Scalable Temporal Domain Generalization via Prompting

Sepidehsadat Hosseini [1] **Mengyao Zhai** [1] **Hossein Hajimirsadeghi** [1] **Frederick Tung** [1]

Abstract

Machine learning models typically assume that training and testing data follow the same independent and identically distributed (i.i.d.) distribution. However, in real-world deployment, data often evolves over time. Addressing this challenge requires models that can efficiently adapt at test time without retraining. This paper introduces a prompting-based test-time adaptation framework for temporal domain generalization that enables pre-trained models to efficiently adapt to evolving distributions without re-training. Our method is both parameter- and time-efficient, leveraging global prompts, domain-specific prompts, and drift-aware prompts to model and forecast temporal shifts in data distributions. By extrapolating these learned adaptations, our approach enables pre-trained models being adaptive to dynamic environments. We demonstrate the adaptability, scalability and generality of our framework across classification, regression, time-series forecasting, and NLP tasks, highlighting its effectiveness in adapting foundation models to real-world temporal shifts.

1. Introduction

Most machine learning algorithms rely on the assumption that the training and testing data are independently and identically distributed (i.i.d.). However, in many in-the-wild deployment settings, the testing data can fall outside the training distribution. This discrepancy is particularly pronounced in the deployment of pre-trained foundation models, where distribution shifts and concept drifts over time. This results in significant performance degradation at test time, highlighting the urgent need for models that can efficiently adapt to dynamic, non-stationary environments without retraining.

[1]RBC Borealis, Vancouver, Canada. Correspondence to: Mengyao Zhai <mengyao.zhai@borealisai.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

While prior research in domain generalization (DG) has explored strategies for handling distribution shifts by improving generalization to unseen domains (Yue et al., 2019; Prakash et al., 2019; Shankar et al., 2018; Volpi et al., 2018; Hu et al., 2023; Triantafillou et al., 2021; Kim et al., 2021; Wang et al., 2021), these approaches often overlook the critical setting of test-time adaptation (TTA) - they often do not account for temporal evolution in the data, limiting their ability to handle dynamic, real-world settings where domain shifts are driven by time-dependent factors. More discussion on related work is in Appendix A.1.

In this paper, we introduce a novel algorithm for *temporal domain generalization* (Bai et al., 2023; Nasery et al., 2021) that enables pre-trained models to generalize and adapt to evolving data distributions at test-time. Our approach is both parameter- and time-efficient, leveraging prompt-based adaptation to dynamically condition a pre-trained model for future domain shifts. Specifically, we propose a mechanism that learns three types of prompts: (1) Global prompts, capturing general knowledge shared across domains; (2) Domain-specific prompts, adapting the model to particular domain characteristics; and (3) Drift-aware prompts, which model temporal shifts in domain-specific prompts and extrapolate future adaptations. By forecasting the temporal evolution of these prompts, our method equips foundation models with the capability to anticipate and adapt to distributional shifts. Moreover, our approach optimizes adaptation through lightweight, real-valued prompts, making it highly scalable and adaptive for real-world applications. The key benefits of our prompting-based temporal DG include:

- **Scalability and efficiency.** Only a small number of parameters shared across all domains are needed for prompt generation and no retraining on future-domain data is required, ensuring efficient adaptation.
- **Adaptability.** By explicitly modeling temporal drift, our method enhances the reliability of foundation models in dynamic environments, making them robust against real-world distributional changes at test time.
- **Generality.** The approach seamlessly integrates with various architectures and tasks, including classification, regression, time-series forecasting, and natural language processing (NLP), as demonstrated in our experiments.

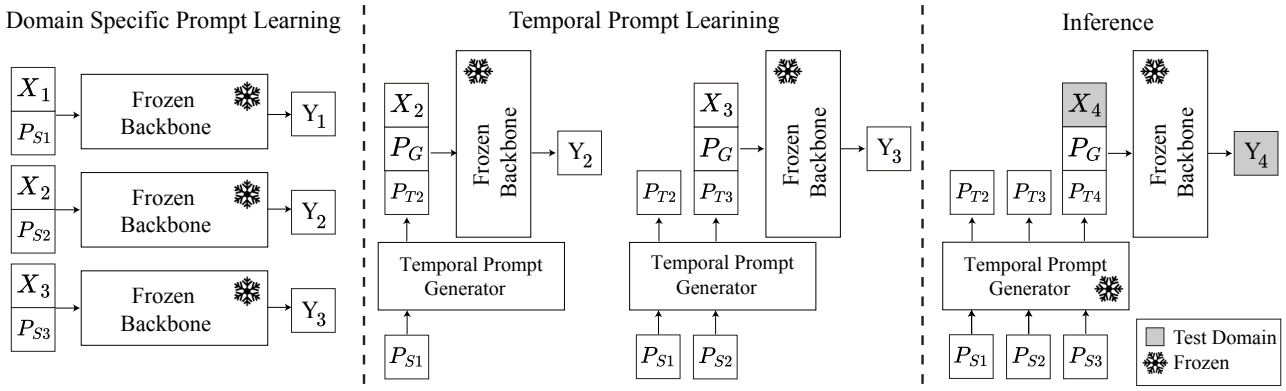


Figure 1: Illustration of the proposed method with a set of source domains D_1, D_2, and D_3 given during training and a target (unobserved future) domain D_4 used only during testing. The goal is to adapt the frozen backbone network, which has been pre-trained on the combined source domains. First, domain-specific prompts P_{S1}, P_{S2}, P_{S3} are learned independently on each source domain to learn the characteristics of each indexed domain separately. Next, a temporal prompt generator is trained to transform the domain-specific prompts to temporal prompts (P_{T2}, P_{T3}, P_{T4}) which can capture the temporal dynamics and concept drifts within the sequence of domains. Finally, to capture the general knowledge across all domains, the general prompts P_G are learned. During inference, the combination $[P_{T4}; P_G, X_4]$ is fed to the frozen backbone to perform the task on the target domain D_4.

2. Method

We address the problem of adapting a pre-trained model to future time periods under the realistic setting where the data distribution evolves over time. Prompt tuning is a well-known mechanism for adapting a pre-trained model to downstream tasks efficiently (see Appendix A.1). Prompt tuning comprises two main components: prompts and a frozen backbone transformer network. The prompts are prepended to the inputs, guiding the frozen backbone transformer in adapting to different downstream tasks. In this section, we present our novel prompting-based method that efficiently makes pre-trained models adaptable to unseen future domains.

2.1. Frozen Backbone Network

The model to be adapted could be a pre-trained LLM for NLP tasks, or a backbone network pre-trained on the data combining all source domains for general tasks.

Denote a set of temporal domains by $\{D_t = (X_t, Y_t)\}$, where $\{D_t | 1 \leq t \leq \tau\}$ represents the source domains, $\{D_t | t > \tau\}$ represents the target (unseen future) domains, and X_t and Y_t are the inputs and outputs for domain t, respectively. f_θ is the transformer-based backbone model.

2.2. Domain-Specific Prompt Learning

We first learn prompts to capture domain-specific information. For each domain D_t, we prepend the input X_t with a prompt $P_{S(t)} \in \mathbb{R}^n$, which are learnable parameters. The combined result, represented as $[P_{S(t)}; X_t]$, is then processed by the frozen backbone network f_θ. To learn prompt $P_{S(t)}$, the model is trained to maximize the likelihood $\mathcal{P}_\theta(Y_t | [P_{S(t)}; X_t])$ while freezing the pre-trained model parameters θ. Learning on each domain independently, we derive domain-specific prompts $P_{S1}, P_{S2}, ..., P_{S(\tau)}$, effectively condensing domain knowledge into a concise set of parameters.

2.3. Temporal Prompt Learning

To capture concept drift over time, we employ a temporal prompt generator g_ω to encode the temporal dynamics into temporal prompts. Starting from $t = 2$, for each domain D_t the temporal prompt generator g_ω receives domain-specific prompts, $P_{S1}, P_{S2}, \ldots, P_{S(t-1)}$, as input tokens. It then uses those prompts to generate the temporal prompts $P_{T2}, P_{T3}, \ldots, P_{T(t)}$. Specifically, as shown in Equation 1, it generates the temporal prompt $P_{T(t)} \in \mathbb{R}^n$ for domain D_t from previous domain-specific prompts.

$$P_{T(t)} = g_\omega(P_{S1:(t-1)}), \quad t = 2, \ldots, \tau \tag{1}$$

Moreover, to help capture generic information across all domains, we incorporate a learnable general prompt $P_G \in \mathbb{R}^n$. The input X_t is prepended by the generic prompt P_G and the temporal prompt $P_{T(t)}$. The result, represented as $[P_{T(t)}; P_G; X_t]$, is fed into the frozen backbone network f_θ. Both P_G and the temporal prompt generator g_ω are trained to maximize the likelihood $\mathcal{P}_\theta(Y_t | [P_{T(t)}; P_G; X_t])$, while keeping the backbone network f_θ frozen. Temporal prompts $P_{T2}, P_{T3}, \ldots, P_{T(\tau)}$ effectively capture temporal drift and help the pre-trained network to adapt to changes in the data distribution over time, and to anticipate future changes by capturing temporal trends.

2.4. Inference

During inference, the model utilizes the domain-specific prompts $P_{S1}, P_{S2}, \ldots, P_{S(\tau)}$ and generates temporal prompts $P_{T2}, P_{T3}, \ldots, P_{T(\tau+1)}$. To perform the target domain task, the frozen backbone receives the input $[P_{T(\tau+1)}; P_G; X_{(\tau+1)}]$ and predicts the output.

3. Experiments

We demonstrate the generality of prompting-based temporal generalization using public datasets covering classification (Rotated Moons (2-Moons) (Nasery et al., 2021), Online News Popularity (ONP) (Ben-David et al., 2010), Electrical Demand (Elec2) (Nasery et al., 2021)), regression (House prices (House) (Nasery et al., 2021), Appliances energy prediction (Appliance) (Bai et al., 2023)), time series forecasting tasks (Crypto (Arik et al., 2022)), and NLP sentiment prediction tasks (AmazPantry (Ni et al., 2019)).

Baseline Methods. We compare our model with several state-of-the-art methods, including temporal domain generalization methods DRAIN (Bai et al., 2023) and GI (Nasery et al., 2021), continuous domain adaption methods CDOT (Jimenez et al., 2019) and CIDA (Wang et al., 2020), and prompting method ATTEMPT (Asai et al., 2022), and 2 baseline methods Vanilla-MLP and Vanilla-Transformer. More details of the baseline methods can be found in Appendix A.2.

Implementation Details. We utilize the Adam optimizer (Kingma & Ba, 2014) and consistently set the learning rate to $1e-4$ across all datasets. Our system is implemented in PyTorch and runs on a workstation powered by a 2.10GHz Intel Xeon(R) Gold 6230 CPU with 20 cores, paired with an NVIDIA RTX 5000 GPU. For each dataset, we tune the hyperparameters based on the suggestions from (Bai et al., 2023). Additional experiment settings and results (e.g., network architectures and additional ablation results) are provided in Appendix A.2.

3.1. Experimental Results

Synthetic data. To evaluate the proposed method under a controlled setting, we constructed synthetic datasets derived from the Mackey-Glass equations (Mackey & Glass, 1977) and sums of cosines. Due to the page limit, these results are included in Appendix A.4.

Classification and regression tasks. Table 1 summarizes the classification and regression results. For the classification and regression datasets, we followed the procedure outlined in (Bai et al., 2023) to partition the dataset into distinct temporal domains. It is observed that our method shows superior results on almost all datasets compared with SOTA methods, except for the 2-Moons dataset. It may due to the low dimensionality of the 2-Moons dataset (only 2 dimensions).

Time series forecasting tasks. Table 2 shows the time series forecasting results on the Crypto dataset. For Crypto dataset, we used 90% of entries from each month in 2018, 2019, and 2020 for training (across 36 domains), the remaining 10% of entries for *in-domain* testing, the first month of 2021 for validation, and the subsequent three months of 2021 for actual testing.

Experimental results show that our approach enables time and parameter efficient adaptation of the pre-trained model. The state-of-the-art temporal domain generalization method DRAIN (Bai et al., 2023) is a hypernet-like approach, where all the weights of the pre-trained model are predicted for the unseen future domain. DRAIN's parameter count grows quickly, while our method requires three orders of magnitude fewer parameters for training and takes less than half the time to train, without loss in generalization performance.

NLP tasks. We demonstrate our method on an NLP dataset AmazPantry (Ni et al., 2019), and our method is applied to the pre-trained BERT (Devlin et al., 2018). Note that DRAIN (Bai et al., 2023) is not applicable to this experiment since generating BERT using a hypernetwork is not feasible. Table 3 show that our method can well capture concept drifting and generalize well to unseen future.

3.2. Ablation Studies and Additional Experiments

We conducted ablation studies on the Crypto and Elec2 datasets to see the impact of the different types of proposed prompts. Table 4 shows that both prompting mechanisms P_T and P_G contribute to better performance.

As an additional experiment, we further evaluate our proposed method with a state-of-the-art multi-scale transformer architecture, Scaleformer (Shabani et al., 2023), on Mackey-Glass synthetic data. Table 5 shows the results, where Scaleformer has been applied on Informer (Zhou et al., 2021). The results confirm that our method generalizes to the state-of-the-art transformers and can support further improvements by incorporating complementary advances in network design.

Further ablation studies on the number of training domains, sequential versus non-sequential training paradigms, and hyperparameter configurations can be found in Appendix A.3.

4. Conclusion

In this paper, we introduced a prompting-based approach to temporal domain generalization, enabling pre-trained models to efficiently adapt to evolving data distributions without requiring access to future-domain data. Leveraging global prompts, domain-specific prompts, and drift-aware prompts

Table 1: Performance comparison of all methods in terms of classification error (in %) for classification tasks and mean absolute error (MAE) for regression tasks (both smaller the better.) Results of comparison methods on all datasets are reported from (Bai et al., 2023). "-" denotes that the method could not converge on the specific dataset.

Method	Classification [% error ↓]			Regression [MAE ↓]	
	2-Moons	ONP	Elec2	House	Appliance
Vanilla-MLP	22.4 ± 4.6	33.8 ± 0.6	23.0 ± 3.1	11.0 ± 0.36	10.2 ± 1.1
CDOT (Jimenez et al., 2019)	9.3 ± 1.0	34.1 ± 0.0	17.8 ± 0.6	-	-
CIDA (Wang et al., 2020)	10.8 ± 1.6	34.7 ± 0.6	14.1 ± 0.2	9.7 ± 0.06	8.7 ± 0.2
GI (Nasery et al., 2021)	3.5 ± 1.4	36.4 ± 0.8	16.9 ± 0.7	9.6 ± 0.02	8.2 ±0.6
DRAIN (Bai et al., 2023)	**3.2 ± 1.2**	38.3 ± 1.2	12.7 ± 0.8	9.3 ± 0.14	6.4 ± 0.4
Vanilla-Transformer	25.2 ± 0.9	33.6 ± 0.5	22.5 ± 0.6	11.8± 0.3	5.6 ± 0.4
Attempt (Asai et al., 2022)	21.1 ± 1.1	34.1 ±0.6	12.3 ±0.8	9.0 ±0.4	4.9 ±0.5
Ours	8.1 ± 1.0	**32.7 ± 0.7**	**10.6± 0.9**	**8.9± 0.20**	**4.7 ± 0.3**

Table 2: Performance comparison of our method with DRAIN (Bai et al., 2023) and ATTEMPT (Asai et al., 2022) on Crypto dataset in terms of root mean square error $\times 10^3$.

Method	#params	time (s)	In domain	D_{t1}	D_{t2}	D_{t3}
DRAIN-2FC	8M	1634	4.97± 0.08	5.22± 0.08	7.78± 0.12	7.98 ± 0.11
DRAIN-3FC	239M	2520	4.61 ± 0.09	4.95 ±0.07	7.38 ± 0.09	7.47 ± 0.13
DRAIN-4FC	254M	2827	3.66± 0.08	3.74± 0.08	6.82 ± 0.10	7.03 ± 0.15
Vanilla-Trans.	69k	239	4.08 ± 0.08	4.44± 0.07	7.28 ± 0.09	7.55 ± 0.08
Attempt	93K	684	3.85±0.10	4.29± 0.11	7.51 ± 0.12	7.75 ±0.13
Attempt-m	93K	684	3.79± 0.12	4.12± 0.10	7.16± 0.17	7.43 ± 0.15
Ours	94K	717	**3.53± 0.06**	**3.57± 0.07**	**6.66 ±0.10**	**6.89 ± 0.09**

Table 3: Experimental results on AmazPantry Dataset.

AmazPantry Dataset		precision	recall	f1 score
BERT	D_{t1}	0.69	0.46	0.56
	D_{t2}	0.52	0.35	0.42
	D_{t3}	0.52	0.77	0.62
Ours	D_{t1}	0.68	0.50	0.58
	D_{t2}	0.55	0.43	0.49
	D_{t3}	0.56	0.76	0.64

Table 4: Ablation of effect of P_G, P_T using Crypto and Elec2 datasets. ✓indicates the prompt is used.

P_G	P_T	Crypto [RMSE $\times 10^3$ ↓] D_{t1}	D_{t2}	D_{t3}	Elec2 [MAE ↓] D_t
✓		3.57	6.66	6.84	14.9
	✓	3.53	6.71	6.80	14.7
✓	✓	3.53	6.61	6.74	10.6

Table 5: Additional experimental results on Mackey-Glass with state-of-the-art transformers Scaleformer (Shabani et al., 2023) and Informer (Zhou et al., 2021).

Method	RMSE $\times 10^1$ ↓	
	Mackey-Glass with σ-modification	Mackey-Glass with variable cosine
Vanilla-Trans. (VT)	1.315 ±0.07	2.511±0.09
Informer	1.127 ±0.05	2.074 ±0.08
Ours (on VT)	0.982 ±0.06	1.975±0.06
Scaleformer	0.934 ±0.00	1.967 ±0.01
Ours (on Scaleformer)	**0.802 ±0.01**	**1.852 ±0.04**

enables effective modeling and anticipating temporal shifts, and enhances model robustness in dynamic environments. Our approach is both parameter- and time-efficient, making it well-suited for real-world deployment scenarios where data distributions evolve over time. Through extensive experiments on classification, regression, time-series forecasting, and NLP tasks, we demonstrated the scalability, generality, and effectiveness of our framework across diverse applications.

Impact Statements

Our research addresses the problem of adapting pre-trained models to data distribution changes over time. To our knowledge, it does not introduce new ethical or societal risks that need to be specifically highlighted here.

References

Arik, S. O., Yoder, N. C., and Pfister, T. Self-adaptive forecasting for improved deep learning on non-stationary time-series. *arXiv preprint arXiv:2202.02403*, 2022.

Asai, A., Salehi, M., Peters, M. E., and Hajishirzi, H. Attempt: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Empirical Methods in Natural Language Processing*, 2022.

Bai, G., Ling, C., and Zhao, L. Temporal domain generalization with drift-aware dynamic neural networks. In *International Conference on Learning Representations*, 2023.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Machine learning*, 2010.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 2020.

Cai, Z., Bai, G., Jiang, R., Song, X., and Zhao, L. Continuous temporal domain generalization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 2016.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *North American Chapter of the Association for Computational Linguistics*, 2018.

Dubey, A., Ramanathan, V., Pentland, A., and Mahajan, D. Adaptive methods for real-world domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Fan, X., Wang, Q., Ke, J., Yang, F., Gong, B., and Zhou, M. Adversarially adaptive normalization for single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The journal of machine learning research*, 2016.

Gao, T., Fisch, A., and Chen, D. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

Garg, V. K., Kalai, A., Ligett, K., and Wu, Z. S. Learn to expect the unexpected: Probably approximately correct domain generalization. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Gong, B., Shi, Y., Sha, F., and Grauman, K. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE conference on computer vision and pattern recognition*, 2012.

Gu, Y., Han, X., Liu, Z., and Huang, M. Ppt: Pre-trained prompt tuning for few-shot learning. In *Proceedings of Annual Meeting of the Association for Computational Linguistic*, 2022.

Hoffman, J., Darrell, T., and Saenko, K. Continuous manifold based adaptation for evolving visual domains. In *Conference on Computer Vision and Pattern Recognition*, 2014.

Hu, S., Liao, Z., Zhang, J., and Xia, Y. Domain and content adaptive convolution for domain generalization in medical image segmentation. *IEEE Transactions on Medical Imaging*, 2023.

Jimenez, G. O., Gheche, M. E., Simou, E., Maretic, H. P., and Frossard, P. Cdot: Continuous domain adaptation using optimal transport. *Optimal Transport & Machine Learning Workhop (NeurIPS)*, 2019.

Khirodkar, R., Yoo, D., and Kitani, K. Domain randomization for scene-specific car detection and pose estimation. In *Winter Conference on Applications of Computer Vision*, 2019.

Kim, J., Lee, J., Park, J., Min, D., and Sohn, K. Self-balanced learning for domain generalization. In *International Conference on Image Processing*, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lao, Q., Jiang, X., Havaei, M., and Bengio, Y. Continuous domain adaptation with variational domain-agnostic feature replay. *arXiv preprint arXiv:2003.04382*, 2020.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *CoRR*, 2021.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

Liu, A. H., Liu, Y.-C., Yeh, Y.-Y., and Wang, Y.-C. F. A unified feature disentangler for multi-domain image translation and manipulation. *Advances in neural information processing systems*, 2018.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 2023a.

Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., and Tang, J. Gpt understands, too. *AI Open*, 2023b.

Mackey, M. C. and Glass, L. Oscillation and chaos in physiological control systems. *Science*, 1977.

Mancini, M., Bulo, S. R., Caputo, B., and Ricci, E. Adagraph: Unifying predictive and continuous domain adaptation through graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Mitrovic, J., McWilliams, B., Walker, J. C., Buesing, L. H., and Blundell, C. Representation learning via invariant causal mechanisms. In *International Conference on Learning Representations*, 2021.

Nam, H., Lee, H., Park, J., Yoon, W., and Yoo, D. Reducing domain gap by reducing style bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Nasery, A., Thakur, S., Piratla, V., De, A., and Sarawagi, S. Training for the future: A simple gradient interpolation loss to generalize along time. *Advances in Neural Information Processing Systems*, 2021.

Nazari, N. H. and Kovashka, A. Domain generalization using shape representation. In *European Conference on Computer Vision*, 2020.

Ni, J., Li, J., and McAuley, J. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 188–197, 2019.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

Prakash, A., Boochoon, S., Brophy, M., Acuna, D., Cameracci, E., State, G., Shapira, O., and Birchfield, S. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *International Conference on Robotics and Automation*, 2019.

Qi, L., Wang, L., Shi, Y., and Geng, X. Unsupervised domain generalization for person re-identification: A domain-specific adaptive framework. *arXiv preprint arXiv:2111.15077*, 2021.

Qiao, F., Zhao, L., and Peng, X. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Qin, T., Wang, S., and Li, H. Generalizing to evolving domains with latent structure-aware sequential autoencoder. In *International Conference on Machine Learning*. PMLR, 2022.

Rame, A., Dancette, C., and Cord, M. Fishr: Invariant gradient variances for out-of-distribution generalization. *International Conference on Machine Learning*, 2022.

Shabani, A., Abdi, A., Meng, L., and Sylvain, T. Scaleformer: Iterative multi-scale refining transformers for time series forecasting. *International Conference on Learning Representations*, 2023.

Shankar, S., Piratla, V., Chakrabarti, S., Chaudhuri, S., Jyothi, P., and Sarawagi, S. Generalizing across domains via cross-gradient training. *International Conference on Learning Representations*, 2018.

Sun, X., Wu, B., Zheng, X., Liu, C., Chen, W., Qin, T., and Liu, T.-Y. Recovering latent causal factor for generalization to distributional shifts. In *Conference on Neural Information Processing Systems*, 2021.

Tian, C. X., Li, H., Xie, X., Liu, Y., and Wang, S. Neuron coverage-guided domain generalization. *The IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Triantafillou, E., Larochelle, H., Zemel, R., and Dumoulin, V. Learning a universal template for few-shot dataset generalization. In *International conference on machine learning*, 2021.

Volpi, R., Namkoong, H., Sener, O., Duchi, J. C., Murino, V., and Savarese, S. Generalizing to unseen domains via adversarial data augmentation. *Conference on Neural Information Processing Systems*, 2018.

Vu, T., Lester, B., Constant, N., Al-Rfou, R., and Cer, D. Spot: Better frozen model adaptation through soft prompt transfer. *Association for Computational Linguistics*, 2021.

Wang, B., Lapata, M., and Titov, I. Meta-learning for domain generalization in semantic parsing. In *North American Chapter of the Association for Computational Linguistics*, 2021.

Wang, H., He, H., and Katabi, D. Continuously indexed domain adaptation. *International Conference on Machine Learning*, 2020.

Wang, M. and Deng, W. Deep visual domain adaptation: A survey. *Neurocomputing*, 2018.

Wu, G. and Gong, S. Collaborative optimization and aggregation for decentralized domain generalization and adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Yang, Y. and Hospedales, T. M. Multivariate regression on the grassmannian for predicting novel domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Yue, X., Zhang, Y., Zhao, S., Sangiovanni-Vincentelli, A., Keutzer, K., and Gong, B. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *International Conference on Computer Vision*, 2019.

Zeng, Q., Wang, W., Zhou, F., Xu, G., Pu, R., Shui, C., Gagné, C., Yang, S., Ling, C. X., and Wang, B. Generalizing across temporal domains with koopman operators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Zhao, Y., Zhong, Z., Yang, F., Luo, Z., Lin, Y., Li, S., and Sebe, N. Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of Association for the Advancement of Artificial Intelligence*, 2021.

A. Appendix

A.1. Related Work

Domain generalization and adaptation. Domain adaptation (DA) aims at tailoring models to specific target domains, using the similarities that exist between these domains (Ben-David et al., 2010; Wang & Deng, 2018; Hoffman et al., 2014; Jimenez et al., 2019; Lao et al., 2020; Wang et al., 2020; Yang & Hospedales, 2016; Courty et al., 2016; Gong et al., 2012; Mancini et al., 2019; Shabani et al., 2023; Wang et al., 2020; Ganin et al., 2016). Domain generalization (DG) methods build upon the insights from domain adaptation and aim to enhance the generalization capability of models to unseen target domains, mostly by data augmentation or discovering domain-invariant features (Yue et al., 2019; Prakash et al., 2019; Shankar et al., 2018; Volpi et al., 2018; Nazari & Kovashka, 2020; Khirodkar et al., 2019; Qiao et al., 2020; Liu et al., 2018; Zhao et al., 2021; Garg et al., 2021; Qi et al., 2021; Fan et al., 2021; Mitrovic et al., 2021; Hu et al., 2023; Triantafillou et al., 2021; Nam et al., 2021; Sun et al., 2021; Wu & Gong, 2021; Dubey et al., 2021; Kim et al., 2021; Wang et al., 2021; Tian et al., 2022; Rame et al., 2022).

Temporal domain generalization. Temporal DG, compard with standard DG, remains an under-explored area with relatively little research dedicated to addressing its challenges. Unlike standard DG, which aims for domain-invariant representations across different domains, temporal DG focuses on capturing the temporal dynamics of the data, enabling generalization to unseen future temporal domains. The Gradient Interpolation (GI) (Nasery et al., 2021) method uses adversarial training to generalize over time, altering the leaky ReLU activation for time dependence. LSSAE (Qin et al., 2022) proposed a probabilistic framework modeling the underlying continuous characteristics in the latent space of deep neural networks. TKNets (Zeng et al., 2024) leverages Koopman operator theory to model temporal domain shifts by establishing linear transition relations between evolving domains. Koodos (Cai et al., 2024) learns three flows including Data Flow, the Model Flow, and the Koopman Representation Flow, and encourages consistent characteristics among the three flows. DRAIN (Bai et al., 2023) models model temporal dynamics by predicting network weights evolving over time. Most existing methods are not ideal for scalable and efficient temporal adaptation, either due to architectural constraints (such as LSSAE) or high computational and memory costs (such as DRAIN, TKNets, and Koodos). In contrast, our approach leverages lightweight prompting mechanisms to dynamically adapt to temporal shifts without excessive parameter overhead, achieving efficient generalization while remaining compatible with large-scale foundation models.

Prompting Mechanism: The concept of prompt-based learning has gained significant traction in the field of natural language processing (NLP) for adapting pre-trained language models (PLMs) to various downstream tasks. This framework involves conditioning the model with additional instructions to perform specific tasks (Peters et al., 2018; Devlin et al., 2018; Brown et al., 2020; Lester et al., 2021; Vu et al., 2021; Gu et al., 2022; Gao et al., 2021; Liu et al., 2023b;a; Li & Liang, 2021).

A.2. Network architectures and experimentation details

Below, we detail the architecture and other specific experiment details for each dataset.

Architecture of frozen backbone network:

We choose backbones for each dataset to enable a fair comparison with state-of-the-art methods.

For the time series dataset **Crypto**, the initial inputs are passed through a linear layer, resulting in 64-dimensional embeddings. These embeddings are then processed by a transformer encoder layer. The transformer comprises a single encoder layer with four heads, and the hidden layers with dimensionality of 128. Finally, the output is passed through another linear layer to achieve the desired output size. We utilize the mean squared error (MSE) loss for Crypto dataset.

For the datasets that are reported in DRAIN (Bai et al., 2023), the initial inputs for **Elec2**, **2Moons**, **House**, and **Appliance** are transformed through a linear layer to produce 128-dimensional embeddings, whereas for **ONP** it is a 32-dimensional embedding. These embeddings are subsequently processed by a transformer encoder layer. Notably, to align closely with the DRAIN paper's structure, our transformer encoder employs just one linear layer in the feed-forward segment, as opposed to the conventional two. The transformer setup involves a single encoder layer with one head. The hidden layers maintain a 128-dimensional structure for all datasets, with the exception of **ONP**, which is set at 64. Outputs are then channeled through another linear layer to derive the desired size. For regression datasets, we adopt the mean squared error (MSE) loss, and for classification datasets, we use binary cross-entropy loss.

Algorithm 1 Prompt Learning and Inference

Require: Source domains, target domains, pre-trained model f_θ, temporal prompt generator g_ω
Ensure: Domain-specific prompts $P_{S1}, P_{S2}, \ldots, P_{S\tau}$, temporal prompts $P_{T2}, P_{T3}, \ldots, P_{T(\tau+1)}$, generic prompt P_G

function DomainSpecificPromptLearning
  for each domain D_t in source domains ($1 \leq t \leq \tau$) **do**
    Prepend X_t with $P_{S(t)}$
    Process $[P_{S(t)}; X_t]$ using frozen backbone f_θ
    Train to maximize likelihood $\mathcal{P}_\theta(Y_t|[P_{S(t)}; X_t])$ with θ fixed
  end for
  Return prompts $P_{S1}, P_{S2}, \ldots, P_{S\tau}$
end function
function TemporalPromptLearning
  Initialize temporal prompt generator g_ω
  for each domain D_t with $2 \leq t \leq \tau$ **do**
    Provide $P_{S1}, \ldots, P_{S(t-1)}$ to g_ω
    Generate $P_{T(t)}$
    Prepend X_t with P_G and $P_{T(t)}$
    Process $[P_{T(t)}; P_G; X_t]$ with f_θ
    Train to maximize $\mathcal{P}_\theta(Y|[P_{T(t)}; P_G; X_t])$ with θ fixed
  end for
end function
function Inference
  Forecast $P_{T(\tau+1)}$ given $P_{S1}, \ldots, P_{S\tau}$ and P_G
  Predict $Y_{(\tau+1)}$ using f_θ and $[P_{T(\tau+1)}; P_G; X_{(\tau+1)}]$
end function=0

Domain-specific prompts: Domain-specific prompts are learnable parameters, whose sizes match the embedding dimensions for each dataset.

Temporal prompt generator: We employ a transformer with a single encoder layer and 1 heads as our temporal prompt generator. The transformer's hidden layers have a consistent 128-dimensional configuration.

Baseline Methods:

- DRAIN-2FC, DRAIN-3FC, DRAIN-4FC. The original DRAIN paper employed two fully connected layers (DRAIN-2FC) in both encoding and decoding functions to transform the latent representations between LSTM units. To potentially boost DRAIN's performance, we also explored using three and four linear layers in both encoding and decoding functions. We denote these models DRAIN-3FC and DRAIN-4FC, respectively. DRAIN-Best refers to the model achieving the highest performance using these configurations for the encoding/decoding functions.
- Vanilla-MLP, the MLP-based backbone network from DRAIN (Bai et al., 2023), which is trained on the combined source domains.
- Vanilla-Transformer, our method's transformer-based backbone network, which is trained on the combined source domains.

A.3. Ablation Study

The number of training domains.: To study the impact of the number of training domains on method performance, we conducted an ablation study on Mackey-Glass synthetic data with varying numbers of training domains. Table 6 shows that the performance of our method improves as the number of source domains increases, which is expected as a greater number of observed source domains make temporal patterns more evident.

Non-sequential temporal prompt learning: In the main paper, temporal prompts are generated sequentially. An alternative option is to generate them non-sequentially. In this experiment, we opt for a non-sequential training paradigm, wherein

Table 6: Impact of number of training domains on vanilla transformer and our method.

# Training domains	MSE $\times 10^1 \downarrow$			
	Mackey-Glass with σ-modification		Mackey-Glass with variable cosine	
	Vanilla-Trans.	Ours	Vanilla-Trans.	Ours
4	1.818	1.305	3.007	2.581
9	1.315	0.982	2.511	1.975
19	0.877	0.787	3.326	2.547
49	0.930	0.739	1.645	1.440

the model is exposed to all source domains simultaneously during the training process. To be precise, the temporal prompt generator, denoted as g_ω, takes all domain-specific prompts $P_{S1}, P_{S2}, \ldots, P_{S(\tau)}$, and generates temporal prompts $P_{T2}, P_{T3}, \ldots, P_{T(\tau+1)}$. Table 7 compares performance of sequential temporal prompt generation vs non-sequential prompt generation, and it can be seen that performance is on par with the main method.

Table 7: Comparing sequential temporal prompt generation vs non-sequential one.

Method	Classification error [in % $\downarrow$]			Regression [MSE $\downarrow$]	
	2-Moons	ONP	Elec2	House	Appliance
Vanilla-Transformer	25.2 $\pm$ 0.9	33.6 $\pm$ 0.5	22.5 $\pm$ 0.6	11.8$\pm$ 0.3	5.6 $\pm$ 0.4
Attempt (Asai et al., 2022)	21.15 $\pm$ 1.1	34.10 $\pm$0.6	12.26 $\pm$0.8	9.0 $\pm$0.4	4.9 $\pm$0.5
Ours	**8.1 $\pm$ 1.0**	32.7 $\pm$ 0.7	**10.6$\pm$ 0.9**	8.9$\pm$ 0.20	**4.7 $\pm$ 0.3**
Ours (not sequential)	8.4 $\pm$ 0.9	**31.8 $\pm$ 0.7**	11.2$\pm$ 0.8	**8.6$\pm$ 0.14**	4.9 $\pm$ 0.4

Impact of embedding and prompt size on model performance : Table 8 shows ablations on embedding and prompt size. It is observed that for Crypto dataset, embedding/prompting size 64 and 128 provide similar better performance, and smaller embedding/prompting size results in a more parameter-efficient network; 64 is selected for better model size and performance tradeoff.

Table 8: Impact of prompt size and embedding size using Crypto dataset, in terms of root mean square error $\times 10$.

Prompt size & Embedding size	Vanilla Transformer			Temporal prompting		
	D_{t1}	D_{t2}	D_{t3}	D_{t1}	D_{t2}	D_{t3}
32	4.20	7.20	7.45	3.57	6.64	6.85
64	4.42	7.19	7.43	3.53	6.61	6.74
128	4.52	7.59	7.79	3.45	6.58	6.79
256	4.45	7.25	7.39	3.45	6.64	6.79

Impact of temporal prompting module layers on model performance : Table 9 presents an additional study on the effect of the number of layers in the temporal prompt generation module using the Mackey-Glass data.

A.4. Results on Synthetic Data

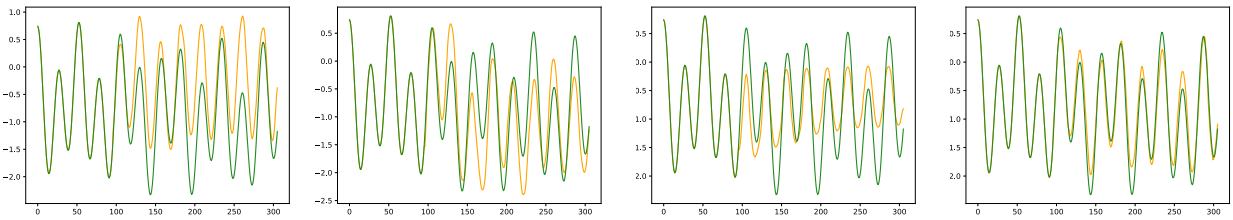
To evaluate the proposed method under a controlled setting, we constructed synthetic datasets derived from the Mackey-Glass equations (Mackey & Glass, 1977) and sums of cosines. For the former, we used

$$x(t+1) = x(t) + \beta \frac{x(t-\sigma)}{1 + x(t-\sigma)^n} - \gamma x(t) \tag{2}$$

where $\beta = 0.2, \gamma = 0.1, n = 15, \sigma = 18, t_{max} = 2600$, and $x(t) = 0.1$ *if* $t < 18$. For the latter, we used

Table 9: Impact of temporal prompting module layers on model performance in terms of MSE.

Number of Layers	Mackey-Glass with σ-modification	Mackey-Glass with variable cosine
Vanilla Transformer (0)	0.1315	0.2511
1	0.0982	0.1975
2	0.0950	0.2053
3	0.1022	0.2119



Vanilla Transformer DRAIN Attempt Ours

Figure 2: Qualitative results on Sum of Cosines synthetic dataset generated with phase-frequency modification and addition of a variable cosine wave.

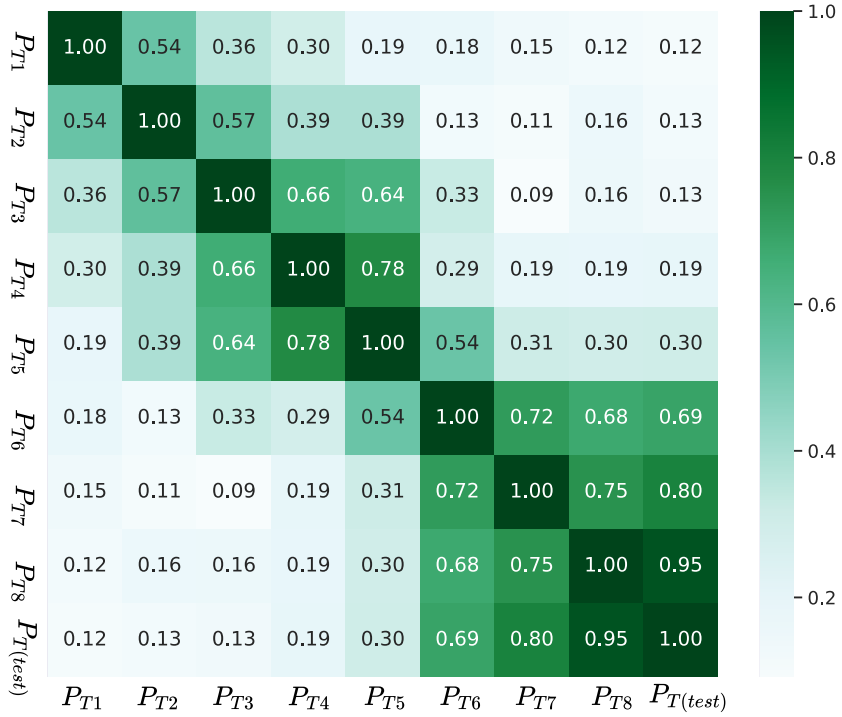


Figure 3: Pairwise comparisons of learned temporal prompts ($P_T s$) across domains in the Sum of Cosines synthetic dataset with both types of synthetic drift. The values shown are cosine similarities.

$$x(t) = \cos\left(a + \frac{\pi h}{\alpha} t\right) + \cos\left(b + \frac{\pi}{\beta} t\right) \tag{3}$$

Table 10: Comparison of proposed method, vanilla transformer, and state-of-the-art approaches on synthetic datasets generated based on Mackey-Glass equations and Sum of Cosines, in mean square error $\times 10^1$.

Method	MSE ↓			
	Mackey-Glass		Sum of Cosines	
	with σ-modification	with variable cosine	with phase-frequency modification	with phase-frequency modification + var. cosine
DRAIN-Best	1.140 ±0.08	2.164 ± 0.08	0.085 ±0.01	2.93 ±0.07
Vanilla Transformer	1.315 ±0.07	2.511 ±0.09	0.1191 ±0.19	3.708 ±0.07
Attempt	1.278 ±0.10	2.199 ±0.12	0.091 ±0.02	2.974 ±0.10
Ours	**0.982 ±0.06**	**1.975 ±0.05**	**0.068±0.00**	**2.489±0.07**

where $\alpha = 100, \beta = 13, a = 40, b = 10, h = 1$, and $0 < t < 2600$.

We employed two strategies to induce temporal drift: parameter modification, and addition of a cosine wave with variable phases and frequencies across domains, given by

$$0.5 \times \cos\left(100i + \frac{\pi(i+1)}{300}t\right) \tag{4}$$

for domain i. For Mackey-Glass, we generated two datasets: one dataset modifying $\sigma = 8 + i \times 2$ for each domain i, and one dataset adding Equation (4) to Equation (2). For Sum of Cosines, we generated two datasets: one dataset modifying $a = i$ and $h = i + 1$ for each domain i, and one dataset adding Equation (4) to the parameter-modified dataset. Visualizations of the synthetic datasets are shown in the appendix.

Results on the synthetic datasets are summarized in Table 10. We also qualitatively visualize the results on Sum of Cosines in Figure 2. The proposed method consistently outperformed the Vanilla Transformer, DRAIN, and Attempt models on the synthetic data. Quantitatively, our model achieved the lowest MSE across both Mackey-Glass and Sum of Cosines datasets with either type of the temporal drifts. Qualitatively, it also demonstrated superior adaptability and accuracy. Figure 3 visualizes the pairwise cosine similarities among the learned temporal prompts across different domains for the Sum of Cosines dataset with both types of synthetic drift. It is observed that temporal prompts from neighboring domains have higher similarity than other domains.

Below we provide visualizations of the synthetic data.

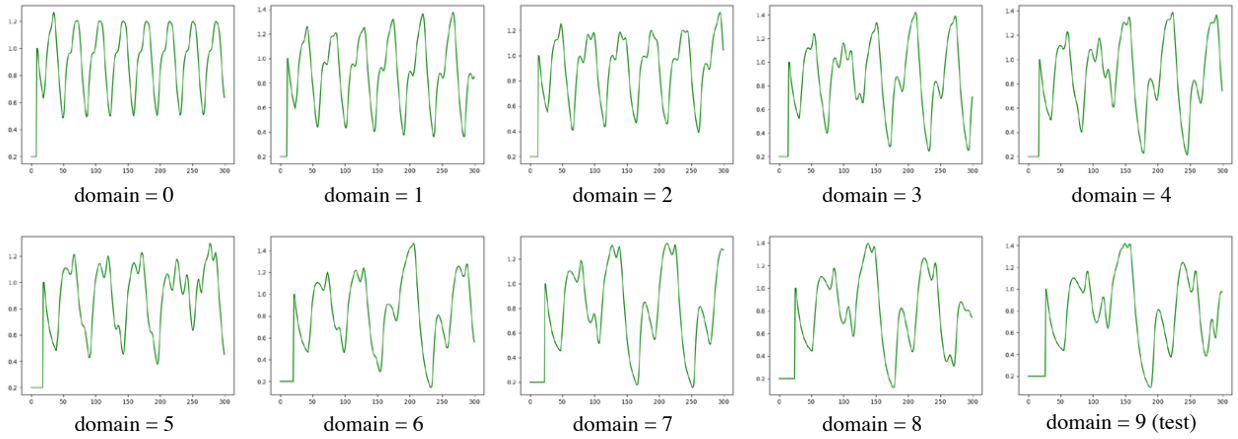


Figure 4: Applying temporal shift to Mackey-Glass time series by modifying σ.

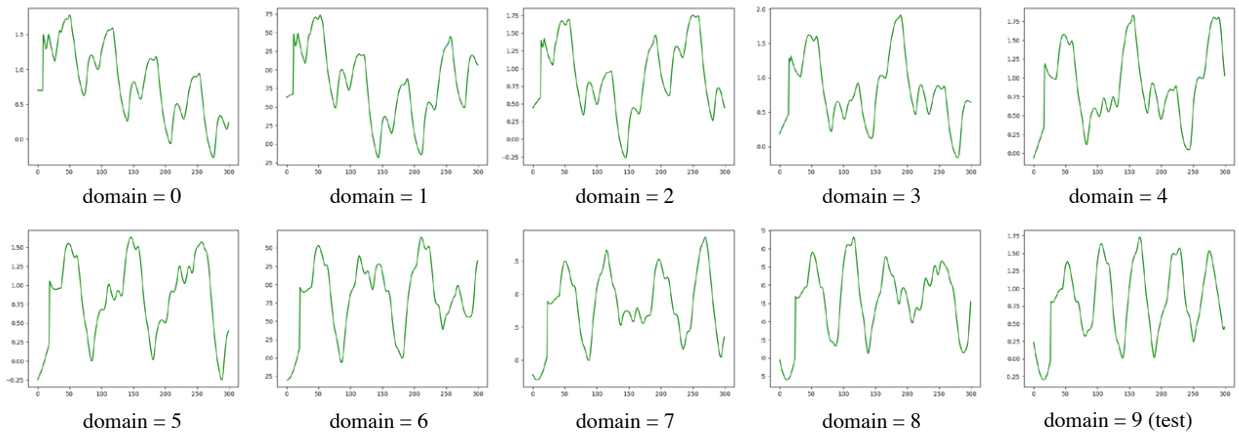


Figure 5: Applying temporal shift to Mackey-Glass time series by adding variable cosine wave.

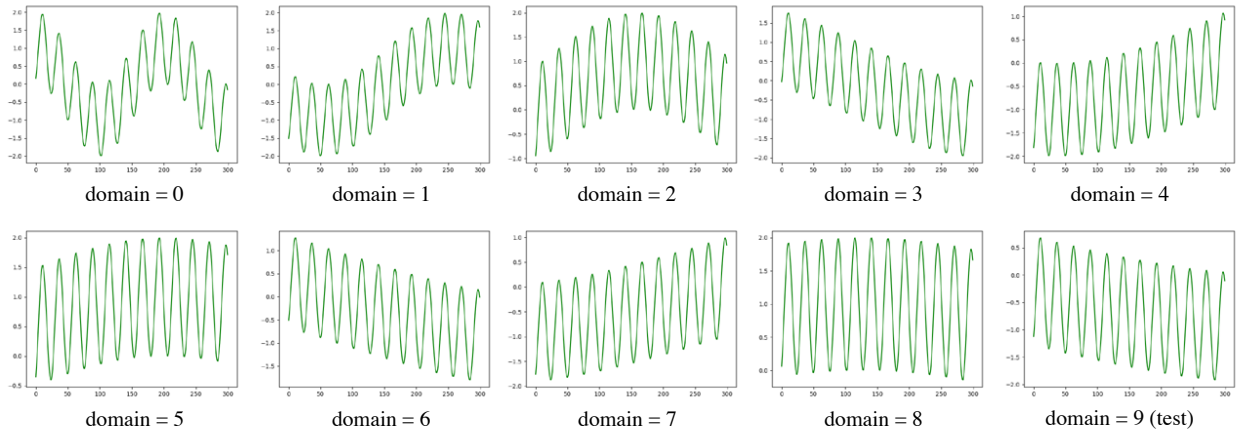


Figure 6: Applying temporal shift to Sum of Cosines time series by modifying phase and frequency.

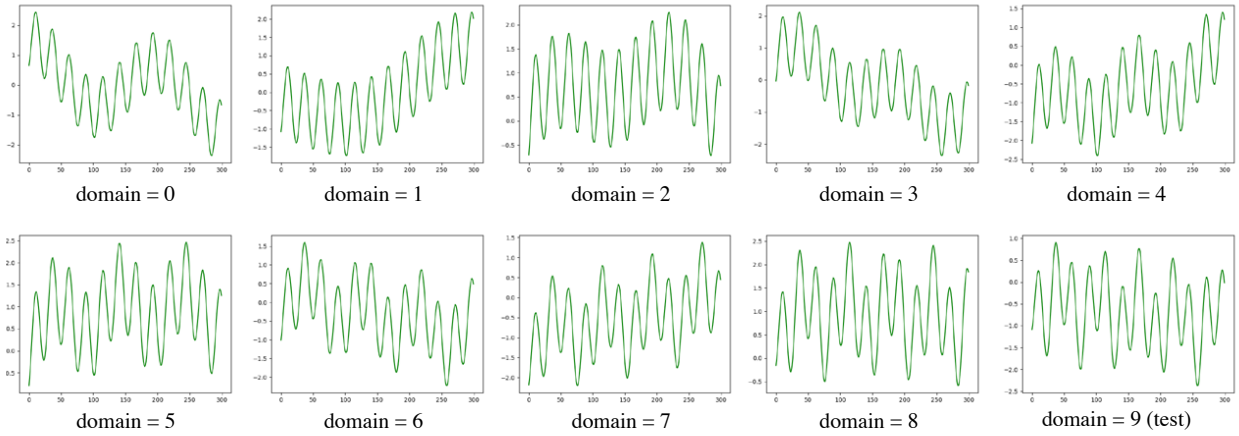


Figure 7: Applying temporal shift to Sum of Cosines time series by modifying phase and frequency, and adding another variable cosine wave.