Decoder-only Foundation Model applications on Time Series Forecasting with Air Quality Datasets

Cuong Anh Pham
Machine Learning Department, MBZUAI
cuong.pham@mbzuai.ac.ae

Abstract

Time series forecasting is critical for air quality monitoring, enabling proactive public health measures. In this study, we adapt TimesFM, a novel decoder-only foundation model pre-trained on diverse time series data, for forecasting air quality metrics across three heterogeneous datasets from three urban areas in Asia and Europe: Beijing, Seoul, and Dublin. The experiments show a promising result of finetuning TimesFM with additional layers for air quality prediction tasks, however, the efficiency of this method is not too competitive with traditional statistical and deep learning models, raising the need for improvement to achieve both efficiency and accuracy and become the best choice in practice. Our code is available at: https://github.com/pacman-ctm/ml703_timesfm.

1 Introduction

Time series forecasting (TSF) involves predicting future values or trends of an event, based on historical data recorded at different time intervals. Air pollution is a significant global concern impacting human health and environmental quality [13]. Reliable forecasting of air quality through time series forecasting (TSF) can facilitate effective policymaking and public safety measures. Traditional statistical models such as ARIMA and GARCH, although effective, typically handle each dataset individually and often fail to model complex temporal dependencies and nonlinear dynamics characteristic of air quality data.

Early TSF research relied on statistical models such as ARIMA and GARCH [10]. Deep learning architectures have revolutionized the performance of forecasting methods in recent years [10]. Belong with that, foundation models are actively researched nowadays because of their optimistic performance [1, 11]. The advent of foundation models, especially Large Language Models (LLM) with great progress in advanced reasoning, promises applications on real-world problems [3].

An issue with traditional methods for TSF is that although effective, they are trained individually for each time series and often struggle with capturing long-range dependencies and non-linear relationships inherent in air quality data. For more efficient forecasting, TimesFM [4] was proposed as a decoder-only foundation model that was pre-trained on a massive corpus of time-series data, which allows the model to learn general temporal patterns and perform zero-shot forecasting.

This study is motivated by the need to: (i) Understand urban air pollution through robust statistical inference; (ii) Evaluate foundation models' effectiveness and efficiency in handling statistical forecasting tasks.

Motivated by the strength of TimesFM [4], we aim to capture complex temporal dependencies in environmental data and improve prediction accuracy. However, we still do not know if this approach works well with air quality datasets. Because of that, we propose an application project that utilizes, improves, and evaluates TimesFM on multiple air quality datasets. In detail, we will:

- Apply and evaluate on various Air Quality Datasets to confirm the efficiency of TimesFM.

- Propose experiments to benchmark the performance of applying TimesFM with Air Quality Datasets compared to statistical & machine learning, and traditional deep learning approaches.

2 Literature Reviews

2.1 Classical Statistical Forecasting

In the early stage of TSF, ARIMA variants remain a staple for regulatory reporting owing to their transparency and low data requirements [2, 10]. GARCH extensions capture conditional randomness in pollutant volatility, but both families presume linear dynamics and short memory. Kalman-filter state-space formulations partly alleviate these issues yet require expert tuning [7].

2.2 Deep Learning-based TSF

With the rise of Deep Learning and its performance, it has gradually replaced the traditional statistical methods. For example, CNN–LSTM hybrids extract local spatial patterns from sensor grids before temporal modelling [9]. Graph neural networks integrate road-network topology and wind trajectories but suffer from scalability limits on city-scale meshes. Transformer variants, such as Temporal Fusion Transformer [12]) employ multi-head attention and gating to capture variable importance, or Informer's ProbSparse attention reduces $\mathcal{O}(T^2)$ cost to $\mathcal{O}(T \log T)$, enabling week-ahead horizons [18, 19].

Despite strong accuracy, these deep learning-based models are dataset-specific and must relearn common seasonalities from scratch.

2.3 Foundation Models and Zero-shot TSF

Motivated by the rapid progress of Large Models in Natural Language Processing and recently Computer Vision, Large Models are also proposed for the used of TSF. Recently, large pre-trained sequence models such as TimeGPT [6] and TimesFM [4] were proposed to adapt the LLMs to continuous signals (such as time series) with masked-span prediction, next-token modeling, and rotary embeddings. They use billions of series for the training process and unlock zero-shot or few-shot transfer.

However, these benchmarks mostly focus on electricity load, finance, and weather; and comprehensive studies on air quality still remain missing. In this project, we bridge the gap by providing an evaluation of a fine-tuned foundation TSF model on urban air pollution.

2.4 Evaluation Protocols and Metrics

Standard metrics for pollutant forecasting include MAE, RMSE, and R^2,

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Root Mean Squared Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Coefficient of Determination (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the mean of the true values.

However, extreme-event monitoring demands tail-sensitive scores such as CRPS and peak-recall [14]. Normally, the benchmark approach typically reports 6, 24, and 72 hours horizons; longer windows exacerbate error accumulation in the autoregressive process. Besides, foundation models are always stated about their efficiency, and also from the motivation of the need to have a good inference time, we will try to figure out if the fine-tuning foundation model approach will work well in practice or not. We follow these recommendations and additionally record compute-time and training-data size to quantify the efficiency of these methods.

Our analysis also revisits the trade-off between computational cost and inference accuracy in the fine-tuned foundation models, which is also motivated by the debates in latent ODE forecasting [15] and climate downscaling [16].

3 Methodology

3.1 Problem Formulation

Let $\{\mathbf{x}_t\}_{t=1}^{T}$ denote a multivariate time series where, at each timestamp t, the vector $\mathbf{x}_t \in \mathbb{R}^d$ contains p target pollutants (e.g. PM2.5, PM 10, NO_2) and $q = d - p$ exogenous drivers (e.g. temperature, humidity, wind speed, calendar features). For a sliding window of length L and a forecasting horizon H we define the input–output pair at reference time t as

$$\mathbf{X}_t^{(L)} \;=\; \big[\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_t\big] \in \mathbb{R}^{d\times L}, \qquad \mathbf{Y}_t^{(H)} \;=\; \big[\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+H}\big] \in \mathbb{R}^{d\times H}. \tag{1}$$

Goal. Learn a mapping $\mathcal{F}_\theta : \mathbb{R}^{d\times L} \to \mathbb{R}^{d\times H}$ with parameters θ that produces forecasts $\hat{\mathbf{Y}}_t^{(H)} = \mathcal{F}_\theta\big(\mathbf{X}_t^{(L)}\big)$ such that the statistical discrepancy between $\hat{\mathbf{Y}}_t^{(H)}$ and the ground-truth $\mathbf{Y}_t^{(H)}$ is minimised over all windows in the training corpus.

Learning Objective. We cast forecasting as supervised learning and minimise an empirical risk

$$\min_{\theta}\; \mathcal{L}(\theta) \;=\; \frac{1}{N}\sum_{t=L}^{T-H} \ell\big(\hat{\mathbf{Y}}_t^{(H)}, \mathbf{Y}_t^{(H)}\big) \;+\; \lambda\,\Omega(\theta), \tag{2}$$

where $\ell(\cdot,\cdot)$ is the loss function, such as mean-squared error (MSE) or mean-absolute error (MAE). $\Omega(\boldsymbol{\theta})$ is the regulization parameter with $\lambda \geq 0$ denoted its strength, and $N = T - L - H + 1$ is the number of training windows.

3.2 TimesFM

TimesFM [4] is a novel time-series foundation model based on a decoder-only attention architecture, pretrained on a large and diverse corpus including both real-world and synthetic time-series data. Our project focuses on analyzing around TimesFM, and this is the brief information about the architecture and the key claims of TimesFM.

The key target of TimesFM is to generally propose a zero-shot forecaster that takes in the past C time-points of a time-series as context and predicts the future H time-points. Let the context be denoted by $\mathbf{y}_{1:L} := \{y_1, \cdots, y_L\}$ where indices follow numpy-like notations. Similarly, the actual values in the horizon are denoted by $\mathbf{y}_{L+1:L+H}$. The task is then to learn a foundation model that can map any time-series context to horizon:

$$f : (\mathbf{y}_{1:L}) \longrightarrow \hat{\mathbf{y}}_{L+1:L+H} \tag{3}$$

3.2.1 Architecture Components

The architecture of TimesFM includes three main components:

- **Input Layers:** These layers preprocess the time-series data into input tokens. The input is broken down into non-overlapping patches. Each patch is then processed by a residual block into a vector (size = model dimension). A binary padding mask is also provided to the model.

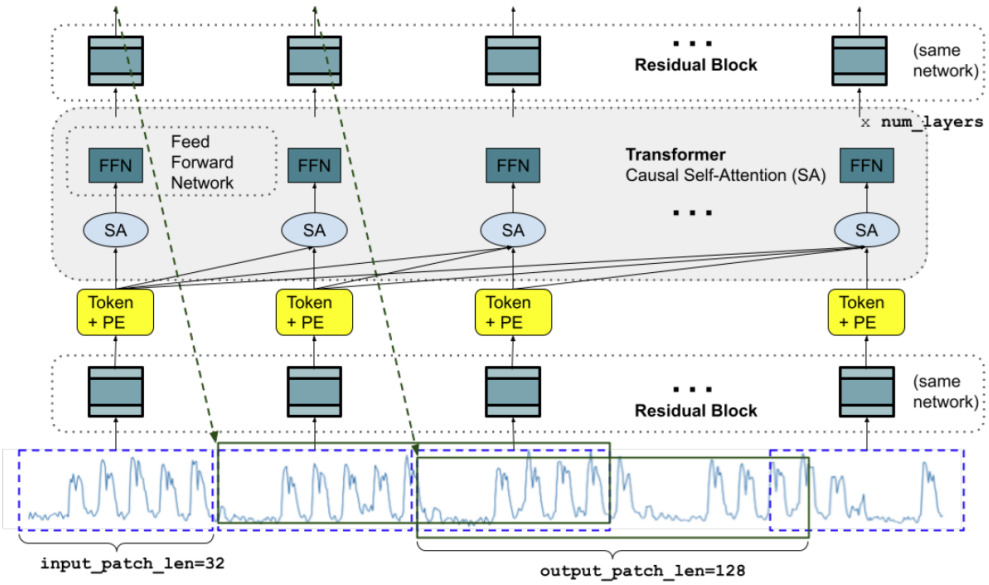

Figure 1: The training architecture of TimesFM (from the original paper [4])

- **Stacked Transformer:** This is the core of the model, comprising multiple transformer layers stacked on top of each other. Each layer includes multi-head self-attention (SA) and a feed-forward network (FFN). Causal attention is used, ensuring that each output token only attends to previous input tokens.
- **Output Layers:** These layers map the output tokens into predictions. A residual block is used to map the output tokens to the predictions of size is the length of the output patch, which is the forecast for the time window following the last input patch seen by the model so far.

3.2.2 Key Characteristics

The key characteristics claimed by the work of TimesFM [4] include:

- **Zero-Shot Forecasting:** The goal of TimesFM is to create a model that performs well without any fine-tuning. This approach will not require any task-specific fine-tuning, **and we will use this to be the hypothesis for our experiments with TimesFM.**
- **Decoder-only with Input Patching:** The model architecture leverages a decoder-only transformer, similar to those used in LLMs, but it involves breaking down the time-series into smaller patches to be the input for Transformer. This approach improves inference speed, **and we will use this to be the hypothesis for our experiments with TimesFM.**
- **Large and Diverse Pre-training Corpus:** TimesFM is pre-trained on both real-world and synthetic time-series datasets to gain diversity, sufficient volume, and cover various temporal patterns and granularity. TimesFM also shows the effect of synthetic data for the time-series forecasting training process.
- **Longer Output Patches:** Typical LLMs only generate one token at a time, but TimesFM uses longer output patches for prediction, and they also achieved better accuracies than the traditional one-token approach.
- **Patch Masking:** The work claimed that with proper masking during training, the model can see all possible context lengths, starting from 1 to the maximum length.
- **Scaling Laws:** TimesFM also showed promising results with model-size and FLOPS increasing.

These characteristics make TimesFM a promising candidate for air quality forecasting, where datasets are heterogeneous, and computational efficiency is critical for real-time applications.

3.2.3 Pre-Training Strategy

For the training of TimesFM, they used multiple large-scale datasets:

- **Real-world Datasets:** Google Trends [1] (search interest data for a wide range of queries across different time granularities), Wiki Pageviews [2] (Hourly pageview statistics from Wikimedia, aggregated into different granularities).
- **Synthetic Datasets:** ARMA processes, seasonal patterns, trends, and step functions.

The diversity and volume of this data are crucial for the model to learn generalizable temporal patterns, handle varying time granularities, and adapt to different forecasting domains. The inclusion of synthetic data also helps the model generalize to underrepresented frequencies.

3.3 Approach

We adapt TimesFM, a decoder-only transformer model, for a specific task: air quality forecasting. As the conclusions from the original TimesFM paper, we will use two main characteristics of TimesFM as the hypotheses in our evaluation:

- We will evaluate if the finetuning TimesFM can work well with a case study - air quality.
- We will evaluate the inference speed of fine-tuned TimesFM on those datasets.

For the fine-tuning process, we will fine-tune TimesFM with two additional Fully-Connected Layers (FCL). This process will update the weights of the FCLs and selectively update the transformer layers using a smaller learning rate for the latter to preserve pre-trained knowledge. This approach mitigates the poor zero-shot performance observed in preliminary experiments. For the loss function for the fine-tuning process, we use the negative log-likelihood loss $\mathcal{L}_{\text{NLL}}(\theta)$:

$$\mathcal{L}_{\text{NLL}}(\theta) \;=\; -\sum_{t=1}^{T} \log p_\theta\big(y_t \mid \mathbf{x}_{1:t-1}\big) \tag{4}$$

with $p_\theta\big(y_t \mid \mathbf{x}_{1:t-1}\big)$ is the model's predictive density for the target y_t given the past context $\mathbf{x}_{1:t-1}$. And for detail, the log-likelihood is:

$$\log p_\theta(y_t \mid \mathbf{x}_{1:t-1}) = -\frac{1}{2}\log(2\pi\sigma_t^2) - \frac{(y_t - \hat{y}_t)^2}{2\sigma_t^2}. \tag{5}$$

with $\hat{y}_t$ and σ_t^2 as the predicted mean and variance, respectively, conditioned on the context $\mathbf{x}_{1:t-1}$. To prevent overfitting, we add L2 regularization to the total loss:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{NLL}}(\theta) + \lambda\|\theta\|_2^2, \tag{6}$$

where λ is the regularization parameter that balances model complexity and generalization, and $\|\cdot\|_2$ denotes the L2-norm.

4 Experiments

4.1 Datasets

For datasets to conduct experiments, we will use these three datasets with a diverse range of environmental conditions:

- **Beijing Multi-Site Air Quality Dataset**[3]: Contains $\sim$ 420,768 rows with 18 columns of pollution features collected in Beijing, China from March 2013 to February 2017. This dataset provides high-resolution pollutant measurements across multiple urban sites.

[1] `https://trends.google.com/trends/`

[2] `https://en.wikipedia.org/wiki/Wikipedia:Pageview_statistics`

[3] `https://archive.ics.uci.edu/dataset/501/beijing+multi+site+air+quality+data`

Table 1: Results for evaluating with Beijing Multi-Site Air Quality Dataset

Model	MSE	RMSE	MAE
ARIMA	0.775	1.123	0.710
RNN	0.720	0.849	0.645
Bi-LSTM	0.512	0.715	0.566
Transformer	0.429	0.655	0.521
TimesFM & 2 FCL	**0.378**	**0.615**	**0.479**

- **Seoul Air Pollution Dataset**[4]: Contains $\sim$ 650k rows with 11 columns of pollution features collected in Seoul, South Korea from January 2017 to December 2019. This dataset captures unique seasonal and diurnal variations typical of a metropolitan region in East Asia.
- **Google Project Air View Data (Dublin City)**[5]: Contains 10 million rows with 7 main columns of pollution features collected in Dublin, Ireland. However, we only use the partition of $\sim$ 25k rows from May 2021 to August 2022. This dataset contains crowdsourced measurements that present a challenging mix of reliability and granularity.

4.2 Experiments Design

Due to the lack of computational resources, we can only conduct experiments by finetuning TimesFM with two more fully-connected layers, and applying the model with the three datasets as above.

We hypothesize that the finetuned TimesFM will achieve comparable or superior forecasting accuracy with reduced computational requirements. All the experiments will be conducted with RTX A6000 or T4-GPU (of Google Colab). Following that, the baselines, metrics and hyperparameters chosen are:

Baselines. We will benchmark the finetuned TimesFM with a statistical method - ARIMA [2], basic deep learning models (RNN [5], Bi-LSTM [8]), and the Transformer-based model [17].

Metrics. We will evaluate the models by the (i) prediction accuracy (Test loss, MAE, MSE, RMSE); (ii) computational efficiency (training time, inference latency, memory usage). To be fair for all datasets, we will calculate the average of all kinds of pollution in each dataset.

Hyperparameters. We trained these models with batch_size = 32, 50 epochs, learning_rate = 0.001 and the drop_out rate (for Transformer) is 0.3. For the regularization of the fine-tuning process, we used $\lambda = 0.001$.

Also, for each dataset, we will first normalize the data and split the train/validation/test as: (i) for the **Beijing dataset**, we splitted the training dataset up to January 2016, the validation set from February 2016 to July 2016, and the test set from August 2016 to February 2017; (ii) for the **Seoul dataset**, we splitted the training dataset up to June 2019, the validation set from July 2019 to September 2019 and the test set from October 2019 to December 2019; (iii) for the **Dublin dataset**, we splitted the validation set in May and June of 2022, the test set in July and August 2022 and the remaining for the training.

4.3 Results

4.3.1 Accuracy Analysis

The results after the experiments are reported in the following Table 1, 2, 3. Note that for ARIMA, we also use grid-search to find the best ARIMA model, and the ARIMA results are based on that best model for each dataset.

Based on these results, it can be seen that after finetuning TimesFM with 2 FCLs, the model consistently outperforms the baseline models in terms of all the accuracy metrics (including MAE, MSE, and RMSE).

In general, the ARIMA shows acceptable results in all three datasets, compared to the worst deep learning model result - RNN, but it is still much worse than the other deep learning methods. We can

[4] `https://www.kaggle.com/datasets/bappekim/air-pollution-in-seoul`

[5] `https://data.gov.ie/dataset/google-airview-data-dublin-city`

Table 2: Results for evaluating with Seoul Air Pollution Dataset

Model	MSE	RMSE	MAE
ARIMA	0.871	0.779	0.601
RNN	0.803	0.896	0.731
Bi-LSTM	0.603	0.776	0.612
Transformer	0.482	0.694	0.551
TimesFM & 2 FCL	**0.421**	**0.648**	**0.499**

Table 3: Results for evaluating with Project Air View Data (Dublin City)

Model	MSE	RMSE	MAE
ARIMA	0.819	0.823	0.536
RNN	0.780	0.883	0.702
Bi-LSTM	0.550	0.741	0.592
Transformer	0.451	0.671	0.530
TimesFM & 2 FCL	**0.447**	**0.634**	**0.516**

also notice that the Transformer is better than RNN and Bi-LSTM. Bi-LSTM shows the better results in most of the metrics across datasets.

For the Beijing dataset (Table 1), the finetuned TimesFM with 2 FCLs has the best performance in terms of all the accuracy metrics (including MAE, MSE, and RMSE) in all three datasets, with around 5% better in each metric compared to the best baseline - Transformer. Similar trends are observed in the Seoul dataset (Table 2), and the Dublin dataset (Table 3) shows more modest gains, likely due to the dataset's higher noise and irregular sampling from crowdsourced measurements.

4.3.2 Computational Efficiency

Table 4 summarizes the average computational efficiency across models, averaged over the three datasets. ARIMA still shows its performance in the speed as it is a statistical method. For the deep learning methods, the inference latency of TimesFM finetuned with 2 FCLs is comparable to the Transformer (0.018ms vs. 0.019ms per sample) and higher than RNN (0.012ms) and Bi-LSTM (0.015ms). In case of one sample, the difference is small, but in practice, with millions or billions of records, this will be days of difference. Memory usage for TimesFM is also higher compared to all three baselines, reflecting the larger model size. Also, notice that we added two FCLs to the finetuned TimesFM, so this model size, in theory, can still be reduced.

4.3.3 About the Experiments Hypotheses

From the above results, the fine-tuned TimesFM & 2 FCL achieves superior accuracy across all datasets, with notable improvements in modeling high-pollution events. This also shows the robustness with the noisy and crowdsourced data (because of the properties of the datasets). This method is promising but also indicates room for improvement in handling sparse or heterogeneous inputs.

However, the pre-trained foundation model (TimesFM) with two additional FCLs increases model size, resulting in inference latency comparable to the Transformer and higher memory usage. It can be concluded that the TimesFM fine-tuned model failed to achieve both the best performance and the best inference speed This accuracy-efficiency trade-off suggests that while TimesFM is effective

Table 4: Computational Efficiency comparison (averaged across datasets)

Model	Inference Latency (ms/sample)	Memory Usage (GB)
ARIMA	0.007	0.5
RNN	0.012	1.8
Bi-LSTM	0.015	2.3
Transformer	0.019	3.2
TimesFM & 2 FCL	0.017	4.4

for high-accuracy forecasting, deployment on resource-constrained devices may require additional techniques to reduce the model size, such as using pruning or quantization.

However, due to computational constraints, we could not evaluate zero-shot TimesFM or explore alternative fine-tuning strategies (such as prompt-tuning). Preliminary experiments with zero-shot TimesFM yielded poor performance, necessitating the fully-connected layers, which improved accuracy but compromised efficiency. Due to the lack of experiments, we can partly (but cannot fully) reject the hypotheses as we stated in Part 3.3 - Approach.

5 Discussions and Conclusions

About the failure of the fine-tuned only TimesFM. Due to the limit in computation usages and also the time pressure, we did not run more experiments with more models and more complex datasets to ensure the statistical claims are practically correct. In fact, we also tried to run experiments with only fine-tuned TimesFM, but the performance is bad, and it is not likely to catch any information from the given datasets, so we decided to add two more fully-connected layers, and it shows promising results in accuracy metrics. However, this will slow the training and also the inference speed because of the larger model size.

Future Works. For the Future Works, the comparable inference latency of fine-tuned TimesFM with 2 FCLs to the Transformer raises another hypothesis about the efficiency of the additional FCLs in the design of the best model, and also raise questions about the differents techniques for finetuning (such as tuning the hyperparameters, trying with different additional layers), we also aim to run more experiments with more different kinds of deep learning/foundation models and more complex datasets to strengthen our evaluation. Besides, we will also try to explain and understand why the finetuning process in our process doesn't bring good performance.

Conclusions. In this project, we conduct analysis and experiments with TimesFM model in terms of fine-tuning for the air quality forecasting task. Based on the results, we have a small claim that the finetuning on this foundation model for Time Series Forecasting has better results, but not too outstanding and proper for practical usage compared to traditional statistical methods and deep learning models. Answering the questions from the 3.3 - Approach part, we noticed that the fine-tuned TimesFM itself cannot work well with the three air quality datasets we used; but with two additional fully-connected layers, the performance is better and even better than the traditional deep learning models. Besides, the inference time for finetuned TimesFM is not very competitive compared to the traditional approaches in the case of the accuracy-efficiency trade-off.

References

[1] Muhammad Awais, Muzammal Naseer, Salman Khan, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, and Fahad Shahbaz Khan. Foundation models defining a new era in vision: a survey and outlook. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01):1–20, 2025.

[2] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[3] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models. 15(3), March 2024. ISSN 2157-6904. doi: 10.1145/3641289. URL `https://doi.org/10.1145/3641289`.

[4] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, pages 10148–10167, 2024.

[5] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[6] Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

[7] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. Global models for time series forecasting: A simulation study. *Pattern Recognition*, 124:108441, 2022.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[9] Taoying Li, Miao Hua, and XU Wu. A hybrid cnn-lstm model for forecasting particulate matter (pm2. 5). *Ieee Access*, 8:26933–26940, 2020.

[10] Wenxiang Li and K. L. Eddie Law. Deep learning models for time series forecasting: A review. *IEEE Access*, 12:92306–92327, 2024. doi: 10.1109/ACCESS.2024.3422528.

[11] Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6555–6565, 2024.

[12] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[13] World Health Organization et al. Ambient air pollution: A global assessment of exposure and burden of disease. *Clean Air Journal*, 26(2):6–6, 2016.

[14] Madhurima Panja, Tanujit Chakraborty, Anubhab Biswas, and Soudeep Deb. E-stgcn: Extreme spatiotemporal graph convolutional networks for air quality forecasting. *arXiv preprint arXiv:2411.12258*, 2024.

[15] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

[16] Saeid A Vaghefi, Christian Huggel, Veruska Muccione, Hamed Khashehchi, and Markus Leippold. Deep climate change: A dataset and adaptive domain pre-trained language models for climate change related tasks. In *NeurIPS 2022 workshop on tackling climate change with machine learning*, 2022.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[18] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pages 11106–11115. AAAI Press, 2021.

[19] Haoyi Zhou, Jianxin Li, Shanghang Zhang, Shuai Zhang, Mengyi Yan, and Hui Xiong. Expanding the prediction capacity in long sequence time-series forecasting. *Artificial Intelligence*, 318:103886, 2023. ISSN 0004-3702.