Enhancing Financial Time Series Prediction with Wavelet Denoising and LSTM Networks: A Comprehensive Study

First Author[1 [0000−1111−2222−3333]], Second Author[2,3 [1111−2222−3333−4444]], and Third Author[3 [2222−−3333−4444−5555]]

No Institute Given

Abstract. This paper discusses using different kinds of wavelet transformations to reduce noise in time series financial data. Different levels and kinds of wavelet decomposition and reconstruction of time series data are discussed along with their varying effectiveness in noise reduction. Experimental results demonstrate that the performance of pure vanilla LSTM (Long Short Term Memory) neural networks for prediction of time series data undergoes a remarkable improvement when using denoised data instead. This paper further discusses the reasons behind the improvement in prediction accuracy and discusses possible further improvement.

Keywords: LSTM · Wavelets · Time Series Data · Noise Reduction.

1 Introduction

Financial time series data like stock prices and market indices are inherently noisy, non-linear and non-stationary. These characteristics of the data arise as a result of the complex interaction of market sentiment, geopolitical events and the transactional activities of institutional and retail traders. The noise in financial data that arises from high frequency trading, random fluctuations and market microstructure noise can often make it difficult to discern the underlying trends in price movement which makes accurate prediction challenging.[1]

Traditional statistical models for time series forecasting like (Auto Regressive Integrated Moving Average) ARIMA models have been widely used for stock price prediction[2] but they struggle with the non-linear patterns and complex relationships in financial time series data. In contrast, machine learning models like decision trees, Support Vector Machines (SVM) and random forest models have also shown promising results.[3] These models are able to capture non-linear relationships but they often require extensive feature engineering and they are generally unable to detect and utilize long term sequential dependencies in the data.

This is where Recurrent Neural Network (RNN) models tend to perform better because of their ability to capture these long term trends while forecasting. Among RNN models, Long Short Term Memory (LSTM) networks are the

most popular choice because they address the vanishing gradient problem that is common in standard RNNs, allowing them to learn and retain information over extended periods. [4]

Despite their advantages, LSTMs and similar models are sensitive to noisy input data and this can affect their predictive accuracy. This has led to the exploration of a variety of data preprocessing and denoising techniques to improve the quality of the input data to such predictive models. [5][6] Traditional denoising methods like moving averages and exponential smoothing often are unable to remove more complex noise from time series signals. Wavelet transformations are a great alternative that are able to decompose time series signals into true signal and noise at multiple resolution levels.

In this paper, we investigate the impact of wavelet-based denoising on the performance of LSTM neural networks. First, we provide a comprehensive analysis of various wavelet transformation techniques and their effectiveness in denoising financial time series data. Second, we present empirical evidence demonstrating the improvements in LSTM model performance when trained on denoised data. Third, we offer insights on where and why the performance improvements occurred and we discuss potential avenues for further enhancing predictive accuracy through advanced denoising methods.

2 Data Preprocessing with Wavelets

2.1 Data Used

The data used in this paper is historical closing price data for the Indian stock index NIFTY 50. The dataset used is for hourly stock data. The hourly dataset used ranged from 3rd Jan 2022 to 29th Dec 2023. Each day corresponds to 7 hourly data points. This data is available in the GitHub repository.

The data used shows significant seasonality on a daily and weekly level as seen in figures 1 and 2.

2.2 Wavelet Transformations

Wavelet transformations are a powerful mathematical transformation used for signal processing that enable the analysis of time series data at multiple scales and resolutions. Wavelet transformations help in decomposing a signal into components that represent both low frequency details and high frequency details (which include noise).

Discrete Wavelet Transformations (DWT) are a specific type of wavelet transformations specifically used for discrete signals like financial time series data. DWT works by passing a signal through a series of high-pass and low-pass filters to decompose the signal into high frequency and low frequency components. This process can ideally be repeated multiple times to capture components at different frequency levels.

At each level of decomposition, 2 sets of coefficients are generated, approximation coefficients (A) representing the low frequency component and detail

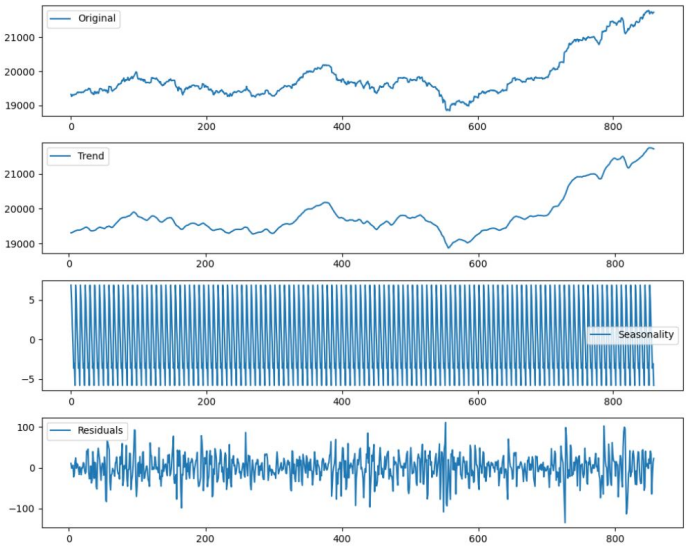

Fig. 1. Daily Decomposition of Hourly Data

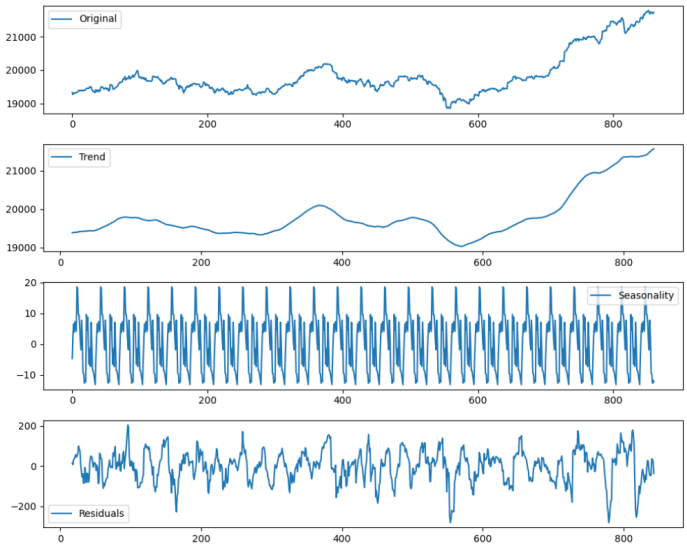

Fig. 2. Weekly Decomposition of Hourly Data

coefficients (D) representing the high frequency component. The approximation coefficients are the values that can be further decomposed at each stage to generate a higher level of decomposition as seen in figure 3.

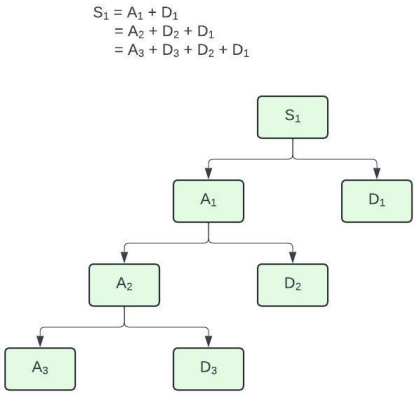

Fig. 3. Multilevel Wavelet Decomposition

2.3 Wavelet Transformations for Denoising

The objective of the denoising process is to reduce noise as much as possible while not affecting the underlying true signal too much. The wavelet based denoising process involves three main steps: decomposition, thresholding, and reconstruction.

Decomposition The input time series data is decomposed into approximation and detail coefficients using DWT. The detail coefficients generated could be at different levels.

Thresholding The noise present in the high frequency detail coefficients is removed or reduced by applying a thresholding technique. The 2 main kinds of thresholding that can be used are soft and hard thresholding:

1. Hard Thresholding: This method sets coefficients below a certain threshold λ to zero, while retaining coefficients above the threshold.

$$T_H(x, \lambda) = \begin{cases} x & \text{if } |x| \geq \lambda \\ 0 & \text{if } |x| < \lambda \end{cases} \tag{1}$$

2. Soft Thresholding: This method sets coefficients below the threshold λ to zero and reduces the remaining coefficients by λ.

$$T_S(x, \lambda) = \begin{cases} \text{sgn}(x) \cdot (|x| - \lambda) & \text{if } |x| \geq \lambda \\ 0 & \text{if } |x| < \lambda \end{cases} \tag{2}$$

Although soft thresholding generally provides better smoothening and noise reduction, it also causes more distortion of the underlying signal.

Choosing an appropriate threshold for the denoising process is crucial and there are a variety of techniques available.

In this paper, we have calculated the results of LSTM performance using hard thresholding with setting λ to ∞ (i.e. setting all the detail coefficients to 0).

Reconstruction The denoised signal is then reconstructed from the approximation and detail coefficients by applying the inverse discrete wavelet transformation (IDWT).

2.4 Wavelet Families Used

The wavelet families used in this paper are the Daubechies Wavelets (db), Symlets (sym) and Coiflets (coif). The wavelets used are db1 - db20, sym2 - sym20 and coif1 - coif17. The wavelet transformations were calculated using the PyWavelets library in Python. [7]

Daubechies Wavelets (db) Daubechies Wavelets are optimal for time-frequency localization because they demonstrate the properties of compact support and orthogonality. Their compact support ensures that they are non-zero over only a finite interval which makes them good at detecting sudden changes or spikes in time series data. In addition, their property of orthogonality means that the wavelet representation is efficient and non-redundant with precise reconstruction of the original signal possible from the calculated wavelet coefficients. However Daubechies wavelets are often asymmetric and this can introduce phase distortion which could lead to a misalignment of the features of the signal in the time domain.

In general, Daubechies wavelets of the order N have a filter length of 2N and N vanishing moments. This means that that the wavelet can effectively represent polynomial trends of a degree up to (N-1).

Symlets (sym) Symlets are a modified version of Daubechies wavelets that were designed to achieve near symmetry while maintaining the properties of orthogonality and compact support. While, the symmetric nature of these wavelets helps to reduce phase distortion so the timing of signal features is preserved more accurately, the larger compact support of sym wavelets slightly reduces their precision in time localization.

In general, symlets of the order N also have a filter length of 2N and N vanishing moments.

Coiflets (coif) Coiflets are a family of wavelets in which both the scaling function and the wavelet function have vanishing moments. This distinguishes them from Daubechies wavelets and Symlets, where typically only the wavelet function has vanishing moments. Coiflets are smoother than Daubechies wavelets of the same order which makes them ideal for capturing gradual and slow changing trends in time series data. However at the same time, their comparatively longer support leads to reduced time localization leading to a lower ability to capture abrupt changes in the data.

In general, coiflets of the order N have a filter length of 6N with both their wavelet and scaling functions having N vanishing moments. This means that that the wavelet can effectively represent polynomial trends of a degree up to (N-1).

2.5 Effects of Wavelet Denoising

In this research paper, we covered 3 separate levels of wavelet denoising of the input time series data. In general, we can see that with each subsequent level of wavelet denoising, there is a greater smoothening of the input signal and lower and lower constituent frequencies of the input signal are treated as noise and discarded. We can see this in the graphs below for the "coif10" wavelet applied to hourly data.

In figures 4 and 5, we can see the original signal plotted alongside the denoised signals. This plot in itself doesn't clearly show the effects of smoothening on the time series signal.

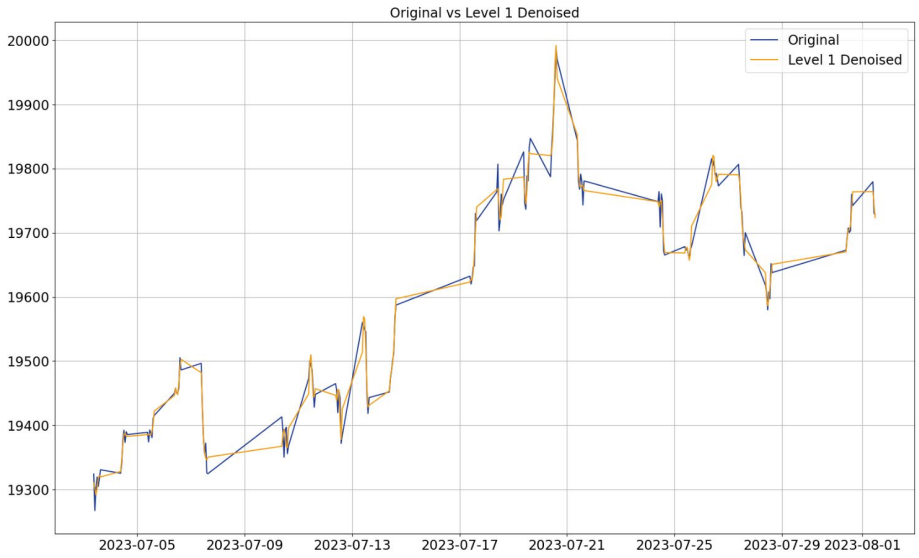

Fig. 4. Original vs Level 1 Denoised Signal

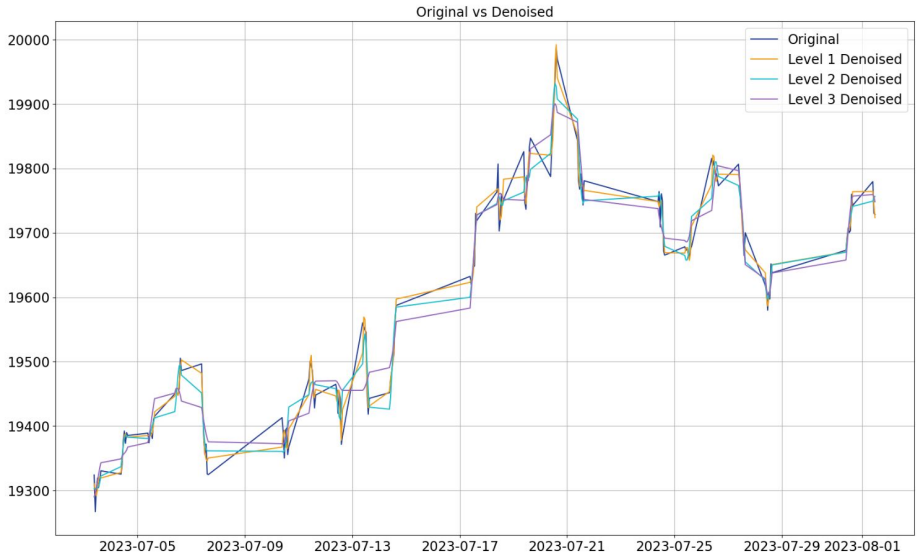

Fig. 5. Original vs Multiple Level Denoised Signals

We can see this more clearly in figures 6 and 7 where we plot the difference between consecutive data points i.e. y(t) - y(t-1) values in a histogram. We see that with each subsequent level of decomposition, the range of differences possible between consecutive data points reduces in magnitude and the likelihood of large differences between consecutive data points decreases. This is a visual representation of data smoothening.

We can see in the Power Spectral Density plot in figure 8 that with each subsequent level of wavelet denoising, lower and lower frequencies of the original signal are treated as noise and removed.

3 LSTM Models

Long Short Term Memory (LSTM) networks are powerful tools for tasks that involve sequential dependencies, such as time series prediction because of their ability to learn and retain information over extended periods.

LSTM networks are composed of units called LSTM cells, which replace the standard neurons in RNNs. Each LSTM cell (See figure 9) maintains a cell state associated with long-term memory and a hidden state associated with short-term memory, both of which are updated over time:

Cell State (C_t)**:** Acts as a long-term memory that contains relevant information for sequence processing.

Hidden State (h_t)**:** Represents the output of the LSTM cell at each time step, which is used for prediction and updating the long-term memory present in the cell state.

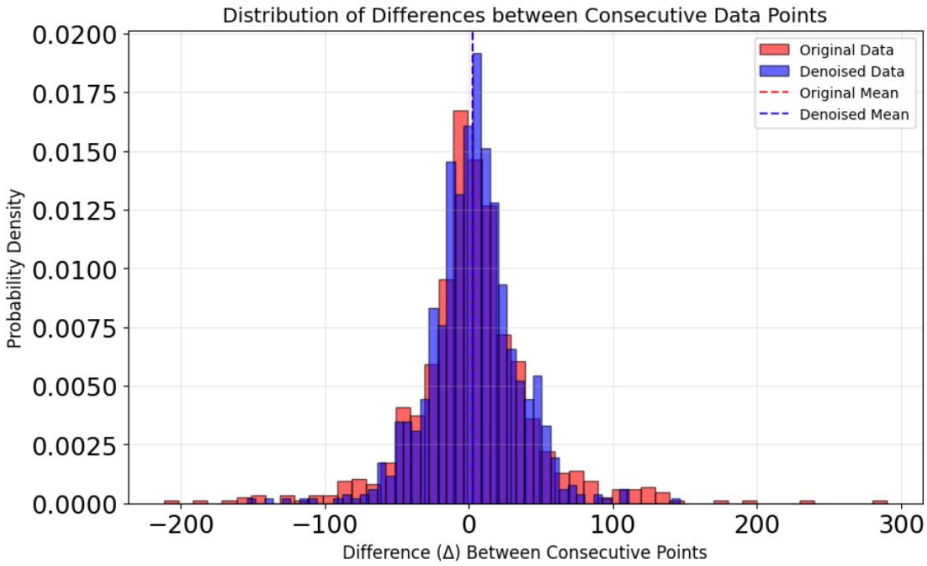

Fig. 6. Frequency Distribution of Differences Between Consecutive Data Points (i.e. y(t) - y(t-1) values) in the Time Series Data. Here, we compare the original data with 1 level of denoising.

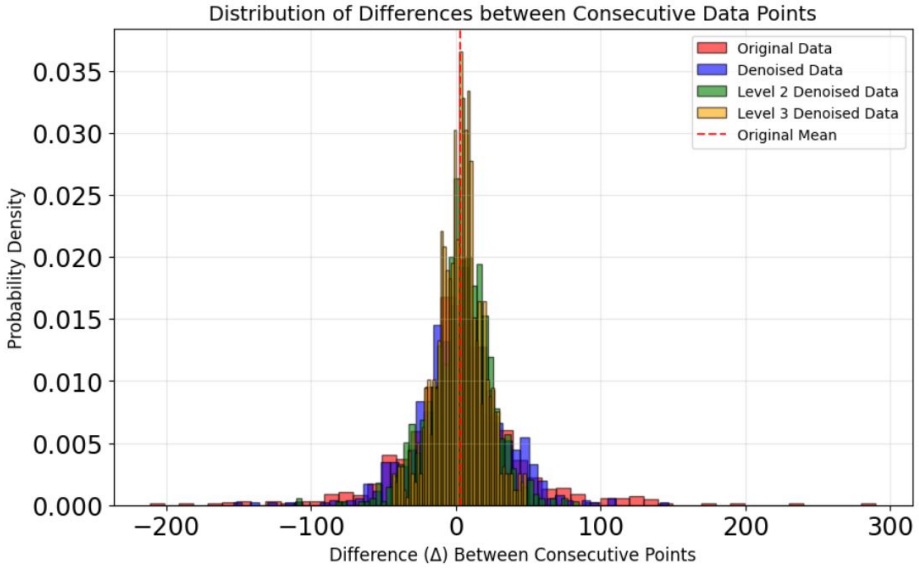

Fig. 7. Frequency Distribution of Differences Between Consecutive Data Points (i.e. y(t) - y(t-1) values) in the Time Series Data. Here, we compare the original data with all 3 levels of denoising.

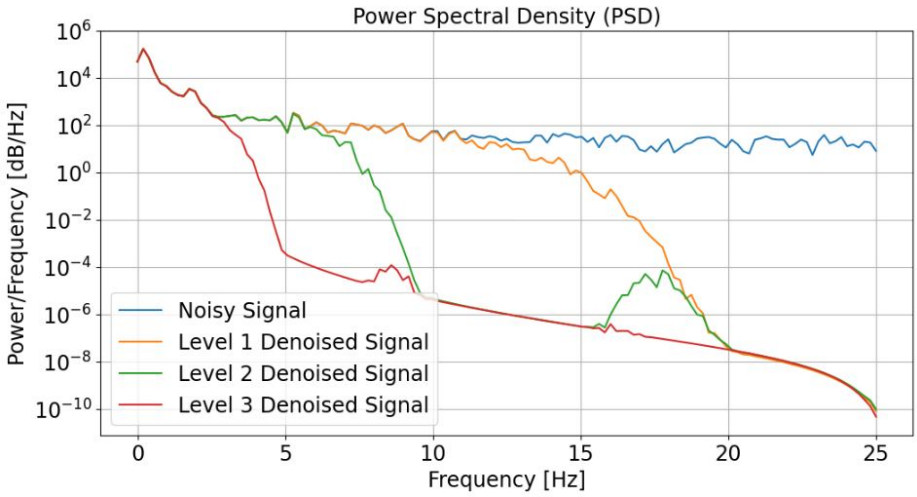

Fig. 8. Constituent Frequencies of Original and Denoised Time Series Data

The internal structure of an LSTM cell includes three main gates that control the flow of information:

Forget Gate (f_t)**:** This gate determines the extent to which information previously stored in the long-term memory should be retained. It uses the sigmoid activation function:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$

Input Gate: This gate decides what new information will be added to the long-term memory in the cell state. It consists of a sigmoid layer and a tanh layer that create values for updating the state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5}$$

The cell state is updated as follows:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{6}$$

Output Gate: This gate calculates the output of the LSTM cell based on the cell state and the input at the current time step. The hidden state of the cell is updated as well.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t \cdot \tanh(C_t) \tag{8}$$

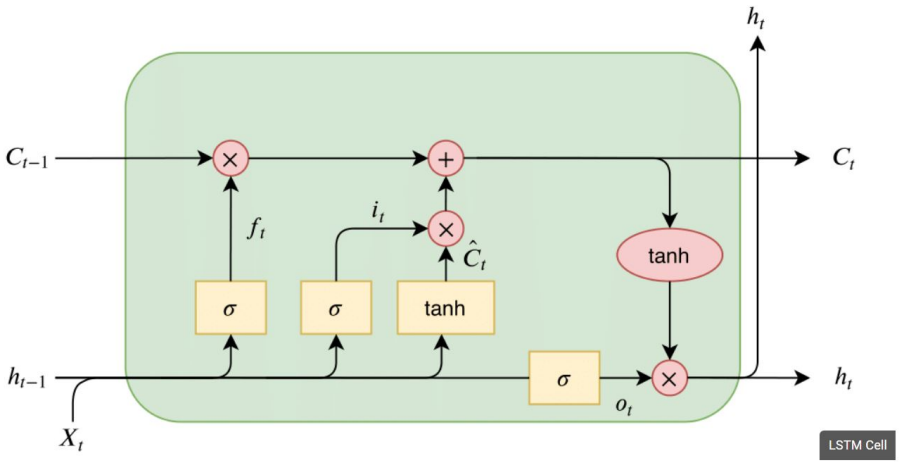

Fig. 9. An LSTM Cell

4 Wavelet LSTM Model Results

In tables 4, 7 and 10 are the results we obtained from training LSTM models on wavelet denoised data. The LSTM model used has 2 LSTM layers of length 500 followed by 3 dense layers and in each case the model is trained with a lookback period of 14 for 30 epochs. The models trained on wavelet denoised data are compared to the baseline model in which the model is trained on the original noisy data. The exact code and model architecture used can be seen in the GitHub repository.

4.1 Metrics Used

The accuracy of the models used is measured not only in terms of **RMSE** (Root Mean Square Error) but also in terms of the percentage of data points in which the predicted price is within 1% or 0.1% of the actual price. Using these percentage threshold values is able to give a more accurate picture of the data than using something like **MAPE** (Mean Absolute Percentage Error) which is strongly affected by outliers.

4.2 Results

The results for a pure LSTM model trained on noisy input data is given in table 1.
The minimum values in each row are in **bold**.

	RMSE	Percentage of Predicted Values Within 1%	Percentage of Predicted Values Within 0.1%
Pure	48.67	99.65	43.7

Table 1: Performance Metrics for the Pure Model

Level Of Decomposition	db1	db2	db3	db4	db5	db6	db7	db8	db9	db10
Level 1	46.26	43.61	42.82	40.68	***40.21***	42.37	44.19	40.41	41.99	43.86
Level 2	47.45	38.76	42.76	42.56	36.56	37.12	38.61	***35.57***	36.77	39.35
Level 3	54.36	51.8	42.21	46.54	41.92	45.5	42.48	44.99	***40.41***	44.23

Level Of Decomposition	db11	db12	db13	db14	db15	db16	db17	db18	db19	db20
Level 1	42.78	40.8	41.79	41.03	41.86	42.75	43.01	40.36	43.14	42.03
Level 2	37.89	35.66	38.39	38.07	36.8	36.79	38.15	40.29	37.18	40.62
Level 3	42.09	43.81	42.06	41.79	43.73	41.6	43.63	40.85	44.76	41.85

Table 4: RMSE Values for LSTM Models Trained on Data Denoised Using Db Wavelets

Level Of Decomposition	coif1	coif2	coif3	coif4	coif5	coif6	coif7	coif8	coif9
Level 1	43.74	42.55	41.96	44.70	45.24	43.01	42.85	40.24	42.83
Level 2	43.05	41.67	40.03	39.28	38.35	40.29	39.67	40.61	39.19
Level 3	45.37	49.01	40.87	46.20	39.92	44.39	41.02	45.86	***40.27***

Level Of Decomposition	coif10	coif11	coif12	coif13	coif14	coif15	coif16	coif17
Level 1	43.35	42.52	43.57	***39.79***	44.42	42.33	41.75	41.47
Level 2	36.31	38.51	37.49	37.73	40.32	***36.25***	37.57	36.53
Level 3	44.50	42.74	43.81	41.30	43.40	43.07	44.85	41.59

Table 10: RMSE Values for LSTM Models Trained on Data Denoised Using Coif Wavelets

Here are the average results for each wavelet type and level of decomposition:

Level Of Decomposition	sym2	sym3	sym4	sym5	sym6	sym7	sym8	sym9	sym10	sym11
Level 1	45.66	44.76	43.37	43.44	41.04	43.86	42.76	41.94	44.06	45.70
Level 2	40.42	41.84	44.84	40.65	35.97	37.20	38.93	38.27	36.01	35.92
Level 3	52.51	43.57	41.69	44.31	43.47	41.20	45.36	46.43	43.15	46.87

Level Of Decomposition	sym12	sym13	sym14	sym15	sym16	sym17	sym18	sym19	sym20
Level 1	43.80	45.24	42.47	42.18	41.83	49.07	41.88	***39.32***	46.02
Level 2	38.75	40.66	35.19	37.64	38.44	38.85	***34.76***	36.68	38.57
Level 3	40.94	43.75	42.01	***40.77***	44.19	43.22	43.88	43.74	41.67

Table 7: RMSE Values for LSTM Models Trained on Data Denoised Using Sym Wavelets

Level Of Decomposition	db	sym	coif	**Average**
Level 1	42.30	43.60	42.72	**42.87**
Level 2	38.77	38.40	38.99	**38.72**
Level 3	44.03	43.83	43.42	**43.76**
Average	**41.70**	**41.94**	**41.71**	**41.78**

Table 11: Comparison of Average RMSE Values for Different Wavelet Families

From Table 11, we can see that the most important factor affecting effectiveness of the denoised LSTM models is the level of wavelet denoising with level 2 denoising being clearly more effective than level 1 or level 3 denoising regardless of wavelet family used. In addition, we can see that the average results displayed by the 3 wavelet families are quite similar. Further, the average RMSE for level 2 denoising (38.72) shows a 20.44% decrease from the RMSE of the pure LSTM (48.67).
So, when selecting wavelets to denoise your data, first calculate the average performance of a few wavelets at multiple levels of decomposition and identify the optimal level of decomposition. After that, the wavelet family used does not seem to have a particularly meaningful effect on performance. On selecting a wavelet family, one can tune for the wavelet with the optimal filter length.

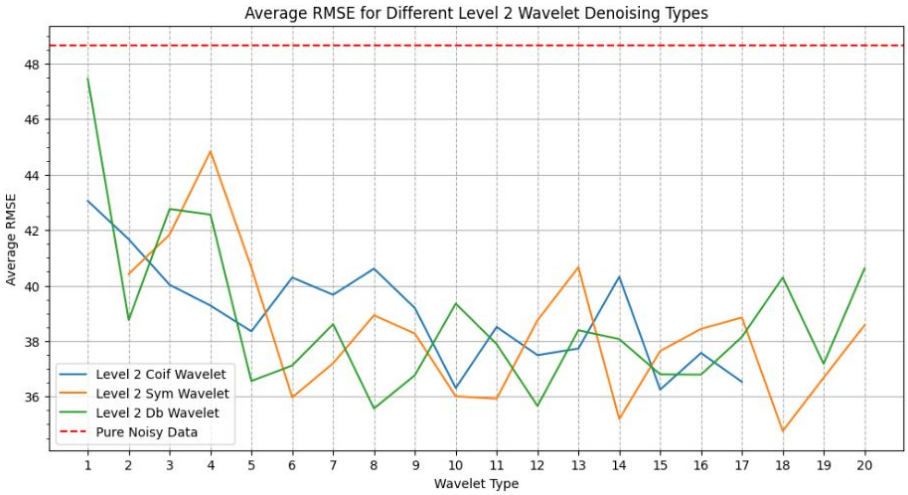

Fig. 10. RMSE Values for LSTM Models Trained on Different Kinds of Level 2 Denoised Data

In figure 10, we can see that all 3 wavelet families - db, sym and coif offer similar levels of performance at the same level of denoising with the performance being poor at low filter lengths and gradually increasing with

increase in filter length till a point after which no further meaningful change occurs.
Something to note here is that there are no peaks or troughs in the graph corresponding to filter lengths matching up with seasonality shown in the time series data. For example, one might expect there to be a minima in the graph near sym3 because it corresponds to a filter length of 6 and the input hourly data shows daily seasonality (at an interval of 7). No such minima is observed however.

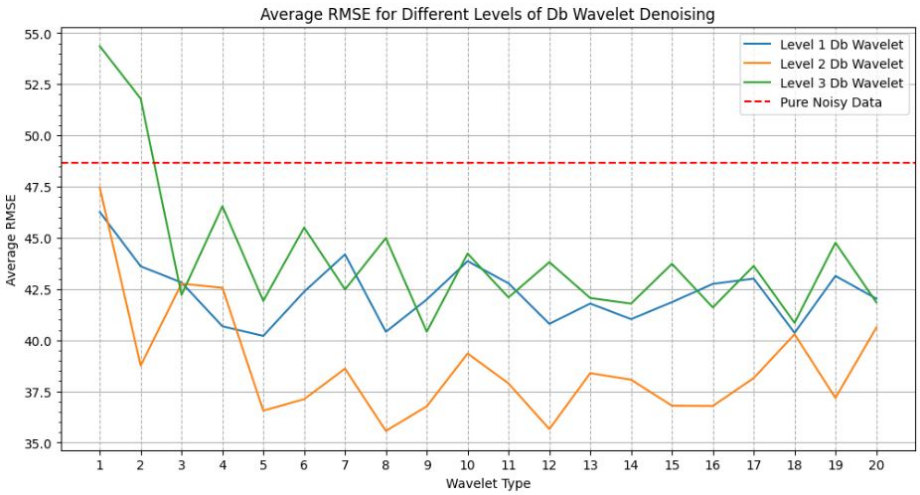

Fig. 11. RMSE Values for LSTM Models Trained on Different Levels of Db Denoised Data

In figure 11, we can see the performance for all 3 levels of denoising using the db wavelet family (which has the lowest average RMSE). Here, again we can see that for large filter lengths, level 2 denoising clearly offers the best performance.
In figure 12, we can see that the plot of the actual price vs the predicted price calculated by the LSTM trained on the original noisy data. From the shaded areas, we can see that the model tends to reduce in accuracy at local peaks and troughs.
In figure 13, we can see that although both the original LSTM (41.5% accuracy) and the denoised LSTM (48.5% accuracy) have similar accuracy levels, they do not predict accurately for the same data points. Only 20% of data points are predicted accurately by both with the overall spread being over 70% of the data. We can also see that the 2 different kinds of LSTM models seem to specialize in different sections of the data due to different levels of smoothening of the LSTM input data. For example, in this case, the original LSTM (blue) alone seems to do well in sections of the graph that are relatively

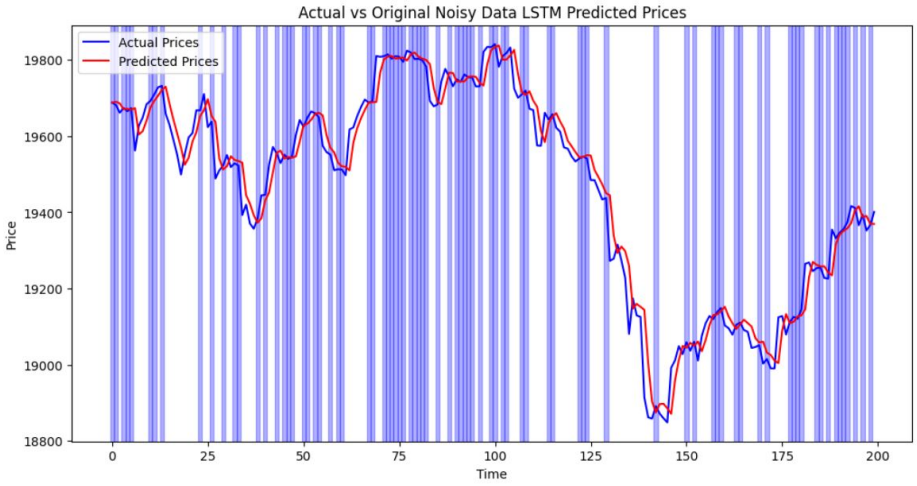

Fig. 12. Actual Prices vs Original Noisy Data LSTM Predicted Prices. Here the 41.5% of data points at which the predicted and actual price are within 0.1% of the other are shaded in Blue

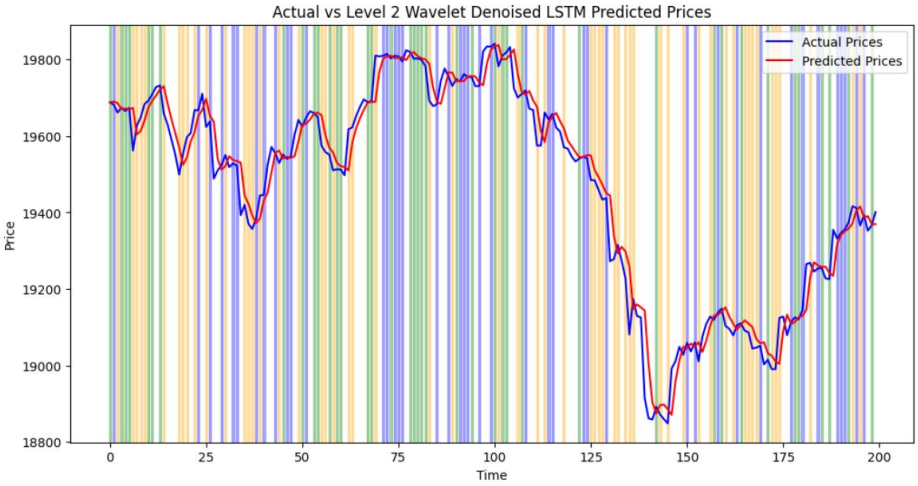

Fig. 13. Actual Prices vs Level 2 Denoised LSTM Predicted Prices. Here the 21.5% of data points at which only the original LSTM predicted price and the actual price are within 0.1% of the other are shaded in blue. The 28.5% of data points at which only the denoised LSTM predicted price and the actual price are within 0.1% of the other are shaded in orange. The 20% of data points at which the predicted price of both LSTMs lies within 0.1% of the actual price are shaded in green.

flat while the denoised LSTM (orange) alone seems to perform better at places where the price undergoes swift reversals.
As a result, the ideal model that could be designed in such a time series prediction task could be some sort of hybrid of the original LSTM and 1 or more denoised LSTMs. Such a hybrid model if well designed would likely yield better performance than any model used alone.

5 Conclusion

Through the research discussed in this article, we have comprehensively demonstrated the effects of different levels and kinds of wavelet denoising and proven that using wavelet transformation based denoising to denoise the input data for LSTM networks can cause a significant improvement in performance with roughly 20% reduction in RMSE in the case discussed in this paper.
When using wavelets to denoise and smoothen data, the level of decomposition and subsequent denoising seems to be the most significant factor affecting performance. The wavelet family used doesn't seem to have much influence, at least for db, sym and coif wavelets. The optimal filter length used also seems to only be determinable via parameter tuning and this value seems to be unaffected by any underlying seasonality in the data.
Further, we see from plots that although denoised LSTMs have better performance than regular "pure" LSTMs, the data points predicted correctly are generally not supersets of the data points predicted correctly by regular LSTMs with both kinds of LSTMs specializing in different sections of the input time series data with some common sections. As a result, some sort of well designed hybrid model that combines the predictive ability of pure and 1 or more denoised LSTM models may offer better performance than any of these models alone.

6 Disclosure of Interests.

The authors of this article have no competing interests to declare that are relevant to the content of this article.

References

1. Jinsong Leng: Modelling and Analysis on Noisy Financial Time Series. Edith Cowan University (2015)
2. A. A. Ariyo, A. O. Adewumi and C. K. Ayo, "Stock Price Prediction Using the ARIMA Model," 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, Cambridge, UK, 2014, pp. 106-112
3. Payal Soni, Yogya Tewari and Deepa Krishnan, "Machine Learning Approaches in Stock Price Prediction: A Systematic Review", Journal of Physics: Conference Series, Volume 2161, 1st International Conference on Artificial Intelligence, Computational Electronics and Communication System (AICECS 2021) 28-30 October 2021, Manipal, India

4. Achyut Ghosh, Soumik Bose, Giridhar Maji, Narayan C. Debnath, Soumya Sen, Stock Price Prediction Using LSTM on Indian Share Market, EPiC Series in Computing Volume 63, 2019, Pages 101–110, Proceedings of 32nd International Conference on Computer Applications in Industry and Engineering
5. D. Song, A. M. Chung Baek and N. Kim, "Forecasting Stock Market Indices Using Padding-Based Fourier Transform Denoising and Time Series Deep Learning Models," in *IEEE Access*, vol. 9, pp. 83786-83796, 2021
6. Torsten Kohler, Dirk Lorenz: A comparison of denoising methods for one dimensional time series, Pennsylvania State University
7. Gregory R. Lee, Ralf Gommers, Filip Wasilewski, Kai Wohlfahrt, Aaron O'Leary (2019). PyWavelets: A Python package for wavelet analysis. Journal of Open Source Software, 4(36), 1237
8. LNCS Homepage, http://www.springer.com/lncs