Stochastic Diffusion: A Diffusion Probabilistic Model for Stochastic Time Series Forecasting

Yuansan Liu [1], Sudanthi Wijewickrema [2], Dongting Hu [3], Christofer Bester [2]

Stephen O'Leary [2], James Bailey [1]

[1] School of Computing and Information Systems, The University of Melbourne
[2] Department of Surgery (Otolaryngology), The University of Melbourne
[3] School of Mathematics and Statistics, The University of Melbourne

Abstract

Recent innovations in diffusion probabilistic models have paved the way for significant progress in image, text and audio generation, leading to their applications in generative time series forecasting. However, leveraging such abilities to model highly stochastic time series data remains a challenge. In this paper, we propose a novel **Stoch**astic **Diff**usion (StochDiff) model which learns data-driven prior knowledge at each time step by utilizing the representational power of the stochastic latent spaces to model the variability of the multivariate time series data. The learnt prior knowledge helps the model to capture complex temporal dynamics and the inherent uncertainty of the data. This improves its ability to model highly stochastic time series data. Through extensive experiments on real-world datasets, we demonstrate the effectiveness of our proposed model on stochastic time series forecasting. Additionally, we showcase an application of our model for real-world surgical guidance, highlighting its potential to benefit the medical community.

1 Introduction

Time series forecasting plays a pivotal role across various domains of human society, including health [5, 14], meteorology [11, 10] and economics [4, 17]. One effective approach to this task is conditional generation, which approximates the distribution of future data conditioned on historical data. The recent development of diffusion probabilistic models has demonstrated the superior generation ability of this model family across various data types including time series. This has led to the application of diffusion models for generative time series forecasting.

Existing diffusion based time series forecasting models mostly apply forecasting architectures such as Recurrent Neural Networks (RNN) or Transformers to capture global temporal dynamics from historical sub-series and learn the underlying patterns that can be used for predicting future values. Then, diffusion modules are implemented to generate high quality time series data from a standard Gaussian distribution, conditioning on the learnt temporal features. Although these models have achieved success with various long-period time series data, challenges still remain: (1) For individual time series data like clinical patient data, variability between each series can introduce extra stochasticity and make it challenging to model the historical data. (2) As a latent variable model, the prior distribution of the diffusion models is commonly fixed to be the standard Gaussian distribution, which might not be able to encode the full stochasticity of real-world time series data. (3) Many real-world time series are multivariate and single-dimensional latent vectors may not be accurate enough for modeling high-dimensional sub-series.

Preprint. Under review.

To address these challenges, we propose a novel diffusion based time series forecasting model called **Stoch**astic **Diff**usion (StochDiff). StochDiff first models the time series at each time step, allowing the latent variables to focus on encoding dimensional dependence only. Secondly, data-driven prior knowledge is learnt during time series modelling to improve the model's ability to extract temporal features from highly stochastic time series data. Finally, a Gaussian Mixture Model is fitted on sampled future data to obtain more accurate point-wise forecasting results. Experiments on six real-world datasets across different domains show that StochDiff can achieve highly competitive performance compared to existing state-of-the-art diffusion based time series forecasting models. Additionally, a case study using real-world surgical data highlights the model's potential in the biomedical domain.

Our contributions can be summarized as follows:

- To the best of our knowledge, our proposed StochDiff approach is the first diffusion based model that learns data-driven prior knowledge at each time step for more effective time series modelling.
- Through extensive experiments on real-world datasets, we demonstrate the competitive performance of StochDiff, especially on highly stochastic time series data.
- We showcase the model's ability to address a real-world problem with a case study using intra-operative data from cochlear implant surgery.

2 Preliminaries

We begin with some necessary background and terminology on diffusion models.

2.1 Denoising Diffusion Probabilistic Model

Diffusion probabilistic models generate a data distribution with a series of latent variables [21]:

$$p_\theta(\boldsymbol{x}^0, \boldsymbol{x}^1, \ldots, \boldsymbol{x}^N) = p_\theta(\boldsymbol{x}^{0:N}) := p(\boldsymbol{x}^N) \prod_{n=1}^{N} p_\theta(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n) \tag{1}$$

where, $\boldsymbol{x}^0$ is the data variable and $\boldsymbol{x}^{1:N}$ are latent variables of the same dimension as the data variables. By integrating over the latent variables we can get the data distribution as $p_\theta(\boldsymbol{x}^0) := \int p_\theta(\boldsymbol{x}^{0:N}) d\boldsymbol{x}^{1:N}$.

In construction of these latent variables, Diffusion Models use a *forward diffusion process* that follows a Markov Chain to gradually add Gaussian noise into the data and finally convert data into standard Gaussian noise over N scheduled steps:

$$q(\boldsymbol{x}^{1:N}|\boldsymbol{x}^0) := \prod_{n=1}^{N} q(\boldsymbol{x}^n|\boldsymbol{x}^n - 1), \quad q(\boldsymbol{x}^n|\boldsymbol{x}^{n-1}) := \mathcal{N}(\boldsymbol{x}^n; \sqrt{1-\beta_n}\boldsymbol{x}^{n-1}, \beta_n \boldsymbol{I}) \tag{2}$$

where, β_n are learnable parameters representing the scheduled variance level over N steps. In practice, we often fix β_n to small positive constants to simplify the model.

Based on the mathematical property of forward diffusion, we can directly sample $\boldsymbol{x}^n$ at an arbitrary step n from $\boldsymbol{x}^0$ using the close form expression:

$$q(\boldsymbol{x}^n|\boldsymbol{x}^0) = \mathcal{N}(\boldsymbol{x}^n; \sqrt{1-\overline{\alpha}_n}\boldsymbol{x}^0, (1-\overline{\alpha}_n)\boldsymbol{I}) \tag{3}$$

where, $\alpha^n := 1 - \beta_n$ and $\overline{\alpha}_n := \prod_{i=1}^{n} \alpha_i$.

Once the latent variables are created, a *reverse diffusion process* in the form of equation 1 will learn to remove noise from each $\boldsymbol{x}^n$ to reconstruct the original data $\boldsymbol{x}^0$. The start point $p(\boldsymbol{x}^N)$ is commonly fixed as $\mathcal{N}(\boldsymbol{x}^N; 0, \boldsymbol{I})$, and each subsequent transition is given by the following parametrization:

$$p_\theta(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n) := \mathcal{N}(\boldsymbol{x}^{n-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}^n, n), \boldsymbol{\delta}_\theta(\boldsymbol{x}^n, n)). \tag{4}$$

Training the diffusion model is accomplished by uniformly sampling n from $\{1, \ldots, N\}$ and optimizing the Kullback-Leibler (KL) divergence between the forward process posteriors and the reverse process:

$$\mathcal{L}_n = D_{KL}(q(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n)||p_\theta(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n)). \tag{5}$$

Although $q(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n)$ is unknown, it can be tractable when conditioned on $\boldsymbol{x}^0$ given equation 3:

$$q(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n, \boldsymbol{x}^0) = \mathcal{N}(\boldsymbol{x}^{n-1}; \tilde{\boldsymbol{\mu}}_n(\boldsymbol{x}^n, \boldsymbol{x}^0), \tilde{\beta}_n \boldsymbol{I}) \tag{6}$$

where, $\tilde{\boldsymbol{\mu}}_n(\boldsymbol{x}^n, \boldsymbol{x}^0) := \frac{\sqrt{\overline{\alpha}_{n-1}}\beta_n}{1-\overline{\alpha}_n}\boldsymbol{x}^0 + \frac{\sqrt{\alpha_n}(1-\overline{\alpha}_{n-1})}{1-\overline{\alpha}_n}\boldsymbol{x}^n, \quad \tilde{\beta}_n := \frac{1-\overline{\alpha}_{n-1}}{1-\overline{\alpha}_n}\beta_n.$

For parametrization in equation 4, normally we follow Ho et al. [12] to fix $\boldsymbol{\delta}_\theta(\boldsymbol{x}^n, n)$ at $\boldsymbol{\sigma}_n^2\boldsymbol{I}$ where $\boldsymbol{\sigma}_n^2 = \overline{\beta}_n := \frac{1-\overline{\alpha}_{n-1}}{1-\overline{\alpha}_n}\beta_n, \overline{\beta}_1 = \beta_1$, and use neural networks to model $\boldsymbol{\mu}_\theta(\boldsymbol{x}^n, n)$.

With all the simplifications and transformations, we can rewrite the objective in equation 5 as:

$$\mathcal{L}_n = \frac{1}{2\boldsymbol{\sigma}_n^2}||\tilde{\boldsymbol{\mu}}_n(\boldsymbol{x}^n, \boldsymbol{x}^0, n) - \boldsymbol{\mu}_\theta(\boldsymbol{x}^n, n)||^2. \tag{7}$$

Note that in equation 7, due to the flexibility of neural networks, $\boldsymbol{\mu}_\theta(\boldsymbol{x}^n, n)$ can be modelled in two ways: $\boldsymbol{\mu}(\boldsymbol{\epsilon}_\theta)$ and $\boldsymbol{\mu}(\boldsymbol{x}_\theta)$. The former is widely used in [12] and its derivations which involves a noise prediction model. The later one trains a data prediction model $\boldsymbol{x}_\theta(\boldsymbol{x}^n, n)$ to obtain the $\boldsymbol{\mu}(\boldsymbol{x}_\theta)$:

$$\boldsymbol{\mu}(\boldsymbol{x}_\theta) = \frac{\sqrt{\alpha_n}(1-\overline{\alpha}_{n-1})}{1-\overline{\alpha}_n}\boldsymbol{x}^n + \frac{\sqrt{\overline{\alpha}_{n-1}}\beta_n}{1-\overline{\alpha}_k}\boldsymbol{x}_\theta(\boldsymbol{x}^n, n). \tag{8}$$

and the corresponding objective function is:

$$\mathcal{L}_{\boldsymbol{x}} = \mathbb{E}_{n,\boldsymbol{x}^0}[||\boldsymbol{x}^0 - \boldsymbol{x}_\theta(\boldsymbol{x}^n, n)||^2]. \tag{9}$$

2.2 Probabilistic Time Series Forecasting

Assume we have a multivariate time series $\boldsymbol{x}_{1:T}^0 = \{\boldsymbol{x}_1^0, \boldsymbol{x}_2^0, \ldots, \boldsymbol{x}_{t_0}^0, \ldots, \boldsymbol{x}_T^0\}$, where, $\boldsymbol{x}_t^0 \in \mathbb{R}^d$ is a $d-$dimensional vector representing d measurements for an event happening at time step t. Probabilistic Time Series Forecasting involves: (1) modelling the observed data $\boldsymbol{x}_{1:t_0}^0$ and (2) predicting the unseen future data $\boldsymbol{x}_{t_0:T}^0$ by modelling the conditional distribution of future data given the observed data:

$$q_\chi(\boldsymbol{x}_{t_0:T}^0|\boldsymbol{x}_{1:t_0-1}^0) = \prod_{t=t_0}^{T} q_\chi(\boldsymbol{x}_t^0|\boldsymbol{x}_{1:t-1}^0). \tag{10}$$

3 Stochastic Diffusion for Time Series Forecasting

In this section, we discuss the details of our proposed Stochastic Diffusion (StochDiff) model. An overview of the model structure is provided in Figure 1. The model comprises two parts: Modelling

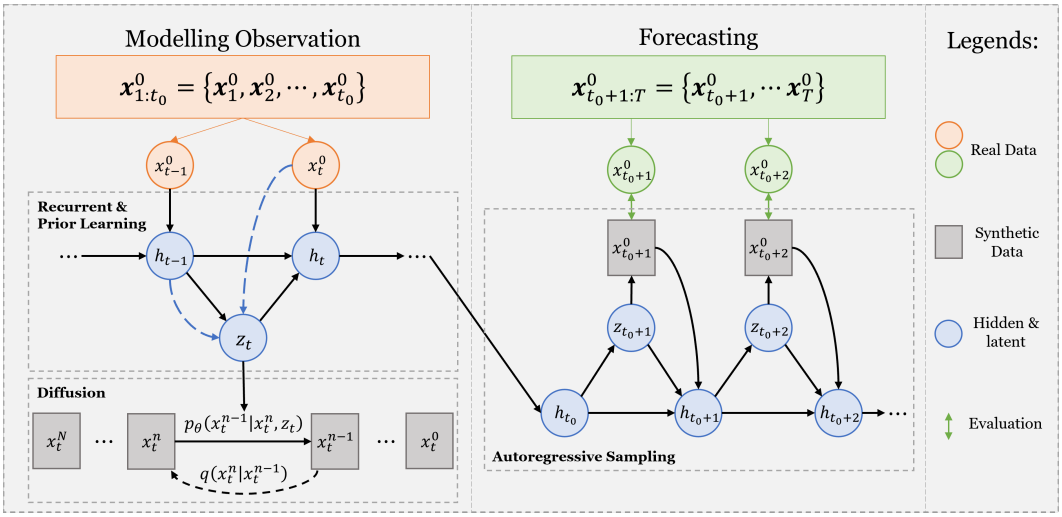


Figure 1: Stochastic Diffusion for Time Series Forecasting

and Forecasting. In the Modelling part, *Recurrent and Prior Learning* firstly learns the prior vector $\boldsymbol{z}_t$: it consists of an *RNN* module that converts the original time series data $\boldsymbol{x}_t^0$ into the hidden units h_t, and a *Prior Encoder* that encodes the previous hidden units h_{t-1} into the current prior vector $\boldsymbol{z}_t$. Secondly, the *Diffusion* module integrates the learnt prior vector $\boldsymbol{z}_t$ into its latent variables to reconstruct the original data $\boldsymbol{x}_t^0$. The Forecasting part involves autoregressive generation of future data: the last hidden unit h_{t_0} from the observation (Modelling part) is taken to learn the first prior vector $\boldsymbol{z}_{t_0+1}$ in the prediction window. The predicted $\boldsymbol{x}_{t_0+1}^0$ is sampled from the diffusion model and is input to the LSTM module to obtain its corresponding hidden unit h_{t_0+1} and the prior vector for the next time step z_{t_0+2}.

3.1 Data-Driven Prior Knowledge with Step-Wise Modelling

In most existing diffusion models developed for the time series forecasting tasks [20, 9, 16], the time series data is modelled as sub-sequences. This allows the model to extract the intrinsic temporal properties that can be used to predict future dynamics. However, several challenges exist for this strategy: Firstly, when modelling highly stochastic data, such as clinical data which are highly variable between different patients, the model may not be able to learn informative knowledge from sub-sequences, leading to failures in forecasting. Secondly, given that real-world time series data are mostly multivariate, the latent variables of one single dimension (latent vectors) might not be enough to represent sub-sequences with multiple dimensions. Additionally, the prior distribution over the latent variable $p(\boldsymbol{x}^N)$ is commonly fixed to be the standard Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$ in order to simplify the computation and parameter estimation. This may not be the best choice when processing highly stochastic and non-linear real-world time series data. The choice of normally distributed latent codes may introduce inflexibility and inexpressiveness into the data modelling, making it less appropriate to represent complex temporal systems.

To address these challenges and develop an efficient learning paradigm for highly stochastic time series, we build on previous works [6, 1] to design a data-driven prior variable of time series data at each time-step:

$$\boldsymbol{z}_t \sim p_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}^0, \boldsymbol{z}_{1:t-1}) := \mathcal{N}(\hat{\boldsymbol{\mu}}_\theta(h_{t-1}), \hat{\boldsymbol{\delta}}_\theta(h_{t-1})), \quad h_{t-1} = \boldsymbol{f}_\theta(\boldsymbol{x}_{t-1}^0, \boldsymbol{z}_{t-1}, h_{t-2}) \tag{11}$$

where, $\hat{\boldsymbol{\mu}}_\theta(\cdot), \hat{\boldsymbol{\delta}}_\theta(\cdot)$ are functions that simulate the parameters of prior distribution, and can be approximated by neural networks. h_t are the hidden states of the neural network $\boldsymbol{f}_\theta$ which is used to model the time series data.

We follow the common setting to use an *RNN* to model the temporal dynamics. Specifically, we use *Long Short Term Memory Networks (LSTM)* as the time series modelling backbone. Hence, $\boldsymbol{f}_\theta$ in equation 11 is *LSTM* and each h_t is the hidden state of it. The *Prior Encoder* includes both $\hat{\boldsymbol{\mu}}_\theta(\cdot), \hat{\boldsymbol{\delta}}_\theta(\cdot)$ that learn the distribution parameter and a Fully Connected Network (FCN) that projects the sampled random variable into $\boldsymbol{z}_t$ for the purpose of optimization.

With equation 11, the prior distribution at each time-step can be learnt by the model rather than fixed as a standard Gaussian Distribution. The latent variables sampled from these prior distributions only compress the information of each time step, which makes them able to (1) encode more informative representations, (2) better approximate the real data distributions, and subsequently, (3) benefit the time series data modelling.

In addition to the prior encoder (equation 11), the model has another encoder that takes both the previous hidden unit h_{t-1} and the current data $\boldsymbol{x}_t$ (blue dash lines in Figure 1) to guide the approximation of the posterior distribution of $\boldsymbol{z}_t$:

$$q_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t}^0, \boldsymbol{z}_{1:t-1}) := \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{z},t}(h_{t-1}, \boldsymbol{x}_t^0), \boldsymbol{\delta}_{\boldsymbol{z},t}(h_{t-1}, \boldsymbol{x}_t^0)). \tag{12}$$

Similarly, it also includes both $\boldsymbol{\mu}_{\boldsymbol{z},t}(\cdot), \boldsymbol{\delta}_{\boldsymbol{z},t}(\cdot)$ that learn the distribution parameters and the FCN that projects the learnable $\boldsymbol{z}_t$.

3.2 Time Series Forecasting via Conditional Generative Diffusion

With the learnt prior distribution $p(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}^0, \boldsymbol{z}_{1:t-1})$, we can introduce prior knowledge into the diffusion latent variables. Note that, in previous work [16], prior knowledge is learnt from the observed sub-sequences and directly replaces the latent variable for both the forward (at the end

point) and reverse (at the start point) diffusion process. However, due to the statistical properties of the forward process, the covariance matrix of the prior vector must be fixed to identity matrix $\boldsymbol{I}$. This introduces extra constraints to the learning of prior knowledge. Thus, in our method, to enable fully learnt prior knowledge, we retain the original end point $\boldsymbol{x}_t^N \sim \mathcal{N}(0, \boldsymbol{I})$ for the forward diffusion process, and integrate the prior knowledge into the the reverse diffusion process as a condition. Thus, the distributions of the latent variables become conditional distributions over the prior variable $\boldsymbol{z}_t$:

$$\begin{aligned} p_\theta(\boldsymbol{x}_t^{n-1}|\boldsymbol{x}_t^n, \boldsymbol{z}_t) &= \mathcal{N}(\boldsymbol{x}_t^{n-1}; \mathcal{C}(\boldsymbol{\mu}_\theta(\boldsymbol{x}_t^n, n), \hat{\boldsymbol{\mu}}_\theta(h_{t-1})), \mathcal{C}(\boldsymbol{\delta}_\theta(\boldsymbol{x}^n, n), \hat{\boldsymbol{\delta}}_\theta(h_{t-1}))) \\ &= \mathcal{N}(\boldsymbol{x}_t^{n-1}; \boldsymbol{\mu}_\mathcal{C}(\boldsymbol{x}_t^n, n, \boldsymbol{z}_t), \boldsymbol{\delta}_\mathcal{C}(\boldsymbol{x}_t^n, n, \boldsymbol{z}_t)) \end{aligned} \tag{13}$$

where, $\mathcal{C}$ is a fusion function that can be simulated by a neural network. In our case, we use a cross-attention mechanism to learn the proper integration of two parameter vectors. Additionally, we also fix the fused covariance matrix $\boldsymbol{\delta}_\mathcal{C}(\boldsymbol{x}_t^n, n, \boldsymbol{z}_t)$ at $\boldsymbol{\sigma}_n^2 \boldsymbol{I}$ for computational simplicity.

Integrating equation 13 into the generative diffusion part (equation 1), and applying the probabilistic time series forecasting formulation (equation 10), we have a variational generative diffusion process for time series forecasting. The details of this formula's integration are provided in Appendix A:

$$\begin{aligned} &p_\theta(\boldsymbol{x}_{t_0:T}^0|\boldsymbol{x}_{1:t_0-1}^0, \boldsymbol{z}_{1:t_0-1}) = \\ &\int_{\boldsymbol{x}_{t_0:T}^{1:N}} \int_{\boldsymbol{z}_{t_0:T}} \prod_{t=t_0}^{T} p(\boldsymbol{x}_t^N|\boldsymbol{z}_t) \prod_{n=1}^{N} p_\theta(\boldsymbol{x}_t^{n-1}|\boldsymbol{x}_t^n, \boldsymbol{z}_t) p_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}^0, \boldsymbol{z}_{1:t-1}) d\boldsymbol{z}_{t_0:T} d\boldsymbol{x}_{t_0:T}^{1:N}. \end{aligned} \tag{14}$$

Following equation 8, our new data prediction model is now built on $\boldsymbol{x}_t^n$, n and $\boldsymbol{z}_t$. The sampled data at each diffusion step is:

$$\boldsymbol{x}_t^{n-1} = \frac{\sqrt{\alpha_n}(1-\overline{\alpha}_{n-1})}{1-\overline{\alpha}_n} \boldsymbol{x}_t^n + \frac{\sqrt{\overline{\alpha}_{n-1}}\beta_n}{1-\overline{\alpha}_k} \boldsymbol{x}_\theta(\boldsymbol{x}_t^n, n, \boldsymbol{z}_t). \tag{15}$$

3.3 Dual Objective Optimization

Our proposed StochDiff merges the learning of prior knowledge and the denoising diffusion process at each time step. Its objective, hence, becomes the step-wise combination of both parts. The prior knowledge learning is associated with the variational inference that optimizes the approximate posterior $q_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t}^0, \boldsymbol{z}_{1:t-1})$ by minimizing the KL divergence between it and the prior:

$$D_{KL}(q_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t}^0, \boldsymbol{z}_{1:t-1})||p_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}^0, \boldsymbol{z}_{1:t-1})). \tag{16}$$

Then, putting this together with the aforementioned diffusion objective (equation 9), we have a step-wise dual objective function for our proposed StochDiff:

$$\mathcal{L}_{dual} = \sum_{t=1} D_{KL}(q_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t}^0, \boldsymbol{z}_{1:t-1})||p_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}^0, \boldsymbol{z}_{1:t-1})) + \mathbb{E}_{\boldsymbol{x}_t^0, n, \boldsymbol{z}_t}[||\boldsymbol{x}_t^0 - \boldsymbol{x}_\theta(\boldsymbol{x}_t^n, n, \boldsymbol{z}_t)||^2]. \tag{17}$$

Notice here, we use a data prediction module in the diffusion model rather than the common setting of a noise prediction model. Our rationale is that in previous diffusion models, the added noise in each diffusion step is Gaussian noise with scheduled variances, which makes the noise more predictable. However, the learnt prior in our model introduces extra noise into the latent variable, making the noise less predictable. Thus, a data prediction model can be more effective here. We designed an *Attention-Net* that extracts the correlations between dimensions at each time step, and reconstructs the data point with auxiliary prior knowledge. The details of this reconstruction network are provided in Appendix C.

The model is trained by modelling the time series data in the training set. Once trained, the model is used to autoregressively forecast the future time series with the following steps: (1) model the observations to setup the hidden units, (2) at each future time step obtain the prior knowledge from the hidden units, and (3) sample the data with auxiliary prior knowledge. This sampled data is later used for these computations at the next step. The details of training and forecasting process are provided in Algorithm 1 and 2, and $\mathcal{U}$ denotes the uniform distribution.

Algorithm 1 Training via Time Series Modelling

Input Training time series data $\boldsymbol{x}_{1:t_0}$.
Initialize $h_0 = 0, \mathcal{L}_{total} = 0$.
repeat
  for $t = 1$ to t_0 **do**
    Sample $n \sim \mathcal{U}(\{1, 2, \ldots, N\})$.
    Sample $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$.
    Observe $\boldsymbol{x}_t$ as $\boldsymbol{x}_t^0$.
    Obtain $\boldsymbol{z}_t \sim p_z$ and $\boldsymbol{z}_t \sim q_z$ via equation 11,12.
    Obtain prior knowledge vector $\boldsymbol{z}_t \sim q_z$.
    Obtain reconstructed $\boldsymbol{x}_t^0$ based on $\boldsymbol{z}_t$.
    Calculate loss function $\mathcal{L}_{dual}$ in equation 17.
    $\mathcal{L}_{total} += \mathcal{L}_{dual}$.
  end for
  Take gradient descent step on $\nabla \mathcal{L}_{total}$, and update model parameters.
until converged

Algorithm 2 Autoregressive Forecasting

Require: trained denoising network $\boldsymbol{x}_\theta$, recurrent network $\boldsymbol{f}_\theta$.
Input Test time series data $\boldsymbol{x}_{1:T}$.
Initialize $h_0 = 0$.
for $t = 1$ to t_0 **do**
  Observe $\boldsymbol{x}_t$ as $\boldsymbol{x}_t^0$.
  Obtain $\boldsymbol{z}_t \sim q_z$ from equation 12.
  Update h_t via $\boldsymbol{f}_\theta(\boldsymbol{x}_t, \boldsymbol{z}_t, h_{t-1})$.
end for
Obtain h_{t_0} (end point of the previous **for** loop).
Initialize $\hat{\boldsymbol{x}}_{t_0+1:T} = \{\}$.
for $t = t_0 + 1$ to T **do**
  Obtain $\boldsymbol{z}_t \sim p_z$ from equation 11
  for $n = N$ to 1 **do**
    Sample $\hat{\boldsymbol{x}}_t^{n-1}$ using equation 15.
  end for
  Update h_t via $\boldsymbol{f}_\theta(\hat{\boldsymbol{x}}_t^0, \boldsymbol{z}_t, h_{t-1})$.
  Append $\hat{\boldsymbol{x}}_t^0$ into $\hat{\boldsymbol{x}}_{t_0+1:T}$.
end for
return $\hat{\boldsymbol{x}}_{t_0+1:T}$

3.4 Improved Point-Wise Solution

Similar to other probabilistic models, diffusion models always generate a data distribution instead of point estimation. This introduces the problem of choosing a single result from the distribution for a downstream application. Existing diffusion models directly use the median values as the final results to represent the central tendency of predictions. However, this strategy might introduce errors given a complex data distribution such as mixture of Gaussians. To address this issue, inspired by Hu et al. [13], we propose an additional step to train a Gaussian Mixture Model (GMM) on the outputs and identify the largest cluster center as the selected point-wise result.

4 Experiments

We evaluate our proposed StochDiff on six real-world time series datasets, against several state of the art diffusion models for time series forecasting tasks.

4.1 Datasets

The time series datasets we used include 4 commonly used datasets in time series forecasting. They mainly consist of data that have been collected over a long period. These data include *Exchange, Weather, Traffic*, and *Solar*. For these datasets, we divide each of them into training sets and test sets based on their recording time. More information of these datasets are provided in Appendix B.

In addition to these long-period datasets, we also include two clinical datasets to evaluate our model's ability for learning highly stochastic multivariate time series data. They include: (1) *ECochG*[23], which contains 78 patients' records from cochlear implant surgeries and (2) *MMG*[1], which contains uterine magnetomyographic (MMG) signals of 25 subjects. These patient datasets contain high inter and intra variances across different patients, and bring extra challenges to the forecasting task. For these patient datasets, we randomly select 70% patients and use their data for training, and use the rest of the patients' data for testing.

4.2 Baseline Methods & Evaluation Metrics

Focusing on the diffusion based time series forecasting models, we compare our proposed StochDiff with existing state-of-the-art diffusion models developed for time series forecasting tasks, including the well-known time series diffusion models: *TimeGrad* [18], *SSSD*[2], and *TimeDiff*[20], as well as the most recent models: *TMDM*[16] and *MG-TSD*[9]. We use their original settings as optimised by their authors. Details of our model's setup can be found in Appendix C.

To quantitatively evaluate model performance, we use two common metrics, namely Normalized Root Mean Squared Error ($NRMSE$) and Continuous Ranked Probability Score ($CRPS$). The former evaluates the quality of the final point-wise results, and the latter measures the compatibility of the model's predictions with the real data. Here, instead of using original $CRPS$, following Rasul et al. [18], we use $CRPS_{sum}$. The details of these metrics can be found in Appendix D. For both metrics, lower values are better.

4.3 Experimental Results

Quantitative results are provided in Tables 1 and 2, The best results are marked with bold font. We

Table 1: $NRMSE$ results. Lower values are better.

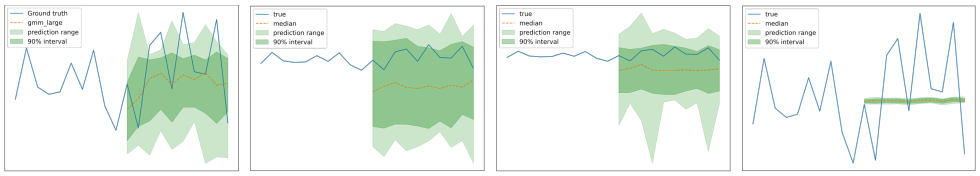
Model	Exchange	Weather	Traffic	Solar	ECochG	MMG
TimeGrad	0.066±0.007	0.691±0.017	0.603±0.021	1.112±0.014	0.952±0.025	2.977±0.101
SSSD	0.065±0.006	0.755±0.041	0.674±0.018	1.458±0.023	0.965±0.041	2.184±0.105
TimeDiff	0.074±0.002	0.702±0.025	0.615±0.013	1.082±0.016	0.916±0.033	1.565±0.091
TMDM	0.063±0.005	0.564±0.019	0.692±0.017	0.989±0.010	1.074±0.052	5.366±0.114
MG-TSD	0.075±0.005	0.696±0.023	**0.602±0.015**	**0.976±0.015**	0.954±0.045	1.716±0.087
StochDiff (ours)	**0.052±0.007**	**0.491±0.014**	0.611±0.012	1.081±0.013	**0.859±0.031**	**1.529±0.094**

Table 2: $CRPS_{sum}$ results. Lower values are better.

Model	Exchange	Weather	Traffic	Solar	ECochG	MMG
TimeGrad	0.010±0.002	0.527±0.015	0.487±0.010	0.416±0.036	0.398±0.019	1.057±0.041
SSSD	0.010±0.001	0.701±0.047	0.605±0.014	0.506±0.053	0.445±0.016	1.372±0.025
TimeDiff	0.012±0.002	0.635±0.019	0.551±0.023	0.423±0.071	0.458±0.015	0.948±0.020
TMDM	0.010±0.001	0.618±0.029	0.542±0.021	0.382±0.039	0.671±0.015	1.495±0.043
MG-TSD	0.009±0.000	0.596±0.027	**0.475±0.017**	**0.375±0.067**	0.409±0.017	0.961±0.031
StochDiff (ours)	**0.008±0.001**	**0.521±0.012**	0.521±0.011	0.391±0.052	**0.345±0.013**	**0.818±0.023**

can see our proposed *StochDiff* method performs best in most of the datasets except *Traffic* and *Solar* where the recent model *MG-TSD* achieves the best results. We highlight the performance of our StochDiff model on clinical datasets for which the model is specifically designed. Figure 2 provides some visual forecasting results on the *ECochG* dataset to help with assessment. As shown,

[1]https://physionet.org/content/mmgdb/1.0.0/



StochDiff MG-TSD TimeGrad TimeDiff

Figure 2: Forecasting results of 4 the best models (based on quantitative results). We display the entire prediction range together with the 90% prediction interval via green bars. The orange dashed lines represent the point-wise results which are medians for baseline methods and the largest centers of fitted GMM for our model.

the baselines fail to capture the variance of the data and tend to predict stationary values for this dataset, and their prediction distribution is either too wide or too narrow. On the other hand, our StochDiff model can learn the dynamics from this highly complex and stochastic data, and is able to narrow its predictions around the true values. Note here, the *TimeDiff* model was originally built for point-wise prediction, and thus, its prediction distributions are relatively concentrated. We provide more visual results in Appendix E.

4.4 Ablation Study

To investigate the contribution of the different components in our proposed model, we evaluate several basic component models on two clinical datasets: (1) A basic RNN model, *LSTM* and (2) a variational derivation of LSTM, *vLSTM-*$\mathcal{N}(0,1)$ that models the data at each time step with a VAE [15] and Standard Gaussian Prior and turns the deterministic RNN model into a probabilistic model, (3) The vLSTM model with diffusion module replacing VAE, *vLSTM-diffusion*, and (4) our proposed *StochDiff* that learns data-driven prior knowledge at each time step. By comparing their performance, we can justify the improvements brought by the step-wise modelling (1,2), the generative diffusion process (2,3) and learnt prior knowledge (3,4) for time series forecasting. Details are provided

Table 3: Ablation Study on *ECochG* and *MMG* datasets

Model	ECochG		MMG	
	NRMSE	CRPS	NRMSE	CRPS
LSTM	1.003±0.042	-	3.657±0.138	-
vLSTM-$\mathcal{N}(0,1)$	0.982±0.036	0.623±0.014	3.315±0.114	1.767±0.041
vLSTM-diffusion	0.956±0.035	0.405±0.017	3.001±0.112	1.639±0.032
StochDiff (ours)	**0.859±0.031**	**0.345±0.013**	**1.529±0.094**	**0.818±0.023**

in Table 3. Here, since *LSTM* is a deterministic method, the $CRPS_{sum}$ metric is not applicable for it. From the results, we can see each component contributes to the performance improvement. Our proposed StochDiff takes advantages of all these components and exhibits better modelling of stochastic time series data.

4.5 Case Study on Cochlear Implant Surgery

Real-world application has become a crucial benchmark to evaluate the practical benefits of machine learning models. While many models achieve state-of-the-art results on various datasets, their applicability in real-world scenarios often remains an open question. In this section, we aim at bridging this gap for our model by simulating the Cochlear Implant Surgery process and evaluating the model's ability to assist with the operation.

The ECochG dataset comprises inner ear responses during Cochlear Implant Surgery for each patient. The raw data, containing 218 channels, can be converted into a univariate signal called the 'Cochlear Microphonic' (CM). Drops larger than 30% in the amplitude of the CM signal has been proven to be capable of reflecting the damage to a patient's inner ear structure [3, 7]. Thus, researchers have developed models to monitor the CM signal and automatically detect these 'traumatic drops' [23].

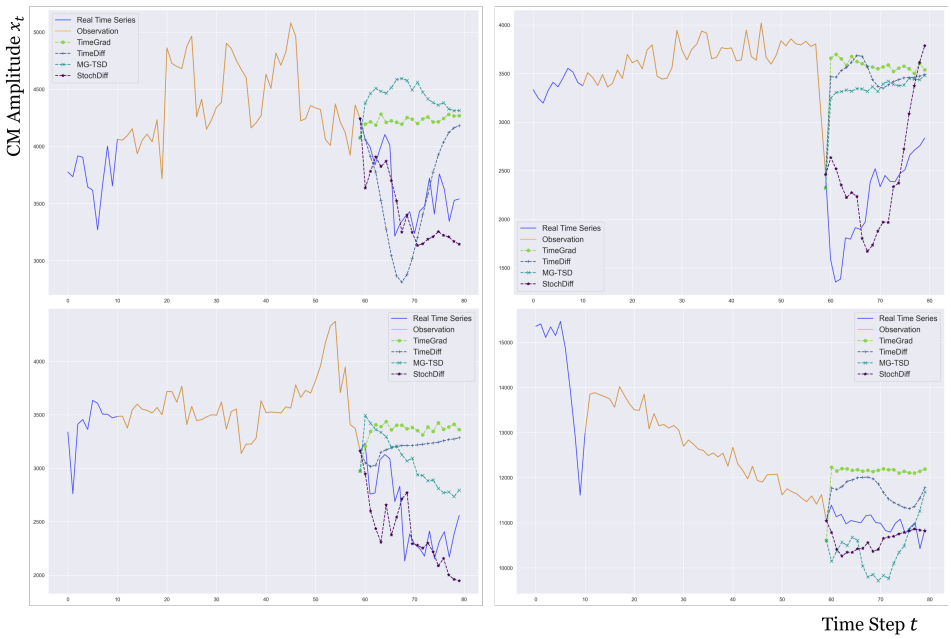


Figure 3: Cochlear Implant Forecasting Simulations. The blue line is the real CM amplitude, and the orange part is the observation that was fed to the model. Forecasting from different models are marked with differently according to the legend inside the plots.

Now, with forecasting models, we can attempt to predict these drops even before they occur. To simulate the data stream in real-world surgeries, we use a sliding window to sample the patient's data so that the model can only observe the data up to the current time step. Then, we let the model predict the future steps based on the historical observations. Additionally, considering that in real-world scenarios, the models are expected to work with new data, we use new patients data (out of the aforementioned 78 patients) to test the models' robustness. Figure 3 shows several cases where possible 'traumatic drops' occurred.

Here, we follow the same steps as [3, 7] to convert the predicted raw values into CM amplitudes, and compare with the real CM amplitudes. With these simulations, we can obtain more straightforward insights on the goodness of the models. While the baseline models show some capability to predict the 'traumatic drops', our StochDiff model can forecast these drops in all four cases. This result demonstrates our model's robustness on new data and showcases its potential in a clinical context, for use in Cochlear Implant Surgeries.

5 Limitations & Border Impacts

Although we demonstrated the strength of our proposed StochDiff, we acknowledge its limitations: (1) The current step-wise modelling and autoregressive prediction can be time consuming, a faster sampling method, such as the one used by Song et al. [22], and parallel sampling strategy could be implemented to speed up the forecasting process and reduce the latency of decision making. (2) The backbone of the current model is an LSTM which might fail to learn long term temporal dynamics. Other networks such as Transformers or Temporal Convolutional Networks could be used as a replacement or alongside to improve long term feature learning. Since our current model is only designed for research purposes, more comprehensive testing and validation are required to prepare it for use in real-world surgeries. It is also important to follow ethical practices when using generative models to avoid any negative human impact. E.g. closely integrating the model with human in the loop oversight.

6 Conclusion

In this paper, we proposed StochDiff, a novel diffusion based forecasting model for stochastic time series data. By applying step-wise modelling and data-driven prior knowledge, StochDiff was better able to model the temporal dynamics and uncertainty from highly stochastic time series data. Experimental results on six real-world datasets demonstrated the competitive performance of StochDiff when compared with state-of-the-art diffusion based time series forecasting models. Additionally, a case study on cochlear implant surgery showcased the model's ability for highly stochastic patient data.

References

[1] E. Aksan and O. Hilliges. STCN: Stochastic temporal convolutional networks. In *International Conference on Learning Representations*, 2019.

[2] J. L. Alcaraz and N. Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

[3] L. Campbell, A. Kaicer (Umansky), D. Sly, C. Iseli, B. Wei, R. Briggs, and S. O'Leary. Intraoperative real-time cochlear response telemetry predicts hearing preservation in cochlear implantation. *Otology & Neurotology*, 37:1, 2016.

[4] J. Cao, Z. Li, and J. Li. Financial time series forecasting model based on ceemdan and lstm. *Physica A: Statistical mechanics and its applications*, 519:127–139, 2019.

[5] V. K. R. Chimmula and L. Zhang. Time series forecasting of covid-19 transmission in canada using lstm networks. *Chaos, solitons & fractals*, 135:109864, 2020.

[6] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28, 2015.

[7] A. Dalbert, A. Huber, D. Veraguth, C. Roosli, and F. Pfiffner. Assessment of cochlear trauma during cochlear implantation using electrocochleography and cone beam computed tomography. *Otology & Neurotology*, 37(5):446–453, 2016.

[8] E. de Bézenac, S. S. Rangapuram, K. Benidis, M. Bohlke-Schneider, R. Kurle, L. Stella, H. Hasson, P. Gallinari, and T. Januschowski. Normalizing kalman filters for multivariate time series analysis. *Advances in Neural Information Processing Systems*, 33:2995–3007, 2020.

[9] X. Fan, Y. Wu, C. Xu, Y. Huang, W. Liu, and J. Bian. MG-TSD: Multi-granularity time series diffusion models with guided learning process. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] B. S. Freeman, G. Taylor, B. Gharabaghi, and J. Thé. Forecasting air quality time series using deep learning. *Journal of the Air & Waste Management Association*, 68(8):866–886, 2018.

[11] A. Grover, A. Kapoor, and E. Horvitz. A deep hybrid model for weather forecasting. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 379–386, 2015.

[12] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[13] D. Hu, L. Peng, T. Chu, X. Zhang, Y. Mao, H. Bondell, and M. Gong. Uncertainty quantification in depth estimation via constrained ordinal regression. In *European Conference on Computer Vision*, pages 237–256. Springer, 2022.

[14] D. H. Kerem and A. B. Geva. Forecasting epilepsy from the heart rate signal. *Medical and biological engineering and computing*, 43:230–239, 2005.

[15] D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2022.

[16] Y. Li, W. Chen, X. Hu, B. Chen, baolin sun, and M. Zhou. Transformer-modulated diffusion models for probabilistic multivariate time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

[17] J. Qiu, B. Wang, and C. Zhou. Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one*, 15(1):e0227222, 2020.

[18] K. Rasul, C. Seward, I. Schuster, and R. Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pages 8857–8868. PMLR, 2021.

[19] D. Salinas, M. Bohlke-Schneider, L. Callot, R. Medico, and J. Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes, 2019.

[20] L. Shen and J. Kwok. Non-autoregressive conditional diffusion models for time series prediction. In *International Conference on Machine Learning*, pages 31016–31029. PMLR, 2023.

[21] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[22] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[23] S. Wijewickrema, C. Bester, J.-M. Gerard, A. Collins, and S. O'Leary. Automatic analysis of cochlear response using electrocochleography signals during cochlear implant surgery. *Plos one*, 17(7), 2022.

A Math Formulation

During the reverse diffusion process, when we introduce an additional latent variable into the prior distribution, we have a prior distribution conditioned on $\boldsymbol{z}$, and the equation 1 becomes:

$$p(\boldsymbol{x}^{0:N}) = p(\boldsymbol{x}^N|\boldsymbol{z}) \prod_{n=1}^{N} p_\theta(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n, \boldsymbol{z}) p(\boldsymbol{z}). \tag{18}$$

By integrating out all other variables $\boldsymbol{x}^{1:N}$, we can obtain the data distribution $p(\boldsymbol{x}^0)$:

$$p(\boldsymbol{x}^0) = \int_{\boldsymbol{x}^{1:N}} \int_{\boldsymbol{z}} p(\boldsymbol{x}^N|\boldsymbol{z}) \prod_{n=1}^{N} p(\boldsymbol{x}^{n-1}|\boldsymbol{x}^n, \boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z} d\boldsymbol{x}^{1:N}. \tag{19}$$

The equation 19 defines a latent variable reverse diffusion process, which we call *variational diffusion*. By introducing the time series indexes and prior distribution (equation 11) into the *variational diffusion*, we have the RHS of equation 14:

$$\int_{\boldsymbol{x}_{t_0:T}^{1:N}} \int_{\boldsymbol{z}_{t_0:T}} \prod_{t=t_0}^{T} p(\boldsymbol{x}_t^N|\boldsymbol{z}_t) \prod_{n=1}^{N} p_\theta(\boldsymbol{x}_t^{n-1}|\boldsymbol{x}_t^n, \boldsymbol{z}_t) p_z(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}^0, \boldsymbol{z}_{1:t-1}) d\boldsymbol{z}_{t_0:T} d\boldsymbol{x}_{t_0:T}^{1:N}. \tag{20}$$

Following the derivation below, we proved equation 20 is the valid formulation of the generative process of *variational diffusion* on *time series forecasting* task, and we name it Stochastic Diffusion Model.

$$\begin{aligned}
&\int_{x_{t_0:T}^{1:N}} \int_{z_{t_0:T}} \prod_{t=t_0}^{T} \left(p(x_t^N|z_t) \prod_{n=1}^{N} p(x_t^{n-1}|x_t^n, z_t) \right) p(z_t|x_{1:t-1}^0, z_{1:t-1}) dz_{t_0:T} dx_{t_0:T}^{1:N} \\
&= \int_{x_{t_0:T}^{1:N}} \left(\int_{z_{t_0:T}} \prod_{t=t_0}^{T} p(x_t^{0:N}|z_t) p(z_t|x_{1:t-1}^0, z_{1:t-1}) dz_{t_0:T} \right) dx_{t_0:T}^{1:N} \\
&= \int_{x_{t_0:T}^{1:N}} \left(\int_{z_{t_0:T}} p(x_{t_0:T}^{0:N}, z_{t_0:T}|x_{1:t_0-1}^0, z_{1:t_0-1}) dz_{t_0:T} \right) dx_{t_0:T}^{1:N} \\
&= \int_{x_{t_0:T}^{1:N}} p(x_{t_0:T}^{0:N}|x_{1:t_0-1}^0, z_{1:t_0-1}) dx_{t_0:T}^{1:N} \\
&= p(x_{t_0:T}^0|x_{1:t_0-1}^0, z_{1:t_0-1})
\end{aligned}$$

B Datasets Details

- *Exchange*[2], which records the daily exchange rate of 22 countries (to US dollars) from 2000 to 2019.
- *Weather*[3], which consists of records of 21 meteorological reads at 10-minute intervals from 2022 to 2023.
- *Traffic*[4], which contains the collection of hourly road occupancy rates on San Francisco Bay area freeways from 2015 to 2016.
- *Solar*[5], which contains solar power production records in year of 2006, data are collected from 56 sensors at Texas State with 5-minute intervals.

C Network Details & Experiment Setup

The proposed StochDiff consists of several components that responsible for various learning tasks. The Recurrent backbone (*LSTM*) is the basis of time series modelling: it takes time series data $\boldsymbol{x}_t$ at

[2]https://www.kaggle.com/datasets/brunotly/foreign-exchange-rates-per-dollar-20002019/data

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

[4]https://github.com/laiguokun/multivariate-time-series-data/tree/master/traffic

[5]https://www.nrel.gov/grid/solar-power-data.html

time t and its corresponding prior knowledge $\boldsymbol{z}_{t-1}$ as inputs, then extracts the temporal dynamics and stores them inside 128-length hidden units h_t. The prior knowledge is learnt via a Fully Connected Network (FCN) that takes $\boldsymbol{x}_t$ and h_{t-1} as inputs and maps them into a 128-length vector $\boldsymbol{z}_t$. In the meantime, the approximate posterior is estimated with another FCN that takes h_{t-1} as the only input. The diffusion module is responsible for the data reconstruction and generation. The forward diffusion process is mainly involved with mathematical calculations. And during the reverse diffusion process, a data prediction network *Attention-Net* is designed for multivariate time series modelling. Since the temporal dynamics is captured during the recurrent and the prior learning process, the *Attention-Net* here is mostly responsible for learning the correlation between each dimension of $\boldsymbol{x}_t$ via its self-attention mechanism. Then, a cross-attention extracts the interaction between two learning outcomes, and properly combines them for the final data reconstruction.

The model is trained using the Algorithm 1 provided in main text. Noted here, to enhance the optimization process, we reduce the learning rate by 50% whenever loss stops dropping for 10 epochs (patient time). To simulate a real-time forecasting scenario, after dividing the datasets with the train-test split, we sample the data using a sliding window with varying size. The model are trained to model the sampled time series data in the training set. Then, during the test time, the model first update the hidden units based on the observations (inside windows) and forecast the future data with specified duration. The statistical details are listed in Table 4.

Table 4: Statistical details of the datasets. Values in parenthesis represents the number of sampled data points.

Datasets	Window Size	Forecasting	Dimension
Exchange	3 months (90)	1 week (7)	22
Weather	1 day (144)	1 day (144)	21
Traffic	4 days (96)	1 day (24)	862
Solar	4 days (96)	1/2 day (12)	56
ECochG	1/2 minute (50)	7 seconds (10)	218
MMG	3 seconds (100)	0.3 second (10)	148

All experiments are undertaken using a High Performance Computer (HPC) with Intel Xeon Gold 6326 CPU (2.90GHz) and NVIDIA A100 GPU (80G).

D Evaluation Metrics

The $NRMSE$ firstly calculates the mean squared error which is the squared differences between estimated future values and the real values. Then it takes the square root of the results and normalize it using the mean of real data to facilitates the different scales of different datasets:

$$NRMSE := \frac{\sqrt{\mathbb{E}((\boldsymbol{x} - \hat{\boldsymbol{x}}))}}{\mathbb{E}(\boldsymbol{x})}.$$

The $CRPS$ measures the quality of the predicted distribution by comparing the Cumulative Distribution Function (CDF) F of the predictions against the real data $x \in \mathbb{R}$:

$$CRPS(F, x) := \int_{\mathbb{R}} (F(y) - \mathbb{I}(x \leq y))^2 dy$$

where $\mathbb{I}(x \leq y)$ is an indicator function, $\mathbb{I} = 1$ when $x \leq y$, and $\mathbb{I} = 0$ otherwise. Following Salinas et al. [19], we calculate the $CRPS_{sum}$ for multivariate time series data by summing sampled and real data across dimensions and then averaging over the prediction horizon:

$$CRPS_{sum} = \mathbb{E}_t \left[CRPS(\hat{F}_{sum}(y), \sum_{i=1}^{d} x_t^{(i)}) \right]$$

where $\hat{F}_{sum}(y) = \frac{1}{S} \sum_{s=1}^{S} \mathbb{I}(x_s \leq y)$ is the empirical CDF with S samples used to approximate the predictive CDF. This score has be proved to be a proper score for multivariate time series data [8].

E Visual Results

Figure 4 provides more visual results on ECochG dataset from *StochDiff*, *MG-TSD*, and *TimeGrad*.

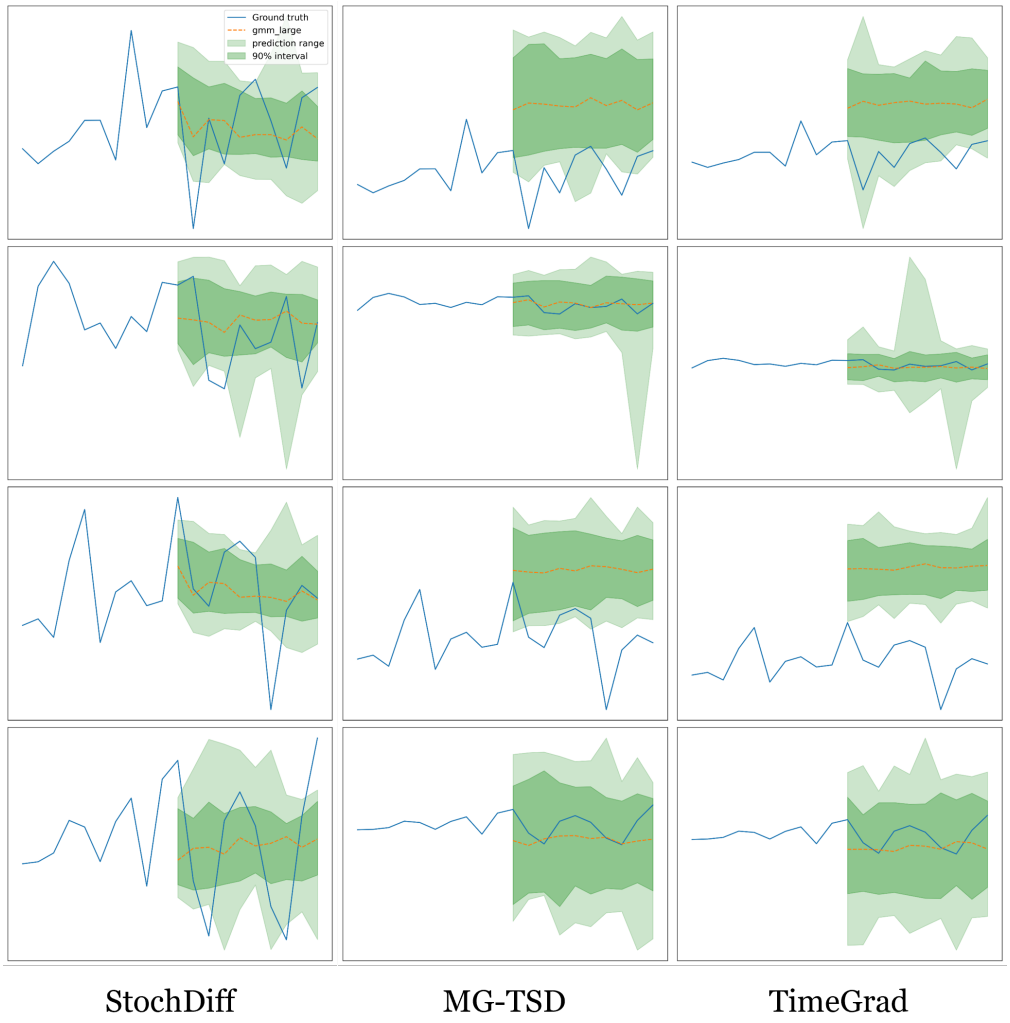


Figure 4: Visual Results on ECochG dataset.