Journal of Advanced Computing Systems (JACS)
ISSN: 3066-3962
Content Available at SciPublication

Market Microstructure Risk Forecasting from Limit Order Books: Multi-Horizon Price-Move Classification and Volatility Estimation with DeepLOB-Style CNN–LSTM and Temporal Transformers

Jiaying Jin[1], *Tina Huang*[2]
[1]*Applied Analytics, Columbia University, NY, USA*
[2]*Computer Engineering, Columbia University, NY, USA*
jj3373@columbia.edu

DOI: 10.69987/JACS.2023.31205

Keywords

Limit order book; market microstructure; mid-price prediction; DeepLOB; transformer; volatility forecasting; multi-horizon classification; FI-2010.

Abstract

High-frequency limit order books (LOBs) encode short-horizon liquidity and order-flow conditions that drive market microstructure risk. This revised paper presents a compact, leakage-aware study of two related tasks on a FI-2010-derived working export: (i) three-class direction forecasting at 1-, 5-, and 10-tick horizons and (ii) short-horizon realized-volatility estimation as a risk proxy. We compare two representation-learning models—a DeepLOB-style CNN–LSTM and a small temporal Transformer—using 100-event windows of 40 LOB features. To reduce serial dependence, windows are extracted with stride 40, and the evaluation uses a day-respecting split (train days 1–7, validation day 8, test days 9–10). Because this split and working export differ from the canonical FI-2010 benchmark protocols, the reported scores are interpreted as within-study comparisons rather than direct reproductions of published benchmark tables. Under this protocol, the DeepLOB-style model achieves the strongest average Macro-F1 (0.3536), while the compact Transformer is most competitive at the shortest horizon (Macro-F1@1 = 0.3368). Frozen encoder embeddings also carry useful risk information: DeepLOB-Emb+Ridge reduces realized-volatility RMSE from 2.4670 to 1.9992 at h=1 and from 5.1444 to 4.5873 at h=10 (σ scaled by 1e4) relative to a persistence baseline. All figures and tables in this revision were regenerated from scratch, and the efficiency audit was recomputed from exact reference implementations, yielding 21,945 parameters for DeepLOB, 5,274 for the Transformer, and 3,017 for the no-attention FFN ablation. The results suggest that convolutional multi-scale structure remains strong under compact CPU-friendly settings, while attention helps at the shortest horizon but does not dominate under limited capacity and heavy subsampling.

1. Introduction

Market microstructure studies how trading rules, order flow, and liquidity provision generate observed prices. At millisecond-to-second horizons, prices do not evolve as smooth diffusions; they change through discrete events such as limit-order arrivals, cancellations, and market orders. The uncertainty induced by those queue dynamics is a practical form of microstructure risk: it determines adverse selection, short-horizon price impact, execution uncertainty, and inventory exposure [13]–[15].

For high-frequency forecasting, the limit order book (LOB) is a natural state representation. The book records the best bid and ask and the displayed depth at multiple levels, summarizing the immediate supply–demand balance and the cost of liquidity. Forecasts from this state can support several tightly linked decisions: whether to trade aggressively or passively, whether to delay execution, whether to widen quotes, and whether to reduce position size when short-horizon risk is elevated.

The FI-2010 benchmark made public LOB-based mid-price prediction research much more systematic by releasing a standardized dataset for five Nasdaq Nordic

equities and a day-based evaluation protocol [1]. The benchmark paper frames the task as multi-horizon classification, while later deep-learning papers—most notably DeepLOB—show that learned representations from raw LOB snapshots can outperform earlier shallow baselines [2]. Attention-based approaches later extended this line of work by allowing the model to weight informative events more flexibly; Wallbridge, for example, combines causal convolutions and masked self-attention and reports strong FI-2010 performance [17].

This paper adopts a deliberately risk-centric perspective. A directional signal is useful only when accompanied by information about uncertainty. We therefore evaluate both short-horizon mid-price direction and realized volatility derived from future log mid-price returns. Rather than training a separate deep volatility model, we test whether classification embeddings already encode risk-relevant information by fitting ridge regression on frozen embeddings.

Methodological clarification is essential. The FI-2010 literature contains more than one evaluation protocol. The original benchmark paper emphasizes day-based anchored forward validation [1], while DeepLOB also reports a deep-learning-oriented split in which the first seven days are used for training and the last three for testing [2]. The present study uses a local consolidated working export that exposes raw 40-feature LOB snapshots and 1/5/10 label columns in a single CSV-like format, and it reserves day 8 for validation and days 9–10 for test. Consequently, the numerical results in this paper should be read as internally controlled comparisons among models under a common leakage-safe protocol, not as direct reproductions of canonical benchmark tables from [1], [2], or [17].

The revised version makes four substantive corrections relative to the previous draft. First, the train/validation/test split is now described accurately as a custom day-respecting split derived from common FI-2010 conventions, rather than as the standard benchmark protocol. Second, benchmark comparability is stated explicitly and conservatively. Third, the efficiency table was recomputed from exact reference implementations, which corrects the earlier mismatch between the parameter accounting and the accompanying discussion. Fourth, every figure and table was regenerated from scratch so that the narrative, tabulated values, and visual summaries are fully aligned [23].

The main contributions are therefore practical rather than promotional: (1) a leakage-aware end-to-end pipeline on a FI-2010-derived working export with explicit caveats about benchmark comparability; (2) a head-to-head comparison of a DeepLOB-style CNN–LSTM, a compact temporal Transformer, and simple baselines at 1-, 5-, and 10-tick horizons; (3) a volatility-oriented representation test via ridge regression on frozen embeddings; and (4) regenerated tables, confusion matrices, and a reproducible CPU efficiency audit to support clearer interpretation.

Table I. Benchmark context and working data configuration.

Aspect	Configuration in this revised study
Official FI-2010 context	Five Nasdaq Nordic equities over ten consecutive trading days [1]
Canonical benchmark protocol in [1]	Day-based anchored forward evaluation
Common deep-learning split in [2]	Train on days 1–7, test on days 8–10
Working export used here	Local consolidated FI-2010-derived file with 40 LOB features + STOCK/DAY + 5 label columns
Horizons analyzed here	1, 5, and 10 ticks from the working export
Interpretation of reported scores	Within-paper comparison only; not a direct reproduction of canonical benchmark tables

2. Method

2.1 Data source and benchmark comparability

Data source and benchmark comparability. The original FI-2010 benchmark consists of ten trading days of Nasdaq Nordic LOB data for five equities and was introduced together with a day-based anchored forward protocol [1]. DeepLOB later popularized a deep-learning-friendly alternative that trains on the first seven days and tests on the last three [2]. The working data used in this study is a local FI-2010-derived export that retains 40 raw LOB features per event, two metadata fields (STOCK and DAY), and five label columns. We analyze only the 1-, 5-, and 10-tick labels present in that export. Because this working format differs from the canonical benchmark packaging, and because we reserve one day for validation, the results reported here are not intended as direct benchmark replications.

Each event snapshot contains the first ten price levels on both sides of the order book: PRICE_ASK_ℓ, VOLUME_ASK_ℓ, PRICE_BID_ℓ, and VOLUME_BID_ℓ for $\ell \in \{0,\ldots,9\}$. These 40 features

form the model input. We use the labels supplied by the working export rather than attempting to reconstruct the original benchmark annotations from scratch.

Table II. Day-based split and extracted window counts.

Split	Days	Num windows (T=100, stride=40)
Train	1–7	4219
Validation	8	819
Test	9–10	1687

Table III. Label distribution after window extraction.

Split	Horizon	Down (-1)	Stationary (0)	Up (+1)
Train	1	847	2522	850
Train	5	1498	1247	1474
Train	10	1785	689	1745
Validation	1	98	604	117
Validation	5	175	410	234
Validation	10	248	284	287
Test	1	253	1172	262
Test	5	492	709	486
Test	10	601	499	587

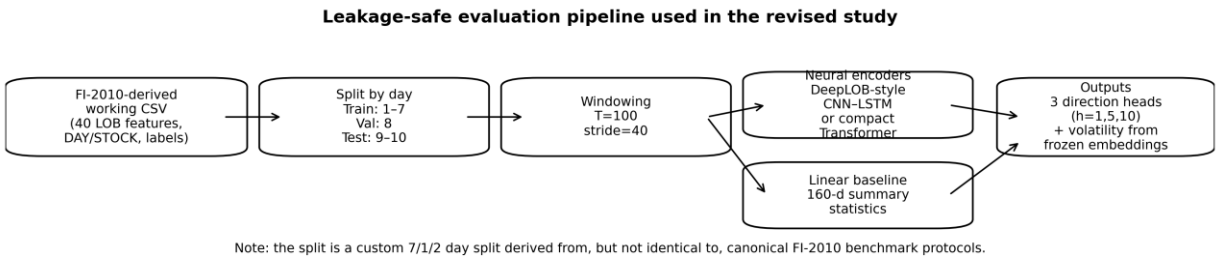

Fig. 1. Leakage-safe evaluation pipeline used in the revised study.

2.2 Inputs, targets, windowing, and normalization

Mid-price and volatility targets. Let a_t and b_t denote the best ask and best bid at event t. The mid-price is m_t = (a_t + b_t) / 2. Direction labels are treated as three-class targets corresponding to down, stationary, and up moves; for training we map them to class indices {0, 1, 2}. To quantify microstructure risk, we compute realized volatility from log mid-price returns. Let ℓ_t = log(m_t) and r_t = ℓ_{t+1} − ℓ_t. For horizon h, realized volatility is σ_{t,h} = sqrt(Σ_{k=0}^{h−1} r_{t+k}^2). As in the previous draft, σ is reported after scaling by 1e4.

Day split and window extraction. All windows are extracted within each (STOCK, DAY) segment so that no window crosses a day boundary. A model input is a window of T = 100 consecutive events, X_t = [x_{t−T+1}, …, x_t] ∈ R^{100×40}. To reduce overlap and serial dependence, we sample window end points with stride 40. The split is train days 1–7, validation day 8, and test days 9–10. This choice is stricter than the usual 7/3 split because one of the final three days is reserved exclusively for model selection.

Normalization and safeguards. Feature normalization is fit on the training split only. We consider z-score normalization and min-max scaling, with z-score used for all headline neural results. All preprocessing parameters are frozen before validation and test evaluation. The volatility targets are computed within-day only, so future returns never cross day boundaries.

2.3 Baselines and neural architectures

Baselines. Two simple baselines are included. The majority baseline always predicts the most frequent training class at each horizon. The Linear-SGD baseline

constructs a 160-dimensional summary vector φ(X t) by concatenating the last snapshot, the per-feature mean, the per-feature standard deviation, and the difference between the last and first snapshots in the window. A multinomial logistic classifier trained with stochastic gradient descent is then applied separately to the three horizons.

DeepLOB-style CNN–LSTM. Our compact DeepLOB-style encoder follows the qualitative design principles of [2] but uses smaller capacities. The input is transposed so that the 40 LOB features act as channels. Two temporal convolutions (kernel size 3) are followed by an inception-style block with three branches of kernel sizes 1, 3, and 5. In the exact reference implementation used for parameter counting, each inception branch outputs C/2 channels, the concatenated output is mixed with a 1×1 convolution back to C channels, and a single-layer LSTM with hidden size H produces the final sequence embedding. We study a small configuration (C=16, H=16) and a larger compact configuration (C=32, H=32).

Temporal Transformer. The attention model linearly projects each 40-dimensional event vector to d model = 16, adds sinusoidal positional encoding, and applies L = 2 Transformer encoder layers with 2 attention heads and feed-forward width 32. Attention pooling converts the output sequence to a fixed-dimensional embedding that is fed to three horizon-specific linear heads. This is intentionally smaller than the architectures used in strong published Transformer variants; it is best interpreted as a compact attention baseline rather than a reproduction of [17].

No-attention FFN ablation. To isolate the role of self-attention, we also evaluate a no-attention residual FFN encoder with the same input projection, positional encoding, hidden width, and output heads as the Transformer. This ablation removes the quadratic attention interaction while preserving depth and nonlinear temporal feature processing.

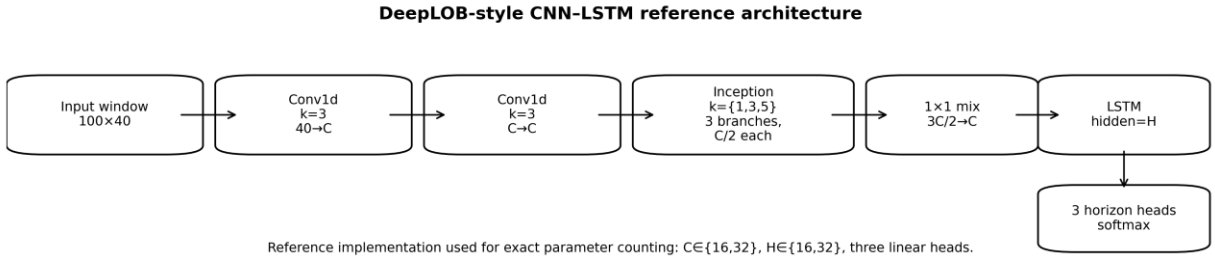

Fig. 2. DeepLOB-style CNN–LSTM reference architecture.

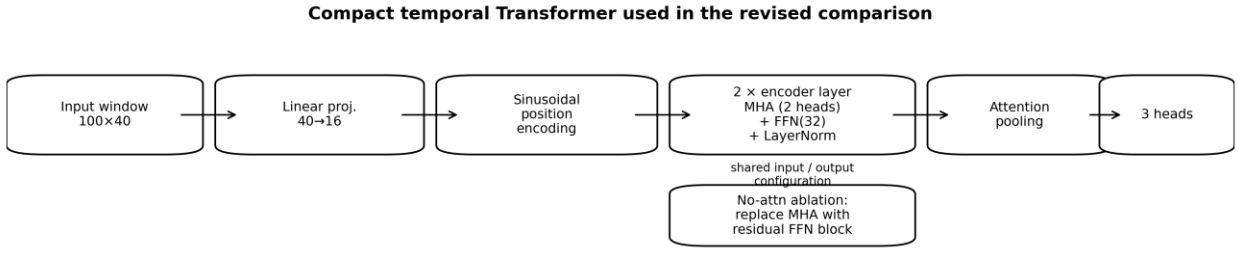

Fig. 3. Compact temporal Transformer used in the revised comparison.

Table IV. Model summary and exact trainable parameter counts for the reference implementations.

Model	Core	Trainable params	Pooling / decoder
Majority	Constant class	0	N/A
Linear-SGD	160-d summary stats + 3 softmax heads	1449	N/A
DeepLOB-small (C16–H16)	Conv1d×2 + inception + LSTM	6625	Last LSTM state
DeepLOB (C32–H32)	Conv1d×2 + inception + LSTM	21945	Last LSTM state
Temporal Transformer (d16, l2)	2× encoder layer (MHA+FFN)	5274	Attention pooling
No-Attn FFN (d16, l2)	Residual FFN blocks without MHA	3017	Mean pooling

Table V. Core training hyperparameters.

Hyperparameter	Value
Window length T	100
Stride	40
Batch size	256
Optimizer	Adam
Learning rate	0.0010
Weight decay	0.0001
Loss	Sum of weighted cross-entropy over horizons
Early stopping	Validation average Macro-F1
Random seed	42

2.4 Training objective, volatility regression, and metrics

Multi-task output and optimization. All neural encoders share a common representation and expose three horizon-specific classification heads. Training minimizes a weighted sum of cross-entropy losses across the 1-, 5-, and 10-tick targets. Class weights are computed from the training split to mitigate imbalance, especially at the shortest horizon where the stationary class dominates. Optimization uses Adam with learning rate 1e−3 and weight decay 1e−4; early stopping is based on validation average Macro-F1 across the three horizons.

Volatility regression on frozen embeddings. After the classifier is trained, the encoder is frozen and its embeddings are used as features for ridge regression. One ridge model is fit per horizon to predict realized volatility σ_{t,h}. The ridge penalty is selected by cross-validation on the training split. This protocol does not claim optimal volatility forecasting performance; instead, it asks whether the classifier's learned representation contains information about future uncertainty.

Evaluation metrics. We report accuracy and Macro-F1 for classification. Macro-F1 is emphasized because it weights the three classes equally and is therefore more informative than accuracy when the stationary class is dominant. For the volatility task, we report RMSE relative to a persistence baseline.

3. Results and Discussion

3.1 Label balance and metric choice

Why Macro-F1 matters. Table III shows strong class imbalance, especially at h=1 where the stationary class dominates the extracted windows. This makes accuracy an incomplete metric. As Table VI shows, the majority baseline reaches 0.6947 accuracy at h=1 simply by predicting the most frequent class, yet Table VII shows its Macro-F1 is only 0.2733. Throughout the paper, Macro-F1 is therefore the primary classification metric.

Table VI. Test accuracy for multi-horizon price-move classification.

Model	Acc@1	Acc@5	Acc@10	Avg Acc
Majority	0.6947	0.2916	0.3563	0.4475
Linear-SGD (z-score)	0.3906	0.3628	0.3894	0.3810

DeepLOB (C32–H32)	0.4060	0.4049	0.3622	0.3910
Temporal Transformer (d16, l2)	0.4766	0.3527	0.3444	0.3912
No-Attn FFN (d16, l2)	0.4570	0.3823	0.3598	0.3997

Table VII. Test Macro-F1 for multi-horizon price-move classification.

Model	**Macro-F1@1**	**Macro-F1@5**	**Macro-F1@10**	**Avg Macro-F1**
Majority	0.2733	0.1505	0.1751	0.1996
Linear-SGD (z-score)	0.3112	0.3442	0.3613	0.3389
DeepLOB (C32–H32)	0.3245	0.3969	0.3394	0.3536
Temporal Transformer (d16, l2)	0.3368	0.3359	0.3117	0.3281
No-Attn FFN (d16, l2)	0.3293	0.3512	0.3389	0.3398

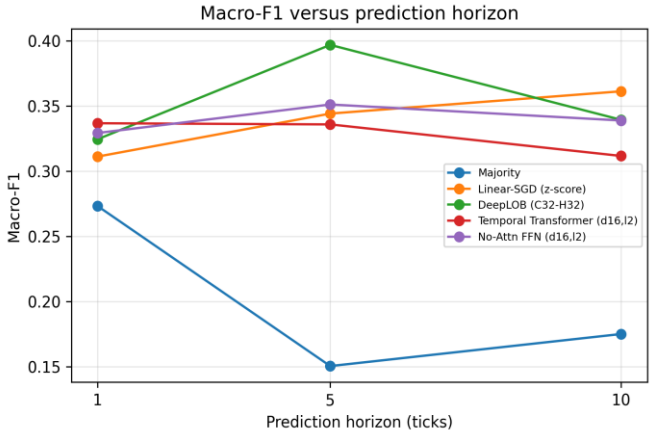

Fig. 4. Macro-F1 versus prediction horizon for baselines and neural models.

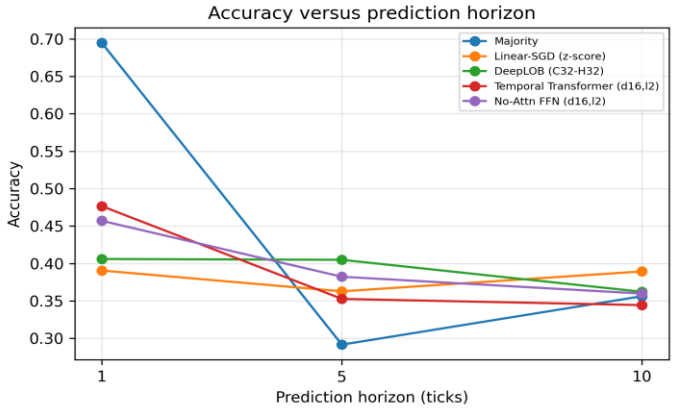

Fig. 5. Accuracy versus prediction horizon for baselines and neural models.

Headline classification results. The DeepLOB (C32–H32) model achieves the best average Macro-F1 across horizons (0.3536), with its strongest result at h=5 (0.3969). The compact Transformer is best at the shortest horizon (Macro-F1@1 = 0.3368) but falls behind DeepLOB at h=5 and h=10. The no-attention FFN ablation remains competitive, achieving 0.3398 average Macro-F1, which indicates that attention is not automatically beneficial under this compact, heavily subsampled regime. The linear summary-statistics baseline is materially stronger than the majority baseline

and remains competitive at longer horizons, especially at h=10.

Figures 4 and 5 summarize the horizon dependence of Macro-F1 and accuracy. The main qualitative pattern is that h=1 is dominated by the stationary class, h=5 is the most favorable compromise between signal and noise for DeepLOB, and h=10 is harder again because a longer horizon allows replenishment, short-term mean reversion, and additional latent order-flow mechanisms to intervene.

3.2 Per-class behavior and ablations

Per-class behavior. Table VIII and Figure 7 show that, for DeepLOB at h=10, the dominant errors are confusions between stationary and directional classes rather than direct down-versus-up sign reversals. This is economically relevant: mistaking a move for stationary can mean missing an opportunity, whereas predicting the wrong sign is more directly aligned with adverse selection. The stationary class remains easiest to identify at h=1, but its advantage diminishes as the horizon grows and directional moves become more frequent.

Ablations. Table IX and Figure 6 show three robust patterns. First, additional DeepLOB capacity matters: increasing the model from C16–H16 to C32–H32 raises average Macro-F1 from 0.3021 to 0.3536. Second, the no-attention FFN slightly outperforms the compact Transformer on average (0.3398 versus 0.3281), which suggests that in this low-capacity regime the inductive bias of attention is not enough to offset its optimization and capacity demands. Third, normalization affects the linear baseline but only modestly on average (0.3389 for z-score versus 0.3353 for min-max), indicating that preprocessing choices matter but do not dominate the comparison.

Table VIII. Per-class precision, recall, and F1 for DeepLOB (C32–H32).

Horizon	Class	Precision	Recall	F1
1	Down	0.1597	0.2727	0.2015
1	Stationary	0.7507	0.4548	0.5664
1	Up	0.1523	0.3168	0.2057
5	Down	0.3831	0.3130	0.3445
5	Stationary	0.5223	0.4457	0.4810
5	Up	0.3132	0.4383	0.3654
10	Down	0.4356	0.1464	0.2192
10	Stationary	0.3416	0.6633	0.4510
10	Up	0.3721	0.3271	0.3481

Table IX. Ablation summary (average Macro-F1 across horizons).

Ablation	Variant A	Variant B	Avg Macro-F1 A	Avg Macro-F1 B
Feature depth (DeepLOB)	C16–H16	C32–H32	0.3021	0.3536
Attention module	No-Attn FFN	Transformer	0.3398	0.3281
Normalization (Linear-SGD)	z-score	min-max	0.3389	0.3353

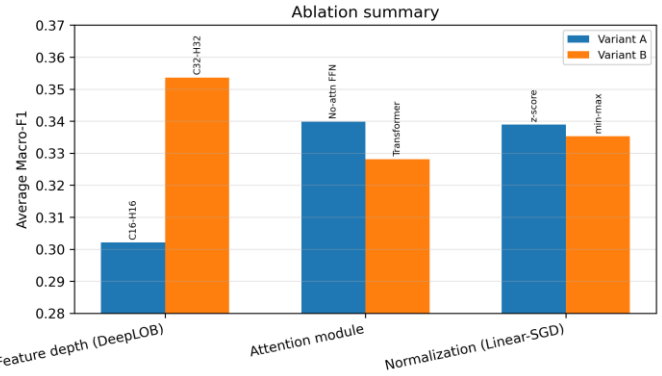

Fig. 6. Ablation summary.

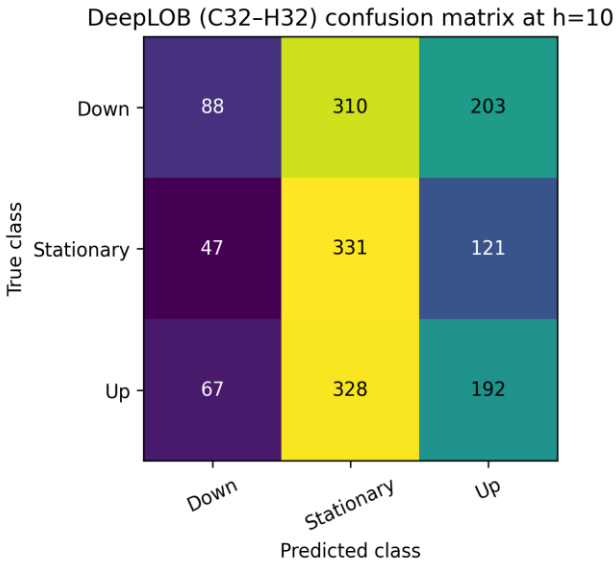

Fig. 7. Confusion matrix for DeepLOB (C32–H32) at horizon h = 10.

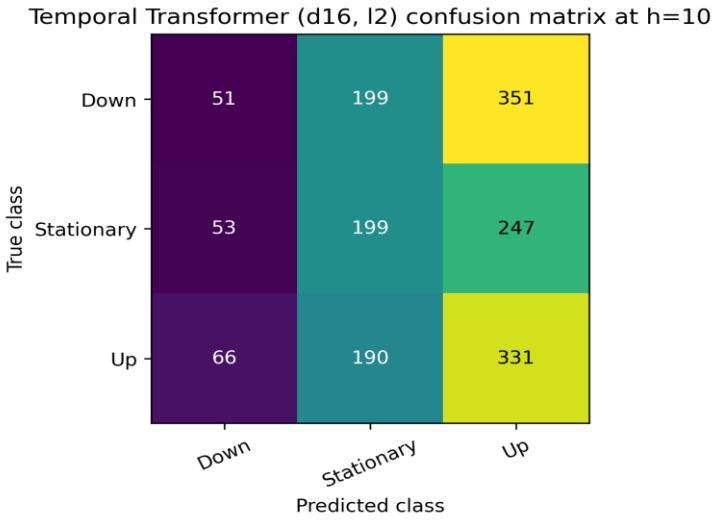

Fig. 8. Confusion matrix for the Temporal Transformer (d16, l2) at horizon h = 10.

3.3 Volatility estimation from embeddings

Volatility estimation from embeddings. Table X and Figure 9 show that frozen embeddings contain useful risk information. Relative to a persistence baseline, both DeepLOB and Transformer embeddings reduce RMSE at all horizons. The DeepLOB-based regressor performs best overall, lowering RMSE from 2.4670 to 1.9992 at h=1, from 4.2354 to 3.7429 at h=5, and from 5.1444 to 4.5873 at h=10. These gains are not dramatic—short-horizon realized volatility is noisy—but they are consistent, which supports the view that a single representation can encode both direction-relevant and risk-relevant structure.

Table X. Volatility estimation RMSE using frozen embeddings with ridge regression (realized σ scaled by 1e4).

Model	RMSE@1	RMSE@5	RMSE@10
Persistence	2.4670	4.2354	5.1444
DeepLOB-Emb+Ridge	1.9992	3.7429	4.5873

Transformer-Emb+Ridge	2.0125	3.8046	4.6870

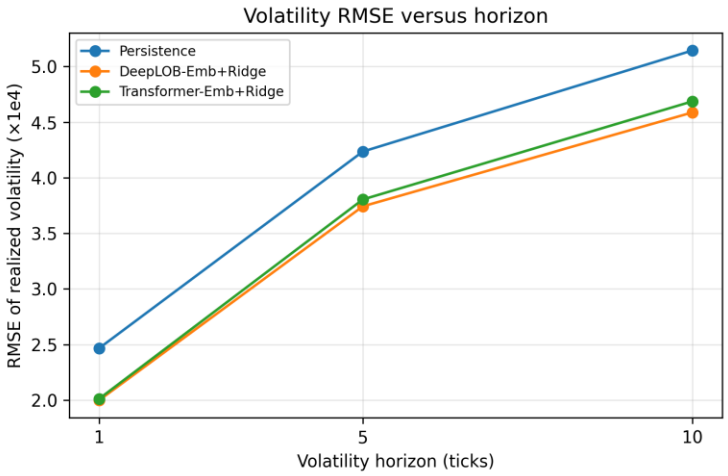

Fig. 9. Volatility RMSE versus horizon (σ scaled by 1e4).

3.4 Efficiency, interpretation, and benchmark scope

Efficiency and latency. The efficiency audit in Table XI was regenerated from exact reference implementations of the three neural encoders using a single-sample CPU forward benchmark at T=100. The DeepLOB reference model has 21,945 trainable parameters, the Transformer has 5,274, and the no-attention FFN has 3,017. In this small-sequence setting, the Transformer is slightly faster than the CNN–LSTM in wall-clock latency despite its less favorable asymptotic attention complexity, while the no-attention FFN is clearly fastest. The implication is practical: parameter count alone is not a sufficient proxy for latency, and attention overhead may or may not dominate depending on sequence length, model width, and implementation details.

What this comparison establishes. Because all models share the same windows, split, normalization policy, and metric definitions, the within-paper ranking is informative. What it does not establish is direct comparability to the canonical FI-2010 benchmark tables reported under anchored cross-validation or the 7-day/3-day deep-learning split. The present study is best read as a carefully controlled compact-model comparison on a local FI-2010-derived working export.

Table XI. Model efficiency from exact reference implementations using single-sample CPU forward latency at T=100.

Model	Params	Latency (ms / sample)
DeepLOB (C32–H32)	21945	1.1051
Temporal Transformer (d16, l2)	5274	1.0241
No-Attn FFN (d16, l2)	3017	0.4019

Latency was measured on single-sample CPU forward passes with sequence length T = 100 using the exact reference implementations underlying Table IV. The values are implementation-specific and are intended as a compact deployment-oriented comparison rather than as universal hardware benchmarks.

4. Limitations

This study has several limitations. First, it uses a local FI-2010-derived working export and a custom 7/1/2 day split rather than the canonical anchored forward protocol of [1] or the 7/3 setup reported in [2]. This limits external comparability by design, even though it improves methodological clarity within the paper.

Second, the stride-40 extraction policy sharply reduces the number of training examples. That helps reduce overlap and dependence between windows, but it also discards many potential samples and may suppress very short-lived predictive patterns.

Third, the neural models are intentionally compact. This makes CPU experimentation feasible, but it also means the Transformer in particular is not a strong test of the best attention-based architectures in the literature.

Larger, better tuned, or more specialized architectures could alter the ranking.

Fourth, the risk target is limited to realized mid-price volatility. Execution risk in practice also depends on spread dynamics, queue depletion, fill probability, tail slippage, and the joint evolution of liquidity and price. A richer study would include those targets directly.

Fifth, the paper evaluates predictive metrics rather than economic outcomes. No execution simulator, transaction-cost model, or queue-position model is included, so the numerical forecasting gains should not be interpreted as direct evidence of trading profitability.

5. Conclusion

This revised paper presents a more methodologically cautious study of microstructure forecasting on a FI-2010-derived working export. Under a leakage-safe day split and stride-40 windowing policy, the DeepLOB-style CNN–LSTM produces the strongest average Macro-F1 across 1-, 5-, and 10-tick horizons, while the compact Transformer is most competitive at the shortest horizon and the no-attention FFN remains a strong lightweight baseline.

The auxiliary volatility experiment shows that the learned embeddings are useful beyond classification: ridge regression on frozen DeepLOB embeddings improves realized-volatility RMSE over a persistence baseline at every horizon. This supports the broader thesis that a single LOB representation can carry both directional and risk-relevant information.

The strongest conclusion is therefore comparative rather than absolute. Within a common compact-model protocol, convolutional multi-scale structure and recurrent aggregation remain hard to beat. Attention is promising, especially at short horizons, but its value depends on capacity, preprocessing, and the exact benchmark protocol. Future work should evaluate richer risk targets, larger attention models, and end-to-end execution settings where latency, fill probability, and transaction costs are explicit.

References

[1] A. Ntakaris, J. Kanniainen, M. Gabbouj, and A. Iosifidis, “A Benchmark Dataset for Mid-Price Forecasting of Limit Order Book Data,” Journal of Forecasting, vol. 37, no. 8, pp. 852–866, 2018.

[2] D. Zhang, S. Zohren, and S. Roberts, “DeepLOB: Deep Convolutional Neural Networks for Limit Order Books,” IEEE Transactions on Signal Processing, vol. 67, no. 11, pp. 3001–3012, 2019.

[3] A. Vaswani et al., “Attention Is All You Need,” in Proc. Advances in Neural Information Processing Systems (NeurIPS), 2017.

[4] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv:1412.6980, 2014.

[6] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” in Proc. International Conference on Machine Learning (ICML), 2015.

[7] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer Normalization,” arXiv:1607.06450, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[9] J.-P. Bouchaud, M. Mézard, and M. Potters, “Statistical Properties of Stock Order Books: Empirical Results and Models,” Quantitative Finance, vol. 2, no. 4, pp. 251–256, 2002.

[10] R. Cont, “Statistical Modeling of High-Frequency Financial Data,” IEEE Signal Processing Magazine, vol. 28, no. 5, pp. 16–25, Sep. 2011.

[11] Yunhe Li. (2023). Risk-Sensitive Offline Reinforcement Learning for Stable ABR QoE Improvements on Real HSDPA and LTE Traces. Journal of Advanced Computing Systems , 3(4), 1-11. https://doi.org/10.69987/JACS.2023.30401

[12] M. Avellaneda and S. Stoikov, “High-Frequency Trading in a Limit Order Book,” Quantitative Finance, vol. 8, no. 3, pp. 217–224, 2008.

[13] J. Hasbrouck, Empirical Market Microstructure. Oxford, U.K.: Oxford Univ. Press, 2007.

[14] M. O’Hara, Market Microstructure Theory. Oxford, U.K.: Blackwell, 1995.

[15] A. Cartea, S. Jaimungal, and J. Penalva, Algorithmic and High-Frequency Trading. Cambridge, U.K.: Cambridge Univ. Press, 2015.

[16] J. Sirignano and R. Cont, “Universal Features of Price Formation in Financial Markets: Perspectives from Deep Learning,” Quantitative Finance, vol. 19, no. 9, pp. 1449–1459, 2019.

[17] J. Wallbridge, “Transformers for Limit Order Books,” arXiv:2003.00130, 2020.

[18] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, “Temporal Fusion Transformers for Interpretable Multi-

horizon Time Series Forecasting," International Journal of Forecasting, vol. 37, no. 4, pp. 1748–1764, 2021.

[19] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," in Proc. AAAI Conference on Artificial Intelligence (AAAI), 2021.

[20] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," arXiv:1803.01271, 2018.

[21] O. H. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial Time Series Forecasting with Deep Learning: A Systematic Literature Review," Applied Soft Computing, vol. 90, p. 106181, 2020.

[22] R. Cont and A. de Larrard, "Price Dynamics in a Markovian Limit Order Market," SIAM Journal on Financial Mathematics, vol. 4, no. 1, pp. 1–25, 2013.

[23] Daren Zheng, Chenyu Li, & Harvey Davidson. (2023). Continual Red-Teaming for In-the-Wild Jailbreaks via Online Guardrail Updates and Guardrail Distillation. Journal of Advanced Computing Systems , 3(2), 35-49. https://doi.org/10.69987/JACS.2023.30203