DOES CROSS-DOMAIN PRE-TRAINING TRULY HELP TIME-SERIES FOUNDATION MODELS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Inspired by the success of pre-training large language models, recent efforts have explored cross-domain pre-training for time-series foundation models (TSFMs). However, the distinct data generation dynamics and contextual limitations of time-series data challenge the direct transferability of LLM strategies to TSFMs. In this paper, we investigate *whether cross-domain pre-training truly benefits TSFMs*. Through systematic experiments, we reveal that while cross-domain pre-training can enhance performance in certain domains, it may also cause severe negative transfer in others due to domain disparities in sampling frequencies and evolution patterns. Surprisingly, transfer effects are often counterintuitive: unrelated domains can yield significant gains, whereas related domains may induce degradation. These findings highlight the need for tailored pre-training strategies that address the unique characteristics of time-series data. Our study provides actionable insights to guide the development of more effective TSFMs.

1 INTRODUCTION

Time-series forecasting is a fundamental need across key domains such as energy, climate, and commerce. Inspired by the success of pre-training large language models (LLMs) on web-scale corpora (Brown et al., 2020; Kaplan et al., 2020), recent years have seen growing interest in pre-training time-series foundation models (TSFMs) using cross-domain data (Woo et al., 2024; Ansari et al., 2024; Rasul et al., 2023; Liu et al., 2024a; Das et al., 2023).

However, despite the shared sequential nature of time-series and language data, fundamental differences between these two types of data challenge the effectiveness of cross-domain pre-training for TSFMs. The first key difference lies in the underlying data generation dynamics. Language data, even across different languages and generations, reflects how humans describe the world and exchange information. In contrast, time-series data from different domains follow fundamentally distinct evolution patterns. For example, electricity consumption is driven by social and economic activities (Fan et al., 2022), climate variations are governed by advection mechanics (Verma et al., 2024), and product sales reflect shifting consumer preferences (Fan et al., 2017). These differences imply that effective forecasting requires domain-specific modeling, raising concerns about the validity of cross-domain pre-training. For instance, how would learning from product sales improve forecasting for humidity?

Even if a sufficiently large TSFM could harmonize these diverse dynamics, a second major challenge arises: historical observations in time series often lack the necessary contextual information to fully govern future variations. In language modeling, previous tokens typically constrain text generation within a limited manifold of plausible continuations—more context usually leads to more deterministic outcomes. In contrast, while past time-series data provides a foundation for forecasting, many real-world systems depend on external factors that historical time series alone cannot capture, such as policy changes, product innovations, or climate shifts. Cross-domain pre-training may further exacerbate this issue, as models trained on similar historical patterns will struggle when future dynamics diverge significantly across domains (Bergmeir, 2024).

In this study, we take a deeper look at a fundamental research question for TSFMs: ***Does cross-domain pre-training truly help?*** To answer this, we propose a simple yet effective experimental protocol to systematically evaluate whether—and to what extent—time-series datasets from different domains enhance or hinder forecasting performance in other domains. Inspired by prior research on task-transfer effects in multi-task learning (Standley et al., 2020; Fifty et al., 2021; Song et al., 2022), our approach aims to disentangle the benefits and limitations of cross-domain pre-training. We specifically examine both performance gains and negative transfer and provide insights to guide the development of future TSFMs from a data-centric perspective.

Through preliminary experiments, we provide the following findings and insights:

- While cross-domain pre-training can improve time-series forecasting performance in certain domains, it may also lead to severe and unexpected performance degradation in other scenarios, highlighting the impact of domain disparities in forecasting patterns.
- Cross-domain transfer effects appear to be highly data-driven, often counterintuitive and beyond human-perceived prior knowledge. For example, some seemingly unrelated domains exhibit significant performance gains from cross-domain learning, while closely related domains may suffer from severe negative transfer effects.
- These findings suggest that to develop more effective TSFMs, we need distinct pre-training strategies compared to LLMs, considering the challenges posed by domain disparities and insufficient contextual information in time-series data.

2 Experimental Protocols for Cross-Domain Pre-Training

Here, we refine our research question more specifically as: *Given a target application domain of interest, what kind of pre-training data is most suitable for building effective TSFMs?* Ideally, we aim to determine the optimal pre-training dataset combination (at the dataset level) that yields the best generalization performance on the given target domain.

However, this task is highly challenging due to the complexity of interactions between datasets. To address this, we simplify our study in two key ways. First, we perform a domain-level simplification by restricting the search to the domain level instead of considering arbitrary combinations of datasets. Second, we focus on a pairwise setting by examining only pairwise interactions—i.e., for a given target domain, we investigate the effect of adding one auxiliary domain at a time. This approach allows us to better isolate and understand the impact of cross-domain interactions.

2.1 Problem Formulation

Identifying Domain Combinations for Pre-Training. Let $\{D^i : i = 1, \ldots, N\}$ denote a collection of domains. For a specified target domain D^T, our goal is to identify the most appropriate pre-training combination D that helps produce a TSFM parameterized by $\theta^*(D)$ with optimal generalization performance on D^T_{test}. This can be formulated as:

$$D^* = \arg\min_{D} \ \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D^T_{\text{test}}} \left[\ell\Big(f\big(\mathbf{x}; \theta^*_D\big), \mathbf{y} \Big) \right], \text{where } \theta^*_D = \arg\min_{\theta} \ \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D} \left[\ell\Big(f\big(\mathbf{x}; \theta\big), \mathbf{y} \Big) \right].$$

Here, $f(\mathbf{x}; \theta)$ denotes the forecasting model with parameters θ, and $\ell(\cdot, \cdot)$ is the loss function. The notation D and D^T_{test} indicate the pre-training data combination and the test set of the target domain, respectively. $\mathbf{x} \in \mathbb{R}^l$ denotes the historical context vector of length l, and $\mathbf{y} \in \mathbb{R}^h$ represents the target forecast vector of length h.

Pairwise Simplification. Due to the high complexity of searching over all possible dataset combinations, we focus on a pairwise setting. In this case, for a given target domain D^T, we study the effect of incorporating a single auxiliary domain D^j. Specifically, the training set is defined as: $D^{T,j} = D^T \cup D^j$, where we omit the "train" subscript for simplicity. The optimal auxiliary domain is then identified by solving:

$$j^* = \arg\min_{j} \ \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D^T_{\text{test}}} \left[\ell\Big(f\big(\mathbf{x}; \theta^*_{D^{T,j}}\big), \mathbf{y} \Big) \right],$$

where $\theta^*_{D^{T,j}} = \arg\min_{\theta} \ \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D^{T,j}} \left[\ell\Big(f(\mathbf{x}; \theta), \mathbf{y} \Big) \right]$. This formulation allows us to quantify the benefit of cross-domain pre-training using an auxiliary domain D^j on the performance of the target domain D^T.

2.2 Protocols and Baselines

Based on the above formulations, our experiments are conducted under three distinct settings: (1) **single-domain pre-training**, where each domain D^i is trained independently to establish in-domain baseline performance; (2) **all-domain pre-training**, wherein a model is pre-trained on the combined dataset of all domains to assess the overall effect of large-scale cross-domain learning; and (3) **multi-domain (pairwise) pre-training**, which focuses on pairwise combinations $D^{i,j}$ to analyze how incorporating an auxiliary domain improves performance on the target domain.

3 Experimental Setup

Data. The experiments are conducted on a diverse set of time-series datasets spanning 10 domains, grouped into five sectors, from the LOTSA dataset (Woo et al., 2024). The **Energy** sector includes `Buildings900K` (`B900K`) (Emami et al., 2023), `BuildingsBench` (`BBench`) (Emami et al., 2023), and `ProEnFo` (Wang et al., 2023). The **Transportation** sector consists of `LargeST` (Liu et al., 2023) and `LibCity` (Jiang et al., 2023). The **Climate** sector includes `ERA5` (Nguyen et al., 2024), `CMIP6` (Nguyen et al., 2024), and `Subseasonal` (`Sub`) (Mouatadid et al., 2024). The **Cloud Service** sector contains a single domain, `CloudOps` (Woo et al., 2023). Finally, the **Sales** sector, which also contains only one domain named `Sales`, includes datasets such as the M5 competition dataset (Makridakis et al., 2022), Favorita Sales, Favorita Transactions Restaurant, and Hierarchical Sales datasets (Binkhonain & Zhao, 2023).

For more detailed information and analysis of these domains, see Appendix B. The following experiments are all conducted at the domain level. In subsequent sections, the terms "dataset" and "domain" may be used interchangeably, but they both refer to the different domains discussed here.

Model. We adopt the MOIRAI-small (Woo et al., 2024) architecture as the backbone model for all experiments. To ensure consistency across experiments, we set the patch size to 16, following the same approach as Liu et al. (2024b) and Yao et al. (2025). All other settings remain consistent with the original paper. Moirai-small is one of the first foundation models designed specifically for time series forecasting, containing $10.7M$ trainable parameters. We acknowledge that larger models with more parameters might exhibit different scaling behaviors. However, due to resource constraints, we have not yet conducted experiments with larger models, leaving this for future research.

Evaluation. Each domain is split into training and test sets to ensure a fair and reasonable data distribution for in-domain performance evaluation. The test set is selected by holding out the last portion of each dataset (details in Appendix C) during the pre-training phase, ensuring these samples are not used for training. The evaluation metrics include four key measures to comprehensively assess the in-domain performance (NLL-loss, NMAE, NRMSE, SMAPE; their definitions can be found in the Appendix C.1).

4 Results and Analysis

Table 1 presents the averaged NLL-loss of all pre-training experiments following our protocols. We use NLL-loss as the main evaluation metric here because it directly assesses the predicted probability distribution, capturing both central tendency and uncertainty. We also report other sampled point prediction metrics, such as NMAE, NRMSE, and SMAPE, which are included in Appendix C.1. These metrics exhibit similar patterns to NLL-loss, though the degree of degradation is less severe in some domains.

All-domain pre-training does not always outperform single-domain pre-training. The *"all"* column reports the relative performance when using all-domain pre-training (the default approach for TSFMs) compared to single-domain pre-training. This result reveals clear sector-specific trends: in the `Energy` and `Climate` sectors, all-domain pre-training consistently outperforms single-domain pre-training, with the largest improvement observed in the energy sector's proenfo dataset, where performance increased by 16.22%. Conversely, for the `Transportation`, `Cloud Services` and `Sales` sector, all-domain pre-training performs worse than single-domain pre-training. Notably, for sales data, all-domain pre-training results in the most significant degradation,

Table 1: **NLL-loss of Multi-Domain Pre-training *Relative* to Single-Domain Pre-training.** This table compares the test Negative Log-Likelihood loss across various domains when models are pretrained on single-domain data versus multi-domain data. Each row represents a target domain used for test, with the second column ("single") showing the single-domain pre-training NLL-loss. The third column ("all") displays the relative performance (in percentage) of all-domain pre-training compared to single-domain pre-training. The subsequent columns present the relative performances (in percentage) when combining the target domain i with an auxiliary domain j during pre-training. For example, in the `B900k` target domain, a value of -6.56% for `BBench` indicates that incorporating the auxiliary domain `BBench` during multi-domain pre-training leads to a 6.56% improvement compared to single-domain pre-training. Diagonal elements are omitted ("-") as they align with single-domain pre-training and remain near 0.00%. Results are averaged over five trials.

NLL-loss↓	single	all	B900k	BBench	ProEnFo	LargeST	LibCity	ERA5	CMIP6	Sub	CloudOps	Sales
`B900k`	3.66	-5.71%	–	-6.56%	-3.56%	-0.74%	-1.64%	-1.29%	-0.53%	1.53%	-2.23%	1.21%
`BBench`	3.46	-0.27%	-1.17%	–	-0.01%	-0.76%	-0.60%	-0.30%	-0.18%	-0.73%	-0.92%	1.12%
`ProEnFo`	8.30	-16.22%	-13.52%	-13.66%	–	-9.21%	-15.49%	-3.68%	-3.63%	-12.53%	-16.80%	-11.78%
`LargeST`	4.51	7.15%	3.20%	4.36%	1.13%	–	6.59%	0.98%	0.96%	5.20%	5.92%	9.13%
`LibCity`	2.34	3.57%	0.58%	0.51%	0.20%	-0.63%	–	-0.12%	-0.19%	1.09%	2.52%	3.40%
`ERA5`	2.42	-12.37%	-19.13%	-16.54%	-17.67%	-5.97%	-6.01%	–	-11.71%	5.64%	-23.00%	7.31%
`CMIP6`	2.62	-16.57%	-19.77%	-15.06%	-8.92%	-16.22%	-11.62%	-10.79%	–	4.47%	-30.56%	72.03%
`Sub`	3.46	-9.28%	-0.49%	-2.66%	-1.16%	-0.23%	-6.24%	-0.52%	0.97%	–	-8.52%	-9.35%
`CloudOps`	0.43	23.85%	4.92%	6.51%	-1.81%	-0.54%	-0.96%	-0.70%	3.40%	9.08%	–	30.62%
`Sales`	0.67	67.09%	-3.14%	1.26%	3.22%	4.04%	43.73%	3.76%	-2.21%	5.87%	45.59%	–

with performance dropping by 67.09%. Overall, this illustrates that while all-domain pre-training can deliver substantial benefits for certain datasets, it can also significantly hinder performance in others, emphasizing the need for tailored pre-training approaches.

Cross-domain transfer effects appear to be highly data-driven, and they are not symmetrical. Cross-domain transfer effects exhibit either mutual enhancement or conflict, and we hypothesize that their behavior is influenced by the underlying characteristics of the datasets. For example, some domains, like `CloudOps`, significantly enhance performance in seeming unrelated domains like `CMIP6`, with an impressive 30.56% improvement over single-domain pre-training — a result that stands out given differences in sampling frequency and sector. Conversely, certain domains consistently impair each other's performance. For example, `Sales` and `Subseasonal` datasets, characterized by their daily sampling frequency, often cause negative transfer when paired with datasets sampled at finer temporal resolutions (e.g., minute or hourly data). Moreover, these observed transfer effects are not symmetric; `CloudOps` improves `CMIP6`, but the reverse effect is not observed. This asymmetry underlines the complexity of cross-domain transfer effects in time-series forecasting, highlighting the need for deeper understanding of these dynamics.

Data from the same sector does not always enhance each other. Even within the same sector, cross-dataset transfer effects vary significantly. Among the three sectors studied, `Energy` stands out with consistent positive transfer across datasets, likely due to all datasets in this sector being uniformly sampled at an hourly rate. In contrast, the `Transportation` and `Climate` sector frequently exhibits mixed results, where combining datasets may lead to degraded performance, possibly due to heterogeneity in sampling rates or forecasting patterns. These observations suggest that even within the same sector, pre-training strategies need to carefully account for variations in data properties such as sampling frequency and pattern similarity.

5 Conclusion

In this study, we systematically evaluated the impact of cross-domain pretraining for TSFMs and uncovered key insights. Our findings show that cross-domain transfer effects are highly data-driven and sometimes counterintuitive, with unrelated domains occasionally providing significant gains while closely related ones may cause degradation. These results emphasize the need for tailored pretraining strategies that account for the unique characteristics of time-series data, rather than directly adopting approaches from language models. Future work should explore adaptive methods to mitigate negative transfer and better leverage cross-domain knowledge.

References

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Gift-eval: A benchmark for general time series forecasting model evaluation. *arXiv preprint arXiv:2410.10393*, 2024.

Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Bernie Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=gerNCVqqtR`. Expert Certification.

Christoph Bergmeir. Llms and foundational models: Not (yet) as good as hoped. *Foresight: The International Journal of Applied Forecasting*, 2024.

Manal Binkhonain and Liping Zhao. A machine learning approach for hierarchical classification of software requirements. *Machine Learning with Applications*, 12:100457, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

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

Patrick Emami, Abhijeet Sahu, and Peter Graf. Buildingsbench: A large-scale dataset of 900k buildings and benchmark for short-term load forecasting. *Advances in Neural Information Processing Systems*, 36:19823–19857, 2023.

Wei Fan, Shun Zheng, Xiaohan Yi, Wei Cao, Yanjie Fu, Jiang Bian, and Tie-Yan Liu. DEPTS: Deep Expansion Learning for Periodic Time Series Forecasting. In *ICLR*, 2022.

Zhi-Ping Fan, Yu-Jie Che, and Zhen-Yu Chen. Product sales forecasting using online reviews and historical sales data: A method combining the bass model and sentiment analysis. *Journal of business research*, 2017.

Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. In *NeurIPS*, 2021.

Federico Garza, Kin Gutierrez, Cristian Challu, Jose Moralez, Ricardo Olivares, and Max Mergenthaler. tsfeatures: Time series feature extraction in python, 2024. URL `https://github.com/Nixtla/tsfeatures`. Accessed: 2024-09-24.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don‘t stop pretraining: Adapt language models to domains and tasks. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740. URL `https://aclanthology.org/2020.acl-main.740/`.

John Haslett and Adrian E. Raftery. Space-time modelling with long-memory dependence: Assessing ireland's wind power resource. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 38(1):1–50, 1989. doi: 10.2307/2347679. URL `https://doi.org/10.2307/2347679`.

Huan He, Owen Queen, Teddy Koker, Consuelo Cuevas, Theodoros Tsiligkaridis, and Marinka Zitnik. Domain adaptation for time series under feature and label shifts. In *International Conference on Machine Learning*, pp. 12746–12774. PMLR, 2023.

Jiawei Jiang, Chengkai Han, Wenjun Jiang, Wayne Xin Zhao, and Jingyuan Wang. Towards efficient and comprehensive urban spatial-temporal prediction: A unified library and performance benchmark. *arXiv e-prints*, pp. arXiv–2304, 2023.

Xiaoyong Jin, Youngsuk Park, Danielle Maddix, Hao Wang, and Yuyang Wang. Domain adaptation for time series forecasting via attention sharing. In *International Conference on Machine Learning*, pp. 10280–10297. PMLR, 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. Largest: A benchmark dataset for large-scale traffic forecasting. *Advances in Neural Information Processing Systems*, 36:75354–75371, 2023.

Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. Unitime: A language-empowered unified model for cross-domain time series forecasting. In *Proceedings of the ACM on Web Conference 2024*, pp. 4095–4106, 2024a.

Xu Liu, Juncheng Liu, Gerald Woo, Taha Aksu, Yuxuan Liang, Roger Zimmermann, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Moirai-moe: Empowering time series foundation models with sparse mixture of experts. *arXiv preprint arXiv:2410.10469*, 2024b.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m5 competition: Background, organization, and implementation. *International Journal of Forecasting*, 38(4):1325–1336, 2022.

Soukayna Mouatadid, Paulo Orenstein, Genevieve Flaspohler, Miruna Oprescu, Judah Cohen, Franklyn Wang, Sean Knight, Maria Geogdzhayeva, Sam Levang, Ernest Fraenkel, et al. Subseasonalclimateusa: a dataset for subseasonal forecasting and benchmarking. *Advances in Neural Information Processing Systems*, 36, 2024.

Tung Nguyen, Jason Jewik, Hritik Bansal, Prakhar Sharma, and Aditya Grover. Climatelearn: Benchmarking machine learning for weather and climate modeling. *Advances in Neural Information Processing Systems*, 36, 2024.

Mohamed Ragab, Emadeldeen Eldele, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, and Xiaoli Li. Self-supervised autoregressive domain adaptation for time series data. *IEEE Transactions on Neural Networks and Learning Systems*, 35(1):1341–1351, 2022.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Hassen, Anderson Schneider, et al. Lag-llama: Towards foundation models for time series forecasting. In *R0-FoMo: Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 4933–4941, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL `https://aclanthology.org/2020.lrec-1.607/`.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=e1wDDFmlVu`.

Xiaozhuang Song, Shun Zheng, Wei Cao, James Yu, and Jiang Bian. Efficient and effective multi-task grouping via meta learning on task combinations. In *NeurIPS*, 2022.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *ICML*, 2020.

Yogesh Verma, Markus Heinonen, and Vikas Garg. ClimODE: Climate and weather forecasting with physics-informed neural ODEs. In *ICLR*, 2024.

Zhixian Wang, Qingsong Wen, Chaoli Zhang, Liang Sun, Leandro Von Krannichfeldt, and Yi Wang. Benchmarks and custom package for electrical load forecasting. *arXiv preprint arXiv:2307.07191*, 2023.

Garrett Wilson, Janardhan Rao Doppa, and Diane J Cook. Multi-source deep domain adaptation with weak supervision for time-series sensor data. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1768–1778, 2020.

Gerald Woo, Chenghao Liu, Akshat Kumar, and Doyen Sahoo. Pushing the limits of pre-training for time series forecasting in the cloudops domain. *arXiv preprint arXiv:2310.05063*, 2023.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024.

Qingren Yao, Chao-Han Huck Yang, Renhe Jiang, Yuxuan Liang, Ming Jin, and Shirui Pan. Towards neural scaling laws for time series foundation models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uCqxDfLYrB.

APPENDIX

TABLE OF CONTENTS

A RELATED WORK

A.1 TIME SERIES FOUNDATION MODELS

The development of TSFMs has gained momentum (Woo et al., 2024; Ansari et al., 2024; Rasul et al., 2023; Liu et al., 2024a; Das et al., 2023). These models aim to generalize across diverse time-series datasets, enabling zero-shot and few-shot forecasting capabilities. Early approaches, such as Chronos (Ansari et al., 2024) and Lag-LLaMA (Rasul et al., 2023), employed unified architectures that struggled with the heterogeneity of input patterns, leading to increased learning complexity and parameter demands. Recent methods like UniTime (Liu et al., 2024a) and MOIRAI (Woo et al., 2024) have addressed these challenges by incorporating specialization mechanisms, such as frequency embeddings or dataset-level prompts, to better adapt to specific data characteristics.

Scaling TSFMs has also been a key focus. For example, Time-MoE (Shi et al., 2025) and Moirai-MoE (Liu et al., 2024b) leverage mixture-of-experts (MoE) architectures to increase model capacity while maintaining computational efficiency. These studies demonstrate that scaling laws—originally established for language models—are applicable to time-series forecasting (Yao et al., 2025). However, significant challenges remain in addressing the heterogeneity of time-series data across domains, such as variations in sampling frequencies and dominant patterns.

A.2 DOMAIN ADAPTATION

Domain adaptation (DA) for time series addresses the challenges posed by distribution shifts between source and target domains, requiring methods that handle unique temporal dependencies and dynamic sequence patterns (Wilson et al., 2020; Jin et al., 2022). Unlike traditional DA approaches in CV and NLP (Rietzler et al., 2020; Gururangan et al., 2020), time series DA must account for these complexities. Recent advancements include He et al. (2023), which aligns temporal and frequency features to address feature and label shifts, and Ragab et al. (2022), which uses self-supervised learning with forecasting as an auxiliary task to improve feature transferability.

However, most existing DA methods remain task-specific and are often evaluated on small-scale datasets with limited domain pairs, restricting their generalizability to diverse scenarios. Our work aims to explore the cross-domain transferability of TSFMs on a larger data scale. Instead of adopting a specific DA method, we follow the common approach used by most current TSFMs, where tokenization uniformly represents the source and target domains in a shared space.

B DETAILED ANALYSIS OF DIFFERENT DOMAINS

B.1 STATISTICAL ANALYSIS

Although the original LOTSA dataset[1] provides a general categorization of different domains, some domains contain an overwhelming amount of data, some domains contain an overwhelming amount of data or are mixed with data that does not belong to specific domains. To address this, we refined the selection process and chose 10 representative domains for our experiments.

The datasets of the ten selected domains cover a wide range of sampling frequencies (e.g., seconds for transportation data vs. days for sales data). Table 2 summarizes the sampling frequencies covered by each domain. Overall, the domains tend to align with their respective sectors' characteristics,

[1] `https://github.com/SalesforceAIResearch/uni2ts`

showing a clustering pattern based on their sampling frequency needs. For instance, the *transportation* sector (e.g., `LibCity`) commonly include high-frequency data (e.g., 2T, 5T, 15T, 30T, where T represents seconds), reflecting the need to capture rapid changes in urban mobility. By contrast, *weather and climate* sector (e.g., `ERA5, CMIP6, Subseasonal`) are typically sampled at lower frequencies (`hourly, 6-hourly, or daily`) in line with the slower progression of atmospheric and environmental changes. Similarly, the *sales* sector operate on a daily frequency, as sales data evolves relatively slowly compared to other domains. Through the multi-domain pre-training between these domains with distinct frequencies, further insights can be drawn concerning the transfer potential across domains with similar or complementary temporal properties.

Table 3 demonstrates that each domain contains a substantial volume of data, ensuring sufficient resources for training a Time Series Foundation Model (TSFM). For example, large datasets such as `CMIP6` and `ERA5` provide over 25 billion target points, while smaller datasets like `ProEnFo` (2.21M) still maintain adequate data for effective model training. The sequence lengths also vary significantly across domains, with some datasets having consistent lengths (`Buildings900K`, `ERA5`, `CMIP6`) and others showing wide variability (`BuildingsBench`, `LibCity`, `Sales`).

Despite these differences, Table 3 highlights that the diverse range of data points and sequence lengths reflects the natural characteristics of each domain. This variability leads to practical constraints, and we did not enforce equal data volumes across domains due to the workload involved. Importantly, this diversity allows for robust evaluation of TSFM models across varied temporal and structural scenarios.

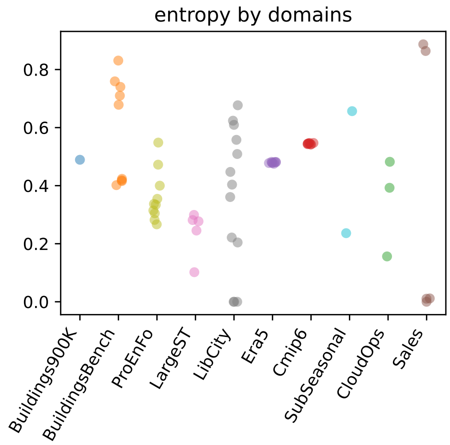
Table 2: **Sampling Frequencies Covered by Each Domain.** Sampling frequencies: 2T, 5T, 15T, 30T (seconds), H (hours, including subcategories such as H, 6H), and D (days). A checkmark (✓) indicates that the dataset contains data at the corresponding sampling frequency.

Domain	2T	5T	15T	30T	H	6H	D
`Buildings900K`					✓		
`BuildingsBench`					✓		
`ProEnFo`					✓		
`LargeST`		✓					
`LibCity`	✓	✓	✓	✓			
`ERA5`					✓		
`CMIP6`						✓	
`Subseasonal`							✓
`CloudOps`		✓					
`Sales`							✓

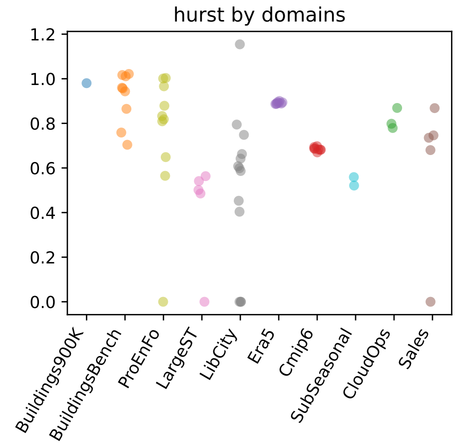
Table 3: **Summary of Datasets Across Different Domains.** The table includes the following metrics: **Target points (M)** (total number of data points in millions), **AvgLen** (average target length as an integer), **MinLen** (minimum target length), and **MaxLen** (maximum target length).

Domain	Target points (M)	AvgLen	MinLen	MaxLen
`Buildings900K`	15,728.24	8761	8761	8761
`BuildingsBench`	20.47	14196	193	34,223
`CloudOps`	2151.01	4304	97	8064
`CMIP6`	25,355.88	7300	7300	7300
`ERA5`	25,763.51	8736	8736	8736
`LargeST`	4452.51	105178	105120	105408
`LibCity`	388.75	19753	1572	105120
`ProEnFo`	2.21	25870	17520	39414
`Sales`	198.09	1441	47	1913
`SubSeasonal`	66.55	15715	11323	16470

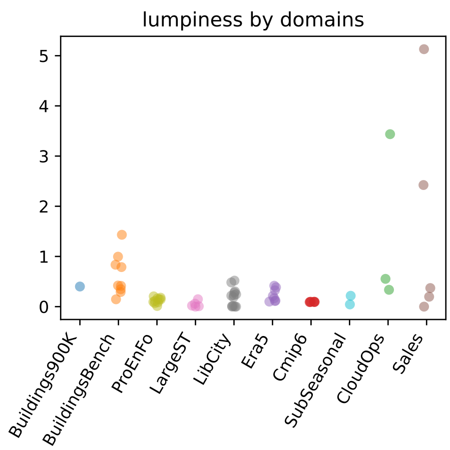
B.2 Calculation of Time Series Features



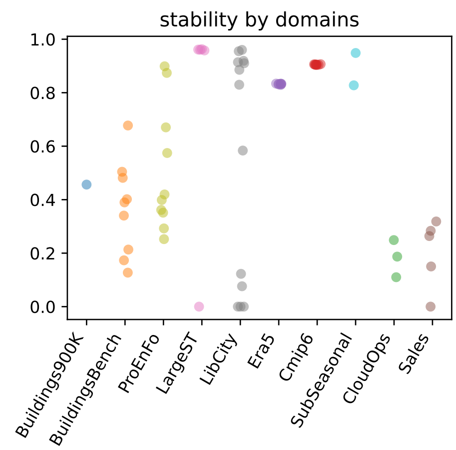
(a) Entropy



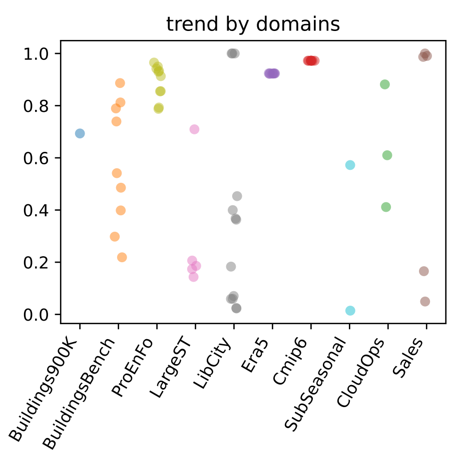
(b) Hurst



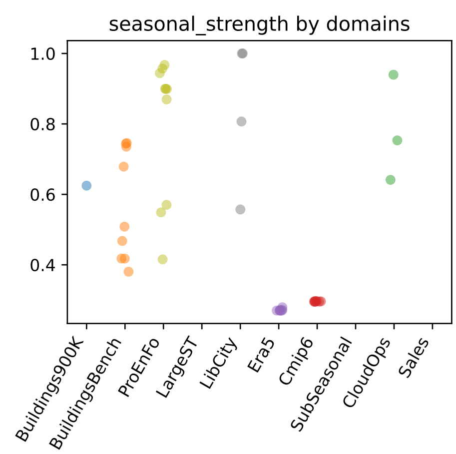
(c) Lumpiness



(d) Stability



(e) Trend



(f) Seasonal

Figure 1: Time Series Features of Different Domains.

In this section, we analyze the distribution of different time series features across datasets from various domains, as shown in Figure 1. For this analysis, we adopt the methodology proposed in Aksu et al. (2024) and utilize the `tsfeatures` library (Garza et al., 2024) to calculate these features. The diversity of the datasets provides an opportunity to systematically study the impact of cross-domain pretraining on time series forecasting tasks.

The calculation of time series features involves two main steps: data preparation and feature extraction. For each dataset, time series are analyzed based on their values, timestamps, and frequencies. If necessary, the target sequence can be shortened to a specified proportion of its original length (e.g., 5% or 20%), preserving the most recent information. The prepared data is then processed using the `tsfeatures` library to compute statistical properties such as trend, seasonal strength, and entropy. For datasets with multiple variables, features are calculated separately for each variable, and the final dataset-level results are obtained by averaging across all variables. This process ensures that the extracted features summarize the overall characteristics of each dataset effectively.

Below, we introduce each feature and its corresponding definition.

Trend. Time series were decomposed using STL (Seasonal and Trend decomposition using Loess) into trend f_t, multiple seasonal components $s_{i,t}$ for $i = 1, \ldots, M$, and remainder e_t:

$$x_t = f_t + s_{1,t} + \cdots + s_{M,t} + e_t.$$

The strength of the trend is:

$$\text{trend} = 1 - \frac{\mathrm{Var}(e_t)}{\mathrm{Var}(f_t + e_t)}.$$

Values less than 0 are set to 0, and values greater than 1 are set to 1. Higher values indicate stronger trends.

Seasonal. Seasonal strength for each component is derived as:

$$\text{seasonal\_strength}_i = 1 - \frac{\mathrm{Var}(e_t)}{\mathrm{Var}(s_{i,t} + e_t)}.$$

Values are clipped between 0 and 1, with non-seasonal series yielding 0.

Entropy. Entropy measures the complexity of the time series using the spectral density estimate $\hat{f}(\lambda)$:

$$\text{Entropy} = -\int_{-\pi}^{\pi} \hat{f}(\lambda) \log \hat{f}(\lambda)\, d\lambda.$$

Lower entropy implies predictable patterns, while higher entropy indicates complexity.

Hurst Exponent. The `Hurst exponent` (`hurst`) is computed as:

$$\text{Hurst} = 0.5 + d,$$

where d is the maximum likelihood estimate of fractional differencing order (Haslett & Raftery, 1989). Higher values (~ 1.0) reflect smoother trends, less volatility, and less roughness.

Stability. Stability quantifies shifts in mean values across tiles. For N tiles with means $\bar{x}_i$, stability is:

$$\text{Stability} = \mathrm{Var}\left(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_N\right).$$

Lower values indicate consistency, while higher values suggest irregularities.

Lumpiness. Lumpiness measures the variability of variances across tiles. For N tiles with variances s_i^2, lumpiness is:

$$\text{Lumpiness} = \mathrm{Var}\left(s_1^2, s_2^2, \ldots, s_N^2\right).$$

Higher lumpiness indicates periods of volatility.

B.3 Feature Analysis

Figure 1 illustrates the distribution of six key time series features across various domains. The results reveal distinct patterns and variability among domains:

- **Entropy** (Figure 1(a)): Higher entropy is observed in domains such as `BuildingsBench` and `Sales`, indicating more complex and less predictable time series. In contrast, domains like `LargeST` exhibit lower entropy, suggesting simpler and more structured patterns.
- **Hurst Exponent** (Figure 1(b)): Most domains display higher Hurst values, signifying smoother trends. Conversely, a few datasets in `LibCity` and `Sales` show a value of 0, which may be due to a calculation error.
- **Lumpiness (Figure 1(c)):** High lumpiness, as seen in `CloudOps` and `Sales`, suggests significant variability in volatility across time.
- **Stability (Figure 1(d)):** `CloudOps` and `Sales` demonstrate low stability, indicating notable shifts in the mean over time. In contrast, `LargeST` and climate sector show consistently higher stability with less variation.
- **Trend (Figure 1(e)):** Strong trends are evident in `ProEnfo`, `ERA5` and `CMIP6`, indicating a clear directional component. This aligns with common sense, as weather data often changes gradually, exhibiting more trend information. Conversely, domains like `LibCity` and `LargeST` have weaker trends and are predominantly driven by other factors.
- **Seasonal Strength (Figure 1(f)):** Domains such as `CloudOps` and `LibCity` display notable seasonal strength, reflecting regular periodic patterns.

These variations highlight the diversity in time series characteristics across different domains, providing insight into the challenges and opportunities for cross-domain time series forecasting.

It is worth noting that the calculation of these time-series features involves processing large volumes of data and intricate pre-processing and data transformations, which may introduce some errors or inaccuracies. As a result, the insights provided are limited. We plan to further refine these calculations in future work to ensure accuracy and provide more comprehensive analytical perspectives.

C MORE ON EXPERIMENTAL SETUP

We conducted validation on datasets across all domains, each containing thousands of samples. The input context length was fixed at 512 time points (equivalent to 32 patches, with a patch size of 16), while the prediction length varied between 14 and 720 across different tasks.

The MOIRAI-small model (Woo et al., 2024) was trained for 10^5 steps using a batch size of 256. The AdamW optimizer was employed with the following hyperparameters: a learning rate (lr) of 1×10^{-3}, a weight decay of 1×10^{-1}, $\beta_1 = 0.9$, and $\beta_2 = 0.98$. A learning rate scheduler was utilized, incorporating a linear warmup over the initial 10,000 steps, followed by cosine annealing. The models were trained using NVIDIA V100-32G GPUs with TF32 precision.

Given that `Buildings900K` is a synthetic dataset specifically designed to enhance `BuildingsBench`, we aligned its test set with that of `BuildingsBench`. Both test sets consist of the final time series segments from datasets included in `BuildingsBench`.

For each experiment, test evaluation was performed across all domains. However, we presented results based on the primary domain of interest. For example, in experiments involving two-domain combinations (e.g., $i + j$), the results for domain i were evaluated using the test set of D_i, and those for domain j were evaluated using the test set of D_j.

Following the pre-training stage, the foundation model could potentially undergo fine-tuning on the target dataset to improve performance on downstream tasks. However, as this paper focuses on studying in-domain transfer capabilities, we did not perform fine-tuning, leaving it as an avenue for future research.

C.1 METRICS

We utilize four metrics to evaluate the performance of the model: **NLL-loss**, **NMAE**, **NRMSE**, and **SMAPE**. Below are their definitions and formulas:

- **Negative Log-Likelihood Loss (NLL-loss)**:

The Negative Log-Likelihood Loss measures the likelihood of the ground truth under the predicted probability distribution. For a Gaussian distribution with mean $\hat{y}$ and variance $\hat{\sigma}^2$, it is defined as:

$$\text{NLL-loss} = \frac{1}{N} \sum_{i=1}^{N} \left[\frac{(y_i - \hat{y}_i)^2}{2\hat{\sigma}_i^2} + \frac{1}{2} \log(2\pi\hat{\sigma}_i^2) \right], \tag{1}$$

where:

- y_i: ground truth value of the i-th sample.
- $\hat{y}_i$: predicted mean value of the i-th sample.
- $\hat{\sigma}_i^2$: predicted variance for the i-th sample.
- N: total number of samples.

This metric penalizes both inaccurate predictions (mean error) and poor uncertainty estimation (variance error).

- **Normalized Mean Absolute Error (NMAE)**:
  NMAE measures the average absolute error between predictions and ground truth, normalized by the sum of the absolute ground truth values:

$$\text{NMAE} = \frac{\frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|}{\sum_{i=1}^{N} |y_i|}. \tag{2}$$

- **Normalized Root Mean Squared Error (NRMSE)**:
  NRMSE is defined as the square root of the mean squared error, normalized by a denominator computed as the squared sum of the absolute target values. Based on the provided implementation logic, the formula can be expressed as:

$$\text{NRMSE} = \sqrt{\frac{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\left(\sum_{i=1}^{N} |y_i|\right)^2}}. \tag{3}$$

  Here, the denominator is computed as the square of the sum of the absolute target values.

- **Symmetric Mean Absolute Percentage Error (SMAPE)**:
  SMAPE is a percentage-based metric that measures the average relative error between predictions and ground truth. It is symmetric with respect to over-predictions and under-predictions:

$$\text{SMAPE} = \frac{100\%}{N} \sum_{i=1}^{N} \frac{|y_i - \hat{y}_i|}{\frac{|y_i| + |\hat{y}_i|}{2}}. \tag{4}$$

  SMAPE is bounded between 0% and 200%, making it scale-independent and suitable for comparing datasets with different ranges.

D Complete Experimental Results

In Section 4, we presented the results corresponding to NLL-loss. Here, we also display the results for the other three metrics: NMAE is shown in Table 4, NRMSE is shown in Table 5, and SMAPE is shown in Table 6. By comparison, it can be observed that the conclusions of the four metrics are generally similar. However, the degradation of NLL-loss is more severe, which may be related to the fact that the other three metrics are point-level and are calculated by taking the median after sampling.

Figures 2 to 11 present the raw data with error bars for each domain, which further validates the conclusions discussed in the paper.

Table 4: **NMAE of Multi-Domain *relative* to Single-Domain Pretraining.** The "single" column shows NMAE for single-domain pretraining on test set of the corresponding domain. The "all" column presents results using all-domain data. The remaining columns report multi-domain pretraining, combining the current domain with another. For example, in the first row (b900k), -1.15% for bbench indicates a 1.15% improvement over single-domain pretraining. Results are averaged over five trials.

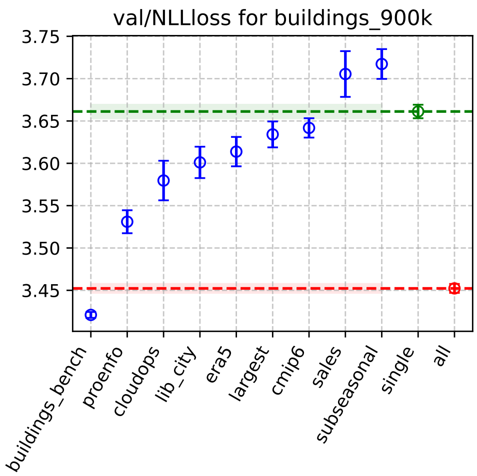
NMAE↓	single	all	b900k	bbench	proenfo	largest	city	era5	cmip6	sub	cloudops	sales
b900k	0.18	-1.55%	–	-1.15%	-0.12%	1.12%	-1.69%	-1.40%	-0.28%	1.43%	-2.93%	2.05%
bbench	0.18	-1.98%	-1.58%	–	0.65%	-0.14%	1.25%	-0.17%	0.11%	-0.80%	-1.80%	5.05%
proenfo	0.28	-16.95%	-4.86%	-11.49%	–	-2.83%	-10.89%	-1.16%	0.98%	-4.67%	-15.68%	2.67%
largest	0.20	-17.80%	-22.06%	-20.46%	-4.92%	–	-15.83%	-22.27%	-22.00%	-20.24%	-21.20%	-13.87%
city	0.14	3.66%	-0.00%	-0.18%	-0.03%	-0.55%	–	-0.08%	-0.69%	0.34%	2.03%	3.48%
era5	1.34	-68.44%	-71.45%	-73.56%	-72.26%	-73.04%	-70.62%	–	-57.50%	-36.22%	-72.85%	-74.11%
cmip6	1.15	-59.07%	-39.03%	-53.07%	-48.93%	-50.76%	-41.28%	27.09%	–	24.23%	-61.12%	-53.83%
sub	0.39	-4.04%	-1.19%	-2.36%	-0.81%	-0.51%	-4.10%	-0.02%	0.06%	–	-3.66%	-5.53%
cloudops	0.10	6.56%	0.25%	-0.12%	0.01%	1.55%	-0.57%	0.89%	1.30%	2.19%	–	8.07%
sales	0.61	-0.44%	-0.69%	-0.74%	0.04%	0.23%	-0.27%	0.02%	-0.01%	0.02%	0.18%	–

Table 5: **NRMSE of Multi-Domain *relative* to Single-Domain Pretraining.** The "single" column shows NRMSE for single-domain pretraining on test set of the corresponding domain. The "all" column presents results using all-domain data. The remaining columns report multi-domain pretraining, combining the current domain with another. For example, in the first row (b900k), 0.64% for bbench indicates a 0.64% degradation over single-domain pretraining. Results are averaged over five trials.

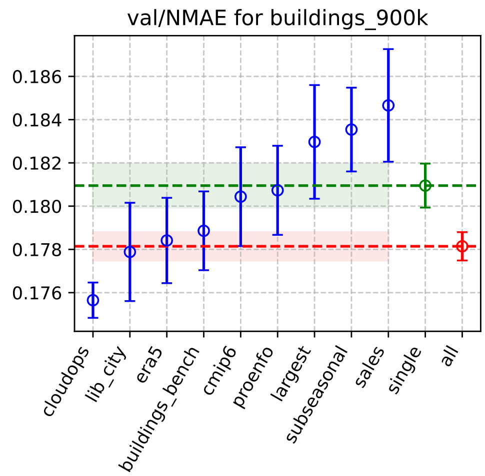
NRMSE↓	single	all	b900k	bbench	proenfo	largest	city	era5	cmip6	sub	cloudops	sales
b900k	0.26	-0.49%	–	0.64%	0.48%	1.49%	-0.94%	-1.11%	-0.18%	0.90%	-2.05%	1.70%
bbench	0.26	-2.40%	-1.30%	–	0.59%	-0.16%	0.84%	-0.34%	0.22%	-0.98%	-2.39%	1.78%
proenfo	0.33	-17.01%	-6.22%	-11.50%	–	-2.92%	-12.14%	-1.89%	0.56%	-6.16%	-16.57%	-0.22%
largest	0.35	-31.86%	-36.04%	-34.21%	-12.09%	–	-29.61%	-36.28%	-36.16%	-33.89%	-34.94%	-31.32%
city	0.23	3.13%	-0.07%	-0.27%	-0.01%	-0.38%	–	-0.26%	-0.92%	0.08%	1.62%	2.78%
era5	1.68	-71.22%	-73.97%	-75.89%	-74.51%	-75.28%	-73.08%	–	-60.20%	-41.16%	-75.27%	-76.60%
cmip6	1.44	-59.59%	-41.74%	-54.80%	-50.62%	-53.14%	-43.01%	25.39%	–	23.04%	-61.05%	-55.83%
sub	0.53	-3.96%	-0.76%	-1.95%	-0.63%	-0.50%	-3.08%	0.04%	0.24%	–	-3.85%	-5.16%
cloudops	0.14	4.71%	-0.08%	-0.48%	0.10%	1.04%	-0.25%	0.64%	1.13%	1.58%	–	5.64%
sales	0.99	-0.48%	-0.58%	-0.67%	0.03%	0.17%	-0.26%	-0.05%	-0.04%	0.03%	0.12%	–

Table 6: **SMAPE of Multi-Domain *relative* to Single-Domain Pretraining.** The "single" column shows SMAPE for single-domain pretraining on test set of the corresponding domain. The "all" column presents results using all-domain data. The remaining columns report multi-domain pretraining, combining the current domain with another. For example, in the first row (b900k), -2.49% for bbench indicates a -2.49% improvement over single-domain pretraining. Results are averaged over five trials.

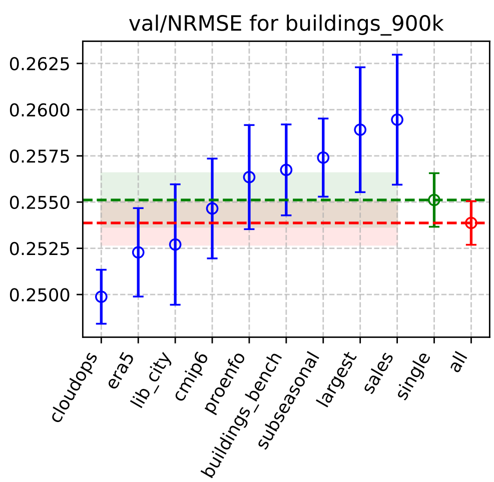
SMAPE↓	single	all	b900k	bbench	proenfo	largest	city	era5	cmip6	sub	cloudops	sales
b900k	20.08	-0.16%	–	-2.49%	-0.67%	-0.67%	-0.77%	-1.04%	0.07%	0.53%	-0.55%	0.76%
bbench	19.89	0.78%	-1.58%	–	0.12%	-1.16%	1.36%	-0.47%	-0.26%	-0.76%	0.83%	5.41%
proenfo	27.86	-20.72%	-6.66%	-14.04%	–	-3.19%	-13.93%	-1.44%	1.67%	-6.83%	-19.75%	-1.54%
largest	17.97	6.53%	0.47%	2.65%	-0.65%	–	7.53%	0.44%	0.66%	2.88%	2.15%	17.28%
city	16.12	3.12%	-0.07%	-0.20%	-0.27%	-0.61%	–	-0.31%	-0.72%	0.03%	2.00%	2.82%
era5	59.68	-14.38%	-11.29%	-15.60%	-13.97%	-14.84%	-11.62%	–	-6.29%	0.64%	-16.38%	-16.42%
cmip6	65.90	-27.39%	-9.18%	-19.06%	-18.58%	-16.10%	-12.65%	-4.82%	–	-3.40%	-29.79%	-17.35%
sub	61.97	-2.80%	-0.54%	-2.13%	-0.78%	-0.60%	-3.01%	0.22%	0.25%	–	-2.82%	-5.05%
cloudops	14.72	3.68%	0.16%	-0.18%	-0.04%	0.97%	-0.43%	0.62%	0.74%	1.38%	–	5.56%
sales	79.64	-0.59%	-0.35%	-0.57%	-0.02%	0.45%	-0.02%	-0.08%	-0.05%	0.04%	-0.04%	–



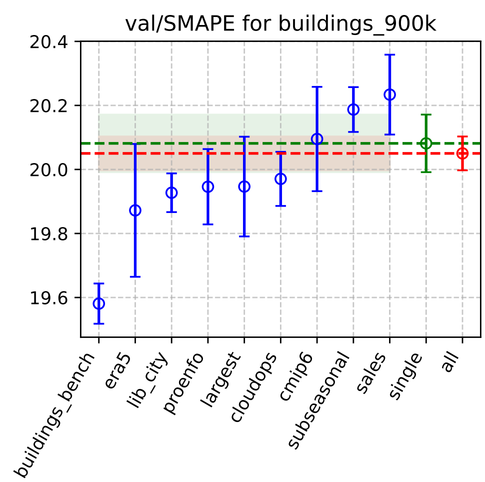
(a) NLL



(b) NMAE

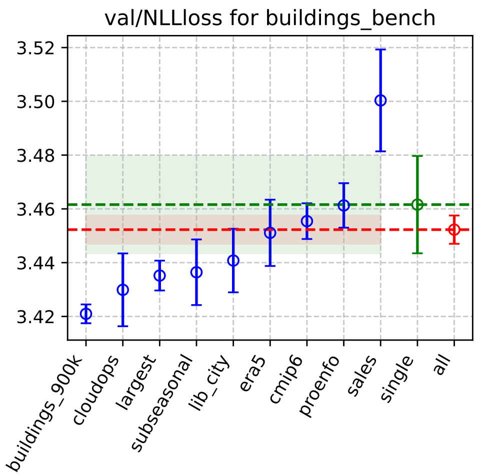


(c) NRMSE

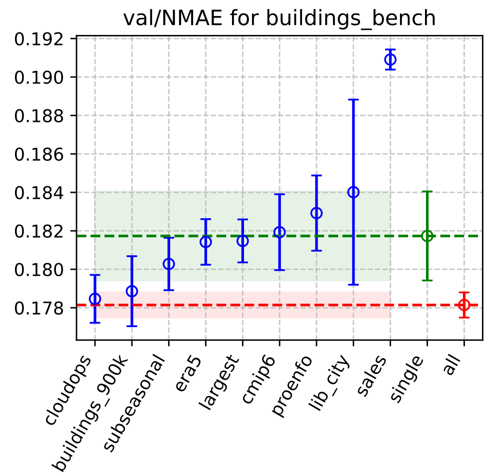


(d) SMAPE

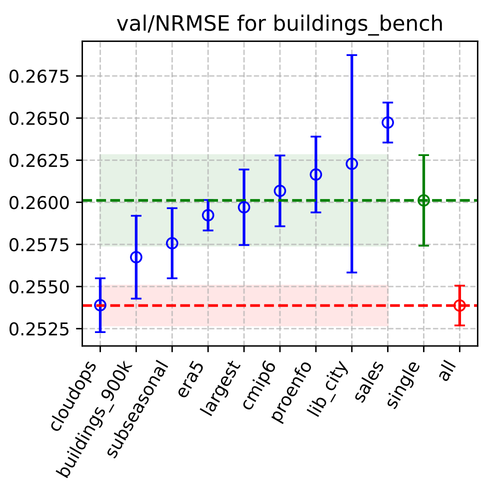
Figure 2: **Metrics on `Buildings900k` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining ("single"), all-domain pretraining ("all"), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



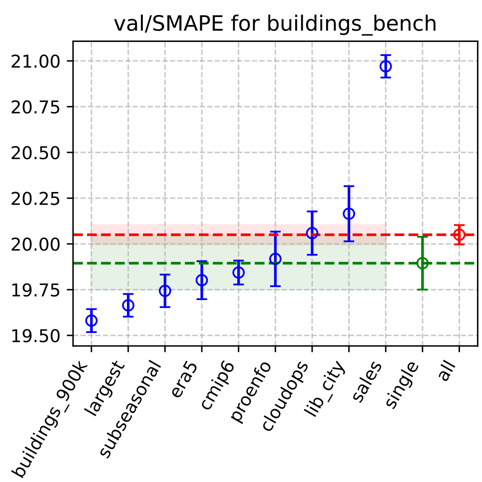
(a) NLL



(b) NMAE

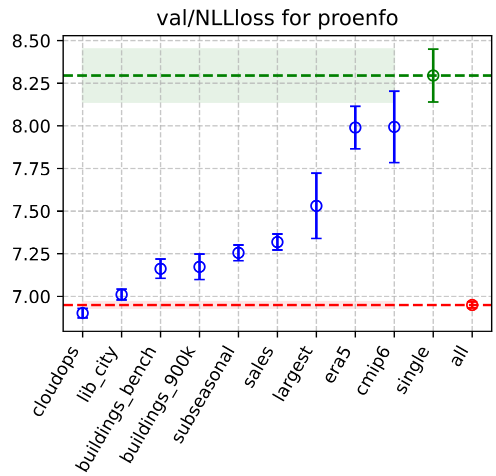


(c) NRMSE

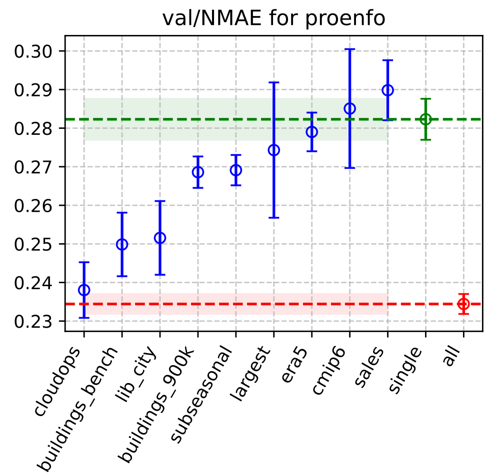


(d) SMAPE

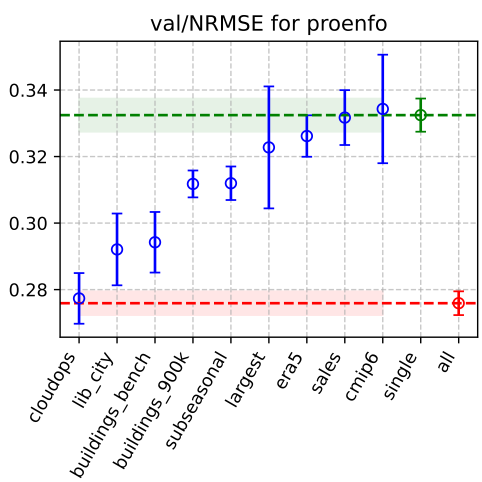
Figure 3: **Metrics on `BuildingsBench` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining (”single”), all-domain pretraining (”all”), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



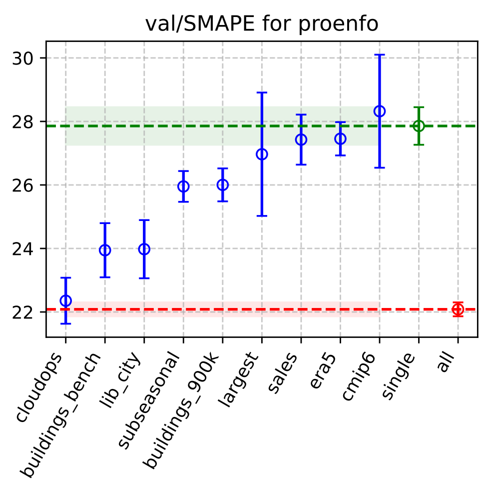
(a) NLL



(b) NMAE

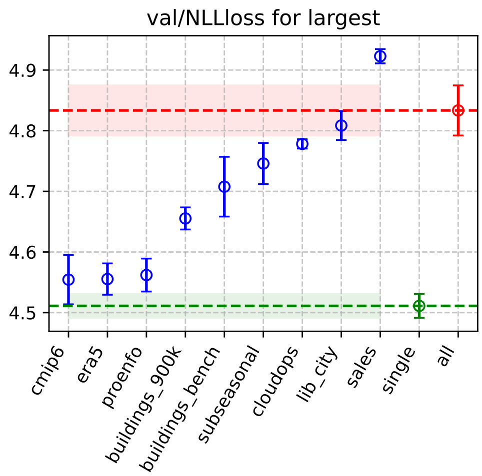


(c) NRMSE

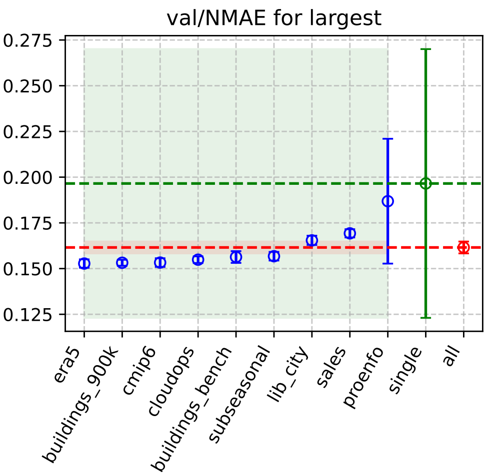


(d) SMAPE

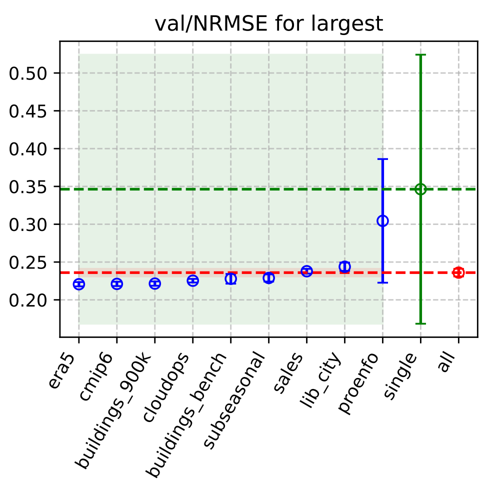
Figure 4: **Metrics on `ProEnFo` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining ("single"), all-domain pretraining ("all"), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



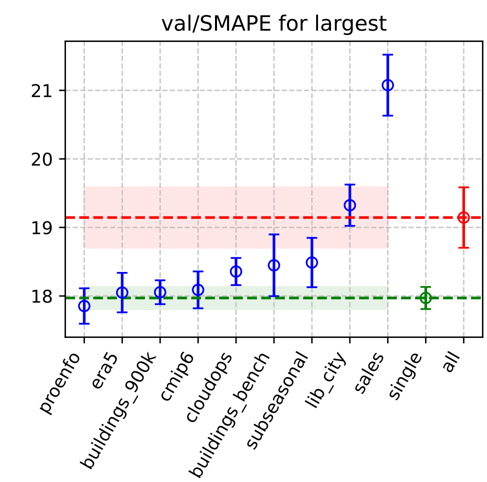
(a) NLL



(b) NMAE

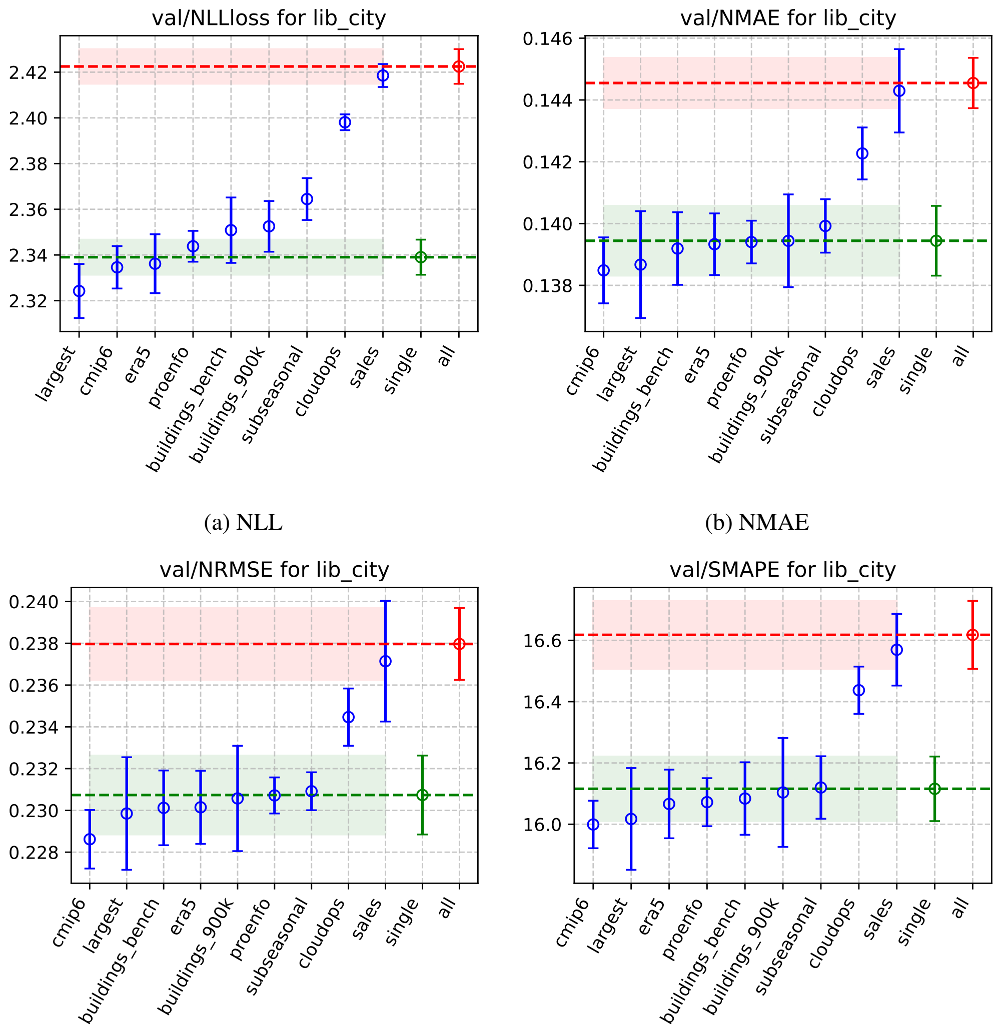


(c) NRMSE



(d) SMAPE

Figure 5: **Metrics on `LargeST` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining (”single”), all-domain pretraining (”all”), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



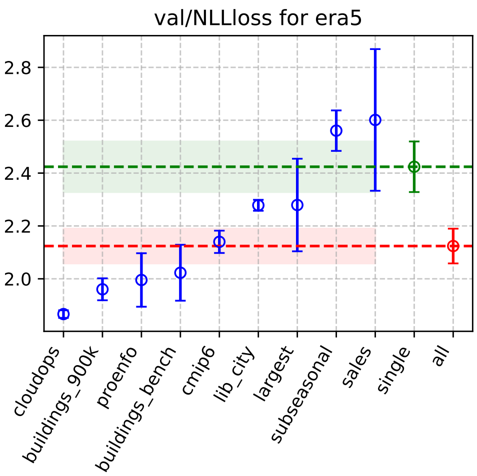
(a) NLL

(b) NMAE

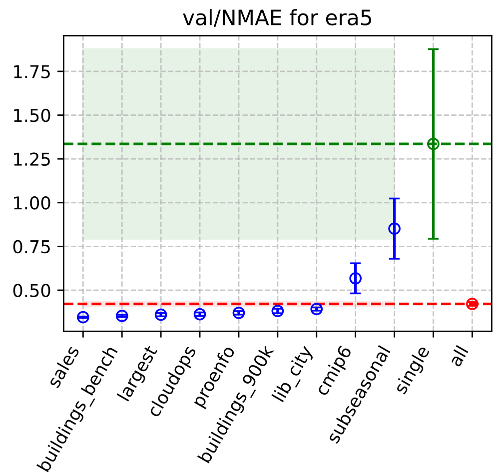
(c) NRMSE

(d) SMAPE

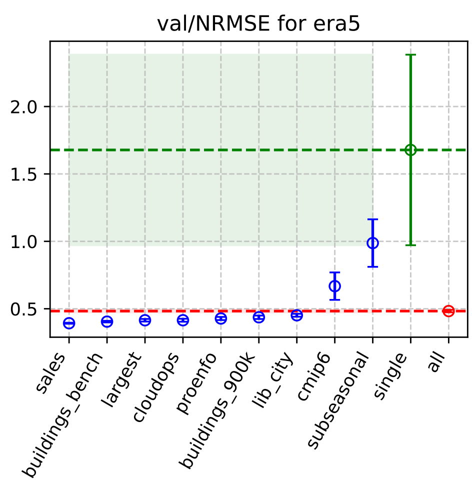
Figure 6: **Metrics on `LibCity` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining (”single”), all-domain pretraining (”all”), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



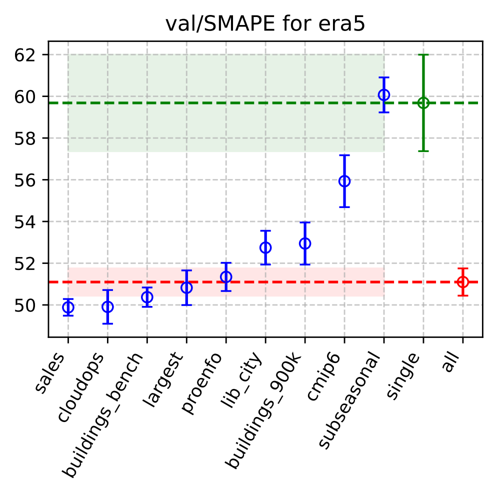
(a) NLL



(b) NMAE

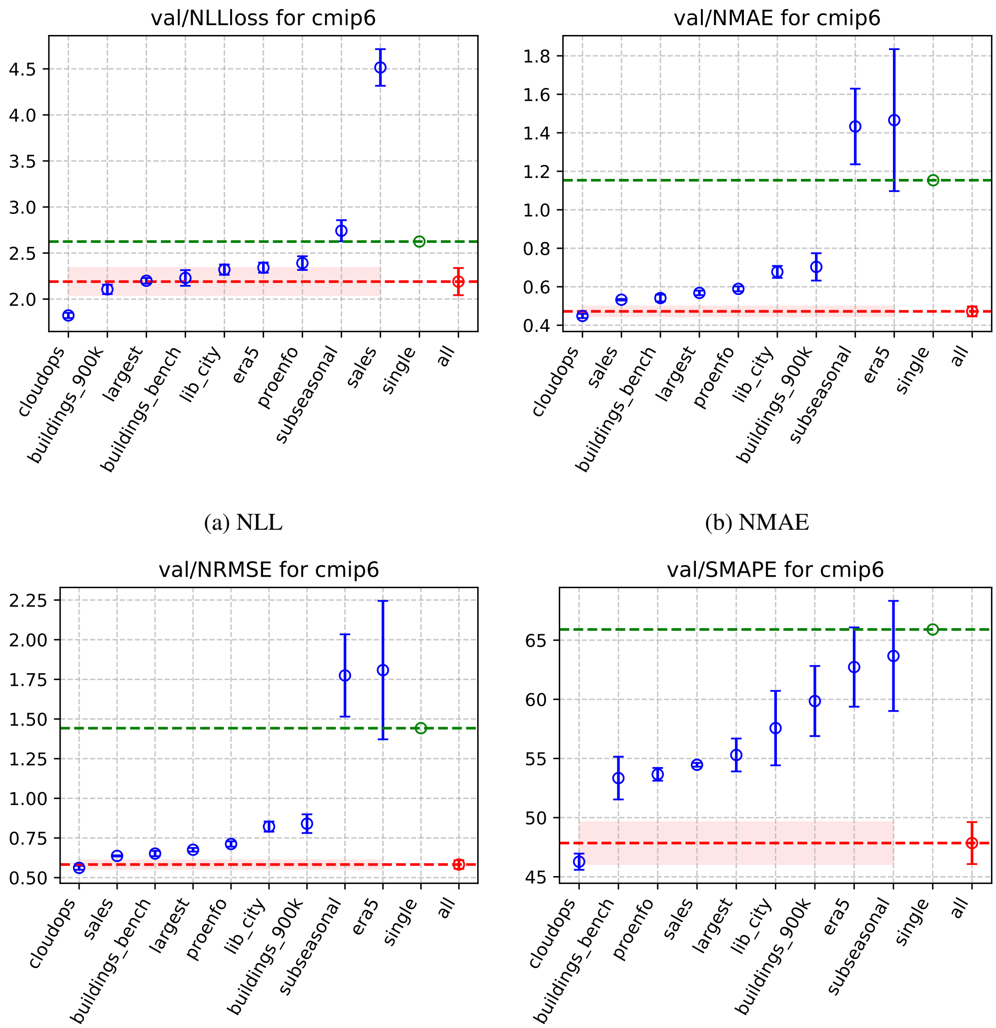


(c) NRMSE



(d) SMAPE

Figure 7: **Metrics on `ERA5` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining (”single”), all-domain pretraining (”all”), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



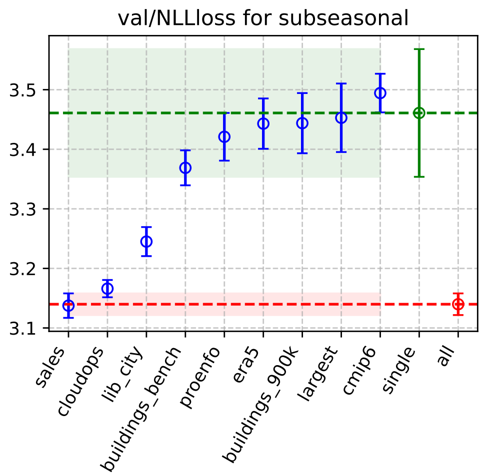
(a) NLL

(b) NMAE

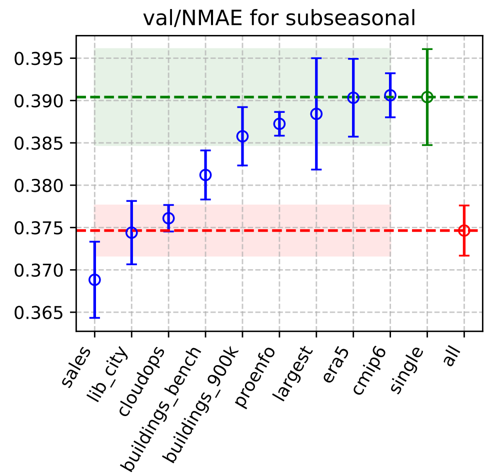
(c) NRMSE

(d) SMAPE

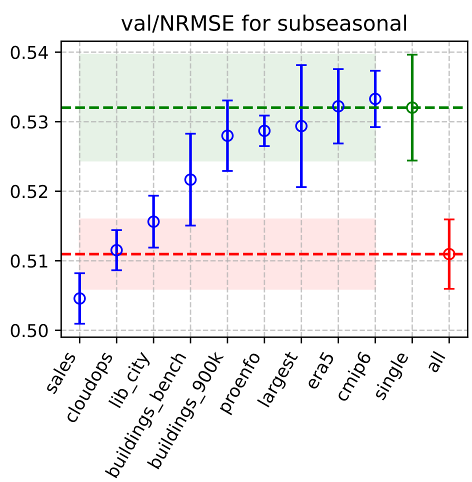
Figure 8: **Metrics on `CMIP6` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining ("single"), all-domain pretraining ("all"), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



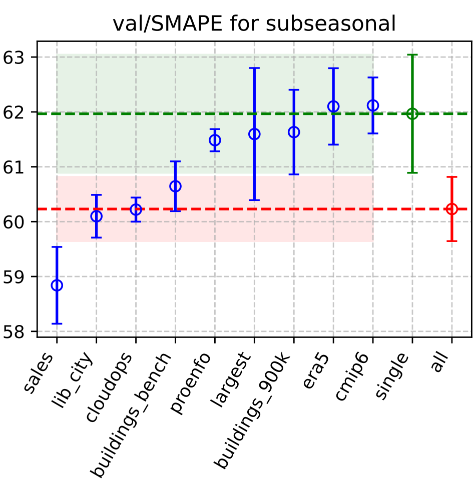
(a) NLL



(b) NMAE

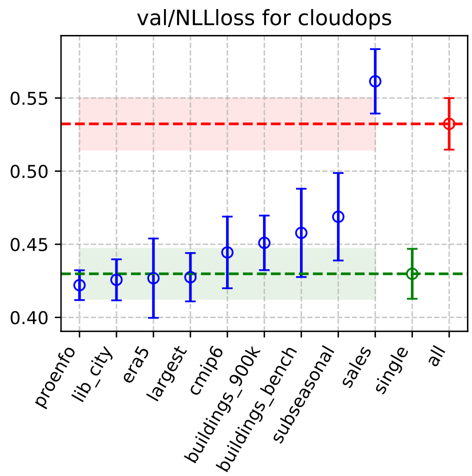


(c) NRMSE

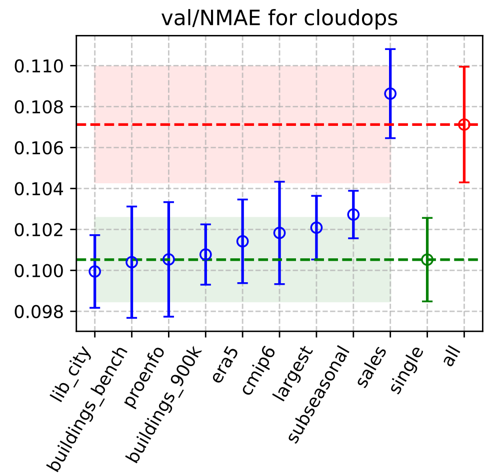


(d) SMAPE

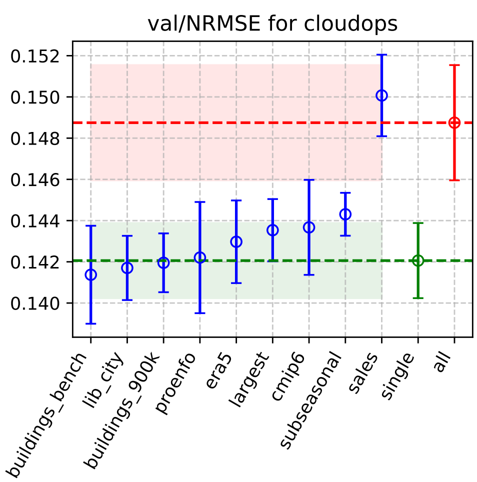
Figure 9: **Metrics on `Subseasonal` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining ("single"), all-domain pretraining ("all"), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



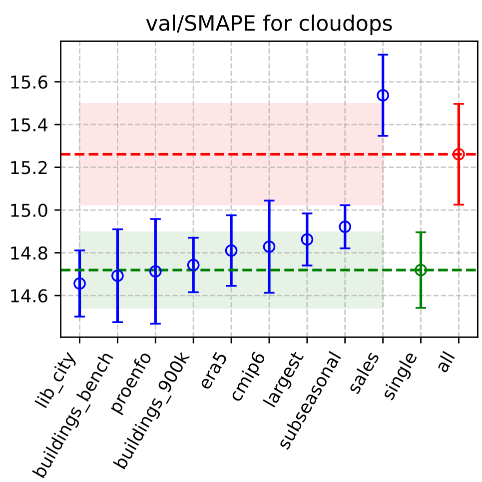
(a) NLL



(b) NMAE

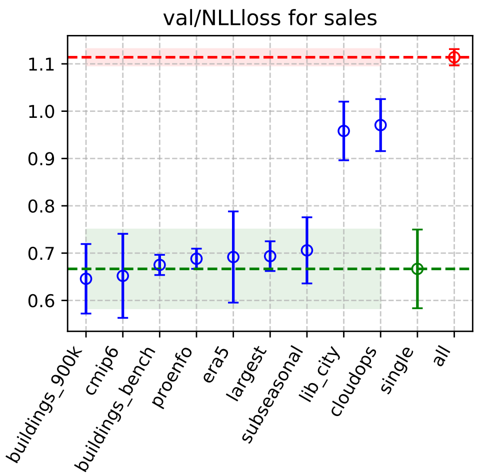


(c) NRMSE

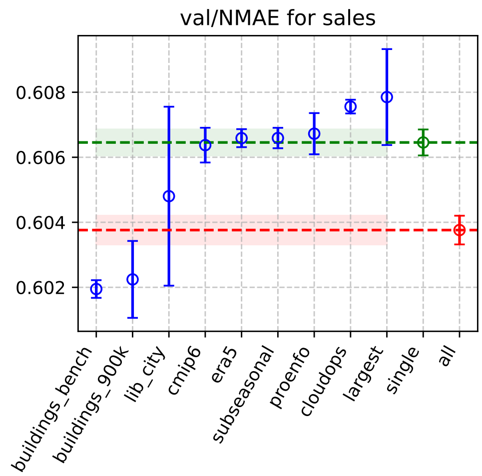


(d) SMAPE

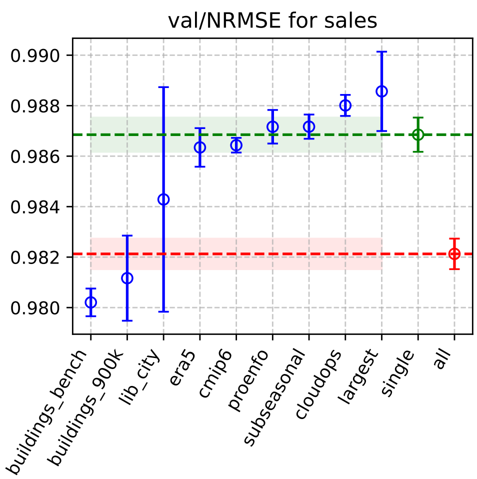
Figure 10: **Metrics on `CloudOps` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining ("single"), all-domain pretraining ("all"), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.



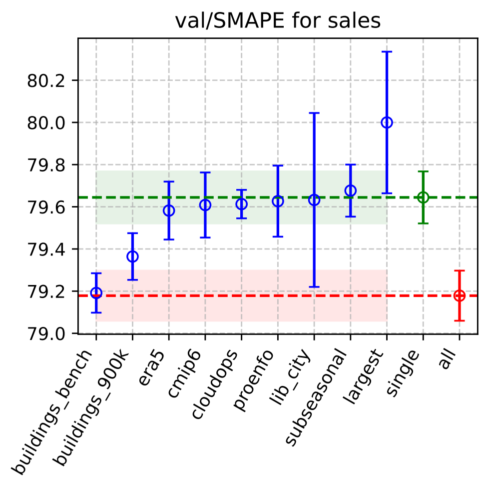
(a) NLL



(b) NMAE



(c) NRMSE



(d) SMAPE

Figure 11: **Metrics on `Sales` Across Different Pretraining Domains.** This figure evaluates test performance across models pretrained on the target domain and various auxiliary domains, using four metrics: (a) NLL-loss, (b) NMAE, (c) NRMSE, and (d) SMAPE. The x-axis in each subplot including: single-domain pretraining (”single”), all-domain pretraining (”all”), and multi-domain pretraining (any other auxiliary domain, sort from smallest to largest). The y-axis indicates the corresponding metric values, with error bars showing standard deviations across 5 random trials. Red dashed lines highlight the performance of single-domain pretraining, while green dashed lines indicate the performance under all-domain pretraining.