The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

Visual Attention Model for Cross-Sectional Stock Return Prediction and End-to-End Multimodal Market Representation Learning

Ran Zhao
Carnegie Mellon University
rzhao1@cs.cmu.edu

Yuntian Deng
Harvard University
dengyuntian@seas.harvard.edu

Mark Dredze
Johns Hopkins University
mdredze@cs.jhu.edu

Arun Verma
Bloomberg
averma3@bloomberg.net

David Rosenberg
Bloomberg
drosenberg44@bloomberg.net

Amanda Stent
Bloomberg
astent@bloomberg.net

Abstract

Technical and fundamental analysis are traditional tools used to analyze stocks; however, the finance literature has shown that the price movement of each individual stock is highly correlated with that of other stocks, especially those within the same sector. In this paper we propose a general-purpose market representation that incorporates fundamental and technical indicators and relationships between individual stocks. We treat the daily stock market as a 'market image' where rows (grouped by market sector) represent individual stocks and columns represent indicators. We apply a convolutional neural network over this market image to build market features in a hierarchical way. We use a recurrent neural network, with an attention mechanism over the market feature maps, to model temporal dynamics in the market. Our model outperforms strong baselines in both short-term and long-term stock return prediction tasks. We also show another use for our market image: to construct concise and dense market embeddings suitable for downstream prediction tasks.

Introduction

In recent years there have been multiple proposals for methods to adopt machine learning techniques in quantitative finance research. Modeling stock price movement is very challenging since stock prices are affected by many external factors such as political events, market liquidity and economic strength. However, the rapidly growing volume of market data allows researchers to upgrade trading algorithms from simple factor-based linear regression to complex machine learning models such as reinforcement learning (Lee 2001), k-nearest neighbors (Alkhatib et al. 2013), Gaussian processes (Mojaddady, Nabi, and Khadivi 2011) and many deep learning approaches, e.g. (Kwon, Choi, and Moon 2005; Rather, Agarwal, and Sastry 2015; Singh and Srivastava 2017).

A variety of financial theories for market pricing have been proposed, which can serve as the theoretical foundation for designing tailored machine learning models. First, the efficient market hypothesis (Malkiel and Fama 1970) states that all available information is reflected in market prices. Fluctuations in stock prices are a result of newly released information. Therefore, through analyzing individual stock price movements, a machine learning-based model should be able to decode the embedded market information.

Second, value investing theory (Graham and Dodd 2002) suggests to buy stocks below their intrinsic value to limit downside risk. The intrinsic value of a company is calculated by fundamental indicators which are revealed in quarterly and annual financial reports. A machine learning-based model should therefore be capable of discovering the relationships between different types of fundamental indicator and the intrinsic value of a company.

Third, the methodology of technical analysis introduced in (Murphy 1999) includes well-known context-dependent leading indicators of price movement such as relative strength index (RSI) and moving average convergence/divergence (MACD). A machine learning-based model should be able to estimate the predictive power of traditional technical indicators in different market situations.

Fourth, the stock market has a well-defined structure. In the macro, people have invented different financial indexes for major markets such as the NASDAQ-100 and Dow Jones Industrial; these are composite variables that may indicate market dynamics. In the micro, the stock market is usually divided into 10 major sectors and tens of subsectors for key areas of the economy. Stocks in the same sector have a shared line of business and are expected to perform similarly in the long run (Murphy 2011). Traditional ways of dealing with market information are to include hand-crafted microeconomic indicators in predictive models, or to construct covariance matrixes of returns among groups of stocks. However, those hand-crafted features can become gradually lagged and unable to dynamically adjust to market changes. Therefore, a machine learning-based model should leverage information from the whole market as well as the sector of each included company.

Inspired by these financial theories, we implement an end-to-end market-aware system that is capable of capturing market dynamics from multimodal information (fundamental indicators (Graham and Dodd 2002), technical indicators (Murphy 1999), and market structure) for stock return

Copyright © 2019, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Indicator Set	Time Scale	Indicators
Price-Volume	Daily	Close-to-Open ratio, High-to-Open ratio, Low-to-Open ratio, Close-to-High ratio, Close-to-Low ratio, High-to-Low ratio
Historical Return	Daily	last {1,2,3,4,5}-day return, last {5,10,15,20,25,30}-day cumulative return
Technical Indicators	Daily	BOLL,DMI,RSI, MACD,ROC,MOMENTUM
Fundamental Indicators	Quartly	EPS,CUR_RATIO, TOT_DEBT_TO_TOT_EQY, FNCL_LVGR, RETURN_TOT_EQY, PE_RATIO, SHORT_INT_RATIO

Table 1: Indicators used in our 'market image'

prediction[1]. First, we construct a 'market image' as in Figure 1, in which each row represents one stock and each column represents an indicator from the three major categories shown in Table 1. Stocks are grouped in a fixed order by their sector and subsector (industry). Then we apply state-of-the-art deep learning models from computer vision and natural language processing on top of the market image. Specifically, our contributions in this work are to: (1) leverage the power of attention-based convolutional neural networks to model spatial relationships between stocks in the market dimension, and of recurrent neural networks for time series forecasting of stock returns in the temporal dimension, and (2) use a convolutional encoder-decoder architecture to reconstruct the market image for learning a generic and compact market representation.

In the following sections, we present our market image, then our models for market-aware stock prediction, and finally our method for computing generic and compact market representations. We present empirical results showing that our model for market-aware stock prediction beats strong baselines and that our market representation beats PCA.

The Market Image

We represent the daily market as an image M, a $m \times n$ matrix where m is the number of unique stocks and n is the number of extracted traditional trading indicators. In our experiments, we used the indicators from Table 1. A sample market image is depicted in Figure 1. The market image serves as a snapshot of market dynamics. These market images can be stacked to form a market cube as shown in Figure 2, thus incorporating a temporal dimension into the representation.

For our experiments later in this paper, we collected the 40 indicators from Table 1 for each of the S&P 500 index constituents on a daily basis from January 1999 to Dec 2016, and used this to construct daily market images. The size of

[1]Stock return is appreciation in price (plus any dividends) divided by the original price of the stock.

(sector)	Ticker	CUR_RATIO	MACD	PE_RATIO	...	RSI
Technology	AAPL	1.0801	2.9012	17.9642	...	61.7531
	GOOGL	4.4707	-4.9249	26.9981	...	35.8913
	MSFT	2.5185	0.3666	18.1137	...	52.9148
...	....	...	...	...	...	...
Basic Materials	XOM	0.888	-0.2489	11.9434	...	51.3308

Figure 1: 1-day market image snapshot

the daily image is 500 (stocks) X 40 (multimodal indicators), denoted as $M_d = \{m_{i=1,...500}, n_{j=1,...40} \in \mathbb{R}\}_d$. In each market image, stocks are grouped first by the ten sectors in the Global Industry Classification Standard (GICS), and within each sector, by the GICS subsectors. We normalize the values for each indicator into a 0-1 scale by applying a min-max scalar using the min and max values of that indicator in the training data (see equation (1)).

$$\{M_{i,j'}\}_d = \frac{\{M_{i,j}\}_d - min(\{M_j\}_{\{d\}})}{(max(\{M_j\}_{\{d\}}) - min(\{M_i\}_{\{d\}})} \quad (1)$$

Some fundamental indicators are updated quarterly; to fill these blank cells in our market images, we applied a backward fill policy to the affected columns.

Market-Aware Stock Return Prediction

Let us assume that we want to predict the return of $stock_m$ at day d based on information from the previous t days. This means that we have to learn a market representation with respect to $stock_m$ given the previous t market images as the market context. First we describe our Market Attention model (MA; right side of Figure 2), which builds market-aware representations for individual stocks. Second, we describe how we add temporal modeling to this model to get our Market-Aware Recurrent Neural Network model (MA-RNN; left side of Figure 2). Third, we present empirical results demonstrating that these models outperform strong baselines for stock return prediction.

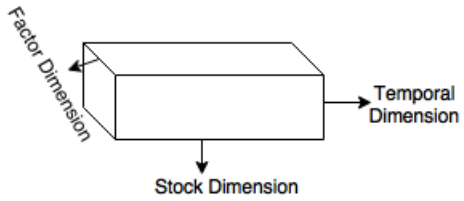

Figure 2: Market Cube

Market Attention Model

We rotate and stack t market images to construct a 3-D market cube $E \in \mathbb{R}^{t \times m \times n}$. Rows ($t$) index the temporal dimension, columns (m) index stocks, and channels (n) index indicators, as shown in Figure 2. Let $x_t^n \in \mathbb{R}^m$ refer to the m-dimensional vector indexed by t in the temporal dimension and n in the factor dimension of the market cube E and

$y_t^m \in \mathbb{R}^n$ refer to the n-dimensional vector indexed by t in the temporal dimension and m in the stock dimension.

Separately, we initialize stock embeddings $S = \{s^1, s^2, ...s^m\}$ to non-zero vectors, where $s^m \in \mathbb{R}^{|v|\times 1}$ indexes the m-th column's stock embedding.

Then, we use a convolutional neural network (CNN) to generate multiple feature maps of the market cube through multiple convolution operations (right side of Figure 3). Each convolution operation involves a filter $w \in \mathbb{R}^{1\times m}$, which is applied to a window of one day to produce a new feature c^j by:

$$c^j = f(\sum_t^T \sum_{n=1}^N w_j^n \cdot x_t^n + b), b \in \mathbb{R} \quad (2)$$

j denotes the j-th kernel; in our experiments, we use 192 different kernels. f is a ReLU active function for introducing non-linearities into the model. So we have a 1-D convolution filter that slides its window vertically with stride=1 along the first dimension of the market cube to produce a feature map column vector $c^j =< c_1^j, c_2^j, ..., c_t^j >^\intercal$.

Given a target stock embedding s^m, the attention model will return a weighted arithmetic mean of the $\{c^j\}$, where the weights are chosen according the relevance of each c^j to the stock embedding s^m. We use the additive attention mechanism explained in (Bahdanau, Cho, and Bengio 2014). In equation 3, w_{sz} and $w_{\bar{c}z}$ are learned attention parameters.

$$z_j = \tanh(w_{sz} \cdot s^m + w_{cz} \cdot c^j) \quad (3)$$

We compute attention weights using a softmax function

$$a_j = \frac{\exp(v_j \cdot z_j)}{\sum_i \exp(v_i \cdot z_i)} \quad (4)$$

The conditioned market embedding p^m is calculated by

$$p^m = \sum_j a_j c^j \quad (5)$$

Intuitively, each filter serves to summarize correlations between different stocks across multiple indicators. Each kernel is in charge of finding different types of patterns among the raw indicators. The attention mechanism of the CNN is responsible for selecting the patterns on which to focus for a particular target stock. The conditioned market embedding summarizes the information contained in the market cube E that is pertinent to the target stock.

Market-Aware RNN

In parallel, we deploy a long-short-term memory recurrent neural network (LSTM) to model the temporal dependencies in a sequence of multidimensional features $y_i^m$of a specific $stock_m$. Recurrent neural networks (Hochreiter and Schmidhuber 1997; Mikolov et al. 2010) are widely applied in natural language processing applications to capture long-range dependencies in time series data (Sutskever, Vinyals, and Le 2014; Cho et al. 2014; Dyer et al. 2015; Yang et al. 2017). The attention mechanism (Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015) has become an indispensable component in time series modeling with recurrent neural networks; it provides an evolving view of the input sequence as the output is being generated.

The sequence of multidimensional features for $stock_m$ $(y_1^m, y_2^m, ...y_t^m)$ are sequentially encoded using a LSTM cell of size 25. The mechanism of the LSTM is defined as:

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ j_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{bmatrix} W[h_{t-1}, x_t]$$

$$c_t = f_t \odot c_{t-1} + i_t \odot j_t$$

$$h_t = o_t \odot tanh(c_t)$$

We treat the last hidden LSTM output, q^m, as the representation of the target $stock_m$'s performance in the past t days.

Finally, we feed both our learned dense market performance embedding p^m and stock performance embedding q^m to a feedforward neural network. They are non-linearly transformed separately in the first layer ϕ and concatenated together to predict the target stock return:

$$f(x) = g(\sum_i W[\phi_1(p^m), \phi_2(q^m)] + b) \quad (6)$$

Evaluation

We conducted an evaluation of our Market-Attention RNN model (MA-RNN). For labels, we built stock return matrices for each market image. We used 1-day and 5-day returns for short-term predictions and 15-day and 30-day returns for long-term predictions, denoted as $R_d = \{r_{i=1,...500,j=1,...10}\}_d$. In order to reduce the effect of volatility on returns, we divide the individual daily return by its recent past standard deviation of return (*cf* the Sharpe ratio). The moving window size to calculate the standard deviation is 10, see equation (7).

$$\{r_{i,j'}\}_d = \frac{r_{i,j}}{\sigma(r_{j,\{d-10:d-1\}})} \quad (7)$$

We divided our input market images into training, validation and backtest sets by time, as shown in Table 2.

	Training	Validation	Backtest
Period	1999-2012	2012-2015	2015-2016
#Trading Days	3265	754	504

Table 2: Data Split

We trained our MA-RNN model with the following hyperparameters: a convolution stride size of 1; a dimensionality of 100 for the trainable stock embeddings; a dimensionality of 32 for the attention vector of the convolutional neural work; a dimensionality of 40 for the final market representation vector; a cell size of 32 for the LSTM; hidden layers of dimensionality 100 and 50 respectively for our fully connected layers; ReLu non-linearity; and a time window t of 10. All the initialized weights were sampled from a uniform distribution [-0.1,0.1]. The mini-batch size was 10. The

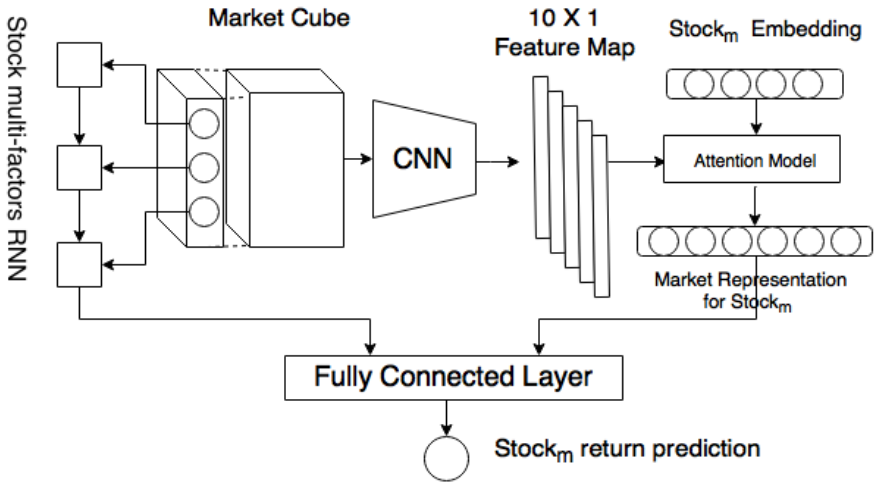

Figure 3: Architecture of Market-Attention Recurrent Neural Network Model (MA-RNN)

models were trained end-to-end using the Adam optimizer (Kingma and Ba 2014) with a learning rate of 0.001 and gradient clipping at 5.

For benchmarking our MA-RNN model, we chose several standard machine learning models. We report MSE of the % return prediction as our metric.

We conducted two experiments. First, we compared the performance of models *with* and *without* market information. Linear regression (LR), a feedforward neural network[2] (FFNN), a long-short term memory recurrent neural network (LSTM-RNN) that uses only individual stocks' price histories[3] and support vector regression[4] (SVR) (Drucker et al. 1997) serve as our market info-free comparison models. Our Market-Attention model (MA) relies solely on the learned market representation, p_m (with reference to Figure 3, it uses only the CNN with attention, and ignores the output of the LSTM). We found that **market awareness can be successfully modeled to improve stock return prediction**. As shown in Table 3, at every time interval (n = 1 day, 5 days, 15 days and 30 days) the Market-Attention (MA) model has lower MSE than the other models, which have no information about the market as a whole[5].

Model	n=1	n=5	n=15	n=30
LR	3.711	6.750	12.381	18.429
SVR	2.411	4.917	8.149	11.930
FFNN	1.727	3.952	6.967	9.088
LSTM-RNN	1.426	2.896	5.854	7.923
MA	0.91	1.63	4.383	5.114

Table 3: Mean Squared Error of % Return Prediction

[2]We used two hidden layers of size 50 and sigmoid non-linearity.

[3]We used a LSTM cell size of 25.

[4]We used a linear kernel function with penalty parameter c=0.3.

[5]Obviously, the further away from the current day, the higher the error is expected to be.

Second, we compared the MA model with the full MA-RNN model to show the value of explicitly modeling temporal dependencies. We found that **temporal awareness can be successfully used in a market-aware model for improved stock return prediction**. As shown in Table 4, our MA-RNN model has lower MSE than our baseline MA model.

Model	n=1	n=5	n=15	n=30
MA	0.91	1.63	4.383	5.114
MA-RNN	0.790	1.210	3.732	4.523

Table 4: Mean Squared Error of % Return Prediction

Generic Market Representation: MarketSegNet

Based on our finding from the previous section that market awareness leads to improved stock prediction accuracy, we propose a novel method to learn a generic market representation (MarketSegNet) in a end-to-end manner. The market representation learning problem is to convert market images (potentially of variable dimensions) to fixed-size dense embeddings for general purpose use. As a test of the fidelity of this representation, from the generic market embedding it should be possible to reconstruct the input market image pixel wise.

Inspired by (Badrinarayanan, Kendall, and Cipolla 2017), we developed a deep fully convolutional autoencoder architecture for pixel-wise regression (Figure 4). The convolutional encoder-decoder model was originally proposed for scene understanding applications, such as semantic segmentation (Long, Shelhamer, and Darrell 2015; Badrinarayanan, Kendall, and Cipolla 2017) and object detection (Ren et al. 2015). A convolutional encoder builds feature representations in a hierarchical way, and is able to take in images of arbitrary sizes, while a convolutional decoder is able to pro-

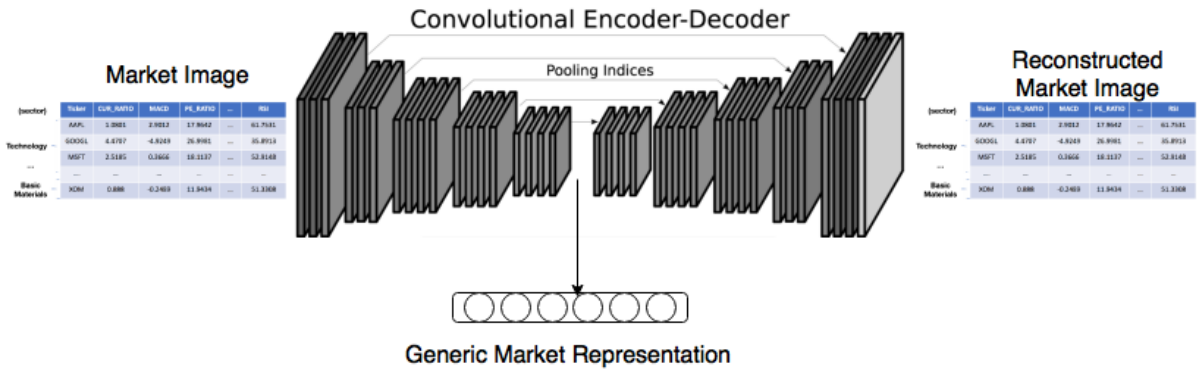

Figure 4: Architecture of MarketSegNet

duce an image of a corresponding size. By using convolutional neural networks, the extracted features exhibit strong robustness to local transformations such as affine transformations and even truncations (Zheng, Yang, and Tian 2017). In a stock market modeling application, after representing each day's overall stock market as an image, we believe that (1) building features in a hierarchical way can provide a better summary of the market, since stocks exhibit an inherent hierarchical structure, and (2) robustness to local transformations is desirable, since the stock universe is constantly changing, with new companies being added, and other companies removed, while we do not want the overall market representation to be greatly affected by the addition or removal of a single company.

Since our market image has a different spatial configuration from a normal image, we customize the structure of our end-to-end architecture. The encoder network is composed of traditional convolutional and pooling layers which are used to reduce the resolution of the market image through max-pooling and subsampling operations. Meanwhile, the encoder network stores the max-pooling indices used in the pooling layer, to be applied in the upsampling operation in the corresponding decoder network. The decoder network upsamples the encoder output using the transferred pool indices to produce sparse feature maps, and uses convolutional layers with a trainable filter bank to densify the feature map so as to recover the original market image. Since companies are grouped in the market image by sector, max-pooling in the encoder network can capture the trends of stocks in the same sector.

To evaluate MarketSegNet, we compare its ability to reconstruct input market images with that of a well-known algorithm for dimensionality reduction, Principal Component Analysis (PCA). PCA uses singular value decomposition to identify variables in the input data that account for the largest amount of variance. We used our training data to train our MarketSegNet model and to fit a PCA model. We then used the MarketSegNet and PCA models to compress and then reconstruct the market images in our test data. We compared the reconstruction error rates of PCA and our MarketSegNet model. Since we varied the sizes of our learned market embeddings from 16 to 128, for each size we created a

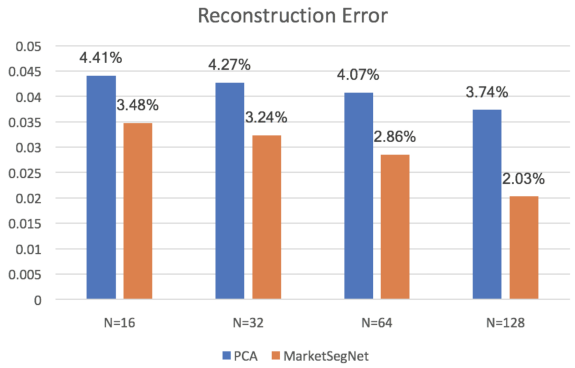

Figure 5: Market Image Reconstruction Error Rates

PCA model with that number of principal components.

Our results are shown in Figure 5. For every size of market embedding, MarketSegNet has lower reconstruction error than PCA.

Conclusions and Future Work

In this paper, we present a method for constructing a 'market image' for each day in the stock market. We then describe two applications of this market image:

1. As input to ML-based models for stock return prediction. We demonstrate (a) that market awareness leads to reduced error vs non-market-aware methods, and (b) that temporal awareness across stacks of market images leads to further reductions in error.
2. As input to a ML-based method for constructing generic market embeddings. We show that the learned market embeddings are better able to reconstruct the input market image than PCA across a range of dimensionality reductions, indicating that they capture more information about the input market image.

We should emphasize that our MA model, our MA-RNN model and our MarketSegNet market embeddings do not represent trading strategies. They are agnostic to trading costs, lost opportunity cost while out of the market, and

other factors that matter with an active trading strategy. That said, they may provide information that is useful for other AI-driven financial prediction tasks. Other research groups that have used the models described here have reported improved performance in predicting the directionality of stock price moves on earnings day, and in assessing which events will move markets. We leave further exploration of the applications of these models to future work.

References

Alkhatib, K.; Najadat, H.; Hmeidi, I.; and Shatnawi, M. K. A. 2013. Stock price prediction using k-nearest neighbor (kNN) algorithm. *International Journal of Business, Humanities and Technology* 3(3):32–44.

Badrinarayanan, V.; Kendall, A.; and Cipolla, R. 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(12):2481–2495.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Drucker, H.; Burges, C. J.; Kaufman, L.; Smola, A. J.; and Vapnik, V. 1997. Support vector regression machines. In *Advances in Neural Information Processing Systems*, 155–161.

Dyer, C.; Ballesteros, M.; Ling, W.; Matthews, A.; and Smith, N. A. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

Graham, B., and Dodd, D. 2002. *Security Analysis*. McGraw Hill Professional.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kwon, Y.-K.; Choi, S.-S.; and Moon, B.-R. 2005. Stock prediction based on financial correlation. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, 2061–2066. ACM.

Lee, J. W. 2001. Stock price prediction using reinforcement learning. In *Proceedings of the IEEE International Symposium on Industrial Electronics*, volume 1, 690–695.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Malkiel, B. G., and Fama, E. F. 1970. Efficient capital markets: A review of theory and empirical work. *The Journal of Finance* 25(2):383–417.

Mikolov, T.; Karafiát, M.; Burget, L.; Cernockỳ, J.; and Khudanpur, S. 2010. Recurrent neural network based language model. In *Proceedings of the Annual Conference of the International Speech Communication Association*.

Mojaddady, M.; Nabi, M.; and Khadivi, S. 2011. Stock market prediction using twin Gaussian process regression. Technical report, Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran.

Murphy, J. 1999. *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance Series. New York Institute of Finance.

Murphy, J. 2011. *Intermarket analysis: profiting from global market relationships*, volume 115. John Wiley & Sons.

Rather, A. M.; Agarwal, A.; and Sastry, V. 2015. Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Systems with Applications* 42(6):3234–3241.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 91–99.

Singh, R., and Srivastava, S. 2017. Stock prediction using deep learning. *Multimedia Tools and Applications* 76(18):18569–18584.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 3104–3112.

Yang, Z.; Hu, Z.; Deng, Y.; Dyer, C.; and Smola, A. 2017. Neural machine translation with recurrent attention modeling. In *Proceedings of the Meeting of the European Chapter of the Association for Computational Linguistics*, 383.

Zheng, L.; Yang, Y.; and Tian, Q. 2017. SIFT meets CNN: A decade survey of instance retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(5):1224–1244.