Diagnosing Breast Cancer with Neural Network & Optimising with Resilient Backpropagation and Genetic Algorithm

Johnny Yang, Research School of Computer Science, Australian National University

Abstract. The purpose of this study was to use neural network to diagnose breast cancer based on 30 attributes. The study explored the limitations of standard backpropagation algorithm and attempted to improve the model using the resilient backpropagation (RPROP) algorithm. The optimised model with RPROP outperformed the baseline with standard backpropagation in terms of both training time and accuracy. The effect of learning rate on the two algorithms was also explored. An attempt was made to further optimise the RPROP model with genetic algorithm, but it did not lead to an improvement in performance.

Keywords: Breast Cancer Diagnosis; Neural Network; Supervised Learning; Limits of Backpropagation; Resilient Backpropagation (RPROP); Learning Rate, Genetic Algorithm.

1 Introduction

1.1 Background

Caused by abnormal growth of cells lining the breast ducts, breast cancer is the most common cancer for women worldwide [1] and the second most common cancer to cause death for Australian women [2]. The abnormal cell growth has the potential to spread to other parts of the body if uncontrolled. Fortunately, breast cancer can be correctly diagnosed through several mechanisms, and the survival rate is over 90% if diagnosed before abnormal cell growth spread to other parts of the body [2]. The purpose of this study is to diagnose breast cancer with neural network.

1.2 Dataset Selection

The chosen dataset was created and donated by researchers at the University of Wisconsin in November 1995 [3]. The dataset had appeared in several medical literatures and been used multiple times in the past to train a network for diagnosing breast cancer [3].

The dataset contains 569 instances and 32 features. Each instance is a digitized image of a fine needle aspirate (FNA) of a breast mass, which describes characteristics of the cell nuclei in the image. The first feature contains the ID of the instance, while the second feature is the label ('B' for 'Benign' and 'M' for 'Malignant'). The remaining 30 features are composed of the (1) mean, (2) standard error, and (3) "worst" (mean of the three largest values) of the following ten characteristics of each cell nucleus:

Table 1. Attributes of the dataset: (1) mean, (2) standard error, and (3) "worst" of the following 10 real-valued characteristics of each cell nucleus[3].

Characteristic	Notes
(a) Radius	
(b) Texture	Standard deviation of grey-scale values
(c) Perimeter	
(d) Area	
(e) Smoothness	Local variation in radius lengths
(f) Compactness	(perimeter^2 / area – 1.0)
(g) Concavity	Severity of concave portions of the contour
(h) Concave points	Number of concave portions of the contour
(i) Symmetry	
(j) Fractal dimension	"coastline approximation" - 1

There are several reasons why this particular dataset was chosen. First and foremost, several studies have been able to use this dataset to train their model to diagnose breast cancer with high accuracy [4] [5], suggesting the attributes chosen by the creators of this dataset are likely highly correlated with breast cancer. Second, the dataset contains no missing data [3]. Third, with 357 of the instances belonging to one class and the remaining 212 instances belonging to

the other, both classes of the dataset are fairly well represented. This is an important characteristic and can not be said for all datasets available on UCI Machine Learning Repository [6]. For example, in the cardiac arrhythmia dataset [7], one class makes up about 54% of the instances, while the remaining 15 classes make up the rest of the dataset. In fact, nine of the sixteen classes each makes up less than 3% of the dataset as outlined in **Table 2**. When the dataset is dominated by one class and the remaining classes are negligible, it is easy for the network to overfit and simply classify instances to the dominant class [8]. Different from the arrhythmia dataset, the breast cancer dataset [3] is fairly well distributed, minimizing the potential harm of this problem. For these reasons, the breast cancer dataset [3] donated by researchers at the University of Wisconsin was chosen to train the network for breast cancer diagnosis.

Table 2. Class distribution of the cardiac arrhythmia dataset [7].

Class Code	Number of Instances (& percentage)
01	245 (54.20%)
02	44 (9.73%)
03	15 (3.32%)
04	15 (3.32%)
05	**13 (2.88%)**
06	25 (5.53%)
07	**3 (0.66%)**
08	**2 (0.44%)**
09	**9 (1.99%)**
10	50 (11.06%)
11	**0 (0%)**
12	**0 (0%)**
13	**0 (0%)**
14	**4 (0.88%)**
15	**5 (1.11%)**
16	22 (4.87%)

1.3 Supervised Learning with Artificial Neural Network

The study utilised a supervised learning with a neural network made up of several hidden layers of artificial neurons and weights associated with connections in the network. The objective of the network was to incrementally adjust its weights through epochs of training to minimise loss and map inputs to the correct class. One common approach for this optimisation is the backpropagation algorithm with gradient descent.

The underlying principle of supervised learning with artificial neural network is to repeatedly apply the chain rule to calculate the partial derivative for each weight in the network; once the partial derivatives are determined, the corresponding weight updates are determined using the backpropagation algorithm [9]. In the standard backpropagation algorithm, the weight update is calculated as the product of the negative derivative and a constant value known as the learning rate to produce a scaled move in the opposite direction of the gradient [9].

1.4 Limitations of Backpropagation

Though popular and relatively straightforward, the backpropagation algorithm has its limits. First, the performance of the algorithm is largely dependent on the choice of the learning rate. A learning rate that is too small will require numerous epochs to reach an acceptable accuracy, while a learning rate that is too large can lead to oscillation thereby preventing the error to fall below an acceptable level [10]. Second, besides the learning rate, the performance of backpropagation depends on the partial derivative [10]. Large derivates translate to large weight steps, which can potentially take the algorithm to a entirely different region of weight space [10]. Although a momentum parameter was proposed to scale the influence of weight steps, it has been documented that it "is not a general technique for gaining stability or speeding up convergence" as comparable and sometimes even better results can be obtained without the momentum term [9].

1.5 Resilient Backpropagation (RPROP)

In response to backpropagation's limitations, Riedmiller and Braun proposed resilient backpropagation (RPROP). Different from standard backpropagation, RPROP determines the size of the weight update without considering the size of the partial derivative [10] to avoid the problems addressed in the previous paragraph. Under RPROP, each weight between neurons i and j has its own update value Δ_{ij}, which is determined during training based on the error function E

based on the following: Whenever the partial derivative of weight w_{ij} switches sign, indicating the last update was too big and a local minimum was jumped over, update value Δ_{ij} gets decreased by a constant factor η^- [10]. On the other hand, when the derivative retains its sign, update value Δ_{ij} gets increased by the a constant factor η^+ to expedite training [10]. After the update value for each weight is adjusted, the weight update follows:

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ +\Delta_{ij}^{(t)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ 0 & , \text{ else} \end{cases}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \quad [10]$$

with the exception that when the partial derivative switches sign—when the previous step was too big and thereby missed the minimum—the previous weight update is reverted:

$$\Delta w_{ij}^{(t)} = -\Delta w_{ij}^{(t-1)} \text{ , if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \quad [10]$$

In summary, the underlying principle of RPROP is the direct modification of the weight update values Δ_{ij}; with the introduction of these resilient update values that do not depend on the size of the partial derivatives, RPROP's performance is not dependent on unforeseeable gradient behaviour like the standard backpropagation algorithm [9].

This study utilised RPROP in an attempt to improve the neural network for diagnosing breast cancer.

1.6 Genetic Algorithm (GA)

Genetic algorithm was explored in attempt to find better hyperparameters for the neural network to further optimise the RPROP model. Similar efforts have been made in the past, and some researchers successfully improved the performance of their neural networks using genetic algorithms to find optimal hyperparameters [11].

2 Method

All the codes can be found in the attached zip file.

2.1 Data Pre-processing

The 'data.csv' file in the 'data' directory contains the original data downloaded from the UCI Machine Learning Repository [12]. The first column of the dataset, which contains the ID numbers of each instance, was not considered as IDs add unnecessary noise and are unlikely to be correlated with breast cancer. The second column, containing the labels, was transformed to numeric values by mapping 'B' (benign) to 0 and 'M' (malignant) to 1. The remaining 30 columns contain the attributes discussed previously in **Table 1**; these values were normalised to the range of zero to one and fed to the network for training. All of above pre-processing was done by the 'get_data' function (and the helper functions it called).

2.2 Neural Network Architecture & Performance Measure

The study employed a standard multi-layer artificial neural network. The baseline model and the RPROP model used the same network architecture; the only difference between them was their optimiser (standard backpropagation vs RPROP). The network architecture for both the baseline and the RPROP model is summarised in **Table 3** below.

Table 3. Network architecture (hyperparameters) for the baseline and RPROP models.

	Baseline Model	RPROP Model
# of hidden layers	5	5
Learning rate	0.05	0.05
# of epochs	600	600
Optimiser	**Standard Backpropagation**	**RPROP**

For both models, the sigmoid function was used as the activation function at the hidden layers, whereas the softmax function was used at the output layer. 600 epochs were used for training, unless the loss reached zero beforehand, in

which case the training would stop early. 10-fold cross validation was used to evaluate the two models. All the code can be found in 'baseline.py' and 'rprop.py' for the two corresponding models.

Additionally, to examine the effectiveness of RPROP, an experiment was set up to visualise how its performance would change in response to changes in the learning rate. The code for this experiment can be found in 'learning_rate.py', which plots the effects of learning rate on the performances of the two models.

To see any of the code files mentioned above in action, simply run 'python [filename]' in the unzipped directory. (For example, use 'python learning_rate.py' to visualise the experiment.)

As an attempt to further improve the RPROP model, genetic algorithm (GA) was applied to find the best number of hidden neurons at each of the hidden layer. The GA model used a population size of 30, each of which was initialised with random numbers of hidden neurons. 500 hundred generations were applied. The crossover rate was 0.8, while the mutation rate was 0.01. The fitness of each individual in the population was assessed by how well the RPROP model can correctly diagnose breast cancer on the dataset using the number of hidden neurons specified by the individual. Crossover was defined as taking the average of the both parents, i.e. an individual with the DNA [500, 400, 300, 200, 100] (500 hidden neurons in the first hidden layer, 400 hidden neurons in the second hidden layer, etc) and an individual with the DNA [400, 300, 200, 100, 50] will produce an offspring with the DNA [450, 350, 250, 150, 75]. Mutation was defined as replacing all five slots of the DNA with random numbers (shapes were assured to match). After the most fitted DNA (number of hidden neurons at each layer) was determined after 500 generations, 10-fold cross validation was used to evaluate a RPROP model using that many hidden neurons with all else being equal (same learning rate, same number of epochs, etc). This GA model is defined in 'ga.py' and can be run using 'python ga.py'.

3 Results and Discussion

The result are summarised in **Table 4** and **Table 5** below.

Table 4. Accuracy of the baseline model, the RPROP model, and the RPROP with GA model.

	Baseline	**RPROP**	**RPROP with GA**
	Accuracy		
1st fold	57.14%	94.64%	69.64%
2nd fold	69.64%	92.86%	91.07%
3rd fold	60.71%	64.29%	82.14%
4th fold	62.5%	96.43%	91.07%
5th fold	76.79%	94.64%	96.43%
6th fold	76.78%	96.43%	91.07%
7th fold	73.21%	71.43%	91.07%
8th fold	69.64%	94.64%	89.29%
9th fold	73.21%	89.29%	87.5%
10th fold	66.07%	96.43%	80.36%
Average	**68.57%**	**89.11%**	**86.96%**

Table 5. Time took for each model. (Recall that training would not use all epochs if loss reached zero beforehand)

	Baseline	**RPROP**	**RPROP with GA**
	(Time, # of epochs took)		
1st fold	7.35s, 600 epochs	4.23s, 189 epochs	Getting through the 500 generations took **more than 10 hours**.
2nd fold	7.30s, 600 epochs	4.94s, 186 epochs	
3rd fold	7.84s, 600 epochs	10.31s, 600 epochs	
4th fold	8.02s, 600 epochs	3.09s, 112 epochs	
5th fold	7.72s, 600 epochs	3.19s, 156 epochs	
6th fold	7.60s, 600 epochs	2.87s, 135 epochs	
7th fold	7.35s, 600 epochs	10.69s, 600 epochs	
8th fold	7.53s, 600 epochs	6.88s, 207 epochs	
9th fold	7.58s, 600 epochs	10.58s, 173 epochs	
10th fold	7.75s, 600 epochs	5.65s, 217 epochs	
Average	**7.62s, 600 epochs** Took less than 2 minutes total (training & evaluating) for the entire execution.	**6.24s, 257.5 epochs** Took less than 2 minutes total (training & evaluating) for the entire execution.	

As shown above, the RPROP model outperformed the baseline model in terms of both accuracy and training time. The accuracy for the RPROP model was about 20% higher than that of the baseline model, while taking less than half of the epochs on average. The genetic algorithm with 500 generations took more than 10 hours to complete, which was significantly longer than the other two models. Unfortunately, the performance of the RPROP with GA model did not surpass that of the RPROP model. The RPROP with GA model might have been able to performed better if initialised

with a greater population and used more generations. However, doing so would also add to the time required, which was already greater than 300 times more than the time required for the baseline or the RPROP model.

Although the 89.11% accuracy might not be as impressive as the model trained by researchers at the University of Wisconsin that reached "100% chronological correctness in diagnosing 131 subsequent patients" [4], the RPROP model was a significant upgrade over the baseline model in terms of both training time and accuracy. With the same network architecture, the RPROP model outperformed the baseline model with standard backpropagation. This was likely due to the fact that RPROP's performance is generally less dependent on the size of the partial derivatives as discussed in the 'Introduction' section. This result is consistent with other studies in the field which also found RPROP to outperform standard backpropagation [13] [14].

Furthermore, RPROP's performance did not seem to depend much on the choice of learning rate like standard backpropagation as suggested by the result of the learning rate experiment (summarised in **Fig. 1**). RPROP was able to achieve an accuracy above 90% with low learning rate when standard backpropagation got about 50% accuracy. This was possible because of RPROP's adaptive weight update as explained in the 'Introduction' section.

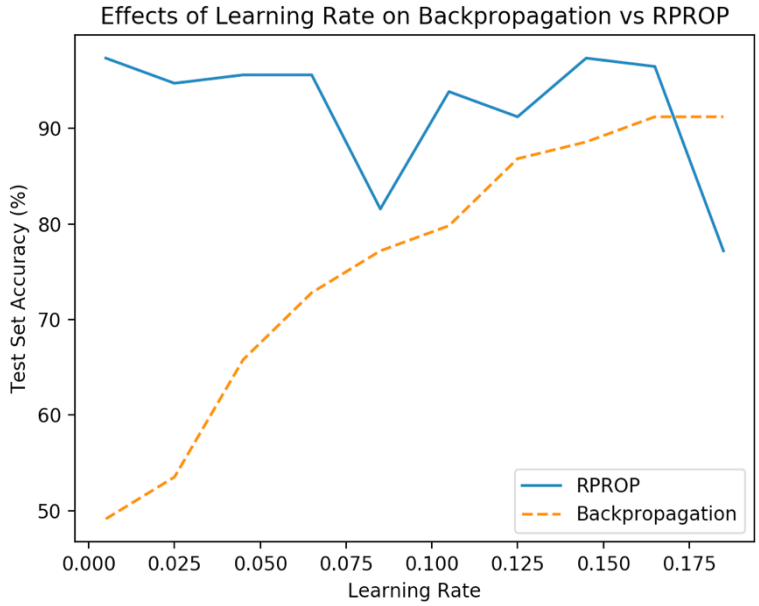

Fig. 1. Result of the learning rate experiment. Different from standard backpropagation whose performance depends largely on the choice of the learning rate, RPROP managed to maintain a relatively high level of accuracy on the test set for most learning rates. This graph can be generated by running 'python learning_rate.py' in the unzipped directory.

Overall, the RPROP model outperformed the baseline model with standard backpropagation in terms of both training time and accuracy as due to its adaptive nature.

4 Conclusion and Future Work

In the end, the neural network was able to cut the number training epochs required by more than half and increase its accuracy by about 20% with the adoption of RPROP. This, however, does not suggest RPROP is perfect and should be applied to every network. Like all methods, RPROP has its limitations. This was evident in the far right of the **Fig. 1** above which showed that at some learning rates, the standard backpropagation outperformed RPROP. As pointed out by other researches already, RPROP's "restricted local adaptation scheme inherently lacks the overall view that global techniques may have" and therefore can underperform compared to standard backpropagation and other methods in certain circumstances [9]. Hence, it remains to be seen how RPROP can be improved in future research.

With better computational power, the GA model can be experimented further more easily. Increasing the number of population size and the number of generations both are likely to contribute to a better performance. Other hyperparameters, such as the number of hidden layers or the learning rate, can also be explored with GA.

As to improving the accuracy of diagnosing breast cancer, several directions other than GA can be considered for future work. One approach will be improving through data. Improvements may be possible through rescaling data, getting more data, or better feature selection. Another approach will be applying algorithms other than backpropagation and RPROP to see if improvements can be made. Finally, the other approach is to adjust the network architecture by tuning the number of hidden layers, the number of neurons, the learning rate, activation functions, etc. Overall, it would be great if one can find an algorithm that addresses the limitations of RPROP as such an algorithm can be applied to all datasets as opposed to the other two approaches whose effect may be limited to this particular dataset.

References

1. Breast cancer statistics. In: World Cancer Research Fund International. https://www.wcrf.org/int/cancer-facts-figures/data-specific-cancers/breast-cancer-statistics. Accessed 25 Apr 2018
2. Australia CC (2018) Breast cancer. In: Cancer Council Australia. https://www.cancer.org.au/about-cancer/types-of-cancer/breast-cancer/. Accessed 25 Apr 2018
3. Wolberg WH, Street N, Mangasarian O (1995)In: UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.names. Accessed 14 Apr 2018
4. Mangasarian OL, Street WN, Wolberg WH (1995) Breast Cancer Diagnosis and Prognosis Via Linear Programming. Operations Research 43:570–577. doi: 10.1287/opre.43.4.570
5. Street WN, Wolberg WH, Mangasarian OL (1993) . Biomedical Image Processing and Biomedical Visualization. doi: 10.1117/12.148698
6. In: UCI Machine Learning Repository: Data Sets. http://archive.ics.uci.edu/ml/datasets.html. Accessed 14 Apr 2018
7. Guvenir H, Acar B, Muderrisoglu H (1998)In: UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/machine-learning-databases/arrhythmia/arrhythmia.names. Accessed 14 Apr 2018
8. Japkowicz N, Stephen S (2002) The class imbalance problem: A systematic study. Intelligent Data Analysis 6:429–449.
9. Riedmiller M (1994) Advanced supervised learning in multi-layer perceptrons — From backpropagation to adaptive learning algorithms. Computer Standards & Interfaces 16:265–278. doi: 10.1016/0920-5489(94)90017-5
10. Riedmiller M, Braun H A direct adaptive method for faster backpropagation learning: the RPROP algorithm. IEEE International Conference on Neural Networks. doi: 10.1109/icnn.1993.298623
11. Bashiri M, Geranmayeh AF (2011) Tuning the parameters of an artificial neural network using central composite design and genetic algorithm. Scientia Iranica 18:1600–1608. doi: 10.1016/j.scient.2011.08.031
12. Wolberg WH, Street N, Mangasarian OLIn: archive.ics.uci.edu. http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data. Accessed 14 Apr 2018
13. Kişi Ö, Encuoğlu E (2005) Comparison of three back-propagation training algorithms for two case studies. Indian Journal of Engineering & Materials Sciences 12:434–442.
14. Magoulas GD, Vrahatis MN, Androulakis GS (1999) Improving the Convergence of the Backpropagation Algorithm Using Learning Rate Adaptation Methods. Neural Computation 11:1769–1796. doi: 10.1162/089976699300016223