# Analysis of Bimodal Distribution Removal on real world ANN classification problems

Nikhil Mathew[1]

[1] Australian National University
0200 Canberra, Australia
u6166704@anu.edu.au

**Abstract.** An analysis of Bimodal Distribution Removal (BDR) techniques on two real world datasets in different domains. Both datasets have specific classes that pose a challenge to a single layer neural network and do not have clearly identifiable outliers. We implement simple Artificial Neural Network (ANN) classifier methods from relevant work on each dataset as baselines and then analyse the effects of adding BDR as an outlier detection and removal mechanism. Areas of improvement for BDR are identified and discussed.

**Keywords:** outlier detection, bimodal distribution removal, neural networks

## 1 Introduction

The success of a neural network as a classifier is dependent on the quality of the training data provided. Real world data however may be subject to errors or special cases which do not follow the general trend of the data. These aberrant data points are outliers, more carefully defined as "An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism [1]."

Thus we use outlier detection as a mechanism to remove these misleading data points from the training data set. Outlier detection can be broadly classified into distribution based and distance based methods [2]. Bimodal Distribution Removal is one such distribution based outlier detection method. It has been shown that this technique works in a real world data setting for non-parametric regression problems [3]. In this report we aim to analyse its effectiveness on real world classification problems. It will be interesting to see if the method can be easily applied to such problems.

The first dataset chosen for this task is the ISOLET dataset. It facilitates a speech recognition task where the objective is to recognize english alphabets in speech [4]. The data is already well preprocessed in this dataset. The neural network described in [4] is a single layer neural network that shows an accuracy of 95.9% on the test set. This dataset was chosen to examine the effects of BDR on a very clean dataset. The second dataset was the COVTYPE dataset. It facilitates a cover type classification task based on geographical data. Like [4] the authors use a single layer neural network to classify this data [5]. However this task is much harder and [5] describes non-linearities in the dataset that cause the data to be hard to classify. The neural network described demonstrates a 70.53% accuracy on the test set. This dataset was chosen to examine the effects of BDR on a more difficult classification task. Finally we use the OPTDIGITS dataset, which is an image classification task developed for optical recognition of handwritten digits [8]. This set was chosen to show the effects of this method on deep convolutional networks and more importantly provide interpretability on the outliers detected. This was a good choice for interpretability of the outliers as we can visually see and judge the digits and chosen by the model for removal.

The next section provides a brief introduction on Bimodal Distribution Removal and discusses how it can be used to improve the training on sets with noise. Next, we discuss the methods used in this paper to judge BDR's performance on parametric classification tasks and the

evaluation metrics chosen. We then present the results of the experiments and provide an analysis and discussion during which we point out some shortcomings of this method that need to be addressed. Finally we present our conclusion and discuss some interesting avenues for future work in this domain.

## 2 Bimodal Distribution Removal

In terms of a statistical framework there are two main ways a data point pair, consisting of an input and a label, can be considered an outlier. Either the input pattern does not conform to the distribution of the class of inputs it belongs to or the output label does not obey the conditional distribution based on the input it is mapped to [3]. In either case there is a seemingly faulty mapping between input and output. Outlier detection mechanisms are designed to identify these erroneous data points so as to prevent their influence on the training of the statistical model developed from the data [1].

The key feature of BDR is to use the model being trained to implicitly perform outlier detection while training. This is done by calculating the the prediction errors on the training set, a usual pattern in neural network models is that the model will fit to the majority of the dataset. There will however be a few data points that do not conform to the input class distribution and thus result in large errors. We thus get an error distribution graph which is bimodal similar to the one below (Fig. 0). Thus we chose to remove there misleading data points by trimming them out of the training set.

To do this, we calculate the mean of the training error distribution. This mean will be close to the mode of the examples with low error as seen by the dotted line to the left of the figure. We then set this as the threshold and calculate the mean $\bar{\delta}_{ss}$ and variance $\sigma_{ss}$ of all the errors above the threshold indicated by the middle and right dotted line. We then take the examples with error $> \bar{\delta}_{ss} + \alpha\sigma_{ss}$ , where $\alpha$ is a hyperparameter that we can tune. In this way the model being trained itself is used for outlier detection. All that remains is the choice of start and stop points for this mechanism. If we start to early this could trim out a considerable portion of the training set, if we start to early there will be no effect of BDR and the outliers will be assimilated into the model. [3] provides a solution by saying around 200-500 epoch's is generally a good starting point and if the variance of the test error crosses a threshold $V_s$ (usually 0.01) we can stop the mechanism.

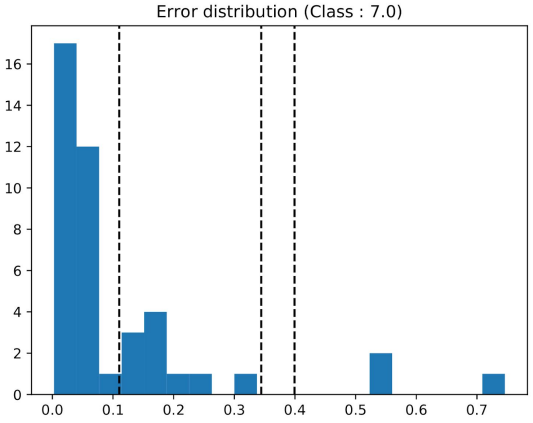



**Fig. 0.** Example of training error distribution for a given class. As we can see there are two groups of error on either side of the graph. Majority of the examples have low training error on the left and few classes have high training error on the right.

## 3 Method and Evaluation

In this section we apply BDR to two datasets and compare the effects with a reference paper on the same dataset as the baseline. For a fair comparison, for both datasets a neural network model was created directly based on the one mentioned in the paper. This was done as the objective is to observe the effects of BDR on an already established ANN model. Only minor modifications were made, more specifically instead of the Mean Squared Error loss used in [4] and [5], cross entropy loss was used; also the ADAM optimiser was used instead of standard stochastic gradient descent with momentum. Both changes were made in order to speed up the running time of experiments. We then apply BDR to a third dataset using a more complicated deep learning model and pull out the outliers for visual interpretation of the mechanisms results.

The ISOLET dataset contains 617 features and 26 classes [4]. Size of the training set and testing set were 12738 and 1559 respectively. A single hidden layer Neural Network with 52 hidden units and the sigmoid activation function was used. Softmax is used at the output layer to normalise the output scores to resemble a probability distribution. The network was then trained with cross entropy loss on and ADAM optimiser with a learning rate of 0.01. The dataset was already preprocessed by its creators and thus raw values of the dataset were directly fed into the neural network.

The COVTYPE dataset contains 54 features and 7 classes [5]. For this data, a single hidden layer Neural Network with 120 hidden units and a sigmoid activation function was used, softmax was used at the output layer. Cross entropy loss was used as an optimisation objective on an ADAM optimiser with a learning rate of 0.05. This dataset required considerable preprocessing. The preprocessing followed [6] where all distances were scaled down based on the training set distances; angles were normalized using the sine function. Preprocessing had an enormous impact on the performance of the model, the test accuracy went up by 20% when compared to training using raw data. As described in [6] the first 11,340 records are used for training, the next 3780 are used for validation and the remaining 565892 are used for testing. Note that to keep this experiment consistent with the ISOLET experiments [4] we do not use the validation set.

The OPTDIGITs dataset consists of 946 examples of 32x32 pixel handwritten digits. For our purposes these images were divided using a 70%-30% split of 662 training examples and 284 testing examples. To perform classification a convolutional neural network (CNN) was applied. Two convolutional layers with 10 and 20 filters were used along with max pooling layers. The output from the final max pooling layer was fed to a two layer feed forward network with 30 and 20 hidden units respectively. This was then reduced to an output layer of 10 units which was subsequently normalized using softmax. For the loss as above, Cross Entropy loss was used and the Adam Optimiser with a learning rate of 0.01 was used to optimise the problem.

For all the datasets we injected a BDR removal technique as described in [3]. The start of the BDR process was kept as a hyper-parameter, generally set to 200. The BDR process was made to halt when the test variance was less than 0.01 as specified in [3]. For OPTDIGITS the start epoch needed to be set much lower at around 30-40 epochs. Similar to the cases above the stop criteria of test variance less that 0.01 was used. Evaluation was done using the overall system accuracy as an evaluation metric. Measures of mean and variance on the training set errors and training loss were also provided for further insight.

# 4 Result and Discussion

## 4.1 ISOLET dataset

The ISOLET dataset proved to be an easy set to classify. The preprocessing of the dataset is really done well [4]. The model converged to a good solution within 200 epochs. The variance of the errors also dropped below the halting threshold before the 200th epoch. As a result BDR halted as soon as it started on epoch 200 and did not influence the training. This is the expected behaviour of BDR as it should not influence models with clean and easily classified data.

While this is a satisfactory result, a further case was explored by being more aggressive with the start condition. The error variance is above the halting threshold at epoch 100 and thus the new BDR start epoch was set to a more aggressive epoch 100 to asses the difference.

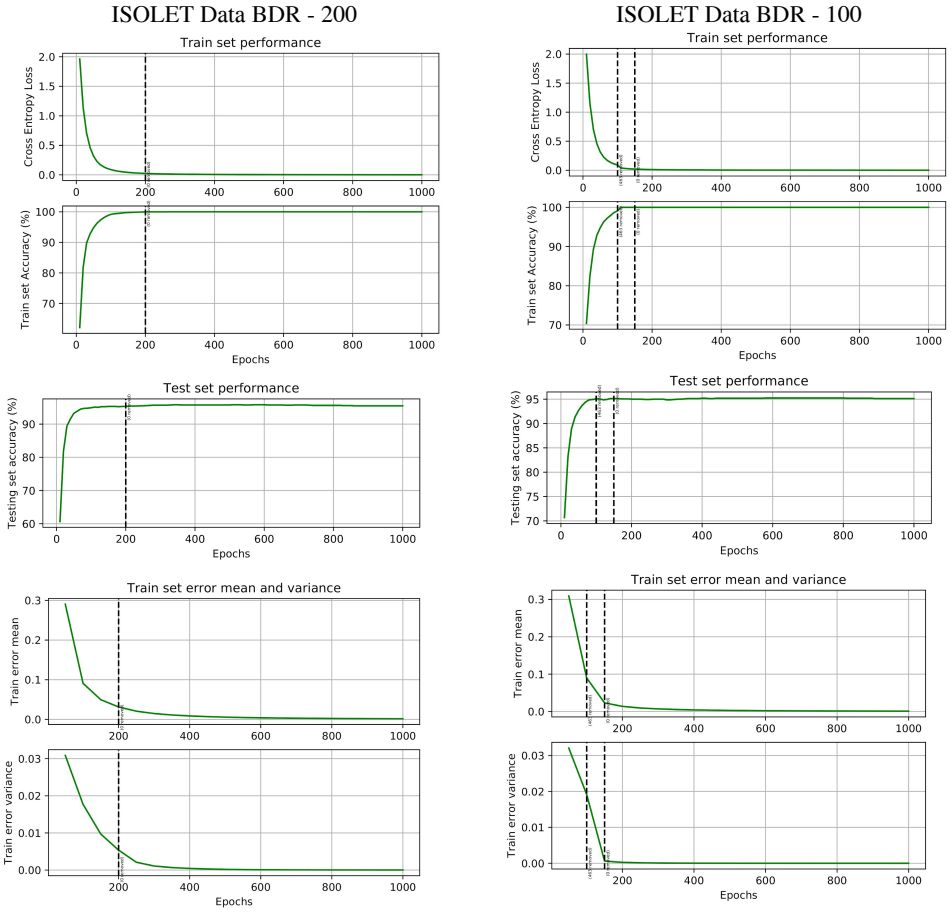



**Fig. 1.** The set of graphs on the left column displays results of BDR starting at the 200th epoch, the set of graphs on the right display results of BDR starting at the 100th epoch. For BDR-200 : Test set accuracy monotonically increases and saturates at ~95% before the 200th epoch. The dotted line indicates that BDR was attempted at this epoch but the halting condition had already been established. Thus no instances were removed BDR was halted at the 200th epoch. Training loss decrease and training accuracy increases monotonically. Since BDR halted at epoch 200, this pattern was mainained there after. The mean and variance both decrease rapidly. The variance is below the halting threshold of 0.01 before the experiment starts. As a result the BDR-200 does not influence the training in anyway. For BDR-100, 463 instances were removed at epoch 100; the aggressive start epoch of 100 doesn't affect the

overall result too drastically, test accuracy saturates at ~95%. There is however a sharp decrease in train error variance as soon as the method is incorporated.

The more aggressive setting leads to only a minor deterioration in test performance (as seen in Fig. 1). The test accuracy dropped by 0.5% to 95.19%. By setting the BDR start epoch to 100, 463 training examples were removed. While the number of examples removed is considerable, this is on the whole encouraging as the decrease in performance is not drastic. At epoch 100, the train error variance is around twice the threshold as seen in Fig. 1. It is too early to begin BDR at this stage. Thus, overall BDR worked as expected on a clean dataset and did not influence the networks learning too much.

It must be noted however that there was a sharp decrease in variance as soon as the method was used. This is an early indication that if the neural network model has not learnt to classify the problem well before the start of BDR, BDR will tend to reinforce the models existing patterns learnt by removing patterns it has not recognized yet. This indicates that the BDR start epoch is a critical hyperparameter. If BDR start epoch is set too low in a setting where the model has not sufficiently decreased the train error variance, it could potentially harm the performance of the system by removing difficult patterns too early. Note that the 200 - 500 epochs mentioned in [3] is situational and can fluctuate dramatically based on the learning rate and choice of optimizer used. A new criteria to dynamically identify the starting point will have to be determined for this method to be used more generally. Potentially the start point could be determined as a function of the train error variance.

### 4.2 COVTYPE Dataset

The COVTYPE dataset is a much tougher classification problem. [5] reports a neural network as designed above achieves 70.58% accuracy on the test set. The authors attribute this to a difficulty in classifying certain classes owing to a non linearity in one of its determining features (the distance to nearest fire-ignition points) [5]. Since a single neural network is used, these non-linearities are not easy to deal with. While this is interesting, the focus of this experiment is not to compete with these results and beat them by using a more complex network architecture. Rather, the main objective is to analyse the effect of BDR on this problem with the given setting. Running a neural network without BDR results in a performance similar to the accuracy mentioned in [5]. Note that a lot of preprocessing of the data, as described in [6], was required to achieve this performance.

Turning on the BDR functionality on this dataset leads to a degradation in results. It is seen that the overall test accuracy immediately dips after the BDR function is used. At epoch 200, 1768 records were removed. Then at epoch 250, 1008 epochs were removed. These removals are very aggressive. Notice how in this setting at epoch 200, the error variance was more than twice the threshold value, whereas in the previous dataset the error was under the threshold and BDR was skipped entirely.

Another point to note is that error distributions per class at this stage are radically different. Some classes are easy to predict where as some classes are much harder. As shown below in Fig 2. The Lodgepole Pine class is difficult to predict due to non-linear relation to the attribute “historic wildfire ignition point” [5]. At this stage too much information would be lost for this class if BDR was to take place. This mechanism needs to account for cases of class imbalance and varying difficulty in classification per class.

A potential way to overcome this drawback is to perform BDR per class instead of on the whole training set. This would need to make use of a dynamic start condition to be determined on the fly for each class. Starting an outlier detection mechanism for all classes at the same time does not offer the model sufficient time to learn the nuances in the difficult classes.

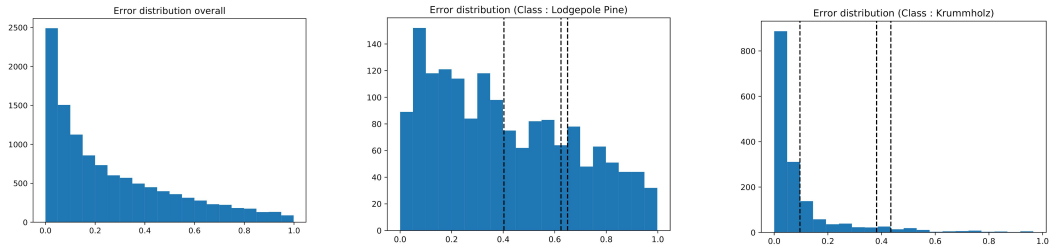



**Fig. 2.** Overall error distribution at epoch 200 shown on the left. Error distribution of a difficult to predict class (Lodgepole Pine) at epoch 200 in the middle. Error distribution of a relatively easy class (Krummholtz) at epoch 200 on the right. The first vertical line is the mean of the graph, the second vertical line is the mean of the erroneos subset and the third is the subset mean + the subset standard deviation. The graph shows, it is too early to perform BDR on the hard class, the network has not yet learnt how to properly classify this yet. Meanwhile BDR can be beneficial for the easily learnt class to prevent overfitting.

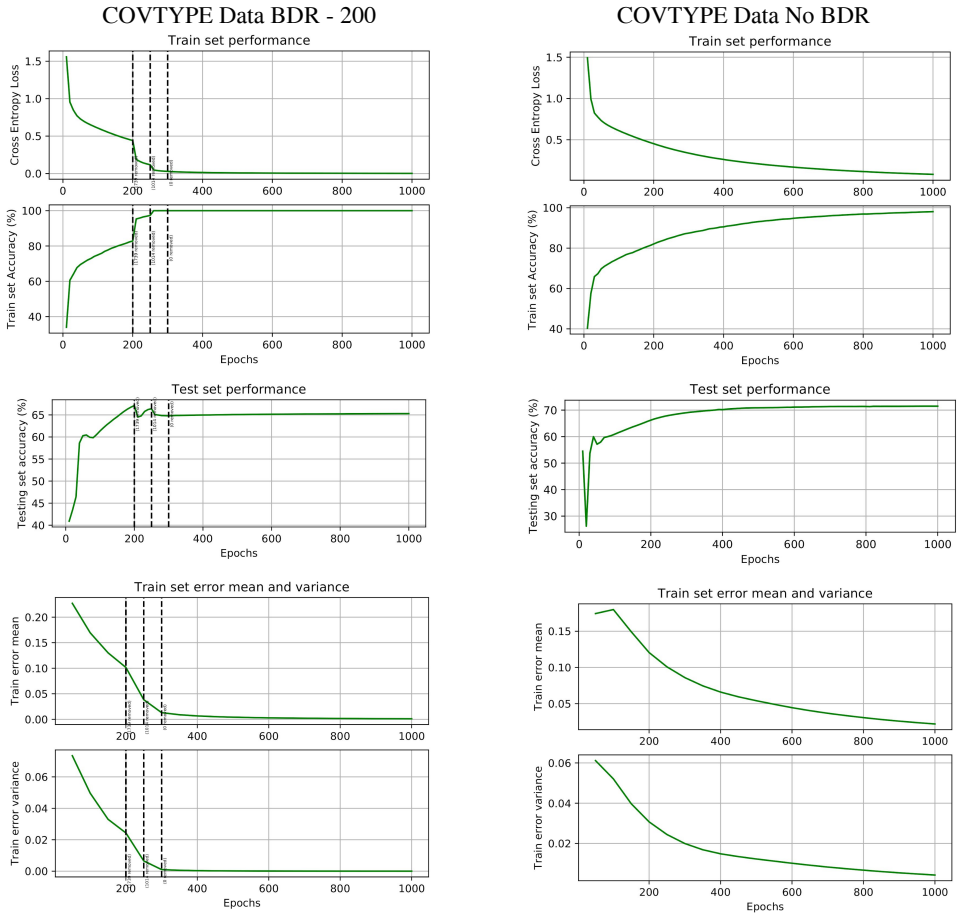



**Fig. 3.** The graphs on the left shows the model's performance on the train set with BDR active at 200 epochs. The graphs on the right show performance with BDR disabled. At epoch 200, 1768 instances were removed. At epoch 250, 1008 instances were removed. The model accuracy went down on the whole when BDR was used. BDR dramatically boosts the train accuracy and sharply decrease test accuracy, indicating that the BDR will simply reinforce the models current learning instead of detecting outliers.

On examining the train performance (Fig. 3), it is evident that BDR reinforces the learnt patterns of the model as all the patterns that the model is having a hard time to classify are removed from the train set. This is evidenced by the dramatic increase in training set accuracy, and sharp decrease in train error variance at this point. BDR is too aggressive in this setting,

when applied early it removes a significant chunk of the training patterns and serves to reinforce the easily identified patterns. This results in a sharp decrease of the test set performance. Notice that early starting showed a much larger decrease in performance when compared to the previous classification problem. This is due to the fact that this classification problem is much harder. This is confirmed by the gradual reduction in variance when not using BDR in Fig. 3, it is observed that the variance reaches the threshold at epoch 600.

### 4.2 Adding Noise to Dataset

Finally, we test the effect of BDR on the datasets above with artificially introduced noise. For the ISOLET dataset when the start epoch was set to 200, the BDR halting condition was met before the it could start and hence it did not influence the test performance. On setting the starting point to the more aggressive 100 epochs there was a slight deterioration in results (Fig. 4). Overall the system behaved similarly to the setup without noise with the only differences being the overall test performance was lower due to the noise and the number of instances removed. This is strange as it is expected that the BDR mechanism should help the model increase performance in this case by identifying outliers. BDR removed a much larger portion of the noisy training data.

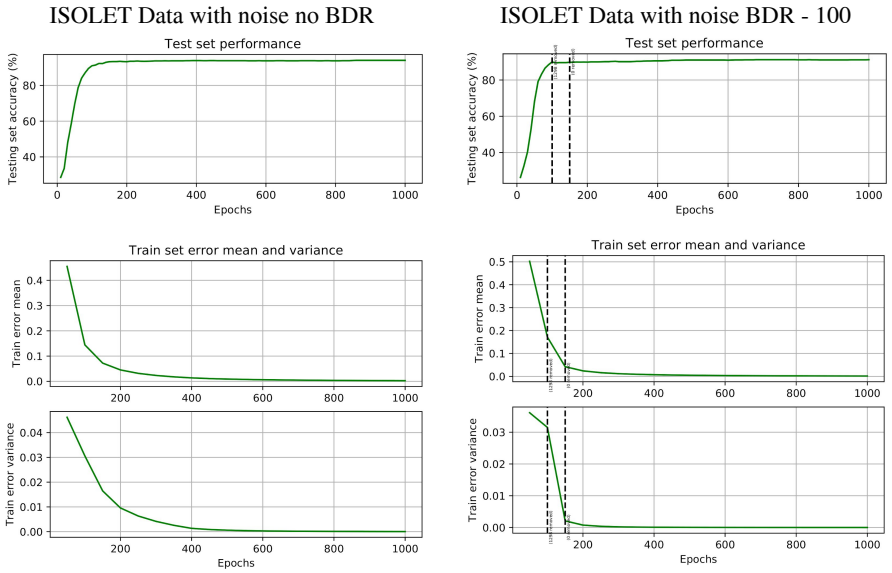



**Fig. 4.** The graphs on the left shows the model's performance on a noisy ISOLET train set without BDR. The graphs on the right show performance with BDR enabled. At epoch 100, 1298 instances were removed. At epoch 150, BDR halted. Test accuracy went down slightly when BDR was enabled. There was also dramatic decrease in train error variance when BDR was enabled.

On the COVTYPE dataset, there was little impact of adding noise. This has been examined in [7] where it is observed that adding randomness is a simple way to prevent overfitting in hard problems, also more specifically [6] shows that adding random noise in GIS data can improve performance. The model was able to achieve a similar performance as before with 71.3% accuracy on the test set despite the addition of noise. When BDR was enabled however the performance dropped dramatically, just as before. The test accuracy decreased by over 5% after using BDR. It seems that all the points discussed before remain valid even in the case of artificially added noise (Fig.5). This is surprising, as this is an area that BDR should help improve.

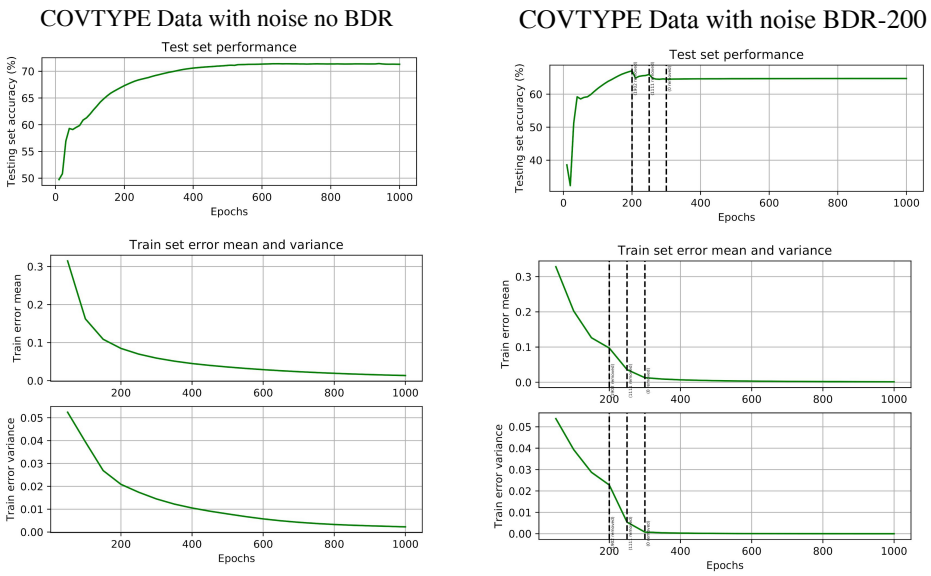



**Fig. 5.** The graphs on the left shows the model's performance on a COVTYPE train set with added noise without BDR. The graphs on the right show performance with BDR enabled. At epoch 200, 1902 instances were removed. At epoch 250, 1111 instances were removed. At epoch 300 BDR halted. Test accuracy deteriorated by over 5% on using BDR.

**Table 1.** Summary of Results

| Method | BDR Start point | Instances Removed | Test Accuracy (%) |
|---|---|---|---|
| ISOLET-NO-BDR-NOISY | - | - | 94.03 |
| ISOLET-BDR-NOISY | 100 | 1298, 0 | 91.21 |
| ISOLET-BDR-NOISY | 200 | 0 | 94.03 |
| ISOLET-BDR | 200 | 0 | 95.77 |
| ISOLET-BDR | 100 | 463, 0 | 95.19 |
| ISOLET-NO-BDR | - | - | 95.83 |
| ISOLET-REFERENCE | - | - | 95.9 |
| COVTYPE-NO-BDR-NOISY | - | - | 71.30 |
| COVTYPE-BDR-NOISY | 200 | 1911, 1074, 0 | 64.17 |
| COVTYPE-BDR | 200 | 1739, 1014, 0 | 65.31 |
| COVTYPE-BDR | 300 | 1387, 962, 0 | 68.16 |
| COVTYPE-BDR | 400 | 1325, 0 | 70.11 |
| COVTYPE-BDR | 500 | 1172, 0 | 69.91 |
| COVTYPE-NO-BDR | - | - | 71.48 |
| COVTYPE-REFERENCE | - | - | 70.53 |

### 4.4 OPTDIGITS Dataset

The results for the OPTDIGITS dataset follow the trend above. We observe that Bimodal Distribution Removal negatively affects the training by reinforcing the patterns that the model has already learnt. As discussed above the start point varies dramatically based on the task and this is a clear example for this. Anything above 60 epochs will not activate BDR and thus the generalization of 200 epochs being a good starting point is invalid.

As before BDR resulted in a drop in test performance when compared to the model being allowed to run freely. An interesting note about this model is that the variance is quite high when we compare to the other tasks, Multiple runs were required to produce good results where in some cases the model failed to learn completely. Similarly in some runs the variance

dropped down quickly and thus BDR would not be used even when set to start at 30 epochs. For 20 epochs we've attached the examples picked as outliers in Fig. 7.

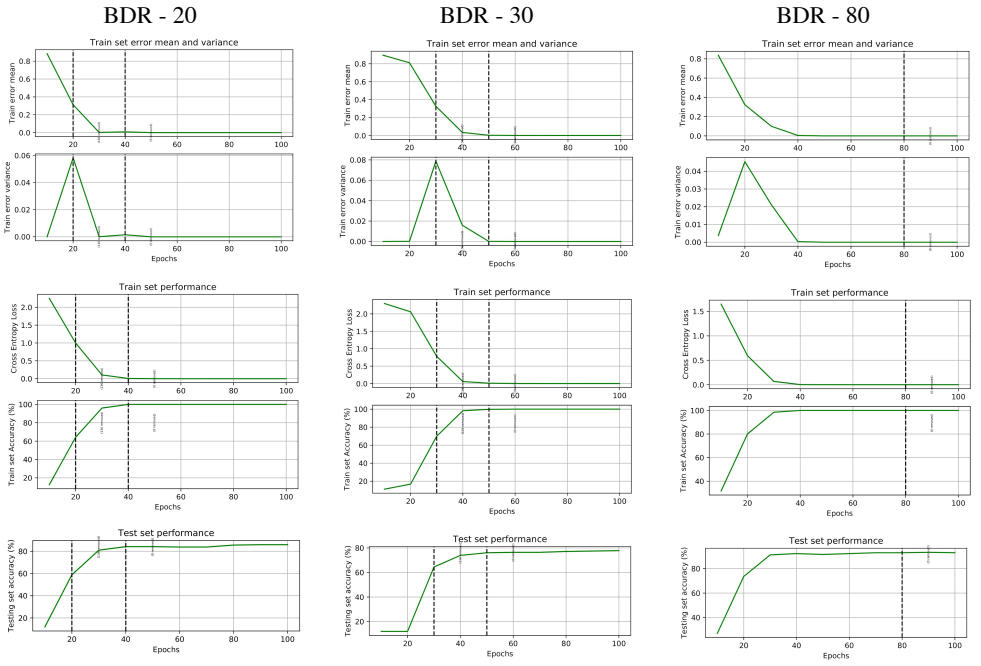
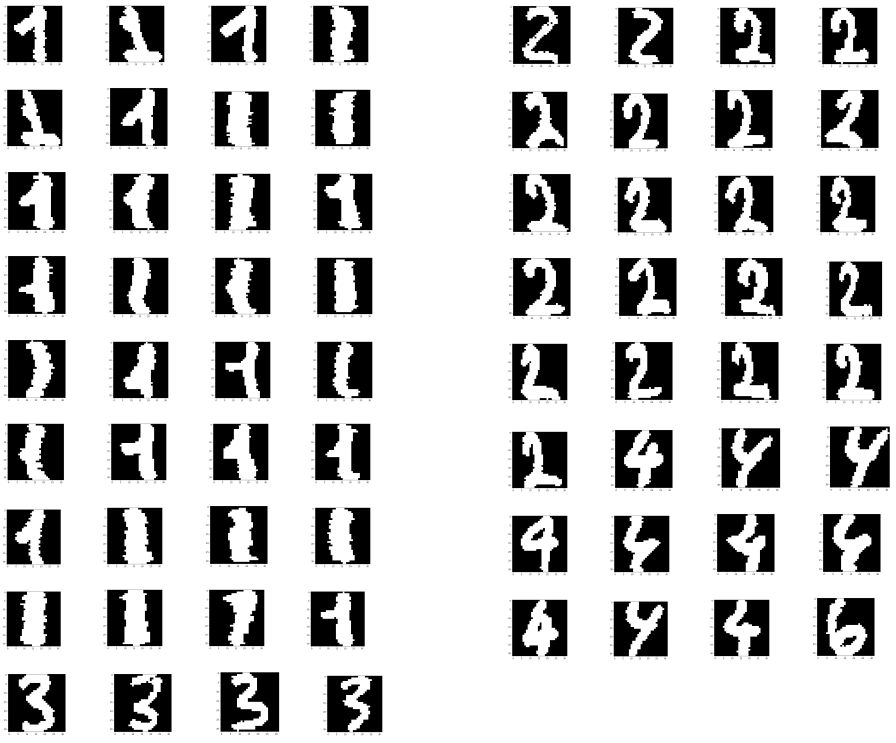



**Fig. 6.** The graphs above show the performance of BDR on OPTDIGITS at different start point. On the left we see the effects of BDR starting at epoch 20, it is observed that on this dataset the effects are dramatic, with a considerable game in test performance. In the middle similar performance for BDR-30 and on right BDR-80 where BDR did not activate shows that if left alone the model performs much better.



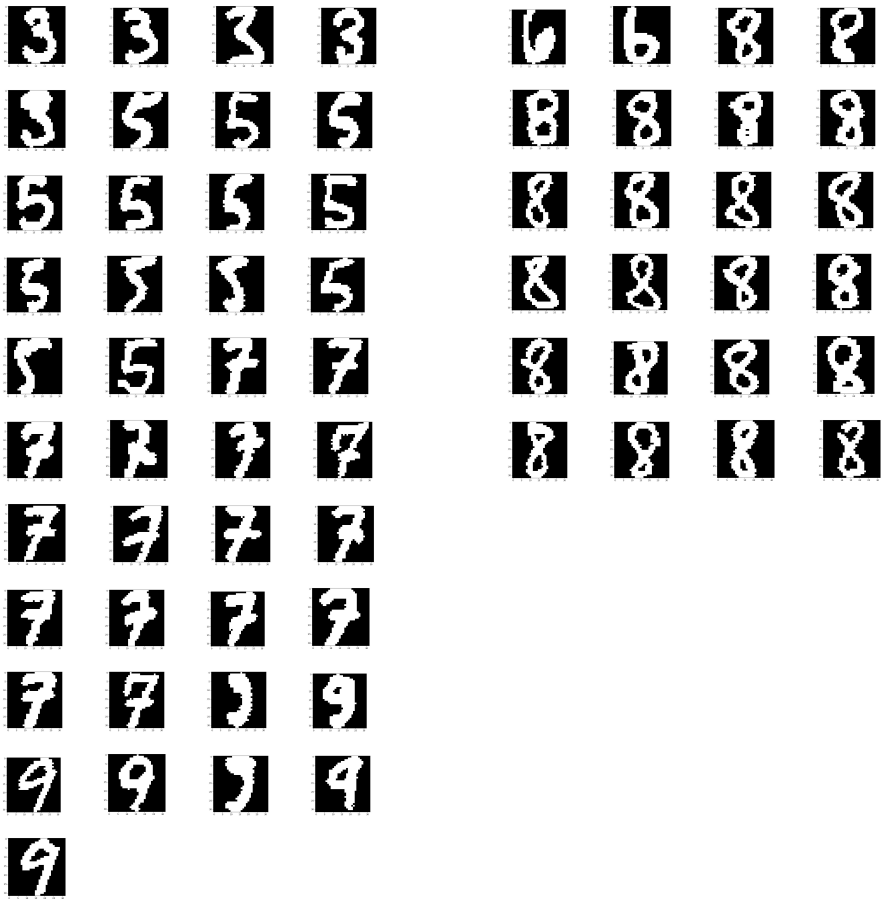
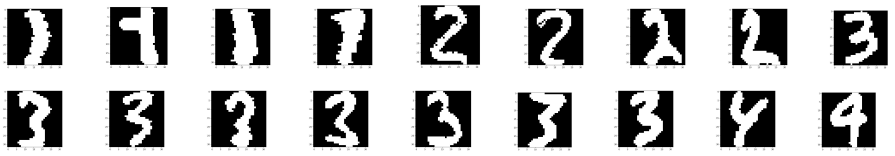


**Fig. 7.** Shows the outliers detected by the model starting at epoch 20. While some of them are true outliers, chances are that you are able to recognize most of them despite their deformities. Also the evident is the presence of class imbalance in outlier detection.

The figure above clearly illustrates a couple of important points. Firstly, that even though the outlier detection is working as we can see in the badly formed examples of 1 and 9, the cases in the other digits are easily classified by humans. This shows us that there is no one size fits all starting epoch, while epoch 20 was sufficient to pick out outliers for some numbers, other numbers like 8 and 3 still need sufficient time and the outliers picked out here could be useful to make the training more robust.

In contrast the outliers detected when the starting epoch is 30 are fewer and more precise outliers. While we still see that there are some valid cases for the class 3 indicating the class imbalance problem discussed earlier, we see that on the whole just moving the starting epochs by 10 epochs can have a dramatic impact on how the system performs.



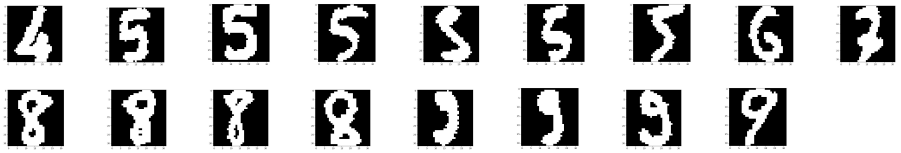


**Fig. 8.** Shows the outliers detected by the model starting at epoch 30. In general there are fewer and more valid outliers, indicating the criticality of the starting epoch.

## 5 Conclusion

The effects of using BDR in real world classification with neural networks was examined. Two different datasets were used, one relatively easy problem and a harder GIS based classification problem. It was observed that Bimodal Distribution Reduction is not easily applied to ANN classification problems. In the simpler problem BDR was skipped entirely, and in the harder problem BDR caused a decrease in performance. Furthermore, the performance was examined after adding artificial noise to the training set. It was observed that previous conclusions hold in this case as well despite it being theoretically favorable to use an outlier detection mechanism to denoise the data. Finally, the effects of BDR were examined on deep convolutional networks used for Image classification. The interpretability of the image classification task allowed us to better analyse insights on the systems performance.

Three issues were pointed out in the application of BDR. Firstly, the starting point of BDR is a critical hyperparameter and there is no universal mechanism to set this. While [3] prescribes 200-500 as a general guideline, in practice the actual start point is heavily dependant on other parameters such as learning rate and optimiser. Secondly it is observed that ANN learns to classify each of the classes at different rates. Applying BDR on the whole train will cause loss of useful patterns that are not yet learnt from the tougher classes. It would be more beneficial to have a per class outlier detection mechanism with individual starting points for each. Thirdly, it was observed that in general BDR serves to reinforce the networks learnt patterns, by removing the patterns that it finds hard to classify. This is evidenced by sharp decrease in training error variances and increase in training accuracy after BDR is applied. This will be beneficial only after the model has successfully generalised the problem. This further emphasises the need to have a more accurately chosen starting epoch.

## 6 Future Work

Further inspection needs to be done to find cases where BDR is beneficial to an ANN classification problem. After identifying this, constructing a function that dynamically calculates problem specific start epochs would greatly help the application of BDR to varied real world problems. It also seems to be that distance based outlier detection may be more suited to the classification problem than distribution based methods. This intuition comes from examining Fig. 1 and Fig. 2, which show that classification problems tend to reduce variance quite rapidly. Therefore, error distribution based outlier detection mechanisms may not be as useful. It would be interesting to explore alternative distance based outlier detection mechanisms for this problem. It would then be useful to examine performance of a per class outlier detection mechanism vs a general outlier detection mechanism.

## References


1. D. M. Hawkins. Identification of outliers Chapman and Hall, London, 1980
2. Hawkins, S., He, H., Williams, G. and Baxter, R., 2002, September. Outlier detection using replicator neural networks. In *International Conference on Data Warehousing and Knowledge Discovery* (pp. 170-180). Springer, Berlin, Heidelberg.
3. Slade, P. and Gedeon, T.D., 1993, June. Bimodal distribution removal. In *International Workshop on Artificial Neural Networks* (pp. 249-254). Springer, Berlin, Heidelberg.
4. Fanty, M. and Cole, R., 1991. Spoken letter recognition. In *Advances in Neural Information Processing Systems* (pp. 220-226).
5. Blackard, J.A. and Dean, D.J., 1999. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Computers and electronics in agriculture*, *24*(3), pp.131-151.
6. Milne, L.K., Gedeon, T.D. and Skidmore, A.K., 1995. Classifying Dry Sclerophyll Forest from Augmented Satellite Data: Comparing Neural Network, Decision Tree & Maximum Likelihood. In *Proceedings Australian Conference on Neural Networks* (pp. 160-163).
7. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), pp.1929-1958.
8. Garris, M.D., Blue, J.L. and Candela, G.T., 1997. NIST form-based handprint recognition system (release 2.0).