Deep Temporal Deaggregation: Large-Scale Spatio-Temporal Generative Models

David Bergström
Linköping University
Linköping, Sweden
david.bergstrom@liu.se

Mattias Tiger
Linköping University
Linköping, Sweden
mattias.tiger@liu.se

Fredrik Heintz
Linköping University
Linköping, Sweden
fredrik.heintz@liu.se

Abstract

Most of today's data is time-series data from sensors, transactions systems, and production systems. However, much of this data is sensitive and consequently unusable. Generative models have shown promise in generating non-sensitive synthetic data, to share and drive applications with. However, current generative time-series models are limited in their ability to capture the data distribution, limiting their usability. In this paper we propose a transformer-based diffusion model, TDDPM, for time-series which outperforms and scales substantially better than state-of-the-art. The focus is primarily on mobility data, such as trajectories of people's movement in cities, and we propose a conditional distribution approach which demonstrate out-of-distribution generalization to city-areas not trained on. We further propose a comprehensive benchmark across several sequence lengths, standard datasets, and evaluation measures, considering key distribution properties.

1 Introduction

Time-series data of human mobility is important because it enables pandemic forecasting and management [1], smart city development [2], urban governance [3], human rights violation detection [4] and monitoring of global migration induced by war and climate change [5, 6]. Two major challenges stand in the way for using time-series data to these ends. The first is a *shortage of publicly available data* [7]. Data can only be collected and shared in limited capacity due to privacy concerns, business concerns and national security, creating a silo effect. Secondly, *predictions* about unobserved parts in space, about the future or even possible futures - given that different actions are taken - is often a necessary complement to the readily collectable data. One such case is generating high-fidelity realistic spatio-temporal mobility data, such as individual pedestrians navigating a city or a building.

A solution to both challenges is to use time-series generative models to accurately capture the data distribution. These generative models can then be adapted to generate samples that are privacy-preserving [8, 9], resulting in synthetic non-private datasets that can be made publicly available. Further adaptation allows for tasks such as imputation [10] or forecasting [10, 11]. However, current approaches have crucial limitations limiting their real-world applicability: they can only generate short sequence length and they struggle to model complex distributions with sparse support.

In this work, *we introduce a new method for generating long and realistic sequences of spatio-temporal data, Temporal Denoising Diffusion Probabilistic Model (TDDPM)*, capable of out-of-distribution generalization via deaggregation from spatial statistics to temporal time-series samples. More specifically, we first adapt the denoising diffusion model to generate time-series data and show that it scales to significantly longer sequences than what is possible with previous models. Secondly, we ask: *Can a deep generative model (in our case a diffusion model) be made to stay true to a marginal probability distribution, while generating samples from the full distribution?* We demonstrate how this can be achieved via conditioning the model on a non-temporal marginal

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

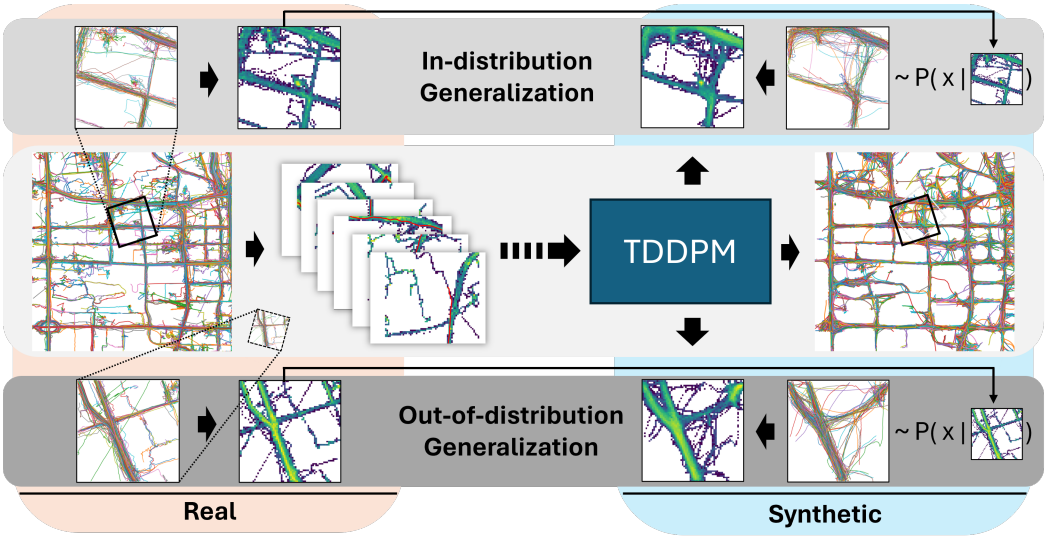

Figure 1: TDDPM is trained on real 2D trajectories (left) to generate synthetic trajectories (right), conditioned on how likely it should be for the population of synthetic trajectories to occupy space on the 2D plane. The latter is represented as a discrete distribution over occupancy frequency, i.e., the marginal distribution over the trajectory probability distribution if integrating out time. This yields both high-fidelity in-distribution generalization (top) and out-of-distribution generalization (bottom), the latter when conditioning on a marginal distribution not part of the training data (dashed rectangle).

distributions of the trajectory data, making deaggregation to individual trajectories possible. A consequence of this is that the model can generalize to environment changes and even to new environments without retraining, requiring only a statistic (i.e., an aggregate) of the new environment as input. Figure 1 illustrates the full capabilities of TDDPM in a setting of mobility data.

2 Temporal Denoising Diffusion Probabilistic Model (TDDPM)

The task of learning an unconditional generative model is defined as learning a mapping f from samples drawn from a known distribution $\mathcal{D}_{\text{known}}$, e.g. standard normal distribution, to samples from an unknown target distribution $\mathcal{D}_{\text{unknown}}$. The mapping is learned without direct access to the unknown distribution, and is instead limited to a set of samples $X_{\text{train}} = \{x_1, \ldots, x_N\}$, where $x_i \sim \mathcal{D}_{\text{unknown}}$. Once the mapping has been learned, synthetic data can be generating by first sampling the known distribution, then passing the individual samples through the mapping function $X_{\text{synthetic}} = \{f(y_i)\}_{i=0}^{M}$, where $y_i \sim \mathcal{D}_{\text{known}}$. The goal of this mapping is for the synthetic samples to be *similar* to samples from the known distribution.

The notion of similarity is often broken into several desirable properties[12–14] for the mapping function and the resulting synthetic data points. In this work we propose the following set of properties to evaluate the quality of synthetic data: (I) *Fidelity:* [13] The individual synthetic samples should have similar characteristics to, or be indistinguishable from, samples from the original distribution. (II) *Diversity:* [13] It should be possible for synthetic data to be drawn from any part of the unknown distribution's support. (III) *Proportionality:* [14] The probability of a sample occurring in the synthetic distribution should be proportional to the probability of a sample occurring in the unknown distribution. (IV) *Usefulness:* [12] The synthetic data should capture aspects of the unknown distribution that is useful for downstream tasks. (V) *Generalization:* [13] Synthetic samples should not be mere copies of the training data.

Further, we separate the generative task into two steps by extracting local information l from the training data. We can then condition the generation on this local information. This allows for more accurate modelling on challenging distributions, as well as enabling generalization in some cases where l can include sufficient information. More precisely, instead of learning to generate samples from $p(x)$ directly, we learn to generate samples from the joint distribution $p(x, l)$ via the chain

rule: $p(x, l) = p(x \mid l)\, p(l)$, where $p(x \mid l)$ is a conditional deep generative model and $p(l)$ can be modeled using a explicit and more interpretable distribution.

Creating synthetic samples from $p(x)$ is then done by first sampling $p(l)$ and then use these samples to sample the conditional distribution $p(x \mid l)$. In practice, we learn a conditional mapping function g which maps samples from a known distribution and local information l to synthetic samples, which should be *similar* to samples from the unknown distribution $\mathcal{D}_{\text{unknown}}$. More precisely, a dataset for a collection of local information $L = \{l_i\}_{i=0}^{M}$ is generated $X_{\text{synthetic}} = \left\{g(y_i, l_i)\right\}_{i=0}^{M}$, where $y_i \sim \mathcal{D}_{\text{known}}$. For certain problems, if the choice of l is sufficiently informative, we can learn to generate data for a new distribution $p^*(x)$ without having to retrain $p(x \mid l)$. This is achieved by estimating a new distribution $p^*(x, l)$, s.t. $p^*(x, l) \approx p(x \mid l)\, p^*(l)$. This is not only a computationally efficient way to estimate $p^*(x)$, but can also be useful in cases where we only have access to l, e.g. to model hypothetical scenarios or account for data distribution shifts.

To approximate $p(x \mid l)$ we propose an architecture based on the denoising diffusion architecture [15], using a transformer encoder [16] for denoising. At each denoising step, the transformer takes the entire noisy sequence, the current denoising step and the optional local information as input and predicts the noise added at the current step. See Appendix A.2 for full details, including tokenization.

The training data consists of a set of N trajectories, consisting of trajectories $t_i = \{p_1, p_2, \ldots, p_{L_i}\}$, each with observations $p_j \in \mathbb{R}^D$. We set l to be a local heatmap of the training data trajectories for a given region $r = (x, y, \theta)$ where θ is rotation in $[0,\, 2\pi)$. See Figure 1 for examples of heatmaps. During training we sample these regions uniformly (including rotation) and extract pairs of heatmap and sub-trajectories in the region. When generating synthetic datasets for evaluation, the regions are placed on a grid that spans the target geographical area. We start by calculating the heatmap for each region. Then, the model is sampled once for each region by conditioning on the corresponding heatmap. This allows for both interpolation, i.e. generating for regions that are not part of the training set, but that are from the same physical space and time, as well as extrapolation, generating sequences for previously unseen regions and different points in time. To ensure a representative dataset, the number of samples from each region is proportional to the total number of trajectories in each region.

3 Evaluation

In this section we first study the performance in unconditional mode, i.e. without local information, on a mix of spatio-temporal and time-series (non-spatial) data. Second, we evaluate TDDPM in conditional mode, i.e. using local information, which the other methods do not support, and validate the out-of-distribution generalization capability. Last, we show how local information can be used to generalize to new areas, and a proof-of-concept of generating data for a hypothetical what-if scenario.

Unconditional Generation
We evaluate TDDPM w.r.t. unconditional SOTA approaches representing GAN, VAE and Diffusion approaches: TimeGAN [17], TimeVAE [18], COSCI-GAN [19] and DiffusionTS [20]. We use six real-world datasets, four non-spatial datasets from the generative time-series field: Stock [17], Energy [17], Solar [21] and Electricity [21] and two spatio-temporal datasets from robotics / large-scale mobility: ATC [22] and Geolife [23]. The synthetic datasets from the trained models are compared to the original data using the standard measures: train on synthetic, test on real (TSTR) [17], t-SNE [24] and KL divergence. We also introduce TimeFID, an adaptation of the widely used Fréchet Inception Distance (FID) [25] to time-series generation by replacing the pre-trained image-specific inception network with a domain-specific time-series embedding network [26]. Prevalent numerical issues has also been addressed [27].

Evaluation using TimeFID (Table 1) show that TDDPM is on-par with top contenders, which vary between datasets, apart from for Solar and Electricity where TDDPM is better. TDDPM is also the only method which can scale to long sequences for all datasets. Both TimeGAN and COSCI-GAN can both take a long time to run, especially for larger datasets and longer sequence lengths. Other models crash due to allocating too much RAM or in rare circumstances VRAM. This indicates that the quadratic scaling of the Transformer architecture is not the limiting factor for time-series generation, rather optimistic memory allocation during preprocessing and sampling. We address this by dynamically loading windows, see Algorithms 1 and 2. We also evaluated TSTR (2) for which most methods are on-par on most datasets. In addition to the quantitative measures, we investigate qualitative aspects by plotting the training and synthetic datasets using t-SNE, see Figure 6. For

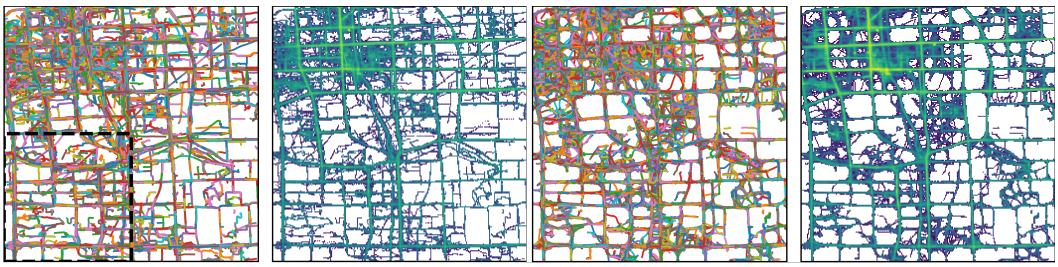

Figure 2: Generalization experiment. TDDPM is trained on the lower left quadrant and then sampled to generate trajectories for the remaining geographical area. From the left: (1) Data from Geolife, dashed area used for training, (2) heatmap of training data, (3) synthetic trajectories, and (4) heatmap of the synthetic data.

Geolife, we also calculate the spatial marginal distribution for the resulting synthetic trajectories, shown in Figure 7. These are then used to calculate the divergence between synthetic and training data distributions (Table 3, 4). Low KL(real ‖ synthetic) indicate that the synthetic distribution captures the real distribution well, which is distinctly seen for TDDPM. DiffusionTS, which is slightly worse, show better results for KL(synthetic ‖ real), indicating that its synthetic distribution has captured some modes in the real data well, but not across the full support of the real distribution. In both tables we can see that conditional TDDPM on out-of-distribution generation outperforms all other methods.

Large-scale human mobility conditional generative model
In these experiments, we validate the conditional method for generating large-scale spatio-temporal data. For this we use a quadrant of Geolife (Large) which is 40 km^2. After having trained on the region, we use the model to generate trajectories for the entire region (Figure 5). We observe that the synthetic trajectories are similar to the training data and when used to calculate a spatial marginalized distribution, the resulting distribution is almost indistinguishable from the original.

Generalizing to new environments
We generate heatmaps outside of the area used for training the model. In regions outside the geographical area, the heatmaps are generated with data previously not observed by the model. To evaluate the quality of the resulting synthetic trajectories, we calculate a spatial marginalized distribution with all synthetic trajectories from all regions. This distribution can then be compared to a distribution of the original, previously unobserved, data. The distribution and trajectories are all shown in Figure 2. We observe that the model successfully manages to generate trajectories for areas outside of the training region and the distribution of the synthetic data is similar to that of the original data. Finally, a proof of concept of what-if-scenario modelling is shown in Figure 4.

4 Conclusion

Scaling time-series generative models to long sequences and large-scale settings, not least for mobility data applications, has been very challenging. By conditioning denoising diffusion probabilistic models on a spatial marginal distribution, we demonstrate that TDDPM scales to problem sizes far beyond current state-of-the-art, without compromising fidelity in the generated trajectories. Moreover, TDDPM stays on-par in performance on unconditional tasks at smaller scales while including less induction bias on trajectories and being more efficient at both training time and sampling time than comparable high-fidelity approaches. Finally, high-quality out-of-distribution generalization is demonstrated at scale, including for what-if scenario modeling of road traffic. The accompanying comprehensive benchmark invite continued improvements as future work and a range of applications.

Acknowledgments and Disclosure of Funding

This work was supported by the Knut and Alice Wallenberg Foundation, and grants from the Excellence Center at Linköping-Lund for Information Technology (ELLIIT), the TAILOR Project, and the CUGS graduate school. The computations were enabled by the Berzelius resource (NSC) provided by the Knut and Alice Wallenberg Foundation and the Stellar facilities funded by ELLIIT.

References

[1] Cornelia Ilin, Sébastien Annan-Phan, Xiao Hui Tai, Shikhar Mehra, Solomon Hsiang, and Joshua E Blumenstock. Public mobility data enables covid-19 forecasting and management at local and global scales. *Scientific reports*, 11(1):13531, 2021.

[2] Yun Wang, Faiz Currim, and Sudha Ram. Deep learning of spatiotemporal patterns for urban mobility prediction using big data. *Information Systems Research*, 33(2):579–598, 2022.

[3] Gang Xiong, Zhishuai Li, Meihua Zhao, Yu Zhang, Qinghai Miao, Yisheng Lv, and Fei-Yue Wang. Trajsgan: A semantic-guiding adversarial network for rban trajectory generation. *IEEE Transactions on Computational Social Systems*, 11(2):1733–1743, 2024. doi: 10.1109/TCSS.2023.3235923.

[4] Xiao Hui Tai, Shikhar Mehra, and Joshua E Blumenstock. Mobile phone data reveal the effects of violence on internal displacement in afghanistan. *Nature human behaviour*, 6(5):624–634, 2022.

[5] Venla Niva, Alexander Horton, Vili Virkki, Matias Heino, Maria Kosonen, Marko Kallio, Pekka Kinnunen, Guy J Abel, Raya Muttarak, Maija Taka, et al. World's human migration patterns in 2000–2019 unveiled by high-resolution data. *Nature Human Behaviour*, 7(11):2023–2037, 2023.

[6] Alfredo Alessandrini, Daniela Ghio, Silvia Migali, et al. *Estimating net migration at high spatial resolution*. Publications Office of the European Union, 2020.

[7] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

[8] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

[9] Huandong Wang, Changzheng Gao, Yuchen Wu, Depeng Jin, Lina Yao, and Yong Li. Pategail: a privacy-preserving mobility trajectory generator with imitation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14539–14547, 2023.

[10] Juan Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

[11] Marcel Kollovieh, Abdul Fatir Ansari, Michael Bohlke-Schneider, Jasper Zschiegner, Hao Wang, and Yuyang (Bernie) Wang. Predict, refine, synthesize: Self-guiding diffusion models for probabilistic time series forecasting. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 28341–28364. Curran Associates, Inc., 2023.

[12] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. *arXiv preprint arXiv:1706.02633*, 2017.

[13] Ahmed Alaa, Boris Van Breugel, Evgeny S. Saveliev, and Mihaela van der Schaar. How faithful is your synthetic data? Sample-level metrics for evaluating and auditing generative models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 290–306. PMLR, 17–23 Jul 2022.

[14] Qitian Wu, Rui Gao, and Hongyuan Zha. Bridging explicit and implicit deep generative models via neural stein estimators. *Advances in Neural Information Processing Systems*, 34: 11274–11286, 2021.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, 2017.

[17] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series Generative Adversarial Networks. *Advances in neural information processing systems (NeurIPS)*, 32, 2019.

[18] Abhyuday Desai, Cynthia Freeman, Ian Beaver, and Zuhui Wang. TimeVAE: A variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*, 2021.

[19] Ali Seyfi, Jean-Francois Rajotte, and Raymond Ng. Generating multivariate time series with COmmon Source CoordInated GAN (COSCI-GAN). *Advances in Neural Information Processing Systems*, 35:32777–32788, 2022.

[20] Xinyu Yuan and Yan Qiao. Diffusion-TS: Interpretable diffusion for general time series generation. In *The Twelfth International Conference on Learning Representations*, 2024.

[21] Paul Jeha, Michael Bohlke-Schneider, Pedro Mercado, Shubham Kapoor, Rajbir Singh Nirwan, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. PSA-GAN: Progressive Self Attention GANs for Synthetic Time Series. In *International Conference on Learning Representations (ICLR)*, 2021.

[22] Dražen Brščić, Takayuki Kanda, Tetsushi Ikeda, and Takahiro Miyashita. Person tracking in large public spaces using 3-d range sensors. *IEEE Transactions on Human-Machine Systems*, 43(6):522–534, 2013.

[23] Yu Zheng, Hao Fu, Xing Xie, Wei-Ying Ma, and Quannan Li. *Geolife GPS trajectory dataset - User Guide*, geolife gps trajectories 1.1 edition, July 2011. URL `https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/`.

[24] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.

[25] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[26] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[27] Nicholas J Higham. Computing a nearest symmetric positive semidefinite matrix. *Linear algebra and its applications*, 103:103–118, 1988.

[28] Jinsung Jeon, Jeonghak Kim, Haryong Song, Seunghyeon Cho, and Noseong Park. Gt-gan: General purpose time series synthesis with generative adversarial networks. *Advances in Neural Information Processing Systems*, 35:36999–37010, 2022.

[29] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021.

[30] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24804–24816. Curran Associates, Inc., 2021.

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

[32] Yun Dai, Chao Yang, Kaixin Liu, Angpeng Liu, and Yi Liu. Timeddpm: Time series augmentation strategy for industrial soft sensing. *IEEE Sensors Journal*, 2023.

[33] Shibo Feng, Chunyan Miao, Zhong Zhang, and Peilin Zhao. Latent diffusion transformer for probabilistic time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11979–11987, 2024.

[34] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[35] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

[36] Ibai Lana, Javier Del Ser, Manuel Velez, and Eleni I Vlahogianni. Road traffic forecasting: Recent advances and new challenges. *IEEE Intelligent Transportation Systems Magazine*, 10 (2):93–109, 2018.

A Appendix / supplemental material

A.1 Related Work

Previous work on unconditional generation of time-series data has focused on variations of the generative adversarial networks (GAN) architecture [12, 17, 21, 28] and, more recently diffusion models [29, 20, 11]. There has also been an interest in using time-series generation for imputation and forecasting [10, 30–33]. Transformer-based time-series foundation models has been proposed as a general purpose forecasting tool [7], but has not been evaluated on the unconditional generation task.

TimeGAN [17] consists of a generative adversarial network (GAN) operating inside the latent space of an autoencoder. To further improve performance, they add an additional network with the task of predicting one time step ahead. The encoder, decoder, supervisor, generator and discriminator are all implemented using autoregressive models and in practice they use gated recurrent units (GRUs).

The TimeVAE [18] architecture is a variant of the popular variational autoencoder architecture.The autoencoder is trained with an additional loss component to have the latent space conform to a known statistical distribution, in this instance a multivariate normal distribution. The autoencoder is trained to both minimize the reconstruction loss, as well as minimizing the divergence between the embedded data and the prior set for the latent space.

COSCI-GAN [19] proposes to use a separate generative adversarial network for each channel of the data. The individual GANs all share single source of noise as input to the generator and, additionally they a central discriminator that is given the stacked output from the all the generators as input.

DiffusionTS [20] adapts the denoising diffusion architecture [15] to generate time-series data by implementing the denoising step with a multilayer neural network each consisting of a transformer block, a fully connected neural network as well as time-series specific layers with the aim of improved interpretability. The transformer architecture [16] proposed attention as an alternative to recurrence in neural networks and the architecture itself became a common architecture. The vision transformer (ViT) [34] is based on the transformer encoder and splits images into several equally sized patches which are projected before given as input tokens to the encoder.

Another track of generative models are based on the structured state-space architecture (S4) [35]. The SSSD architecture [10] combines DiffWave [29], a denoising diffusion model for audo synthesis, with S4 to use denoising diffusion for imputation and forecasting. This was later extended into the TSDiff architecture with various capabilities, including unconditional generation, although limited to univariate data.

In this work, we focus on using denoising diffusion to generate time-series data. This is similar to [20], but we introduce less inductive bias by using a simpler architecture without any time-series specific layers, and instead we rely on time-embeddings [16]. In practice, local information is expressed as heatmaps and similar to ViT processes images, we split our heatmaps into 8x8 equally sized patches and use linear projection to convert them into tokens.

A.2 Architecture Additional Details

An overview of the approach is shown in Figure 3. Positional encoding [15]:

$$\mathrm{PE}_{(pos,2i)} = \sin\left(-e^{i}\frac{\log(10000)}{\frac{d}{2}-1}\right) \tag{1}$$

$$\mathrm{PE}_{(pos,2i+1)} = \cos\left(-e^{i}\frac{\log(10000)}{\frac{d}{2}-1}\right) \tag{2}$$

The input to the transformer encoder is:

- L input tokens, each token corresponding to a time point in the noisy sequence. **Note:** The token size depends on the number of features of the dataset. It is a concatenation of:
  - $x \in \mathbb{R}^{DN}$, each corresponding to a dimension observed at each time point encoded using positional encoding

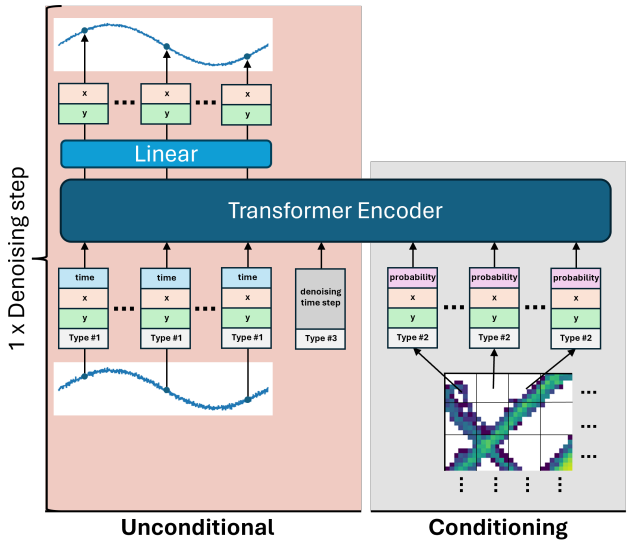

Figure 3: Overview of the architecture. In the unconditional part, each time point of the noisy trajectory is converted into a separate token with positional embedding [16] used to embed its values and the time point, as well as a learned vector representing its type. The denoising step token encodes the denoising step, the step is encoding using positional encoding and then concatenated with a type vector. The transformer can also optionally take a marginal distribution to guide the denoising process to generate samples with particular properties, improving in-distribution performance as well as enabling generalization to previously unobserved areas. The marginal distribution is split into tokens, taking inspiration from ViT [34], concatenated with a learned type vector and the corresponding position using positional embedding.

  - $x \in \mathbb{R}^N$, the time point encoded using the positional encoding introduced in [16]
  - $x \in \mathbb{R}^N$, a learned vector encoding denoting that this is a token that corresponds to a noisy sequence
- **Optional conditional information:** 64 tokens, each corresponding to a patch of the heatmap and being a concatenation of:
  - $x \in \mathbb{R}^N$, corresponding to the x position of the patch. Encoded using positional encoding [16].
  - $x \in \mathbb{R}^N$, corresponding to the y position of the patch. Encoded using positional encoding [16].
  - $x \in \mathbb{R}^N$, corresponding to the intensity of the heatmap. Encoded using a linear projection [34].
  - $x \in \mathbb{R}^N$, a learned vector encoding denoting that this is a token that corresponds to the conditional information
- A token encoding the current denoising step:
  - $x \in \mathbb{R}^{N(D+1)}$, the denoising step encoded using positional encoding [16]
  - $x \in \mathbb{R}^N$, a learned vector encoding denoting that this is a token that corresponds to the denoising step

A.3 Data Preprocessing Details

The datasets used are summarized in Table 5. The ATC shopping center dataset consist of pedestrians being tracked indoors between 2012 and 2013. We use the first day (2012/10/24) of the dataset, and drop all features except position (x,y) scaled to meters. Time is also dropped in favor for index since time between observations is roughly constant. Geolife is a GPS trajectory dataset collected by Microsoft Research Asia, consists of 178 users and was collected between 2007 and 2011. The

Algorithm 1 Pre-compute and save indices for (valid) sliding windows

Input: Raw dataset $X = x_1, \ldots, x_N$, Length of the raw sequences $T \in \mathbb{N}^N$, Sequence length L
Output: List l which maps from window indices to indices in X
Initialize $l = []$.
for $i = 1$ **to** N **do**
  for $j = 1$ **to** $T_i - L$ **do**
    Optional: **if not valid(**$X_{i,j:j+T}$**)** continue
    Append (i, j) to l
  end for
end for

Algorithm 2 Lookup window index to sequence window

Input: Raw dataset X, Sequence length L, list that maps from window indices to sequence indices and time points l, a window index k
Output: A window w_k
$(i, j) \leftarrow l_k$
$w_k \leftarrow X_{i,j:j+T}$

majority of the data is centered on Beijing, China. We use two subsets of the full Geolife dataset: Geolife (Small) at 10 km^2 and Geolife (Large) at 161 km^2.

To save on memory in order to not crash on long sequences, it is often necessary to save a list of valid sliding window indices rather than save all sliding windows in memory. To calculate these, we use Algorithm 1. For Geolife, we only save windows that are within the target geographic area, where less than 10 seconds have elapsed between observations and where the velocity is less than 140 km/h. Once the indices are pre-computed, we look up the window sequences when collecting the current batch using Algorithm 2.

A.4 Additional Metrics Details

TSTR: Train on synthetic, test on real. We follow the evaluation done by TimeGAN [17] and use the synthetic data to train a GRU-based RNN on task of one step prediction using the synthetic data. The resulting model is then evaluated on the training data and the mean absolute value is reported. This evaluates the usefulness, fidelity and diversity of the synthetic data. A lower score is better.

t-SNE: We refine the process of visual evaluation by using an embedding network to first embed the real and synthetic time-sequences into fixed-size vectors. The vectors are then passed to t-SNE which places the vectors on a 2D plane, similar vectors are placed close to each other and vectors that are different further away. The points are then colored, with different colors for the training and synthetic datasets. This allows us to visually evaluate the similarity between the synthetic and real data. More similar data are more similar in terms of fidelity and diversity, while having points that cover the entire range of real data points indicates support coverage. If all synthetic points share the same location as the training data, this suggests poor generalization.

KL Divergence: Evaluates support coverage as well as proportionality. KL(real $||$ synthetic) measures how well the synthetic distribution represents the real distribution, with 0 being identical. KL(synthetic $||$ real) measures how well the synthetic distribution fits inside the real distribution, i.e. it is sufficient that the synthetic distribution matches a single mode of the real distribution in order to get a low divergence value. The synthetic distribution consequently do not need to have the same support, nor have any probability density outside of this mode. Symmetric KL weight these two together and Jensen–Shannon divergence is a more stable version of symmetric KL with regions without probability in either of the distributions.

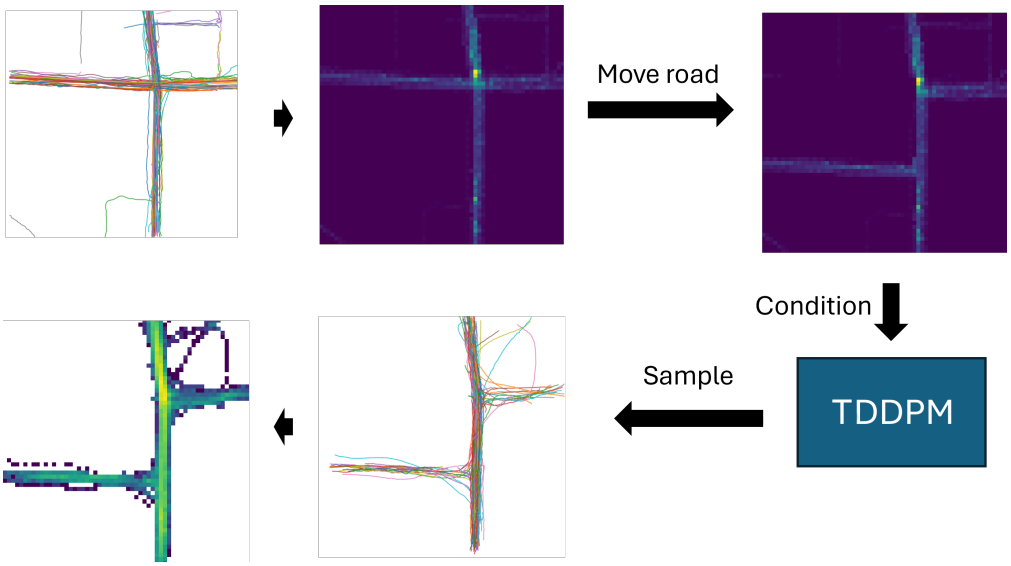

Figure 4: What-if-scenario where a road is removed and added.

A.5 Additional Results

A.5.1 What-if analysis

Current approaches also struggle with environmental and context changes. The environment (and how people behave within it) often undergo rapid changes. This necessitates *data ageing mechanisms that allow models to adapt their prediction to changing circumstances* [36], such as for example road construction, traffic accidents, traffic-light malfunction or other traffic flow and road topology changes. For analysis and what-if scenario modeling, it is further important that the distributions of synthetic data is proportional to the probability distribution of the real data: Certain locations and motion patterns are more frequently occurring than others. This is important for policy making, planning and decisions to be resting on correct risk assessments using for example Bayesian inference.

Proof-of-concept of this capability by TDDPM is shown in Figure 4.

A.5.2 Unconditional and conditional experiments

Table 1: Evaluation of unconditional generation using TimeFID. Lower is better. Several experiments ran out of 128 GB system memory (RAM) or 24GB of video memory (VRAM) (*oom*) or did not finish (*dnf*) in under 100 hours.

Dataset	Len.	TimeGAN	TimeVAE	COSCI-GAN	DiffusionTS	TDDPM
Stock	24	0.49 ± 0.49	0.15 ± 0.15	$\mathbf{0.12 \pm 0.12}$	0.15 ± 0.15	0.13 ± 0.13
	32	0.56 ± 0.56	0.13 ± 0.13	$\mathbf{0.12 \pm 0.12}$	$\mathbf{0.12 \pm 0.12}$	0.19 ± 0.19
	64	0.25 ± 0.25	0.19 ± 0.19	0.21 ± 0.21	$\mathbf{0.11 \pm 0.11}$	0.20 ± 0.20
	128	2.40 ± 2.40	0.52 ± 0.52	0.22 ± 0.22	$\mathbf{0.16 \pm 0.16}$	0.40 ± 0.40
	256	1.87 ± 1.87	0.29 ± 0.29	0.14 ± 0.14	$\mathbf{0.09 \pm 0.09}$	0.15 ± 0.15
	512	dnf	1.37 ± 1.37	$\mathbf{0.05 \pm 0.05}$	oom	0.28 ± 0.28
	1024	dnf	130.71 ± 130.71	$\mathbf{0.04 \pm 0.04}$	oom	0.30 ± 0.30
Energy	24	0.69 ± 0.69	0.43 ± 0.43	1.73 ± 1.73	$\mathbf{0.27 \pm 0.27}$	0.40 ± 0.40
	32	0.77 ± 0.77	0.44 ± 0.44	2.14 ± 2.14	$\mathbf{0.30 \pm 0.30}$	0.45 ± 0.45
	64	0.50 ± 0.50	0.35 ± 0.35	1.47 ± 1.47	$\mathbf{0.22 \pm 0.22}$	0.37 ± 0.37
	128	1.42 ± 1.42	0.52 ± 0.52	1.74 ± 1.74	$\mathbf{0.19 \pm 0.19}$	0.51 ± 0.51
	256	3.46 ± 3.46	0.50 ± 0.50	0.90 ± 0.90	$\mathbf{0.21 \pm 0.21}$	0.52 ± 0.52
	512	dnf	$\mathbf{0.37 \pm 0.37}$	0.81 ± 0.81	oom	0.43 ± 0.43
	1024	dnf	oom	0.56 ± 0.56	oom	$\mathbf{0.37 \pm 0.37}$
Solar	24	oom	3.54 ± 3.54	$2e8 \pm 2e8$	0.73 ± 0.73	$\mathbf{0.66 \pm 0.66}$
	32	oom	2.87 ± 2.87	$2e7 \pm 2e7$	0.74 ± 0.74	$\mathbf{0.48 \pm 0.48}$
	64	oom	1.73 ± 1.73	$4e6 \pm 4e6$	0.74 ± 0.74	$\mathbf{0.53 \pm 0.53}$
	128	oom	$\mathbf{0.54 \pm 0.54}$	$4e7 \pm 4e7$	0.88 ± 0.88	0.80 ± 0.80
	256	oom	oom	$4e7 \pm 4e7$	0.79 ± 0.79	$\mathbf{0.38 \pm 0.38}$
	512	oom	oom	$4e7 \pm 4e7$	oom	$\mathbf{0.52 \pm 0.52}$
	1024	oom	oom	dnf	oom	$\mathbf{1.24 \pm 1.24}$
Electricity	24	oom	5.27 ± 5.27	dnf	4.86 ± 4.86	$\mathbf{2.29 \pm 2.29}$
	32	oom	4.42 ± 4.42	dnf	4.49 ± 4.49	$\mathbf{2.13 \pm 2.13}$
	64	oom	4.65 ± 4.65	dnf	4.73 ± 4.73	$\mathbf{1.35 \pm 1.35}$
	128	oom	oom	dnf	4.73 ± 4.73	$\mathbf{1.09 \pm 1.09}$
	256	oom	oom	dnf	4.67 ± 4.67	$\mathbf{1.52 \pm 1.52}$
	512	oom	oom	dnf	oom	$\mathbf{0.68 \pm 0.68}$
	1024	oom	oom	dnf	oom	$\mathbf{0.77 \pm 0.77}$
ATC	24	dnf	50.34 ± 50.34	dnf	$\mathbf{0.13 \pm 0.13}$	0.14 ± 0.14
	32	dnf	76.20 ± 76.20	dnf	$\mathbf{0.14 \pm 0.14}$	0.19 ± 0.19
	64	dnf	413.52 ± 413.52	dnf	$\mathbf{0.11 \pm 0.11}$	0.18 ± 0.18
	128	dnf	oom	dnf	$\mathbf{0.14 \pm 0.14}$	0.16 ± 0.16
	256	dnf	oom	dnf	$\mathbf{0.22 \pm 0.22}$	0.24 ± 0.24
	512	dnf	oom	dnf	oom	$\mathbf{0.29 \pm 0.29}$
	1024	dnf	oom	dnf	oom	$\mathbf{0.42 \pm 0.42}$
Geolife (Small)	24	1.05 ± 1.05	0.24 ± 0.24	0.29 ± 0.29	$\mathbf{0.12 \pm 0.12}$	0.18 ± 0.18
	32	0.54 ± 0.54	0.41 ± 0.41	0.98 ± 0.98	$\mathbf{0.16 \pm 0.16}$	0.19 ± 0.19
	64	0.36 ± 0.36	0.35 ± 0.35	0.27 ± 0.27	$\mathbf{0.17 \pm 0.17}$	0.25 ± 0.25
	128	0.76 ± 0.76	0.21 ± 0.21	$\mathbf{0.16 \pm 0.16}$	0.21 ± 0.21	0.21 ± 0.21
	256	0.96 ± 0.96	$\mathbf{0.18 \pm 0.18}$	0.25 ± 0.25	0.28 ± 0.28	0.37 ± 0.37
	512	dnf	0.26 ± 0.26	$\mathbf{0.21 \pm 0.21}$	oom	0.40 ± 0.40
	1024	dnf	0.46 ± 0.46	$\mathbf{0.14 \pm 0.14}$	oom	0.17 ± 0.17

Table 2: Evaluation of unconditional generation using TSTR. Lower is better. Several experiments ran out of 128 GB system memory (RAM) or 24GB of video memory (VRAM) (*oom*) or did not finish (*dnf*) in under 100 hours.

Dataset	Len.	TimeGAN	TimeVAE	COSCI-GAN	DiffusionTS	Ours
Stock	24	$\mathbf{0.01 \pm 0.01}$	0.03 ± 0.03	0.02 ± 0.02	0.02 ± 0.02	0.02 ± 0.02
	32	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.01}$	0.02 ± 0.02	$\mathbf{0.01 \pm 0.01}$
	64	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.01}$	0.02 ± 0.02	$\mathbf{0.01 \pm 0.01}$	0.02 ± 0.02
	128	0.02 ± 0.02	0.02 ± 0.02	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.01}$
	256	0.04 ± 0.04	0.03 ± 0.03	$\mathbf{0.01 \pm 0.01}$	0.02 ± 0.02	$\mathbf{0.01 \pm 0.01}$
	512	dnf	0.02 ± 0.02	0.02 ± 0.02	oom	$\mathbf{0.01 \pm 0.01}$
	1024	dnf	0.04 ± 0.04	$\mathbf{0.01 \pm 0.01}$	oom	$\mathbf{0.01 \pm 0.01}$
Energy	24	0.07 ± 0.07	0.06 ± 0.06	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$
	32	0.06 ± 0.06	0.06 ± 0.06	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$
	64	0.06 ± 0.06	0.06 ± 0.06	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$
	128	0.09 ± 0.09	0.06 ± 0.06	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$
	256	0.21 ± 0.21	0.06 ± 0.06	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{0.05 \pm 0.05}$
	512	dnf	$\mathbf{0.05 \pm 0.05}$	0.06 ± 0.06	oom	0.06 ± 0.06
	1024	dnf	oom	0.13 ± 0.13	oom	$\mathbf{0.05 \pm 0.05}$
Solar	24	oom	$\mathbf{0.01 \pm 0.01}$	0.92 ± 0.92	0.07 ± 0.07	0.07 ± 0.07
	32	oom	$\mathbf{0.01 \pm 0.01}$	0.54 ± 0.54	0.07 ± 0.07	0.07 ± 0.07
	64	oom	$\mathbf{0.01 \pm 0.01}$	0.07 ± 0.07	0.07 ± 0.07	0.07 ± 0.07
	128	oom	$\mathbf{0.01 \pm 0.01}$	0.80 ± 0.80	$\mathbf{0.01 \pm 0.01}$	0.07 ± 0.07
	256	oom	oom	0.92 ± 0.92	$\mathbf{0.07 \pm 0.07}$	$\mathbf{0.07 \pm 0.07}$
	512	oom	oom	0.93 ± 0.93	oom	$\mathbf{0.07 \pm 0.07}$
	1024	oom	oom	dnf	oom	$\mathbf{0.07 \pm 0.07}$
Electricity	24	oom	$\mathbf{0.00 \pm 0.00}$	dnf	$\mathbf{0.00 \pm 0.00}$	$\mathbf{0.00 \pm 0.00}$
	32	oom	$\mathbf{0.00 \pm 0.00}$	dnf	$\mathbf{0.00 \pm 0.00}$	$\mathbf{0.00 \pm 0.00}$
	64	oom	$\mathbf{0.00 \pm 0.00}$	dnf	$\mathbf{0.00 \pm 0.00}$	$\mathbf{0.00 \pm 0.00}$
	128	oom	oom	dnf	0.01 ± 0.01	$\mathbf{0.00 \pm 0.00}$
	256	oom	oom	dnf	$\mathbf{0.00 \pm 0.00}$	$\mathbf{0.00 \pm 0.00}$
	512	oom	oom	dnf	oom	$\mathbf{0.00 \pm 0.00}$
	1024	oom	oom	dnf	oom	$\mathbf{0.00 \pm 0.00}$
ATC	24	dnf	0.31 ± 0.31	dnf	$\mathbf{0.04 \pm 0.04}$	0.09 ± 0.09
	32	dnf	0.27 ± 0.27	dnf	$\mathbf{0.04 \pm 0.04}$	$\mathbf{0.04 \pm 0.04}$
	64	dnf	0.26 ± 0.26	dnf	$\mathbf{0.04 \pm 0.04}$	$\mathbf{0.04 \pm 0.04}$
	128	dnf	oom	dnf	$\mathbf{0.04 \pm 0.04}$	$\mathbf{0.04 \pm 0.04}$
	256	dnf	oom	dnf	$\mathbf{0.04 \pm 0.04}$	$\mathbf{0.04 \pm 0.04}$
	512	dnf	oom	dnf	oom	$\mathbf{0.04 \pm 0.04}$
	1024	dnf	oom	dnf	oom	$\mathbf{0.05 \pm 0.05}$
Geolife (Small)	24	0.12 ± 0.12	0.10 ± 0.10	0.11 ± 0.11	0.13 ± 0.13	$\mathbf{0.09 \pm 0.09}$
	32	$\mathbf{0.09 \pm 0.09}$	$\mathbf{0.09 \pm 0.09}$	0.10 ± 0.10	$\mathbf{0.09 \pm 0.09}$	$\mathbf{0.09 \pm 0.09}$
	64	0.13 ± 0.13	$\mathbf{0.08 \pm 0.08}$	0.11 ± 0.11	0.09 ± 0.09	0.10 ± 0.10
	128	$\mathbf{0.08 \pm 0.08}$	0.12 ± 0.12	$\mathbf{0.08 \pm 0.08}$	$\mathbf{0.08 \pm 0.08}$	$\mathbf{0.08 \pm 0.08}$
	256	0.07 ± 0.07	$\mathbf{0.06 \pm 0.06}$	0.07 ± 0.07	0.07 ± 0.07	0.07 ± 0.07
	512	dnf	0.05 ± 0.05	$\mathbf{0.04 \pm 0.04}$	oom	$\mathbf{0.04 \pm 0.04}$
	1024	dnf	0.10 ± 0.10	$\mathbf{0.02 \pm 0.02}$	oom	$\mathbf{0.02 \pm 0.02}$

Table 3: KL divergence between spatial marginal distributions. TDDPM in unconditional mode except for sequence length 128^*. Then TDDPM is trained on one quarter of the map and KL is calculated over full dataset including both in- and out-of-distribution data. Lower is better.

Metric	Len.	TimeGAN	TimeVAE	COSCI-GAN	DiffusionTS	TDDPM
KL(real || synthetic)	24	2.78	1.51	1.07	**0.75**	**0.75**
	32	1.41	1.49	1.30	0.70	**0.69**
	64	1.40	1.25	1.00	**0.68**	0.72
	128	1.77	1.08	1.01	0.80	**0.76**
	128^*	-	-	-	-	**0.41**
	256	2.73	1.47	1.78	2.23	**1.01**
	512	dnf	3.57	2.88	dnf	**1.19**
	1024	dnf	6.40	3.98	dnf	**0.61**
KL(synthetic || real)	24	3.31	1.83	1.65	**0.94**	**0.94**
	32	1.80	1.92	1.82	**0.85**	0.86
	64	1.83	1.71	1.44	**0.73**	0.89
	128	2.01	1.46	1.49	**0.77**	1.02
	128^*	-	-	-	-	**0.45**
	256	4.08	2.07	2.93	1.79	**1.25**
	512	dnf	6.05	4.98	dnf	**1.22**
	1024	dnf	8.76	6.50	dnf	**0.61**

Table 4: Divergence between spatial marginal distributions. TDDPM in unconditional mode except for sequence length 128^*. Then TDDPM is trained on one quarter of the map and KL is calculated over full dataset including both in- and out-of-distribution data. Lower is better.

Dataset	Len.	TimeGAN	TimeVAE	COSCI-GAN	DiffusionTS	TDDPM
Symmetric KL	24	3.04	1.67	1.36	**0.84**	0.85
	32	1.60	1.70	1.56	**0.78**	**0.78**
	64	1.61	1.48	1.22	**0.71**	0.80
	128	1.89	1.27	1.25	**0.79**	0.89
	128^*	-	-	-	-	**0.43**
	256	3.40	1.77	2.36	2.01	**1.13**
	512	dnf	4.81	3.93	oom	**1.20**
	1024	dnf	7.58	5.24	oom	**0.61**
JS	24	0.44	0.28	0.23	**0.16**	**0.16**
	32	0.26	0.28	0.26	0.15	**0.14**
	64	0.26	0.24	0.20	**0.14**	0.15
	128	0.29	0.21	0.21	**0.15**	0.16
	128^*	-	-	-	-	**0.09**
	256	0.42	0.27	0.34	0.31	**0.20**
	512	dnf	0.51	0.48	dnf	**0.19**
	1024	dnf	0.66	0.58	dnf	**0.11**

Table 5: Selected properties of the datasets used in evaluation of unconditional generation. A dataset consists of one or more sequences, each sequence consist of one or more observation across time and at each time point one or more dimensions are observed.

Name	Number of sequences	Sequence length	Observations	Dimensions
Stock	1	3,685	3,865	6
Energy	1	19,735	19,735	28
Solar	137	105,121	14,401,440	1
Electricity	370	[16,032, 140,256]	41,855,506	1
ATC	49,688	[24, 187766]	47,290,292	2
Geolife (Smaller)	16,708	[24, 3,322]	1,957,039	2

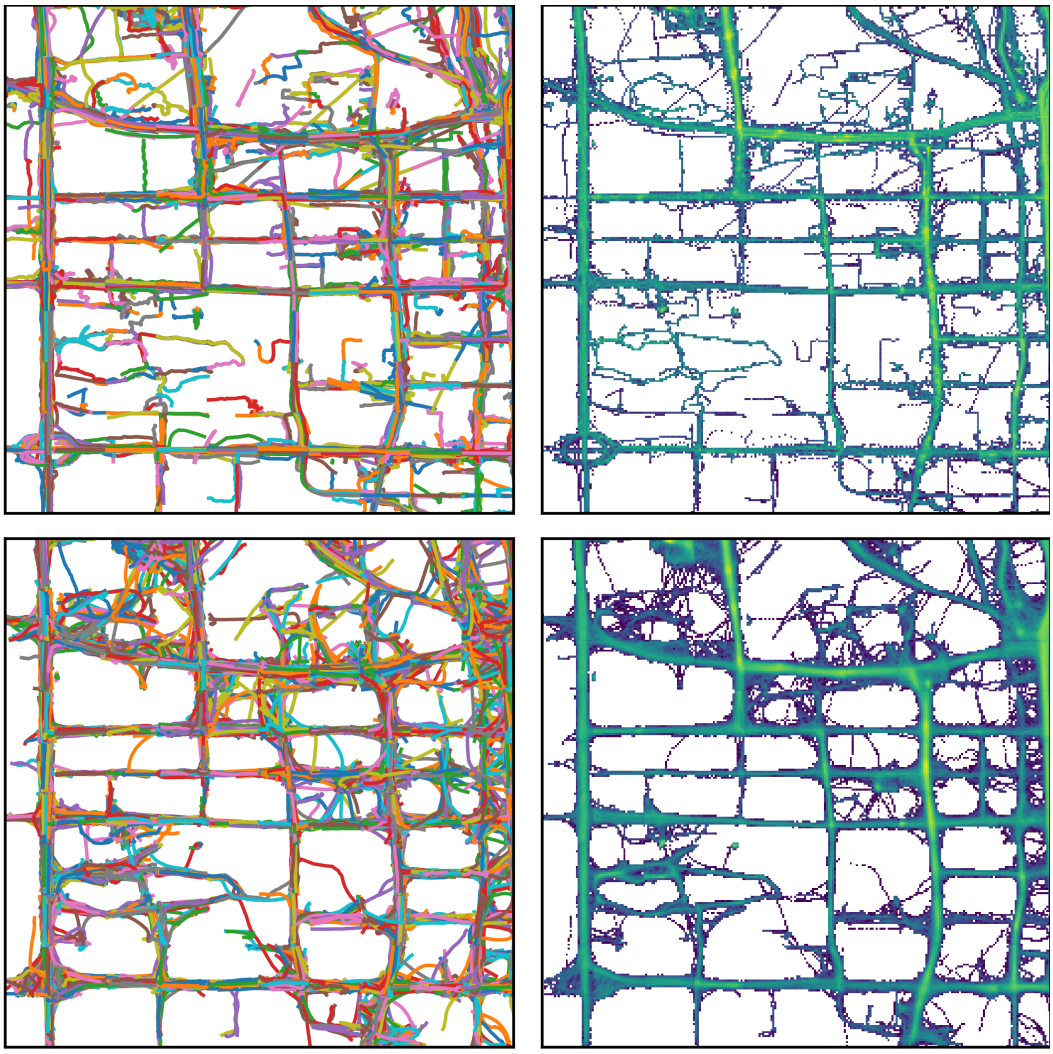

Figure 5: Interpolation experiment. The model has trained on this region is tasked to reconstruct it from the heatmaps. *Top left:* Data from Geolife used for training. *Top right:* heatmap of training data and areas used for creating query heatmaps for sampling the model, *Bottom left:* synthetic trajectories. *Bottom right:* heatmap of the synthetic data.

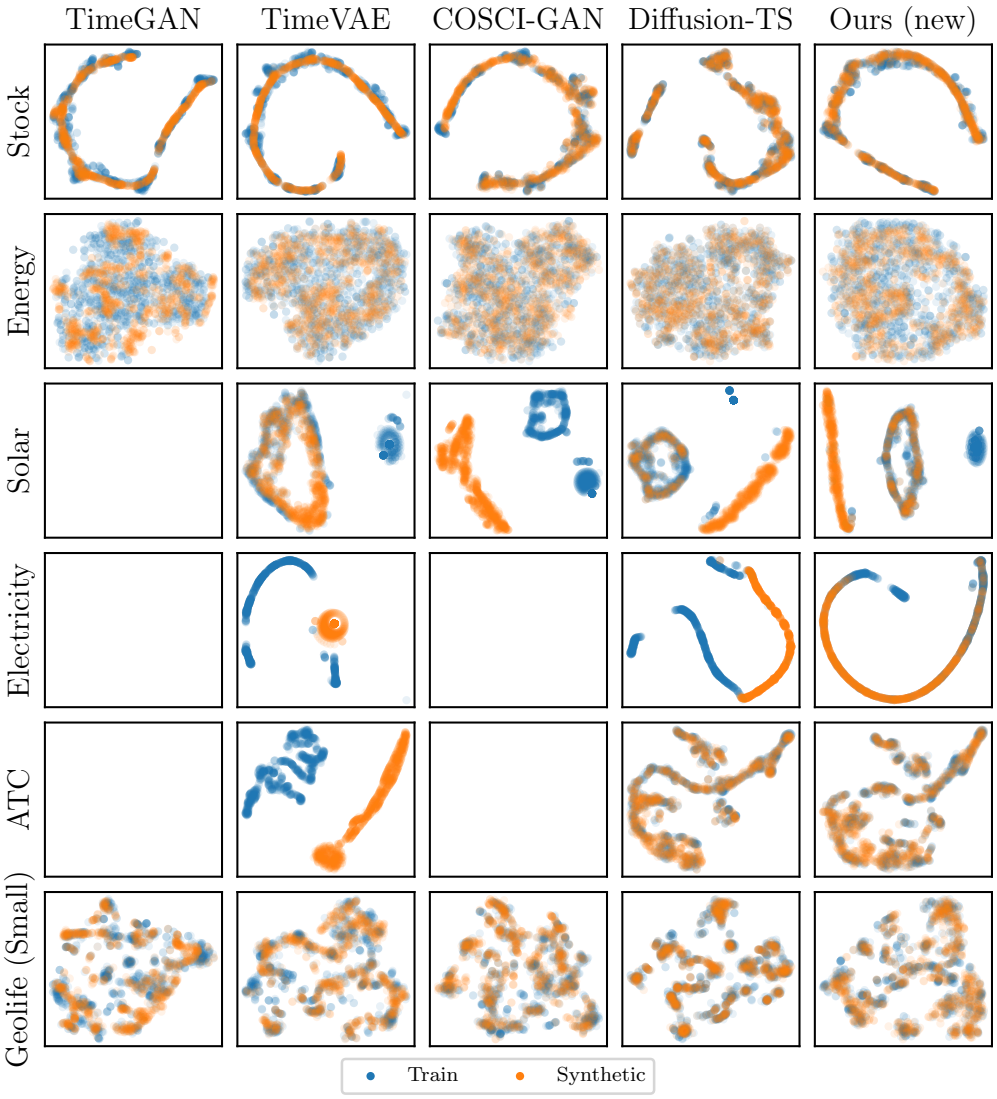

Figure 6: t-SNE analysis for sequence length 64. Blank figures are unsuccessful experiments (Table 1).

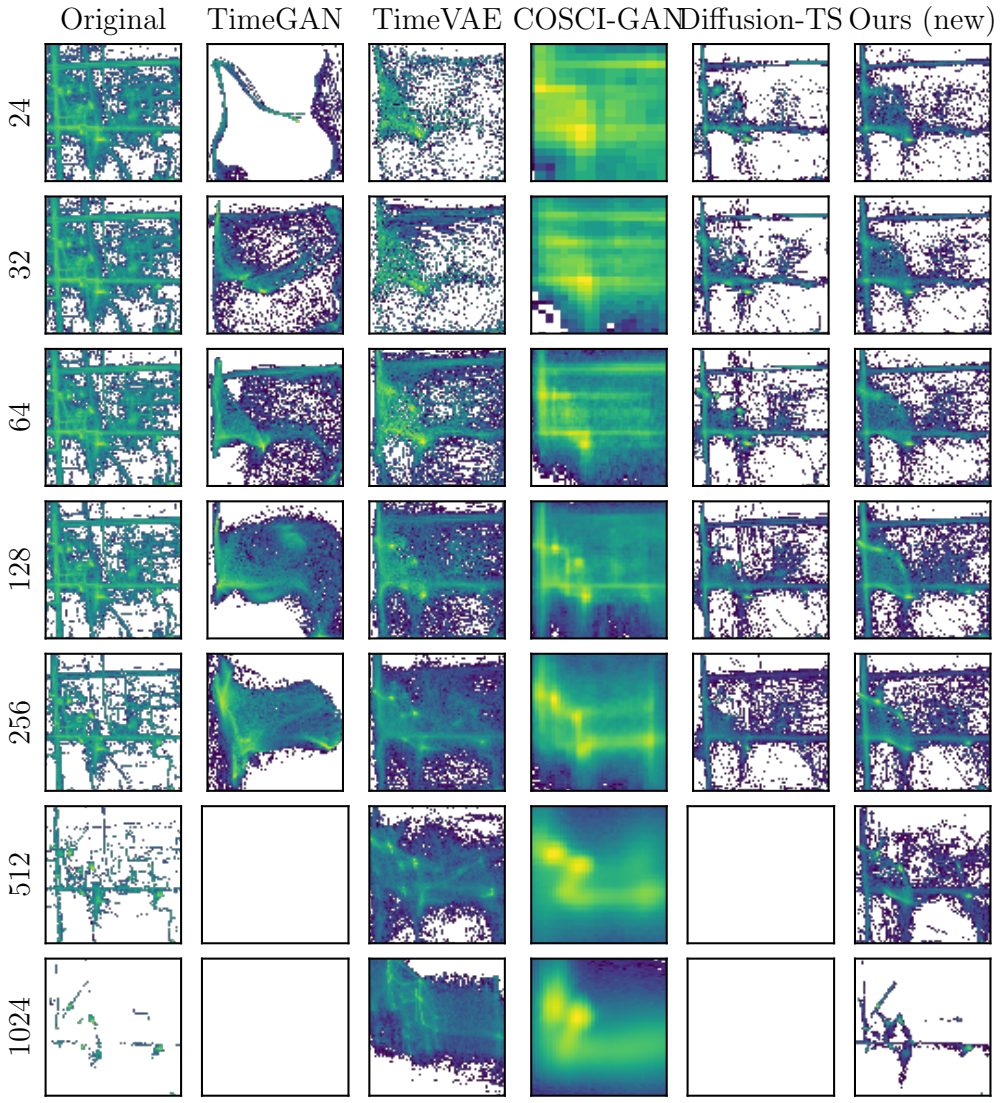

Figure 7: Heatmaps of the unconditionally generated trajectories for Geolife. Training dataset is shown in the left column. Blank figures are unsuccessful experiments (Table 1).