Price Prediction with CNN and Limit Order Book Data: 5th Workshop on Engineering Applications, WEA 2018, Medellín, Colombia, October 17–19, 2018, Proceedings, Part I

Chapter · September 2018

DOI: 10.1007/978-3-030-00350-0_11

CITATIONS
0

READS
2,964

5 authors, including:

Jaime Nino
National University of Colombia
10 PUBLICATIONS **21** CITATIONS
SEE PROFILE

Diego León
Externado University of Colombia
10 PUBLICATIONS **6** CITATIONS
SEE PROFILE

German Hernandez
National University of Colombia
51 PUBLICATIONS **176** CITATIONS
SEE PROFILE

Javier Sandoval
Externado University of Colombia
17 PUBLICATIONS **43** CITATIONS
SEE PROFILE

Price Prediction with CNN and Limit Order Book Data

Jaime Niño[1], Andrés Arévalo[1], Diego Leon[2], German Hernandez[1], and Javier Sandoval[2]

[1] Universidad Nacional de Colombia, Bogotá, Colombia.
{jhninop,gjhernandezp,ararevalom}@unal.edu.co
[2] Universidad Externado de Colombia, Bogotá, Colombia.
{diego.leon,javier.sandoval}@uexternado.edu.co

Abstract. This work introduces how to use Limit Order Book Data (LOB) and transaction data for short-term forecasting of stock prices. LOB registers all trade intentions from market participants, as a result, it contains more market information that could enhance predictions. We will be using Deep Convolutional Neural Networks (CNN), which are good at pattern recognition on images. In order to accomplish the proposed task we will make an image-like representation of LOB and transaction data, which will feed up into the CNN, therefore it can recognize hidden patterns to classify Financial Time Series (FTS) in short-term periods. Data enclose information from 11 NYSE instruments, including stocks, ETF and ADR. We will present step by step methodology for encoding financial time series into an image-like representation. Results present an impressive performance, 74.15% in Directional Accuracy (DA).

Keywords: Short-term Forecasting; Deep Learning; Convolutional Neural Networks; Limit Order Book; Pattern Recognition

1 Introduction

In recent years developments in telecommunications, information technology and computer science have helped to transform Finance into a highly sophisticated scientific discipline capable to analyze huge flows of real time data. Large datasets and nonlinearities, data noisiness and complexity make Machine Learning (ML) techniques very appropriated to handle FTS analysis. As a result, vast number of literature reports Machine Learning applications for price prediction[11,12,7,10,5]. Works include Artificial Neural Networks, Support Vector Machines and Deep Learning. For this work, we use a Convolutional Neural Network to predict price movements. We will be working with both LOB and transaction (tick) data. As mentioned before, LOB data contains all traders intentions to negotiate an asset at a particular price and quantity at certain time t. LOB information is richer than transaction data, which only records prices and quantities exchanged at certain time t. In order to use CNN, we represent

both LOB and tick data as images. Results are very competitive when compare to other DL approaches reported in [8,4,2,16,13], with the advantage of using the same trained model for different assets.

This paper continues as follows: section two gives a brief summary of CNN, section three explains how LOB and tick data are transformed into images, section four explains full methodology from data acquisition up to image data classification, section five shows results and section six gives final remarks, conclusions, and further work opportunities.

2 Convolutional Neural Networks (CNN)

CNN are one type of network topology, widely used to applied Deep Learning (DL) principles. They have been used for Image Processing and Classification task. A CNN is a variation of a Multilayer Perceptron, which means that it is a feed-forward network, however, it requires less processing when compared to a MLP, due to the mechanism used to process input data. One of its main characteristic is to be space invariant, that is due to the convolution operator that transforms data inputs.

CNN are biological inspired, trying to emulate what happens in mammal's Visual Cortex, where neural cells are specialized to distinguish particular features. Building blocks of a CNN architecture are in charge of doing this feature detection by activating or de-activating a set of neurons. Since market agents decisions are mostly made from visual analysis of order changes in the LOB, we expect that a CNN based algorithm can learn patterns in order to help trigger trading decisions. In fact, [15,14] shown that a visual dictionary could be constructed from LOB data and that dictionary has predicting capabilities.

The two main build blocks of a CNN are the convolution layer and the pooling layer, which in conjunction with a dense layer, complete a CNN.

Convolution layer It is in charge of applying convolution operator to the input matrix, in other words it applies a kernel to filter data input. Depending on the parameters used, it can reduce or maintain input's dimensionality. The reason to convolve is to identify edges. That means to identify or separate features that later on can be used to construct more complex representations in deeper layers.

Pooling layer It is a local operator, that takes convolution output and maps subregions into a single number. The pooling operator can extract the max value of the mapped subregion (Max pooling) or the average value of the mapped subregion (Average Pooling). In other words, it gets subsamples out of the Convolution Layer.

Dense layer Finally, the deeper convolutional layer is connected to a dense layer (fully connected), from which network obtains its outputs. As mentioned before, the CNN topology may have one or more dense layers.

3 Limit Order Book and Tick data transformation

3.1 Definitions

Limit Order Book Order Book Data records market agents buy/sell intentions. It includes a time-stamp, quantity and price to buy/sell. This data is known as Limit Order Book (LOB). Formally, an order $x = (p, q, t, s)$ sent at time t_x with price p_x, quantity q_x (number of shares) and side s_x (buy / sell), is a commitment to buy/sell up to q_x units of an asset at price p_x. Orders are sorted by quoted price p and arrival time t. Sell orders have larger prices than buy orders.[9,14,6] Some other useful concepts include [9,14]:

- Spread size is the difference between the best sell and buy price.
- Bid price is the highest price among all active buy orders at time t. Conversely, Ask price is the lowest price among all active sell orders at time t. Both are called best quotes.
- An LOB $L(t)$ is the set of all active orders at time t.

Dynamics of LOB are complex[6,9], since it reflects interactions among market agents with a different point of views and different trading strategies. For a particular time t, LOB concepts are illustrated on figure 1.

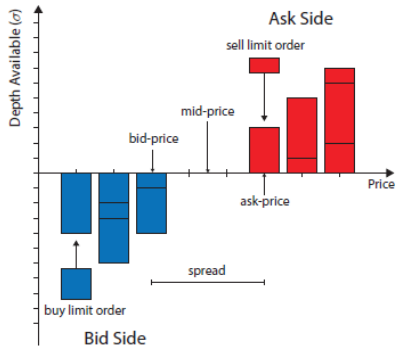


Fig. 1. LOB Snapshot, taken from [9]

When all recorded intentions are joined, they can be seen as an Image 2. On this image representation, y-axis represent prices, the x-axis is time and each point is a quantity willing to be traded.The darker the color the most quantity q at certain price p. In [15], authors used this graphic representation to cluster LOB-Patterns in order to build a classifier. Based on this work, LOB data can be seen as a list of tuples (prices-quantities) where agents expect to negotiate. Numerically, this representation can be seen as a multivariate FTS[3].

[3] Some considerations should be done, particularly related to the dimensionality of the FTS

LOB representation For a set of successive timestamps, LOB data can be represented as a matrix-like object, where column labels are timestamps, row labels are prices and the content of each cell is the number of shares to bid/ask. Each cell contains a quantity q, with subindex side s, time t and price line p. Order side could be either ask a or bid b. Because there are order imbalances, price lines subindex are k for the ask side and j for the bid side.

Table 1. LOB Matrix Representation

	t_0	t_1 ... t_n
$AskPrice_k$	q_{a0_k}	... ... q_{an_k}
$AskPrice_{k-1}$	$q_{a0_{k-1}}$	... ... $q_{an_{k-1}}$
...	...	... ...
$AskPrice_0$	q_{a0_0}	... ... q_{an_0}
$BidPrice_0$	q_{b0_0}	... ... q_{bn_0}
...	...	... ...
$BidPrice_{j-1}$	$q_{b0_{j-1}}$	... ... $q_{bn_{j-1}}$
$BidPrice_j$	q_{b0_j}	... ... q_{bn_j}

Normalizing each q_{sti} between 0-255, will produce a LOB gray scale image. However, there is a lot more information in LOB data. Because each order is recorded individually and sorted by arrival time, it is possible to aggregate volumes at the same price. By doing so you can get how many different orders (quotes) are placed at the same price. Formally, for each unique price p adds all quantities q_k, where $q = [q_1, q_2, ...q_m]$, being m the last entered order at price p.

This information is very important because is different to have many distinct agents interested at one particular price that just a few ones. However this fact, under real market conditions, goes hand in hand with how much volume (quantity) of the asset is available at that particular price p. In other words, it is important to have some sense of the distribution. It is different to have a lot volume concentrated in just one participant that distributed across many. To introduce this information in our representation, we used $max_{p_k}(q)$, for each unique price p at line k, signaling a sense of the volume distribution.

As a result, we will represent LOB data in a 4-channel representation, which can be seen as a RGBA image, where:

- R channel is only used for ask volumes q_a, 0 otherwise.
- G channel is only used for bid volumes q_b, 0 otherwise.
- B channel is only used to represent total number of placed orders at a unique price p.
- A channel is only used to represent volume distribution for a unique price p, taking $max_{p_k}(q)$

Tick Data Tick data records transactions, that is prices and quantities exchanged for a particular asset. Formally, a transaction occurs when at time t

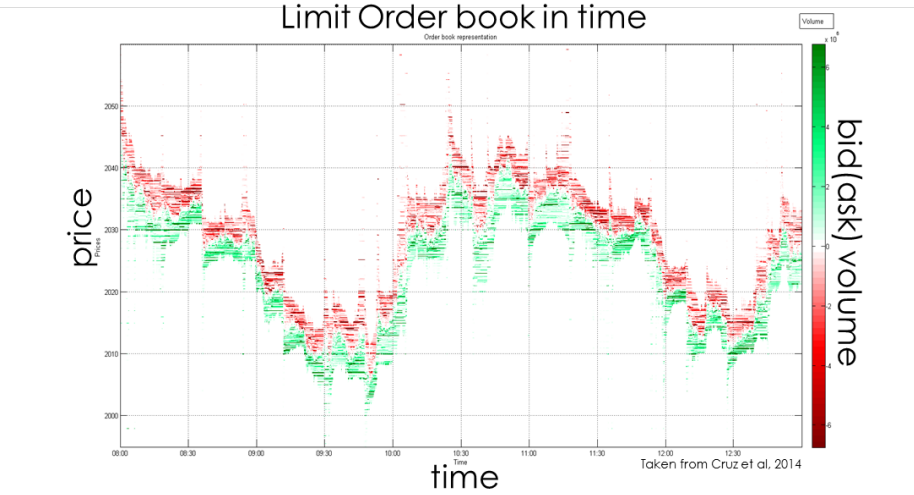


Fig. 2. LOB as Image, taken from [15]

the bid price equals the ask price. At this point, a transaction $T = (p, q, t)$ occurs, where p_T is the price, q_T is the shares quantity exchanged and t_T is the transaction time-stamp [14]. Tick data is a univariate time series[4].

Tick Data Graphical Representation As mentioned before, tick data is the most widely used data when modeling FTS. This is because is easier to obtain. LOB data is more difficult to get and usually cost a lot, not just in money terms but also in storage terms. Transactions are heavily influenced by the intentions recorded in the LOB, but they do not have the richness of LOB. Nevertheless, we expect, that in conjunction with the LOB, to yield better results. In other to homogenize inputs, it is necessary to transform tick data into a matrix-like representation. In [18], authors show a step by step methodology that transforms univariate time series into an image representation. This transformation is called Grammian Angular Field (GAF), which consists of the following steps[5]:

- Time series normalization between $[-1, 1]$
- Time series is converted from Cartesian to Polar coordinates

$$\phi = arccos(x_t); r = \frac{t_i}{N}, t_i \in N \tag{1}$$

[4] Bivariate if volumes are included
[5] For full details please refer to [18]

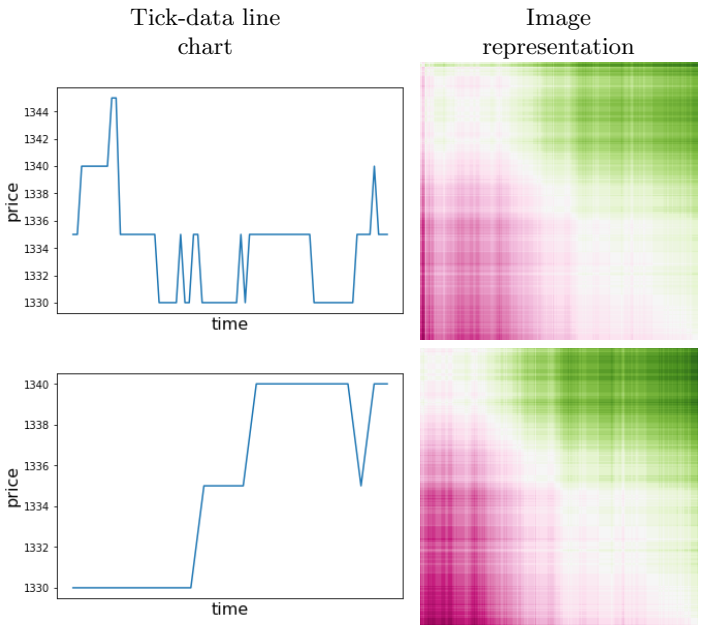
– GAF matrix deduction, defined as:

$$\begin{bmatrix} < \boldsymbol{x_1}, \boldsymbol{x_1} > & \dots & < \boldsymbol{x_1}, \boldsymbol{x_n} > \\ < \boldsymbol{x_2}, \boldsymbol{x_1} > & \dots & < \boldsymbol{x_2}, \boldsymbol{x_n} > \\ \vdots & \ddots & \vdots \\ < \boldsymbol{x_n}, \boldsymbol{x_1} > & \dots & < \boldsymbol{x_n}, \boldsymbol{x_n} > \end{bmatrix}$$

, where $< \boldsymbol{x}, \boldsymbol{y} >= x \cdot y - \sqrt{I - x^2} \cdot \sqrt{I - y^2}$

Authors in [18] used for non-Financial Time Series. In this paper, we apply the same general steps in order to obtain a graphical version of the tick data, as illustrated in table 2

Table 2. Original tick data vs Image representation of tick data



One advantage of this transformation is that marks peaks of the input signal, based on intensity levels 2. This is useful for pattern recognition because it helps to differentiate price variances within the original signal. On the other hand, the transformed input can be rolled back to the original signal[18]. We expect that on this new space, patterns could be easier to identify since CNN's learning capabilities have been proven good in frequency spaces. In fact, in a previous work we show how a wavelet transformation improve results over a

pure time-space approach[2][6]. Next section will give step by step explanation for our experiment.

4 Classifying Financial Time Series with CNN

4.1 Why a CNN for FTS Classification

- Firstly, DL models have demonstrated a greater effectiveness in both classification and prediction tasks. Its superiority is due to the fact that they are able to learn useful representations from raw data, avoiding the local minimum issue of ANNs, by learning in a layered way using a combination of supervised and unsupervised learning to adjust weights W.
- Secondly, DL applications in computational finance are limited [3,4,8,17,19] and as long as it goes to our knowledge, there is no publication applying CNN to FTS using LOB data for short-term periods forecasting.
- Thirdly, CNN are good for pattern recognition, real traders have told us that they try to identify patterns by following buy/sell intentions in a numeric form. In a previous work, [14] identified volume barriers patterns to translate them into trading decisions and [15] identified visual patterns and cluster them into a bag of words model to predict market movements. As a result of these works, we decided to extend them and use a more suitable technique for pattern recognition such as CNN on image-like representation of market data.
- Finally, by applying input space change (from time to a frequency), we expect that CNN will recognize patterns more effective, indeed authors in [2] improved their results by using wavelets to represent high frequency data of several financial assets[1]. Even tough our images are not natural ones, we expect that CNN's layers are capable to distinguish simple frequency changes (edges) at lower layers in order to identify more complex patterns at deeper ones.

4.2 Experimental setup

- Data acquisition: Original dataset is compose of LOB and transaction data for 11 instruments listed on NYSE, from Dec 24, 2015 to Apr 27, 2017, from EDGX venue. Dataset includes information from US stocks (AIG, C, MSFT, FB), ETF (SPY) ADR (EC, CIB, AVAL, AVH, RBS, DB). 4,371,945 LOB files transformed into 1,491,240 image like representation, including LOB and tick data totaling 6.2 GB in disk. For training and testing purposes we use a dataset containing 350,000 and 35,000 images respectively.[7]
- Data preparation: For each stock, data normalization was conducted, taking into account some considerations which include handling of no orders at some price levels in LOB data, some liquidity constrains and event of LOB. Details are given in the next subsection.

[6] We used other DL topologies

[7] Data provided by DataDrivenMarket Corporation

- Data transformation: For each stock both LOB data and tick data are transformed to an image-like representation, following the methodology previously explained.
- CNN modeling: We chose a base CNN architecture. We trained and test it with transformed data.
- CNN comparison again other DL topologies: We compare results achieved results obtained in this work against others, which have been used for similar problems (Short-term forecasting) but different Deep Learning topologies (RNN, LSTM, Multilayer Perceptron, DBN)

Following paragraphs will provide further details of our experimental setup.

4.3 Data preparation

Data Normalization For each stock, prices, volumes (quantities) and a number of orders at the same price were normalized between (0-1]. Given the fact that LOB data may have price levels with no demand /offer, minimum values were reduced by a small factor so that minimum values had a small value above zero. Data normalization by stock facilitates magnitude equilibrium across all stock data, regardless their nominal prices or volumes.

Handling of LOB events We took an event-based approach, that is to analyze a fixed number of LOB events (10 in this case). Each event is record at 5 seconds interval. This means that the LOB matrix explained in section 2 1, was partitioned into fixed segments of 10. We use the same window to create the corresponding image for tick data. Figure 3 illustrates procedure described above.

Handling of LOB Deepness Given deepness asymmetry between buy and sell sides, we decided to work with LOB data of 5 lines depth for each book side. Prices start from the best quotes (down/up) side depending (bid/ask).

Additional considerations It is important to note the following:

- Prices with no volume will have a 0 value. This value will be always different for the lowest volume after normalization, as mentioned before.
- For modeling purposes, we resize each image to be 10 width and 40 height.

4.4 CNN Modeling

Data input Four channel images are used, one for LOB data another for tick data. A five dimensional tensor is used for data input, with size [n,2,10,40,4]. The first dimension is the number of samples, second one the number of images categories (LOB / tick) an the other three, image dimensions (Width, Height, Channels).

	TimeStamps							
Prices	**0**	**1**	**...**	**9**	**10**	**...**	**n-1**	**n**
AskPrice$_k$	$q_{a\_(0,k)}$	$q_{a\_(1,k)}$	...	$q_{a\_(9,k)}$	$q_{a\_(10,k)}$	...	$q_{a\_(n-1,k)}$	$q_{a\_(n,k)}$
AskPrice$_{k-1}$	$q_{a\_(0,k-1)}$	$q_{a\_(1,k-1)}$	...	$q_{a\_(9,k-1)}$	$q_{a\_(10,k-1)}$	...	$q_{a\_(n-1,k-1}$	$q_{a\_(n,k-1)}$
...	...	...	...	...	...	...	...	...
AskPrice$_1$	$q_{a\_(0,1)}$	$q_{a\_(1,1)}$	...	$q_{a\_(9,1)}$	$q_{a\_(10,1)}$	...	$q_{a\_(n-1,1)}$	$q_{a\_(n,1)}$
AskPrice$_0$	$q_{a\_(0,0)}$	$q_{a\_(1,0)}$	...	$q_{a\_(9,0)}$	$q_{a\_(10,0)}$	...	$q_{a\_(n-1,0)}$	$q_{a\_(n,0)}$
BidPrice$_0$	$q_{b\_(0,0)}$	$q_{b\_(1,0)}$	...	$q_{b\_(9,0)}$	$q_{b\_(10,0)}$	...	$q_{b\_(n-1,0)}$	$q_{b\_(n,0)}$
BidPrice$_1$	$q_{b\_(0,1)}$	$q_{b\_(1,1)}$	...	$q_{b\_(9,1)}$	$q_{b\_(10,1)}$	...	$q_{b\_(n-1,1)}$	$q_{b\_(n,1)}$
...	...	...	...	...	...	...	...	...
BidPrice$_{j-1}$	$q_{b\_(0,j-1)}$	$q_{b\_(1,j-1)}$	...	$q_{b\_(9,j-1)}$	$q_{b\_(10,j-1)}$	...	$q_{b\_(n-1,j-1}$	$q_{b\_(n,j-1)}$
BidPrice$_j$	$q_{b\_(0,j)}$	$q_{b\_(1,j)}$	...	$q_{b\_(9,j)}$	$q_{b\_(10,j)}$	...	$q_{b\_(n-1,j)}$	$q_{b\_(n,j)}$

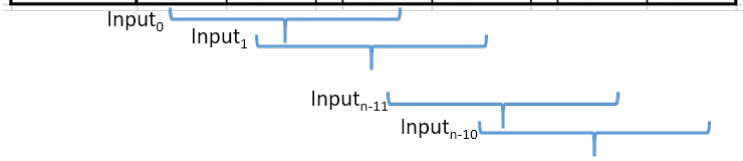


Fig. 3. LOB Events

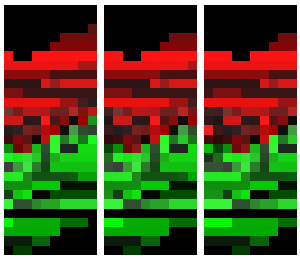


Fig. 4. LOB data images

Data labeling Data will be classified in three different classes as shown on Table 3. Classes exhibit unbalance, which could be adjusted by reducing the interval that determines flat movements.

CNN architecture We use a standard CNN architecture, which consists of (Input + Conv + Pool + Conv + Pool + Dense + Dropout), input images' size is 10 X 40. Experiments include both LOB and tick data. We used TensorFlow.

Special considerations for training size included dropout at 40% and batch size at 100. The dataset was split into 90%for training, 10% for testing.

Table 3. Three class Rules

Price direction	Rule	Class
Upward movement (27.92%)	Last tick price above 0.015% vs last tick of the previous window	0
Downward movement (27.75%)	Last tick price below -0.015% vs last tick of the previous window	1
Flat movements (44.32%)	Otherwise	2

5 Results

5.1 Model comparison against other DL topologies

The CNNs were used to classify the three target classes (Up, Down, Flat). Table 4 shows the proposed model performance (74.15%). As observed on table 4, proposed model is very competitive against similar works, with the advantage that one model runs for several assets. Metrics are displayed on table 5

Table 4. Comparison against other DL topologies

DL Topology	Classes	Data used	Directional Accuracy
Multilayer Perceptron[2]	2	1-Stock, tick data	66%
Deep Belief Network[13]	2	1-Stock, LOB + Tick data	57%
Deep Neural Network + Wavelet[1]	2	19 stocks, Tick data, one model per stock	[64% - 72%]
Proposed Model (CNN)	**3**	**12-stocks, LOB + Tick data**	**74.15%**

Table 5. Metrics at different steps

Metric	Step 150,000	300,000	450,000	600,000
Accuracy	73.89%	73.99%	74.05%	**74.15%**
Mean Class Accuracy	74.73%	74.70%	74.72%	**74.90%**
MSE	0.642	0.640	0.634	**0.629**
True Positives	8495	8438	8440	**8501**
False Positives	2576	2508	2468	**2497**
False Negatives	2399	2456	2454	**2393**

6 Conclusion and Future Research

CNN for FTS prediction purpose worked well. DA shows that results are very competitive, in fact, better than other approaches tested before [13,15,2]. As ex-

pected, performance improves when both LOB and tick data is used in conjunction, and the main reason is simple: there is more market information. Image-like representation is useful and even could be extended, that is it is possible to have more channels in the original input image (matrix).

Perceived advantages

- One network for multiple assets. It is not usually the case, given the fact that each asset has it owns dynamics. Image-like representation homogenize inputs, resulting in an image representing market information, finding patterns across all image set, regardless the asset.
- Lifetime of trained model. In financial applications frequent retraining is the norm. This approach extends the lifetime of the trained model due to the time invariance fact associated with images.

Perceived disadvantages

- It is a data intensive technique. As there are more images for training, results will improve.
- Preprocessing could be tricky. There are a lot of details to take into account when transforming raw data.

It is necessary to perform out of sample test, to check DA with unseen data, as well as possible re-training periods. For FTS it is necessary to re-train models frequently given the fact that Financial Markets exhibit trends in both long and short periods of times. Moreover, it is necessary to test different flat intervals. We think that there are a lot of possibilities for improvement, including a combination with other topologies such as (LSTM and CNN), and to code more information in more channels, for example, technical information, time among others.

References

1. Arévalo, A., Nino, J., León, D., Hernandez, G., Sandoval, J.: Deep learning and wavelets for high-frequency price forecasting pp. 385–399 (2018), https://link.springer.com/chapter/10.1007/978-3-319-93701-4_29
2. Arévalo, A., Nino, J., Hernández, G., Sandoval, J.: High-Frequency Trading Strategy Based on Deep Neural Networks pp. 424–436 (2016), http://link.springer.com/10.1007/978-3-319-42297-8_40
3. Arnold, L., Rebecchi, S., Chevallier, S., Paugam-Moisy, H.: An Introduction to Deep Learning. ESANN (2011), https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2011-4.pdf
4. Chao, J., Shen, F., Zhao, J.: Forecasting exchange rate with deep belief networks. In: The 2011 International Joint Conference on Neural Networks. pp. 1259–1266. IEEE (7 2011), http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6033368http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6033368
5. Chen, M., Ebert, D., Hagen, H., Laramee, R.S., van Liere, R., Ma, K.L., Ribarsky, W., Scheuermann, G., Silver, D.: Data, information, and knowledge in visualization. IEEE Computer Graphics and Applications 29(1), 12–19 (Jan 2009)

6. Cont, R., Stoikov, S., Talreja, R.: A Stochastic Model for Order Book Dynamics. Operations Research 58, 549–563 (2010)
7. De Goijer, J., Hyndman, R.: 25 Years of Time Series Forecasting. Journal of Forecasting 22, 443–473 (2006)
8. Ding, X., Zhang, Y., Liu, T., Duan, J.: Deep learning for event-driven stock prediction. Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (ICJAI) (2015), http://ijcai.org/papers15/Papers/IJCAI15-329.pdf
9. Gould, M.e.a.: Limit Order Books. Quantitative Finance 13, 42 (2010)
10. Hamid, S., Habib, A.: Financial FOrecasting with Neura Networks. Academy of Accounting and Financial Studies Journal 18, 37–56 (2014)
11. Huang, G.e.a.: Trends in extreme learning machines: A Review. Neural Networks 61, 32–48 (2015)
12. Längkvist, M., Karlsson, L., Loutfi, A.: A review of unsupervised feature learning and deep learning for time-series modeling. Pattern Recognition Letters 42, 11 – 24 (2014), http://www.sciencedirect.com/science/article/pii/S0167865514000221
13. Nino, J., Hernandez, G.: Price direction prediction on high frequency data using deep belief networks. In: Applied Computer Sciences in Engineering. pp. 74–83. Springer International Publishing (2016)
14. Sandoval, J.: Empirical Shape Function of the Limit-Order Books of the USD/COP Spot Market. ODEON 7 (2013), https://ssrn.com/abstract=2408087
15. Sandoval, J., Nino, J., Hernandez, G., Cruz, A.: Detecting informative patterns in financial market trends based on visual analysis. Procedia Computer Science 80, 752 – 761 (2016), http://www.sciencedirect.com/science/article/pii/S1877050916308407, international Conference on Computational Science 2016, ICCS 2016, 6-8 June 2016, San Diego, California, USA
16. Shen, F., Chao, J., Zhao, J.: Forecasting exchange rate using deep belief networks and conjugate gradient method. Neurocomput. 167(C), 243–253 (Nov 2015), http://dx.doi.org/10.1016/j.neucom.2015.04.071
17. Takeuchi, L., Lee, Y.: Applying Deep Learning to Enhance Momentum Trading Strategies in Stocks (2013)
18. Wang, Z., Oates, T.: Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks (2015), https://pdfs.semanticscholar.org/32e7/b2ddc781b571fa023c205753a803565543e7.pdf
19. Yeh, S., Wang, C., Tsai, M.: Corporate Default Prediction via Deep Learning (2014), http://teacher.utaipei.edu.tw/~cjwang/slides/ISF2014.pdf