ONLINE CONTINUAL LEARNING WITH FEEDFORWARD ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recently deep learning has been widely used in time-series prediction tasks. Although a trained deep neural network model typically performs well on the training set, performance drop significantly in a test set under slight distribution shifts. This challenge motivates the adoption of online adaptation algorithms to update the prediction models in real-time to improve the prediction performance. Existing online adaptation methods optimize the prediction model by feeding back the latest prediction error computed with respect to the latest observation. However, feedback based approach is prone to forgetting past information. In this work, we propose an online adaptation method with feedforward compensation, which uses critical data samples from a memory buffer, instead of the latest samples, to optimize the prediction model. We prove that the proposed feedforward approach has a smaller error bound than the feedback approach in slow time-varying systems. The experiments on several time-series prediction tasks show that the proposed feedforward adaptation outperforms conventional feedback adaptation by more than 10%. In addition, the proposed feedforward adaptation method is able to estimate an uncertainty bound of the prediction that is agnostic from specific optimizers, while existing feedback adaptation could not.

1 INTRODUCTION

Time-series prediction (or forecasting) has been widely studied in many fields, including control, energy management, and financial investment Box et al. (2015); Brockwell & Davis (2002). Among the research applications, acquiring future trends and tendencies of the time-series data is one of the most important subjects. With the emergence of deep learning, many deep neural network models have been proposed to solve this problem Lim & Zohren (2021), e.g., Recurrent Neural Networks Lai et al. (2018) and Temporal Convolutional Networks Bai et al. (2018). Inspired by the great success of Transformer in the NLP and CV community Vaswani et al. (2017); Dosovitskiy et al. (2020), Transformer-style methods have been introduced to capture long-term dependencies in time series prediction tasks Zhou et al. (2021). Benefiting from the self-attention mechanism, Transformers obtain a great advantage in modeling long-term dependencies for sequential data Brown et al. (2020). Although a trained Transformer model (or other big deep neural network models) typically performs well on the training set, performance can significantly drop in a slightly different test domain or under a slightly different data distribution Popel & Bojar (2018); Si et al. (2019).

In practical time-series prediction problems, there are often significant distributional discrepancies between the offline training set and the real-time testing set. These differences may be attributed to multiple factors. In some cases, it is too expensive to collect large unbiased training datasets, e.g., for weather prediction or medical time-series prediction. In other cases, it may be difficult to obtain the training instances from a specific domain. For example, in human-robot collaboration, it is hard to collect data from all potential future users. In these cases, adaptation techniques are applied to deal with the distribution mismatch between offline training and real-time testing Blum (1998); Abuduweili & Liu (2021). Besides, some tasks require the system to adapt itself after every observation. For example, in human-robot collaboration, the robot needs to continually adapt its behaviors to different users. In these scenarios, complete retraining is infeasible, and online adaptation techniques are often embraced Abuduweili et al. (2019).

Online adaptation continually learns from real-time streaming data, which belongs to online continual learning. In online adaptation, a prediction model receives sequential observations. After each observation, the online optimization algorithm (e.g. SGD) may update the prediction model according to the prediction loss measured by the observed data. The goal of online adaptation is to improve prediction accuracy in subsequent rounds. Online adaptation is currently applied to many researches like time-series prediction Pastor et al. (2011); Abuduweili & Liu (2020), image recognition Lee & Kriegman (2005), and machine translation Martínez-Gómez et al. (2012). In this paper, we mainly focus on time-series prediction tasks, but the proposed methods also can be used for other online adaptation (or online learning) tasks.

Most existing online adaptation approaches are based on feedback compensation Abuduweili & Liu (2021), analogous to feedback control. In feedback adaptation, a prediction model only utilizes the latest receiving data. After observing a new sample, the online optimization algorithm updates the prediction model according to the prediction loss measured between the prediction and the latest received ground truth. However, this kind of passive and feedback compensation is not efficient.

In this work, we propose to enable feedforward compensation in online adaptation to maximize information extraction from existing data, especially those that are more critical. A critical sample means which is more helpful to reduce the objective (loss) of the model when training. In the proposed **feedforward adaptation**, we will not only have forgetting as is done in conventional online adaptation Paleologu et al. (2008), but also enable recalling to compensate for potential shortsighted behaviors due to forgetting. There is a balance between forgetting and recalling. On the one hand, to rapidly learn the new function value in a time-varying system, we need to forget some of the old data. On the other hand, too much forgetting may cause unstable and incorrect predictions when we encounter a similar pattern with historical data. To achieve the balance between forgetting and recalling, we design a novel mechanism for feedforward compensation using a memory buffer similar to the functionality of the Hippocampus in the human brain Barron (2021). We will maintain the memory buffer by storing recent T-steps observations (or hidden features) of samples. When the prediction model experiences similar observations, it will pull the corresponding data (critical sample) from the memory buffer to enhance learning. For example, in human behavior prediction tasks, a human subject may exhibit similar behavior patterns on different days. These would be extremely difficult to discover if we only passively learn from the most recent data like conventional online adaptation but can be identified using the feedforward adaptation methods with memory buffer. We can also use the related information between the current sample and critical samples to measure the uncertainty bound to the current prediction. The uncertainty-bound estimation can be incorporated into any optimizers.

Our main contributions can be summarized in the following points.

- To the best of our knowledge, we are the first to propose a feedforward compensation in online adaptation. Which is theoretically reasonable and empirically applicable.
- Including feedforward and feedback adaptation method, we provide a general online adaptation framework.
- The proposed feedforward adaptation method provides the uncertainty bounds of the prediction which is agnostic from specific optimizers.
- We conduct extensive experiments and show that the proposed feedforward adaptation is superior to conventional feedback adaptation. Furthermore, the feedforward adaptation is model agnostic, which can be combined with any deep neural network to further exploit its learning ability.

2 Related Works

Time series prediction. Time-series prediction historically has very diverse applications across various domains Lim & Zohren (2021). These applications trigger many methods for time-series prediction. Traditional methods have focused on domain knowledge — such as autoregressive (AR) Box et al. (2015), exponential smoothing models Gardner Jr (1985). Modern machine learning methods provide a purely data-driven manner without domain knowledge Ahmed et al. (2010). In recent years, deep neural networks becomes the dominant approach, inspired by notable achievements of deep learning in computer vision Dosovitskiy et al. (2020) and natural language processing Vaswani et al. (2017). Because Transformer-style models have the potential to capture long-term depen-

dencies Brown et al. (2020), which is becoming more and more popular in time-series prediction tasks. Most Transformer-style models focus on long-term time series forecasting problem, include Informer Zhou et al. (2021), Autoformer Wu et al. (2021), Pyraformer [18] Liu et al. (2021).

Continual Learning, also known as Lifelong learning, aims at providing incrementally updated knowledge in an ever-changing environment. From the data statistics perspective, continual learning studies the problem of learning from non-i.i.d data, with the goal of preserving and extending the acquired knowledge. Conventional continual learning assumes the new data arrive one task at a time, and the data distribution for each task is stationary De Lange et al. (2021). Which is also called offline continual learning. Different from the batch training on offline continual learning, **Online Continual Learning** focuses on the more realistic problem, where data arrive one tiny batch at a time and previously seen batches from the current or the previous tasks are not accessible Mai et al. (2022). Therefore, online continual learning is required to efficiently learn from a single sample from the online data stream in a single step. Recent works of continual learning can be roughly divided into four families: dynamic architectures, regularization-based, rehearsal, and generative replay approach Lesort et al. (2020). Among these methods, Rehearsal-based methods are more related to our work, which use memory replays to enhance the knowledge from the previous data or processes such as iCaRL Rebuffi et al. (2017), GEM Lopez-Paz & Ranzato (2017), which use a small subset of the previous dataset to balance the knowledge distribution for different tasks. However, most of these methods focus on offline continual learning.

Online adaptation continually learning from streaming data, which is belong to the online continual learning family. However, unlike works that focus on avoiding catastrophic forgetting in offline continual learning Kirkpatrick et al. (2017), online adaptation focuses on the ability to rapidly learn and adapt in the presence of non-stationarity. Online adaptation is a special case of online continual learning in that the model is always offline trained on an offline dataset (which can be very small) before online adaptation Cheng et al. (2019). Most existing online adaptation approaches are based on feedback compensation with SGD (Stochastic Gradient Descent) or EKF (Extended Kalman Filter) optimizers Bhasin et al. (2012); Abuduweili & Liu (2021).

3 Online Adaptation Framework

This section discusses the general online adaptation framework for time-series prediction tasks.

3.1 Time-series prediction

The time series prediction problem is to make inferences on future time-series data given the past and current observations. We consider a multi-step prediction problem: using recent I steps' observations to predict future O steps' data. Assume the transition model is composed of a feature extractor (or Encoder) E and a predictor (or decoder) f. At time step t, the input to the model is $X_t = [x_{t-I+1}, x_{t-I+2}, \cdots, x_t]$, which denotes the stack of I-step recent observations. The output of the model is $Y_{t+1} = [y_{t+1}, y_{t+2}, \cdots, y_{t+O}]$, which denotes the stack of O-step future predictions. The observations x_t, y_t are vectors which may contain trajectory or feature, and in some cases (e.g. univariate prediction) $x_t = y_t$.

The transition model for time series prediction can be formulated as

$$Z_t = E(X_t), \tag{1}$$

$$Y_{t+1} = f_t(Z_t), \tag{2}$$

where Z_t is a hidden feature representation of input X_t. Feature extractor E does not change overtime, while predictor f_t changes over time. Let f_t denotes the ground-truth predictor, that generates ground-truth input-output tuples (X_t, Y_{t+1}). In online adaptation, we use the parameterized model (e.g. Neural Networks) $\hat{f}(\theta_t, Z_t)$ with learnable parameters θ_t to estimate the ground-truth predictor $f_t(Z_t)$. This paper assumes that the encoder E is fixed in online adaptation, which could be trained offline before online adaptation or be a non-parametric feature extractor[1].

[1]In a special case, the feature extractor can be an identity mapping $E(X) = X$, when we adapt the end-to-end neural network model in online adaptation

3.2 FEEDBACK AND FEEDFORWARD ADAPTATION

Online adaptation of the model parameters θ_t is important to make the inference framework robust to time-varying and heterogeneous behaviors in the prediction. Online adaptation also can be called *adaptable prediction*, since it makes an inference concurring with updating model parameters. Online adaptation explores local overfitting to minimize the prediction error. At time step t, the prediction error e_{t+1} is to be minimized. The optimization objective is shown below

$$\mathcal{L}_{err} = \min_{\theta_t} e_{t+1} = \min_{\theta_t} \|Y_{t+1} - \hat{f}(\theta_t, Z_t)\|_p, \tag{3}$$

where $f_t(Z_t) = Y_{t+1} = [y_{t+1}, y_{t+2}, \cdots, y_{t+O}]$ is the ground truth observation (to be received in the future) and $\hat{f}(\theta_t, Z_t)) = \hat{Y}_{t+1} = [\hat{y}_{t+1}, \hat{y}_{t+2}, \cdots, \hat{y}_{t+O}]$ is the predicted outcome using the learned model parameter θ_t. If the ground-truth model parameter changes slowly over time, the estimated model parameter that minimizes the prediction error in the future can be approximated by the estimated parameter that minimizes the fitting error in the past. The adaptation objective can be in any ℓ_p norm. This paper considers ℓ_2 norm.

In conventional (feedback) online adaptation, the estimate of the model parameter θ_t is updated iteratively when new observations are received. Then a new prediction is made using the new estimate. In the next time step, the estimate will be updated again given a new observation, and the process repeats. It is worth noting that the observation we received at time t is y_t. The other terms in Y_t remain unknown. This paper focused on adaptation methods using one-step ahead observation. It is also possible to conduct online adaptation with multi-step ahead observations as discussed in Abuduweili & Liu (2021).

Algorithm 1 General Online Adaptation Framework (Adaptable Prediction)

Input: Initial predictor $f(\theta_0, :)$ with parameters θ_0, Feature Extractor E, Optimizer $\mathcal{O}(:, :, :)$
Output: Sequence of predictions $\{\hat{Y}_{t+1}\}_{t=1}^{T}$

1: **for** $t = 1, 2, \cdots, T$ **do**
2: Receive the ground truth observation values x_t, y_t
3: Find the critical input-output pairs (Z_s, y_{s+1}) for $1 \leq s < t$, where $Z_s = E(X_s)$
4: Adaptation: $\theta_t = \mathcal{O}(\theta_{t-1}, \hat{y}_{s+1}, y_{s+1})$, where $\hat{Y}_{s+1} = [\hat{y}_{s+1}, \cdots, \hat{y}_{s+O}] = f(\theta_{t-1}, Z_s)$
5: Prediction: $\hat{Y}_{t+1} = [\hat{y}_{t+1}, \cdots, \hat{y}_{t+O}] = f(\theta_t, Z_t)$, where $Z_t = E(X_t)$
6: **end for**

We provide a general online adaptation framework by incorporating conventional feedback with the proposed feedforward adaptation as shown in algorithm 1. At time step t, after receiving the current observations (x_t, y_t), we find the critical input-output pair (Z_s, y_{s+1}). The critical pair was used to adjust the parameters of the prediction model by an online optimizer (e.g. SGD). Then we obtain the current prediction result with the adapted model. The main difference between feedback and feedforward adaptation lies in the selection of the critical pair.

- **Feedback adaptation**. In conventional feedback adaptation, the critical input-output pairs are composed by the latest observations $s = t - 1$. It is called feedback adaptation, because at step t we adapt the model by measuring the error of the previous step prediction $f(\theta_{t-1}, Z_{t-1})$ and adjust the model to get the desired output value of the previous step. Note that, the goal of the online adaptation is to make the current prediction $f(\theta_t, Z_t)$ more desirable. If the current observation x_t and the previous observation x_{t-1} are not very related, the feedback adapted model, which provides desired output value of the previous step, may not provide a desired output of the current step.
- **Feedforward adaptation**. In the proposed feedforward adaptation, the critical input-output pairs are the most related samples to the current observation (see details in section 4). Feedforward adaptation calculates the required value of the most related sample and then adjusts the model to get desired output of the most related sample. The feedforward adapted model, which provides desired output value of the most related sample, may provide a desired output of the current sample.

In algorithm 1, an online optimizer was used to adjust the model with the selected critical samples: $\theta_t = \mathcal{O}(\theta_{t-1}, \hat{y}_{s+1}, y_{s+1})$. SGD and EKF are two widely used optimizers for online adaptation, please check the details in .

4 FEEDFORWARD ADAPTATION APPROACH

This section derives a feedforward adaptation method from some basic assumptions and provides an error bound and uncertainty estimation of the proposed feedforward adaptation method. Then this section discusses the effectiveness of the proposed approach in a special case.

4.1 ERROR BOUND FOR GENERAL ONLINE ADAPTATION

The theoretical analysis of the paper is based on two basic conditions about the local smoothness property of ground-truth predictors f_t. The series of ground-truth prediction functions within recent L steps $(f_{t-L+1}, f_{t-L+2}, \cdots, f_t)$ has K-Lipschitz continuous and bounded time-varying properties.

K-Lipschitz continuity condition. For a time-step t and $\forall s \in [t-L, t-1]$, we have local K Lipschitzness for ground-truth prediction function f_t and recent L steps input data:

$$\|f_t(Z_t) - f_t(Z_s)\| \leq K\|Z_t - Z_s\|, \tag{4}$$

where K is the bound (real number) for the change of the value of the function over input space. Intuitively, a Lipschitz continuous function is limited in how fast function value can change over input space. It is proven that every function that has bounded first derivatives is Lipschitz continuous Sohrab (2003). K-Lipschitz continuity is a very common condition in time-series prediction tasks because most of the time-series transition model has bounded first derivatives.

Similar to Eq. (4), we assume our parameterized function $\hat{f}_t$ has Lipschitz continuity. This assumption is reasonable because neural network models should have bounded first derivative with proper training.

$$\|\hat{f}(\theta_t, Z_s) - \hat{f}(\theta_t, Z_t)\| \leq \hat{K}\|Z_t - Z_s\|. \tag{5}$$

δ time-varying condition. For a time-step t and $\forall s \in [t-L, t-1]$, assume the ground-truth prediction functions f_t has bounded changes within recent L steps compared to f_s on input data Z_s.

$$\|f_t(Z_s) - f_s(Z_s)\| \leq \delta, \tag{6}$$

where δ is the bound (real number) for change of the value of the function sequences over time on a fixed input. A bounded δ time-varying condition is a very common condition in time-series tasks, because we obtain similar future time-series signals with the same input in most cases.

Bound for prediction error of online adaptation. If the sereis of ground-truth functions f_t within recent L steps follows Eq. (5) and (6), the (prior) prediction error e_{t+1} has the following inequality. Please check the proof on appendix B.1.

$$e_{t+1} \leq K\|Z_t - Z_s\| + \delta + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| + \|\hat{f}(\theta_t, Z_s) - \hat{f}(\theta_t, Z_t)\| \quad \text{(Eq. (26))}\| \tag{7}$$

$$\leq (K + \hat{K})\|Z_t - Z_s\| + \delta + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| \tag{8}$$

4.2 FEEDFORWARD ADAPTATION

As discussed in section 3.2, our goal is to minimize the prediction error Eq. (3). Because of the lack of ground-truth value in the current steps, we cannot directly minimize the prediction error. Conventional online adaptation methods approximate Eq. (3) with minimizing the fitting error in the past. However, minimizing the bound of the prediction error provides better results, especially in the worst-case scenarios. Under the following assumption, the bound of the prediction error Eq. (29) is to be minimized in the proposed feedforward adaptation method,

Assumption: For the δ time-varying function f_t, assume δ is independent to time or its supremum is a small value: $\frac{\partial \delta}{\partial t} = 0 \text{ or } \sup(\delta) \approx 0$

The above assumption is reasonable, because the time-varying factor δ is very small in many cases, and the major changes in function value are caused by the change of inputs. In other cases, δ is caused by random noises, then which is time-independent. Under the above assumption, we can ignore the term δ in minimization. Then the optimization objective for the error bound is shown below:

$$\mathcal{L}_{Bound} = \min_{\theta_t, s \in [t-L, t-1]} (K + \hat{K})\|Z_t - Z_s\| + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| \tag{9}$$

However, we still can not directly minimize the above objective, because we do not know $K, \hat{K}$. Then a simplification is applied to the above objective to make it applicable. We change the joint minimization over s and θ_t to a bi-level optimization which first minimizes the first term of objective over s, then minimizes the second term of objective over θ_t. Thus, the simplified objective function is shown below:

$$\mathcal{L}_{final} = \min_{\theta_t} \|Y_{s^\star+1} - \hat{f}(\theta_t, Z_{s^\star})\| \tag{10}$$

$$\text{s.t.} \quad s^\star = \underset{s\in[t-L,t-1]}{\arg\min} \|Z_t - Z_s\| \tag{11}$$

In a summary, Feedforward adaptation selects the most similar samples as critical pair $(Z_{s^\star}, y_{s^\star})$ by Eq. (11). Then using the critical pair to optimize the prediction model by Eq. (10). Feedback adaptation is equivalent to set $s^\star = t-1$ and optimize the prediction model by Eq. (10). In the error bound Eq. (9), the last term is the posterior (fitting) error, which can be very small and which is independent of the feedback or feedforward criteria. For the comparison between feedback and feedforward, the major error is caused by the first term $\|Z_t - Z_s\|$. Since feedforward adaptation can find more similar samples, it results in a smaller error bound than feedback adaptation.

4.3 Uncertainty Estimation

Feedforward adaptation could provide uncertainty estimation of the prediction by the error bound Eq. (28). We know all the items in Eq. (28), except K and δ. Here we use estimation of $\tilde{K}_t$ and $\tilde{\delta}$ to approximate real K and δ. We use confidence factor $\sigma \in (0, 1]$ to decay the error bound as uncertainty estimation. The uncertainty estimation U_{t+1} for prediction $\hat{Y}_{t+1}$ is shown below:

$$\hat{U}_{t+1} = \sigma \cdot \Big(\tilde{K}_t\|Z_t - Z_s\| + \tilde{\delta} + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| + \|\hat{f}(\theta_t, Z_s) - \hat{f}(\theta_t, Z_t)\|\Big) \tag{12}$$

Where $s = t-1$ for feedback adaptation and $s = s^\star$ (Eq. (11)) for feedforward adaptation. The confidence factor σ is a predefined hyperparameter, e.g. $\sigma = 0.9$ for 90% confidence of uncertainty. For $\tilde{\delta}$, we can set it as a small value, e.g. $\tilde{\delta} = 0.001$. For $\tilde{K}_t$, we can iteratively update its value according to the estimated uncertainty $\hat{U}_t$ and real error e_t. If the previous uncertainty estimation is much larger than the real error then we shrink the $\tilde{K}_t$ value. If the previous uncertainty estimation is much smaller than the real error then we enlarge the $\tilde{K}_t$ value. The $\tilde{K}_t$ estimation criteria are shown below.

$$\tilde{K}_t = \begin{cases} \tilde{K}_{t-1}/1.5, & \text{if } \hat{U}_t \geq 2e_t \\ \tilde{K}_{t-1} \times 1.5, & \text{if } \hat{U}_t \leq 0.5e_t \\ \tilde{K}_{t-1}, & \text{otherwise} \end{cases} \tag{13}$$

The overall algorithm is shown in algorithm 2.

4.4 When feedforward is better than feedback?

As described in section 4.2, under the assumption 4.2, feedforward adaptation achieves a smaller error bound than feedback adaptation. In this section, we analyze the exact condition for δ in a special case. Consider a function f_t which can be separated to time-invariant function $g(z_t)$ and linearly time-dependent part αt. Input z_t is a random variable from uniform distribution $\mathcal{U}(0, 1)$.

$$z_t \sim \mathcal{U}(0, 1) \tag{14}$$

$$f_t(z_t) = g(z_t) + \alpha t \tag{15}$$

For the above functions, we have the following lemma. Please check the proof in appendix B.1.1.

Lemma 1. If $\alpha \leq \frac{4-\sqrt{12}}{6}(K + \hat{K})$ for Eq. (31), with proper buffer size $L = \sqrt{\frac{2(K+\hat{K})}{\alpha}} - 1$, feedforward adaptation achieves smaller error bound than feedback adaptation.

Actually, we can extend the above conclusion to a slightly larger set of functions. α can be time-dependent, but the supremum of it needs to be small.

Algorithm 2 Online Adaptation with Feedforward Compensation

Input: Initial predictor $f(\theta_0, :)$ with parameters θ_0, Feature Extractor E, Optimizer $\mathcal{O}(:,:,:)$
Input: Empty L-size buffer B ;
Output: Sequence of predictions $\{\hat{Y}_{t+1}\}_{t=1}^{T}$ and estimated uncertainty $\{\hat{U}_{t+1}\}_{t=1}^{T}$

```
for t = 1, 2, ··· , T do
    Receive the ground truth observation values x_t, y_t; Construct input X_t = [x_{t-I+1}, ··· , x_t]
    Find the critical (similar) input-output pairs (Z_{s*}, y_{s*+1}) from buffer B by Eq. (11)
    Feedforward Adaptation: θ_t = O(θ̃_t, ŷ_{s*+1}, y_{s*+1}), where [ŷ_{s*+1}, ··· , ŷ_{s+O}] = f(θ̃_t, Z_{s*})
    Prediction: Ŷ_{t+1} = [ŷ_{t+1}, ··· , ŷ_{t+O}] = f(θ_t, Z_t), where Z_t = E(X_t)
    Uncertainty Û_{t+1} Estimation by Eq. (12) and Eq. (13)
    Add current data to buffer: B.append(Z_t, y_t)
    if size(B) > L then
        B ← keep_more_recent_samples (B, L)
    end if
end for
```

Lemma 2. Given a function f_t, let $\alpha_t = \frac{\partial f_t}{\partial t}$. Let B and $\hat{B}$ denotes the Lipschitz constant for ground-truth f_t and learnable approximator $\hat{f}(\theta)$. If $\sup(\alpha_t) \leq \frac{4-\sqrt{12}}{6}(K + \hat{K})$, with proper buffer size, feedforward adaptation achieves smaller error bound than feedback adaptation.

Thus feedforward adaptation can be used to learn time-invariant functions and slow-varying linear time-variant functions. The slow-varying linear time-variant systems are very common in the world. For example, electronic measurement devices have a slow time-dependent shift on the results, because of aging and wear.

The special case of discussion about the time-invariant polynomial is shown in appendix B.1.2.

5 Experiments

In this section, we present empirical results on both synthetic and real-world datasets.

5.1 Synthetic Experiments

To evaluate the conclusion in section 4.4, we consider the following time-varying function.

$$z_t \sim \mathcal{U}(0, 1) \tag{16}$$

$$y_{t+1} = f_t(z_t) = \sin z_t + \alpha t \tag{17}$$

We have the following property (please check details in appendix C.2).

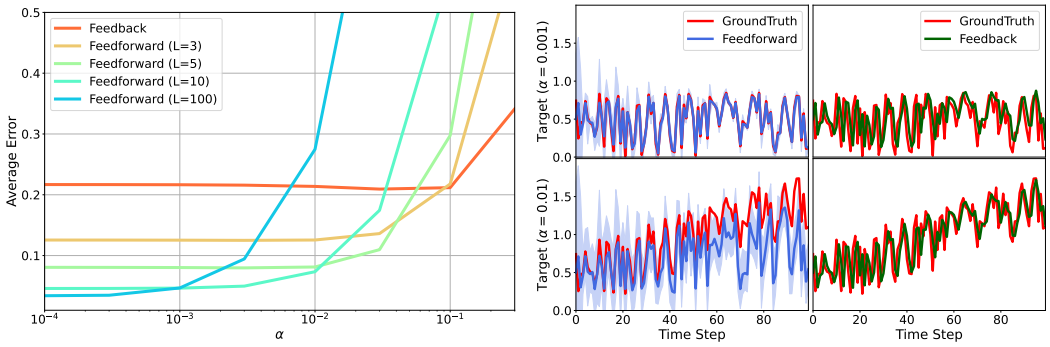
$$\mathbb{E}[B_e^{fb}] = \frac{2}{3} + \alpha, \quad \mathbb{E}[B_e^{ff}] \leq \frac{5}{4(L+1)} + \alpha\frac{L+1}{2} \tag{18}$$

$$\mathbb{E}[B_e^{ff}] \leq \sqrt{\frac{5}{2}\alpha} \text{ with } L = \sqrt{\frac{5}{2\alpha}} - 1 \tag{19}$$

$$\alpha^\star \approx 0.089(K + \hat{K}) \approx 0.11, \quad L^\star = \sqrt{\frac{5}{2\alpha^\star}} - 1 \approx 3 \tag{20}$$

Figure 1a shows the average error of feedback and feedforward adaptation under different time-varying factor α. Note that, feedforward adaptation with $L = 1$ is the same as feedback adaptation. As can be seen, for a smaller time-varying factor α feedforward adaptation with a larger buffer size L achieves better performance, which is consistent with Eq. (60). We also find that feedforward adaptation with buffer size $L = 3$ performs better than feedback adaptation if $\alpha < 0.1$. The intersection point $\alpha \approx 0.1$ is consistent with Eq. (61). If α is larger, the performance of the feedforward adaptation drops. Figure 1b illustrate some reasons for it. In Fig. 1b, the first row shows the prediction results for $\alpha = 10^{-3}$ target function of feedforward adaptation with $L = 100$ (left subfigure) and feedback adaptation (right subfigure). As can be seen, feedforward adaptation can learn the

ground truth more precisely. The second row in Fig. 1b shows the prediction results for $\alpha = 10^{-2}$ target function. For this case, prediction with feedforward adaptation has a significant shift from the ground truth, even if it still is able to learn some input-dependent details. Compared to feedback adaptation, feedforward adaptation is more focused on the input-dependent part of the ground truth function and pays less attention to the time-varying shift of the function. These empirical results validate the conclusions in section 4.4.



(a) Average error

(b) Prediction of feedforward adaptation

Figure 1: (a) Average error of feedback and feedforward adaptation with different buffer size L and time-varying factor α. (b) Prediction of feedforward adaptation with buffer size $L = 100$ on time-varing factor $\alpha = 10^{-3}$ and $\alpha = 10^{-2}$.

5.2 Experiments on real wrold data

In this section, we evaluate the proposed feedforward adaptation on four real-world benchmarks, including ETT (Electricity Transformer Temperature) Zhou et al. (2021), Exchange-Rate Lai et al. (2018), ILI (Influenza-like Illness) [2], THOR human motion trajectory dataset Rudenko et al. (2020). We include two models as our learnable prediction function $\hat{f}(\theta, :)$: Informer Zhou et al. (2021) and simple MLP (multi-layer perception). Please check details of Dataset, backbone model and hyperparameters in appendix C.3.

Experimental Design. First we offline train models (Informer and MLP) on a train set. After training, just like in real-world applications, we will incrementally receive the data point from the test set. At each time step, we conduct an online adaptation to optimize the model with some available previous observations by feedback or feedforward compensation. Then the prediction results come from the updated model. We evaluate the prediction results with mean squared error (MSE) and mean absolute error (MAE). In the experiments, we only adapt the decoder of the model and make the encoder fixed. We conduct experiments on SGD and EKF as optimizers in adaptation.

Baselines. We compare the proposed method with three baselines. 1) **w/o adapt** directly conduct prediction without adaptation. Which is a lower bound for adaptation methods. 2) **Feedback adaptation** is the most important baseline to us. 3) **Random adaptation** is a method that selects the critical pair from the recent L steps with random sampling. A comparison between the proposed feedforward adaptation and the random adaptation would evaluate the critical pair selection strategy within the same memory buffer.

Results. The detailed results of the experiment are shown in table 1. The last row Gain denotes the performance gain of the proposed feedforward adaptation over feedback adaptation. As can be seen, the proposed feedforward adaptation outperforms feedback adaptation by 11% in terms of average MSE and 8% in terms of average MAE. Feedforward adaptation achieves the best results on all four datasets. The performance gain on ILI and Exchange dataset is the most obvious, and the gain on the THOR dataset is the smallest. It is reasonable because ILI and Exchange dataset is a slow time-varying dataset, the future value is mostly dependent on the current input. While the THOR dataset is more heterogeneous. The behavior of one agent might be totally different from other agents, so when incrementally learning behavior from different agents directly causes the time-varying property of the ground-truth function.

[2] `https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html`

We found another interesting result by comparison between Informer and MLP on table 1. Without adaptation, the simple MLP outperforms the Informer on average. This is caused by the overfitting of the Informer on the training set. However, by online adaptation, especially with feedforward adaptation, the performance of Informer is greatly increased. For example, the performance of Informer on the ETTh1 dataset increased from 0.211 to 0.102 by 51% and outperform the MLP model. That shows Informer has greater representation ability but may perform poorly on the test set, because of overfitting or other problems. In this case, it is embarrassing to directly use the big Transformer-like models on time-series prediction. This phenomenon emphasizes the importance of online adaptation in real-world time-series prediction. With online adaptation, especially with the proposed feedforward adaptation on the slow time-varying system, the great representation ability of the big models is exploited.

Table 1: Performance comparison between the proposed feedforward adaptation method and other baselines. Avg denotes the average results of two models (MLP and Informer) with two optimizers (SGD and EKF) under the specific adaptation strategy. In the last row, Gain denotes the performance gain of feedforward adaptation over feedback adaptation in terms of average result.

Method \Dataset			ETTh1		Exchange		ILI		THOR	
Adaptation	Model	Optimizer	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
w/o Adaptation	MLP	SGD	0.195	0.371	0.549	0.540	4.348	1.413	0.135	0.117
		EKF	0.195	0.371	0.549	0.540	4.348	1.413	0.135	0.117
	Informer	SGD	0.211	0.389	1.128	0.858	4.942	1.531	0.137	0.171
		EKF	0.211	0.389	1.128	0.858	4.942	1.531	0.137	0.171
	Avg		0.203	0.380	0.839	0.699	4.645	1.472	0.136	0.144
Random Adaptation	MLP	SGD	0.145	0.312	0.365	0.445	3.835	1.307	0.132	0.148
		EKF	0.174	0.335	0.404	0.468	3.318	1.262	0.123	0.149
	Informer	SGD	0.131	0.133	0.497	0.553	3.766	1.280	0.204	0.271
		EKF	0.191	0.352	0.847	0.684	3.573	1.328	0.277	0.313
	Avg		0.163	0.283	0.528	0.537	3.623	1.294	0.184	0.220
Feedback Adaptation	MLP	SGD	0.153	0.317	0.347	0.442	3.868	1.334	0.112	0.119
		EKF	0.129	0.288	0.356	0.459	3.873	1.341	0.107	0.106
	Informer	SGD	0.116	0.269	0.326	0.461	4.383	1.411	0.134	0.173
		EKF	0.109	0.249	0.359	0.462	4.320	1.401	0.124	0.166
	Avg		0.127	0.288	0.347	0.456	4.111	1.411	0.119	0.141
Feed Forward Adaptation	MLP	SGD	0.128	0.286	0.349	0.432	3.041	1.193	0.102	0.108
		EKF	0.123	0.280	0.352	0.450	2.847	1.154	0.101	0.103
	Informer	SGD	0.109	0.259	0.276	0.399	3.363	1.312	0.125	0.156
		EKF	0.102	0.248	0.254	0.327	3.395	1.244	0.134	0.178
	Avg		0.115	0.268	0.307	0.402	3.161	1.225	0.115	0.136
	Gain		9.45%	6.9%	11.5%	11.8%	23.1%	13.1%	3.3%	3.5%

6 CONCLUSION

This paper studied an effective feedforward adaptation algorithm for time-series prediction tasks. Firstly we propose the general framework for online adaptation which includes feedback and feedforward adaptation. Then we propose the feedforward adaptation algorithm by selecting the most similar critical samples for optimization. We prove that, in a time-invariant or slow time-varying system, the feedforward adaptation has a smaller error bound than conventional feedback adaptation. In the experiments, we empirically validate the effectiveness of the proposed algorithm both on synthetic and real-world data. In the end, we show that online adaptation can greatly improve the performance of Informer or other models.

REFERENCES

Abulikemu Abuduweili and Changliu Liu. Robust online model adaptation by extended kalman filter with exponential moving average and dynamic multi-epoch strategy. In *Learning for Dynamics and Control*, pp. 65–74. PMLR, 2020.

Abulikemu Abuduweili and Changliu Liu. Robust nonlinear adaptation algorithms for multitask prediction networks. *International Journal of Adaptive Control and Signal Processing*, 35(3): 314–341, 2021.

Abulikemu Abuduweili, Siyan Li, and Changliu Liu. Adaptable human intention and trajectory prediction for human-robot collaboration. *arXiv preprint arXiv:1909.05089*, 2019.

Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric reviews*, 29(5-6): 594–621, 2010.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Helen C Barron. Neural inhibition for continual learning and memory. *Current opinion in neurobiology*, 67:85–94, 2021.

Shubhendu Bhasin, Rushikesh Kamalapurkar, Huyen T Dinh, and Warren E Dixon. Robust identification-based state derivative estimation for nonlinear systems. *IEEE Transactions on Automatic Control*, 58(1):187–192, 2012.

Avrim Blum. On-line algorithms in machine learning. *Online algorithms*, pp. 306–325, 1998.

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

Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. Springer, 2002.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Yujiao Cheng, Weiye Zhao, Changliu Liu, and Masayoshi Tomizuka. Human motion prediction using semi-adaptable neural networks. In *2019 American Control Conference (ACC)*, pp. 4884–4890, 2019. doi: 10.23919/ACC.2019.8814980.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

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

Everette S Gardner Jr. Exponential smoothing: The state of the art. *Journal of forecasting*, 4(1): 1–28, 1985.

Andrew H Jazwinski. *Stochastic processes and filtering theory*. Courier Corporation, 2007.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. Online learning with kernels. *IEEE transactions on signal processing*, 52(8):2165–2176, 2004.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Kuang-Chih Lee and David Kriegman. Online learning of probabilistic appearance manifolds for video-based recognition and tracking. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pp. 852–859. IEEE, 2005.

Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information fusion*, 58:52–68, 2020.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

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

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

Pascual Martínez-Gómez, Germán Sanchis-Trilles, and Francisco Casacuberta. Online adaptation strategies for statistical machine translation in post-editing scenarios. *Pattern Recognition*, 45(9): 3193–3203, 2012.

Constantin Paleologu, Jacob Benesty, and Silviu Ciochina. A robust variable forgetting factor recursive least-squares algorithm for system identification. *IEEE Signal Processing Letters*, 15: 597–600, 2008.

Peter Pastor, Ludovic Righetti, Mrinal Kalakrishnan, and Stefan Schaal. Online movement adaptation based on previous sensor experiences. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 365–371. IEEE, 2011.

Martin Popel and Ondřej Bojar. Training tips for the transformer model. *arXiv preprint arXiv:1804.00247*, 2018.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Dennis W. Ruck, Steven K. Rogers, Matthew Kabrisky, Peter S. Maybeck, and Mark E. Oxley. Comparative analysis of backpropagation and the extended kalman filter for training multilayer perceptrons. *IEEE transactions on pattern analysis & machine intelligence*, 14(06):686–691, 1992.

Andrey Rudenko, Tomasz P Kucner, Chittaranjan S Swaminathan, Ravi T Chadalavada, Kai O Arras, and Achim J Lilienthal. Thör: Human-robot navigation data collection and accurate motion trajectories dataset. *IEEE Robotics and Automation Letters*, 5(2):676–682, 2020.

Wenwen Si, Tianhao Wei, and Changliu Liu. Agen: Adaptable generative prediction networks for autonomous driving. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 281–286. IEEE, 2019.

Houshang H Sohrab. *Basic real analysis*, volume 231. Springer, 2003.

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

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.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

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, pp. 11106–11115, 2021.

A ONLINE OPTIMIZERS

In algorithm 1, an online optimizer was used to adjust the model with the selected critical samples: $\theta_t = \mathcal{O}(\theta_{t-1}, \hat{y}_{s+1}, y_{s+1})$. Unlike offline machine learning, online optimizers need to handle iterative input-output pairs with single batch size. We assume our objective function (loss function) has suitable smoothness properties (e.g. differentiable and locally Lipschitz continuous).

Stochastic Gradient Descent (SGD) is one of the widely used optimizers in online adaptation Kivinen et al. (2004). SGD is memory efficient and stable. The update formula of the SGD is shown below:

$$\theta_t = \theta_{t-1} - \eta \left(\frac{\partial}{\partial \theta} \|y_{s+1} - \hat{y}_{s+1}(\theta)\|^2 \right) |_{\theta=\theta_{t-1}}, \tag{21}$$

where η is learning rate.

Extended Kalman Filter (EKF) is another strong optimizer in online adaptation Jazwinski (2007). In the original EKF, the object being estimated is the state value of a dynamic system. In online adaptation, we can apply the EKF approach to adapt model parameters by regarding model parameters as system states. The EKF approach has a faster convergence rate than SGD, but it is very memory-exhaustive. When optimizing small neural networks, EKF approach has been demonstrated to be superior to the SGD-based algorithms Ruck et al. (1992); Abuduweili & Liu (2021). If the adapted model size is not big, EKF is a strong optimizer for online adaptation. The update formula of the EKF is shown below:

$$H_t = \frac{\partial \hat{y}_{s+1}(\theta)}{\partial \theta} |_{\theta=\theta_{t-1}} \tag{22}$$

$$G_t = P_{t-1} \cdot H_t^{\mathsf{T}} \cdot (H_t \cdot P_{t-1} \cdot H_t^{\mathsf{T}} + \lambda I)^{-1} \tag{23}$$

$$\theta_t = \theta_{t-1} + G_t \cdot (y_{s+1} - \hat{y}_{s+1}) \tag{24}$$

$$P_t = \lambda^{-1}(P_{t-1} - G_t \cdot H_t \cdot P_{t-1}) \tag{25}$$

where $\lambda \in (0, 1]$ is a forgetting factor, H_t is the gradient matrix, G_t is the Kalman gain, and P_t is a matrix representing the uncertainty in the estimates of parameters. The initial value of the matrix P_t can be set as to be proportional to the identity matrix $P_0 = p_0 \cdot \mathrm{diag}(1, 1, \cdots, 1)$ for $p_0 > 0$.

B SOME PROOF ON THE SECTION 4

B.1 BOUND FOR PREDICTION ERROR

Bound of ground-truth difference. If the sereis of ground-truth functions f_t within recent L steps follows K-Lipschitz continuity and δ time-varying conditions, then the ground-truth value Y_{t+1} and Y_{s+1} has the following property:

$$\|Y_{t+1} - Y_{s+1}\| \leq K\|Z_t - Z_s\| + \delta \tag{26}$$

The prove is shown below:

$$\begin{aligned}
\|Y_{t+1} - Y_{s+1}\| &= \|f_t(Z_t) - f_s(Z_s)\| \\
&= \|f_t(Z_t) - f_t(Z_s) + f_t(Z_s) - f_s(Z_s)\| \\
&\leq \|f_t(Z_t) - f_t(Z_s)\| + \|f_t(Z_s) - f_s(Z_s)\| \quad \text{(triangle inequality)} \\
&\leq K\|Z_t - Z_s\| + \|f_t(Z_s) - f_s(Z_s)\| \quad \text{(K Lipschitzness)} \\
&\leq K\|Z_t - Z_s\| + \delta \quad (\delta \text{ time} - \text{varying})
\end{aligned} \tag{27}$$

Bound for prediction error of online adaptation. For time step t, the (prior) prediction error e_{t+1} has the following inequality:

$$\begin{aligned}
e_{t+1} &= \|Y_{t+1} - \hat{Y}_{t+1}\| = \|Y_{t+1} - \hat{f}(\theta_t, Z_t)\| \\
&= \|Y_{t+1} - Y_{s+1} + Y_{s+1} - \hat{f}(\theta_t, Z_s) + \hat{f}(\theta_t, Z_s) - \hat{f}(\theta_t, Z_t)\| \\
&\leq \|Y_{t+1} - Y_{s+1}\| + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| + \|\hat{f}(\theta_t, Z_s) - \hat{f}(\theta_t, Z_t)\| \quad \text{(triangle inequality)} \\
&\leq K\|Z_t - Z_s\| + \delta + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| + \|\hat{f}(\theta_t, Z_s) - \hat{f}(\theta_t, Z_t)\| \quad \text{(Eq. (26))}\|
\end{aligned} \tag{28}$$

The first two terms come from the difference between ground-truth $Y_{t+1} - Y_{s+1}$, the third term is a posterior error for input-output tuple (Z_s, Y_{s+1}), and the last term is the difference between two predictions. Combining Eq. (28) with Eq. (5), we obtain error bound for general online adaptation is shown below:

$$e_{t+1} \leq (K + \hat{K})\|Z_t - Z_s\| + \delta + \|Y_{s+1} - \hat{f}(\theta_t, Z_s)\| \tag{29}$$

B.1.1 Discussion of error bound

$$z_t \sim \mathcal{U}(0, 1) \tag{30}$$

$$f_t(z_t) = g(z_t) + \alpha t \tag{31}$$

We use neural network $f(\hat{\theta}, :)$ to learn the above function. Let K is a Lipschitz constant for f_t (it is also equal to Lipschitz constant for function g), and $\hat{K}$ is a Lipschitz constant for $\hat{f}$. The δ time varying condition becomes $\|f_t(z_s) - f_s(z_s)\| = \alpha\|t - s\|$, and $\delta = \alpha\|t - s\|$. Then we have an error bound from Eq. (29):

$$e_{t+1} \leq (K + \hat{K})\|z_t - z_s\| + \alpha\|t - s\| + \|y_{s+1} - \hat{f}(\theta_t, z_s)\| \tag{32}$$

Now we consider the comparison of the error bound between feedforward and feedback adaptation. Because the last term of the above equation is a fitting error, which can be very small with proper optimization. In addition, the fitting error is independent of feedback or feedforward adaptation. Then we ignore the fitting error in comparison. We now compare the expectation of the bound:

$$\mathbb{E}[B_e] = (K + \hat{K})\mathbb{E}[\|z_t - z_s\|] + \alpha\mathbb{E}[|t - s|] \tag{33}$$

In feedback adaptation, $s = t - 1$. The input z_t and z_{t-1} are independent random variables from $\mathcal{U}(0, 1)$. The expectation of the distance between these two independent variables is $\frac{1}{3}$. We have the expectation for the error bound of feedback adaptation is:

$$\mathbb{E}[B_e^{fb}] = \frac{K + \hat{K}}{3} + \alpha \tag{34}$$

In feedforward adaptation, $s = s^\star$ 11. $z_\star$ is a most similar sample to z_t. $\mathbb{E}[\|z_t - z_s\|]$ means the average minimum distance between z_t and other samples. If $t \leq l$, the average minimum distance for t samples is no greater than $\frac{1}{t}$. If $t > l$, the average minimum distance for $L + 1$ samples is no greater than $\frac{1}{L+1}$. Similarly, $\mathbb{E}[\|t - s\|]$ is an average distance between t indices (if $t \leq l$), which is $\frac{t}{2}$, or average distance from $L + 1$ indices (if $t \geq l$) , which is $\frac{L+1}{2}$. We have the expectation for the error bound of feedforward adaptation is:

$$\mathbb{E}[B_e^{ff}] \leq (K + \hat{K}) \cdot \max(\frac{1}{t}, \frac{1}{L+1}) + \alpha \cdot \min(\frac{t}{2}, \frac{L+1}{2}) \tag{35}$$

We mainly consider the final performance $t > L$, then the above equation can be simplified as:

$$\mathbb{E}[B_e^{ff}] \leq \frac{K + \hat{K}}{L+1} + \alpha\frac{L+1}{2} \tag{36}$$

With $L = \sqrt{\frac{2(K+\hat{K})}{\alpha}} - 1$ we could achieve the smallest error bound for feedforward adaptation:

$$\mathbb{E}[B_e^{ff}] \leq \sqrt{2\alpha(K + \hat{K})} \tag{37}$$

To find the condition for $\mathbb{E}[B_e^{ff}] \leq \mathbb{E}[B_e^{fb}]$, we solve the following inequality

$$\sqrt{2\alpha(K + \hat{K})} \leq \frac{K + \hat{K}}{3} + \alpha \quad \text{and} \quad L = \sqrt{\frac{2(K + \hat{K})}{\alpha}} - 1 \geq 1 \tag{38}$$

We obtain a δ condition for feedforward adaptation:

$$\alpha \leq \frac{4 - \sqrt{12}}{6}(K + \hat{K}) \approx 0.089(K + \hat{K}) \tag{39}$$

Then we have the lamma 1 in section 4.4.

B.1.2 SPECIAL CASE: TIME INVARIANT POLYNOMIAL

In this subsection, we analyze the effectiveness of feedforward adaptation as a special case: input z_t is a random variable sampled from uniform distribution $\mathcal{U}(0,1)$, and ground-truth predictor $f_t(z_t)$ is a time-invariant polynomial function. Assume our parameterized prediction model is a linear projection:

$$z_t \sim \mathcal{U}(0,1) \tag{40}$$

$$y_{t+1} = f_t(z_t) = \sum_{i=1}^{N} W_i z_t^i \tag{41}$$

$$\hat{y}_t = \hat{f}(V_t, z_t) = V_t z_t \tag{42}$$

At time step t, we uses the critical pair (z_s, y_{s+1}) to optimize the linear prediction model.

$$V_t^\star = \arg\min \|V_t z_s - y_{s+1}\| = \frac{y_{s+1}}{z_s} = \frac{\sum_{i=1}^{N} W_i z_s^i}{z_s} = \sum_{i=0}^{N-1} W_{i+1} z_s^i \tag{43}$$

Then the absolute prediction error (l_1 norm) is shown below:

$$\begin{aligned} e_{t+1} &= |y_{t+1} - V_t^\star z_t| = |\sum_{i=1}^{N} W_i z_t^i - \sum_{i=0}^{N-1} W_{i+1} z_s^i \cdot z_t| \\ &= |z_t| \cdot |\sum_{i=0}^{N-1} W_{i+1}(z_t^i - z_s^i)| = z_t \cdot |\sum_{i=0}^{N-1} W_{i+1}(\sum_{j=0}^{i-1} z_t^{i-j-1} z_s^j)(z_t - z_s)| \\ &= |z_t| \cdot |z_t - z_s| \cdot |\sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} z_t^{i-j-1} z_s^j| \end{aligned} \tag{44}$$

We consider the expectation of the prediction error. For $z_t \sim \mathcal{U}(0,1)$, we have $\mathbb{E}(z_t^n) = \frac{1}{n+1}$. For simplicity, assume $W_i > 0$. The expectation of the prediction error is shown below.

$$\mathbb{E}(e_{t+1}) = \mathbb{E}(z_t) \cdot \mathbb{E}(|z_t - z_s|) \cdot \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \mathbb{E}[z_t^{i-j-1} z_s^j] \tag{45}$$

In feedback adaptation $x_s = x_{t-1}$ and $\mathbb{E}(|z_t - z_s|) = \frac{1}{3}$ (section 4.4). Then we derive an expectation of the prediction error for feedback adaptation:

$$\begin{aligned} \mathbb{E}(e_{t+1}^{fb}) &= \frac{1}{2} \cdot \frac{1}{3} \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \mathbb{E}[z_t^{i-j-1} x_s^j] = \frac{1}{6} \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \mathbb{E}[z_t^{i-j-1}] \cdot \mathbb{E}[z_s^j] \\ &= \frac{1}{6} \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \frac{1}{i-j} \frac{1}{j+1} \\ &\geq \frac{1}{6} \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \frac{2}{(i+1)^2} = \frac{1}{3} \sum_{i=0}^{N-1} W_{i+1} \frac{i}{(i+1)^2} \\ &\geq \frac{1}{3} \frac{N-1}{N^2} \sum_{i=1}^{N-1} W_{i+1} \end{aligned} \tag{46}$$

As can be seen, the expected error of feedback adaptation cannot converge. $\mathbb{E}(e_{t+1}^{fb})$ in each step t is almost random and lower bounded by an non-zero constant.

In feedforward adaptation x_s is a most close point for x_t, and $\mathbb{E}(|z_t - z_s|) = \max(\frac{1}{t}, \frac{1}{L+1})$ (section 4.4) . Then we derive an expectation of the prediction error for feedforward adaptation:

$$\begin{aligned}
\mathbb{E}(e_{t+1}^{ff}) &= \frac{1}{2}\max(\frac{1}{t}, \frac{1}{L+1}) \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \mathbb{E}[z_t^{i-j-1} z_s^j] \\
&\leq \frac{1}{2}\max(\frac{1}{t}, \frac{1}{L+1}) \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \mathbb{E}[z_t^{i-j-1} z_t^j] \qquad (47)\\
&\leq \frac{1}{2}\max(\frac{1}{t}, \frac{1}{L+1}) \sum_{i=0}^{N-1} W_{i+1} \sum_{j=0}^{i-1} \frac{1}{i} \\
&\leq \frac{1}{2}\max(\frac{1}{t}, \frac{1}{L+1}) \sum_{i=1}^{N-1} W_{i+1} \qquad (48)
\end{aligned}$$

$$\lim_{L\to\infty, t\to\infty} \mathbb{E}(e_{t+1}^{ff}) \leq \lim_{L\to\infty, t\to\infty} \frac{1}{2}\max(\frac{1}{t}, \frac{1}{L+1}) \sum_{i=1}^{N-1} W_{i+1} = 0 \qquad (49)$$

As can be seen, the expectation of the prediction error of feedforward compensation converges to zero with a large buffer size L. Then we provide that, feedforward adaptation is superior to feedback adaptation in estimating polynomial function with random inputs. The experimental evaluation of the above formula is shown in appendix C.1.

C Details of Experiments

C.1 Synthetic Experiments: Time-invariant Quadratic Function

To demonstrate the superiority of feedforward adaptation over feedback adaptation in time-invariant function, as described in **??**, we consider a quadratic function with random input. Our parameterized prediction model is the linear projection:

$$z_t \sim \mathcal{U}(0, 1) \qquad (50)$$

$$y_{t+1} = f_t(z_t) = z_t^2 \qquad (51)$$

$$\hat{y}_t = \hat{f}(V_t, z_t) = V_t z_t \qquad (52)$$

Set the buffer size $L = 99$. According to Eq. (46) and Eq. (48), the expected error for feedback and feedforward adaptation are:

$$\mathbb{E}(e_{t+1}^{fb}) = \frac{1}{6}, \qquad \mathbb{E}(e_{t+1}^{ff}) \leq \frac{1}{2}\max(\frac{1}{t}, \frac{1}{L+1}) = \max(\frac{1}{2t}, \frac{1}{200}) \qquad (53)$$

Fig. 2a shows the prediction error for feedback and feedforward adaptation. As can be seen, the error curve for feedback adaptation is not converged. While the error curve for feedforward adaptation converges by the trend of $\frac{1}{2t}$. The experimental results are consistent with Eq. (53). Fig. 2a shows the prediction results for feedforward adaptation. The dashed blue region denotes the estimated uncertainty. In the figure, the ground truth is always within the estimated uncertainty, which shows the validity of the uncertainty estimation.

C.2 Synthetic Experiments: Time-varying function

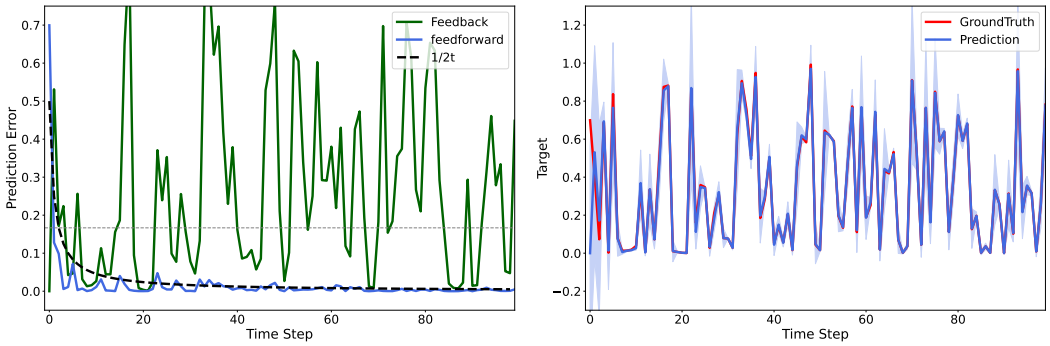
To evaluate the conclusion in section 4.4, we consider the following time-varying function.

$$z_t \sim \mathcal{U}(0, 1) \qquad (54)$$

$$y_{t+1} = f_t(z_t) = \sin z_t + \alpha t \qquad (55)$$

And our parameterized prediction model is a one-layer perception with a Sigmoid activation function.

$$\hat{y}_t = S(V_t z_t) + b_t = \frac{1}{1 + e^{-V_t z_t}} + b_t \qquad (56)$$



(a) Error Curve (b) Prediction of feedforward adaptation

Figure 2: Error curve and prediction results on quadratic function.

The Lipschitz constant of a function is equal to upper bound of its gradient. Then we have:

$$K = \sup |\frac{\partial f_t}{\partial z_t}| = \sup |\cos(z_t)| = 1 \tag{57}$$

$$\hat{K} = \sup(|\frac{\partial \hat{f}}{\partial z_t}|) = \sup |V_t \cdot S(V_t x_t) \cdot (1 - S(V_t x_t))| = 0.25 \sup |V_t| \tag{58}$$

We use SGD as an optimizer in feedback and feedforward adaptation. During training, we keep the $\|V_t\|$ bounded, i.e. $\|V_t\| \leq 1$, then $\hat{K} = \frac{1}{4}$. We use Eq. (34) and Eq. (36) to estimate the error bound for feedback and feedforward adaptation. Then use Eq. (39) to calculate the critical α for comparison.

$$\mathbb{E}[B_e^{fb}] = \frac{2}{3} + \alpha, \quad \mathbb{E}[B_e^{ff}] \leq \frac{5}{4(L+1)} + \alpha \frac{L+1}{2} \tag{59}$$

$$\mathbb{E}[B_e^{ff}] \leq \sqrt{\frac{5}{2}\alpha} \text{ with } L = \sqrt{\frac{5}{2\alpha}} - 1 \tag{60}$$

$$\alpha^\star \approx 0.089(K + \hat{K}) \approx 0.11, \quad L^\star = \sqrt{\frac{5}{2\alpha^\star}} - 1 \approx 3 \tag{61}$$

C.3 Experimental design for real-world experiment

Dataset.

- **ETT** Zhou et al. (2021) dataset contains the data collected from electricity transformers, including load and oil temperature that are recorded every 15 minutes between July 2016 and July 2018. Which consists of two hourly-level datasets (ETTh) and two 15-minute-level datasets (ETTm). In our experiments, we used the first hourly-level datasets ETTh1 as an univariance prediction task.
- **Exchange-Rate** Lai et al. (2018) records the daily exchange rates of eight countries from 1990 to 2016.
- **ILI** [3] describes the ratio of patients seen with ILI and the total number of patients. Which includes the weekly recorded influenza-like illness (ILI) patients data from the Centers for Disease Control and Prevention of the United States between 2002 and 2021.
- **THOR** Rudenko et al. (2020) is a public dataset of human motion trajectories, recorded in a controlled indoor experiment. Which includes the motion trajectories with diverse and accurate social human motion data in a shared indoor environment. In our experiments, we use No. $2 \sim 6$ agent's trajectory as a train set, No. $7 \sim 8$ as a validation set, and No. $9 \sim 10$ agent's trajectory as a test set.

Backbone models. We include two models as our learnable prediction function $\hat{f}(\theta, :)$: Informer Zhou et al. (2021) and simple MLP (multi-layer perception).

[3] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

- **Informer** Zhou et al. (2021) is a widely used transformer-based time-series prediction model. Which extends Transformer with KL-divergence based ProbSparse attention.
- **MLP** is a simple but robust baseline for time-series prediction. Our MLP consists of 2 layers. The first layer can be considered as Encoder $Z_t = W \cdot X_t$. After the encoder, MLP has a layer normalization, activation function and a final linear projection $Y_{t+1} = V \cdot \mathrm{Relu}(\mathrm{LayerNorm}(Z_t))$. The layer normalization and the final projection can be served as a decoder. Note that, we did not flatten the input for MLP, the expression $Z_t = W \cdot X_t$ is a linear layer along the temporal axis.

Hyperparameters. For offline training, we follow the strategy in Zeng et al. (2022). In adaptation, we set the learning rate of SGD is $\eta = 0.1$, and set the EKF hyperparameters as $p_0 = 0.1, \lambda = 1$. Buffer size for feedforward adaptation is $L = 1000$. For ETT dataset, we use $I = 192$ step recent observations to predict future $O = 192$ step output data. For the Exchange-Rate dataset, we use $I = 96$ step recent observations to predict future $O = 192$ step output data. For the ILI dataset, we use $I = 36$ step recent observations to predict future $O = 36$ step output data. For the THOR dataset, we use $I = 20$ step recent observations to predict future $O = 20$ step output data.