Hierarchical Feature Selection for Random Projection

Qi Wang, *Senior Member, IEEE,* Jia Wan, Feiping Nie, Bo Liu, Xuelong Li, *Fellow, IEEE*

Abstract—Rdndom projection is a popular machine learning algorithm which can be trained with a very efficient manner. However, the number of features should be large enough when applied to a rather large scale dataset, which results in slow speed in testing procedure. Furthermore, some of the features are redundant and even noisy since they are randomly generated, so the performance may be affected by these features. To remedy these problems, an effective feature selection method is proposed to select useful features hierarchically. The training time and accuracy of the proposed method are improved compared with traditional methods and some variations on both classification and regression tasks. Extensive experiments confirm the effectiveness of the proposed method.

1 Introduction

NEURAL networks have been proved to be effective in solving tasks like object detection [1] and scene classification [2]. Gradient decent (GD) algorithm [3] and its variants are the most widely adopted methods used in the learning of neural networks. However, GD algorithms need to tune parameters iteratively, which makes the training of neural networks very slow especially on some large scale datasets.

ELM [4] is a single layer network which can be efficiently trained with a closed-form solution. It has been utilized in many applications and confirmed to be efficient in training and has good performance based on some small datasets. However, more hidden neurons are necessary when ELM is applied on rather large datasets since the weights of neurons are randomly generated. Under this circumstance, the training of ELM is rather slow as a large matrix reversion problem should be solved. Furthermore, the selection of neurons can be seen as a random projection, in which redundant and noisy features might affect the final performance and slow down the testing speed.

In order to select useful features in ELM, some algorithms are proposed to prune redundant features. Pruned-ELM [5] utilizes statistical methods as measurements of the relevance between hidden nodes and class labels, in which the relevant nodes are removed to achieve compact networks and robust performance. This method is faster than the original ELM in training, but it can only be applied on classification problems. OP-ELM [6] is proposed to rank neurons by multiresponse sparse regression (MRSR) algorithm, and then select useful features through leave-

- *Qi Wang is with the School of Computer Science, with the Center for OPTical IMagery Analysis and Learning, and with the Unmanned System Research Institute, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: crabwq@gmail.com).*
- *Jia Wan and Feiping Nie are with the School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, China (e-mail: jiawan1998@gmail.com, feipingnie@gmail.com).*
- *Bo Liu is with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL36849, U.S.A. (e-mail: boliu@auburn.edu).*
- *Xuelong Li is with the Center for OPTical IMagery Analysis and Learning, State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China (email: xuelongli@opt.ac.cn).*

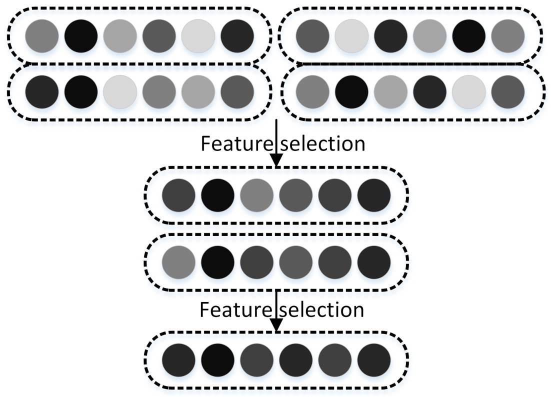


Fig. 1. The pipeline of the proposed hierarchical feature selection method. In this figure, each node represents a feature. The feature pool which contains 24 features is split into 4 groups. For each group, half of the features are selected each time. Then, a new feature pool which contains 12 features is generated. Finally, a compact network contains only 6 features are constructed. Note that, darker nodes indicate the features have higher weights.

one-out (LOO) validation. It can be used in both classification and regression tasks, but the training speed is slower than the original ELM. To accelerate the training of ELM for both classification and regression tasks, a hierarchical feature selection method (abbreviated as HFS-ELM) is proposed in which a very simple yet efficient criteria is utilized to determine the usefulness of features. To further accelerate the training procedure, a hierarchical selection scheme is also presented. Through our experiments, this method can achieve superior performance than ELM and some widely used classification methods with faster testing speed. And it can be used in both classification and regression tasks.

The contributions of this paper are summarized as follows:

1. A simple yet efficient feature selection method is proposed to select useful features generated by ELM. The criteria used for ranking the usefulness of features, which is based on the value of output weights is very easy to calculate and effective to eliminate redundant and noisy features.
2. A hierarchical selection scheme is proposed to choose useful features in a very efficient manner. To accelerate the

training of ELM on large scale datasets, useful features are selected hierarchically, which avoids the selection from the whole feature pool. Therefore,the large matrix reversion is evaded which largely accelerates the training of ELM.

3. Extensive experiments are performed on a variety of datasets including both classification and regression tasks and confirm the effectiveness and efficiency of the proposed method.

2 Related Works

In this section, we briefly review the compression algorithms based on ELM which can be divided in 2 categories.

The first type of algorithms perform network compression before or during the learning of output weights (i.e. matrix reversion calculation) which is faster in training than the original ELM since the matrix reversion is performed on a smaller matrix. EI-ELM [7] is proposed to incrementally add features from a candidate pool during the training process in which the feature that results in larger error decreasing will be appended to the existing network. In Bidirectional ELM [8], the relationship between a newly added feature and residual error is presented so that the weight of new feature can be calculated efficiently. A pruned-ELM (P-ELM) [5] is proposed to evaluate the relevance between features and class labels, in which the irrelevant features are pruned to generate a compact network. Another pruned method [9] is proposed which utilizes the defined sensitivities to determine the necessary features. To remedy overfitting problem and generate compact network, Sparse Bayesian ELM (SBELM) [10] is proposed to learn sparse weights by Bayesian Inference, in which the features with 0 weight can be pruned.

Another type of methods are proposed to select useful features based on the results of the original ELM, which means that the training is longer than the original ELM. OP-ELM [6] which utilizes MRSR and LOO validation to determine the usefulness of features can generate very compact network than the original ELM with comparable performance. Parsimonious ELM [11] is proposed to rank and select significant features based on partial orthogonal decomposition through recursive orthogonal least squares method.

3 Extreme Learning Machine

ELM is a very fast algorithm since the learning procedure don't need iterative optimization of parameters. The learning of ELM is rather simple. First of all, the weights and bias of input weights of the hidden layer is randomly assigned. Then, the output weight of ELM is calculated to minimize the mean squared error between the ground truth and the prediction.

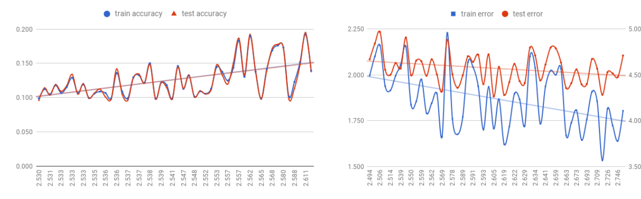
Formally, in a single hidden layer neural network with N nodes, the output label y with respect to a d dimensional input vector x is denoted as:

$$y = \sum_{i=1}^{N} \beta_i h_i(x), \tag{1}$$

where β_i is the weight of the i-th node, and $h(x)$ indicates the non-linear feature mapping which can be written as:

$$h_i(x) = \theta(a_i x + b_i), \tag{2}$$

where a_i, b_i are randomly assigned parameters and θ can be any non-linear mapping function like Sigmoid function, Gaussian function or Cosine function.



(a) Classification (b) Regression

Fig. 2. The relationship between output weights and the performance of ELM. In this figure, the horizontal axis is the values of the output weight in ascending order. The vertical axis is the accuracy or mean squared error. (a) The classification accuracy is proportional to the output weight. (b) The regression error is inversely proportional to the output weight.

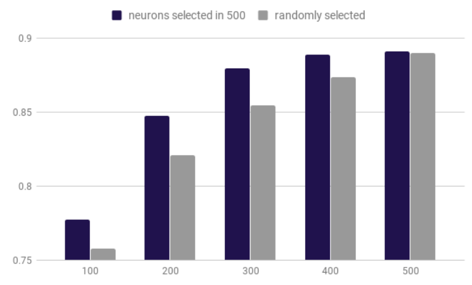


Fig. 3. The comparison of ELM and ELM with feature selection on classification task. The horizontal axis is the number of features randomly generated or selected from a feature pool with 500 features. The vertical axis is classification accuracy.

In this model, the only unknown parameter is β which can be learned by minimizing the mean squared error between the ground truth and the prediction:

$$\beta = \min_{\beta} \|H\beta - G\|^2, \tag{3}$$

where $\|\cdot\|$ denotes the Frobenius norm and H is the output of hidden layer with randomly assigned parameters:

$$H = \begin{bmatrix} h(x_1) \\ h(x_2) \\ \vdots \\ h(x_d) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & h_2(x_1) & \dots & h_N(x_1) \\ h_1(x_2) & h_2(x_2) & \dots & h_N(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ h_1(x_d) & h_2(x_d) & \dots & h_N(x_d) \end{bmatrix}, \tag{4}$$

and G is the ground truth matrix. If ELM is performed on a regression problem where the output target is a real value, G could be a vector:

$$G = [g_1, g_2, \cdots, g_n]^\top, \tag{5}$$

otherwise G is a matrix representing category labels in classification problems.

The optimization of Equation 3 is very efficient since it has a closed-form solution:

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

Since it has no iteration steps, the optimization of ELM is very efficient especially when performed on a small dataset in which the number and dimension of training samples are small. Under this condition, a small number of hidden nodes are necessary to obtain good performance so the matrix reversion can be easily calculated. However, large amount of hidden nodes is usually needed to guarantee performance on large scale datasets, because the matrix reversion is performed on a large matrix, the calculation of which is often time-consuming.

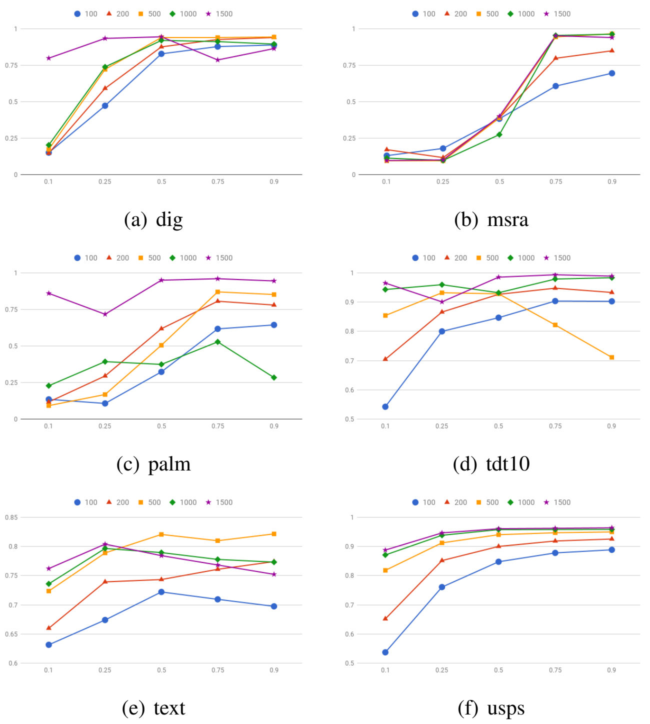


Fig. 4. The experimental results regarding to how many features should be remained in each selection procedure on different datasets. In this figure, horizontal axis is the ratio of remained features in selection procedure. The vertical axis is accuracy. Different lines represent performance of HFS-ELM with different number of final remained features.

4 Hierarchical Feature Selection

In order to alleviate redundant features and avoid large matrix reversion, in this paper, a very simple yet effective feature selection method is proposed to hierarchically choose significant features in large amount of randomly generated features. In particular, the first step of the proposed method is to generate features randomly. Then, the features are split into several groups. For each group of the features, the original ELM is performed to learn output weights which can be utilized to evaluate the importance of features. Specifically, larger weight indicates that the feature is more important, thus the features can be ranked based on the weights of them. After the ranking and selection of features, all of them chosen from different groups are aggregated together. The selection procedure is performed several times hierarchically, and a very compact yet effective network can be generated from numerous features very efficiently.

Since the input weight of hidden neurons is randomly assigned, it can be seen as random projection in which many redundant features might be yielded during the training process when a good performance is achieved. To relieve this problem, a simple criteria for feature selection is proposed to rank and choose features based on the output weight which can be utilized to evaluate the importance of features. Particularly, after the learning of the original ELM, the output weight β is obtained by calculating Equation 6. As shown in experiments, the features with larger absolute value have higher probability to generate better performance. So, the output weights are sorted in descending order:

$$I = \text{sort\_index}(\text{norm}(\beta)), \tag{7}$$

where norm() is a function that calculate the norm of $\beta_1, \beta_2, \cdots \beta_n$ independently. sort_index() is used to rank the norm of weight in descending order and obtain the index I. After the importance of features is ranked, a subset of feature can be selected through index I. Proved by experiments, this method is very effective to prune redundant and noisy features and generate better performance than the original ELM. However, the training time is increased especially when the feature pool is large as the calculation of output weight needs to solve a larger matrix inversion problem than the original ELM.

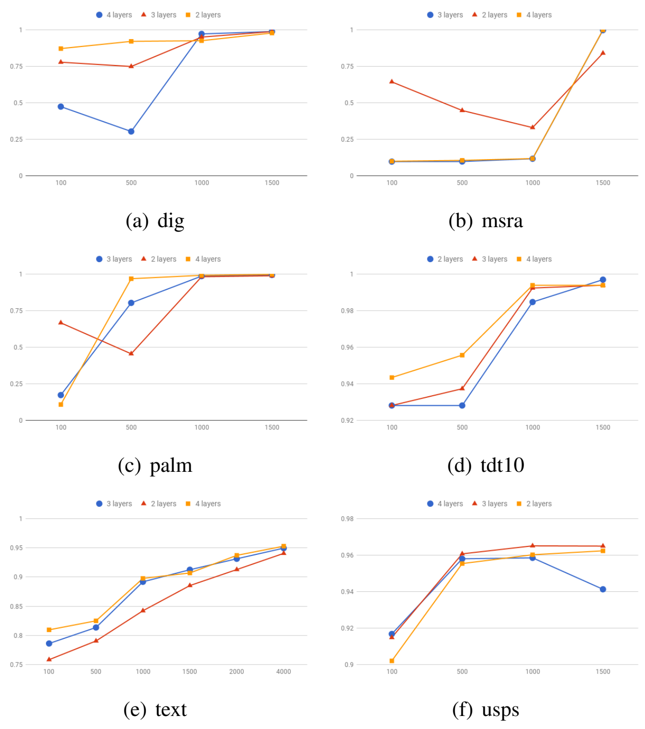
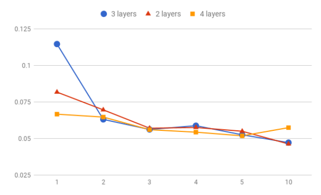


Fig. 5. The experimental results about the number of layers in the selection scheme and the size of features remained at last on different datasets. In the figures, the horizontal axis is number of final remained features and the vertical axis is accuracy. Different lines represent the performance of HFS-ELM with different selection layers.

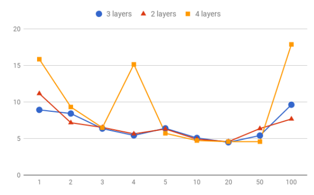
To accelerate the training of ELM by avoiding large matrix inversion, a hierarchical selection strategy is proposed. First of all, numerous features are randomly generated to serve as the feature pool. Then, features in the pool are split into several groups on which the feature selection procedure is performed. To obtain more compact network, the same feature selection can be performed on the selected features which forms a hierarchical scheme.

Although there are some parameters which will affect the performance and efficiency of the proposed method, they are not very sensitive to different datasets. In experiments, some parameters can be used to both classification and regression tasks over different datasets. The parameters are summarized as follows and the selection of them are investigated in experiments.

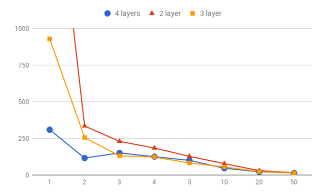
1. The first parameter that we should determine is the ratio of remained features in selection procedure since large ratio tends to yield better performance but the training speed are limited while small ratio based algorithm can be trained more efficiently. A proper ratio can yield good performance and fast training speed at the same time.



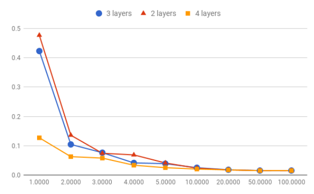
(a) NWPU_Cong



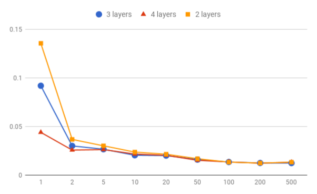
(b) abalone



(c) cpusmall



(d) mg



(e) space_ga

Fig. 6. The experimental results about how many layers the selection procedure should include on different datasets. In the figures, the horizontal axis is the number of final remained features and the vertical axis is accuracy. Different lines represent the performance of hierarchical feature selection with HFS-ELM with different selection layers.

2. The final size of selected features is a factor that affect performance and the testing speed of the algorithm. A large number of final features will yield better performance while a small number of features will be faster in training and testing.
3. Based on the ratio in feature selection procedure and the final remained number of features, the size of feature pool can be determined by the number of layers in hierarchical selection scheme. The more the number of layers, the larger the feature pool. A large feature pool can produce better performance at the most of the time. However, the training speed will increase a lot if hierarchical selection procedure with more layers are performed during training.

The proposed hierarchical feature selection method is summarized in Algorithm 1.

TABLE 1
The detailed information of evaluated datasets for classification.

	size of dataset	dimension
dig	1797	64
msra	1799	25
palm	2000	256
tdt10	653	36771
text	1946	7511
usps	9298	256

5 Experimental Results and Discussion

To confirm the efficiency of the proposed method, extensive experiments are conducted. Through the experimental results, we find that the efficiency of ELM can be largely improved with comparable performance, so that the proposed method can be used for rather large datasets. In this section, some experiments are first conducted to prove that the feature with larger output weight has higher probability to generate better performance. Then, the performance and efficiency of the proposed method is evaluated on both classification and regression tasks.

Algorithm 1 Hierarchical Feature Selection for Extreme Learning Machine

```
Require: $X \in \Re^{n\times d}$: the training features
         $Y \in \Re^{n\times 1}$: the training labels
         $S$: the size of feature pool
         $s$: the size of each group
         $r$: the ratio of selected features from each group
  // for each node
  for $i = 1, 2, ..., S$ do
      Randomly assign parameters $a^{(i)}$, $b^{(i)}$
  end for
  //Suppose the hierarchical scheme has three layers
  for $i = 1, 2, 3$ do
      // Split features in pool into groups with size $s$
      for each group do
          Compute matrix $H$ through Equation 2
          Calculate $\beta_j$ through Equation 6
          Rank features' importance through Equation 7
          Select features based on $r$ and the ranking result
      end for
      Merge all selected features into a new feature pool
  end for
  Compute $\beta_f$ of the final selected features by Equation 6
Ensure: Final selected features and the corresponding $\beta_f$
```

5.1 Why the proposed method is working?

Empirically, features which have larger weights tend to generate better performance since large weights indicate the significance of features. To prove this idea, some experiments are conducted, including classification and regression tasks.

In particular, we randomly generate some ELM with only one neuron and calculate the corresponding output weight and accuracy/MSE (mean squared error). Then, these trained ELMs are ranked by the value of their output weights. The relationship of the output weight and performance is thus shown in Figure 2. In this figure, the classification accuracy is proportional to the output weight and the regression error is inversely proportional to the weight which indicates that the larger weight has higher probability to yield better performance in both classification and regression tasks.

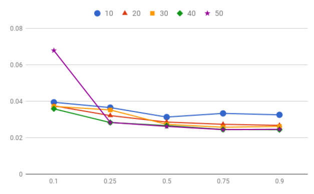
To further confirm the effectiveness of feature selection, the performance of ELM is compared with the performance of ELM with feature selection. In details, the original ELM is compared with the ELM with same number of neurons which have rather higher output weights selected from a feature pool with 500 neurons. The experimental result is shown in Figure 3 in which the classification accuracy of ELMs with neurons selected from feature pool is higher than the ELMs with randomly generated neurons. The experimental result indicates that the ELMs with selected neurons always have better performance.

TABLE 2
Comparison of different methods on classification tasks. The best results are indicated in **bold**.

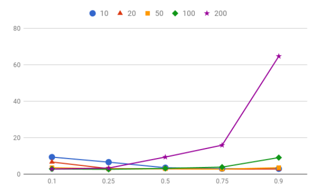
	SVM		Adaboost		ELM		HFS-ELM	
	accuracy	testing time	accuracy	testing time	accuracy	testing time	accuracy	testing time
dig	**97.96%**	0.039	95.34%	0.088	96.98%	0.053	97.20%	**0.028**
msra	99.85%	0.093	99.27%	0.142	99.61%	0.130	**99.94%**	**0.057**
palm	99.60%	**0.020**	96.50%	0.060	99.00%	0.049	**99.80%**	**0.02**
tdt	97.57%	1.283	97.71%	1.193	98.21%	0.033	**99.17%**	**0.030**
text	**96.92%**	5.464	94.60%	0.764	93.93%	2.496	95.29%	**1.676**
usps	94.95%	0.997	91.84%	0.358	**96.74%**	0.334	96.51%	**0.172**

TABLE 3
The detailed information of datasets fro regression.

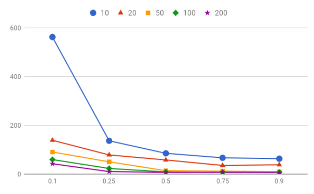
	size of dataset	dimension
NWPU_Cong	5585	10
abalone	4177	8
cpusmall	8192	12
mg	1385	6
space_ga	3107	6



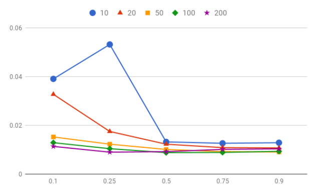
(a) NWPU_Cong



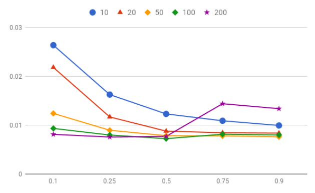
(b) abalone



(c) cpusmall



(d) mg



(e) space_ga

Fig. 7. The experimental results about how many features should be remained on different datasets. In this figure, the horizontal axis is the ratio of remained features in selection procedure. The vertical axis is accuracy. Different lines represent performance of HFS-ELM with different number of final remained features.

5.2 HFS-ELM for Classification

Classification is one of the fundamental problems in Artificial Intelligence. To assess the efficiency of the proposed method, 3 popular classifiers (ELM, SVM, and Adaboost) are included for comparison on several widely used benchmarks (dig [12], msra [13], palm [14], tdt10 [15], text [16] and usps [17]) that the detailed information is concluded in Table 1.

We first investigate three hyper-parameters that should be considered: how many features should be remained when we select useful features, how many features should be remained at last and the number of layers in hierarchical scheme. The first parameter is the number of features selected each time. Based on the aforementioned declaration that it is the trade-off between the performance and efficiency, the performance of ELM with different compression level is compared, to find out a proper number that can maximumly compress the network without much performance decrease. The experimental results are shown in Figure 7 in which we can see that the performance can be guaranteed with a rather high compression degree when ratio equals to 0.5 over most of the datasets. Thus, ratio is set as 0.5 in the following experiments.

The number of final remained features and the number of layers are explored at the same time. The ELMs with different layers and remained features are trained and compared in the experiment. The results are shown in Figure 5. In this figure, we can see that a 2 layers selection scheme can not achieve the best or comparable results than a 3 layers or 4 layers selection scheme. And a 4 layers selection method may generate unstable performance on some datasets like dig or may cause overfitting on usps. Thus, the number of layers of hierarchical scheme is set as 3 in the following experiments.

Table 2 shows the experimental results regarding to the comparison of some widely used classification algorithms. The proposed method is the most efficient one of all compared algorithms with comparable classification performance since the hierarchical feature selection can prune redundant and noisy features effectively and generate more compact networks. Comparing the proposed algorithm with the original ELM, better performance can be obtained with more compact networks and faster testing speed, which proves that the effectiveness of the proposed hierarchical selection method.

5.3 HFS-ELM for Regression

In this section, the performance and efficiency of the proposed method is evaluated on regression tasks. Specifically, the proposed algorithm (HFS-ELM) is compared with Adaboost, OPELM and ELM. The detailed information of datasets (abalone [18] cpusmall[1], mg [19], space_ga [20]) can be seen in Table 3. Most of these datasets are come from UCI repository and the NWPU_Cong dataset [21] is a real world congestion detection dataset. Note that, the mean squared error (MSE) is utilized to evaluate the performance of regression.

Similar to the experiments in classification tasks, the parameters are first exploited. We find that the ratio can be set as 0.5 over both classification and regression tasks, as shown in Figure

1. http://www.cs.toronto.edu/ delve/data/datasets.html

TABLE 4
Comparison of different methods on regression tasks. The best results are indicated in **bold**.

	Adaboost		OPELM		ELM		HFS-ELM	
	MSE	testing time	MSE	testing time	MSE	testing time	MSE	testing time
NWPU_Cong	**0.021**	0.038	0.026	0.019	0.029	0.026	0.029	**0.014**
abalone	7.864	0.020	5.610	0.010	5.250	0.012	**4.482**	**0.006**
cpusmall	13.75	0.041	17.27	0.02	11.99	0.02	**11.52**	**0.02**
mg	0.024	0.020	**0.015**	0.0064	0.0153	0.0060	**0.015**	**0.004**
space_ga	0.019	0.024	0.013	0.005	**0.012**	0.005	**0.012**	**0.004**

7, a smaller ratio may cause worse performance on most of the dataset while a larger ration may cause overfitting on abalone and space datasets. The number of layers can be set as 2 or 3 based on the results shown in Figure 6 that only 4 layers scheme may cause overfitting on abalone dataset. Since the size of regression datasets is rather small, a 4 layers selection scheme is far enough to fit training data. However, overfitting is occurred on only one dataset which confirms that the proposed method is very robust to avoid overfitting.

The experimental results are summarized in Table 4. In most cases, the proposed method achieves superior performance than others. What's more, the testing time is always shorter than others which confirmed the proposed selection criteria is very effective to compress network by avoid redundant and noisy features.

6 Conclusion and Future Works

To improve the speed of the original ELM through pruning redundant and noisy features in single layer neural networks, a hierarchical feature selection method is proposed. The selection method which can be used to select useful features from feature pool is based on a simple but effective criteria regarding to the value of the learned weight. Based on the selection criteria, a hierarchical selection scheme is proposed to accelerate the efficiency of the algorithm. Utilizing hierarchical feature selection in ELM, very compact networks can be generated which have comparable performance with some widely used algorithms in classification tasks and better results than traditional methods in regression tasks. And the testing speed has shown its priority in both classification and regression tasks.

Although both performance and speed are improved based on proposed method, it can be further improved by automatically estimating hyper-parameters like the layer of the hierarchical scheme. In the future, adaptive parameter selection methods should be exploited.

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[2] Y. Yuan, J. Wan, and Q. Wang, "Congested scene classification via efficient unsupervised feature learning and density estimation," *Pattern Recognition*, vol. 56, pp. 159–169, 2016.

[3] S.-Y. Zhao and W.-J. Li, "Fast asynchronous parallel stochastic gradient descent: A lock-free approach with convergence guarantee." in *AAAI Conference on Artificial Intelligence*, 2016, pp. 2379–2385.

[4] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.

[5] H.-J. Rong, Y.-S. Ong, A.-H. Tan, and Z. Zhu, "A fast pruned-extreme learning machine for classification problem," *Neurocomputing*, vol. 72, no. 1, pp. 359 – 366, 2008.

[6] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, "Op-elm: Optimally pruned extreme learning machine," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 158–162, 2010.

[7] G.-B. Huang and L. Chen, "Enhanced random search based incremental extreme learning machine," *Neurocomputing*, vol. 71, no. 16, pp. 3460–3468, 2008.

[8] Y. Yang, Y. Wang, and X. Yuan, "Bidirectional extreme learning machine for regression problem and its learning effectiveness," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 9, pp. 1498–1505, 2012.

[9] L. Ying and L. Fan-Jun, "A pruning algorithm for extreme learning machine," in *International Conference on Intelligent Data Engineering and Automated Learning*, 2013, pp. 1–7.

[10] J. Luo, C.-M. Vong, and P.-K. Wong, "Sparse bayesian extreme learning machine for multi-classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 4, pp. 836–843, 2014.

[11] N. Wang, M. J. Er, and M. Han, "Parsimonious extreme learning machine using recursive orthogonal least squares," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 10, pp. 1828–1841, 2014.

[12] L. Xu, A. Krzyzak, and C. Y. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[13] S. A. Nene, S. K. Nayar, H. Murase *et al.*, "Columbia object image library (coil-20)," 1996.

[14] C. Hou, F. Nie, C. Zhang, and Y. Wu, "Learning an orthogonal and smooth subspace for image classification," *IEEE Signal Processing Letters*, vol. 16, no. 4, pp. 303–306, 2009.

[15] F. Wang, C. Tan, P. Li, and A. C. König, "Efficient document clustering via online nonnegative matrix factorizations," in *Proceedings of the International Conference on Data Mining*, 2011, pp. 908–919.

[16] O. Chapelle and A. Zien, "Semi-supervised classification by low desity separation." in *International Conference on Artificial Intelligence and Statistics*, 2005, pp. 57–64.

[17] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1.

[18] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.

[19] G. W. Flake and S. Lawrence, "Efficient svm regression training with smo," *Machine Learning*, vol. 46, no. 1, pp. 271–290, 2002.

[20] R. Kelley and R. Barry, "Quick computation of regressions with a spatially autoregressive dependent variable," *Geographical Anal*, vol. 29, pp. 232–247, 1997.

[21] J. Wan, Y. Yuan, and Q. Wang, "Traffic congestion analysis: A new perspective," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 1398–1402.