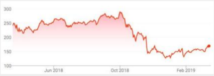
Winton Stock Market Challenge: Using a Deep Learning Framework for Time Series Stock Market Prediction

Joaquin Borggio, Sagar Maheshwari, Cheyenne Sadeghi, Maxim Serebriakov

Motivation

- It is commonly thought that the stock market is unpredictable due to the presence of random factors



- Growing prevalence of High Frequency Trading (HFT) in markets reliant on AI and algorithms
- With the use of a deep learning framework, we attempt to predict trends within the stock market with a higher accuracy than random guessing

Data

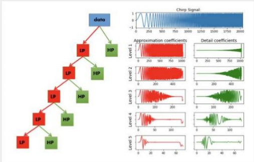
- Dataset provided by Winton: Global Investment Management

WINTON

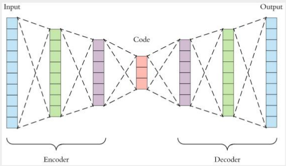
- 40,000 training/test stocks
- Each stock includes the following features:
  - 25 anonymized features
  - Close price from two days ago
  - Close price from one day ago
  - Stock prices for first 120 time-steps of current day
- Predict:
  - Stock prices for next 60 time-steps of current day
  - Close price for the day after
  - Close price for two days after

Network Architectures

Wavelet Transform: Captures information about frequency distribution and location in time.

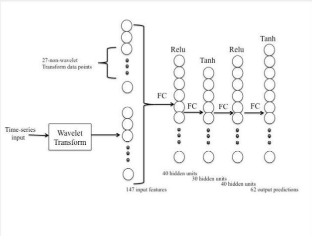


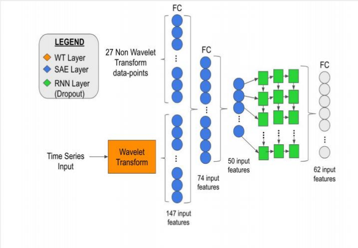
Stacked Autoencoders: A form of unsupervised learning where encoder and decoder functions are learned.



<u>Fully Connected Network</u>

1) Wavelet Transform - Using the Discrete Wavelet Transform (DWT) allows us to denoise data and separate the time series data into frequency components
2) Fully Connected Layers - After the DWT, we implemented fully connected layers with 40, 30, 40, and 62 units, respectively





<u>Recurrent Neural Network</u>

1) Wavelet Transform - Same implementation as fully connected network
2) Stacked Autoencoder (SAE) - After training the SAE, we use the encoding functions to process our DWT data.
3) Recurrent Neural Network (RNN) - A 3 layer RNN with dropout implemented to prevent overfitting.

Results

<u>Hyperparameter Tuning</u>

After searching for the best number of epochs to train on for our neural network, we arrived at the optimal value of 25 epochs.

Total Epochs (Dropout = 0.1)	Training Set	Test Set
1	51.9913%	51.8105%
5	51.4072%	51.4071%
10	57.1145%	57.1596%
25	65.4812%	65.4697%
30	64.0671%	64.1763%
50	64.1439%	64.0503%
100	60.8273%	60.7885%

Dropout	Train Set	Test Set
0.08	63.6774%	63.6786%
0.1	65.6912%	65.2534%
0.15	64.3821%	64.3584%
0.2	64.8197%	64.8297%
0.25	63.8641%	63.8105%

Once we established our epoch number of 25, we also trained on various dropout probabilities, with probability 0.1 yielding the best results.

<u>Comparing Two Models</u>

Fully Connected Network:

Data Input	1	2	3	4	5	6	7	8	9	10	Average
Square Difference	51.131	56.888	49.907	50.105	50.450	49.294	47.365	51.358	47.102	47.937	49.950
Cross Entropy Loss	47.373	49.490	49.996	53.623	55.485	53.186	53.385	54.189	48.338	51.735	51.680

Recurrent Neural Network:

	1	2	3	4	5	6	7	8	9	10	Average
Train	64.8	65.09	64.87	65.41	65.22	64.96	65.24	65.13	65.4	63.75	64.99
Test	64.79	65.15	64.91	65.45	65.06	64.96	65.18	65.02	65.54	63.63	64.97

Future Work

- Long Short-Term Memory (LSTM) : Long term retention of stock price information may be needed to predict future prices. Due to such long-term dependencies, the use of LSTMs could significantly improve model performance.
- Feature Selection: Use Principal Component Analysis (PCA) or other techniques in order to determine the most relevant features when training and testing
- Data Heterogeneity: Scrape data for eclectic group of stocks to ensure the distributions of the training and testing data are not highly similar