

A Systematic Hyperparameter Analysis of Deep Learning Models for Limit Order Book Mid-price Prediction

Hongyi Li

*School of Management and Marketing, College of Business, Murdoch University,
Perth, Australia
36124563@student.murdoch.edu.au*

Abstract. Limit order book (LOB) mid-price prediction plays an important role in high-frequency trading. This paper presents a systematic evaluation of deep learning models and hyperparameter settings for LOB prediction using the FI-2010 benchmark. Under the same experimental settings, eleven representative neural network architectures, including recurrent, CNN-based, bilinear-based, and transformer-based models are compared. The results show that BiNTABL achieves the best overall performance in terms of macro-F1, precision, recall, and accuracy. Extensive ablation experiments are then conducted to investigate the effects of network layer number, learning rate, hidden dimensions, batch size, historical input length, and prediction horizon. The results show that prediction performance is highly sensitive to these settings. And the existence of bilinear layer, learning rate together with forecast horizon are the top three most important factors in model performance. The findings provide practical guidance for neural network design and hyperparameter selection in LOB prediction and offer a useful reference for future research on financial time-series forecasting.

Keywords: Deep learning, Limit order book, Price prediction, Empirical study

1. Introduction

High-frequency Limit Order Book (LOB) forecasting has become a core research topic in financial engineering, serving as a fundamental module for automated market making, intraday trend trading and market microstructure risk monitoring. LOB data records real-time bid–ask depth, pending order volumes and incremental order flows, which encode short-term supply–demand imbalance signals that determine the immediate movement of asset mid-prices.

Over the past decade, a rich set of deep learning architectures, ranging from classical recurrent networks e.g. Long Short-Term Memory (LSTM) [1], modern Convolutional Convolutional Networks (CNN) [2], LOB-specialized CNN-LSTM [3, 4] to transformer-based time-series models [5, 6] have been proposed for LOB prediction. Most existing limit order book forecasting studies take the FI-2010 benchmark [7], which is a 2018 low-volatility Nordic equity dataset, as the primary testbed for model comparison. However, existing literature [7-10] typically adopts heuristically selected hyperparameter configurations and omits systematic hyperparameter sensitivity ablation studies. Key hyperparameters including input lookback length, hidden dimension size and dropout

rate are tuned manually without rigorous analysis. To better understand the hyperparameters in LOB prediction models, this paper carries out comprehensive ablation experiments over core hyperparameters. To better improve the robustness and interpretability of the findings, this paper sets a three-class mid-price movement classification task (price down, stationary, price up), consistent with the FI-2010 standard experimental paradigm. Accuracy, macro-precision, macro-recall and macro-F1 are adopted as evaluation metrics, which eliminate bias from imbalanced stationary samples and enable fair comparisons against established LOB baselines.

In this paper, eleven representative models covering recurrent, convolutional and transformer architectures are tested uniformly. At the same time, controlled variable ablation further reveals how core hyperparameters alter model performance under separate market regimes. The findings confirm that the efficiency of models strongly depends on hyperparameter sets, providing actionable guidelines for high-frequency microstructure modeling in financial engineering practice. And the existence of bilinear layer, learning rate together with forecast horizon are the top three most important factor in model performance.

2. Related work

Numerous studies leverage deep neural networks for short-term mid-price movement prediction based on LOB data, with most experiments conducted on the FI-2010 benchmark dataset [7]. Early research adopted generic time-series models such as multilayer perceptrons (MLPs), which is used as a benchmark by Tsantekidis et.al. [1]. And then CNN [2] and LSTM [1] networks are used to capture sequential market dynamics. As a pioneering LOB-specific architecture, DeepLOB [3] combines CNNs and LSTMs to extract static order book patterns and temporal evolution sequentially, yet it fails to explicitly model second-order interactions between feature and temporal dimensions. TransLOB [6] introduces transformer self-attention to capture long-range temporal dependencies, though it suffers from excessive parameters and severe overfitting on limited high-frequency data.

To better characterize the intrinsic two-dimensional structure of LOB snapshots, the Temporal Attention-Augmented Bilinear Network (TABL) proposes separated bilinear projections to model cross-dimension interactions between temporal slices and feature slices [11]. Follow-up work further presents bilinear normalization and BiNTABL [12], a tailored normalization module stabilizing the training of bilinear layers by jointly normalizing inputs along two dimensions, overcoming limitations of batch normalization, instance normalization and DAIN [13]. Comprehensive benchmark researches [7-10] reproduce mainstream LOB prediction models under unified experimental settings for fair comparison, but they concentrate solely on architectural performance gaps without systematic parameter analysis.

Existing literature mainly relies on ablation studies to verify the effectiveness of individual network components [14, 15]. As observed in large-scale benchmark reproductions, few investigations systematically explore how critical hyperparameters affect model performance, especially for bilinear architectures tailored for LOB forecasting. To the best of current knowledge, no prior work analyses the mutual influence between hyperparameter configurations and BiN normalization. This paper fills such research gaps by conducting systematic sensitivity experiments on BiNTABL. Further, this paper explores how window size, hidden dimension and learning rate interact with bilinear projection, offering practical configuration guidelines for bilinear LOB prediction models.

3. Method

Figure 1 summarizes the progress of typical deep learning models for LOB prediction, which can be divided into the five groups: feed forward networks, recurrent networks, CNN-based networks, bilinear networks and Transformer-based networks.

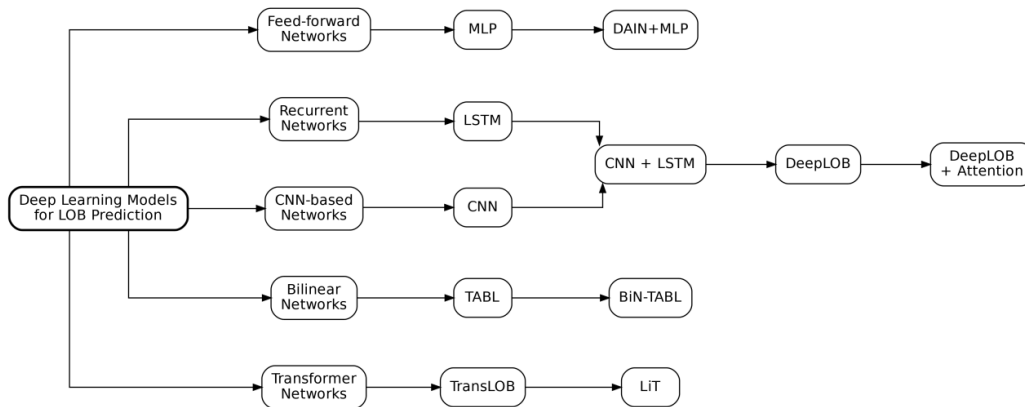


Figure 1. Typical deep learning models for LOB prediction (picture credit: original)

3.1. Feed-forward neural networks

Early deep learning works on LOB prediction were mainly feed-forward neural networks based, such as MLP, which is used as the baseline model [1, 13]. Later, a learnable normalization layer was proposed, which adaptively normalizes the mean and variance of the input features to fit the input data distribution [13]. The basic idea of feed-forward neural networks is that fully connected layers could straightforwardly map hand-crafted or raw LOB features to future price levels.

3.2. Recurrent neural networks

RNN can be used to learn temporal dependence between successive LOB snapshots. In the variant of RNN called LSTMs, the memory cells and gating operation allow information from past observations to enter the current hidden state. The model [1] takes one LOB snapshot at each time step and updates this hidden state as the sequence unfolds. The last hidden state is then fed to a fully-connected and a softmax layer for the 3-class prediction.

3.3. Convolutional and hybrid networks

Tsantekidis et al. [2] proposed a CNN-based approach to represent the LOB as a structured matrix, using convolution layers to capture local patterns across prices and volumes of orders. The CNN-LSTM [4] adds a recurrent component to the model and uses CNN layers to account for the spatial structure of each snapshot and LSTM layers to model its evolution over time. The DeepLOB [3] extends this combination by adding an inception layer that allows the model to capture features on different scales before the temporal stage. The DeepLOB-Attention [16] goes one step further from this architecture by adding an attention component, with the model being able to focus on the more informative temporal representation for making predictions.

3.4. Bilinear networks

Bilinear networks use bilinear mappings to learn interactions between the temporal and feature spaces. TABL [11] uses bilinear mapping layers and temporal attention. It allows the model to weigh the time steps differently while staying somewhat lightweight. BiNTABL [12] adds a normalization stage, BiN, before the TABL layers. The normalization stage aims to mitigate the impact of distributional differences of the inputs and model robustness to such changes.

3.5. Transformer-based models

Transformer-based approaches have also been applied to LOB prediction, in particular to model dependencies that may exist at distant time steps. Whereas recurrent models update a hidden state recursively, Transformers use self-attention to directly relate different positions in the input sequence. A typical Transformer model has an input embedding stage, a stage to encode positions, several Transformer encoder blocks and a prediction layer. Inside each Transformer encoder block, multihead self-attention is combined with a feed-forward network to deal with information coming from the entire sequence. Both TransLOB and LiT [6] follow this general architecture, and use it for high-frequency order book data. The main advantage with respect to CNN-based or LSTM-based models is that attention computations can be parallelized and the same sequential updates of the hidden state do not have to occur, but Transformer also allows for a direct way of modelling long-term temporal dependencies.

4. Experiments & discussions

Most existing works compared the neural networks' models. The influence of the model hyperparameters on network training result is not fully understood, like learning rate, network depth and network parameter number. So, in this paper, two progressive experiments on FI-2010 data for high-frequency tick price forecast are designed. The paper first performs horizontal comparison of mainstream neural networks to select the most suitable base model and then do hyperparameter study of the best performing BiNTABL.

4.1. Dataset

The FI-2010 Limit Order Book Dataset is a widely adopted benchmark for short-term mid-price prediction research on order book dynamics. It contains order book records of five stocks traded on Nasdaq Nordic (Finland) across 10 consecutive trading days in June 2010. Data is split chronologically: the first 7 days for training and the last 3 days for testing without shuffling. FI-2010 contains 149 features, listed as Table 1.

Table 1. FI-2010 structure

Component	Dimensions	Description
Raw LOB features	40	Bid 1–10 & Ask 1–10 (price, volume per level)
Handcrafted microstructure features	104	Spread, cumulative volume, order book slope and other statistics

Table 1. (continued)

Prediction labels	5	3-class mid-price movement (down/stable/up) for horizons of 1,2,3,5,10 events.
-------------------	---	--

4.2. Horizontal comparison of multiple models

This paper presents a comparison of five network architectures including MLP, LSTM, CNN, CNN+LSTM and BiNTABL. The goal is to identify the base model best suited for short-term tick price prediction, and it is selected as the research object for hyperparameter study. Under FI-2010 datasets, all models take 100 ticks temporal historical market data with 40 features as input. The features are limited order book first 10 levels. Furthermore, outputs are the classification results of price movements 5 ticks ahead ($K=5$). Cross entropy loss function, stochastic gradient descent optimizer and 0.01 learning rate are adopted for training. Four evaluation metrics (F1, Precision, Accuracy, Recall) on the test set are recorded for each model. As showed in Table 2, BiNTABL outperforms all baseline models across all four metrics, which confirms it as the target network for the following ablation study.

Table 2. Model performance

Model	F1	Precision	Accuracy	Recall
BiNTABL	86.61%	87.63%	87.07%	85.80%
LSTM	59.79%	59.55%	61.41%	60.21%
MLP	52.92%	53.86%	54.15%	53.88%
CNN	67.62%	67.94%	70.27%	67.44%
CNN+LSTM	69.74%	73.11%	75.23%	67.76%
DAIN	42.86%	50.23%	59.55%	43.40%
DeepLOB	69.04%	70.35%	74.43%	68.06%
TABL	63.61%	66.17%	70.20%	62.11%
TransLOB	59.86%	61.02%	66.67%	59.07%
DeepLOB-Attention	72.47%	74.83%	77.23%	70.86%
LiT	69.53%	72.92%	74.92%	67.59%

4.3. Hyperparameter study on BiNTABL

This paper sets BiNTABL in the Section 4.3 as baseline, which contains 2 BLs and a learning rate of 0.0, 64 as batch size and horizon is 100, and forecast 5 ticks future. A bilinear layer models the interactions between temporal and feature dimensions through two learnable projection matrices, W_1 and W_2 . Matrices W_1 change the height of the hidden feature, while matrices W_2 change width. And W_1 of 2 bilinear layers in the baseline, change height from 40 to 120 to 60 to 3. And W_2 of 2 bilinear layers in the baseline, change width from 1100 to 10 to 5 to 1. The control variable method is adopted: only one parameter is modified per group while all other configurations remain unchanged. The parameter tests are divided into six groups. All models are trained with 30 epochs.

4.4. Changing the number of bilinear layer

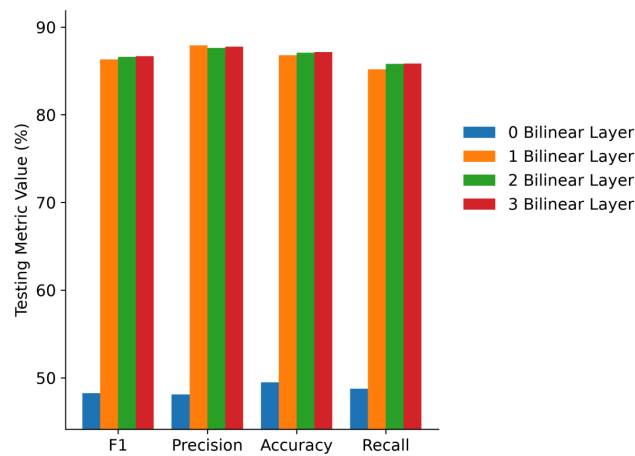


Figure 2. Changing the number of bilinear layer (picture credit: original)

Bilinear layer is the basic feature extractor module of the BiNTABL. It learns representation of contexts by inputting LOB sequences forwards and backwards. Figure 2 shows the effects of the bilinear layer. Removing the bilinear layer leads to an immediate plunge of F1 score by 38.4 percentage points and bringing the model close to random guess level in the 3-class classification. Adding a bilinear layer forms second, the F1 score is improved by 0.0029, and adding a third bilinear layer only leads to an increase of 0.0006 in F1 score. The training–test gap also remains stable as the network becomes deeper. It is -0.0158 with three bilinear layers, compared with -0.0161 for the base line.

These results highlight the central role of bilinear layers in exploiting discriminative information contained in the high-frequency LOB data. Most of the gain of the performance is obtained when the first bilinear layer is introduced; the gain is much smaller once the second bilinear layer has been introduced. This shows that a shallow bilinear architecture is sufficient to capture the useful feature interactions in the input. Additional layers only add little predictive power but also add to the computational burden. At the same time, the similar training–test gaps for the different tested depths provide no definite indication of additional overfitting as the network becomes deeper.

4.5. Changing learning rates

The learning rate controls the size of steps used to update the model parameters as training progresses. It thus has a direct effect on the convergence of the model. Figure 3 illustrates strongly the effect of the learning rate on model performance. This study indeed observe a decrease of about 10 percentage points in the F1 score by reducing the learning rate by one order of magnitude. Further decreasing the learning rate by another order of magnitude renders the model unable to converge. Among the tested settings, a learning rate of 0.01 gives the highest F1 score. When the learning rate is decreased to 0.001, the training F1 score is only 0.3441, which is much lower than that obtained using the baseline setting.

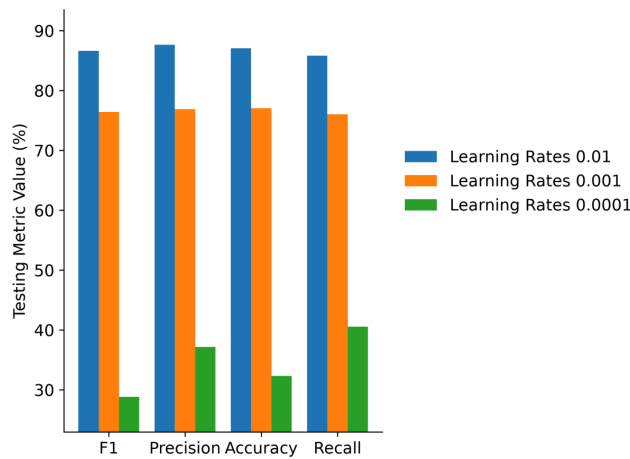


Figure 3. Changing learning rates (picture credit: original)

From these results, it is found that the model works better for a relatively large learning rate. With a learning rate of 0.01, the model makes fairly large parameter updates during the training process. This is important for learning from a rapidly changing LOB data. If the learning rate is too small, the parameter updates are restricted, and the model learns very slowly. Within a fixed number of training epochs, this prevents the model from learning useful patterns and leads to underfitting. Therefore, learning rate selection is critical for this model.

4.6. Changing matrices structure of bilinear layer

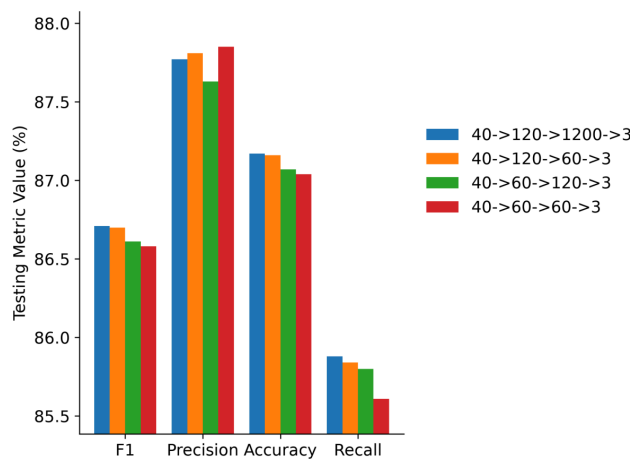


Figure 4. Changing matrices W1 structure (picture credit: original)

The bilinear layer is the main component of the TABL network for multivariate financial time series. Given an input tensor with D features and T time steps, bilinear layer uses two learnable weight matrices, $W1$ and $W2$, to perform linear projections along the feature and temporal dimensions, respectively. The resulting representation is then passed through an element-wise nonlinear transformation function.

Figure 4 shows that changes in the size of $W1$ have little effect on model performance. The F1 scores differ by only 0.0013 across all tested configurations. This small difference suggests that the $W1$ dimension is not a key factor in this model. Increasing the hidden dimension of the first $W1$

layer from 60 to 120 produces a small improvement in F1 score. The effect of W2 is more noticeable, as shown in Figure 5. The best configuration, 100→30→5→1, improves the F1 score by 0.0036 over the baseline. When the first hidden layer is fixed at 30, reducing the second hidden layer from 20 to 10 and then to 5 also improves the F1 score, from 0.8672 to 0.8687 and 0.8697, respectively.

The bar charts show that the size of W1 has only a minor effect on prediction performance. Its dimension can therefore be selected mainly according to computational cost. In comparison, the structure of W2 has a stronger effect on the results. The best performance is obtained with a wider first hidden layer and a narrower second layer. The bottleneck configuration may retain a richer representation in the first layer while compressing less useful information in the second. The results further indicate that a larger second layer does not necessarily lead to better performance. In the tested configurations, reducing the size of the second hidden layer produces a more compact representation and is associated with higher prediction performance.

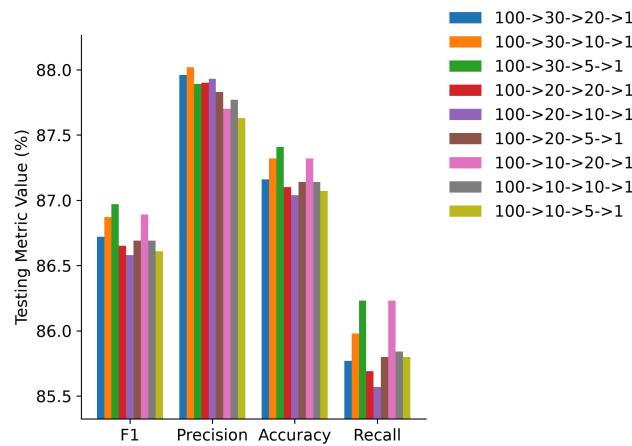


Figure 5. Changing matrices W2 structure (picture credit: original)

4.7. Changing batch size

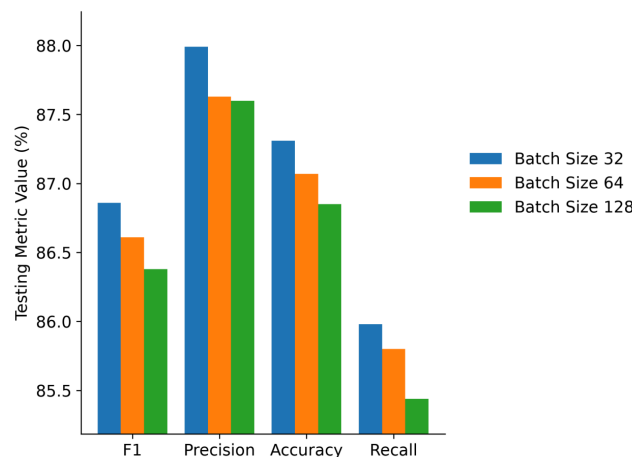


Figure 6. Changing batch size (picture credit: original)

The batch size indicates how many training samples are fed to the neural network before parameters are updated and therefore also has an impact on the optimization time and training time.

As Figure 6 shows, the batch sizes used in the experiments give very small but constant differences in the F1 score. A lower batch size than the one considered in the base experiment (64) of 32 leads to an F1 score improvement of 0.25 percentage points, whereas a higher batch size of 128 leads to a decrease of F1 score by 0.23 percentage points. A batch size of 32 thus yields the best prediction performance of the three batch sizes. This improvement, however, requires a very high computational cost. Training with a batch size of 32 requires 2336 s, compared with 1317 s in the baseline of 64, corresponding to an increase of roughly 77.3%.

Higher performance with small batches reflects the more frequent updates of parameters and larger variation in the gradients between updates. This variation can introduce a kind of stochasticity during optimization, which is potentially useful for generalization. The extra updates also explain the longer training time observed for the smaller batch size. Overall, batch size has a relatively small impact on F1 (as some of the other hyperparameters examined do) but does introduce a clear performance–efficiency trade-off. A batch size of 32 is preferable when prediction performance is foremost, while 64 gives a more practical compromise in terms of performance to computational cost.

4.8. Changing history horizon

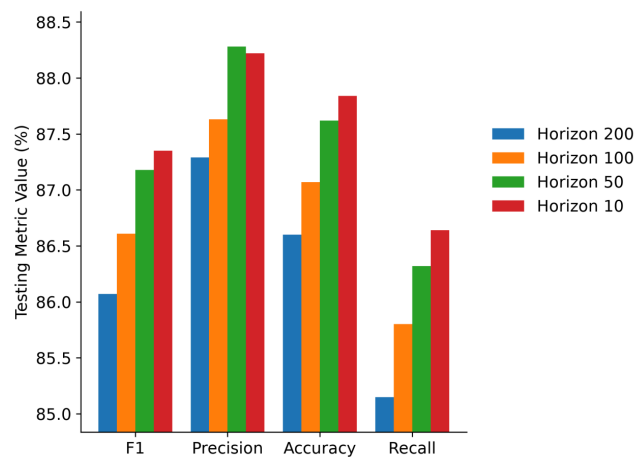


Figure 7. Changing history horizon (picture credit: original)

The history horizon determines the number of previous LOB snapshots input to the model and therefore the amount of historical information available to each prediction step. Figure 7 shows the effect of changing the window of the history horizon. Changing the history horizon between 100 and 10 snapshots has an evident effect on the F1 score. If the history horizon is reduced from 100 to 10 snapshots, the F1 is improved by 0.74 percentage points; if the history horizon is extended to 200 snapshots, the F1 score reduces by 0.54 percentage points. Of the tested configurations, the 10-snapshot window gives the highest F1 score.

Preference for a shorter history indicates that very recent order book states actually have more useful information for this short-term prediction task considered. Information stored in older snapshots may be of less relevance to an evolving market condition, and that including more snapshots in the input sequence also introduces noisy or duplicate observations. It is found that a window of 10 snapshots is already sufficient for the prediction task to capture the recent changes necessary for prediction, and very much more historical observations is not helpful for the additional performance achieved.

4.9. Changing forecast horizon

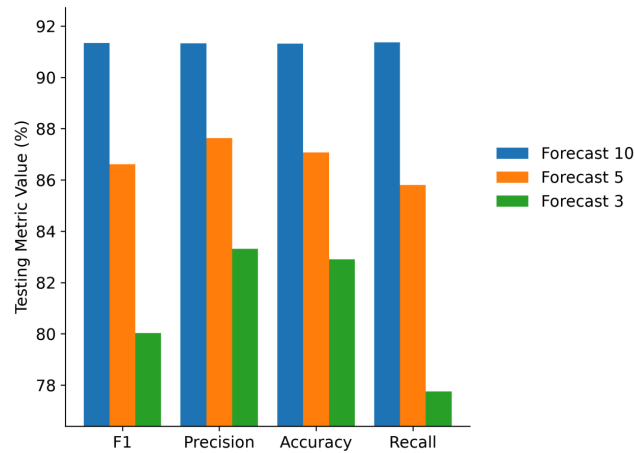


Figure 8. Changing forecast horizon (picture credit: original)

The forecast horizon defines the number of future events in the prediction target. Figure 8 shows a dramatic change in F1 scores over the forecast horizon tested. F1 goes up by 4.7 percentage points when going from predicting 5 events to 10 events, and F1 goes down by 6.6 percentage points when reducing the forecast horizon from 5 to 3 events. Highest F1 score is for 10-event horizon.

The difference between these settings, however, is much larger than that for most of the other hyperparameters. One possible explanation for this is that predictions for only a few events are more sensitive to short-lived fluctuations of the order book. Extending the horizon to 10 events reduces the effect of these short-lived movements and gives a distinct price-movement target. Extending it further does not necessarily give a further increase, because the prediction target itself changes as the forecast gets further into the future. 10 events give the strongest predictive performance.

5. Conclusion

This paper studies the effects of neural network architectures and hyperparameter settings on LOB mid-price prediction using the FI-2010 benchmark. Eleven deep learning models are compared under the same experimental settings. Results show that BiNTABL outperforms other models, which demonstrates the effectiveness of bilinear projection and bilinear normalization of LOB data. Further more, this paper presents ablation experiments by controlling variables. Extensive ablation experiments show that the number of network layer, learning rate, hidden dimensions, batch size, historical input length, and prediction horizon all affect prediction performance. Among these factors, the existence of bilinear layer, learning rate together with forecast horizon are the top three most important factor in model performance. This paper enables LOB prediction works select hyperparameter better. And future works can extend this analysis to higher-volatility markets, such as crypto market, and emerging neural network architectures to further improve prediction performance.

References

- [1] Tsantekidis, A., Passalis, N., Tefas, A., Kannianen, J., Gabbouj, M., & Iosifidis, A. (2017). Using deep learning to detect price change indications in financial markets. In *2017 25th European Signal Processing Conference (EUSIPCO)* (pp. 2511-2515). IEEE.

- [2] Tsantekidis, A., Passalis, N., Tefas, A., Kannianen, J., Gabbouj, M., & Iosifidis, A. (2017). Forecasting stock prices from the limit order book using convolutional neural networks. In *2017 IEEE 19th Conference on Business Informatics (CBI) (Vol. 1)* (pp. 7-12). IEEE.
- [3] Zhang, Z., Zohren, S., & Roberts, S. (2019). Deeplob: Deep convolutional neural networks for limit order books. *IEEE Transactions on Signal Processing*, 67(11), 3001-3012.
- [4] Tsantekidis, A., Passalis, N., Tefas, A., Kannianen, J., Gabbouj, M., & Iosifidis, A. (2020). Using deep learning for price prediction by exploiting stationary limit order book features. *Applied Soft Computing*, 93, 106401.
- [5] Xiao, Y., Ventre, C., Wang, Y., Li, H., Huan, Y., & Liu, B. (2025). LiT: Limit order book transformer. *Frontiers in Artificial Intelligence*, 8, 1616485.
- [6] Wallbridge, J. (2020). Transformers for limit order books. *arXiv preprint arXiv:2003.00130*.
- [7] Ntakaris, A., Magris, M., Kannianen, J., Gabbouj, M., & Iosifidis, A. (2018). Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods. *Journal of Forecasting*, 37(8), 852-866.
- [8] Prata, M., Masi, G., Berti, L., Arrigoni, V., Coletta, A., Cannistraci, I., ... & Bartolini, N. (2024). Lob-based deep learning models for stock price trend prediction: A benchmark study: M. Prata et al. *Artificial Intelligence Review*, 57(5), 116.
- [9] Zhong, M., Lin, Y., & Yang, P. (2025). Representation learning of limit order book: A comprehensive study and benchmarking. *arXiv preprint arXiv:2505.02139*.
- [10] Nagy, P., Frey, S., Li, K., Sarkar, B., Vyetenko, S., Zohren, S., ... & Foerster, J. (2025). LOB-Bench: Benchmarking generative AI for finance--an application to limit order book data. *arXiv preprint arXiv:2502.09172*.
- [11] Tran, D. T., Iosifidis, A., Kannianen, J., & Gabbouj, M. (2018). Temporal attention-augmented bilinear network for financial time-series data analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 30(5), 1407-1418.
- [12] Tran, D. T., Kannianen, J., Gabbouj, M., & Iosifidis, A. (2021). Data normalization for bilinear structures in high-frequency financial time-series. In *2020 25th International Conference on Pattern Recognition (ICPR)* (pp. 7287-7292). IEEE.
- [13] Passalis, N., Tefas, A., Kannianen, J., Gabbouj, M., & Iosifidis, A. (2019). Deep adaptive input normalization for time series forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, 31(9), 3760-3765.
- [14] Hestness, J., Narang, S., Ardalani, N., Diamos, G., Jun, H., Kianinejad, H., ... & Zhou, Y. (2017). Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*.
- [15] Derikvand, T. (2025). Determining the optimal neural network architecture and parameters: A comprehensive review and practical manual for data scientists. *International Journal of Operations Research and Artificial Intelligence*, 1(3), 139-147. <https://doi.org/10.48314/ijorai.v1i3.73>
- [16] Zhang, Z., & Zohren, S. (2021). Multi-horizon forecasting for limit order books: Novel deep learning approaches and hardware acceleration using intelligent processing units. *arXiv preprint arXiv:2105.10430*.