Journal of Engineering Science and Technology
Vol. 17, No. 5 (2022) 3174 - 3186

STOCK MARKET PREDICTION USING DEEP LEARNING APPROACH

JOHN-SYIN ANG*, KOK-WHY NG, FANG-FANG CHUA

Faculty of Computing and Informatics, Multimedia University,
Persiaran Multimedia, 63100, Cyberjaya, Selangor, Malaysia
*Corresponding Author: johnsyin97@gmail.com

Abstract

LOB tracks the outstanding limit order for a stock or other security. LOB data is often used as an input for high-frequency trading and price prediction. Recent studies have shown that the DL methods are superior to the ML methods in stock prediction because they can extract important features for prediction. There is no existing study on stock prediction that profiles stocks. This can be overgeneralised as it predicts each stock equally when they have similar historical price movements. Moreover, the existing high-frequency stock price prediction models do not consider learning the long-term information. The objective of this study is to find out whether profiling the stocks and extending the length of the input data can further improve the prediction accuracy. The method used for this study is ESET-ConvNet, a method, which is proposed to combine two existing models, TCN and SENet with an additional embedding layer to learn the embedding of each stock. The result shows that profiling stocks and extending the input size led to an improvement in accuracy, with a trade-off in computation time. This study conclude that the main contribution of this study is a model architecture with better accuracy in stock prediction problem.

Keywords: Deep learning, High frequency trading, Limit order book modelling, Stock prediction, Time series classification.

1. Introduction

To better optimise the portfolio, many studies have attempted to use various quantitative techniques to predict stock movements. Stock price prediction approaches often differ in the context of input, output, and horizon. Some variants include prediction of prices, price movements, trading signals, or self-defined labels to support a particular trading strategy. The goal of a stock price prediction model may be to optimise a portfolio in terms of risk-adjusted return over the long run or simply to increase the number of profitable trades in a given period. Portfolio construction is one of the main tasks in asset management and can be further divided into weight optimisation and trading strategies [1]. Nevertheless, most researchers have been tackling the problem of stock prices prediction separately from asset management problems.

Stock price prediction is one of the sub-problems to support decision-making in the stock market. However, there are still various objective functions in prediction models. The common approaches define it as a price prediction problem, which can be further divided into a regression [2] or a classification [3, 4] problem. The regression problem predicts the actual continuous value of prices or their changes, while the classification problem predicts discretized categorical values of price changes; such as the classification of the future direction of stocks.

The factors involved in the movement of stock prices come in different forms and lead to chaotic prediction problems with a lot of noise. Researchers often focus on a few factors in stock market forecasting because they need to be transformed independently using specific techniques. Since stock prices are driven by purchases and demands, it is logical to think in terms of an investor. The common features that an investor would look for are fundamental factors [5], technical factors [6], macroeconomic factors [7], and investor sentiment [8].

In addition, it is popular to use raw LOB data or raw stock price data to support decisions; these models are also mimicked using artificial intelligence [9]. Modelling stock price data and LOB data is often considered a time series problem. They are in the form of sequences and are not meaningful unless interpreted in the context of multiple timesteps. LOB data is widely used in the industry because it contains finer details from the market's limit orders, but data collection can be expensive, and pre-processing can also be time-consuming. Nonetheless, Ntakaris et al. [10] collected and pre-processed the LOB data of 5 stocks with 10 days of trading events from the ITCH feed in 2010, which is referred to as FI-2010 data. So far, this dataset is the ideal benchmarking dataset for modelling LOB data.

DL has been massively used in multiple ML problems with unstructured input, such as sequenced data. The main difference between DL and other ML models is that DL can engineer features that are correlated to the correct target or label from raw data in a black box method. DL is known to be an excellent model for learning image, video, and sequential data, including natural language and time-series data. DL eliminates the need to handcraft features, although they are not mutually exclusive, DL can extract more important features by learning the labels directly.

An DL model was proposed for predicting future mid-prices from modelling LOB data. Our model is a hybrid of Temporal Convolutional Layers [11], Squeeze-and-Excitation Layers [12], and embedding layers inspired by word embedding.

The advantages of our model are its ability to learn the representation of each stocks and the expansion of learnt input.

This section gives the structure of the rest of the paper. Section 2 gives an overview of the background of time-series and stock prediction problems. In Section 3, the architecture of our model is presented. In Section 4, the experimental results are analysed. In Section 5, the study is concluded and possible direction for future research is suggested.

2. Background

A large number of researchers have implemented non-DL models to solve the stock prediction problem. ML models can be referred to as both single and ensemble models; single models include Naïve Bayes and SVM while ensemble models include RF and Gradient Boosting Trees [13, 14]. Recently, DL has received much attention. DL models are capable of solving problems with sequential data, including time-series data. Model architectures such as CNN, RNN, and LSTM have been evaluated to investigate the capability of these black-box models in time series forecasting [15, 16].

Traditionally, a common method for aggregating price data is to compute technical indicators that can summarise the entire period [17]. Many researchers using non-DL models and price data have used this method [14, 18-21]. Popular indicators include stochastic and momentum [14, 20]. There are some problems with this method of using technical indicators. First, there may be information loss because the calculated characteristics were aggregated from the price. Arguably, this method of using technical indicators is based on assumptions and subjective knowledge that they are meaningful for prediction; therefore, the success of a model depends on the accuracy of these assumptions, in the worst case, these indicators provide noises but no information for the model [16]. Recently, the use of DL has become trendier. One of the main use cases of DL is the extraction of features from time-series data that do not contain subjective information. Some empirical analyses argued that DL is superior to some ML models such as SVM and RF [16, 21].

2.1. Mid-price prediction with LOB data

The detail of LOB data makes it suitable for providing information for automated algorithmic trading in real-time at high frequency. Since Ntakaris et al. [10] have compiled a benchmark dataset for LOB data, the FI-2010 dataset, this problem has been standardised and continues to receive considerable attention.

Existing works

One of the earliest mid-price prediction approaches is to use the BoF method to extract features extraction before feeding it to an MLP model [22]. The BoF method was originally used to extract information from images using histogram representations. In this problem, the BoF method outperforms many feature extraction methods, including PCA, LDA, and Autoencoder. However, many analyses show that the DL methods significantly outperform the aforementioned feature extraction methods.

Passalis et al. [23] have proposed a novel normalisation method specifically for time series data, DAIN. By using DAIN to normalise MLP, CNN, or RNN, the performance is significantly better than other normalisation methods such as standard normalisation, sample average normalisation, batch normalisation, and instance normalisation. The authors argued that standard normalisation is suboptimal when the input time series is not from a unimodal Gaussian distribution, they proposed DAIN to normalise time series in a mode-aware manner. Moreover, Tsantekidis et al. [24] proposed to use a hybrid of CNN and LSTM. They have obtained better results than SVM, MLP, CNN, and LSTM. The focus of DAIN is mainly on preprocessing time-series data.

Moreover, a novel model, TABL, is proposed to use temporal attention on linear layers [25]. The main concept is to give different weights to temporal instances based on their importance, rather than giving them equal weights. By paying temporal attention to the first linear layer in a bilinear layer model, it is able to outperform SVM, MLP, CNN, and LSTM by a large margin.

A model called DeepLOB achieves the state-of-the-art result on the FI-2010 dataset recently [16]. They used a hybrid of Inception Network and LSTM. They used CNN and Inception Modules for feature extraction from sequence input. The authors have also done a comprehensive analysis of their model. First, they tested the performance of their model on FI-2010. Their model outperformed many other models including CNN, LSTM, BoF, and TABL, which was already discussed. They also trained their model with 5 stocks from another stock exchange and then test the model with 5 other stocks that did not fit the model during training. Surprisingly, they get two indistinguishable results, which means that the model is able to generalise to the price patterns of different stock counters. They also ran a trading simulation, with a few simple assumptions for intraday trading. Their model has been shown to be significantly profitable based on the t-statistics. Last but not least, they have implemented a method called Local Interpretable Model-agnostic Explanations [26] for sensitivity analysis, a method that helps to highlight the important inputs that contribute to each prediction and facilitates interpretability of a prediction.

2.2. Research Gap

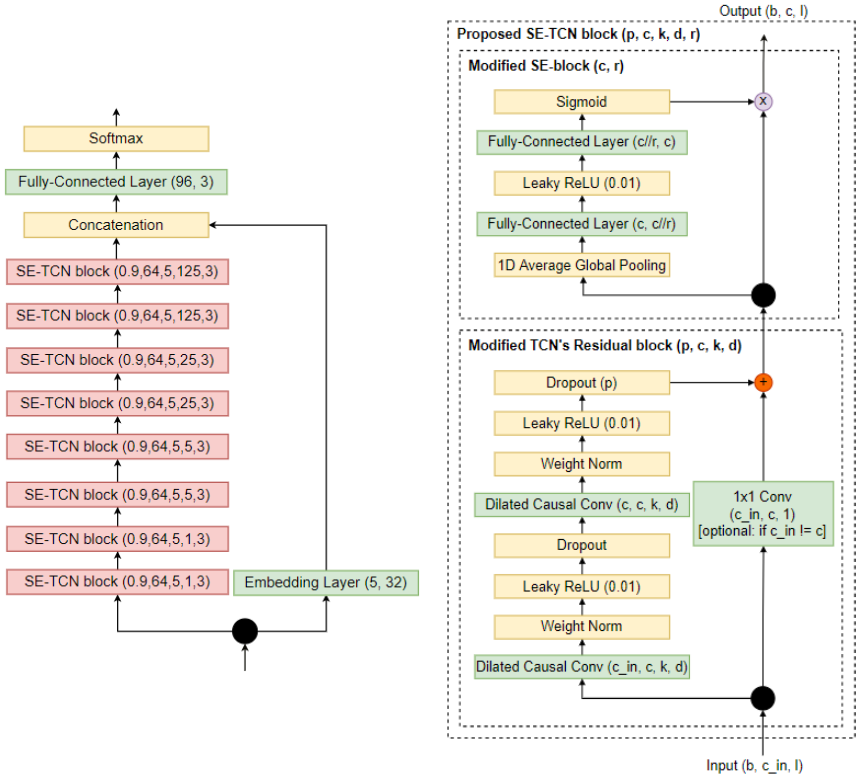
As far is known, there is no existing study that deals with the representation of the latent spaces of individual stocks. Moreover, existing works have used only the most recent input horizon and have omitted long-term information. In this study, stock embedding is investigated and sparse learning of larger inputs.

3. Model Architecture

This study proposed a model architecture called ESET-ConvNet. ESET-ConvNet is a hybrid network consisting of many SE-TCN blocks and an embedding layer as shown in Fig. 1(a). As Fig. 1(b) shows, a SE-TCN block consists of a modified block of TCN [11] and a modified block of SENet [12].

Similar to DeepLOB [16], ESET-ConvNet is proposed to use a CNN-like architecture to automate feature extraction. TCN [11] is a variant of CNN proposed to efficiently learn long sequences of time series data. The main difference between DeepLOB and ESET-ConvNet is that the latter did not use LSTM because the

training or the forwarding propagation process can take a lot of time if the sequence length is long. In the DeepLOB model [16] which uses LSTM, each input sequence is chopped to a length of 100 time steps, which inspired us to try to learn from the entire sequence for possible long-term dependency. ESET-ConvNet also consists of an embedding layer to characterise and profile each stock. The purpose is to train the model in a stock-aware manner.



(a) ESET-ConvNet architecture and hyperparameters used in this study. **(b) SE-TCN block, the key component of ESET-ConvNet [11, 12].**

Fig. 1. Architectural elements in an ESET-ConvNet.

In addition to the TCN and embedding layers, this architecture also modified a SE-block from the SENet [12]. SENet was originally proposed to solve the problem that the features in a CNN are always weighted equally. SENet can dynamically scale the weights based on the learnt importance in a computationally inexpensive fully-connected layer. The key idea is to extend TCN architecture to incorporate this concept and have achieved significant improvements as a result. In ESET-ConvNet, the SE-block is slightly modified such that it suits the time series input because the original architecture was initially proposed to solve computer vision problems.

ESET-ConvNet has slightly modified versions of SE-block and TCN's residual block compared to the original work [11, 12]. Each ReLU layer of both TCN and SE-blocks was changed to Leaky ReLU [27]. Without this change, the optimisation process would get stuck at a local minimum after the first few epochs. Leaky ReLU

helps to prevent dead ReLU at a zero slope. In ESET-ConvNet, the SE-block has been slightly modified to accommodate the time-series input as the original architecture was originally proposed for solving computer vision problems. Originally, the SE-block uses a 2-dimensional global pooling layer. In our work, a 1-dimensional pooling is used as the input data is one-dimensional time-series data.

Details of each layer

The main concept of ESET-ConvNet is to use SE-TCN blocks to extract information from time-series data and embedding layer to profile stock characteristics.

a) Embedding Layer: the embedding layer is identical to learning the word embedding representation in natural language processing problems. This layer will be used to learn the feature or representation of the stock itself. The output of this layer will be concatenated with static features extracted from time-series since the output is already static and does not need to be processed by a temporal network.

b) SE-TCN block

- Dilated Causal Convolutional Layer [11]: The main concept of TCN consists of a dilated causal convolutional layer, with exponentially increasing dilation parameters. In this way, the receptive field covered by the network grows exponentially with the number of layers. The word "causal" here means that the output is chopped so that no future input leaks to the layer. Each layer requires 1 additional hyperparameter d compared to a normal convolutional layer. Fig. 2 [11] shows an illustrative example visualising 3 dilated causal convolutional layers with d=1, 2, and 4 respectively.

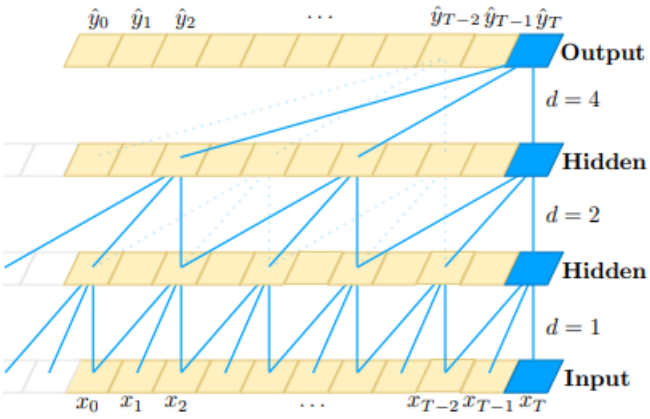


Fig. 2. 3 Causal Dilated Convolutional Layers with d=1, 2, and 4 [11].

- WN [28]: WN is used to reduce training time by normalising the weight tensor based on the direction of the weight. The WN layer also reduces the time for tuning the hyperparameters because WN can automatically decay the learning rate faster if it is too large. WN is also cheaper to implement and less sensitive to mini-batch noise than Batch Normalization [28]. In Eq. (1), *g* represents the size of the weights and *v* represents the direction of the weights.

$$w = g\frac{v}{||v||} \tag{1}$$

- Leaky ReLU [29]: ReLU is a popular activation function in convolutional networks because it is much cheaper to compute than sigmoid and tanh, but it suffers from the dead-ReLU problem. When dead-ReLU problem occurs, the output and gradient will be stuck at zero. Leaky ReLU is a modification of ReLU to solve this problem and ensure that there is at least a very small gradient to recover learning in the long run. In Eq. (2), a is usually set to a small value such as 0.01.

$$f(x) = \begin{cases} ax, if\ x < 0, \\ x, if\ x \geq 0 \end{cases} \tag{2}$$

- Dropout [30]: Dropout is applied to regularise the network. Dropout randomly zeroes out 1-p probability of the input in each training step. This process can effectively prevent the network from overfitting.
- Modified SE-block [12]: The squeeze-and-excitation mechanism works by first pooling each channel and then using 2 fully-connected layers to learn the weight for scaling the channel. This model is effective in finding out important feature channels. This model improves LSVRC 2016's winner model by a large margin with low computational cost. The hyperparameter r represents the reduction, where the intermediate "squeezing" hidden layer would have $c//r$ hidden units.

4. Experiment Details

4.1. Data, software and hardware

The FI-2010 dataset consists of 394,337 records, where one record represents one time step for one stock. It consists of 5 stocks with an unequal number of time steps depending on the number of limit order events for each stock. Each time step consists of 144 input features and 5 target variables. Each target variable specifies a different prediction horizon, k, i.e., the number of future time steps to predict. For each k, there are 3 classes that indicate the direction of the future mid-price, which can be either upward, stationary, or downward. Table 1 has shown the class distribution for k=10, 20, and 50, which are focused on in this study.

Table 1. Class distribution.

	k=10	k=20	k=50
Upward	71,429	92,079	126,072
Stationary	252,845	212,480	146,681
Downward	70,063	89,778	121,584

All 144 input features are used to include both raw features and features engineered in the original compilation of FI-2010 [10]. This study follows a common setup of previous work for the train-test split process and the selection of prediction horizon, k=10, 20, and 50. The dataset consists of 10 days of data, and the first 7 days of data is used for training and the last 3 days of data is used for testing. The original dataset contains 3 sets of inputs preprocessed using different normalization techniques. This study chose the input that was normalized with the decimal precision technique. This study is conducted with the use of the Pytorch framework [31] to train the models on Google Colab.

4.2. Optimisation

The AdamW optimisation algorithm [32, 33] is used with a learning rate of 0.001, β1 of 0.9, β2 of 0.999, ε of 1e-8 without any weight decay. For each prediction horizon, 50 training epochs has been runned with a batch size of 128. The sum of cross-entropy loss is optimized with 3 classes weighted by the class ratio in the train set since the dataset is highly imbalanced. Gradient clipping is applied to clip the gradient below 1 to prevent gradient from exploding or vanishing.

Unlike other problems, the input data has 5 samples or stocks with long sequence to predict, which makes batch sampling for optimisation slightly different. For each sequence, 128 consecutive labels are taken as one batch, then use the input from the beginning until the end of the correspond 128 time step. For example, if the current batch predicts y_{128} to y_{255}, it will use the entire sequence from x_0 to x_{255} as input.

4.3. Experiment result

Table 2 shows the performance of ESET-ConvNet and other existing models. Each sub-table represents a different prediction horizon k, e.g., k=10 means the prediction of the mid-price after 10 trading events. The performance is measured by calculating the weighted average of the F1 scores. Since the dataset is highly imbalanced, it is recommended to use weighted metrics such as the F1 score for comparison with fairness [10]. In addition to the performance of our model, Table 2 also report the results of several models including CNN [34], LSTM [35], DAIN [23], TABL [25], and DeepLOB [16]. All the results reported in the table are the best result when there is more than one variant in any of the included studies.

Table 2. Experiment results.

	F1 % (*k*=10)	**F1 % (*k*=20)**	**F1 % (*k*=50)**
CNN [34]	55.21	59.17	59.44
LSTM [35]	66.33	62.37	61.43
DAIN (MLP) [23]	66.92	65.31	-
TABL [25]	77.63	66.93	78.44
DeepLOB [16]	83.40	72.82	80.35
ESET-ConvNet	**84.61**	**80.04**	**83.34**

Table 3 shows the computation time for a forward pass in milliseconds (ms), the number of parameters, and the input length of the existing models. Since the ESET-ConvNet is proposed for learning longer dependencies, it uses a longer input length that causes a longer forward propagation time.

Table 3. Computation time, number of parameters, and input length.

	Forward (ms)	**No. parameters (k)**	**Input length**
CNN [34]	**0.025**	768	100
LSTM [35]	0.061	-	100
TABL [25]	0.229	-	100
DeepLOB [16]	0.253	**60**	100
ESET-ConvNet	14.814	396	1373

4.4. Discussion

By embedding each stock and learning a larger input, ESET-ConvNet is able to achieve the highest F1-score among the existing models. However, ESET-ConvNet takes much more time to perform one forward propagation because the number of parameters and the length of input are much higher than DeepLOB. In short, it is shown that the information of larger input size and stock embedding is useful to improve the prediction accuracy.

In practise, although ESET-ConvNet may have a large impact, it is more suitable for a larger horizon. In the benchmark dataset, the interval of 2 consecutive events ranges from a few milliseconds to minutes [10]. It is very likely that ESET-ConvNet misses the opportunity to find the desired sequence for a horizon like k=10. The biggest improvement is the performance for a prediction horizon of k=20, it has also been shown that ESET-ConvNet is more robust regardless of the prediction horizon, so more use cases can be explored for stock prediction, e.g., swing trading and day trading rather than high-frequency trading.

This study also presents the model architecture of ESET-ConvNet as a general approach for other use cases of time-series modelling. ESET-ConvNet was proposed based on the study [36] which explores existing DL models for modelling time-series in different domains or even general use cases and not only for stock prediction models. The latent space of each input series offers great potential for performance improvements with a very small additional number of parameters. For example, in the temperature prediction use case, embedding countries allows the model to adjust the prediction or learnt representation when a similar pattern occurs in another instance of a country. To accommodate different use cases with different effective input length, the length of the receptive field can be easily adjusted by changing the number of layers, the kernel size, or the dilation of the SE-TCN block.

Our model does not contain stateful components such as LSTM. This means that the model can be further parallelised, which is another way to make this model more practical. The convolutional filters can be processed in a distributed system so that the computation time can be split.

5. Conclusion

Existing work lacks some information from the input that could improve the prediction accuracy. In this study, these research gaps are being addressed, which include learning the latent space of the stock and increasing input length. This study has also shown that these actions lead to a better F1-score, although the problem of inefficient forward propagation needs to be further addressed.

This study introduce a hybrid model of embedding layer, TCN, and SENet to predict stock prices using LOB data. DL is able to extract features from LOB data and the latent space of each stock profile. It is the first model to learn the profiles or latent space of each stock among the existing stock prediction studies. Also, this study has experimented the first LOB model that does not omit any long-term information from the data unlike the other existing studies. The experiment compared the model with other previous models using the FI-2010 benchmark dataset. By creating profiles for each stock each stock and extending input size, the result has shown higher accuracy but not the computation speed.

This study suggest the future work to focus on improving ESET-ConvNet in the context of reducing the computation time. This study has attempted to learn a larger input sparsely compared to previous work, however, it is significantly less efficient to perform a forward pass. In practise, the trade-off between the size of the input and the time needs to be further tuned to meet the standard of high-frequency trading.

Nomenclatures

a	Small slope of negative input of Leaky-ReLU
b	Batch size
c	Number of output channel
c_in	Number of input channel
d	Dilation Rate
g	Magnitude of weights
k	Kernel size or Prediction horizon
l	Length of sequence
p	Probability of keeping neuron in dropout
r	Reduction Ratio
v	Direction of weights
x	Input of model
$\hat{y}$	Predicted output of model

Greek Symbols

β_1	Adam's decay rate of running average of gradients.
β_2	Adam's decay rate of running average of square of gradients.
ε	Tiny constant to prevent division by zero.

Abbreviations

BoF	Bag-of-Features
CNN	Convolutional Neural Network
DAIN	Deep Adaptive Input Normalization
DL	Deep Learning
ESET-ConvNet	Embedded Squeeze-and-Excitation Temporal
LDA	Latent Dirichlet Allocation
LOB	Limit Order Book
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
SE-block	Squeeze-and-Excitation block
SENet	Squeeze-and-Excitation Network
SVM	Support Vector Machine
TABL	Temporal Attention augmented Bilinear Layer
TCN	Temporal Convolutional Network
WN	Weight Normalization

References

1. Snow, D. (2020). Machine learning in asset management - Part 1: Portfolio construction - Trading strategies. *The Journal of Financial Data Science*, 2(1), 10-23.
2. Ljung, C.; and Maria, S. (2018). A study of momentum effects on the Swedish stock market using Time Series Regression. Retrieved October 5, 2020, from http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-228996.
3. Chiang, W.C.; Enke, D.; Wu, T.; and Wang, R. (2016). An adaptive stock index trading decision support system. *Expert Systems with Applications*, 59, 195-207.
4. Dey, S.; Kumar, Y.; Saha, S.; and Basak S. (2016). Forecasting to classification: Predicting the direction of stock market price using xtreme gradient boosting. Retrieved November 5, 2020, from https://doi.org/10.13140/RG.2.2.15294.48968.
5. Shen, K.Y.; and Tzeng, G.H. (2015). Combined soft computing model for value stock selection based on fundamental analysis. *Applied Soft Computing*, 37, 142-155.
6. Lin, Q. (2018). Technical analysis and stock return predictability: An aligned approach. *Journal of financial markets*, 38, 103-123.
7. Liu, J.; and Kemp, A. (2019). Forecasting the sign of us oil and gas industry stock index excess returns employing macroeconomic variables. *Energy Economics*, 81, 672-686.
8. Derakhshan, A.; and Beigy, H. (2019). Sentiment analysis on stock social media for stock price movement prediction. *Engineering Applications of Artificial Intelligence*, 85, 569-578.
9. Sirignano, J.A. (2019). Deep learning for limit order books. *Quantitative Finance*, 19.4, 549-570.
10. Ntakaris, A.; Magris, M.; Kannianen, J.; Gabbouj, M.; and Iosifidis, A. (2018). Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods. *Journal of Forecasting*, 37.8, 852-866.
11. Bai, S.; Kolter, J.Z.; and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:*1803.01271.
12. Hu, J.; Shen, L.; and Sun, G. (2018). Squeeze-and-excitation networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*. Salt Lake City, USA, 7132-7141.
13. Krauss, C.; Do, X.A.; and Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*, 259.2, 689-702.
14. Patel, J.; Shah, S.; Thakkar, P.; and Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert systems with applications*, 42.1, 259-268.
15. Fischer, T.; and Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270.2, 654-669.

16. Zhang, Z.; Zohren, S.; & Roberts, S. (2019). Deeplob: Deep convolutional neural networks for limit order books. *IEEE Transactions on Signal Processing*, 67(11), 3001-3012.
17. Shynkevich, Y.; McGinnity, T.M.; and Coleman, S.A. (2017). Forecasting price movements using technical indicators: Investigating the impact of varying input window length. *Neurocomputing*, 264, 71-88.
18. Dash, R.; and Dash, P.K. (2016). A hybrid stock trading framework integrating technical analysis with machine learning techniques. *The Journal of Finance and Data Science*, 2.1, 42-57.
19. Henrique, B.M.; Sobreiro, V.A.; and Kimura, H. (2018). Stock price prediction using support vector regression on daily and up to the minute prices. *The Journal of finance and data science*, 4.3, 183-201.
20. Zhou, F.; Zhang, Q.; Sornette, D.; and Jiang, L. (2019). Cascading logistic regression onto gradient boosted decision trees for forecasting and trading stock indices. *Applied Soft Computing*, 84, 105747.
21. Naik, N.; and Mohan, B.R. (2019). Stock price movements classification using machine and deep learning techniques-the case study of Indian stock market. *International Conference on Engineering Applications of Neural Networks*. Xersonisos, Greece, 445-452.
22. Passalis, N.; Tefas, A.; Kanniainen, J.; Gabbouj, M.; and Iosifidis, A. (2018). Temporal bag-of-features learning for predicting mid price movements using high frequency limit order book data. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4.6, 774-785.
23. Passalis, N.; Tefas, A.; Kanniainen, J.; Gabbouj, M.; and Iosifidis, A. (2019). Deep adaptive input normalization for time series forecasting. *IEEE transactions on neural networks and learning systems*, 31.9, 3760-3765.
24. Tsantekidis, A.; Passalis, N.; Tefas, A.; Kanniainen, J.; Gabbouj, M.; and Iosifidis, A. (2020). Using deep learning for price prediction by exploiting stationary limit order book features. *Applied Soft Computing*, 93, 106401.
25. Tran, D.T.; Iosifidis, A.; Kanniainen, J.; and Gabbouj, M.(2018). Temporal attention-augmented bilinear network for financial time-series data analysis. *IEEE transactions on neural networks and learning systems*, 30.5, 1407-1418.
26. Ribeiro, M.T.; Singh, S.; and Guestrin, C. (2016). " Why should i trust you?" Explaining the predictions of any classifier. *Proceedings of the* 22*nd ACM SIGKDD international conference on knowledge discovery and data mining*. Virtual Event, Singapore, 1135-1144.
27. Xu, B.; Wang, N.; Chen, T.; and Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv:*1505.00853.
28. Salimans, T.; and Kingma, D.P. (2016). Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *arXiv:*1602.07868.
29. Maas, A.L.; Hannun, A.Y.; and Ng, A.Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. icml*. Atlanta, USA, 30.1, 3.
30. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15.1, 1929-1958.

31. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; and Desmaison, A. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 8026-8037.
32. Kingma, D.P.; and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:*1412.6980.
33. Loshchilov, I.; and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv:*1711.05101.
34. Tsantekidis, A.; Passalis, N.; Tefas, A.; Kanniainen, J.; Gabbouj, M.; and Iosifidis, A. (2017). Forecasting stock prices from the limit order book using convolutional neural networks. 2017 *IEEE* 19*th Conference on Business Informatics* (*CBI*). Thessaloniki, Greece, 1, 7-12.
35. Tsantekidis, A.; Passalis, N.; Tefas, A.; Kanniainen, J.; Gabbouj, M.; and Iosifidis, A. (2017). Using deep learning to detect price change indications in financial markets. 2017 25*th European Signal Processing Conference* (*EUSIPCO*). Kos Island, Greece, 2511-2515.
36. Ang, J.S.; Ng, K.W., and Chua, F.F. (2020). Modeling Time Series Data with Deep Learning: A Review, Analysis, Evaluation and Future Trend. 2020 8*th International Conference on Information Technology and Multimedia* (*ICIMU*). Langkawi, Malaysia, 32-37.