12-12-2018

Neural Network Fraud Detection Dilemma: The Curious Role of Activation Functions

J Locke
Auburn University, jml0062@tigermail.auburn.edu

Jay Claiborne
Auburn University, jzc0019@tigermail.auburn.edu

Neural Network Fraud Detection Dilemma: The Curious Role of Activation Functions

Research-in-Progress

J. Locke
Auburn University
jml0062@auburn.edu

J. Claiborne
Auburn University
jzc0019@auburn.edu

Abstract

Neural networks are excellent candidates for uncovering fraudulent transactions and have been proven effective by credit card companies, banks, large retailers and other organizations dependent on large numbers of transactions and structured data for daily business activities. Neural networks were born in theory as computing power was insufficient for demonstrating the techniques. With the proliferation of graphical processing units (GPU) for processing and the connectivity of the Internet for rapidly gathering and sharing data, the promise of quickly responding to criminal and petty fraud activity with computer power is apparent. Now that we may routinely use these powerful tools it makes sense to test them, to look under the hood and see what is going on with the myriad of math processes and connections that are augmenting our decisions. This paper looks at fraud in the public transit sector, a massive generator of transaction data and a frequent target of fraud, and investigates how well neural network activation functions, critical to automating learning and predicting, perform in identifying fraud and suggests research and teaching avenues for management information systems (MIS) researchers and academics to consider.

Keywords

Fraud, Neural Network, Activation Function, Node, Back Propagation

Introduction

Neural networks are commonly used in efforts to identify transaction fraud. The available data for particular transaction domains is typically promising and reliable for artificial intelligence (AI) classifiers. The data sets are normally very large as measured in numbers of individual transactions. Data set growth parallels the normal progress of business. The number of features relative to transactions are normally few and fall safely within classifier model design norms (Beleites, et.al, 2015). Large repositories of data are available for training models, verifying validity and testing for accuracy. Transactions can be tested automatically and in real time as business activities progress throughout transit activity periods. New transaction data becomes available daily for continually updating the model with new indicators of fraudulent activity and trends. Overall, the combination of data and model seem to offer an excellent platform for implementing AI for the purpose of fraudulent transaction identification.

Applying AI to fraud identification problems, while common, is not without inherent problems. Though promising, and often thought of as a reliable "black box" solution, AI is still essentially a statistical exercise, albeit a sophisticated one (Hutton, L. 1992). AI models are still affected by the various challenges of statistical models such as autocorrelation, confounding variables and missing or corrupt data. Thus, awareness and competence in statistical design is crucial for building reliable AI classifiers. Likewise, it is crucial to understand that building the AI algorithm itself relies on human skill. Successful and reliable AI modeling is a labor-intensive activity that requires deep AI expertise (Wistuba, 2018). The structured data used for fraud identification avoids certain difficulties typical of many AI models; that is, extracting usable high-level features from raw data (Goodfellow, Bengio and Courville, 2016). Financial transaction data typically requires little pre-processing compared to, for example, the abstract representations of image or construct classifiers. Adding to the mix that must be considered for AI fraud identification modeling is the balanced data problem. While it is generally regarded as sound method by practitioners to train AI models using balanced data—in this case a balance between fraudulent and non-fraudulent transactions—there is

strong theoretical support to construct training models with unbalanced data that may indeed lead to models that are better at generalizing (Murphey, Guo & Feldkamp, 2004).

Data Environment

For high-volume low product cost transactions, such as those common to the transit industry as studied herein, the pattern of settlement transaction per activity can mask fraud identification until well after the activity has concluded. Too, the currencies used are often in the form of tokens or some form of swipe card; thus, offering unique avenues to fraud by those inclined to profit from such behavior. These data characteristics can offer not only an excellent understanding of customer behavior and preference trends but also provide valuable insight into fraudulent behaviors, techniques and sophistication. This insight, along with timely identification of fraudulent transactions, becomes critical for preventing fraud and uncovering new and novel efforts to steal.

The features for training can be generalized as those that identify types of settlement method, transaction amount, media type (i.e., token, swipe card, etc.), time of day, location, repetitive transactions of the same or similar characteristics and the like. Thus, the feature set, whether for training or testing, is typically straight-forward and requires little pre-processing or disambiguation. Likewise, the label for each transaction, being a binary indicator of fraudulent or non-fraudulent, is equally straight-forward.

This body of structured data, characterized by minimal pre-processing and binary labeling, presents the opportunity for an intermediate feature. As fraud is confirmed and the algorithm is updated with newly learned information, it is likely that fraud may be suspected in some cases by the AI but does not cross the threshold as being a true positive. An argument can be made that such transactions be labeled as possible fraud pending more learning or pattern identification. Statistically, this is a plausible option. Consider that, with an alpha of five percent ($\alpha = 0.05$), transactions labeled as non-fraudulent among one-hundred total transactions in a sample (where one or more transaction is indeed fraudulent), may reveal a pattern of feature combinations that suggest possible fraudulent activity in test data (Hutton, L. 1992).

Combining supervised and unsupervised learning in any suite of AI algorithms will likely prove beneficial to fraud researchers and practitioners. Supervised learning to identify with high confidence particular transactions whose features indicate fraud is the goal of the application but uncovering those subtle combinations or patterns of features that indicate a new fraud technique is equally important. Clustering algorithms based on unlabeled transactions—unsupervised learning—can lead to identifying both new information about possible fraud as well as confirming results from supervised learning algorithms.

AI Modeling

Modeling an AI algorithm for fraud identification can take a number of forms. Historically, these have ranged from data mining to neural networks, fuzzy neural nets, supervised and unsupervised techniques, support vector machines (SVM) and the like (Phua, Lee, Smith-Miles & Gayler, 2005). AI deep learning techniques have been recently employed for training algorithms, largely driven by commercial applications targeted to specific industries and broader cloud-based offerings such as Amazon Web Services (AWS), Google AI and similar along with numerous consultants offering AI application services. As reported above, these approaches are all constrained by the skill of the individual or team charged to build the AI algorithm and develop the data.

With the large volume of data typically available to transit operations, deep learning neural networks (NN) offer a viable option for modeling (Goodfellow, Bengio and Courville, 2016). A deep learning model is, quintessentially, a multi-layer perceptron; that is, multiple hidden layers of nodes separating the input and output layers of the model. These layers contribute to creating an algorithm by slowly adjusting weights and biases applied throughout the model based on comparing model output to correct label values. A slow learning rate is necessary to prevent the models from suffering a variety of problems such as vanishing or exploding gradient (Wilson, D., Martinez, T., 2001). The goal is to use data to train a model to correctly identify true positives and be generalizable enough to recognize a range of fraud while avoiding overfitting.

The model builder has a number of parameters and hyperparameters available for creating the initial algorithm and making adjustments for fine-tuning the model based on the learning outcomes of the training data set. Neural networks are constrained by the number of data features and output for determining the

number of input and output layer nodes, respectively. The number of input nodes will match the number of features plus the label and, given that fraud identification models will classify a data instance as fraudulent or not, the output layer will be a single node. Examples of parameters and hyperparameters include learning rate, number of hidden layers, number of nodes in each layer, activation functions to be applied to the entire model or to various layers and the number of iterations to be used for training the model. Too, the model can be a feed-forward network or one that also employs backpropagation for training.

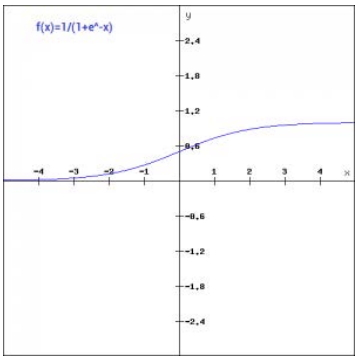
Backpropagation is the process that measures the differences in output values with known values, the labels of the training data. The difference in values is computed by a cost function that then adjusts the weights and biases throughout the network. Weights are values that connect the layers of nodes and can be thought of as determinants of how each feature impacts the predicted value. Biases are adjustments to the activation function output ranges of the nodes. Biases do not change the AF; they change the range of the AF calculation.

Activation Functions

Activation functions (AF) are mathematical functions in the general form,

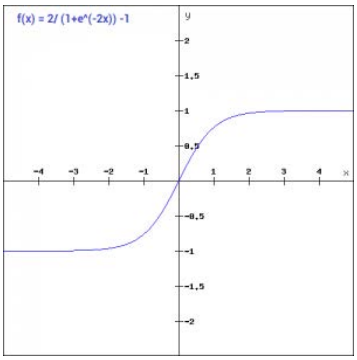
$$Y = AF(\Sigma(\text{weight} * \text{input}) + \text{bias}),$$

are assigned to nodes for adjusting node input (feature value multiplied by weight) and applying a bias for delivery to the next layer of nodes. Common choices of AFs for binary output NNs include rectified linear unit (ReLU), sigmoid and tanh. Key to these AFs are their ability to converge features to a binary output value.



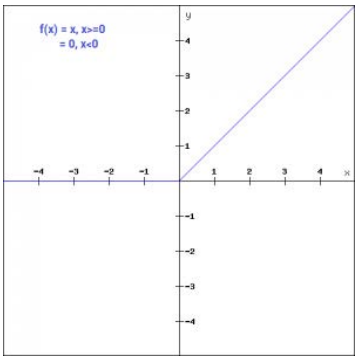
`f(x)=1/(1+e^-x)`

Figure 1. Sigmoid



`tanh(x)=2/(1+e^(-2x))-1`

Figure 2. Tanh



`f(x)=max(0,x)`

Figure 3. ReLU

Figures 1, 2 and 3 (Gupta, 2017)

The sigmoid AF (Figure 1) is non-linear, S-shaped and ranges from zero to one. The steeper gradient in the middle assures a greater change in Y for changes in X which pushes Y toward the extremes, a valuable property for modeling binary output.

Tanh (Figure 2) is similar to the sigmoid AF but handles negative values as it ranges from -1 to 1 and is more aggressive in pushing Y values to the extremes.

The rectified linear unit (Figure 3) is the most widely used activation function. It is non-linear yet sets positive Y values linearly relative to non-zero X values. Negative X values result in Y values of zero. This gives ReLU its unique value to neural networks in that not all nodes activate with each feed-forward or backpropagation iteration. If the value of a node results in a zero value that node is not activated. This results in a sparse network that ignores values not valued for training while focusing computing power on the positive values.

All three of these activation functions handle backpropagation well due to their non-linearity. They can be combined in various combinations of feedforward and backpropagation AF assignments to affect better matching of features to a binary outcome for a more efficient algorithm. Thus, generally, neural network

models constructed for fraud detection using these modeling techniques are being deployed by practitioners with increasing frequency.

This study investigates neural networks for fraud identification efficacy and analyzes their ability and reliability for this purpose.

Research Questions

To confidently employ neural networks for identifying fraudulent transactions it is prudent to understand the modeling and how NNs function to classify transactions as being fraudulent. Particularly, it is important to accurately identify true positives (a fraudulent transaction is indeed fraud) and avoid false negatives (a non-fraudulent transaction identified as being fraudulent). Accuracy in both cases is critical for transit authorities.

1. Can artificial neural networks equal or exceed other models in identifying fraudulent transactions?

To assess the suitability of neural networks for modeling transaction fraud it is necessary to look inside the "black box" of neural networks, consider what can be adjusted for increasing both accuracy and generalizability, what adjustments are outside control of model builders and which aspects of neural network algorithms may improve or inhibit modeling goals.

2. What modeling adjustments are typically hidden from model builders and how do those impact the usability and reliability of classification predictions?

Data

The data for this research was sampled from the Metro Atlanta Rapid Transit Authority (MARTA) transactional database where individual transactions had been flagged as fraudulent or non-fraudulent. 15,000 records were randomly extracted with a ratio of one fraudulent transaction per two non-fraudulent transactions. Thus, the NN model was built with balanced data consisting of 10,000 non-fraudulent and 5,000 fraudulent transactions. Features (Figure 4) were selected by MARTA for their likely correlation with fraudulent activity.

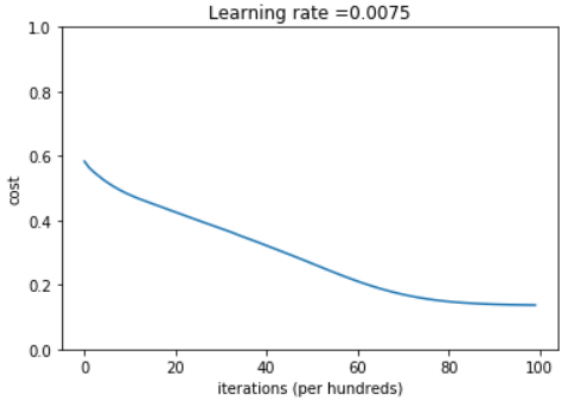
Header	Description
SERIAL_NBR	Unique serial number for transit ticket/card
HOTLISTED_FLAG	Fraud =1, not Fraud =0
MEDIA_TYPE_DESC	Product type. Customers can load multiple products i.e. weekly pass, single trip, monthly pass
RC_DESC	Rider class description
MODES	Transportation modes accessed (rail, bus, handicap bus)
USE_TYPES	Use types include entries, exists, purchases, balance checks, etc.
DEVICES	Gates, vending machines, bus terminals etc.
FACILITIES	This is primarily the number of rail stations utilized
FARE_INTRUMENTS	Quantity of products utilized
FARE_CATEGORIES	Quantity of fare categories utilized
ENTRIES	Bus or rail entry
EXITS	Bus or rail exit
ENT_EXT_RATIO	Difference between entries and exits

Figure 4. Data features of the MARTA data set

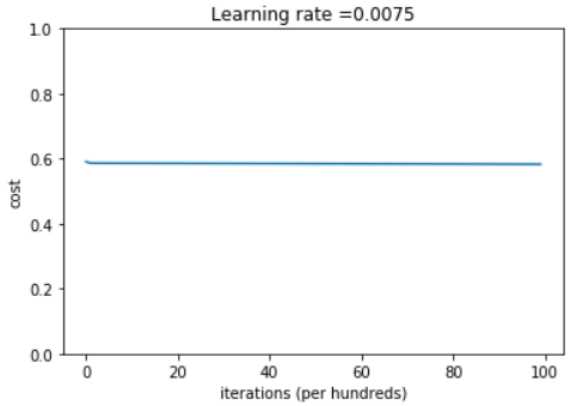
Method / Model

The 15,000 sample data records were further balanced to produce 10,000 records and then randomly sampled and split into 8,000 training and 2,000 testing records. A neural network was constructed using sklearn libraries and the Python language in a Jupyter notebook in the Anaconda environment.

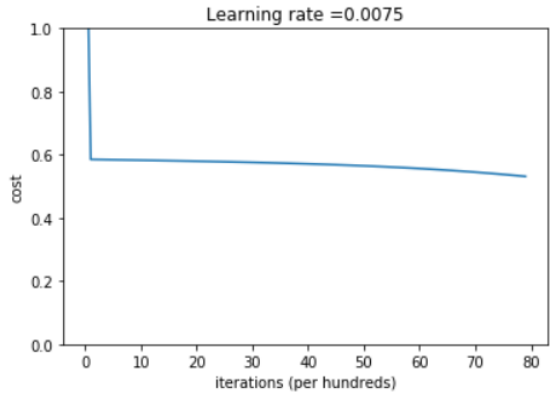
The model was constructed with five layers: The input layer flexible with the number of nodes to match the number of features plus the label (Figure 4), three layers of 128 nodes each and a single node output layer for the binary prediction. ReLU, sigmoid and tanh activation functions were programmed to allow maximum flexibility, as allowed by the sklearn library, for assigning AFs during training and testing. Thus, the model was built with the ability to assign activation functions for the three hidden layers of 128 nodes each and for the single output layer. The learning rate was set to 0.0075. Additional code was added to the model to report cost (Figure 5) and accuracy at every 100 iterations of training (Figure 6). After data splitting, the models were trained using four combinations of activation functions and various iterations. No other hyperparameters were changed.



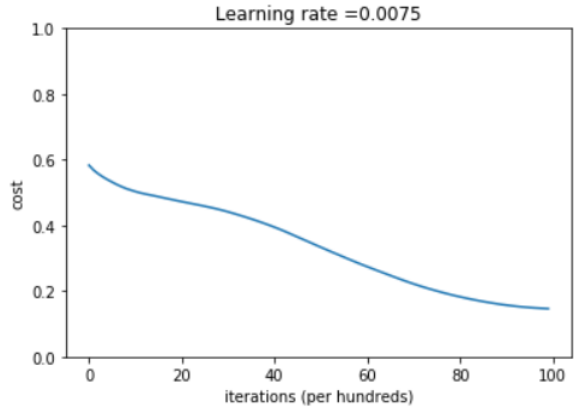
a. Forward hidden/output: ReLU/ReLU
Backprop hidden/output: ReLU/ReLU



b. Forward hidden/output: sigmoid/sigmoid
Backprop hidden/output: sigmoid/sigmoid

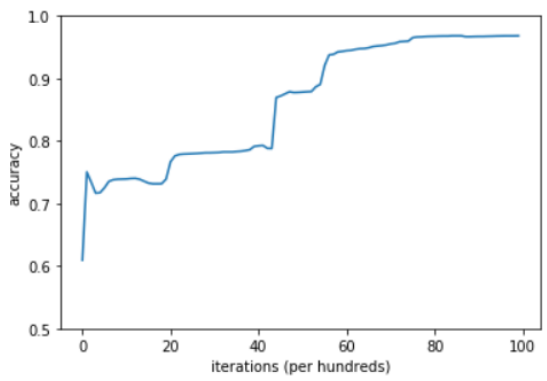


c. Forward hidden/output: sigmoid/tanh
Backprop hidden/output: sigmoid/tanh

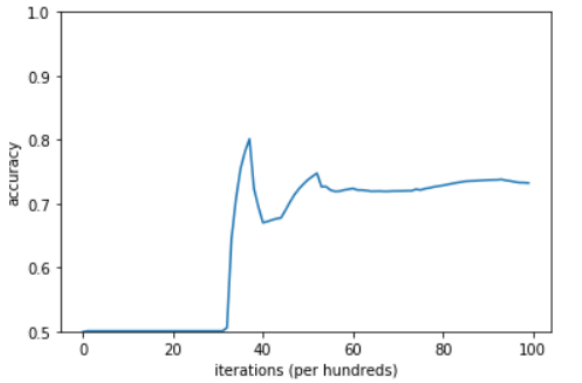


d. Forward hidden/output: ReLU/sigmoid
Backprop hidden/output: tanh/sigmoid

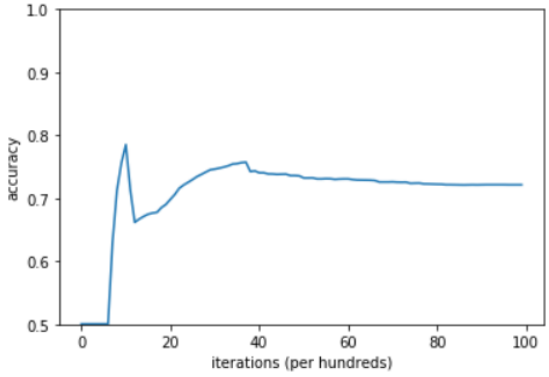
Figure 5. Cost over 10,000 iterations of indicated activation function combinations



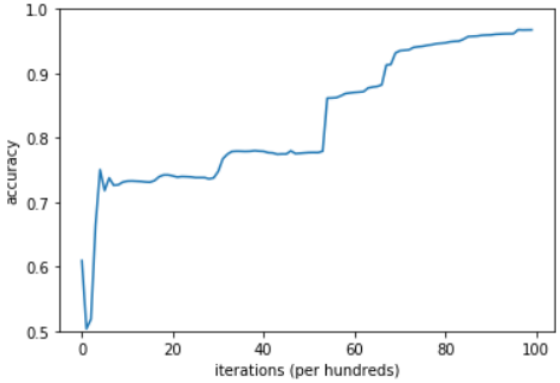
a. Forward hidden/output: ReLU/ReLU
Backprop hidden/output: ReLU/ReLU



b. Forward hidden/output: sigmoid/sigmoid
Backprop hidden/output: sigmoid/sigmoid



c. Forward hidden/output: sigmoid/tanh
Backprop hidden/output: sigmoid/tanh



d. Forward hidden/output: ReLU/sigmoid
Backprop hidden/output: tanh/sigmoid

Figure 6. Accuracy over 10,000 iterations of indicated activation function combinations

Immediately obvious from the graphs in Figure 6 is how the cost function reveals very different values over the 10,000 iteration period of training the neural network. The perception of training NNs is that the cost function should reduce over iterations as the network adjusts out differences in predicted and actual values. That appears to be the case for Figure 6a and Figure 6d. However, the other combinations of AFs result in very flat cost over the training iterations.

The accuracy plots generally align with the cost plots in terms of indicating eliminating cost while progressing to increased accuracy. However, there are numerous concerns. The progressions of Figure 6a and Figure 6d, while overall improving accuracy, exhibit reversals and plateaus that raise questions about the model. Where does the practitioner confidently claim success and end the model training? It certainly would be tempting, and is common, to stop training when a learning plateau is reached. However, this investigation into what is happening in the "black box" over a large number of training iterations suggests that optimal training is elusive, and that AF choice can lead to a false confidence in the model. With the ability to track accuracy and cost over the life of the experiment it can be confirmed that the activation function combinations in Figure 6b and Figure 6c should not be trusted.

Limitations / Future Research

Limiting this research was the relatively small data set and scope. Regardless, it revealed that future investigation of the internal metrics of established neural network models is deserved. The observation that building and evaluating NN models is heavily dependent on modeler skills, experience and insight cannot be overstated. The varying metrics over the training lives of these four AF combinations indicate that choice of activation function is critical to solid and reliable modeling yet extra steps not normally considered are required to gain insight into the "black box." These models were trained and tested using the same training and testing data sets while holding all hyperparameters fixed, with the exception of activation functions. Given the overall variances in the accuracies reported at the completion of testing and training (Table 1) and the variances observed in Figure 5 and Figure 6, additional research into the performance and impact of activation functions is warranted.

Forward hidden/output	ReLU/sigmoid	sigmoid/sigmoid	sigmoid/tanh	ReLU/sigmoid
Backprop hidden/output	ReLU/sigmoid	sigmoid/sigmoid	sigmoid/tanh	tanh/sigmoid
Accuracy – training	0.9685	0.7310	0.7215	0.9676
Accuracy -- testing	0.9670	0.7335	0.7170	0.9685

Table 1. Training and testing accuracies for activation function combinations

Conclusion

As AI researcher, author and academic Michael Jordan observed, artificial intelligence is not yet an engineering discipline (Jordan, 2018). As machine learning and AI become more common and commercialized, it will be imperative that MIS academics and researchers investigate these algorithms, as they do, for example, database management systems, so as to better understand reliable model selection, algorithm construction, proper application of the numerous parameters and tools internal to modeling systems, match data to models, and to consider research areas. The tangible necessity of this understanding is revealed in Table 1, where such results, which are too often used to measure model performance, would lead the modeler to select one of the two algorithms expressing greater than 96% accuracy while being completely unaware of the underlying issues. We are, as Professor Jordan observed, at a point in AI where bridge builders were prior to civil engineering. We see the value in AI as our ancestors did in bridges and we are anxious to get there. But as their bridges often failed, with tragic results, our attempts at solid and reliable AI will require us to better understand the mysteries that gird our learning machines as builders learned the mysterious combinations of materials and techniques to successfully cross chasms.

Acknowledgement

The authors wish to thank and acknowledge the generous work of Mr. David Edwards, privately employed by industry, for his contributions of coding assistance creating the neural network code and edits in Python in support of this research.

References

Beleites, C., Neugebauer, U., Bocklitz, T., Krafft, C., and Popp, J. (2012). Sample Size Planning for Classification Models. Analytica Chimica Acta, 2013, 760 (Special Issue: Chemometrics in Analytical Chemistry 2012), 25-33, DOI: 10.1016/j.aca.2012.11.007.

Goodfellow, I., Bengio, Y., & Courville, A., (2016), "Deep Learning." The MIT Press, Cambridge, Massachusetts, USA. (2016).

Gupta, D. (2017). Fundamentals of Deep Learning – Activations Functions and When to Use Them? Analytics Vidhya, https://www.analyticsvidhya.com/blog/2017/10/fundamentals-deep-learning-activation-functions-when-to-use-them/ accessed September 12, 2018.

Hutton, L. (1992). Using Statistics to Assess the Performance of Neural Network Classifiers. Johns Hopkins APL Technical Digest, Volume 13, Number 2 (1992).

Jordan, M. (2018). Artificial Intelligence – The Revolution Hasn't Happened Yet. Medium (2018, April 18). https://medium.com/@mijordan3/artificial -intelligence-the-revolution-hasnt-happened-yet-5e1d5812e1e7 accessed September 20, 2018.

Murphey, Y., Guo, H. & Feldkamp, L. (2004). Neural Learning from Unbalanced Data. L.A. Applied Intelligence 21:117-128 (2004).

Phua, C., Lee, V., Smith-Miles, K. & Gayler, R. (2005). A Comprehensive Survey of Data Mining-based Fraud Detection Research. Clayton School of Information Technology, Monash University.

Wilson, D. & Martinez, T. (2001). The Need for Small Learning Rates on Large Problems. Proceedings of the 2001 International Conference of Neural Networks (IJCNN'01), 115-119.

Wistuba, M. (2018). Finding Competitive Network Architectures Within a Day Using UCT. arXiv: 1712.07420v2, July 23, 2018.