Learning Latent Seasonal-Trend Representations for Time Series Forecasting

Zhiyuan Wang[1], **Xovee Xu**[1], **Weifeng Zhang**[1], **Goce Trajcevski**[2], **Ting Zhong**[1], **Fan Zhou**[1*]
[1] University of Electronic Science and Technology of China
[2] Iowa State University
zhy.wangcs@gmail.com, xovee@live.com, weifzh@outlook.com,
gocet25@iastate.edu, {tingzhong, fan.zhou}@uestc.edu.cn

Abstract

Forecasting complex time series is ubiquitous and vital in a range of applications but challenging. Recent advances endeavor to achieve progress by incorporating various deep learning techniques (e.g., RNN and Transformer) into sequential models. However, clear patterns are still hard to extract since time series are often composed of several intricately entangled components. Motivated by the success of disentangled variational autoencoder in computer vision and classical time series decomposition, we plan to infer a couple of representations that depict seasonal and trend components of time series. To achieve this goal, we propose LaST, which, based on variational inference, aims to disentangle the seasonal-trend representations in the latent space. Furthermore, LaST supervises and disassociates representations from the perspectives of themselves and input reconstruction, and introduces a series of auxiliary objectives. Extensive experiments prove that LaST achieves state-of-the-art performance on time series forecasting task against the most advanced representation learning and end-to-end forecasting models. For reproducibility, our implementation is publicly available on Github[1].

1 Introduction

Time series forecasting plays a significant role in plethora of modern applications, ranging from climate analysis [1], energy production [2], traffic flows [3] to financial markets and various industrial systems [4]. The ubiquity and importance of time series data have recently attracted researcher efforts resulting in a myriad of deep learning forecasting models [5, 6] ameliorating the time series forecasting. Based on advanced techniques such as RNN and Transformer [7]), these methods usually learn latent representations to epitomize every instant of the signals, and then derive forecasting results by a predictor, achieving great progress on forecasting tasks.

However, these models have difficulties to extract exact/clear information related to temporal patterns (e.g., seasonality, trend, and level), especially in supervised end-to-end architecture without any constraint on representations [8]. As a consequence, efforts have been made to apply the variational inference into time series modeling [9, 10, 11], where improved guidance for latent representations with probabilistic form has been proved beneficial to downstream time series tasks [12]. However, when various intricately co-evolving constituents exist a time series data, analyzing with a single representation will result in superficial variable and models' non-reusablity and lack of interpretability, due to the highly entangled nature of neural networks [13, 14]. Thus, while providing efficiency and effectiveness, existing approaches with a single high-dimensional representation sacrifice the

*Corresponding author.
[1] https://github.com/zhycs/LaST

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

information utilization and explainability, which may further lead to overfitting and degenerated performance.

To address the above limitations and seek a new disentangled time series learning framework, we leverage the ideas from the decomposition strategy [15, 16] to split time series data into several components, each of which captures an underlying pattern category. The decomposition assists the analysis process and reveals underlying insights more consistently with human intuition. This insight motivates us to produce a couple of latent representations that respond to different time series characteristics (in our case the seasonal and trend), from which we predict the results by formulating sequence as the sum of these characteristics. The representations should be as independent as possible to avoid a model prone to feature entanglement, while also have sufficient information to the input sequence.

Towards that, we propose a novel framework **LaST** to learn the **La**tent **S**easonal-**T**rend representations for time series forecasting. LaST exploits an encoder-decoder architecture and follows variational inference theory [17] to learn a couple of disentangled latent representations that describes seasonal and trend of time series. To achieve this goal, LaST enforces the representations into disentanglement under two constraints: (1) from the input reconstruction, we dissect intrinsic seasonal-trend patterns which can be readily obtained from raw time series and off-the-shelf measurement methods, and accordingly design a series of auxiliary objectives; (2) from representations themselves, we minimize the mutual information (MI) between seasonal and trend representations on the premise that grantee the consistency between input data and each of them. Our main contributions are threefold:

- We start with the variational inference and information theory to design the seasonal-trend representations learning and disentanglement mechanisms, and practically demonstrate their effectiveness and superiority (over the existing baselines) on time series forecasting task.
- We propose LaST, a novel latent seasonal-trend representations learning framework, which encodes input as disentangled seasonal-trend representations and provides a practicable approach that reconstructs seasonal and trend separately to avoid chaos.
- We introduce MI terms as a penalty and present a novel tractable lower bound and an upper bound for their optimizations. The lower bound ameliorates the biased gradient issue in prevalent MINE approach and ensures informative representations. The upper bound provides the feasibility to further reduce the overlapping of seasonal-trend representations.

2 Related work

Most of the deep learning methods for time series forecasting are designed as an end-to-end architecture. Various basic techniques (e.g., residual structure [18, 19], autoregressive network [20, 21], and convolutions [22, 23]) are exploited to produce expressive non-linear hidden states and embeddings that reflect the temporal dependencies and patterns. There is also a body of works that apply Transformer [7] structure into time series forecasting tasks [24, 25, 26, 6], aiming to discover the relationships across the sequence and focus on the important time points. Deep learning methods have achieved superior performance in comparison to the classical algorithms such as ARIMA [27] and VAR [28], and have become prevalent in multiple applications.

Learning flexible representations has been demonstrated to be beneficial for downstream tasks by numerous researches [12, 29]. In time series representations domain, early methods, employing the variational inference, jointly train an encoder and corresponding decoder that reconstructs raw signals to learn approximate latent representations [10, 30, 31]. Recent efforts have improve these variational methods [32, 33] by establishing more complex and flexible distributions using techniques such as copula [32] and normalizing flow [34, 35]. Another group of works exploited the burgeoning contrastive learning to obtain invariant representations from augmented time series [36, 37, 38], which avoids the reconstruction process and improves representations without additional supervision.

Time series decomposition [15, 16] is a classical method that splits complex time series into several components to obtain the temporal patterns and interpretability. Recent works have applied machine learning and deep learning approaches [39, 40, 41] to robustly and efficiently achieve the decomposition on large-scale datasets. There are also research results which tackle forecasting with the assistance of decomposition. For example, Autoformer [26] decomposes time series into seasonal-trend parts by average pooling and introduces an autocorrelation mechanism to empower

Transformer [7] for better relations discovery. CoST [38] encodes signals into seasonal and trend representations in frequency and temporal domain, respectively, and introduces contrastive learning to supervise their learning. Different from our work, these methods exploit simple average pooling decomposition mechanism which may provide incompatible periodical assumptions, or intuitively disentangle the representations by processing in different domains. Meanwhile, LaST adaptively epitomizes the seasonality and trend by disentangled representations and boosts their disassociation from a probabilistic perspective in the latent space.

3 Latent seasonal-trend representations learning framework

We now formalize the problem definition and introduce the proposed LaST framework. We note that in LaST we use seasonal and trend characteristics for disentanglement learning but our framework can be easily extended to adapt situations that have more than two components to dissociate.

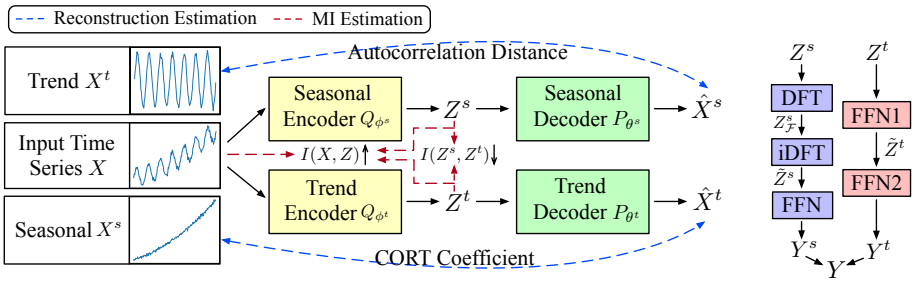
Problem definition. Consider a time series dataset $\mathcal{D}$ consisting of N i.i.d. sequences denoted as $X_{1:T}^{(i)} = \{x_1^{(i)}, x_2^{(i)}, \cdots, x_t^{(i)}, \cdots, x_T^{(i)}\}$, where $i \in \{1, 2, \ldots, N\}$, and each $x_t^{(i)}$ is univariate or multivariate value representing the current observation(s) at time instant t (e.g., price and rainfall). We aim to derive a model that outputs expressive representations $Z_{1:T}$, suitable for predicting future sequence(s) $Y = \hat{X}_{T+1:T+\tau}$. Hereafter, when there is no ambiguity we omit the superscripts and the subscript $_{1:T}$. A model that infers the likelihood between observation X and future Y with latent representation Z can be formulated as follows:

$$P(X, Y) = P(Y|X)P(X) = \int_Z P(Y|Z)P(Z|X)dZ \int_Z P(X|Z)P(Z)dZ. \tag{1}$$

From the perspective of variational inference (cf. [17]), the likelihood $P(X|Z)$ is calculated by a posterior distribution $Q_\phi(Z|X)$ and maximized by the following evidence lower bound (ELBO):

$$\begin{aligned}\log P_\Theta(X, Y) \geq \log \int_Z P_\psi(Y|Z)Q_\phi(Z|X)dZ + \mathbb{E}_{Q_\phi(Z|X)}[\log P_\theta(X|Z)] \\ - \mathbb{KL}(Q_\phi(Z|X)||P(Z)) = \mathcal{L}_{ELBO},\end{aligned} \tag{2}$$

where Θ is composed of ψ, ϕ, and θ denotes learned parameters.



(a) Seasonal-trend representations learning (b) Predictor

Figure 1: Overview of proposed LaST framework.

However, as pointed out in Sec. 1, this faces and entanglement problem and cannot clearly extract complicated temporal patterns. To ameliorate this limitation, we incorporate the decomposition strategy into our LaST that learns a couple of disentangled representations to depict seasonal and trend dynamics. Specifically, we formulate the temporal signals X and Y as the sum of seasonal and trend components, i.e., $X = X^s + X^t$. Accordingly, the latent representation Z is factorized into Z^s and Z^t, assumed to be independent of each other – i.e., $P(Z) = P(Z^s)P(Z^t)$. Figure 1 illustrates the two parts of the LaST framework: (a) representations learning, producing disentangles seasonal-trend representations (separate reconstructions and MI constraints); and (b) prediction based on learned representations.

Theorem 1. *With the decomposition strategy, Eq.* (2) *(i.e., the ELBO) naturally has the following factorized form:*

$$\mathcal{L}_{ELBO} = \log \int_{Z^s} \int_{Z^t} P_\psi(Y|Z^s, Z^t) Q_{\phi^s,\phi^t}(Z^s, Z^t|X) dZ^s dZ^t \qquad \textit{(predictor)} \quad (3)$$

$$+ \mathbb{E}_{Q_{\phi^s}(Z^s|X)}[\log P_{\theta^s}(X^s|Z^s)] + \mathbb{E}_{Q_{\phi^t}(Z^t|X)}[\log P_{\theta^t}(X^t|Z^t)] \qquad \textit{(reconstruction)} \quad (4)$$

$$- \mathbb{KL}(Q_{\phi^s}(Z^s|X)||P(Z^s)) - \mathbb{KL}(Q_{\phi^t}(Z^t|X)||P(Z^t)). \qquad \textit{(KL divergence)} \quad (5)$$

The detailed inference process of the above formula is provided in Appendix A.1. The ELBO is split into three main units, i.e., Eqs. (3), (4), and (5). The *predictor* makes forecasting and measures the accuracy (e.g,. L1 or L2 losses), *reconstruction* and *KL divergence* are served as regularization terms aiming to improve the learned representations. The three units are described in the following.

Predictor: The *predictor* (cf. Eq. (3) and Figure 1(b)) can be regarded as the sum of two independent parts: $\log \int_{Z^s} P_{\psi^s}(Y^s|Z^s) Q_{\phi^s}(Z^s|X) dZ^s$ and $\log \int_{Z^t} P_{\psi^t}(Y^t|Z^t) Q_{\phi^t}(Z^t|X) dZ^t$. Here we introduce two specialized approaches to harness the seasonal-trend representations combining their own characteristics. Given the seasonal latent representation $Z^s \in \mathbb{R}^{T \times d}$, the seasonal predictor first employs the discrete Fourier transform (DFT) algorithm to detect the seasonal frequencies, i.e., $Z^s_{\mathcal{F}} = \text{DFT}(Z^s) \in \mathbb{C}^{F \times d}$, where $F = \lfloor \frac{T+1}{2} \rfloor$ due to the Nyquist theorem [42]. Then, we inverse the frequencies back to the temporal domain to extend the representation to the future part, i.e., $\tilde{Z}^s = \text{iDFT}(Z^s_{\mathcal{F}}) \in \mathbb{R}^{\tau \times d}$. More details of the DFT and iDFT functions can be found in Appendix B.1. Given Z^t, trend predictor provides a feed forward network (FFN) $f : T \to \tau$ to produce a predictable representation $\tilde{Z}^t \in \mathbb{R}^{\tau \times d}$. We end the predictor with two FFNs to map $\tilde{Z}^s$ and $\tilde{Z}^t$ into Y^s and Y^t, respectively, and obtain the forecasting result Y by their sum.

Reconstruction and KL divergence: Among these two terms, the *KL divergence* can be easily estimated by Monte Carlo sampling with prior assumptions. Here we take a widely used setting that priors both follow $\mathcal{N}(0, I)$ for efficiency, more discussions of our priors can be found in Appendix C. As for *reconstruction* term, it cannot be directly measured owing to the unknown X^s and X^t. Besides, merging these two terms into $\mathbb{E}_{Q_{\phi^s,\phi^t}(Z^s,Z^t|X)}[\log P_{\theta^s,\theta^t}(X|Z^s, Z^t)]$ will result in chaos since the decoder is prone to reconstruct the intricate time series from every representation.

Theorem 2. *With the Gaussian distribution assumption, the reconstruction loss $\mathcal{L}_{rec}$ can be estimated without leveraging X^s and X^t, and Eq.* (4) *can be replaced with the following formula:*

$$\mathcal{L}_{rec} = -\sum_{\kappa=1}^{T-1} \left\| \mathcal{A}_{XX}(\kappa) - \mathcal{A}_{\hat{X}^s\hat{X}^s}(\kappa) \right\|^2 + \textit{CORT}(X, \hat{X}^t) - \left\| \hat{X}^t + \hat{X}^s - X \right\|^2, \quad (6)$$

$$\textit{CORT}(X, \hat{X}^t) = \frac{\sum_{i=1}^{T-1} \Delta X_i^t \Delta \hat{X}_i^t}{\sqrt{\sum_{i=1}^{T-1} \Delta X^t} \sqrt{\sum_{i=1}^{T-1} \Delta \hat{X}^t}}, \quad (7)$$

where $\mathcal{A}_{XX}(\kappa) = \sum_{i=1}^{T-\kappa} (X_t - \bar{X})(X_{t+\kappa} - \bar{X})$ is the autocorrelation coefficient with lagged value κ (we employ an efficient implementation in frequency domain [43], details are in Appendix B.2), CORT$(X, \hat{X}^t)$ is the temporal correlation coefficient, and $\Delta X_i = X_i - X_{i-1}$ is the first difference.

The proof is provided in Appendix A.2. According to Eq. (6), the reconstruction loss now can be estimated and, conversely, used to supervise disentangled representation learning. However, we find that this framework still holds certain drawbacks: (1) The KL divergence tends to narrow down the distance between posterior and prior. The modeling choice tends to sacrifice the variational inference vs. data fit when modeling capacity is not sufficient to achieve both [44]. The posterior may become almost non-informative for the inputs, which causes the forecastings irrelevant to the observations. (2) The disentanglement of the seasonal-trend representations is boosted indirectly by the separate reconstruction, where we need to impose a direct constraint on the representations themselves. We alleviate these limitations by introducing additional mutual information regularization terms. Specifically, we increase the mutual information between Z^s, Z^t and X to alleviate the divergence narrowing problem [44, 45], while decreasing mutual information between Z^s and Z^t to further dissociate their representations. The maximizing objective of LaST becomes

$$\mathcal{L}_{LaST} = \mathcal{L}_{ELBO} + I(X, Z^s) + I(X, Z^t) - I(Z^s, Z^t), \quad (8)$$

where $I(\cdot, \cdot)$ denotes the mutual information between two representations. However, the two mutual information terms are untraceable [46, 47, 48]. We address this problem in the next section.

4 Mutual information bounds for optimization

We now address the traceable mutual information bounds, maximizing $I(X, Z^s)$ and $I(X, Z^t)$, and minimizing $I(Z^s, Z^t)$ in Eq. (8), and provide lower and upper bounds for the model optimization.

Lower bound for $I(X, Z^s)$ or $I(X, Z^t)$. We omit the superscript s or t when analyzing lower bound. Among the prior approaches exploring the lower bounds for MI [49, 50, 51], MINE [51], for example, employs KL divergence between the joint distribution and marginals and defines an energy-based variational family to achieve a flexible and scalable lower bound. This can be formulated as $I(X, Z) \geq \mathbb{E}_{Q_\phi(X,Z)}[\gamma_\alpha(X, Z)] - \log \mathbb{E}_{Q(x)Q_\phi(z)}[e^{\gamma_\alpha(X,Z)}] = I_{\text{MINE}}$, where γ_α is a learned normalized critic with parameters α. However, this bound suffers from the biased gradient owing to the parametric logarithmic term (see Appendix A.3 for a proof). Inspired by [47], we substitute the logarithmic function by its tangent family to ameliorate the above biased bound:

$$\begin{aligned} I_{\text{MINE}} &\geq \mathbb{E}_{Q_\phi(X,Z)}[\gamma_\alpha(X, Z)] - (\frac{1}{\eta}\mathbb{E}_{Q(x)Q_\phi(z)}[e^{\gamma_\alpha(X,Z)}] + \log \eta - 1) \\ &\geq \mathbb{E}_{Q_\phi(X,Z)}[\gamma_\alpha(X, Z)] - \frac{1}{\eta}\mathbb{E}_{Q(x)Q_\phi(z)}[e^{\gamma_\alpha(X,Z)}], \end{aligned} \tag{9}$$

where η denotes the different tangent points. The first inequality relies on the concave negative logarithmic function – the values on the curve are upper bounds for that on the tangent line, and is tight when the tangent point overlaps the independent variable, i.e., the true value of $\mathbb{E}_{Q(x)Q(z)}[e^{\gamma(X,Z)}]$. The closer the distance between tangent point and independent variable, the greater the lower bound. Therefore, we set η as the variational term $\mathbb{E}_{Q(x)Q_\phi(z)}[e^{\gamma_\alpha(X,Z)}]$ that estimates the independent variable to obtain as great lower bound as possible. In the second inequality, $\gamma_\alpha(x, z)$ – a critic function activated by Sigmoid – is limited within $[0, 1]$ and thus $-(\log \eta - 1) \geq 0$. This inequality is tight only if $\mathbb{E}_{Q(x)Q_\phi(z)}[\gamma_\alpha(X, Z)] = 1$, which means γ_α can discriminate whether a pair of variables (X, Z) is sampled from the joint distribution or marginals. Similarly to MINE, this consistency problem can be addressed by the universal approximation theorem for neural networks [52]. Thus, Eq. (9) provides a flexible and scalable lower bound for $I(X, Z)$ with unbiased gradient.

For the evaluation, we exploit a traceable manner [53, 51] that draws joint samples $(X^{(i)}, Z^{(i)})$ by $Q(Z^{(i)}|X^{(i)})P_{\mathcal{D}}(X^{(i)})$. As for the marginal $Q_\phi(Z)$, we randomly select a datapoint j and then sample it from $Q_\phi(Z|X^{(j)})P_{\mathcal{D}}(X^{(j)})$. Details of the optimization are shown in Algorithm 1.

Upper bound for $I(Z^s, Z^t)$. Few efforts have been made that explore the traceable upper bound for mutual information [54, 47, 55]. Existing upper bounds (listed in Appendix D.1) are traceable with known probabilistic density of joint or conditional distributions here being $Q(Z^s|Z^t)$, $Q(Z^t|Z^s)$ or $Q(Z^s, Z^t)$. However, these distributions lack the interpretability and can hardly be directly modeled, which leads to untraceable estimations of the above upper bounds.

To avoid the direct estimation of unknown probabilistic densities, we introduce an energy-based variational family for $Q(Z^s, Z^t)$ that uses a normalized critic $\gamma_\beta(Z^s, Z^t)$ like Eq. (9) to establish a traceable upper bound. Specifically, we incorporate the critic γ_β into the upper bound I_{CLUB} [55] to obtain a traceable Seasonal-Trend Upper Bound (STUB) for $I(Z^s, Z^t)$, which is defined as:

$$I(Z^s, Z^t) \leq \mathbb{E}_{Q(Z^s,Z^t)}[\log Q(Z^s|Z^t)] - \mathbb{E}_{Q(Z^s)Q(Z^t)}[\log Q(Z^s|Z^t)] = I_{\text{CLUB}} \tag{10}$$

$$= \mathbb{E}_{Q_{\phi^s,\phi^t}(Z^s,Z^t)}[\gamma_\beta(Z^s, Z^t)] - \mathbb{E}_{Q_{\phi^s}(Z^s)Q_{\phi^t}(Z^t)}[\gamma_\beta(Z^s, Z^t)] = I_{\text{STUB}}. \tag{11}$$

The derivation details of this formula are provided in Appendix D.2. The inequality in Eq. (10) is tight only if Z^s and Z^t are a pair of independent variables [55]. This is exactly a sufficient condition for I_{STUB}, since MI and Eq. (11) are both zeros on the independent situation, which is our seasonal-trend disentanglement optimal objective. The critic γ_β, similar to γ_α, takes on the discriminating responsibility but provides converse scores, constraining the MI to a minimum. However, Eq. (11) may get negative values during the learning of parameter β, resulting an invalid upper bound for MI. To alleviate this problem, we additionally introduce a penalty term $\|I^{neg}_{STUB}\|^2$ to assist the model optimization, which is a L2 loss of the negative parts in I_{STUB}.

For the evaluation, we take the same sampling manner as the one in the lower bound and optimization details are also shown in Algorithm 1.

Algorithm 1 An epoch of the optimization of LaST.

Initialize the parameters of LaST: $\Theta = \{\psi^s, \psi^t, \phi^s, \phi^t, \theta^s, \theta^t\}$, $\Gamma = \{\alpha^s, \alpha^t, \beta\}$.
for a mini-batch with size $\mathcal{B}$ consisting of $\{X^{(i)}, Y^{(i)}\}_{i\in\mathcal{B}}$ **in** training set **do**
  Get samples of the latent representations $\{Z^{s(i)}\}_{i\in\mathcal{B}}$ and $\{Z^{t(i)}\}_{i\in\mathcal{B}}$ from distributions $\{Q_{\phi^s}(Z^s|X^{(i)})\}_{i\in\mathcal{B}}$ and $\{Q_{\phi^t}(Z^t|X^{(i)})\}_{i\in\mathcal{B}}$, respectively;
  Shuffle the $\{Z^{s(i)}\}_{i\in\mathcal{B}}$ and $\{Z^{t(i)}\}_{i\in\mathcal{B}}$ and form $\{Z^{s(j)}\}_{j\in\mathcal{B}}$ and $\{Z^{t(j)}\}_{j\in\mathcal{B}}$, respectively;
  Compute the η^s, η^t: $\eta^s \leftarrow \frac{1}{\mathcal{B}}\sum_{i=j=1}^{\mathcal{B}} e^{\gamma_{\alpha^s}(X^{(i)},Z^{s(j)})}$, $\eta^t \leftarrow \frac{1}{\mathcal{B}}\sum_{i=j=1}^{\mathcal{B}} e^{\gamma_{\alpha^t}(X^{(i)},Z^{t(j)})}$;
  Update parameters Θ: $\Theta \leftarrow G(\nabla_\Theta)[\mathcal{L}_{ELBO} + \frac{1}{\mathcal{B}}\sum_{i=j=1}^{\mathcal{B}}(\gamma_{\alpha^s}(X^{(i)}, Z^{s(i)}) - \frac{1}{\eta}e^{\gamma_{\alpha^s}(X^{(i)},Z^{s(j)})} + \gamma_{\alpha^t}(X^{(i)}, Z^{t(i)}) - \frac{1}{\eta}e^{\gamma_{\alpha^t}(X^{(i)},Z^{t(j)})}) - \frac{1}{\mathcal{B}}\sum_{i=j=1}^{\mathcal{B}}(\gamma_\beta(Z^{s(i)}, Z^{t(i)}) - \gamma_\beta(Z^{s(i)}, Z^{t(j)})) + average(\left\|(\gamma_\beta(Z^{s(i)}, Z^{t(i)}) - \gamma_\beta(Z^{s(i)}, Z^{t(j)}))^{neg}\right\|^2)]$;
  Update parameters Γ: $\Gamma \leftarrow G(\nabla_\Gamma)[\frac{1}{\mathcal{B}}\sum_{i=j=1}^{\mathcal{B}}(\gamma_{\alpha^s}(X^{(i)}, Z^{s(i)}) - \frac{1}{\eta}e^{\gamma_{\alpha^s}(X^{(i)},Z^{s(j)})} + \gamma_{\alpha^t}(X^{(i)}, Z^{t(i)}) - \frac{1}{\eta}e^{\gamma_{\alpha^t}(X^{(i)},Z^{t(j)})}) - \frac{1}{\mathcal{B}}\sum_{i=j=1}^{\mathcal{B}}(\gamma_\beta(Z^{s(i)}, Z^{t(i)}) - \gamma_\beta(Z^{s(i)}, Z^{t(j)}) + average(\left\|(\gamma_\beta(Z^{s(i)}, Z^{t(i)}) - \gamma_\beta(Z^{s(i)}, Z^{t(j)}))^{neg}\right\|^2)]$;
end for

5 Experiments

We now present the results of our extensive experimental evaluations comparing LaST with state-of-the-art baselines and report a series of empirical results, along with ablation study and visualizations of seasonal-trend representations. Further details and results are provided in Appendix F.

5.1 Settings

Datasets and Baselines. We conducted our experiments on seven real-world benchmark datasets from four categories of mainstream time series forecasting applications: (1) **ETT** [2][25]: Electricity Transformer Temperature consists of the target value "oil temperature" and six "power load" features, recorded hourly (i.e., ETTh1 and ETTh2) and every 15 minutes (i.e., ETTm1 and ETTm2) over two years. (2) **Electricity**, from the UCI Machine Learning Repository [3] and preprocessed by [56], is composed of the hourly electricity consumption of 321 clients in kWh from 2012 to 2014. (3) **Exchange** [56] with daily exchange rates of eight countries from 1990 to 2016. (4) **Weather** [4] contains 21 meteorological indicators (e.g., temperature and humidity) and is recorded every 10 minutes in 2020. We compare our LaST with latest state-of-the-art methods on time series modeling and forecasting tasks from two categories: (1) representation learning techniques, including COST [38], TS2Vec [37], and TNC [36]; (2) end-to-end forecasting models, including VAE-GRU [10], Autoformer [26], Informer [25], and TCN [22]. Further descriptions and settings of these baselines are provided in appendix F.1.

Evaluation setup. Following the prior work, we run our model on both univariate and multivariate forecasting settings. In multivariate forecasting, LaST accepts and forecasts all variables in datasets. In univariate forecasting, LaST only consider a specific feature in each dataset. We employ the standard normalization and set input length $T = 201$ for all datasets. For the dataset split, we follow standard protocol that categorizes all datasets into training, validation and test set in chronological order by the ratio of 6:2:2 for all datasets. We report the evaluation results on the test set while the model achieves the best performance on the validation set.

Implementation details. As for the network structure of LaST, we use a single-layer fully connected network as the feed forward network (FFN), which is applied in the modeling of posterior, reconstruction, and predictor. Besides, we employ the 2-layer MLP for the critic γ in MI bound estimations. Dimensions of seasonal and trend representations are consistent. We set them as 32 in univariate forecasting and as 128 in multivariate forecasting on other datasets. MAE loss is used to measure

[2] https://github.com/zhouhaoyi/ETDataset

[3] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams

[4] https://www.bgc-jena.mpg.de/wetter

Table 1: Univariate forecasting comparisons. Complete results on ETT benchmark are shown in appendix F.4. Best performance is highlighted in bold font and the second best results are underlined.

Method		**LaST**		CoST		TS2Vec		TNC		VAE-GRU		Autoformer		Informer		TCN	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	**0.030**	**0.131**	0.040	0.152	0.039	0.151	0.057	0.184	0.042	0.155	0.057	0.189	0.098	0.247	0.104	0.254
	48	**0.051**	**0.169**	0.060	0.186	0.062	0.189	0.094	0.239	0.077	0.218	0.070	0.207	0.158	0.319	0.206	0.366
	168	**0.078**	**0.211**	0.097	0.236	0.142	0.291	0.171	0.329	0.172	0.344	0.108	0.260	0.183	0.346	0.462	0.586
	336	**0.100**	**0.246**	0.112	0.258	0.160	0.316	0.179	0.345	0.140	0.301	0.119	0.281	0.222	0.387	0.422	0.564
	720	0.138	0.298	0.148	0.306	0.179	0.345	0.235	0.408	0.204	0.381	**0.109**	**0.264**	0.269	0.435	0.438	0.578
ETTm1	24	**0.011**	**0.077**	0.015	0.088	0.016	0.093	0.019	0.103	0.013	0.082	0.022	0.115	0.030	0.137	0.027	0.127
	48	**0.021**	**0.108**	0.025	0.117	0.028	0.126	0.045	0.162	0.026	0.120	0.032	0.138	0.069	0.203	0.040	0.154
	96	**0.033**	**0.134**	0.038	0.147	0.045	0.162	0.054	0.178	0.046	0.164	0.045	0.168	0.194	0.372	0.097	0.246
	288	**0.069**	**0.197**	0.077	0.209	0.095	0.235	0.142	0.290	0.127	0.294	0.071	0.207	0.401	0.554	0.305	0.455
	672	**0.100**	**0.239**	0.113	0.257	0.142	0.290	0.136	0.290	0.217	0.399	0.102	0.254	0.512	0.644	0.445	0.576
Electricity	24	**0.151**	**0.277**	0.243	0.264	0.260	0.288	0.252	0.278	0.330	0.406	0.290	0.411	0.251	0.275	0.243	0.367
	48	**0.186**	**0.307**	0.292	0.300	0.313	0.321	0.300	0.308	0.437	0.481	0.310	0.408	0.346	0.339	0.283	0.397
	168	**0.243**	**0.346**	0.405	0.375	0.429	0.392	0.412	0.384	0.433	0.476	0.435	0.490	0.544	0.424	0.357	0.449
	336	**0.286**	**0.379**	0.560	0.473	0.565	0.478	0.548	0.466	0.472	0.504	0.646	0.606	0.713	0.512	0.355	0.446
	720	**0.322**	**0.422**	0.889	0.645	0.863	0.651	0.859	0.651	0.543	0.563	0.609	0.587	1.182	0.806	0.387	0.477
Average		**0.121**	**0.236**	0.208	0.268	0.222	0.289	0.234	0.328	0.219	0.326	0.201	0.306	0.345	0.400	0.278	0.403

the forecasting derived from the predictor. For the training strategy, we use the Adam [57] optimizer and training process is early stopped within 10 epoches. We initialize the learning rate with 10^{-3} and decay it with 0.95 weight every epoch.

5.2 Performance comparisons and model analysis

Effectiveness. Tables 1 and 2 summarize the results of univariate and multivariate forecastings respectively. LaST achieves state-of-the-art performance against the advanced representation baselines on five real-world datasets. The relative improvements on MSE and MAE are 25.6% and 22.1% against the best representation learning method CoST and are 22.0% and 18.9% against the best end-to-end models Autoformer. We note that Autoformer achieves better performance on long horizons forecasting on hourly ETT datasets and think there are two reasons: (1) Transformer-based models intrinsically establish long-range dependencies, which plays a crucial role in long sequence forecasting; (2) it employs a simple decomposition by average pooling with a fixed kernel size, which is more suitable for strongly periodic dataset like hourly ETT. This phenomenon is beneficial to long-term forecasting but limits the sensitivity to local context, and the bonus does not have significant impact on other datasets. Compared with baselines, LaST extracts the seasonal and trend patterns with disentangled representations adaptively and thus can be applied to intricate time series.

Ablation study. We investigated the performance benefits brought by each mechanism of LaST on a synthetic dataset (generation details are provided in appendix F.3) and ETTh1. The results are shown in Table 3, consisting of two groups: **M1** validates the mechanisms of seasonal-trend representations learning framework. In it, "w/o seasonal" and "w/o trend" denote LaST without the seasonal and trend components respectively; "w/o coe" denotes LaST without autocorrelation and CORT coefficients while estimating the reconstruction loss. **M2** judges the introduction and estimations of MI, where"w/o lower" and "w/o upper" indicate the removal of the lower and upper bounds for MI in regularization terms respectively; "with MINE" denotes that we replace our lower bound with MINE. The results show that all mechanisms improve the performance on the forecasting task. We notice that the quality drops a lot when removing the trend component. The reason is the seasonal forecasting derives from the iDFT algorithm, which is essentially a periodical repetition of the history observations. However, it captures the seasonal patterns and assists the trend component in complete LaST to bring the superiority, especially in the long-term settings and strongly periodical synthetic dataset. Besides, we observe that with biased regularization term MINE, the performance becomes unstable and sometimes even worse than LaST without MI lower bound, while our unbiased bound (cf. Eq.(9)) continuously outperforms it.

Representation disentanglement. We visualize the seasonal-trend representations with the t-SNE [58] technique in Figure 2. We also visualize the embeddings in last layer of Autoformer decoder as a comparison. The points with same color have a clearer and closer clustering in LaST, while they mix together without decomposition mechanisms ("w/o dec" indicates removal of the two decomposition mechanisms (autocorrelation and CORT coefficients, and the upper bound to MI).

Table 2: Multivariate forecasting comparisons. Complete results on ETT benchmark are shown in appendix F.4. Best performance is highlighted in bold font and the second best results are underlined.

Method		**LaST**		CoST		TS2Vec		TNC		VAE-GRU		Autoformer		Informer		TCN	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	**0.324**	**0.368**	0.386	0.429	0.590	0.531	0.708	0.592	0.529	0.534	0.384	0.428	0.577	0.549	0.583	0.547
	48	**0.351**	**0.380**	0.437	0.464	0.624	0.555	0.749	0.619	0.612	0.593	0.392	0.419	0.685	0.625	0.670	0.606
	168	**0.468**	**0.453**	0.643	0.582	0.762	0.639	0.884	0.699	0.758	0.647	0.490	0.481	0.931	0.752	0.811	0.680
	336	0.566	0.512	0.812	0.679	0.931	0.728	1.020	0.768	0.844	0.692	**0.505**	**0.484**	1.128	0.873	1.132	0.815
	720	0.740	0.650	0.970	0.771	1.063	0.799	1.157	0.830	1.045	0.816	**0.498**	**0.500**	1.215	0.896	1.165	0.813
ETTm1	24	**0.218**	**0.289**	0.246	0.329	0.453	0.444	0.522	0.472	0.509	0.452	0.383	0.403	0.453	0.444	0.522	0.472
	48	**0.280**	**0.329**	0.331	0.386	0.592	0.521	0.695	0.567	0.642	0.543	0.454	0.453	0.494	0.503	0.542	0.508
	96	**0.323**	**0.360**	0.378	0.419	0.635	0.554	0.731	0.595	0.600	0.540	0.481	0.463	0.678	0.614	0.666	0.578
	288	**0.392**	**0.403**	0.472	0.486	0.693	0.597	0.818	0.649	0.769	0.678	0.634	0.528	1.056	0.786	0.991	0.735
	672	**0.491**	**0.466**	0.620	0.574	0.782	0.653	0.932	0.712	0.799	0.673	0.606	0.542	1.192	0.926	1.032	0.756
Electricity	24	**0.125**	**0.222**	0.136	0.242	0.287	0.375	0.354	0.423	0.190	0.250	0.165	0.286	0.312	0.387	0.235	0.346
	48	**0.146**	**0.245**	0.153	0.258	0.309	0.391	0.376	0.438	0.228	0.280	0.178	0.295	0.392	0.431	0.253	0.359
	168	**0.170**	**0.265**	0.175	0.275	0.335	0.410	0.402	0.456	0.240	0.297	0.215	0.327	0.515	0.509	0.278	0.372
	336	**0.188**	**0.280**	0.196	0.296	0.351	0.422	0.417	0.466	0.262	0.318	0.218	0.329	0.759	0.625	0.287	0.382
	720	**0.223**	**0.309**	0.232	0.327	0.378	0.440	0.442	0.483	0.296	0.347	0.252	0.356	0.969	0.788	0.287	0.381
Exchange	24	**0.033**	**0.122**	**0.033**	0.127	0.108	0.252	0.105	0.236	0.064	0.178	0.060	0.178	0.611	0.626	2.483	1.327
	48	**0.056**	**0.162**	0.058	0.165	0.200	0.341	0.162	0.270	0.133	0.262	0.091	0.222	0.680	0.644	2.328	1.256
	168	**0.190**	**0.320**	0.198	0.327	0.412	0.492	0.397	0.480	0.334	0.432	0.405	0.473	1.097	0.825	2.372	1.279
	336	**0.430**	**0.482**	0.512	0.523	1.339	0.901	1.008	0.866	0.614	0.606	0.509	0.524	1.672	1.036	3.113	1.459
	720	1.521	**0.898**	1.855	0.998	2.114	1.125	1.989	1.063	2.285	1.117	**1.447**	0.941	2.478	1.310	3.150	1.458
Weather	24	**0.105**	**0.134**	0.293	0.369	0.170	0.309	0.200	0.312	0.117	0.147	0.180	0.263	0.162	0.235	0.170	0.287
	48	**0.131**	**0.174**	0.558	0.515	0.231	0.375	0.284	0.367	0.227	0.270	0.241	0.310	0.348	0.400	0.327	0.365
	168	**0.197**	**0.238**	0.812	0.671	0.470	0.532	0.475	0.502	0.234	0.280	0.295	0.355	0.444	0.463	0.517	0.569
	336	**0.257**	**0.285**	1.196	0.832	1.360	0.875	1.405	0.881	0.309	0.339	0.359	0.395	0.578	0.523	0.639	0.608
	720	**0.315**	**0.327**	1.620	1.002	2.173	1.120	2.034	1.018	0.444	0.410	0.419	0.428	1.059	0.741	0.639	0.610
Average		**0.330**	**0.347**	0.533	0.482	0.654	0.575	0.731	0.591	0.487	0.468	0.394	0.415	0.819	0.661	1.008	0.663

Table 3: Ablation study results on the two parts: **M1** and **M2**.

Method		Original		M1						M2					
		LaST		w/o seasonal		w/o trend		w/o coe		w/o lower		w/o upper		with MINE	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Synthetic	24	**0.129**	**0.250**	0.179	0.298	2.870	1.089	0.135	0.257	0.133	0.256	0.146	0.266	0.131	0.252
	168	**0.631**	**0.617**	0.830	0.644	2.868	1.032	0.635	0.619	0.632	0.618	0.634	0.620	0.633	0.620
	336	**1.054**	**0.797**	1.505	0.925	2.864	1.016	1.061	0.805	1.056	0.801	1.055	0.798	1.055	0.801
ETTh1	24	**0.324**	**0.368**	0.328	0.373	0.631	0.548	0.334	0.380	0.339	0.383	0.332	0.376	0.332	0.375
	48	**0.351**	**0.380**	0.360	0.389	0.973	0.695	0.358	0.386	0.360	0.389	0.361	0.390	0.358	0.387
	168	**0.468**	**0.453**	0.482	0.463	1.106	0.788	0.497	0.474	0.508	0.482	0.477	0.459	0.502	0.484
	336	**0.566**	**0.512**	0.579	0.531	1.206	0.843	0.598	0.546	0.604	0.543	0.582	0.534	0.603	0.545
	720	**0.740**	**0.650**	0.788	0.672	1.240	0.854	0.803	0.689	0.766	0.665	0.768	0.670	0.780	0.669
Average		**0.533**	**0.503**	0.631	0.537	1.720	0.858	0.553	0.520	0.550	0.517	0.544	0.514	0.549	0.517

Notably, though Autofomer with the simple moving average block achieves satisfying decomposition from the time series perspective, their representations are still prone to entanglement. These results suggest that (1) learning disentangled seasonal-trend representations is not trivial, and (2) the proposed decomposition mechanisms successfully disentangle the seasonal-trend representations in latent space, each paying attention to a specific temporal pattern.

Input settings. We further investigate the influence of hyperparameter input length to validate the sensitivity and Table 4 shows the results. Long look-back window improves the performance especially in long-term forecasting, while others even have performance degradation. This verifies that LaST can effectively utilize the past information to understand patterns and make predictions.

Table 4: Multivariate forecasting performance with different input lengths on ETTm1 datasets.

Input length		96			168			201			672			Average
Output length		48	288	672	48	288	672	48	288	672	48	288	672	
LaST	MSE	**0.321**	**0.427**	**0.532**	**0.286**	**0.401**	**0.516**	**0.280**	**0.392**	**0.491**	**0.279**	**0.373**	**0.463**	**0.397**
	MAE	**0.359**	**0.421**	**0.493**	**0.338**	**0.413**	**0.493**	**0.329**	**0.403**	**0.466**	**0.334**	**0.391**	**0.448**	**0.407**
CoST	MSE	0.390	0.514	0.644	0.335	0.467	0.611	0.331	0.472	0.620	0.349	0.474	0.612	0.485
	MAE	0.422	0.507	0.585	0.392	0.484	0.569	0.386	0.486	0.574	0.399	0.486	0.567	0.488
AutoFormer	MSE	0.454	0.634	0.606	0.460	0.578	0.529	0.463	0.545	0.562	0.562	0.656	0.695	0.562
	MAE	0.453	0.528	0.542	0.456	0.514	0.504	0.452	0.512	0.516	0.503	0.571	0.586	0.511

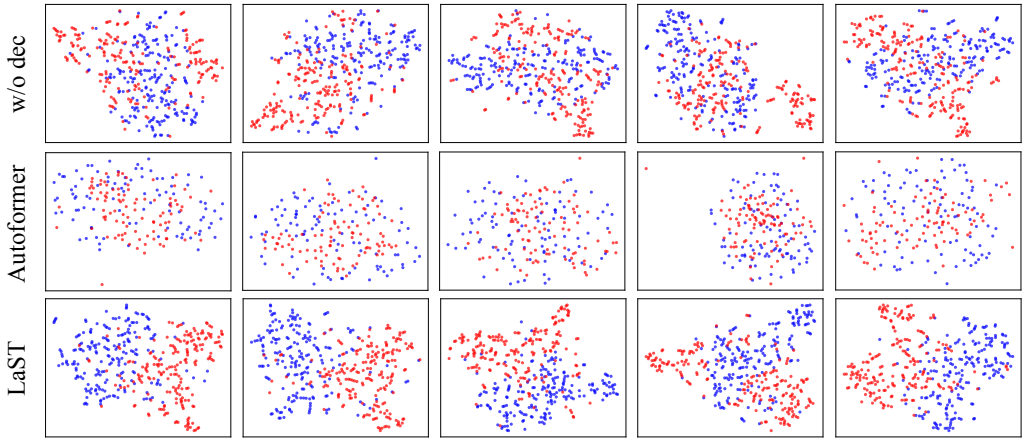
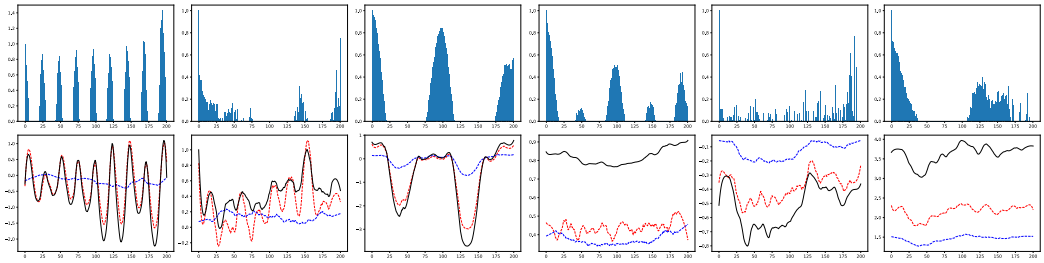


Figure 2: Visualizations of seasonal (red) and trend (blue) representations on ETTh1 dataset.



(a) ETTh1 (b) ETTm1 (c) Exchange

Figure 3: *Top*: learned seasonality visualizations (autocorrelation statistics of reconstructed seasonal sequences). *Bottom*: seasonal (red) and trend (blue) reconstructions to the ground truths (black).

Observations from a case-study. We further validate LaST by by visualizing the extracted seasonality and trend in specifica cases. As shown in Figure 3, LaST can capture the seasonal patterns on real-world datasets. For example, a strong daily period is indicated on hourly and 15-minutes ETT datasets. Even the period on Exchange dataset is not obvious, LaST still provides some long-term periods on the daily data. Besides, trend and seasonal components jointly accurately restore the original sequence with their own perspective, which supports that LaST can produce workable disentangled representations for intricate time series.

6 Conclusion

We presented LaST, a disentangled variational inference framework with mutual information constraints to disassociate a couple of seasonal-trend representations in latent space, for effective forecasting of time series. Our extensive experiments demonstrated that LaST successfully disentangles the seasonal-trend representations and achieves state-of-the-art performance. Our future work will focus on tackling other challenging downstream tasks in the time series domain, e.g., generation and imputation. In addition, we plan to model stochastic factors explicitly in decomposition strategies, which will better understand the real-world time series.

Acknowledgments and Disclosure of Funding

This work was supported in part by National Natural Science Foundation of China (Grant No.62072077 and No.62176043), Natural Science Foundation of Sichuan Province (Grant No. 2022NSFSC0505), and National Science Foundation SWIFT (Grant No.2030249).

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Our source code for reproducibility is included in the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See "Datasets" and "Evaluation setup" in Sec. 5.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Appendix F.5.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix F.2.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See URLs in Sec. 5.
   (b) Did you mention the license of the assets? [Yes] See folder "license" in supplementary materials.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Our source code for reproducibility is included in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See folder "license" in supplementary materials.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] They do not contain personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

References

[1] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878):672–677, 2021.

[2] Zhiyuan Wang, Xovee Xu, Goce Trajcevski, Kunpeng Zhang, Ting Zhong, and Fan Zhou. PrEF: Probabilistic electricity forecasting via copula-augmented state space model. In *AAAI*, 2022.

[3] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

[4] Amadou Ba, Mathieu Sinn, Yannig Goude, and Pascal Pompey. Adaptive learning of smoothing functions: Application to electricity load forecasting. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[5] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

[6] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[8] Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020.

[9] Matthew Johnson and Alan Willsky. Stochastic variational inference for bayesian time series models. In *International Conference on Machine Learning*, pages 1854–1862. PMLR, 2014.

[10] Otto Fabius, Joost R van Amersfoort, and Diederik P Kingma. Variational recurrent auto-encoders. In *ICLR (Workshop)*, 2015.

[11] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. Gp-vae: Deep probabilistic time series imputation. In *International conference on artificial intelligence and statistics*, pages 1651–1661. PMLR, 2020.

[12] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[14] William Whitney. Disentangled representations in neural models. *arXiv preprint arXiv:1602.02383*, 2016.

[15] Cleveland Robert, C William, and Terpenning Irma. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of official statistics*, 6(1):3–73, 1990.

[16] Mike West. Time series decomposition. *Biometrika*, 84(2):489–494, 1997.

[17] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[18] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2019.

[19] Wei Fan, Shun Zheng, Xiaohan Yi, Wei Cao, Yanjie Fu, Jiang Bian, and Tie-Yan Liu. Depts: Deep expansion learning for periodic time series forecasting. In *International Conference on Learning Representations*, 2022.

[20] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

[21] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31, 2018.

[22] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[23] Minhao Liu, Ailing Zeng, Zhijian Xu, Qiuxia Lai, and Qiang Xu. Time series is a special sequence: Forecasting with sample convolution and interaction. *arXiv preprint arXiv:2106.09305*, 2021.

[24] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, 32, 2019.

[25] 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 AAAI*, 2021.

[26] Jiehui Xu, Jianmin Wang, Mingsheng Long, et al. Autoformer: Decomposition transformers with autocorrelation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34, 2021.

[27] G Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.

[28] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[29] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[30] Pankaj Malhotra, Vishnu TV, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Timenet: Pre-trained deep recurrent neural network for time series classification. *arXiv preprint arXiv:1706.08838*, 2017.

[31] Xinrui Lyu, Matthias Hueser, Stephanie L Hyland, George Zerveas, and Gunnar Raetsch. Improving clinical predictions through unsupervised time series representation learning. *arXiv preprint arXiv:1812.00490*, 2018.

[32] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. *Advances in neural information processing systems*, 32, 2019.

[33] Emmanuel de B'ezenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. Normalizing kalman filters for multivariate time series analysis. In *Advances in Neural Information Processing Systems*, volume 33, pages 2995–3007, 2020.

[34] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[35] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs M Bergmann, and Roland Vollgraf. Multivariate probabilistic time series forecasting via conditioned normalizing flows. In *International Conference on Learning Representations*, 2020.

[36] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations*, 2021.

[37] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of AAAI*, 2022.

[38] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International Conference on Learning Representations*, 2022.

[39] Qingsong Wen, Jingkun Gao, Xiaomin Song, Liang Sun, Huan Xu, and Shenghuo Zhu. Robuststl: A robust seasonal-trend decomposition algorithm for long time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5409–5416, 2019.

[40] Qingsong Wen, Zhe Zhang, Yan Li, and Liang Sun. Fast robuststl: Efficient and robust seasonal-trend decomposition for time series with complex patterns. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2203–2213, 2020.

[41] Luke B Godfrey and Michael S Gashler. Neural decomposition of time-series data for effective generalization. *IEEE transactions on neural networks and learning systems*, 29(7):2973–2985, 2017.

[42] Michail Vlachos, Philip Yu, and Vittorio Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 449–460. SIAM, 2005.

[43] Autocorrelation: Efficient computation. `https://en.wikipedia.org/wiki/Autocorrelation#Efficient_computation`. Accessed: 2022-05-19.

[44] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Infovae: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*, 2017.

[45] Ali Lotfi Rezaabad and Sriram Vishwanath. Learning representations by maximizing mutual information in variational autoencoders. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2729–2734. IEEE, 2020.

[46] Liam Paninski. Estimation of entropy and mutual information. *Neural computation*, 15(6):1191–1253, 2003.

[47] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

[48] David McAllester and Karl Stratos. Formal limitations on the measurement of mutual information. In *International Conference on Artificial Intelligence and Statistics*, pages 875–884. PMLR, 2020.

[49] Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time. iv. *Communications on pure and applied mathematics*, 36(2):183–212, 1983.

[50] David Barber and Felix V Agakov. The im algorithm: A variational approach to information maximization. In *Advances in Neural Information Processing Systems*, 2003.

[51] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pages 531–540. PMLR, 2018.

[52] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[53] Miguel A Arcones and Evarist Gine. On the bootstrap of u and v statistics. *The Annals of Statistics*, pages 655–674, 1992.

[54] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations, ICLR*, 2017.

[55] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. Club: A contrastive log-ratio upper bound of mutual information. In *International conference on machine learning*, pages 1779–1788. PMLR, 2020.

[56] 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.

[57] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*, 2015.

[58] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Learning Latent Seasonal-Trend Representations for Time Series Forecasting (Technical Appendix)

In this body of supplementary materials, we first provide the proofs of ELBO inference with decomposition, reconstruction loss alternative, and MINE biased gradient. Next, we discuss the use of Discrete Fourier transform and autocorrelation methods in our work. This is followed by a more detailed discussion of the use of prior distribution, mutual information upper bounds, along with the complexity results of our algorithmic solutions. Lastly, we present the details of the experimental settings and provide some additional results to support the claims in the main text.

A Proofs

Following are the details of the proofs of the claims from the main text – Theorem 1, Theorem 2 and the MI lower bound (cf. Section 4).

A.1 Inference of ELBO with decomposition

We follow the decomposition strategy, in the sense that time series and their representations are composed of multiple components such as seasonal Z^s and trend Z^t. Sequence X and Y is decomposed as a sum form, i.e., $X = X^s + X^t$, $Y = Y^s + Y^t$. Representations Z^s and Z^t are independent from each other, i.e., $P(Z^s, Z^t) = P(Z^s)P(Z^t)$ and, additionally, Z^t and Z^s are only associated with their own components. With the above assumptions, we have:

$$\begin{aligned}
\mathcal{L}_{ELBO} &= \log \int_Z P_\psi(Y|Z) Q_\phi(Z|X) dZ + \mathbb{E}_{Q_\phi(Z|X)}[\log P_\theta(X|Z)] - \mathbb{KL}(Q_\phi(Z|X)||P(Z)) \\
&= \log \int_{Z^s} \int_{Z^t} P_\psi(Y|Z^s, Z^t) Q_{\phi^s,\phi^t}(Z^s, Z^t|X) dZ^s dZ^t \\
&\quad + \mathbb{E}_{Q_{\phi^s,\phi^t}(Z^s,Z^t|X)}[\log P_\theta(X|Z^s, Z^t)] - \mathbb{KL}(Q_{\phi^s,\phi^t}(Z^s, Z^t|X)||P(Z^s, Z^t)) \\
&= \log \int_{Z^s} \int_{Z^t} P_\psi(Y|Z^s, Z^t) Q_{\phi^s,\phi^t}(Z^s, Z^t|X) dZ^s dZ^t \\
&\quad + \mathbb{E}_{Q_{\phi^s}(Z^s|X)}[\log P_\theta(X|Z^s)] + \mathbb{E}_{Q_{\phi^s}(Z^t|X)}[\log P_\theta(X|Z^t)] \\
&\quad - \mathbb{KL}(Q_{\phi^s}(Z^s|X)||P(Z^s)) - \mathbb{KL}(Q_{\phi^t}(Z^t|X)||P(Z^t)) \\
&= \log \int_Z P_\psi(Y|Z^t, Z^s) Q_{\phi^s,\phi^t}(Z^s, Z^t|X) dZ^s dZ^t \\
&\quad + \mathbb{E}_{Q_{\phi^s}(Z^s|X)}[\log P_{\theta^s}(X^s|Z^s)] + \mathbb{E}_{Q_{\phi^t}(Z^t|X)}[\log P_{\theta^t}(X^t|Z^t)] \\
&\quad - \mathbb{KL}(Q_{\phi^s}(Z^s|X)||P(Z^s)) - \mathbb{KL}(Q_{\phi^t}(Z^t|X)||P(Z^t)).
\end{aligned} \tag{A1}$$

A.2 Proof of the alternative of reconstruction loss

The loss function of $\log P_\theta(X|Z)$ can be written with Gaussian distribution as:

$$-\log P_\theta(X|Z) = \left(\log \sigma_\theta(Z) + \frac{1}{2} \log 2\pi + \frac{1}{2} \frac{\|X - \mu_\theta(Z)\|^2}{\sigma_\theta(Z)} \right) \propto \|X - \mu_\theta(Z)\|^2, \tag{A2}$$

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

where $\mu_\theta(Z)$ and $\sigma_\theta(Z)$ are neural networks that reconstruct the X from latent representations. By the above equation, maximizing the reconstruction loss is regarded as minimizing the euclidean distance between inputs and reconstructions.

Thus, we opt to optimize the reconstruction loss in LaST by finding a way that imposes closer distances between raw input X^s, X^t, and reconstructed input $\hat{X}^s$, $\hat{X}^t$. There are three terms in Eq. (6) in main text:

$$\mathcal{L}_{rec} = -\sum_{\kappa=1}^{T-1} \left\| \mathcal{A}_{XX}(\kappa) - \mathcal{A}_{\hat{X}^s\hat{X}^s}(\kappa) \right\|^2 + \text{CORT}(X, \hat{X}^t) - \left\| \hat{X}^t + \hat{X}^s - X \right\|^2,$$

1. The first term reflects the similarity between every instant value in X and its κ-lagged value. We can regard the function $\mathcal{A}_{XX}(\kappa)$ as an unnormalized score that measures the confidence of period κ [1, 2]. A lower distance between the two autocorrelation sequences means more similar periodic characteristics. With identical autocorrelations, reconstructed $\hat{X}^s$ will be same as X^s in the seasonal view but still holds interference. The optimized reconstructed $\hat{X}^s$ can be formulated as $\hat{X}^s = X^s + \varepsilon^s$, where ε denotes the interference error.
2. The second normalized term reflects the similarity between the first differences of the input time series, which further measures the simultaneity of two signals' rising or falling patterns, i.e., the trend. With this objective, the reconstructed trend input is formulate as $\hat{X}^t = X^t + \varepsilon^t$ similar to $\hat{X}^s$.
3. The third term now naturally becomes $\|\varepsilon^s + \varepsilon^t\|^2$ aiming to minimize the interference terms.

These three terms are used to reconstruct the pure seasonal and trend components and optimize the above reconstruction loss.

A.3 Biased gradient of MINE

The lower bound of MI between X and Z is estimated by MINE [3], which is defined as

$$I(X, Z) \geq \mathbb{E}_{Q_\phi(X,Z)}[\gamma_\alpha(X, Z)] - \log \mathbb{E}_{Q(x)Q_\phi(z)}[\exp(\gamma_\alpha(X, Z))] = I_{MINE}. \tag{A3}$$

When sampling the time series in a mini-batch $\mathcal{B}$ (see details in Algorithm 1), it becomes

$$I_{MINE}(X, Z) = \frac{1}{\mathcal{B}} \sum_{i=j=1}^{\mathcal{B}} \gamma_\alpha(X^{(i)}, Z^{(i)}) - \log \frac{1}{\mathcal{B}} \sum_{i=j=1}^{\mathcal{B}} \exp(\gamma_\alpha(X^{(i)}, Z^{(j)})), \tag{A4}$$

where $Z^{(i)}$ and $Z^{(j)}$ are derived from parameters-driven encoders $Q_\phi(Z|X)$ with input X. The gradient of MINE is calculated by general stochastic optimizer such as Adam [4] or others:

$$G_{\mathcal{B}} = \frac{1}{\mathcal{B}} \sum_{i=j=1}^{\mathcal{B}} \nabla_{\alpha,\phi}\gamma_\alpha(X^{(i)}, Z^{(i)}) - \frac{\sum_{i=j=1}^{\mathcal{B}} \nabla_{\alpha,\phi}\gamma_\alpha(X^{(i)}, Z^{(j)} \exp(\gamma_\alpha(X^{(i)}, Z^{(j)})}{\sum_{i=j=1}^{\mathcal{B}} \exp(\gamma_\alpha(X^{(i)}, Z^{(j)})}, \tag{A5}$$

where the second term holds bias since the derivative of logarithmic function.

B Discrete Fourier transform and autocorrelation

The details of our use of Discrete Fourier Transform and efficient autocorrelation are presented next.

B.1 Discrete Fourier Transform

We apply discrete Fourier Transform (DFT) and inverse Discrete Fourier Transform (iDFT) to construct the mapping between time and frequency domain (cf. Figure 1 in the main text). First, given a time domain sequence of seasonality $Z^s = \{Z_1^s, Z_2^s, \ldots, Z_T^s\}$, we consider the first $\lfloor \frac{T+1}{2} \rfloor$ Fourier coefficients according to the Nyquist theorem [5]:

$$Z_{\mathcal{F},k}^s = \mathcal{F}(Z^s)_k = \sum_{t=1}^{T} Z_t^s \cdot \exp(\frac{-2\pi ikt}{T}), \tag{B1}$$

where $1 \leq k \leq \lfloor \frac{T+1}{2} \rfloor$. Then time domain seasonal representation is extended and mapped from frequency domain with iDFT:

$$Z_t^s = \mathcal{F}^{-1}(Z_{\mathcal{F}^s})_t = \frac{1}{T} \sum_{k=1}^{T} Z_{\mathcal{F},k}^s \cdot \exp(\frac{2\pi i k t}{T}), \text{ where } 1 \leq t \leq T + \tau, \tag{B2}$$

where we derive predictable representations $\tilde{Z}^s$ by taking the last τ time steps. In LaST, we use the fast version of DFT [6] based on halving lemma and divide-and-conquer strategy, based on which the time complexity becomes $\mathcal{O}(T \log T)$, as opposed to the original $\mathcal{O}(T^2)$.

B.2 Efficient autocorrelation

The autocorrelation of sequence X with κ lagged steps is defined as

$$\mathcal{A}_{XX}(\kappa) = \sum_{i=1}^{T-\kappa} (X_t - \bar{X})(X_{t+\kappa} - \bar{X}). \tag{B3}$$

To obtain the autocorrelation sequence, it requires $T(T-1)/2$ times multiplications, each of which holds $\mathcal{O}(d^2)$ time complexity, where d is the dimension of X. To improve the efficiency, we employ the Wiener–Khinchin theorem [7] that allows computing the autocorrelation with fast Fourier transform (FFT) and its inverse transform as:

$$X_{\mathcal{F}} = FFT(X^s), \tag{B4}$$

$$S_{\mathcal{F}} = X_{\mathcal{F}} X_{\mathcal{F}}^*, \tag{B5}$$

$$\mathcal{A}_{XX}(0 : T-1) = iFFT(S_{\mathcal{F}}), \tag{B6}$$

where the asterisk denotes complex conjugate, and the output $\mathcal{A}_{XX}(0 : T-1)$ denotes sequence $\{\mathcal{A}_{XX}(\kappa)\}_{\kappa \in [0:T-1]}$. In this setting, the number of multiplication calculations drops to $T \log T$ (from the original quadratic complexity).

C Discussions of prior distributions

Choosing an appropriate prior for the latent representations in VAE is important for mediating between the encoder and decoder. The standard Gaussian $\mathcal{N}(0, I)$ is a commonly used prior assumption because: (1) it is efficient without additional calculation overhead and (2) the input data are preprocessed with standard normalization. We note that any other advanced approaches on prior learning can be easily incorporated into our model. In the following, we take VampPrior [8] – which produces prior distribution through the encoder Q_ϕ given pseudo inputs – as an example to show the effects of different priors on the forecasting performance of LaST. The priors $P(Z^s)$ and $P(Z^t)$ are obtained by $Q_{\phi^s}(Z^s|I)$ and $Q_{\phi^t}(Z^t|I)$, respectively, with identity matrix as pseudo inputs.

Table C1: Multivariate forecasting performance with different prior distributions.

Dataset		ETTh1			ETTm1			Exchange			Weather		
		48	336	720	48	288	672	48	336	720	48	336	720
Standard Gaussian	MSE	0.351	**0.566**	**0.740**	**0.280**	0.392	0.491	0.056	0.430	**1.521**	**0.131**	0.257	0.315
	MAE	0.380	**0.512**	**0.650**	**0.329**	**0.403**	**0.466**	0.162	**0.482**	**0.898**	**0.174**	0.285	0.327
VampPrior	MSE	**0.349**	0.567	0.765	0.284	**0.390**	**0.490**	**0.054**	**0.426**	1.763	0.137	**0.250**	**0.313**
	MAE	**0.377**	0.522	0.666	0.337	0.411	0.468	**0.161**	0.484	0.927	0.196	**0.279**	**0.326**

Table C1 shows the experimental results. We can see that VampPrior and standard Gaussian are well matched in LaST on time series forecasting task. However, none shows clear superiority over the other on four datasets. Thus, we use the standard Gaussian as the default prior in LaST due to its efficiency.

D Existing upper bounds and the proposed I_{STUB}

We now discuss in detail the portion pertaining to the upper bounds (cf. Section 4 of the main paper).

D.1 Existing upper bounds

Existing research results have concluded several traceable and appropriate upper bounds to MI between X and Y [9, 10, 11], they include:

$$I_{VUB}(X,Y) = \mathbb{E}_{P(X,Y)}\left[\frac{\log P(Y|X)}{Q(Y)}\right], \tag{D1}$$

$$I_{L1Out}(X,Y) = \mathbb{E}\left[\frac{1}{\mathcal{B}}\sum_{i=1}^{\mathcal{B}}\left[\log\frac{P(Y^{(i)}|X^{(i)})}{\frac{1}{\mathcal{B}-1}\sum_{j\neq i}P(Y^{(i)}|X^{(j)})}\right]\right], \tag{D2}$$

$$I_{CLUB}(X,Y) = \mathbb{E}_{P(X,Y)}[\log P(Y|X)] - \mathbb{E}_{P(X)P(Y)}[\log P(Y|X)], \tag{D3}$$

where $Q(Y)$ is usually a learned marginal density approximation to $P(Y)$ and $\mathcal{B}$ is the batch size. These upper bounds are all based on conditional distributions which, defined as $Q(Z^s|Z^t)$ or $Q(Z^t|Z^s)$ in our LaST, are still untraceable and ineffective. Though we can replace them with joint distribution via deductions, for example,

$$\begin{aligned} I(Z^s,Z^t) &\leq \mathbb{E}_{Q(Z^s,Z^t)}[\log Q(Z^s|Z^t) + \log Q(Z^t)] - \mathbb{E}_{Q(Z^s)Q(Z^t)}[\log Q(Z^s|Z^t) + \log Q(Z^t)] \\ &= \mathbb{E}_{Q(Z^s,Z^t)}[\log Q(Z^s,Z^t)] - \mathbb{E}_{Q(Z^s)Q(Z^t)}[\log Q(Z^s,Z^t)], \end{aligned} \tag{D4}$$

it is challenging to establish a joint distribution and estimate its density directly without any additional information [12].

D.2 Derivation of proposed upper bound I_{STUB}

To overcome the limitations of the existing upper bounds, we introduce a normalized energy-based function that uses a critic $\gamma_\beta(Z^s,Z^t)$ over the variational joint distribution to estimate the probability:

$$Q(Z^s,Z^t) = \frac{Q(Z^s)Q(Z^t)}{\mathcal{Z}_\beta}e^{\gamma_\beta(Z^s,Z^t)}, \text{ where } \mathcal{Z}_\beta = \mathbb{E}_{Q(Z^s)Q(Z^t)}[e^{\gamma_\beta(Z^s,Z^t)}]. \tag{D5}$$

Applying this formula into Eq. (D4), we can obtain a traceable upper bound as:

$$\begin{aligned} I(Z^s,Z^t) \leq\ & \mathbb{E}_{Q(Z^s,Z^t)}[\log Q(Z^s) + \log Q(Z^t) + \gamma_\beta(Z^s,Z^t) - \log \mathcal{Z}_\beta] \\ & - \mathbb{E}_{Q(Z^s)Q(Z^t)}[\log Q(Z^s) + \log Q(Z^t) + \gamma_\beta(Z^s,Z^t) - \log \mathcal{Z}_\beta] \\ =\ & \mathbb{E}_{Q(Z^s,Z^t)}[\gamma_\beta(Z^s,Z^t)] - \mathbb{E}_{Q(Z^s)Q(Z^t)}[\gamma_\beta(Z^s,Z^t)] = I_{STUB}, \end{aligned} \tag{D6}$$

which helps us addressing the joint density calculation challenge.

E Complexity analysis and algorithm

This section presents the complexity analysis and, for completeness, also shows the pseudo-code of the algorithms for training and prediction phases.

E.1 Complexity analysis

The corresponding time complexities of the components in LaST and are shown in Table E1. The time complexity of seasonal and trend encoders/decoders are linear of fully connected network, i.e., $\mathcal{O}(Tnd)$, where n is the number of observations and d is the dimension of latent representation. For the reconstruction phrase, calculations of autocorrelation and CORT incur $\mathcal{O}(nT \log T)$ and $\mathcal{O}(nT)$ time complexities, respectively. The MI upper and lower bounds are both based on the energy function implemented by a 2-layer MLP, which has a time complexity of $\mathcal{O}(Td^2)$. As for the predictor, the time complexity of fast Fourier transform is equal to $\mathcal{O}(nT \log T)$. The first FFN in trend forecasting consumes $\mathcal{O}(nT\tau)$, while the time complexity of other FFNs becomes $\mathcal{O}(\tau nd)$. We note that only the encoders and predictor are required after training. Thus, the time complexity of training is $\mathcal{O}(T(nd + n \log T + d^2) + \tau(nT + nd))$ and that of prediction turns to $\mathcal{O}(nT\tau + nT \log T + \tau nd)$.

Table E1: Complexity analysis of main components in LaST.

Component	**Training**	**Prediction**	**Time Complexity**
seasonal-trend encoder	✓	✓	$\mathcal{O}(Tnd)$
seasonal-trend decoder	✓	✗	$\mathcal{O}(Tnd)$
reconstruction	✓	✗	$\mathcal{O}(nT \log T)$
MI estimation	✓	✗	$\mathcal{O}(Td^2)$
seasonal predictor	✓	✓	$\mathcal{O}(nT \log T + \tau nd)$
trend predictor	✓	✓	$\mathcal{O}(nT\tau + \tau nd)$

E.2 Algorithm of LaST

The pseudo-codes of LaST covering training and prediction phases are summarized in Algorithm E1 and E2, respectively.

Algorithm E1 Training phase of LaST.

Input:
Historical time series $X_{1:T}$ and future time series $X_{T+1:T+\tau}$;
Hyperparameters and initialized parameters of LaST.

Output: LaST with optimized parameters.

1: **for** a mini-batch with size $\mathcal{B}$ consisting of $\{X^{(i)}, Y^{(i)}\}_{i \in \mathcal{B}}$ **in** training set **do**
2: Encode historical time series into latent seasonal-trend representations Z^s and Z^t;
3: Reconstruct seasonal and trend $\hat{X}^s$ and $\hat{X}^t$ from representations via decoders;
4: Calculate the reconstruction loss $\mathcal{L}_{rec}$ via autocorrelation and CORT (cf. Eq. (6));
5: Estimate the mutual information bounds via Eq. (9) and Eq. (11);
6: Derive the forecasting $Y = \hat{X}_{T+1:T+\tau}$ from Z^s and Z^t via predictor;
7: Calculate the MAE loss between $\hat{X}_{T+1:T+\tau}$ and $X_{T+1:T+\tau}$;
8: Update parameters of LaST via Adam optimizer;
9: **end** *until convergence*

Algorithm E2 Prediction phase of LaST.

Input: Historical time series $X_{1:T}$ and optimized LaST;

Output: Predicted future time series $\hat{X}_{T+1:T+\tau}$.

1: Encode historical time series into latent seasonal-trend representations Z^s and Z^t;
2: Derive the forecasting $\hat{X}_{T+1:T+\tau}$ from Z^s and Z^t via predictor;

F Experiment supplementary

In this section, we present more details of the baselines, computing infrastructure, dataset generation process, and full experimental results on ETT benchmark (ETTh1, h2, m1, and m2). In addition, we provide fluctuation analysis and memory cost comparison.

F.1 Details of baselines

The implementations and settings of the seven baselines are provided below. Unless otherwise specified, we use the suggested settings described in the respective papers.

CoST [13]: analyses raw signals in time and frequency domain with temporal convolutions and Fourier transform, and separately produces seasonal and trend representations for time series forecasting task. We run experiments with their publicly available code: `https://github.com/salesforce/CoST`.

TS2Vec [14]: is a universal time series representation learning framework that performs contrastive learning in a hierarchical way over augmented context views, which enables a robust contextual representation for each timestamp. We run the code from their open source repository: `https://github.com/yuezhihan/ts2vec`.

TNC [15]: is a self-supervised framework that introduces the concept of a temporal neighborhood with stationary properties and learns generalizable time series representations. We use their open-source code: `https://github.com/sanatonek/TNC_representation_learning`. As for hyperparameters, we set $w = 0.005$ in the loss function, $mc_sample_size = 20$, batch size as 8, and learning rate as 10^{-3} with Adam optimizer.

VAE-GRU [16]: combines the strengths of recurrent network and stochastic gradient variational Bayes, which has been applied for time series forecasting among various domains [17, 18]. We implement it by the GRU [19] variant with initial state sampling from $\mathcal{N}(0, I)$ and keep the variational settings consistent with our LaST.

Autoformer [20]: designs a novel decomposition architecture with an autocorrelation mechanism to improve Transformer [21]. It exploits average pooling strategy to split time series into seasonal and trend components and regards autocorrelation coefficient as attention score, which boosts the performance of long-term forecasting. We use their publicly available code: `https://github.com/thuml/Autoformer`.

Informer [22]: designs an efficient transformer-based model for time series forecasting with a ProbSparse self-attention mechanism, highlights distilling module, and generative style decoder. We use the publicly available code: `https://github.com/zhouhaoyi/Informer2020`.

TCN [23]: introduces an architecture that is applied across all tasks for sequence modeling. It is also popular in time series modeling and forecasting [24, 25]. We use their public repository: `https://github.com/locuslab/TCN` and set kernel sizes as $\{2^1, 2^2, \cdots, 2^7\}$.

F.2 Evaluation environment

LaST is implemented by `PyTorch` and all experiments are run on a server with a `Intel Xeon Platinum 8124M` CPU, a `RTX-3090` GPU, and 128GB memory.

F.3 Synthetic dataset generation

Following the method in [13], we synthesize time series with sinusoidal seasonal patterns, as well as linear and non-linear trend patterns. The seasonal signal consists of three sine waves with the following period, phase, and amplitudes: $\{(20, 3, 0), (50, 3, 0.2), (100, 3, 0.5)\}$. The trend signal is composed of two components: (1) a nonlinear, saturating pattern $X_t = \frac{1}{1+\exp \beta_0(t-\beta_1)}$, where $\beta_0 = 0.2$ and $\beta_1 = 200$; (2) an ARMA process whose parameters of AR and MA are $\{(0.9, -0.1), (0.2, -0.5)\}$. The final time series is generated by the sum of these seasonal and trend signals.

F.4 More results on ETT benchmark

Table F1: Performance comparisons on univariate forecasting on complete ETT benchmark. Best performance is highlighted in bold font and the second best results are underlined.

Method		**LaST**		CoST		TS2Vec		TNC		VAE-GRU		Autoformer		Informer		TCN	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	**0.030**	**0.131**	0.040	0.152	0.039	0.151	0.057	0.184	0.042	0.155	0.057	0.189	0.098	0.247	0.104	0.254
	48	**0.051**	**0.169**	0.060	0.186	0.062	0.189	0.094	0.239	0.077	0.218	0.070	0.207	0.158	0.319	0.206	0.366
	168	**0.078**	**0.211**	0.097	0.236	0.142	0.291	0.171	0.329	0.172	0.344	0.108	0.260	0.183	0.346	0.462	0.586
	336	**0.100**	**0.246**	0.112	0.258	0.160	0.316	0.179	0.345	0.140	0.301	0.119	0.281	0.222	0.387	0.422	0.564
	720	0.138	0.298	0.148	0.306	0.179	0.345	0.235	0.408	0.204	0.381	**0.109**	**0.264**	0.269	0.435	0.438	0.578
ETTh2	24	**0.070**	**0.197**	0.079	0.207	0.097	0.230	0.097	0.238	0.073	0.202	0.112	0.259	0.093	0.240	0.109	0.251
	48	**0.099**	**0.239**	0.118	0.259	0.124	0.274	0.131	0.281	0.110	0.252	0.122	0.269	0.155	0.314	0.147	0.302
	168	**0.169**	**0.321**	0.189	0.339	0.198	0.355	0.197	0.354	0.179	0.340	0.178	0.331	0.232	0.389	0.209	0.366
	336	**0.202**	0.362	0.206	**0.360**	0.205	0.364	0.207	0.366	0.219	0.370	0.236	0.386	0.263	0.417	0.237	0.391
	720	0.247	0.404	0.214	0.371	0.208	0.371	0.207	0.370	0.294	0.439	0.284	0.427	0.277	0.431	**0.200**	**0.367**
ETTm1	24	**0.011**	**0.077**	0.015	0.088	0.016	0.093	0.019	0.103	0.013	0.082	0.022	0.115	0.030	0.137	0.027	0.127
	48	**0.021**	**0.108**	0.025	0.117	0.028	0.126	0.045	0.162	0.026	0.120	0.032	0.138	0.069	0.203	0.040	0.154
	96	**0.033**	**0.134**	0.038	0.147	0.045	0.162	0.054	0.178	0.046	0.164	0.045	0.168	0.194	0.372	0.097	0.246
	288	**0.069**	**0.197**	0.077	0.209	0.095	0.235	0.142	0.290	0.127	0.294	0.071	0.207	0.401	0.554	0.305	0.455
	672	**0.100**	**0.239**	0.113	0.257	0.142	0.290	0.136	0.290	0.217	0.399	0.102	0.254	0.512	0.644	0.445	0.576
ETTm2	24	**0.026**	**0.108**	0.027	0.112	0.038	0.139	0.045	0.151	0.030	0.113	0.076	0.208	0.036	0.141	0.048	0.153
	48	**0.051**	**0.157**	0.054	0.159	0.069	0.194	0.080	0.201	0.052	0.163	0.115	0.260	0.069	0.200	0.063	0.191
	96	**0.069**	**0.191**	0.072	0.196	0.089	0.225	0.094	0.229	0.073	0.200	0.091	0.230	0.095	0.240	0.129	0.265
	288	**0.119**	**0.260**	0.137	0.279	0.153	0.307	0.161	0.306	0.155	0.309	0.169	0.320	0.211	0.367	0.208	0.352
	672	**0.171**	**0.318**	0.183	0.329	0.201	0.357	0.197	0.352	0.201	0.357	0.197	0.346	0.267	0.417	0.222	0.377

Table F2: Performance comparisons on multivariate forecasting on complete ETT benchmark. Best performance is highlighted in bold font and the second best results are underlined.

Method		**LaST**		CoST		TS2Vec		TNC		VAE-GRU		AutoFormer		Informer		TCN	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	**0.324**	**0.368**	0.386	0.429	0.590	0.531	0.708	0.592	0.529	0.534	0.384	0.428	0.577	0.549	0.583	0.547
	48	**0.351**	**0.380**	0.437	0.464	0.624	0.555	0.749	0.619	0.612	0.593	0.392	0.419	0.685	0.625	0.670	0.606
	168	**0.468**	**0.453**	0.643	0.582	0.762	0.639	0.884	0.699	0.758	0.674	0.490	0.481	0.931	0.752	0.811	0.680
	336	0.566	0.512	0.812	0.679	0.931	0.728	1.020	0.768	0.844	0.692	**0.505**	**0.484**	1.128	0.873	1.132	0.815
	720	0.758	0.659	0.970	0.771	1.063	0.799	1.157	0.830	1.045	0.816	**0.498**	**0.500**	1.215	0.896	1.165	0.813
ETTh2	24	**0.175**	**0.272**	0.447	0.502	0.423	0.489	0.612	0.595	0.267	0.364	0.261	0.341	0.720	0.665	0.935	0.754
	48	**0.229**	**0.312**	0.699	0.637	0.619	0.605	0.840	0.716	0.523	0.529	0.312	0.373	1.457	1.001	1.300	0.911
	168	0.722	0.609	1.549	0.982	1.845	1.074	2.359	1.213	2.519	1.301	**0.457**	**0.455**	3.489	1.515	4.017	1.579
	336	1.261	0.828	1.749	1.042	2.194	1.197	2.782	1.349	3.589	1.577	**0.471**	**0.475**	2.723	1.340	3.460	1.456
	720	1.780	1.094	1.971	1.092	2.636	1.370	2.753	1.394	3.788	1.683	**0.474**	**0.484**	3.467	1.473	3.106	1.381
ETTm1	24	**0.218**	**0.289**	0.246	0.329	0.453	0.444	0.522	0.472	0.509	0.534	0.383	0.403	0.453	0.444	0.522	0.472
	48	**0.280**	**0.329**	0.331	0.386	0.592	0.521	0.695	0.567	0.642	0.543	0.454	0.453	0.494	0.503	0.542	0.508
	96	**0.323**	**0.360**	0.378	0.419	0.635	0.554	0.731	0.595	0.600	0.540	0.481	0.463	0.678	0.614	0.666	0.578
	288	**0.392**	**0.403**	0.472	0.486	0.693	0.597	0.818	0.649	0.769	0.678	0.634	0.528	1.056	0.786	0.991	0.735
	672	**0.491**	**0.466**	0.620	0.574	0.782	0.653	0.932	0.712	0.799	0.673	0.606	0.542	1.192	0.926	1.032	0.756
ETTm2	24	**0.102**	**0.206**	0.141	0.282	0.179	0.296	0.185	0.297	0.178	0.306	0.153	0.261	0.173	0.301	0.180	0.324
	48	**0.135**	**0.237**	0.209	0.347	0.243	0.352	0.264	0.360	0.214	0.332	0.178	0.280	0.303	0.409	0.204	0.327
	96	**0.182**	**0.274**	0.325	0.436	0.336	0.415	0.389	0.458	0.279	0.375	0.255	0.339	0.365	0.453	3.041	1.330
	288	**0.299**	**0.360**	0.816	0.698	0.707	0.632	0.920	0.788	0.809	0.691	0.342	0.378	1.047	0.804	3.162	1.337
	672	0.790	0.611	1.633	1.025	1.801	1.022	2.164	1.135	1.838	1.038	**0.434**	**0.430**	3.126	1.302	3.624	1.484

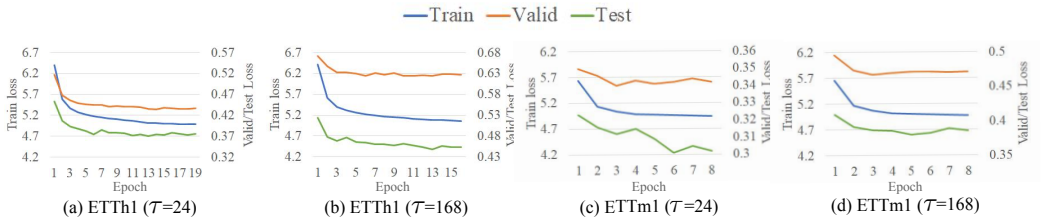
F.5 Fluctuation analysis

To validate the models' robustness, we conduct fluctuation analysis and have three runs of all the experiments with different seeds for baselines and LaST. Table F3 reports the performance results in terms of MSE and MAE with standard deviations.

Table F3: Standard deviations results on multivariate forecasting.

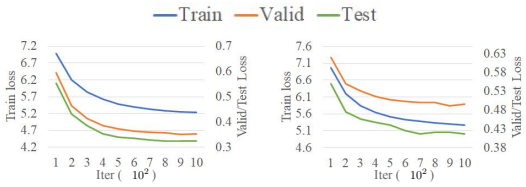
Method		**LaST**		CoST		VAE-GRU		AutoFormer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	**0.324** $\pm$0.003	**0.368** $\pm$0.004	0.386 $\pm$0.006	0.429 $\pm$0.005	0.529 $\pm$0.016	0.531 $\pm$0.013	0.384 $\pm$0.020	0.428 $\pm$0.017
	48	**0.351** $\pm$0.002	**0.380** $\pm$0.003	0.437 $\pm$0.003	0.464 $\pm$0.006	0.612 $\pm$0.019	0.593 $\pm$0.010	0.392 $\pm$0.020	0.419 $\pm$0.026
	168	**0.468** $\pm$0.004	**0.453** $\pm$0.004	0.643 $\pm$0.018	0.582 $\pm$0.011	0.758 $\pm$0.035	0.647 $\pm$0.031	0.490 $\pm$0.016	0.481 $\pm$0.009
	336	0.566 $\pm$0.002	0.512 $\pm$0.003	0.812 $\pm$0.003	0.679 $\pm$0.008	0.844 $\pm$0.030	0.692 $\pm$0.028	**0.505** $\pm$0.013	**0.484** $\pm$0.015
	720	0.758 $\pm$0.001	0.659 $\pm$0.001	0.970 $\pm$0.030	0.771 $\pm$0.006	1.045 $\pm$0.014	0.816 $\pm$0.015	**0.498** $\pm$0.011	**0.500** $\pm$0.012
ETTm1	24	**0.218** $\pm$0.002	**0.289** $\pm$0.001	0.246 $\pm$0.018	0.329 $\pm$0.010	0.509 $\pm$0.023	0.402 $\pm$0.027	0.383 $\pm$0.030	0.403 $\pm$0.035
	48	**0.280** $\pm$0.003	**0.329** $\pm$0.005	0.331 $\pm$0.024	0.386 $\pm$0.008	0.642 $\pm$0.026	0.543 $\pm$0.028	0.454 $\pm$0.029	0.453 $\pm$0.038
	96	**0.323** $\pm$0.001	**0.360** $\pm$0.001	0.378 $\pm$0.023	0.419 $\pm$0.012	0.600 $\pm$0.022	0.540 $\pm$0.020	0.481 $\pm$0.018	0.463 $\pm$0.020
	288	**0.392** $\pm$0.003	**0.403** $\pm$0.027	0.472 $\pm$0.011	0.486 $\pm$0.007	0.769 $\pm$0.038	0.678 $\pm$0.019	0.634 $\pm$0.024	0.528 $\pm$0.016
	672	**0.491** $\pm$0.006	**0.466** $\pm$0.002	0.620 $\pm$0.029	0.574 $\pm$0.010	0.799 $\pm$0.008	0.673 $\pm$0.010	0.606 $\pm$0.010	0.542 $\pm$0.014
Electricity	24	**0.125** $\pm$0.002	**0.222** $\pm$0.003	0.136 $\pm$0.006	0.242 $\pm$0.005	0.190 $\pm$0.004	0.250 $\pm$0.002	0.165 $\pm$0.002	0.286 $\pm$0.004
	48	**0.146** $\pm$0.001	**0.245** $\pm$0.001	0.153 $\pm$0.002	0.258 $\pm$0.003	0.228 $\pm$0.003	0.280 $\pm$0.005	0.178 $\pm$0.007	0.295 $\pm$0.010
	168	**0.170** $\pm$0.002	**0.265** $\pm$0.003	0.175 $\pm$0.005	0.275 $\pm$0.008	0.240 $\pm$0.005	0.297 $\pm$0.009	0.215 $\pm$0.003	0.327 $\pm$0.004
	336	**0.188** $\pm$0.006	**0.280** $\pm$0.004	0.196 $\pm$0.005	0.296 $\pm$0.005	0.262 $\pm$0.007	0.318 $\pm$0.007	0.218 $\pm$0.006	0.329 $\pm$0.004
	720	**0.223** $\pm$0.003	**0.309** $\pm$0.002	0.232 $\pm$0.008	0.327 $\pm$0.003	0.296 $\pm$0.004	0.347 $\pm$0.003	0.252 $\pm$0.007	0.356 $\pm$0.008
Exchange	24	**0.033** $\pm$0.001	**0.122** $\pm$0.001	0.033 $\pm$0.003	0.127 $\pm$0.009	0.064 $\pm$0.004	0.178 $\pm$0.005	0.060 $\pm$0.008	0.178 $\pm$0.007
	48	**0.056** $\pm$0.001	**0.162** $\pm$0.002	0.058 $\pm$0.002	0.165 $\pm$0.004	0.133 $\pm$0.007	0.262 $\pm$0.010	0.091 $\pm$0.010	0.222 $\pm$0.009
	168	**0.190** $\pm$0.009	**0.320** $\pm$0.003	0.198 $\pm$0.015	0.327 $\pm$0.008	0.334 $\pm$0.013	0.432 $\pm$0.012	0.405 $\pm$0.017	0.473 $\pm$0.013
	336	**0.430** $\pm$0.018	**0.482** $\pm$0.016	0.512 $\pm$0.026	0.523 $\pm$0.013	0.614 $\pm$0.030	0.606 $\pm$0.024	0.509 $\pm$0.041	0.524 $\pm$0.016
	720	1.521 $\pm$0.162	**0.898** $\pm$0.105	1.855 $\pm$0.103	0.998 $\pm$0.037	2.285 $\pm$0.078	1.117 $\pm$0.054	**1.447** $\pm$0.084	0.941 $\pm$0.028
Weather	24	**0.105** $\pm$0.005	**0.134** $\pm$0.004	0.293 $\pm$0.013	0.369 $\pm$0.009	0.117 $\pm$0.004	0.147 $\pm$0.002	0.180 $\pm$0.015	0.263 $\pm$0.018
	48	**0.131** $\pm$0.006	**0.174** $\pm$0.006	0.558 $\pm$0.020	0.515 $\pm$0.014	0.227 $\pm$0.008	0.270 $\pm$0.003	0.241 $\pm$0.010	0.310 $\pm$0.012
	168	**0.197** $\pm$0.010	**0.238** $\pm$0.011	0.812 $\pm$0.033	0.671 $\pm$0.026	0.234 $\pm$0.007	0.280 $\pm$0.005	0.295 $\pm$0.027	0.355 $\pm$0.023
	336	**0.257** $\pm$0.013	**0.285** $\pm$0.014	1.196 $\pm$0.024	0.832 $\pm$0.022	0.309 $\pm$0.022	0.339 $\pm$0.015	0.359 $\pm$0.035	0.395 $\pm$0.031
	720	**0.315** $\pm$0.003	**0.327** $\pm$0.009	1.620 $\pm$0.036	1.002 $\pm$0.020	0.444 $\pm$0.037	0.410 $\pm$0.024	0.419 $\pm$0.017	0.428 $\pm$0.014

F.6 Convergence



(a) ETTh1 ($\mathcal{T}$=24) (b) ETTh1 ($\mathcal{T}$=168) (c) ETTm1 ($\mathcal{T}$=24) (d) ETTm1 ($\mathcal{T}$=168)

Figure F1: Train, valid, and test losses changes in the training process.



(a) ETTm1 ($\mathcal{T}$=24) (b) ETTm1 ($\mathcal{T}$=168)

Figure F2: Train, valid, and test losses changes at the first epoch.

We conduct additional experiments to validate the convergence property of our model. Figure F1 shows the descending process of training, valid, and test loss of our model as the epochs increase, where we can see that all losses first drop and then goes to level off. The downward trend is not obvious on ETTm1 dataset. This is because ETTm1 is a larger-scale dataset and it requires over 1,000 times stochastic gradient descent at every epoch. We visualize the losses change at the first

epoch in figure F2. We can observe that losses drop a lot at the first epoch. These results support that our model converges well on real-world datasets.

F.7 Memory cost

We keep the batch size $\mathcal{B}$, input length T and forecasting length τ consistent among baselines and LaST, and report the memory cost of in Table F4, where we can observe that LaST is $\sim$30% more memory-efficient than CoST and over 40% more memory-efficient than AutoFormer. CoST contains multiple TCNs with different kernel sizes, which enable it to collect multi-horizon information, but lead the memory consumption to multiply. Transformer-based Autoformer establishes the correlations between every time step, which captures the long-term dependencies but incurs $\mathcal{O}((T+\tau)^2)$ space complexity. For example, AutoFormer on Weather dataset with $\{\mathcal{B}=32, T=201, \tau=720\}$ allocates 9045MB memory while LaST only needs 1751MB of memory. Based on variational inference theory, our proposed LaST model regards the time series as a whole and extracts seasonal and trend without additional memory consumption.

Table F4: Memory usage (MB) comparisons among different methods.

Batch size		16						32					
Input length		96			201			96			201		
Output length		48	168	720	48	168	720	48	168	720	48	168	720
LaST	ETTh1	1071	1073	1149	1127	1139	1383	1133	1175	1317	1245	1403	1741
	Weather	1075	1081	1175	1131	1167	1395	1139	1187	1337	1255	1407	1751
CoST	ETTh1	1597	1597	1597	1909	1909	1909	1759	1759	1759	1977	1977	1977
	Weather	1598	1598	1598	1911	1911	1911	1791	1761	1761	1981	1981	1981
AutoFormer	ETTh1	1831	2501	4371	2209	2857	4771	2433	3477	8213	3263	4213	9035
	Weather	1833	2505	4381	2211	2859	4777	2437	3481	8229	3265	4219	9045

References

[1] Michail Vlachos, Philip Yu, and Vittorio Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 449–460. SIAM, 2005.

[2] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[3] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pages 531–540. PMLR, 2018.

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*, 2015.

[5] Harry Nyquist. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, 47(2):617–644, 1928.

[6] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[7] Norbert Wiener, Norbert Wiener, Cyberneticist Mathematician, Norbert Wiener, Norbert Wiener, and Cybernéticien Mathématicien. *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*, volume 113. MIT press Cambridge, MA, 1949.

[8] Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223. PMLR, 2018.

[9] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

[10] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

[11] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. Club: A contrastive log-ratio upper bound of mutual information. In *International conference on machine learning*, pages 1779–1788. PMLR, 2020.

[12] Malik Magdon-Ismail and Amir Atiya. Neural networks for density estimation. In *Advances in Neural Information Processing Systems*, volume 11, 1998.

[13] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International Conference on Learning Representations*, 2022.

[14] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of AAAI*, 2022.

[15] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations*, 2021.

[16] Otto Fabius, Joost R van Amersfoort, and Diederik P Kingma. Variational recurrent auto-encoders. In *ICLR (Workshop)*, 2015.

[17] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *Proceedings of the IEEE international conference on computer vision*, pages 3332–3341, 2017.

[18] Abdelhafid Zeroual, Fouzi Harrou, Abdelkader Dairi, and Ying Sun. Deep learning methods for forecasting covid-19 time-series data: A comparative study. *Chaos, Solitons & Fractals*, 140:110121, 2020.

[19] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[20] Jiehui Xu, Jianmin Wang, Mingsheng Long, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34, 2021.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[22] 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 AAAI*, 2021.

[23] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[24] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *Advances in neural information processing systems*, 32, 2019.

[25] Shumin Deng, Ningyu Zhang, Wen Zhang, Jiaoyan Chen, Jeff Z Pan, and Huajun Chen. Knowledge-driven stock trend prediction and explanation via temporal convolutional network. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 678–685, 2019.