A Proof of Proposition 1

Given two Gaussian distributions $p(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $p(\hat{\boldsymbol{x}}) \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{L})$, the KL-divergence between these two distributions, i.e. $D_{KL}(p(\boldsymbol{x})||p(\hat{\boldsymbol{x}})) = \mathbb{E}_{p(\boldsymbol{x})}\left(\ln \frac{p(\boldsymbol{x})}{p(\hat{\boldsymbol{x}})}\right)$, can be simplified as follows:

$$
\begin{aligned}
&D_{KL}(p(\boldsymbol{x})||p(\hat{\boldsymbol{x}})) = \mathbb{E}_{p(\boldsymbol{x})}\left(\ln \frac{p(\boldsymbol{x})}{p(\hat{\boldsymbol{x}})}\right) \\
=&\frac{1}{2}\left\{\ln \frac{|\boldsymbol{L}|}{|\boldsymbol{\Sigma}|} + Tr(\boldsymbol{L}^{-1}\boldsymbol{\Sigma}) + (\boldsymbol{\mu}-\boldsymbol{m})^T \boldsymbol{L}^{-1}(\boldsymbol{\mu}-\boldsymbol{m}) - d\right\}
\end{aligned} \tag{15}
$$

In this problem, from the same origin $\boldsymbol{x}_0$, $\boldsymbol{x}_L$ and $\hat{\boldsymbol{x}}_L$ both obey Gaussian distribution, as follows:

$$
\begin{aligned}
\boldsymbol{x}_L &\sim \mathcal{N}\left(\boldsymbol{\Phi}^L \boldsymbol{x}_0, \sum_{m=0}^{L-1} \boldsymbol{\Phi}^{2m}\right) \\
\hat{\boldsymbol{x}}_L &\sim \mathcal{N}\left(\hat{\boldsymbol{\Phi}}^L \boldsymbol{x}_0, \sum_{m=0}^{L-1} \hat{\boldsymbol{\Phi}}^{2m}\right)
\end{aligned} \tag{16}
$$

We can calculate the KL-divergence of these two Gaussian distributions as follows:

$$
\begin{aligned}
&D_{KL}(p(\boldsymbol{x}_L|\boldsymbol{x}_0)||p(\hat{\boldsymbol{x}}_L|\boldsymbol{x}_0)) \\
=&\frac{1}{2}\left\{\ln \frac{|\sum_{m=0}^{L-1}\hat{\boldsymbol{\Phi}}^{2m}|}{|\sum_{m=0}^{L-1}\boldsymbol{\Phi}^{2m}|} + Tr\left\{\left[\sum_{m=0}^{L-1}\hat{\boldsymbol{\Phi}}^{2m}\right]^{-1}\left[\sum_{m=0}^{L-1}\boldsymbol{\Phi}^{2m}\right]\right\}\right. \\
&\left. +(\boldsymbol{\Phi}^L\boldsymbol{x}_0 - \hat{\boldsymbol{\Phi}}^L\boldsymbol{x}_0)^T\left[\sum_{m=0}^{L-1}\hat{\boldsymbol{\Phi}}^{2m}\right]^{-1}(\boldsymbol{\Phi}^L\boldsymbol{x}_0 - \hat{\boldsymbol{\Phi}}^L\boldsymbol{x}_0) - d\right\}
\end{aligned} \tag{17}
$$

As $\boldsymbol{\Phi}, \hat{\boldsymbol{\Phi}}$ are symmetric, assume they can be decomposed into $\boldsymbol{\Phi} = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}^T$ and $\hat{\boldsymbol{\Phi}} = \boldsymbol{P}\hat{\Lambda}\boldsymbol{P}^T$. In addition, $diag(\boldsymbol{\Lambda}) = (\lambda_1, ..., \lambda_d)$ and $diag(\hat{\boldsymbol{\Lambda}}) = (\hat{\lambda}_1, ..., \hat{\lambda}_d)$. First, we simplify $|\sum_{m=0}^{L-1}\boldsymbol{\Phi}^{2m}|$ as follows:

$$
\begin{aligned}
\left|\sum_{m=0}^{L-1}\boldsymbol{\Phi}^m(\boldsymbol{\Phi}^T)^m\right| &= \left|\sum_{m=0}^{L-1} P\boldsymbol{\Lambda}^m P^T P \boldsymbol{\Lambda}^m P^T\right| \\
&= \left|\sum_{m=0}^{L-1} P\boldsymbol{\Lambda}^{2m}P^T\right| \\
&= \left|P\left(\sum_{m=0}^{L-1}\boldsymbol{\Lambda}^{2m}\right)P^T\right| \\
&= \prod_{i=1}^{d}\sum_{m=0}^{L-1}\lambda_i^{2m}.
\end{aligned} \tag{18}
$$

For the term $\left[\sum_{m=0}^{L-1}\hat{\boldsymbol{\Phi}}^{2m}\right]^{-1}$, we have the following:

$$
\begin{aligned}
\left[\sum_{m=0}^{L-1}\hat{\boldsymbol{\Phi}}^{2m}\right]^{-1} &= \left[P\left(\sum_{m=0}^{L-1}\hat{\boldsymbol{\Lambda}}^{2m}\right)P^T\right]^{-1} \\
&= \left[P\left(\sum_{m=0}^{L-1}\hat{\boldsymbol{\Lambda}}^{2m}\right)^{-1}P^T\right].
\end{aligned} \tag{19}
$$

Then we can simplify the Eq. (17) as follows:

$$
\begin{aligned}
&D_{KL}(p(\boldsymbol{x}_L|\boldsymbol{x}_0)||p(\hat{\boldsymbol{x}}_L|\boldsymbol{x}_0)) \\
=&\frac{1}{2}\left\{\ln \frac{\prod_{i=1}^{d}\sum_{m=0}^{L-1}\hat{\lambda}_i^{2m}}{\prod_{i=1}^{d}\sum_{m=0}^{L-1}\lambda_i^{2m}} + \sum_{i=1}^{d}\frac{\sum_{m=0}^{L-1}\lambda_i^{2m}}{\sum_{m=0}^{L-1}\hat{\lambda}_i^{2m}}\right. \\
&\left. + \sum_{i=1}^{d}\frac{(\lambda_i^L - \hat{\lambda}_i^L)^2}{\sum_{m=0}^{L-1}\hat{\lambda}_i^{2m}}(x_0^i)^2 - d\right\} \\
\geq&\frac{1}{2}\left\{\ln \frac{\prod_{i=1}^{d}\sum_{m=0}^{L-1}\hat{\lambda}_i^{2m}}{\prod_{i=1}^{d}\sum_{m=0}^{L-1}\lambda_i^{2m}} + \sum_{i=1}^{d}\frac{\sum_{m=0}^{L-1}\lambda_i^{2m}}{\sum_{m=0}^{L-1}\hat{\lambda}_i^{2m}} - d\right\}
\end{aligned} \tag{20}
$$

Denote the maximum eigenvalue as $\lambda = \max\{\lambda_i\}$, $\hat{\lambda} = \max\{\hat{\lambda}_i\}$. And they differ by a small quantity δ, i.e., $|1 - \hat{\lambda}/\lambda| = \delta$. As the function $f(x) = \ln x + \frac{1}{x} - 1 \geq 0$, we can transform the above inequality, leaving only terms of maximum eigenvalue:

$$
\begin{aligned}
&D_{KL}(p(\boldsymbol{x}_L|\boldsymbol{x}_0)||p(\hat{\boldsymbol{x}}_L|\boldsymbol{x}_0)) \\
\geq&\frac{1}{2}\left\{\ln \frac{\sum_{m=0}^{L-1}\hat{\lambda}^{2m}}{\sum_{m=0}^{L-1}\lambda^{2m}} + \frac{\sum_{m=0}^{L-1}\lambda^{2m}}{\sum_{m=0}^{L-1}\hat{\lambda}^{2m}} - 1\right\}.
\end{aligned} \tag{21}
$$

Denote $\frac{\sum_{m=0}^{L-1}\hat{\lambda}^{2m}}{\sum_{m=0}^{L-1}\lambda^{2m}}$ as a function of L, i.e., $h(L)$, where $L \in \mathbb{Z}_+$. So we get

$$
D_{KL}(p(\boldsymbol{x}_L|\boldsymbol{x}_0)||p(\hat{\boldsymbol{x}}_L|\boldsymbol{x}_0)) \geq \frac{1}{2}\left\{\ln h(L) + \frac{1}{h(L)} - 1\right\}. \tag{22}
$$

If $|\lambda_i| > |\hat{\lambda}_i|$, $h(L)$ is monotonically decreasing and $h(L) < 1$. In addition to that $f(x)$ is monotonically decreasing under where $x < 1$, so the above lower bound is monotonically increasing with respect to L. If $|\lambda_i| < |\hat{\lambda}_i|$, we can get the similar result that above lower bound is monotonically increasing. Till now, we get the first conclusion:

- The lower bound of the KL-divergence is monotonically increasing as the generated length L increases.

If $\lambda > 1$, for a small quantity δ, we can also assume $\hat{\lambda}$ is close to λ and $\hat{\lambda} > 1$. Since the generated length L can be arbitrarily large as the generation proceeds, then $h(L) = (1 \pm \delta)^{2L}$ when L is large. For a small quantity δ and a large generation length L, then we have get the following:

$$
\begin{aligned}
\ln(1+\delta)^{2L} + (1+\delta)^{-2L} &= \Theta(\ln(1+\delta)^{2L}) = \Theta(\delta L) \\
\ln(1-\delta)^{2L} + (1-\delta)^{-2L} &= \Theta((1-\delta)^{-2L}) = \Theta(c^L),
\end{aligned} \tag{23}
$$

where $c = (1-\delta)^{-2} > 1$. As for the case when $\lambda = 1$, we can get similar results. So we get the second conclusion:

- $D_{KL}(p(\boldsymbol{x}_L|\boldsymbol{x}_0)||p(\hat{\boldsymbol{x}}_L|\boldsymbol{x}_0)) = \Omega(\delta L)$, when $\lambda \geq 1$.

For the case when $\lambda < 1$, the function $h(L)$ will converge to a certain constant when L is large so that the KL-divergence also converges to a certain constant. But we can still apply the first conclusion that the distribution shift is more severe as the generation proceeds.

B Experimental Details

B.1 Datasets Information

ETT (Zhou et al. 2021) datasets include the power load and the oil temperature data, which are collected from the electricity transformer and used for the long-term deployment of electric power. They consist of two granularities: 15-minute and 1-hour. **US Births** (Godahewa et al. 2021) records the daily number of births in the US from 1969/01/01 to 1988/12/31. **ILI** (Fluview 2022) contains the weekly records of the patients who suffer from influenza-like illness (ILI) from 2002/01/01 to 2020/06/30. We list the characteristics of the six datasets in Tab. 4. All these datasets are split into training (80%) and testing (20%) sets in chronological order, and the times-series GANs are trained on the training set.

Table 4: Characteristics of the datasets

Dataset	Length	Attributes	Granularity
ETTh1	17420	7	1hour
ETTh2	17420	7	1hour
ETTm1	69680	7	15mins
ETTm2	69680	7	15mins
US Births	7305	1	1day
ILI	966	7	1week

B.2 Baselines Descriptions

In this work, we have compared the quality of synthetic data with the following four GANs for time-series generation:

- **QuantGAN** (Wiese et al. 2020) utilizes the temporal convolutional networks (TCNs) to capture distributional properties and dependence properties in high fidelity.
- **TimeGAN** (Yoon, Jarrett, and Van der Schaar 2019) learns an embedding space that is additionally optimized with the supervised temporal correlations.
- **Cot-GAN** (Xu et al. 2020) formulates a fancy loss function inspired by Causal Optimal Transport (COT).
- **SigCWGAN** (Ni et al. 2020) integrates GANs with the signature of a path, and the explicit representation can relieve the training burden.

Among these time-series GANs, QuantGAN mainly focuses on the generation of financial time-series, considering the volatility clusters and leverage effects in the typical financial time-series data. However, QuantGAN does not perform well in the general time-series datasets, as shown in our experiments. SigCWGAN is a conditional GAN for financial time-series generation, and it regards the past sequence as the conditional variable.

As the generated data can be used as augmentation data to facilitate the downstream tasks, we also implement several augmentation methods to compare their performance on the downstream forecasting task.

- **Scaling** (Um et al. 2017) changes the scale of the original time series by multiplying a random scalar.
- **Jitter** (Um et al. 2017) adds an additive Gaussian noise to the original time series.

Algorithm 2: AEC-GAN Training Algorithm

Input: Training set $\boldsymbol{S} = \{x_t \in \mathbb{R}^d\}_{1 \le t \le T}$, l_2 radius δ_{max}, learning rate $\eta_\theta, \eta_\phi, \eta_\omega$.
Output: $G_\theta, D_\phi, M_\omega$.
while *not converged* **do**
  for *D_step = 1 ... MAX_D_STEP* **do**
    $\boldsymbol{c}_t, \boldsymbol{x}_\tau \leftarrow \text{Sample}(\boldsymbol{S})$
    $\boldsymbol{z} \leftarrow \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
    Obtain $\boldsymbol{\delta}$ according to Eq. (3)
    Generate $\{\hat{\boldsymbol{x}}_\tau^i\}_{i \in [4]}$ according to Eq. (9)
    $g_\phi \leftarrow \nabla_\phi \mathcal{L}_D^{Aug}$
    $\phi \leftarrow \phi + \eta_\phi \cdot g_\phi$
  $\boldsymbol{c}_t, \boldsymbol{x}_\tau \leftarrow \text{Sample}(\boldsymbol{X}_t)$
  $\boldsymbol{z} \leftarrow \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
  Obtain $\boldsymbol{\delta}$ according to Eq. (3)
  Generate $\{\hat{\boldsymbol{x}}_\tau^i\}_{i \in [4]}$ according to Eq. (9)
  $g_\theta \leftarrow \nabla_\theta \mathcal{L}_G^{Aug}; \theta \leftarrow \theta - \eta_\theta \cdot g_\theta$
  $g_\omega \leftarrow \nabla_\omega \mathcal{L}_M; \omega \leftarrow \omega - \eta_\omega \cdot g_\omega$
Return: $G_\theta, D_\phi, M_\omega$.

- **Mixup** (Zhang et al. 2017) makes a convex combination of pairs of examples and their labels.

In order to qualify how much the generated data can boost the performance of the downstream time-series forecasting models, we deploy the following three well-known time-series forecasting methods as our experimental backbones. All experimental settings are kept the same as their corresponding setups.

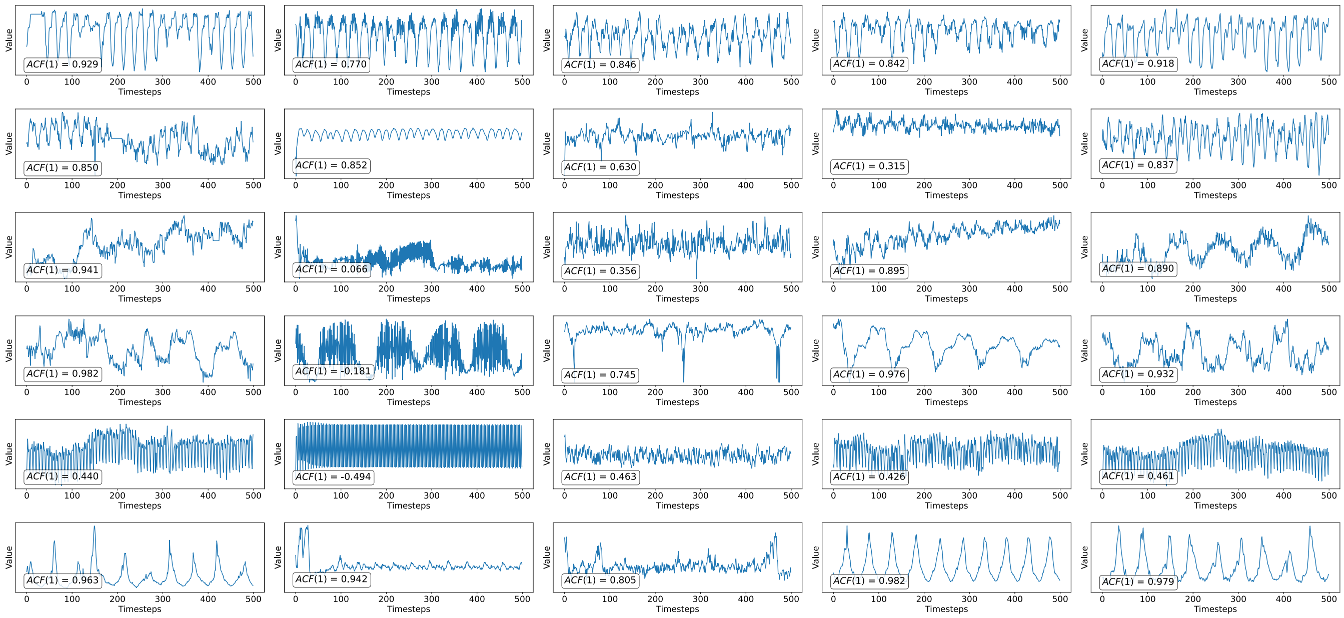
- **SCINet** (Liu et al. 2021) proposes sample convolution and interaction for temporal modeling, which performs well in forecasting tasks.
- **Informer** (Zhou et al. 2021) proposes a novel self-attention mechanism, which is more efficient and works well for long sequence time-series forecasting (LSTF).
- **Autoformer** (Wu et al. 2021) alternates the pre-processing convention with an Auto-Correlation mechanism which outperforms self-attention in both efficiency and accuracy.

Among these three time-series forecasting models, there are some differences between the input and forecasting horizons. Take the forecasting horizon 168 in ETTh1 for example, the input horizon for SCINet is 336, but the input horizon for the transformer-based methods is 96. Moreover, the transformer-based methods only predict the last 48 steps of the target 168 steps in the future. Thus, the performances of SCINet and the transformer-based methods are not comparable essentially. However, we keep their original experimental settings and only compare the performance differences of the generated data with these time-series forecasting models.

B.3 Experimental Setups

The implementations of the baselines and corresponding setups can be found in the following repositories:

- **QuantGAN**: https://github.com/KseniaKingsep/quantgan
- **TimeGAN**: https://github.com/jsyoon0823/TimeGAN



(a) Realistic Data (b) TimeGAN (c) Cot-GAN (d) SigCWGAN (e) AEC-GAN

Figure 8: Generation examples of length 500 on six datasets. Each row represents a dataset (ETTh1, ETTh2, ETTm1, ETTm2, US Births, ILI). **(a)** reports the real data samples. **(b-e)** are generated samples. Each figure is annotated with the $ACF(1)$ of each sequence.

- **Cot-GAN**: https://github.com/tianlinxu312/cot-gan
- **SigCWGAN**: https://github.com/SigCGANs/Conditional-Sig-Wasserstein-GANs
- **Scaling**: https://github.com/terryum/Data-Augmentation-For-Wearable-Sensor-Data
- **Jitter**: https://github.com/terryum/Data-Augmentation-For-Wearable-Sensor-Data
- **Mixup**: https://github.com/facebookresearch/mixup-cifar10
- **SCINet**: https://github.com/cure-lab/SCINet
- **Autoformer**: https://github.com/thuml/Autoformer
- **Informer**: https://github.com/thuml/Autoformer

For the time-series GANs, we only change the generation length of each model and keep other parameters unchanged. All the GAN models are trained with 10,000 iterations for a fair comparison.

For the time-series forecasting models, the input and forecasting horizons are different between SCINet and transformer-based methods, and we keep their corresponding settings, respectively. In addition, the transformer-based methods exploit the date information for temporal embedding. However, the deep generative models always ignore the date information, and we remove the temporal embedding in all experiments for a fair comparison.

To evaluate the quality of the generated time-series data in the downstream tasks, we generate a fake dataset to replace the original training set. As different experiments expect different lengths of time-series data, we utilize the GAN models to generate the desirable length of time-series data to fit in the corresponding setting in the forecasting tasks. For instance, SCINet forecasts forward 168 steps given an input sequence with 336 steps, so we generate a fake dataset containing sequences with $504 = 168+336$ timesteps. To maintain the same dataset size, we generate the same amount of such sequences as the number of slices of the original training set sliced with sliding window 504.

B.4 Pseudocode of Training

The detailed training procedure of our AEC-GAN is shown in Algorithm 2. The three modules, G_θ, D_ϕ and M_ω, are jointly optimized. In addition, we adopt the two time-scale update rule (TTUR (Heusel et al. 2017)), which updates the discriminator D_ϕ twice often as the generator G_θ.

B.5 Implementation Details

We implement the generator G_θ, D_ϕ by the conditional AR-FNN (Ni et al. 2020) and M_ω by four layers of one-dimensional convolutional neural networks (Conv1d). The illustration of the implementation is shown in Figure 2. The discriminator is trained twice during each training iteration. The learning parameters are shown in Table 5. The hidden dimension of all models on six datasets is 50, and all the models have trained 10,000 iterations with a batch size of 200.

Table 5: Learning parameters.

Module	Optimizer	Learning rate	Betas
G_θ	Adam	1e-4	(0, 0.9)
D_ϕ	Adam	2e-4	(0, 0.9)
M_ω	Adam	1e-3	(0.9, 0.999)

Table 6: The MSE and MAE errors of our AEC-GAN performs on **SCINet** with different synthetic data size. AEC-GAN**xN** denotes that we use **N** times the original training data. The metric is the lower the better. Best results are shown in **bold**. Each dataset has experimented with three different forecasting horizons.

		ETTh1			ETTh2			ETTm1			ETTm2			US Births			ILI		
Methods	Metric	168	336	720	168	336	720	96	288	672	96	288	672	168	336	720	36	48	60
Original	mse	0.450	0.528	0.597	0.554	0.657	1.118	0.197	0.350	1.214	0.330	**0.383**	0.501	0.376	0.737	1.140	1.400	1.401	1.452
	mae	0.453	0.513	0.571	0.517	0.576	0.776	0.294	0.405	0.836	0.377	0.408	0.490	0.268	0.865	1.775	4.089	4.076	4.216
AEC-GAN	mse	**0.403**	0.498	0.501	0.393	0.390	0.495	**0.190**	0.306	0.376	0.315	0.416	0.490	0.231	0.611	0.567	3.498	3.531	3.598
	mae	**0.432**	0.496	0.504	0.420	0.428	0.510	**0.279**	0.354	0.395	0.345	0.455	0.429	0.345	0.602	0.581	1.278	1.297	1.329
AEC-GANx2	mse	0.411	0.434	0.427	0.390	0.364	0.414	0.191	**0.305**	**0.363**	**0.304**	0.402	0.485	**0.230**	0.435	0.413	2.597	2.547	2.619
	mae	0.436	0.450	0.451	0.420	**0.413**	0.455	**0.279**	**0.353**	**0.390**	**0.341**	0.394	0.444	**0.343**	0.497	0.486	1.062	1.064	1.101
AEC-GANx5	mse	0.409	**0.387**	**0.398**	**0.385**	**0.354**	**0.387**	**0.190**	0.307	0.365	0.310	0.400	0.468	0.253	**0.276**	**0.286**	**2.204**	**2.167**	**2.235**
	mae	0.434	**0.415**	**0.427**	**0.416**	**0.413**	**0.432**	0.281	0.356	**0.390**	0.343	**0.393**	0.440	0.368	**0.375**	**0.386**	**0.957**	**0.956**	**0.994**

As for the adversarial attacks, we conduct 10 steps PGD-attacks. Conditioning on previous p steps and generating forward q steps, we set the maximum l_2 radius $\delta_{max} = \sqrt{(p+q)*c^2}$, where c is a small constant to limit the perturbation scale, and we let $c = 0.2$ across all experiments.

To evaluate the generation quality, we conduct experiments on six datasets and quantitatively assess the generation quality. In the training process, we condition the GAN on previous p steps to generate q steps forward. We let $p/q = 168/336$ for ETTh* and US Birth, $p/q = 96/192$ for ETTm*, $p/q = 18/36$ for ILI.

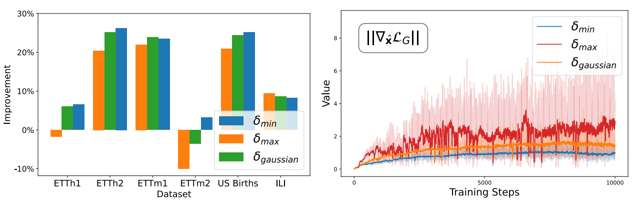
C Additional Experiments

C.1 Perturbation Analysis

We studied whether our adversarial attacks benefit generation. Besides the PGD attack perturbation $\boldsymbol{\delta}_{max}$ and $\boldsymbol{\delta}_{min}$ shown in Eq. (24),

$$\begin{aligned}\boldsymbol{\delta}_{min} &= \underset{||\boldsymbol{\delta}||_2 \le \delta_{max}}{\arg\min} \ \log\left[1 - D_\phi(G_\theta(\boldsymbol{z}|\boldsymbol{c}_t), \boldsymbol{c}_t + \boldsymbol{\delta})\right] \\ \boldsymbol{\delta}_{max} &= \underset{||\boldsymbol{\delta}||_2 \le \delta_{max}}{\arg\max} \ \log\left[1 - D_\phi(G_\theta(\boldsymbol{z}|\boldsymbol{c}_t), \boldsymbol{c}_t + \boldsymbol{\delta})\right],\end{aligned} \tag{24}$$

we also conduct experiments with the Gaussian perturbation $\boldsymbol{\delta}_{gaussian} = \mathcal{N}(\mathbf{0}, \sqrt{\delta_{max}/pd}\boldsymbol{I})$, maintaining the same expected l_2 radius δ_{max}. We also respectively run the downstream task training on the synthetic data. The average promotion to six datasets is shown in Fig. 9(a). It can be seen that $\boldsymbol{\delta}_{min}$ performs the best and $\boldsymbol{\delta}_{max}$ performs the worst generally. For a more intuitive explanation, we record the $\|\nabla_{\hat{\boldsymbol{x}}}\mathcal{L}_G\|$ during the training on ILI in Fig. 9(b). It shows that the model trained with $\boldsymbol{\delta}_{min}$ exhibits the smallest $\|\nabla_{\hat{\boldsymbol{x}}}\mathcal{L}_G\|$ so that it can reduce the gap between the training and testing phases significantly. However, the model trained with $\boldsymbol{\delta}_{max}$ exhibits the largest $\|\nabla_{\hat{\boldsymbol{x}}}\mathcal{L}_G\|$. We conjecture that this is because $\boldsymbol{\delta}_{min}$ is the adversarial perturbation concerning the discriminator. So training with the adversarial examples will improve the robustness of the discriminator and the gradient $\|\nabla_{\hat{\boldsymbol{x}}}\mathcal{L}_G\|$ will be lower.



(a) Improvements (b) Records $\|\nabla_{\hat{\boldsymbol{x}}}\mathcal{L}_G\|$

Figure 9: **(a)** shows the performance improvement of MAE error trained with different perturbations under SCINet. The results are averaged among all corresponding forecasting horizons. **(b)** shows the shows $\|\nabla_{\hat{\boldsymbol{x}}}\mathcal{L}_G\|$ during the training of ILI with different perturbations.

C.2 Generation Visualization

We have conducted extensive experiments on six widely used datasets. In order to have a comprehensive understanding of these datasets visually, as well as the comparison of the generation results of different models, we randomly generate samples for these six datasets and visualize them in Fig. 8. As can be seen from the figure, there will be bias amplification in the long-range generation of the realistic datasets, which is more evident in TimeGAN and Cot-GAN. The conditional GANs (SigCWGAN and AEC-GAN) usually perform better than other GANs, thanks to the conditioning information. Meanwhile, our AEC-GAN performs better than SigCWGAN, especially in ETTh2 and ETTm1. In summary, our excellent performance may be derived from the bias mitigation effect of our error correction module.

C.3 Additional Results

The additional experimental results on ETTm2, which are omitted in the main paper due to the space limit, are shown in Tab. 9. With our synthetic time-series data, the forecasting errors are much lower than other GAN models, indicating our excellent generation performance (high quality and wide diversity). In addition, the performance of SCINet can be improved by 2.54% on average.

Table 7: The MSE and MAE errors of **SCINet** based on the generated data from our method and other augmentation methods. The metric is the lower the better. Best results are shown in **bold**. Improvement indicates the improvement of our method relative to the original training set. Each dataset has experimented with three different forecasting horizons (as shown in the second row).

		ETTh1			ETTh2			ETTm1			US Births			ILI		
Methods	Metric	168	336	720	168	336	720	96	288	672	168	336	720	36	48	60
Original	mse	0.450	0.528	0.597	0.554	0.657	1.118	0.197	0.350	1.214	0.268	0.865	1.775	4.089	4.076	4.216
	mae	0.453	0.513	0.571	0.517	0.576	0.776	0.294	0.405	0.836	0.376	0.737	1.140	1.400	1.401	1.452
Augmentation																
Scaling	mse	0.448	0.510	0.580	0.562	0.622	1.073	**0.187**	0.368	1.235	0.266	0.865	1.761	4.101	4.087	4.234
	mae	0.454	0.502	0.559	0.533	0.562	0.761	0.287	0.417	0.852	0.374	0.737	1.135	1.401	1.410	1.457
Jitter	mse	0.450	0.510	0.583	0.562	0.619	1.076	0.197	0.366	1.303	0.268	0.865	1.771	4.092	4.081	4.219
	mae	0.455	0.502	0.560	0.533	0.560	0.762	0.296	0.416	0.882	0.376	0.737	1.139	1.401	1.409	1.453
Mixup	mse	0.457	0.537	0.604	0.625	0.679	1.093	0.216	0.391	1.913	0.268	0.921	1.935	4.232	4.234	4.365
	mae	0.457	0.516	0.571	0.563	0.592	0.773	0.310	0.427	1.057	0.380	0.760	1.188	1.421	1.434	1.476
AEC-GAN	mse	**0.403**	**0.498**	**0.501**	**0.393**	**0.390**	**0.495**	0.190	**0.306**	**0.376**	**0.231**	**0.611**	**0.567**	**3.498**	**3.531**	**3.598**
	mae	**0.432**	**0.496**	**0.504**	**0.420**	**0.428**	**0.510**	**0.279**	**0.354**	**0.395**	**0.345**	**0.602**	**0.581**	**1.278**	**1.297**	**1.329**

Table 8: The MSE and MAE errors of **transformer-based** forecasting methods based on generated data from our method and other GANs. The metric is the lower the better. Best results are shown in **bold**. Improvement indicates the average improvement of our method relative to the original training set on Autoformer and Informer. Each dataset has experimented with three different forecasting horizons (as shown in the second row).

		ETTh1			ETTh2			ETTm1			US Births			ILI		
Methods	Metric	168	336	720	168	336	720	96	288	672	168	336	720	36	48	60
Autoformer																
Original	mse	0.520	0.641	0.676	0.403	0.451	0.462	**0.445**	**0.618**	**0.700**	0.756	1.135	1.356	3.863	3.778	3.918
	mae	**0.485**	0.545	0.582	0.419	0.458	0.472	**0.453**	**0.525**	**0.563**	0.697	0.868	0.989	1.339	1.357	1.403
QuantGAN	mse	1.351	1.292	1.323	1.678	1.886	1.679	0.898	0.807	0.796	3.026	2.457	1.998	4.592	4.627	4.508
	mae	0.851	0.862	0.920	0.969	1.039	0.970	0.680	0.614	0.608	1.454	1.225	1.084	1.527	1.527	1.522
TimeGAN	mse	0.772	0.762	0.740	0.429	0.456	0.452	0.728	0.731	0.752	1.991	1.325	1.748	4.455	4.290	4.259
	mae	0.615	0.620	0.623	0.439	0.461	0.463	0.584	0.577	0.587	1.190	0.949	1.140	1.500	1.474	1.471
Cot-GAN	mse	0.579	0.662	0.696	0.414	**0.450**	**0.451**	0.692	0.718	0.744	1.623	1.565	1.573	4.043	3.955	4.065
	mae	0.510	0.556	0.586	0.422	**0.456**	**0.463**	0.547	0.563	0.580	1.097	1.076	1.085	1.396	1.396	1.439
SigCWGAN	mse	0.528	0.622	0.682	**0.401**	0.450	0.474	0.701	0.737	0.745	0.797	1.172	1.381	**3.672**	3.826	4.050
	mae	0.486	0.539	0.590	**0.414**	0.458	0.488	0.571	0.579	0.582	0.717	0.883	0.974	1.342	1.393	1.441
AEC-GAN	mse	**0.519**	**0.614**	**0.663**	0.413	0.453	0.454	0.673	0.731	0.751	**0.669**	**1.108**	**1.325**	3.730	**3.686**	**3.832**
	mae	0.489	**0.535**	**0.577**	0.425	0.461	0.470	0.553	0.572	0.587	**0.643**	**0.835**	**0.937**	**1.323**	**1.342**	**1.392**
Informer																
Original	mse	0.871	1.121	1.138	6.245	5.608	4.557	**0.680**	0.905	0.955	1.851	1.839	2.114	5.843	6.309	6.112
	mae	0.708	0.848	0.839	2.077	1.997	1.847	**0.572**	0.739	0.750	1.169	1.213	1.313	1.675	1.762	1.739
QuantGAN	mse	1.181	1.258	1.302	3.907	3.996	3.789	1.449	1.296	1.195	4.080	2.623	1.858	8.330	8.334	8.411
	mae	0.791	0.860	0.908	1.531	1.556	1.501	0.910	0.871	0.832	1.738	1.362	1.140	2.111	2.108	2.115
TimeGAN	mse	1.288	1.070	1.016	0.633	0.643	0.572	0.983	1.008	1.049	2.136	2.255	2.198	5.321	4.891	5.002
	mae	0.879	0.808	0.804	0.543	0.562	0.554	0.684	0.729	0.763	1.296	1.346	1.340	1.622	1.558	1.589
Cot-GAN	mse	0.663	0.705	0.710	0.392	0.430	0.428	0.815	**0.763**	**0.767**	1.560	1.563	1.590	5.292	4.907	5.202
	mae	0.544	0.568	0.590	0.413	0.452	0.452	0.573	**0.580**	**0.574**	1.117	1.129	1.142	1.622	1.527	1.591
SigCWGAN	mse	**0.523**	0.697	0.773	0.385	0.545	0.551	0.735	0.786	0.818	1.530	1.794	1.941	5.368	5.057	5.453
	mae	**0.494**	0.584	0.648	0.414	0.481	0.499	0.565	0.598	0.599	1.059	1.202	1.276	1.582	1.551	1.622
AEC-GAN	mse	0.570	**0.673**	**0.648**	**0.349**	**0.360**	**0.379**	0.861	0.895	0.885	**1.266**	**1.397**	**1.479**	**4.645**	**4.173**	**4.495**
	mae	0.509	**0.553**	**0.550**	**0.403**	**0.412**	**0.437**	0.626	0.646	0.632	**0.900**	**1.025**	**1.094**	**1.451**	**1.357**	**1.437**
Improvement	mse	17.38%	22.09%	22.49%	45.97%	46.57%	46.71%	-38.93%	-8.59%	0.02%	21.56%	13.21%	16.16%	11.97%	18.15%	14.44%
	mae	13.64%	18.31%	17.65%	39.58%	39.36%	38.38%	-15.76%	1.82%	6.27%	15.38%	9.65%	10.97%	7.28%	12.05%	9.08%

Table 9: The MSE and MAE errors of our method compared to other GAN-based methods. The metric is the lower the better. Best results are shown in **bold**. Improvement indicates the improvement of our method relative to the original training set.

		ETTm2		
Methods	Metric	96	288	672
Original	mse	0.330	**0.383**	0.501
	mae	0.377	0.408	0.490
QuantGAN	mse	0.861	0.754	0.780
	mae	0.639	0.593	0.590
TimeGAN	mse	0.909	1.144	0.743
	mae	0.609	0.721	0.541
Cot-GAN	mse	0.484	0.607	0.677
	mae	0.438	0.501	0.525
SigCWGAN	mse	0.489	0.547	0.893
	mae	0.436	0.493	0.643
AEC-GAN	mse	**0.315**	0.416	**0.490**
	mae	**0.345**	**0.402**	**0.455**
Improvement	mse	4.55%	-8.62%	2.20%
	mae	8.49%	1.47%	7.14%

C.4 Improvement of Dataset Size

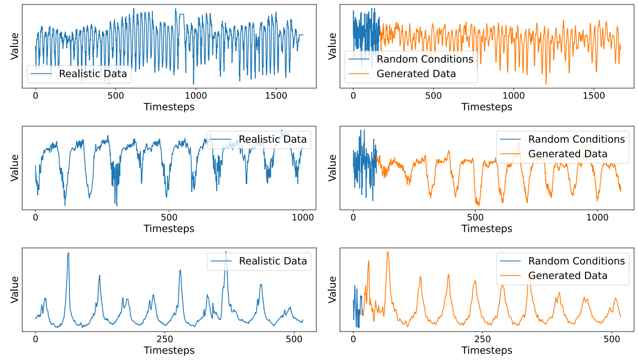
As generative models can generate an extensive amount of data, we explore how much can our synthetic data improve the performance of the time-series forecasting models with different synthetic data sizes. Tab. 6 shows the performances of SCINet with different synthetic data sizes, and the significant improvement under a large amount of data indicates the advantages of synthetic data.

C.5 Comparison with Augmentation Method

In the downstream forecasting task, the time-series GANs can provide high-quality synthetic time-series data to facilitate the training of the forecasting models. Meanwhile, the augmentation methods (e.g., **Mixup**) have shown excellent capability in improving the downstream models' performance. Hence, we also compare our AEC-GAN with several well-known augmentation methods. Similar to the setting in Sec 5.3, the forecasting model (SCINet) is only trained on the augmented data and is evaluated on the test set. The results are shown in Tab. 7. It shows that the generated data generated by AEC-GAN outperforms the augmented data on these datasets.

C.6 Effects of Error Correction Module

We develop an error correction module M_ω to mitigate the distribution shifts. In previous experiments, we measure the distribution shifts in the long sequence generation (4000 steps). In this experiment, we directly generate sequences from Gaussian noises, which are regarded as the conditioning variables. In Fig. 10, we can clearly see that the generated sequences, which start with Gaussian noises, gradually become more similar to the realistic sequences as the generation proceeds. This phenomenon demonstrates that our AEC-GAN can mitigate the distribution shifts (even from Gaussian noise) and generate high-quality long sequences.



(a) Real Data (b) Generated Data

Figure 10: Generation examples. Each row represents a dataset (ETTh1, ETTm2, ILI). **(a)** shows the real data samples. **(b)** are generated samples given Gaussian noises as the conditioning variables.

References

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*, 214–223. PMLR.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Brophy, E.; Wang, Z.; She, Q.; and Ward, T. 2021. Generative adversarial networks in time series: A survey and taxonomy. *arXiv preprint arXiv:2107.11098*.

Chakraborty, A.; Alam, M.; Dey, V.; Chattopadhyay, A.; and Mukhopadhyay, D. 2018. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*.

Ding, Q.; Wu, S.; Sun, H.; Guo, J.; and Guo, J. 2020. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In *IJCAI*, 4640–4646.

Donahue, C.; McAuley, J.; and Puckette, M. 2018. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*.

Esteban, C.; Hyland, S. L.; and Rätsch, G. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*.

Fluview. 2022. National, Regional, and State Level Outpatient Illness and Viral Surveillance. https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

Franceschi, J.-Y.; Dieuleveut, A.; and Jaggi, M. 2019. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32.

Fréchet, M. 1957. Sur la distance de deux lois de probabilité. *Comptes Rendus Hebdomadaires des Seances de L Academie des Sciences*, 244(6): 689–692.

Godahewa, R.; Bergmeir, C.; Webb, G. I.; Hyndman, R. J.; and Montero-Manso, P. 2021. Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643*.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Hamilton, J. D. 2020. *Time series analysis*. Princeton university press.

Hazra, D.; and Byun, Y.-C. 2020. SynSigGAN: Generative adversarial networks for synthetic biomedical signal generation. *Biology*, 9(12): 441.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Jarrett, D.; Bica, I.; and van der Schaar, M. 2021. Time-series generation by contrastive imitation. *Advances in Neural Information Processing Systems*, 34: 28968–28982.

Jiang, L.; Dai, B.; Wu, W.; and Loy, C. C. 2021. Deceive D: Adaptive pseudo augmentation for gan training with limited data. *Advances in Neural Information Processing Systems*, 34: 21655–21667.

Kapoor, A.; Ben, X.; Liu, L.; Perozzi, B.; Barnes, M.; Blais, M.; and O'Banion, S. 2020. Examining covid-19 forecasting using spatio-temporal graph neural networks. *arXiv preprint arXiv:2007.03113*.

Karras, T.; Aittala, M.; Hellsten, J.; Laine, S.; Lehtinen, J.; and Aila, T. 2020a. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33: 12104–12114.

Karras, T.; Aittala, M.; Laine, S.; Härkönen, E.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2021. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34: 852–863.

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020b. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8110–8119.

Li, J.; Wang, X.; Lin, Y.; Sinha, A.; and Wellman, M. 2020. Generating realistic stock market order streams. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 727–734.

Lim, B.; and Zohren, S. 2021. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194): 20200209.

Liu, M.; Zeng, A.; Xu, Z.; Lai, Q.; and Xu, Q. 2021. Time series is a special sequence: Forecasting with sample convolution and interaction. *arXiv preprint arXiv:2106.09305*.

Liu, X.; and Hsieh, C.-J. 2019. Rob-gan: Generator, discriminator, and adversarial attacker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11234–11243.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Mao, X.; Li, Q.; Xie, H.; Lau, R. Y.; Wang, Z.; and Paul Smolley, S. 2017. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2794–2802.

Mirza, M.; and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Ni, H.; Szpruch, L.; Wiese, M.; Liao, S.; and Xiao, B. 2020. Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421*.

Paul, J.; Michael, B.-S.; Pedro, M.; Rajbir, S. N.; Shubham, K.; Valentin, F.; Jan, G.; and Tim, J. 2021. PSA-GAN: Progressive Self Attention GANs for Synthetic Time Series. *arXiv preprint arXiv:2108.00981*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tran, N.-T.; Tran, V.-H.; Nguyen, N.-B.; Nguyen, T.-K.; and Cheung, N.-M. 2021. On data augmentation for gan training. *IEEE Transactions on Image Processing*, 30: 1882–1897.

Um, T. T.; Pfister, F. M.; Pichler, D.; Endo, S.; Lang, M.; Hirche, S.; Fietzek, U.; and Kulić, D. 2017. Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM international conference on multimodal interaction*, 216–220.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wiese, M.; Knobloch, R.; Korn, R.; and Kretschmer, P. 2020. Quant GANs: deep generation of financial time series. *Quantitative Finance*, 20(9): 1419–1440.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34: 22419–22430.

Xiao, C.; Li, B.; Zhu, J.-Y.; He, W.; Liu, M.; and Song, D. 2018. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*.

Xu, T.; Wenliang, L. K.; Munn, M.; and Acciaio, B. 2020. Cot-gan: Generating sequential data via causal optimal transport. *Advances in Neural Information Processing Systems*, 33: 8798–8809.

Yoon, J.; Jarrett, D.; and Van der Schaar, M. 2019. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 11106–11115.