International Journal of Engineering & Technology, 7 (2.14) (2018) 53-56

SPC

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

An Analysis of Large Data Classification using Ensemble Neural Network

Mumtazimah Mohamad[1]*, Wan Nor Shuhadah Wan Nik[1], Zahrahtul Amani Zakaria[1], Arifah Che Alhadi[2]

[1]Faculty of Informatics & Computing, Universiti Sultan Zainal Abidin, Besut, Terengganu
[2]School of Informatics & Applied Mathematics, Universiti Malaysia Terengganu, Kuala Nerus, Terengganu
**Corresponding author E-mail: mumtaz@unisza.edu.my*

Abstract

In this paper, operational and complexity analysis are investigated for a proposed model of ensemble Artificial Neural Networks (ANN) multiple classifiers. The main idea to this is to employ more classifiers to obtain a more accurate prediction as well as to enhance the classification capabilities in case of larger data. The classification result analyzed between a single classifier and multiple classifiers followed by the estimates of upper bounds of converged functional error with the partitioning of the benchmark dataset. The estimates derived using the Apriori method shows that the proposed ensemble ANN algorithm with a different approach is feasible where such problems with a high number of inputs and classes can be solved with time complexity of O(n^k) for some k, which is a type of polynomial. This result is in line with the significant performance achieved by the diversity rule applied with the use of reordering technique. As conclusion, an ensemble heterogeneous ANN classifier is practical and relevant to theoretical and experimental of combiners for the ensemble ANN classifier systems for a large dataset.

Keywords: *Classification; Complexity Approximation; Ensemble Neural Network, Large Data Neural Network Classifier*

1. Introduction

Many current computational models used in soft computing are described as input-output producing devices of linear combinations functions derived from a simple computational unit. Classification tasks model is an example of coefficient of linear combination using inner parameters of computational units that are adjustable. This classification tasks model is commonly constrained by model complexity. Thus, the selection of computational unit type is important to allow efficient learning from the given data by networks with a reasonable unit size.

The classification for large data using the Artificial Neural Network (ANN) is rarely discussed. However, previous research shows a bright side on large dataset classification tasks using ANN since many other researchers ignore the ANN advantage of universal approximation[1]. ANN, first known as the 'connectionist model', emerged from the simplified neuron introduced by McCulloch and Pitts in 1943. These neurons present a biological model as a conceptual component that could perform computational tasks and able to learn the complex nonlinear input relationships for sufficient data in training times that scales linearly with data size [2]. An enhanced alternative is required to utilize ANN ability in transforming traditional optimization algorithm to complex classification tasks that can be formulated as optimization problems [3]. Methods such as SVM or other advanced methods are not likely to be appropriate for large datasets classification. This is because most other methods need to solve a quadratic programming problem in order to find a separation hyper-plane, which causes an intense computational complexity[4].

The use of a single ANN classifier for large datasets leads to the unstable result of learning classifier and sensitive to several parameters. Furthermore, it works differently for different training data[5]. Previous researchers found that the performance of more than two classifiers is better than their base model since a single ANN tends to make errors on different examples[6]. Combining a group of different output of multiple classifiers using the ensemble strategy is used to solve the variety network's output[7]. It makes sense only if the classifiers are diverse or in other words, statistically independent.

This asymptotic method is one of the Apriori technique principle that emphasizes the analysis task on the specific problem prior to the design part. Analyzing the asymptotic usually requires the analysis of the running time or the space usage of the program and essentially needs the time and space estimates for the function of input size. This paper proposed techniques which are embedded in several strategies for a large dataset by having clusters of ANN classifiers that work independently to allow improvement of the classification task in order to classify a large dataset. This ensemble ANN is measured in terms of complexity measures in investigating the complexity of ensemble ANN for a large dataset.

This paper is organized in five sections starting with the introduction of the study. Section 2 describes the related proposed technique and the framework in detail. Section 3 describes the complexity model used for approximation for the proposed framework. Section 4 describes the experimental setup and elaborates the result of performance of classifiers respectively. Finally, Section 5 presents the summarization drawn from the study.

2. Ensemble learning algorithm for ANN cluster

In general, the idea behind the ensemble methods is the use of a cluster of classifiers and combining their predictive capabilities into a single classification task to obtain a more accurate and stronger prediction. Success result and finding of previous research on diversity [8, 9], which suggests that a single classifier diversity measures might not be accurate enough to capture all the relevant independency of each available classifier. In this study, the combination of prediction capabilities applies by adopting the reordering technique for the multiple classifiers as a diversity rule.
The development of combinational independent classifiers comprises of four phases in a framework. First, the dataset was pre-processed using input normalization and output denormalization. Secondly, the training data was then divided into a training set, testing set and validation set. The training dataset was adapted to the reordering technique for data re-sampling. The result of the reordering was configured with the partitioning task. Then, in the third phase, each partitioned parts were passed as individual ANN to the corresponding thread of multi-classifiers by a master classifier. Finally, the output was transformed into a reliability value and then passed back to the master ANN thread for an aggregated ensemble strategy.
In the second phase, the use of reordering is important to ensure the independent classifier is generated in the third phase. The original sequence and configuration can make all ANN fall in the same error the training variability chances are very low[10]. For multiple ANN classifiers, the original reordering is impractical because the error and output generated reflected dependent and insignificant classifiers where the training error is small and very similar to the other classifiers in ensemble networks[11]. The independent ensemble classifier in this study is dependent on the proposed enhancement for the reordering algorithm. The Distributed Reordering Technique as in the following algorithm is an enhanced reordering technique based on [10]. The enhanced distributed reordering shuffles the data sequence in training data for a given p classifier from P classifiers. The random weights are initialized for each of the classifiers to adapt to the probability of threading.

```
Function: Distribute Reordering sampling without replacement to multiple classifiers
1   BEGIN
2   for t=1 to J
3       Generate $DS_e$ by sampling DS without replacement
4       Draw $X_i$ from DS using P thread probability
5       for i=1 to N pattern
6           $Tr_i = x_i$(i, all columns)= DS(randRow, AllColumns)
7           Adjust probabilities $P$
8       END for
9   END for
10  Output the final training subsets (Tr$^1$, Tr$^2$,...Tr$^n$)
11  END
```

The probability P of threads is $\frac{p_i}{N}, i \in [1,..,N]$. In sampling without replacement, the two sample values are not independent. This means that the first sample can affect the second sample. In this case, the covariance between the two is not zero. Therefore, the sampling rather complicates the computations. This sampling generates different partitioned parts. The data DS results of training datasets $Tr_1, Tr_2, ... Tr_n$ have undergone the reordering algorithm in order to ensure sufficient training data with the diversity rule[12].

In the third phase, the carry load of partitioned datasets are passed to n multi-classifier threads. The individual classifier adopts a stochastic learning mechanism as it promotes efficiency of large data scalability[13] that is aligned with[14] that similarly showed the good performance on stochastic learning for I/O overhead communication. Each ANN thread q presents the network assigned partitioned datasets Tr_n of a batch pattern which is called a component gradient[15]. Each ensemble classifier with different datasets partitions are trained with back propagation neural network (BPNN) algorithm. A BPNN schemes minimizes the error E function by obtaining the new weights and threshold after epochs of training. Two phases are repeated until E converges to a possible minimum value. In this case, the stopping criteria is either achieving the error of 0.001 or up to 10000 iterations. The number of output y, which is F_3 is used as an indicator of reliable sufficiency of the ensemble classifier in integrating ensemble members.
The last phase is motivated by the decorrelation maximization method that has been used in [16] to select a suitable number of neural network ensemble members. This method created the diverseness for the ensemble neural network due to a smaller correlation value obtained from the available neural networks. It is assumed that there are p ANN classifiers result $(f_1, f_2, ... f_p)$ with n values of output. For each ANN classifier, the plural-correlation coefficient p_i is calculated with (1) as follows:

$$p_i^2 = r_i^T R_{-i}^T r_i \; (i = 1,2, ..., p) \quad (1)$$

where R is the variance of the error for a classifier divided by the square root of variance multiplied with the variance-covariance of classifier's error. For a specified threshold θ, if $\rho_i^2 < \theta$, then the classifier f_i should be removed from P classifiers. On the contrary, the classifier f_i should be retained. This phase involves the master ANN that locates the aggregated output. After all ANNs are decorrelated with each other, the respected error will be reduced as the number of ensemble members is increased after output aggregation. Therefore, the final output is aggregated for this simple average as given in (2).

$$\hat{y}_c(x) = \frac{1}{N_{networks}} \sum_{net=1}^{N_{networks}} y^{net}(x) \quad (2)$$

Then the class c, generates maximum of the average values, $\hat{y}_c$ to the respected pattern. The output average technique is the easiest combiner and it averages the individual classifier outputs, y^{net}, across the different classifiers, N^{net}. The notation of the variables shown in the previous chapter is modified by adding a super index referring to the network of the ensemble. The number of networks of the ensemble is given by N^{nets}.

3. Analysis of complexity model

The complexity of ANN demonstrates the quantification of interactions needed in order to approximate the error (k). Three layers of ANN are represented in order to compute output q such as follows:

$$q = \sum w_i y_i = \sum w_i r\,(x) \quad (3)$$

where w_i is the random weight, y_i is an input node of input x, and $r = \{r_i(x)\}_i$ is a dictionary of sigmoid and hyperbolic tangent activation functions. Such networks can implement nonlinear approximations of desired input-output functions f in approximation theory. For subsets $J \subseteq I$ of cardinality, where n refers to input x, $(x \in \mathbb{R}^k)$. $\mathbb{R}^k$ is the ridge function for $f_{ij}(x) = r(x\,.a - \theta)$, where r: $\mathbb{R} \rightarrow \mathbb{R}$ is the real number of monotone hyperbolic tangent functions which increase from -1 to 1.
The time complexity of a computational P is the minimum time needed by an algorithm to solve the problem. The complexity of a program is measured based on input size and the procedure is

shown in the following algorithm. This complexity calculation requires the $f(x)$ to be fitted as $f(n)$ and find the frequency of each similar asymptote function to be determined[17] in order to find the complexity, as follows.

```
Input: size n, $f(x)$,Output: Complexity $C$
1   BEGIN
2     Find the number of frequent count
3     Fit an equation as $f(n)$ for time f($x$) , $n$
        input
4     Find the asymptote function similar to
        $f(x)$.
5   Output $C$
6   END
```

Based on the above algorithm, the frequent count is calculated and will be fitted to the according function for each time of each input. After that, calculate the similar asymptote function of the respective function. The asymptotic behavior of a function $f(n)$ refers to the growth of $f(n)$ as n gets larger. The slower the asymptotic growth rate, the better the algorithm. If an algorithm runs within $T(n)$ time, it means that $T(n)$ is an upper bound of the running time that holds for all inputs size (n). This is called the worst case scenario. The worst case for a function considers how long it takes to run a function of the length of the input list.
Alternatively, an average case analysis can be chosen in order to focus on calculating the expected time spent on a randomly chosen input. The worst case scenario is used in this analysis in order to justify the large data case study. The asymptote function is derived by using the Big O notation to calculate the complexity of a program. The time complexity of P threads for multiple ANN has the properties of ANN with the communication part in the partitioning of the dataset and selection and combination of output from threads P. Each frequency count represents the notion of reordering techniques, communication of nodes of ANN threads, ANN training, and the selection and combination of outputs to ensemble the results.

4. Result and discussion

The MNIST dataset was selected in the study. It has 60000 examples as training set and 10000 examples for testing set. The validation dataset was drawn from the training set which is 10000 examples, which made the training set with 50000 examples. The solution of multiple ANN systems was tested using multithreading in a single computer representing a cluster that runs synchronously. All ANN classifiers were trained with different dataset partitions and parameters. For example, two ANN work with the same hidden nodes setup, but with different learning rate and momentum, two classifiers using different hidden node setup and learning rate/momentum with re-sampling techniques for input and etc. The accuracy of the recognition rate obtained from the result is measured by the total corrected recognition.
The proposed distributed reordering technique shows the better approximation for a high performance for de-correlation and combination of ANN classifiers. The calculated algorithm approximation for ANN cluster shows that $T(n)$ grows asymptotically not faster than n^4. The network size, excluding the number of non-input nodes, may increase to $O(n^5)$. This explains that for large enough input sizes n, the network size is less than cn^4 for some constant c. Such function f is said to have polynomial size complexity for N because n^4 belongs to a function of n. The function of (f) is shown in Fig.1 is asymptotically bounded below by $g(n)$ where there is some positive constant value and a particular value of n_0. The $f(n)$ function also has no bounds related because the function does not satisfy two conditions. First, it does not have a function of $\Omega(n^3)$ where $f(n) > n^3$ for all constant of n. Second, it does not have a function of $O(n^4)$ that implies both above and below by $g(n)$ asymptotically.

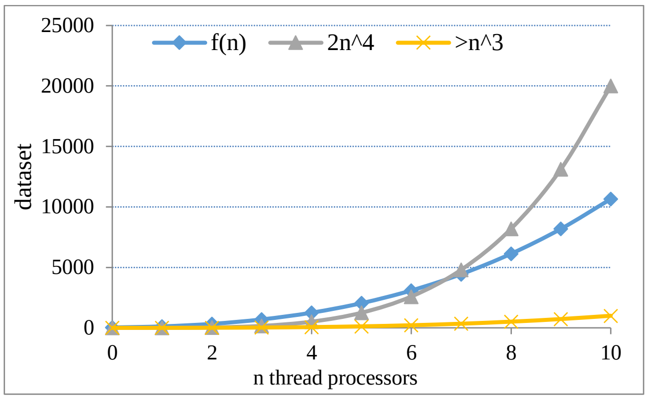

Fig.1: Result of Complexity Analysis

There exists an asymptote upper bound and a lower bound as represented by $f(n)\epsilon\ O(g(n))$ and $f(n)\epsilon\ \Omega\ (g(n))$. These growth functions show that for sufficiently large input parameters, f grows at a rate that may henceforth be greater than c(g) in regards to $g\in O(f)$. The time complexity for ensemble ANN,$T(n)$ asymptotically is bounded above by g up to a constant factor of 2 and with particular values of n_0 at 8 and the greater number than 8. It can be said that $T(n)$ grows asymptotically no faster than n^4. The function n^4 requires an amount of time that is in polynomial proportion to the number of input elements. The network size, excluding the number of non-input nodes, may increase to $O(n^5)$. This explains that for large enough input sizes n, the network size is less than cn^4 for some constant c. Such a function f is said to have polynomial size complexity for N because n^4 belongs to a function of n. The function of (f) is also asymptotically bounded below by $g(n)$ where there is some positive constant value and a particular value of n_0.
The result of ensemble ANN shows that the complexity between the multiple of ANN with additional processes for reordering and the other ensemble method have no significant difference of performance [18, 19]. The above result is in contradiction to the investigation done by [20] that found an ensemble strategy with a voting system with $O(n^2)$. However, they agree that the maximum computing complexity is $O(k^3)$ for all k clusters. The voting approach is very simple for an ensemble since no matrix calculation is needed for a large dataset.

5. Conclusion

Classifiers ensemble makes sense only if the classifiers are diverse. The reordering techniques that promote diversity have contributed a reduction error and training time for clusters of ANN. The proposed technique has enhanced the generalization ability among different ANN classifiers that provide a promising solution to other binary-class classification and recognition problems. From the analysis, with computing time complexity of $O(n^4)$, classifier ensemble should be considered as computationally feasible with an improved computing complexity for a large dataset. In other words, the time used by the algorithm is bounded by a polynomial in the size of the input. Additionally, the proposed solution is a kind of NP-complete problem. It is possible to reduce any other NP problem X to Y in polynomial time. This means that a Y problem can be solved if X problem has the solution. Moreover, there is very strong evidence that it is impossible to solve any NP-complete problem in less than exponential time. This proposed ANN clusters solution is a P class of problem where $P = NP$ because this problem can be verified in polynomial time as well as the subset problem.

Acknowledgement

This work was supported by Fundamental Research Grant Scheme (FRGS/1/2015/ICT02/UNISZA/02/1) by Malaysia Ministry of Higher Education and the Center of Research and Innovation Management of Universiti Sultan Zainal Abidin, Terengganu, Malaysia.

References

[1] Ciresan, D.C., et al., “Deep big simple neural nets excel on handwritten digit recognition”. *Neural Computation*, (2010). 22(12): 3207 - 3220.
[2] Bishop, C.M., *Neural networks for pattern recognition*. (1995): Oxford: Clarendon.
[3] Egmont-Petersen, M., D.d. Ridder, and H. Handels, Image processing with neural networks—a review. *The Journal of Pattern Recognition Society*, (2002). 35: 2279–2301.
[4] Stahl, F., M. Gaber, and M. Bramer, Scaling up data mining techniques to large datasets using parallel and distributed processing, in *Business intelligence and performance management*, P. Rausch, A.F. Sheta, and A. Ayesh, Editors. (2013), Springer London. 243-259.
[5] Windeatt, T., Accuracy diversity and ensemble MLP classifier design. *IEEE Transactions on Neural Networks*, (2006). 17(5): 1194-1211.
[6] Sospedra, J.T., Ensembles of artificial neural networks: *Analysis and development of design methods*, in Department of Computer Science and Engineering. (2011), Universitat Jaume I: Castellon.
[7] Ceamanosa, X., et al., A classifier ensemble based on fusion of support vector machines for classifying hyperspectral data. *International Journal of Image and Data Fusion*, (2010). 1(3): 293–307.
[8] Zang, W., et al., Comparative study between incremental and ensemble learning on data streams: Case study. *Journal of Big Data*, (2014). 1(1): 5.
[9] Kuncheva, L. and J. Rodríguez, A weighted voting framework for classifiers ensembles. *Knowledge and Information Systems*, (2014). 38(2): 259-275.
[10] Torres-Sospedra, J., C. Hernández-Espinosa, and M. Fernández-Redondo, Introducing reordering algorithms to classic well-known ensembles to improve their performance, in *Neural information processing*, B.-L. Lu, L. Zhang, and J. Kwok, Editors. (2011), Springer Berlin Heidelberg. 572-579.
[11] Woźniak, M., M. Graña, and E. Corchado, A survey of multiple classifier systems as hybrid systems. *Information Fusion*, (2014). 16(0): 3-17.
[12] Shields, M.W. and M.C. Casey, A theoretical framework for multiple neural network systems. *Neurocomputing*, (2008). 71(7–9): 1462-1476.
[13] Mohamad, M., M.Y.M. Saman, and M.S. Hitam, The use of output combiners in enhancing the performance of large data for ANNs. *IAENG International Journal of Computer Science*, (2014). 41(1): 38-47.
[14] Zinkevich, M., et al. Parallelized stochastic gradient descent. in *Advances in neural information processing systems*, (2010), 2595-2603.
[15] Babii, S. Performance evaluation for training a distributed backpropagation implementation. in *4th International Symposium on Applied Computational Intelligence and Informatics*. Timisoara IEEE, (2007), 273-278.
[16] Yu, L., S. Wang, and K.K. Lai, Credit risk assessment with a multistage neural network ensemble learning approach. *Expert Systems with Applications*, (2008). 34(2): 1434-1444.
[17] Engel, A., Complexity of learning in artificial neural networks. *Theoretical computer science*, (2001). 265(1–2): 285-306.
[18] Sharma, K. and D. Garg, Complexity analysis in heterogeneous system. *Computer and Information Science*, (2009). 2(1): P48.
[19] Alizadeh, H., M.-B. Behrouz, and H. Parvin, To improve the quality of cluster ensembles by selecting a subset of base clusters. *Journal of Experimental & Theoretical Artificial Intelligence*, (2014). 26(1): 127-150.
[20] Ghaemi, R., et al., A survey: Clustering ensembles techniques. *World Academy of Science, Engineering and Technology*, (2009)