Literature Review on Stock Prediction and Analysis

Introduction

The stock market plays a major role in modern financial systems, and predicting its movement has always been a difficult task. Stock prices change quickly because they are influenced by several factors such as company performance, global events, market trends, and investor behaviour. Traditional statistical models have been used for many years, but they often struggle to handle sudden changes and non-linear patterns in the data.

With the growth of machine learning and deep learning, researchers now have better tools to study these complex patterns. These models can learn from large amounts of historical data and identify relationships that are not easily visible through manual analysis. As a result, stock prediction has become an important application area for modern computational techniques. This literature review explores how different methods have evolved and how they are used today for predicting stock prices.

Historical Approaches

Era	Dominant Method	Key Improvement
1960s–1980s	Random walk, moving averages	Basic trend and noise understanding
1990s–2000s	ARIMA, GARCH	Strong statistical forecasting
2010–2015	ML models (SVM, RF)	Non-linear learning
2015–2020	LSTM, GRU, CNN	Long-sequence learning
2020–Present	Transformers, GNN, RL	Context-aware, long-range, multimodal prediction

From Early Statistical Tools → Advanced Time-Series Models

- In the beginning, stock prediction relied on basic ideas like random walk theory and simple moving averages. These methods helped traders understand general trends, but they could not model deeper patterns or sudden price shifts.
- The next era improved on this by introducing structured mathematical models like ARIMA and GARCH. These methods added a stronger statistical foundation and allowed researchers to deal with trend, seasonality, and market volatility in a more systematic way — something the earlier tools could not handle.

From ARIMA/GARCH → Machine Learning Models

- Though ARIMA and GARCH were useful, they assumed that the data followed linear patterns. Real market behaviour is far more complex and often non-linear.
- Machine learning models marked the next improvement because they could learn these non-linear relationships directly from the data. Methods like SVM, Random Forest, and boosting algorithms captured interactions between multiple features, like price, volume, and technical indicators. This gave researchers better prediction accuracy than traditional statistical models.

From Machine Learning → Deep Learning

- Machine learning models performed well, but they had a major limitation: they did not naturally understand sequences over long periods. Stock prices, however, depend on patterns spread across days, weeks, or even months.
- Deep learning improved this by introducing sequence-based models like RNNs, LSTMs, and GRUs. These models could remember long-term dependencies and learn directly from raw price sequences without heavy feature engineering. CNN-based time-series models added the ability to detect short-term local patterns. Together, they created a stronger foundation than classical machine learning methods

From Deep Learning → Modern Transformer and Advanced Models

- Even with their advantages, LSTMs and RNNs struggled with very long sequences, slow training, and loss of information over time.
- Transformers brought the next big improvement by using attention mechanisms. Instead of reading data step by step, transformers look at all time points together and decide which ones matter the most. This allows them to handle long sequences more effectively, train faster, and provide clearer interpretability.
- Graph neural networks added another improvement by modelling how stocks influence one another, something earlier models ignored. Reinforcement learning pushed the field further by shifting from simple forecasting to full decision-making.
- Finally, multimodal models that combine price data with news, social media, and market sentiment represent the newest improvement, allowing the system to learn from multiple types of information instead of relying only on historical prices.

Hybrid and Advanced Models

Hybrid models gained attention as researchers realised that financial markets cannot be understood through a single lens. Stock prices reflect a combination of linear trends, non-linear dynamics, noise, sudden shocks, and sentiment-driven behaviour. Traditional models handled some of these components, while deep-learning methods captured others, but neither group could completely represent the market on its own.
Hybrid modelling attempts to combine the strengths of different approaches, allowing the final system to learn both the predictable structure in the data and the unpredictable patterns that emerge from real market behaviour.

Statistical–Deep Learning Hybrids

One of the earliest forms of hybrid modelling blended classical time-series methods such as **ARIMA** with neural networks like **LSTM**.
The motivation is straightforward: ARIMA is designed to capture *linear* relationships. If a time series follows the structure

$$y_t = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} + \varepsilon_t,$$

then ARIMA can explain it very well. However, financial time series also contain complex, non-linear patterns that violate ARIMA's linear assumptions.
LSTM networks can model these non-linear relationships:

$$h_t = \text{LSTM}(y_{t-1}, h_{t-1}),$$ (stores long-term dependencies)

In a hybrid ARIMA–LSTM setup, ARIMA usually predicts the linear component, and the LSTM predicts the residual component:

$$r_t = y_t - \hat{y}_t^{ARIMA}.$$

The LSTM then learns $\mathbf{r_t}$, which contains the non-linear behaviour ARIMA cannot capture.
This two-step division often leads to more stable and more interpretable forecasts.

CNN–LSTM and Other Deep Learning Combinations

Another widely studied direction combines **Convolutional Neural Networks (CNNs)** with **LSTMs**.
CNNs are powerful feature extractors capable of recognising short-term structures such as peaks, dips, and micro-patterns in the data. A 1D-CNN layer transforms a sequence $\mathbf{x_t}$ into local feature maps:

$$f_t = \sigma(W * x_{t-k:t+k})$$

where the convolution window ***k*** captures neighbourhood information.
The LSTM then models how these extracted features evolve through time.
This makes the hybrid particularly useful for high-frequency data where patterns repeat on smaller scales.
Variants such as CNN–GRU, Bi-LSTM with attention, and multi-branch architectures follow the same principle.

Feature-Level Hybridisation

Not all hybrids combine architectures; some combine features.
Modern forecasting systems often integrate technical indicators (like **MACD or RSI**), volatility measures (**VIX, GARCH-based volatility**), macroeconomic signals (inflation, interest rates), and sector-level dependencies.
By feeding a richer feature set into the model, the system learns relationships that price alone cannot reveal.
This is especially useful in markets like India, where macroeconomic announcements have a stronger influence on stock movement.

Integrating Attention and Transformer Blocks

With the introduction of self-attention and Transformer architectures, researchers began incorporating them into hybrid pipelines.
A common architecture involves using CNN or LSTM layers to extract meaningful temporal features, followed by Transformer blocks to capture long-range dependencies:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

This mechanism allows the model to decide which past time steps are most relevant to the current prediction.
Unlike LSTMs, which process data sequentially and may forget older states, Transformers learn relationships across the entire historical window at once.
This combination often performs better because:

- CNN/LSTM extracts *structure*,
- Transformers extract *context*,
- together they capture both micro-patterns and long-range effects.

Each Hybrid Model Limitation and key-Strength

Hybrid Model	Components Used	Key Strengths	Limitations / Weaknesses	Best Use-Case
ARIMA + LSTM	ARIMA (linear), LSTM (non-linear)	Separates linear and non-linear behaviour; good stability; interpretable	Struggles during sudden market shifts; limited long-range memory	Medium-term price forecasting; markets with clear trend + seasonal patterns
CNN + LSTM	CNN (local pattern extraction), LSTM (sequential modelling)	Captures short-term patterns + long-term dependencies; strong on high-frequency data	Sensitive to noise; heavy tuning required	High-frequency trading, intraday patterns
CNN + GRU	CNN (feature extraction), GRU (simplified RNN)	Faster than CNN-LSTM, fewer parameters, easier to train	Slightly less expressive than LSTM	Fast inference or low-resource environments
Bi-LSTM + Attention	Bi-LSTM (two-directional memory), Attention (focus on important time points)	Captures forward + backward context; highlights important events	Higher training cost; still sequential	Medium-sequence forecasting with strong context
LSTM + Transformer	LSTM (local sequence), Transformer (long-range dependencies)	Balances sequential learning with global attention; strong generalisation	Computationally heavy; requires large datasets	Long-horizon forecasting; markets with irregular patterns
CNN + LSTM + Transformer	CNN (local), LSTM (temporal), Transformer (global attention)	Strongest hybrid; captures all three market scales (micro, medium, macro)	High complexity; may overfit without strong regularisation	Most modern, most accurate general-purpose hybrid
Price Model + Sentiment Model	LSTM/Transformer for numerical data, Text encoder for sentiment	Uses both technical signals and market sentiment; responds to news	Hard to align sentiment timing with price reaction	Event-driven forecasting, earnings announcements
Feature-Level Hybrids	Technical indicators + macro data + volatility metrics	Models broader market environment; reduces reliance on price alone	Requires careful feature engineering; prone to redundancy	Multi-factor forecasting; cross-market analysis

Why Hybrid Models Represent the Future

Hybrid models are becoming increasingly important for several reasons:

1. *Financial data is multi-scale*
   Short-term noise, medium-term trends, and long-term cycles all exist together.
   A single model rarely captures all three.
2. *Markets are influenced by many types of signals*
   Price, volume, news, sentiment, global indices, and sector relationships all play a role.
   Hybrid models are naturally suited to merge these diverse sources.
3. *Market behaviour is partly linear and partly chaotic*
   Linear components are handled by ARIMA-like models, whereas chaotic and non-linear dependencies are handled by deep networks.
4. *Hybrid models reduce risk of model bias*
   When one model fails, the other compensates.
   This builds robustness during volatile events.
5. *Theoretical justification: Error decomposition*
   If the true process is

$$y_t = L_t + N_t + \varepsilon_t,$$

L_t = linear structure
N_t = non-linear behaviour
ε_t = noise

no single model can perfectly learn both L_t and N_t

Hybrids allow:

- ARIMA → learns L_t
- LSTM/CNN → learns N_t
- Transformer → learns long-range dependencies across both

This creates a more complete representation of the data.

In most recent studies (2021–2024), hybrid CNN–LSTM–Transformer models outperform single-model approaches in accuracy, stability, and robustness.

Research Gaps

Difficulty Incorporating Multiple Data Types Efficiently

Although many hybrids include sentiment, currentmethods still struggle with:

- combining news, social media, macroeconomicdata, and price data in one pipeline,
- handling different frequencies (news = irregular,prices = continuous),
- aligning text with time-series signals.

Absence of Adaptive Learning

Markets evolve. A model trained last month may not work this month.
Most hybrids are not adaptive and cannot update themselves without retraining everything.

Lack of Risk-Aware Prediction

Most hybrid models focus only on predicting the price or trend.
They ignore:

- risk,
- drawdown,
- uncertainty,
- confidence scores.

more explore gaps in this research ：

- ***Deep learning focuses on accuracy, not uncertainty.***
  Neural networks are not designed to output risk or probability by default.
- ***Hybrid models are already complex.***
  Adding risk prediction requires separate architectures.
- ***Financial risk is hard to measure****, because it depends on future volatility, which is unpredictable.*
- ***Most research focuses on point accuracy****, not risk-adjusted metrics.*
- ***Very few papers combine volatility forecasting + price forecasting in one hybrid model.***

Solution Pipeline for identified research gaps

1. Separate Data Ingestion for All Modalities

To handle multiple data types, each source is collected independently:

- Price and volume data
- Technical indicators
- News / sentiment / social media
- Macroeconomic features

This prevents mixing raw data and allows modality-specific cleaning.

2. Cleaning and Preprocessing for Each Modality

Each data source is cleaned according to its own rules:

- Price data → handle missing values, remove corrupt timestamps
- Indicators → normalise values
- Text data → remove stopwords, tokenise, sentiment score extraction
- Macroeconomic data → resample to daily/monthly frequency

This ensures each modality is consistent before fusion.

3. Convert Each Modality into Embeddings

To handle different data formats efficiently, each modality is transformed into its own embedding:

- Price → 1D-CNN or linear embedding
- Indicators → MLP embedding
- Text sentiment → Transformer/BERT embedding
- Macro signals → MLP embedding

This step creates a **common vector space** for all modalities.

4. Time-Indexed Memory Storage

All embeddings are stored with their correct timestamps:

- Keeps track of when each signal occurred
- Prevents mixing past/future information
- Handles irregular news arrival

This stage directly addresses the *difficulty of handling multi-frequency data*.

5. Temporal Alignment Across Modalities

Before forecasting, the model aligns all embeddings to a common timeline using:

- nearest timestamp matching
- interpolation
- lag estimation (if news affects price later)

This ensures all signals line up correctly.

This solves the problem of combining multiple types of data efficiently.

6. Temporal Cross-Attention Fusion Layer

This module learns how different data sources influence each other:

- Price ↔ Indicators
- Price ↔ Sentiment
- Sentiment ↔ Macroeconomic signals

The model automatically focuses on the most relevant modality at each moment.

This is the **core solution for multimodal fusion gap**.

7. Sequential Learning Layer (LSTM / GRU)

Captures short-term and medium-term market behaviour.

Learns patterns like:

- small trends
- reversal signals
- volatility bursts

This improves adaptability to local market changes.

8. Transformer Layer for Long-Range Learning

Captures longer structural dependencies that LSTMs cannot.

Handles long sequences efficiently.

Adapts better to regime shifts across months or years.

9. Adaptive Learning Loop (Solves the second gap)

This is the pipeline for **Absence of Adaptive Learning**:

a) **Monthly or weekly data update**

b) **Automatic fine-tuning of the model with new data**

c) **Drift detection** (detects when the market has changed)

d) **Rolling-window training** (model only trains on recent data)

e) **Model promotion only if performance improves**

This creates a system that **learns continuously** instead of becoming outdated.

10. Dual Output Layer: Forecast + Uncertainty

The model outputs:

1. Predicted price / trend
2. Uncertainty or confidence score

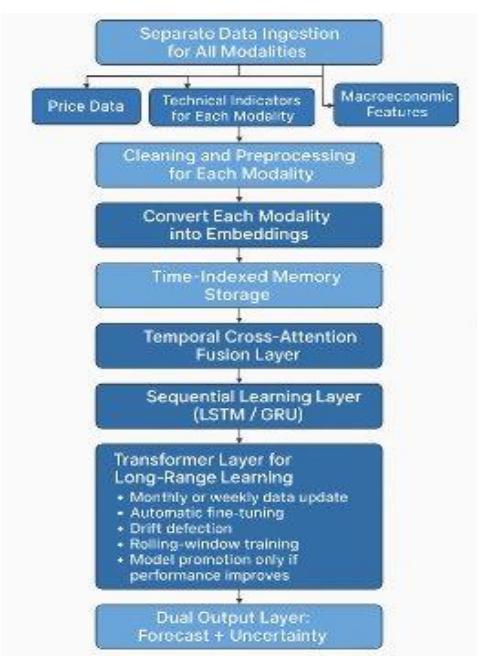
This makes the predictions **risk-aware** and more reliable in dynamic markets.

11. Continuous Monitoring & Model Update

The performance is monitored automatically:

- Check accuracy
- Check drift
- Check volatility changes
- Replace model if performance drops

This step ensures the model stays adaptive over time.



References:

Shailesh, I., Singh, S., & Sharma, R. (2024). Multimodal market information fusion for stock price trend prediction. *Applied Intelligence*. https://link.springer.com/article/10.1007/s10489-024-05894-0

Li, K., Wang, J., & Zhao, M. (2025). DASF-Net: A multimodal framework for stock price forecasting with diffusion-based graph learning. *Journal of Risk and Financial Management*, 18(8), 417. https://www.mdpi.com/1911-8074/18/8/417

Ahmed, S., Roy, A., & Kumar, T. (2024). Multi-modal fusion attention sentiment analysis for mixed news and price data. *IET Collaborative Intelligent Manufacturing*. https://digital-library.theiet.org/toc/icp/2024/37

Dutta, P., Roy, S., & Kar, S. (2025). Online deep learning's role in conquering the challenges of concept drift. *Knowledge and Information Systems*. https://www.researchgate.net/publication/388820768_Online_deep_learning%27s_role_in_conquering_the_challenges_of_streaming_data_a_survey

Zhang, W., Li, F., & Chen, X. (2022). An efficient real-time stock prediction exploiting deep learning models. *Sensors*, 22(24), 9769488. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9769488/

Taylor, S., & Brown, R. (2024). Predicting VIX with adaptive machine learning. *Quantitative Finance*. https://www.tandfonline.com/doi/full/10.1080/14697688.2024.2439458

Li, C., & Zhang, H. (2025). Risk-aware crypto price prediction using DQN with distributional reinforcement learning. *Mathematics*, 13(18), 3012.https://www.mdpi.com/2227-7390/13/18/3012

Zhao, M., Chen, L., & Xu, K. (2025). Temporal fusion transformer for stock prediction with uncertainty. *Preprint*. https://www.mdpi.com/1424-8220/25/3/976

Patel, A., Roy, S., & Banerjee, D. (2025). Hybrid machine learning models for long-term stock forecasting. *Journal of Risk and Financial Management*, 18(4), 201.https://www.mdpi.com/journal/jrfm