A Bilevel Optimization Framework for Peak Trough Aware Time Series Forecasting

Jungmin Kim[1]
Jaesik Choi[*1,2]
[1]Korea Advanced Institute of Science and Technology (KAIST)
[2]INEEJI
{aldirl7, jaesik.choi}@kaist.ac.kr

Abstract

In Time Series Forecasting(TSF), accurately predicting peaks and troughs is critical but challenging with traditional metrics like mean squared error (MSE). We propose a bilevel optimization framework that combines point-wise L_p loss and dynamic time warping (DTW) loss to better capture these critical features. In our bilevel optimization formulation, the upper-level problem minimizes the L_p norm distance, while the lower-level problem targets DTW loss, allowing dynamic adjustment between the two objectives. By separating parameters within a single TSF model, our approach remains model-agnostic and efficient. Our experiment results show that the bilevel optimization framework effectively achieves the primary TSF objective and excels in peak-trough accuracy, outperforming baseline methods in the newly defined Peak-Trough MSE (PTMSE) metric. Qualitative visualizations confirm that our framework enables sharp predictions, and better-capturing peaks and troughs.

1 Introduction

Time series forecasting (TSF) aims to predict the time series values over a forecast horizon based on the observed historical values. In many practical applications — such as energy management, traffic control, and financial trading — accurately predicting peaks and troughs is crucial [Hyndman and Fan, 2009; Yin and Shang, 2016; Andersen *et al.*, 2005] as failure to predict sharp changes in the signal can lead to substantial losses [Palshikar and others, 2009]. For instance, while gradual changes in electricity demand are manageable, sudden spikes or drops can be critical because these can lead to system overload or collapse.

The primary objective in TSF is to closely align predicted values with actual outcomes over the forecast horizon. Common metrics such as the mean absolute error (MAE) and mean squared error (MSE) are widely used to measure this accuracy using point-wise L_p norm distance. However, these metrics are limited in capturing critical features of the data, such as sharp peaks and troughs [Esling and Agon, 2012; Le Guen and Thome, 2019; Lee *et al.*, 2022]. Furthermore, when the point-wise losses are employed as a loss function to train parametric TSF models, the error from each point contributes equally towards the overall loss; hence, there is no inherent mechanism to put more importance on accurately predicting peaks and troughs.

Relatedly, Dynamic Time Warping (DTW) has been proposed as an alternative loss function designed to train TSF models to more closely reflect the shapes of ground truth data [Cuturi and Blondel, 2017; Abid and Zou, 2018; Mensch and Blondel, 2018; Le Guen and Thome, 2019; Le Guen and Thome, 2020]. Unlike the point-wise distance metrics, DTW initially aligns the time indices of predicted and actual data for comparison [Sakoe and Chiba, 1978]. This alignment allows DTW to detect and emphasize significant variations, such as peaks and troughs, by ensuring that extreme values in one series correspond to multiple indices in another [Cuturi and Blondel, 2017].

While DTW is effective at aligning time series data to match patterns [Esling and Agon, 2012], its inherent design to be invariant to time shifts and distortions may lead to the neglect of precise timing of events. This invariance can result in inaccurately predicted time series patterns, such as periodicity and trends. Consequently, TSF models trained with DTW loss often demonstrate suboptimal performance in point-wise distance metrics [Lee *et al.*, 2022].

To develop a TSF model that captures both the point-wise accuracy and the peaks and troughs of the data, we propose a bilevel optimization framework. This framework simultaneously minimizes the point-wise L_p loss and dynamic time warping (DTW) loss. The upper-level (leader) problem focuses on minimizing the L_p distance, while the lower-level (follower) problem targets minimizing the DTW loss. This hierarchical structure allows the follower's optimization to adjust to the leader's decisions dynamically. By separating parameters within a single TSF model to designate leader and follower roles, our approach maintains model size, ensures model agnosticism, and avoids the need for additional sub-models.

To validate the effectiveness of the bilevel optimization framework, we evaluate MSE, DTW, and MultiObj losses as baselines, comparing them using the state-of-the-art TSF model, PatchTST. Evaluation results show that the bilevel optimization framework effectively achieved primary TSF objectives and peak-trough accuracy, outperforming baselines in a newly defined Peak-Trough MSE (PTMSE). In simple qualitative visualization of prediction, our framework allows the model to make sharp predictions, better capturing peaks and troughs.

Therefore, our contributions are as follows:

- We highlight the need for peak-aware time series forecasting (TSF).
- We propose using DTW loss as a sub-objective to better capture the peak-trough characteristics of time series.
- We develop a bilevel optimization framework that maintains the main objective of TSF while efficiently enabling peak-trough prediction, implemented in a model-agnostic manner applicable to any TSF model.

2 Background

2.1 DTW and differentiable soft-DTW

Dynamic Time Warping (DTW) is a distance measure between two temporal sequences by aligning time indices, so-called the measure of sequence similarity. Consider $\mathbf{X} \in \mathbb{R}^{m \times d}$ and $\mathbf{Y} \in \mathbb{R}^{n \times d}$ as two d-dimensional time series of lengths m and n. We represent their elements Given two time series $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_m)$, both in $\mathbb{R}^{p \times n}$ and $\mathbb{R}^{p \times m}$, respectively, and a differentiable cost function $\delta : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}_+$, the Dynamic Time Warping (DTW) distance can be defined as:

$$\mathrm{DTW}(\mathbf{x}, \mathbf{y}) := \min_{A \in \mathcal{A}_{n,m}} \langle A, \Delta(\mathbf{x}, \mathbf{y}) \rangle,$$

where $\Delta(\mathbf{x}, \mathbf{y}) = [\delta(x_i, y_j)]_{ij} \in \mathbb{R}^{n \times m}$ is the cost matrix, and $\mathcal{A}_{n,m} \subset \{0,1\}^{n \times m}$ represents the set of alignment matrices, which are paths on a $(n \times m)$ matrix that connect the upper-left $(1,1)$ matrix entry to the lower-right (n,m) entry using only $\downarrow$, $\rightarrow$, and $\searrow$ moves. In our setting, the length of each time series n and m are the same as h and the dimension p is 1. L_2 norm Euclidean distance is used as a differentiable cost function δ.

Soft-DTW is a differentiable variant of DTW designed to address the non-differentiability of the original DTW, making it more suitable for gradient-based optimization. Instead of selecting the optimal alignment path, soft-DTW considers all possible paths by applying a soft-minimum operator. Formally, soft-DTW is defined as:

$$\mathrm{softDTW}_\gamma(\mathbf{x}, \mathbf{y}) := -\gamma \log \left(\sum_{A \in \mathcal{A}_{n,m}} \exp \left(-\frac{\langle A, \Delta(\mathbf{x}, \mathbf{y}) \rangle}{\gamma} \right) \right)$$

where $\gamma > 0$ is a smoothing parameter. As $\gamma \to 0$, soft-DTW converges to the standard DTW distance. The smoothing parameter γ controls the trade-off between the exact DTW distance and a smoother approximation, allowing for differentiability and more stable gradient computations during model training. In this work, we use DTW as a metric and loss function for the TSF deep learning model. Hence, any subsequent mention of DTW refers to the differentiable form, Soft-DTW, and for simplicity, we denote this as DTW.

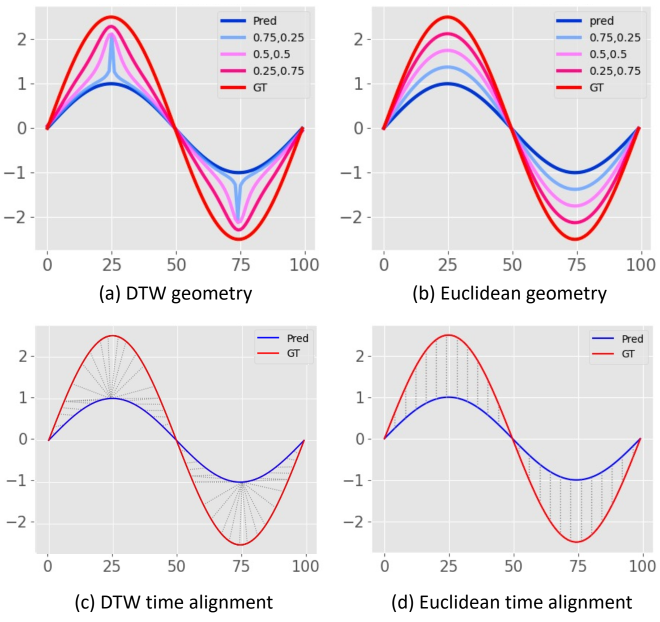

Figure 1: Interpolation between two sine functions (blue and red) by using DTW and Euclidean distance. For Euclidean distance, linear interpolation is applied and a barycenter is computed for DTW. The interpolation weights (λ_1, λ_2) are (0.25, 0.75), (0.5, 0.5), and (0.75, 0.25) corresponding to the ground truth and prediction. We refer to these interpolation figures from [Cuturi and Blondel, 2017].

2.2 Euclidean and DTW geometry

In this section, we examine DTW and L_p norm Euclidean geometries to understand how DTW loss enhances sharper predictions compared to Euclidean loss [Cuturi and Blondel, 2017; Le Guen and Thome, 2019]. Note that the direction in which a prediction loss decreases indicates the direction the loss function promotes. Figures 1-(a) and 1-(b) illustrate interpolation between two series (assuming red as ground truth and blue as prediction) using DTW and Euclidean distances, respectively. Under DTW (Figure 1-(a)), the blue prediction adjusts sharply to match the peaks and troughs of the red ground truth, highlighting its focus on extreme points. Conversely, Euclidean geometry (Figure 1-(b)) results in a uniform shift without special emphasis on extremes.

The geometric difference between the two measures stems from their methods of comparing time indices: lock-step for L_p and elastic for DTW. DTW (Figure 1-(c)) aligns time indices (dotted lines) to better match extreme values and reduce distances at these points, while Euclidean (Figure 1-(d)) maintains original indices and measures all distances uniformly. This difference in time index matching illustrates how each loss function penalizes predictions based on different index priorities.

3 Problem Formulation

3.1 Time series forecasting problem

We consider the univariate TSF problem with a dataset $\mathcal{D} = \{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^N$, where $\mathbf{x}^i = [y_1^i, ..., y_T^i] \in \mathbb{R}^T$ is the ith historical time series of length T and $y_t^i \in \mathbb{R}$ represents the

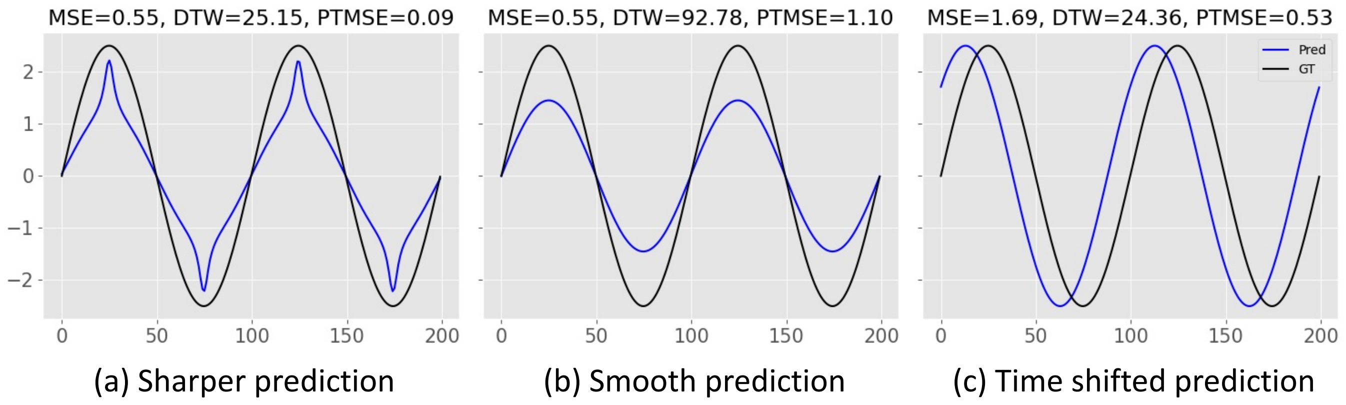

Figure 2: Three types of predictions are illustrated: (a) Sharp prediction, (b) Smooth prediction, and (c) Prediction with temporal error, using black for ground truth (GT) and blue for predictions (Pred). Pred-(a) is sharper than pred-(b), both having a similar mean squared error (MSE) of 0.55 but Pred-(a) has a significantly lower DTW (25.15) and peak-trough MSE (PTMSE) (0.09) compared to Pred-(b) (92.78, 1.10). Pred-(c), while having a similar DTW loss to Pred-(a) (a:25.15,c:24.36), shows a higher MSE of 1.69, reflecting that DTW by itself cannot achieve TSF's basic goal.

scalar value at time t. $\mathbf{y}^i = [y^i_{T+1}, ..., y^i_{T+H}] \in \mathbb{R}^H$ is the forecasting target of length H [Chatfield, 2000]. Let $f(\mathbf{x};\theta) : \mathbb{R}^T \mapsto \mathbb{R}^H$ be a parametric TSF model that maps inputs $\mathbf{x}$ to predictions $\hat{\mathbf{y}}$. Then, we consider learning the optimal parameters θ^* under the empirical risk minimization (ERM) framework with a loss function $\mathcal{L}$ [Wasserman, 2010]:

$$\theta^* \in \arg\min_{\theta} \; Risk_{\mathcal{D}}(\theta) := \mathbb{E}_{\mathcal{D}}\big[\mathcal{L}(\hat{\mathbf{y}}^i, \mathbf{y}^i)\big] \tag{1}$$

The most commonly used loss function is a point-wise L_p norm distance; specifically, when $p = 2$, we get the mean squared error (MSE) loss, or $\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{H}\sum_{t=1}^{H}(\hat{y}_t - y_t)^2$.

3.2 Optimization problem formulation

We aim to develop a TSF model that minimizes the point-wise L_p distance and is adept at making precise predictions that capture the peaks and troughs within a time series. More intuitively, as demonstrated in Figure 2, we prefer the prediction of (a) 'Sharper prediction' over (b) 'Smooth prediction' when both have similar MSE errors, and we want to train the model to reflect this preference. Building on prior work [Cuturi and Blondel, 2017; Le Guen and Thome, 2019; Le Guen and Thome, 2020], we hypothesize that incorporating DTW loss encourages models to produce sharper predictions, which we further explore in Section 5.

The most straightforward way to train a TSF model that minimizes DTW distances without significantly compromising the original L_p norm loss is to reframe the TSF problem as a multi-objective (MultiObj) optimization problem through scalarization [Jahn, 1985]. Specifically, given a DTW loss weight $\lambda_{dtw} > 0$, MultiObj optimizes the following:

$$\min_{\theta} \mathbb{E}_{\mathcal{D}}\big[L_p(\hat{\mathbf{y}}^i, \mathbf{y}^i) + \lambda_{dtw} * \mathrm{DTW}(\hat{\mathbf{y}}^i, \mathbf{y}^i)\big]. \tag{2}$$

Despite its apparent simplicity, selecting the appropriate value for λ_{dtw} to achieve the optimal balance between the two objectives is nontrivial. For instance, as noted in [Le Guen and Thome, 2019], focusing primarily on minimizing DTW loss ($\lambda_{dtw} \gg 1$) results in significantly suboptimal point-wise predictions that fail to capture major trends and periodicity in the dataset. Conversely, setting $\lambda_{dtw} \ll 1$ can render the effect of DTW negligible. Moreover, when these two loss functions compete, optimization becomes more challenging as the model receives conflicting gradient signals.

To ensure that the model effectively learns the major trends and periodicity in a given dataset, we prioritize the L_p loss as the primary objective and treat DTW loss as secondary. One approach to maintaining this hierarchy of objectives is to formulate a constrained optimization problem [Bertsekas, 2014]. Here, we minimize the L_p loss while imposing a constraint on the DTW loss. Formally, given a threshold $\alpha > 0$ for DTW, we define a feasible set $\mathcal{F}_\alpha = \{\theta : \mathbb{E}_{\mathcal{D}}[\mathrm{DTW}(\hat{\mathbf{y}}, \mathbf{y})] \leq \alpha\}$. Then, the constrained optimization problem is defined as follows:

$$\min_{\theta \in \mathcal{F}_\alpha} \mathbb{E}_{\mathcal{D}}\big[L_p(\hat{\mathbf{y}}^i, \mathbf{y}^i)\big]. \tag{3}$$

The constrained optimization approach refines the solution space of the original unconstrained problem (1) to include only those solutions where the expected DTW is below the threshold α. This method is advantageous as it effectively filters out solutions with undesirably high DTW values. However, a significant challenge with this approach is determining the optimal for α. If α is set too low, the feasible set may become unrealistically restrictive; conversely, if α is too high, it may not adequately refine the solution space to reflect the desired preferences. Additionally, this approach lacks flexibility, as the constraint does not dynamically interact with the main objective during the optimization process.

4 Method

4.1 Bilevel optimization framework

To overcome the limitations of both the constrained and multi-objective optimization approaches, we propose a bilevel optimization framework that treats the peak and trough-aware TSF problem as a Stackelberg game

[Von Stackelberg, 2010]. In this framework, the upper-level (leader) problem focuses on minimizing the point-wise L_p distance, while the lower-level (follower) problem targets the minimization of the DTW. Importantly, the model predictions $\hat{\mathbf{y}}$ are determined by two distinct sets of parameters: θ_p for the leader and θ_{dtw} for the follower. The bilevel optimization can then be formulated as:

$$\min_{\theta_p} \mathbb{E}_{\mathcal{D}}\big[L_p(\mathbf{y}, \hat{\mathbf{y}}(\theta_p, \theta^*_{dtw}))\big] \tag{4}$$

$$\text{s.t. } \theta^*_{dtw}(\theta_p) = \arg\min_{\tilde{\theta}} \ \mathbb{E}_{\mathcal{D}}\big[\mathrm{DTW}(\mathbf{y}, \hat{\mathbf{y}}(\theta_p, \tilde{\theta}))\big]. \tag{5}$$

Since the follower selects the optimal response $\theta^*_{dtw}(\theta_p)$ based on the leader's decision θ_p, the follower's parameters are inherently dependent on the leader's parameters. The leader, aware of this dependency, can leverage this information when adjusting its parameters [Rajeswaran *et al.*, 2020].

Compared to the other optimization approaches, the bilevel approach presents several distinct advantages. First, it addresses the challenge of manually setting an optimal α in the constrained approach by transforming the constraint into the follower's optimization problem that adapts in response to the leader's decisions [Colson *et al.*, 2007]. Additionally, by clearly defining a hierarchical relationship between minimizing the L_p distance and the DTW, our approach ensures that the optimization of the DTW loss occurs within the optimal L_p solution space. This structure allows for more effective prioritization of objectives and provides a robust mechanism for achieving sharp predictions while still maintaining reasonable point-wise prediction performance.

4.2 Parameter separation for bilevel optimization

Bilevel optimization necessitates distinct objectives for leader and follower parameters. Typically, this involves two different models acting as the leader and the follower, with the follower's parameters influencing the leader's. Drawing inspiration from the dropout layer [Srivastava *et al.*, 2014] and mixture-of-experts [Zhou *et al.*, 2022], we implement this concept by designating a specific percentage of a single TSF model's parameters as leader parameters, with the remainder serving as follower parameters.

In the bilevel optimization formulation (5), both θ_p and θ_{dtw} are used during the model's forward process, yet each loss function updates only its corresponding parameters. This strategy of parameter separation within one model provides two key benefits. First, it renders our bilevel optimization framework model-agnostic. Unlike approaches that necessitate constructing a sub-follower model and integrating it within an existing leader model's architecture — which can demand alterations to the leader model and may not be viable for all model types — simply separating parameters within the existing model preserves its original architecture and enhances the framework's applicability and simplicity. Furthermore, this approach eliminates the need for an additional sub-model, thus maintaining the model's parameter size and enhancing parameter efficiency. We provide the pseudo-code of the parameter separation and the bilevel optimization process in Alg 1.

Algorithm 1 Bilevel Optimization Training Schema

```
Input series $X = \{\mathbf{x}^i\}_{i=1}^N$ and target series $Y = \{\mathbf{y}^i\}_{i=1}^N$;
 1: Initialize parameters $\theta$
 2: Designate parameters $\theta_p, \theta_{dtw}$
 3: while not converge do
 4:     for $\mathbf{x}^i \in X$, target $\mathbf{y}^i \in Y$ do
 5:         Forecast $\hat{\mathbf{y}}^i = f(\mathbf{x}^i; \theta)$
 6:         Update $\theta_{dtw}$ using loss function DTW
 7:     end for
 8: end while{Training of the follower parameter}
 9: while not converge do
10:     for $\mathbf{x}^i \in X$, horizon $\mathbf{y}^i \in Y$ do
11:         Forecast $\hat{\mathbf{y}}^i = f(\mathbf{x}^i; \theta)$
12:         Update $\theta_p$ using loss function $L_p$
13:     end for
14: end while{Training of the the leader parameter}
```

5 Experiments

5.1 Experiment Setting

Dataset

We evaluate the performance of our bilevel optimization framework on six widely-used datasets, including Traffic, Electricity, and four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2). These datasets have been extensively used for benchmarking purposes and are publicly available as referenced in [Wu *et al.*, 2021]. We chose the energy usage and traffic volume datasets from the benchmark because they have periodic peaks and troughs, making accurate predictions of them critical.

Baseline loss and evaluation metric

To validate the effectiveness of the bilevel loss, we use MSE, DTW, and MultiObj loss as baseline losses. To compare the effectiveness of these losses, we use the state-of-the-art TSF model, PatchTST [Nie *et al.*, 2022], as the base model. The PatchTST model trained with the losses mentioned above is evaluated using MSE, DTW, and a newly defined metric called PTMSE (Peak-Trough MSE), which we developed to measure peak-trough prediction accuracy. Detailed explanations of PTMSE are provided below.

PTMSE: Evaluation metric for peak-trough prediction

We introduce the Peak-Trough (PT) distance as a new metric to measure the accuracy of peak and trough predictions in time series. Let P_l and Q_l denote the sets of time indices within the forecasting horizon H where the values are local maxima (peaks) and local minima (troughs) within each predefined l-length PT unit window. The PT unit window size depends on the dataset's sampling rate, corresponding to the upper-level time unit. For example, for the benchmark ETTh dataset with an hourly sampling rate, the PT unit window size is 24 (one day). For a minute-based dataset, the window size is 60 (one hour). Each PT unit window contains one peak and one trough, so the number of time indices in P_l and Q_l equals the number of peaks and troughs in the forecasting horizon H. While a peak detection algorithm could be used

Loss		MSE			DTW			MultiObj			Bilevel		
Metric		MSE	DTW	PTMSE	MSE	DTW	PTMSE	MSE	DTW	PTMSE	MSE	DTW	PTMSE
Traffic	96	**0.131**	8.02	9.504	1.645	**7.84**	2.65	0.140	7.92	**2.603**	0.170	9.31	2.940
	192	**0.136**	16.80	9.749	2.593	16.41	4.357	0.152	**15.78**	3.548	0.160	17.02	**3.347**
	336	**0.139**	29.93	9.566	2.778	28.06	3.96	0.148	**25.69**	3.684	0.148	27.12	**3.460**
	720	**0.156**	66.43	9.638	3.313	63.34	3.69	0.158	**54.27**	**3.198**	0.158	60.62	3.212
Electricity	96	0.228	13.05	1.263	0.641	12.42	0.43	**0.218**	**11.53**	0.412	0.239	12.29	**0.038**
	192	**0.259**	27.55	1.229	1.142	27.31	0.70	0.266	**26.32**	**0.962**	0.276	26.54	1.004
	336	**0.297**	53.30	1.220	1.041	60.64	0.61	0.303	**50.01**	0.519	0.324	54.42	**0.471**
	720	**0.340**	121.4	1.245	1.110	150.4	1.19	0.365	**114.1**	**1.070**	0.365	118.2	1.101
ETTh1	96	**0.054**	6.09	0.159	0.070	**5.31**	0.13	0.059	5.68	**0.082**	0.058	5.86	**0.082**
	192	**0.070**	15.40	0.172	0.089	**11.49**	0.18	0.076	13.08	0.109	0.076	13.97	**0.085**
	336	**0.081**	29.43	0.161	0.103	**20.72**	0.22	0.091	22.12	0.137	0.091	26.18	**0.094**
	720	**0.086**	61.13	0.155	0.116	**43.60**	0.27	0.100	44.83	0.171	0.100	49.64	**0.109**
ETTh2	96	**0.128**	9.84	0.524	0.232	**8.45**	0.29	0.140	9.59	0.144	0.139	9.41	**0.141**
	192	**0.168**	23.64	0.517	0.308	**17.83**	0.45	0.181	20.10	0.298	0.179	21.55	**0.205**
	336	**0.184**	43.63	0.514	0.340	**31.22**	0.57	0.208	35.16	0.343	0.195	38.97	**0.200**
	720	**0.223**	106.9	0.534	0.401	**77.32**	0.85	0.292	79.57	0.687	0.258	93.56	**0.331**
ETTm1	96	**0.025**	3.28	0.195	0.033	**3.29**	0.05	0.030	3.50	0.049	0.030	3.58	**0.038**
	192	**0.039**	9.61	0.159	0.049	**8.85**	0.10	0.044	9.68	0.090	0.044	9.68	**0.060**
	336	**0.052**	21.80	0.093	0.067	**18.79**	0.16	0.058	18.89	0.130	0.058	21.22	**0.037**
	720	**0.074**	63.20	0.136	0.092	**47.69**	0.18	0.080	49.39	0.132	0.080	57.14	**0.029**
ETTm2	96	**0.064**	5.80	1.306	0.106	**5.68**	0.39	0.076	5.85	0.443	0.076	5.85	**0.432**
	192	**0.093**	15.89	1.233	0.163	**13.66**	0.49	0.105	14.60	**0.613**	0.105	15.15	0.751
	336	**0.120**	34.15	0.561	0.213	**28.21**	0.74	0.137	28.34	0.407	0.137	32.14	**0.174**
	720	**0.172**	101.1	0.829	0.318	**70.20**	0.74	0.187	72.72	0.592	0.187	82.53	**0.208**

Table 1: Univariate forecasting results with supervised PatchTST with different losses. The best performance in each loss is bolded. All of the experiments follow the same experimental setup with prediction length $H \in 96, 192, 336, 720$ and historical time series length $T = 336$

[Alves *et al.*, 2024], it may yield different labels depending on the dataset and hyperparameters and detect trivial peaks and troughs, which may not be suitable for forecasting tasks used as alerts.

We set a PT prediction window of ± indices around each true peak/trough to allow flexibility in time index differences. This flexibility is reasonable in our alert context, as predicting extreme values is important even with slight time deviations. The PT distance calculates the L_2 distance between the true peak/trough values and the maximum/minimum prediction values within this PT prediction window. The PTMSE for a single sample is defined as:

$$\text{PTMSE}_l = \frac{1}{|P_l|} \sum_{p \in P_l} L_2(\hat{y}_p, y_p) + \frac{1}{|Q_l|} \sum_{q \in Q_l} L_2(\hat{y}_q, y_q)$$

where T is the historical time series length T. This metric quantitatively evaluates a model's ability to forecast peaks and troughs in the data.

Implementation details

Like most TSF studies, we set a forecasting target length $H \in \{96, 192, 336, 720\}$. In the case of PatchTST, it follows the settings presented in the paper. In the original paper, PatchTST uses a historical time series length T=336, transforms this series to patches of length=16 with stride=8, and finally uses 42 historical time series patches as model input. Other detailed model architectures of PatchTST also follow the official code in PatchTST github. However, information about univariate forecasting is not provided for the Electricity and Traffic dataset, so we apply the multivariate setting by reducing only the number of features. The smoothing parameter γ of soft-DTW is set to 10^{-2} as proposed in previous work [Le Guen and Thome, 2019]. The soft-DTW loss is always positive with 'normalize=True', and a DTW loss weight=10^{-2} is applied to match the scale with MSE loss in both MultiObj and Bilevel approaches. In the Bilevel optimization framework, the leader and follower parameters are each set to 50%.

5.2 Quantitative results

Table 1 presents the univariate forecasting results for the PatchTST model trained with different loss functions (MSE, DTW, MultiObj, and Bilevel). As expected, each loss function performs best on its corresponding metric: MSE Loss provides the best MSE results, DTW Loss excels in DTW scores, and Bilevel Loss demonstrates strong performance in PTMSE, particularly for peak-trough accuracy. The Bilevel optimization framework minimizes MSE while achieving the best PTMSE accuracy in most cases, compared to baselines. Although MultiObj balances MSE and DTW, it shows relatively lower PTMSE performance than Bilevel Loss. This indicates that the Bilevel framework effectively achieves the primary TSF objectives while enabling peak-trough aware forecasting. Notably, despite promoting peak-trough awareness, DTW Loss alone fails to meet the fundamental TSF objectives, resulting in suboptimal peak-trough predictions.

5.3 Qualitative results

As shown in Figure 3, Each mse-pred (red) and bi-pred (green) is a prediction based on MSE loss and bi-level op-

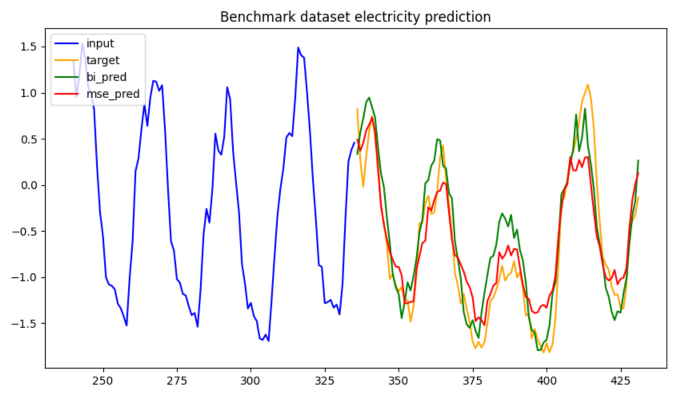

Figure 3: Comparison of PatchTST [Nie *et al.*, 2022] model predictions using different losses on an electricity benchmark dataset [Zhou *et al.*, 2021]. Blue shows historical data (model input), and yellow shows ground truth targets. Red and green represent the prediction based on MSE (red) and bilevel optimization (green).

timization loss, respectively. mse-pred has a lower MSE (0.0869) than bi-pred (green) (MSE: 0.0949), but when comparing PTMSE, bi-pred has a smaller PTMSE (0.1212) and better captures the peaks and troughs than mse-pred (0.1652). This visualization of predictions indicates that our bilevel optimization framework allows the model to make a sharp prediction better capturing peaks and troughs.

6 Conclusion and Future work

In this paper, we addressed the limitations of traditional TSF metrics in capturing sharp peaks and troughs. We proposed a bilevel optimization framework that combines L_p loss and DTW loss to enhance peak-trough prediction accuracy. Our approach maintains the primary TSF objectives while being model-agnostic and efficient. Evaluation results demonstrated the effectiveness of our framework in achieving superior peak-trough accuracy compared to baseline methods. We will focus on validating the efficacy of the bilevel optimization framework in a multivariate forecasting setting and across various TSF models.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2022-00143911, AI Excellence Global Innovative Leader Education Program)

References

[Abid and Zou, 2018] Abubakar Abid and James Y Zou. Learning a warping distance from unlabeled time series using sequence autoencoders. *Advances in neural information processing systems*, 31, 2018.

[Alves *et al.*, 2024] Emilly Pereira Alves, Felipe Alberto Barbosa Simão Ferreira, Francisco Madeiro, Paulo Salgado Gomes de Mattos Neto, and João Fausto Lorenzato de Oliveira. A hybrid multi-objective approach for time series forecasting considering peak instants. *Energy Reports*, 11:4537–4551, 2024.

[Andersen *et al.*, 2005] Torben G Andersen, Tim Bollerslev, Peter Christoffersen, and Francis X Diebold. Volatility forecasting, 2005.

[Bertsekas, 2014] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[Chatfield, 2000] Chris Chatfield. *Time-series forecasting*. CRC press, 2000.

[Colson *et al.*, 2007] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of operations research*, 153:235–256, 2007.

[Cuturi and Blondel, 2017] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, pages 894–903. PMLR, 2017.

[Esling and Agon, 2012] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):1–34, 2012.

[Hyndman and Fan, 2009] Rob J Hyndman and Shu Fan. Density forecasting for long-term peak electricity demand. *IEEE Transactions on Power Systems*, 25(2):1142–1153, 2009.

[Jahn, 1985] Johannes Jahn. *Scalarization in multi objective optimization*. Springer, 1985.

[Le Guen and Thome, 2019] Vincent Le Guen and Nicolas Thome. Shape and time distortion loss for training deep time series forecasting models. *Advances in neural information processing systems*, 32, 2019.

[Le Guen and Thome, 2020] Vincent Le Guen and Nicolas Thome. Probabilistic time series forecasting with shape and temporal diversity. *Advances in Neural Information Processing Systems*, 33:4427–4440, 2020.

[Lee *et al.*, 2022] Hyunwook Lee, Chunggi Lee, Hongkyu Lim, and Sungahn Ko. Tilde-q: A transformation invariant loss function for time-series forecasting. *arXiv preprint arXiv:2210.15050*, 2022.

[Mensch and Blondel, 2018] Arthur Mensch and Mathieu Blondel. Differentiable dynamic programming for structured prediction and attention. In *International Conference on Machine Learning*, pages 3462–3471. PMLR, 2018.

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

[Palshikar and others, 2009] Girish Palshikar et al. Simple algorithms for peak detection in time-series. In *Proc. 1st Int. Conf. advanced data analysis, business analytics and intelligence*, volume 122, 2009.

[Rajeswaran *et al.*, 2020] Aravind Rajeswaran, Igor Mordatch, and Vikash Kumar. A game theoretic framework for model based reinforcement learning. In *International conference on machine learning*, pages 7953–7963. PMLR, 2020.

[Sakoe and Chiba, 1978] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[Von Stackelberg, 2010] Heinrich Von Stackelberg. *Market structure and equilibrium*. Springer Science & Business Media, 2010.

[Wasserman, 2010] Larry Wasserman. *All of statistics : a concise course in statistical inference*. Springer, New York, 2010.

[Wu *et al.*, 2021] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

[Yin and Shang, 2016] Yi Yin and Pengjian Shang. Forecasting traffic time series with multivariate predicting method. *Applied Mathematics and Computation*, 291:266–278, 2016.

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

[Zhou *et al.*, 2022] Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.