Kronos: A Foundation Model for the Language of Financial Markets

Yu Shi[1,†], Zongliang Fu[2,†], Shuo Chen[1], Bohan Zhao[1], Wei Xu[1], Changshui Zhang[2], Jian Li[1]

[1]Institute for Interdisciplinary Information Sciences, [2]Department of Automation
Tsinghua University
{shi-y23, fzl22, zhaobh23, s-chen25}@mails.tsinghua.edu.cn
weixu@tsinghua.edu.cn, zcs@mail.tsinghua.edu.cn, lapordge@gmail.com

Abstract

The success of large-scale pre-training paradigm, exemplified by Large Language Models (LLMs), has inspired the development of Time Series Foundation Models (TSFMs). However, their application to financial candlestick (K-line) data remains limited, often underperforming non-pre-trained architectures. Moreover, existing TSFMs often overlook crucial downstream tasks such as volatility prediction and synthetic data generation. To address these limitations, we propose **Kronos, a unified, scalable pre-training framework tailored to financial K-line modeling**. Kronos introduces an instance-based tokenizer that discretizes continuous market information into token sequences, preserving both price dynamics and trade activity patterns. We pre-train Kronos using an autoregressive objective on a massive, multi-market corpus of over 12 billion K-line records from 45 global exchanges, enabling it to learn nuanced temporal and cross-asset representations. Kronos excels in a zero-shot setting across a diverse set of financial tasks. On benchmark datasets, Kronos boosts price series forecasting RankIC by 93% over the leading TSFM and 87% over the best non-pre-trained baseline. It also achieves a 9% lower MAE in volatility forecasting and a 22% improvement in generative fidelity for synthetic K-line sequences. These results establish Kronos as a robust, versatile foundation model for end-to-end financial time series analysis. Our code and models are available at `https://github.com/shiyu-coder/Kronos`.

1 Introduction

Foundation Models (FMs) have reshaped AI research in language and vision [1, 2, 3, 4], and are now emerging for temporal data as Time Series Foundation Models (TSFMs) [5, 6, 7]. Within this expanding research landscape, financial markets stand out as a critical and challenging application area for TSFMs, given their inherent data richness, high-frequency observations, and complex, non-stationary temporal market dynamics. Financial markets center around K-line sequences—multivariate time series encoding **O**pen, **H**igh, **L**ow, **C**lose prices, **V**olume, and **A**mount (OHLCVA) over fixed intervals. These sequences constitute a highly compact, information-dense "language" through which market participants interpret price movements, volatility regimes, liquidity shifts, and collective sentiment [8].

However, applying general-purpose TSFMs to financial K-line data faces two key challenges. First, K-line sequences exhibit distinctive characteristics—low signal-to-noise ratios, strong non-stationarities, and complex cross-attribute dependencies [9, 10]—that conflict with the inductive biases of generic TSFMs. Second, finance remains underrepresented in mainstream TSFM research: financial data account for only a minor share of pre-training corpora [11, 12, 7], and critical downstream tasks such as volatility forecasting, sequence generation, and risk management are largely neglected. Conse-

[†]Equal contribution

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

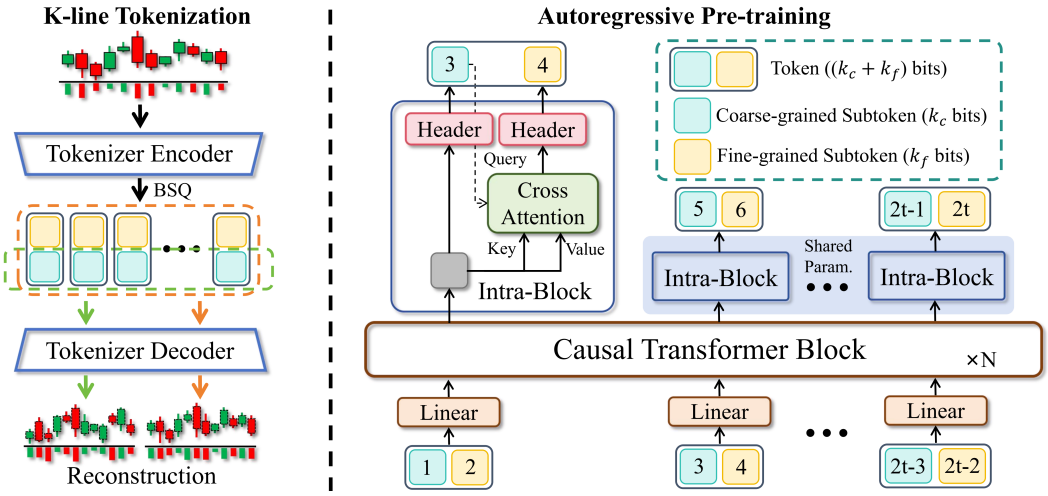


Figure 1: The two-stage framework of Kronos. **(1) K-line Tokenization:** A Transformer-based autoencoder with a dual reconstruction objective quantizes continuous K-line data into a vocabulary of hierarchical discrete tokens, each comprising a coarse and a fine subtoken. **(2) Autoregressive Pre-training:** A decoder-only Transformer is pre-trained to model the temporal dynamics by sequentially predicting the hierarchical subtokens for the next time step, conditioned on the past.

quently, as we empirically demonstrate, general-purpose TSFMs often underperform specialized non-pre-trained models (e.g., iTransformer [13]) and fail to generalize across financial tasks.

To address these shortcomings, we introduce **Kronos, a unified, scalable pre-training framework designed specifically for financial K-line data**. Kronos features a specialized tokenizer that quantizes each multivariate K-line record into structured, dual-component (coarse and fine) tokens, coupled with a tailored autoregressive objective that predicts these subtokens sequentially. This coarse-to-fine prediction scheme allows Kronos to explicitly model multi-scale market dynamics. We conduct large-scale pre-training for a family of Kronos models with varying capacities on an expansive, heterogeneous corpus of 12.1 billion K-line records drawn from over 45 global markets and 7 temporal granularities. Kronos achieves state-of-the-art performance in price series forecasting, significantly outperforming TSFMs and full-shot baselines, while demonstrating strong versatility across volatility forecasting and synthetic K-line generation(see Figure 6 for performance overview).

2 Methodology

Kronos abstracts financial K-line sequences as a discrete language and implements this via a two-phase framework illustrated in Figure 1: **(1) K-line Tokenization** and **(2) Autoregressive Pre-training**.

2.1 K-line Tokenization

Kronos transforms continuous D-dimensional K-line sequences $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ (encoding OHLCVA indicators) into discrete tokens using a Transformer-based autoencoder with encoder E_{enc}, quantizer Q, and decoder E_{dec}. Drawing inspiration from video quantization approaches in generative modeling [14, 15], we adapt Binary Spherical Quantization (BSQ) [16], a variant of Look-up Free Quantization (LFQ) [15], to quantize latent vectors $\boldsymbol{\xi}_t$ into k-bit binary codes $b_t \in \{-1, 1\}^k$ by projecting them onto a set of learnable hyperplanes. See Appendix I (Q2) for the rationale behind this choice. A large bit-width k improves expressivity but yields a vocabulary of size 2^k, inflating autoregressive compute and parameters. Following video quantization [15, 17], we factorize the k-bit code into n subspaces and set $n = 2$ to balance computational cost and latency (see Appendix I, Q3), splitting each token into coarse and fine subtokens $b_t = [b_t^c, b_t^f]$, $b_t^c, b_t^f \in \{-1, 1\}^{k/2}$. This reduces complexity from one prediction over 2^k to two sequential predictions over $2^{k/2}$ entries.

To enforce a coarse-to-fine structure within each token, the tokenizer is trained with:

$$\mathcal{L}_{\text{tokenizer}} = \mathcal{L}_{\text{coarse}} + \mathcal{L}_{\text{fine}} + \lambda \mathcal{L}_{\text{quant}} \tag{1}$$

where: $\mathcal{L}_{\text{coarse}} = \mathbb{E}[\|\mathbf{x} - E_{\text{dec}}(\mathbf{b}^c)\|^2]$ trains coarse subtokens for low-fidelity reconstruction; $\mathcal{L}_{\text{fine}} = \mathbb{E}[\|\mathbf{x} - E_{\text{dec}}(\mathbf{b})\|^2]$ ensures high-fidelity reconstruction using complete tokens; $\mathcal{L}_{\text{quant}}$ is the BSQ quantization loss regularizing continuous-discrete alignment.

2.2 Hierarchical Autoregressive Modeling

We model the tokenized sequence with a causal decoder-only Transformer E_{ar}, estimating

$$p(\mathbf{b}) = \prod_{t=1}^{T} p(b_t \mid \mathbf{b}_{<t}).$$

With hierarchical tokens $b_t = [b_t^c, b_t^f]$, each step factorizes as

$$p(b_t \mid \mathbf{b}_{<t}) = p(b_t^c \mid \mathbf{b}_{<t})\, p(b_t^f \mid \mathbf{b}_{<t}, b_t^c).$$

At time i, subtokens are embedded and fused

$$\mathbf{v}_i = W_{\text{fuse}}\Big(\big[\, e_c(b_i^c);\, e_f(b_i^f)\,\big]\Big), \qquad (2)$$

then processed by E_{ar} to yield a context $\mathbf{h}_t$. The coarse subtoken is predicted via

$$p(b_t^c \mid \mathbf{b}_{<t}) = \text{softmax}(W_c \mathbf{h}_t).$$

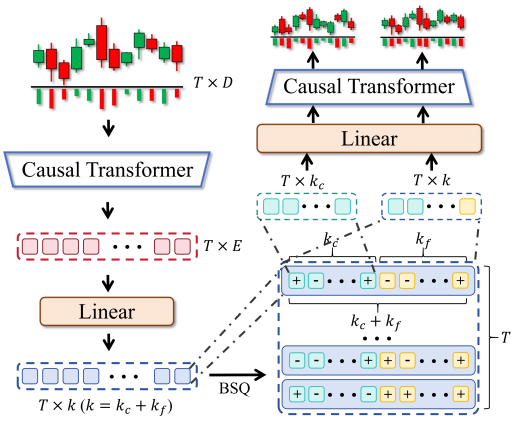


Figure 2: Architecture of the K-line Tokenizer. It employs a Transformer-based autoencoder with a Binary Spherical Quantization (BSQ) layer. A composite loss objective ($\mathcal{L}_{\text{coarse}}, \mathcal{L}_{\text{fine}}, \mathcal{L}_{\text{quant}}$) guides the learning of a coarse-to-fine representational structure within each token.

To predict the fine subtoken, we condition on a sampled coarse prediction $\hat{b}_t^c$ (no teacher-forcing) to reduce exposure bias:

$$\mathbf{h}_t^{\text{upd}} = \text{CrossAttn}\big(q = e_c(\hat{b}_t^c),\, k = v = \mathbf{h}_t\big), \quad p(b_t^f \mid \mathbf{b}_{<t}, \hat{b}_t^c) = \text{softmax}(W_f \mathbf{h}_t^{\text{upd}}).$$

Training minimizes the negative log-likelihood, where $\mathcal{D}$ represents the data distribution.

$$\mathcal{L}_{\text{ar}} = -\mathbb{E}_{\mathbf{b}\sim\mathcal{D}} \sum_{t=1}^{T} \Big[\log p(b_t^c \mid \mathbf{b}_{<t}) + \log p(b_t^f \mid \mathbf{b}_{<t}, \hat{b}_t^c) \Big].$$

2.3 Model Pre-training

Dataset: We curate a large-scale dataset of 12.1 billion K-line observations across 7 sampling frequencies from 45 global exchanges, with a specialized cleaning pipeline described in Appendix C.

Model Training: Informed by the scaling laws observed in LLMs [18], we train three Kronos variants as shown in Table 9. Complete training details are provided in Appendix D.

Inference: We generate future sequences autoregressively with temperature scaling and nucleus sampling [19]. For enhanced accuracy, multiple trajectories can be averaged via Monte Carlo rollouts.

3 Experiments

3.1 Experimental Setup

We evaluate Kronos across five representative tasks: price series forecasting, return forecasting, realized volatility forecasting, K-line generation, and investment simulation. We benchmark against 25 baselines spanning non-pre-trained models (e.g., iTransformer [13]), zero-shot TSFMs (e.g., TimeMOE [7]), econometric models (e.g., GARCH [20]), and generative models (e.g., DiffusionTS [21]). Task details and baselines are provided in Appendix E.

3.2 Main Results

All key experimental findings are summarized in Figure 3, while detailed results are presented in Appendix G. Further discussions and insights are available in Appendix I.

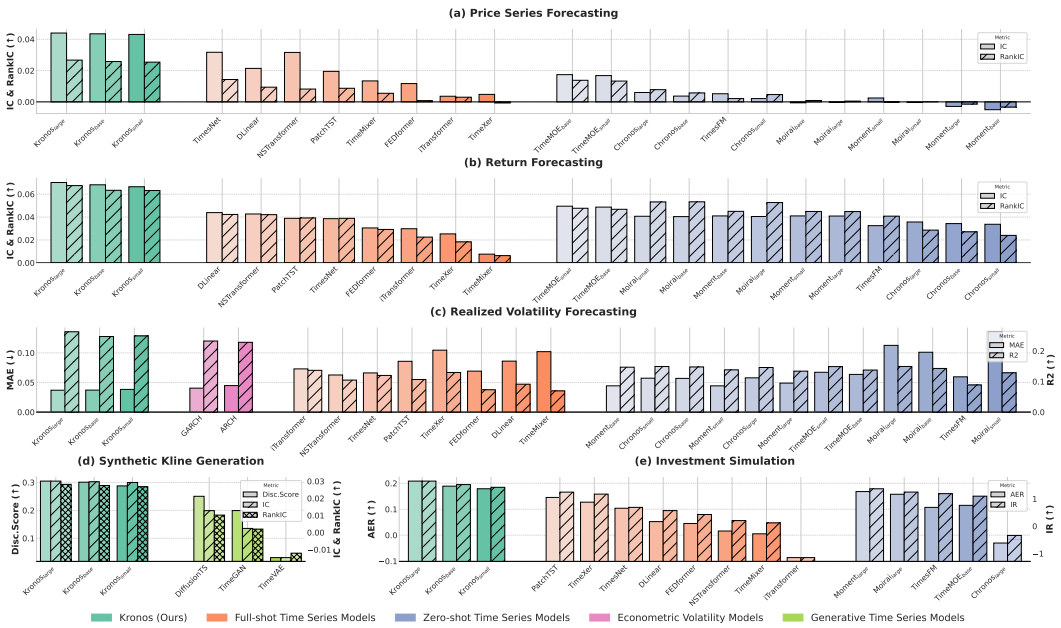


Figure 3: Main experimental results across five representative financial tasks. Subfigures (a-c) show forecasting performance on price series, return, and realized volatility. Subfigure (d) displays generative model performance in terms of fidelity and usefulness. Subfigure (e) presents the investment simulation backtesting results. Full experimental results can be found in Appendix G.

Prediction Tasks: Kronos establishes a new state-of-the-art, boosting the RankIC by 93% over the leading TSFM and by 87% over the best-performing non-pre-trained baseline. Furthermore, it demonstrates strong versatility by achieving a 9% lower MAE in volatility forecasting.

Generative Tasks: Following established evaluation protocols [22], we assess synthetic data quality via diversity (t-SNE/KDE visualization), fidelity (discriminative score), and usefulness (Train-on-Synthetic, Test-on-Real protocol). In Figure 3(d), Kronos achieves 22% improvement in generative fidelity and 176% relative improvement in downstream usefulness.

Investment Simulation: In realistic portfolio construction scenarios on Chinese A-shares, Kronos achieves the highest Annualized Excess Return and Information Ratio, as shown in Figure 3(e), translating predictive accuracy into tangible investment gains.

Scaling Effects: Model performance demonstrates consistent improvement with increased scale across all tasks, confirming the scaling laws for financial TSFM, as clearly illustrated in Figure 3.

3.3 Ablation Studies

Modeling Paradigms: Comparing against continuous-space variants (Direct-AR with MSE, Prob-AR with Student-t mixture), our discrete approach substantially outperforms alternatives. A parallel subtoken prediction variant (Kronos-Parallel) underperforms sequential prediction, validating our conditional dependency modeling. Detailed descriptions are provided in Appendix E.4.

Vocabulary Size: As illustrated in Figure 8, larger vocabularies improve both reconstruction quality and downstream performance, with finer-grained representations reducing quantization error.

Test-Time Scaling: Our probabilistic framework enables performance enhancement at inference through trajectory ensembling. Averaging multiple sampled paths consistently improves IC and RankIC by reducing prediction variance, shown in Figure 9.

4 Conclusion

We introduce Kronos, a foundation model specifically designed for financial K-line sequences. Kronos employs a novel two-stage framework, where an instance-based tokenizer first discretizes continuous market data into hierarchical coarse-to-fine tokens, which are then modeled by a large autoregressive Transformer. Comprehensive empirical evaluations demonstrate that Kronos establishes new state-

of-the-art benchmarks in price series forecasting, as well as in other tasks such as synthetic K-line generation and volatility forecasting, significantly outperforming existing baselines. These results position Kronos as a robust and versatile foundation for a range of applications in quantitative finance.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[5] Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

[6] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *International Conference on Machine Learning*, pages 53140–53164. PMLR, 2024.

[7] Shi Xiaoming, Wang Shiyu, Nie Yuqi, Li Dianqi, Ye Zhou, Wen Qingsong, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. In *ICLR 2025: The Thirteenth International Conference on Learning Representations*. International Conference on Learning Representations, 2025.

[8] Steve Nison. *Japanese candlestick charting techniques: a contemporary guide to the ancient investment techniques of the Far East*. Penguin, 2001.

[9] Lu Zhang and Lei Hua. Major issues in high-frequency financial data analysis: A survey of solutions. *Mathematics*, 13(3):347, 2025.

[10] Ranjai Baidya and Sang-Woong Lee. Addressing the non-stationarity and complexity of time series data for long-term forecasts. *Applied Sciences*, 14(11):4436, 2024.

[11] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

[12] Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. Units: Building a unified time series model. *arXiv e-prints*, pages arXiv–2403, 2024.

[13] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

[14] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[15] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.

[16] Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and video tokenization with binary spherical quantization. *arXiv preprint arXiv:2406.07548*, 2024.

[17] Yuqing Wang, Zhijie Lin, Yao Teng, Yuanzhi Zhu, Shuhuai Ren, Jiashi Feng, and Xihui Liu. Bridging continuous and discrete tokens for autoregressive visual generation. *arXiv preprint arXiv:2503.16430*, 2025.

[18] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[19] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[20] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[21] Xinyu Yuan and Yan Qiao. Diffusion-ts: Interpretable diffusion for general time series generation. *arXiv preprint arXiv:2403.01742*, 2024.

[22] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[23] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

[24] Sabera Talukder, Yisong Yue, and Georgia Gkioxari. Totem: Tokenized time series embeddings for general time series analysis. *arXiv preprint arXiv:2402.16412*, 2024.

[25] Fengyuan Shi, Zhuoyan Luo, Yixiao Ge, Yujiu Yang, Ying Shan, and Limin Wang. Scalable image tokenization with index backpropagation quantization, 2025.

[26] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Hassen, Anderson Schneider, et al. Lag-llama: Towards foundation models for time series forecasting. In *R0-FoMo: Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023.

[27] Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. *arXiv preprint arXiv:2402.02368*, 2024.

[28] Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. *arXiv preprint arXiv:2502.00816*, 2025.

[29] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

[30] Yuanjian Xu, Anxian Liu, Jianing Hao, Zhenzhuo Li, Shichang Meng, and Guang Zhang. Plutus: A well pre-trained large unified transformer can unveil financial time series regularities. *arXiv preprint arXiv:2408.10111*, 2024.

[31] Xueying Ding, Aakriti Mittal, and Achintya Gopal. Delphyne: A pre-trained model for general and financial time series. *arXiv preprint arXiv:2506.06288*, 2025.

[32] Aaron Wheeler and Jeffrey D Varner. Marketgpt: Developing a pre-trained transformer (gpt) for modeling financial time series. *arXiv preprint arXiv:2411.16585*, 2024.

[33] Junjie Li, Yang Liu, Weiqing Liu, Shikai Fang, Lewen Wang, Chang Xu, and Jiang Bian. Mars: a financial market simulation engine powered by generative foundation model. *arXiv preprint arXiv:2409.07486*, 2024.

[34] Deniz Ozenbas et al. Intra-day trading volume patterns of equity markets: A study of us and european stock markets. *International Business & Economics Research Journal (IBER)*, 7(8), 2008.

[35] Raj K Kohli and Theodor Kohers. The week-of-the-month effect in stock returns: The evidence from the s&p composite index. *Journal of economics and finance*, 16(2):129–137, 1992.

[36] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[37] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR, 2020.

[38] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

[39] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[40] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. *arXiv preprint arXiv:2407.13278*, 2024.

[41] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. Timexer: Empowering transformers for time series forecasting with exogenous variables. *arXiv preprint arXiv:2402.19072*, 2024.

[42] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

[43] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

[44] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

[45] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in neural information processing systems*, 35:9881–9893, 2022.

[46] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

[47] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR, 2022.

[48] Robert F Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007, 1982.

[49] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. Timevae: A variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*, 2021.

[50] Xiao Yang, Weiqing Liu, Dong Zhou, Jiang Bian, and Tie-Yan Liu. Qlib: An ai-oriented quantitative investment platform. *arXiv preprint arXiv:2009.11189*, 2020.

[51] Qingren Yao, Chao-Han Huck Yang, Renhe Jiang, Yuxuan Liang, Ming Jin, and Shirui Pan. Towards neural scaling laws for time series foundation models. *arXiv preprint arXiv:2410.12360*, 2024.

[52] Oliver Kim and Robert E Verrecchia. Trading volume and price reactions to public announcements. *Journal of accounting research*, 29(2):302–321, 1991.

[53] Mark J Flannery and Aris A Protopapadakis. Macroeconomic factors do influence aggregate stock returns. *The review of financial studies*, 15(3):751–782, 2002.

[54] Malcolm Baker and Jeffrey Wurgler. Investor sentiment and the cross-section of stock returns. *The journal of Finance*, 61(4):1645–1680, 2006.

[55] Zhi Da, Joseph Engelberg, and Pengjie Gao. In search of attention. *The journal of finance*, 66(5):1461–1499, 2011.

[56] Christian T Brownlees and Giampiero M Gallo. Financial econometric analysis at ultra-high frequency: Data handling concerns. *Computational statistics & data analysis*, 51(4):2232–2245, 2006.

[57] Stephan Rabanser, Tim Januschowski, Valentin Flunkert, David Salinas, and Jan Gasthaus. The effectiveness of discretization in forecasting: An empirical study on neural time series models. *arXiv preprint arXiv:2005.10111*, 2020.

[58] Yongxin Zhu, Bocheng Li, Yifei Xin, Zhihua Xia, and Linli Xu. Addressing representation collapse in vector quantized models with one linear layer. *arXiv preprint arXiv:2411.02038*, 2024.

[59] Benoit Mandelbrot et al. The variation of certain speculative prices. *Journal of business*, 36(4):394, 1963.

[60] Boris Podobnik, Davor Horvatic, Alexander M Petersen, and H Eugene Stanley. Cross-correlations between volume change and price change. *Proceedings of the National Academy of Sciences*, 106(52):22079–22084, 2009.

[61] Warwick J McKibbin, Marcus Noland, and Geoffrey Shuetrim. The global economic effects of trump's 2025 tariffs. *Peterson Institute for International Economics Working Paper*, 0(25-13), 2025.

Appendix

Overview of Appendix

This appendix provides supplementary materials to support the main paper. We detail our data preprocessing pipeline, model and training configurations, experimental setups for all tasks, and present additional results including hyperparameter sensitivity analyses, full result tables, and forecasting showcases. For further details, please see the full paper at `https://arxiv.org/abs/2508.02739`.

A Related Work

A.1 Time Series Tokenization

The recent success of large, token-based models has spurred a growing interest in discretizing continuous time series. This tokenization process is pivotal for adapting such architectures for time series analysis, yet dedicated research in this area remains sparse. Early efforts like Chronos [23] employ scaling and uniform quantization, while TOTEM [24] utilizes a Vector Quantized Variational Autoencoder (VQ-VAE) [14]—a seminal approach that maps encoder outputs to learned discrete latent codes—for codebook-based tokenization. Given this nascent landscape, we draw inspiration from the more mature field of visual tokenization. Beyond the foundational VQ-VAE, innovations include Lookup-Free Quantization (LFQ) [15], achieving high-fidelity reconstruction via an implicit codebook without explicit lookups. Binary Spherical Quantization (BSQ) [16] advances implicit codebooks using spherical projection for an exponentially growing vocabulary, offering bounded quantization error and improved trainability over LFQ. Further, Index Backpropagation Quantization (IBQ) [25] tackles codebook collapse by making all code entries differentiable, enabling stable joint optimization of large-scale codebooks and the visual encoder. While primarily designed for visual data, these methods can also be applied to discretize general multivariate time series.

A.2 General-Purpose Time Series Foundation Models

The paradigm of time series analysis has recently been reshaped by Time Series Foundation Models (TSFMs), drawing inspiration from the success of Large Language Models in leveraging massive pre-trained Transformers. These models are trained on vast, multi-domain corpora—some with over a hundred billion data points—to achieve remarkable zero-shot or few-shot performance on general forecasting benchmarks. This versatility is enabled by diverse architectures, including decoder-only models like Lag-Llama [26], TimesFM [11], Timer [27], Time-MoE [7], and Sundial [28]; encoder-only frameworks like MOMENT [29] and Moirai [6]; encoder-decoder structures such as TimeGPT [5]; and models with modified Transformer blocks for multi-task learning like UniTS [12]. At the input level, they employ generic representations such as direct value patching (e.g., TimesFM [11], MOMENT [29]), value quantization into a fixed vocabulary (e.g., Chronos [23]), or treating consecutive time points as tokens (e.g., Timer [27]). Several of these models also extend to probabilistic forecasting (e.g., Lag-Llama [26], Moirai [6], Chronos [23] and Sundial [28]).

However, the very generality that drives their success on broad benchmarks becomes a limitation in specialized domains. To provide a concrete comparison, we summarize key attributes of prominent TSFMs in Table 1. An important observation from the table is the minuscule proportion of financial data within the pre-training corpora of these general-purpose models, with most dedicating less than 1% of their data to this domain. This data imbalance means that the unique structural properties, non-stationarity, and complex dynamics of financial K-line sequences are largely overlooked or averaged out during pre-training, often resulting in suboptimal performance for financial tasks. To address this fundamental gap in pre-training, we introduce Kronos, a foundation model built from the ground up on a massive corpus composed exclusively of financial K-line data.

A.3 Financial Time Series Foundation Models

The development of foundation models specifically for finance time series is a nascent but rapidly growing field. These efforts can be divided into two main streams. The first focuses on general financial time series, including K-line data. For instance, PLUTUS [30] introduces an invertible embedding and multi-scale temporal attention, pre-trained on massive datasets to uncover market regularities. DELPHYNE [31] is designed explicitly to counteract the negative transfer from non-financial data. While promising, neither of these works has released their code or models, precluding direct empirical comparison. The second stream targets order flow data, where models like MarketGPT [32] and MarS [33] act as generative engines for realistic market simulation. These pioneering efforts validate the value of domain-specific pre-training. However, K-line data possesses broader applicability than order flow, as it is readily available across all markets and suitable for diverse time horizons where order flow data is often inaccessible. Despite its central importance, a versatile and open-source foundation model for K-line analysis remains a notable gap. We introduce Kronos to fill this void, offering a unified, scalable framework designed specifically for financial K-line data.

Table 1: Comparison of time series foundation models. The table highlights architectural choices, tokenization methods, probabilistic forecasting capabilities, and the estimated proportion of financial data in their pre-training corpora.

Model	Architecture	Tokenization	Probabilistic	Financial Data Ratio (Est.)	Primary Domain
Kronos (Ours)	**Decoder-only**	**Discrete (BSQ)**	**Yes**	**100%**	**Financial K-lines**
Sundial [28]	Decoder-only	Continuous	Yes	1.02%	General
Time-MoE [7]	Decoder-only	Continuous	No	<0.01%	General
Moirai [6]	Encoder-only	Continuous	Yes	0.10%	General
MOMENT [29]	Encoder-only	Continuous	No	1.60%	General
Chronos [23]	Encoder-Decoder	Discrete (Quantization)	Yes	0.45%	General
Timer [27]	Decoder-only	Continuous	No	0.03%	General
TimesFM [11]	Decoder-only	Continuous	No	<0.01%	General
UniTS [12]	Encoder-only	Continuous	No	Unknown	General
Lag-Llama [26]	Decoder-only	Continuous	Yes	0.01%	General

Table 2: Frequency-specific parameters for the low-quality data filtering pipeline. Thresholds are adjusted to reflect the distinct dynamics of different time frequencies.

Frequency	Min. Length (bars)	Price Jump Threshold	Max. Consecutive Bars	
			Illiquid	Stagnant
1min	2048	0.10	15	45
5min	1024	0.15	3	10
10min	512	0.15	3	6
15min	512	0.15	2	5
20min	512	0.15	2	5
30min	512	0.20	2	3
40min	256	0.20	1	3
60min	256	0.20	1	3
2H	128	0.25	1	3
4H	128	0.25	1	3
Daily	128	0.30	1	3
Weekly	16	0.50	0	2

B Preliminary

Let D-dimensional vector $\mathbf{x}_t \in \mathbb{R}^D$ denote the K-line observation at discrete time t, comprising D key financial indicators. In this work, we fix the dimension $D = 6$ to represent OHLCVA attributes (Open, High, Low, Close prices, trading Volume, and Amount). The rationale for this input choice is detailed in Appendix I (Q1). Given a historical sequence $\mathbf{x}_{1:T} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$, our objective is to predict the following H observations $\hat{\mathbf{x}}_{T+1:T+H} = (\hat{\mathbf{x}}_{T+1}, \hat{\mathbf{x}}_{T+2}, \ldots, \hat{\mathbf{x}}_{T+H})$.

Rather than operating on raw continuous inputs, Kronos first quantizes each multivariate observation $\mathbf{x}_t$ into a discrete token b_t via a learnable codebook $\mathcal{C}$. Consequently, the original sequence $\mathbf{x}_{1:T} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ is mapped to $\mathbf{b}_{1:T} = (b_1, \ldots, b_T)$. The forecasting task then reduces to an autoregressive token-sequence modeling problem:

$$p(\mathbf{b}_{T+1:T+H} \mid \mathbf{b}_{1:T}) = \prod_{h=1}^{H} p\big(b_{T+h} \mid \mathbf{b}_{1:T+h-1}\big). \tag{3}$$

Such a discrete formulation is inherently scalable and naturally extends to other tasks that can be framed generatively, such as synthetic data generation and volatility forecasting.

Algorithm 1 Low-Quality Segment Filtering Pipeline

Input: Raw K-line series S_{raw}, Parameter set Θ for a given frequency (from Table 2)
Output: A set of clean K-line segments $\mathcal{C}$

function FILTERLOWQUALITYSEGMENTS(S_{raw}, Θ)
  $\mathcal{C} \leftarrow \emptyset$ ▷ Initialize the set of final clean segments
  $\mathcal{S}_{initial} \leftarrow$ PartitionByPriceJumps($S_{raw}, \Theta_{\text{jump}}$) ▷ Split by structural breaks
  for all segment S in $\mathcal{S}_{initial}$ **do**
    $M_{illiquid} \leftarrow$ FlagConsecutiveIlliquid($S, \Theta_{\text{illiquid}}$) ▷ Identify illiquid periods
    $M_{stagnant} \leftarrow$ FlagConsecutiveStagnant($S, \Theta_{\text{stagnant}}$) ▷ Identify stagnant periods
    $M_{invalid} \leftarrow M_{illiquid} \vee M_{stagnant}$ ▷ Combine masks for all invalid points
    $\mathcal{S}_{clean} \leftarrow$ ExtractValidSubsequences($S, M_{invalid}$) ▷ Split segment on invalid boundaries
    for all subsequence S_{sub} in $\mathcal{S}_{clean}$ **do**
      if Length(S_{sub}) $\geq \Theta_{\text{min\_len}}$ **then**
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{S_{sub}\}$ ▷ Add valid, sufficiently long segment
      end if
    end for
  end for
  return $\mathcal{C}$
end function

C Dataset Details

C.1 Data Preprocessing and Cleaning

This section details the preprocessing and cleaning pipeline applied to the large-scale K-line dataset used for pre-training. The dataset is aggregated from over 40 exchanges across more than 30 countries, comprising a diverse range of asset classes at multiple temporal frequencies (1-minute to weekly). A statistical overview is provided in Table 13. The integrity of large-scale pre-training is contingent upon high-quality input data. Raw K-line series, however, are frequently contaminated by artifacts stemming from low liquidity, price limits, or data feed errors. To mitigate the impact of such issues, we implement a rigorous, two-stage pipeline designed to process missing values and filter out low-quality data segments.

C.1.1 Missing Value Processing

We employ a field-specific strategy to handle missing values, which are typically represented as 'NaN' (Not a Number) or 'Inf' (Infinity).

- **Price Fields (Open, High, Low, Close):** For price-related fields, we treat missing values as hard boundaries. Inspired by TimeMOE [7], we partition the time series into contiguous, valid subsequences at each occurrence of a missing price value. This approach ensures that each resulting segment maintains its internal temporal integrity without unwarranted imputation.
- **Volume and Amount Fields:** In contrast, for volume and amount fields, which primarily serve as auxiliary covariates, we impute missing values with zero. To enhance model robustness to sparse or unavailable volumetric data, we introduce a regularization technique: during training, both volume and amount are randomly set to zero for 5% of the input samples. This encourages the model to learn to make effective predictions from price information alone.

C.1.2 Low-Quality Segment Filtering

Beyond addressing discrete missing values, our pipeline systematically identifies and removes entire segments of low-quality data. This is achieved through a multi-stage filtering process where tolerance thresholds are dynamically adjusted according to the data's temporal frequency, as detailed in Table 2. The procedure, formalized in Algorithm 1, consists of the following steps:

- **Structural Break Segmentation.** The initial filtering stage partitions the series based on significant price discontinuities. We identify these breaks by calculating the relative price jump between the previous bar's close and the current bar's open ($|\text{open}_t/\text{close}_{t-1} - 1|$). If this jump exceeds a frequency-specific threshold, the sequence is split. This step effectively isolates artifacts arising from events such as contract rollovers, stock splits, or dividend distributions.

Table 3: Hyperparameter configurations for the Kronos model series. All models are trained with the AdamW optimizer.

Model	FFN Dropout	Residual Dropout	Attention Dropout	Token Dropout	Learning Rate	Weight Decay
Kronos$_{small}$	0.25	0.25	0.1	0.1	1×10^{-3}	0.01
Kronos$_{base}$	0.20	0.20	0.0	0.0	5×10^{-4}	0.05
Kronos$_{large}$	0.00	0.00	0.0	0.0	2×10^{-4}	0.10

- **Filtering of Illiquid Periods.** Within each segment from the previous step, we screen for periods of sustained illiquidity. A bar is deemed illiquid if its trading volume is zero or near-zero. If the number of consecutive illiquid bars exceeds a frequency-dependent threshold, the corresponding period is flagged as invalid.
- **Filtering of Price Stagnation.** We apply a similar method to filter periods of price stagnation, where the closing price remains constant over an extended duration. This often indicates potential data feed issues or market inactivity. If the length of a stagnant streak surpasses its frequency-specific tolerance, it is also flagged as an invalid period.
- **Final Segment Validation.** After flagging all illiquid and stagnant periods, the initial segments are further split at the boundaries of these flagged regions. Finally, only the resulting sub-segments that meet the frequency-specific minimum length requirement ($\Theta_{\text{min\_len}}$ in Table 2) are retained for the final pre-training dataset. This ensures each sample is sufficiently long to support meaningful model learning.

D Implementation Details

In this section, we provide further details on the implementation of Kronos, covering data preprocessing, model architecture, and configurations for training and inference.

D.1 Input Preprocessing

Each input K-line sequence $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathbb{R}^D$, is normalized in a two-step procedure before being passed to the tokenizer. First, we apply z-score normalization independently to each of the D feature dimensions (e.g., Open, High, Low, Close, Volume and Amount). Second, to mitigate the potential impact of extreme outliers on training stability, the normalized values are clipped to the range $[-5, 5]$. This process ensures that all input features have a consistent scale while preserving the model's robustness against anomalous data points.

D.2 Model Architecture

Temporal Embeddings. To capture cyclical patterns inherent in financial markets, such as intraday, weekly, and monthly seasonality [34, 35], we incorporate learnable temporal embeddings. We extract five time-related features for each K-line entry: minute-of-day, hour-of-day, day-of-week, day-of-month, and month-of-year. Each feature is mapped to a dense vector via a dedicated embedding layer. These temporal embeddings are summed and then added to the input representation of each corresponding token, providing the model with explicit temporal context.

K-line Tokenization. The tokenizer's autoencoder is designed to be lightweight. The encoder and decoder each consist of 3 Transformer layers, with a model dimension of 256, a feed-forward network dimension of 512, and 4 attention heads. Following the official open-source implementation of BSQ, we configure the key quantization hyperparameters as follows: a commitment weight $\beta = 0.05$, entropy penalty weights $\gamma_0 = 1.0$ and $\gamma = 1.1$, and an overall entropy scale $\zeta = 0.05$. The balancing hyperparameter λ for the quantization loss in our objective is set to 1. The quantization group size is set to 5 for tractable entropy computation.

Transformer Block Architecture. To encode the sequential nature of the data, we employ causal self-attention with Rotary Position Embeddings (RoPE) [36], which injects relative positional information. The attention operation is formulated as follows:

$$\text{Attention}(Q, K, V) = \text{CausalMask}\left(\frac{Q'(K')^T}{\sqrt{d_k}}\right)V \tag{4}$$

`https://github.com/zhaoyue-zephyrus/bsq-vit`

Table 4: Inference hyperparameters for downstream tasks. T denotes the temperature for sampling, Top-p controls nucleus sampling, and N is the number of inference samples generated for each test instance.

Task	Temperature (T)	Top-p	Number of Inference Samples (N)
Price Series Forecasting	0.6	0.90	10
Return Forecasting	0.6	0.90	10
Realized Volatility Forecasting	0.9	0.90	1
Synthetic K-line Generation	1.0	0.95	1
Investment Simulation	0.6	0.90	10

where d_k is the dimension of the key vectors, and CausalMask prevents attending to future positions. The matrices Q' and K' represent the original query and key matrices with RoPE transformations applied. Furthermore, we adopt the Pre-Layer Normalization (Pre-LN) [37] to improve training stability, specifically utilizing Root Mean Square Layer Normalization (RMSNorm) [38] for its computational efficiency and performance.

D.3 Training Configuration

The training hyperparameters are carefully selected for each model size to ensure a stable pre-training process. As model scale increases, we decrease the peak learning rate and dropout probability while increasing the weight decay. We employ the AdamW optimizer [39] and a cosine learning rate schedule with a linear warm-up phase. The learning rate warms up from 10% of its peak value over the first 15,000 training steps. Table 3 details the specific hyperparameter settings for each model variant.

D.4 Inference Hyperparameters

The generation process at inference time is controlled by temperature scaling (T) and nucleus (top-p) sampling. The optimal choice of these hyperparameters is task-dependent. For example, forecasting tasks generally benefit from lower temperatures to reduce randomness, whereas generative tasks may require higher temperatures to increase diversity. A detailed analysis of hyperparameter sensitivity is available in Appendix F.1. The inference hyperparameters used for each task are detailed in Table 4.

D.5 Pre-training Data Rebalancing

The raw pre-training corpus exhibits a natural imbalance across asset classes, with equities being more prevalent than cryptocurrencies, futures, and foreign exchange (forex) assets. To prevent potential underfitting on these less-represented classes, we apply strategic resampling to the training data. Specifically, we increase the sampling weights for data from crypto, futures, and forex markets. This rebalancing ensures the model gains more balanced exposure to the diverse dynamics across different financial instruments.

E Experimental Design and Implementation

In this section, we present the comprehensive experimental design and implementation for the evaluation of Kronos. We begin by outlining the core evaluation tasks and their corresponding metrics. Next, we introduce the suite of baseline models used for comparison and detail their specific configurations. Finally, we provide a detailed account of the implementation for each experimental task, covering the datasets, parameters, and specific protocols used in our evaluation.

E.1 Tasks and Evaluation Metrics

We evaluate Kronos on a diverse set of tasks that are central to quantitative finance. The tasks and their respective evaluation metrics are as follows:

- **Price Series Forecasting:** We assess the model's ability to predict future price series. Performance is measured by the Information Coefficient (IC) and Rank Information Coefficient (RankIC) between the predicted and actual values.
- **Return Forecasting:** Similarly, we evaluate the model's proficiency in forecasting asset returns, also using IC and RankIC as the metrics to gauge predictive accuracy.

- **Realized Volatility Forecasting:** We use the model's high-frequency forecasts to estimate realized volatility. The accuracy of these estimations is evaluated using Mean Absolute Error (MAE) and the Coefficient of Determination (R^2).
- **Synthetic K-line Generation:** Following established practices in time series generation [22], we assess the quality of synthetic K-line sequences from three perspectives: *diversity*, assessing how well the generated samples cover the distribution of the real data; *fidelity*, assessing whether synthetic samples are indistinguishable from real data; and *usefulness*, evaluating if synthetic data is as effective as real data for downstream predictive tasks (i.e., the Train-on-Synthetic, Test-on-Real paradigm).
- **Investment Simulation:** To measure the practical applicability of the model's forecasts, we perform backtesting simulations. The performance is reported using Annualized Excess Return (AER) and Information Ratio (IR).

E.2 Baselines and Configurations

For a rigorous evaluation, we benchmark Kronos against a comprehensive suite of 25 baseline models. These models are selected from prior works (e.g., [7, 40, 21]) to represent a diverse range of established and state-of-the-art approaches across different paradigms. They are organized into four distinct groups:

- **Full-shot Time Series Models:** This category consists of modern, non-pre-trained time series models that are trained from scratch on the specific downstream task. It includes TimeXer [41], TimesNet [42], TimeMixer [43], PatchTST [44], Non-stationary Transformer (NSTransformer) [45], DLinear [46], FEDformer [47], and iTransformer [13].
- **Zero-shot Time Series Models:** This group comprises large-scale, pre-trained foundation models designed for general time series analysis. The baselines are TimeMOE [7], Moirai [6], TimesFM [11], Moment [29], and Chronos [23], which we evaluate in a zero-shot setting.
- **Econometric Volatility Models:** For the volatility forecasting task, we include established econometric models as specialized baselines, namely ARCH [48] and GARCH [20].
- **Generative Time Series Models:** For the K-line generation task, we compare Kronos against models representing three mainstream generative architectures: DiffusionTS (diffusion-based) [21], TimeVAE (VAE-based) [49], and TimeGAN (GAN-based) [22].

Full-shot Time Series Models. For all non-pre-trained deep learning models, we employ a composite loss function that combines Mean Squared Error (MSE) with an Information Coefficient (IC) term. We find this objective empirically improves predictive performance on financial tasks compared to using MSE alone, as it directly rewards the model for capturing the directional accuracy of price movements. The loss function is defined as:

$$\mathcal{L} = \frac{1}{M \cdot H} \sum_{i=1}^{M} \sum_{j=1}^{H} (y_{i,j} - \hat{y}_{i,j})^2 - \lambda \cdot \frac{1}{M} \sum_{i=1}^{M} \text{IC}(y_i, \hat{y}_i) \tag{5}$$

where y_i and $\hat{y}_i$ are the true and predicted sequences for the i-th feature, respectively, M is the number of features, H is the prediction horizon, and λ is a balancing hyperparameter, set to 4 in our experiments.

All models are trained with a batch size of 256 and an Adam optimizer with a learning rate of 5×10^{-4}. We train for a maximum of 12 epochs, employing an early stopping mechanism with a patience of 3 epochs based on the validation loss. For each model, we test two sets of hyperparameters corresponding to smaller and larger model sizes to ensure a fair and robust comparison. The configuration that yields the best performance on the validation set is selected for final evaluation. For DLinear, instead of varying model dimensions, we evaluate two configurations based on its 'individual' parameter: one where a single linear layer is shared across all variates ('individual=False') and another where a separate linear layer is trained for each variate ('individual=True'). The specific hyperparameter configurations are detailed in Table 5.

Econometric Volatility Models. For the specialized volatility forecasting baselines, we follow standard econometric practices for model selection.

- **ARCH**: For each time series, we fit ARCH models with lag orders $p \in \{1, 2, 3\}$. The model with the lowest Bayesian Information Criterion (BIC) is selected for forecasting. The BIC penalizes model complexity, helping to prevent overfitting.
- **GARCH**: We perform a grid search over the lag orders for both the autoregressive term (p) and the moving average term (q), with $p, q \in \{1, 2, 3\}$. Similar to ARCH, the GARCH(p,q) model with the minimum BIC is chosen as the final model for that series.

E.3 Task Implementation Details

Below, we describe the specific setups for each of our evaluation tasks.

Table 5: Hyperparameter configurations for the baseline models. Values for the two evaluated sets are separated by a slash (/). We detail the number of layers, model dimension ($\mathbf{d}_{\text{model}}$), feed-forward dimension ($\mathbf{d}_{\text{ff}}$), and the number of attention heads.

Model	Layers	$\mathbf{d}_{\text{model}}$	$\mathbf{d}_{\text{ff}}$	Heads
TimeXer	3 / 5	128 / 256	256 / 512	4 / 8
TimesNet	3 / 5	128 / 256	256 / 512	—
TimeMixer	3 / 5	128 / 256	256 / 512	4 / 8
PatchTST	3 / 5	128 / 256	256 / 512	4 / 8
NSTransformer	2 / 3	128 / 256	256 / 512	4 / 8
FEDformer	2 / 3	128 / 256	256 / 512	4 / 8
iTransformer	3 / 5	128 / 256	256 / 512	4 / 8

Forecasting Task Setup The pre-training data for Kronos extends up to June 2024. Consequently, our test period for all tasks begins in July 2024 to ensure a strict temporal separation between training and evaluation. We select a diverse set of assets and K-line frequencies to rigorously test model generalization.

Assets We evaluate on three major asset classes:

- **Stocks:** To test both in-distribution and out-of-distribution generalization, we use data from nine global stock exchanges.
  - *In-distribution exchanges:* Shanghai (XSHG), NASDAQ (XNAS), Japan (XJPX), India (XNSE), Korea (XKRX), and Hong Kong (XHKG).
  - *Out-of-distribution exchanges:* Indonesia (XIDX), Malaysia (XKLS), and Taiwan (XTAI).
- **Cryptocurrency:** All spot trading pairs available on the Binance exchange.
- **Forex:** A comprehensive dataset of over 1,000 foreign exchange pairs.

For cryptocurrency and forex assets, we intentionally exclude volume and amount fields, providing only the OHLC price series. This setup tests the models' ability to make predictions based solely on price dynamics, a common scenario where reliable volume data is unavailable.

Frequencies and Horizons We test on a range of K-line frequencies, again including both in-distribution and out-of-distribution settings. For each frequency, we define look-back and forecast horizons that are relevant to practical applications in quantitative finance. These settings are detailed in Table 6.

Table 6: Look-back and forecast horizon settings for each K-line frequency in the forecasting tasks.

Frequency	Look-back Window	Forecast Horizon
5min	480	96
10min	240	48
15min	160	32
20min	120	24
40min	90	24
1-hour	80	12
2-hour	60	12
4-hour	90	18
Daily	40	12

Metric Calculation Details

- **Price Series Forecasting:** For each sample, the IC and RankIC are calculated between the predicted and true series for each of the four price channels (Open, High, Low, Close). The final reported metrics are the average across these four channels.
- **Return Forecasting:** We define the predicted return $\hat{r}$ based on the last value of the predicted close price sequence $\hat{p}_{t+H}$ and the last value of the historical close price sequence p_t:

$$\hat{r} = \frac{\hat{p}_{t+H}}{p_t} - 1 \tag{6}$$

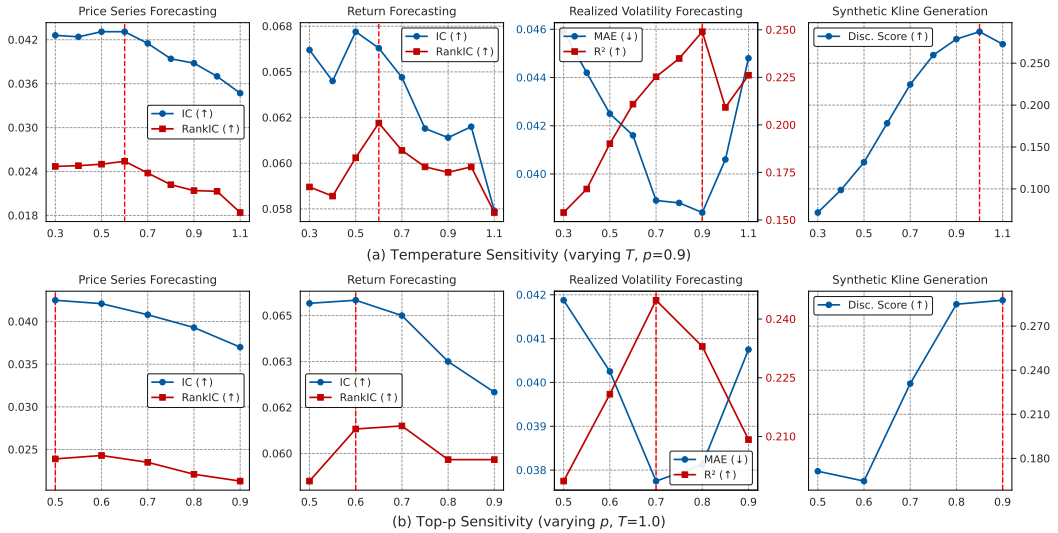


Figure 4: Sensitivity analysis of Kronos's performance on downstream tasks with respect to inference sampling hyperparameters. (a) Varying temperature T while keeping top-$p = 0.9$ fixed. (b) Varying top-p while keeping temperature $T = 1.0$ fixed. Optimal values, indicated by red dashed lines, are task-dependent, highlighting different requirements for precision versus diversity.

The IC and RankIC are then computed between the vector of predicted returns and the vector of actual returns for all samples within a given asset class and frequency.

- **Realized Volatility Forecasting:** We estimate the realized volatility from a high-frequency price series. Using the model's predicted closing prices $\{\hat{p}_i\}_{i=1}^{H}$ over the forecast horizon, the realized volatility is calculated as the sum of squared log returns:

$$\hat{\sigma}^2 = \sum_{i=1}^{H-1} \left(\log(\hat{p}_{i+1}) - \log(\hat{p}_i)\right)^2 \tag{7}$$

We then compute the Mean Absolute Error (MAE) and Coefficient of Determination (R^2) between the predicted and actual realized volatilities across all samples.

Synthetic K-line Generation Setup

Datasets and Generation Parameters We use data from two stock exchanges (in-distribution XSHG and out-of-distribution XTAI), as well as the cryptocurrency and forex datasets. We evaluate generation on two frequencies: 15-minute and daily. For the 15-minute frequency, we use a look-back window of 120 and generate a future sequence of length 96. For the daily frequency, the look-back is 96 and the generation horizon is 35. For each asset-frequency pair, we generate 6,000 synthetic sequences for evaluation.

Evaluation Metrics

- **Discriminative Score:** To assess the fidelity of the generated data, we employ a post-hoc LSTM-based classifier to distinguish between real and synthetic sequences. The classifier consists of a single LSTM layer with a hidden dimension of 32. For training, we construct a balanced dataset of 6,000 samples (3,000 real, 3,000 synthetic) and a held-out test set of the same size and composition. The model is trained for 20 epochs with a batch size of 64, using the Adam optimizer (learning rate = 0.0005) and the binary cross-entropy (BCE) loss function. The Discriminative Score is defined as the classification error on the test set. A score approaching 0.5 indicates higher fidelity, signifying that the classifier struggles to differentiate generated data from real data.
- **Usefulness (TSTR):** To measure the practical usefulness of the synthetic data, we adopt the Train-on-Synthetic, Test-on-Real (TSTR) methodology. We train a post-hoc LSTM prediction model to forecast a future K-timestep window given a historical one. This model comprises two LSTM layers with a hidden dimension of 64. It is trained exclusively on 6,000 generated synthetic sequences for 20 epochs using the Adam optimizer (learning rate = 0.001) and a batch size of 64, with the Mean Squared Error (MSE) loss as the objective function. The look-back and horizon windows are set to (80,

16) for 15-minute data and (30, 5) for daily data, respectively. The trained model is then evaluated on the original, real test data. The final usefulness score is reported as the average Information Coefficient (IC) and Rank Information Coefficient (RankIC) of the predicted price series.

Investment Simulation Setup To evaluate the practical profitability of Kronos and other baselines in real-world markets, we conduct an investment simulation on the Chinese A-share market. For simplicity, regarding the Zero-shot Time Series Models, we only select the largest-sized model from each family for comparison.

Data Our empirical analysis utilizes daily market data for the Chinese A-share market, sourced from the Qlib platform [50], an open-source framework for quantitative finance. To promote transparency and reproducibility, we apply no additional filtering or preprocessing to the data, using it in its original, unprocessed state. Furthermore, we conduct all backtesting simulations within the Qlib framework. This approach leverages its integrated backtesting engine to ensure a standardized and consistent evaluation protocol for all models under review.

Strategy We employ the top-k/drop-n portfolio construction strategy. On each trading day, all stocks in the investment universe are ranked based on their predicted return signal. An equal-weight portfolio is formed by taking long positions in the top k stocks. To manage turnover and trading costs, a maximum of n stocks are bought or sold daily, and a minimum holding period of 5 days is enforced for all positions.

Signal and Backtest The predictive signal is formulated as an expected return derived from a multi-step price forecast over a horizon of H days. This signal generation pipeline is applied uniformly to all models under evaluation, including Kronos and the baselines, to ensure a fair comparison. For any given stock on trading day t, a sequence of forecasted closing prices for the subsequent H days, denoted as $\{\hat{p}_{t+i}\}_{i=1}^{H}$, is first generated by the respective model. The signal, which we term the H-day average expected return ($R_{t\to t+H}$), is then calculated by comparing the arithmetic mean of these forecasted prices to the current closing price p_t:

$$R_{t\to t+H} = \frac{\left(\frac{1}{H}\sum_{i=1}^{H}\hat{p}_{t+i}\right) - p_t}{p_t} \tag{8}$$

In our experiments, we set the forecast horizon to $H = 10$. All price forecasts are generated using daily K-line data with a 90-day look-back window. This methodology is designed to produce a robust signal by averaging the forecasted price path, thereby mitigating the influence of short-term prediction noise and capturing the underlying trend more effectively.

Backtests are performed on the constituents of the CSI 300 and CSI 800 indices. These indices are chosen as they represent two key segments of the Chinese A-share market: the CSI 300 comprises large-cap, highly liquid stocks, while the CSI 800 provides broader market coverage by including both large- and mid-cap stocks. This allows for a comprehensive assessment of the model's performance across different market segments.

Parameters and Costs For the CSI 300 index, we set $k = 50$ and $n = 5$. For the broader CSI 800 index, we set $k = 200$ and $n = 10$. The relatively large portfolio sizes are chosen to ensure diversification and produce more stable backtesting results, reducing the influence of idiosyncratic stock movements. To ensure a realistic performance assessment, a conservative transaction cost of 0.15% is applied to each trade.

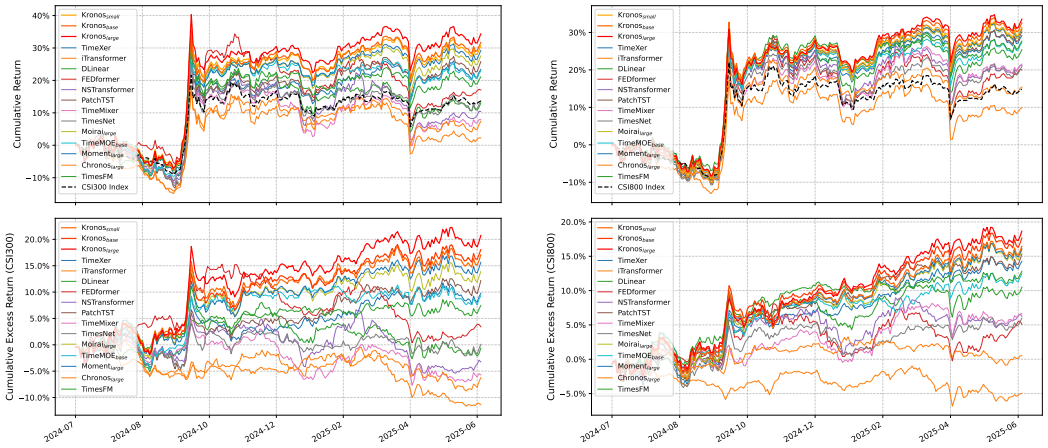
E.4 Details of Ablation Study Baselines

To investigate the architectural choices of Kronos, we design three baseline variants for our ablation study (Table 10). Each variant targets a different modeling paradigm, allowing us to isolate the benefits of our proposed discrete, sequential framework. Below we provide a detailed description of each model.

Direct-AR. This model serves as a standard autoregressive forecasting baseline in the continuous space. Given a sequence of input features $\{x_1, \ldots, x_T\}$, each feature vector $x_t \in \mathbb{R}^D$ is first mapped to a higher-dimensional embedding via a linear projection. The sequence of embeddings is then processed by a Transformer decoder backbone. The model is trained to directly predict the value of the next time step, $\hat{x}_{T+1}$, from the historical context. The training objective is to minimize the Mean Squared Error (MSE) between the predicted and ground-truth values. This approach represents the most common regression-based formulation for time series forecasting.

Prob-AR. This is a probabilistic forecasting model operating in the continuous space. Following established practices [51], instead of a point estimate, Prob-AR predicts the parameters of a probability distribution for the next time step. We use a mixture of four Student-t distributions to model the predicted distribution. The probability density function (PDF) for a random variable x following a single Student-t distribution is:

$$p(x|\nu,\mu,\sigma) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{\pi\nu}\sigma}\left(1 + \frac{1}{\nu}\left(\frac{x-\mu}{\sigma}\right)^2\right)^{-\frac{\nu+1}{2}} \tag{9}$$



(a) CSI300 Index (b) CSI800 Index

Figure 5: Cumulative return curves of backtest using signals generated by different models.

where $\nu > 0$, $\mu \in \mathbb{R}$, and $\sigma > 0$ are the degrees of freedom, location, and scale parameters, respectively, and $\Gamma(\cdot)$ is the gamma function. The model employs independent linear layers to predict the parameters for each of the four components—degrees of freedom (ν_k), location (μ_k), scale (σ_k), and mixture weights (w_k). To ensure parameter validity, a softplus transformation is applied to ν_k and σ_k to enforce positivity, and a softmax function is applied to the weights w_k to ensure they form a valid probability distribution. The model is trained by minimizing the Negative Log-Likelihood (NLL) of the true value under the predicted mixture distribution.

Kronos-Parallel. This variant is a direct ablation of the sequential subtoken generation mechanism within Kronos. While it shares the same input quantization and discrete prediction space as Kronos, it removes the intra-block module. After the Transformer backbone produces a context vector from the input history, a single prediction head is used to concurrently predict the logits for both subtokens of the next time step. The training objective is the sum of the cross-entropy losses for each subtoken, optimized jointly.

E.5 Experimental Environment

All experiments are conducted within a Kubernetes (k8s) cluster. For all computational tasks, we utilize three identical pods. Each pod is provisioned with a dedicated set of resources comprising 96 CPU cores (Intel Xeon Gold 6330 @ 2.00 GHz), 200 GB of system memory (RAM), and eight NVIDIA GeForce RTX 4090D GPUs. This configuration provides a total of 24 GPUs, which are collectively employed for model training and all subsequent evaluations.

The software environment is containerized and standardized across all pods. The primary components and their versions are detailed below:

- **Operating System:** Ubuntu 24.04.1 LTS
- **Software versions:** Python 3.13.2, PyTorch 2.7.0, NumPy 1.26.2, Pandas 2.2.2, Matplotlib 3.9.3, Hugging Face Hub ('huggingface_hub') 1.57.4

F Additional Results

F.1 Impact of Inference Sampling Hyperparameters

The autoregressive generation process of Kronos is governed by sampling strategies that introduce controlled stochasticity, namely temperature scaling (T) and top-p (nucleus) sampling. The choice of these hyperparameters can significantly influence model performance on different downstream tasks. To provide guidance on their optimal settings, we conduct a sensitivity analysis. Figure 4 illustrates the performance of Kronos across our four main tasks while varying one hyperparameter and holding the other constant.

As shown in Figure 4, the optimal sampling hyperparameters are task-dependent. For forecasting tasks (price series and return), which demand precision, lower temperatures (e.g., $T \approx 0.6$) are preferable. This sharpens the next-token distribution, compelling the model towards more deterministic and high-confidence predictions. Conversely, realized volatility forecasting and synthetic K-line generation benefit from greater stochasticity,

achieving optimal performance at temperatures closer to 1.0. A higher temperature encourages the generation of more diverse sequences, which is essential for capturing the probabilistic nature of volatility and for producing realistic, non-repetitive market data.

The analysis of top-p sampling reveals a similar pattern: forecasting tasks favor smaller p values to restrict the sampling pool, whereas generative tasks perform better with a larger nucleus ($p \geq 0.9$) to preserve diversity. When comparing the two techniques, we observe that temperature scaling generally offers more effective and nuanced control, leading to slightly better peak performance across tasks. This suggests that the global probability rescaling of temperature may be a more suitable tuning mechanism than the hard truncation of nucleus sampling.

F.2 Ablation on Tokenizer Architecture

We perform an ablation study on the tokenizer architecture to justify our design choices. We compare our proposed Transformer-based tokenizer using a hierarchical loss against two alternatives: (1) a Transformer-based tokenizer with a standard, non-hierarchical reconstruction loss and (2) a CNN-based architecture with a comparable parameter count. All models are trained with a vocabulary size of 2^{18}.

Table 7: Ablation study on the K-line tokenizer architecture. We compare our proposed Transformer-based tokenizer, which employs a hierarchical reconstruction loss, against two key variants: a Transformer-based tokenizer with a standard reconstruction loss and a CNN-based architecture. All models are trained with a vocabulary size of 2^{18}. The table reports reconstruction quality measured by MAE and MSE.

Tokenizer Architecture	**MAE** (↓)	**MSE** (↓)
Transformer w/ Hierarchical Loss (Ours)	0.0785	0.0203
Transformer w/ Standard Loss	0.0781	0.0202
CNN-based	0.0916	0.0251

As shown in Table 7, the results indicate that Transformer-based architectures outperform the CNN-based model in reconstruction quality, highlighting the effectiveness of self-attention for capturing dependencies in K-line data. More importantly, our model with hierarchical loss achieves reconstruction quality nearly identical to that of the standard loss variant. This confirms that our approach successfully engineers a coarse-to-fine structure within the tokens—a property beneficial for the subsequent autoregressive model—without a notable trade-off in representational fidelity.

F.3 K-line Reconstruction Visualizations

Figure 10 visualizes our tokenizer's reconstruction results on a diverse set of financial instruments. The plots show that the reconstructed 'Close Price' and 'Volume' series closely track the ground truth, confirming that our tokenizer effectively preserves the essential dynamics of the original continuous data within its discrete token representation.

F.4 Cumulative Return Curve Visualizations

Figure 5 presents the cumulative return curves derived from backtesting using predictive signals by different models. As illustrated, Kronos consistently demonstrates superior performance, achieving the highest cumulative returns among the evaluated models.

G Full Experiment Results

In this section, we present the complete experimental results for three forecasting tasks and the synthetic K-line generation task. For the forecasting tasks, we report the results for each asset, averaged over all tested frequencies. Tables 14 and 15 show the results of the price series forecasting experiments. The outcomes for return forecasting are presented in Tables 16 and 17, while those for realized volatility forecasting are in Tables 18 and 19. Furthermore, for the synthetic K-line generation task, Figures 13 and 14 provide visualizations of the *diversity* of the generated sequences by different models. The results for the discriminative score and predictive usefulness are presented in Table 20 and Table 21, respectively. Finally, the results of the investment simulation experiment are presented in Table 8.

Table 8: Full results of investment simulation. We report Annualized Excess Return (AER) and Information Ratio (IR). Best and second best results are marked with red underline and blue underline, respectively.

Model	CSI300 Index		CSI800 Index		Average	
	AER	IR	AER	IR	AER	IR
TimeXer	0.1035	0.7988	0.1509	1.5471	0.1272	1.1730
TimeMixer	−0.0600	−0.5721	0.0705	0.8113	0.0053	0.1196
iTransformer	−0.1202	−1.4441	−0.0525	−0.8558	−0.0864	−1.1500
PatchTST	0.1289	0.9895	0.1620	1.5033	0.1455	1.2464
TimesNet	0.1441	0.6558	0.0634	0.7225	0.1038	0.6892
DLinear	−0.0066	−0.0605	0.1112	1.2003	0.0523	0.5699
FEDformer	0.0362	0.2943	0.0539	0.5602	0.0451	0.4273
NSTransformer	−0.0343	−0.2889	0.0664	0.6979	0.0161	0.2045
Time-MOE$_{base}$	0.0985	0.8230	0.1315	1.3726	0.1150	1.0978
Moirai$_{large}$	0.1470	0.9747	0.1683	1.5215	0.1577	1.2481
TimesFM	0.0788	0.7357	0.1355	1.6427	0.1072	1.1892
Moment$_{large}$	0.1655	1.1993	0.1707	1.5361	0.1681	1.3677
Chronos$_{large}$	−0.0659	−0.7670	0.0056	0.0902	−0.0302	−0.3384
Kronos$_{small}$	0.1805	1.2394	0.1772	1.6050	0.1789	1.4222
Kronos$_{base}$	0.1911	1.3782	0.1867	1.6652	0.1889	1.5217
Kronos$_{large}$	0.2193	1.4177	0.1974	1.8805	0.2084	1.6491

On the core task of price series forecasting, Kronos establishes a new state-of-the-art, boosting the RankIC by 93% over the leading TSFM and by 87% over the best-performing non-pre-trained baseline. Furthermore, it demonstrates strong versatility by achieving a 9% lower MAE in volatility forecasting and a 22% improvement in generative fidelity for synthetic K-line generation.

These findings highlight the broad effectiveness of our approach and underscore Kronos's potential as a robust foundation model for interpreting the complex "language" of financial markets.

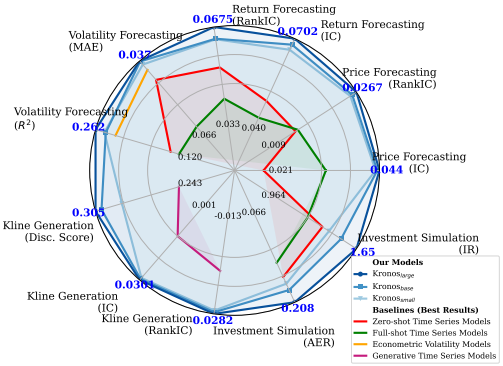


Figure 6: Comprehensive performance of Kronos across several quantitative finance tasks. The chart benchmarks our Kronos models (blue family) against several categories of specialized baselines. A greater distance from the center signifies superior performance.

H Forecast Showcases

Figures 15 to 19 present the forecasting results of our proposed model, Kronos, against several baselines. We select a few representative assets and showcase the predictions for two key features: closing price and trading volume. As observed, the forecasts from Kronos not only achieve competitive predictive performance but also exhibit a strong qualitative resemblance to the ground-truth series. Notably, our model adeptly captures the characteristic dynamics and patterns of the actual price and volume sequences, producing forecasts that are not only accurate but also visually plausible.

Table 9: Model configurations for the Kronos family. We detail the number of Transformer layers, model dimension ($\mathbf{d}_{\text{model}}$), feed-forward dimension ($\mathbf{d}_{\text{ff}}$), number of attention heads, vocabulary size, and the total number of parameters.

Model	Layers	$\mathbf{d}_{\text{model}}$	$\mathbf{d}_{\text{ff}}$	Heads	Vocab. (2^k)	Params
Kronos$_{small}$	8	512	1024	8	20	24.7M
Kronos$_{base}$	12	832	2048	16	20	102.3M
Kronos$_{large}$	18	1664	3072	32	20	499.2M

Table 10: Ablation study dissecting the architectural choices of Kronos. We compare our model against variants targeting different **Prediction Spaces** (continuous vs. discrete) with corresponding **Training Objectives**. *Direct-AR* serves as a standard regression baseline. *Prob-AR* evaluates the benefit of probabilistic modeling in the continuous space. *Kronos-Parallel* ablates our sequential subtoken design by predicting subtokens concurrently. Arrows (↑/↓) indicate preferred direction. Best results are in **bold**. **MSE** stands for Mean Squared Error, **NLL** stands for Negative Log-Likelihood, and **CE** stands for Cross-Entropy.

Model	Prediction Space	Training Objective	Price Series Forecasting		Return Forecasting		Volatility Forecasting	
			IC (↑)	RankIC (↑)	IC (↑)	RankIC (↑)	MAE (↓)	R^2 (↑)
Direct-AR	Continuous	MSE	0.0212	0.0149	0.0416	0.0399	0.0565	0.1608
Prob-AR	Continuous	NLL	0.0179	0.0102	0.0356	0.0329	0.0464	0.1383
Kronos-Parallel	Discrete	CE	0.0345	0.0226	0.0529	0.0505	0.0461	0.1784
Kronos$_{small}$	**Discrete**	**CE**	**0.0431**	**0.0254**	**0.0665**	**0.0622**	**0.0384**	**0.2490**

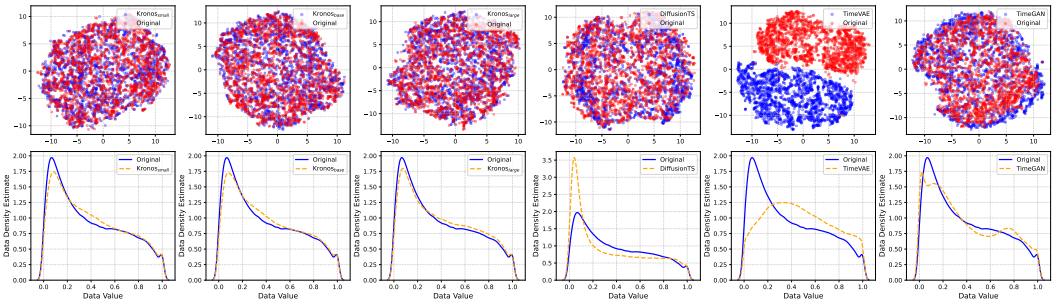


Figure 7: Visual comparison of generative models on the dataset of Shanghai Stock Exchange, 15-minute frequency. **Top row:** t-SNE embeddings of original (red) versus synthetic (blue) data. **Bottom row:** Kernel Density Estimates (KDE) of original versus synthetic data.

I Discussion

I.1 Has K-line data embedded enough information to drive the price movement of capital market in short term? (Q1)

In capital markets, the determinants of price dynamics are conventionally bifurcated into:

- **Long-term driving factors**, which manifest as persistent trends and exert a lasting influence on intrinsic value;
- **Short-term driving factors**, which are typified by elevated volatility and immediate market impact.

Long-term driving factors establish the market's prevailing trajectory and valuation benchmarks, whereas short-term ones introduce transient volatility and generate discrete trading opportunities.

Extensive empirical evidence demonstrates that kline data (**OHLCVA**, including **price** and **trading volume**) [52], when analyzed in tandem, effectively encapsulate the informational content of short-term driving factors—such as macroeconomic data releases [53], corporate event disclosures [52], and shifts in investor sentiment [54, 55].

The detail discussion about the above empirical evidences is beyond the scope of this paper.

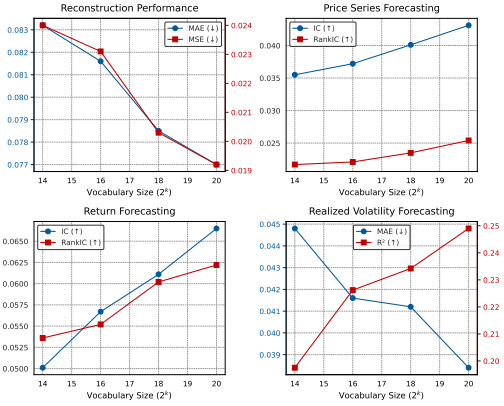


Figure 8: Impact of vocabulary size on model performance. We plot reconstruction quality and downstream forecasting performance as vocabulary size increases.

I.2 What makes Krono's tokenizer work? (Q2)

The effectiveness of our vision-inspired quantization (BSQ) tokenizer can be analyzed from two key perspectives: its inherent noise suppression and its ability to create a structured, discrete state space suitable for sequence modeling.

I.2.1 Noise Suppression and Stability

Financial time-series data is often corrupted by noise and subject to extreme outliers, such as "flash-crash" events caused by anomalous trades. A primary challenge for regression-based models is that such outliers can lead to unbounded approximation errors, severely degrading model stability [56].

Our approach addresses this by transforming the representation learning into a more robust, classification-like framework. By quantizing continuous price-volume embeddings, we effectively cap the influence of any single data point. Specifically, BSQ's projection of embeddings onto a unit sphere prior to binarization guarantees that the expected distortion is strictly upper-bounded [16]:

$$\mathbb{E}_u \left\| u - \widehat{u} \right\| \; < \; \sqrt{2 - 2/\sqrt{L}} \; < \sqrt{2}.$$

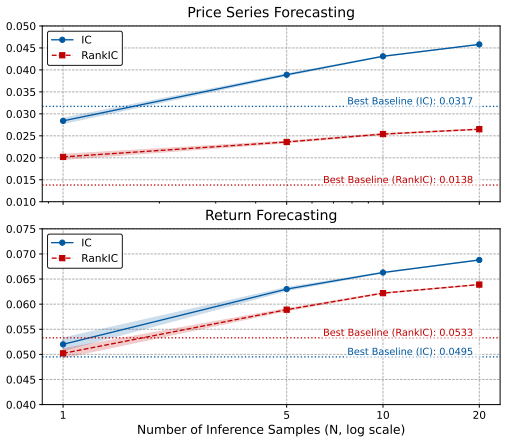


Figure 9: Impact of the number of inference samples (N) on forecasting performance. The lines represent the mean performance over 5 runs with different random seeds, while the shaded areas indicate the standard deviation.

This bound tightens as the codebook dimension L increases. In contrast, simpler methods like sign-based quantization without normalization (e.g., LFQ) lack such a guarantee, leaving them vulnerable to arbitrarily large errors from outlier inputs [16]. This bounded error property is crucial for building reliable financial forecasting models.

I.2.2 Learning in a Compact and Discrete State Space

High-frequency financial data exists in a high-dimensional, continuous state space, posing significant challenges for sequence models. Our tokenizer maps these infinite states into a finite, discrete vocabulary of tokens. This discretization serves as a powerful form of regularization with two main benefits [57]:

- **Improved Sample Efficiency and Generalization**: Instead of learning a complex function over a continuous space, a downstream model like a Transformer learns to predict transitions and patterns among a finite set of abstract states (tokens). This simplifies the learning task. Different but semantically similar input vectors can be mapped to the same token, effectively increasing the number of observations for each discrete state. This allows the model to learn robust patterns from fewer examples, which is particularly critical for modeling rare market phenomena like responses to liquidity shocks, where data is sparse.
- **Reduced Overfitting**: The quantization process inherently discards fine-grained, potentially noisy variations within each quantization cell. This prevents the model from fitting to spurious artifacts in the training data.

Table 11: Codebook usage for coarse-level subtoken and fine-level subtoken.

Codebook Type	Codebook Size	Code Usage
Coarse-Level-Subtoken Codebook	2^{10}	97.66%
Fine-Level-Subtoken Codebook	2^{10}	85.25%

The effectiveness of our tokenizer is further evidenced by its codebook utilization. As shown in Table 11, the code usage of BSQ reaches 97.66% at the coarse level and 85.25% at the fine level. Such high utilization indicates that our method creates an expressive vocabulary, effectively partitioning the feature space without suffering from codebook collapse (where many codes are left unused) [58]. This expressiveness provides the rich foundation necessary for a model to capture the nuanced and diverse states of market microstructure.

Additionally, the vocabulary is stratified into three categories based on usage frequency: (a) high-frequency, (b) low-frequency, and (c) unused tokens. To investigate their representational characteristics, we conduct an

Table 12: Trade-off analysis for factorizing a $k = 20$ bit token into n subtokens, based on the Kronos$_{base}$ architecture. The model's core Transformer blocks have ≈97.5M parameters.

Setup	Splits (n)	Sub-Vocab ($2^{k/n}$)	Core Params (M)	Vocab Params (M)	Fusion Params (M)	Total Params (M)	Inference Steps per Token
No Split	1	1,048,576	97.5	1744.8	0.0	1842.3	1×
Ours	**2**	**1,024**	**97.5**	**3.4**	**1.4**	**102.3**	**2**×
More Splits	4	32	97.5	0.2	2.8	100.5	4×
	5	16	97.5	0.1	3.5	101.1	5×

analysis where we replace the final token of an encoded sequence with a token from each category and then decode it back to a K-line. Figure 12 presents the results of this procedure. We observe a clear correspondence between token frequency and pattern typicality. High-frequency tokens (a) map to common K-bar shapes, indicative of stable market conditions. Conversely, low-frequency (b) and unused (c) tokens generate more extreme and atypical K-bars, such as those with long bodies or wicks, signifying rare, high-volatility events. This suggests that the learned codebook captures a meaningful semantic hierarchy, effectively distinguishing between common and significant market patterns based on token frequency.

I.2.3 Hyperspherical geometry for tail sensitivity

In financial contexts, market returns and price changes often exhibit heavy tails (or fat tails) [59]. The heavy-tail distribution of price changes is one of the key sources of trading profits in quantitative investment and cannot be ignored.

Unlike standard vector-quantization on the Euclidean sphere, BSQ's binary encoding preserves angular information very efficiently, making it more sensitive to fat-tail data that manifest as sharp directional changes in feature space. This aligns well with how microstructure events often appear as abrupt shifts in the "direction" of the joint price-volume vector [60].

Figure 11 illustrates the tokenizer's ability to capture and reconstruct the long-tailed market microstructure under short-term high volatility and during extreme gap events (in the economic context of Trump's Trade War [61]).

Above all, we summarize the concrete advantages of BSQ for K-line time series data, leveraging its ability to preserve angular information and capture sharp directional changes, which are crucial for modeling financial time series with heavy tails and abrupt shifts due to microstructure events.

I.3 Analysis of Subtoken Factorization (Q3)

Our methodology factorizes a k-bit token into n subtokens to manage a large vocabulary size. A key design choice is the number of factors, n. While further factorization (e.g., $n > 2$) could reduce sub-vocabulary sizes even more (e.g., from 2^{10} to 2^5 for a $k = 20$ token), we argue that $n = 2$ offers the best trade-off between parameter efficiency and inference latency.

This factorization introduces a fundamental trade-off. On one hand, it significantly reduces the size of **vocabulary-dependent parameters** in the input embedding and output projection layers, replacing a single large table for a 2^k vocabulary with n smaller tables for $2^{k/n}$ sub-vocabularies. On the other hand, it introduces two costs: (1) a new **fusion layer** (W_{fuse} in Equation 2), whose parameters $(n \times \mathbf{d}_{\text{model}}) \times \mathbf{d}_{\text{model}}$ grow linearly with n, and (2) increased **inference latency**, as generating a full token requires n sequential autoregressive steps.

Table 12 quantifies this trade-off for our Kronos$_{base}$ model. The most significant parameter reduction is achieved by moving from no factorization ($n = 1$) to a 2-way split. This single step reduces vocabulary-dependent parameters by over 99.8% (from ≈1.7B to 3.4M), shrinking the total model size by nearly 95% and making a large effective vocabulary computationally feasible.

However, further factorization yields diminishing returns while incurring rising costs. Moving from $n = 2$ to $n = 4$ reduces vocabulary parameters by only 3.2M, a saving that is partially offset by a 1.4M increase in fusion layer parameters. This results in a marginal total parameter reduction of just ≈2%. As n increases to 5, the overhead from the fusion layer outweighs the savings from the smaller vocabularies, causing the total parameter count to increase. Crucially, these marginal or negative parameter benefits come at a direct and substantial latency cost: moving from $n = 2$ to $n = 4$ doubles the number of sequential generation steps required per token.

In summary, our choice of $n = 2$ represents an effective balance. It captures the vast majority of the parameter-reduction benefits, making our large vocabulary practical, while avoiding the significant latency penalties and growing architectural overhead associated with finer-grained splits.

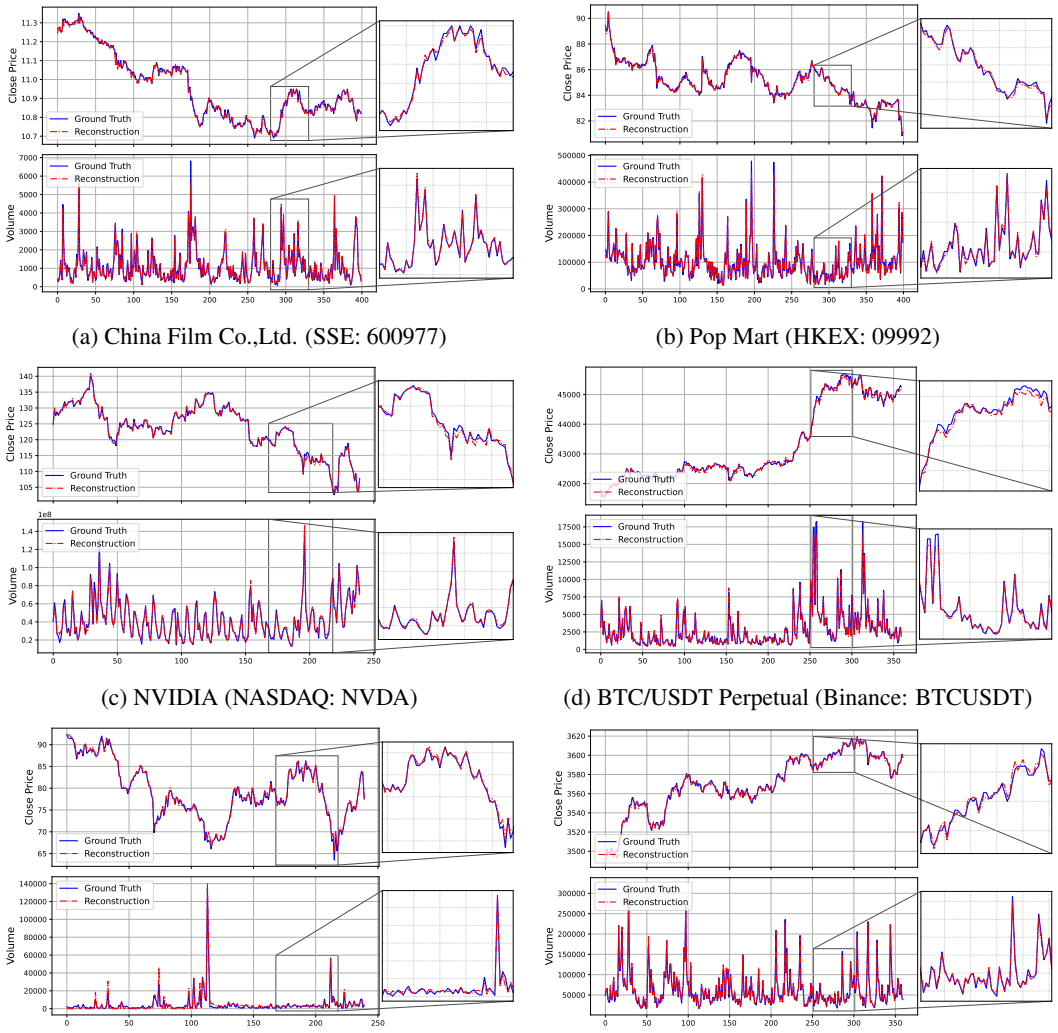
Table 13: Descriptive statistics of the multi-exchange, multi-asset K-line dataset. The timeframe abbreviations are: T (1-min), H (1-hour), D (1-day), W (1-week).

Exchange / Country	Asset Types	Timeframes	#Assets	#Observations	Start Date
Binance	Crypto, Swap	T, 5T, 15T, 30T, H, D, W	997	1,237,002,843	2021/1/31
Athens Stock Exchange	Stock, ETF	D, W	180	226,315	2023/4/11
Beijing Stock Exchange	Stock	5T, 15T, 30T, H, D, W	272	10,197,628	2021/11/19
Brazil Stock Exchange	Stock, ETF	D, W	2,058	1,315,290	2020/1/31
Moscow Exchange	Stock, ETF	D, W	514	567,351	2020/1/31
Euronext Amsterdam	Stock, ETF	D, W	514	602,083	2020/1/31
Australian Securities Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	3,381	86,613,897	2020/1/31
Stock Exchange of Thailand	Stock, ETF	5T, 15T, 30T, H, D, W	1,664	49,590,394	2020/1/31
Bombay Stock Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	5,491	284,428,211	2020/1/31
Euronext Brussels	Stock, ETF	D, W	166	195,491	2020/1/31
Bucharest Stock Exchange	Stock, ETF	D, W	247	176,080	2020/1/31
Budapest Stock Exchange	Stock, ETF	D, W	50	57,586	2022/1/14
Buenos Aires Stock Exchange	Stock	D, W	183	225,352	2020/1/31
Colombo Stock Exchange	Stock	D, W	292	372,627	2020/1/31
Copenhagen Stock Exchange	Stock	D, W	825	617,464	2020/1/31
Frankfurt Stock Exchange	Stock, ETF	D, W	17,054	21,547,744	2020/1/31
Ghana Stock Exchange	Stock	D, W	44	57,690	2020/1/31
Hong Kong Stock Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	3,500	359,434,220	2020/1/31
Japan Exchange Group	Stock, ETF	5T, 15T, 30T, H, D, W	4,467	280,601,980	2020/1/31
Indonesia Stock Exchange	Stock	5T, 15T, 30T, H, D, W	935	38,627,125	2020/1/31
Borsa Istanbul	Stock	D, W	627	784,147	2020/1/31
Johannesburg Stock Exchange	Stock, ETF	D, W	562	681,587	2020/1/31
Pakistan Stock Exchange	Stock, ETF	D, W	660	595,505	2020/1/31
Kuala Lumpur Stock Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	1,150	45,938,559	2020/1/31
Korea Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	2,928	205,061,301	2020/1/31
Lima Stock Exchange	Stock, ETF	D, W	166	63,503	2020/1/31
Euronext Lisbon	Stock, ETF	D, W	60	65,753	2020/1/31
London Stock Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	8,660	177,947,624	2020/1/31
Luxembourg Stock Exchange	Stock	D, W	5	7,598	2020/1/31
Madrid Stock Exchange	Stock, ETF	D, W	309	331,745	2020/1/31
Mexican Stock Exchange	Stock, ETF	D, W	775	937,637	2020/1/31
Nasdaq Stock Exchange	Stock, ETF	T, 5T, 15T, 30T, H, D, W	8,725	2,478,662,459	2000/1/1
National Stock Exchange of India	Stock, ETF	5T, 15T, 30T, H, D, W	2,554	242,429,169	2020/1/31
New York Stock Exchange	Stock, ETF	T, 5T, 15T, 30T, H, D, W	7,073	2,133,143,549	2000/1/1
Euronext Paris	Stock, ETF	D, W	1,781	1,981,059	2020/1/31
Philippine Stock Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	351	4,388,378	2020/1/31
Prague Stock Exchange	Stock	D, W	50	62,666	2020/1/31
Santiago Stock Exchange	Stock	D, W	225	160,638	2020/1/31
Shenzhen Stock Exchange	Stock, ETF	T, 5T, 15T, 30T, H, D, W	3,519	1,754,519,331	1990/12/19
Shenzhen Stock Exchange (B-shares)	Stock	5T, 15T, 30T, H, D, W	46	4,198,702	2020/2/3
Shanghai Stock Exchange	Stock, ETF	T, 5T, 15T, 30T, H, D, W	3,064	1,967,996,343	1990/12/19
Shanghai Stock Exchange (B-shares)	Stock	5T, 15T, 30T, H, D, W	50	4,526,152	2020/2/3
Stockholm Stock Exchange	Stock, ETF	D, W	1,305	1,463,722	2020/1/31
SIX Swiss Exchange	Stock, ETF	D, W	1,981	2,451,675	2020/1/31
Taiwan Stock Exchange	Stock, ETF	5T, 15T, 30T, H, D, W	1,252	71,619,260	2020/1/31
Toronto Stock Exchange	Stock, ETF	D, W	3,035	3,356,561	2020/1/31
Vienna Stock Exchange	Stock	D, W	98	123,643	2020/1/31
China	Future	T, 5T, 15T, D	75	63,318,960	2010/1/1
–	Foreign Exchange	5T, 15T, 30T, H, D, W	1,023	462,434,562	2020/1/31
Australia	Stock Index	5T, 15T, 30T, H, D, W	40	183,158	2020/1/31
Belgium	Stock Index	D, W	5	8,109	2020/1/31

Continued on next page

Table 13 – Continued from previous page

Exchange / Country	Asset Types	Timeframes	#Assets	#Observations	Start Date
Brazil	Stock Index	D, W	3	4,766	2020/1/31
Canada	Stock Index	D, W	18	27,622	2020/1/31
China	Stock Index	5T, 15T, 30T, H, D, W	597	55,884,065	2020/2/3
Germany	Stock Index	D, W	18	28,622	2020/1/31
Spain	Stock Index	D, W	2	3,257	2020/1/31
France	Stock Index	D, W	38	55,945	2020/1/31
Britain	Stock Index	5T, 15T, 30T, H, D, W	51	5,355,869	2020/1/31
Greece	Stock Index	D, W	1	1,589	2020/1/31
Hong Kong, China	Stock Index	5T, 15T, 30T, H, D, W	4	453,016	2020/1/31
Hungary	Stock Index	D, W	1	1,602	2020/1/31
Indonesia	Stock Index	5T, 15T, 30T, H, D, W	2	47,816	2020/1/31
India	Stock Index	5T, 15T, 30T, H, D, W	113	3,189,450	2020/1/31
Japan	Stock Index	5T, 15T, 30T, H, D, W	9	125,024	2020/1/31
Korea	Stock Index	5T, 15T, 30T, H, D, W	5	274,292	2020/1/31
Mexico	Stock Index	D, W	1	1,619	2020/1/31
Malaysia	Stock Index	D, W	2	3,145	2020/1/31
Netherlands	Stock Index	D, W	4	6,475	2020/1/31
Pakistan	Stock Index	D, W	3	3,184	2020/1/31
Philippines	Stock Index	D, W	2	3,187	2020/1/31
Portugal	Stock Index	D, W	1	1,632	2020/1/31
Romania	Stock Index	D, W	5	7,726	2020/1/31
Russia	Stock Index	D, W	15	19,079	2020/1/31
Sweden	Stock Index	D, W	11	16,389	2020/1/31
Thailand	Stock Index	D, W	4	5,005	2020/1/31
Taiwan, China	Stock Index	5T, 15T, 30T, H, D, W	1	85,318	2020/1/31
America	Stock Index	5T, 15T, 30T, H, D, W	670	37,887,535	2020/1/31
Approximate Totals			**96569**	**12.11B**	–



(a) China Film Co.,Ltd. (SSE: 600977)

(b) Pop Mart (HKEX: 09992)

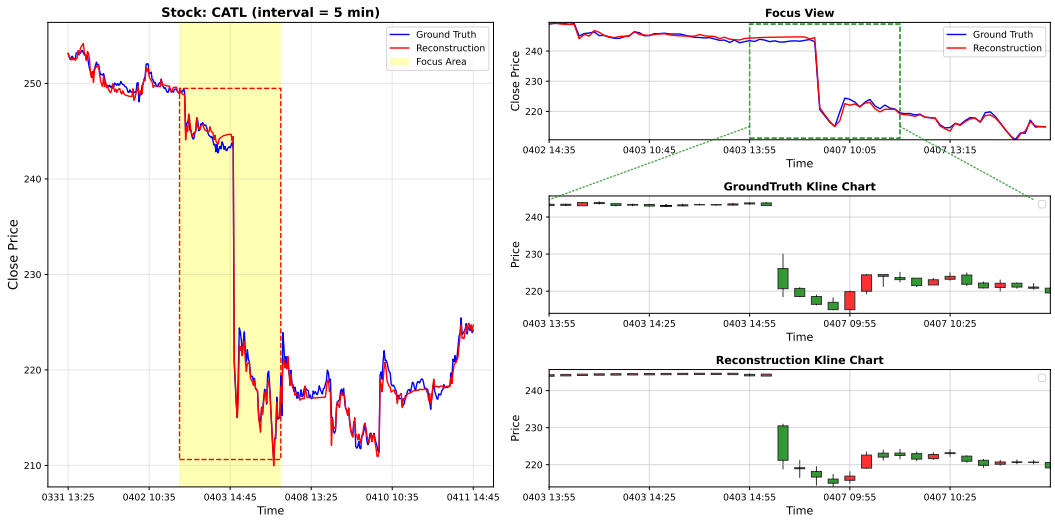
(c) NVIDIA (NASDAQ: NVDA)

(d) BTC/USDT Perpetual (Binance: BTCUSDT)

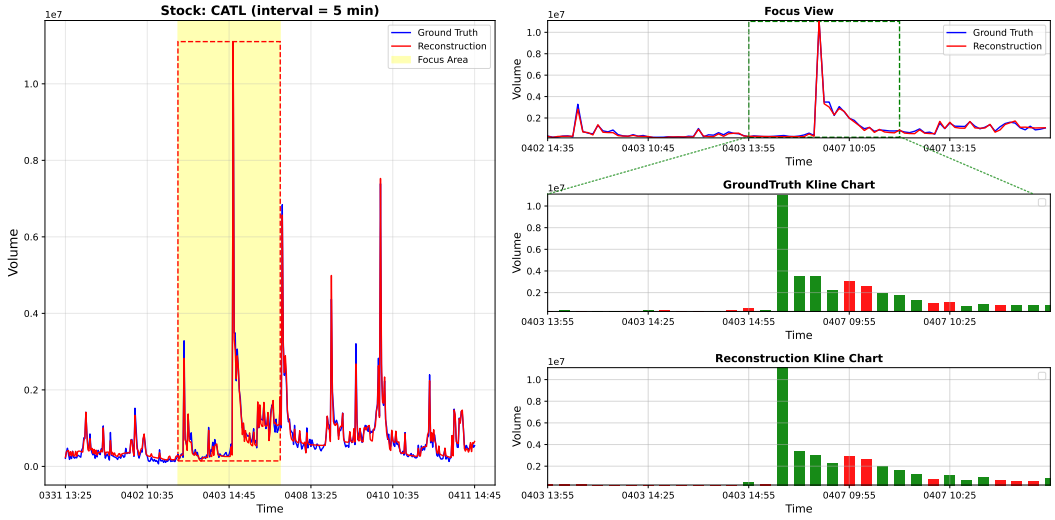
(e) BMW (FWB: BMW)

(f) Rebar Steel Futures (SHFE: RB)

Figure 10: Visualization of reconstruction results for the 'Close Price' and 'Volume' from our K-line Tokenizer. Blue lines denote the ground truth, while red lines indicate the reconstructions generated by our model.



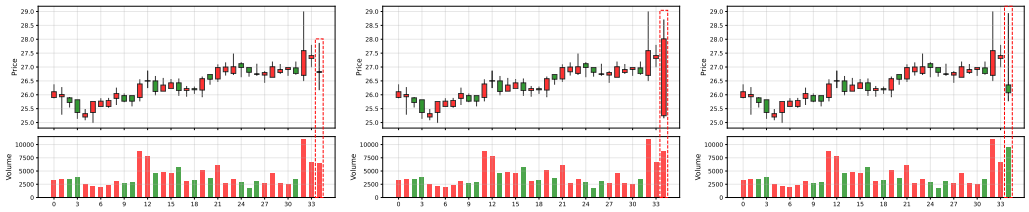
(a) Stock K-line Reconstruction, Price Part



(b) Stock K-line Reconstruction, Volume Part

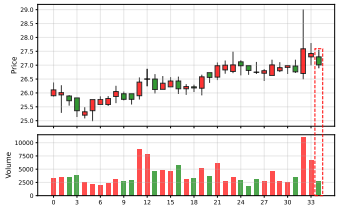
Figure 11: Illustration of the reconstruction performance of 5-minute K-line data for CATL (Contemporary Amperex Technology Co., Limited) on April 7th, 2025, in the economic context of Trump's Trade War [61]. In the visualization, the candlesticks follow a "red for up, green for down" convention (where up/down is determined by the close price relative to the open price), and the volume bars are colored accordingly.

(a) Examples of high-frequency tokens.



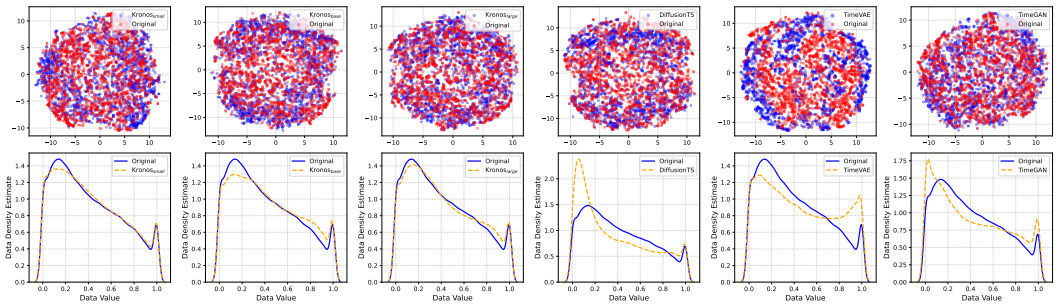
(b) Examples of low-frequency tokens.

(c) Examples of unused tokens from the vocabulary.

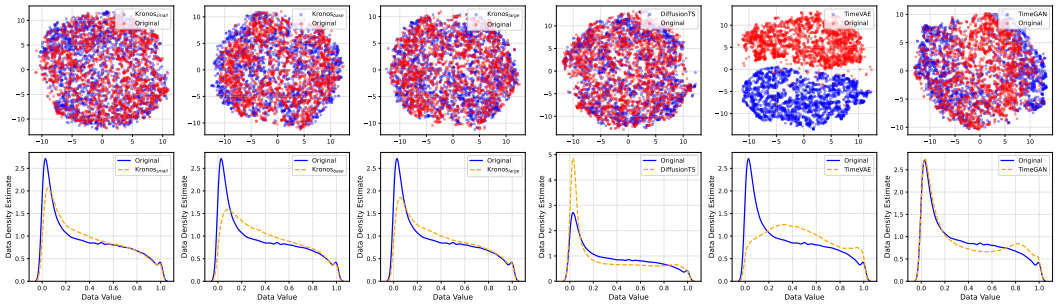


(d) A sample of an original token sequence.

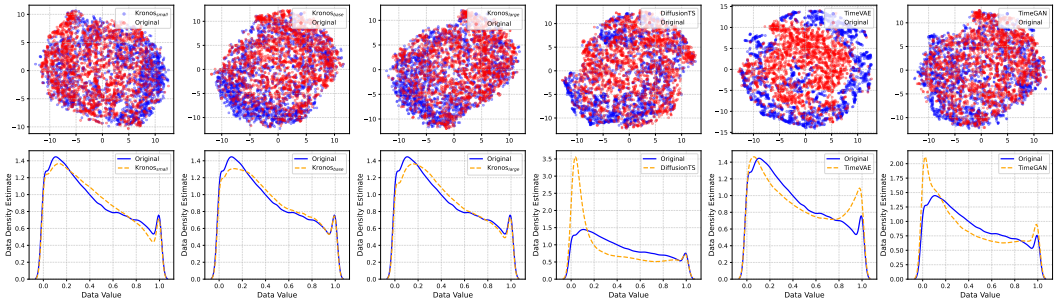
Figure 12: Visualization of token usage patterns. The figure illustrates token categories based on their occurrence frequency in the corpus: (a) high-frequency, (b) low-frequency, and (c) unused (zero-frequency) tokens. A sample from an original sequence (d) is shown for reference. The sequences in (a), (b), and (c) are constructed by replacing the last token of (d) with a randomly sampled token from the corresponding category. In the visualization, the candlesticks follow a "red for up, green for down" convention (where up/down is determined by the close price relative to the open price), and the volume bars are colored accordingly.



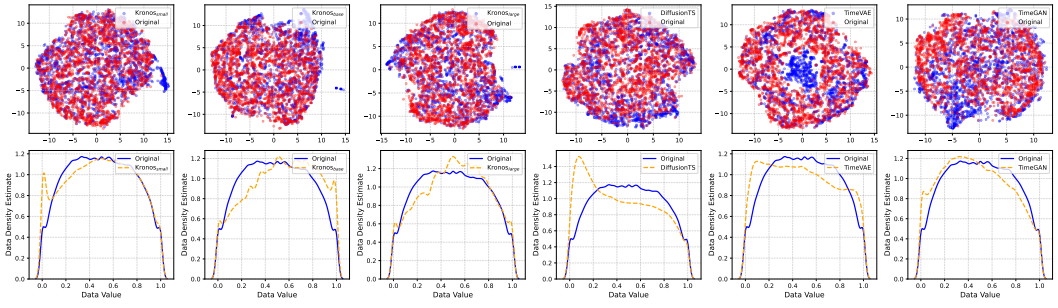
(a) Shanghai Stock Exchange (XSHG), Daily frequency



(b) Taiwan Stock Exchange (XTAI), 15-minute frequency

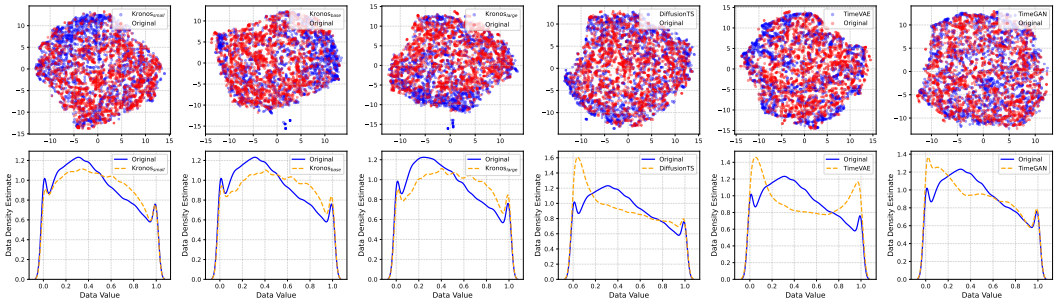


(c) Taiwan Stock Exchange (XTAI), Daily frequency

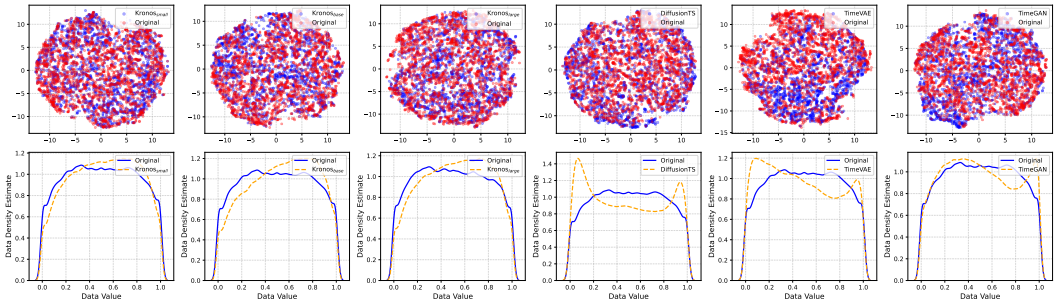


(d) Cryptocurrency (Crypto), 15-minute frequency

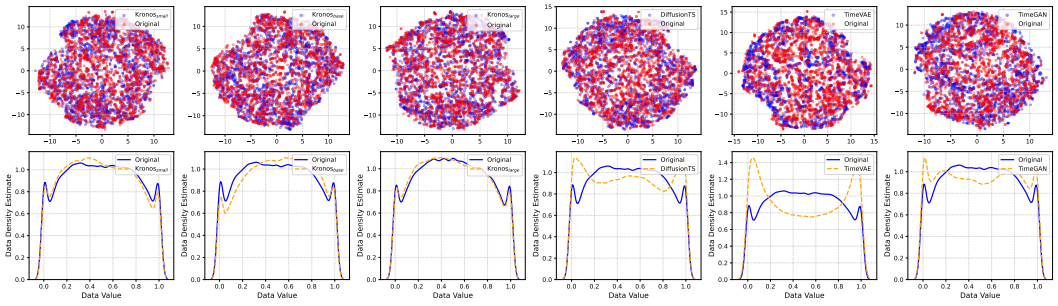
Figure 13: Visual comparison of generative models on different datasets. **Top row in each subfigure:** t-SNE embeddings of original (red) versus synthetic (blue) data. **Bottom row in each subfigure:** Kernel Density Estimates (KDE) of original versus synthetic data.



(a) Cryptocurrency (Crypto), Daily frequency



(b) Foreign Exchange (Forex), 15-minute frequency



(c) Foreign Exchange (Forex), Daily frequency

Figure 14: Visual comparison of generative models on different datasets. **Top row in each subfigure:** t-SNE embeddings of original (red) versus synthetic (blue) data. **Bottom row in each subfigure:** Kernel Density Estimates (KDE) of original versus synthetic data.

Table 14: Full results of price series forecasting experiments (Part 1): Our model (Kronos) and full-shot time series models. A higher IC or RankIC indicates a better prediction. Best and second best results are marked with red underline and blue underline, respectively.

Models		Kronos (Ours)			Full-shot Time Series Models							
Metrics		Kronos_S	Kronos_B	Kronos_L	TimeXer	TimeMixer	iTransformer	PatchTST	TimesNet	DLinear	FEDformer	NSTransformer
XSHG	IC	0.0549	0.0564	0.0546	0.0280	0.0291	0.0350	0.0450	0.0424	0.0405	0.0233	0.0433
	RankIC	0.0375	0.0390	0.0381	0.0053	0.0079	0.0128	0.0088	0.0175	0.0181	0.0107	0.0155
XNAS	IC	0.0343	0.0322	0.0361	0.0132	0.0097	0.0204	0.0116	0.0174	0.0197	0.0165	0.0253
	RankIC	0.0155	0.0190	0.0191	0.0106	0.0048	0.0111	0.0083	0.0084	0.0084	−0.0014	0.0016
XJPX	IC	0.0314	0.0332	0.0360	0.0094	0.0017	0.0137	0.0053	0.0099	0.0118	0.0046	0.0281
	RankIC	0.0199	0.0209	0.0277	0.0159	0.0036	0.0271	0.0056	0.0127	0.0149	0.0024	0.0212
XNSE	IC	0.0634	0.0648	0.0634	−0.0055	0.0094	−0.0252	0.0082	0.0566	0.0024	0.0063	0.0514
	RankIC	0.0434	0.0464	0.0486	−0.0371	0.0024	−0.0248	0.0084	0.0379	−0.0024	0.0003	0.0225
XKRX	IC	0.0550	0.0575	0.0567	−0.0328	0.0036	−0.0442	0.0248	0.0416	0.0001	−0.0070	0.0416
	RankIC	0.0362	0.0393	0.0373	−0.0160	0.0033	−0.0284	0.0214	0.0285	0.0006	−0.0049	0.0058
XHKG	IC	0.0435	0.0439	0.0428	0.0318	0.0322	0.0336	0.0401	0.0333	0.0392	0.0296	0.0366
	RankIC	0.0226	0.0236	0.0228	−0.0051	−0.0009	−0.0021	−0.0068	−0.0040	−0.0009	−0.0078	−0.0017
XIDX	IC	0.0551	0.0551	0.0573	−0.0139	0.0116	−0.0233	0.0194	0.0468	0.0158	0.0169	0.0381
	RankIC	0.0214	0.0216	0.0223	0.0025	0.0046	0.0011	0.0149	0.0171	0.0037	0.0084	0.0051
XKLS	IC	0.0411	0.0408	0.0466	−0.0283	0.0079	−0.0281	−0.0037	0.0341	0.0306	−0.0102	0.0101
	RankIC	0.0215	0.0149	0.0167	0.0051	0.0171	0.0024	−0.0078	−0.0025	0.0208	−0.0169	−0.0103
XTAI	IC	0.0424	0.0443	0.0448	0.0282	0.0197	0.0275	0.0328	0.0312	0.0394	0.0249	0.0334
	RankIC	0.0301	0.0320	0.0342	−0.0042	0.0015	0.0111	0.0147	0.0095	0.0192	0.0059	0.0129
Crypto	IC	0.0247	0.0209	0.0211	0.0105	0.0128	0.0155	0.0149	0.0192	0.0137	0.0081	0.0164
	RankIC	0.0138	0.0135	0.0129	0.0022	0.0038	0.0134	0.0192	0.0146	0.0040	0.0000	0.0096
Forex	IC	0.0279	0.0292	0.0244	0.0124	0.0102	0.0142	0.0158	0.0167	0.0227	0.0153	0.0228
	RankIC	0.0177	0.0141	0.0137	0.0134	0.0128	0.0090	0.0085	0.0175	0.0168	0.0120	0.0079
Average	IC	0.0431	0.0435	0.0440	0.0048	0.0134	0.0036	0.0195	0.0317	0.0214	0.0117	0.0316
	RankIC	0.0254	0.0258	0.0267	−0.0007	0.0055	0.0030	0.0087	0.0143	0.0094	0.0008	0.0082
1st Count		4	7	10	0	0	0	1	0	0	0	0

Table 15: Full results of price series forecasting experiments (Part 2): Zero-shot time series models. A higher IC or RankIC indicates a better prediction. Best and second best results are marked with red underline and blue underline, respectively.

Models		Zero-shot Time Series Models											
Metrics		Time-MOE_S	Time-MOE_B	Moirai_S	Moirai_B	Moirai_L	TimesFM	Moment_S	Moment_B	Moment_L	Chronos_S	Chronos_B	Chronos_L
XSHG	IC	0.0463	0.0493	−0.0007	−0.0005	−0.0002	0.0174	0.0028	−0.0032	−0.0009	0.0147	0.0069	0.0195
	RankIC	0.0304	0.0317	0.0000	−0.0012	0.0003	0.0020	0.0003	−0.0037	−0.0017	−0.0026	−0.0108	0.0025
XNAS	IC	−0.0032	−0.0045	−0.0008	−0.0005	0.0000	0.0076	0.0010	−0.0023	−0.0003	−0.0025	−0.0005	0.0020
	RankIC	−0.0033	−0.0042	−0.0008	0.0013	0.0007	0.0112	−0.0015	−0.0027	−0.0007	−0.0001	0.0008	0.0030
XJPX	IC	0.0268	0.0280	0.0012	0.0004	0.0000	0.0076	−0.0010	−0.0003	−0.0027	0.0117	0.0113	0.0067
	RankIC	0.0228	0.0230	0.0025	0.0019	0.0016	0.0073	−0.0031	0.0010	−0.0006	0.0110	0.0123	0.0070
XNSE	IC	0.0173	0.0190	−0.0005	−0.0008	−0.0006	0.0025	0.0063	−0.0129	−0.0039	−0.0012	−0.0049	0.0014
	RankIC	0.0155	0.0169	−0.0021	−0.0021	−0.0029	0.0009	0.0060	−0.0104	−0.0055	−0.0041	−0.0066	−0.0005
XKRX	IC	0.0113	0.0141	−0.0014	0.0006	−0.0011	−0.0105	0.0056	−0.0083	−0.0114	−0.0009	−0.0018	0.0061
	RankIC	0.0088	0.0118	−0.0020	0.0006	−0.0002	−0.0097	0.0041	−0.0082	−0.0069	−0.0006	−0.0009	0.0072
XHKG	IC	0.0174	0.0189	0.0000	−0.0001	−0.0013	0.0117	0.0013	−0.0050	−0.0003	0.0159	0.0140	0.0166
	RankIC	0.0186	0.0201	0.0003	0.0031	0.0011	0.0058	−0.0034	−0.0013	0.0009	0.0190	0.0179	0.0192
XIDX	IC	−0.0053	−0.0052	−0.0009	−0.0009	−0.0003	0.0026	0.0052	−0.0094	−0.0007	0.0021	0.0042	0.0080
	RankIC	0.0002	0.0000	0.0008	0.0012	0.0007	0.0042	−0.0015	−0.0029	0.0014	0.0087	0.0122	0.0153
XKLS	IC	0.0123	0.0125	−0.0003	−0.0028	0.0005	0.0106	0.0045	−0.0093	−0.0065	−0.0080	−0.0076	−0.0077
	RankIC	0.0112	0.0135	0.0010	0.0027	0.0047	−0.0052	0.0000	0.0017	−0.0031	0.0113	0.0118	0.0114
XTAI	IC	0.0296	0.0292	0.0005	0.0001	−0.0004	−0.0002	0.0025	−0.0047	−0.0046	0.0028	−0.0002	0.0080
	RankIC	0.0234	0.0224	0.0011	0.0013	0.0003	−0.0028	0.0001	−0.0023	−0.0009	0.0088	0.0060	0.0125
Crypto	IC	0.0054	0.0037	−0.0008	−0.0006	−0.0004	−0.0009	−0.0002	−0.0004	−0.0030	−0.0114	−0.0129	−0.0096
	RankIC	0.0069	0.0050	0.0004	0.0011	0.0000	0.0014	−0.0011	−0.0061	−0.0007	−0.0051	−0.0061	−0.0045
Forex	IC	0.0265	0.0267	−0.0011	−0.0011	0.0000	0.0092	−0.0007	0.0008	0.0024	0.0176	0.0143	0.0155
	RankIC	0.0115	0.0114	−0.0010	0.0005	−0.0003	0.0076	−0.0014	−0.0010	0.0022	0.0168	0.0147	0.0127
Average	IC	0.0168	0.0174	−0.0004	−0.0006	−0.0003	0.0052	0.0025	−0.0050	−0.0029	0.0037	0.0021	0.0060
	RankIC	0.0133	0.0138	0.0000	0.0009	0.0005	0.0021	−0.0001	−0.0033	−0.0014	0.0057	0.0047	0.0078
1st Count		0	0	0	0	0	0	0	0	0	0	0	0

Table 16: Full results of return forecasting experiments (Part 1): Our model (Kronos) and full-shot time series models. A higher IC or RankIC indicates a better prediction. Best and second best results are marked with red underline and blue underline, respectively.

Models		Kronos (Ours)			Full-shot Time Series Models							
Metrics		Kronos_S	Kronos_B	Kronos_L	TimeXer	TimeMixer	iTransformer	PatchTST	TimesNet	DLinear	FEDformer	NSTransformer
XSHG	IC	0.0677	0.0652	0.0662	0.0456	0.0114	0.0371	0.0467	0.0563	0.0626	0.0589	0.0777
	RankIC	0.0617	0.0653	0.0642	0.0306	−0.0072	0.0266	0.0437	0.0421	0.0461	0.0568	0.0595
XNAS	IC	0.0563	0.0626	0.0639	0.0051	0.0270	0.0340	0.0569	−0.0193	0.0144	0.0219	0.0377
	RankIC	0.0513	0.0544	0.0601	0.0061	0.0204	0.0251	0.0446	0.0352	0.0518	0.0254	0.0335
XJPX	IC	0.0618	0.0667	0.0668	0.0309	0.0211	0.0439	0.0655	0.0656	0.0621	0.0409	0.0436
	RankIC	0.0583	0.0623	0.0687	0.0474	0.0145	0.0399	0.0446	0.0556	0.0253	0.0373	0.0428
XNSE	IC	0.0501	0.0523	0.0585	−0.0021	−0.0126	0.0117	0.0216	0.0238	0.0144	0.0238	0.0314
	RankIC	0.0541	0.0550	0.0639	0.0031	0.0044	0.0146	0.0238	0.0277	0.0442	0.0130	0.0312
XKRX	IC	0.0749	0.0778	0.0792	0.0389	0.0253	0.0309	0.0589	0.0844	0.0704	0.0726	0.0754
	RankIC	0.0707	0.0763	0.0790	−0.0024	−0.0071	0.0282	0.0422	0.0801	0.0439	0.0354	0.0792
XHKG	IC	0.0678	0.0661	0.0654	0.0666	−0.0276	0.0106	0.0470	0.0276	0.0404	0.0496	0.0210
	RankIC	0.0671	0.0646	0.0703	0.0707	−0.0063	0.0091	0.0631	0.0288	0.0558	0.0605	0.0264
XIDX	IC	0.0998	0.0990	0.1046	0.0039	−0.0095	0.0393	0.0003	0.0301	0.0195	−0.0007	0.0244
	RankIC	0.0943	0.0924	0.1007	−0.0111	−0.0018	0.0341	0.0280	0.0304	0.0184	0.0358	0.0610
XKLS	IC	0.1213	0.1153	0.1359	0.0144	0.0074	0.0252	0.0605	0.0941	0.0781	−0.0016	0.1046
	RankIC	0.1047	0.1009	0.1145	−0.0261	0.0097	0.0237	0.0685	0.0712	0.0800	−0.0050	0.0851
XTAI	IC	0.0549	0.0524	0.0511	0.0382	−0.0038	0.0313	0.0421	0.0216	0.0514	0.0489	0.0143
	RankIC	0.0597	0.0584	0.0609	0.0404	−0.0027	0.0163	0.0363	0.0261	0.0431	0.0444	0.0159
Crypto	IC	0.0373	0.0376	0.0368	0.0286	0.0250	0.0372	0.0163	0.0348	0.0446	0.0065	0.0274
	RankIC	0.0332	0.0336	0.0333	0.0154	0.0135	0.0151	0.0213	0.0272	0.0283	0.0027	0.0111
Forex	IC	0.0398	0.0555	0.0441	0.0079	0.0203	0.0266	0.0124	0.0054	0.0254	0.0146	0.0122
	RankIC	0.0289	0.0343	0.0274	0.0275	0.0322	0.0152	0.0148	0.0037	0.0279	0.0148	0.0169
Average	IC	0.0665	0.0682	0.0702	0.0253	0.0076	0.0298	0.0389	0.0386	0.0439	0.0305	0.0427
	RankIC	0.0622	0.0634	0.0675	0.0183	0.0063	0.0225	0.0392	0.0389	0.0423	0.0292	0.0421
1st Count		2	3	10	1	0	0	0	2	1	0	1

Table 17: Full results of return forecasting experiments (Part 2): Zero-shot time series models. A higher IC or RankIC indicates a better prediction. Best and second best results are marked with red underline and blue underline, respectively.

Models		Zero-shot Time Series Models											
Metrics		Time-MOE_S	Time-MOE_B	Moirai_S	Moirai_B	Moirai_L	TimesFM	Moment_S	Moment_B	Moment_L	Chronos_S	Chronos_B	Chronos_L
XSHG	IC	0.0507	0.0501	0.0507	0.0579	0.0534	0.0322	0.0575	0.0579	0.0575	−0.0152	−0.0055	−0.0019
	RankIC	0.0612	0.0621	0.0657	0.0647	0.0661	0.0445	0.0527	0.0530	0.0525	−0.0277	−0.0116	−0.0048
XNAS	IC	0.0416	0.0399	0.0275	0.0281	0.0271	0.0226	0.0290	0.0288	0.0287	0.0545	0.0504	0.0572
	RankIC	0.0480	0.0457	0.0280	0.0290	0.0304	0.0271	0.0300	0.0297	0.0296	0.0448	0.0405	0.0461
XJPX	IC	0.0639	0.0642	0.0441	0.0417	0.0446	0.0498	0.0509	0.0508	0.0512	0.0326	0.0323	0.0276
	RankIC	0.0473	0.0487	0.0790	0.0790	0.0793	0.0579	0.0490	0.0491	0.0493	0.0175	0.0174	0.0126
XNSE	IC	0.0348	0.0343	0.0356	0.0357	0.0354	0.0068	0.0356	0.0357	0.0354	0.0190	0.0179	0.0168
	RankIC	0.0476	0.0483	0.0518	0.0518	0.0514	0.0180	0.0518	0.0518	0.0514	0.0116	0.0175	0.0161
XKRX	IC	0.0573	0.0566	0.0545	0.0546	0.0512	0.0392	0.0545	0.0546	0.0544	0.0523	0.0508	0.0532
	RankIC	0.0599	0.0592	0.0617	0.0619	0.0545	0.0465	0.0617	0.0619	0.0618	0.0348	0.0347	0.0394
XHKG	IC	0.0373	0.0385	0.0324	0.0314	0.0304	0.0281	0.0358	0.0357	0.0357	0.0271	0.0286	0.0297
	RankIC	0.0439	0.0431	0.0485	0.0487	0.0486	0.0369	0.0485	0.0487	0.0486	0.0315	0.0331	0.0328
XIDX	IC	0.0611	0.0565	0.0487	0.0475	0.0474	0.0555	0.0487	0.0488	0.0489	0.0514	0.0560	0.0615
	RankIC	0.0638	0.0597	0.0586	0.0586	0.0587	0.0582	0.0586	0.0586	0.0587	0.0404	0.0486	0.0522
XKLS	IC	0.0971	0.0963	0.0815	0.0782	0.0852	0.0585	0.0856	0.0854	0.0854	0.0804	0.0788	0.0772
	RankIC	0.0954	0.0952	0.1004	0.1001	0.0999	0.0710	0.0803	0.0800	0.0799	0.0723	0.0698	0.0697
XTAI	IC	0.0386	0.0369	0.0418	0.0414	0.0412	0.0332	0.0418	0.0414	0.0412	0.0361	0.0359	0.0338
	RankIC	0.0238	0.0202	0.0494	0.0488	0.0487	0.0505	0.0394	0.0388	0.0387	0.0264	0.0326	0.0312
Crypto	IC	0.0291	0.0293	−0.0051	−0.0081	−0.0046	−0.0042	−0.0042	−0.0039	−0.0043	0.0041	0.0067	0.0107
	RankIC	0.0122	0.0112	0.0157	0.0172	0.0159	0.0105	0.0058	0.0071	0.0059	−0.0069	−0.0064	0.0009
Forex	IC	0.0334	0.0336	0.0355	0.0357	0.0347	0.0353	0.0155	0.0157	0.0157	0.0289	0.0255	0.0274
	RankIC	0.0217	0.0215	0.0262	0.0264	0.0264	0.0276	0.0162	0.0164	0.0164	0.0194	0.0218	0.0184
Average	IC	0.0495	0.0487	0.0407	0.0404	0.0405	0.0325	0.0410	0.0410	0.0409	0.0337	0.0343	0.0357
	RankIC	0.0477	0.0468	0.0532	0.0533	0.0527	0.0408	0.0449	0.0450	0.0448	0.0240	0.0271	0.0286
1st Count		0	0	0	0	2	0	0	0	0	0	0	0

Table 18: Full results of realized volatility forecasting experiments (Part 1): Our model (Kronos) and full-shot time series models. A lower MAE or higher R^2 indicates a better prediction. Best and second best results are marked with red underline and blue underline, respectively.

Models		Kronos (Ours)			Full-shot Time Series Models								Eco. Volatility Models	
Metrics		Kronos_S	Kronos_B	Kronos_L	TimeXer	TimeMixer	iTransformer	PatchTST	TimesNet	DLinear	FEDformer	NSTransformer	ARCH	GARCH
XSHG	MAE	0.0199	0.0205	0.0203	0.0510	0.0349	0.0593	0.0356	0.0348	0.0398	0.0231	0.0348	0.0247	0.0219
	R^2	0.2597	0.2630	0.2809	0.1500	0.1585	0.2191	0.2401	0.1429	0.2400	0.2301	0.1232	0.1969	0.1986
XNAS	MAE	0.1540	0.1407	0.1503	0.3323	0.3473	0.3223	0.2926	0.2492	0.2416	0.2223	0.2168	0.1472	0.1259
	R^2	0.1169	0.0961	0.0978	0.0819	0.0071	0.0876	0.1036	0.0452	0.1192	0.0512	0.0963	0.2174	0.2271
XJPX	MAE	0.0198	0.0198	0.0196	0.1309	0.1324	0.0425	0.0842	0.0365	0.1527	0.0316	0.0353	0.0320	0.0271
	R^2	0.1626	0.1912	0.1996	0.1818	0.0229	0.1245	0.0383	0.1277	0.0133	0.0467	0.1531	0.2421	0.2434
XNSE	MAE	0.0264	0.0269	0.0267	0.0667	0.0347	0.0502	0.0784	0.0555	0.1272	0.0614	0.0497	0.0269	0.0271
	R^2	0.1803	0.1445	0.1815	0.1184	0.0708	0.1140	0.0153	0.0486	0.0152	0.0286	0.0365	0.1424	0.1548
XKRX	MAE	0.0271	0.0255	0.0246	0.0332	0.0424	0.0408	0.0449	0.0537	0.0608	0.0715	0.0552	0.0347	0.0316
	R^2	0.5936	0.6190	0.6156	0.1966	0.0175	0.1967	0.1792	0.2695	0.0795	0.0842	0.2223	0.4617	0.4641
XHKG	MAE	0.0352	0.0402	0.0349	0.0435	0.0746	0.0679	0.0547	0.0608	0.0529	0.0702	0.0499	0.0464	0.0402
	R^2	0.1935	0.1875	0.1824	0.1423	0.0515	0.0394	0.0408	0.0396	0.0482	0.0176	0.0051	0.3294	0.3295
XIDX	MAE	0.0566	0.0544	0.0501	0.1412	0.2504	0.0925	0.0728	0.0827	0.1263	0.0987	0.0836	0.0647	0.0592
	R^2	0.1275	0.1884	0.1467	0.1443	0.0163	0.1730	0.0433	0.1053	0.0322	0.0391	0.1065	0.2209	0.2092
XKLS	MAE	0.0370	0.0367	0.0376	0.1570	0.0823	0.0456	0.1355	0.0759	0.0533	0.0787	0.0827	0.0397	0.0406
	R^2	0.5369	0.4781	0.4967	0.1867	0.1378	0.2245	0.1201	0.1409	0.0529	0.0540	0.1172	0.2148	0.2247
XTAI	MAE	0.0217	0.0220	0.0213	0.0230	0.0254	0.0267	0.0229	0.0318	0.0262	0.0223	0.0271	0.0263	0.0240
	R^2	0.2607	0.2074	0.2915	0.1755	0.1797	0.1740	0.2171	0.1591	0.2592	0.1853	0.1783	0.2021	0.2320
Crypto	MAE	0.0147	0.0148	0.0145	0.1438	0.0705	0.0346	0.0926	0.0289	0.0446	0.0642	0.0375	0.0286	0.0292
	R^2	0.1772	0.2179	0.2658	0.0468	0.0711	0.1212	0.1475	0.2372	0.0547	0.0286	0.1095	0.1642	0.1575
Forex	MAE	0.0097	0.0074	0.0069	0.0277	0.0277	0.0205	0.0300	0.0187	0.0212	0.0171	0.0176	0.0219	0.0185
	R^2	0.1301	0.1235	0.1277	0.0002	0.0302	0.0290	0.0270	0.0029	0.0901	0.0382	0.0034	0.1169	0.1141
Average	MAE	0.0384	0.0372	0.0370	0.1046	0.1021	0.0730	0.0858	0.0662	0.0861	0.0692	0.0627	0.0448	0.0405
	R^2	0.2490	0.2470	0.2624	0.1295	0.0694	0.1366	0.1066	0.1199	0.0913	0.0731	0.1047	0.2281	0.2323
1st Count		4	2	11	0	0	0	0	0	0	0	0	1	3

Table 19: Full results of realized volatility forecasting experiments (Part 2): Zero-shot time series models. A lower MAE or higher R^2 indicates a better prediction. Best and second best results are marked with red underline and blue underline, respectively.

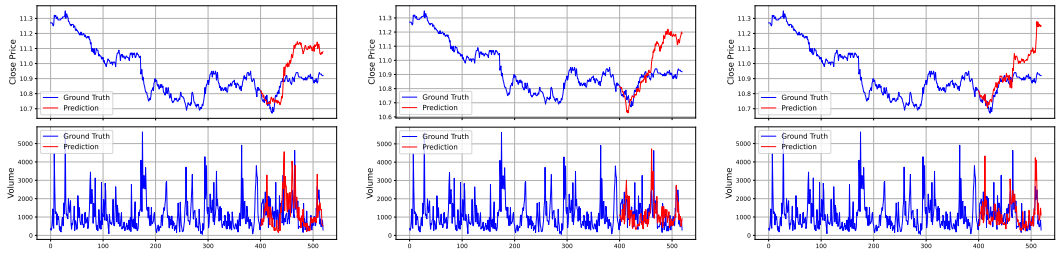
Models		Zero-shot Time Series Models											
Metrics		Time-MOE_S	Time-MOE_B	Moirai_S	Moirai_B	Moirai_L	TimesFM	Moment_S	Moment_B	Moment_L	Chronos_S	Chronos_B	Chronos_L
XSHG	MAE	0.0462	0.0471	0.1158	0.0994	0.1048	0.0408	0.0357	0.0343	0.0366	0.0386	0.0384	0.0382
	R^2	0.2423	0.2417	0.2118	0.2233	0.2191	0.0995	0.2479	0.2461	0.2336	0.1946	0.1922	0.1663
XNAS	MAE	0.2713	0.2498	0.3537	0.1927	0.2502	0.1902	0.1034	0.1020	0.1168	0.1896	0.1863	0.1881
	R^2	0.1255	0.0901	0.1782	0.1228	0.1306	0.0740	0.0872	0.0882	0.0804	0.0811	0.0340	0.0982
XJPX	MAE	0.0372	0.0367	0.1065	0.0829	0.0878	0.0345	0.0291	0.0278	0.0306	0.0331	0.0331	0.0329
	R^2	0.1392	0.1374	0.1150	0.1541	0.1493	0.1213	0.1489	0.1450	0.013 75	0.1812	0.1794	0.1769
XNSE	MAE	0.0420	0.0415	0.1029	0.0873	0.0924	0.0437	0.0364	0.0358	0.0397	0.0414	0.0413	0.0411
	R^2	0.0411	0.0457	0.0455	0.0588	0.0554	0.0394	0.0483	0.0468	0.0422	0.0454	0.0563	0.0439
XKRX	MAE	0.0452	0.0447	0.1109	0.0909	0.0982	0.0508	0.0418	0.0413	0.0461	0.0485	0.0484	0.0482
	R^2	0.2248	0.2321	0.2235	0.2576	0.2229	0.1249	0.2914	0.2811	0.2588	0.3357	0.3371	0.3132
XHKG	MAE	0.0701	0.0671	0.1824	0.1367	0.1499	0.0551	0.0500	0.0475	0.0499	0.0526	0.0523	0.0521
	R^2	0.1757	0.1475	0.0900	0.1838	0.1576	0.1862	0.1537	0.1502	0.1432	0.1064	0.1018	0.1090
XIDX	MAE	0.0725	0.0718	0.2321	0.1687	0.1876	0.0766	0.0652	0.0695	0.0663	0.0744	0.0735	0.0732
	R^2	0.1558	0.1572	0.1228	0.1118	0.1144	0.0952	0.1607	0.1093	0.1471	0.1445	0.1820	0.1692
XKLS	MAE	0.0572	0.0553	0.1142	0.0914	0.1037	0.0733	0.0571	0.0597	0.0699	0.0706	0.0705	0.0703
	R^2	0.0828	0.1021	0.1451	0.1559	0.1669	0.0541	0.1714	0.1725	0.1393	0.1673	0.1745	0.1645
XTAI	MAE	0.0387	0.0384	0.1047	0.0900	0.0954	0.0386	0.0335	0.0319	0.0341	0.0371	0.0369	0.0366
	R^2	0.1901	0.1913	0.1611	0.1704	0.1729	0.0789	0.1885	0.1850	0.1672	0.1868	0.1804	0.1588
Crypto	MAE	0.0374	0.0373	0.0570	0.0574	0.0572	0.0352	0.0209	0.0236	0.0327	0.0341	0.0340	0.0339
	R^2	0.1416	0.1387	0.1061	0.1004	0.2016	0.0881	0.1685	0.1310	0.1758	0.1566	0.1608	0.1584
Forex	MAE	0.0225	0.0110	0.0119	0.0151	0.0120	0.0171	0.0151	0.0155	0.0158	0.0102	0.0124	0.0218
	R^2	0.1173	0.0286	0.0145	0.0306	0.0504	0.0141	0.0744	0.0592	0.0717	0.0391	0.0245	0.0453
Average	MAE	0.0673	0.0637	0.1356	0.1011	0.1127	0.0596	0.0444	0.0444	0.0490	0.0573	0.0570	0.0579
	R^2	0.1487	0.1375	0.1285	0.1427	0.1492	0.0887	0.1380	0.1468	0.1339	0.1490	0.1475	0.1458
1st Count		0	0	0	0	0	0	0	1	0	0	0	0

Table 20: Full discriminative score results for synthetic K-line generation experiments. A higher score indicates a better generation quality. Best and second best results are marked with red underline and blue underline, respectively.

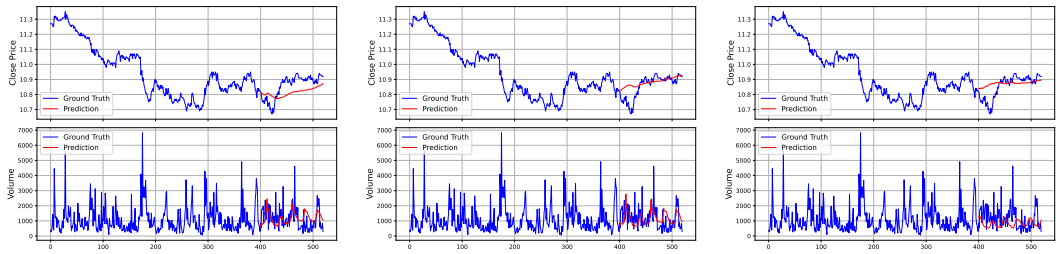
Models		Kronos (Ours)			Time-series Generative Models		
Metrics		Kronos_{small}	Kronos_{base}	Kronos_{large}	DiffusionTS	TimeVAE	TimeGAN
XSHG	15min	0.2313	0.2317	0.2393	0.0885	0.0015	0.2241
	daily	0.1865	0.2227	0.2105	0.2532	0.0142	0.1193
XTAI	15min	0.1733	0.1478	0.1788	0.1420	0.0387	0.2689
	daily	0.2088	0.2023	0.2235	0.1712	0.0097	0.0622
Crypto	15min	0.4100	0.4185	0.4187	0.3005	0.0637	0.0680
	daily	0.2792	0.2575	0.2835	0.3188	0.0402	0.2114
Forex	15min	0.4783	0.4903	0.4688	0.4112	0.0492	0.4015
	daily	0.3337	0.4363	0.4152	0.3177	0.0295	0.2387
Average		0.2876	0.3009	0.3048	0.2504	0.0308	0.1993
1st Count		0	2	4	2	0	1

Table 21: Full results of predictive usefulness (IC and RankIC) for synthetic K-line generation experiments. Higher IC and RankIC scores suggest the generated data is more useful for building predictive financial models. Best and second best results are marked with red underline and blue underline, respectively.

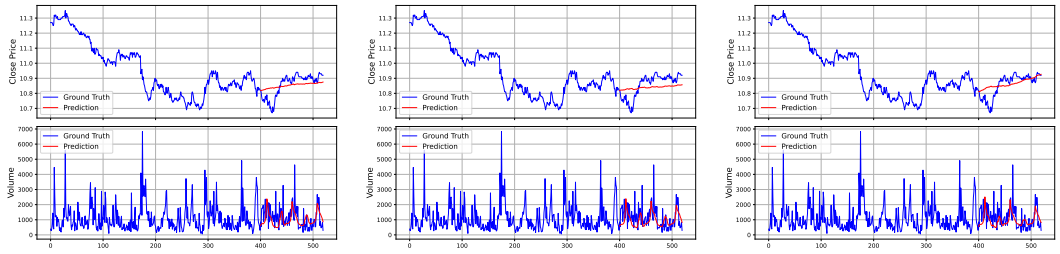
Models			Kronos (Ours)			Time-series Generative Models		
Metrics			Kronos_{small}	Kronos_{base}	Kronos_{large}	DiffusionTS	TimeVAE	TimeGAN
XSHG	15min	IC	0.0223	0.0231	0.0236	0.0103	0.0098	0.0102
		RankIC	0.0144	0.0147	0.0151	0.0087	0.0134	0.0081
	daily	IC	0.0918	0.0902	0.0845	0.0760	−0.0789	0.0108
		RankIC	0.0854	0.0839	0.0796	0.0684	−0.0720	0.0150
XTAI	15min	IC	0.0230	0.0274	0.0281	0.0074	−0.0118	0.0045
		RankIC	0.0226	0.0276	0.0299	0.0037	−0.0092	−0.0003
	daily	IC	0.0460	0.0437	0.0560	0.0013	−0.0213	0.0118
		RankIC	0.0445	0.0431	0.0551	−0.0001	−0.0193	0.0118
Crypto	15min	IC	0.0237	0.0243	0.0237	−0.0016	−0.0012	0.0096
		RankIC	0.0222	0.0231	0.0231	−0.0026	−0.0016	0.0079
	daily	IC	0.0027	0.0051	0.0037	−0.0085	−0.0130	−0.0330
		RankIC	0.0028	0.0049	0.0031	−0.0111	−0.0100	−0.0301
Forex	15min	IC	0.0202	0.0172	0.0171	0.0156	−0.0150	0.0095
		RankIC	0.0183	0.0158	0.0150	0.0142	−0.0140	0.0094
	daily	IC	0.0044	0.0069	0.0042	0.0016	0.0140	−0.0044
		RankIC	0.0042	0.0066	0.0045	0.0007	0.0160	−0.0058
Average		IC	0.0293	0.0297	0.0301	0.0128	−0.0147	0.0024
		RankIC	0.0268	0.0275	0.0282	0.0102	−0.0121	0.0020
1st Count			4	4	9	0	2	0



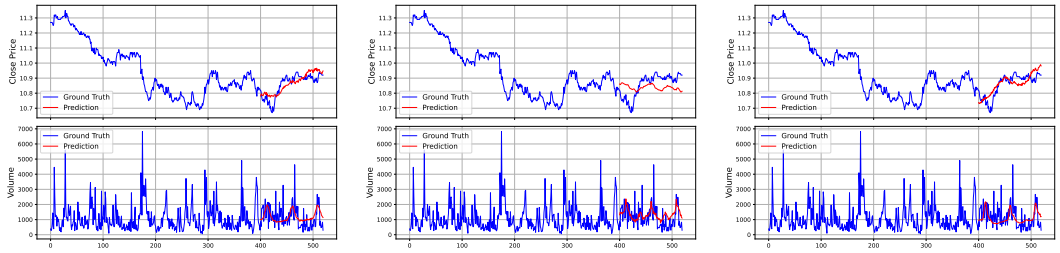
(a) Kronos$_{small}$ (b) Kronos$_{base}$ (c) Kronos$_{large}$



(d) TimeMOE$_{small}$ (e) TimeMOE$_{large}$ (f) TimesFM

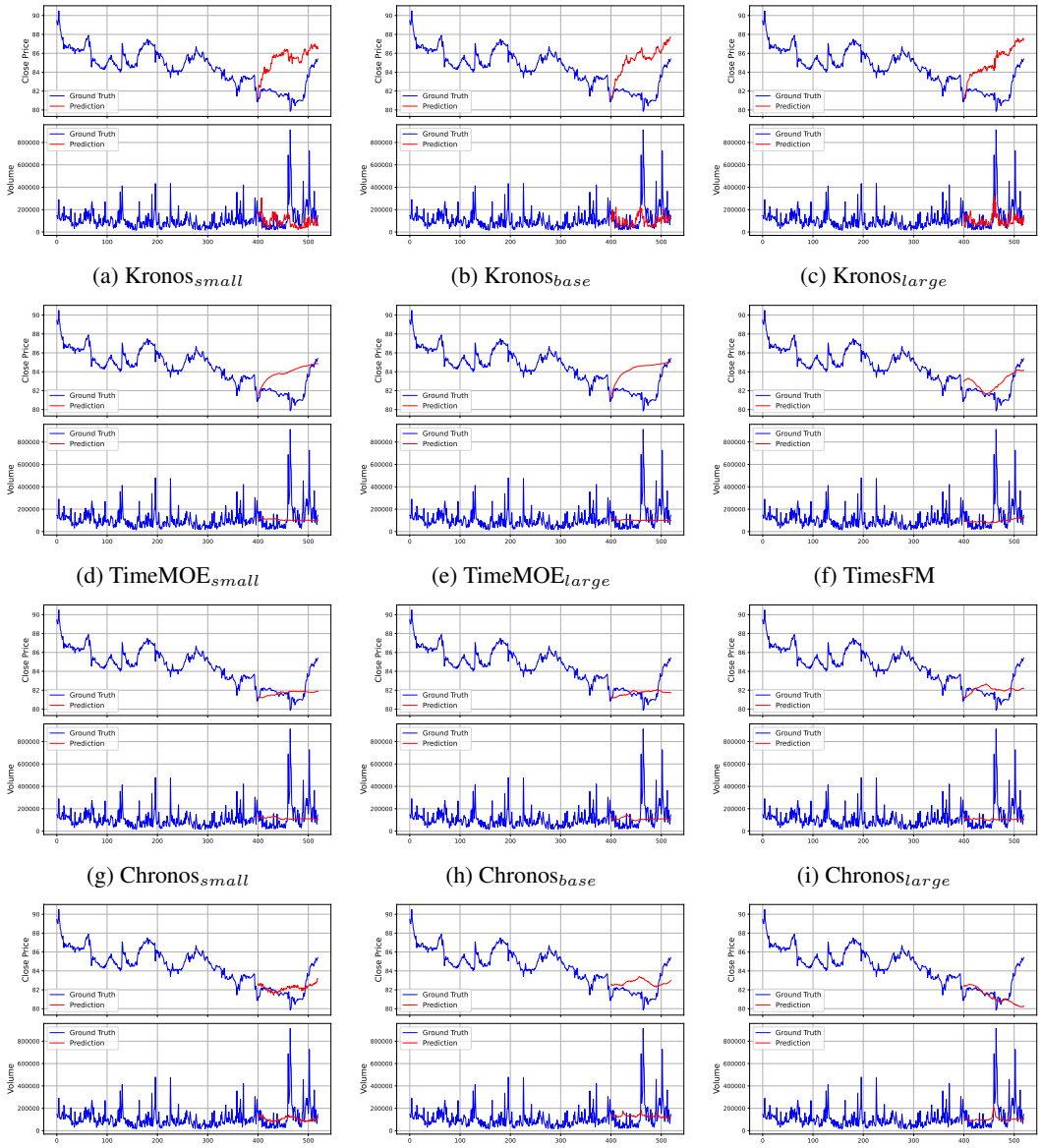


(g) Chronos$_{small}$ (h) Chronos$_{base}$ (i) Chronos$_{large}$



(j) iTransformer (k) DLinear (l) TimesNet

Figure 15: Forecasting results for the 'Close Price' and 'Volume' of China Film Co.,Ltd. (SSE: 600977), based on 5-minute K-line data. The model uses a 400-step look-back window to predict a 120-step horizon. Blue lines represent the ground truths and red lines are the model's predictions.



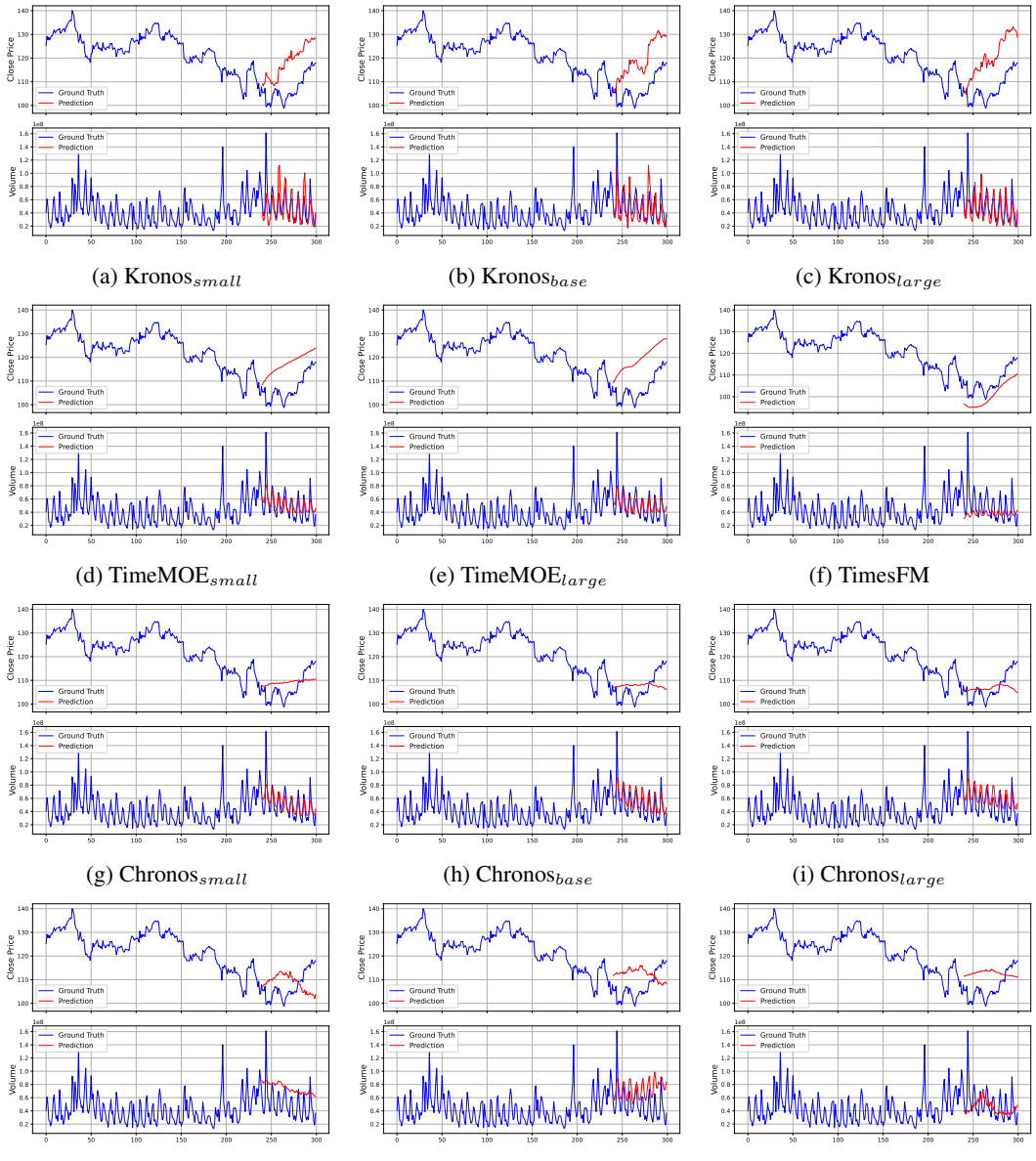
(a) Kronos$_{small}$ (b) Kronos$_{base}$ (c) Kronos$_{large}$

(d) TimeMOE$_{small}$ (e) TimeMOE$_{large}$ (f) TimesFM

(g) Chronos$_{small}$ (h) Chronos$_{base}$ (i) Chronos$_{large}$

(j) iTransformer (k) DLinear (l) TimesNet

Figure 16: Forecasting results for the 'Close Price' and 'Volume' of Pop Mart (HKEX: 09992), based on 5-minute K-line data. The model uses a 400-step look-back window to predict a 120-step horizon. Blue lines represent the ground truths and red lines are the model's predictions.



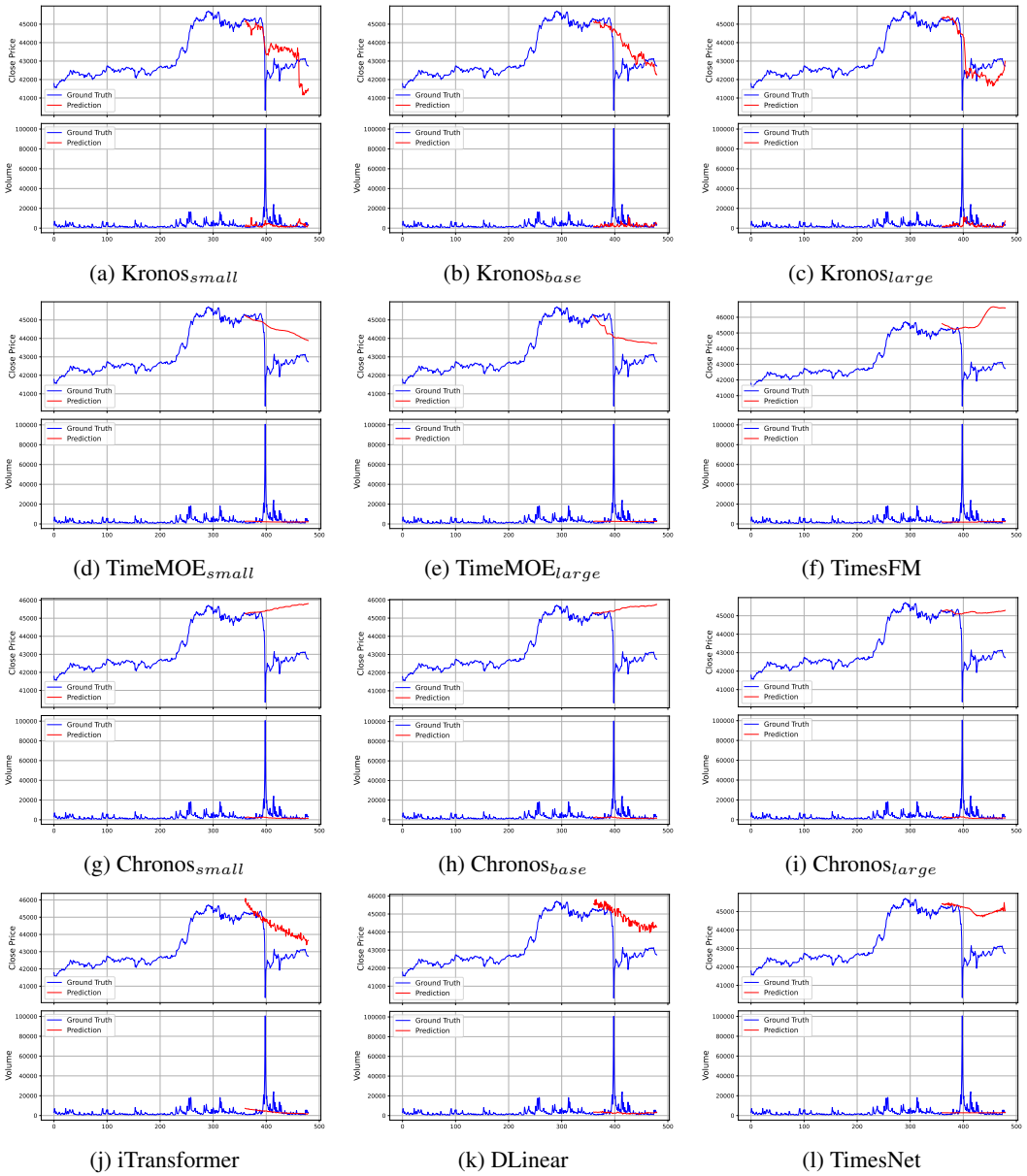
(a) Kronos$_{small}$ (b) Kronos$_{base}$ (c) Kronos$_{large}$

(d) TimeMOE$_{small}$ (e) TimeMOE$_{large}$ (f) TimesFM

(g) Chronos$_{small}$ (h) Chronos$_{base}$ (i) Chronos$_{large}$

(j) iTransformer (k) DLinear (l) TimesNet

Figure 17: Forecasting results for the 'Close Price' and 'Volume' of NVIDIA (NASDAQ: NVDA), based on 1-hour K-line data. The model uses a 240-step look-back window to predict a 60-step horizon. Blue lines represent the ground truths and red lines are the model's predictions.



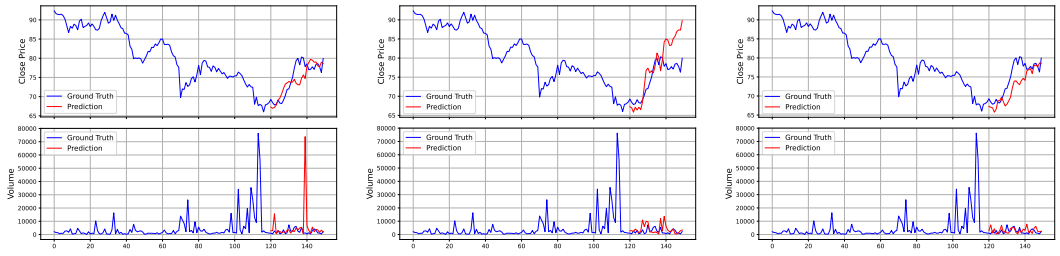
(a) Kronos$_{small}$ (b) Kronos$_{base}$ (c) Kronos$_{large}$

(d) TimeMOE$_{small}$ (e) TimeMOE$_{large}$ (f) TimesFM

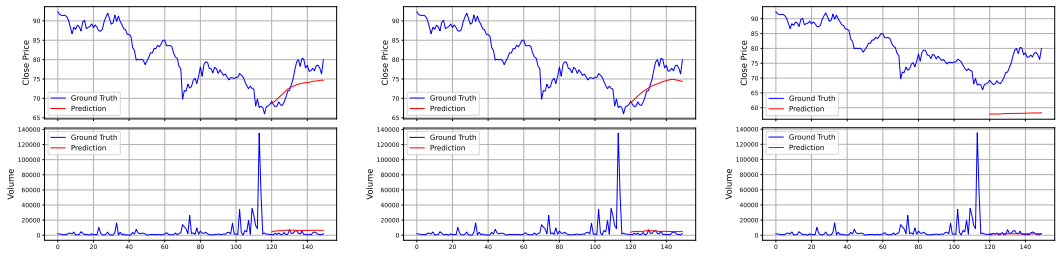
(g) Chronos$_{small}$ (h) Chronos$_{base}$ (i) Chronos$_{large}$

(j) iTransformer (k) DLinear (l) TimesNet

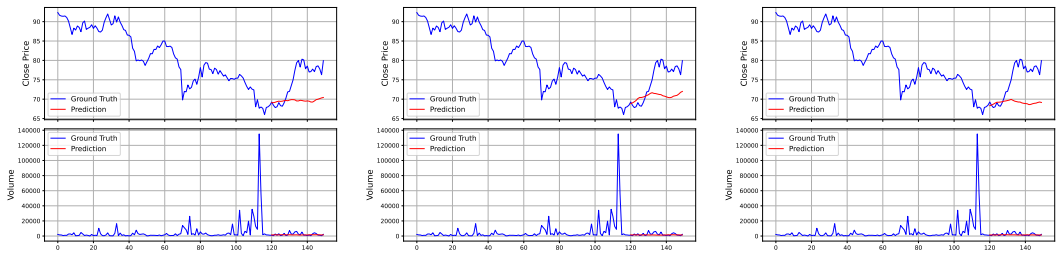
Figure 18: Forecasting results for the 'Close Price' and 'Volume' of the BTC/USDT perpetual contract on Binance, based on 15-minute K-line data. The model uses a 360-step look-back window to predict a 120-step horizon. Blue lines represent the ground truths and red lines are the model's predictions.



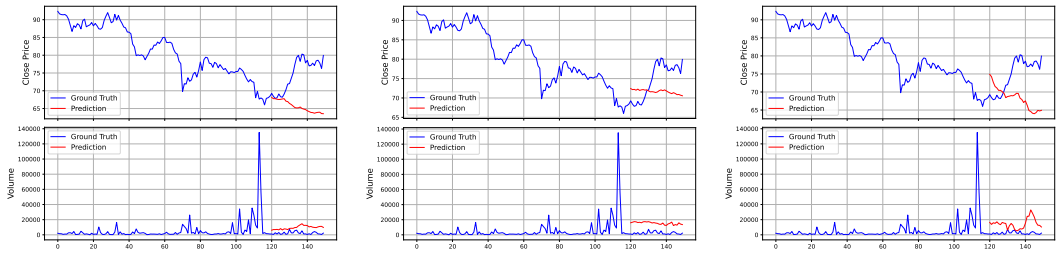
(a) Kronos$_{small}$ (b) Kronos$_{base}$ (c) Kronos$_{large}$



(d) TimeMOE$_{small}$ (e) TimeMOE$_{large}$ (f) TimesFM



(g) Chronos$_{small}$ (h) Chronos$_{base}$ (i) Chronos$_{large}$



(j) iTransformer (k) DLinear (l) TimesNet

Figure 19: Forecasting results for the 'Close Price' and 'Volume' of BMW (FWB: BMW), based on daily K-line data. The model uses a 120-step look-back window to predict a 30-step horizon. Blue lines represent the ground truths and red lines are the model's predictions.