A Comparison between Bimodal Distribution Removal Algorithm and Genetic Algorithm feature selection

Ruiqi Zhou

Research School of Computer Science, Australian National University
ACT, Australia

U5909724@anu.edu.au

Abstract. Bimodal Distribution Removal (BDR) algorithm has been proposed to be an effective method to eliminate noises in data set and optimize the training result of neural network, while Genetic Algorithm (GA) is another method which can be used to select features from training sets. This paper uses a neural network model to perform a vehicle classification problem and examines the effect of the above two enhancement techniques. The result is evaluated by accuracy and loss. Compared to other techniques such as constructive neural network (CNN) learning algorithm (77.5 ± 6.2), the performances of BDR (71.28) and GA feature selection (76.59 ± 3.2) on vehicle data set are worse on Vehicle classification problem, based on the experiments in this paper, but GA tends to perform better than BDR.

Keywords: Neural network, bimodal distribution removal, classification, genetic algorithm, feature selection

1 Introduction

1.1 Background of the problem

Neural network combined with statistical methods has been utilized for performing classification problems [1]. This paper aims to examine how to train a network and can apply such a model to a noisy data set and see the performance of training result and BDR. This paper will use Statlog (Vehicle Silhouettes) Data Set [2] as an example. Basically, this data set is made up of 946 instances of vehicle silhouettes. These silhouettes are described by 18 different features and are classified into 4 types of vehicles, which are OPEL, SAAB, BUS, VAN. The goal is to use a trained neural network to classify the 3D objects using 2D features.

According to the website record, this data set has been tested for many neural network algorithms and methods, which can provide many evaluation results for BDR and GA to compare with. Furthermore, there are also some noises in the data set, which makes it nonlinear separable [3]. This implies that some preprocessing such as BDR or GA feature selection on data set tend to be effective for improving the result of model training.

1.2 Investigation outline

Based on the vehicle data, this paper will firstly discuss the method to pre-process the data set. After the preprocessing, the data will be randomly divided into training set and data set. Secondly, we will discuss the techniques used to design the network model and a way of evaluating the accuracy of pattern classification. Thirdly, the above techniques will be utilized to train the model and generate a result. Then, the BDR will be applied onto the training set and we can see how this algorithm can improve the performance of pattern classification. For comparison, GA feature selection will also be applied to the data set as another enhancement of data processing. Furthermore, this paper will discuss other researcher's work on the same data set, examine what techniques they have used and consider their results as a reference. Finally, we will summarize on the findings and implications for the future work.

2 Method

2.1 Pre-processing

The last column of the data set is the target of each pattern, so the string value of this column will be replaced with numbers that represent classes. The data set contains 946 instances in total. Each instance is described by 18 features, and there are 4 types of instance. Firstly, randomly split the instances into train set (around 850 instances) and test set (around 100 instances). Then, in case there has been some data preprocessing on each vector of features [4], this paper uses the following formula to normalize the value of each feature in this data set:

$$result_i = \frac{(value_i - m_i)}{sd_i} \ (\mathrm{i} \in \{1, \ldots, \mathrm{N}\}) \quad (1)$$

N is the number of features, m_i represents the mean value of column i and sd_i means the standard deviation of column i. This formula enables each feature has a zero-mean and unit-variance and makes the distribution of each feature similar. After the pre-processing, randomly split the data set into train set and test set.

2.2 Network Design

This network takes features as the input, and the classes of patterns as the output. Thus, there are 18 input neurons and 4 output neurons. Besides, the structure of hidden neurons needs to be simple to ensure that GA is efficient enough. Based on the research from T.D. Gedeon & D. Harris [10], the middle layer is set to 36 hidden neurons, which is the doubled number of input neurons. After several iterations of experiments, the 36 hidden neurons' middle layer reflects a decent performance.

Since the network only has one hidden layer, there will be only little information loss among each layer. Thus, the choice of activation function between hidden layer and output is the sigmoid function. As for the prediction of each epoch, use Cross Entropy (CE) Loss to evaluate the predictions of model and as the error for back propagation. The CE loss will be discussed in section 2.4. The optimizer of this network is Stochastic Gradient Descent (SGD). The reason for choosing SGD is that it randomly selects an observation in each iteration and updates the weight more frequently. As a result, it performs much fewer computations per iteration and is more capable of dealing with noises [5], which makes it converge faster. Also, set up a momentum to 0.5, such that the SGD can jump out from local minimum. This decision is based on several iterations of experiment. The result of accuracy on test sets implies that a momentum with value 0.5 helps to accelerate the process of reaching a global optimum.

In general, a network combining with back propagation tends to return a model with high accuracy [9]. In this case, we update the weights between different layers by tracing error backward to find the most error neurons. Followed by the step of back propagation, performing zero grad to clear the gradients so that the back propagation does not accumulate the former gradients.

2.3 Evaluation method

This paper uses generalization accuracy of training set and test set to evaluate the performance of network and BDR algorithm. This is because there is another paper that also tests its network on the vehicle data set and gives a result by showing a prediction accuracy [3]. Based on several experiments, we find that the rise of accuracy is positively related to the descend of CE loss. Besides, accuracy is another form of confusion matrix, which makes it an intuitional method for evaluating model [7]. The formula of accuracy is:

$$\text{accuracy} = \text{the number of correctly classified patterns / the number of patterns} \quad (2)$$

Some other techniques (e.g. network size) are not considered in this scenario, as the BDR algorithm does not change the structure of network. For every 50 epochs, the network predicts the class of patterns on both train set and test set, compares its prediction with the target and show the accuracy and loss.

2.4 Bimodal Distribution Removal

As mentioned in the above section, this dataset contains many noises, which will result in a fluctuation on the convergence of network training and enhance the probability of overfitting. BDR is an effective approach to minimize the effect of patterns with large error by permanently remove the noisy patterns from the data set [6]. It uses statistical method to divide training pattern into valid points and noisy points. To identify the noisy points in the training set, calculate the error of each pattern and plot the distribution of errors. The error of each pattern is evaluated by cross entropy error as follow [7][8]:

$$\mathrm{E} = -\sum_{n=1}^{N} \mathrm{y}_{\mathrm{target}} \log(\mathrm{y}_{\mathrm{pred}}) + (1 - \mathrm{y}_{\mathrm{target}}) \log(1 - \mathrm{y}_{\mathrm{pred}}) . \quad (3)$$

Having a way of calculating error for each pattern, a customized BDR can be implemented as [6]:

- Start training the network using the overall training patterns.
- Let the training last for several epochs, such that the error of network model decreases to an ideal level.
- Once the error is below an upper bound for applying BDR, calculate the average error M of all patterns.
- Get a subset of patterns with error > M, calculate the mean M1 and the standard deviation S of the subset.
- Remove those patterns with error > M1 + kS, where $k \in [0,1]$.
- Repeat the above steps, until the error of whole training set exceeds a lower bound for applying BDR.

As each removal step is followed by a certain number of epochs, BDR allows the network itself to identify outliers and makes the training faster [6]. In the context of vehicle data set, the customized BDR has some changes to the original BDR: an upper bound and a lower bound of the error are designed to limit the execution of BDR. Specifically, the BDR will be processed only when the error of model falls into a certain range. The reason is that the size of data set is relatively small. If the original BDR is applied, then after several iterations the training set becomes too small for training process. Therefore, the BDR process is only being processed in a certain period, and the termination condition is also different from the original BDR. The training process stops when the loss of evaluation on test set converges to a certain level. Moreover, to ensure that each execution of BDR does not remove too many noises, the k is set to 1.

In this paper, the process of BDR is designed as following: firstly, calculate the error of each pattern using CE loss function as mentioned. In general, each 50 epochs are followed by an execution of BDR until the variance decreases to a low constant (usually 0.01) [6]. However, as this data set tends to be noisy and the size of patterns is small, the convergence condition should be customized so that the training set does not become too small. Thus, the BDR process will not be called until the CE loss of overall training set jumps into the range of [0.8, 1.5]. The domain is supposed to be an intrinsic attribute of the vehicle data set and is determined by conducting many tests.

As mentioned above, the BDR will reduce the size of training patterns, which may be detrimental to small data sets. In addition, it performs much better on dealing with data set with a lot of noises. However, this also limits the range of its application on other kind of data sets.

2.5 Genetic Algorithm

Genetic algorithm offers an alternative method for solving feature subset selection problem. This paper will use a standard genetic algorithm to change the weights of features iteratively. According to the research of Chtioui, Y., Bertrand, D., & Barba, D, the process of GA feature selection is described as following [11]:

- Define the chromosomes of a certain problem;
- Choose a fitness function to optimize;
- Define the parameters for GA, this includes: population size, number of generation, probability of crossover, probability of mutation, probability of selection of the highest ranked individual;
- Generate an initial population of chromosomes. Commonly the chromosomes are binary representations, where 1 implies active and 0 implies inactive;
- Evaluate each chromosome by fitness value, and pick out better individuals as the chromosomes of next generation; Also process crossover and mutation;
- Repeat until terminate or the iteration meets the target number of generation.

Generally, the size of chromosome is determined by the number of features. Each feature is corresponding to a bit. If the value of this bit is greater than 1, then the feature is selected in this chromosome. Otherwise, it is not selected. In this paper, the chromosome is randomly assigned with the values ranges from 0 to 4. As a result, roughly 1/5 features will be dropped by each chromosome in the first generation.

The fitness function is the accuracy of the network model's performance on test set prediction. Each selected subset is used to train the network model and evaluate the fitness. According to Yang and Honavar's literature [12], to avoid extreme conditions (e.g. a high accuracy with a high loss), another fitness function will also be tested, which involves two evaluation criterions:

$$\text{fitness}(x) = accuracy(x) - \frac{loss(x)}{accuracy(x) + 1} + loss_{max} \quad (4)$$

The loss max can be a constant, which implies the upper bound of loss. Individual with a higher fitness value has a higher probability of becoming a parent of next generation. The crossover operation will select two decent chromosomes and mix them up as a new descendant. Furthermore, mutation operator will randomly revise some of the chromosome in the population based on a mutation probability. The purpose of mutation is to avoid early convergence of GA [11]. The following is the parameter setting of this experiment:

- Population size: 10
- Number of generation: 100
- Probability of crossover: 0.5
- Probability of mutation: 0.01

- Convergence: fitness > 90

Since the performance of network model on vehicle data set prediction has a relatively large fluctuation, we repeat the training process for each chromosome for 3 times and calculates the mean fitness value; Also, to avoid detrimental effects of fluctuation, we reduce the size of population to ensure that the distribution of fitness value is sparse in early generations. Conversely, the number of generation then be extended to ensure that the GA can reach a global optimum. The probability of each chromosome in a generation being selected is calculated by:

$$\text{Probability(x)} = (fitness(x) - fitness_{min})/(\sum fitness - fitness_{min} * POPSIZE) \quad (5)$$

This formula shows that chromosomes with a higher fitness value tend to be selected in the next generation. An exception is that all the fitness values in one generation are the same. This leads to a "divide by zero" exception. Thus, we consider this situation as an alternative flow, which assigns each chromosome with the same probability(x). Other settings in this paper is based on experiments.

3 Result and Discussion

3.1 Initial Result of Model

There are 1500 epochs in total to train the model, and the following table shows the result:

Table 1. The result of network model without BDR

Performance parameter	Accuracy	Loss
Train Accuracy % (1200 epoch)	72.74	0.5577
Test Accuracy % (1200 epoch)	73.40	0.7907
Train Accuracy % (1500 epoch)	79.79	0.4932
Test Accuracy % (1500 epoch)	74.47	0.7881

For the line chart, it is clearer when the loss of model converges:

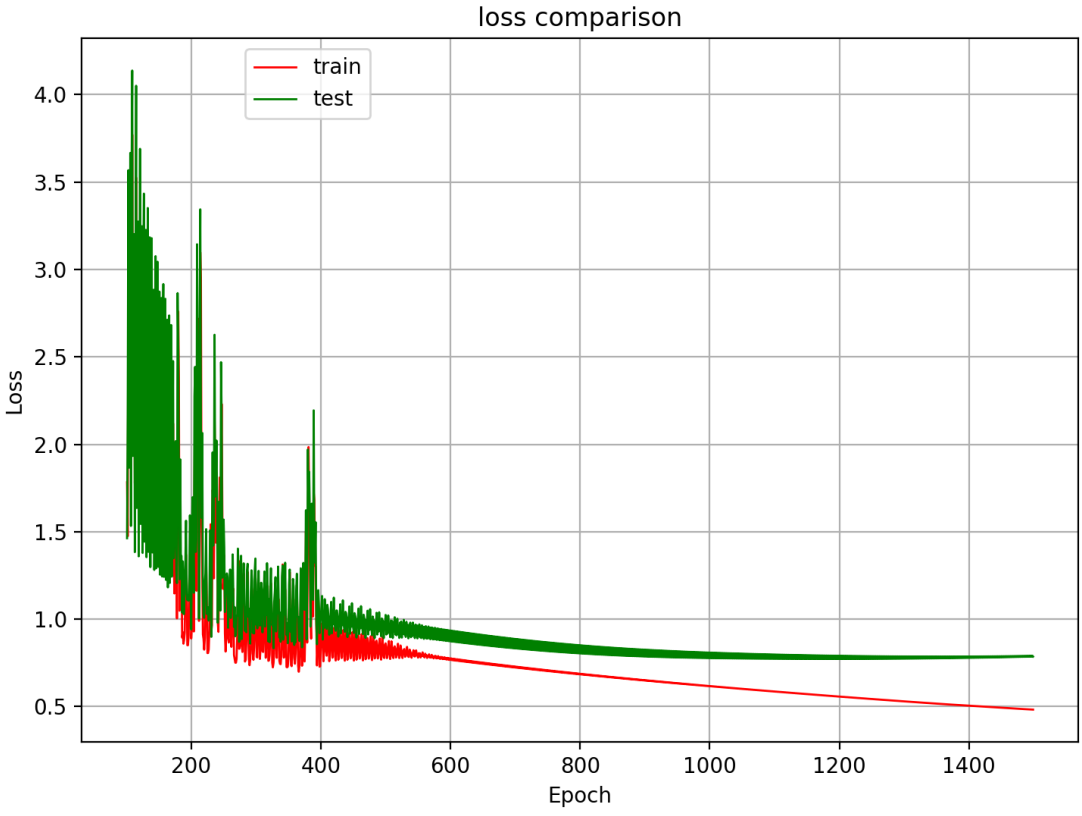

Fig. 1. A line chart that illustrates the tendency of loss comparison.

At the first 100 epoch of training, the value of loss is significant, which will affect the domain of y-axis, so in this figure the early period was dropped. From the line chart, the error of test set fluctuates in the early stage (from 100 to 400 epoch) and finally converges at around the 1400th epoch of training. It implies that the loss of test set has reached a local minimum for prediction. Accordingly, the changing curve of accuracy is:

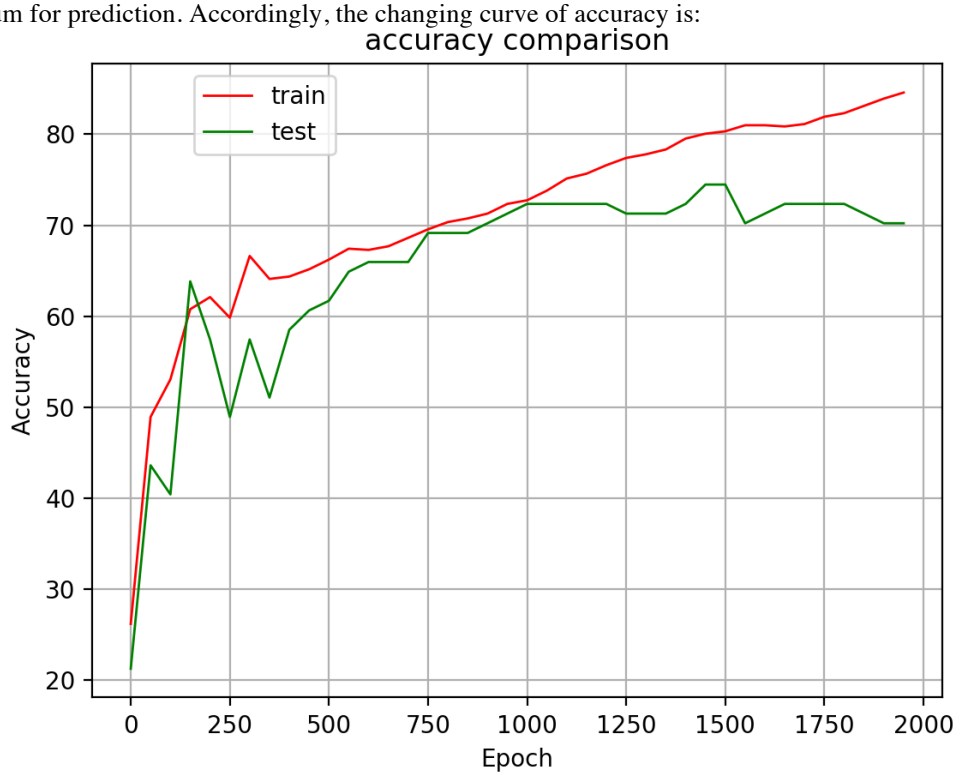

Fig. 2. A line chart that illustrates the tendency of the accuracy of train set and test set without BDR.

The accuracy of test set reaches its local maximum at round 1400-1500 epoch, which exactly match the curve of its loss changing. It is interesting to note that the accuracy of train set remains climbing and finally converges to approximately 88.5% at 3300 epochs. The may be caused by the effect of noisy training patterns.

3.2 The Performance of BDR

From the above discussion, BDR can be used to identify and remove noisy patterns in the training set. As a result, the training process converges faster. Before applying BDR, plot the error distribution onto a histogram when the error firstly falls below 1.5:

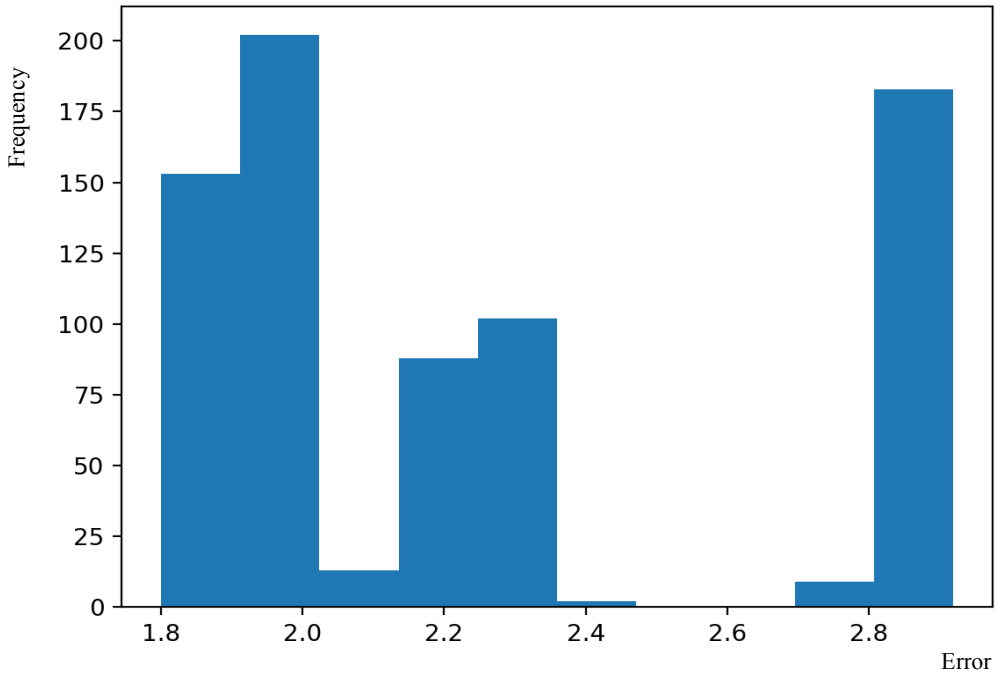

Fig. 3. A histogram that illustrates the distribution the error of train set.

This histogram illustrates the distribution of pattern errors after a 50 epochs' training process. Noisy patterns are recognized by the network itself, and it roughly obeys a bimodal distribution. Therefore, apply BDR on the training set to permanently remove the outliers and then continue to train the model.

After the first removal of noises, the loss of training set encountered a reasonable drop, while the loss of test set rose significantly. However, after several epochs, the loss declined to less than 1.5 again, which is an indicator of another execution of BDR. The error distribution before outlier removal is looked like:

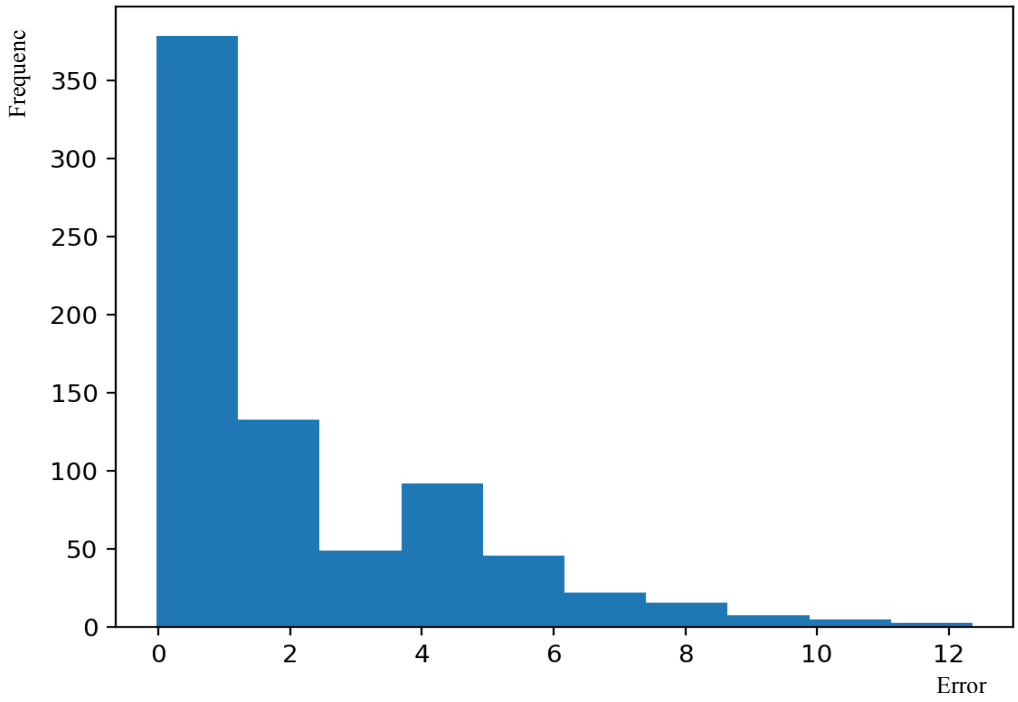

Fig. 4. A histogram that illustrates the distribution the error of train set after first BDR

This histogram shows that the error has centralized due to the removal of noisy patterns, which implies that the BDR's performance of the vehicle data set is decent. There are several executions of BDR afterwards, and the histograms are not displayed for being concise. When the CE loss of the whole remaining training set falls below 0.8, the size of training set declined from around 850 to around 650. Considering about keeping enough patterns for training, the BDR will no longer be applied to the training set.

Below is a table that shows the result of accuracy:

Table 2. The result of network model with BDR

Performance parameter	Accuracy	Loss
Train Accuracy % (1200 epoch)	83.88	0.3824
Test Accuracy % (1200 epoch)	69.15	1.0936
Train Accuracy % (1500 epoch)	86.24	0.3420
Test Accuracy % (1500 epoch)	71.28	1.1705

Compared to the result of the model without BDR, as both accuracy and loss are higher at a same epoch, the performance of classification on training set is much better. However, the performance on test set decreases around 3-5%. An explanation is that the decreased number of training set may lead to insufficient fitting of data. Besides, noisy patterns in the test set are not recognized properly. This is because the outliers in training set also helps the model to identify outliers in test set. If the training set becomes clean, then the accuracy on test set can be harmed.

This idea can be validated by adjusting the size of training set and test set. Reassign the instances as follow: randomly pick 300 instances into test sets and pick 650 instances into training set, and see the performance of BDR:

Table 3. The result of reassignment of data set

Performance parameter	Accuracy	Loss
Train Accuracy % (1200 epoch)	89.85	0.2291
Test Accuracy % (1200 epoch)	59.93	1.9176
Train Accuracy % (1500 epoch)	91.17	0.2088
Test Accuracy % (1500 epoch)	60.28	2.0869

From the result of the validation, the convergence of training set appears earlier. Conversely, the performance of model on test set becomes worse. The shrunk size of training set may be a reason for this change, and the increased number of noisy patterns in test set may be another.

Here is a changing curve of loss and accuracy on training set and test set:

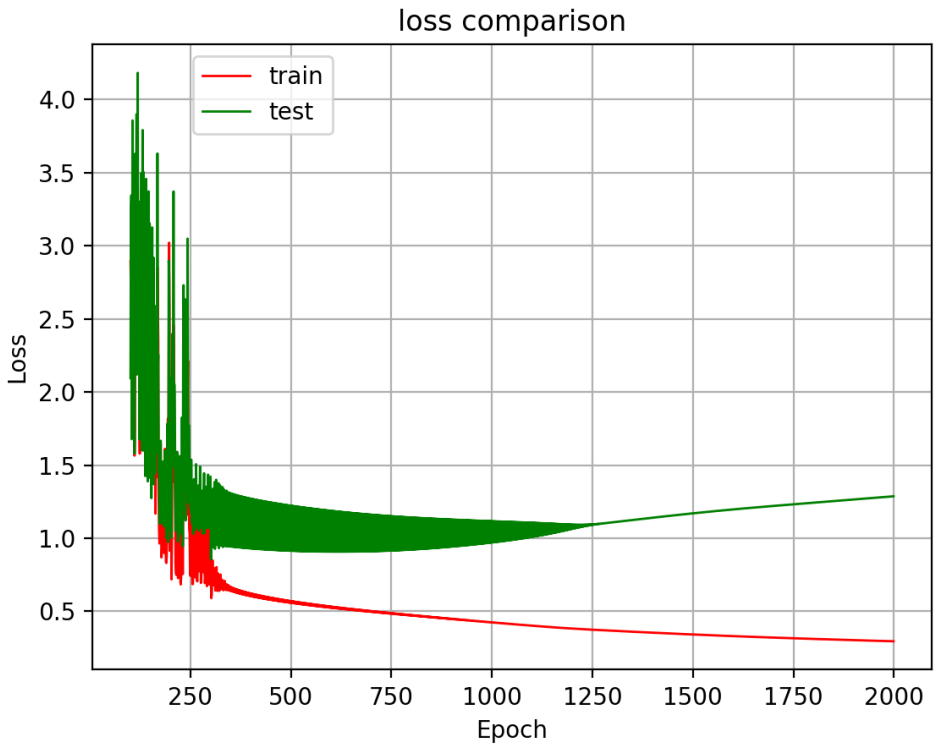

Fig. 5. A line chart that illustrates the tendency of loss comparison applying BDR

The loss of test set converges at around 1250-1300 epochs. Compare this line chart to the one without BDR, the loss of training set converges faster, while the convergence of test set slows down. The upgrade is a benefit that BDR have on training process [6], but the noisy patterns in test set becomes tougher for the model to identify. Accordingly, the accuracy is shown below:

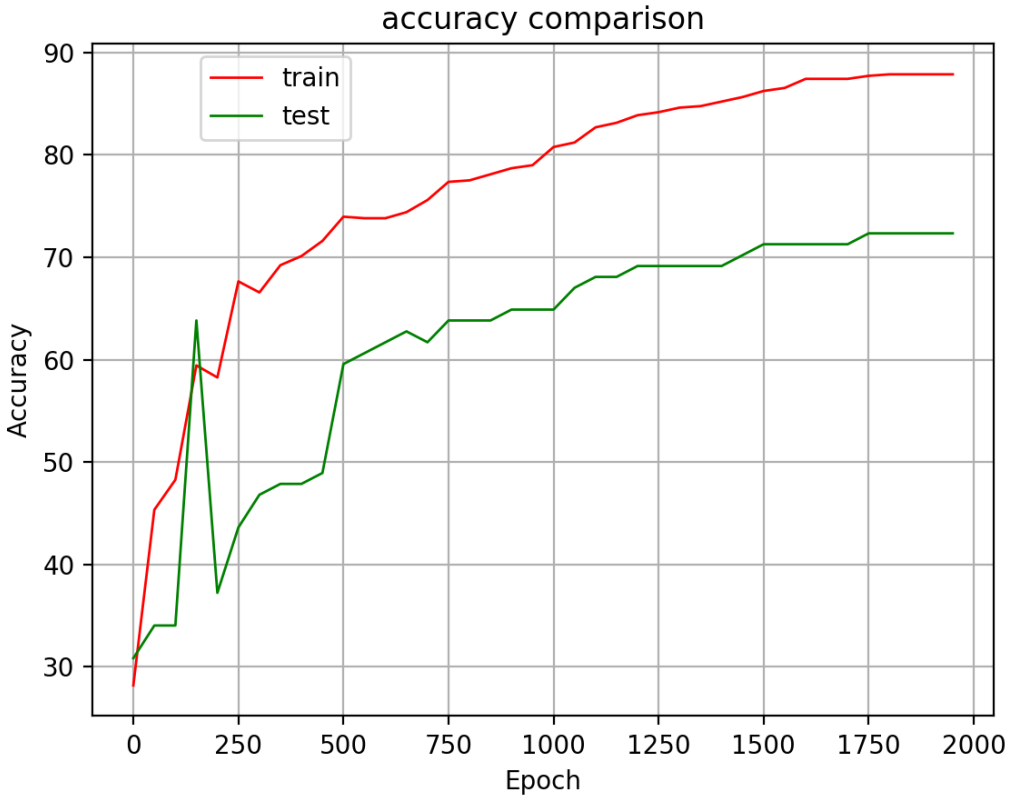

Fig. 6. A line chart that illustrates the tendency of the accuracy of train set and test set.

After BDR has been performed, the accuracy of training set increases and converges faster. On the other hand, it becomes more difficult for the model to recognize the noisy patterns in the test set.

3.3 The Performance of GA feature selection

The following table shows the performance of GA feature selection on the initial network model with different fitness functions:

Table 4. The result of GA with two parameters' fitness

Generation	Best Chromosome	Best Fitness
0	[4 2 0 1 3 3 1 0 2 1 3 4 4 3 1 2 1 1]	75.9
25	[4 0 3 1 3 1 2 0 2 3 3 2 3 2 1 0 3 1]	73.1
50	[4 0 0 1 3 1 2 1 2 3 0 2 3 2 1 0 3 1]	71.2
100	[4 0 1 1 3 0 0 0 0 3 0 2 3 2 1 0 3 1]	78.4

Table 5. The result of GA with accuracy fitness

Generation	Best Chromosome	Best Fitness
0	[2 0 0 1 2 0 0 4 2 1 2 2 3 3 2 2 1 1]	67.4
25	[2 1 4 1 2 0 0 4 2 0 2 2 0 3 2 0 1 1]	70.2
50	[2 1 4 1 2 1 0 4 0 0 2 2 0 3 2 0 1 1]	73.4
100	[2 1 4 1 2 0 0 4 0 1 1 2 0 0 2 0 1 1]	72.3

Since the genetic feature selection is a probability event, it is understandable that some better feature subsets disappear. The first table shows that the combination of loss and accuracy fitness leads to a significant fluctuation on genetic selection. Conversely, the accuracy fitness shows a steady enhancement for this data set. Furthermore, the second table indicates that some features in this subset did not contribute much to the training process. For instance, after the removal of the 7th, 9th, 10th, 13th and16th feature, the fitness remains at a similar level, as showed in Table 1. Therefore, GA feature selection can be an effective method to enhance the efficiency of training process. With a pruned feature set, the network size can be reduced.

3.4 Comparison with Another Algorithm

There are some other techniques of neural network that have been applied on the vehicle data set. One of which is Constructive Neural Network (CNN) learning algorithm. In 2000, an approach to build a near-minimal neural network structure for pattern classification, which is MTilling-real algorithm, is proposed by Parekh, Yang and Honavar [3]. It is an extension of CNN: the network itself can grow from a small architecture into an ideal multi-layered model to solve the problem. Specifically, it starts from a single layer network with several master neurons; while the accuracy does not satisfy the requirement and the number of layer is below the limitation, then patterns relates to more than one class will be picked, and the corresponding output neurons will be divided into 2 neurons for classifying the patterns. The strict convergence condition is guaranteed to return an efficient network model for classification, but it also takes much more time to converge.

Here is a performance of MTilling-real algorithm on vehicle dataset:

Table 6. The result of MTilling-real algorithm

Performance parameter	Accuracy
Train Accuracy %	87.5 ± 4.4
Test Accuracy %	77.5 ± 6.2

The accuracy of training set finally converges to a similar level of the model with BDR, while the test accuracy is higher than the model with BDR. The reason may be that the architecture of this result is trained to be able to identify noisy patterns in both training set and test set.

4 Conclusion and Future Work

In this paper, a neural network model has been designed to solve classification problem for vehicle dataset. The dataset was randomly divided into training set and test set. The network model is evaluated by accuracy and loss function. The first network model reaches a fixed point with roughly 73.4% accuracy. Then, BDR was applied to perform noise removal in training set. However, the accuracy falls to 71.28%. This may be caused by the insufficient instances of data set, and a stubborn strategy of division on data set. Another technique used to improve the network model is GA, which

can be an effective approach for feature selection. As a result, around 6 features can be deactivated to enhance the efficiency of training process with its accuracy of prediction remains at a same level (72.3%~73.4%). Since GA is extremely computationally expensive, a small data set and simple network structure is required for GA to converge swiftly. Therefore, GA feature selection tends to be more acceptable for the vehicle classification problem.

Optimizing the structure of network model is another way to enhance the performance on pattern classification, as MTilling-real algorithm based on CNN has provided a successful attempt. Furthermore, there are some other customized operations that can be applied in BDR algorithm. One of which is re-splitting the data set after each execution of BDR, such that the noisy patterns in both training set and test set can be removed synchronously. In addition, from the process of GA we can see that there are some decent individuals disappear. The idea to tackle with this problem is to find a balance between the size of population and the number of generation.

References

1. Giacinto, G., & Roli, F.: Design of effective neural network ensembles for image classification purposes. Image and Vision Computing, 19(9), 699--707. doi:10.1016/S0262-8856(01)00045-2 (2001)
2. Pete, M., & Barry, S.: UCI Repository of Machine Learning Databases, Turing Institute Research Memorandum TIRM-87-018 http://archive.ics.uci.edu/ml/datasets/Statlog+%28Vehicle+Silhouettes%29 (1987)
3. Parekh, R., Yang, J., & Honavar, V.: Constructive neural-network learning algorithms for pattern classification. IEEE Transactions on neural networks, 11(2), 436--451 (2000)
4. Bustos R.A., Gedeon T.D.: Decrypting Neural Network Data: A Gis Case Study. In: Artificial Neural Nets and Genetic Algorithms. Springer, Vienna (1995)
5. Recht, B., Re, C., Wright, S., & Niu, F.: Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In Advances in neural information processing systems. 693--701. (2011)
6. Slade, P., and Gedeon T.D.: Bimodal Distribution Removal, vol. 686 (1993)
7. Chamkalani A. et al: Pattern recognition insight into drilling optimization of shaly formations. Journal of Petroleum Science and Engineering, vol. 156, 322--339 (2017)
8. McCaffrey J. D.: Why You Should Use Cross-Entropy Error Instead Of Classification Error Or Mean Squared Error For Neural Network Classifier Training, https://jamesmccaffrey.wordpress.com/2013/11/05/why-you-should-use-cross-entropy-error-instead-of-classification-error-or-mean-squared-error-for-neural-network-classifier-training/
9. Zhao Z. et al: Investigation and improvement of multi-layer perceptron neural networks for credit scoring, Expert Systems with Applications, vol. 42, (7), 3508-3516 (2015)
10. Gedeon T.D. and Harris D.: Network Reduction Techniques. Proceedings International Conference on Neural Networks Methodologies and Applications, AMSE, San Diego, vol. 1: 119-126 (1991)
11. Chtioui, Y., Bertrand, D., & Barba, D.: Feature selection by a genetic algorithm. Application to seed discrimination by artificial vision. Journal of the Science of Food and Agriculture, 76(1): 77-86. (1998)
12. Yang, J., & Honavar, V.: Feature subset selection using a genetic algorithm. In Feature extraction, construction and selection. Springer, Boston, MA.: 117-136 (1998)