TimeGrad

2025년 7월 29일 화요일 오후 5:45

TimeGrad

Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting

Kashif Rasul [1] **Calvin Seward** [1] **Ingmar Schuster** [1] **Roland Vollgraf** [1]

Abstract

In this work, we propose TimeGrad, an autoregressive model for multivariate probabilistic time series forecasting which samples from the data distribution at each time step by estimating its gradient. To this end, we use diffusion probabilistic models, a class of latent variable models closely connected to score matching and energy-based methods. Our model learns gradients by optimizing a variational bound on the data likelihood and at inference time converts white noise into a sample of the distribution of interest through a Markov chain using Langevin sampling. We demonstrate experimentally that the proposed autoregressive denoising diffusion model is the new state-of-the-art multivariate probabilistic forecasting method on real-world data sets with thousands of correlated dimensions. We hope that this method is a useful tool for practitioners and lays the foundation for future research in this area.

1. Introduction

Classical time series forecasting methods such as those in (Hyndman & Athanasopoulos, 2018) typically provide univariate point forecasts and are trained individually on each time series in a data set which does not scale with millions of series. Deep learning based time series models (Benidis et al., 2020) are popular alternatives due to their end-to-end training of a global model, ease of incorporating exogenous covariates, and automatic feature extraction abilities. The task of modeling uncertainties is of vital importance for downstream problems that use these forecasts for (business) decision making. More often the individual time series for a problem data set are statistically dependent on each other. Ideally, deep learning models need to incorporate this inductive bias in the form of multivariate (Tsay, 2014) probabilistic methods to provide accurate forecasts.

To model the full predictive distribution, methods typically resort to tractable distribution classes or some type of low-rank approximations, regardless of the true data distribution. To model the distribution in a general fashion, one needs probabilistic methods with tractable likelihoods. Till now several deep learning methods have been proposed for this purpose such as autoregressive (van den Oord et al., 2016c) or generative ones based on normalizing flows (Papamakarios et al., 2021) which can learn flexible models of high dimensional multivariate time series. Even if the full likelihood is not tractable, one can often optimize a tractable lower bound to the likelihood. But still, these methods require a certain structure in the functional approximators, for example on the determinant of the Jacobian (Dinh et al., 2017) for normalizing flows. *Energy-based models* (EBM) (Hinton, 2002; LeCun et al., 2006) on the other hand have a much less restrictive functional form. They approximate the unnormalized log-probability so that density estimation reduces to a non-linear regression problem. EBMs have been shown to perform well in learning high dimensional data distributions at the cost of being difficult to train (Song & Kingma, 2021).

In this work, we propose autoregressive EBMs to solve the multivariate probabilistic time series forecasting problem via a model we call TimeGrad and show that not only are we able to train such a model with all the inductive biases of probabilistic time series forecasting, but this model performs exceptionally well when compared to other modern methods. This autoregressive-EBM combination retains the power of autoregressive models, such as good performance in extrapolation into the future, with the flexibility of EBMs as a general purpose high-dimensional distribution model, while remaining computationally tractable.

The paper is organized as follows. In Section 2 we first set up the notation and detail the EBM of (Ho et al., 2020) which forms the basis of our distribution model. Section 3 introduces the multivariate probabilistic time series problem and we detail the TimeGrad model. The experiments with extensive results are detailed in Section 4. We cover related work in Section 5 and conclude with some discussion in Section 6.

[1]Zalando Research, Mühlenstraße 25, 10243 Berlin, Germany. Correspondence to: Kashif Rasul <kashif.rasul@zalando.de>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021.

/

Abstract

TimeGrad, an autoregressive model for multivariate probabilistic time series forecasting which samples from the data distribution at each time step by estimating its gradient.
직역: TimeGrad는/ 하나의/ autoregressive(자가회귀)/ 모델이다/ 다변량/ 확률적/ 시계열/ 예측을 위한/ 그리고/ 샘플링한다/ 데이터/ 분포로부터/ 각/ 시간/ 단계에서/ 그것의/ 그래디언트를/ 추정함으로써.
의역: TimeGrad는 다변량 확률 시계열 예측을 위해 각 시간 단계마다 그래디언트를 추정하여 데이터 분포에서 샘플링하는 자가회귀 모델이다.
To this end, we use diffusion probabilistic models, a class of latent variable models closely connected to score matching and energy-based methods.
직역: 이/ 목적을 위해,/ 우리는/ 사용한다/ diffusion(확산)/ 확률/ 모델들을,/ 하나의/ 클래스의/ 잠재/ 변수/ 모델들/ 밀접하게/ 연결된/ score matching(스코어 정합)/ 및/ energy-based methods(에너지 기반 기법)과.
의역: 이를 위해 우리는 score matching과 에너지 기반 기법과 밀접한 연관이 있는 잠재 변수 모델의 일종인 확산 확률 모델을 사용한다.
Our model learns gradients by optimizing a variational bound on the data likelihood and at inference time converts white noise into a sample of the distribution of interest through a Markov chain using Langevin sampling.
직역: 우리의/ 모델은/ 학습한다/ 그래디언트를/ 데이터/ 우도에 대한/ 변분/ 경계를/ 최적화함으로써/ 그리고/ 추론/ 시점에/ 변환한다/ 백색/ 잡음을/ 관심/ 분포의/ 하나의/ 샘플로/ 마르코프/ 체인을/ 통해/ Langevin(랑주뱅)/ 샘플링을/ 사용하여.
의역: 본 모델은 데이터의 우도에 대한 변분 경계를 최적화하여 그래디언트를 학습하고, 추론 시에는 랑주뱅 샘플링을 활용한 마르코프 체인을 통해 백색 잡음을 원하는 분포의 샘플로 변환한다.
We demonstrate experimentally that the proposed autoregressive denoising diffusion model is the new state-of-the-art multivariate probabilistic forecasting method on real-world data sets with thousands of correlated dimensions.
직역: 우리는/ 실험적으로/ 입증한다/ 제안된/ autoregressive/ denoising/ diffusion/ 모델이/ 새로운/ state-of-the-art(최신)/ 다변량/ 확률적/ 예측/ 방법임을/ 실제/ 데이터/ 셋들/ 위에서/ 수천의/ 상관된/ 차원을/ 가진.
의역: 우리는 제안된 자가회귀 디노이징 확산 모델이 수천 개의 상관된 차원을 가진 실제 데이터셋에서 최신 성능을 보이는 다변량 확률 예측 방법임을 실험적으로 입증하였다.
We hope that this method is a useful tool for practitioners and lays the foundation for future research in this area.
직역: 우리는/ 희망한다/ 이/ 방법이/ 하나의/ 유용한/ 도구이기를/ 실무자들에게/ 그리고/ 놓기를/ 기반을/ 미래의/ 연구를 위한/ 이/ 영역에서.
의역: 우리는 이 방법이 실무자들에게 유용한 도구가 되며, 이 분야의 향후 연구를 위한 기반이 되기를 바란다.

2. Diffusion Probabilistic Model

At a very high level the diffusion models sample from a distribution by a gradual reversal of a noising process, starting from white noise to the actual signal. These models learn to produce a slightly less noisy signal at each step of the process and as we will see the training loss can be expressed as a regression problem when we assume that the denoising process is given by a learned diagonal Gaussian.

Formally, let $\mathbf{x}^0 \sim q_{\mathcal{X}}(\mathbf{x}^0)$ denote a multivariate training vector from some input space $\mathcal{X} = \mathbb{R}^D$ and let $p_\theta(\mathbf{x}^0)$ denote the probability density function (PDF) which aims to approximate $q_{\mathcal{X}}(\mathbf{x}^0)$ and allows for easy sampling. Diffusion models (Sohl-Dickstein et al., 2015) are latent variable models of the form $p_\theta(\mathbf{x}^0) := \int p_\theta(\mathbf{x}^{0:N})\, \mathrm{d}\mathbf{x}^{1:N}$, where $\mathbf{x}^1, \ldots, \mathbf{x}^N$ are latents of dimension $\mathbb{R}^D$. Unlike in variational autoencoders (Kingma & Welling, 2019) the approximate posterior $q(\mathbf{x}^{1:N}|\mathbf{x}^0)$ denoted as:

$$q(\mathbf{x}^{1:N}|\mathbf{x}^0) = \Pi_{n=1}^N q(\mathbf{x}^n|\mathbf{x}^{n-1})$$

is not trainable. Rather it's fixed to a Markov chain (called the *forward* process) that gradually adds Gaussian noise to the signal:

$$q(\mathbf{x}^n|\mathbf{x}^{n-1}) := \mathcal{N}(\mathbf{x}^n; \sqrt{1-\beta_n}\mathbf{x}^{n-1}, \beta_n \mathbf{I}),$$

via a given increasing variance schedule $\beta_1, \ldots, \beta_N$ with $\beta_n \in (0, 1)$. The joint distribution $p_\theta(\mathbf{x}^{0:N})$ is called the *reverse* process, and is defined as a Markov chain with learned Gaussian transitions starting with $p(\mathbf{x}^N) = \mathcal{N}(\mathbf{x}^N; \mathbf{0}, \mathbf{I})$, where each subsequent transition of

$$p_\theta(\mathbf{x}^{0:N}) := p(\mathbf{x}^N)\Pi_{n=N}^1 p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n)$$

is given by a parametrization of our choosing denoted by

$$p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n) := \mathcal{N}(\mathbf{x}^{n-1}; \mu_\theta(\mathbf{x}^n, n), \sigma_\theta(\mathbf{x}^n, n)\mathbf{I}), \quad (1)$$

with shared parameters θ. Both $\mu_\theta : \mathbb{R}^D \times \mathbb{N} \to \mathbb{R}^D$ and $\sigma_\theta : \mathbb{R}^D \times \mathbb{N} \to \mathbb{R}^+$ take two inputs, namely the variable $\mathbf{x}^n \in \mathbb{R}^D$ as well as the noise index $n \in \mathbb{N}$. The goal of $p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n)$ is to eliminate the Gaussian noise added in the diffusion process. The parameters θ are learned to fit the data distribution $q_{\mathcal{X}}(\mathbf{x}^0)$ by minimizing the negative log-likelihood via a variational bound using Jensen's inequality:

$$\min_\theta \mathbb{E}_{q(\mathbf{x}^0)}[-\log p_\theta(\mathbf{x}^0)] \leq$$
$$\min_\theta \mathbb{E}_{q(\mathbf{x}^{0:N})}[-\log p_\theta(\mathbf{x}^{0:N}) + \log q(\mathbf{x}^{1:N}|\mathbf{x}^0)].$$

This upper bound can be shown to be equal to

$$\min_\theta \mathbb{E}_{q(\mathbf{x}^{0:N})}\left[-\log p(\mathbf{x}^N) - \sum_{n=1}^N \log \frac{p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n)}{q(\mathbf{x}^n|\mathbf{x}^{n-1})}\right]. \quad (2)$$

As shown by (Ho et al., 2020), a property of the forward process is that it admits sampling $\mathbf{x}^n$ at any arbitrary noise level n in closed form, since if $\alpha_n := 1 - \beta_n$ and $\bar{\alpha}_n := \Pi_{i=1}^n \alpha_i$ its cumulative product, we have:

$$q(\mathbf{x}^n|\mathbf{x}^0) = \mathcal{N}(\mathbf{x}^n; \sqrt{\bar{\alpha}_n}\mathbf{x}^0, (1-\bar{\alpha}_n)\mathbf{I}). \quad (3)$$

By using the fact that these processes are Markov chains, the objective in (2) can be written as the KL-divergence between Gaussian distributions:

$$-\log p_\theta(\mathbf{x}^0|\mathbf{x}^1) + D_{\mathrm{KL}}(q(\mathbf{x}^N|\mathbf{x}^0)||p(\mathbf{x}^N)) + \sum_{n=2}^N D_{\mathrm{KL}}(q(\mathbf{x}^{n-1}|\mathbf{x}^n, \mathbf{x}^0)||p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n)), \quad (4)$$

and (Ho et al., 2020) shows that by the property (3) the forward process posterior in these KL divergences when conditioned on $\mathbf{x}^0$, i.e. $q(\mathbf{x}^{n-1}|\mathbf{x}^n, \mathbf{x}^0)$ are tractable given by

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

where

$$\tilde{\mu}_n(\mathbf{x}^n, \mathbf{x}^0) := \frac{\sqrt{\alpha_{n-1}}\beta_n}{1-\alpha_n}\mathbf{x}^0 + \frac{\sqrt{\alpha_n}(1-\alpha_{n-1})}{1-\alpha_n}\mathbf{x}^n$$

and

$$\tilde{\beta}_n := \frac{1-\bar{\alpha}_{n-1}}{1-\alpha_n}\beta_n. \quad (5)$$

Further, (Ho et al., 2020) shows that the KL-divergence between Gaussians can be written as:

$$D_{\mathrm{KL}}(q(\mathbf{x}^{n-1}|\mathbf{x}^n, \mathbf{x}^0)||p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n)) = \mathbb{E}_q\left[\frac{1}{2\sigma_\theta}\|\tilde{\mu}_n(\mathbf{x}^n, \mathbf{x}^0) - \mu_\theta(\mathbf{x}^n, n)\|^2\right] + C, \quad (6)$$

where C is a constant which does not depend on θ. So instead of a parametrization (1) of p_θ that predicts $\tilde{\mu}$, one can instead use the property (3) to write $\mathbf{x}^n(\mathbf{x}^0, \epsilon) = \sqrt{\bar{\alpha}_n}\mathbf{x}^0 + \sqrt{1-\bar{\alpha}_n}\epsilon$ for $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ together with the formula for $\tilde{\mu}$ to obtain that μ_θ must predict $(\mathbf{x}^n - \beta_n\epsilon/\sqrt{1-\bar{\alpha}_n})/\sqrt{\alpha_n}$. But, since $\mathbf{x}^n$ is available to the network, we can choose:

$$\mu_\theta(\mathbf{x}^n, n) := \frac{1}{\sqrt{\alpha_n}}\left(\mathbf{x}^n - \frac{\beta_n}{\sqrt{1-\bar{\alpha}_n}}\epsilon_\theta(\mathbf{x}^n, n)\right),$$

so that the objective in (6) simplifies to:

$$\mathbb{E}_{\mathbf{x}^0, \epsilon}\left[\frac{\beta_n^2}{2\sigma_\theta\alpha_n(1-\bar{\alpha}_n)}\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_n}\mathbf{x}^0 + \sqrt{1-\bar{\alpha}_n}\epsilon, n)\|^2\right] \quad (7)$$

resembling the loss in Noise Conditional Score Networks (Song & Ermon, 2019; 2020) using score matching. Thus we see that ϵ_θ is a network that needs to predict Gaussian noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, trained via the MSE loss between the true noise and predicted noise. Once trained, to sample from the reverse process $\mathbf{x}^{n-1} \sim p_\theta(\mathbf{x}^{n-1}|\mathbf{x}^n)$ (1) we can predict the noise for any stage of the process and compute

$$\mathbf{x}^{n-1} = \frac{1}{\sqrt{\alpha_n}}\left(\mathbf{x}^n - \frac{\beta_n}{\sqrt{1-\bar{\alpha}_n}}\epsilon_\theta(\mathbf{x}^n, n)\right) + \sqrt{\sigma_\theta}\mathbf{z}$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ for $n = N, \ldots, 2$ and $\mathbf{z} = \mathbf{0}$ when $n = 1$. The full sampling procedure for $\mathbf{x}^0$, starting from white noise sample $\mathbf{x}^N$, resembles Langevin dynamics where we sample from the most noise-perturbed distribution and reduce the magnitude of the noise scale until we reach the smallest one.

3. `TimeGrad` Method

We denote the entities of a multivariate time series by $x_{i,t}^0 \in \mathbb{R}$ for $i \in \{1, \ldots, D\}$ where t is the time index. Thus the multivariate vector at time t is given by $\mathbf{x}_t^0 \in \mathbb{R}^D$. We are tasked with predicting the multivariate distribution some given prediction time steps into the future and so in what follows consider time series with $t \in [1, T]$, sampled from the complete time series history of the training data. We will split this contiguous sequence into a context window sized interval $[1, t_0)$ and given prediction length sized interval $[t_0, T]$, reminiscent of seq-to-seq models (Sutskever et al., 2014) in language modeling.

In the univariate probabilistic `DeepAR` model (Salinas et al., 2019b), the log-likelihood of each entity $x_{i,t}^0$ at a time step $t \in [t_0, T]$ is maximized over an individual time series' prediction window. This is done with respect to the parameters of some chosen distributional model via the state of an recurrent neural network (RNN) derived from its previous time step $x_{i,t-1}^0$ and the current time covariates $\mathbf{c}_{i,t}$. The emission distribution model, which is typically Gaussian for real-valued data or negative binomial for count data, is selected to best match the statistics of the time series and the network incorporates activation functions that satisfy the constraints of the distribution's parameters, e.g. a `softplus()` for the scale parameter of the Gaussian.

A straightforward time series model for multivariate real-valued data could take the full multivariate vector $\mathbf{x}_t$ and covariates as inputs to the RNN and use a factorizing output distribution. Shared parameters can then learn patterns across the individual time series entities through the temporal component—but the model falls short of capturing dependencies in the emissions of the model. For this, a full joint distribution at each time step has to be modeled, for example by using a multivariate Gaussian. However, modeling the full covariance matrix not only increases the number of parameters of the neural network by $O(D^2)$ fects. Approximating Gaussians with diagonal or low-rank covariance matrices give strong baselines and these models are referred to as `Vec-LSTM` in (Salinas et al., 2019a).

Instead, in this work we propose `TimeGrad` which aims to learn a model of the conditional distribution of the future time steps of a multivariate time series given its past and covariates as:

$$q_{\mathcal{X}}(\mathbf{x}_{t_0:T}^0|\mathbf{x}_{1:t_0-1}^0, \mathbf{c}_{1:T}) = \Pi_{t=t_0}^T q_{\mathcal{X}}(\mathbf{x}_t^0|\mathbf{x}_{1:t-1}^0, \mathbf{c}_{1:T}), \quad (8)$$

were we assume that the covariates are known for all the time points and each factor is learned via a *conditional* denoising diffusion model introduced above. To model the temporal dynamics we employ the RNN architecture from (Graves, 2013; Sutskever et al., 2014) which utilizes the LSTM (Hochreiter & Schmidhuber, 1997) or GRU (Chung et al., 2014) to encode the time series sequence up to time point $t-1$, given the covariates of the next time point $\mathbf{c}_t$, via the updated hidden state $\mathbf{h}_{t-1}$:

$$\mathbf{h}_{t-1} = \mathrm{RNN}_\theta(\mathrm{concat}(\mathbf{x}_{t-1}^0, \mathbf{c}_t), \mathbf{h}_{t-2}), \quad (9)$$

where RNN_θ is a multi-layer LSTM or GRU parameterized by shared weights θ and $\mathbf{h}_0 = \mathbf{0}$. Thus we can approximate (8) by the `TimeGrad` model

$$\Pi_{t=t_0}^T p_\theta(\mathbf{x}_t^0|\mathbf{h}_{t-1}), \quad (10)$$

where θ now comprises the weights of the RNN as well as the denoising diffusion model. This model is autoregressive as it consumes the observations at the time step $t-1$ as input to learn the distribution of, or sample, the next time step as shown in Figure 1.

3.1. Training

Training is performed by randomly sampling context and adjoining prediction length sized windows from the training time series data and optimizing the parameters θ that minimize the negative log-likelihood of the model (10):

$$\sum_{t=t_0}^T -\log p_\theta(\mathbf{x}_t^0|\mathbf{h}_{t-1}),$$

starting with the hidden state $\mathbf{h}_{t_0-1}$ obtained by running the RNN on the chosen context window. Via a similar derivation as in the previous section, we have that the conditional variant of the objective (4) for time step t and noise level n is given by the following simplification of (7) (Ho et al., 2020):

$$\mathbb{E}_{\mathbf{x}_t^0, \epsilon, n}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_n}\mathbf{x}_t^0 + \sqrt{1-\bar{\alpha}_n}\epsilon, \mathbf{h}_{t-1}, n)\|^2\right], \quad (11)$$

when we choose the variance in (1) to be $\sigma_\theta := \tilde{\beta}_n$ (5)

---

2. 확산 확률 모델(Diffusion Probabilistic Model)

아주 높은 수준에서 보면, 확산 모델은 노이즈가 추가되는 과정을 점진적으로 역전시키며 어떤 분포로부터 샘플을 생성하는 방식이다. 이 모델들은 과정의 각 단계에서 점점 덜 노이즈가 낀 신호를 생성하도록 학습하며, 디노이징 과정이 학습된 대각 가우시안(Gaussian)을 따른다고 가정하면, 학습 손실은 회귀 문제로 표현될 수 있다.

형식적으로, X_0 ~ q_X(X_0)는 입력 공간 X = ℝᴰ로부터의 다변량 학습 벡터를 나타내며, p_θ(X_0)는 q_X(X_0)를 근사하고 샘플링이 용이하도록 하는 확률 밀도 함수(PDF)를 나타낸다. 확산 모델(Sohl−Dickstein et al., 2015)은 다음과 같은 형태의 잠재 변수 모델이다:

p_θ(X_0) := ∫ p_θ(X_0:N) dX_1:N

여기서 X_1, ..., X_N은 차원이 ℝᴰ인 잠재 변수들이다. 변분 오토인코더(variational autoencoders, Kingma & Welling, 2019)와는 달리, 근사 사후 분포 q(X_1:N|X_0)는 다음과 같이 고정된 형태로 정의된다:

q(X_1:N|X_0) = Π_{n=1}^{N} q(X_n|X_{n−1})

이는 학습 불가능하며, 신호에 가우시안 노이즈를 점진적으로 추가하는 마르코프 연쇄로 고정된다(이를 순방향 과정이라 한다):

q(X_n|X_{n−1}) := N(X_n; √(1 − β_n) X_{n−1}, β_n I)

여기서 증가하는 분산 스케줄 β_1, ..., β_N ∈ (0, 1)이 주어진다. 결합 분포 p_θ(X_0:N)은 역방향 과정(reverse process)이라 불리며, 다음과 같은 학습된 가우시안 전이 구조의 마르코프 연쇄로 정의된다:

p_θ(X_0:N) := p(X_N) Π_{n=1}^{N} p_θ(X_{n−1}|X_n)

여기서 p(X_N) = N(X_N; 0, I)이고, 각 전이 확률은 다음과 같이 매개변수화된다:

p_θ(X_{n−1}|X_n) := N(X_{n−1}; μ_θ(X_n, n), σ_θ(X_n, n) I) (1)

이때 μ_θ : ℝᴰ × ℕ → ℝᴰ, σ_θ : ℝᴰ × ℕ → ℝ⁺는 입력 X_n ∈ ℝᴰ와 노이즈 인덱스 n ∈ ℕ를 입력으로 받는다. p_θ(X_{n−1}|X_n)의 목적은 확산 과정에서 추가된 가우시안 노이즈를 제거하는 것이다.

매개변수 θ는 다음과 같은 Jensen 부등식을 이용한 변분 경계(variational bound)를 최소화함으로써, 데이터 분포 q_X(X_0)를 적합하도록 학습된다:

min_θ E_{q(X_0)}[−log p_θ(X_0)] ≤ min_θ E_{q(X_0:N)}[−log p_θ(X_0:N) + log q(X_1:N|X_0)]

이 상계는 다음과 같이 쓸 수 있다:

min_θ E_{q(X_0:N)} [log p(X_N) − Σ_{n=1}^{N} log (p_θ(X_{n−1}|X_n) / q(X_n|X_{n−1}))]

(Ho et al., 2020)에 의해 보여진 바와 같이, 순방향 과정의 한 가지 성질은, 임의의 노이즈 레벨 n에 대해 X_n을 폐형식(closed−form)으로 샘플링할 수 있다는 점이다. 만약 α_n := 1 − β_n 이고, **ᾱ_n := Π_{i=1}^{n} α_i** (누적곱)이라면, 우리는 다음을 얻는다:

q(X_n | X_0) = N(X_n; √ᾱ_n X_0, (1 − ᾱ_n) I) (식 3)

이 과정들이 마르코프 체인이라는 사실을 사용하면, (식 2)의 목적함수는 다음과 같이 정리될 수 있다 — 가우시안 분포 간의 Kullback−Leibler 발산(KL 발산)으로 표현된다:

log p_θ(X_0 | X_1) + D_KL(q(X_N | X_0) || p(X_N)) + Σ_{n=2}^{N} D_KL(q(X_{n−1} | X_n, X_0) || p_θ(X_{n−1} | X_n)) (식 4)

그리고 (Ho et al., 2020)은 식 (3)의 성질에 따라, 위 KL 발산 항에 등장하는 순방향 과정의 사후분포 q(X_{n−1} | X_n, X_0)는 다음과 같이 폐형식으로 계산 가능하다고 보여준다:

q(X_{n−1} | X_n, X_0) = N(X_{n−1}; μ̃_n(X_n, X_0), σ̃_n I)

여기서

μ̃_n(X_n, X_0) := (√ᾱ_{n−1} * β_n) / (1 − ᾱ_n) * X_0 + (√α_n * (1 − ᾱ_{n−1})) / (1 − ᾱ_n) * X_n

σ̃_n := (1 − ᾱ_{n−1}) / (1 − ᾱ_n) * β_n (식 5)

또한, (Ho et al., 2020)은 두 개의 가우시안 분포 사이의 KL 발산을 다음과 같이 쓸 수 있다고 보여준다:

D_KL(q(X_{n−1} | X_n, X_0) || p_θ(X_{n−1} | X_n)) = E_q [1/(2σ̃_n²) ‖μ̃_n(X_n, X_0) − μ_θ(X_n, n)‖²) + C] (식 6)

여기서 C는 θ와 무관한 상수이다.

따라서 p_θ의 파라미터화를 μ̃_n를 예측하는 형태로 하지 않고, 식 (3)의 성질을 이용하여 다음과 같이 X_n을 샘플링할 수 있다:

X_n(X_0, ε) = √ᾱ_n X_0 + √(1 − ᾱ_n) ε , ε ~ N(0, I)

이를 μ̃_n의 식과 결합하면, 네트워크 μ_θ는 다음 값을 예측해야 함을 알 수 있다:

(X_n − √(1 − ᾱ_n) ε) / √ᾱ_n

그러나 X_n은 네트워크의 입력으로 제공되므로, 우리는 다음과 같이 정의할 수 있다:

μ_θ(X_n, n) := 1 / √α_n * (X_n − √(1 − ᾱ_n) ε_θ(X_n, n))

이에 따라 (식 6)의 목적함수는 다음과 같이 단순화된다:

E_{X_0, ε} [(1 / 2σ_n²) * α_n * (1 − ᾱ_n) * ‖ε − ε_θ(√ᾱ_n X_0 + √(1 − ᾱ_n) ε , n)‖²] (식 7)

이 형태는 Noise Conditional Score Networks(Song & Ermon, 2019; 2020)에서의 score matching 손실과 유사한 구조를 가진다.

따라서 우리는 ε_θ가 가우시안 노이즈 ε ~ N(0, I)를 예측해야 하는 네트워크임을 알 수 있고, 이는 **실제 노이즈와 예측된 노이즈 간의 MSE 손실**을 통해 학습된다.

poral component—but the model falls short of capturing dependencies in the emissions of the model. For this, a full joint distribution at each time step has to be modeled, for example by using a multivariate Gaussian. However, modeling the full covariance matrix not only increases the number of parameters of the neural network by $O(D^2)$, making learning difficult but computing the loss is $O(D^3)$ making it impractical. Furthermore, statistical dependencies for such distributions would be limited to second-order ef-

variant of the objective (4) for time step t and noise level n is given by the following simplification of (7) (Ho et al., 2020):

$$\mathbb{E}_{\mathbf{x}_t^0,\epsilon,n}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_n}\mathbf{x}_t^0 + \sqrt{1-\bar{\alpha}_n}\epsilon, \mathbf{h}_{t-1}, n)\|^2\right], \tag{11}$$

when we choose the variance in (1) to be $\sigma_\theta := \tilde{\beta}_n$ (5), where now the ϵ_θ network is also *conditioned* on the hidden state. Algorithm 1 is the training procedure for each time step in the prediction window using this objective.

Algorithm 1 Training for each time series step $t \in [t_0, T]$

Input: data $\mathbf{x}_t^0 \sim q_{\mathcal{X}}(\mathbf{x}_t^0)$ and state $\mathbf{h}_{t-1}$
repeat
  Initialize $n \sim \text{Uniform}(1, \ldots, N)$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Take gradient step on

$$\nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_n}\mathbf{x}_t^0 + \sqrt{1-\bar{\alpha}_n}\epsilon, \mathbf{h}_{t-1}, n)\|^2$$

until converged

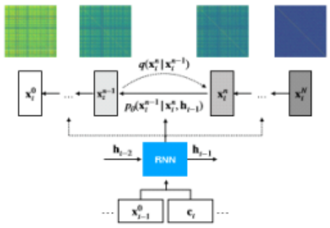

Figure 1. TimeGrad schematic: an RNN *conditioned* diffusion probabilistic model at some time $t-1$ depicting the fixed forward process that adds Gaussian noise to the signal and the learned reverse processes. The cross-correlation of the time series is also depicted during the stages of the process, from white noise (right) to the original signal (left).

3.2. Inference

After training, we wish to predict for each time series in our data set some prediction steps into the future and compare with the corresponding test set time series. As in training, we run the RNN over the last context sized window of the training set to obtain the hidden state $\mathbf{h}_T$ via (9). Then we follow the sampling procedure in Algorithm 2 to obtain a sample $\mathbf{x}_{T+1}^0$ of the next time step, which we can pass autoregressively to the RNN together with the covariates $\mathbf{c}_{T+2}$ to obtain the next hidden state $\mathbf{h}_{T+1}$ and repeat until the desired forecast horizon has been reached. This process of sampling trajectories from the "warm-up" state $\mathbf{h}_T$ can be repeated many times (e.g. $S = 100$) to obtain empirical quantiles of the uncertainty of our predictions.

3.3. Scaling

In real-world data, the magnitudes of different time series entities can vary drastically. To normalize scales, we divide each time series entity by their context window mean (or 1 if it's zero) before feeding it into the model. At inference,

Algorithm 2 Sampling $\mathbf{x}_t^0$ via annealed Langevin dynamics

Input: noise $\mathbf{x}_t^N \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and state $\mathbf{h}_{t-1}$
for $n = N$ **to** 1 **do**
  if $n > 1$ **then**
    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  else
    $\mathbf{z} = \mathbf{0}$
  end if
  $\mathbf{x}_t^{n-1} = \frac{1}{\sqrt{\alpha_n}}(\mathbf{x}_t^n - \frac{\beta_n}{\sqrt{1-\bar{\alpha}_n}}\epsilon_\theta(\mathbf{x}_t^n, \mathbf{h}_{t-1}, n)) + \sqrt{\sigma_\theta}\mathbf{z}$
end for
Return: $\mathbf{x}_t^0$

the samples are then multiplied by the same mean values to match the original scale. This rescaling technique simplifies the problem for the model, which is reflected in significantly improved empirical performance as shown in (Salinas et al., 2019b). The other method of a short-cut connection from the input to the output of the function approximator, as done in the multivariate point forecasting method LSTNet (Lai et al., 2018), did not give good results.

3.4. Covariates

We employ embeddings for categorical features (Charrington, 2018), that allows for relationships within a category, or its context, to be captured when training time series models. Combining these embeddings as features for forecasting yields powerful models like the first place winner of the Kaggle Taxi Trajectory Prediction[1] challenge (De Brébisson et al., 2015). The covariates $\mathbf{c}_t$ we use are composed of time-dependent (e.g. day of week, hour of day) and time-independent embeddings, if applicable, as well as lag features depending on the time frequency of the data set we are training on. All covariates are thus known for the periods we wish to forecast.

4. Experiments

We benchmark TimeGrad on *six* real-world data sets and evaluate against several competitive baselines. The source code of the model is available at https://github.com/zalandoresearch/pytorch-ts.

4.1. Evaluation Metric and Data Set

For evaluation, we compute the Continuous Ranked Probability Score (CRPS) (Matheson & Winkler, 1976) on each time series dimension, as well as on the sum of all time series dimensions (the latter denoted by CRPS_{sum}). CRPS measures the compatibility of a cumulative distribution func-

[1] https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i

tion (CDF) F with an observation x as

$$\mathrm{CRPS}(F, x) = \int_{\mathbb{R}} (F(z) - \mathbb{I}\{x \leq z\})^2 \, \mathrm{d}z,$$

where $\mathbb{I}\{x \leq z\}$ is the indicator function which is one if $x \leq z$ and zero otherwise. CRPS is a *proper scoring function*, hence CRPS attains its minimum when the predictive distribution F and the data distribution are equal. Employing the empirical CDF of F, i.e. $\hat{F}(z) = \frac{1}{S}\sum_{s=1}^{S} \mathbb{I}\{x^{0,s} \leq z\}$ with S samples $x^{0,s} \sim F$ as a natural approximation of the predictive CDF, CRPS can be directly computed from simulated samples of the conditional distribution (8) at each time point (Jordan et al., 2019). Finally, $\mathrm{CRPS}_{\mathrm{sum}}$ is obtained by first summing across the D time-series—both for the ground-truth data, and sampled data (yielding $\hat{F}_{\mathrm{sum}}(t)$ for each time point). The results are then averaged over the prediction horizon, i.e. formally

$$\mathrm{CRPS}_{\mathrm{sum}} = \mathbb{E}_t \left[\mathrm{CRPS}\left(\hat{F}_{\mathrm{sum}}(t), \sum_{i=1}^{D} x_{i,t}^0 \right) \right].$$

As proved in (de Bézenac et al., 2020) $\mathrm{CRPS}_{\mathrm{sum}}$ is also a proper scoring function and we use it, instead of likelihood based metrics, since not all methods we compare against yield analytical forecast distributions or likelihoods are not meaningfully defined.

For our experiments we use `Exchange` (Lai et al., 2018), `Solar` (Lai et al., 2018), `Electricity`[2], `Traffic`[3], `Taxi`[4] and `Wikipedia`[5] open data sets, preprocessed exactly as in (Salinas et al., 2019a), with their properties listed in Table 1. As can be noted in the table, we do not need to normalize scales for `Traffic`.

4.2. Model Architecture

We train `TimeGrad` via SGD using Adam (Kingma & Ba, 2015) with learning rate of 1×10^{-3} on the training split of each data set with $N = 100$ diffusion steps using a linear variance schedule starting from $\beta_1 = 1 \times 10^{-4}$ till $\beta_N = 0.1$. We construct batches of size 64 by taking random windows (with possible overlaps), with the context size set to the number of prediction steps, from the total time steps of each data set (see Table 1). For testing we use a rolling windows prediction starting from the last context window history before the start of the prediction and compare it to the ground-truth in the test set by sampling $S = 100$ trajectories.

[2] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[3] https://archive.ics.uci.edu/ml/datasets/PEMS-SF

[4] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[5] https://github.com/mbohlkeschneider/gluon-ts/tree/mv_release/datasets

Table 1. Dimension, domain, frequency, total training time steps and prediction length properties of the training data sets used in the experiments.

DATA SET	DIM. D	DOM.	FREQ.	TIME STEPS	PRED. STEPS
EXCHANGE	8	$\mathbb{R}^+$	DAY	6,071	30
SOLAR	137	$\mathbb{R}^+$	HOUR	7,009	24
ELEC.	370	$\mathbb{R}^+$	HOUR	5,833	24
TRAFFIC	963	$(0, 1)$	HOUR	4,001	24
TAXI	1,214	$\mathbb{N}$	30-MIN	1,488	24
WIKI.	2,000	$\mathbb{N}$	DAY	762	30

The RNN consists of 2 layers of an LSTM with the hidden state $\mathbf{h}_t \in \mathbb{R}^{40}$ and we encode the noise index $n \in \{1, \ldots, N\}$ using the Transformer's (Vaswani et al., 2017) Fourier positional embeddings, with $N_{\max} = 500$, into $\mathbb{R}^{32}$ vectors.

The network ϵ_θ, which needs to predict the noise injected in the noisy signal (11), consists of conditional 1-dim dilated ConvNets with residual connections adapted from the `WaveNet` (van den Oord et al., 2016a) and `DiffWave` (Kong et al., 2021) models. Figure 2 shows the schematics of a single residual block $b = \{0, \ldots, 7\}$ together with the final output from the sum of all the 8 skip-connections. All, but the last, convolutional network layers have an output channel size of 8 and we use a *bidirectional* dilated convolution in each block b by setting its dilation to $2^{b \% 2}$. We use a validation set from the training data of the same size as the test set to tune the number of epochs for early stopping.

All experiments run on a single Nvidia V100 GPU with 16GB of memory.

4.3. Results

Using the $\mathrm{CRPS}_{\mathrm{sum}}$ as an evaluation metric, we compare test time predictions of `TimeGrad` to a wide range of existing methods including classical multivariate methods:

- `VAR` (Lütkepohl, 2007) a multivariate linear vector auto-regressive model with lags corresponding to the periodicity of the data,
- `VAR-Lasso` a Lasso regularized `VAR`,
- `GARCH` (van der Weide, 2002) a multivariate conditional heteroskedastic model and
- `VES` a innovation state space model (Hyndman et al., 2008);

as well as deep learning based methods namely:

Table 2. Test set CRPS_{sum} comparison (lower is better) of models on six real world data sets. Mean and standard error metrics for `TimeGrad` obtained by re-training and evaluating 10 times.

Method	Exchange	Solar	Electricity	Traffic	Taxi	Wikipedia
VES	**0.005**±0.000	0.9±0.003	0.88±0.0035	0.35±0.0023	-	-
VAR	**0.005**±0.000	0.83±0.006	0.039±0.0005	0.29±0.005	0.292±0.000	3.4±0.003
VAR-Lasso	0.012±0.0002	0.51±0.006	0.025±0.0002	0.15±0.002	-	3.1±0.004
GARCH	0.023±0.000	0.88±0.002	0.19±0.001	0.37±0.0016	-	-
KVAE	0.014±0.002	0.34±0.025	0.051±0.019	0.1±0.005	-	0.095±0.012
Vec-LSTM ind-scaling	0.008±0.001	0.391±0.017	0.025±0.001	0.087±0.041	0.506±0.005	0.133±0.002
Vec-LSTM lowrank-Copula	0.007±0.000	0.319±0.011	0.064±0.008	0.103±0.006	0.326±0.007	0.241±0.033
GP scaling	0.009±0.000	0.368±0.012	0.022±0.000	0.079±0.000	0.183±0.395	1.483±1.034
GP Copula	0.007±0.000	0.337±0.024	0.0245±0.002	0.078±0.002	0.208±0.183	0.086±0.004
Transformer MAF	**0.005**±0.003	0.301±0.014	0.0207±0.000	0.056±0.001	0.179±0.002	0.063±0.003
TimeGrad	0.006±0.001	**0.287**±0.02	**0.0206**±0.001	**0.044**±0.006	**0.114**±0.02	**0.0485**±0.002

- `KVAE` (Fraccaro et al., 2017) a variational autoencoder to represent the data on top of a linear state space model which describes the dynamics,
- `Vec-LSTM-ind-scaling` (Salinas et al., 2019a) which models the dynamics via an RNN and outputs the parameters of an *independent* Gaussian distribution with mean-scaling,
- `Vec-LSTM-lowrank-Copula` (Salinas et al., 2019a) which instead parametrizes a low-rank plus diagonal covariance via Copula process,
- `GP-scaling` (Salinas et al., 2019a) which unrolls an LSTM with scaling on each individual time series before reconstructing the joint distribution via a low-rank Gaussian,
- `GP-Copula` (Salinas et al., 2019a) which unrolls an LSTM on each individual time series and then the joint emission distribution is given by a low-rank plus diagonal covariance Gaussian copula and
- `Transformer-MAF` (Rasul et al., 2021) which uses Transformer (Vaswani et al., 2017) to model the temporal conditioning and Masked Autoregressive Flow (Papamakarios et al., 2017) for the distribution emission model.

Table 2 lists the corresponding CRPS_{sum} values averaged over 10 independent runs together with their empirical standard deviations and shows that the `TimeGrad` model sets the new state-of-the-art on all but the smallest of the benchmark data sets. Note that flow based models must apply continuous transformations onto a continuously connected distribution, making it difficult to model disconnected modes. Flow models assign spurious density to connections between these modes leading to potential inaccuracies. Similarly the generator network in variational autoencoders must learn to map from some continuous space to a possibly disconnected space which might not be possible to learn. In contrast EBMs do not suffer from these issues (Du & Mordatch, 2019).

4.4. Ablation

The length N of the forward process is a crucial hyperparameter, as a bigger N allows the reverse process to be approximately Gaussian (Sohl-Dickstein et al., 2015) which assists the Gaussian parametrization (1) to approximate it better. We evaluate to which extent, if any at all, larger N affects prediction performance, with an ablation study where we record the test set CRPS_{sum} of the `Electricity` data set for different total diffusion process lengths $N = 2, 4, 8, \ldots, 256$ while keeping all other hyperparameters unchanged. The results are then plotted in Figure 3 where we note that N can be reduced down to ≈ 10 without significant performance loss. An optimal value is achieved at $N \approx 100$ and larger levels are not beneficial if all else is kept fixed.

To highlight the predictions of `TimeGrad` we show in Figure 4 the predicted median, 50% and 90% distribution intervals of the first 6 dimensions of the full 963 dimensional

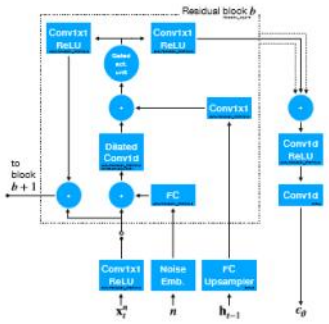

Figure 2. The network architecture of ϵ_θ consisting of `residual_layers` $= 8$ conditional residual blocks with the Gated Activation Unit $\sigma(\cdot) \odot \tanh(\cdot)$ from (van den Oord et al., 2016b); whose skip-connection outputs are summed up to compute the final output. `Conv1x1` and `Conv1d` are 1D convolutional layers with filter size of 1 and 3, respectively, circular padding so that the spatial size remains D, and all but the last convolutional layer has output channels `residual_channels` $= 8$. `FC` are linear layers used to up/down-sample the input to the appropriate size for broadcasting.

multivariate forecast of the `Traffic` benchmark.

5. Related Work

5.1. Energy-Based Methods

The EBM of (Ho et al., 2020) that we adapt is based on methods that learn the gradient of the log-density with respect to the *inputs*, called Stein Score function (Hyvärinen, 2005; Vincent, 2011), and at inference time use this gradient estimate via Langevin dynamics to sample from the model of this complicated data distribution (Song & Ermon, 2019). These models achieve impressive results for image generation (Ho et al., 2020; Song & Ermon, 2020) when trained in an unsupervised fashion without requiring adversarial optimization. By perturbing the data using multiple noise scales, the learnt Score network captures both coarse and fine-grained data features.

The closest related work to `TimeGrad` is in the recent non-autoregressive conditional methods `WaveGrad` (Chen et al., 2021) and `DiffWave` (Kong et al., 2021) for high fidelity waveform generation. Although these methods learn

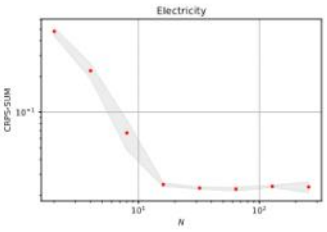

Figure 3. `TimeGrad` test set CRPS_{sum} for `Electricity` data by varying total diffusion length N. Good performance is established already at $N \approx 10$ with optimal value at $N \approx 100$. The mean and standard errors obtained over 5 independent runs. We see similar behaviour with other data sets.

the distribution of vector valued data via denoising diffusion methods, as done here, they do not consider its temporal development. Also neighboring dimensions of waveform data are highly correlated and have a uniform scale, which is not necessarily true for multivariate time series problems where neighboring entities occur arbitrarily (but in a fixed order) and can have different scales. (Du & Mordatch, 2019) also use EBMs to model one and multiple steps for a trajectory modeling task in an non-autoregressive fashion.

5.2. Time Series Forecasting

Neural time series methods have recently become popular ways of solving the prediction problem via univariate point forecasting methods (Oreshkin et al., 2020; Smyl, 2020) or univariate probabilistic methods (Salinas et al., 2019b). In the multivariate setting we also have point forecasting methods (Lai et al., 2018; Li et al., 2019) as well as probabilistic methods, like this method, which explicitly model the data distribution using Gaussian copulas (Salinas et al., 2019a), GANs (Yoon et al., 2019), or normalizing flows (de Bézenac et al., 2020; Rasul et al., 2021). Bayesian neural networks can also be used to provide *epistemic* uncertainty in forecasts as well as detect distributional shifts (Zhu & Laptev, 2018), although these methods often do not perform as well empirically (Wenzel et al., 2020).

6. Conclusion and Future Work

We have presented `TimeGrad`, a versatile multivariate probabilistic time series forecasting method that leverages the exceptional performance of EBMs to learn and sample

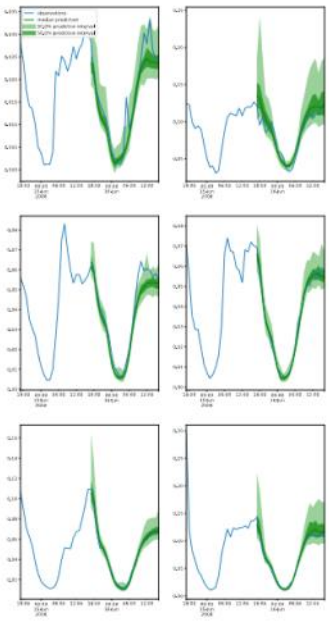

Figure 4. `TimeGrad` prediction intervals and test set ground-truth for `Traffic` data of the first 6 of 963 dimensions from first rolling-window. Note that neighboring entities have an order of magnitude difference in scales.

from the distribution of the next time step, autoregressively. Analysis of `TimeGrad` on six commonly used time series benchmarks establishes the new state-of-the-art against competitive methods.

We note that while training `TimeGrad` we do not need to loop over the EBM function approximator ϵ_θ, unlike in the normalizing flow setting where we have multiple stacks of bijections. However while sampling we do loop N times over ϵ_θ. A possible strategy to improve sampling times introduced in (Chen et al., 2021) uses a combination of improved variance schedule and an L_1 loss to allow sampling with fewer steps at the cost of a small reduction in quality if such a trade-off is required. A recent paper (Song et al., 2021) generalize the diffusion processes via a class of non-Markovian processes which also allows for faster sampling.

The use of normalizing flows for discrete valued data dictates that one dequantizes it (Theis et al., 2016), by adding uniform noise to the data, before using the flows to learn. Dequantization is not needed in the EBM setting and future work could explore methods of explicitly modeling discrete distributions.

As noted in (Du & Mordatch, 2019) EBMs exhibit better out-of-distribution (OOD) detection than other likelihood models. Such a task requires models to have a high likelihood on the data manifold and low at all other locations. Surprisingly (Nalisnick et al., 2019) showed that likelihood models, including flows, were assigning higher likelihoods to OOD data whereas EBMs do not suffer from this issue since they penalize high probability under the model but low probability under the data distribution explicitly. Future work could evaluate the usage of `TimeGrad` for anomaly detection tasks.

For long time sequences, one could replace the RNN with a Transformer architecture (Rasul et al., 2021) to provide better conditioning for the EBM emission head. Concurrently, since EBMs are not constrained by the form of their functional approximators, one natural way to improve the model would be to incorporate architectural choices that best encode the inductive bias of the problem being tackled, for example with graph neural networks (Niu et al., 2020) when the relationships between entities are known.

Software

We wish to acknowledge and thank the authors and contributors of the following open source libraries that were used in this work: GluonTS (Alexandrov et al., 2020), NumPy (Harris et al., 2020), Pandas (Pandas development team, 2020), Matplotlib (Hunter, 2007) and PyTorch (Paszke et al., 2019).

Acknowledgements

We would like to thank and acknowledge the hard work of the reviewers whose comments and suggestions have been tremendously helpful.

K.R.: I would like to thank Stefano Ermon, Alexander März and Zaeem Burq for the helpful discussions and suggestions; as well as Antonia and Georges Herzog for their hospitality at the Restaurant Rössli in Büren zum Hof, Switzerland.

I would like to acknowledge the traditional owners of the land on which I have lived and worked, the Wurundjeri people of the Kulin nation who have been custodians of their land for thousands of years. I pay my respects to their elders, past and present as well as past and present aboriginal elders of other communities.

References

Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D. C., Rangapuram, S., Salinas, D., Schulz, J., Stella, L., Türkmen, A. C., and Wang, Y. GluonTS: Probabilistic and Neural Time Series Modeling in Python. *Journal of Machine Learning Research*, 21(116):1–6, 2020. URL http://jmlr.org/papers/v21/19-820.html.

Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, B., Maddix, D., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., Callot, L., and Januschowski, T. Neural forecasting: Introduction and literature overview, 2020.

Charrington, S. TWiML & AI Podcast: Systems and Software for Machine Learning at Scale with Jeff Dean, 2018. URL https://bit.ly/2G0LmGg.

Chen, N., Zhang, Y., Zen, H., Weiss, R. J., Norouzi, M., and Chan, W. WaveGrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations 2021 (Conference Track)*, 2021. URL https://openreview.net/forum?id=NsMLjcFaO8O.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

de Bézenac, E., Rangapuram, S. S., Benidis, K., Bohlke-Schneider, M., Kurle, R., Stella, L., Hasson, H., Gallinari, P., and Januschowski, T. Normalizing Kalman Filters for Multivariate Time series Analysis. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020.

De Brébisson, A., Simon, E., Auvolat, A., Vincent, P., and Bengio, Y. Artificial Neural Networks Applied to Taxi Destination Prediction. In *Proceedings of the 2015th International Conference on ECML PKDD Discovery Challenge - Volume 1526*, ECMLPKDDDC'15, pp. 40–51, Aachen, Germany, Germany, 2015. CEUR-WS.org. URL http://dl.acm.org/citation.cfm?id=3056172.3056178.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

Du, Y. and Mordatch, I. Implicit Generation and Modeling with Energy Based Models. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 3608–3618. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/378a063b8fdb1db941e34f4bde584c7d-Paper.pdf.

Fraccaro, M., Kamronn, S., Paquet, U., and Winther, O. A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30, pp. 3601–3610. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/7b7a53e239400a13bd6be6c91c4f6c4e-Paper.pdf.

Graves, A. Generating Sequences With Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*, 2013.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del R'ıo, J. F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Hinton, G. E. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14 (8):1771—1800, August 2002. ISSN 0899-7667. doi: 10.1162/089976602760128018. URL https://doi.org/10.1162/089976602760128018.

Ho, J., Jain, A., and Abbeel, P. Denoising Diffusion Probabilistic Models. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020. URL https://papers.nips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

Hyndman, R. and Athanasopoulos, G. *Forecasting: Principles and practice*. OTexts, 2018. ISBN 9780987507112.

Hyndman, R., Koehler, A., Ord, K., and Snyder, R. *Forecasting with exponential smoothing. The state space approach*, chapter 17, pp. 287–300. Springer-Verlag, 2008. doi: 10.1007/978-3-540-71918-2.

Hyvärinen, A. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005. URL http://jmlr.org/papers/v6/hyvarinen05a.html.

Jordan, A., Krüger, F., and Lerch, S. Evaluating Probabilistic Forecasts with scoringRules. *Journal of Statistical Software, Articles*, 90(12):1–37, 2019. ISSN 1548-7660. doi: 10.18637/jss.v090.i12. URL https://www.jstatsoft.org/v090/i12.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Kingma, D. P. and Welling, M. An Introduction to Variational Autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019. doi: 10.1561/2200000056. URL https://doi.org/10.1561/2200000056.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. DiffWave: A Versatile Diffusion Model for Audio Synthesis. In *International Conference on Learning Representations 2021 (Conference Track)*, 2021. URL https://openreview.net/forum?id=a-xFK8Ymz5J.

Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pp. 95–104, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5657-2. doi: 10.1145/3209978.3210006. URL http://doi.acm.org/10.1145/3209978.3210006.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. A Tutorial on Energy-Based Learning. In Bakir, G., Hofman, T., Schölkopf, B., Smola, A., and Taskar, B. (eds.), *Predicting Structured Data*. MIT Press, 2006.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 5244–5254. Curran Associates, Inc., 2019.

Lütkepohl, H. *New Introduction to Multiple Time Series Analysis*. Springer Berlin Heidelberg, 2007. ISBN 9783540262398. URL https://books.google.de/books?id=muorJ6FHIiEC.

Matheson, J. E. and Winkler, R. L. Scoring Rules for Continuous Probability Distributions. *Management Science*, 22(10):1087–1096, 1976.

Nalisnick, E., Matsukawa, A., Teh, Y. W., Gorur, D., and Lakshminarayanan, B. Do Deep Generative Models Know What They Don't Know? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1xwNhCcYm.

Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., and Ermon, S. Permutation Invariant Graph Generation via Score-Based Generative Modeling. In Chiappa, S. and Calandra, R. (eds.), *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pp. 4474–4484. PMLR, 2020.

Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1ecqn4YwB.

Pandas development team, T. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked Autoregressive Flow for Density Estimation. *Advances in Neural Information Processing Systems 30*, 2017.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing Flows for Probabilistic Modeling and Inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. URL http://jmlr.org/papers/v22/19-1028.html.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang,

L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8026–8037. Curran Associates, Inc., 2019.

Rasul, K., Sheikh, A.-S., Schuster, I., Bergmann, U., and Vollgraf, R. Multivariate Probabilistic Time Series Forecasting via Conditioned Normalizing Flows. In *International Conference on Learning Representations 2021 (Conference Track)*, 2021. URL https://openreview.net/forum?id=WiGQBFuVRv.

Salinas, D., Bohlke-Schneider, M., Callot, L., Medico, R., and Gasthaus, J. High-dimensional multivariate forecasting with low-rank Gaussian Copula Processes. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 6824–6834. Curran Associates, Inc., 2019a.

Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2019b. ISSN 0169-2070. URL http://www.sciencedirect.com/science/article/pii/S0169207019301888.

Smyl, S. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1):75–85, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.03.017. URL http://www.sciencedirect.com/science/article/pii/S0169207019301153. M4 Competition.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 2015. PMLR. URL http://proceedings.mlr.press/v37/sohl-dickstein15.html.

Song, J., Meng, C., and Ermon, S. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations 2021 (Conference Track)*, 2021. URL https://openreview.net/pdf?id=St1giarCHLP.

Song, Y. and Ermon, S. Generative Modeling by Estimating Gradients of the Data Distribution. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 11918–11930. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

Song, Y. and Ermon, S. Improved Techniques for Training Score-Based Generative Models. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/92c3b916311a5517d9290576e3ea37ad-Paper.pdf.

Song, Y. and Kingma, D. P. How to Train Your Energy-Based Models. 2021. URL https://arxiv.org/abs/2101.03288.

Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to Sequence Learning with Neural Networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 3104–3112. Curran Associates, Inc., 2014.

Theis, L., van den Oord, A., and Bethge, M. A note on the evaluation of generative models. In *International Conference on Learning Representations*, 2016. URL http://arxiv.org/abs/1511.01844. arXiv:1511.01844.

Tsay, R. S. *Multivariate Time Series Analysis: With R and Financial Applications*. Wiley Series in Probability and Statistics. Wiley, 2014. ISBN 9781118617908.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. WaveNet: A Generative Model for Raw Audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*, pp. 125. ISCA, 2016a. URL http://www.isca-speech.org/archive/SSW_2016/abstracts/ssw9_DS-4_van_den_Oord.html.

van den Oord, A., Kalchbrenner, N., Espeholt, L., kavukcuoglu, k., Vinyals, O., and Graves, A. Conditional Image Generation with PixelCNN Decoders. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 4790–4798. Curran Associates, Inc., 2016b. URL https://proceedings.neurips.cc/paper/2016/file/b1301141feffabac455e1f90a7de2054-Paper.pdf.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel Recurrent Neural Networks. In Balcan, M. F. and Weinberger, K. Q. (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1747–1756, New York, New York, USA, 20–22 Jun 2016c. PMLR. URL http://proceedings.mlr.press/v48/oord16.html.

van der Weide, R. GO-GARCH: a multivariate generalized orthogonal GARCH model. *Journal of Applied Econometrics*, 17(5):549–564, 2002. doi: 10.1002/jae.688.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is All you Need. In Guyon, I., Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Vincent, P. A Connection Between Score Matching and Denoising Autoencoders. *Neural Computation*, 23(7):1661–1674, 2011. URL https://doi.org/10.1162/NECO_a_00142.

Wenzel, F., Roth, K., Veeling, B., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How good is the Bayes posterior in deep neural networks really? In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10248–10259. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/wenzel20a.html.

Yoon, J., Jarrett, D., and van der Schaar, M. Time-series Generative Adversarial Networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 5508–5518. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf.

Zhu, L. and Laptev, N. Deep and Confident Prediction for Time Series at Uber. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, volume 00, pp. 103–110, November 2018. doi: 10.1109/ICDMW.2017.19. URL doi.ieeecomputersociety.org/10.1109/ICDMW.2017.19.