Ensemble of online sequential extreme learning machine based on cross-validation

Liyun Wei¹, Lidan Wang¹,³,⁴,⁵, Yunfei Li¹ and Shukai Duan¹,²,³,⁴,⁵,*

¹College of Electronic and Information Engineering, Southwest University, Chongqing, 400715, China
²College of Artificial Intelligence, Southwest University, Chongqing, 400715, China
³Brain-inspired Computing & Intelligent Control of Chongqing Key Lab, Chongqing, 400715, China
⁴National & Local Joint Engineering Laboratory of Intelligent Transmission and Control Technology, Chongqing, 400715, China
⁵Chongqing Brain Science Collaborative Innovation Center Chongqing, 400715, China

*Corresponding author’s e-mail: ldwang@swu.edu.cn

Abstract. The online sequential extreme learning machine (OS-ELM) was proposed as an evolution of the Extreme learning machine (ELM) in 2006. The learning algorithm can learn data by fixing or changing the size of the chunk (a block of data). Compared with other online algorithms, it has faster training speed, and the generalization performance is better. The proposed ensemble of neural networks further demonstrates that the use of the neural network sets with multiple consensus schemes has a greater improvement over the stability of a single network. The basic idea of cross-validation is to group the original data sets, part of which is to train as the training set and the other part of which is to verify as the test set, so the algorithm can reduce the over-fitting problem of the data to a certain extent, and extract more valid information in the limited data set. In this paper, a new ensemble of online sequential extreme learning machine based on cross-validation (ENOS-ELM) is proposed. Through application of cross-validation and integration to the training stage, the accuracy of the algorithm in classification is further improved.

1. Introduction
In many real-world applications, learning is actually an ongoing process, because what we usually get is not a complete data set, and new data often arrives during the learning process. When new data arrives, batch learning takes a lot of time by combining previously trained data with untrained new data for retraining. Therefore, the sequential learning algorithm can become a cure for learning new information in the network, and only learn new data without training, they achieving the purpose of saving time. Liang et al. have proposed an online sequential extreme learning machine (OS-ELM), [1] which is a not only fast but accurate online sequential learning algorithm for single hidden layer feedforward network (SLFN) with additive and radial basis function (RBF) hidden nodes. OS-ELM is developed on the basis of ELM [2-4]. ELM algorithm can be used as batch learning, and has been proved that is very fast and
has good generalization performance [3]. Compared OS-ELM with ELM, it can learn data by chunk-by-chunk with fixed or variable block size individually.

In order to improve the performance of OS-ELM and use the integrated network [5-7] for sequential learning mode, LAN et al. proposed an ensemble network structure called Ensemble of Online Sequence Extreme Learning Machine (EOS-ELM) [8]. EOS-ELM includes several OS-ELM networks. The final outcome of the overall network performance is obtained by summing the outputs of each OS-ELM and taking the average. However, due to the instability of the algorithm, the output of an OS-ELM may be quite different from most of the other output results, it has a great impact on the overall output.

In this article, we present an ensemble of online sequential extreme learning machine based on cross-validation (ENOS-ELM). First, the original training set is divided into several subsets using a cross-validation scheme. In the cross-validation, each subset has opportunity to play the role of training set and test set. Then in the new training process, each learner uses one of the subsets as a test set and trains with each of the remaining subsets. This process is repeated until all subsets have been used as test sets, and the accuracy of each verification is averaged and used as a comparison object for subsequent training, and the first phase ends. After that, it enters the selection process. This part includes several small networks. Each time the training set passes through a small network, its accuracy will be calculated and compared with the average accuracy obtained in the previous stage. Finally, enter the integration phase, and the remaining parameters of each group are tested and the average of the results is taken.

In this article, we will review the concepts of OS-ELM and EOS-ELM in the second chapter. In the third chapter, we will detail the integrated cross-validation online over-limit learning machine. In the fourth chapter, we will show the experimental results of the algorithm (the activation functions are additive function and radial basis function respectively), and compare them with the experimental results of OS-ELM and EOS-ELM without cross-validation. The fifth chapter is the conclusion.

2. The Reviews of OS-ELM and EOS-ELM

2.1. The Review of OS-ELM

OS-ELM is based on the development of ELM which based on the single hidden layer feedforward neural network. And the hidden nodes are generally additive or radial basis functions. If there is a SLFN with L hiding nodes and can approximate N samples almost no mistake, it can be express as the parameters $a_i$, $\beta_i$ and $b_i$ are existence and satisfy mathematical relations:

$$f_L(x_i) = \sum_{i=1}^{L} \beta_i G(a_i, b_i, x_i) = t_j, \quad j = 1, \ldots, N. \quad (1)$$

Where $a_i$ and $b_i$ are the learning parameters of the hiding nodes, and $\beta_i$ is the output weight which links output layer and hidden layer. In addition, $G(a_i, b_i, x_i)$ represents the output of the corresponding $i$th hidden node. When the activation function is $\text{sig}$, $G(a_i, b_i, x_i) = g(a_i \cdot x_j + b_i)$, where $a_i$ is the input weight, and $b_i$ is the bias of the corresponding $i$th hidden node. But if the activation function is RBF, $G(a_i, b_i, x_j) = g(b_i \|x_j - a_i\|)$, where $a_i$ is the center and $b_i$ is the impact width of the $i$th hidden node, remember that $b_i$ is greater than 0 in the case.

Assuming that the network has L hidden nodes, the OS-ELM is divided into two phases, one is the initialization phase, and the other is the sequential learning phase. In the first phase, in order to ensure that OS-ELM achieve the same learning performance as ELM, the condition that $\text{rank}(H_0) = L$ is necessary. And $H_0$ represents the hidden output matrix of the initialization phase. This means that the number of training data required in initialization phase $N_0$ must be equal to or greater than L, namely $N_0 \ge L$. But if $N_0 = N$, OS-ELM is the same as batch ELM. Therefore, ELM can be regarded as a special case of OS-ELM when all data exists in an iteration.

2.2. The Review of EOS-ELM

The EOS-ELM network is made up by many OS-ELM networks, all of which have the same number of hidden nodes and function. Article [8] builds the original EOS-ELM with M OS-ELM networks. All M OS-ELM networks are trained in the original OS-ELM training format. That is, each OS-ELM network
first randomly generates learning parameters, obtains output weights by least squares method, and applies the parameters of the training phase to the testing phase to obtain actual output. The average of all the outputs of the M OS-ELM networks is then calculated, which is the final output of the EOS-ELM.

EOS-ELM should be regarded as a whole system. We hope that EOS-ELM is better than OS-ELM alone. Since the input weight and bias of each OS-ELM network are randomly generated, it can be said that this is the whole. Every OS-ELM network is unique. When the data sequentially enters the overall network, some of the OS-ELM networks can adapt to new data faster and better than others. However, different OS-ELM networks may be well-adapted networks for different input data, but it is not possible to avoid some OS-ELM networks that can not adapt well to new data. In this case, the final result of this single OS-ELM network is the worst in the simulated round compared to the other rounds. As a result, the overall test results are poor. Statistically, the population mean is always closer to the expected value than the value of the individual, which means that the results obtained by the EOS-ELM always fluctuate within a small range compared to the results obtained with a single OS-ELM network. What is obvious that the larger M is, the closer the actual output is to the desired output. Therefore, we can conclude that EOS-ELM can be more stable when M is larger [6-7]. However, as M increases, the computation time also increases and the network becomes more complicated. Therefore, for the selection of the parameter M, we must not only consider the overall stability, but also fully consider the calculation time and complexity of the network.

3. ENOS-ELM
Since the weights and bias of hidden nodes are randomly generated, ELM significantly shortens the learning time. However, because parameters cannot contain prior knowledge of the input, the parameters we use maybe not contain optimal parameters, and generalization performance may be degraded. The [8] proposed EOS-ELM algorithm improves the stability and generalization ability of the network to a certain extent through multiple iterations, but the improvement is not obvious, and even increases the probability of over-fitting. This paper suggests using a variety of random parameter sets to construct a set of predictors on the training set, where the parameters of each predictor are chosen according to specific criteria. The so-called specific standard refers to the accuracy as a reference [10]. The proposed ENOS-ELM is divided into three phases, an initialization phase, a select parameters phase, and a test phase. Cross-validation is used in the first two phases. On one hand, the cross-validation scheme prohibits over-fitting; on the other hand, it expands the number of predictors in the whole to ensure the stability and accuracy of the decision.

3.1. Initialization
In this phase, the whole training set is divided into K subsets, each subset contains roughly the same number of samples, and learning parameters are set randomly. Then each learner is trained with (k-1) subset and verified with the remaining subset. After K times of training, the average accuracy CA of K times of cross validation and the average value of output weight norm \( \| \beta \| \) will be obtained. It is generally believed that smaller \( \| \beta \| \) or larger CA can represent better generalization performance. Here we use CA as a criterion for later comparison to determine whether \( \omega \) and \( b \) can proceed to the next step or not.

3.2. Selection of parameters
In this stage, we can use OS-ELM to perform M iterations, each iteration will randomly generate learning parameters \( \omega_m \) and \( b_m \) (m represents the current iteration number). At the same time, each iteration will use cross-validation method to calculate the training accuracy of this iteration, and compare it with the training accuracy of initialization stage, if it is initialization training accuracy higher, then the learning parameters of the group will be replaced by \( \omega \) and \( b \); on the contrary, if the training accuracy of this iteration is higher, the learning parameters of the group will be applied to the next training process. In other words, after M iterations, we will carry out M times cross validation, compare with the CA
obtained in the initialization stage, take the parameters corresponding to the higher average accuracy CA, and generate N sets of input parameters.

3.3. Integrated Construction

As can be seen from the above, in this process, the variables \( \omega_m \) and \( b_m \) always maintain the relatively good value of the current m iterations. After 20 iterations, we use the final set of N variables \( \omega \) and \( b \) to test the new data. Get the corresponding outputs and take the average of them as the final result. So we have:

\[
f(x_i) = \frac{1}{N} \sum_{j=1}^{N} f^{(j)}(x_i) \quad j = 1, 2, ..., N
\]

where \( f^{(j)}(x_i) \) is the output of each OS-ELM, and \( f(x_i) \) is the output of the input of \( x_i \) passes through the whole network.

4. Experimental results and comparison

In this part, we compare the performance of ENOS-ELM with EOS-ELM which don’t have cross-validation section, and the original OS-ELM. What data sets we use in this article is shown in Table 1.

| Dataset      | Training samples | Testing samples | Classes number | Feature dimension |
|--------------|------------------|-----------------|----------------|------------------|
| Wine         | 120              | 58              | 3              | 13               |
| Monks-1      | 300              | 132             | 2              | 6                |
| New-thyroid  | 140              | 75              | 3              | 5                |
| Segmentation | 1500             | 810             | 7              | 19               |

4.1. Model selection

According to the previous literature, we know that one of the important tasks for OS-ELM is to select the best number of hidden nodes. Because hidden nodes have a great impact on online learning performance, the number of optimal hidden nodes corresponding to different data sets is different. Referring to article [9], we use the same hidden nodes for three experiments on the same data set, and each data set chooses the hidden nodes. Tibetan nodes are given in Table 3 and 4. At the same time, for OS-ELM learning block by block. Another thing we need to determine is the size of the block. Generally speaking, OS-ELM uses two fixed block modes, 20-by-20 and 1-by-1, and a variable block mode [10, 30]. We will compare these three modes separately. For integrated networks, the size of the integrated network determines the network test time, complexity, stability, and a combination of three factors. It is showed in Table 2 that the accuracies of classification increase monotonically until the number of networks is larger than 20. So the two integrated networks EOS-ELM and ENOS-ELM, we choose an integration number of 20. For cross-validation, K usually takes from 5 to 10, now we choose ten folds cross-validation method for experiment.

| Dataset      | The testing accuracies (%) |
|--------------|-----------------------------|
|              | 5   | 10  | 15  | 20  | 25  | 30  |
| EOS-ELM      | 89.38 | 88.80 | 88.80 | 90.80 | 90.34 | 89.36 |
| ENOS-ELM     | 93.67 | 94.25 | 94.21 | 95.78 | 95.66 | 93.25 |

4.2. Performance comparison

In this section, we will show a comparison of ENOS-ELM, EOS-ELM without cross-validation, and original ELM experimental results. All experiments were run in the Matlab R2014b environment, and
the computer's properties were CPU 3.20 GHz, RAM 4.00 GB, and the training and testing process was run 50 times. Table 3 shows the experimental results of sig as the activation function.

Table 3. Comparison of OS-ELM, EOS-ELM and ENOS-ELM

| Dataset         | Algorithm | Learning Mode | Training time (s) | Testing Accuracy (%) | Hidden nodes |
|-----------------|-----------|---------------|-------------------|----------------------|--------------|
| Wine            | OS-ELM    | 1-by-1        | 0.0068            | 97.22                | 25           |
|                 |           | 20-by-20      | 0.0011            | 97.19                | 25           |
|                 | EOS-ELM   | 1-by-1        | 0.0357            | 97.38                | 25           |
|                 |           | 20-by-20      | 0.0085            | 97.40                | 25           |
|                 | ENOS-ELM  | 1-by-1        | 0.8543            | 98.71                | 25           |
|                 |           | 20-by-20      | 0.2524            | 98.69                | 25           |
| Monks-1         | OS-ELM    | 1-by-1        | 0.0543            | 78.74                | 80           |
|                 |           | 20-by-20      | 0.0098            | 78.76                | 80           |
|                 | EOS-ELM   | 1-by-1        | 1.0872            | 78.77                | 80           |
|                 |           | 20-by-20      | 0.1486            | 78.74                | 80           |
|                 | ENOS-ELM  | 1-by-1        | 5.3691            | 79.65                | 80           |
|                 |           | 20-by-20      | 1.0132            | 79.68                | 80           |
| New-thyroid     | OS-ELM    | 1-by-1        | 0.0066            | 89.68                | 20           |
|                 |           | 20-by-20      | 0.0010            | 89.65                | 20           |
|                 | EOS-ELM   | 1-by-1        | 0.0335            | 90.82                | 20           |
|                 |           | 20-by-20      | 0.0031            | 90.82                | 20           |
|                 | ENOS-ELM  | 1-by-1        | 2.5142            | 95.78                | 20           |
|                 |           | 20-by-20      | 0.4524            | 95.57                | 20           |
| Segmentation    | OS-ELM    | 1-by-1        | 10.154            | 94.30                | 180          |
|                 |           | 20-by-20      | 1.4624            | 94.31                | 180          |
|                 | EOS-ELM   | 1-by-1        | 7.553             | 94.18                | 180          |
|                 |           | 20-by-20      | 7.553             | 94.19                | 180          |
|                 | ENOS-ELM  | 1-by-1        | 95.451            | 95.11                | 180          |
|                 |           | 20-by-20      | 8.175             | 95.09                | 180          |

As can be seen from Tables 3, the ENOS-ELM algorithm requires more training time than the original OS-ELM and EOS-ELM algorithms, but its accuracy is improved. However, in terms of the choice of learning models, ENOS-ELM has the same characteristics as the other two algorithms, that is, the learning model 20-by-20 saves time compared to the learning model 1-by-1. And as we can see in the tables, the 20-by-20 model spend less time than 1-by-1 model, and when ENOS-ELM choose those models, the training times are not too longer than others. So we can sum up that the ENOS-ELM is more suitable big chunk of data

In addition, we validate the algorithm using the California Institute of Technology database Leaves 1999. Leaves 1999 consists of 186 leaf photographs taken in different backgrounds with a size of 896*502. Some of the three leaf photographs are shown in Fig.1.
Figure 1. Two types of leaves: (a) Class A leaves (b) Class B leaves

In the experiment, all three algorithms use Sigmoid function. Similarly, 8-fold cross-validation is used. The number of basic networks is 20, one-by-one mode, and the number of hidden nodes is 20, 50 and 80, respectively. The average of 30 test runs is the final result. For ease of calculation, we set the size of each picture to 16*16. The comparison results are shown in Table 4. From the experimental results, we can see that the improved algorithm can still achieve better experimental results for image classification experiments. High accuracy can be achieved without too many hidden nodes.

| Hidden nodes | OS-ELM (%) | EOS-ELM (%) | ENOS-ELM (%) |
|--------------|------------|-------------|--------------|
| 20           | 85.94      | 86.85       | 86.98        |
| 50           | 91.93      | 93.75       | 94.33        |
| 80           | 98.16      | 98.28       | 98.95        |

5. Conclusion
This paper proposes an ensemble of online sequential extreme learning machine based on cross-validation (ENOS-ELM). By introducing ensemble learning methods and cross-validation strategies into the training process, the goal of reducing over-fitting and improving generalization ability is achieved. The experimental results prove that ENOS-ELM is superior to several classification tasks of the original OS-ELM algorithm and EOS-ELM learning algorithm. Although the proposed method takes more time to train than the previous algorithms, compared with the OS-ELM learning algorithm and the EOS-ELM learning algorithm, the generalization performance is better and the training accuracy is improved, especially in the block comparison. In the bigger chunk, this advantage is more prominent.

Acknowledgments
The work was supported by National Key R&D Program of China (Grant Nos. 2018YFB1306600), National Natural Science Foundation of China (Grant Nos. 61571372, 61672436, 61601376), Fundamental Science and Advanced Technology Research Foundation of Chongqing (cstc2017jcyjBX0050, cstc2016jcyjA0547), Fundamental Research Funds for the Central Universities (Grant Nos. XDJK2016A001, XDJK2017A005).

References
[1] Liang N.-Y.; Huang G.-B.; Saratchandran.P; Sundararajan. N; (2006) A fast and accurate online sequential learning algorithm for feedforward networks, IEEE Transactions on Neural Networks. 17, 6:1411–1423.
[2] Huang G.-B.; Zhu Q.-Y.; Siew C.-K.; (2004) Extreme learning machine: a new learningscheme of feedforward neural networks, in: Proceedings of International Joint Conference on Neural Networks. Budapest, Hungary. 985–990.

[3] Huang G.-B.; Zhu Q.-Y.; Siew C.-K. (2006) Extreme learning machine: theory and applications, Neurocomputing 70:489–501.

[4] Huang G.-B.; Siew C.-K. (2004) Extreme learning machine: RBF network case, in: Proceedings of the 8th International Conference on Control, Automation, Robotics and Vision (ICARCV’04), Kunming, China. 1029–1036.

[5] Breiman. L. (1996) Bagging predictors, Machine Learning 24, 2:123–140.

[6] Schapire.R.E. (1990) The strength of weak learnability, Machine Learning 5, 2:197–227.

[7] Freund.Y; Soh.Y.C; Huang G.-B. (2006) Ensemble of online sequential extreme learning machine, Neurocomputing 72: 3391–3395.

[8] Zhou Z. H.; Wu J.; Tang W. Ensembling neural networks: Many could be better than all, Artif. Intell., 137:39–263.

[9] Hansen L. K. and Salamon P. (1990) Neural network ensembles, IEEE Trans. Pattern Anal. Machine Intell., 12:993–1001.

[10] Liu N. and Wang H.,(2010) Ensemble Based Extreme Learning Machine, IEEE SIGNAL PROCESSING LETTERS, 17.