IST-Africa 2018 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2018
ISBN: 978-1-905824-60-1

ANN-MIND: A Comparative Study on the Training of Neural Networks with Incomplete Datasets

Tshilidzi MUDAU, Duncan COULTER
University of Johannesburg, Kingsway and University Road, Auckland Park, Johannesburg, 2092, South Africa
Tel: +27 716258789, Email: mudau7t@gmail.com, dcoulter@uj.ac.za

Abstract: The quality of a dataset plays a central role in the results and conclusion that can be drawn from analysis such a dataset. As it is often said; garbage in, garbage out. In recently years, neural networks have displayed good performance in solving a diverse number of problems. Unfortunately, artificial neural networks are not immune to this misfortune presented by missing values. Furthermore, in most real word settings, it is often the case that, the only data available for the training of artificial neural networks consists of a significant amount of missing values. In such cases, we are left with little choice but to use this data for the purposes of training neural networks, although doing so may result in a poorly performing neural network. In this paper, we describe the use of neural network dropout as a technique for training neural networks in the presence of missing values. We test the performance of different neural network architectures on different levels of artificial generated missing values introduces on the MNIST handwriting recognition dataset, Cifar-10 and the Pima Indians Diabetes Dataset and find that in most cases it results in significantly better performance of the neural network compared to other missing data handling techniques.

Keywords: Artificial neural networks, neural network dropout, missing data.

1. Introduction

For one reason or another, some records in a database might contain missing fields. A malfunctioning data collection device might record some pieces of data and not others. In a questionnaire, for one reason or another, some applicants might choose to not answer some of the question. This missing information is what we call missing data. Regardless of the reason(s) that lead to the missing data, missing data is a common occurrence in real life settings and it plays a crucial role in the quality of machine learning models built on top of this datasets. Most systems currently in use, merely discard the missing observation from the training datasets, while others just proceed to use this data and ignore the problems presented by the missing values. Still other approaches choose to impute this missing value with fixed constants such as means and mode.

Like most machine learning algorithms, most neural network architectures work under the assumption that the supplied data contains no missing values. Whist neural networks are generally known to be resilient to noise, in the presence of a significant amount of missing values, this resilience soon fades away [1] and [2]. This dissertation explores a method for training neural networks in the event where the training dataset consists missing values. Its contribution is the introduction of four new datasets containing artificial generated missing values and the proposed use of neural network dropout (a widely used regularisation technique) as a technique for handling missing values. We refer to this use of neural network dropout as Artificial Neural Network Missing INputs Dropout (ANN-MIND).

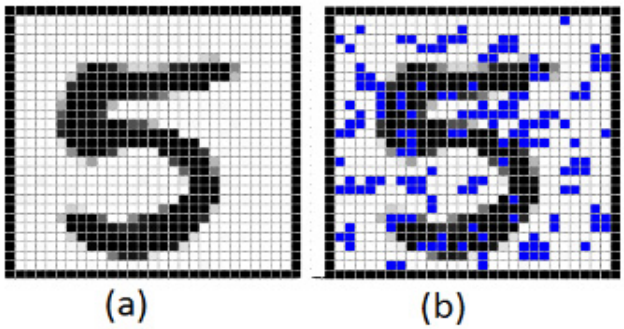
2. Objectives

The objective of this paper is to answer the following questions:

1. Can ANNs be successfully trained in the presence of a significant amount of missing value, that is; a high quality of missing data which cannot be discarded or replaced with random values and still yield good accuracy (as compared to benchmark accuracy for the dataset in question) for an ANN?
2. Is it possible to train ANNs in the presence of missing data without risking introducing noise?
3. Is it possible to put the task of dealing with missing values on the hands of ANNs and thus eliminate an intermediate step which handles missing data?
4. Does a ANN produce different levels of accuracy depending on the type of missing value imputation technique used? If so, which imputation method yields better accuracy?
5. How can ANNs be trained in the presence of missing values in such a way that no additional noise is introduced in the form of incorrect prediction or estimation of the missing value(s)?

3. Methodology

Pixels missing completely at random: The first method we used to introduce missing pixels into random MNIST images was to draw a random sample of images from the original dataset and remove pixels using a random Gaussian sampling. Figure 1 shows an image picked at random from the dataset followed by 20% Gaussian random pixel removal. In figure 1 removed/missing pixels are represented by blue pixels, however in code these pixels are represented by a placeholder NaN. Furthermore, recall that MNIST images are grey-scaled, thus the blue colour is for purely visual purposes and not an indication that the image has three channels(Red, Green, Blue).



(a)An image of a handwritten digit 5 from the MNIST dataset. (b) The image shown in (a) with 20% of its pixels selected via a random Gaussian selection and removed.

Data in real world settings is rarely missing completely at random. Furthermore, according to [18], if a dataset that is missing completely at random then it meets the criteria minimum criteria in [18] for the dataset to use to predict the missing values within the dataset. For this reason, we decided to try another form of missingness as well as missing completely at random (see figure 2). We refer to this method as the localized missing pixels method.

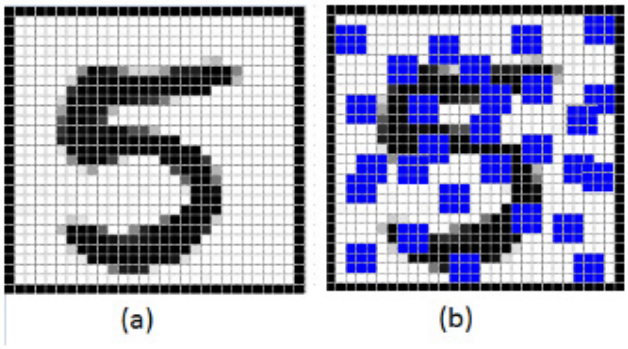


Figure 2: (a) A random image selected from the MNIST dataset. (b) The image in (a) after a localized imputation at 10% probability of a pixel being selected and neighbourhood radius r=2.

In this method, pixels are selected at random using via a random Gaussian distribution. Once a pixel p has been selected, a neighbourhood of radius of pixels is built around p. Finally, all pixels within the radius (p included) are discarded, and flagged as missing (i.e NaN). The resulting image is similar to that shown in figure 2 (b) where the missing pixels are shown in blue. In figure 2 (b) r=2 and the probability of a pixel being selected for such removal was set at 10%.

[3] used a combination of K-Nearest neighbour and a NN to predict missing values within datasets. Auto encoders have also been used to create a dimensionality reduced representation of the input data consisting of missing values such works include [5], producing a complete representation of an otherwise missing input data which is then used for training subsequent models/algorithms. A similar technique relying on neighbourhood was presented in [5] where instead of a K-nearest neighbour, a shell neighbour is used instead. By combining several standard models generally used with complete data, (Marlin, 2008) built a collaborative predicting for predicting non-random missing values. [4] trained a feature extracting unsupervised NN to learn important features of datasets consisting of missing values and used this features as inputs into a swarm intelligence algorithm which the authors used to predict the missing values. [9] used genetic algorithm to predict and impute missing values within datasets, whilst [10] used this same algorithm (genetic algorithm) in combination with an ANN, a principle component analysis (PCA). What makes ANN-MIND unique is that it does not intend to predict the missing values, nor do we intend to eliminate all missing values before passing them through to the neural network.

Neural network dropout [13], a regularization technique used for training large/deep neural networks. Neural network dropout (or simply dropout) is a regularisation technique used for regularising ANNs. It reduces the complexity network which in turn reduces the likelihood of the network over-fitting. According to [13], [18], the use of dropout on a network reduces the likelihood of neurons co-adaption in the network, a characteristic that is often associated with over-fitting. NN dropout works but randomly selecting and discarding this neuron from the network during the feed-forward step of the NN training. Furthermore, neurons dropped out on a given iteration of the mini-batches or feed-forwards step also seize to become part of the corresponding back-propagation step. This process is repeated over all samples in the training set, see figure 3 for an example of NN dropout on a hypothetical run of the feed-forward step. The random discarding of neurons forces the network to learn to generalise with only a subset of its overall neurons present, see figure 3. Since every training example pushed to the network is trained with a usually a different sub-network of the overall network, the end results are that the final trained NN is in effect an ensemble of this sub-network.

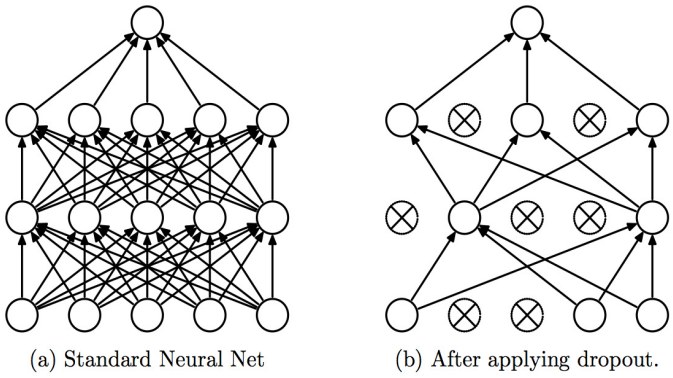


Figure 3: (a) A standard neural network. (b) Neural network dropout applied on the network shown in (a).

According to [17], dropout has the same effect as training multiple NNs and creating an ensemble (see section below) with them. Since [13] first proposed dropout, there have been several variations proposed, this includes see fast dropout [14] and drop-connect [15] works the same as dropout however, weight connecting to other neurons are dropped instead of the entire neuron.

4. Developments

The model we propose in this paper(ANN-MIND) is to introduce a neural network dropout layer in place of the normal input layers. However, this dropout layer drops out only those neurons corresponding to the missing values. When an input vector which possibly contains some observations that have missing values, ANN-MIND applies a neural network dropout on all the elements in the vector that are missing. This is a one to one, direct mapping from the input vector to the layer directly after the ANN-MIND layer with the only difference being that all the missing inputs are dropped out from the next layer(see figure 4).

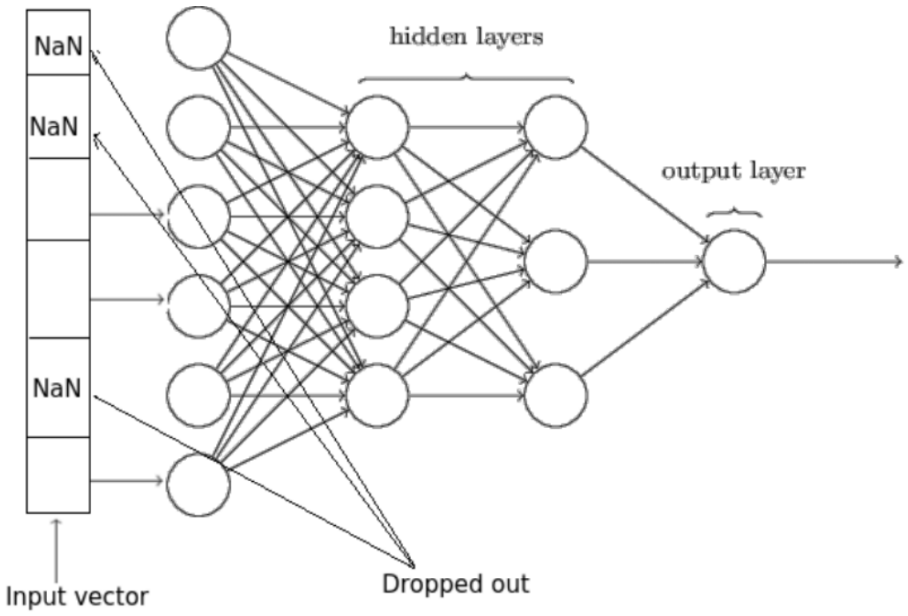


Figure 4: An example showing ANN-MIND applied on missing elements (denoted by NaN) on input vector.

The introduction of ANN-MIND in the network doesn't result in any constrain in as far as how the resulting network is/can be constructed. The construction of the network is

carried forward as it would be if the first layer of the network where the dropout layer. The algorithm for ANN-MIND is as shown figure 5 below.

Algorithm 2: Training a NN using MissedOut model

Input : A dataset D of N observations(length L) some of which consists of missing values(labelled as NaN)

Output: Weights $W_{learned}$ learned by the NN

```
while Not finished training do
    for i = 0 N do
        for j = 0 N do
            if D[i][j] = NaN then
                Dropout input D[i][j]
            end
            Feed D[i][j] to the first next layer of the network
        end
    end
end
return W_learned
```

Figure 5: The ANN-MIND algorithm.

5. Results

The simplest of our architectures is a shallow network consisting of only an input layer followed by one hidden layer with 784 neurons and an output layer. Figure 6 shows the learning curve of the small network described above when trained on MNIST dataset without any missing values. The figure indicates that the model reaches an accuracy of close to 99% without any sign of overfitting.

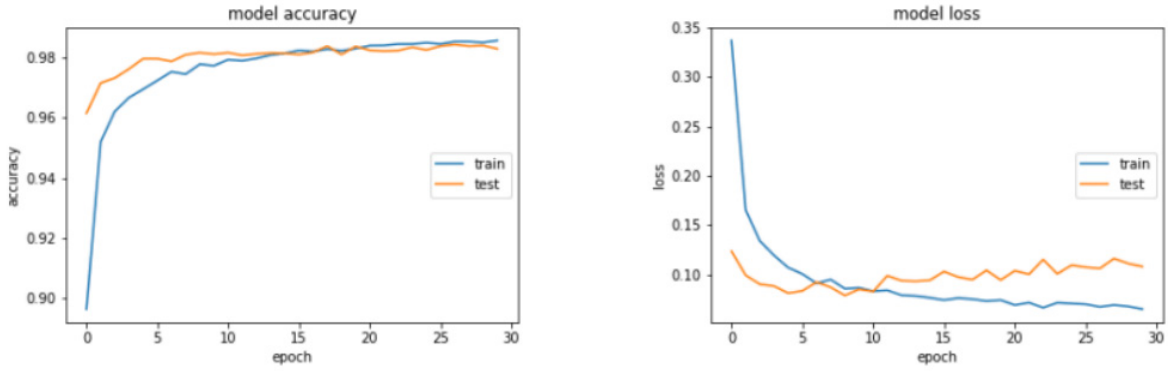


Figure 6: On the left is the accuracy observed during the training the net-work discussed above for 30 epochs. The networks achieve a test accuracy of 0.9829. The loss observed during the training of the same network for 30 epochs. The networks achieve a corresponding test loss of 0.10797932473.

Upon increasing the number of missing values from 20% to 35%, we observe that the neural network fails to learn completely, with both the training and the validation loss remaining constant at 0.00020 for the entire 60 epochs (see figure 7). We attribute this to the small training dataset left after we performed listwise deletion on all the missing data.

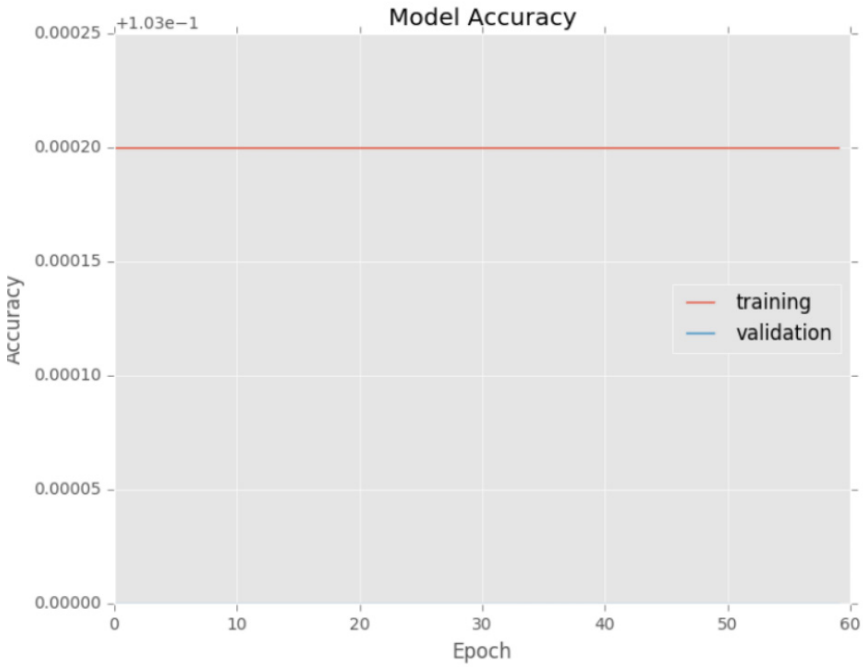


Figure 7: A shallow neural network trained on MNIST dataset with 20% of the pixels randomly removed from every image. This network shows clear signs of under-fitting indicated by the validation loss and the training loss which remain constant at0.0020 over the course of the entire 60 epochs.

This indicates that 20% pixels selected at random on the MNIST dataset results in the removal of a significant amount of missing values. As such further experiments were carried at 20% missing values. Figure 8 below shows the performance of ANN-MIND on MNIST with 20% of the pixels on every image missing via Localised missing scheme discussed previously.

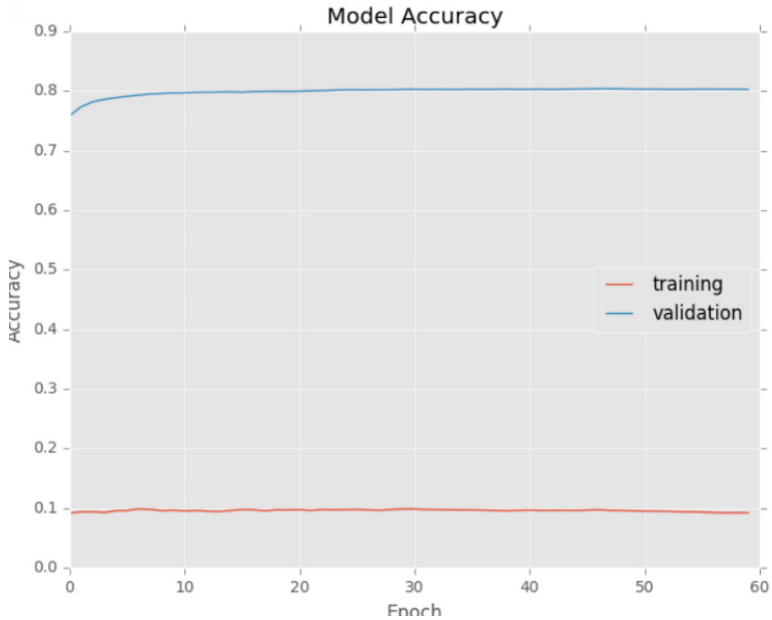


Figure 8: A shallow neural network trained using ANN-MIND on MNIST dataset with 20% of the pixels removed completely using localized missing value removal.

From the figure above we observe that the model performs well even in the presence of such a high amount of missing values. The constant training rate is the result of the high dropout rate being carried out by ANN-MIND on the 20% missing pixels.

6. Business Benefits

Missing data is a common problem in real world setting (including in business). Following the success of neural networks in solving a wide variety of business problems, neural networks are getting more and more used to solve business problems. This brings up the need for ways in which neural networks can be trained in solving business problems in those cases where the only data available consists of a significant amount of missing values.

7. Conclusions

This paper discussed results observed when training numerous neural network models using ANN-MIND. The results showed that compared to other techniques we looked at, ANN-MIND performs well (in terms of accuracy) in the presence of missing values. Furthermore, we saw that in the presence of no missing data, ANN-MIND performs just as well as a similar network without ANN-MIND applied. The model we proposed in this paper showed impressive performance in the cases where the data at our disposal was missing. The model showed impressive performance on both the MNIST dataset (in which we artificially created missing data) and the mushroom edibility classification dataset (which has missing values in its natural state and as such didn't require any artificial data manipulation). Over the cause of conducting this research study, the author found that, whilst the presence of missing data is a norm in industry, finding a public dataset that has a lot of missing values is a time-consuming process. The absence of a standardised dataset with which any proposed missing data model's efficiency could be verified against means that measuring the efficiency of a model proposed by other researchers is extremely difficult.

To compare the efficiency of a new model proposed, a researcher would have to compare his/her model against the same dataset used by other researchers. However, since there is no standardised dataset for evaluating the efficiency of an algorithm on dealing with missing values, different researchers use different datasets. To this end, the author hopes that in future such a standardised dataset will be made available. The author hopes that introduction of such a data could have a similar impact to that which the ImageNet dataset (see [19]) had in the field of computer vision. The introduction of the [19] lead many researchers and industry alike to evaluate the efficiency of their proposed computer vision model on a single dataset and compare how their model(s) fared to those of other researchers. We also performed what we feel to be good analysis and gave results supported by statistical significant results. It is the wish of the author that the model should be tested on more datasets, on more datasets consisting of missing values which have not been artificially generated as is the case for our MNIST dataset. As part of the author's future work, the author hopes to collect a large dataset of missing values which the author hopes could be used as a dataset for evaluating the efficiency of algorithms that predicts or learns on missing data. The motivation for this is as started above; to provide a standardised dataset which researchers in this area can all use as a benchmark for their models. The introduction of such a dataset would reduce the need to test one's model on multiple datasets.

References

[1] Brunel, N. and Hansel, D. (2006). How Noise Affects the Synchronization Properties of Recurrent Networks of Inhibitory Neurons. Neural Computation, 18(5), pp.1066-1110.

[2] Dodge, Samuel, and Lina Karam. "Understanding how image quality affects deep neural networks." Quality of Multimedia Experience (QoMEX), 2016 Eighth International Conference on. IEEE, 2016.

[3] Aydilek, I. B. and Arslan, A. (2012). A novel hybrid approach to estimating missing values in databases using k-nearest neighbors and neural networks. International Journal of Innovative Computing, Information and Control, 7(8):4705–4717.

[4] Leke, C., Twala, B., and Marwala, T. (2014). Modeling of missing data prediction: Computational intelligence and optimization algorithms. In Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on, pages 1400–1404. IEEE.
[5] Zhang, S. (2011). Shell-neighbor method and its application in missing data imputation. Applied Intelligence, 35(1):123–133.
[6] Marlin, B. M. (2008). Missing data problems in machine learning. PhD thesis, University of Toronto.
[7] Leke, C. and Marwala, T. (2016). Missing data estimation in high-dimensional datasets: A swarm intelligence-deep neural network approach. In International Conference in Swarm Intelligence, pages 259–270. Springer.
[8] Little, R. J. and Rubin, D. B. (2014). Statistical analysis with missing data. John Wiley & Sons.
[9] Abdella, M. and Marwala, T. (2005). The use of genetic algorithms and neural networks to approximate missing data in database. In Computational Cybernetics, 2005. ICCC 2005. IEEE 3rd International Conference on, pages 207–212. IEEE.
[10] Mistry, F. J., Nelwamondo, F. V., and Marwala, T. (2009). Missing data estimation using principle component analysis and autoassociative neural networks. Journal of Systemics, Cybernatics and Informatics, 7(3):72–79.
[11] Little, R. J. and Rubin, D. B. (2014). Statistical analysis with missing data. John Wiley & Sons.
[12] Kalaycioglu, O., Copas, A., King, M., and Omar, R. Z. (2015). A comparison of multiple-imputation methods for handling missing data in repeated measurements observational studies. Journal of the Royal Statistical Society: Series A (Statistics in Society).
[13] Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015). Highway networks. ArXiv preprint arXiv:1505.00387.
[9] Warde-Farley, D., Goodfellow, I. J., Courville, A., and Bengio, Y. (2013). An empirical analysis of dropout in piecewise linear networks. arXiv preprint arXiv:1312.6197.
[15] Wang, S., & Manning, C. (2013). Fast dropout training. In Proceedings of the 30th International Conference on Machine Learning (ICML-13) (pp. 118-126).
[16] Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., & Fergus, R. (2013). Regularization of neural networks using dropconnect. In Proceedings of the 30th international conference on machine learning (ICML-13) (pp. 1058-1066).
[17] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012b). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
[18] Imhof, L. A., Song, D., and Wong, W. K. (2002). Optimal design of experiments with possibly failing trials. Statistica Sinica, pages 1145–1155.
[19] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009b). ImageNet: ALarge-Scale Hierarchical Image Database. InCVPR09.