Enhance Time Series Modeling by Integrating LLM

Can (Sam) Chen
Borealis AI & MILA & McGill University
chencan421@gmail.com

Gabriel L. Oliveira
Borealis AI

Hossein Sharifi-Noghabi
Borealis AI

Tristan Sylvain
Borealis AI

{gabriel.oliveira, hossein.sharifi, tristan.sylvain}@borealisai.com

Abstract

Time series (TS) modeling is critical in dynamic systems like weather prediction and anomaly detection. Recent work leverages Large Language Models (LLMs) for TS modeling due to their strong pattern recognition abilities. However, these approaches often prioritize LLMs as the predictive backbone, neglecting traditional TS models' mathematical aspects, like periodicity. Conversely, ignoring LLMs overlooks their pattern recognition strengths. To bridge this gap, we propose *LLM-TS Integrator*, a framework that integrates LLM capabilities with traditional TS modeling. At its core is the *mutual information* module, where a traditional TS model is enhanced with LLM-derived insights, improving predictive performance by maximizing mutual information between TS representations and their LLM-generated textual counterparts. We also address the varying importance of samples for traditional prediction and mutual information maximization. To handle this, we introduce the *sample reweighting* module, which assigns dual weights to each sample—one for prediction loss and another for mutual information loss—dynamically optimized through bi-level optimization. Our method achieves state-of-the-art or comparable performance across five key TS tasks: short-term and long-term forecasting, imputation, classification, and anomaly detection. Our code is available.

1 Introduction

Time series (TS) modeling is vital across various applications, such as weather forecasting [67], economic data imputation [22], anomaly detection in industrial monitoring [23], and action recognition [20], as highlighted by [32]. Its practical significance continues to draw considerable interest [38, 63].

Recent trends in TS increasingly incorporate Large Language Models (LLMs) for their pattern recognition abilities [33, 80, 35, 54, 24, 5]. However, these approaches often neglect specialized mathematical techniques utilized in traditional TS models, such as Fourier Transform for periodic patterns [66]. Balancing the strengths of LLMs with traditional TS models can enhance predictive performance. To this end, we propose *LLM-TS Integrator*, a framework that integrates LLM capabilities with traditional TS models.

At the core of our framework is a *mutual information* module, shown in Figure 1(a). We enhance traditional models, primarily TimesNet [66], by integrating insights from LLMs. This enhancement maximizes mutual information [55] between TS representations from traditional models and their LLM-derived textual counterparts. Given that textual descriptions are often absent from TS data, we generate them using a template enriched with background and statistical details relevant to the TS, aiding LLMs in better understanding the context.

NeurIPS 2024 Workshop on Time Series in the Age of Large Models.

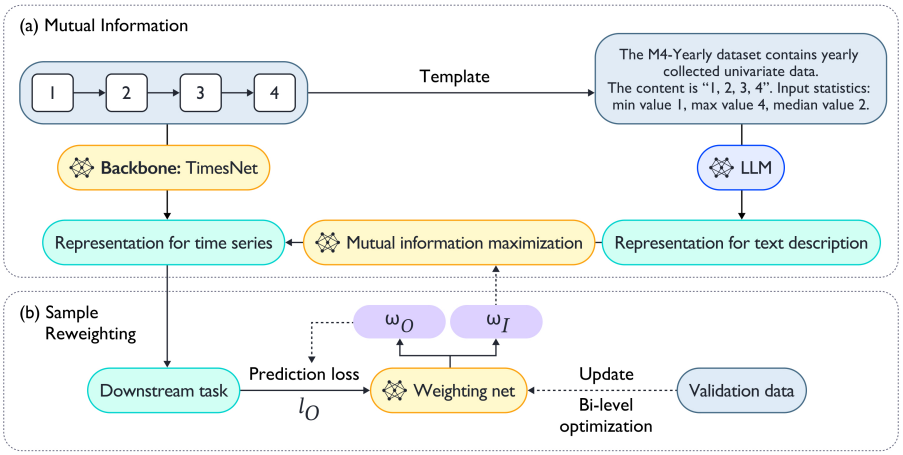

Figure 1: Illustration of *LLM-TS Integrator*. Module (a) enhances the traditional TS model (TimesNet) with LLM-derived insights by mutual information maximization. Module (b) optimizes sample importance for both prediction loss and mutual information loss to improve information utilization.

Our dual loss framework handles traditional prediction and mutual information, acknowledging that sample importance varies between the two. To address this, we introduce a *sample reweighting* module using an MLP (Figure 1(b)). This module adjusts weights for prediction and mutual information losses through bi-level optimization, enhancing information utilization.

Our primary contributions are as follows:

- Our method consists of *mutual information* and *sample reweighting*. The first module enhances traditional TS modeling with capabilities from LLMs through mutual information maximization.
- The second module optimizes sample importance for both prediction loss and mutual information loss, which improves information utilization.
- Extensive experiments across five TS tasks demonstrate the effectiveness of our framework.

2 Preliminaries

TimesNet. In this paper, we mainly choose TimesNet as the traditional predictive model due to its exceptional performance [66] and also explore other additional traditional models including ETSformer [65], Stationary [42], and FreTS [70] in Section A.9. Previous studies on modeling temporal variations in 1D time series often struggle with complex temporal patterns. TimesNet addresses this challenge by decomposing these complex variations into multiple intra-period and inter-period variations.

For the time series $\boldsymbol{x}$, we derive its representation $\boldsymbol{h}_{\boldsymbol{\theta}}^{m}(\boldsymbol{x})$ using the TimesNet model parameterized by $\boldsymbol{\theta}$ where m represents the *model*.

Large Language Models. Language models are trained on extensive collections of natural language sequences, with each sequence consisting of multiple tokens. Each language model uses a tokenizer that breaks down an input string into a sequence of recognizable tokens. However, the training of current large language models is solely focused on natural language, not encompassing time series data. This limitation presents challenges for the direct application of LLMs to time series analysis.

3 Method

This section introduces the *LLM-TS Integrator* framework, whose overall process is in Algorithm 1.

3.1 Mutual Information

Previous studies [80, 34] often prioritize LLMs as the primary predictive tool in TS analysis, neglecting the specialized mathematical modeling in traditional TS models, such as periodicity.

Our approach uses a traditional TS model, enhanced by LLMs, as the predictive backbone. In this work, we use TimesNet (Section 2) and explore other models in Section A.9. This hybrid method

Algorithm 1 LLM-TS Integrator

Input: TS dataset $\mathcal{D}$, number of training iterations T.
Output: Trained TS model parameterized by $\boldsymbol{\theta}^*$.

```
/* Mutual Information Module */
Train a traditional TS model (e.g., TimesNet) parameterized by θ using D.
Generate text description t for TS sample x via a designed template.
Derive hidden representations h^m_θ(x) from the TS model and h^l(t) from the LLM.
while τ ≤ T − 1 do
    Sample x, t, y from D, where y are the labels.
    Optimize a discriminator model T_β to estimate mutual information.
    /* Sample Reweighting Module */
    Process sample loss l_O with the weighting net to produce dual weights (Eq. 2, 3).
    Apply bi-level optimization to update the weighting net (Eq. 8, 9).
    Re-calculate dual weights using the updated weighting net (Eq. 2, 3).
    Calculate the overall loss to update the TS model (Eq. 4).
end while
Return the trained TS model parameterized by θ*.
```

combines traditional TS models' strengths with modern LLMs via a *mutual information* module, which maximizes the mutual information between TS representations from the traditional model and their textual representations from LLMs.

Estimating mutual information between a TS sample $\boldsymbol{x}$ and its textual description $\boldsymbol{t}$ is crucial. We derive $\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x})$ using TimesNet, parameterized by $\boldsymbol{\theta}$, and $\boldsymbol{h}^l(\boldsymbol{t})$ from a pre-trained LLM. We primarily use the LLaMA-3b model [59], while also evaluating others (Section A.9). Mutual information is estimated using the Jensen-Shannon MI estimator [55, 45], with further exploration of the MINE estimator [27] (Appendix A.19). Given TS set $\mathbb{S}$, mutual information is calculated as:

$$I(\boldsymbol{\theta}, \boldsymbol{\beta}) = \mathbb{E}_{\mathbb{S}}[-sp(-T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{h}^l(\boldsymbol{t}))] - \mathbb{E}_{\mathbb{S}\times\tilde{\mathbb{S}}}[sp(T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{h}^l(\tilde{\boldsymbol{t}}))], \tag{1}$$

where $T_{\boldsymbol{\beta}}$ is the discriminator and sp is the softplus function. The mutual information estimation alternates between optimizing $\boldsymbol{\beta}$ and refining $\boldsymbol{\theta}$ to maximize mutual information.

We assume each TS sample $\boldsymbol{x}$ is paired with a corresponding textual description, $\boldsymbol{t}$. Since textual descriptions are often unavailable, we generate them using a template that captures essential background and statistical details of the TS as detailed in Appendix A.1.

3.2 Sample Reweighting

Our *mutual information* module introduces two distinct loss functions: (1) the original sample prediction loss, $l_O(\boldsymbol{x}, \boldsymbol{y})$, hereafter referred to as l_O, which corresponds to the prediction loss for a TS sample $\boldsymbol{x}$ and its label $\boldsymbol{y}$, and (2) the mutual information maximization loss, denoted as $-I(\boldsymbol{\theta}, \boldsymbol{\beta})$. We acknowledge that the significance of samples varies between these two losses. Specifically, a large prediction loss l_O indicates a sample's substantial learning potential, thereby justifying a higher weight ω_O for its prediction loss. Conversely, this suggests that the sample's representation may be suboptimal for mutual information computation, warranting a lower weight ω_I.

To address this disparity, we have developed a novel *sample reweighting* module. This module employs a two-layer MLP weighting network parameterized by $\boldsymbol{\alpha}$, which processes the sample prediction loss to produce a pair of weights:

$$\omega_O(\boldsymbol{\alpha}), \omega_I(\boldsymbol{\alpha}) = MLP_{\boldsymbol{\alpha}}(l_O) \tag{2}$$

This process involves converting the sample loss l_O into a latent code z through a hidden layer. The network then outputs dual weights:

$$\omega_O(\boldsymbol{\alpha}), \omega_I(\boldsymbol{\alpha}) = \sigma(m_O \cdot z), \sigma(m_I \cdot z) \tag{3}$$

where $m_O > 0$, $m_I < 0$ ensures a negative correlation between ω_O and ω_I. $\sigma(\cdot)$ denotes sigmoid.

For a batch of N samples, the weight vector $\boldsymbol{\omega_O(\alpha)} \in \mathbb{R}^N$ is directly applied to the original prediction loss vector $\boldsymbol{l_O} \in \mathbb{R}^N$, resulting in the weighted average loss calculated as $mean(\boldsymbol{\omega_O(\alpha)} \cdot \boldsymbol{l_O})$.

Similarly, the weight vector $\boldsymbol{\omega_I(\alpha)} \in \mathbb{R}^N$ not only reflects the overall significance of the mutual information but also each sample's individual contribution to this metric. The mean of these weights $mean(\boldsymbol{\omega_I(\alpha)})$ represents the overall importance. For mutual information computations, $\boldsymbol{\omega_I(\alpha)}$ is transformed into a probability distribution $p_I^i = \frac{\omega_I^i}{\sum_{i=1}^N \omega_I^i}$. This adjustment affects the distribution used in mutual information calculations, necessitating a recalculation of mutual information as $I(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\alpha})$, with details provided in Appendix A.2. As a result, the overall loss is formulated as:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha}) = mean(\boldsymbol{\omega_O(\alpha)} \cdot \boldsymbol{l_O}) mean(\boldsymbol{\omega_I(\alpha)}) \cdot [-I(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\omega_I(\alpha)})] \tag{4}$$

We optimize the weighting network $\boldsymbol{\alpha}$ by leveraging the supervision signals from a small validation dataset as detailed in Appendix A.3

4 Experimental Results

Baselines. Our evaluation employs a comprehensive array of baseline models across several architectural designs **(1)** CNN-based models, specifically TimesNet [66]; **(2)** MLP-based models, including LightTS [74] and DLinear [73]; **(3)** Transformer-based models, such as Reformer [36], Informer [77], Autoformer [67], FEDformer [79], Nonstationary Transformer [42], ETSformer [65], and PatchTST [44]; **(4)** LLM-based models, represented by GPT4TS [80].

Additional comparisons for forecasting tasks include LLM-based models like Time-LLM [34] and TEST [54]. For short-term forecasting, models like N-HiTS [7] and N-BEATS [46] are included. Anomaly detection tasks are assessed using Anomaly Transformer [68], and for classification tasks, models such as XGBoost [16], Rocket [17], LSTNet [37], LSSL [26], Pyraformer [41], TCN [20], and Flowformer [30] are considered.

Main Results. Figure 2 demonstrates that our *LLM-TS Integrator* consistently outperforms other

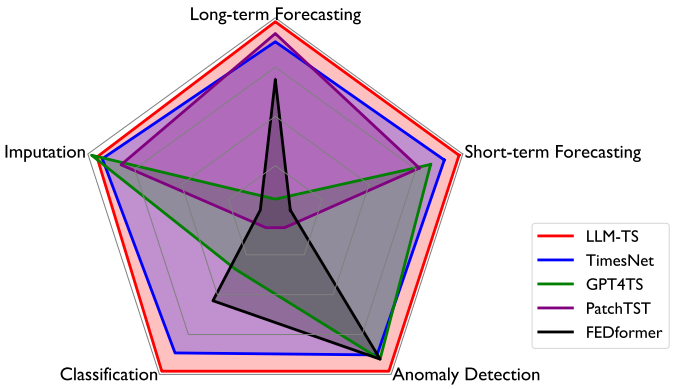

Figure 2: Model performance across different tasks.

methods in various tasks, underscoring its efficacy. We will refer to our method as *LLM-TS* in the tables for brevity. Unless otherwise indicated, we cite results from TimesNet [66]. We reproduce TimesNet and GPT4TS [80] experiments for all tasks. All results are averages from three runs with different seeds. Standard deviations for ablation studies are detailed in Appendix 8. The best results are highlighted in bold, with the second-best underlined. Detailed results are in Appendix A.5, A.6, A.7 and A.8 and we also conduct ablation studies in Appendix A.9. We also (1) present several showcases of our method in Appendix A.11 and (2) discuss the model efficiency in Appendix A.13. We detail the experimental setting in Appendix A.4.

5 Conclusion and Discussion

In conclusion, the *LLM-TS Integrator* offers a promising approach to integrating Large Language Models (LLMs) with traditional TS methods. By encouraging high mutual information between textual and TS data, our method aims to maintain the distinct characteristics of time series while benefiting from the advanced pattern recognition capabilities of LLMs. We also discuss the related work in Appendix A.20. Building on this workshop paper, we present an extended version in [10].

References

[1] Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. Practical approach to asynchronous multivariate time series anomaly detection and localization. *KDD*, 2021.

[2] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[3] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: Mutual information neural estimation, 2021.

[4] Anthony J Bell and Terrence J Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7(6):1129–1159, 1995.

[5] Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *ICLR*, 2024.

[6] CDC. Illness. `https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html`, 2024. Online; accessed 10 August 2024.

[7] Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[8] Can Chen, Xi Chen, Chen Ma, Zixuan Liu, and Xue Liu. Gradient-based bi-level optimization for deep learning: A survey. *arXiv preprint arXiv:2207.11719*, 2022.

[9] Can Chen, Chen Ma, Xi Chen, Sirui Song, Hao Liu, and Xue Liu. Unbiased implicit feedback via bi-level optimization. *arXiv preprint arXiv:2206.00147*, 2022.

[10] Can Chen, Gabriel Oliveira, Hossein Sharifi Noghabi, and Tristan Sylvain. Llm-ts integrator: Integrating llm for enhanced time series modeling. *arXiv preprint arXiv:2410.16489*, 2024.

[11] Can Chen, Yingxue Zhang, Xue Liu, and Mark Coates. Bidirectional learning for offline model-based biological sequence design. In *International Conference on Machine Learning*, pages 5351–5366. PMLR, 2023.

[12] Can Chen, Yingxueff Zhang, Jie Fu, Xue Steve Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. *Advances in Neural Information Processing Systems*, 2022.

[13] Can Chen, Shuhao Zheng, Xi Chen, Erqun Dong, Xue Steve Liu, Hao Liu, and Dejing Dou. Generalized data weighting via class-level gradient manipulation. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

[14] Can Chen, Jingbo Zhou, Fan Wang, Xue Liu, and Dejing Dou. Structure-aware protein self-supervised learning. *Bioinformatics*, 39, 2023.

[15] Can Sam Chen, Christopher Beckham, Zixuan Liu, Xue Steve Liu, and Chris Pal. Parallel-mentoring for offline model-based optimization. *Advances in Neural Information Processing Systems*, 2024.

[16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[17] Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[19] Tongtong Fang, Nan Lu, Gang Niu, and Masashi Sugiyama. Generalizing importance weighting to a universal solver for distribution shift problems. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2024.

[20] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

[21] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 1997.

[22] Milton Friedman. The interpolation of time series by related series. *Journal of the American Statistical Association*, 1962.

[23] Jingkun Gao, Xiaomin Song, Qingsong Wen, Pichao Wang, Liang Sun, and Huan Xu. Robusttad: Robust time series anomaly detection via decomposition and convolutional neural networks. *arXiv preprint arXiv:2002.09545*, 2020.

[24] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36, 2024.

[25] Nate Gruver, Marc Anton Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[26] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[27] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[28] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization, 2019.

[29] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[30] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer: A transformer architecture for optical flow. In *European conference on computer vision*, pages 668–685. Springer, 2022.

[31] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Söderström. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. *KDD*, 2018.

[32] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[33] Yushan Jiang, Zijie Pan, Xikun Zhang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. Empowering time series analysis with large language models: A survey. *arXiv preprint arXiv:2402.03182*, 2024.

[34] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

[35] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[36] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020.

[37] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.

[38] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philos. Trans. Royal Soc. A*, 2021.

[39] Ralph Linsker. Self-organization in a perceptual network. *Computer*, 21:105–117, 1988.

[40] Risheng Liu, Jiaxin Gao, Jin Zhang, Deyu Meng, and Zhouchen Lin. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[41] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *ICLR*, 2021.

[42] 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.

[43] Aditya P. Mathur and Nils Ole Tippenhauer. Swat: a water treatment testbed for research and training on ICS security. In *CySWATER*, 2016.

[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] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems*, 29, 2016.

[46] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *ICLR*, 2019.

[47] PeMS. Traffic. `http://pems.dot.ca.gov/`, 2024. Online; accessed 10 August 2024.

[48] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[49] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *Proc. Int. Conf. Machine Lea. (ICML)*, 2018.

[50] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2019.

[51] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information, 2017.

[52] Spyros Makridakis. M4 dataset, 2018.

[53] Ya Su, Y. Zhao, Chenhao Niu, Rong Liu, W. Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. *KDD*, 2019.

[54] Chenxi Sun, Yaliang Li, Hongyan Li, and Shenda Hong. Test: Text prototype aligned embedding to activate llm's ability for time series. *arXiv preprint arXiv:2308.08241*, 2023.

[55] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000*, 2019.

[56] Yanmin Sun, Mohamed S Kamel, Andrew KC Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern recognition*, 2007.

[57] Tristan Sylvain, Linda Petrini, and Devon Hjelm. Locality and compositionality in zero-shot learning. In *International Conference on Learning Representations*, 2019.

[58] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method, 2000.

[59] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[60] UCI. Electricity. `https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014`, 2015. Online; accessed 10 August 2024.

[61] 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. In *ICLR*, 2023.

[62] Zitai Wang, Qianqian Xu, Zhiyong Yang, Yuan He, Xiaochun Cao, and Qingming Huang. A unified generalization analysis of re-weighting and logit-adjustment for imbalanced learning. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2024.

[63] Qingsong Wen, Linxiao Yang, Tian Zhou, and Liang Sun. Robust time series analysis and applications: An industrial perspective. In *KDD*, 2022.

[64] Wetterstation. Weather. `https://www.bgc-jena.mpg.de/wetter/`, 2024. Online; accessed 10 August 2024.

[65] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven C. H. Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*, 2022.

[66] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023.

[67] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *NeurIPS*, 2021.

[68] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *ICLR*, 2021.

[69] Hao Xue and Flora D Salim. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[70] Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain mlps are more effective learners in time series forecasting. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2024.

[71] Huining Yuan, Hongkun Dou, Xingyu Jiang, and Yue Deng. Task-aware world model learning with meta weighting via bi-level optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2024.

[72] Ye Yuan, Can Sam Chen, Zixuan Liu, Willie Neiswanger, and Xue Steve Liu. Importance-aware co-teaching for offline model-based optimization. *Advances in Neural Information Processing Systems*, 2024.

[73] 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*, 2023.

[74] Tianping Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.

[75] Xiaoying Zhang, Junpu Chen, Hongning Wang, Hong Xie, Yang Liu, John Lui, and Hang Li. Uncertainty-aware instance reweighting for off-policy learning. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2024.

[76] Hao Zhou, Chengming Hu, Ye Yuan, Yufei Cui, Yili Jin, Can Chen, Haolun Wu, Dun Yuan, Li Jiang, Di Wu, et al. Large language model (llm) for telecommunications: A comprehensive survey on principles, key techniques, and opportunities. *arXiv preprint arXiv:2405.10825*, 2024.

[77] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, 2021.

[78] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

[79] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, 2022.

[80] Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36, 2024.

A Appendix

A.1 Template

```
template = (
    "{task_description}. The content is: {TS}. "
    "Input statistics: min value {min(TS)}, max value {max(TS)}, "
    "median value {median(TS)}, top 5 lags {compute_lags(TS)}."
)
```

A.2 Mutual Information Recalculation

Recall that mutual information can be calculated using the equation:

$$I(\boldsymbol{\theta}, \boldsymbol{\beta}) = \mathbb{E}_{\mathbb{S}}[-sp(-T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{h}^l(\boldsymbol{t}))] - \mathbb{E}_{\mathbb{S}\times\tilde{\mathbb{S}}}[sp(T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{h}^l(\tilde{\boldsymbol{t}}))]\,, \tag{5}$$

where $T_{\boldsymbol{\beta}}$ signifies the discriminator characterized by parameters $\boldsymbol{\beta}$, and sp denotes the softplus function. Notably, $(\boldsymbol{x}, \boldsymbol{t})$ symbolizes a sample from the dataset $\mathbb{S}$, while $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{t}})$ represents a different sample from the dataset $\tilde{\mathbb{S}} = \mathbb{S}$.

This formulation presumes a uniform distribution of samples. However, we have already computed probabilities p^i_I for each sample, which introduces a non-uniform distribution. For a batch of N samples, the expected value is computed as

$$\mathbb{E}_{\mathbb{S}}[-sp(-T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{h}^l(\boldsymbol{t}))] = -\sum_{i=1}^{N} p^i_I sp(-T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x^i}), \boldsymbol{h}^l(\boldsymbol{t^i})). \tag{6}$$

$$\mathbb{E}_{\mathbb{S}\times\tilde{\mathbb{S}}}[sp(T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{h}^l(\tilde{\boldsymbol{t}}))] = \sum_i \sum_{i\neq j} \hat{p}^{ij} sp(T_{\boldsymbol{\beta}}(\boldsymbol{h}^m_{\boldsymbol{\theta}}(\boldsymbol{x^i}), \boldsymbol{h}^l(\tilde{\boldsymbol{t}}^{\boldsymbol{j}})). \tag{7}$$

Here, $\hat{p}^{ij}$ is defined as $\frac{p^i_I \cdot p^j_I}{\sum_i \sum_{i\neq j} p^i_I \cdot p^j_I}$, adjusting for the non-uniform distribution of sample probabilities. As p^i_I is produced from the weighting network $\boldsymbol{\alpha}$, we can also write $I(\boldsymbol{\theta}, \boldsymbol{\beta})$ as $I(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\alpha})$.

A.3 Bi-level Optimization

The ensuing challenge is optimizing the weighting network $\boldsymbol{\alpha}$. We achieve this by leveraging the supervision signals from a small validation dataset. If the weighting network is properly optimized, the model trained with these weights is expected to show improved performance on the validation dataset in terms of the validation loss $\mathcal{L}_V(\boldsymbol{\theta}) = \frac{1}{M}\sum_j^M l^j_O(\boldsymbol{x}_j, \boldsymbol{y}_j)$, where M denotes the size of the validation set. This constitutes a bi-level optimization problem, which is widely used in machine learning [29, 8, 40, 15, 9, 12, 14, 11]. At the inner level, model training is conducted through:

$$\hat{\boldsymbol{\theta}}(\boldsymbol{\alpha}) = \boldsymbol{\theta} - \eta_1 \cdot \frac{\partial \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha})}{\partial \boldsymbol{\theta}} \tag{8}$$

The objective is to ensure that the model performs optimally on the validation dataset:

$$\hat{\boldsymbol{\alpha}} = \boldsymbol{\alpha} - \eta_2 \cdot \frac{\partial \mathcal{L}_V(\boldsymbol{\theta}(\boldsymbol{\alpha}))}{\partial \boldsymbol{\alpha}} \tag{9}$$

Both η_1 and η_2 represent the learning rates for the respective optimization steps. Through the minimization of the validation loss, we aim to optimize the weighting network $\boldsymbol{\alpha}$.

A.4 Experimental Settings

Following [50], the weighting network comprises a two-layer MLP with a hidden size of 100, and we set the learning rate η_2 for this network at 0.001. The learning rate η_0 of the discriminator is set as 0.001 at the first epoch and then decreases to 0.0001 for the rest of epochs.

A.5 Short- and Long- Term Forecasting

Setup.

To comprehensively assess our framework's forecasting capabilities, we engage it in both short- and long-term forecasting settings. In the realm of short-term forecasting, we utilize the M4 dataset [52], which aggregates univariate marketing data on a yearly, quarterly, and monthly basis. For long-term forecasting, we examine five datasets following [80]: ETT [78], Electricity [60], Traffic [47], Weather [64], and ILI [6]. We adhere to the TimesNet setting with an input length of 96. For LLM-based methods like GPT4TS and Time-LLM, which use different input lengths, we rerun the experiments using their code. For PatchTST, we cite the results from [61], as the original PatchTST uses an input length of 512. Due to shorter input lengths in this study compared to the original, the reported performance is lower.

Results.

As shown in Tables 1 and 2, our *LLM-TS* performs exceptionally well in both short- and long-term settings. It consistently surpasses TimesNet, highlighting the effectiveness of incorporating LLM-derived insights. Furthermore, it generally outperforms other LLM-based methods such as GPT4TS, TIME-LLM, and TEST, underscoring the advantages of integrating traditional TS modeling.

Table 1: Short-term M4 forecasting. The prediction lengths are in [6, 48] and results are obtained by weighting averages across multiple datasets with varying sampling intervals. Full results are in Appendix A.14.

Methods	LLM-TS	TimesNet	GPT4TS	TIME-LLM	TEST	PatchTST	N-HiTS	N-BEATS	FEDformer	Stationary	Autoformer
SMAPE	**11.819**	11.908	11.991	11.983	11.927	12.059	11.927	11.851	12.840	12.780	12.909
MASE	**1.588**	1.612	1.600	1.595	1.613	1.623	1.613	1.599	1.701	1.756	1.771
OWA	**0.851**	0.860	0.861	0.859	0.861	0.869	0.861	0.855	0.918	0.930	0.939

Table 2: Long-term forecasting: Averages over 4 lengths: 24, 36, 48, 60 for ILI, and 96, 192, 336, 720 for others. Full results in Appendix A.15.

Methods	LLM-TS		TimesNet		TIME-LLM		DLinear		PatchTST		GPT4TS		FEDformer		TEST		Stationary		ETSformer	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MAE	MSE	MAE	MSE
Weather	**0.257**	**0.285**	0.265	0.290	0.279	0.296	0.265	0.317	0.265	**0.285**	0.275	0.292	0.309	0.360	0.291	0.315	0.288	0.314	0.271	0.334
ETTh1	0.454	**0.451**	0.470	0.462	0.474	0.459	0.456	0.452	0.516	0.484	0.473	**0.451**	**0.440**	0.460	**0.440**	0.460	0.570	0.537	0.542	0.510
ETTh2	0.396	0.413	0.413	0.426	0.398	0.415	0.559	0.515	0.391	**0.411**	**0.383**	0.410	0.437	0.449	0.414	0.432	0.526	0.516	0.439	0.452
ETTm1	**0.401**	0.409	0.414	0.418	0.437	0.421	0.403	0.407	0.406	0.407	0.408	**0.400**	0.448	0.452	0.402	0.411	0.481	0.456	0.429	0.425
ETTm2	0.295	**0.331**	0.294	**0.331**	0.298	0.342	0.350	0.401	**0.290**	0.334	**0.290**	0.335	0.305	0.349	0.323	0.359	0.306	0.347	0.293	0.342
ILI	**1.973**	**0.894**	2.266	0.974	2.726	1.098	2.616	1.090	2.184	0.906	5.117	1.650	2.847	1.144	3.324	1.232	2.077	0.914	2.497	1.004
ECL	0.194	0.299	0.198	0.298	0.229	0.315	0.212	0.300	0.216	0.318	0.206	**0.285**	0.214	0.327	0.237	0.324	**0.193**	0.296	0.208	0.323
Traffic	0.618	**0.333**	0.627	0.335	0.606	0.395	0.625	0.383	**0.529**	0.341	0.561	0.373	0.610	0.376	0.581	0.388	0.624	0.340	0.621	0.396
Average	**0.574**	**0.427**	0.618	0.442	0.681	0.468	0.686	0.483	0.600	0.436	0.964	0.525	0.701	0.489	0.756	0.491	0.633	0.465	0.662	0.473

A.6 Imputation

Setup.

To assess our method's imputation capabilities, we employ three datasets: ETT [78], Electricity [60], and Weather [64], serving as our benchmarks. To simulate various degrees of missing data, we randomly obscure time points at proportions of $\{12.5\%, 25\%, 37.5\%, 50\%\}$ following [66].

Results.

Table 3 illustrates that our method achieves performance comparable to GPT4TS and surpasses other baselines, highlighting its effectiveness. We attribute the robust performance of GPT4TS primarily to its backbone feature extractor: the pre-trained language model, which excels at capturing time series patterns, enhancing its imputation proficiency.

A.7 Classification

Setup.

We focus on the application of our method to sequence-level time series classification tasks, a crucial test of its ability to learn high-level representations from data. Specifically, we employ 10 diverse

Table 3: Imputation task: Randomly masked $\{12.5\%, 25\%, 37.5\%, 50\%\}$ of points in 96-length series, averaging results over 4 mask ratios. Full results are in Appendix A.16.

Methods	LLM-TS		TimesNet		GPT4TS		PatchTST		LightTS		DLinear		FEDformer		Stationary		Autoformer		Reformer	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.025**	**0.103**	0.028	0.109	0.028	0.108	0.047	0.140	0.104	0.218	0.093	0.206	0.062	0.177	0.036	0.126	0.051	0.150	0.055	0.166
ETTm2	**0.021**	**0.087**	0.022	0.089	0.023	0.088	0.029	0.102	0.046	0.151	0.096	0.208	0.101	0.215	0.026	0.099	0.029	0.105	0.157	0.280
ETTh1	0.087	0.198	0.090	0.199	**0.069**	**0.174**	0.115	0.224	0.284	0.373	0.201	0.306	0.117	0.246	0.094	0.201	0.103	0.214	0.122	0.245
ETTh2	**0.050**	0.148	0.051	0.150	**0.050**	**0.144**	0.065	0.163	0.119	0.250	0.142	0.259	0.163	0.279	0.053	0.152	0.055	0.156	0.234	0.352
ECL	0.094	0.211	0.095	0.212	0.091	0.207	**0.072**	**0.183**	0.131	0.262	0.132	0.260	0.130	0.259	0.100	0.218	0.101	0.225	0.200	0.313
Weather	**0.030**	0.056	0.031	0.059	0.032	0.058	0.034	**0.055**	0.055	0.117	0.052	0.110	0.099	0.203	0.032	0.059	0.031	0.057	0.038	0.087
Average	0.051	0.134	0.053	0.136	**0.049**	**0.130**	0.060	0.144	0.123	0.228	0.119	0.224	0.112	0.229	0.056	0.142	0.061	0.151	0.134	0.240

multivariate datasets sourced from the UEA Time Series Classification repository [2]. These datasets encompass a wide range of real-world applications, including gesture and action recognition, audio processing, medical diagnosis, among other practical domains. We reproduce the results of TEST based on their code [54].

Results.

As depicted in Figure 3, our *LLM-TS Integrator* achieves superior performance with an average accuracy of 73.4%. As detailed in Appendix A.17, it consistently outperforms other LLM-based methods across most tasks, including GPT4TS and TEST. We attribute this enhanced capability to the traditional TS modeling techniques in our framework, which effectively capture classification characteristics more adeptly than LLMs.

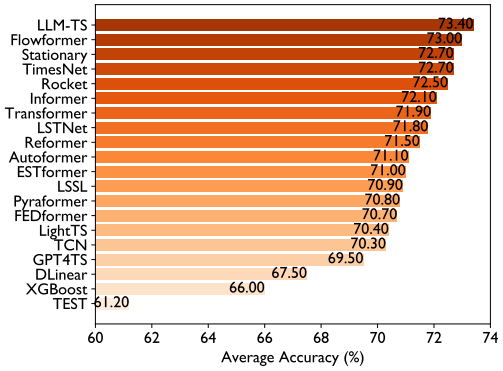

Figure 3: Model comparison in classification.

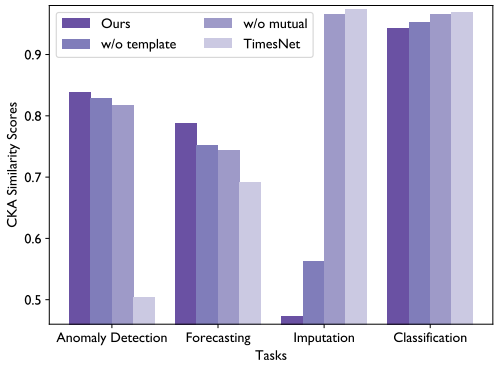

Figure 4: CKA by Task.

A.8 Anomaly Detection

Setup.

Our study concentrates on unsupervised time series anomaly detection, aiming to identify aberrant time points indicative of potential issues. We benchmark our method against five established anomaly detection datasets: SMD [53], MSL [31], SMAP [31], SWaT [43], and PSM [1]. These datasets span a variety of applications, including service monitoring, space and earth exploration, and water treatment processes. For a consistent evaluation framework across all experiments, we employ the classical reconstruction error metric to determine anomalies following [66].

Results.

As indicated in Table 4, our *LLM-TS Integrator* exhibits superior performance with an average F1-score of 85.17%. This result underscores the versatility of *LLM-TS*, demonstrating its capability not only in classifying complete sequences, as discussed previously, but also in effectively detecting anomalies in time series data.

Table 4: Anomaly detection task. F1-score (as %) is calculated per dataset. $*$. in the Transformers represents the name of $*$former. Full results are in Appendix A.18.

Methods	LLM-TS	TimesNet	GPT4TS	PatchTS.	ETS.	FED.	LightTS	DLinear	Stationary	Auto.	Pyra.	Anomaly.**	In.	Re.	Trans.
SMD	84.69	84.57	84.32	84.62	83.13	85.08	82.53	77.10	84.72	85.11	83.04	85.49	81.65	75.32	79.56
MSL	81.11	80.34	81.73	78.70	85.03	78.57	78.95	84.88	77.50	79.05	84.86	83.31	84.06	84.40	78.68
SMAP	69.41	69.18	68.86	68.82	69.50	70.76	69.21	69.26	71.09	71.12	71.09	71.18	69.92	70.40	69.70
SWaT	93.23	93.12	92.59	85.72	84.91	93.19	93.33	87.52	79.88	92.74	91.78	83.10	81.43	82.80	80.37
PSM	97.43	97.27	97.34	96.08	91.76	97.23	97.15	93.55	97.29	93.29	82.08	79.40	77.10	73.61	76.07
Average	**85.17**	84.90	<u>84.97</u>	82.79	82.87	84.97	84.23	82.46	82.08	84.26	82.57	80.50	78.83	77.31	76.88

Table 5: Results averaged over 4 prediction lengths.

Methods	Ours		w/o mutual		w/o template		w/o reweight		TimesNet	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	**0.257**	**0.285**	0.264	0.290	0.263	0.288	0.264	0.291	0.265	0.290
ETTh1	**0.454**	**0.451**	0.467	0.460	0.465	0.460	0.464	0.463	0.470	0.462
ETTm1	**0.401**	**0.409**	0.411	0.417	0.406	0.415	0.403	0.411	0.414	0.418
ILI	**1.973**	**0.894**	2.221	0.942	2.173	0.950	2.173	0.947	2.266	0.974

A.9 Ablations

In this section, we first verify the effectiveness of our framework by sequentially removing key components: (1) *mutual information* module and (2) *sample reweighting* module. Additionally, for *mutual information*, we explore the impact of removing the template while retaining the raw time series data inputs to the LLM. We denote these variants as *w/o mutual*, *w/o reweight* and *w/o template*. Our experiments span long-term forecasting tasks including Weather, ETTh1, ETTm1 and ILI. As detailed in Table 5, the removal of any component leads to a decrease in performance, confirming the value of each design element. Additionally, we explore the use of the MINE estimator [27] instead of the Jensen-Shannon MI estimator in our main paper, with further details provided in Appendix A.19. Lastly, we showcase various case studies to demonstrate the enhancements facilitated by our method in Appendix A.11 and explore template variations in Appendix A.12.

Mutual Information.

We further explore the *mutual information* module from a representation learning perspective, following the findings in [66]. They adopt a CKA (Centered Kernel Alignment) metric which measures similarity between representations obtained from the first and last layer of a model and they find that forecasting and anomaly detection benefits from high CKA similarity, contrasting with that imputation and classification tasks benefits from lower CKA similarity.

Experiments are conducted using the MSL dataset for the anomaly detection task, the Weather dataset for forecasting, the ETTh1 dataset for imputation, and the PEMS-SF dataset for classification. As depicted in Figure 4, the removal of components in our method results in decreased CKA similarity in anomaly detection and forecasting tasks, but an increase in imputation and classification tasks. This observation further substantiates the effectiveness of our components.

Sample Reweighting.

Regarding the *sample reweighting* module, we illustrate the behavior of the learned weighting network in Appendix A.19. The trend confirms our hypothesis: sample weight ω_O increases with the prediction loss l_O, and weight ω_I decreases as l_O increases. This pattern validates our *sample reweighting* module. Further discussion comparing this module to a fixed weight scheme are presented in Appendix A.19.

To verify the effectiveness of our method, we conduct ablation studies focusing on (1) traditional time series (TS) models and (2) language models.

Traditional Models.

Although we utilize TimesNet as our primary model, our framework is applicable to other traditional models. We explored additional traditional models including ETSformer [65], Stationary [42], and FreTS [70]. As shown in Table 6, integrating *LLM-TS* generally enhances performance across all traditional models, underscoring the benefits of our method.

Language Models.

In the main paper, the LLaMA-3b model [59] is used to generate embeddings for the TS language description. We compare it with GPT2 [48] and BERT [18] to assess different embeddings' performance. Table 7 reveals that LLaMA-3b generally outperforms the alternatives, and all LMs improve results compared to non-LLM approaches, validating the effectiveness of *LLM-TS Integrator*.

Table 6: Ablation results on different traditional models. Full results are in Appendix A.19.

Methods	ETSformer		ETS LLM-TS		Stationary		Stat LLM-TS		FreTS		FreTS LLM-TS	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$Weather$	0.313	0.382	0.307	0.375	0.282	0.307	0.284	0.309	0.262	0.306	0.255	0.302
$ETTh1$	0.799	0.684	0.791	0.678	0.667	0.582	0.653	0.572	0.484	0.473	0.478	0.466
$ETTm1$	0.638	0.583	0.555	0.528	0.527	0.477	0.522	0.471	0.415	0.422	0.407	0.415
ILI	3.922	1.367	3.740	1.320	2.722	1.041	2.205	0.935	3.449	1.279	3.158	1.211

Table 7: Ablation results on different LLM embeddings. Full results are in Appendix A.19.

Methods	LLM-TS (LLaMA)		LLaMA w/o template		GPT2		BERT		No LLM	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$Weather$	**0.257**	**0.285**	0.263	0.288	0.261	0.287	0.260	0.287	0.264	0.290
$ETTh1$	**0.454**	**0.451**	0.465	0.460	0.464	0.458	0.467	0.460	0.467	0.460
$ETTm1$	**0.401**	**0.409**	0.406	0.415	0.406	0.413	0.406	0.412	0.411	0.417
ILI	**1.973**	**0.894**	2.173	0.950	2.169	0.936	2.193	0.952	2.221	0.942

A.10 Standard Deviation Results

Table 8 presents the results along with standard deviations to underscore the consistency and reliability of our method's performance.

Table 8: Ablation results with standard deviation.

Methods		Ours		w/o mutual		w/o reweight		TimesNet	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$Weather$	96	0.166 ± 0.002	0.217 ± 0.002	0.168 ± 0.003	0.218 ± 0.005	0.181 ± 0.003	0.232 ± 0.001	0.174 ± 0.003	0.224 ± 0.002
	192	0.229 ± 0.003	0.269 ± 0.003	0.227 ± 0.004	0.268 ± 0.003	0.230 ± 0.002	0.270 ± 0.004	0.235 ± 0.001	0.272 ± 0.003
	336	0.278 ± 0.002	0.302 ± 0.003	0.298 ± 0.004	0.318 ± 0.003	0.283 ± 0.004	0.306 ± 0.002	0.285 ± 0.002	0.307 ± 0.002
	720	0.354 ± 0.001	0.351 ± 0.001	0.361 ± 0.002	0.356 ± 0.001	0.361 ± 0.002	0.355 ± 0.001	0.365 ± 0.001	0.358 ± 0.000
$ETTh1$	96	0.403 ± 0.005	0.420 ± 0.003	0.402 ± 0.004	0.422 ± 0.002	0.408 ± 0.003	0.428 ± 0.002	0.414 ± 0.006	0.431 ± 0.004
	192	0.440 ± 0.009	0.441 ± 0.004	0.459 ± 0.006	0.455 ± 0.005	0.469 ± 0.005	0.460 ± 0.003	0.463 ± 0.010	0.456 ± 0.006
	336	0.471 ± 0.006	0.457 ± 0.004	0.471 ± 0.005	0.457 ± 0.004	0.492 ± 0.004	0.474 ± 0.004	0.487 ± 0.007	0.466 ± 0.005
	720	0.503 ± 0.005	0.487 ± 0.004	0.535 ± 0.003	0.507 ± 0.003	0.485 ± 0.006	0.478 ± 0.007	0.517 ± 0.004	0.494 ± 0.004
$ETTm1$	96	0.329 ± 0.014	0.371 ± 0.006	0.341 ± 0.010	0.377 ± 0.008	0.350 ± 0.011	0.387 ± 0.005	0.340 ± 0.011	0.377 ± 0.007
	192	0.380 ± 0.009	0.398 ± 0.004	0.404 ± 0.010	0.413 ± 0.005	0.383 ± 0.010	0.397 ± 0.005	0.406 ± 0.012	0.408 ± 0.004
	336	0.418 ± 0.004	0.425 ± 0.004	0.432 ± 0.005	0.428 ± 0.002	0.410 ± 0.004	0.411 ± 0.003	0.424 ± 0.006	0.425 ± 0.003
	720	0.476 ± 0.008	0.440 ± 0.005	0.468 ± 0.009	0.449 ± 0.004	0.467 ± 0.007	0.448 ± 0.003	0.485 ± 0.010	0.461 ± 0.006
ILI	24	1.921 ± 0.201	0.898 ± 0.033	2.170 ± 0.174	0.947 ± 0.039	1.934 ± 0.170	0.925 ± 0.034	2.072 ± 0.211	0.948 ± 0.026
	36	2.151 ± 0.061	0.933 ± 0.035	2.093 ± 0.119	0.889 ± 0.041	2.505 ± 0.179	1.020 ± 0.017	2.494 ± 0.125	1.019 ± 0.007
	48	2.062 ± 0.090	0.892 ± 0.019	2.418 ± 0.058	0.959 ± 0.014	2.325 ± 0.201	0.948 ± 0.062	2.298 ± 0.066	0.964 ± 0.011
	60	1.759 ± 0.214	0.853 ± 0.061	2.203 ± 0.181	0.971 ± 0.048	1.926 ± 0.152	0.896 ± 0.039	2.198 ± 0.070	0.963 ± 0.017

A.11 Showcases

To provide a clear comparison among different models, we showcase the forecasting task results on ETTh1 (96-96) and ETTm1 (96-336) using three models: *LLM-TS*, TimesNet, and GPT4TS. As shown in Figures 5 and 6, our LLM-TS model produces significantly more accurate predictions, demonstrating its effectiveness.

To illustrate the performance improvements achieved by the LLM-TS Integrator framework, we introduce a case study. We created a training set with a weighted sine function:

$$\sum_{i=1}^{4} \omega_i \sin(f_i t + p_i) + \epsilon N(0,1) \tag{10}$$

where $w_1 = 0.1$, $w_2 = 0.2$, $w_3 = 0.3$, $w_4 = 0.4$; $f_1 = \frac{1}{40}$, $f_2 = \frac{1}{45}$, $f_3 = \frac{1}{50}$, $f_4 = \frac{1}{55}$; $p_1 = 0$, $p_2 = 1$, $p_3 = 2$, $p_4 = 3$; and $\epsilon = 0.1$ is the noise level. We generated a long sequence of length

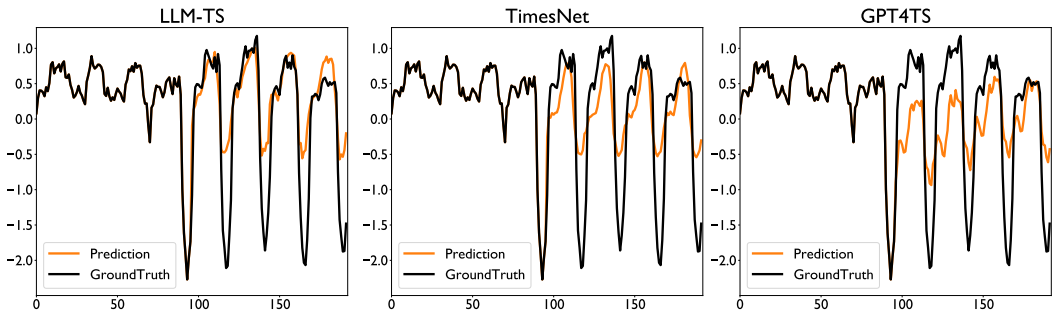

Figure 5: ETTh1

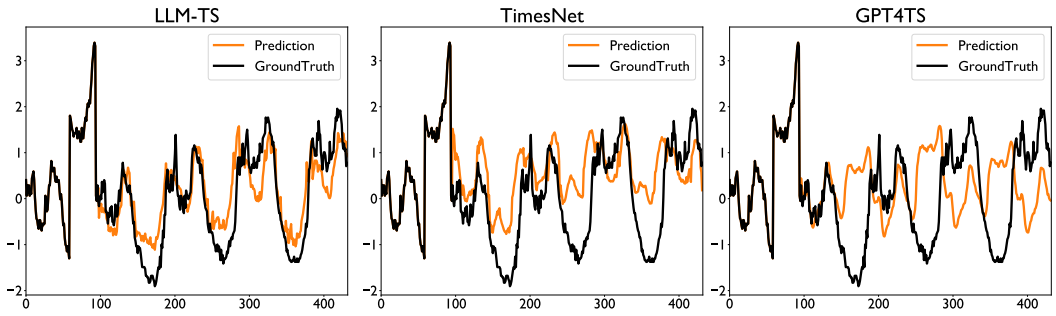

Figure 6: ETTm1

$10,000$ and then sampled a batch of size 64 with a sequence length of 96 and a prediction length of 336 to train GPT4TS, TimesNet, and LLM-TS on this data for $1,000$ iterations. For testing, we created a test set with frequency $f = \frac{1}{20}$, which is greater than $\max(f_1, f_2, f_3, f_4)$, and used $p = 2.5$, $w = 1$ and $\epsilon = 0.1$.

As shown in Figure 7, Figure 8 an Figure 9, we can know:

- GPT4TS fails to accurately capture periodic information as it relies solely on a language model without incorporating traditional mathematical modelling.

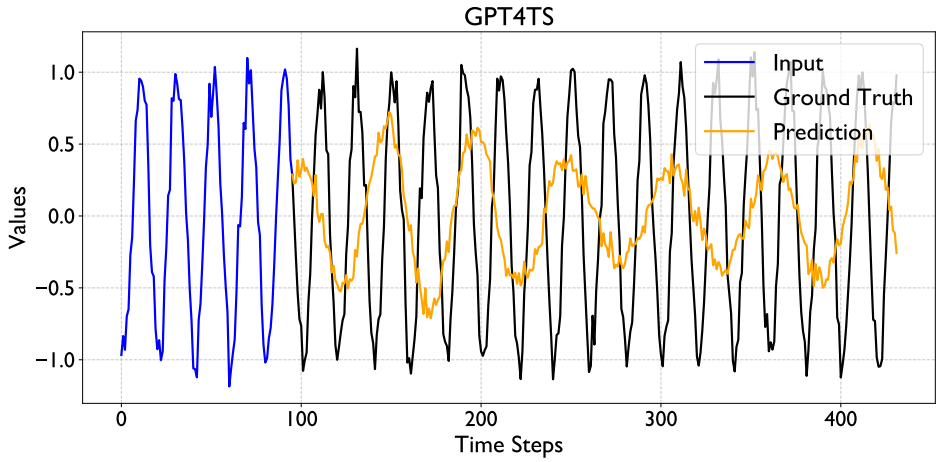

Figure 7: GPT4TS on synthetic data

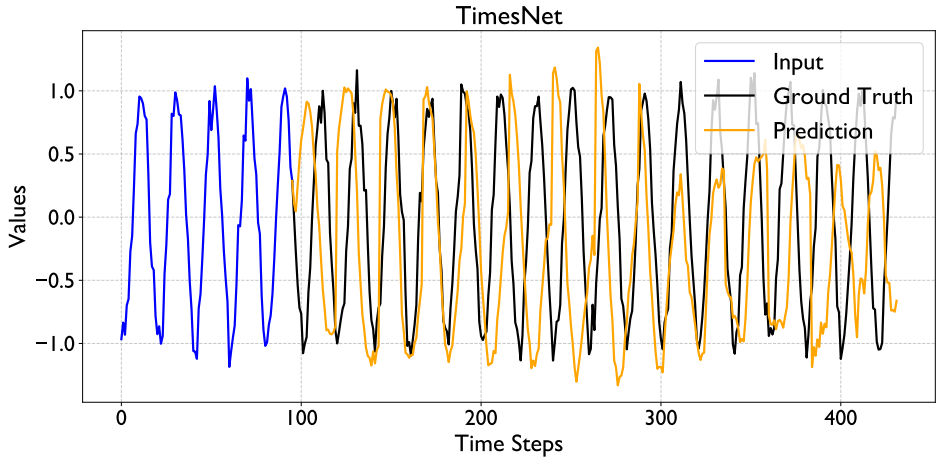

Figure 8: TimesNet on synthetic data

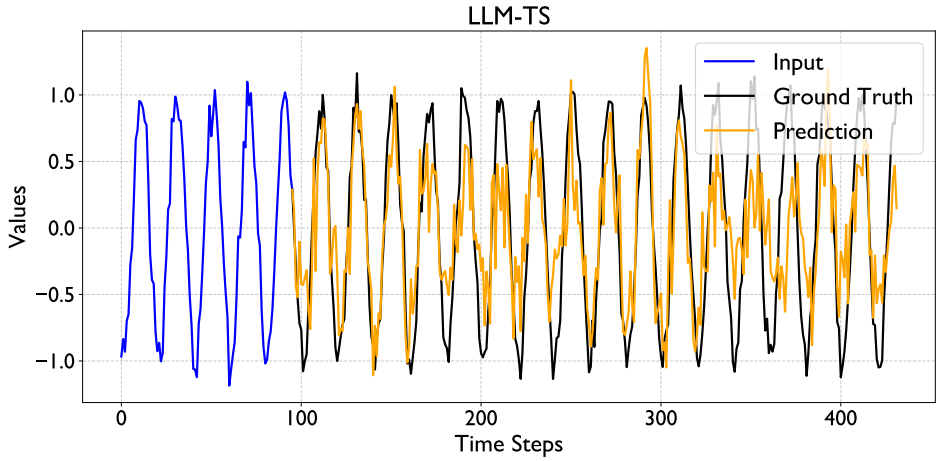

Figure 9: LLM-TS on synthetic data

- TimesNet generally captures periodic information due to the use of the FFT mathematical operator, but it still does not perfectly match the ground truth.
- LLM-TS captures periodic information and better matches the ground truth by integrating rich language model insights into the traditional TimesNet model.

This case study highlights how the LLM-TS Integrator framework benefits from both inherent properties of traditional TS models and pattern recognition abilities of LLMs, demonstrating the effectiveness of our approach.

A.12 Template Variation

We conducted additional experiments on the ETTh1 dataset for long-term forecasting with GPT2. The original template achieves a Mean Squared Error (MSE) of 0.464 and a Mean Absolute Error (MAE) of 0.458. We tested variations of the template by changing the original context from "The Electricity Transformer Temperature is a crucial indicator in electric power long-term deployment." to:

- Variation 1: "The temperature of the electricity transformer is a vital metric for long-term electric power deployment."

- Variation 2: "Monitoring the temperature of electricity transformers is essential for the long-term deployment of electric power."
- Variation 3: "The temperature of electricity transformers serves as a key indicator in the long-term deployment of electric power."
- Variation 4: No template.

Besides, we also consider the following changes:

- w/o Input Statistics: excluding input statistical data from our analysis.
- w/o Mean, Max, Median: remove mean, max and median informatino.
- w/o Lags: remove lags information.

The performance of these variations is summarized in Table 9. These results indicate that the performance is quite similar across different variations, supporting the robustness of our approach regardless of minor template modifications. For further details on the template implementation, refer to our code repository at `https://anonymous.4open.science/r/llm_ts_anonymous-F07D/utils/tools.py`.

Table 9: Performance across different template variations

Template Variation	MSE	MAE
Original Template	0.464 ± 0.004	0.458 ± 0.005
Variation 1	0.460 ± 0.005	0.456 ± 0.003
Variation 2	0.465 ± 0.006	0.460 ± 0.005
Variation 3	0.464 ± 0.004	0.459 ± 0.003
Variation 4 (No template)	0.466 ± 0.005	0.460 ± 0.005
w/o Input Statistics	0.468 ± 0.004	0.462 ± 0.004
w/o Mean/Max/Median	0.465 ± 0.004	0.459 ± 0.003
w/o Lags	0.467 ± 0.003	0.460 ± 0.005

A.13 Model Efficiency Analysis

Compared to TimesNet, our *LLM-TS integrator* introduces additional costs due to the mutual information and sample weighting modules. However, after training, the inference cost of our method is the same as TimesNet. We detail the time cost of each component for ETTh1 and ETTm1 tasks, using a batch size of 32 on a 32G V100 GPU. As shown in Table 10, the training cost of our method is reasonable, given that it achieves the best performance across most tasks.

It is important to note that we use the pre-trained LLM to obtain the text embeddings only once. These embeddings can then be used throughout the training process. For instance, obtaining the embeddings for the ETTh1 dataset using the llama-3b model on an A100 GPU takes approximately 1 hour. After this, the embeddings are utilized in our framework to train the model, and in the final output of the TimesNet model. This ensures that the inference time of our method is identical to that of the TimesNet model.

As detailed in the TimesNet paper, our backbone model TimesNet is relatively small with 0.067 MB parameters. For comparison, other models have the following sizes: Non-stationary Transformer has 1.884 MB, Autoformer has 1.848 MB, FEDformer has 2.9 MB, LightTS has 0.163 MB, DLinear has 0.296 MB, ETSformer has 1.123 MB, Informer has 1.903 MB, Reformer has 1.157 MB, and Pyraformer has 1.308 MB. The introduced mutual information network consists of only two linear layers of size $64x64$ and $64x4096$, which is negligible in terms of additional parameters. Similarly, the introduced MLP network consists of four layers: 1×100, 100×1, 1×1, and 1×1, and the number of parameters is also negligible.

Thus, our model remains very small and efficient, with inference time identical to TimesNet (as the mutual information component is only used during training). Given that many TS models are primarily used for inference, our approach offers effective performance gains with minimal additional computational cost.

Table 10: Cost Comparison per step(s).

Methods	Overall	TimesNet	Mutual Information	Sample Reweighting
ETTh1	3.177	0.126	0.577	2.474
Weather	5.563	0.436	1.094	4.033

A.14 Full Results of Short-term Forecasting

Table 11 displays the comprehensive results for short-term forecasting.

Table 11: Full results of short-term forecasting.

Methods		LLM-TS	TimesNet	GPT4TS	TIME-LLM	PatchTST	N-HiTS	N-BEATS	FEDformer	Stationary	Autoformer
Yearly	$SMAPE$	13.369	13.512	13.531	13.419	13.477	13.418	13.436	13.728	13.717	13.974
	$MASE$	3.021	3.065	3.015	3.0050	3.019	3.045	3.043	3.048	3.078	3.134
	OWA	0.789	0.799	0.793	0.789	0.792	0.793	0.794	0.803	0.807	0.822
Quarterly	$SMAPE$	10.020	10.069	10.177	10.110	10.38	10.202	10.124	10.792	10.958	11.338
	$MASE$	1.162	1.178	1.194	1.178	1.233	1.194	1.169	1.283	1.325	1.365
	OWA	0.878	0.887	0.898	0.889	0.921	0.899	0.886	0.958	0.981	1.012
Monthly	$SMAPE$	12.696	12.783	12.894	12.980	12.959	12.791	12.677	14.260	13.917	13.958
	$MASE$	0.936	0.949	0.956	0.963	0.970	0.969	0.937	1.102	1.097	1.103
	OWA	0.880	0.889	0.897	0.903	0.905	0.899	0.880	1.012	0.998	1.002
Others	$SMAPE$	4.916	4.954	4.940	4.795	4.952	5.061	4.925	4.954	6.302	5.485
	$MASE$	3.310	3.364	3.228	3.178	3.347	3.216	3.391	3.264	4.064	3.865
	OWA	1.039	1.052	1.029	1.006	1.049	1.040	1.053	1.036	1.304	1.187
Average	$SMAPE$	11.819	11.908	11.991	11.983	12.059	11.927	11.851	12.840	12.780	12.909
	$MASE$	1.588	1.612	1.600	1.595	1.623	1.613	1.599	1.701	1.756	1.771
	OWA	0.851	0.860	0.861	0.859	0.869	0.861	0.855	0.918	0.930	0.939

A.15 Full Results of Long-Term Forecasting

Full results for long-term forecasting are presented in Table 12.

A.16 Full Results of Imputation.

Table 13 contains the detailed results of our imputation tasks.

A.17 Full Results of Classification

Table 14 contains the comprehensive results for classification.

A.18 Full Results of Anamoly Detection

Full results for anamoly detection are detailed in Table 15.

A.19 Further Ablation Studies

Mutual Information Estimator. In the main paper, we utilize the Jensen-Shannon mutual information (MI) estimator. Additionally, we explore the Mutual Information Neural Estimator (MINE) [27]. We evaluate both estimators on two tasks, ETTh1 and ETTm1, with results averaged over four prediction lengths. For ETTh1, the MSE and MAE using the original Jensen-Shannon estimator are 0.454 and 0.451, respectively, compared to 0.460 and 0.457 with MINE. For ETTm1, the MSE and MAE are 0.401 and 0.409 with the original estimator, and 0.402 and 0.410 with MINE. These comparisons highlight the robustness of our method across different mutual information estimators.

Sample Reweighting Illustration. Figures 11, 10, 12, and 13 display the learned weighting network applied to various datasets: MSL for anomaly detection, Weather for forecasting, ETTh1 for imputation, and PEMS-SF for classification. These visualizations corroborate our hypothesis:

Table 12: Full results for long-term forecasting. We use prediction length $O \in \{96, 192, 336, 720\}$ except for ILI and $O \in \{24, 36, 48, 60\}$ for ILI. A lower MSE indicates better performance.

Methods		LLM-TS		TimesNet		TIME-LLM		DLinear		PatchTST		GPT4TS		FEDformer		TEST		Stationary		ETSformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.166	0.217	0.174	0.224	0.202	0.239	0.196	0.255	0.186	0.227	0.196	0.234	0.217	0.296	0.214	0.264	0.173	0.223	0.197	0.281
	192	0.229	0.269	0.235	0.272	0.245	0.277	0.237	0.296	0.234	0.265	0.241	0.271	0.276	0.336	0.262	0.298	0.245	0.285	0.237	0.312
	336	0.278	0.302	0.235	0.272	0.300	0.313	0.283	0.335	0.284	0.301	0.296	0.308	0.339	0.380	0.310	0.329	0.321	0.338	0.298	0.353
	720	0.354	0.351	0.365	0.358	0.369	0.356	0.345	0.381	0.356	0.349	0.367	0.354	0.403	0.428	0.378	0.370	0.414	0.410	0.352	0.288
	Avg	0.257	0.285	0.265	0.290	0.279	0.296	0.265	0.317	0.265	0.285	0.275	0.292	0.309	0.360	0.291	0.315	0.288	0.314	0.271	0.334
ETTh1	96	0.403	0.420	0.414	0.431	0.414	0.422	0.386	0.400	0.460	0.447	0.409	0.415	0.376	0.419	0.411	0.426	0.513	0.491	0.494	0.479
	192	0.440	0.441	0.463	0.456	0.466	0.450	0.437	0.432	0.512	0.477	0.468	0.446	0.420	0.448	0.475	0.461	0.534	0.504	0.538	0.504
	336	0.471	0.457	0.487	0.466	0.515	0.475	0.481	0.459	0.546	0.496	0.503	0.461	0.459	0.465	0.508	0.482	0.588	0.535	0.574	0.521
	720	0.503	0.487	0.517	0.494	0.503	0.487	0.519	0.516	0.544	0.517	0.510	0.482	0.506	0.507	0.504	0.494	0.643	0.616	0.562	0.535
	Avg	0.454	0.451	0.470	0.462	0.474	0.459	0.456	0.452	0.516	0.484	0.473	0.451	0.440	0.460	0.475	0.466	0.570	0.537	0.542	0.510
ETTh2	96	0.322	0.366	0.340	0.374	0.306	0.353	0.333	0.387	0.308	0.355	0.298	0.350	0.358	0.397	0.328	0.374	0.476	0.458	0.340	0.391
	192	0.400	0.409	0.399	0.410	0.386	0.399	0.477	0.476	0.393	0.405	0.376	0.399	0.429	0.439	0.403	0.418	0.512	0.493	0.430	0.439
	336	0.432	0.435	0.452	0.452	0.460	0.458	0.594	0.541	0.427	0.436	0.430	0.439	0.496	0.487	0.455	0.458	0.552	0.551	0.485	0.479
	720	0.430	0.442	0.462	0.468	0.442	0.451	0.831	0.657	0.436	0.450	0.428	0.451	0.463	0.474	0.470	0.477	0.562	0.560	0.500	0.497
	Avg	0.396	0.413	0.413	0.426	0.398	0.415	0.559	0.515	0.391	0.411	0.383	0.410	0.437	0.449	0.414	0.432	0.526	0.516	0.439	0.452
ETTm1	96	0.329	0.371	0.340	0.377	0.393	0.398	0.345	0.372	0.352	0.374	0.350	0.369	0.379	0.419	0.336	0.373	0.386	0.398	0.375	0.398
	192	0.380	0.398	0.406	0.408	0.412	0.405	0.380	0.389	0.390	0.393	0.387	0.387	0.426	0.441	0.381	0.399	0.459	0.444	0.408	0.410
	336	0.418	0.425	0.424	0.425	0.442	0.425	0.413	0.413	0.421	0.414	0.418	0.407	0.445	0.459	0.411	0.418	0.495	0.464	0.435	0.428
	720	0.476	0.440	0.485	0.461	0.502	0.457	0.474	0.453	0.462	0.449	0.477	0.437	0.543	0.490	0.478	0.454	0.585	0.516	0.499	0.462
	Avg	0.401	0.409	0.414	0.418	0.437	0.421	0.403	0.407	0.406	0.407	0.408	0.400	0.448	0.452	0.402	0.411	0.481	0.456	0.429	0.425
ETTm2	96	0.189	0.266	0.185	0.264	0.193	0.281	0.193	0.292	0.183	0.270	0.185	0.271	0.203	0.287	0.230	0.307	0.192	0.274	0.189	0.280
	192	0.253	0.307	0.252	0.306	0.254	0.315	0.284	0.363	0.255	0.314	0.250	0.312	0.269	0.328	0.284	0.338	0.280	0.339	0.253	0.319
	336	0.315	0.345	0.323	0.350	0.320	0.355	0.369	0.427	0.309	0.347	0.314	0.351	0.325	0.366	0.340	0.370	0.334	0.361	0.314	0.357
	720	0.421	0.408	0.415	0.403	0.426	0.416	0.554	0.522	0.412	0.404	0.410	0.408	0.421	0.415	0.436	0.420	0.417	0.413	0.414	0.413
	Avg	0.295	0.331	0.294	0.331	0.298	0.342	0.350	0.401	0.290	0.334	0.290	0.335	0.305	0.349	0.323	0.359	0.306	0.347	0.293	0.342
ILI	24	1.921	0.898	2.072	0.948	2.589	1.054	2.398	1.040	2.229	0.894	5.259	1.689	3.228	1.260	3.371	1.231	2.294	0.945	2.527	1.020
	36	2.151	0.933	2.494	1.019	2.996	1.194	2.646	1.088	2.330	0.925	6.136	1.831	2.679	1.080	3.725	1.322	1.825	0.848	2.615	1.007
	48	2.062	0.892	2.298	0.964	2.714	1.095	2.614	1.086	2.140	0.894	4.670	1.562	2.622	1.078	3.291	1.237	2.010	0.900	2.359	0.972
	60	1.759	0.853	2.198	0.963	2.605	1.050	2.804	1.146	2.037	0.912	4.402	1.517	2.857	1.157	2.907	1.136	2.178	0.963	2.487	1.016
	Avg	1.973	0.894	2.266	0.974	2.726	1.098	2.616	1.090	2.184	0.906	5.117	1.650	2.847	1.144	3.324	1.232	2.077	0.914	2.497	1.004
ECL	96	0.167	0.271	0.169	0.273	0.207	0.292	0.197	0.282	0.190	0.296	0.186	0.273	0.193	0.308	0.218	0.309	0.169	0.273	0.187	0.304
	192	0.178	0.280	0.186	0.288	0.209	0.297	0.196	0.285	0.199	0.304	0.190	0.278	0.201	0.315	0.220	0.311	0.182	0.286	0.199	0.315
	336	0.198	0.302	0.206	0.305	0.224	0.312	0.209	0.301	0.217	0.319	0.204	0.291	0.214	0.329	0.234	0.323	0.200	0.304	0.212	0.329
	720	0.233	0.344	0.231	0.327	0.277	0.359	0.245	0.333	0.258	0.352	0.245	0.297	0.325	0.355	0.276	0.354	0.222	0.321	0.233	0.345
	Avg	0.194	0.299	0.198	0.298	0.229	0.315	0.212	0.300	0.216	0.318	0.206	0.285	0.214	0.327	0.237	0.324	0.193	0.296	0.208	0.323
Traffic	96	0.587	0.315	0.589	0.313	0.609	0.402	0.650	0.396	0.526	0.347	0.563	0.378	0.587	0.366	0.589	0.390	0.612	0.338	0.607	0.392
	192	0.612	0.326	0.627	0.337	0.586	0.382	0.598	0.370	0.522	0.332	0.549	0.367	0.604	0.373	0.567	0.380	0.613	0.340	0.621	0.399
	336	0.634	0.338	0.635	0.341	0.593	0.390	0.605	0.373	0.517	0.334	0.566	0.376	0.621	0.383	0.583	0.389	0.618	0.328	0.622	0.396
	720	0.640	0.351	0.658	0.349	0.636	0.405	0.645	0.394	0.552	0.352	0.567	0.372	0.626	0.382	0.585	0.391	0.653	0.355	0.632	0.396
	Avg	0.618	0.333	0.627	0.335	0.606	0.395	0.625	0.383	0.529	0.341	0.561	0.373	0.610	0.376	0.581	0.388	0.624	0.340	0.621	0.396
Average		0.574	0.427	0.618	0.442	0.681	0.468	0.686	0.483	0.600	0.436	0.964	0.525	0.701	0.489	0.756	0.491	0.633	0.465	0.662	0.473

the sample weight ω_O increases with the prediction loss l_O, while the weight ω_I decreases as l_O increases. This observed pattern supports the efficacy of our reweighting strategy.

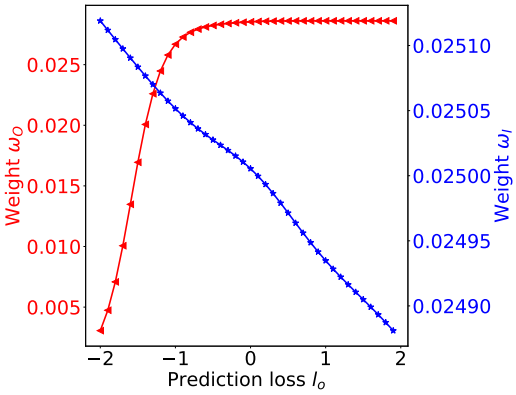

Figure 10: Forecasting.

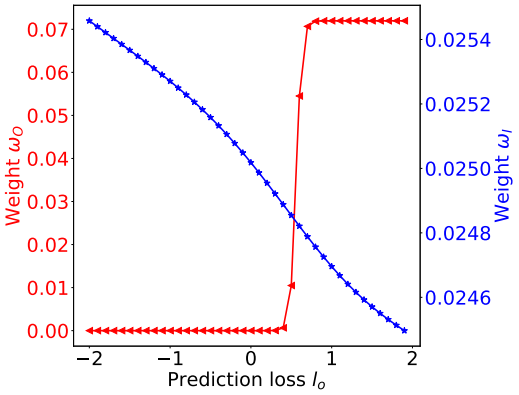

Figure 11: Anomaly detection

Static Weighting Scheme. We also explore a static weighting scheme as a contrast to the dynamic weighting used in our sample reweighting module. This scheme balances the prediction loss and mutual information loss, with a ratio of 0.0 representing pure prediction loss and 1.0 representing

Table 13: Full results for the imputation task. Randomly masked {12.5%, 25%, 37.5%, 50%} of points in 96-length series, averaging results over 4 mask ratios.

Methods		LLM-TS		TimesNet		GPT4TS		PatchTST		LightTS		DLinear		FEDformer		Stationary		Autoformer		Reformer	
Mask	Ratio	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTm1$	12.5%	0.018	0.088	0.023	0.101	0.018	0.089	0.041	0.130	0.093	0.206	0.080	0.193	0.052	0.166	0.032	0.119	0.046	0.144	0.042	0.146
	25%	0.022	0.097	0.023	0.101	0.023	0.099	0.044	0.135	0.093	0.206	0.080	0.193	0.052	0.166	0.032	0.119	0.046	0.144	0.042	0.146
	37.5%	0.027	0.108	0.029	0.112	0.030	0.112	0.049	0.143	0.113	0.231	0.103	0.219	0.069	0.191	0.039	0.131	0.057	0.161	0.063	0.182
	50%	0.033	0.120	0.035	0.123	0.042	0.131	0.055	0.151	0.134	0.255	0.132	0.248	0.089	0.218	0.047	0.145	0.067	0.174	0.082	0.208
	Avg	0.025	0.103	0.028	0.109	0.028	0.108	0.047	0.140	0.104	0.218	0.093	0.206	0.062	0.177	0.036	0.126	0.051	0.150	0.055	0.166
$ETTm2$	12.5%	0.018	0.079	0.019	0.081	0.019	0.078	0.108	0.239	0.034	0.127	0.062	0.166	0.056	0.159	0.021	0.088	0.023	0.092	0.108	0.228
	25%	0.020	0.085	0.021	0.087	0.021	0.084	0.028	0.099	0.042	0.143	0.085	0.196	0.080	0.195	0.024	0.096	0.026	0.101	0.136	0.262
	37.5%	0.022	0.089	0.023	0.092	0.024	0.090	0.030	0.104	0.051	0.159	0.106	0.222	0.110	0.231	0.027	0.103	0.030	0.108	0.175	0.300
	50%	0.025	0.096	0.025	0.097	0.027	0.098	0.034	0.110	0.059	0.174	0.131	0.247	0.156	0.276	0.030	0.108	0.035	0.119	0.211	0.329
	Avg	0.021	0.087	0.022	0.089	0.023	0.088	0.029	0.102	0.046	0.151	0.096	0.208	0.101	0.215	0.026	0.099	0.029	0.105	0.157	0.280
$ETTh1$	12.5%	0.058	0.165	0.064	0.170	0.043	0.141	0.093	0.201	0.240	0.345	0.151	0.267	0.070	0.190	0.060	0.165	0.074	0.182	0.074	0.194
	25%	0.077	0.189	0.082	0.192	0.056	0.159	0.107	0.217	0.265	0.364	0.180	0.292	0.106	0.236	0.080	0.189	0.090	0.203	0.102	0.227
	37.5%	0.096	0.209	0.098	0.209	0.074	0.182	0.120	0.230	0.296	0.382	0.215	0.318	0.124	0.258	0.102	0.212	0.109	0.222	0.135	0.261
	50%	0.118	0.228	0.116	0.226	0.104	0.214	0.141	0.248	0.334	0.404	0.257	0.347	0.165	0.299	0.133	0.240	0.137	0.248	0.179	0.298
	Avg	0.087	0.198	0.090	0.199	0.069	0.174	0.115	0.224	0.284	0.373	0.201	0.306	0.117	0.246	0.094	0.201	0.103	0.214	0.122	0.245
$ETTh2$	12.5%	0.039	0.131	0.040	0.132	0.041	0.129	0.057	0.152	0.101	0.231	0.100	0.216	0.095	0.212	0.042	0.133	0.044	0.138	0.163	0.289
	25%	0.046	0.143	0.048	0.146	0.046	0.137	0.061	0.158	0.115	0.246	0.127	0.247	0.137	0.258	0.049	0.147	0.050	0.149	0.206	0.331
	37.5%	0.053	0.154	0.055	0.156	0.053	0.148	0.067	0.166	0.126	0.257	0.158	0.276	0.187	0.304	0.056	0.158	0.060	0.163	0.252	0.370
	50%	0.061	0.165	0.061	0.165	0.060	0.160	0.073	0.174	0.136	0.268	0.183	0.299	0.232	0.341	0.065	0.170	0.068	0.173	0.316	0.419
	Avg	0.050	0.148	0.051	0.150	0.050	0.144	0.065	0.163	0.119	0.250	0.142	0.259	0.163	0.279	0.053	0.152	0.055	0.156	0.234	0.352
ECL	12.5%	0.087	0.203	0.090	0.204	0.080	0.194	0.055	0.160	0.102	0.229	0.092	0.214	0.107	0.237	0.093	0.210	0.089	0.210	0.190	0.308
	25%	0.091	0.207	0.092	0.209	0.087	0.203	0.065	0.175	0.121	0.252	0.118	0.247	0.120	0.251	0.097	0.214	0.096	0.220	0.197	0.312
	37.5%	0.095	0.213	0.096	0.213	0.094	0.211	0.076	0.344	0.141	0.273	0.144	0.276	0.136	0.266	0.102	0.220	0.104	0.229	0.203	0.315
	50%	0.101	0.220	0.102	0.221	0.101	0.220	0.091	0.208	0.160	0.293	0.175	0.305	0.158	0.284	0.108	0.228	0.113	0.239	0.210	0.319
	Avg	0.094	0.211	0.095	0.212	0.091	0.207	0.072	0.183	0.131	0.262	0.132	0.260	0.130	0.259	0.100	0.218	0.101	0.225	0.200	0.313
$Weather$	12.5%	0.026	0.048	0.025	0.047	0.027	0.049	0.029	0.049	0.047	0.101	0.039	0.084	0.041	0.107	0.027	0.051	0.026	0.047	0.031	0.076
	25%	0.029	0.055	0.031	0.062	0.030	0.054	0.031	0.053	0.052	0.111	0.048	0.103	0.064	0.163	0.029	0.056	0.030	0.054	0.035	0.082
	37.5%	0.032	0.059	0.034	0.064	0.034	0.062	0.035	0.058	0.058	0.121	0.057	0.117	0.107	0.229	0.033	0.062	0.032	0.060	0.040	0.091
	50%	0.033	0.061	0.035	0.062	0.037	0.066	0.038	0.063	0.065	0.133	0.066	0.134	0.183	0.312	0.037	0.068	0.037	0.067	0.046	0.099
	Avg	0.030	0.056	0.031	0.059	0.032	0.058	0.060	0.144	0.055	0.117	0.052	0.110	0.099	0.203	0.032	0.059	0.031	0.057	0.038	0.087

Table 14: Complete classification task results. $*$. in the Transformers indicates the name of $*$former.

Methods	Classical		RNN		TCN	Transformers									MLP		TimesNet	LLM		LLM-TS
	XGB	Roc	LSTNet	LSSL		Trans.	Re.	In.	Pyra.	Auto.	Station.	FED.	ETS.	Flow.	DL	LTS.		GPT4TS	TEST	
Ethanol	43.7	45.2	39.9	31.1	28.9	32.7	31.9	31.6	30.8	31.6	32.7	31.2	28.1	33.8	32.6	29.7	30.4	26.2	25.1	31.9
FaceD	63.3	64.7	65.7	66.7	52.8	67.3	68.6	67.0	65.7	68.4	68.0	66.0	66.3	67.6	68.0	67.5	68.6	67.8	50.1	68.9
HandW	15.8	58.8	25.8	24.6	53.3	32.0	27.4	32.8	29.4	36.7	31.6	28.0	32.5	33.8	27.0	26.1	32.1	28.9	20.1	32.7
HeartB	73.2	75.6	77.1	72.7	75.6	76.1	77.1	80.5	75.6	74.6	73.7	73.7	71.2	77.6	75.1	75.1	77.6	72.2	73.7	77.1
JapanV	86.5	96.2	98.1	98.4	98.9	98.7	97.8	98.9	98.4	96.2	99.2	98.4	95.9	98.9	96.2	96.2	97.2	98.4	78.4	98.1
PEMS	98.3	75.1	86.7	86.1	68.8	82.1	82.7	81.5	83.2	82.7	87.3	80.9	86.0	83.8	75.1	88.4	89.6	79.2	59.5	90.8
SCP1	84.6	90.8	84.0	90.8	84.6	92.2	90.4	90.1	88.1	84.0	89.4	88.7	89.6	92.5	87.3	89.8	90.4	90.1	84.0	91.8
SCP2	48.9	53.3	52.8	52.2	55.6	53.9	56.7	53.3	53.3	50.6	57.2	54.4	55.0	56.1	50.5	51.1	57.1	50.0	54.4	57.8
SpokenA	69.6	71.2	100.0	100.0	95.6	98.4	97.0	100.0	99.6	100.0	100.0	100.0	100.0	98.8	81.4	100.0	98.6	97.9	82.1	98.6
UWave	75.9	94.4	87.8	85.9	88.4	85.6	85.6	85.6	83.4	85.9	87.5	85.3	85.0	86.6	82.1	80.3	85.5	85.6	84.4	86.6
Avg	66.0	72.5	71.8	70.9	70.3	71.9	71.5	72.1	70.8	71.1	72.7	70.7	71.0	<u>73.0</u>	67.5	70.4	72.7	69.5	61.2	**73.4**

pure mutual information loss. As shown in Table 18, the static approach underperforms relative to our dynamic sample weighting module, demonstrating the superior effectiveness of our method.

Comprehensive Results.

The detailed performance of various traditional TS models and LLMs is presented in Table 16 and Table 17.

A.20 Related Work

LLM for TS Modeling. FPT [80] suggests utilizing pre-trained language models to extract features from time series for improved predictions. TIME-LLM [35] and TEST [54] adapt LLMs for general time series forecasting by maintaining the original language model structure while reprogramming the input to fit time series data [76]. LLMTIME [24] interprets time series as sequences of numbers, treating forecasting as a next-token prediction task akin to text processing, applying pre-trained LLMs for this purpose. Given that it is not a state-of-the-art method and primarily targets zero-shot forecasting, it has not been incorporated into our experimental framework. TEMPO [5] utilizes essential inductive biases of the TS task for generative pre-trained transformer models.

Table 15: Full results for the anomaly detection.

Methods	SMD			MSL			SMAP			SWaT			PSM			Avg F1
Metrics	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	%
LLM-TS	88.09	81.54	84.69	89.04	74.49	81.11	89.95	56.51	69.41	91.16	95.40	93.23	98.44	96.45	97.43	85.17
TimesNet	87.93	81.45	84.57	88.62	73.48	80.34	89.59	56.35	69.18	91.00	95.33	93.12	98.40	96.18	97.27	84.90
GPT4TS	87.70	81.19	84.32	82.15	81.32	81.73	90.04	55.75	68.86	92.12	93.06	92.59	98.37	96.34	97.34	84.97
PatchTST	87.26	82.14	84.62	88.34	70.96	78.70	90.64	55.46	68.82	91.10	80.94	85.72	98.84	93.47	96.08	82.79
ETSformer	87.44	79.23	83.13	85.13	84.93	85.03	92.25	55.75	69.50	90.02	80.36	84.91	99.31	85.28	91.76	82.87
FEDformer	87.95	82.39	85.08	77.14	80.07	78.57	90.47	58.10	70.76	90.17	96.42	93.19	97.31	97.16	97.23	84.97
LightTS	87.10	78.42	82.53	82.40	75.78	78.95	92.58	55.27	69.21	91.98	94.72	93.33	98.37	95.97	97.15	84.23
DLinear	83.62	71.52	77.10	84.34	85.42	84.88	92.32	55.41	69.26	80.91	95.30	87.52	98.28	89.26	93.55	82.46
Stationary	88.33	81.21	84.62	68.55	89.14	77.50	89.37	59.02	71.09	68.03	96.75	79.88	97.82	96.76	97.29	82.08
Autoformer	88.06	82.35	85.11	77.27	80.92	79.05	90.40	58.62	71.12	89.85	95.81	92.74	99.08	88.15	93.29	84.26
Pyraformer	85.61	80.61	83.04	83.81	85.93	84.86	92.54	57.71	71.09	87.92	96.00	91.78	71.67	96.02	82.08	82.57
Anomaly Transformer	88.91	82.23	85.49	79.61	87.37	83.31	91.85	58.11	71.18	72.51	97.32	83.10	68.35	94.72	79.40	80.50
Informer	86.60	77.23	81.65	81.77	86.48	84.06	90.11	57.13	69.92	70.29	96.75	81.43	64.27	96.33	77.10	78.83
Reformer	82.58	69.24	75.32	85.51	83.31	84.40	90.91	57.44	70.40	72.50	96.53	82.80	59.93	95.38	73.61	77.31
Transformer	83.58	76.13	79.56	71.57	87.37	78.68	89.37	57.12	69.70	68.84	96.53	80.37	62.75	96.56	76.07	76.88

Table 16: Different traditional models. We use prediction length $O \in \{96, 192, 336, 720\}$ for ILI and $O \in \{24, 36, 48, 60\}$ for others.

Methods		PatchTST		PatchTST INT		ETSformer		ETS INT		Stationary		Stat INT		FreTS		FreTS INT	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.174	0.216	0.172	0.214	0.196	0.282	0.200	0.285	0.178	0.226	0.201	0.246	0.187	0.243	0.179	0.235
	192	0.222	0.258	0.219	0.255	0.282	0.364	0.278	0.361	0.235	0.278	0.238	0.280	0.227	0.274	0.221	0.278
	336	0.280	0.298	0.279	0.298	0.344	0.409	0.322	0.382	0.327	0.339	0.312	0.329	0.281	0.325	0.276	0.320
	720	0.356	0.349	0.356	0.348	0.430	0.472	0.427	0.470	0.387	0.383	0.386	0.383	0.352	0.382	0.344	0.376
	Avg	0.258	0.280	0.257	0.279	0.313	0.382	0.307	0.375	0.282	0.307	0.284	0.309	0.262	0.306	0.255	0.302
ETTh1	96	0.381	0.398	0.382	0.401	0.554	0.536	0.550	0.532	0.534	0.499	0.523	0.486	0.398	0.412	0.395	0.409
	192	0.421	0.426	0.422	0.428	0.686	0.619	0.690	0.621	0.639	0.560	0.609	0.560	0.454	0.449	0.455	0.451
	336	0.464	0.449	0.460	0.441	0.869	0.730	0.868	0.728	0.790	0.648	0.780	0.634	0.512	0.483	0.502	0.474
	720	0.527	0.500	0.510	0.496	1.085	0.849	1.054	0.830	0.706	0.620	0.701	0.606	0.572	0.547	0.560	0.530
	Avg	0.448	0.443	0.444	0.442	0.799	0.684	0.791	0.678	0.667	0.582	0.653	0.572	0.484	0.473	0.478	0.466
ETTm1	96	0.332	0.368	0.332	0.372	0.526	0.495	0.424	0.434	0.417	0.417	0.412	0.410	0.340	0.375	0.339	0.374
	192	0.368	0.388	0.367	0.388	0.565	0.538	0.458	0.461	0.446	0.437	0.445	0.435	0.395	0.408	0.384	0.399
	336	0.397	0.405	0.396	0.405	0.658	0.603	0.537	0.519	0.582	0.507	0.570	0.491	0.431	0.433	0.420	0.423
	720	0.457	0.445	0.460	0.446	0.801	0.696	0.802	0.696	0.661	0.546	0.660	0.546	0.494	0.470	0.484	0.462
	Avg	0.389	0.402	0.389	0.403	0.638	0.583	0.555	0.528	0.527	0.477	0.522	0.471	0.415	0.422	0.407	0.415
ILI	24	2.229	0.894	2.172	0.856	4.043	1.410	3.607	1.305	2.722	1.024	1.905	0.872	3.226	1.231	3.202	1.213
	36	2.330	0.925	2.347	0.978	3.809	1.358	3.705	1.315	3.026	1.071	2.790	1.068	3.363	1.259	3.000	1.173
	48	2.140	0.894	1.984	0.869	3.851	1.351	3.714	1.309	2.622	1.032	2.132	0.900	3.456	1.285	3.132	1.213
	60	2.037	0.912	1.770	0.831	3.983	1.349	3.935	1.350	2.520	1.035	1.991	0.901	3.749	1.340	3.298	1.243
	Avg	2.184	0.906	2.068	0.884	3.922	1.367	3.740	1.320	2.722	1.041	2.205	0.935	3.449	1.279	3.158	1.211

Time Series to Text. PromptCast [69] proposes to transform the numerical input and output into prompts, which enables forecasting in a sentence-to-sentence manner. Time-LLM [35] incorporates background, instruction and statistical information of the time series data via natural language to facilitate time series forecasting in LLM. LLMTIME [25] converts time series data into a string of numbers and predicts future values as if completing a text.

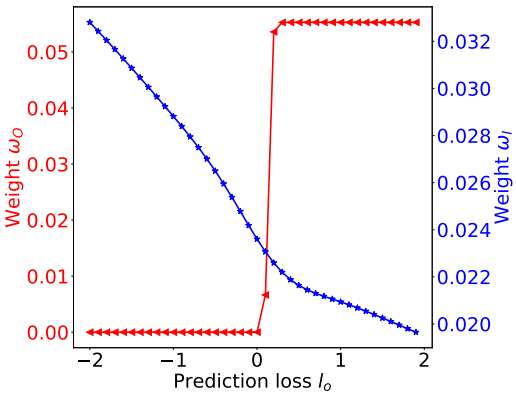

Figure 12: imputation.

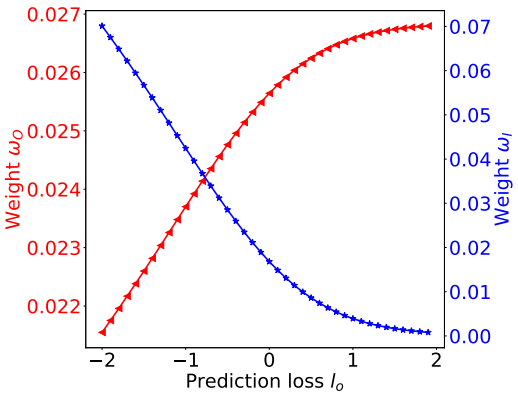

Figure 13: classification

Table 17: Different LLM embeddings. We use prediction length $O \in \{96, 192, 336, 720\}$ for ILI and $O \in \{24, 36, 48, 60\}$ for others.

Methods		LLM-TS (LLaMA)		LLaMA w/o text		GPT2		BERT		No LLM	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.166	0.217	0.170	0.218	0.168	0.218	0.167	0.217	0.168	0.218
	192	0.229	0.269	0.227	0.266	0.226	0.267	0.229	0.270	0.227	0.268
	336	0.278	0.302	0.295	0.314	0.292	0.310	0.283	0.305	0.298	0.318
	720	0.354	0.351	0.360	0.354	0.359	0.354	0.360	0.354	0.361	0.356
	Avg	0.257	0.285	0.263	0.288	0.261	0.287	0.260	0.287	0.264	0.290
ETTh1	96	0.403	0.420	0.409	0.427	0.408	0.426	0.402	0.421	0.402	0.422
	192	0.440	0.441	0.445	0.445	0.442	0.444	0.452	0.450	0.459	0.455
	336	0.471	0.457	0.490	0.472	0.487	0.467	0.494	0.472	0.471	0.457
	720	0.503	0.487	0.518	0.496	0.517	0.494	0.520	0.497	0.535	0.507
	Avg	0.454	0.451	0.465	0.460	0.464	0.458	0.467	0.460	0.467	0.460
ETTm1	96	0.329	0.371	0.350	0.387	0.338	0.370	0.340	0.375	0.341	0.377
	192	0.380	0.398	0.383	0.398	0.392	0.404	0.401	0.408	0.404	0.413
	336	0.418	0.425	0.423	0.426	0.416	0.423	0.414	0.421	0.432	0.428
	720	0.476	0.440	0.467	0.449	0.477	0.454	0.470	0.445	0.468	0.449
	Avg	0.401	0.409	0.406	0.415	0.406	0.413	0.406	0.412	0.411	0.417
ILI	24	1.921	0.898	1.998	0.929	1.997	0.929	1.917	0.915	2.170	0.947
	36	2.151	0.933	2.422	0.957	2.333	0.958	2.431	1.004	2.093	0.889
	48	2.062	0.892	2.198	0.964	2.269	0.937	2.333	0.961	2.418	0.959
	60	1.759	0.853	2.072	0.948	2.077	0.921	2.089	0.926	2.203	0.971
	Avg	1.973	0.894	2.173	0.950	2.169	0.936	2.193	0.952	2.221	0.942

Table 18: Static Weighting Scheme with Different ratios.

Ratio	0.0		0.2		0.4		0.6		0.8		1.0		Ours	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh1$	0.478	0.468	0.471	0.459	0.465	0.462	0.470	0.463	0.473	0.450	0.471	0.463	0.454	0.451
$ETTm1$	0.415	0.417	0.408	0.414	0.405	0.412	0.406	0.412	0.417	0.419	0.416	0.419	0.401	0.409

Mutual Information The Infomax principle [39, 4], applied in the context of neural networks, advocates for maximizing mutual information between the inputs and outputs of a network. Traditionally, quantifying mutual information was challenging outside a few specific probability distributions, as discussed in [51]. This complexity led to the development of various heuristics and approximations [58]. More recently, a breakthrough came with MINE [3], which introduced a neural estimator capable of assessing mutual information between two arbitrary quantities with a precision that depends on the capacity of the encoding network. This innovative approach has spearheaded advancements in the field of representation learning [28, 57]. The estimator we utilize is based on the Jensen-Shannon divergence variant of the MINE mutual information estimator.

Sample Reweighting. Sample reweighting is commonly used to improve training efficacy [19, 62, 71, 75, 72, 13]. Traditional approaches [21, 56] assign larger weights to samples with higher loss values, as these hard samples have greater learning potential. Recent studies [49] suggest using a validation set to guide the learning of sample weights, which can enhance model training. Notably, meta-weight-net [50] proposes learning a mapping from sample loss to sample weight. In this work, we adopt an MLP network that takes sample prediction loss as input and outputs dual weights for prediction loss and mutual information loss.