A Template of Research Papers

Peng Cheng
East China Normal University
Shanghai, China
pcheng@sei.ecnu.edu.cn

ABSTRACT

"Usually write this section when are creating an entry in the system, which means that leave this section at the end."
Sentence1: Background of this paper.
Sentence2: What is your point? What is the novelty of your paper? The key point!
Sentence3: What problem you modeled? Briefly introduce your problem.
Sentence4: What is the challenge in your problem?
Sentence5: How you solve it? Any heuristic algorithms or approximation algorithms proposed? If your algorithms have some interesting or theory results, show them here! For example, good approximation ratios, low time complexities, and efficient structures.
Sentence6: Introduce that the experiments have demonstrated your algorithms. For example, "Through extensive experiments, we demonstrate the efficiency and effectiveness of our XXX approaches on both real and synthetic data sets."

1 INTRODUCTION

Time series data is ubiquitous in real-world applications, playing a crucial role in decision-making processes across various domains. From stock price forecasting in finance and patient health monitoring in healthcare to machine condition monitoring in manufacturing and traffic flow optimization in transportation, accurate time series prediction enables users to identify underlying patterns and make informed decisions based on historical data. Over the years, significant progress has been made in time series analysis through the development of deep neural networks, including Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Transformers. These models have demonstrated remarkable capabilities in capturing temporal dependencies and improving prediction accuracy.

Recently, diffusion models have emerged as a powerful generative framework, showing great potential in time series forecasting due to their ability to model complex data distributions. Several diffusion-based approaches have been proposed to enhance time series prediction by iteratively refining noise into structured signals. However, time series data often exhibits intricate multi-scale patterns, making it challenging for standard diffusion models to capture both long-term trends and short-term fluctuations effectively. To address this, some studies have integrated trend decomposition techniques with diffusion models, demonstrating improved performance by explicitly modeling different temporal scales (e.g., trend, seasonality, and residuals).

Despite these advancements, existing diffusion-based time series models typically require fixed-length input windows, limiting their flexibility in real-world scenarios where time series may vary in length. A recent study by Ilan Naiman et al. explored a unified diffusion model capable of handling variable-length time series by transforming sequential data into image-like representations. While promising, this approach may not fully exploit the inherent temporal structures of time series data.

To overcome these limitations, we propose a novel diffusion model framework that (1) adapts to time series of varying lengths without requiring fixed window sizes and (2) incorporates multi-scale trend decomposition to enhance prediction accuracy by separately modeling different temporal patterns. Our method leverages the generative power of diffusion models while ensuring robustness across diverse time series lengths and scales. Experimental results on real-world datasets demonstrate that our approach outperforms existing diffusion-based and non-diffusion-based baselines, providing more accurate and reliable forecasts.

The contributions of this work are summarized as follows:

- We introduce a flexible diffusion model architecture capable of processing time series of arbitrary lengths, eliminating the constraint of fixed input windows.
- We integrate multi-scale trend decomposition into the diffusion process, enabling explicit modeling of both long-term trends and fine-grained fluctuations.
- We conduct extensive experiments on multiple real-world datasets, demonstrating the superiority of our method over state-of-the-art time series forecasting models.

This paper is organized as follows: Section 2 reviews related work on time series forecasting and diffusion models. Section 3 presents our proposed framework in detail. Section 4 describes the experimental setup and results. Finally, Section 5 concludes the paper and discusses future research directions.

2 RELATED WORK

Time series forecasting has undergone three evolutionary phases: statistical modeling, deep learning revolution, and the emerging diffusion paradigm. We critically analyze these paradigms with a focus on their capabilities and limitations.

2.1 Statistical and Classical Machine Learning Methods

Early approaches relied on mathematical assumptions about temporal patterns:

- **Linear Models**:
  - ARIMA [1] dominated linear time series analysis, with SARIMA extending to seasonal data
  - VAR [2] introduced multivariate linear dependence
- **Nonparametric Smoothing**:
  - Holt-Winters [3] empirically captured trend-seasonality interactions

Table 1: Comparative Analysis of Time Series Forecasting Approaches

Paradigm	Method	Model/Technique	Variable-Length Support	Multi-Scale Modeling	Key Innovations	Limitations & Challenges	
Traditional Statistical	ARIMA/SARIMA	Auto-regressive + Moving Average + Differencing	No	No	Handles linear trends and seasonality	Only works for stationary linear data	fails on complex nonlinear patterns
	VAR	Vector Auto-regression	No	No	Multivariate joint modeling	Computational complexity grows exponentially with variables	
	Holt-Winters	Triple Exponential Smoothing	No	Partial (Seasonal only)	Intuitive trend-seasonality decomposition	Requires predefined seasonal period parameters	
	Gaussian Processes	Kernel-based temporal modeling	Yes	No	Probabilistic forecasting with uncertainty quantification	Sensitive to kernel choice	computationally expensive for big data
Deep Learning	LSTM/GRU	Gated Recurrent Networks	Yes	No	Solves long-term dependency problems	Difficult to parallelize	gradient vanishing in long sequences
	DeepAR	Auto-regressive LSTM + Probabilistic output	No	No	Probabilistic time series forecasting	Slow autoregressive generation	error accumulation
	WaveNet/TCN	Dilated Causal Convolutions	Yes	Partial (Depends on dilation)	Parallel training and long-range capture	Receptive field limited by dilation rates	high memory usage
	Transformer	Self-attention mechanism	Yes	No	Global dependency modeling	O(n²) computational complexity	struggles with long sequences
	Informer	Sparse Attention	Yes	No	Reduces attention computation complexity	Sparse patterns may lose local details	
	Autoformer	Auto-correlation mechanism	Yes	Partial (Seasonal decomp)	Frequency-domain time series modeling	Requires predefined seasonal periods	
Diffusion Models	TimeGrad	Diffusion process + RNN conditioning	No	No	First TS diffusion framework	Slow autoregressive generation	no variable-length support
	CSDI	Conditional score-based diffusion	No	No	Unified imputation and forecasting	Dual Transformer architecture computationally expensive	
	SSSD	Structured State Space + Diffusion	Yes	No	Continuous-time modeling and efficient long-range capture	State space dimensions require tuning	
	TSDiff	Time-series-as-image representation	Yes	No	Variable-length input handling	2D representation may disrupt temporal locality	
	DiffSTG	Graph Neural Network + Diffusion	No	No	Spatio-temporal graph structure modeling	Graph construction relies on prior knowledge	
	mr-Diff	Multi-resolution trend decomposition + Diffusion	Yes	Yes	Explicit multi-scale modeling & progressive coarse-to-fine generation	High training complexity from multi-stage approach	

  - ETS framework [4] provided a unified error-trend-seasonality formulation
- **Probabilistic Approaches**:
  - Gaussian Processes [5] enabled Bayesian uncertainty quantification

While interpretable, these methods face fundamental limitations in modeling nonlinear dynamics and high-dimensional dependencies.

2.2 Deep Learning Revolution

Neural networks transformed forecasting through hierarchical feature learning:

2.2.1 Sequential Modeling.

- LSTM [6] overcame gradient vanishing via gated mechanisms
- GRU [7] simplified gating while preserving performance
- DeepAR [8] integrated autoregressive LSTMs with probabilistic outputs

2.2.2 Parallelizable Architectures.

- WaveNet [9] used dilated convolutions for long-range dependencies
- TCNs [10] established residual connections for efficient training

2.2.3 Attention-Based Paradigms.

- Transformer [11] enabled global dependency modeling via self-attention
- Informer [12] addressed quadratic complexity through sparse attention
- Autoformer [13] replaced attention with frequency-domain autocorrelation

2.3 Diffusion Models in Time Series

Building on deep learning successes, diffusion models have emerged as powerful generative tools:

2.3.1 Foundational Works.

- TimeGrad [15] combined diffusion with RNN-based conditioning
- CSDI [16] adapted score-based diffusion for missing value imputation

2.3.2 Architectural Innovations. Recent advances address key limitations:

- SSSD [17] integrated state space models for efficient long-range modeling
- TSDiff [18] encoded variable-length series as 2D representations
- DiffSTG [19] incorporated graph structures for spatio-temporal data
- mr-Diff [20] pioneered multi-scale decomposition via seasonal-trend analysis

Critical Gap: Existing diffusion approaches exhibit three key limitations: (1) rigidity in handling variable-length inputs, (2) absence of explicit multi-scale modeling, and (3) computational inefficiency in inference - which our framework systematically addresses through... [briefly preview your method's innovations].

3 BACKGROUND

3.1 Problem Statement

Time series forecasting aims to predict future values $\mathbf{x}^0_{1:H} \in \mathbb{R}^{d\times H}$ given past observations $\mathbf{x}^0_{-L+1:0} \in \mathbb{R}^{d\times L}$, where d is the number of variables, H is the forecast horizon, and L is the lookback window length. The core challenge lies in modeling the conditional distribution:

$$p(\mathbf{x}^0_{1:H} \mid \mathbf{x}^0_{-L+1:0}), \tag{1}$$

which captures the temporal dependencies in the data. Traditional approaches (e.g., ARIMA, RNNs) often assume simple parametric forms or struggle with high-dimensional, non-Gaussian distributions.

3.2 Diffusion Models

Diffusion models [**?**] are latent variable models with forward (noising) and backward (denoising) processes.

3.2.1 Forward Diffusion. Given input $\mathbf{x}^0$, the forward process gradually adds Gaussian noise over K steps:

$$q(\mathbf{x}^k \mid \mathbf{x}^{k-1}) = \mathcal{N}(\mathbf{x}^k; \sqrt{1-\beta_k}\mathbf{x}^{k-1}, \beta_k\mathbf{I}), \quad k = 1, \dots, K, \tag{2}$$

where $\beta_k \in [0,1]$ controls noise scales. A closed-form for step k is:

$$\mathbf{x}^k = \sqrt{\bar{\alpha}_k}\mathbf{x}^0 + \sqrt{1-\bar{\alpha}_k}\epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{3}$$

with $\bar{\alpha}_k = \prod_{s=1}^{k}(1-\beta_s)$.

3.2.2 Reverse Denoising. The backward process learns to iteratively denoise:

$$p_\theta(\mathbf{x}^{k-1} \mid \mathbf{x}^k) = \mathcal{N}(\mathbf{x}^{k-1}; \mu_\theta(\mathbf{x}^k, k), \sigma_k^2\mathbf{I}), \tag{4}$$

where μ_θ is parameterized by a neural network. Two training strategies exist:

- **Noise prediction**: Minimize $\mathcal{L}_\epsilon = \mathbb{E}_{k,\mathbf{x}^0,\epsilon}\|\epsilon - \epsilon_\theta(\mathbf{x}^k, k)\|^2$.
- **Data prediction**: Minimize $\mathcal{L}_\mathbf{x} = \mathbb{E}_{\mathbf{x}^0,\epsilon,k}\|\mathbf{x}^0 - \mathbf{x}_\theta(\mathbf{x}^k, k)\|^2$.

3.3 Conditional Diffusion Models

For time series prediction, the denoising process is conditioned on past observations $\mathbf{c} = \mathcal{F}(\mathbf{x}^0_{-L+1:0})$ via:

$$p_\theta(\mathbf{x}^{0:K}_{1:H} \mid \mathbf{c}) = p_\theta(\mathbf{x}^K_{1:H}) \prod_{k=1}^{K} p_\theta(\mathbf{x}^{k-1}_{1:H} \mid \mathbf{x}^k_{1:H}, \mathbf{c}), \tag{5}$$

where $\mathcal{F}$ is a conditioning network (e.g., CNN or Transformer). The conditional mean $\mu_\theta(\mathbf{x}^k, k \mid \mathbf{c})$ leverages both noisy input and context.

3.4 Time Series to Image Transforms

To enhance diffusion models, time series are often mapped to images via invertible transforms:

- **Delay Embedding**: For univariate series $x_{1:L}$, construct matrix:

$$X = \begin{bmatrix} x_1 & x_{m+1} & \cdots \\ \vdots & \ddots & \vdots \\ x_n & \cdots & x_L \end{bmatrix} \in \mathbb{R}^{n \times q}, \tag{6}$$

  where m, n are user-defined parameters and $q = \lfloor (L-n)/m \rfloor$.

These transforms enable diffusion models to leverage spatial inductive biases for improved generation.

4 SOLUTION1

Paragraph 1: what is the general idea of the solution1? Greedy based? DP? Generally introduce solution1.

4.1 some properties

Do you have any properties about the problem to use for developing the algorithm? Show them in this subsection with lemmas.

Or you need define some special values to ease your algorithm description, do it here!

You may need to write some equations. Here I show some example equations styles.

1. Multiple Equations with numbering:

$$A = B \tag{7}$$
$$B = C. \tag{8}$$

2. Single equation with numbering:

$$A = B \tag{9}$$

3. Simple equation without numbering:

$$A = B$$

4. Align equations:

$$\begin{aligned} & function(A) \\ = \; & A^3 + A^2 + A^1 \end{aligned} \tag{10}$$

5. Equation with more than one conditions:

$$A = \begin{cases} A + 1, & A \neq 0 \\ A, & A = 0 \end{cases} \tag{11}$$

6. Linear Programming:

Algorithm 1: ExampleAlgorithm

Input: A set C of n workers, and a set R of m riders
Output: A set of updated scheduling sequences S

foreach $r_i \in R$ **do**
  retrieve a list C_i of workers that are valid to r_i
while $C_i \neq \emptyset$ **do**
  if *rider* r_i *can be arranged in* c_j **then**
    do if.
    break
  else if r_i *can replace rider* r'_i *of* c_j **then**
    do else.
    break
do
  Eample of do-while-loop
while *condition*
return $\mathbb{S}$

$$\begin{aligned} \max \quad & \sum \lambda_{ik} \cdot x_{ik} \\ \text{s.t.} \quad & d(u_i, v) \cdot x_{ik} \leq r, \quad i = 1, \ldots, m; k = 1, \ldots, q, \\ & \sum_{k=1}^{q} x_{ik} \leq 1, \quad i = 1, \ldots, m, \end{aligned} \tag{12}$$

4.2 proposed algorithm

Your algorithm needs an interesting name! Stop simply calling it "the Greedy algorithm" or "the Dynamic Programming Based Algorithm"! Do not use these boring names, please!

Introduce the details of your algorithms with natural language. Below is an example of pseudocode.

4.3 theory analyses

Analyze the approximation ratios, competitive ratios, time complexities here.

5 SOLUTION2

Paragraph1: Explain why you propose solution2. What point will solution2 improve compared with solution1?

The rest is similar to what you have done in solution1 section.

6 EXPERIMENTAL STUDY

6.1 Data Set

In introduce what data sets you have used in your experimental study. Real dataset first, then synthetic dataset.

6.2 Approaches and Measurements

What approaches tested in the experimental study? Introduce each one with one sentence. Your proposed approaches, the compared existing approaches, random approaches, and so on.

Why need this paragraph? Readers may have forgotten your solutions, thus just make them recall your solutions through some simple descriptions.

Table 2: Experimental Settings.

Parameters	Values
the number, m, of riders	1K, **3K**, 5K, 8K, 10K
the number, n, of vehicles	100, **200**, 300, 400, 500
the pickup deadline range $[rt^-_{min}, rt^-_{max}]$	**[1, 10]**, [10, 30], [30, 60]
the capacity of vehicles a_j	2, **3**, 4, 5
the balancing parameters (α, β)	(0, 0), (1, 0), (0, 1), **(0.33, 0.33)**
the flexible factor ε	1.2, **1.5**, 1.7, 2
the length δ_j of time frame f_j	30 mins

Then introduce the measures you will compare on, such as running time, memory usage, and so on. Usually, control variate method will be used in the experimental study. Define a default setting, then change one parameter in a set of experiments. Make a table to show the configuration, like the example in Table 2 [**?**].

Finally, introduce the running environment of your experimental study. Running on what kind of PCs or servers? Using what programming languages?

6.3 Experimental Results

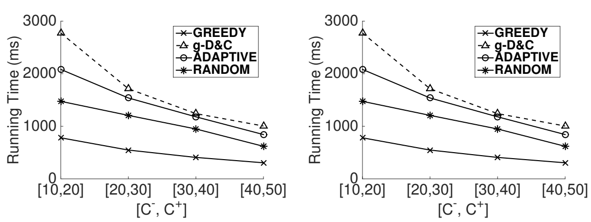
What do you discover or demonstrated in the experimental study? Show the effects of each parameter one by one. Usually, describe results on real dataset first.

Better to have a summary of the interesting points found in the experimental study at the end of this subsection.

6.4 Adding Figures

In this subsection, some example of adding figures are introduces. Usually, put figures on the top of pages. Put figures close to their description.

Below is an example of adding two figures together. Modify the related vspace configuration in head.tex to adjust the global setting. To adjust locally, just add your local spacing command behind them. Parameter "h" means "here"; "t" means "top"; "!" means "mandatory". In vspace command, I like to use "ex", which just a unit and you can use others like "px". For other parameters, you can adjust them to see the effects.



(a) Sub Caption1 (b) Sub Caption2

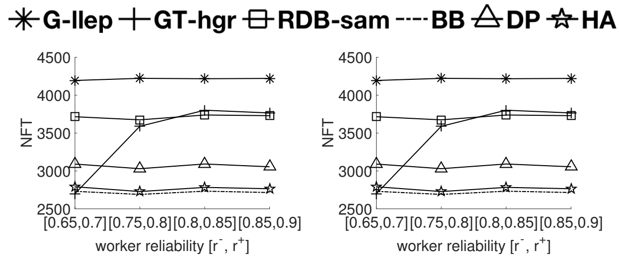
Figure 1: An Example of Figure.

Add a bar on top of a group of subfigures, like in Figure 2.

To draw a figure crossing two columns, like in Figure 3.

Some suggestions:

- Label sub-figures and figures separately. When you describe the particular figure, your can accurately refer to the one you want refer to.
- Use eps files! If you need to convert jpg or png to eps, you can try this website: https://www.online-convert.com, which is the most stable one I can find.
- Put figures on the top of pages for better layout.
- Put figures close to their description to ease your readers.



(a) Sub Caption1 (b) Sub Caption1

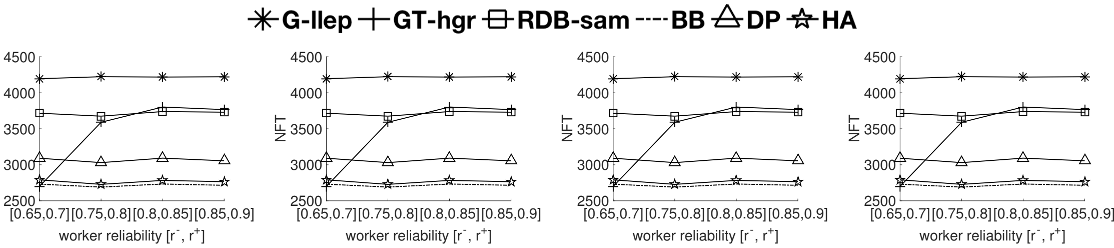
Figure 2: An Example of Figure.

Some suggestions/lessons noticed from helping junior PhD students to revise their drafts:

- Try to avoid using words like "naive", "obviously", "simple" and so on.
- Don't forget leaving a blank space before (. Bad example: abs(e.g., ddd)
- Use aaa [**?**] in stead of aaa [**?**], as the first one can avoid strange [**?**] new line.
- Any Tables, Figures, Algorithms or other similar components in your draft should be discussed with some natural language. Don't simply put them in your draft as no one know their purposes unless you describe them clearly.
- Put blocks of tables, figures, algorithms on the top of pages for better layout.
- Numbered equations should be referred somewhere in your draft. If not, just don't number them. You can add "\notag" behind an equation to suppress its numbering.
- some special commands to use: $log \longrightarrow \log$.
- Avoid starting a sentences with words: "And", "So"... You can use "In addition", "Moreover", "Thus", "Therefore" ...
- Use tools to double check grammar errors, such as Textidote: https://github.com/sylvainhalle/textidote
- Before submitting draft, search: "??" in the pdf file to avoid wrong references or citations.

Experience from Wangze NI after revising drafts with me:

- Do not introduce any technical details in the introduction section.
- Explain the semantic meaning (or usage) of theorems after you prove them.
- Take a running example for each algorithm.'
- Better to use the same example to explain all concepts/algorithms in the paper.
- Do not write sentences too long. Try to cut long sentences into several short sentences when you are not masters in English writing.
- Before writing the paper, it would be helpful to make a presentation to friends. During the preparation of the presentation, organize the content more logically. After presentation, get some feedback from friends.
- Use the same word for one thing/concept in the whole paper. For the important actions, please keep the verbs on describing them consistently.
- Conclude the experimental findings at the end of the experimental study section.



(a) Sub Caption1 (b) Sub Caption2 (c) Sub Caption3 (d) Sub Caption4

Figure 3: An Example of Figure.

7 CONCLUSION

Just describe what are studied in this paper. Do not talk about related work anymore.

8 ACKNOWLEDGMENT

If you think that this template helped you on writing research papers and you work on the related topics, please help to cite some of my publications [**? ? ? ? ? ? ? ? ? ?**]. Thanks a lot!

REFERENCES

[1] E. Stellwagen and L. Tashman, “Arima: The models of box and jenkins,” *Foresight: The International Journal of Applied Forecasting*, pp. 28–33, 2013.

[2] E. Zivot and J. Wang, *Vector Autoregressive Models for Multivariate Time Series*, pp. 369–413. New York, NY: Springer New York, 2003.

[3] A. B. Koehler, R. D. Snyder, and J. Ord, “Forecasting models and prediction intervals for the multiplicative holt–winters method,” *International Journal of Forecasting*, vol. 17, no. 2, pp. 269–286, 2001.

[4] D. J. Hand, “Forecasting with exponential smoothing: The state space approach by rob j. hyndman, anne b. koehler, j. keith ord, ralph d. snyder,” *International Statistical Review*, vol. 77, no. 2, pp. 315–316, 2009.

[5] C. E. Rasmussen and C. K. I. Williams, “Gaussian processes for machine learning (adaptive computation and machine learning),” *The MIT Press*, 2005.

[6] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *Computer Science*, 2014.

[8] V. Flunkert, D. Salinas, and J. Gasthaus, “Deepar: Probabilistic forecasting with autoregressive recurrent networks,” *International Journal of Forecasting*, vol. 36, no. 3, 2020.

[9] A. V. D. Oord, S. Dieleman, H. Zen, K. Simonyan, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” 2016.

[10] S. Bai, J. Z. Kolter, and V. Koltun, “Trellis networks for sequence modeling,” 2018.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *arXiv*, 2017.

[12] H. Zhou, S. Zhang, J. Peng, S. Zhang, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” 2020.

[13] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting,” 2021.

[14] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” *JMLR.org*, 2015.

[15] K. Rasul, C. Seward, I. Schuster, and R. Vollgraf, “Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting,” 2021.

[16] Y. Tashiro, J. Song, Y. Song, and S. Ermon, “Csdi: Conditional score-based diffusion models for probabilistic time series imputation,” 2021.

[17] J. M. L. Alcaraz and N. Strodthoff, “Diffusion-based time series imputation and forecasting with structured state space models,” *Transactions on Machine Learning Research*, 2022. Featured Certification.

[18] I. Naiman and N. Berman, “Utilizing Image Transforms and Diffusion Models for Generative Modeling of Short and Long Time Series,”

[19] H. Wen, Y. Lin, Y. Xia, H. Wan, Q. Wen, R. Zimmermann, and Y. Liang, “Diffstg: Probabilistic spatio-temporal graph forecasting with denoising diffusion models,” in *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, SIGSPATIAL ’23, (New York, NY, USA), Association for Computing Machinery, 2023.

[20] L. Shen, W. Chen, and J. T. Kwok, “MULTI-RESOLUTION DIFFUSION MODELS FOR TIME SERIES FORECASTING,” 2024.