Experimental Approach for the Evaluation of Neural Network Classifier Algorithms

Masoud Ghaffari and Ernest L. Hall

Center for Robotics Research
University of Cincinnati
Cincinnati, Oh 45221-0072

ABSTRACT

The purpose of this paper is to demonstrate a new benchmark for comparing the rate of convergence in neural network classification algorithms. The benchmark produces datasets with controllable complexity that can be used to test an algorithm. The dataset generator uses the concept of random numbers and linear normalization to generate the data. In a case of a one-layer perceptron, the output datasets are sensitive to weight or bias of the perceptron. A Matlab^{TM} implemented algorithm analyzed the sample datasets and the benchmark results. The results demonstrate that the convergence time varies based on some selected specifications of the generated dataset. This benchmark and the generated datasets can be used by researchers that work on neural network algorithms and are looking for a straightforward and flexible dataset to examine and evaluate the efficiency of neural network classification algorithms.

Keywords: Classification, Benchmark, Algorithm learning, Artificial Neural Network, Robotics

1. INTRODUCTION

Artificial neural networks (ANN) have been an active area of research in the last decade and many achievements have been developed. Center for Robotics Research at the University of Cincinnati has applied ANNs in robotics and several progresses have been achieved. After any development in the techniques and algorithms, the question arises about operation in an application. Any improvement in neural network (NN) algorithms needs a test bed type problem to show the results. Some authors have used real data fromtheir applied problems and others have used the well-known problems in the NN literature.

For example, Ng, et al. (1999) developed a fast convergent generalized back-propagation algorithm.[1] They conducted a number of experiments on three different problems, including XOR, 3-bit parity and the 5-bit counting problems to illustrate the various aspects of the new algorithm. The network of XOR problem consists of two input nodes, two hidden nodes and one output node. In the 3-bit parity problem, the network consists of three input nodes, two hidden nodes and one output node.

Another popular benchmark problem is the two-spiral problem. The two-dimensional (2D) spiral data set was proposed by Alexis Wieland of MITRE Corporation and now forms one the important benchmarks at the Carnegie Mellon repository.[2] The two-spiral problem is often used as a test for comparing the quality of different supervised learning algorithms and architectures.[3] Many authors now include it in the benchmarks for speed and quality of learning for new algorithms and architecture types.[4, 5]

There are many common datasets described in the ANN literature that have been used as the applications of techniques. Some examples of those datasets include the IRIS, Fishers Iris Set[6], Cleveland Heart Diseases data[7] , IMOX and 80X, hand printed character sets[8] , Congressional voting dataset (UC-Irvin), Churn dataset[9], BLOOD published by the American Statistical Association[10], Sonar[11], Glass, a collection of glass fragments[12], Tremor, Parkinson's disease data[13] , and Ionosphere, radar data.[14]

Studies show that for almost any ANN a dataset can be constructed for which it solves it well.[15] The performance of a network is based on the class distribution and sample size and therefore on the application. Thus, an application domain has to be defined. The common way to do this is by selecting a collection of datasets.

This method of performance evaluation has some pitfalls. A collection of datasets may show the diversity but does not show the weight of a particular dataset in the overall performance. In addition, some classifiers have many user-adjustable parameters such as step sizes, momentum terms, weights and stopping procedures and therefore the results are user dependent, which makes the performance comparisons more difficult because different researchers may get different results for the same problem. The next problem of traditional datasets is the need for the training data and limitation of sample size. Generally, datasets are divided into three parts training, tuning and testing sections. The common pitfall of this procedure is that most researchers tend to adjust their algorithms after the testing, with the result that they may be using from the testing data, for the training and therefore the results are biased.[15]

Since many researchers and students are developing new methods, a standard benchmark dataset is a necessity. The benchmark should be large enough and it should include a collection of diverse problems. In addition, it should be renewed from time to time. There have been some attempts to build such a standard benchmark.[15, 16, 17] Also a workshop about NN benchmarking at NIPS*95 (Neural Information Processing Systems) addressed some issues regarding a standard benchmark.[18]

The purpose of this paper is to present a new benchmarking dataset that can be used for the test of algorithms associated with neural networks. The proposed data generator will provide a flexible dataset without the limitation of sample size. It also may provide a good tool for parameter adjustment of the network. By choosing different data each time, a specific behavior of a neural network can be studied. This benchmark is not a replacement for real application datasets, rather, it is a complementary benchmark that can be used in standard benchmark datasets.

2. THE BENCHMARK

The main idea of this benchmark is based on generating random numbers from the normal distribution with different complexities for different purposes. What will be described in this paper is a special case of the benchmark data in two-dimensional space. The idea can be extended to higher dimensions.

The benchmark contains two types of datasets, Benchmark 1, shown in Fig. 1 shows the idea of a two-dimensional set of the benchmark data. The sample data has a random distribution in two sectors with the radius of one. Based on a, the angle between two lines of Fig. 1, two classes of data can be separated by a linear perceptron with different complexities. This benchmark is sensitive to weight parameter estimation of the perceptron.

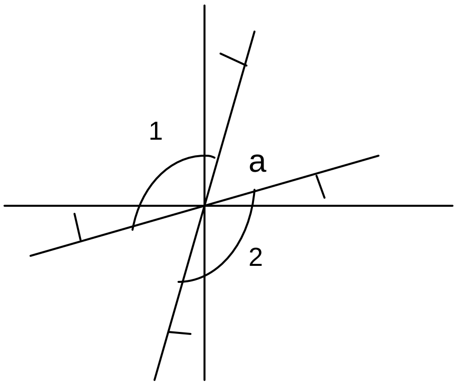


Figure 1. Random data in two sectors

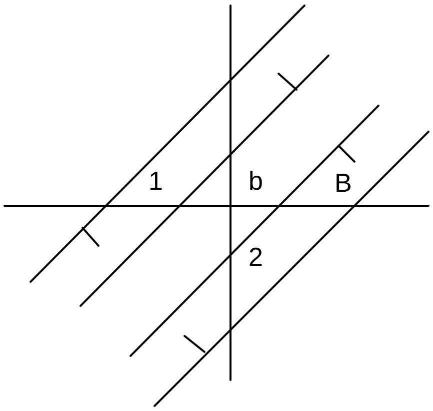


Figure 2. Random data in two strips

Benchmark 2, shown in Fig. 2, represents two classes of random data between two parallel strips. When these two classes are far from each other, separation is easier. By changing the b, different data can be obtained. This benchmark is sensitive to estimation of bias in the perceptron.

Research hypothesis states that ‘when the angle of a and bias b decrease, the data set is more complex and execution time for classification algorithms increase’. If the research hypothesis is proved, it means the benchmark has the ability to generate different standardized datasets for benchmarking and comparing the rate of convergence, in classification algorithms.

The following is an example for benchmark 1:

In the polar coordinate system:

Class 1	*Class2*	
$r = Rand(0,1)$	$r = Rand(0,1)$	
$q = p/4 + a/2 + (p - a) * Rand(0,1)$	$q = p/4 - a/2 + (-p + a) * Rand(0,1)$	(1)

and in the cartesian coordinate system:

Class 1	*Class2*	
$x = r\sin(q)$	$x = r\sin(q)$	
$y = r\cos(q)$	$y = r\cos(q)$	(2)

Each time a new random number for ? and ? should be used to generate independent (iid) x and y in any class. The above formulas generate m two-dimensional random sample datasets in two classes.

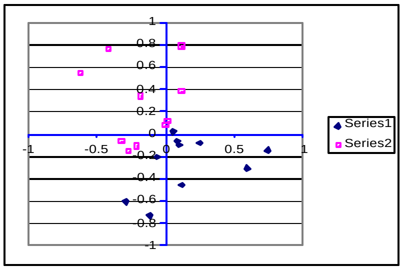
The following example is performed for benchmark 2:

Class1	*Class 2*	
$x - y = b + \sqrt{2} * Rand(0,1)$	$x - y = -b - \sqrt{2} * Rand(0,1)$	
$x = -B + 2B * Rand(0,1)$	$x = -B + 2B * Rand(0,1)$	(3)

This formula generates different samples, by changing b for a specific B, where B is the width of strip. These sets of data are sensitive to bias and can be used in comparing the learning speed in different algorithms.

3. RESULTS

A Microsoft Excel^{TM} program produced several data sets for benchmark 1 and benchmark 2 based on a and b. Figure 3 shows a sample data set for a=30 and Fig. 4 represent the scatter plot of a sample data for $b = 8$.



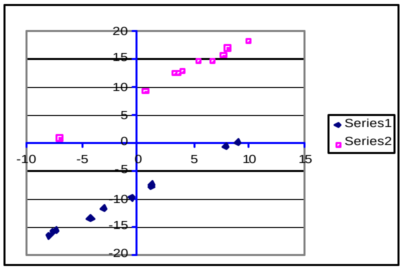


Figure 3. Data set scatter plot for a=30   Figure 4. Data set scatter plot for b=8

These data were used in a MATLAB program and execution time for any dataset was measured. Figures 5 and 6 demonstrate the execution time verses angle and bias, which are two parameters to control the complexity of the datasets. Some researches chose the number of iterations as an indicator for the learning speed of algorithm; however; in this example, the execution time has been chosen as it is a better indicator for computational work.[19] Table 1 and Table 2 demonstrate the results of experiments.

Bias (b)	0.1	5	10	40
Execution time	4.67s	2.14s	2.14s	1.914s

Table 1. Execution time for different bias

Slope (a)(deg)	2.8	10	45	90
Execution time	4.939s	3.127s	2.394s	2.104s

Table 2. Execution time for different slopes

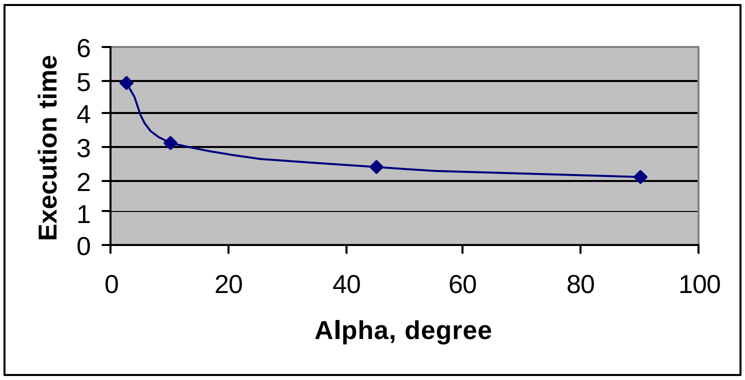


Figure 5. Execution time verses angle

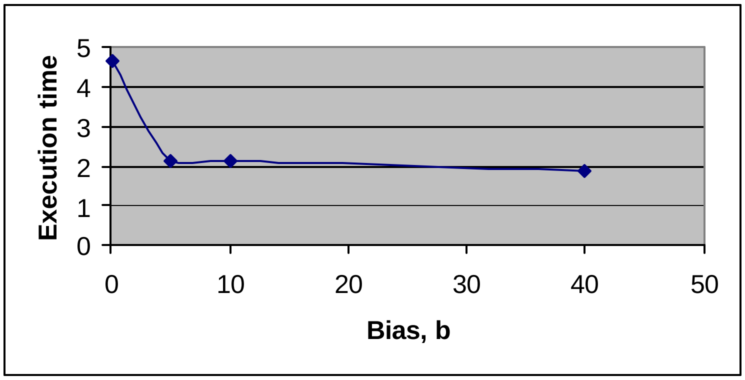


Figure 6. Execution time verses bias

The results confirm the hypothesis of the research. The benchmark is able to generate different data sets with various complexities. Figures 7 and 8 show two samples output of the MATLAB program. The following is the MATLAB code:

```
% time1 measures the start time of program
time1=clock;
```

```
% alpha=30
P=[0.694 0.986 0.028 0.748 -0.074 0.415 0.035 0.663 -0.086 -0.042 -0.399 0.229 -0.259 -0.001 0.037 -0.208 -0.580
0.035 -0.232 -0.014 ; -0.554 0.051 -0.063 0.237 -0.249 0.147 -0.082 -0.158 -0.630 -0.992  -0.090 0.853 0.914 0.064
0.226 -0.111 0.307 0.135 -0.016 .013];
T=[ 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 ; 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1];
plotpv(P,T);
net=newp([-1 1; -1 1], 2);
plotpv(P,T);
linehandle=plotpc(net.iw{1,1}, net.b{1});
E=1;
while(sse(E))
  [net, Y, E]=adapt(net,P,T);
  linehandle=plotpc(net.iw{1,1}, net.b{1}, linehandle);drawnow;
end;
% etime calculates the elapsing time or speed of algorithem
etime(clock,time1)
```

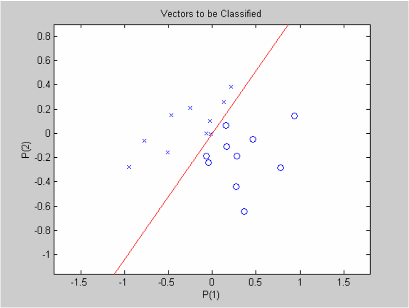


Figure 7. MATLAB output for a=2.8

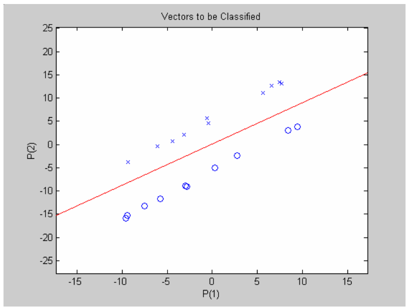


Figure 8. MATLAB output for $b = 5$

4. GENERALIZED BENCHMARK

The demonstrated experiment used the following assumptions:

- Two-dimensional space
- A slope of 45° and $B = 10$ (width of strips) for the benchmark 2
- Random numbers in Cartesian coordinates
- One perceptron classification
- Linear classification
- Ten samples for each class.

Relaxing the assumptions provide a more generalized benchmark. Generalizing from two-dimensional space to m dimensions easily is possible. Also for slope and B different numbers can be chosen. The interesting generalization is from Cartesian coordinate to polar format. Figure 9 shows the 2D spiral problem in Cartesian coordinates and Fig. 10 represents the radius in function of angle for the 2D spiral problem. The points of a spiral obey the equation $r = r(q + 2pn) + r_0$. Figure. 10 demonstrates that a 2D problem can be resulted from benchmark 2.

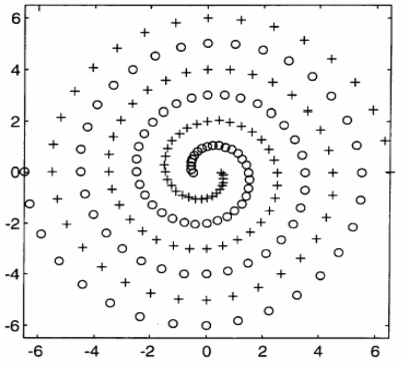


Figure 9. Cartesian plane representation of 2D

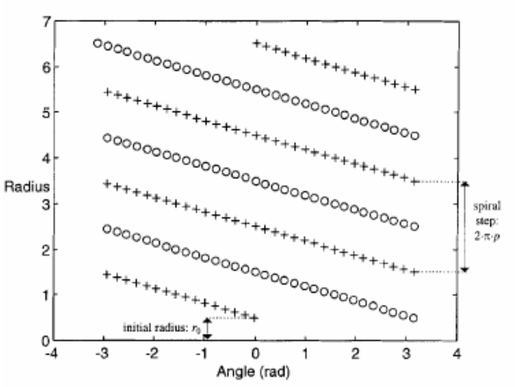


Figure 10. Representation of the radius in function of the angle for 2D
(From Alvarez-Sanchez, 1999)

Relaxing the assumption of one perceptron is also possible. Figure 10 illustrates the idea. In the current format $G_1[a,n,m]$ and $G_2[b,n,m]$ represent the benchmark when a=angle, b =bias, $n =$ dimension and $m =$ number of samples.

5. CONCLUSION

In this paper, a benchmark problem for the neural network classification algorithms was presented. The results demonstrate the flexibility and capability of the bearcat benchmark in generating the variety of data sets with different complexities for comparing the learning speed of algorithms. Using these two benchmarks can also be useful in estimating the initial weight factors in the algorithms. In addition, the generalized benchmark can stimulate ideas for further research in this field. The output of these researches can be installed in the University's website as a data repository.

REFERENCES

1. S.C. Ng,, S.H. Leung, A. Luk, "Fast Convergent Generalized Back-Propagation Algorithm with Constant Learning Rate," *Neural Processing Letters*, 9,1, pp. 13-23, 1999.
2. S. Singh, , "2D spiral pattern recognition with possibilistic measures," *Pattern Recognition Letters*, Vol. 19, pp. 141-147, 1998.
3. J. R Alvarez-Sanchez, , "Injecting Knowledge into the Solution of the Two-Spiral Problem," *Neural Comput & Applic*, Vol. 8, pp. 265-272, 1999.
4. M. Riedmiller, H. Braun, "A direct adaptive method for faster backpropagation learning: the RPROP algorithm," Proceeding, *IEEE International Conference on Neural Networks*, San Francisco, 1993.
5. N. K. Treadgold, TD Geden, "A cascade network algorithm employing progressive RPROP," *Biological and Artificial Computation*, Vol. 1240, 1997.
6. R.A., Fisher, "The use of multiple measurements in taxonomic problems," *Ann Eugenics*, Vol. 7, pp. 280-322, 1936.
7. P. Murphy, D. Aha, UCI repository of machine learning databases, Technical report, University of California, Irvin, 1994.
8. A.K. Jain, M. D. Ramaswami, "Classifier design with Parzen windows in: E.S. Gelsema nad L.N. Kanal, Eds.," *Pattern Recognition and artificial intelligence*, Amsterdam, pp. 211-228, 1988.
9. R. Feraud, R. Clerot, "A methodology to explain neural network classification," *Neural Networks*, 15, pp. 237-246,2002.

10. L.H.Cox, et al., "Exposition of statistical graphing technology," *ASA Proc. Statistical Computation Section*, pp. 55-56, 1982.

11. Gorman, T. J. Sejnowski, , "Learned classification of sonar targets using massively parallel network," *IEEE Tran. Acous. Speech Signal Process.* , 36, 7, pp 1135-1140, 1988.

12. B.D. Riply, "Neural networks and related methods for classification," *J. Roy. Statist. Soc. B*, 56, 3, pp. 409-456, 1994.

13. J. Spyers-Ashby, "The recording and analysis of tremor in neurological disorders," PhD dissertation, Imperial College, London, 1996.

14. V. Sigillito, et al., 1989, "Classification of radar returns from ionosphere using neural networks," *John Hopkins APL Technical Digest*, 10, pp. 262-266, 1989.

15. R. Duin, "A note on comparing classifiers," *Pattern Recognition Letters*, Vol. 17, pp. 529-536, 1996.

16. D. Michie, et al., *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, New York. 1994

17. L. Prechelt, "A study of experimental evaluations of neural network learning algorithms: current research practice," Technical report, 19/94, 1994.

18. http://www-2.cs.cmu.edu/Groups/NIPS/NIPS95/Papers.html

19. G. Auda, M. Kamel, "Modular Neural Network Classifiers: A Comparative Study," *Journal of Intelligent and Robotic Systems,* Vol. 21, Issue 2, pp. 117-129, 1998.