General Time Transformer: an Encoder-only Foundation Model for Zero-Shot Multivariate Time Series Forecasting

Cheng Feng
cheng.feng@siemens.com
Siemens Technology
Beijing, China

Long Huang
huangl22@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Denis Krompass
denis.krompass@siemens.com
Siemens Technology
Munich, Germany

Abstract

We present General Time Transformer (GTT), an encoder-only style foundation model for zero-shot multivariate time series forecasting. GTT is pretrained on a large dataset of 200M high-quality time series samples spanning diverse domains. In our framework, we consider multivariate time series as a distinct category of images characterized by varying number of channels, and represent each time series sample as a sequence of non-overlapping curve shapes (patches) within an unified numerical magnitude. Furthermore, we formulate the task of multivariate time series forecasting as a problem of predicting the next curve shape based on a window of past curve shapes on a channel-wise basis. Experimental results demonstrate that GTT exhibits superior zero-shot multivariate forecasting capabilities on unseen time series datasets, even surpassing state-of-the-art supervised baselines. Additionally, we investigate the impact of varying GTT model parameters and training dataset scales, observing that the scaling law also applies in the context of zero-shot multivariate time series forecasting. The codebase of GTT is available at https://github.com/cfeng783/GTT.

CCS Concepts

• **Computing methodologies** → **Neural networks**; • **Mathematics of computing** → **Time series analysis**.

Keywords

Multivariate time series forecasting; Transformer; Foundation model; Zero-shot forecast

ACM Reference Format:

Cheng Feng, Long Huang, and Denis Krompass. 2024. General Time Transformer: an Encoder-only Foundation Model for Zero-Shot Multivariate Time Series Forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3627673.3679931

1 Introduction

Time series forecasting, the task of predicting future values of one or multiple variables based on their historical values and other potentially relevant information, holds significant importance across diverse domains including manufacturing, traffic, healthcare, finance, and environmental science. In response to its practical significance, a large variety of time series forecasting methods have been developed. Earlier work includes classic statistical approaches such as ARIMA [3, 6], Exponential Smoothing [13] and VAR [33], as well as those leverage deep sequential models like recurrent neural networks (RNNs) [22] and convolutional neural networks (CNNs) [2]. In recent years, two distinct directions have emerged regarding the utilization of deep neural networks for time series forecasting. On one hand, building on the success of the Transformer architecture [23] in natural language processing (NLP) and computer vision (CV), there has been a surge in leveraging Transformer-like architecture for time series forecasting. Examples include Pyraformer [16], LogTrans [15], Informer [30], Autoformer [26], FEDformer [31], Crossformer [29], iTransformer [17] and PatchTST [20], to name a few. Crossformer and iTransformer, in particular, explicitly exploit cross-channel dependencies that play a vital role in achieving precise multivariate forecasting outcomes. PatchTST, on the other hand, aggregates adjacent data points of time series into patches, thereby enhancing the semantic information of input tokens represented as curve shapes. On the other hand, there is also a different voice that simple multilayer perceptrons (MLP)-like models can achieve similar or even better time-series forecasting performance than sophisticated Transformer-based models [7, 28]. This discrepancy may be attributed to the fact that Transformers tend to overfit small datasets, and that the largest publicly available time series dataset is less than 10 GB [10], which is significantly smaller compared to those in NLP and CV domains.

In this work, inspired by the Transformer scaling successes in NLP and CV domains, we experiment with training a Transformer-based foundation model, which we term General Time Transformer (GTT), for zero-shot multivariate time series forecasting on a large dataset containing 200M high-quality time series samples collected from diverse domains. To overcome the challenges of distribution shift, as well as varying channel dimensions of time series samples across different domains, the task of multivariate time series forecasting is formulated as a channel-wise next curve shape prediction problem within our framework. Specifically, we do not introduce any time-series-specific inductive biases, but instead treat each time series sample as a sequence of non-overlapping curve shapes with a unified numerical magnitude. Each curve shape comprises M consecutive time points of a single variable. GTT is trained to use N preceding curve shapes as the context to predict the next curve shape on a channel-wise basis. We adopt an encoder-only architecture for GTT with the fewest possible modifications to the standard Transformer. The only major modification we introduced is a cross-channel attention stage after the temporal attention stage

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679931

in each multi-head self-attention block to capture cross-variate dependency between channels. GTT employs an auto-regressive approach to handle long-term forecasting tasks extend beyond M time steps.

GTT exhibits excellent zero-shot multivariate time series forecasting performance on various benchmark datasets, even outperforming state-of-the-art supervised baselines in several cases. We also demonstrate that GTT can achieve noticeably improved performance with cost-effective fine-tuning on target datasets. Additionally, we have conducted an investigation into the influence of different scales of GTT model parameters and training datasets, which reveals that scaling law also applies in our context.

2 General Time Transformer

We consider building a general purpose zero-shot multivariate time series forecaster that takes in a look-back window of L time points of a time-series and optionally their corresponding covariates as context, and predicts the future H time points. Let $\mathbf{x}_{1:L}$ and $\mathbf{c}_{1:L}$ be the context time series and corresponding covariates, GTT is a function to predict $\hat{\mathbf{x}}_{L+1:L+H}$, such that $\hat{\mathbf{x}}_{L+1:L+H} = f(\mathbf{x}_{1:L}, \mathbf{c}_{1:L})$.

2.1 Pretraining Data

We collected a large scale time series repository containing 2.4B univariate or multivariate time points from internal source and the Monash Time Series Repository [10]. Our repository consists of about 180,000 univariate or multivariate time series spanning diverse domains including manufacturing, transportation, finance, environmental sensing, healthcare, to name some.

For each series, we take its first 90% time points to extract training samples and the remaining 10% time points to extract validation samples. Each extracted time series sample consists of 1088 consecutive time points without missing values. Our model is trained to predict the values of the last 64 time points using the preceding 1024 time points as context. To achieve a unified numerical magnitude for time series samples across different datasets, we normalize each time series sample on a channel-wise basis. Specifically, We use only the standard normalization part of reversible instance normalization (RevIN) [14], i.e, each time series is scaled by the mean and standard deviation of the context window. Normalized samples that have a data point with an absolute value greater than 9 are discarded to exclude samples containing extreme values. Furthermore, we mask 1 to 960 time points in the beginning of 10% randomly chosen samples to zero values. This manipulation allows us to generate samples with shorter context lengths, providing a variation in the length of context within the training data. Lastly, to ensure a balance between the scale and domain diversity of our training data, we restrict the max number of training or validation samples that can be extracted from a single time series to 60,000. In the end, approximately 200M high quality training samples and 24M validation samples are generated from our repository.

2.2 The Model

An overview of GTT is depicted in Figure 1. Specifically, we split an input multivariate time series sample into fixed-size non-overlapping patches channel-wise. Each patch represents a curve shape composed of 64 time points of a single variable. We linearly embed each of the patches, add position encodings, and then feed the resulting sequence of patches to the encoder. The encoder has an extra channel attention stage compared with the standard Transformer. For parameter efficiency, the temporal and channel attention share the same weights. Weight sharing between multi-head self attentions applied on different input dimensions has been proved effective in [27]. Lastly, we add a linear head to the last token to perform forecasting of the next patch (curve shape). During inference, we apply reversible instance normalization (RevIN) [14] to first normalize the input data into zero mean and unit variance and then padding zeros in the front if the length of input time series is shorter than 1024. The predicted output is denormalized into its original scale by using the pre-calculated mean and variance of the input data. In this way, no normalization is needed for input data before using GTT which significantly improves the convenience of model usage. Furthermore, our encoder-only architecture ensures that the predicted values are normalized strictly using the mean and standard deviation of the entire context window such that all time series samples share an unified numerical magnitude.

It is important to note that upon closer examination, the architectural similarities between GTT and Vision Transformer (ViT) [5] become apparent if curve shapes are thought as special type of image patches. However, a significant distinction arises in their treatment of channels. While ViT combines RGB channels of an image within its patching process, GTT independently processes time series channels and incorporates an additional stage for channel attention, which facilitates the learning of cross-variate dependencies with varying channel numbers.

We now describe the key components of GTT architecture. In the presentation, we use the following notations: B: batch size, T: input time series length, C: number of input channels (number of target variables and covariates in total), O: number of output channels (number of target variables), M: number of patches, P: patch size, D: number of embedding dimensions, N: number of encoder layers.

2.2.1 Patching and Positional Encoding. Let $\mathbf{X} \in \mathbb{R}^{B\times T\times C}$ be the input batch of time series samples, we first reshape $\mathbf{X}$ to $\hat{\mathbf{X}} \in \mathbb{R}^{BC\times T\times 1}$, then utilize a one-dimensional convolutional layer (Conv1D) with kernel size and strides equal to patch size P and number of filters equals to D, to segment input series into patches and then embed them into $M \times D$ dimensional patch embeddings channel-wise. We use the Positional Encoding in the original Transformer paper [23] for encoding position information and add position encodings to the patch embeddings to retain sequential information:

$$\hat{\mathbf{X}} = \text{Reshape}(\mathbf{X}), \quad \mathbf{X} \in \mathbb{R}^{B\times T\times C}, \hat{\mathbf{X}} \in \mathbb{R}^{BC\times T\times 1}$$
$$\mathbf{Z}_0 = \text{Conv1D}(\hat{\mathbf{X}}) + \mathbf{E}_{pos}, \quad \mathbf{E}_{pos}, \mathbf{Z}_0 \in \mathbb{R}^{BC\times M\times D}$$

2.2.2 Encoder Layers. To utilize both temporal and cross channel dependencies for the forecasting task, we apply two multi-head self attentions (MSA), which we call temporal attention (T-MSA) and channel attention (C-MSA) in each encoder layer of GTT. A MLP consisting of two layers with a GELU non-linearity [12] is applied after T-MSA and C-MSA. Layernorm (LN) and residual connections

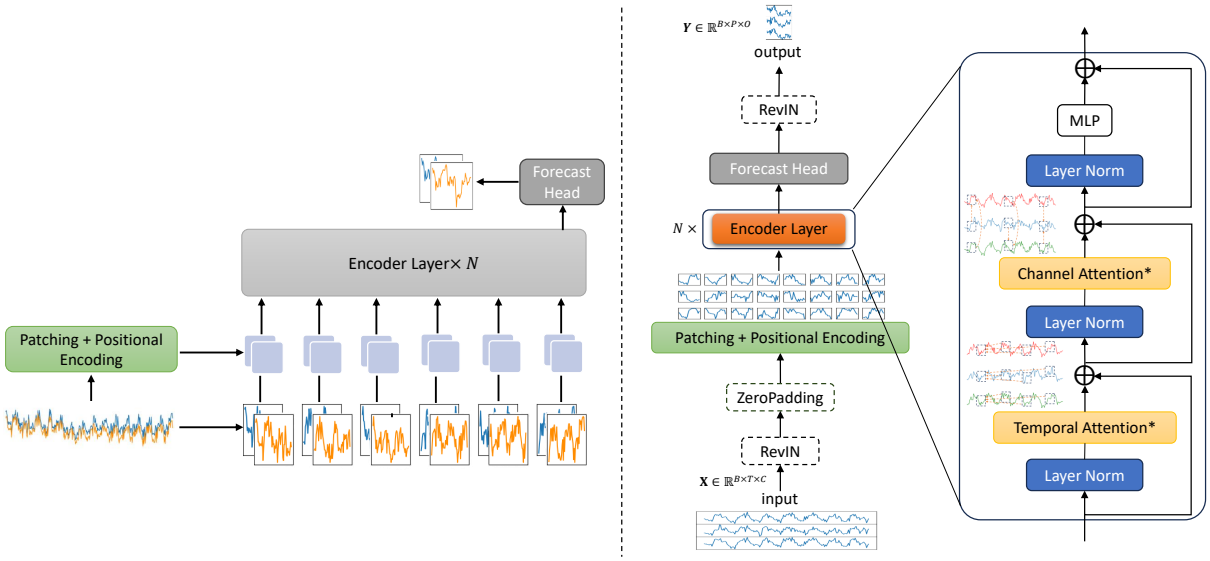


Figure 1: Overview of GTT model structure.

are also applied:

$$
\begin{aligned}
&\mathbf{Z}'_l = \text{T-MSA}(\text{LN}(\mathbf{Z}_{l-1})) + \mathbf{Z}_{l-1}, \quad l = 1, \ldots, N \\
&\hat{\mathbf{Z}}'_l = \text{Reshape}(\mathbf{Z}'_l), \quad \mathbf{Z}'_l \in \mathbb{R}^{BC\times M\times D}, \hat{\mathbf{Z}}'_l \in \mathbb{R}^{BM\times C\times D} \\
&\hat{\mathbf{Z}}''_l = \text{C-MSA}(\text{LN}(\hat{\mathbf{Z}}'_l)) + \hat{\mathbf{Z}}'_l, \quad l = 1, \ldots, N \\
&\mathbf{Z}''_l = \text{Reshape}(\hat{\mathbf{Z}}''_l), \quad \hat{\mathbf{Z}}''_l \in \mathbb{R}^{BM\times C\times D}, \mathbf{Z}''_l \in \mathbb{R}^{BC\times M\times D} \\
&\mathbf{Z}_l = \text{MLP}(\text{LN}(\mathbf{Z}''_l)) + \mathbf{Z}''_l, \quad l = 1, \ldots, N
\end{aligned}
$$

It is important to note that since channel attention requires no positional information, the trained model can generalize to arbitrary channel dimensions in the inference stage

2.2.3 Forecast Head. Following the encoder layers, we retrieve $\mathbf{Z}_N^M$, the last token of the last encoder layer, and then a linear forecast head is attached to $\mathbf{Z}_N^M$ for predicting the next patch of time points for all channels, i.e., the linear head is shared by all channels. Lastly, we retrieve the channels for the target variables as the output:

$$
\begin{aligned}
&\mathbf{Y}' = \mathbf{Z}_N^M W^{D\times P} + \mathbf{b}^P, \quad \mathbf{Z}_N^M \in \mathbb{R}^{BC\times D}, \mathbf{Y}' \in \mathbb{R}^{BC\times P} \\
&\mathbf{Y}'' = \text{Reshape}(\mathbf{Y}'), \quad \mathbf{Y}'' \in \mathbb{R}^{B\times P\times C} \\
&\mathbf{Y} = \text{Retrieve}(\mathbf{Y}''), \quad \mathbf{Y} \in \mathbb{R}^{B\times P\times O}
\end{aligned}
$$

2.2.4 Loss Function. The model is trained to minimize the Mean Absolute Error (MAE) between ground-truth and predicted values. We choose the MAE loss because it is less sensitive to outliers.

Table 1: Details of GTT model variants.

Model	Encoder layers	Embedding Embedding	No. Heads	MLP size	Parameters
GTT-Tiny	4	384	6	1536	7M
GTT-Small	6	512	8	2048	19M
GTT-Large	8	768	12	3072	57M

3 Experiments

3.1 Experimental Settings

Our largest trained model has 57M parameters, which is significantly smaller than those foundation models in NLP and CV domains. Nevertheless, we already observe excellent zero-shot forecasting performance. Details on the GTT model variants are provided in Table 1. All models are trained using the 200M training samples and 24M validation samples as described in Section 2.1. We train all models using the AdamW optimizer [19], training is stopped when the validation loss increases in three consecutive epochs. All GTT variants are trained on a server containing 4 NVIDIA A800 80G GPUs.

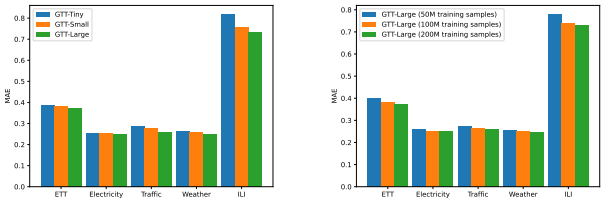
To evaluate the forecasting performance of GTT, we follow the benchmarks used in PatchTST [20]. Specifically, 8 popular datasets, including 4 ETT datasets, Electricity, Traffic, Weather and ILI are used. It is also worthy to mention that all the benchmark datasets are not included in our pretraining data.

3.2 Comparison to Supervised Models

We first compare the zero-shot multivariate forecasting performance of our largest model, GTT-Large, to state-of-the-art supervised forecasting models, including GPT4TS [32], PatchTST [20], Crossformer [29], Fedformer [31], TimesNet [25], DLinear [28],

Table 2: Multivariate time series forecasting. The results are obtained by averaging predictions for four different lengths: 24, 36, 48, and 60 for the ILI dataset, 96, 192, 336, and 720 for other datasets. The best value for each metric is highlighted in red, while the second-best value is highlighted in blue. ZS is short for Zero-Shot and FT is short for Fine-Tune.

Model	GTT (ZS)		GTT (FT)		GPT4TS		PatchTST		Crossformer		Fedformer		TimesNet		DLinear		TSMixer		iTransformer	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	0.398	0.392	0.370	0.383	0.352	0.383	0.353	0.382	0.405	0.424	0.448	0.452	0.400	0.406	0.357	0.379	0.351	0.378	-	-
ETTm2	0.279	0.324	0.253	0.309	0.266	0.326	0.256	0.317	-	-	0.305	0.349	0.291	0.333	0.267	0.332	0.254	0.314	-	-
ETTh1	0.418	0.415	0.420	0.411	0.427	0.426	0.413	0.434	0.457	0.454	0.440	0.460	0.458	0.450	0.423	0.437	0.408	0.430	-	-
ETTh2	0.314	0.359	0.298	0.353	0.346	0.394	0.331	0.381	-	-	0.434	0.447	0.414	0.427	0.431	0.447	0.340	0.387	-	-
Electricity	0.157	0.249	0.155	0.246	0.167	0.263	0.159	0.253	0.305	0.358	0.214	0.327	0.192	0.295	0.166	0.264	0.155	0.251	0.178	0.270
Traffic	0.404	0.260	0.390	0.257	0.414	0.294	0.391	0.264	0.506	0.285	0.610	0.376	0.620	0.336	0.434	0.295	0.386	0.263	0.428	0.282
Weather	0.227	0.247	0.218	0.249	0.237	0.270	0.226	0.264	0.409	0.447	0.309	0.360	0.259	0.287	0.246	0.300	0.224	0.262	0.258	0.279
ILI	1.536	0.732	1.668	0.724	1.925	0.903	1.480	0.807	3.387	1.236	2.847	1.144	2.139	0.931	2.169	1.041	-	-	-	-



(a) (b)

Figure 2: (a): Zero-shot multivariate forecasting performance on benchmark datasets of GTT with different model parameter scales. (b): Zero-shot multivariate forecasting performance on benchmark datasets of GTT-Large with different training data scales. The results for ETT are averaged from the four ETT datasets.

TSMixer [7] and iTransformer [17]. All the above supervised baselines are trained on the train split of each benchmark dataset. Additionally, we also report the performance of GTT (we refer GTT to GTT-Large if not specified hereafter) after fine-tuning on the train split of each benchmark dataset. Note that we only tune the Forecast Head and keep other parameters of GTT fixed during fine-tuning. The results, in terms of Mean Square Error (MSE) and Mean Absolute Error (MAE) on the test split of each benchmark dataset, are given in Table 2. We report the results of the baselines directly from their original papers if available. We find that GTT performs remarkably well even in a zero-shot scenario, where it faces a disadvantage as other methods have the opportunity to train on the benchmark datasets. Furthermore, we find that after fine-tuning on the train split of benchmark datasets, the performance of GTT can be further significantly improved. It achieves the best MAE on 6 datasets, best MSE on 4 datasets. These results clearly demonstrate the superiority of GTT as a foundation model for multivariate time series forecasting.

3.3 Scaling Study

We first study how the parameter scale impacts the zero-shot multivariate forecast performance of GTT models. Figure 2a gives the forecasting performance in terms of MAE on the benchmark datasets for GTT-Tiny, GTT-Small and GTT-Large. It can be observed that when the training data size does not bottleneck, the zero-shot forecasting accuracy increases with a larger model.

We then study how crucial is the training dataset size. Specifically, we also pre-train GTT-Large models on smaller datasets of size: 50M and 100M training samples. Figure 2b gives the zero-shot forecasting performance in terms of MAE on the benchmark datasets for GTT-Large pretrained on 50M, 100M and 200M samples respectively. It can be seen that when the model size does not bottleneck, the zero-shot forecasting accuracy also increases with a larger training dataset.

Importantly, the above results indicates that GTT does not appear to reach saturation within the range of model parameter and dataset sizes explored. This motivates future scaling efforts for our proposed method.

4 Related Work and Conclusion

The exploration of foundation models pretrained on large datasets for zero-shot time series forecasting remains relatively limited in comparison to the advancements made in NLP and CV fields. However, there have been some notable efforts recently. Among the examples are TimeGPT [9], Lag-Llama [21], TimesFM [4], Timer [18], UniTS [8], MOMENT [11], Moirai [24] and Chronos [1]. GTT distinguishes itself from existing models through several notable differences. First, we do not introduce any time-series-specific inductive biases into the GTT architecture. Instead, we interpret an arbitrary multivariate time series as a special category of images characterized by varying channel dimensions. Second, we adopt an encoder-only architecture for GTT with the fewest possible modifications to the standard Transformer. The only major modification we introduced is an extra channel attention stage in the multi-head self-attention block.

Our simple yet scalable approach yields impressive results, particularly when combined with pretraining on large datasets. GTT demonstrates comparable or superior performance to many advanced supervised models in multivariate time series forecasting across various widely used benchmark datasets. To conclude, GTT establishes a straightforward basis for developing large-scale foundation models for multivariate time series forecasting. Our future work includes incorporating uncertainty calibration and further scaling of GTT to yield enhanced performance.

References

[1] 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. 2024. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815* (2024).

[2] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. 2018. Dilated convolutional neural networks for time series forecasting. *Journal of Computational Finance, Forthcoming* (2018).

[3] George EP Box and Gwilym M Jenkins. 1968. Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 17, 2 (1968), 91–109.

[4] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. 2023. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688* (2023).

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

[6] James Durbin and Siem Jan Koopman. 2012. *Time series analysis by state space methods*. Vol. 38. OUP Oxford.

[7] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. TSMixer: Lightweight MLP-Mixer Model for Multivariate Time Series Forecasting. *arXiv preprint arXiv:2306.09364* (2023).

[8] Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. 2024. UniTS: Building a Unified Time Series Model. *arXiv preprint arXiv:2403.00131* (2024).

[9] Azul Garza and Max Mergenthaler-Canseco. 2023. TimeGPT-1. *arXiv preprint arXiv:2310.03589* (2023).

[10] Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I Webb, Rob J Hyndman, and Pablo Montero-Manso. 2021. Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643* (2021).

[11] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. 2024. MOMENT: A Family of Open Time-series Foundation Models. In *International Conference on Machine Learning*.

[12] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* (2016).

[13] Rob Hyndman, Anne B Koehler, J Keith Ord, and Ralph D Snyder. 2008. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media.

[14] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. 2021. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.

[15] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems* 32 (2019).

[16] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. 2021. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*.

[17] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2023. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625* (2023).

[18] Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. 2024. Timer: Transformers for Time Series Analysis at Scale. *arXiv preprint arXiv:2402.02368* (2024).

[19] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[20] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2022. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730* (2022).

[21] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Vincent Hassen, Anderson Schneider, et al. 2023. Lag-llama: Towards foundation models for time series forecasting. *arXiv preprint arXiv:2310.08278* (2023).

[22] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

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

[24] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. 2024. Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592* (2024).

[25] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2022. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186* (2022).

[26] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems* 34 (2021), 22419–22430.

[27] Taojiannan Yang, Yi Zhu, Yusheng Xie, Aston Zhang, Chen Chen, and Mu Li. 2022. AIM: Adapting Image Models for Efficient Video Action Recognition. In *The Eleventh International Conference on Learning Representations*.

[28] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting?. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 11121–11128.

[29] Yunhao Zhang and Junchi Yan. 2022. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*.

[30] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 11106–11115.

[31] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*. PMLR, 27268–27286.

[32] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. 2023. One Fits All: Power General Time Series Analysis by Pretrained LM. *arXiv preprint arXiv:2302.11939* (2023).

[33] Eric Zivot and Jiahui Wang. 2006. Vector autoregressive models for multivariate time series. *Modeling financial time series with S-PLUS®* (2006), 385–429.