International Journal of Innovative Information Systems & Technology Research 14(4):217-226, Oct-Dec, 2026

© SEAHI PUBLICATIONS, 2026 www.seahipublications.org ISSN: 2467-8562

DOI:10.5281/zenodo.23629545

A Model for Anomaly Detection Using Hybrid Deep Neural Network And Long Short Term Autoencoder

Gogo Mbaja Olakayi[1], V. I. E. Anireh[2], O. E. Taylor[3]

ABSTRACT

Abnormal price movements in stock markets can signal manipulation, liquidity shocks, or emerging systemic risk, yet conventional statistical detectors and existing LSTM-Autoencoder systems lose accuracy under volatile conditions and rarely provide real-time alerting or rich reporting. This study develops a model for anomaly detection of stock prices using a hybrid Deep Neural Network (DNN) and LSTM-Autoencoder enhanced with volatility-aware and technical features. Daily Open, High, Low, Close, and Volume data from Yahoo Finance were cleaned, Min-Max normalised, enriched with technical, volatility, and rolling-statistical features, and converted into time-series sequences. The model learns normal market behaviour without labelled anomalies and flags observations whose reconstruction error exceeds a threshold. It was implemented in Python with TensorFlow/Keras, served through a FastAPI framework with real-time alerting, and backed by a PostgreSQL database and an interactive dashboard with downloadable reports. Compared with the existing LSTM-Autoencoder system, the proposed model achieved an accuracy of 97.60% versus 94.80%, precision of 96.90% versus 94.10%, recall of 95.80% versus 93.20%, an F1-score of 96.35% versus 93.65%, and a ROC-AUC of 98.40% versus 95.70%. False positive and false negative rates fell from 5.10% to 2.80% and from 6.80% to 4.20%, and average detection time fell from 1.84 s to 1.12 s. The findings indicate that combining DNN layers with LSTM-Autoencoder learning and volatility-aware features improves both the accuracy and the speed of stock anomaly detection, supporting its use for market surveillance and risk management.

Keywords: Anomaly Detection; Stock Prices; Deep Neural Network; LSTM-Autoencoder; Reconstruction Error; Volatility-Aware Features; Financial Time Series

1. INTRODUCTION

Anomaly detection is the process of identifying observations, events, or patterns that diverge markedly from expected behaviour [1, 2]. In financial markets it is applied to stock prices, trading volumes, and volatility shifts, where deviations may reflect macroeconomic announcements, liquidity shocks, algorithmic-trading distortions, market manipulation, or sudden changes in investor sentiment, as well as errors in data collection [3, 2]. Detecting them early supports risk management, market surveillance, and trading decisions for investors, regulators, and financial institutions, and it matters more as automated trading increases the speed at which disturbances such as flash crashes spread [4].

Statistical models such as ARIMA, GARCH, and linear regression have long been used for forecasting and anomaly detection in financial time series. They offer interpretability, but they typically assume stationarity, linearity, or simplified relationships among variables [5, 6], whereas real price series are non-stationary, show volatility clustering and structural breaks, and contain nonlinear interactions [7]. Deep neural networks (DNNs), including LSTM networks, autoencoders, convolutional networks, and hybrid models, can learn nonlinear mappings and long-range temporal dependencies directly from data [8, 9]. In

anomaly detection they are commonly trained on normal behaviour, and observations that the learned model cannot reproduce are flagged as anomalous [10].

A prominent deep-learning approach is the LSTM-Autoencoder, which combines sequence learning with reconstruction-error scoring [11]. It has three problems that motivate this study. First, it relies mainly on historical price attributes and lacks volatility-aware features and advanced feature engineering, so its accuracy falls during extreme volatility. Second, it offers no real-time processing or automatic notification of detected anomalies, which delays user responses. Third, its reporting is basic and its graphical visualisation limited, which makes results hard to interpret.

1.1 Aim and Objectives of the Study

The aim of this study is to develop a model for anomaly detection of stock prices using a Deep Neural Network. The objectives are to: (i) develop a hybrid DNN and LSTM-Autoencoder model enhanced with volatility-aware and technical features for accurate stock market anomaly detection; (ii) implement a FastAPI-based real-time anomaly detection and alerting framework that connects the user interface, the AI model, and the database; and (iii) design an interactive visualisation and reporting module that presents results through dashboards, charts, and downloadable reports. The study uses publicly available daily stock data (Open, High, Low, Close, and Volume) from Yahoo Finance and targets statistical and structural anomalies such as spikes, drops, and irregular trends; cryptocurrency analysis, macroeconomic forecasting, sentiment analysis, and portfolio optimisation are excluded.

2. LITERATURE REVIEW

2.1 Anomalies in Stock Prices

Stock prices are shaped by macroeconomic indicators, investor psychology, company announcements, and geopolitical events. Under the Efficient Market Hypothesis, prices reflect available information, so persistent abnormal behaviour should be rare [12]; the Adaptive Markets Hypothesis instead allows efficiency to vary with changing market conditions [13], which means that a detector must adapt as regimes change. Volatility clustering and fat-tailed returns are well-documented stylised facts that complicate the separation of noise from genuine anomalies [7, 14]. Anomalies in stock pricing are commonly grouped as statistical (extreme spikes or drops), temporal (unusual patterns over time), contextual (abnormal only in a given market context), collective (a group of observations that is abnormal together), and behavioural (arising from investor behaviour).

2.2 Machine-Learning and Deep-Learning Detectors

Machine-learning detectors are supervised, unsupervised, or semi-supervised [1]. Supervised methods need labelled anomalies, which are rare and costly to obtain in financial markets. Unsupervised and semi-supervised methods, including autoencoders, learn a representation of normal data and flag deviations, so they suit this setting [2]. An autoencoder compresses its input through a bottleneck and reconstructs it, so patterns unlike those seen in training reconstruct poorly [15]; LSTM networks add the ability to retain information over long sequences [9]. Deep learning for stock analysis has been shown to capture nonlinear, high-dimensional structure that classical methods miss [16], and a review of deep financial anomaly detection notes that most studies evaluate models under stable or moderately volatile conditions [17].

2.3 Related Work

Zhang and Zhou proposed an LSTM-Autoencoder for anomaly detection in S&P 500 stocks that outperformed statistical thresholding and ARIMA-based outlier detection, but reported that accuracy deteriorated in extreme turbulence [11]. Variational autoencoders improved detection of subtle anomalies in high-frequency data yet degraded under extreme volatility [18]. A hybrid LSTM-CNN outperformed standalone LSTM and CNN models in moderately volatile markets [19], and attention-based models improved focus on relevant temporal segments [20]. Graph neural networks captured inter-stock dependencies [21]; Transformer models captured long-range dependencies [22]; multi-scale LSTM-CNN-

attention models combined short-, medium-, and long-term patterns [23]; reinforcement-learning-enhanced models adapted anomaly weighting [24]; time-series GANs flagged deviations from generated normal patterns [25]; and an attention-based LSTM-CNN-DNN hybrid detected both subtle and extreme anomalies in moderate markets [26]. Across these studies, the authors report reduced performance during extreme volatility or sudden shocks.

2.4 Research Gap

The reviewed work leaves three gaps. First, models often struggle to detect anomalies during extreme volatility, such as financial crises or sudden shocks. Second, many studies depend on supervised learning, which requires labelled anomalies that are rare and difficult to label. Third, many studies use few input features and neglect interactions among multiple indicators such as Open, High, Low, Close, and Volume. This study addresses them with an unsupervised hybrid DNN and LSTM-Autoencoder enhanced with volatility-aware features, technical indicators, and rolling statistics; a FastAPI-based real-time alerting framework; and an interactive visualisation and reporting module.

3. METHODOLOGY

3.1 Research Design

A quantitative experimental design was adopted, because the study relies on numerical market data and computational techniques. A DNN-based model was built, trained on historical data to learn normal price behaviour, and tested on unseen data, and its results were compared with those of the existing system. The workflow has five stages: input (collection, preprocessing, and feature engineering of stock data), processing (design and training of the model in TensorFlow/Keras), output (anomaly scores and indicators), comparing (benchmarking against the existing system), and evaluation (accuracy, precision, recall, F1-score, and ROC-AUC).

3.2 Existing System and Its Limitations

The existing system is the LSTM-Autoencoder-based stock anomaly detection system of Zhang and Zhou [11]. Historical Open, High, Low, Close, and Volume data are cleaned, normalised, and passed to an LSTM-Autoencoder whose reconstruction error is compared with a predefined threshold; results are stored in PostgreSQL and shown on a basic dashboard. Its main limitations are: poor performance under extreme volatility; no volatility-aware features (such as historical volatility or Bollinger Bands); limited feature engineering; inability to capture short- and long-term patterns simultaneously; no real-time alerting; limited visualisation and reporting; no API layer for integration; limited scalability and adaptability to new market regimes; and a higher likelihood of false positives and false negatives because detection rests on a reconstruction-error threshold alone.

3.3 Proposed System and Requirements

The proposed system is a data-driven, unsupervised deep-learning approach that learns normal stock behaviour from multivariate data without labelled anomalies. Users (analysts, investors, financial institutions, researchers, and administrators) must be able to retrieve historical data from the Yahoo Finance API, monitor behaviour in real time, receive instant notifications of anomalies, view interactive charts, download reports, and access historical records. Functionally, the system acquires, preprocesses, and engineers features from stock data; stores data, scores, and reports in PostgreSQL; classifies each observation as Normal or Anomalous from its reconstruction error and learned features; raises real-time alerts; and provides visualisation, reporting, FastAPI-based communication between the interface, model, and database, and user authentication. Non-functional requirements cover performance, accuracy, reliability, scalability, security, usability, maintainability, availability, portability, interoperability, and robustness under extreme volatility.

3.4 Dataset and Feature Engineering

The system uses historical daily trading records obtained from the Yahoo Finance API (yfinance): Date, Open, High, Low, Close, and Volume (OHLCV). Preprocessing removes duplicates and missing or

invalid values, applies Min-Max normalisation, and transforms the data into time-series sequences with sliding windows. To capture market volatility, momentum, and statistical behaviour, the system derives additional features (Table 1). The core input vector is X = (Open, High, Low, Close, Volume, MA, RSI, MACD, Volatility), where MA is a five-period moving average of closing prices; the exponential moving average, signal line, Bollinger Bands, rolling mean, rolling standard deviation, and daily return are also derived by the feature-engineering module [27].

Table 1. Input feature categories.

Category	Input variables
Price	Open, High, Low, Close
Trading	Volume
Trend indicators	Moving Average (MA), Exponential Moving Average (EMA)
Momentum indicators	RSI, MACD, Signal Line
Volatility indicators	Bollinger Bands, Rolling Standard Deviation, Volatility Index
Statistical features	Rolling Mean, Daily Return

Features are Min-Max normalised so that variables of different scale, such as prices, volume, RSI, and MACD, are comparable:

$$X' = (X - X_{min}) / (X_{max} - X_{min}) \quad (1)$$

2. 3.5 Mathematical Model

The temporal component is an LSTM whose gates at time step t are:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (2)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (3)$$

$$\tilde{C}_t = tanh(W_c[h_{t-1}, x_t] + b_c) \quad (4)$$

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t \quad (5)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (6)$$

$$h_t = o_t tanh(C_t) \quad (7)$$

where f_t, i_t, and o_t are the forget, input, and output gates, C_t is the memory cell, h_t is the hidden state, σ is the sigmoid function, and W and b are weights and biases. The autoencoder encodes the input into a latent representation z and decodes it to a reconstruction:

$$z = f(W_e x + b_e) \quad (8)$$

$$\hat{x} = f(W_d z + b_d) \quad (9)$$

The reconstruction error is the mean squared error between the input and its reconstruction, and an observation is flagged as anomalous when it exceeds a threshold T:

$$RE = (1/n) \Sigma_{i=1}^{n} (x_i - \hat{x}_i)^2 \quad (10)$$

$$A_t = 1 \text{ if } RE_t > T;\ A_t = 0 \text{ if } RE_t \leq T \quad (11)$$

DNN layers add hierarchical feature extraction and anomaly scoring on top of the latent representation, $y = f(Wx + b)$, with binary cross-entropy used where a supervised output is required. Because the latent representation is highly compressed, abnormal patterns are difficult to reconstruct accurately and therefore produce large reconstruction errors.

3.6 System Architecture

The system follows a modular pipeline architecture (Fig. 1) in which data flow sequentially from stock data in the Yahoo Finance API to data ingestion and preprocessing (collection, cleaning, missing-value handling, normalisation, feature engineering, technical and volatility indicators, and rolling statistics); a FastAPI server that receives requests, validates data, and communicates with the model; the hybrid DNN and LSTM-Autoencoder model, which extracts temporal features, learns complex patterns, reconstructs the input, and computes the error; an analysis module that computes anomaly scores and classifies observations as normal or anomalous; a PostgreSQL database that stores stock data, predictions, scores, alerts, reports, and historical records; a real-time alerting module that notifies users by dashboard, email, or other channels; a visualisation and reporting module with interactive charts, trend analysis, anomaly reports, and downloadable reports; and a client dashboard for real-time monitoring and report generation. A model-comparison module benchmarks results against the existing LSTM-Autoencoder system.

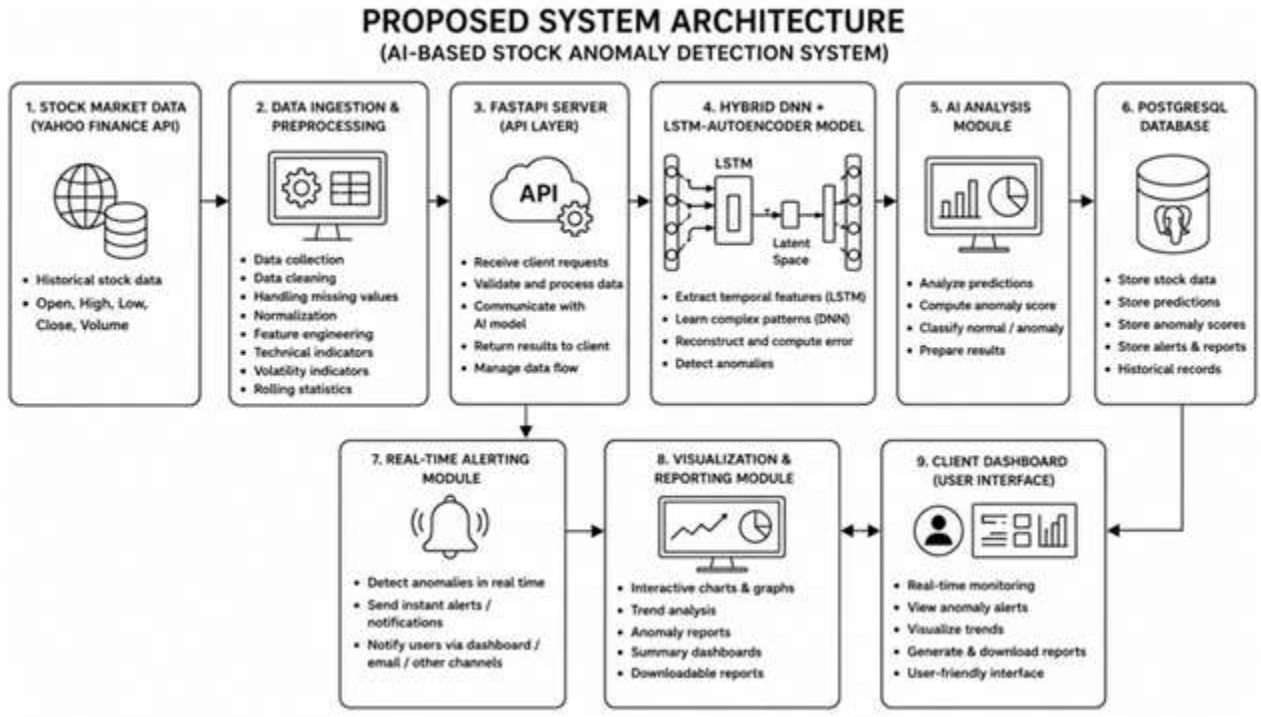

Figure 1. Architecture of the proposed AI-based stock anomaly detection system.

3.7 Algorithm

Algorithm 1 summarises the training and detection procedure.

Algorithm 1: Hybrid DNN and LSTM-Autoencoder-Based Stock Anomaly Detection

Input: historical stock data (Open, High, Low, Close, Volume) from the Yahoo Finance API
Output: Normal/Anomaly label, anomaly score, real-time alert, dashboards and reports

```
Collect stock market data from the Yahoo Finance API
Preprocess: remove missing or invalid values; normalise; build time-series sequences
Engineer features: technical indicators (RSI, MACD, MA), volatility indicators, rolling statistics
Store the processed dataset
Load the hybrid DNN and LSTM-Autoencoder model
Train the model on normal historical data
Feed the processed data into the trained model
Generate reconstructed outputs with the autoencoder
Compute the reconstruction error and the anomaly score
if anomaly score > threshold then
  classify the observation as Anomaly
  store the result in PostgreSQL
  trigger the real-time alerting module
  generate dashboards, charts, and reports
else
  classify the observation as Normal; store the result
end if
display results on the client dashboard
```

3.8 Implementation and Evaluation Set-up

The system was implemented in Python 3.11 with TensorFlow/Keras for the model, Scikit-learn for preprocessing and metrics, NumPy and Pandas for data handling, Matplotlib and Plotly for visualisation, FastAPI for the REST API, and PostgreSQL with SQLAlchemy for data management. The model was trained with the Adam optimiser and a mean-squared-error loss using the settings in Table 2, with 80% of the data used for training and 20% reserved for validation and testing. Unit tests of the ingestion, preprocessing, autoencoder, detection, and visualisation modules, and integration tests of the web interface, FastAPI backend, PostgreSQL database, and model engine, all passed.

Table 2. Model training configuration.

Table 2. Model training configuration.

Parameter	Value	Parameter	Value
Model	Hybrid DNN + LSTM-AE	Optimiser	Adam
Loss function	MSE	Learning rate	0.001
Batch size	32	Epochs	100
Data split	80% / 20%	Activations	ReLU, Sigmoid
Framework	TensorFlow / Keras	Output	Reconstruction error

Performance was measured with accuracy, precision, recall, F1-score, ROC-AUC, false positive rate (FPR), false negative rate (FNR), average detection time, and reconstruction error, where accuracy = (TP + TN)/(TP + TN + FP + FN), precision = TP/(TP + FP), recall = TP/(TP + FN), and F1 = 2 × precision × recall/(precision + recall). The proposed system was benchmarked against the existing LSTM-Autoencoder system on the same evaluation data.

4.1 Training Behaviour

Training and validation loss decreased steadily over the 100 epochs, from 0.1185 and 0.1237 at epoch 1 to 0.0079 and 0.0091 at epoch 100 (Table 3, Fig. 2). The two curves stay close together throughout, with a gap of 0.0012 at epoch 100, which indicates that the model learned the normal characteristics of the data without significant overfitting.

Table 3. Training and validation loss.

Epoch	Training loss	Validation loss
1	0.1185	0.1237
10	0.0652	0.0698
20	0.0416	0.0452
40	0.0243	0.0271
60	0.0168	0.0189
80	0.0105	0.0123
100	0.0079	0.0091

Figure 2. Training and validation loss of the hybrid DNN and LSTM-Autoencoder.

4.2 Comparative Detection Performance

Table 4 and Fig. 3 compare the proposed system with the existing LSTM-Autoencoder system. The proposed model achieved an accuracy of 97.60% against 94.80%, so it classified observations more accurately. Its higher precision (96.90% against 94.10%) means fewer false alarms, and its higher recall (95.80% against 93.20%) means it identified a greater proportion of actual anomalies. The F1-score (96.35% against 93.65%) shows a better balance between the two, and the ROC-AUC (98.40% against 95.70%) shows stronger separation of normal behaviour from anomalies. The false positive rate fell from 5.10% to 2.80%, a relative reduction of about 45%, and the false negative rate fell from 6.80% to 4.20%, about 38%. Average detection time fell from 1.84 s to 1.12 s, which is 39.1% faster and makes the system more suitable for real-time monitoring.

Table 4. Comparative performance of the existing and proposed systems.

Metric	Existing	Proposed	Change
Accuracy	94.80%	97.60%	+2.80 pp
Precision	94.10%	96.90%	+2.80 pp
Recall	93.20%	95.80%	+2.60 pp
F1-score	93.65%	96.35%	+2.70 pp
ROC-AUC	95.70%	98.40%	+2.70 pp
False positive rate	5.10%	2.80%	−2.30 pp
False negative rate	6.80%	4.20%	−2.60 pp

Metric	Existing	Proposed	Change
Detection time	1.84 s	1.12 s	39.1% faster

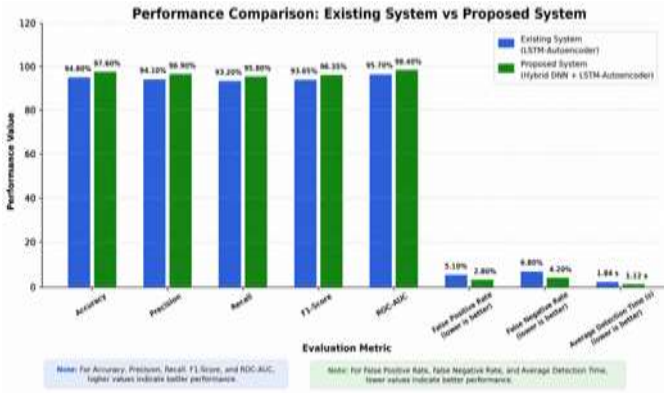

Figure 3. Performance comparison of the existing and proposed systems.

4.3 Reconstruction Error

The proposed model produced lower reconstruction errors on normal observations and a clearer separation between normal and anomalous data (Table 5). The mean reconstruction error fell from 0.0048 to 0.0028, about 42% lower, and the maximum from 0.0245 to 0.0179. The detection threshold was 0.0065 for the proposed model against 0.0080 for the existing one, so the better separation allowed a more reliable threshold, which reduced false detections.

Table 5. Reconstruction error comparison.

Parameter	Existing	Proposed
Mean reconstruction error	0.0048	0.0028
Maximum reconstruction error	0.0245	0.0179
Detection threshold	0.0080	0.0065

4.4 Capability Comparison

Beyond detection performance, the proposed system adds capabilities that the existing system lacks (Table 6): technical and volatility indicators, rolling statistics, DNN layers, real-time alerting, a visualisation dashboard, and FastAPI integration. Both use an LSTM-Autoencoder, and the existing system has only limited PostgreSQL support.

Table 6. Feature comparison.

Feature	Existing	Proposed
OHLCV data	Yes	Yes
Technical indicators	No	Yes
Volatility indicators	No	Yes
Rolling statistics	No	Yes
DNN layers	No	Yes
LSTM-Autoencoder	Yes	Yes
Real-time alerting	No	Yes
Visualisation dashboard	No	Yes
FastAPI integration	No	Yes
PostgreSQL database	Limited	Yes

4.5 DISCUSSION

The three objectives were met. A hybrid DNN and LSTM-Autoencoder with volatility-aware and technical features was developed and outperformed the existing LSTM-Autoencoder on every metric in Table 4. A FastAPI-based framework connects the interface, model, and database and supports real-time alerting. A visualisation and reporting module presents anomaly scores, reconstruction errors, trends, and downloadable reports. The pattern of results is consistent with the rationale of the design: the added indicators give the model information about volatility and momentum that price levels alone do not carry, and the DNN layers add hierarchical feature extraction to the temporal learning of the LSTM.

Several limitations qualify these results. The data come from a single public source, Yahoo Finance, and the performance on other markets or asset classes has not been validated. The results are not broken down by volatility regime, so the improvement under extreme volatility, which motivated the design, is not isolated from the overall improvement. No confidence intervals or significance tests are reported for the differences of about 2.6 to 2.8 percentage points. A fixed reconstruction-error threshold is used, and, as market behaviour changes over time, the model requires periodic retraining. Finally, training and deployment need considerable computational resources.

5. CONCLUSIONS

This study developed and evaluated a hybrid Deep Neural Network and LSTM-Autoencoder system for detecting abnormal stock price movements. By adding DNN layers, volatility-aware indicators, technical features, and rolling statistics to an LSTM-Autoencoder, and by providing FastAPI-based real-time alerting with an interactive dashboard and reports, the system addresses the main limitations of the existing approach. It achieved 97.60% accuracy, 96.90% precision, 95.80% recall, a 96.35% F1-score, and a 98.40% ROC-AUC, with a false positive rate of 2.80%, a false negative rate of 4.20%, and an average detection time of 1.12 s, outperforming the existing LSTM-Autoencoder system on all of them.

The study contributes a hybrid DNN and LSTM-Autoencoder framework that combines hierarchical nonlinear feature extraction with temporal sequence learning; an integrated feature-engineering and reconstruction-error detection pipeline; and an end-to-end monitoring framework with data acquisition, detection, alerting, visualisation, reporting, and database management. The system can support investors, analysts, regulators, and financial technology organisations in market surveillance and risk management.

Future work should integrate real-time market data feeds; incorporate macroeconomic indicators, news, and sentiment; investigate Transformer, graph, bidirectional LSTM, and attention architectures; replace the fixed threshold with adaptive thresholding; add explainability methods such as SHAP and LIME; extend the framework to other assets such as cryptocurrencies, foreign exchange, and commodities; deploy it on cloud platforms with email, SMS, and messaging alerts; and validate it on larger datasets from several exchanges across different economic cycles, including an evaluation by volatility regime.

REFERENCES

[1] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58. https://doi.org/10.1145/1541880.1541882

[2] Aggarwal, C. C. (2017). Outlier analysis (2nd ed.). Springer.

[3] Chan, N. T., & Wang, Y. (2020). Financial anomaly detection using machine learning: A survey. Quantitative Finance, 20(12), 2115–2133.

[4] Kirilenko, A., Kyle, A. S., Samadi, M., & Tuzun, T. (2017). The flash crash: High-frequency trading in an electronic market. Journal of Finance, 72(3), 967–998. https://doi.org/10.1111/jofi.12545

[5] Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time series analysis: Forecasting and control (5th ed.). Wiley.

[6] Tsay, R. S. (2010). Analysis of financial time series (3rd ed.). Wiley.

[7] Cont, R. (2001). Empirical properties of asset returns: Stylised facts and statistical issues. Quantitative Finance, 1(2), 223–236. https://doi.org/10.1080/713665670
[8] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.
[9] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.
[10] Pimentel, M. A. F., Clifton, D. A., Clifton, L., & Tarassenko, L. (2014). A review of novelty detection. Signal Processing, 99, 215–249.
[11] Zhang, Y., & Zhou, X. (2021). LSTM-Autoencoder–based financial anomaly detection for S&P 500 stocks. Journal of Financial Data Science, 3(4), 25–40.
[12] Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. Journal of Finance, 25(2), 383–417.
[13] Lo, A. W. (2004). The adaptive markets hypothesis: Market efficiency from an evolutionary perspective. Journal of Portfolio Management, 30(5), 15–29.
[14] Bollerslev, T. (1986). Generalised autoregressive conditional heteroskedasticity. Journal of Econometrics, 31(3), 307–327.
[15] Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. Science, 313(5786), 504–507.
[16] Chong, E., Han, C., & Park, F. C. (2017). Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Systems with Applications, 83, 187–205.
[17] Li, Z., & Li, J. (2020). Deep learning in financial anomaly detection: A review of recent advancements. IEEE Access, 8, 193347–193362.
[18] Hawkins, D., Patel, R., & Singh, A. (2022). Variational autoencoder models for high-frequency financial anomaly detection. Journal of Financial Technology, 5(2), 112–128.
[19] Patel, R., & Kumar, S. (2022). Hybrid LSTM-CNN approach for anomaly detection in stock markets. Journal of Computational Finance, 12(4), 145–162.
[20] Li, X., Zhang, Y., & Chen, H. (2023). Attention-based deep learning for anomaly detection in stock price series. Journal of Financial Computing, 6(1), 55–72.
[21] Zhou, Y., & Chen, H. (2023). Graph neural networks for anomaly detection in stock price series. Journal of Financial Data Science, 6(2), 77–95.
[22] Sun, Q., & Liu, J. (2024). Transformer-based deep learning for anomaly detection in stock markets. International Journal of Financial Computing, 7(1), 33–52.
[23] Huang, L., Zhang, X., & Li, M. (2024). Multi-scale LSTM-CNN-attention models for anomaly detection in stock price series. Journal of Computational Finance, 14(1), 77–98.
[24] Chen, Y., Liu, H., & Wang, J. (2024). Reinforcement learning-enhanced deep learning for anomaly detection in stock markets. Journal of Financial Technology, 8(1), 66–88.
[25] Liu, X., & Zhang, Y. (2024). Time-series generative adversarial networks for anomaly detection in stock markets. Journal of Financial Computing, 8(2), 101–122.
[26] Yang, F., & Zhao, L. (2024). Attention-based hybrid LSTM-CNN-DNN models for financial anomaly detection. Journal of Financial Technology, 8(2), 121–142.
[27] Bollinger, J. (2002). Bollinger on Bollinger Bands. McGraw-