Multilayer Perceptron and Stacked Autoencoder for Internet Traffic Prediction

Tiago Prado Oliveira, Jamil Salem Barbar, and Alexsandro Santos Soares

Federal University of Uberlândia, Faculty of Computer Science, Uberlândia, Brazil,
`tiago_prado@comp.ufu.br`, `{jamil,alex}@facom.ufu.br`

Abstract. Internet traffic prediction is an important task for many applications, such as adaptive applications, congestion control, admission control, anomaly detection and bandwidth allocation. In addition, efficient methods of resource management can be used to gain performance and reduce costs. The popularity of the newest deep learning methods has been increasing in several areas, but there is a lack of studies concerning time series prediction. This paper compares two different artificial neural network approaches for the Internet traffic forecast. One is a Multilayer Perceptron (MLP) and the other is a deep learning Stacked Autoencoder (SAE). It is shown herein how a simpler neural network model, such as the MLP, can work even better than a more complex model, such as the SAE, for Internet traffic prediction.

Keywords: Internet traffic, time series, prediction, forecasting, neural network, machine learning, multilayer perceptron, deep learning, stacked autoencoder.

1 Introduction

Using past observations to predict future network traffic is an important step to understand and control a computer network. Network traffic prediction can be crucial to network providers and computer network management in general. It is of significant interest in several domains, such as adaptive applications, congestion control, admission control and bandwidth allocation.

There are many studies that focus on adaptive and dynamic applications. They usually present some algorithms, that use the traffic load, to dynamically adapt the bandwidth of a certain network component [1][2][3] and improve the Quality of Service (QoS) [4]. Several works have been developed using Artificial Neural Networks (ANN) and they have shown that ANN are a competitive model, overcoming classical regression methods such as ARIMA [5][6][7][8]. Thus, there are works that combine these two factors, therefore producing a a predictive neural network that dynamically allocates the bandwidth in real-time video streams [3].

Initially, the use of neural networks was limited in relation to the number of hidden layers. Neural networks made up of various layers were not used due to the difficulty in training them [9]. However, in 2006, Hinton presented the Deep

Belief Networks (DBN), with an efficient training method based on a greedy learning algorithm, which trains one layer at a time[10]. Since then, studies have encountered several sets of good results regarding the use of deep learning neural networks. Through these findings this study has as its objective to use the deep learning concept in traffic prediction.

Network traffic is a time series, which is a sequence of data regularly measured at uniform time intervals. For network traffic, these sequential data are the bits transmitted in some network device at a certain period on time. A time series can be a stochastic process or a deterministic one. To predict a time series it is necessary to use mathematical models that truly represent the statistical characteristic of the sampled traffic. The choice of the prediction method must take into account the prediction horizon, computational cost, prediction error and the response time, for adaptive applications that require real-time processing.

This paper analyses two prediction methods that are based on ANN. Evaluations were made comparing Multilayer Perceptron (MLP) and Stacked Autoencoder (SAE). MLP is a feed-forward neural network with multiple layers that uses Backpropagation as supervised training. SAE is a deep learning neural network that uses a greedy algorithm for unsupervised training. The analysis focuses on a short-term forecast and the tests were made using samples of Internet traffic time series, which were obtained on DataMarket database [11].

2 Artificial Neural Networks

Artificial Neural Networks are simple processing structures, which are separated into strongly connected units called artificial neurons (nodes). Neurons are organized into layers, one layer has multiple neurons and any one neural network can have one or more layers, which are defined by the network topology and vary among different network models [12].

Neurons are capable of working in parallel to process data, store experimental knowledge and use this knowledge to infer new data. Each neuron has a synaptic weight, which is responsible for storing the acquired knowledge. Network knowledge is acquired through learning processes (learning algorithm or network training) [12]. In the learning process, the network will be trained to recognize and differentiate the data from a finite set. After learning, the ANN is ready to recognize the patterns in a time series, for example. During the learning process the synaptic weights are modified in an ordered manner until they reach the desired learning. A neural network offers the same functionality as neurons in a human brain for resolving complex problems, such as nonlinearity, high parallelism, robustness, fault tolerance, noise tolerance, adaptability, learning and generalization [5][12].

Deep learning refers to a machine learning method that is based on a neural network model with multiple levels of data representation. Hierarchical levels of representation are organized by abstractions, features or concepts. The higher levels are defined by the lower levels, where the representation of the low-levels may define several different features of the high-levels, this makes the data representation more abstract and nonlinear for the higher levels [9][10]. These hierarchical levels are represented by the layers of the ANN and they allow for the adding of a significant complexity to the prediction model. This complexity is proportional to the number of layers that the neural network has. The neural network depth concerns to the number of composition levels of nonlinear operations learned from trained data, i.e., more layers; more nonlinear and deeper is the ANN.

The main difficulty in using deep neural networks relates to the training phase. Conventional algorithms, like Backpropagation, do not perform well when the neural network has more than three hidden layers [13]. Besides, these conventional algorithms do not optimize the use of more layers and they do not distinguish the data characteristics hierarchically, i.e., the neural network with many layers does not have a better result to that of a neural network with few layers, e.g., shallow neural network with two or three layers [14][15].

3 Review of Literature

Several types of ANN have been studied for network traffic prediction. An advantage of ANN is the response time, i.e., how fast the prediction of future values is made. After the learning process, which is the slowest step in the use of an ANN, the neural network is ready for use, obtaining results very quickly compared to other more complex prediction models as FARIMA [8]. Therefore, ANNs are better at online prediction, obtaining a satisfactory result regarding prediction accuracy and response time [5].

3.1 Multilayer Perceptron and Backpropagation

One of commonest architectures for neural networks is the Multilayer Perceptron. This kind of ANN has one input layer, one or more hidden layers, and an output layer. Best practice suggests one or two hidden layers [14]. This is due to the fact that the same result can be obtained by raising the number of neurons in the hidden layer, rather than increase the number of hidden layers [15].

MLPs are feed-forward networks, where all neurons in the same layer are connected to all neurons of the next layer, yet the neurons in the same layer are not connected to each other. It is called feed-forward because the flow of information goes from the input layer to the output layer. The training algorithm used for MLP is the Backpropagation, which is a supervised learning algorithm, where the MLP learns a desired output from various entry data.

3.2 Stacked Autoencoder and Deep Learning

Stacked Autoencoder is a deep learning neural network built with multiple layers of sparse Autoencoders, in which the output of each layer is connected to the input of the next layer. SAE learning is based on a greedy layer-wise unsupervised training, which trains each Autoencoder independently [16][17][18].

The strength of deep learning is based on the representations learned by the greedy layer-wise unsupervised training algorithm. Furthermore, after a good data representation in each layer is found, the acquired neural network can be used to initialize some new ANN with new synaptic weights. This new initialized neural network can be an MLP, e.g., to start a supervised training if necessary [9]. A lot of papers emphasize the benefits of the greedy layer-wise unsupervised training for deep network initialization [9][10][18][19][20]. Therefore, one of the goals of this paper is to verify if the unsupervised training of deep learning does actually bring advantages over the simpler ANN models.

4 Experiments and Results

The utilized time series data were gathered on DataMarket and created by R. J. Hyndman [11]. The experiments were performed from data collected daily, hourly and at five minute intervals. Altogether, six time series were used, with them being “A-1d”, “A-1h”, “A-5m”, “B-1d”, “B-1h” and “B-5m”.

These time series used are composed of Internet traffic (in bits) from a private Internet Service Provider (ISP) with centres in 11 European cities. The data corresponds to a transatlantic link and was collected from 06:57 hours on 7 June to 11:17 hours on 31 July 2005. This series was collected at different intervals, resulting in three different time series: “A-1d” is a time series with daily data; “A-1h” is hourly data; “A-5m” contains data collected every five minutes.

The remaining time series are composed of Internet traffic from an ISP, collected in an academic network backbone in the United Kingdom. They were collected from 19 November 2004, at 09:30 hours to 27 January 2005, at 11:11 hours. In the same way, this series was divided into three different time series: “B-1d” is daily data; “B-1h” is hourly data; “B-5m”, with data collected at five minute intervals.

The conducted experiments used DeepLearn Toolbox [21], an open source code of different libraries that cover several machine learning and artificial intelligence techniques. Some are, Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Stacked Autoencoders (SAE), Convolutional Autoencoders (CAE) and Deep Belief Networks (DBN). The libraries of DeepLearn Toolbox are coded using the MATLAB environment tool.

4.1 Data Normalization

Before training the neural network, it is important to normalize the data [8], in this case, the time series. In addition, for a better calculation and results, the DeepLearn Toolbox requires that the input data are next to zero. Hence, to decrease the time series scale the z-score was used to normalize the data. After that, a sigmoid function was applied, so that the time series values are in the range $[0, 1]$. The z-score was chosen as through it the data scale are preserved and patterns are not changed.

Table 1. The time interval and size of each time series

Data Set	Time interval	Time Series total size	Training Set size
A-1d	1 day	51	25
A-1h	1 h	1231	615
A-5m	5 min	14772	7386
B-1d	1 day	69	34
B-1h	1 h	1657	828
B-5m	5 min	19888	9944

The original time series is normalized, generating a new normalized time series, which will be used for the training. The range of the 6 time series used varies from 51 values (for the smallest time series, with daily data) to 19888 values(for the largest time series, with data collected at five minute intervals). During the experiments the data range for the training set varied greatly, from 25 values (for the smallest time series) to 9944 values (for the largest time series). The size for the training set was chosen as that of the first half of the time series, the other half of the time series is the test set for evaluating the prediction accuracy. The size of each data set can be seen in Table 1.

4.2 Neural Network Architecture and Topology

For the standard neural network, the MLP was used, with a sigmoid activation function, a low learning rate of 0.01 and Backpropagation as the training algorithm. For the deep learning neural network, the SAE was used, also with sigmoid activation function and a learning rate of 0.01. Higher learning rate accelerates the training, but may generate many oscillations in it, making it harder to reach a low error. On the other hand, a lower learning rate leads to steadier training, however is much slower.

This low value for the learning rate was used because, for our purpose, the training time was not the most important target, in fact, it is the final error achieved by the artificial neural network. Was also tested different ones, such as 0.5, 0.25 and 0.1, yet as expected, the lowest errors were obtained using 0.01 for the learning rate.

The training algorithm of SAE is a greedy algorithm that gives similar weights to similar inputs. Each Autoencoder is trained separately, in a greedy fashion, then it is stacked onto those already trained; thereby, producing a SAE with multiple layers. The greedy training algorithm is used to train unlabeled data, i.e., it does not train the data considering the expected output. On the other hand, for labeled data such as time series, the greedy training is not sufficient and it is used as a pre-training to initialize the neural network weights (instead of a standard random initialization). After that, the Backpropagation algorithm was used as a fine-tuning for the supervised training [9][13][17][18].

Several tests were carried out varying the ANN topology, both in number of neurons per layer as in the number of layers. For the MLP, the best performances

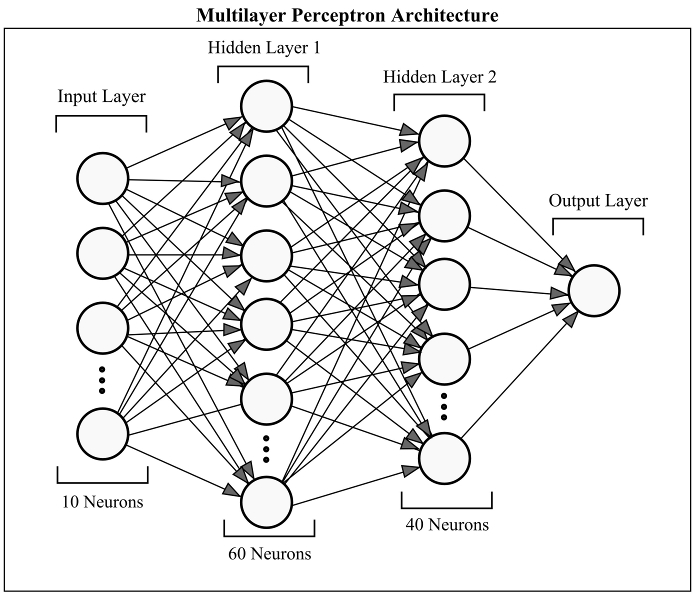

Fig. 1. Neural Network Architecture showing the layers, numbers of each layer and the information feed-forward flow

were obtained with 4 layers, around 10 input neurons, 1 output neuron, 60 and 40 neurons in the hidden layers, respectively, as shown in Fig. 1. It was found that increasing the number of neurons did not result in better performance, the average of Root Mean Square Error (RMSE) was found to be similar. However, for the ANN with 5 or more layers, overly increasing the number of layers was detrimental to performance.

For the SAE, the best results were found with 6 layers, with 20 input neurons, 1 output neuron, 80, 60, 60 and 40 neurons in each of the hidden layers, respectively. Increasing the number of neurons of the SAE did not produce better results; on average the Normalized Root Mean Square Error (NRMSE) was very similar. Similar results were found also with 4 layers, like the MLP, whereas deeper SAE achieved slightly better results. A comparison of the NRMSE of each prediction model will be shown in Table 2.

4.3 Neural Network Training

The neural network training was carried out in a single batch. This way all input data of the training set is trained in a single training epoch, adjusting the weights of the neural network for the entire batch. Tests with more batches (less input data for each training epoch) were also realized and similar error rates were found. Nevertheless, for smaller batches, the training took more time to converge, because a smaller amount of data is trained at each epoch.

The training time is mainly affected by the size of the training set and by the number of neurons of the neural network. The higher the size of the training set and higher the number of neurons, more time is necessary to train the neural network.

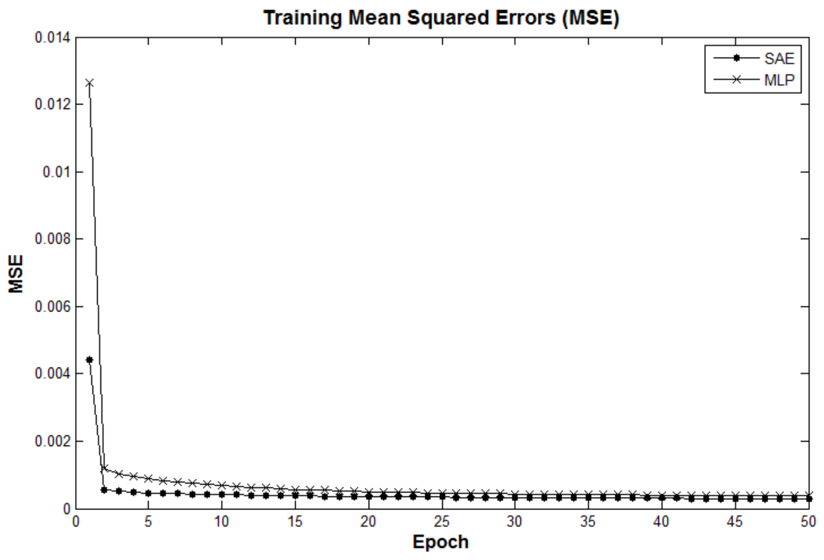

Fig. 2. A MSE comparison of SAE and MLP at each training epoch, for B-5m time series

The MLP training lasted 1000 epochs. The SAE training is separated into two steps. The first one is the unsupervised pre-training, which lasted 900 epochs. The second step is the fine-tuning that uses a supervised training, which lasted 100 epochs. Fig. 2 shows the first 50 training epochs and their respective errors, comparing the fine-tuning training of the SAE with the MLP training. It is possible to observe that, because of the SAE pre-training, the SAE training converges faster than the MLP training. However, more training epochs are enough for them to obtain very similar error rates.

Fig. 3 and Fig. 4 show the time series prediction results for the MLP and SAE, respectively. It is noted that MLP best fits the actual data, nevertheless, both fared well in data generalization. Both, MLP and SAE, learned the time series features and used these features to predict data that are not known a priori.

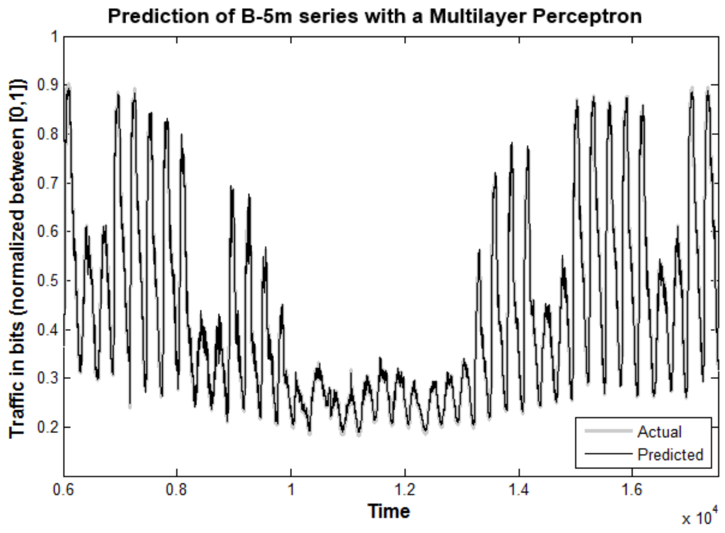

Fig. 3. A prediction comparison of the MLP at each training epoch for B-5m time series. It shows the Actual (the original) time series (represented in grey) and the Predicted one (represented in black). Since the actual and the predicted plot lines are very similar, it is difficult to see the difference with a low scale image. Yet, it is possible to see that the predicted values fit very well to the actual values.

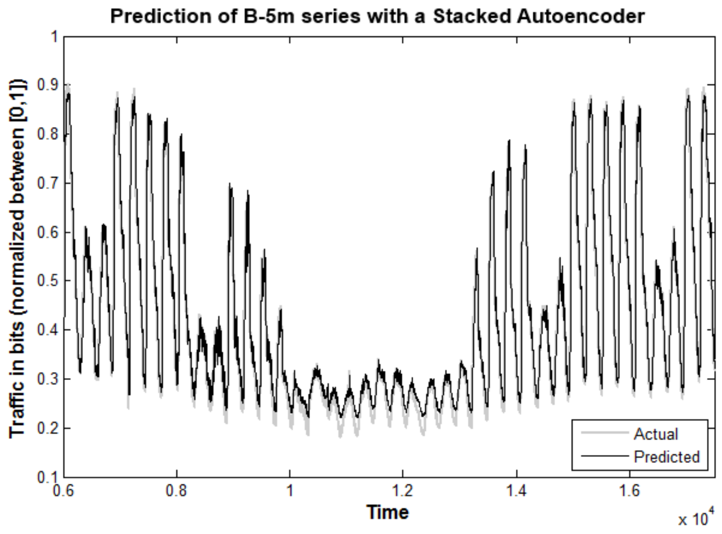

Fig. 4. A prediction comparison of the SAE at each training epoch for B-5m time series. It shows the Actual (the original) time series and the Predicted one. It is noted that the predicted (represented in black) did not fit well from 1×10^4 to 1.4×10^4 period in time, but for the rest of the series the predicted values fit well to the actual values.

4.4 Main Results

The key idea of deep learning is that the depth of the neural network allows learning complexes and nonlinear data [9]. However, the use of SAE for time series prediction was not beneficial, i.e., the pre-training did not bring significant benefits to prediction. The best results for the MLP and SAE with their respective NRMSE are shown in Table 2. Even though the MLP does not have a significant advantage over the SAE, still, the MLP has achieved better results for network traffic prediction.

Table 2. A comparison of Normalized Root Mean Squared Error (NRMSE) results

Data Set	NRMSE	
	MLP	SAE
A-1d	0.1999	0.3660
A-1h	0.0479	0.0756
A-5m	0.0192	0.0222
B-1d	0.1267	0.2155
B-1h	0.0421	0.0556
B-5m	0.0131	0.0184

In the time series prediction, the SAE method has more complexity than the MLP, since it has the extra unsupervised training phase, which initializes the neural network weights for the fine-tuning stage. Even with the additional complexity, the SAE was slightly inferior. Due to this fact, this approach is not recommended for time series prediction.

There are works, in the pattern recognition field, where the use of Autoencoders are advantageous [20], as they are based in unlabeled data. On the other hand, there are works, in time series prediction and labelled data, showing that the Autoencoders approach is worse than classical neural networks [22], such as MLP and Recurrent ANN. Each of these problems has a better method for solving it, so it is important to analyse the entry data type before choosing the most appropriate method to be used.

5 Conslusion

Both types of studied ANN have proven that they are capable of adjusting and predicting network traffic accurately. However, the initialization of the neural network weights through the unsupervised pre-training did not bring an improvement for time series prediction. The result shows that MLP is better than SAE for Internet traffic prediction. In addition, the SAE deep neural network approach reflects on more computational complexity during the training, so the choice of MLP is more advantageous.

The use and importance of deep neural networks is increasing and very good results are achieved in images, audio and video pattern recognition [19][20][23][24]. However, the main learning algorithms for this kind of neural network are unsupervised training algorithms, which use unlabelled data for their training. In contrast, network traffic and time series, in general, are labeled data, requiring an unsupervised pre-training before the actual supervised training as a fine-tuning. Yet, as shown in [24], the DBN and restricted Boltzmann machine (RBM), which are deep learning methods, can be modified to work better with labeled data, i.e., time series data sets.

Future works will focus in other deep learning techniques, like Deep Belief Nets and Continuous Restricted Boltzmann Machine (CRBM) and other models of ANN, such as the Recurrent Neural Network (RNN), with others training algorithms, such as Resilient Backpropagation. Even better results are expected, since they are more optimized for learning sequential and continuous data. Other future works will use the network traffic prediction to create an adaptive bandwidth management tool. This adaptive management tool will first focus on congestion control through bandwidth dynamic allocation, based on the traffic predicted. The objective is to guarantee a better QoS and a fair share of bandwidth allocation for the network devices in a dynamic and adaptive management application.

References

1. Han, M.-S.: Dynamic bandwidth allocation with high utilization for XG-PON. In: 16th International Conference on Advanced Communication Technology (ICACT), pp. 994–997. IEEE (2014)
2. Zhao, H., Niu, W., Qin, Y., Ci, S., Tang, H., Lin, T.: Traffic Load-Based Dynamic Bandwidth Allocation for Balancing the Packet Loss in DiffServ Network. In: 11th International Conference on Computer and Information Science (ICIS), pp. 99–104. IEEE/ACIS (2012)
3. Liang, Y., Han, M.: Dynamic Bandwidth Allocation Based on Online Traffic Prediction for Real-Time MPEG-4 Video Streams. EURASIP Journal on Advances in Signal Processing (2007)
4. Nguyen, T.D., Eido, T., Atmaca, T.: An Enhanced QoS-enabled Dynamic Bandwidth Allocation Mechanism for Ethernet PON. In: International Conference on Emerging Network Intelligence, pp. 135–140. EMERGING (2009)
5. Cortez, P., Rio, M., Rocha, M., Sousa, P.: Multi-scale Internet traffic forecasting using neural networks and time series methods. ExpertSystems: The Journal of Knowledge Engineering 29, 143–155 (2012)
6. Hallas, M., Dorffner, G.: A comparative study of feedforward and recurrent neural networks in time series prediction. In: 14th European Meet. Cybernetics Systems Research, vol. 2, pp. 644–647 (1998)
7. Ding, X., Canu, S., Denoeux, T.: Neural Network Based Models for Forecasting. In: Proceedings of Applied Decision Technologies (ADT 1995), pp. 243–252. Wiley and Sons, Uxbridge (1995)
8. Feng, H., Shu, Y.: Study on network traffic prediction techniques. In: International Conference on Wireless Communications, Networking and Mobile Computing, vol. 2, pp. 1041–1044. WiCOM (2005)

9. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning 2, 1–127 (2009)
10. Hinton, G.E., Osindero, S., Teh, Y.: A fast learning algorithm for deep belief nets. Neural Comput. 18, 1527–1554 (2006)
11. Hyndman, R.J.: Time Series Data Library, `http://data.is/TSDLdemo`
12. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall PTR, Upper Saddle River (1998)
13. Erhan, D., Manzagol, P.-A., Bengio, Y., Bengio, S., Vincent, P.: The difficulty of training deep architectures and the effect of unsupervised pre-training. In: Proceedings of The Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS 2009), pp. 153–160 (2009)
14. Villiers, J., Barnard, E.: Backpropagation neural nets with one and two hidden layers. IEEE Transactions on Neural Networks 4, 136–141 (1993)
15. Hornik, K., Stinchcombe, M., White, H.: Multi- layer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
16. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and Composing Robust Features with Denoising Autoencoders. In: Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML 2008), pp. 1096–1103. ACM, New York (2008)
17. Unsupervised Feature Learning and Deep Learning. Stanford's online wiki. Stacked Autoencoders,
`http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders`
18. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: Schlkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19 (NIPS 2006), pp. 153–160. MIT Press (2007)
19. Larochelle, H., Erhan, D., Vincent, P.: Deep learning using robust interdependent codes. In: Dyk, D.V., Welling, M. (eds.) Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS 2009), vol. 5, pp. 312–319 (2009); Journal of Machine Learning Research - Proceedings Track (2009)
20. Ranzato, M.A., Boureau, Y.-L., LeCun, Y.: Sparse Feature Learning for Deep Belief Networks. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20, pp. 1185–1192. MIT Press, Cambridge (2007)
21. Palm, R.B.: DeepLearnToolbox, a Matlab toolbox for Deep Learning, `https://github.com/rasmusbergpalm/DeepLearnToolbox`
22. Busseti, E., Osband, I., Wong, S.: Deep Learning for Time Series Modeling. Stanford, CS 229: Machine Learning (2012)
23. Arel, I., Rose, D.C., Karnowski, T.P.: Deep Machine Learning - A New Frontier in Artificial Intelligence Research [research frontier]. IEEE Computational Intelligence Magazine 5, 13–18 (2010)
24. Chao, J., Shen, F., Zhao, J.: Forecasting exchange rate with deep belief networks. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 1259–1266 (2011)