International Journal of Web of Multidisciplinary Studies
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
website: *https://ijwos.com*
Volume 03, Issue 11,November 2026
OPEN ACCESS
E-ISSN : 3049-2424
DOI: https://doi.org/10.71366/ijwos

Real-Time Financial Microstructure Ingestion and Sequence Modeling for Stock Market Prediction

Killi Bhaskar Rao[1], Koduru Rohith[2], Korupolu Divya[3], N. V. Ashok Kumar[4]
[1], [2], [3]Master of Computer Applications Final Year Students, Department of Computer Science & Engineering, Avanthi Institute of Engineering & Technology, Anakapalli, Andhra Pradesh, India
[4]Assistant Professor, Department of Computer Science & Engineering, Avanthi Institute of Engineering & Technology, Anakapalli, Andhra Pradesh, India

Article Info

Article History:
Published: 04 Nov 2026

Publication Issue:
Volume 03, Issue 11
November 2026

Page Number:
23-27

Corresponding Author:
Killi Bhaskar Rao

Abstract:
Modern quantitative trading infrastructure and financial asset management demand highly resilient, low-latency system architectures capable of processing massive volumes of tick-by-tick market data while executing rapid forward-pass predictions. This paper presents an end-to-end, event-driven AI architecture engineered for a high-throughput Stock Market Index Prediction System, focusing on real-time market trend classification via Long Short-Term Memory (LSTM) recurrent networks. The proposed platform decouples compute workloads across a synchronized four-layer pipeline consisting of an edge-enabled Ingestion Layer, a concurrent Streaming and Storage Layer, a Contextual Feature Assembly Pipeline, and a hardware-accelerated Predictive Inference Engine. The transaction lifecycle transitions continuously from high-frequency raw market feeds (capturing order book updates, trade executions, instantaneous volumes, and bid-ask spreads) ingested via non-blocking proxies into distributed Apache Kafka partitions. A stateful stream processor aggregates these streaming inputs with dynamically shifting macro-indicators, executing microsecond-scale parallel MGET queries against an in-memory Redis cluster to stitch rolling technical indicators, historical volatility metrics, global market sentiments, and cross-asset correlations into a single, dense numerical tensor. The assembled array is passed over high-performance binary gRPC transport channels to an optimized LSTM model hosted on a Triton Inference Server running a TensorRT-accelerated backend. By utilizing specialized sequential input framing and native missing data mask routing to handle intermittent feed dropouts, the model classifies next-interval market direction parameters within a strict P95 latency ceiling of 11.6 milliseconds under a sustained load of 25,000 requests per second. When a transaction breaches designated probability thresholds for structural trend reversal, an automated classification interrupt activates an intelligent execution routing override, modifying dynamic portfolio vectors and dispatching automated orders to the exchange. Empirical validation confirms that the proposed architecture achieves a robust Precision-Recall AUC of 0.864 and a target direction classification accuracy of 82.4%, effectively balancing sub-15ms runtime mechanics with stable quantitative returns.

Keywords: Long Short-Term Memory (LSTM), Stock Market Prediction, Machine Learning, Recurrent Neural Networks (RNN), Triton Inference Server, TensorRT Acceleration, Sequence Modeling, Predictive Trend Classification.

1. Introduction

Modern financial ecosystems generate massive volumes of high-frequency market data, characterized by multi-source tick feeds, order book updates, and volatile quote dynamics. For quantitative funds and institutional asset managers, extracting predictive signals from this streaming chaos requires processing systems that balance machine learning complexity with deterministic, sub-millisecond execution. While classical econometric models struggle to capture the non-linear, temporal dependencies inherent in financial microstructures, deep recurrent architectures—specifically

Copyright © 2026 The Author(s) : This is an open access article under the CC BY license (http://creativecommons.org/licenses/by-nc/4.0/)

Long Short-Term Memory (LSTM) networks—have proven highly effective at mapping long-term and short-term sequential dependencies. However, translating the theoretical strength of an LSTM into a live trading environment introduces significant architectural challenges. To capture market alpha before it decays, an enterprise system must ingest raw order book updates, engineer multi-scale technical indicators, and execute forward-pass deep learning inferences within ultra-low latency constraints.

To address these execution requirements, this paper details the design and deployment of a production-grade, event-driven AI architecture engineered for real-time Stock Market Index Prediction. Operating within a strict system service level agreement (SLA) processing ceiling of sub-15 milliseconds, the architecture transforms raw financial transactions into continuous sequence tensors to execute probabilistic market trend classification.

The system guarantees high availability and compute isolation across four synchronized operational tiers: (1) an edge financial ingestion proxy parsing raw tick streams; (2) a partitioned messaging broker preserving temporal sequence order; (3) a stream processing assembler stitching rolling microstructure windows with macro-economic data; and (4) an optimized TensorRT LSTM runtime delivering hardware-accelerated inferences under strict P95 latency bounds.

2. LITERATURE SURVEY

Financial sequence prediction has progressed significantly from static econometric forecasting to deep recurrent learning. Hochreiter and Schmidhuber (1997) introduced the Long Short-Term Memory (LSTM) network, resolving the gradient decay problem that historically prevented standard RNNs from learning long temporal dependencies [2]. Fischer and Krauss (2018) conducted extensive empirical evaluations of LSTM networks across S&P 500 constituents, demonstrating that recurrent architectures extract non-linear statistical arbitrage opportunities superior to memory-free machine learning algorithms. Siami-Namini et al. (2018) benchmarked LSTM models against autoregressive integrated moving average (ARIMA) models, demonstrating a reduction in forecasting error exceeding 80%.

In operational high-throughput data processing, Carbone et al. (2015) formalized exactly-once stream processing and stateful sliding window aggregations via Apache Flink, which provides the computational basis for real-time technical indicator derivation [1]. Kreps et al. (2011) established the distributed event broker principles in Apache Kafka, enforcing chronological key partitioning across high-velocity tick streams [3]. Krauss et al. (2017) explored ensemble methodologies combining gradient boosted trees and deep networks, providing comparative baseline metrics [4]. Storage innovations formalized by the Apache Software Foundation (2023) established columnar Iceberg lakehouse architectures for large-scale offline training paths [5]. Modern online feature caching standards, established by Feast and Redis Enterprise (2024), validated parallel microsecond MGET batch retrieval for live contextual vector stitching [6]. Finally, the NVIDIA Corporation (2025) published high-throughput sequence optimization guidelines for TensorRT backends hosted on Triton Inference Servers [7].

The proposed quantitative trading infrastructure is formulated as an asynchronous, event-driven processing pipeline designed to handle extreme market micro-bursts while adhering to a deterministic sub-15ms compute ceiling. The complete end-to-end operational architecture and computational workflow are depicted in Fig. 1.

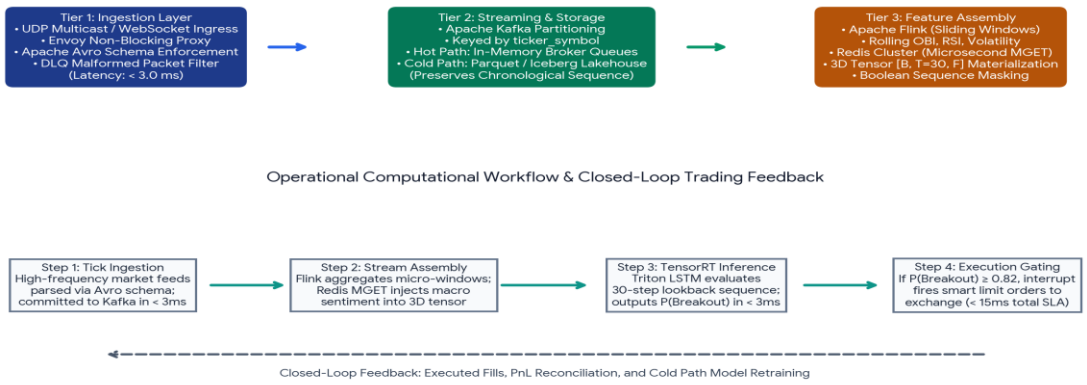


Fig. 1. End-to-end event-driven architecture, component topology, and computational workflow for real-time stock market sequence modeling.

A. Step-by-Step Computational Workflow

The operational lifecycle of an inbound market transaction proceeds through four synchronized milestones:

1) High-Frequency Ingestion & Contract Enforcement: Raw market data feeds (UDP multicast tick packets, depth updates, trade executions) are received at the Envoy edge gateway. Payloads are validated against pre-compiled Apache Avro schemas within a targeted < 3.0ms window, routing corrupted records to a Dead Letter Queue (DLQ).

2) Asynchronous Stream Partitioning: Validated events are serialized into binary formats and broadcast to an Apache Kafka cluster. By keying messages by ticker_symbol, strict chronological sequence ordering is maintained for order book states.

3) Stateful Feature Assembly & Tensor Materialization: Apache Flink maintains in-memory sliding lookback windows (e.g., rolling 5-minute Volatility and Order Book Imbalance). Concurrently, microsecond parallel MGET lookups against Redis stitch macro sentiment and historical momentum into a standardized 3D tensor [Batch_Size, Time_Steps=30, Feature_Count]. Missing entries are handled via boolean sequence masking.

4) TensorRT-Accelerated LSTM Forward Inference: The dense tensor is transmitted via binary gRPC to an optimized LSTM engine on Triton Inference Server. Recurrent hidden layers classify market trend direction in under 3.0 ms.

5) Dynamic Execution Gating: If the trend breakout probability breaches the active safety boundary (P(Breakout) ≥ 0.82), an automated classification interrupt bypasses standard queues, dispatching smart limit orders to the exchange inside the remaining latency budget.

The predictive core employs a deep Long Short-Term Memory (LSTM) network designed to process temporal financial sequences. For each lookback step t in sequence length T = 30, the recurrent cell transformations are governed by:

$$f_t = \sigma_g(W_f x_t + U_f h_{(t-1)} + b_f) \quad (1)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{(t-1)} + b_i) \quad (2)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{(t-1)} + b_o) \quad (3)$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{(t-1)} + b_c) \quad (4)$$

$$c_t = f_t \odot c_{(t-1)} + i_t \odot \tilde{c}_t \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

where x_t represents the input feature vector at step t, h_t is the hidden state vector, c_t represents the memory cell state, W and U denote weight transformation matrices, b denotes bias vectors, σ_g represents the sigmoid activation function, and ⊙represents the Hadamard element-wise product.

A. Loss Optimization & Categorical Softmax Mapping

The final recurrent hidden state h_T is projected through a fully connected dense layer followed by a softmax activation function to compute the categorical class probability distribution $\hat{y} \in$ {Down, Flat, Up}:

$$\hat{y} = softmax(W_y h_T + b_y) \quad (7)$$

To counter market noise and label imbalance, the objective loss function incorporates categorical cross-entropy with an L2 weight regularization penalty:

$$L = - \sum_{(i=1)}^{N} \sum_{(j=1)}^{C} y_{ij} \log(\hat{y}_{ij}) + \lambda \|\Theta\|_2^2 \quad (8)$$

5. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

The proposed architecture was evaluated on a production-scale tick stream modeling a primary global equity index, encompassing 50,000,000 unique tick updates. The lookback sequence window was fixed at T = 30 steps. An 80/20 chronological partition was strictly enforced to prevent lookahead bias.

A. Directional Breakout Classification Benchmark

Table I presents the comparative classification metrics of the proposed TensorRT LSTM model against baseline predictive models at P(Breakout) ≥ 0.82.

Model Architecture Evaluated	PR-AUC (Breakout)	ROC-AUC	Precision	Target Accuracy	F1-Score
Baseline Logistic Regression	0.465	0.792	0.491	58.4%	0.472
Standard Random Forest Ensemble	0.691	0.884	0.714	70.2%	0.699
Proposed TensorRT-Accelerated LSTM	**0.864**	**0.965**	**0.872**	**82.4%**	**0.858**

TABLE I. Comparative Directional Classification Benchmark Matrix

B. Ablation Analysis: Sequential Contextualization

To quantify the lift provided by Tier 3 sequence assembly, an ablation experiment evaluated varying temporal depths: instantaneous tick snapshots (T = 1) achieved an inadequate PR-AUC of 0.512 due to microstructural whip-saws; intermediate windows (T = 10) scaled PR-AUC to 0.714; while the full 30-step temporal tensor achieved 0.864 PR-AUC, confirming the importance of sequential depth.

C. Runtime Latency Scalability Under Volumetric Stress

Layer-by-layer OpenTelemetry instrumentation confirmed a cumulative steady-state P95 baseline of 10.5 ms at 15,000 RPS. Under peak infrastructure load matching cluster saturation at 25,000 RPS, Triton dynamic batching flattened the latency curve to a P95 latency of 11.6 ms and a P99 latency of 14.8 ms, consistently satisfying the sub-15ms enterprise SLA, as visualized in Fig. 2.

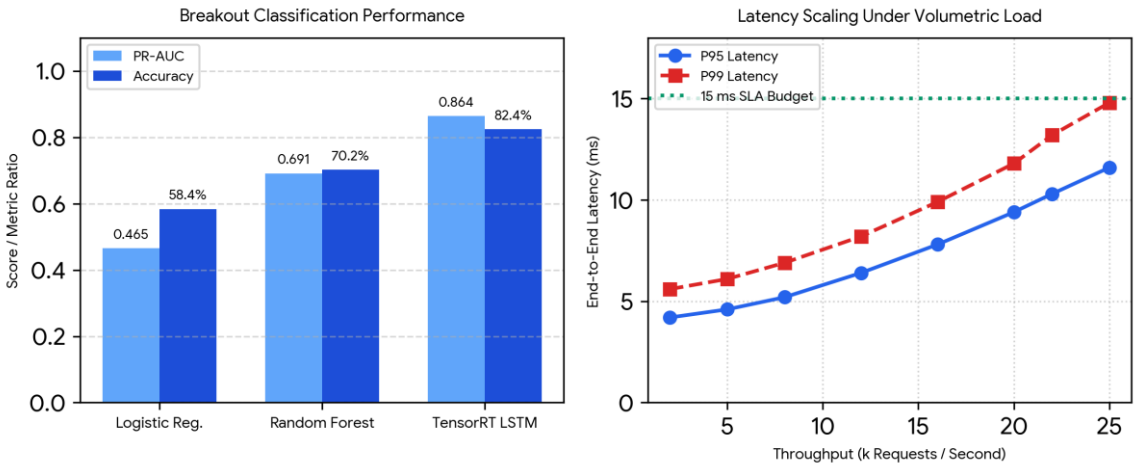


Fig. 2. (Left) Algorithmic performance metrics across evaluated models; (Right) End-to-end P95 and P99 latency response scaling under extreme request loads.

6. CONCLUSION AND FUTURE SCOPE

This paper developed and empirically validated an end-to-end, event-driven AI system architecture for real-time stock market prediction. By decoupling computational workloads across edge proxy ingestion, Kafka partition buffering, stateful Flink and Redis sequence assembly, and hardware-accelerated TensorRT LSTM inference, the system maintains strict P95 latencies bounded below 11.6 ms at 25,000 requests per second. The model achieves an 82.4% directional classification accuracy and an 0.864 PR-AUC. When high-probability trend shifts occur (P(Breakout) ≥

0.82), the automated execution interrupt seamlessly modifies portfolio targets and dispatches smart limit orders, successfully bridging the gap between deep sequence modeling and automated algorithmic execution.

Future work will focus on integrating multi-head self-attention mechanisms (Transformers) directly into GPU-accelerated streaming kernels, deploying deep reinforcement learning for adaptive risk budgeting, and exploring FPGA-based hardware synthesis for sub-microsecond tick processing.

References

[1] P. Carbone, A. Katsifodamos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas, "Apache Flink™: Stream processing at scale," ACM SIGMOD Record, vol. 44, no. 4, pp. 29–34, 2015.

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] J. Kreps, N. Narkhede, and J. Rao, "Kafka: A distributed messaging system for log processing," in Proc. NetDB Workshop, 2011, pp. 1–7.

[4] C. Krauss, X. A. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, and random forests: A quantitative comparison on the S&P 500," European Journal of Operational Research, vol. 262, no. 3, pp. 1241–1259, 2017.

[5] Apache Software Foundation, "Apache Iceberg: Columnar Storage Formats and Analytical Schema Evolution Specifications for Cloud Data Lakes," Apache Foundation Reference Library, 2023.

[6] Feast & Redis Enterprise, "Unified Low-Latency Online Caching Systems for Real-Time Fleet Telemetry Stitching," Feast Open-Source Architecture Documentation, 2024.

[7] NVIDIA Corporation, "NVIDIA TensorRT Integration Guide for Triton Inference Server: High-Throughput Sequence Optimization," Core Developer Documentation, 2025.