Load Forecasting using Deep Neural Networks

Stefan Hosein and Patrick Hosein
The University of the West Indies
St. Augustine, Trinidad and Tobago
stefan.hosein2@my.uwi.edu, patrick.hosein@sta.uwi.edu

Abstract—**Short-term electricity demand prediction is of great importance to power companies since it is required to ensure adequate capacity when needed and, in some cases, it is needed to estimate the supply of raw material (e.g., natural gas) required to produce the required capacity. The deregulation of the power industry in many countries has magnified the importance of this need. Research in this area has included the use of shallow neural networks and other machine learning algorithms to solve this problem. However, recent results in other areas, such as Computer Vision and Speech Recognition, have shown great promise for Deep Neural Networks (DNN). Unfortunately, far less research exists on the application of DNN to short-term load forecasting. In this paper, we apply DNN as well as other machine learning techniques to short-term load forecasting in a power grid. The data used is taken from periodic smart meter energy usage reports. Our results indicate that DNN performs quite well when compared to traditional approaches. We also show how these results can be used if dynamic pricing is introduced to reduce peak loading.**

Index Terms—**Deep Neural Networks, Machine Learning, Smart Grid, Load Forecasting**

I. Introduction

Forecasting electrical loads can be split into three groups, short-term forecasting (one hour to a few weeks), medium-term forecasting (a few weeks to a few months) and long-term forecasting (a few months to years). Recently there has been tremendous research done in the area of short-term load forecasting (STLF) [1]. This increase has been partly due to the deregulation of the power industry throughout many countries. STLF can assist many companies and sectors by providing an accurate means to predict future loads which would lead to precise planning, estimates of supplies and determining prices. This leads to decreases in operating cost, increase in profit and a more reliable electricity supply. Over the past decades of research in STLF, there have been numerous models proposed to solve this very pertinent problem. These models have been classified into classical approaches and machine learning based techniques [2].

In recent years, many deep learning methods have been shown to achieve state-of-the-art performance in many research areas: speech recognition [3], computer vision [4] and natural language processing [5]. This promise has not been demonstrated in other areas of computer science due to a lack of thorough research. Deep in this case refers to having a larger number of hidden layers than the traditional Feed Forward Neural Networks. The concept of having multiple layers is not new, however, but only recently was there an efficient way to train and accurately use Deep Neural Networks (DNN). Deep-learning methods are representation-learning methods with multiple levels of representation obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level [6]. With the composition of enough such transformations, very complex functions can be learned.

In this paper we compare deep architectures and traditional methods when applied to our STLF problem and we also provide a comprehensive analysis of numerous deep learning models. We then show how these methods can be used to assist in the pricing of electrical rates. To the best of out knowledge, there are only a handful of papers showing any sort of comparisons and analysis, but they are not very detailed. This work makes use of Stacked Autoencoders, Convolutional Neural Networks, Recurrent Neural Networks and Long-Short-Term Memory for the deep architectures. For traditional methods we consider moving averages, regression trees and support vector regressions. The data is based on one year of smart meter data collected from residential customers. We apply each of the algorithms to the curated data while also noting the corresponding runtimes. Due to differences in electricity usage between the week and the weekend, we then split the data into two new datasets: weekends and weekly data. The algorithms are applied to these new datasets and the results are analyzed. The results show that the deep architectures are superior to the traditional methods, and achieve the lowest error rate but they have the longest run-time.

II. Background

A. Traditional Methods

The traditional methods that were selected are: weighted moving average (WMA), linear regression (MLR) [7], regression trees (RT) [8], support vector regression (SVR) [9] and multilayer perceptrons (MLP) [10]. These methods were selected due to their popularity in the literature and usage in the industry.

B. Deep Neural Networks

Deep Neural Networks (DNN) prove to be more advantageous than MLP networks since they can succinctly represent a significantly larger set of functions. We use a number of these varied architectures and provide a brief explanation of each.

Stacked Autoencoders (SA) utilize a stacked architecture, where there is an autoencoder in each layer. An autoencoder

978-1-5386-2890-4/17/$31.00 ©2017 IEEE

is an unsupervised learning algorithm that is trained to reconstruct the input in the output layer, i.e., setting the target values $\mathbf{y} = \{y_1, y_2, \ldots\}$ to be equal to the inputs $\mathbf{x} = \{x_1, x_2, \ldots\}$, $y_i = x_i$. There are two main parts of the autoecnoder: the encoder and the decoder. The encoder maps the input space $\mathfrak{X}$ to a feature space $\mathfrak{F}$, while the decoder maps the feature space $\mathfrak{F}$ to the input space $\mathfrak{X}$. Using one hidden layer, the encoder calculates:

$$\mathbf{z} = \sigma_1(\mathbb{W}\mathbf{x} + \mathfrak{b}) \tag{1}$$

and the decoder calculates:

$$\widehat{\mathbf{x}} = \sigma_2(\widehat{\mathbb{W}}\mathbf{z} + \widehat{\mathfrak{b}}) \tag{2}$$

where $\mathbb{W}$ and $\widehat{\mathbb{W}}$ are the weight matrices for the input and output layers and σ is the activation sigmoid function defined in MLP. The loss function for the autoencoder is:

$$J(\mathbf{x}, \widehat{\mathbf{x}}) = \|\mathbf{x} - \widehat{\mathbf{x}}\|^2 = \left\|\mathbf{x} - \sigma_2\Big(\widehat{\mathbb{W}}\big(\sigma_1(\mathbb{W}\mathbf{x} + \mathfrak{b})\big) + \widehat{\mathfrak{b}}\Big)\right\|^2 \tag{3}$$

For a SA, we layer autoencoders one onto the other, where the output of one autoencoder becomes the input of another. A typical MLP can be added to the last autoencoder to perform prediction of time series data.

Recurrent Neural Networks (RNN) are another architecture. RNNs are neural networks containing feedback loops thus allowing the persistence of information. They consist of units interacting in discrete time via directed, weighted connections with weights w_{ij}, from unit j to unit i. Every unit has an activation $\widehat{y}(t)$ updated at every time step t ($t = 1, 2, \ldots$). An activation, $\widehat{y}^i$, of unit i is updated by computing its network input sum $\mathcal{N}^i$ where

$$\mathcal{N}^i(t) = \sum_j w_{ij}\widehat{y}^j(t-1) \tag{4}$$

and squashing it with a differentiable function (such as the sigmoid function) σ gives:

$$\widehat{y}^i(t) = \sigma(\mathcal{N}^i(t)) \tag{5}$$

We define learning in RNNs as optimizing a differentiable objective function E, summed over all the time steps of all sequences by adapting the connection weights. E is based on supervised targets y^k, where k indexes the output units of the network with activation $\widehat{y}^k$. An example is the squared error objective function:

$$E(t) = \frac{1}{2}\sum_k e_k(t)^2 \tag{6}$$

where $e_k = y^k(t) - \widehat{y}^k(t)$ which is the external error. $E(t)$ represents the error at time t for one sequence component called a pattern. For a typical dataset consisting of sequences of patterns, E is the sum of $E(t)$ over all patterns of all sequences. A gradient descent learning algorithm for RNNs, computes the gradient of E with respect to each weight w_{ij} to determine the weight changes δw_{ij}:

$$\delta w_{ij}(t) = -\tau\frac{\partial E(t)}{\partial w_{ij}} \tag{7}$$

where α is the learning rate [11].

III. Analysis

A. Data Description

Our dataset consists of hourly samples over a the period of a year and consisted of 18 features. The dataset was broken into 3 parts for training, validation and testing of size 65%, 15%, 20% respectively. The readings were recorded at hourly intervals throughout the year. Some of the features were electrical load readings for the previous hour, the previous two hours, the previous three hours, the previous day same hour, the previous day previous hour, the previous day previous two hours, the previous 2 days same hour, the previous 2 days previous hour, the previous 2 days previous two hours, the previous week same hour, the average of the past 24 hours and the average of the past 7 days. The rest of the features (which do not contain electrical load readings) are the day of the week, hour of the day, if it is a weekend, if it is a holiday, temperature and humidity. These features were selected as they are the norm for STLF. In addition, the total electrical load in this paper does not change significantly throughout the year, since the households are located in a tropical country where the temperature remains fairly constant throughout the year.

B. Method

As a preprocessing step, the data is cleaned and scaled to zero mean and unit variance . All traditional methods use cross-validation to determine appropriate values for the hyper-parameters. Random grid search was used to determine the hyper-parameters for the deep learning methods.

Several baseline algorithms were chosen. They include the Weighted Moving Average (WMA) where $y_{t+1} = \alpha y_i + \beta y_{i-167}$ with $\alpha = 0.05$ and $\beta = 0.95$, Multiple Linear Regression (MLR) and quadratic (MQR), Regression Tree (RT) with the minimum number of branch nodes being 8, Support Vector Regression (SVR) with a linear kernel and Multilayer Perception (MLP), with the number of hidden neurons being 100.

For our Deep Neural Networks we used[1]: Deep Neural Network without pretraining (DNN-W), DNN with pretraining using Stacked Autoencoders (DNN-SA), Recurrent Neural Networks (RNN), RNNs and Long Short Term Memory (RRN-LSTM).

To evaluate the goodness of fit of these algorithms we use the Mean Absolute Percentage Error (MAPE) defined as:

$$\mathrm{MAPE} = \frac{100}{n}\sum_{t=1}^{n}\frac{|y_t - \widehat{y}_t|}{y_t} \tag{8}$$

[1] https://github.com/smhosein/deep_learning_stlf

TABLE I
BASELINE ALGORITHMS

Algorithm	MAPE	MPE	Time (s)
WMA	9.51	-1.96	100
MLR	24.25	-1.47	1
MQR	12.91	-7.63	7
RT	**7.23**	-1.71	15
SVR	13.65	3.16	19

TABLE II
DEEP NEURAL NETWORK ALGORITHMS

Algorithm	200 Epocs			400 Epocs		
	MAPE	MPE	Time(s)	MAPE	MPE	Time(s)
MLP	5.62	-5.62	14	4.55	-4.54	25
DNN-W_3	**2.64**	1.61	30	2.50	1.98	56
DNN-W_4	5.71	-5.36	37	5.48	-5.32	72
DNN-W_5	4.40	1.79	38	5.98	5.45	69
DNN-SA_3	2.97	1.23	23	2.01	0.74	25
DNN-SA_4	2.88	0.23	29	2.37	0.79	42
DNN-SA_5	2.92	0.91	37	**1.84**	0.53	49
RNN	5.23	0.89	174	5.13	-0.37	359
RNN-LSTM	5.36	-1.26	880	5.27	-1.17	1528

where n is the number of data points, t is the particular time step, y_t is the target or actual value and $\widehat{y}_t$ is the predicted value.
In order to determine the cost of the prediction errors (i.e. whether the prediction is above or below the actual value) the Mean Percentage Error (MPE) is used, which is defined as:

$$\text{MPE} = \frac{100}{n}\sum_{t=1}^{n}\frac{y_t - \widehat{y}_t}{y_t} \tag{9}$$

C. Numerical Results

We first look at the baseline methods, (with the exception of MLP) in Table I. From the table we see that MLR performs the worst, with a MAPE of 24.25%, which would indicate that the problem is not linear (this is later confirmed by Figure 1). However, the RT algorithm outperforms the rest of the methods by a noticeable margin. This shows that the problem can be split into some discrete segments which would accurately forecast the load. This can be confirmed by looking at the load in Figure 1 - it is clear that depending on the time of day, there is significant overlap of the value of the load between days. Thus, having a node in the RT determining the time of the day would significantly improve accuracy. The run-time for these algorithms was quite short with WMA taking the longest due to the cross-validation step where we determined all possible coefficients in steps of 0.05.

Due to the typically long running time of DNN architectures, the algorithms were restricted to 200 and 400 epocs. From Table II, there is a clear difference when looking at the 200 epocs and the 400 epocs MAPE columns, as most of the algorithms have a lower MAPE after running for 400 epocs when compared with 200 epocs. This is especially true for the DNN-SA_3 which saw significant drops in the MAPE.

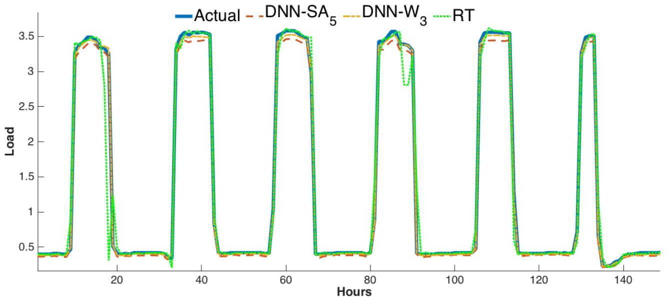


Fig. 1. Actual vs Predicted Loads

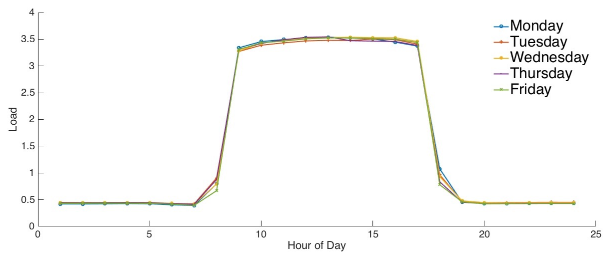
The MLP did not perform the worst in both epocs but it was always in the lower half of accuracy. This indicates that the shallow network might not be finding the patterns or structure of the data as quickly as the DNN architectures. However, it outperformed RT in both the 200 and 400 epocs. This alludes to the fact that the hidden layer is helping to capture some of the underlying dynamics that a RT cannot.

Looking at the 200 epocs column, we see that DNN-W_3 performs the best with a MAPE of 2.64%. On the other hand, the most stable architecture is the DNN-SA with a MAPE consistently less than 3%. This robustness is shown when the epocs are increased to 400 where the DNN-SA architecture outperforms all the other methods (both the baseline and deep methods). The pretraining certainly gave these methods a boost over the methods–especially due to the short training time of the DNNs–as it guides the learning towards basins of attraction of minima that support better generalization from the training data set [12]. RNNs and to an extent LSTM have an internal state which gives it the ability to exhibit dynamic temporal behavior. However, they require a much longer time to compute which is evident in Table II, as these methods had trouble mapping those underlying dynamics of the data in such a small number of epocs.

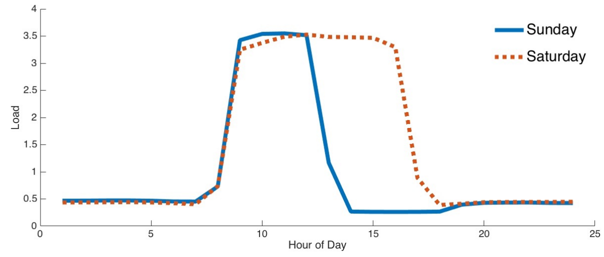
Taking both tables into consideration, most of the DNN architectures vastly outperform the traditional approaches, but DNNs require much more time to run and thus there is a trade-off. For STLF, which is a very dynamic environment, time cannot be wasted on waiting for a new model to complete its training stage. Hence, this is another reason we limited the number of epocs to 200 and 400. Table II shows that limiting the epocs did not adversely affect many of the DNN architectures as most were able to surpass the accuracy of the traditional methods (some by a lot). When selecting a model, one would have to determine if the length of time to run the model is worth the trade-off of selecting a slightly less accurate model but with a much shorter running time.

D. Daily Analysis

We know that people have different electrical usage patterns on weekdays when compared to weekends. This difference can be seen in Figure 2 which illustrates usage for a sample home. This household uses more energy during the weekdays than on weekends. There are electrical profiles that may be opposite,



(a) Weekday Electrical Usage



(b) Weekend Electrical Usage

Fig. 2. Electrical Profiles

i.e., where the weekend electrical load is more. Whatever the scenario, there are usually different profiles for weekdays and weekends.

To see how our models handle weekdays and weekends, we took the results gathered in section III-C and calculated the average MAPE for each day of the week in the test set (the 400 epoc models was used for the DNNs calculations). The average for each day of the week is tabulated in Table III. From the table, it is clear that most of the DNN algorithms have their lowest MAPE during the week. This is indicative that the patterns for weekdays are similar and as a result have more data. By having more data, DNNs are better able to capture the underlying structure of the data and thus are able to predict the electrical load with greater accuracy. Weekend predictions have a higher MAPE since DNNs require a lot of data to perform accurate predictions and for weekends this data is limited. The WMA and MQR seem to have their best day on Sunday, but have a very poor MAPE for the rest of the days. This indicates that the models have an internal bias towards Sunday and as a result fail to accurately predict the values for other days. It is clear, again, that DNN outperform the traditional methods.

E. Mean Percentage Error

In this particular domain, an electricity provider will also be interested in changes of electrical load, as opposed to absolute error in order to adjust generation accordingly, mostly because starting up additional plants takes time. This is why the Mean Percentage Error (MPE) was used. The MPE would tell that a model with a positive value ”under-predicts” the load while a negative value ”over-predicts” the actual value and they can then adjust their operations accordingly.

Many of the traditional methods had predicted more electrical load than the actual load, including MLP. However, most of the DNNs have under-predicted the load value. Looking at the best in Table II, DNN-SAs MPE values (for 400 epocs), they are all under 1% and positive, which indicates that it under-predicts the value. However, one should not use the MPE alone. An example is RNNs which have a low positive MPE, however it's MAPE in both epocs is around 5%. This indicates that RNN had a slightly larger sum of values that ”under-predicts” than ”over-predicts”, but its overall accuracy is not as good as other deep architectures.

TABLE III
DAILY MAPE VALUES

Algorithm	Sun	Mon	Tue	Wed	Thu	Fri	Sat
WMA	5.71	10.05	8.87	10.24	10.74	10.37	10.67
MLR	65.46	27.61	12.55	11.39	9.01	9.38	35.59
MQR	1.17	11.92	9.88	14.24	14.11	17.11	13.24
RT	7.45	5.99	7.63	7.37	5.98	7.26	8.87
SVR	20.70	12.96	10.73	11.53	11.63	10.90	17.40
MLP	5.18	4.62	4.43	4.27	4.31	4.70	4.34
DNN-W$_3$	2.95	1.88	2.12	2.49	2.54	2.46	3.12
DNN-W$_4$	6.67	5.45	5.25	4.88	4.61	5.65	5.83
DNN-W$_5$	7.23	5.53	5.56	6.14	6.13	5.81	5.48
DNN-SA$_3$	2.29	1.84	1.76	1.97	1.87	2.03	2.35
DNN-SA$_4$	2.67	2.19	2.00	2.14	2.27	2.55	2.82
DNN-SA$_5$	2.28	1.47	1.63	1.93	1.60	1.76	2.22
RNN	5.38	5.30	4.41	5.14	5.11	5.35	5.45
RNN-LSTM	4.25	4.34	4.96	4.55	5.64	6.97	6.13

F. Applications to Energy Efficiency

Using the results from STLF (MAPE and MPE), a company can now accurately predict upcoming load. This would mean that a power generating company, can now produce energy at a much more precise amount rather than producing a lot of excess energy that would be wasted. Since most of these companies use fossil fuels which are non-renewable sources of energy, we would be conserving them as well as reducing levels of carbon dioxide released into the atmosphere and the toxic byproducts of fossil fuels.

Another benefit of accurate load forecasting is that of dynamic pricing. Many residential customers pay a fixed rate per kilowatt. Dynamic pricing is an approach that allows the cost of electricity to be based on how expensive this electricity is to produce at a given time. The production cost is based on many factors, which in this paper, is characterized by the algorithms for STLF. By having a precise forecast of electrical load, companies now have the ability to determine trends, especially at peak times.

An example of this would be in the summer months many people may want to turn on their air conditioners and thus electricity now becomes expensive to produce as the company could have to start up additional power generating plants to account for this load. If the algorithms predict that there would be this increase in electrical load around the summer months, this would be reflected in the price that consumers would need to pay. Now, many people would not want to keep their air conditioner on all the time but use it only when necessary. Taking this example and adding on washing machines, lights and other appliances, we can see the immense decrease in energy that can be achieved on the consumer side.

IV. Related Work

The area of short-term load forecasting (STLF) has been around for many decades but deep learning has only recently seen a surge of research into its applications. Significant research has been focused on Recurrent Neural Networks (RNNs). In the thesis by [13], RNNs was used to compare other methods for STLF. These methods included modifications of MLP by training with algorithms like Particle Swarm Optimization, Genetic Algorithms and Artificial Immune Systems. Two other notable papers that attempt to apply DNN for STLF are [14] and [15]. In [14], they compare Deep Feedfoward Neural Networks, RNNs and kernelized regression. In the paper by [15] a RNN is used for forecasting loads and the result is compared to a Feedfoward Neural Network.

In the wider area of time series forecasting, more research has been done on deep learning architectures. Convolutional Neural Networks (CNN) has seen recent interest and promising performance on a number of time series tasks. [16] uses a novel CNN architecture for multivariate time series classification. Also, [17] used CNN for multichannel time series for human activity recognition. CNN was used by [18] to predict the stock market, along with events which are extracted from news text, and represented as dense vectors.

This lack of research into the crucial area of STLF and pricing prompted this paper. In addition, there is no comprehensive source of information that researchers can reference to determine whether deep architectures or classical techniques, would be the best for their scenario.

V. Conclusion

There has been significant success using deep learning for other applications, but this has not yet been illustrated for applications in the power sector. In this paper, we show comparisons of DNN and traditional approaches when applied to short term load forecasting. The results indicate that certain DNN architectures achieve greater accuracy than traditional methods. Even when the results for weekdays and weekends were analyzed, we see that DNNs still outperform traditional methods. However, DNN still suffers from long computational times but we showed that even by having a small number of epocs, DNN architectures are still preferable. Given these detailed comparisons, electrical companies can now make more informed and accurate decisions about their projected load and pricing among others.

References

[1] H. K. Alfares and M. Nazeeruddin, "Electric load forecasting: Literature survey and classification of methods," *International Journal of Systems Science*, vol. 33, no. 1, pp. 23–34, 2002.

[2] J. D. Gooijer and R. Hyndman, "25 years of time series forecasting," *International Journal of Forecasting*, vol. 22, pp. 443–473, 2006.

[3] G. Hinton, L. Deng, D. Yu, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. S. G. Dahl, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, 2012.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1097–1105.

[5] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, 2008.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[7] T. Hong, P. Wang, and H. Willis, "A naïve multiple linear regression benchmark for short term load forecasting," in *Power and Energy Society General Meeting, 2011 IEEE*, 2011.

[8] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, 1984, vol. 19.

[9] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in Neural Information Processing Systems 9*, 1996, pp. 155–161.

[10] F. Rosenblatt, *Principles of neurodynamics: perceptrons and the theory of brain mechanisms*, ser. Report (Cornell Aeronautical Laboratory). Spartan Books, 1962. [Online]. Available: https://books.google.co.uk/books?id=7FhRAAAAMAAJ

[11] F. Gers, "Long short-term memory in recurrent neural networks," Ph.D. dissertation, École Polytechnique Fédérale ds Lausanne, 2001.

[12] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *J. Mach. Learn. Res.*, vol. 11, Mar. 2010.

[13] S. Mishra, "Long short-term memory in recurrent neural networks," Master's thesis, National Institute Of Technology Rourkela, 2008.

[14] E. Busseti, I. Osband, and S. Wong, "Deep learning for time series modeling," Stanford, Tech. Rep., 2012.

[15] J. Connor, L. E. Atlas, and D. R. Martin, "Recurrent networks and narma modeling," in *Advances in Neural Information Processing Systems 4*, 1992.

[16] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, *Web-Age Information Management: 15th International Conference, WAIM 2014*. Springer International Publishing, 2014, ch. Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks.

[17] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Proceedings of the 24th International Conference on Artificial Intelligence*, 2015.

[18] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *Proceedings of the 24th International Conference on Artificial Intelligence*, ser. IJCAI'15, 2015.