A Wavelet Decomposition based Ensemble Learning Framework for Short-Term Stock Prediction

Hieu Nguyen, **Jihye Moon**, **Dongjin Song** and **Joseph Johnson**
University of Connecticut
{hieu.nguyen, jihye.moon, dongjin.song, joseph.2.johnson}@uconn.edu

Abstract

This study proposes an ensemble framework for the short-term forecasting of price movements in the stock market. Our framework introduces a signal decomposition method, the wavelet transform, to machine learning models. We decompose feature signals into high and low-frequency portions, in which the low frequencies approximate the original signals, and the high frequencies can be interpreted as details occurring on smaller time scales. We theorize that the information influencing short-term stock movements may largely be contained in these high-frequency portions. Thus these high-frequency portions may be a valuable data source for training various machine learning methods to be used in forecasting stock movements. In this paper, we apply this concept to build an ensemble machine learning framework on data spanning 10 years (01/01/2010-01/01/2020) and consisting of three major U.S. ETFs: S&P 500 (SPY), Dow Jones Industrial Average (DIA), and NASDAQ (QQQ). The results obtained from this study indicate that the proposed feature decomposition approach outperforms models using unaltered signals, with an accuracy of up to 72 percent.

1 Introduction

Time series analysis has been an appealing domain in many different fields such as economics, social sciences, medicine, physical sciences, and finances. In recent years, forecasting the movement of stocks has become an ever-increasing issue in investment/trading decision-making. Researchers have been building models to analyze and understand historical data to potentially predict future outcomes. With the increased prevalence of commission-free trading platforms, stock forecasting has been garnering increased attention not only within financial institutions but amongst retail investors as well.

One well-known time series analysis method is the autoregressive moving average (ARIMA) model [Fattah *et al.*, 2018]. Though ARIMA has been a successful model in various time series applications, the model does not perform well on high-dimensional data. Other variants of the ARIMA model were developed to tackle this problem such as the nonlinear autoregressive exogenous(NARX) models [Boussaada *et al.*, 2018].

Due to the stock market being nonlinear, non-stationary, and exhibiting many external factors, predicting stock movements has traditionally been a challenging task. Recently, machine learning models have been increasing in popularity due to their ability to solve complex and nonlinear problems, as seen in computer vision and natural language processing applications. [Schmidhuber, 2015; Krizhevsky *et al.*, 2012; Sak *et al.*, 2014]. As technology continues to grow, we witness an increased usage of machine learning methods in financial models.

Over the past 40 years, much of the literature pertaining to time series forecasting uses traditional machine learning methods such as decision trees, random forests, and support vector machines. With the recent developments of deep learning, many studies have shown that deep learning models outperform their machine learning counterparts [Wu, 2020; Chong *et al.*, 2017; Chen *et al.*, 2015; Krauss *et al.*, 2017; Sezer *et al.*, 2020]. The long short-term memory (LSTM) model is a popular method among many natural language applications which has been extended to applications in time series due to its forecasting nature [Rahimyar *et al.*, 2019; Oord *et al.*, 2016; Lai *et al.*, 2018; Oreshkin *et al.*, 2019]. The advantage of machine learning methods is their ability to learn the non-linear mapping between features. The model outperforms traditional methods when inputs are high-dimensional data. However, since many applications of time-series forecasting come with limited data, the deep learning model is prone to overfitting. Thus, researchers propose hybrid methods to combine the best of both worlds, where traditional methods would learn the linear relationship and deep learning methods learn the nonlinearity of the residuals [Pai and Lin, 2005].

Existing researches use the wavelet transform and other signal processing methods to denoise the data, which can result in an incremental improvement in forecasting using low-frequency portions as predictors [Nguyen and Wang, 2016; Rahimyar *et al.*, 2019]. In [Wang *et al.*, 2018], the authors proposed a multilevel Wavelet Decomposition Network (mWDN) that uses multilevel 2-band discrete wavelet decomposition with a deep neural network framework and tested on 40 UCR datasets. Such wavelet decompositions are not uncommon in quantitative finance. A study [Zhao *et al.*, 2018] uses a continuous wavelet transform to decompose time series input into scalograms and then uses CNN and Attention LSTM as models. However, much of this focus is often centered around the low-frequency portions of the data to forecast/analyze stock price, and naively assume a negligible contribution from high-frequency portions. We take another approach and theorize that these high-frequency signals contain information that has more influence on short-term stock movements than traditionally assumed. To tackle this challenge, we use the wavelet transform as a signal processing method to decompose the feature signals into high and low-frequency portions and build an ensemble machine learning pipeline to forecast short-term stock movements.

To the best of our knowledge, we are the first to apply a single level 4-band wavelet decomposition for a multi-featured forecasting model. Our proposed framework yields a significant improvement for short-term stock forecasting by only using high-frequency portions of the data. The summary of our contribution is as follows:

- Propose a relationship between the high-frequency signals (obtained from the wavelet transformed features) and the short-term stock fluctuations.
- A unified framework combining the resulting high-frequency signals and machine learning techniques to forecast short-term stock movement. Our results present a significant improvement for short-term prediction when using high-frequency signals as inputs.

The paper is organized as follows: Section 2 describes the proposed method, Section 3 presents results, and followed by a conclusion.

2 Proposed Method

Financial time series data is often impacted by many external factors, and consequently, an accurate forecast is a very challenging task. In this study, we use a signal processing method called wavelet transformation to decompose the time series data into low and high-frequency portions. The low-frequency signal represents an approximation of the original data, and the high-frequency signals represent data occurring on shorter time scales.

We propose a framework that focuses on decomposing the features into low and high-frequency signals using the discrete wavelet transform (DWT). Each frequency portion is then used in conjunction with machine learning (ML) models. The results are then compared to help determine the best forecasting model (see Figure 1).

2.1 Features Decomposition

One of the most common signal processing methods is the Fourier Transform, which only considers the frequency domain and omits any information pertaining to time. However, this omission is undesirable due to the time-dependent nature of financial data.

A solution to this challenge is to use the Wavelet Transformation, which fulfills both desired conditions of decomposition; information in both frequency and time domains are preserved [Steffen *et al.*, 1993]. For this experiment, we construct a 4-Band wavelet to decompose the features. Below are the filter banks for our 4-Band Wavelet:

$$\alpha = [-0.067371, 0.094195, 0.405805, 0.567372, \\ 0.567372, 0.405805, 0.094195, -0.067372] \tag{1}$$

$$\beta = [-0.094195, 0.067372, 0.567372, 0.405805, \\ -0.405805, -0.567372, -0.067372, 0.094195] \tag{2}$$

$$\gamma = [-0.094195, -0.067372, 0.56737, -0.405805, \\ -0.405805, -0.56737, -0.067372, -0.094195] \tag{3}$$

$$\delta = [-0.067372, -0.094195, 0.405805, -0.567372, \\ 0.567372, -0.405805, 0.094195, 0.067372], \tag{4}$$

where α is the low pass filter bank, and $\beta,\ \gamma,\ \delta$ are the high pass filter banks such that they satisfy the following conditions:

$$\sum_{i=1}^{8} \alpha_i = 2 \tag{5}$$

$$\sum_{i=1}^{8} \beta_i = \sum_{i=1}^{8} \gamma_i = \sum_{i=1}^{8} \delta_i = 0 \tag{6}$$

$$\|\alpha\|_2 = \|\beta\|_2 = \|\gamma\|_2 = \|\delta\|_2 \tag{7}$$

$$\langle \alpha, \beta \rangle = \langle \alpha, \gamma \rangle = \langle \alpha, \delta \rangle = \langle \beta, \gamma \rangle = \langle \beta, \delta \rangle = \langle \gamma, \delta \rangle = 0, \tag{8}$$

where $\| \cdot \|$ denotes the Euclidean norm, and $\langle \cdot \rangle$ denotes the standard inner product on $\mathbb{R}^n$.

An example where a signal $S \in \mathbb{R}^{4^k} (k \in \mathbb{N},\ k \geq 2)$, a 4-Band Wavelet Transform matrix W (see Fig. 2), is constructed by shifting and wrapping around the filter banks.

Let $W \in \mathbb{R}^{4^k} \times \mathbb{R}^{4^k}$ be a wavelet matrix. It is an orthonormal matrix since the column and row vector of W form a set of the orthonormal basis for $\mathbb{R}^{4k}$. To get into the frequency domain (F), we apply the wavelet transformation to the signal by simple matrix multiplication:

$$F = WS = [a \ \ d_1 \ \ d_2 \ \ d_3]^T \in \mathbb{R}^{4^k}, \tag{9}$$

where $a = [a_1, a_2, a_3, ..., a_{4^{k-1}}]$ is the low-frequency component, and $d_i = [d_{i,1}, d_{i,2}, d_{i,3}, ..., d_{i,4^{k-1}}] (i = 1, 2, 3)$, are the high frequency components. We separate each frequency component by setting other frequency components to zero.

$$A = [a \ \ 0 \ \ 0 \ \ 0]^T \tag{10}$$

$$D_1 = [0 \ \ d_1 \ \ 0 \ \ 0]^T \tag{11}$$

$$D_2 = [0 \ \ 0 \ \ d_2 \ \ 0]^T \tag{12}$$

$$D_3 = [0 \ \ 0 \ \ 0 \ \ d_3]^T, \tag{13}$$

We apply the inverse wavelet transform to the frequency component to get back to the signal domain. This transformation can be done as follows:

$$S_a = W^T A \tag{14}$$

$$S_{d_i} = W^T D_i \tag{15}$$

so that $S = S_a + \sum_{i=1}^{3} S_{d_i}$ where S_a approximate S and S_{d_i} represents the details of S. See Fig 3 for an example of this process.

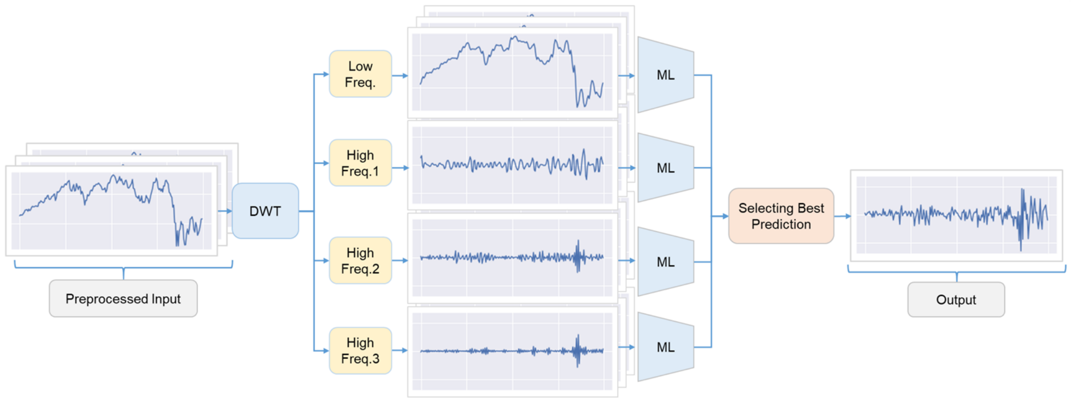

Figure 1: Proposed Framework – The input features are first normalized and then decomposed into frequency bands using Discrete Wavelet Transform (DWT). We assemble various types of machine learning (ML) models to each frequency band of the features. Based on the validation performance, the best model is selected by the combination of the best performing frequency band and machine learning model.

$$\begin{bmatrix}
\alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & \alpha_6 & \alpha_7 & \alpha_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & \alpha_6 & \alpha_7 & \alpha_8 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & \alpha_6 & \alpha_7 & \alpha_8 \\
\alpha_5 & \alpha_6 & \alpha_7 & \alpha_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 \\
\beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & \beta_6 & \beta_7 & \beta_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & \beta_6 & \beta_7 & \beta_8 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & \beta_6 & \beta_7 & \beta_8 \\
\beta_5 & \beta_6 & \beta_7 & \beta_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta_1 & \beta_2 & \beta_3 & \beta_4 \\
\gamma_1 & \gamma_2 & \gamma_3 & \gamma_4 & \gamma_5 & \gamma_6 & \gamma_7 & \gamma_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \gamma_1 & \gamma_2 & \gamma_3 & \gamma_4 & \gamma_5 & \gamma_6 & \gamma_7 & \gamma_8 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma_1 & \gamma_2 & \gamma_3 & \gamma_4 & \gamma_5 & \gamma_6 & \gamma_7 & \gamma_8 \\
\gamma_5 & \gamma_6 & \gamma_7 & \gamma_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma_1 & \gamma_2 & \gamma_3 & \gamma_4 \\
\delta_1 & \delta_2 & \delta_3 & \delta_4 & \delta_5 & \delta_6 & \delta_7 & \delta_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \delta_1 & \delta_2 & \delta_3 & \delta_4 & \delta_5 & \delta_6 & \delta_7 & \delta_8 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \delta_1 & \delta_2 & \delta_3 & \delta_4 & \delta_5 & \delta_6 & \delta_7 & \delta_8 \\
\delta_5 & \delta_6 & \delta_7 & \delta_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \delta_1 & \delta_2 & \delta_3 & \delta_4
\end{bmatrix}$$

Figure 2: An example of $4^2 \times 4^2$ Wavelet Transform matrix.

2.2 Machine Learning Models

After decomposing our features into four sets of data, we feed each data set separately into both traditional machine learning methods and deep learning methods to compare performances.

Linear Regression: Linear Regression (LR) aims to find the curve that best fits the data. This is done by determining the coefficients and variables such that the relationship between the dependent and independent variables is optimally described. The model estimates the best curve by minimizing the residual sum of squares between the labels provided by the data and the targets predicted by the linear approximation.

Decision Tree: A decision tree (DT) is constructed starting from the root node. Each node represents an output class and every branch represents the process that leads to the decision output, and the end node is the result [Rokach and Maimon, 2005].

Random Forest: Random forest (RF) is an extension of

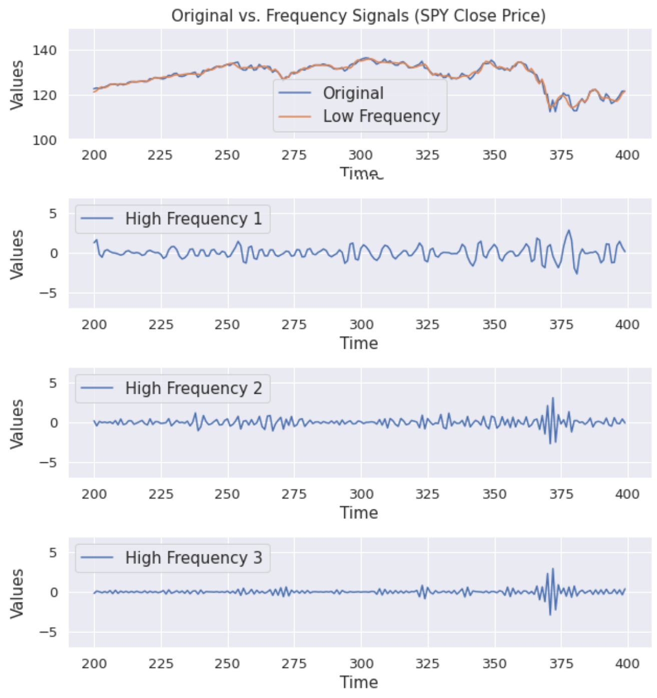

Figure 3: Decomposing an example signal into high- and low-frequency portions. The low-frequency signal closely represents the original signal, and the high-frequency signals represent the details occurring on smaller time scales.

bootstrap aggregation that uses multiple decision trees to help improve the stability and accuracy of the algorithm. Due to the fact the decision trees have high variances, the model tends to over-fit if the training data is complex and presents an irregular pattern or contains a lot of noise [Breiman, 2001; Krauss *et al.*, 2017].

eXtreme Gradient Boosting: The eXtreme Gradient Boosting (XGBoost) is an ensemble end-to-end tree boosting algorithm that applies the boosting for weak learners to convert them to strong learners. The XGBoost generates individual trees using multiple cores and organizes each data point to minimize the lookup times to get better performance and speed. The model has provided in-built cross-validation ability, efficient handling of missing data, regularization for avoiding over-fitting, catch awareness, tree pruning, and parallelized tree building [Nabipour *et al.*, 2020].

Support Vector Machine: Support Vector Machine (SVM) is a non-parametric kernel-based regression method used for extrapolating future values [Hearst *et al.*, 1998; Huang *et al.*, 2005]. More specifically, we shall be focusing on ϵ-SVR, a form of SVR in which a hyperplane is constructed with a loss function within ϵ precision. The SVR function can be expressed as :

$$f(x) = w^T\phi(x) + b, \tag{16}$$

where $\phi(x)$ maps data from the input space to the feature space, w is a weight vector, and b is a bias constant. w and b are estimated by satisfying as follows:

Minimizing: $\frac{1}{2}\|x\|^2$

Subject to:

$$\begin{cases} y_i - \langle w, \phi(x_i)\rangle + b \le \epsilon \\ \langle w, \phi(x_i)\rangle + b - y_i \le \epsilon, \end{cases} \tag{17}$$

where x_i and y_i represent input and target values obtained from the training set. To address points outside this ϵ-insensitive band, we introduce slack variables ξ_i, ξ_i^*:

Minimizing: $\frac{1}{2}\|x\|^2 + C\sum_i^n = 1(\xi_i + \xi_i^*)$

Subject to:

$$\begin{cases} y_i - (\langle w, \phi(x_i)\rangle + b) & \le \epsilon + \xi_i \\ (\langle w, \phi(x_i)\rangle + b - y_i & \le \epsilon + \xi_i^* \\ \xi_i, \xi_i^* & \le 0 \end{cases} \tag{18}$$

The constant $C > 0$ is used to represent the trade off between model complexity and training error. After taking the Lagrangian and optimizing with the above constraints, we are left with:

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(x_i, x) + b, \tag{19}$$

where $K(\cdot,\cdot)$ is a kernel function, α_i and α_i^* are nonzero Lagrangian multipliers and solutions to the dual problem.

Long Short Term Memory: The Long Short Term Memory (LSTM) contains special units called memory blocks in the recurrent hidden layer. These memory blocks contain memory cells with self-connections storing the temporal state of the network in addition to special multiplicative units called gates to control the flow of information. Each memory block in the original architecture contained an input gate and an output gate. The input gate controls the flow of input activation into the memory cell. The output gate controls the output flow of cell activation into the rest of the network. Later, the forget gate was added to the memory block. This addressed a weakness of LSTM models preventing them from processing continuous input streams that are not segmented into sub-sequences. The forget gate scales the internal state of the cell before adding it as an input to the cell through the self-recurrent connection of the cell, therefore adaptively forgetting or resetting the cell's memory. Besides, the modern LSTM architecture contains peephole connections from its internal cells to the gates in the same cell to learn the precise timing of the outputs [Sak *et al.*, 2014]. An LSTM network maps a sequence of input vectors $x = (x_1, ..., x_t)$ to a sequence of output vectors $y = (y_1, ..., y_t)$ by iteratively calculating the network unit activation using the following equation from $t = 1...T$:

$$\begin{array}{lcl} \text{Block input} & : & z_t = g(W_z x_t + V_z y_{t-1} + b_t) \\ \text{Input gate} & : & i_t = \sigma(W_i x_t + V_i y_{t-1} + b_i) \\ \text{Forget gate} & : & f_t = \sigma(W_f x_t + V_f y_{t-1} + b_f) \\ \text{Memory cell} & : & c_t = i_t \odot z_t + f_t \odot c_{t-1} \\ \text{Output gate} & : & o_t = \sigma(W_o x_t + V_o y_{t-1} + b_o) \\ \text{Block output} & : & y_t = o_t \odot g(c_t) \end{array}$$

where W and V denote input and recurrent weight matrices, respectively (e.g. W_i the weight matrix from the current input step to input gate i and V_i the recurrent weight matrix from the output of the previous step to input gate i the b terms denote bias weight vectors (e.g.b_i is the input gate bias weight vector), σ is the logistic sigmoid function, $g(x) = tanh(x)$, and i, f, o, c are output gate and memory cell activation vectors, respectively, all of which are the same size as the cell output activation vector m, $\odot$ is the element-wise product of the vectors.

Convolutional Network-Long Short Term Memory: The convolutional Network-Long Short Term Memory (CNN-LSTM) is one of the variants of the LSTM model [Liu *et al.*, 2017]. The CNN-LSTM utilizes the output of the convolutional networks as the input of the LSTM. A CNN network is formed using convolutional layers which perform a convolutional operation. The model plays an important role in reducing the parameters using filters, max-pooling, dropout, and fully connected layers so that the network only extracts the most meaningful features from input data.

3 Experiments

3.1 Data Collection

For our experiments, we are collecting data from three U.S. ETFs: **SPY**, **DIA**, and **QQQ**, which track the major indices of the S&P 500, Dow Jones Industrial Average, and NASDAQ, respectively. We collect the daily quote data from January 1st, 2010 to January 1st, 2020 using Yahoo! Finance API[1]. The features used in these datasets consist of $Open, High, Low, Close, Volume$ and technical indicators computed from $Close$ prices which are *Moving Average Convergence Divergence (MACD), Bollinger bands (BBANDS), Relative Strength Index (RSI)*. Our target is the percentage change of $Close$ price for the next m days. Thus, the goal is to use n-previous days of features to predict the percentage change of $Close$ price for the next m days. The baseline formulation before wavelet transforms is below:

$$T_m = f(X_0, X_{-1}, ..., X_{-(n-1)}, X_{-n}), \tag{20}$$

where T_m is the percentage change of $Close$ price for the next m days, X is the set of all the features, and $f(\cdot)$ is the

[1]https://finance.yahoo.com/

non-linear mapping function. We then split the data in sequential order into training/testing sets. For this experiment, the data is split into 60%/10%/30% for training, validation, and testing sets, respectively. Technical indicators are often used by many traders to identify the change in patterns. The most popular ones are *Moving Averages, Bollinger Bands*, and *Relative Strength Index*. Many existing studies have only used *Open-High-Low-Close (OHLC)* values as input, but we believe that incorporating technical indicators as additional features would increase the performance of machine learning models. Below is the description of our features:

Features	Description
MACD	A trend-following momentum indicator.
RSI	An oscillator that indicates the internal strength of a signal
BBAND	Standard deviation level above/below a simple moving average
Volume	Number of shares traded
High	Highest price reached in the day
Low	Lowest price reached in the day
Open	The price of the stock opened at market
Close	Close price adjusted for splits
Adj Close	Adjusted close price adjusted for both dividends and splits

Table 1: List of features used in our framework for stock movement forecasting.

3.2 Model Training

We train eight machine learning models using all stock features, including decomposed features, respectively; LR, DT, RF, GB, XGB, SVR, LSTM, and CNN-LSTM. All parameters of the machine learning models are set by the default values of the *sklearn* library. The LSTM consists of four LSTM layers with 50 units and four drop-outs of 0.2. The drop-out is located after each LSTM layer. For CNN-LSTM, three 1-dimensional CNNs consisting of 5 kernel sizes, 1 stride size, and one LSTM layer with 30 units, and one drop-out of 0.2 are utilized. The filter size of the CNN layers varies 64, 64, and 32 for each layer. As hyper-parameters, for both LSTM and CNN-LSTM models, Adam optimizers are selected with an 0.001 learning rate. They are trained using Python.

3.3 Evaluation Metrics

We evaluate our approach by using multiple metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and accuracy. The MAE method represents the difference between the actual and the predicted values over the data set by averaging the absolute difference. The RMSE is the error rate by the square root of the mean squared error between actual and predicted values.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |\overline{y_i} - y_i|}{n} \tag{21}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\overline{y_i} - y_i)^2} \tag{22}$$

where n is the number of data points, y_i is the observed values, and $\overline{y_i}$ is the predicted values.

This paper utilizes the accuracy of the binary classification, which estimates either down or up of the stock movements. We define the down as 0 when $\overline{y_i}$ and y_i are less or equal to 0, up as 1 and $\overline{y_i}$ and y_i are more than 0 (Eq. 19). The accuracy is calculated by the sum of true negative (TN) and true positive (TP) over test sets. The TN is the number of that $\overline{y_i}$ and y_i are correctly identified as down, and the TP is the number of that $\overline{y_i}$ and y_i are correctly identified as up.

$$y_i,\, \overline{y_i} = \begin{cases} 0, & y_i,\, \overline{y_i} \le 0 \\ 1, & y_i,\, \overline{y_i} > 0. \end{cases} \tag{23}$$

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{All Samples}}. \tag{24}$$

3.4 Numerical Experiments

In this experiment, the input features for the baseline models are MACD, RSI, BBAND, and Volume from the feature importance analysis. For our proposed method, we decompose each feature into low and high-frequency signals using the wavelet transformation. We then run various machine learning models for each set of features and calculate their performance using the metrics mentioned earlier.

Predicting Price One Day Ahead

From our results (see Tables 2 - 4), using high-frequency signals yields better results than both the original data and low-frequency signals (see the black boldfaced digits from tables). Across the machine learning models being experimented with, SVR is the best model when using high-frequency #2 as input features (see red boldfaced digits). For all three ETFs, the accuracy is above 72%, and the errors (MAE and RMSE) are the lowest compared to other traditional statistical and machine learning models. It is also worth pointing out that when using frequency signals as input, these models also outperform the models that use the unaltered features. In our study, LSTM and CNN-LSTM did not perform as well as expected. We suspect that the data in this experiment is not complex enough for deep machine learning models, which may explain our observation that traditional statistical and machine learning methods outperform deep learning in our study.

Predicting Price Multiple Days Ahead

To confirm our hypothesis that high-frequency signals have a more significant influence on the short-term stock movement, we run experiments to forecast multiple days ahead by using n-days future as targets. From the results (see Fig. 4), we learn that the performances of the models using high-frequency features drop when attempting to predict more than three days. The results further suggest that short-term stock movement is more influenced by signals in the high frequency portions of its decomposition. From the earlier experiment (forecasting one day ahead) using deep learning methods (LSTM and CNN-LSTM), we observe the accuracy for one-day movement prediction is almost a coin-toss. However, as we predict multiple days, the accuracy increases. We currently do not have an explanation for this behavior; this is left for future research. We also note that as we attempt to predict a long term period, the errors (RMSE) increase due to a higher level of uncertainty.

Method	Original Signals			Low Frequency Signals			High Frequency Signals #1			High Frequency Signals #2			High Frequency Signals #3		
	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)
LR	0.544	0.816	48.32	0.542	0.804	55.57	0.550	0.824	57.99	**0.496**	**0.723**	**69.53**	0.527	0.772	64.16
DT	0.572	0.935	55.97	0.584	0.981	56.78	0.538	0.824	47.38	**0.507**	**0.713**	**67.79**	0.561	0.836	64.56
RF	0.848	1.247	49.80	0.805	1.165	53.15	0.758	1.100	52.48	**0.701**	**0.995**	**65.37**	0.736	1.073	58.12
GB	0.606	0.948	48.46	0.569	0.880	55.17	0.591	0.905	58.79	**0.538**	**0.797**	**69.53**	0.555	0.839	65.10
XGB	0.558	0.834	55.84	0.558	0.832	56.24	0.554	0.823	55.57	**0.523**	**0.785**	**59.60**	0.541	0.806	57.58
SVR	0.547	0.830	52.48	0.534	0.805	58.66	0.543	0.820	58.26	**0.444**	**0.683**	**72.35**	0.515	0.771	64.16
LSTM	0.539	0.813	52.37	0.550	0.805	49.19	0.537	0.805	50.97	0.507	0.735	50.55	**0.523**	**0.761**	**52.59**
CNN LSTM	**0.551**	**0.841**	**55.11**	0.573	0.877	52.73	0.572	0.876	53.05	0.704	1.409	50.75	0.557	0.854	52.11

Table 2: SPY RESULTS COMPARISON

Method	Original Signals			Low Frequency Signals			High Frequency Signals #1			High Frequency Signals #2			High Frequency Signals #3		
	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)
LR	0.556	0.834	47.38	0.549	0.822	57.99	0.562	0.842	55.57	**0.518**	**0.750**	**69.66**	0.537	0.785	65.37
DT	0.596	0.972	55.70	0.554	0.831	57.05	0.576	0.875	54.23	**0.520**	**0.725**	**69.66**	0.540	0.799	66.31
RF	0.811	1.185	52.08	0.764	1.114	55.84	0.802	1.105	52.89	**0.668**	**0.949**	**64.56**	0.698	0.979	60.94
GB	0.623	0.983	49.80	0.592	0.949	57.72	0.609	0.941	57.18	**0.561**	**0.844**	**70.34**	0.561	0.842	65.91
XGB	0.565	0.847	55.44	0.573	0.857	55.84	0.561	0.830	55.17	**0.534**	**0.803**	**59.73**	0.550	0.829	57.45
SVR	0.565	0.853	55.03	0.550	0.832	57.05	0.554	0.832	58.26	**0.463**	**0.708**	**72.08**	0.519	0.777	65.50
LSTM	**0.554**	**0.832**	52.26	0.556	0.821	49.67	0.560	0.841	**52.76**	0.554	0.779	49.92	0.524	0.780	50.79
CNN LSTM	0.579	0.919	**53.01**	0.578	0.849	49.15	**0.571**	**0.873**	51.29	0.680	1.366	50.22	0.557	0.850	52.14

Table 3: DIA RESULTS COMPARISON

Method	Original Signals			Low Frequency Signals			High Frequency Signals #1			High Frequency Signals #2			High Frequency Signals #3		
	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)	MAE	RMSE	Acc.(%)
LR	0.755	1.096	50.07	0.542	0.804	55.57	0.550	0.824	57.99	**0.496**	**0.723**	**69.53**	0.527	0.772	64.16
DT	0.760	1.138	57.85	0.584	0.981	56.78	0.538	0.824	47.38	**0.507**	**0.713**	**67.79**	0.561	0.836	64.56
RF	0.940	1.320	50.87	0.805	1.165	53.15	0.758	1.100	52.48	**0.701**	**0.995**	**65.37**	0.736	1.073	58.12
GB	0.819	1.199	51.95	0.569	0.880	55.17	0.591	0.905	58.79	**0.538**	**0.797**	**69.53**	0.555	0.839	65.10
XGB	0.746	1.096	57.72	0.558	0.832	56.24	0.554	0.823	55.57	**0.523**	**0.785**	**59.60**	0.541	0.806	57.58
SVR	0.747	1.109	56.64	0.534	0.805	58.66	0.543	0.820	58.26	**0.444**	**0.683**	**72.35**	0.515	0.771	64.16
LSTM	0.736	1.087	**57.99**	0.541	0.800	51.21	0.547	0.806	49.38	**0.487**	**0.710**	51.48	0.522	0.761	51.82
CNN LSTM	0.796	1.244	57.69	0.586	0.916	49.72	**0.541**	**0.825**	**51.98**	0.660	1.277	49.49	0.585	0.884	49.94

Table 4: QQQ RESULTS COMPARISON

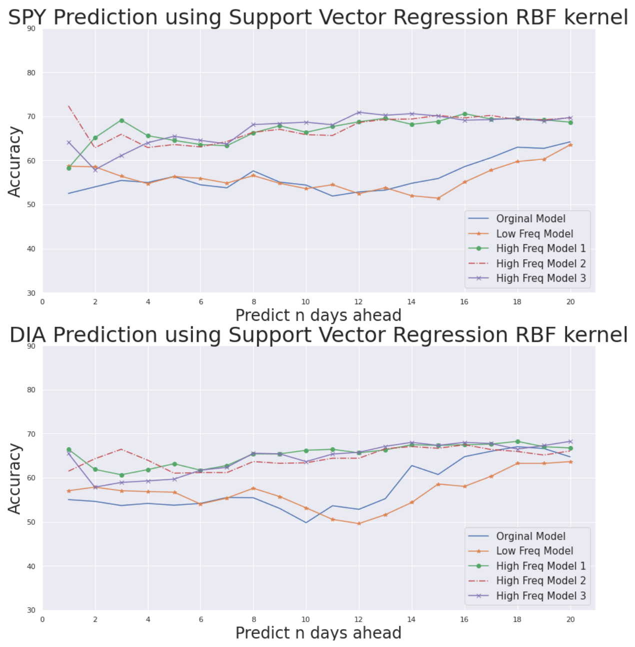

Figure 4: Accuracy of predicting multiple days ahead using Support Vector Machine

4 Conclusion & Further Research

In this paper, we study the use of the wavelet transformation to decompose features into low/high-frequency signals and use them as input for our forecasting models. As demonstrated in our experiments, high-frequency signals of the features yield better short-term prediction results of the three major ETFs (SPY, DIA, & QQQ) in terms of errors and accuracy compared to using the original signals. Short-term market movements are caused by many external factors, and the results of this study suggest that many of which are captured within the high-frequency signals. Thus, being able to decompose signals into high/low frequency portions may prove valuable in forecasting short-term stock movement.

This study investigated the use of a Wavelet transform with a single level of decomposition. We note that additional levels of decomposition may further isolate distinct time scales of the original signal, and consequently improve forecasting accuracy. Future studies will experiment with multiple levels of decomposition and further validate the influence of high frequency signals on short-term stock movements. We then plan to apply this framework to different types type of data and features. Furthermore, we would also like to include portfolio optimization based on the proposed prediction methods.

Ethical Statement

The authors suspect there are no ethical issues in this study.

References

[Boussaada *et al.*, 2018] Zina Boussaada, Octavian Curea, Remaci Ahmed, Haritza Camblong, and mrabet bellaaj Najiba. A nonlinear autoregressive exogenous (narx) neural network model for the prediction of the daily direct solar radiation. *Energies*, 11:620, 03 2018.

[Breiman, 2001] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Chen *et al.*, 2015] Kai Chen, Yi Zhou, and Fangyan Dai. A lstm-based method for stock returns prediction: A case study of china stock market. In *2015 IEEE international conference on big data (big data)*, pages 2823–2824. IEEE, 2015.

[Chong *et al.*, 2017] Eunsuk Chong, Chulwoo Han, and Frank C Park. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Systems with Applications*, 83:187–205, 2017.

[Fattah *et al.*, 2018] Jamal Fattah, Latifa Ezzine, Zineb Aman, Haj Moussami, and Abdeslam Lachhab. Forecasting of demand using arima model. *International Journal of Engineering Business Management*, 10:184797901880867, 10 2018.

[Hearst *et al.*, 1998] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998.

[Huang *et al.*, 2005] Wei Huang, Yoshiteru Nakamori, and Shou-Yang Wang. Forecasting stock market movement direction with support vector machine. *Computers & operations research*, 32(10):2513–2522, 2005.

[Krauss *et al.*, 2017] Christopher Krauss, Xuan Anh Do, and Nicolas Huck. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the s&p 500. *European Journal of Operational Research*, 259(2):689–702, 2017.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[Lai *et al.*, 2018] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

[Liu *et al.*, 2017] Shuanglong Liu, Chao Zhang, and Jinwen Ma. Cnn-lstm neural network model for quantitative strategy analysis in stock markets. In *International Conference on Neural Information Processing*, pages 198–206. Springer, 2017.

[Nabipour *et al.*, 2020] M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, E. Salwana, and Shahab S. Deep learning for stock market prediction. *Entropy*, 22(8):840, Jul 2020.

[Nguyen and Wang, 2016] Hieu Q Nguyen and Xiaodi Wang. Pseudo quantum steganography with "color barcode" in m-band wavelet domain. *International Journal of Signal Processing*, 1:160–168, 2016.

[Oord *et al.*, 2016] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[Oreshkin *et al.*, 2019] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

[Pai and Lin, 2005] Ping-Feng Pai and Chih-Sheng Lin. A hybrid arima and support vector machines model in stock price forecasting. *Omega*, 33(6):497–505, 2005.

[Rahimyar *et al.*, 2019] Abdul Hasib Rahimyar, Hieu Quang Nguyen, and Xiaodi Wang. Stock forecasting using m-band wavelet-based svr and rnn-lstms models. In *2019 2nd International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pages 234–240. IEEE, 2019.

[Rokach and Maimon, 2005] Lior Rokach and Oded Maimon. *Decision Trees*, volume 6, pages 165–192. 01 2005.

[Sak *et al.*, 2014] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *arXiv preprint arXiv:1402.1128*, 2014.

[Schmidhuber, 2015] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[Sezer *et al.*, 2020] Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing*, 90:106181, 2020.

[Steffen *et al.*, 1993] Peter Steffen, Peter N Heller, Ramesh A Gopinath, and C Sidney Burrus. Theory of regular m-band wavelet bases. *IEEE Transactions on Signal Processing*, 41(12):3497–3511, 1993.

[Wang *et al.*, 2018] Jingyuan Wang, Ze Wang, Jianfeng Li, and Junjie Wu. Multilevel wavelet decomposition network for interpretable time series analysis. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2437–2446, 2018.

[Wu, 2020] X.; Su J.; Tang B.; Wu S. Wu, D.; Wang. A labeling method for financial time series prediction based on trends. *Journal of Entropy*, 22:1162, 2020.

[Zhao *et al.*, 2018] Yi Zhao, Yanyan Shen, Yanmin Zhu, and Junjie Yao. Forecasting wavelet transformed time series with attentive neural networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1452–1457. IEEE, 2018.