NEUROCOMPUTATION CONFERENCE PAPER

Module Code: CS2NC19

Student Number: 26019697

Actual Hours Spent For Assignment: 50

February 27th 2020

Reading Assignment Evaluation

- Struggled at first to understand how to choose a dataset and apply it to the neural network, spent a lot of time researching before finally deciding on a classification problem to use.
- Completed a lot of experiments but found difficulty in deciding which were best – eventually settle on using the highest unseen classification rate, but this meant that I excluded a lot of results.
- Overall learnt a lot, but wish that there was a little bit more guidance on what to do – had to do a lot of research, but the ANN we were using validation data sets of which I could find less information about than desired.

TRAINING A MULTI LAYER ARTIFICIAL NEURAL NETWORK WITH SIGMOIDAL ACTIVATION

26019697

ABSTRACT

Over 400 million people globally are affected by diabetes, a serious and large issue which takes a lot of resources to diagnose and develop appropriate treatment for. This paper experiments and discusses whether a multilayer artificial neural network (ANN) can learn a classification problem with 8 data inputs and a binary output. 105 experiments were performed with the highest correct classification rate of 90% on an unseen dataset achieved. The ANN that has been used was programmed in Java and takes advantage of object-oriented principles to complete learning. Additionally, the ANN features both a validation set to stop learning when it peaks and SAM (simple adaptive momentum) which is used to accelerate the learning process.

1. INTRODUCTION

The following experiments document the training of an ANN, using a dataset found on a opensource resource page called machinelearningmastery.com. This dataset contains information about a sample of Pima Indian population and 8 different factors; (number of pregnancies, plasma glucose concentration, diastolic blood pressure (mm Hg), triceps skinfold thickness (mm), 2 hour serum insulin (mm), BMI (weight in kg/(height in m)^2), diabetes pedigree function, age (years) and a binary classification (0 or 1) of whether the subject was diagnosed with diabetes or not.

The ANN that has been created is written in Java, using a heavily object-oriented structure in order to create a multi-layer perceptron network which uses backward propagation in order to process the error backwards and hence supervise learning and create a model which most appropriately maps a set of inputs to their correct output. The network includes training, unseen and specially a validation partition of the dataset which is used to stop learning once the validation sets SSE (sum of the squared of the errors) has begun to increase over 10 epochs, this is discussed in [5]. Additionally, the neural network features optional SAM which has been shown to increase learning rate (number of epochs), although may have some impact on accuracy of the result as discussed in [4].

The ANN has been trained in various experiments in order to find an optimal solution. A combination of different pre-processing, learning rates, starting weights, momentum, number of epochs (learning cycles) and size of the neural net (number of neurons) have been used. By changing these parameters, different final weights within the neural network have been achieved, and hence different solutions have been crafted to this classification problem.

2. CHOICE OF DATASET

Choosing an appropriate dataset is very important when attempting to train a neural network. Due to limitations of the system being used there were restrictions on the maximum size of the dataset that could be chosen.

2.1. Using a large dataset

A small dataset was used in order to reduce the large processing time of the neural network on the system used for the following experiments, especially when training using many neurons and a very small learning rate. If using a large dataset when trying to experiment the data would often take extensive period to train the model, this is due to many calculations (as discussed in [1]).

2.2. Using a small dataset

When using a small dataset, problems such as overfitting can occur, this is due to models that are produced being more easily able to fit into a smaller set of data points. This can cause the model to be seem accurate even when it is not – this is since it can create a solution which perfectly fits the small amount of data but does not create a good model for all data as seen in the image below.

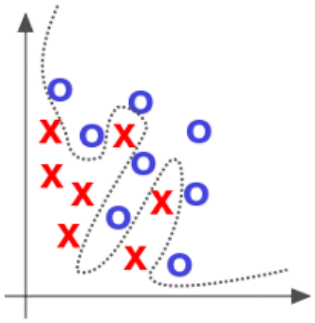

Figure 1 Overfitting in a small dataset (image courtesy of [3])

2.3. The chosen dataset

Due to the above reasons a dataset with 768 data points has been selected as this is enough data for a network to learn from without taking too much processing time and hence reducing the number of experiments able to be attempted. This can limit the effectiveness of the entire process as there would be a restriction on the number of times the network could be trained in different ways which would reduce the number of attempts to find an optimal solution.

Additionally, this dataset was chosen as it is a classification problem with a binary output meaning it is

easier to interpret the results of the problem, and hence interpret the effectiveness of training.

3. DATA PRE-PROCESSING

When machine learning, data pre-processing is very important as this will have a large impact on how the data is learnt by the ANN.

There are multiple different ways of pre-processing data, all with different studies and supporting different methods of pre-processing. However generally agreed, one of the most important things to deal with when processing data for machine learning is dealing with null values. There are a few different ways to deal with null values, these include ignoring them (if there are very few), removing rows containing the nulls values or replacing them with the mean values of the column. The following experiments document all three of the previous methods.

Another major part of data pre-processing data is deciding on how to split the dataset into training, validation and unseen partitions. In order to do this effectively one must have a training set which is large enough to train the dataset, a validation and unseen set which is a not too big and not too small in order to make sure that is the learning progress stops at the correct time when finding the global minimum for the problem.

The data that was pre-processed here has all been done with Python scripts, dividing the data randomly as described at the start of each section to ensure that all the experiments are fair.

There are other forms of pre-processing which must be acknowledged, for example data scaling, data normalisation and binarizing data, all these methods can be used to help with the processing of data, and if more experiments were to be carried out, these methods of processing would be implemented to add more variation.

4. EXPERIMENTATION

4.1. Experiments using raw data

The first experiments were attempted with the mostly untouched data, which had been split into 3 sets equally and put in the correct format to be applied to the neural network. Additionally, the null values from the dataset have all been set to 0, this allows all the data points to be processed

The best results that were gained from these experiments are given in a table below. The table shows 3 different splits of the data using an algorithm written in python designed to split the data into exact thirds – or at least as close as possible. There are 768 data points in the dataset meaning each of the unseen training and validation datasets have a size of 256 points.

The experiments are documented with the following keys: E is the number of epochs taken to learn the problem, HN is the number of hidden neurons in the network, MMT is the simple adaptive momentum, Seed is the value provided to randomly set the weights at the start of the training process. And NE is the number of experiments attempted. The tables show the SSE's and percentage correctly classified at start at end, being presented from the 'best' model achieved with the highest percentage classification correct on the unseen dataset. Where the percentage correctly predicted is the same, the lowest SSE was chosen to be the 'best' result.

Data Set	Train SSE	Unseen SSE	Valid SSE
1 E: 150	0.2627 38%	0.2998 28%	0.2998 28%
	0.1043 75%	0.1019 78%	0.1019 78%
HN: 10 LR: 0.10 MMT 0.00 Seed 100 NE: 10			
2 E: 1770	0.2297 36%	0.2381 31%	0.2381 31%
	0.0980 77%	0.0904 80%	0.0904 80%
HN: 15 LR 0.01 MMT 0.10 Seed 1000 NE: 10			
3 E: 270	0.2600 41%	0.2930 32%	0.2930 32%
	0.0938 80%	0.0989 77%	0.989 77%
HN: 10 LR: 0.01 MMT: 0.00 Seed: 1000 NE:10			

Figure 2. Best solutions found from first 3 training sets

The results on this unprocessed data are generally good – with an accuracy rate of 80% achieved on the unseen dataset for the best model that was created, up from an original 31% before the training started. As we can see the neural network has certainly achieved significant learning, it can now predict whether a subject is diabetes positive with an 80% accuracy rate.

Another interesting observation about these 'best' results is the fact that they have either no or very little momentum, this agrees with [4], suggesting that while the momentum speeds up the learning process it can create some reduction in accuracy in the neural network as the solution given is moved away from the true global minima by the momentum which causes is to learn 'too much' and hence overshot a solution.

4.2. Experiments with null values removed

For the second set of experiments, different pre-processing was used - the dataset was further pre-processed than before by removing rows with null values. Additionally, the dataset has been split differently, allocating 80% of the set to training and 10% for the validation and unseen sets. This will remove all the null values which were replaced previously with 0's which would have created a large number of outliers, and also provide a larger portion of the dataset for training which would allow the dataset to be trained more effectively and

not be affected by outlying null values which had been replaced with 0. This is further explained in [6].

Data Set	Train SSE	Unseen SSE	Valid SSE
1 E: 3630	0.2233 66%	0.2384 64%	0.1849 71%
	0.0801 82%	0.0926 89%	0.0773 76%
HN: 20 LR: 0.02 MMT: 0.10 Seed: 0 NE: 10			
2 E: 3160	0.2244 28%	0.2246 35%	0.2440 20%
	0.0749 84%	0.1060 79%	0.0444 89%
HN: 15 LR 0.01 MMT 0.50 Seed 1000 NE: 10			
3 E: 40	0.3050 32%	0.2518 43%	0.3073 30%
	0.0885 78%	0.1077 79%	0.0888 82%
HN: 10 LR: 0.10 MMT 0.00 Seed: 100 NE: 10			

Figure 3. Best solutions over second 3 data sets

Using the pre-processed data we can see that the optimal solution achieved had a 89% correct prediction accuracy on the unseen dataset, this significantly higher than the previous set, reducing the error rate from 20% to 11%, this shows that by using pre-processing to prepare the data in a more logical way, the model produced has a far higher accuracy rate. This shows that pre-processing data is very important and shows that experimentation with different forms of pre-processing to create the best model is a key factor in obtaining a good model which has learnt the problem well.

Additionally, over all the experiments and datasets the lowest percentage correct achieved was 74%, this was higher than the experiments in 4.1 where the lowest unseen classification percentage was 69%, this shows that by removing some of the noise, in this case null values, having a higher minimum and maximum classification percentage allows use to see that by removing the outliers (null values represented by 0) the data has become easier to model and both the worst and best models are better.

Another interesting observation is that the best results achieved were the results which had the least level of improvement – the first data set had only improvements of 5-25%, this shows that seed (the starting weights) used in experiments are very important, showing how well a problem can be learnt by a neural network. When choosing weights which are nearer to local or global minima's, this can cause the neural network to tend toward one solution or another. However, the only way to know which weights are going to create a good solution is through trial and error. Hence in future experiments, performing repeated experiments with varied start weights could help in finding better solutions to the problem. This is because the starting weights being closer to a minimum point in space which would mean that learning would tend towards this minimum point and a better solution would be more easily achieved.

4.3. Experiments with null values replaced with mean

A final time the dataset was prepared by replacing all the null values with the average value from their column, this keeps the average value for each column the same. Additionally, this allows the whole dataset to be used for the 3 partitions, this is because null values were not dropped. This data was split in the same way as previously (80% training and 10% unseen and validation) as this would allow for all the data to be passed to the neural network, this should hopefully increase the accuracy of the model produced as a larger training set can be used which should create a better model for the network to learn from.

Additionally for the following data partitions 15 different experiments were performed – with 5 experiments using random seeds selected by using a random number generator between 0 and 1000, the idea of this was to create truly random start weights which would mean that the start points for learning would have a higher potential to be closer to 'better' solutions.

Data Set	Train SSE	Unseen SSE	Valid SSE
1 E: 3910	0.2976 33%	0.2801 37%	0.2603 41%
	0.0825 80%	0.0877 81%	0.1262 70%
HN: 10 LR: 0.01 MMT: 0.00 Seed: 100 NE: 15			
2 E: 30	0.2860 36%	0.3494 18%	0.2841 36%
	0.0985 76%	0.0846 85%	0.0978 77%
HN: 10 LR 0.10 MMT 0.10 Seed 100 NE: 15			
3 E: 30	0.2911 34%	0.3122 31%	0.2805 36%
	0.1015 76%	0.0735 90%	0.01143 70%
HN: 10 LR: 0.10 MMT 0.20 Seed: 55 NE: 15			

Figure 4. Best solutions over final 3 data sets

The final three data partitions that were used show an increase in percentage correctly classified and they also show an average increase in classification of the unseen data of 3%, meaning that overall there are improvements on all the results. However, interestingly, the prediction rates on the validation and training sets are much lower with an average decrease across all three sets of 4%, this implies that the models may not be as good as expected – potentially they are just very good at predicting the values within the unseen dataset, however they may not work as well on an actual larger dataset. In order to further this , more experiments could be carried out in

order to determine whether replacing the null values does or doesn't increase overall accuracy.

The highest result yet of a 90% accuracy rate was achieved on the unseen dataset, however this may not be truly accurate and could just be coincidental, as the training and unseen classification percentages are 76% and 70% respectively, showing that the neural network has perhaps not learnt the whole problem effectively.

Finally, from the results we can see that the lowest SSE has some relation but is not directly proportionate to highest percentage correctly classified was above in figure 3, this shows that the SSE shows how much 'less wrong' the data is by having a lower value, not technically meaning how correct the data is. The implications of this are peculiar – by trying to lower the SSE in training the best percentage correctly classified may not be achieved as the percentage correct is binary, either right or wrong, however the SSEs are numerical and can see how 'close' they are to the correct results. While this is an interesting point, there is nothing that can be changed without creating an entirely new neural network which learnt based on percentage correctly classified rather than SSE This leads us to see that there are perhaps improvements to the neural network that could be implemented.

4.4. Further experiments

Due to limitations on both time, processing power and a lack of automated experiments, less experiments that desired were able to be carried out. A total of 105 experiments were carried out during this paper with the results from the best 9 unseen classifications being displayed.

If there was more time available, further experimentation could be carried out. During this experimentation there would be specific areas to focus on. Firstly, there would need to be focus on repeating experiments with different start weights on the neurons (documented as seed value in the tables), this would mean that for each data set used there would be more experiments performed with different start points and hence more chance of a true global minimum being achieved.

Additionally, if more experiments were attempted then different partitions of the data would be used as this has created different models as we can see previously, hence by creating different partitions of the data we would be able to find different solutions to the problem.

5. FALSE POSITIVES AND FALSE NEGATIVES

When dealing with an issue as serious as diabetes, considering false positives and false negatives is very important. False positives could lead to people without diabetes getting a result that they were diabetic, which without any further diagnosis could be rather dangerous, and vice versa for false negatives which arguably could be worse with people who are diabetic being classed as non-diabetic.

The implications of this are that an exceptionally high percentage classification would be required by any model for it to have practical applications as if the classification was too low then every case would be have to be verified by an official diagnosis as the model used is not accurate enough.

6. CONCLUSION

Overall the neural network has learnt the problem to a high level, with a 90% accuracy rate on an unseen dataset being the highest rate achieved. This is a good classification rate, and shows that the network has been learning well, however due to the application of the problem, this is not a good enough rate to be used in diagnosing diabetes as a wrong diagnosis could have drastic impacts on an individual's life. However, further experiments with different partitions and pre-processing could improve the accuracy and comparing it to the accuracy of actual diagnosis could result in a more effective and better model which could potentially be implemented in the diagnosis of diabetes.

7. REFERENCES

[1] N. Kempton "Why is type 1 diabetes diagnosed" https://beyondtype1.org/why-is-type-1-diabetes-misdiagnosed/ (2018)

[2] M. Stewart "Handling Big Datasets For Machine Learning" https://towardsdatascience.com/machine-learning-with-big-data-86bcb39f2f0b (2019)

[3] R. Alencar "Dealing with very small datasets" https://www.kaggle.com/rafjaa/dealing-with-very-small-datasets (2019)

[4] R. J. Mitchell "On Simple Adaptive Momentum" (2008)

[5] J. Brownlee "What is the Difference Between Test and Validation Datasets? https://machinelearningmastery.com/difference-test-validation-datasets/ (2017)

[6] B. Angelov "Working with missing data in machine learning" https://towardsdatascience.com/working-with-missing-data-in-machine-learning-9c0a430df4ce (2017)