Advances in Brain Inspired Cognitive Systems:
Proceedings of the 9th International Conference, BICS 2018, Xi'an, China, July 7–8, 2018
ISBN:978-3-030-00562-7; DOI:10.1007/978-3-030-00563-4_2

How Good a Shadow Neural Network is for Solving Non-linear Decision Making Problems

Hongmei He[1], Zhilong Zhu*[2], Gang Xu[2], Zhenhuan Zhu[3]

[1]Manufacturing Informatics Centre, SATM,Cranfield University, Cranfield, MK43 0AL, UK
Email: h.he@cranfield.ac.uk
[2]School of Electronic Engineering, Anhui Polytechnic University, China
[3]Advanced Laser Ltd, 3 Raleigh Street, Stockport, UK, SK5 7ER

Abstract. The universe approximate theorem states that a shadow neural network (one hidden layer) can represent any non-linear function. In this paper, we aim at examining how good a shadow neural network is for solving non-linear decision making problems. We proposed a performance driven incremental approach to searching the best shadow neural network for decision making, given a data set. The experimental results on the two benchmark data sets, Breast Cancer in Wisconsin and SMS Spams, demonstrate the correction of universe approximate theorem, and show that the number of hidden neurons, taking about the half of input number, is good enough to represent the function from data. It is shown that the performance driven BP learning is faster than the error-driven BP learning, and that the performance of the SNN obtained by the former is not worse than that of the SNN obtained by the latter. This indicates that when learning a neural network with the BP algorithm, the performance reaches a certain value quickly, but the error may still keep reducing. The performance of the SNNs for the two databases is comparable to or better than that of the optimal linguistic attribute hierarchy, obtained by a genetic algorithm in wrapper or in terms of semantics manually, which is much time-consuming.
keywords: Artificial Neural Networks, Shadow Neural Network, Universe Approximate Theorem, BP algorithm, Performance Driven, Incremental approach, Non-linear problems, Decision Making.

1 Introduction

An artificial neuron network (ANN) is a computational model, mimicking the structure and functions of biological neural networks. It provides an easy approach to creating the relations between input attributes and the output based on a limit set of data, in stead of an exact mathematic function, which we may not be able to create. The ability to learn by examples makes ANNs very flexible and powerful. Although there exists bias to the real relation between inputs and outputs, ANN is still a good approach to solving many non-linear mapping problems.

Deep neural networks (DNNs) have been successfully applied in two main areas: image processing and speech recognition. Especially, deep convolutional nets (ConvNets) have brought about breakthroughs in video [1,2], image processing [3], object

*Corresponding author: Zhilong Zhu<zhuzhilong919@ahpu.edu.cn>

detection [4], as well as audio [5] and speech recognition [6]. The properties of compositional hierarchies of images, speech and text promote the capacities of deep neural networks. However, we cannot always see the semantics of higher-level features in many real-world cases as in image and acoustic modelling.

The universal approximation theorem, first with sigmoid activation function proved by Cybenkot in 1989 [7], states that a shallow neural network (with one hidden layer, containing a finite number of neurons) with a non-polynomial activation function can approximate any function, i.e. can in principle learn anything [8,9]. This indicates that we do not always need to use DNNs. A shallow neural network could be enough to solve non-linear approximate problems.

The back-propagation (BP) algorithm is a classic training algorithm of ANNs. Blum and Rivest [10] proved that training a 2-layer, 3 nodes and n inputs neural network with the BP algorithm is NP-Complete. Obviously, the big barrier of blocking the applications of deep neural networks is the computing complexity, although it shows great attractive on solving complex non-linear problems. With the strong capability of GPU, deep learning for 2-20 depth networks is successful (e.g. Google AlphaGo). Also, the success of deep learning in image and acoustic modelling benefits from GPU computing. However, in many cases, we may not need to use GPU, or even we do not have GPU to support the calculation, for example, in an application of embedded intelligence.

Improving learning performance for all ANN applications is necessary. Basically, there are four kinds of approaches to improving the performance of ANNs: (1) improving data, which is important for training an ANN; (2) improving training algorithm, for which many notable algorithms have been developed in addition to the BP algorithm; (3) algorithm tuning, for which, some evolutionary algorithms were developed to optimise the parameters and neural network structures; (4) using ensembles.

Recently, Zhang et al. [11] proposed a dynamic neighborhood learning-based gravitational search algorithm. This approach can improve search performance in convergence and diversity of an evolutionary optimisation. A shadow neural network (SNN) is a feed-forward neural network (FNN) with only one fully connected hidden layer. If we use an evolutionary optimisation to find the best SNN with a specified number of hidden neurons as individuals, then there will be much redundant computing, as SNNs with the same number of hidden neurons but different distribution in the permutation of the individuals in the evolutionary optimisation have the same performance. There was also some research on incremental approach. For example, Bu et al. [12] proposed an incremental back-propagation model for training neural networks by adapting the parameters and the neural structure, and used the Singular Value Decomposition on the weight matrix to reduce some redundant links. The final neural network is not a fully connected FNN. He et al. [13] used the incremental approach based on information gain to select features for SVM spam detector.

In this research, we examine how good a shadow neural network is for solving non-linear decision making problems, and propose a new performance-driven incremental approach to finding the suitable number of hidden neurons in a shadow neural network. This approach overcomes not only the shortages of a population searching in evolutionary algorithm, which is much expensive in computing complexity, but also the shortages in both the randomness and computing complexity of ensembles. We use

two case studies on the two benchmark databases, Breast Cancer in Wisconsin [14] and SMS spams [15], from UCI machine learning repository [16] to validate the correction of the universal approximation theorem.

2 Methodology

2.1 A Multi-layer FNN

A multiple layer FNN is a computational graph whose nodes are computing units and whose directed edges transmit numerical information from low layer nodes to upper layer nodes. One neuron represents a linear classifier, the simplest neural network, using an activation function (e.g. sigmoid function in Eq. (2)) to produce the result. A neuron can be described with the following function:

$$v_j = \sum_{i=0}^{k} w_{ij} x_i, \tag{1}$$

$$y = f(v) = \frac{1}{1 + e^{-\alpha(v-b)}}, \tag{2}$$

where, j represents a neuron, to which the outputs of all neurons (i=1...k) in lower layer are input; b is the bias. The sigmoid's output $y \in [0, 1]$. We use $n_1 - n_2 ... - 1$ to denote the structure of a neural network, where n_k is the number of neurons at the k^{th} hidden layer, the last figure 1 represents one output neuron, and input neuron number is the number of input attributes by default.

2.2 The classic BP algorithm

Neural network learning is to find the optimal weights so that the network function φ approximates the function f representing the given data as closely as possible. Namely, given a training set $(x - 1, t_1), ..., (x_n, t_n)$, it is to minimise the error function of the network, defined as

$$E = \frac{1}{2} \sum_{i=1}^{n} \|o_i - t_i\|^2, \tag{3}$$

where, o_i is the output of the FNN for input sample x_i, t_i is the target output. The basic idea of BP algorithm is to use error back propagation to update weights in a fixed structure of ANN. The process is: (1) initialise the weights of the network randomly, (2) perform feed-forward computation to get the output of the network, and calculate the error between the output of the network and the target value (Eq. (3)), (3) calculate the gradient of the error function for all lower layers, and update the weight in terms of the back-propagated error, and repeat the steps of (2) and (3) until the average error ($\varepsilon = E/n$) is reduced to a specified small value.

Performance calculation A true estimation is the result, when the estimated probability $p(y|\boldsymbol{x})$ that the state of a decision variable y with measurement vector $\boldsymbol{x}$ is '+' or '-' is larger than a threshold (e.g. 05.). Classic performance measurements include confusion matrices, accuracy (A), and F_1 score, ROC curve, and the area under ROC curve (AUC). Assume P positive samples and N negative samples in the tested data set. The confusion matrices include the four parameters: true positive rate or recall (TPR=TP/P), true negative rate(TNR=TN/N), false positive rate(FPR=FP/N), and false negative rate (FNR=FN/P). The accuracy is the ratio of the number of true estimations for both states to the number of testing samples. The F_1 score is the harmonic average of the precision (TP/(TP+FP)) and recall. A ROC curve is a graphical plot of the true positive rate against the false positive rate at various threshold settings, and the area under the ROC curve has been formalized in [17].

The updated BP algorithm In the general BP algorithm, the stop criteria depends on the average error. The question is how small the average error is sufficient. If the average error is too small, then the number of learning iterations could be large. Usually, a maximum number of iterations is set, in case the average error cannot converge to the specified value. Moreover, the neural network system may produce over fitting problem. To avoid over fitting, usually a small data set is used to validate the performance during the training process. Once the error of the FNN on the validation data is increasing, while the error of the FNN on training data is still decreasing, the training process will be stopped. This increases the complexity of training.

In fact, for a decision making problem, the goal of neural network training is to gain high accuracy. The primary experimental results show that when average error arrives a certain value, the performance for decision making could not be improved further. Therefore, the stop criteria can be set to evaluate the neural network performance, such as Accuracy (A), F1-score (F_1), true positive rate (TPR). Namely, the learning process will be stopped until the performance of the network has not been improved for a certain number (T) of iterations (called convergence tolerance) (Algorithm 1). When the average error is used as the stop criteria p, the line 10 in Algorithm 1 should be $p < best_p$.

2.3 Incremental Construction of FNN

Recently, David and Greental [18] proposed using a GA to optimise Deep Neural Networks, but they didn't implement it. As we argued in the introduction, the success of deep neural network benefited from the GPU computing. Assume the complexity of deep neural networks is χ, the number of evolutionary generations is $\mathcal{G}$, and the population size $\mathcal{P}$, and assume we have $\mathcal{P}$ processors to parallelises the GA, the complexity of the optimisation process of deep neural network is $O(\mathcal{G}\chi)$ otherwise, it is $O(\mathcal{G}\chi\mathcal{P})$, which may be intolerant. Hence, we propose an incremental approach to finding the best structure of FNN, by starting from one neuron in one hidden layer, increasing a neuron in the hidden layer each step, and then increasing one hidden layer until the performance does not change.

Algorithm 1 UpdatedBP(D(X,Y),Net, T)

```
Initialise(Net);
I = 0, k = 0;
p = 0;
while (I<MAX_IT) do
    Ŷ=feedforward(X);
    backpropagation (Y, Ŷ);
    Net = updateWeight(Net);
    best_p = p;
    p = calPerformance(Y, Ŷ);
    if (p>best_p) then
        k = 0;
        best_p=p;
    else
        k = k+1;
        if (k>T) then
            break;
        end if
    end if
end while
```

3 Experiments and Evaluation

The test platform is a laptop with Windows 10 and Intel (R) Core (TM)2 Duo CPU T7300 @2GHZ 2GB memory. A software tool embedded with the algorithm is implemented in VC++. The FNNs will be evaluated with the accuracy A, F_1 score, TPR and AUC. For each database, five experiments are conducted: (1) Error-driven FNN;(2) A-driven FNN;(3) $F1$-driven FNN; (4) TPR-driven FNN, and (5) a ten folder crossing validation. The convergence tolerance T is set to 200. The best performance will be recorded for each experiment.

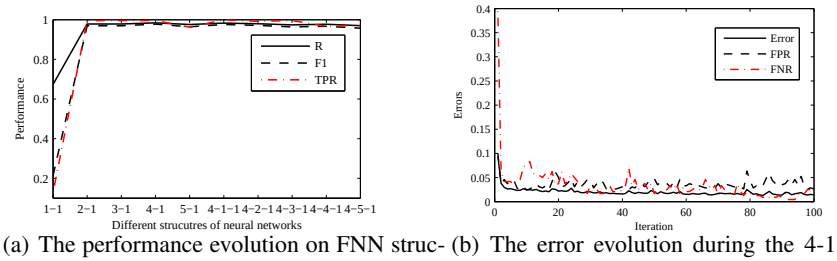
First, we apply the incremental approach to changing the structure of FNN on the whole data set, and observe how the performance changes when structure is changed; secondly, examine the training processes of the best structure of neural network, and observe the effect of different termination criteria on the training process and the performance; finally, perform ten folder crossing validation, 10% of data as test set, and the rest 90% of data as training set. Also we compare the performance of the best structural neural networks with ten-folder crossing validation to the performance in literature.

3.1 Case study on the database of the Wisconsin Breast Cancer

The Wisconsin Breast Cancer (WBC) database was created by Wolberg [14], containing 699 samples, in which 458 samples are benign, and 241 samples are malignant. There are nine basic attributes $x_0, x_1, ..., x_8$, with integer range [1,10]. The missing value of an attribute in an instance of the database is replaced with the mean value of the attribute on the corresponding goal class.

Performance of different structures of FNN The experiment is conducted by incrementally changing the structure of the neural network, and the training process of each

structure of neural network will be stopped when the minimal average error has not be improved up to 200 epoches. Fig. 1 (a) shows the performance evolution when the FNN structure changes. Obviously, one hidden neuron can not well represent the function, but the FNN with more than one hidden neurons can well represent the function, although the performances of different structures of FNNs are slightly different. The structure 4-1 of FNN achieves the best performance in all performance measurements of A, F_1, TPR and AUC. The structure 4-3-1 of FNN obtains almost same AUC and TPR as the structure 4-1 of FNN does, but the performance of F_1 and A are slightly lower than that obtained by the structure 4-1 of FNN. Namely, a shadow neural network is enough to represent the function given the data.



(a) The performance evolution on FNN structures

(b) The error evolution during the 4-1 FNN training

Fig. 1. Performance and Training Process

The evolution process during the 4-1 SNN training Fig. 1 (b) shows the error evolution during the training process for the 4-1 FNN. It can be seen that the error (the solid black line in Fig. 1 (b) is gradually decreasing during the training process. However, the false negative rate has a large vibration after about 10 iterations, while false positive rate has a decreasing trend although there are many fluctuations. After 20 iterations, the FPR (dashed line) and FNR (dashdotted line) seems having an opposite behaviour. Namely, while FPN is increasing, FNR is decreasing.

Performance of the 4-1 SNN for different termination criteria For a critical decision making problem, we do not want to make a wrong decision on any positive instances, namely we expect the TPR is 100%. In this section, we observe the end-loop and performances when the criteria of error (ε), A, F_1 and TPR have not be improved for up to 200 iterations, respectively. Table 1 provides the performances of the 4-1 SNN and the iteration index (best iteration) and the training time (best time) after which the observed performance has not been improved. For the four stop criteria, the performances of the 4-1 SNN are almost same, but the end-loops are very different, and error-driven BP learning has the largest end-loop. It means that the performance, arriving a certain value, is not improved further while the average error is still decreasing.

Ten-folder Crossing Validation Now we validate the performance of the 4-1 SNN for different termination criteria, using ten-folders crossing validation. Table 2 shows the results. Obviously, the performance with ten-folder crossing validation is lower than that in Table 1. This indicates how robust the trained SNN is when it works on unseen data. From Table 2, it can be seen that the performances of A, F_1 and TPR for the

Table 1. Performance parameters of 4-1 FNN for different stop criteria

Criteria	A	F_1	TPR	AUC	$bestIN$	$bestTT$(ms)
ε	0.9828	0.9757	1	0.9970	527	1015
A	0.9828	0.9755	0.9917	0.9938	154	250
F_1	0.9857	0.9797	1	0.9963	461	750
TPR	0.9814	0.9737	1	0.9960	202	328

termination criteria of ε, A and F_1 are very close. The performance of the 4-1 SNN for the termination criterium TPR is lower than that for other criteria. The average end-loop for error-driven BP learning keeps the largest.

The average accuracy and standard deviation for the ten runs of ten folder crossing validation is presented in the form of (a±b), where a is average accuracy, and b is the standard deviation, and it is (0.955±0.006), of which, the average is slightly lower that of the accuracy (0.967± 0.02) obtained by the optimal linguistic attribute hierarchy (LAH) [17], which was obtained by a GA wrapper. Also an SNN obtains more stable accuracies than the LAH, and the accuracies of the SNN fall in the accuracy range of LAH.

Table 2. Performance parameters of the 4-1 FNN on different stop criteria

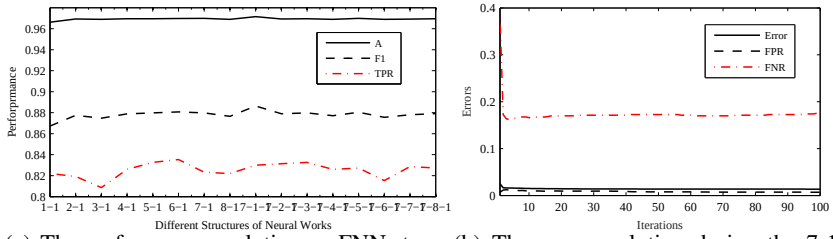
Criteria	A	F_1	TPR	end-loop
ε	0.9551	0.9380	0.9643	1372
A	0.9507	0.9249	0.9351	173
F_1	0.9536	0.9349	0.9347	162
TPR	0.9101	0.8475	0.8634	76

3.2 Case study on the SMSSpamCollection database

The SMSSpamCollection database [15], has 5574 raw messages, including 747 spams. He et al. [13] extracted 20 features from the database, and the number of features was reduced to 14 by combining some features with similar meanings [19]. We use the 14 attribute database for the experiments.

Performance of different structures of FNN Similar to the experiments on WBC, we apply error-driven BP training on the whole data set. Fig. 2 (a) shows the performance evolution when the FNN structure changes. It can be seen that the 1-1 FNN can well represent the function from the SMS spams data, and the TPR of the 3-1 FNN is lower than that of the 1-1 FNN. The performances of different structures of FNNs are slightly different. But the performances, A, F_1 and TPR are clearly separated. The F_1 score keep around 0.88, the TPR is waved in [0.8, 0.84], and the accuracy keeps above 0.96. The 6-1 FNN achieves the best performance in F_1 and TPR, but slightly lower performance in A than the 7-1 FNN, and more importantly, the end-loop for training the 6-1 FNN reaches to the preset maximum iteration 20000, while the end-loop for training the 7-1 FNN is only 923. Therefore, the second hidden layer is constructed with the 7 neurons in the first hidden layer. Similar to the WBC experiments, the experimental results show

that an SNN is enough to represent the function of the specific data. At the same time, we can conclude that the feature extraction still need to be improved, or additional information is needed for improving the true positive rate of neural networks.



(a) The performance evolution on FNN structures

(b) The error evolution during the 7-1 FNN training

Fig. 2. Performance and Training Process

The evolution process during the 7-1 SNN training Fig. 2 (b) shows the error evolution during the training process for the 7-1 SNN. The end-loop of the training process is 923. To clearly show the trend of error evolution, we take the first 100 iterations of the training process for plotting Fig. 2 (b). It can be seen that the error (the solid black line) slightly decreases, and the FNR (the red dashed dot line) drops from above 0.35 to 0.15, but the FNR slightly increases at the second iteration. After the second iteration, the average error, FPR and FNR almost do not change.

Performance of the 7-1 SNN for different termination criteria Similar to the experiments on WBC, we observe the end-loop and performances of SNNs on the SMSSpams data when the criteria of average error (ε), A, F_1 and TPR have not be improved for up to 200 iterations, respectively. Table 3 provides the performances of the trained 7-1 SNN and the iteration index and the training time at the best epoch. It can be seen that the performances in A, F_1 and TPR for the four termination criteria are very close. Especially, for the termination criteria A and F_1, the training stops at the same end-loop. Therefore, the SNNs obtain completely same performances in A, F_1 and TPR. The training process for the termination criterium of average error is the longest, while the termination criterium of TPR produced the shortest training process. The F_1 score and accuracy A do not change after 71 iterations. The TPR does not change after iteration 3, and the SNN at iteration 3 obtains the highest TPR, and the performance A and F_1 are similar to that at iteration 71, and even at iteration 923. This indicates that the performance of the SNN keeps stable, although the average error continues being slightly improved.

Ten-folder Crossing Validation Now we validate the performance of the 7-1 SNN for different termination criteria, using ten-folders crossing validation. Table 4 shows the results. The performance with ten-folder crossing validation is similar to that in Table 3. This indicates the trained SNN is robust when it works on unseen data. From Table 3, it can be seen that the performances of A, F_1 and TPR for the four termination criteria are very close, even the SNN trained with the termination criterium of TPR obtains the best performance. The performance of the 7-1 SNN with ten-folder crossing validation is

Table 3. Performance parameters of 7-1 FNN for different termination criteria

Criteria	A	F_1	TPR	AUC	$bestIN$	$bestTT$(ms)
ε	0.9699	0.8798	0.8233	0.9669	923	27406
A	0.9711	0.8851	0.8300	0.9652	71	1906
F_1	0.9711	0.8851	0.8300	0.9652	71	2094
TPR	0.9686	0.8774	0.8380	0.9623	3	109

much better than that of the linguistic attribute hierarchy in [19], of which, the accuracy on the SMS Spam 0.9458, and the TPR 0.7323. The end-loop for error-driven BP on SMS Spams is much larger than that for performance-driven BP.

Table 4. Performance parameters of the 7-1 FNN for different termination criteria

Criteria	A	F_1	TPR	end-loop
ε	0.9621	0.8615	0.8257	10062
A	0.9627	0.8619	0.8351	97
F_1	0.9645	0.8726	0.8230	53
TPR	0.9670	0.8818	0.8527	54

4 Conclusions

The contribution of the research are summarised as follows: (1) Validating the universe approximate theorem using experiments: a shadow neural network can well represent the function from data, and the number of hidden neurons, taking the half of input number, is good enough. (2) Providing an simple incremental approach to finding the best shadow neural network. This approach overcomes not only the shortages of a population searching in evolutionary algorithm, which is much expensive in computing complexity, but also the shortages in both the randomness and computing complexity of ensembles; (3) Updating the classic BP algorithm with different termination criteria. Performance-driven BP learning could help reduce the training time and avoid over fitting; (4) Having the comparable or better performance of the trained neural network on the two benchmark data bases, compared to the optimal linguist attribute hierarchy. The research results show that a shadow neural network seems matching the property of the human brain: when an individual gets the first impression to a thing, without new information, the individual cannot change the impression. Partial linked neural network will be investigated in future.

Acknowledgements

This research is sponsored by the Key project of natural science in universities in Anhui Province(KJ2018A0111).

References

1. A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, F. F. Li, Large-scale Video Classification with Convolutional Neural Networks, IEEE Conference on Computer Vi-

sion and Pattern Recognition (CVPR), Columbus, OH, USA, 23-28 Jun. 2014. DOI: 10.1109/CVPR.2014.223.
2. Z. Wang, J. Ren, D. Zhang, M. Sun and J. Jiang, A deep-learning based feature hybrid framework for spatiotemporal saliency detection inside videos, Neurocomputing 287, 2018 pp. 68-83.
3. L. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, CoRR, abs/1606.00915, 2016, http://arxiv.org/abs/1606.00915, arXiv: 1606.00915.
4. S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Journal IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), June 2017, pp. 1137-1149.
5. H. Lee, Y. Largman, P. Pham and A. Y. Ng, Unsupervised feature learning for audio classification using convolutional deep belief networks, in Proc. of the 22nd International Conference on Neural Information Processing Systems (NIPS'09), Vancouver, British Columbia, Canada, 07 - 10 Dec. 2009, pp. 1096-1104.
6. Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. Laurent, Y. Bengio and A. C. Courville, Towards End-to-End Speech Recognition with Deep Convolutional Neural Networks, CoRR, abs/1701.02720, 2017, http://arxiv.org/abs/1701.02720, arXiv: 1701.02720.
7. G. Cybenkot, Approximation by Superpositions of a Sigmoidal Function, Math. Control Signals Systems (1989) 2:303-314
8. K. Hornik, Approximation Capabilities of Multilayer Feedforward Networks, Neural Networks, 4(2), 1991, pp. 251–257.
9. M. Leshno, V. Lin, A. Pinkus, and S. Schoken, Multilayer Feedforward Networks With a Nonpolynomial Activation Function Can Approximate Any Function, Neural Networks, 6, 1993, pp. 861-867.
10. A. L. Blum and R. L. Rivest, Training a 3-Node Neural Network is NP-Complete. Neural Networks, 5, 1992, pp. 117-127.
11. A. Zhang, G. Sun, J. Ren, X. Li, Z. Wang and X. Jia, A dynamic neighborhood learning-based gravitational search algorithm, IEEE transactions on cybernetics 48 (1), 2018, pp. 436-447.
12. F. BU, Z Chen and Q Zhang, Incremental updating method for big data feature learning. Computer Engineering and Applications, 3. 2015, 92-101.
13. H. He, A. Tiwari, J. Mehnen, T. Watson, C. Maple, Y. Jin, B. Gabrys, Incremental Information Gain Analysis of Input Attribute Impact on RBF-Kernel SVM Spam Detection, WCCI2016, Vancouver, Canada, 24-29 July, 2016.
14. W. H. Wolberg, and O. L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology. In Proceedings of the National Academy of Sciences, 87, 1990, pp. 9193–9196.
15. T.A.Almeida,J.M.G.Hidalgo,andA.Yamakami.Contributionstothe study of sms spam filtering:new collection and results. In DocEng'11, Mountain View, California, USA, 19-22 September 2011.
16. D. Dua and E. Karra Taniskidou, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2017.
17. H. He and J. Lawry, Linguistic Attribute Hierarchy and Its Optimisation for Classification Problems, Soft Computing, 18(10), Oct. 2014, pp. 1967-1984.
18. E. David and I. Greental, Genetic Algorithms for Evolving Deep Neural Networks, ACM Genetic and Evolutionary Computation Conference (GECCO), pages 1451–1452, Vancouver, Canada, July 2014.
19. H. He, T. Watson, C. Maple, J. Mehnen, A. Tiwari, Semantic Attribute Deep Learning with A Hierarchy of Linguistic Decision Trees for Spam Detection, IJCNN2017, Anchorage, Alaska, USA, 14-19 May 2017.