done by a NN with only one input attribute in the input layer. This NN is trained to fit all the output targets with the specific attribute. Hence, the individual discrimination ability of this attribute can be evaluated.

Here in this paper, we use a new approach to evaluate the discrimination ability of attributes. This approach reduces computations and could get better performance. Before feature selection, input attributes are pre-ranked by their goodness scores based on MSE weights. The attributes with better goodness scores are introduced and detected first, those with worse goodness scores are introduced and detected later. The best-first detection approach can reduce search scope and improve accuracy of feature selection. Four batch removal methods based on attribute performance and network accuracy are developed to discard irrelevant features.

Different from the other wrapper methods, our classifier can detect irrelevant /redundant features directly based on the traced network performance and the evaluation function is simple. Instead of detecting a single irrelevant feature in one iteration, our classifier can detect and discard features in batch. The computational cost involved in searching among a large number of possible solutions is significantly lowered. Further validated by a CBP [14] classifier, the accuracy of our feature selection methods is high. The details of MSE weights and ITID training algorithm are presented in Section II and III. The feature selection algorithm is described in Section IV. The experimental results are reported accordingly in Section V. The last section includes some discussions and conclusions about our feature selection methods.

**II. MSE algorithm**

MSE is often used in statistical classification problems. For example, in a 3-class problem, we set the input attributes as a vector **x** and Class 1, Class2 and Class 3 as $\omega_1$, $\omega_2$ and $\omega_3$ respectively**.** Define $\mathbf{y}_i = [1 \quad \mathbf{x}_i]^T$, such that there are $n_i$ vectors $\mathbf{y}_i$ from the $i^{th}$ class, $i$ = 1; 2; 3, with $\sum_{i=1}^{3} n_i$ = number of total training patterns. According to [2][11][16], the vectors are said to be linearly separable if there exists a linear machine that can classify all of them correctly, i.e. there exist a set of weight vectors, $\mathbf{a}_1, \mathbf{a}_2, \ldots\ldots, \mathbf{a}_c$, such that

$$\text{If } \mathbf{y} \in \omega_i, \text{ then } \mathbf{a}_i^T\mathbf{y} > \mathbf{a}_j^T\mathbf{y}, \text{ for all } j, j \neq i$$

Here

$$t_i = \mathbf{a}_i^T\mathbf{y} = a_{0i} + a_{1i}x_1 + a_{2i}x_2 + \cdots + a_{mi}x_m, \qquad (1)$$

*where m* is the number of attributes.

Our goal is to obtain a weight vector $\mathbf{a}_i$ that is a MSE solution to

$$\mathbf{a}_i^T\mathbf{y} = \begin{cases} +1 & \text{for all } \mathbf{y} \in \omega_i \\ -1 & \text{for all } \mathbf{y} \notin \omega_i \end{cases}$$

The pseudo-inverse solution to the multi-class MSE problem can be written as the following.

Define

$$\mathbf{Y}_i = \begin{bmatrix} \cdots \\ \mathbf{y}^T \\ \cdots \end{bmatrix}_{n_i \times (d+1)} \qquad \mathbf{y} \in \omega_i, \ \mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \\ \cdots \\ \mathbf{Y}_c \end{bmatrix}_{n \times (d+1)} \qquad \mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \ldots & \mathbf{a}_c \end{bmatrix}_{(d+1)\times c} \text{ and say } \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \\ \cdots \\ \mathbf{B}_c \end{bmatrix}_{n \times c}$$

where $\mathbf{B}_i$ are $n_i$-*by*-$c$ matrix with all the elements of it zero except for those in the $i^{th}$ column, which are unity.

Then, the trace of the squared error matrix $(\mathbf{YA} - \mathbf{B})^T(\mathbf{YA} - \mathbf{B})$ is minimized by the solution

$$\mathbf{A} = \mathbf{Y}^{\dagger}\mathbf{B} \qquad (2)$$

where $\mathbf{Y}^{\dagger}$ is the pseudoinverse of **Y.**

The weights in vector $\mathbf{a}_i$ can be used to scale the influence of each input attribute, which means the weights can be used to select relevant features. And this feature selection method can be applied without task decomposition.

Since the first weight will be multiplied by 1, it has no influence. The remaining 8 weights can be used to scale the relevance of each input attribute. For convenience, we can normalize the weights first.

$$b_i = \frac{|a_i|^2}{\sum_i |a_i|^2}$$

For a multi-class problem, it is natural that the final weight vector which will be used to scale the relevance of the input attributes should be the average of the 6 weight vectors.

Now we show that the MSE method is also applicable for the sigmoid neural network. In a backpropagation neural networks, $c$ output nodes are used to represent $c$ output classes. For each

output node, it is also a two-class problem. The task of output node *i* is to see if an input pattern belongs to class *i*, such as $\mathbf{a}_i$ in MSE. So each $\mathbf{a}_i$ in the MSE problem corresponds to an output node *i* in a backpropagation NN. In our three-layer, feedforward networks, sigmoid function is used as activation function in the hidden layer and the output layer. The activation function in a backpropagation NN is

$$f(x) = \frac{1}{1+e^{-x}}$$

For each output node, the output can be rewritten as:

$$t_i^{'} = \frac{1}{1+\exp\left[-\left(A_0 + A_1 x_1 + A_2 x_2 + \cdots + A_m x_m + B_1 f(A_0^{'} + A_0^{'} x_1 + \cdots + A_m^{'} x_m) + B_2 f(\cdots) + \cdots\right)\right]}$$ Using

Taylor's expansion,

$$t_i^{'} = C_{0i} + C_{1i} x_1 + C_{2i} x_2 + \cdots + C_{mi} x_m + D_{1i} x_1^2 + D_{2i} x_1 x_2 + \cdots + D_{(\frac{n+1}{2}-n)i} x_n^2 + o(x^2) \qquad (3)$$

For our problems, the input data set has been normalized between [0 1]. Generally, $x > x^2 > x^3 > \cdots$, we could expect that the first order terms have more influence on the final result especially when *x* is small compared with 1. Comparing the above equation with its counterpart in MSE, we can see that $w_{0i}$ corresponds to $C_{0i}$, $w_{1i}$ corresponds to $C_{1i}$, … , $w_{mi}$ corresponds to $C_{mi}$. The $o(x^2)$ term and $D_{ji}$ terms in equation (3) is used to correct the former terms to generate better result while ensuring $0 < t_i^{'} < 1$. If $x_i << 1$, we can merge the $o(x^2)$ term with the $D_{ji}$ terms in equation (3) then equation (3) can be rewritten as

$$t_i^{'} = C_{0i} + C_{1i} x_1 + C_{2i} x_2 + \cdots + C_{mi} x_m + o(x) \qquad (4)$$

Comparing equation (4) with equation (1), $C_{mi}$ should be close to $w_{mi}$ in proportion. That means $C_{ji} \approx A \cdot w_{ji}$, here $j=1,2,\cdots,m$ and *A* is a constant. $C_{0i}$ and $w_{0i}$ are the bias and may not satisfy the

above relationship. In equation (1), the larger $\|w_{ji}\|$, the more influence $x_j$ has on $t_i$. That means, the larger $\|w_{ji}\|$, the more important $x_j$ is. A larger $w_{ji}$ in equation (1) corresponds to a larger $C_{ji}$ in equation (4), that means $x_j$ has more influence on $t_i^{'}$. In other words, $x_j$ is an important feature to output node $i$. This explains why feature selection based on MSE can be used in backpropagation NNs.

If $x_i$ is not very small, especially when $x_i$ approaches 1, the above analysis may not hold. But first-order terms still have remarkable influence to the final outputs. So feature selection considering only the effect of first-order terms (in another word, MSE) is feasible in backpropagation NNs.

## III. ITID

An ITID classifier is a three-layer feedforward neural network trained using the ITID method. The input space of this classifier is divided into several sub-dimensional input spaces, each of which has one input attribute correspondingly. During training, the input attributes are introduced one by one and the input dimension of this classifier is increased successively. When an input attribute is introduced into this classifier, a corresponding subnetwork is trained to obtain information from this attribute. The subnetwork is further merged with those of the previously trained subnetworks to refine the network performance. The network structure of an ITID classifier is shown in Figure 1. For each subnetwork, there is a set of hidden units that are linked to the corresponding input units. For the whole network, the connections between the input layer and hidden layer are only established between the hidden units to their corresponding

input units. Details of training procedures, error measures and stopping criteria of the ITID classifier are described in Appendix A.

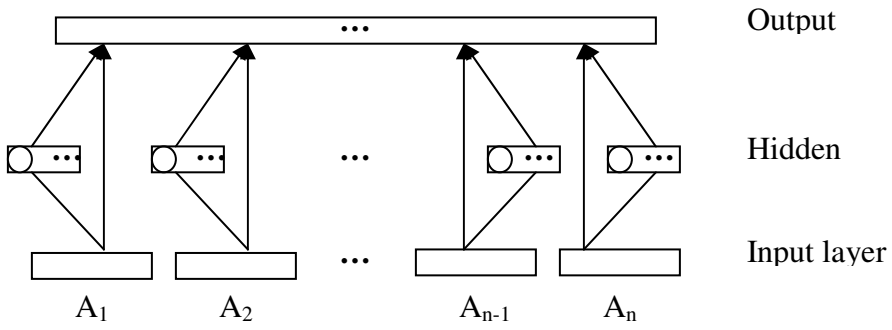


Figure 1. Network structure of the ITID classifier

Legends: "$A_i$" stands for the i-th input attribute.

During ITID training, each input attribute could bring along some information into the neural network being trained. Some of them are useful for the classification target, the others may be useless or even harmful. Network performance can be traced with respect to each input attribute by taking a snapshot of the merged network after an attribute is introduced. This snapshot records some information measuring network performance (i.e. number of epochs, training time, number of hidden units, number of independent parameters, training error, test error and classification error). Through this record, whether an attribute contributes to network training and classification target could be detected according to some criterion described in the next section.

## IV. Feature Selection algorithm

Based on the MSE weights, we present four feature selection methods. The first two methods

only consider MSE weights and do not include the ITID process. The other two methods use ITID to further detect the relevance of the features.

As mentioned above, network performance is traced for each attribute during training. The attribute with the best discrimination ability or the largest MSE weight will be set as the default attribute and introduced first, followed by those attributes with lower discrimination ability. This default attribute will always appear in the subset after feature selection and cannot be discarded. The other features are subject to feature selection. The contribution of an attribute is evaluated by the network error reduction rate, *R*, when this attribute is introduced:

$$R_n = -\frac{E_n - E_{n-1}}{E_{n-1}} \ ;$$

$R_n$ is the reduction rate contributed by the *n*-th input attribute, $E_n$ is the test error (for regression problems) or classification error (for classification problems) of the validation set when the *n*-th input attribute is added to the network. If the reduction rate of classification error/test error is negative, the newly-added attribute is regarded as useless. Therefore this attribute could be an irrelevant or redundant feature and will be removed.

Now some features that are irrelevant or redundant are removed first based on the above observation. Furthermore, some of the remaining features may also be removed. We use the 'knock-out' technique to the remaining attributes to see whether it could be discarded. The 'Knock-out' technique is described as follows:

(1) Order the features according their MSE weights. Set up the ITID network according to the order. The most important feature is introduced to the network first.

(2) Evaluate the features' contribution using their error reduction rate. Starting from the feature with the smallest MSE weight, if the introduction of that attribute is harmful to the overall accuracy, then the attribute will be knocked out. Otherwise, end the 'knock-out' process.

(3) Repeat Step 2 for the remaining features.

Four feature selection methods are provided:

**Method 1:** Set a specific threshold. If the MSE weight of one feature is larger than the threshold, it is selected, otherwise, discarded.

**Method 2**: After the weights of the features are compared with the threshold, the remaining attributes are further validated by knock-out to get the final feature subset.

**Method 3**: Feature selection is performed after incremental training: All the input attributes will go through the incremental ordered training first, after that, the contribution from all the attributes (except the default attribute) will be calculated. Those attributes with negative contribution will be discarded in batch, and those attributes with positive contribution will be selected. Then the remaining attribute set will be validated further by knock-out to get the final feature subset.

**Method 4:** Feature selection is integrated with incremental training: When an input attribute is introduced into a NN, a new subnetwork is trained and network performance is refined. Calculate the contribution of this attribute, if its contribution is negative, then discard this attribute and

continue incremental training without this attribute. If the contribution of this attribute is positive, then it will be selected. We continue incremental training with this attribute. Training continues until we have run through all the attributes. Then the attribute set formed from the selected attributes will be validated further by 'knock-out' to find the final feature set.

For the comparison of the four methods, we list them in the following table:

| **Four Feature Selection methods** | |
|---|---|
| Method 1 | (1) Order the features according their MSE weights.<br>(2) Compare the MSE weight with threshold.<br>MSE weight ≥ threshold, keep the feature<br>MSE weight ≤ threshold, discard the feature |
| Method 2 | (1) Order the features according their MSE weights.<br>(2) Compare the MSE weight with threshold.<br>MSE weight ≥ threshold, keep the feature<br>MSE weight ≤ threshold, discard the feature<br>(3) Check the non-discarded features using knock-out. |
| Method 3 | (1) Order the features according their MSE weights.<br>(2) Set up the ITID network based on the order of the features above.<br>(3) Compute the contribution of the features (except the first feature), in other words, $R_n$ is computed. If $R_n \leq 0$, discard the feature.<br>(4) Check the remaining features using knock-out. |
| Method 4 | (1) Order the features according their MSE weights.<br>(2) Set up the ITID network using the first feature.<br>(3) Add other features one by one to the ITID network. Once a feature is added to the network, its contribution is evaluated using $R_n$. If $R_n \leq 0$, discard the feature and its correspondent sub-network. Otherwise, keep the feature.<br>(4) Check the non-discarded features using knock-out. |

**Validation**

Two validation procedures are commonly used for feature selection: using artificial datasets and using real world datasets [13]. In the first procedure, artificial datasets are constructed for a certain target concept, all the actual relevant features for this concept are known. Validation procedures check whether the selected subset is the same as the actual subset.

The second procedure works by testing the accuracy of the selected subset with the help of a suitable classifier. In this research we use some real world datasets to validate our feature selection results. The validation steps are as follows:

**Step 1**: Order the selected attributes by contribution and train the NN using all selected attributes;

**Step 2**: Knock out the attribute with the least contribution and compare the performance of the current NN (without this attribute) with that of the previous NN (with this attribute inside);

**Step 3**: If knock-out results in performance improvement, then discard this attribute, go to Step 2, otherwise, restore the attribute knocked out and stop the knock-out process.

**V. Experiments and Simulation results**

The proposed feature selection methods are tested on four benchmark problems taken from the PROBEN1 benchmark collection: Diabetes, Cancer, Glass, and Vowel [12].

**1. Diabetes**

Diabetes is a two-class classification problem. There are 8 continuous input attributes in this dataset used to diagnose whether a Pima Indian has diabetes or not. There are 768 patterns in this dataset, 65% of the total patterns belong to class 1 (no diabetes), 35% of the total patterns belong to class 2 (diabetes).

At first, we calculate the MSE weight of each feature. The experimental results are shown in Table 1.

| Attribute | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| MSE weight | 0.0479 | 0.5463 | 0.0321 | 0.0001 | 0.0090 | 0.3087 | 0.0463 | 0.0096 |

Table 1. MSE weights of the input attributes - Diabetes

**Method 1**: Here we set the threshold as 0.05. According to Table 1, only attributes 2 and 6 are selected and the classification error decreases from 23.93 to 21.12.

**Method 2**: Since we will further validate the attributes whose weights are larger than the threshold, we can set a smaller threshold, say 0.03, in order to include more attributes at the beginning. The validation process is shown in Table 2. And the final feature subset is {2,6,1,7,3}. The classification error decreases to 22.19.

| Feature subset | 2,6,1,7,3 | 2,6,1,7 |
|---|---|---|
| C.error | 22.19 | 22.29 |

Table 2. Results of Method 2 - Diabetes

Legend: "C.Error" is the classification error of validation set.

**Method 3**: The ranking order of the input attributes is 2>6>1>7>3>8>5>4. After incremental ordered training, we can get a performance snapshot (Table 3). From the results we can see that attributes 2,6,3,5 are selected. The validation results are shown in Table 4. Therefore, the final feature subset is {2,6}.

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 2 | 23.54166 | | Default |
| 6 | 22.78647 | 3.2079% | Selected |
| 1 | 23.67188 | -3.8857% | Discarded |
| 7 | 23.776045 | -0.4400% | Discarded |
| 3 | 23.02084 | 3.1763% | Selected |
| 8 | 23.072915 | -0.2262% | Discarded |
| 5 | 22.656245 | 1.8059% | Selected |
| 4 | 22.838545 | -0.8046% | Discarded |

Table 3. Ordered incremental training performance snapshot - Diabetes (Method 3)

| Feature subset | 2,6,3,5 | 2,6,3 | 2,6 | 2 |
|---|---|---|---|---|
| C.error | 23.26 | 22.19 | 21.12 | 23.54 |

Table 4. Results of Method 3 - Diabetes

**Method 4**: The performance snapshot is shown in Table 5. Attribute 2,6,5,8,3,7,4 are selected and will be further validated (Table 6). The final subset is {2,6,5,8,3}.

Table 7 shows that in Method 1 and 3, the training cost is significantly reduced by 45.48% for the number of training epochs, 61.01% for training time, 43.84% for the number of hidden units, and 73.39% for the number of independent parameters and the classification error is reduced by 11.75%. In Method 2, the training cost is reduced by 49.36% for the number of training epochs, 69.25% for training time, 37.44% for the number of hidden units and 57.99% for the number of independent parameters and the classification error is reduced by 5.54%. In Method 4, , the training cost is reduced by 39.31% for the number of training epochs, 65.03% for training time, 8.37% for the number of hidden units and 42.07% for the number of independent parameters and the classification error is reduced by 8.02%.

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 2 | 23.54166 | | Default |
| 6 | 22.78647 | 3.2079% | Selected |
| 1 | 23.67188 | -3.8857% | Discarded |
| 7 | 22.73438 | 0.2286% | Selected |
| 3 | 22.50001 | 1.03% | Selected |
| 8 | 22.08333 | 1.85% | Selected |
| 5 | 21.40625 | 3.01% | Selected |
| 4 | 21.3802 | 0.12% | Selected |

Table 5. Ordered incremental training performance snapshot - Diabetes (Method 4)

| Feature subset | 2,6,5,8,3,7,4 | 2,6,5,8,3,7 | 2,6,5,8,3 | 2,6,5,8 |
|---|---|---|---|---|
| C.error | 23.49 | 22.16 | 22.01 | 22.42 |

Table 6. Results of Method 4 – Diabetes

| Method | | Epochs | T. Time (s) | Hidden Units | Indp. Param. | C. Error (%) | Standard Deviation |
|---|---|---|---|---|---|---|---|
| Diabetes1 (Before selection) | | 8869.5 | 49.75 | 10.15 | 129.65 | 23.93229 | 1.15 |
| Method 1 Subset 1 {2,6} | | 4835.25 | 19.4 | 5.7 | 34.5 | 21.119785 | 1.21 |
| | Reduction | 45.48% | 61.01% | 43.84% | 73.39% | 11.75% | -- |
| Method 2 Subset 2 {2,6,1,7} | | 4491.75 | 15.3 | 6.35 | 54.47 | 22.29 | 1.09 |
| | Reduction | 49.36% | 69.25% | 37.44% | 57.99% | 5.54% | -- |
| Method 3 Subset 3 {2,6} | | 4835.25 | 19.4 | 5.7 | 34.5 | 21.119785 | 1.21 |
| | Reduction | 45.48% | 61.01% | 43.84% | 73.39% | 11.75% | -- |
| Method 4 Subset 4 {2,6,5,8} | | 5471 | 17.4 | 9.3 | 75.1 | 22.01 | 1.13 |
| | Reduction | 38.31% | 65.03% | 8.37% | 42.07% | 8.02% | -- |

Table 7. Validation results – Diabetes

**2. Cancer**

Cancer is a two-class classification problem that diagnoses breast cancer. The dataset includes 9 inputs, 2 outputs, and 699 patterns. All inputs are continuous: 66% of the total patterns belong to

class 1 (benign) and 34% of the total patterns belong to class 2 (malignant). The MSE weights are shown in Table 8.

| Attribute | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| MSE weight | 0.2251 | 0.1069 | 0.0554 | 0.0083 | 0.0130 | 0.4379 | 0.0895 | 0.0623 | 0.0017 |

Table 8. MSE weights of the input attributes - Cancer

**Method 1**: Here we set the threshold as 0.05. According to Table 8, attributes 6,1,2,7,8,3 are selected and the classification error decreases from 1.87 to 1.38.

**Method 2**: Set a smaller threshold, say 0.03. The validation process is shown in Table 9. And the final feature subset is {6,1,2,7,8,3}.

| Feature subset | 6,1,2,7,8,3 | 6,1,2,7,8 |
|---|---|---|
| C.error | 1.38 | 2.24 |

Table 9. Results of Method 2 – Cancer

**Method 3**: The ranking order of the input attributes is 6>1>2>7>8>3>5>4>9. After incremental ordered training, we can get a performance snapshot (Table 10). From the results we can see that attributes 6,1,2,7,8,3 are selected. The validation results are shown in Table 11. Therefore, the final feature subset is {6,1,2,7,8,3}.

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 6 | 9.77011 | | Default |
| 1 | 3.563222 | 63.53% | Selected |
| 2 | 2.816089 | 20.97% | Selected |
| 7 | 2.4425275 | 13.27% | Selected |
| 8 | 1.954024 | 20.00% | Selected |
| 3 | 1.9252885 | 1.47% | Selected |
| 5 | 1.9252885 | 0.00% | Discarded |
| 4 | 1.9252885 | 0.00% | Discarded |
| 9 | 1.9252885 | 0.00% | Discarded |

Table 10. Ordered incremental training performance snapshot - Cancer (Method 3)

| Feature subset | 6,1,2,7,8,3 | 6,1,2,7,8 |
|---|---|---|
| C.error | 1.38 | 2.24 |

Table 11. Results of Method 3 – Cancer

**Method 4**: The performance snapshot is shown in Table 12. Attribute 6,1,2,7,8,3 are selected and will be further validated (Table 13). The final subset is {6,1,2,7,8,3}.

The results from these four methods are compared in Table 14. Table 14 shows that the four methods have same performance. The training cost is significantly reduced by 55.96% for the number of training epochs, 77.21% for training time, 53.93% for the number of hidden units, and 61.51% for the number of independent parameters and the classification error is reduced by 26.20%.

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 6 | 9.77011 | | Default |
| 1 | 3.563222 | 63.53% | Selected |
| 2 | 2.816089 | 20.97% | Selected |
| 7 | 2.4425275 | 13.27% | Selected |
| 8 | 1.954024 | 20.00% | Selected |
| 3 | 1.9252885 | 1.47% | Selected |
| 5 | 1.9252885 | 0.00% | Discarded |
| 4 | 2.2126435 | -13.5% | Discarded |
| 9 | 2.183908 | -13.75% | Discarded |

Table 12. Ordered incremental training performance snapshot - Cancer (Method 4)

| Feature subset | 6,1,2,7,8,3 | 6,1,2,7,8 |
|---|---|---|
| C.error | 1.38 | 2.24 |

Table 13. Results of Method 4 – Cancer

| Method | | Epochs | T. Time (s) | Hidden Units | Indp. Param. | C. Error (%) | Standard deviation |
|---|---|---|---|---|---|---|---|
| Cancer1 (Before selection) | | 5927.25 | 32.25 | 13.35 | 180.2 | 1.87 | 0.34 |
| Method 1,2,3 and 4 | | 2669.75 | 7.35 | 6.15 | 69.35 | 1.38 | 0.28 |
| Subset (6,1,2,7,8,3) | Reduction | 55.96% | 77.21% | 53.93% | 61.51% | 26.20% | -- |

Table 14. Validation results – Cancer

**3. Glass**

Glass studies the classification of glass types. There are 9 input attributes, 6 outputs, and 214 patterns in the Glass1 dataset. We calculate their weights in Table 15.

| Attribute | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| MSE weight | 0.1804 | 0.0980 | 0.0713 | 0.1954 | 0.0782 | 0.1732 | 0.1025 | 0.0950 | 0.0059 |

Table 15. MSE weights of the input attributes - Glass

**Method 1**: Here we set the threshold as 0.05. According to Table 15, only attributes 9 is not selected and the classification error decreases from 41.23 to 35.66.

**Method 2**: Set a smaller threshold 0.03. The validation process is shown in Table 16. And the final feature subset is {4,1,6,7,2,8,5}.

| Feature subset | 4,1,6,7,2,8,5,3 | 4,1,6,7,2,8,5 | 4,1,6,7,2,8 |
|---|---|---|---|
| C.error | 36.42 | 32.45 | 34.15 |

Table 16. Results of Method 2 - Glass

**Method 3**: The ranking order of the input attributes is 4>1>6>7>2>8>5>3>9. After incremental ordered training, we can get a performance snapshot (Table 17). From the results we can see that attributes 4,6,8,2,5 are selected. The validation results are shown in Table 18. Therefore, the final feature subset is {4,6,8,2,5}.

**Method 4**: The performance snapshot is shown in Table 19. Attribute 4,8,7,6 are selected and will be further validated (Table 20). The final subset is {4,8,7,6}.

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 4 | 38.49055 | | Default |
| 1 | 42.924505 | -11.52% | Discarded |
| 6 | 38.11319 | 11.21% | Selected |
| 7 | 39.99997 | -4.9505% | Discarded |
| 2 | 38.96224 | 2.5943% | Selected |
| 8 | 35.471715 | 8.9587% | Selected |
| 5 | 35.000005 | 1.3298% | Selected |
| 3 | 36.41509 | -4.0431% | Discarded |
| 9 | 38.301865 | -5.1813% | Discarded |

Table 17. Ordered incremental training performance snapshot - Glass (Method 3)

| Feature subset | 4,6,8,2,5 | 4,6,8,2 |
|---|---|---|
| C.error | 36.60 | 38.02 |

Table 18. Results of Method 3 - Glass

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 4 | 38.49055 | | Default |
| 1 | 42.924505 | -11.52% | Discarded |
| 6 | 38.113225 | 0.98% | Selected |
| 7 | 35.56606 | 6.68% | Selected |
| 2 | 38.490555 | -6.73% | Discarded |
| 8 | 31.79248 | 10.61% | Selected |
| 5 | 33.962285 | -7.10% | Discarded |
| 3 | 36.886785 | -16.77% | Discarded |
| 9 | 34.52831 | -9.68% | Discarded |

Table 19. Ordered incremental training performance snapshot - Glass (Method 4)

| Feature subset | 4,8,7,6 | 4,8,7 |
|---|---|---|
| C.error | 32.45 | 34.06 |

Table 20. Results of Method 4 - Glass

The results from these four methods are compared in Table 21. Table 21 shows that in Method 1, the training cost is increased by 55.41% for the number of training epochs, 25.27% for training

time, 18.21% for the number of hidden units, and 6.56% for the number of independent parameters but the classification error is reduced by 13.50%. In Method 2, the training cost is increased by 116.75% for the number of training epochs, 78.67% for training time, 24.40% for the number of hidden units 2.94% for the number of independent parameters but the classification error is reduced by 21.30%. In Method 3, the number of training epochs increases for 28.87%, 14.43% for number of hidden units, but training time decreases for 23.84% and number of independent parameters for 19.47. The classification error is reduced by 11.23%.In Method 4, the training cost is increased by 90.26% for the number of training epochs, 26.52% for training time, 15.46% for the number of hidden units but the number of independent parameters is decreased for 26.64% and the classification error is reduced by 21.30%.

| Method | | Epochs | T. Time (s) | Hidden Units | Indp. Param. | C. Error (%) | Standard Deviation |
|---|---|---|---|---|---|---|---|
| Glass1 (Before selection) | | 8375 | 27.9 | 14.55 | 292.8 | 41.23 | 4.43 |
| Method 1 Subset1 {1,2,3,4,5,6,7,8} | Reduction | 13016 –55.41% | 34.95 –25.27% | 17.2 –18.21% | 312 -6.56% | 35.66 13.50% | 4.27 -- |
| Method 2 Subset2 {4,1,6,7,2,8,5} | Reduction | 18153 –116.75% | 49.85 –78.67% | 18.1 –24.40% | 301.4 -2.94% | 32.45 21.30% | 3.40 -- |
| Method 3 ,Subset3 {4,6,8,2,5} | Reduction | 10793.25 –28.87% | 21.25 23.84% | 16.65 –14.43% | 235.8 19.47% | 36.60 11.23% | 4.23 -- |
| Method 4 Subset4 {4,8,3,2} | Reduction | 15934.5 –90.26% | 35.3 –26.52% | 16.8 –15.46% | 214.8 26.64% | 32.45 21.30% | 3.86 -- |

Table 21. Validation results – Glass

**4. Vowel**

Vowel studies the classification of vowels. There are 10 input attributes, 11 outputs, and 990 training patterns in the Vowel1 dataset. We calculate their weights in table 22.

| Attribute | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| MSE weight | 0.2078 | 0.2206 | 0.0888 | 0.0394 | 0.1192 | 0.0930 | 0.0562 | 0.1202 | 0.0223 | 0.0326 |

Table 22. MSE weights of the input attributes - Vowel

**Method 1**: Here we set the threshold as 0.05 according to Table 1, only attributes 4,9 and 10 are discarded and the classification error increases from 34.73 to 38.38.

**Method 2**: Set a smaller threshold 0.03. The validation process is shown in Table 23. And the final feature subset is {1,2,3,4,5,6,7,8}.

| Feature subset | 1,2,3,4,5,6,7,8,10 | 1,2,3,4,5,6,7,8 | 1,2,3,5,6,7,8 |
|---|---|---|---|
| C.error | 35.06 | 32.94 | 38.38 |

Table 23. Results of Method 2 - Vowel

**Method 3**: The ranking order of the input attributes is 1>2>8>5>6>3>7>4>10>9. After incremental ordered training, we can get a performance snapshot (Table 24). From the results we can see that attributes 1,2,5,8,4,9 are selected. The validation results are shown in Table 25. Therefore, the final feature subset is {1,2,5,8,4,9}.

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 1 | 76.3360 | | Default |
| 2 | 57.5709 | 24.58% | Selected |
| 8 | 54.9595 | 4.54% | Selected |
| 5 | 52.4299 | 4.60% | Selected |
| 6 | 52.9352 | -0.96% | Discarded |
| 3 | 53.0162 | -0.15% | Discarded |
| 7 | 53.0567 | -0.01% | Discarded |
| 4 | 51.9028 | 2.17% | Selected |
| 10 | 52.2875 | -0.74% | Discarded |
| 9 | 52.0648 | 0.43% | Selected |

Table 24. Ordered incremental training performance snapshot - Vowel (Method 3)

| Feature subset | 1,2,5,8,4,9 | 1,2,5,8,4 |
|---|---|---|
| C.error | 32.79 | 35.61 |

Table 25. Results of Method 3 - Vowel

| Input attribute | C. Error (%) | C.Error Reduction rate | Status |
|---|---|---|---|
| 1 | 76.3360 | | Default |
| 2 | 57.5709 | 24.58% | Selected |
| 8 | 54.9595 | 4.54% | Selected |
| 5 | 52.4299 | 4.60% | Selected |
| 6 | 52.9352 | -0.96% | Discarded |
| 3 | 52.05 | 0.74% | Selected |
| 7 | 52.36 | -0.60% | Discarded |
| 4 | 49.22 | 6.00% | Selected |
| 10 | 53.04 | -7.76% | Discarded |
| 9 | 50.01 | -1.61% | Discarded |

Table 26. Ordered incremental training performance snapshot - Vowel (Method 4)

| Feature subset | 1,2,4,5,8,3 | 1,2,4,5,8 |
|---|---|---|
| C.error | 31.50 | 35.41 |

Table 27. Results of Method 4 - Vowel

**Method 4**: The performance snapshot is shown in Table 26. Attribute 1,2,4,5,8,3 are selected and will be further validated (Table 27). The final subset is {1,2,4,5,8,3}.

| Method | | Epochs | T. Time (s) | Hidden Units | Indp. Param. | C. Error (%) | Standard Deviation |
|---|---|---|---|---|---|---|---|
| Vowel1 (Before selection) | | 19264.25 | 352.95 | 26.65 | 707.3 | 34.73 | 7.41 |
| Method 1 | | 11851.75 | 150.3 | 18.4 | 437.6 | 38.38 | 8.02 |
| Subset1 {1,2,3, 5,6,7,8} | Reduction | 38.48% | 57.51% | 30.96% | 38.05% | -10.51% | -- |
| Method 2 | | 15814.75 | 249.05 | 21.55 | 530 | 32.94 | 6.95 |
| Subset2 {1,2,3,4,5,6,7,8} | Reduction | 17.90% | 29.46% | 19.14% | 25.04% | 5.15% | -- |
| Method 1 | | 13128.75 | 206.35 | 21.15 | 457.7 | 32.79 | 5.83 |
| ,Subset3 {1,2,5,8,4,9} | Reduction | 31.85% | 41.54% | 20.64% | 35.22% | 5.59% | -- |
| Method 2 | | 13293 | 192.4 | 20.95 | 454.03 | 31.50 | 6.03 |
| Subset4 {1,2,4,5,8,3} | Reduction | 31.00% | 45.49% | 21.39% | 35.79% | 9.30% | -- |

Table 28. Validation results – Vowel

The results from these four methods are compared in Table 28. Table 28 shows that in Method 1, the training cost is significantly reduced by 38.48% for the number of training epochs, 57.51% for training time, 30.96% for the number of hidden units, and 38.05% for the number of independent parameters but the classification error is increased by 10.51%. In Method 2, the training cost is reduced by 17.90% for the number of training epochs, 29.46% for training time, 19.14% for the number of hidden units and 25.04% for the number of independent parameters and the classification error is reduced by 5.15%. In Method 3, the training cost is reduced by 31.85% for the number of training epochs, 41.54% for training time, 20.64% for the number of hidden units and 35.22% for the number of independent parameters and the classification error is reduced by 5.59%. In Method 4, , the training cost is reduced by 31.00% for the number of training epochs, 45.49% for training time, 21.39% for the number of hidden units and 35.79% for the number of independent parameters and the classification error is reduced by 9.30%.

**VI. Discussion and Conclusion**

Different from other wrapper methods, our feature selection methods detect relevant features directly, based on attribute contribution. The computational cost involved in searching among a large number of possible solutions is significantly reduced. The experimental results (Table 29) showed that all selected subsets using our feature selection methods could achieve increased accuracy.

The simulation results showed that in all the four problems, method 3 or method 4 can achieve the best performance in NN accuracy. Though method 1 or 2 is able to obtain the best

performance sometimes and is easier to implement compared with the other two methods, the disadvantage of method 1 or 2 is obvious - that is, the threshold. How to set a proper threshold is a problem in these two methods. Extra simulation is required to set an optimal threshold. However, in method 3 and method 4, there is no need to find a threshold. ITID NN provides a natural way to select the initial feature subset.

| Dataset | | Diabetes | Cancer | Glass | Vowel |
|---|---|---|---|---|---|
| Before Selection | Features | 8 | 9 | 9 | 10 |
| | Error (%) | 23.93 | 1.87 | 41.23 | 34.73 |
| | Standard deviation (%) | 1.15 | 0.34 | 4.43 | 7.41 |
| Method 1 | Features | 2 | 6 | 8 | 7 |
| | Error | 21.12 | 1.38 | 35.66 | 38.38 |
| | Standard deviation (%) | 1.21 | 0.28 | 4.27 | 8.02 |
| Method 2 | Features | 4 | 6 | 7 | 8 |
| | Error | 22.29 | 1.38 | 32.45 | 32.94 |
| | Standard deviation (%) | 1.09 | 0.28 | 3.40 | 6.95 |
| Method 3 | Features | 2 | 6 | 5 | 6 |
| | Error | 21.12 | 1.38 | 32.6 | 32.79 |
| | Standard deviation (%) | 1.21 | 0.28 | 4.23 | 5.83 |
| Method 4 | Features | 4 | 6 | 4 | 6 |
| | Error | 22.01 | 1.38 | 32.45 | 31.5 |
| | Standard deviation (%) | 1.13 | 0.28 | 3.86 | 6.03 |

Table 29. List of experimental results

We also have compared our results with the feature selection results reported in the literature such as ADHOC [15], NNFS [22] and Contribution-based ITID (or C-ITID) [19]. Our feature selection results are consistent with the results reported as shown in Table 30. It should be mentioned that the comparison of the error rates obtained by different methods in Table 30 may not be precise (or fair) because the results achieved using different algorithms were not obtained using the same experimental procedure, network structures, and training methods. For example, we used CBP to find suitable network structures for different problems, whereas the researchers

of NNFS used a standard fully connected three-layer neural network with 12 hidden units.

It can be seen from the simulation results that the average performance of method 3 and 4 is better than previous methods except in Glass1 (ADHOC). It should be mentioned that there exist some alternatives in our feature selection methods. For example, some alternative validation methods could also be tried. These will be considered in our future work.

| Dataset | NNFS | | ADHOC | | C-ITID Method 1 | | C-ITID Method 2 | |
|---|---|---|---|---|---|---|---|---|
| | Features | Error | Features | Error | Features | Error | Features | Error |
| Diabetes1 | 2.03(0.18) | 25.7(3.3) | 3 | 26.8 | 2 | 21.12 | 4 | 21.25 |
| Cancer1 | 2.7(1.0) | 5.9(1.0) | .. | .. | 5 | 1.75 | 5 | 1.75 |
| Glass1 | .. | .. | 4 | 29.5 | 5 | 36.23 | 4 | 33.4 |
| Vowel1 | .. | .. | .. | .. | .. | .. | .. | .. |

Table 30. Results of related works

**References:**


[1] John, G.H., Kohavi, R. and Pfleger, K., "Irrelevant features and the subset selection problem", *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 121- 129, 1994.

[2] Boser, B., Guyon, I. and Vapnik, V.N. 1992. "A training algorithm for optimal margin classifiers," *Fifth Annual Workshop on Computational Learning Theory*, pp. 144-152, San Mateo, CA: Morgan Kaufmann.

[3] Chua-Nan Hsu, Hung-Ju Huang, and Dietrich Schuschel, "The ANNIGMA-wrapper approach to fast feature selection for neural nets", *IEEE Transactions on Systems*, *Man, and Cybernetics-Part B: Cybernetics*, vol. 32, no. 2, April 2002.

[4] T. W. S. Chow and D. Huang, "Estimating optimal feature subsets using efficient estimation of high-dimensional mutual information," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 213–224, 2005.

[5] Ichino, M. and Sklansky, J., "Feature selection for linear classifier". *Proceedings of the Seventh International Conference on Pattern Recognition*, volume1, Page 124-127, July-Aug 1984.

[6] Ichino, M. and Sklansky, J., "Optimum feature selection by zero-one programming", *IEEE Transactions On Systems, Man and Cybernetics*, SMC-14 (5):737-746, September/October 1984.

[7] Jihoon Yang and Vasant Honavar "Feature subset selection using a genetic algorithm," *Feature Extraction, Construction, and Subset Selection: A Data Mining Perspective*, H.Motoda and H. Liu Eds. Norwell, MA: Kluwer, 1998, ch. 8.

[8] J.R. Quinlan, "C4.5: Programs for Machine Learning", San Mateo, CA: Morgan Kaufmann, 1993.

[9] K. Kira, and L.A. Rendell, "The feature selection problem: Traditional methods and a new algorithm" *Proceedings of Ninth National Conference on Artificial Intelligence*, pp. 129-134, 1992.

[10] L. Breiman, J. H. Friedman, R. A. Olshen, and C.J. Stone, "Classification and regression trees", *Belmont, CA: Wadsworth and Brooks*, 1984.

[11] Cortes, C. and Vapnik, V. "Support vector networks", *Machine Learning*, pp.273-297, vol. 20, 1995.

[12] L. Prechelt, "PROBEN1: A set of neural network benchmark problems and benchmarking rules," Technical Report 21/94, Department of Informatics, University of Karlsruhe, Germany, 1994.

[13] M. Dash and H. Liu, "Feature selection for classification", *Intelligent Data Analysis*, pp. 131-156, March 1997.

[14] M. Lehtokangas, "Modelling with constructive backpropagation", *Neural Networks*, vol. 12, pp. 707-716, 1999.

[15] M. Richeldi and P. Lanzi, "Performing effective feature selection by investigating the deep structure of the data," *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 379-383. AAAI Press, 1996.

[16] Haykin, S., *neural networks*, Prentice-Hall, 1999, New Jersey.

[17] Kohavi, R. and John, G.H. "Wrappers for feature subset selection", *Artificial Intelligence*, 97, 273–324, 1997.

[18] Quinlan, J., "Introduction of decision trees", *Machine Learning*, Morgan Kaufmann, San Mateo, California, 1993.

[19] Sheng-Uei Guan, Jun Liu and Yinan Qi, "An Incremental approach to contribution based feature selection ", *Journal of Intelligent Systems*, vol. 12, No. 4, 2002.

[20] K. Z. Mao, "Fast orthogonal forward selection algorithm for feature subset selection," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1218–1224, Sep. 2002.

[21] K. Z. Mao, "Orthogonal forward selection and backward elimination algorithms for feature subset selection," *IEEE Transactions on Systems*, *Man, and Cybernetics-Part B: Cybernetics*,, vol. 34, no. 1, pp. 629–634, Feb. 2004.

[23] Sheng-Uei Guan and Shanchun Li, "Incremental learning with respect to new incoming input attributes", *Neural Processing Letters,* vol. 14, issue 3, pp. 241-260, 2001.

[24] Riedmiller, M. and Braun, H. 1993. "A direct adaptive method for faster backpropagation learning: the RPROP algorithm", *Proceedings of the IEEE International Conference on Neural Networks*, 586–591.

[25] V. Sindhwani, S. Rakshit, D. Deodhare, D. Erdogmus, J. Principe, and P. Niyogi, "Feature selection in MLPs and SVMs based on maximum output information," *IEEE Transactions on Neural Networks*, vol. 15, no. 4, pp. 937–948, 2004.

[26] N. Kambhatla and T. K. Leen, "Dimension reduction by local principal component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1493–1516, 1997.

[27] J. T. Kwok and I. W. Tsang, "The pre-image problem in kernel methods," *IEEE Transactions on Neural Networks*, vol. 15, no. 6, pp. 1517–1525, 2004.

[28] A. D. Back and T. P. Trappenberg, "Selecting inputs for modeling using normalized higher order statistics and independent component analysis," *IEEE Transactions on Neural Networks*, vol. 12, no. 3, pp. 612–617, 2001.

## Appendix A

### Training stopping criteria used in this paper

The following is abridged from [17][23]:

The error measure $E$ used in the NN training is *the squared error percentage* [13], derived from the normalization of the mean squared error to reduce the dependency on the number of coefficients in the problem representation and on the range of output values used:

$$E = 100 \cdot \frac{o_{\max} - o_{\min}}{K \cdot P} \sum_{p=1}^{P} \sum_{k=1}^{K} \left(o_{pk} - t_{pk}\right)^2$$

where $o_{\max}$ and $o_{\min}$ are the maximum and minimum values of output coefficients in the problem representation.

$E_{tr}(t)$ is the average error per pattern of the network over the training set, measured after epoch $t$. The value $E_{va}(t)$ is the corresponding error on the validation set after epoch $t$ and is used by the stopping criterion. $E_{te}(t)$ is the corresponding error on the test set; it is not known to the training algorithm but characterizes the quality of the network resulting from training.

The value $E_{opt}(t)$ is defined to be the lowest validation set error obtained in epochs up to epoch $t$:

$$E_{opt}(t) = \min_{t' \le t} E_{va}(t')$$

The *generalization loss* at epoch $t$ is defined as the relative increase of the validation error over the minimum so far (in percentage):

$$GL(t) = 100 \cdot \left( \frac{E_{va}(t)}{E_{opt}(t)} - 1 \right)$$

A high generalization loss is one candidate reason to stop training because it directly indicates overfitting.

To formalize the notion of training progress, a *training strip of length* $k$ is defined to be a sequence of $k$ epochs numbered $n+1 \ldots n+k$ where $n$ is divisible by $k$. The training *progress* measured after a training strip is:

$$P_k(t) = 1000 \cdot \left( \frac{\sum_{t' \in t-k+1 \ldots t} E_{tr}(t')}{k \cdot \min_{t' \in t-k+1 \ldots t} E_{tr}(t')} - 1 \right)$$

It is used to measure how much larger the average training error is than the minimum training error during the training strip.

During the process of growing and training sub-networks, heuristic overall stopping criteria are adopted as follows: $E_{opt} < E_{th}$ **OR** *(Reduction of training set error due to the last new hidden unit is less than 0.01%* **AND** *Validation set error increased due to the last new hidden unit)*. The first part ( $E_{opt} < E_{th}$ ) means that the optimal validation set error is below the threshold and the result is acceptable. The other part means the last insertion of a hidden unit resulted in hardly any progress. The criteria for adding a new hidden unit are as follows: *At least 25 epochs reached for the current network* **AND** (*Generalization loss* $GL(t) > 5$ ***OR*** *Training progress* $P_k(t) < 0.1$). The first part means that the current network should be trained for at least a certain number of epochs before a new hidden unit is installed because the error curves will be turbulent in the beginning. The second part means that the current network has been overfitted or training has little progress. In addition, the RPROP algorithm [24] is adopted to minimize the cost function. The parameters are set as: $\eta^+ = 1.2$, $\eta^- = 0.5$, $\Delta_0 = 0.1$, $\Delta_{max} = 50$, $\Delta_{min} = 1.0e-6$, with initial weights from –0.25 … 0.25.