LOST IN THE NON-CONVEX LOSS LANDSCAPE: HOW TO FINE-TUNE THE LARGE TIME SERIES MODEL?

Anonymous authors
Paper under double-blind review

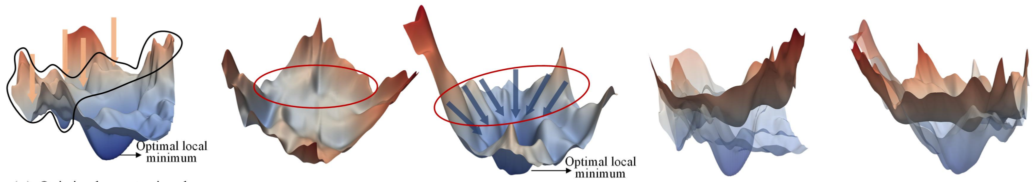
ABSTRACT

Recently, large time series models (LTSM) have become popular and important because they exhibit characteristics similar to large language models, such as flexible context length, scalability, and task generality, outperforming the advanced task-specific models in the domain. However, existing research indicates that the pre-trained LTSM can show a poor non-convex loss landscape (indicating poor trainability). Hence, directly fine-tuning pre-trained LTSM shows overfitting, which leads to poor fine-tuning performance, even worse than training from scratch on the downstream datasets. This severely diminishes the value of the pre-trained LTSM. To address this, we propose a new fine-tuning method called *Smoothed Full Fine-tuning* (SFF). Specifically, before fine-tuning, we first construct an auxiliary LTSM with a smooth loss landscape (indicating good trainability) through random initialization. Second, we utilize it to smooth the loss landscape of the pre-trained LTSM through linear interpolation between their weights. As a result, the smoothed LTSM acquires good trainability while retaining good pre-training knowledge, thereby achieving better performance when fine-tuned on the downstream dataset. We also explain why SFF is effective from the perspective of optimization theory: interpolation perturbs sharp minima without obviously harming originally flat regions, thereby aiding sharp minima escape to better and smoother basins. Extensive experiments on popular datasets show that our method indeed improves the performance of eight popular LTSMs, e.g., Timer, TimesFM, MOMENT, UniTS, MOIRAI, Chronos, TTMs, and Sundial, in different downstream tasks. Our codes are available at the link: `https://anonymous.4open.science/r/SFF-0014`.

1 INTRODUCTION

Large models developed through the generative pre-training transformer (GPT) have exhibited several advanced capabilities not found in smaller models: flexible context length, the generalization ability to fit multiple domains, the versatility to handle various scenarios and tasks, and the scalability where performance improves with the increase in the scale of parameters and pre-training corpora. In this context, large time series models (LTSM), e.g., Timer Liu et al. (2024), TimesFM Das et al. (2024), and MOMENT Goswami et al. (2024), are proposed to introduce the similar power of GPT into time series analysis and improve overall performance in forecasting Box et al. (2015), interpolation Friedman (1962), and anomaly detection Ren et al. (2019) tasks. After pretraining on the large-scale time series dataset, the LTSM can better capture universal features such as trend, amplitude, frequencies, and phases Goswami et al. (2024), thereby benefiting the downstream tasks.

However, theoretically, the current study shows that large-scale training may cause models to converge to sharp minima Keskar et al. (2016) characterized by a non-convex loss landscape Li et al. (2018), which in turn leads to optimization difficulties during fine-tuning. Experimentally, in Figure 1, we visualize the loss landscape of the pretrained LTSM on the downstream datasets, and it shows severe local protrusions (the area enclosed by the black curve in Figure 1), which corresponds to the above theoretical findings. Existing research also shows that such a steep and *non-convex (non-smooth)* loss landscape indicates poor trainability of a model and can lead the model to fall into poor local minimums that exist between the protrusions (e.g., orange arrows in Figure 1(a)), making the model more prone to severe overfitting and resulting in poorer generalization Li et al. (2018). We further examine the training and test losses of directly full fine-tuning the LTSM and found that the training loss is always the lowest, while the test loss is even higher than that of training from scratch. The



(a) Original pre-trained LTSM (b) Random Initialization (c) Utilizing (b) to smooth (a) (d) Placing (b) and semi-transparent (a) in the same coordinate system (e) Placing (b) and semi-transparent (c) in the same coordinate system

Figure 1: Loss landscape comparisons based on the LTSM Timer and exchange rate dataset. Smoother is better. The unsmooth loss landscape shows a non-convex structure and indicates poorer trainability Li et al. (2018), e.g., Figure 1(a). More cases are shown in Appendix Figure 7 and Figure 8.

results are shown in Appendix Figure 9 due to limited space. This is consistent with the characteristics of overfitting and corresponds to our analysis above.

Intuitively, the occurrence of the non-convex loss landscape can be caused by overfitting during the pre-training phase. For example, learning specific features of the pre-training data rather than general features, or falling into sharp minima Keskar et al. (2016). Due to the non-convex loss landscape, no matter which current fine-tuning method is used, such as *Full Fine-tuning (FF)*, *Linear Probing (LP)*, or *Linear Probing then Full Fine-tuning (LPFF)* Kumar et al., a good fine-tuning performance cannot be achieved, because they cannot smooth the non-convex loss landscape. This hinders the pre-trained LTSM from optimizing to the better local minimum during fine-tuning (e.g., dark blue area at the bottom of the loss landscape in Figure 1(a)). Therefore, an effective measure to mitigate the negative impact of the "non-convex loss landscape" on fine-tuning for downstream tasks. Fortunately, we also empirically find that the LTSM initialized randomly generally has a smoother loss landscape, as shown in Figure 1(b). This naturally raises a question: *Can we leverage the smooth landscape in Figure 1(b) to help the pre-trained one in Figure 1(a) achieve better convex structure (smoothed loss landscape) and thus improve its trainability, while still retaining the pre-trained knowledge?*

Based on these insights, we propose Smoothed Full Fine-tuning (SFF) to better exploit the pretrained knowledge of LTSMs for improving their fine-tuning performance. The method consists of two key steps. **First**, we construct an auxiliary LTSM by random initialization. Unlike the pretrained model, this auxiliary one has a smoother and more convex loss landscape (Figure 1(b)), which makes it more trainable. However, it lacks pretrained knowledge, as illustrated in Figure 1(d): the minimum of the randomly initialized model is much higher than that of the pretrained one, explaining why the pretrained model can perform zero-shot prediction. **Second**, we smooth the pretrained model's loss landscape by linearly interpolating its weights with those of the auxiliary LTSM. The resulting smoothed LTSM inherits the pretrained knowledge of the original model while gaining the improved trainability of the auxiliary one. As shown in Figure 1(e), the minimum loss of the smoothed model remains much lower than that of the randomly initialized model, confirming that pretrained knowledge is well preserved. We provide further empirical evidence in Section 5.4.

Overall, smoothing reduces severe protrusions in the pretrained loss landscape (Figure 1(c)), making it more convex and facilitating convergence. Gradient descent is then more likely to reach better local optima (as illustrated by the dark blue arrows), thereby improving fine-tuning stability and effectiveness. The proposed fine-tuning strategy only requires linear interpolation of model parameters before fine-tuning, without increasing memory and computation overhead during fine-tuning. We also explain why SFF is effective from the theory of deep learning optimization: interpolation perturbs sharp minima without obviously harming originally flat regions, thereby aiding the escape of sharp minima to better and smoother basins. Details can be found in section 3.1.

In summary, our contributions are as follows:

- We reveal a key finding that pretrained LTSM may suffer from overfitting during pretraining, exhibiting a poor convex structure in the loss landscape and showing lower trainability. Consequently, the fine-tuning performance in the downstream tasks is limited.
- We are the first to propose a weight-interpolation based fine-tuning strategy called *Smoothed Full Fine-tuning (SFF)* to mitigate the overfitting issue of pretrained LTSM. SFF smooths the loss landscape of the pre-trained LTSM through linear interpolation of model weights

between the pre-trained one (with **good** pre-trained knowledge but **poor** trainability) and a randomly initialized one (with **poor** pre-trained knowledge but **good** trainability). Smoothed LTSM achieves improved trainability while retaining good pre-trained knowledge, thereby facilitating better convergence during fine-tuning. We also explain SFF's effectiveness from an optimization perspective (section 3.1). **SFF does not incur additional memory overhead or time complexity**, and we provide a new insight for fine-tuning large models.

- We have validated the effectiveness of our method on time series forecasting (TSF) and anomaly detection tasks. Our method outperforms the popular *Full Fine-tuning (FF)*, *Linear Probing (LP)*, or *Linear Probing then Full Fine-tuning (LPFF)* strategies. Our method improves the performance of eight popular LTSMs with diverse architectures (encoder-only, decoder-only, encoder-decoder, and MLP only) and model sizes (3.8GB to 3MB).

2 Related works

Popular fine-tuning approaches include full fine-tuning, linear probing, and linear probing first and then full fine-tuning (LP-FF) (Kumar et al.). We have placed the related work about fine-tuning, optimization strategies, and time series foundation models in the appendix A.3 due to limited space.

Difference from weight interpolation in existing works. To the best of our knowledge, weight interpolation hasn't been explored for fine-tuning based on loss landscape theory. Although weight averaging and interpolation (Vlaar & Frankle, 2022) have been studied in model merging (Wortsman et al., 2022) and continual learning (Kozal et al., 2024), these works don't target the core challenge we identify in LSTMs, and our work is fundamentally different from them in the following aspects:

(1) Different goals. Existing interpolation methods (Wortsman et al., 2022; Kozal et al., 2024) are primarily designed for model ensembling—e.g., interpolating among multiple **well-trained models** to improve generalization or mitigate catastrophic forgetting. **In contrast, our method leverages interpolation to *smooth the loss landscape of a single pretrained model* by a randomly initialized model, thereby making it more trainable during fine-tuning.**

(2) Different pipelines. Previous works typically use the interpolated model directly for downstream tasks without further training. In our case, the pretrained model begins with a steep, irregular loss landscape. After interpolation smooths this landscape, **we proceed with additional fine-tuning** to utilize the smoothing effect for better performance.

(3) New theoretical analysis. Our method is built upon a key conceptual contrast: the flat, smooth loss landscape of a randomly initialized model versus the steep, irregular landscape of a pretrained one. We formalize this contrast through theoretical analysis and proof, which further shows—also theoretically—why our interpolation strategy can effectively exploit this difference to enhance fine-tuning performance.

3 Smoothing the loss landscape of the pre-trained LTSM for fine-tuning

We propose the *smoothed fine-tuning* strategy to boost the performance of fine-tuning various pre-trained LTSMs, and the overview of the proposed *smoothed fine-tuning* is shown in Figure 2.

3.1 Motivation and Theoretical Analysis

In this section, we explain why Smoothed Full Fine-tuning (SFF) is effective from the theory of deep learning optimization. Specifically, the region of the loss landscape where the model weights are located can be divided into flat and sharp areas Li et al. (2018). Flat regions imply better generalization because the model is more tolerant to weight changes within flat regions Keskar et al. (2016); Hochreiter & Schmidhuber (1997); Foret et al. (2020). This is also why our SFF can make linear interpolation between randomly initialized weights and the pre-trained model's weights for a smoother landscape and better fine-tuning effect. Doing so does not significantly affect the state of the weights in the flat region due to their high tolerance towards the weight perturbation. **For weights in non-flat regions, SFF's effect is similar to adding momentum (similar to the concept of**

momentum in the Adam Kingma & Ba (2014) optimizer) through random weight interpolation, allowing them to escape the non-flat regions for a smoother one and a better fine-tuning effect.

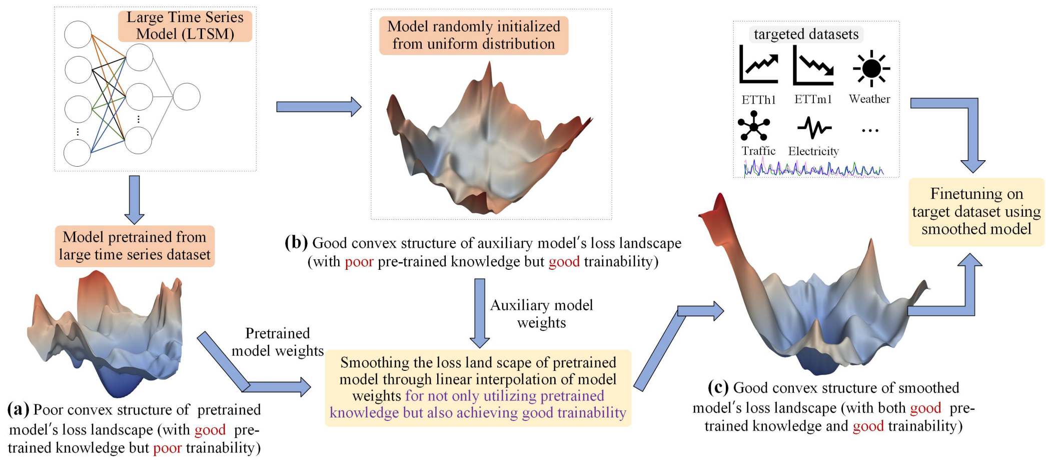


Figure 2: *Smoothed Full Fine-tuning* (SFF). By linearly interpolating the pretrained and randomly initialized LTSMs, we obtain a version that preserves pretrained knowledge while enjoying a smoother and more trainable loss landscape for better fine-tuning effects.

Next, we provide a more rigorous theoretical derivation to prove the effectiveness of SFF.

3.1.1 Influence of Interpolation on Sharp and Flat Loss Landscape

Given an MSE loss function $\mathcal{L}(\Theta)$ and the Θ denotes model parameters. To obtain a minimum $\mathcal{L}(\Theta)$, the corresponding parameters Θ^* can be a sharp minimum or a flat minimum. Inspired by (Keskar et al., 2016), we use the Hessian matrix to formally characterize the sharpness and flatness of the loss landscape in the following analysis. By analyzing the maximum eigenvalue $\lambda_{\text{max}}(\cdot)$ of the Hessian matrix $H = \nabla^2\mathcal{L}(\Theta^*)$ ($H \in \mathbb{R}^{d\times d}$ where d denotes feature dimension), we can obtain the definitions of the sharp and flat minimum as follows:

Theorem 1 (Sharp minimum)**.** *The Hessian $\nabla^2\mathcal{L}(\Theta^*)$ has large eigenvalues (i.e., $\lambda_{max}(\nabla^2\mathcal{L}(\Theta^*)) \gg \tau$ where $\tau > 0$ is a threshold), meaning steep loss landscape and small parameter perturbations lead to large loss increases.*

Theorem 2 (Flat minimum)**.** *The Hessian $\nabla^2\mathcal{L}(\Theta^*)$ has small eigenvalues (i.e., $\lambda_{max}(\nabla^2\mathcal{L}(\Theta^*)) \leq \tau$ where $\tau > 0$), meaning flat loss landscape and the loss is robust to parameter perturbations.*

Proof details for the above Theorems are shown in the Appendix section A.2.

Smoothing (perturbing) sharp minima. SFF interpolation strategy defines smoothed parameters as $\Theta_3 = \alpha\Theta_1^* + (1-\alpha)\Theta_2$, where Θ_1^* (pre-trained LTSM) are easier to lie in a sharp minimum (Figure 1(a)) due to large-scale pretraining (Keskar et al., 2016). In contrast, Θ_2 (randomly initialized) lies in a flat region, as shown in Figure 1(b), and **we will further demonstrate in the next section why mainstream Kaiming or Xavier initializations yield a flat loss landscape**. After pre-training the LTSM, for the sharp minimum points Θ_1^*, its largest eigenvalue is $\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_1^*)) \gg \tau$. For Θ_2 are randomly initialized in a flat region, and so the largest eigenvalue is $\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_2)) \leq \tau$.

Under a local quadratic approximation of the loss function around the interpolation path, the Hessian at Θ_3 can be approximated by a convex combination of the Hessians at Θ_1^* and Θ_2:

$$\nabla^2\mathcal{L}(\Theta_3) \approx \alpha\nabla^2\mathcal{L}(\Theta_1^*) + (1-\alpha)\nabla^2\mathcal{L}(\Theta_2) \tag{1}$$

Consequently, the maximum eigenvalue satisfies:

$$\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_3)) \lesssim \alpha\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_1^*)) + (1-\alpha)\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_2)) \tag{2}$$

Since $\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_2)) \ll \lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_1^*))$ (flat vs. sharp), it follows that $\lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_3)) < \lambda_{\text{max}}(\nabla^2\mathcal{L}(\Theta_1^*))$ for $\alpha \in (0,1)$. This suggests that interpolation reduces local sharpness, i.e., helps "sharp weights" escape the non-flat regions for a smoother one and a better fine-tuning effect. This provides theoretical support for SFF's smoothing effect.

Preservation of flat regions. After pre-training the LTSM, for the flat minimum points $\bar{\Theta}_1^*$, interpolation preserves its flatness.

Both $\bar{\Theta}_1^*$ and Θ_2 lie in flat regions of the loss landscape, i.e.,

$$\lambda_{\max}\left(\nabla^2\mathcal{L}(\bar{\Theta}_1^*)\right) \leq \tau, \quad \lambda_{\max}\left(\nabla^2\mathcal{L}(\Theta_2)\right) \leq \tau \tag{3}$$

Similarly, the Hessian at the interpolated point $\Theta_3 = \alpha\bar{\Theta}_1^* + (1-\alpha)\Theta_2$ satisfies $\nabla^2\mathcal{L}(\Theta_3) \approx \alpha\nabla^2\mathcal{L}(\bar{\Theta}_1^*) + (1-\alpha)\nabla^2\mathcal{L}(\Theta_2)$. Consequently, we can derive that:

$$\lambda_{\max}\left(\nabla^2\mathcal{L}(\Theta_3)\right) \lesssim \alpha\lambda_{\max}\left(\nabla^2\mathcal{L}(\bar{\Theta}_1^*)\right) + (1-\alpha)\lambda_{\max}\left(\nabla^2\mathcal{L}(\Theta_2)\right) \leq \alpha\tau + (1-\alpha)\tau = \tau \tag{4}$$

Hence, Θ_3 remains in a flat region. According to (Foret et al., 2020), this indicates that interpolation does not harm existing flat minima $\bar{\Theta}_1^*$.

In summary, from the perspective of mathematical rigor, we ensure that interpolation smooths (perturbs) sharp minima without obviously harming originally flat regions, thereby aiding sharp minima escape to better and smoother basins.

3.1.2 Discussion on parameter initialization and the smoothness of the corresponding loss landscape

Visualizations show that initialization methods like Kaiming He et al. (2015) and Xavier Glorot & Bengio (2010) yield a smooth loss landscape. Based on this observation, we apply them as auxiliary initialization schemes for LSTM to promote smoothed full finetuning. In this section, we further study this phenomenon from a theoretical standpoint. Prior work Fort & Scherlis (2019) quantifies the smoothness of the loss landscape under Kaiming and Xavier initializations by the ratio of the trace of the Hessian matrix $Tr(H)$ to its Frobenius norm $\|H\|_F$ and has demonstrated $\frac{\mathrm{Tr}(H)}{\|H\|_F} \gg 1$ from both theoretical and experimental perspectives, i.e., mainstream initialization indeed produce a smooth, flat, and more easily optimizable loss landscape. We further provide a theoretical analysis for this conclusion. Specifically, according to the symmetry of H, we explicitly expand the formula as:

$$\frac{\mathrm{Tr}(H)}{\|H\|_F} = \frac{\mathrm{Tr}(H)}{\sqrt{\sum_{i=1}^{d} h_i^2}} = \frac{\mathrm{Tr}(H)}{\sqrt{\mathrm{Tr}(H^T H)}} = \frac{\sum \lambda_i}{\sqrt{\sum \lambda_i^2}} \gg 1 \tag{5}$$

This indicates that the sum of eigenvalues greatly exceeds the square root of the sum of squared eigenvalues, suggesting that most eigenvalues are positive and relatively evenly distributed rather than containing extreme outliers, i.e., most cases belong to $\lambda(\nabla^2\mathcal{L}(\Theta^*)) \leq \tau$ where τ>0. According to the definition of **Theorem 1** and **Theorem 2**, this indicates that the loss surface exhibits a smooth, valley-like geometry dominated by positive curvature. As a result, most random descent directions remain stable and low-curvature, making the optimization process easier and more consistent.

In contrast, if $\frac{\mathrm{Tr}(H)}{\|H\|_F} \lesssim 1$, it indicates a balanced mix of positive and negative eigenvalues λ, leading to a steep and unsmooth loss landscape. As a result, convergence becomes slower and the optimizer is more likely to fall into suboptimal local minima.

Therefore, Kaiming or Xavier initialization offers a stable and smooth loss landscape. In our work, we adopt them for randomly initializing the auxiliary LSTM.

3.2 Smoothing the loss landscape then fine-tuning

We define the training set as $\mathcal{D} = \{(\mathcal{X}_1, Y_1), ..., (\mathcal{X}_N, Y_N)\}$, where $\mathcal{X}_N = [x_1, x_2, \ldots, x_t] \in \mathbb{R}^{t\times v}$ with length t for all v time variables. We divide the pre-trained LTSM into two parts: model backbone $G(\mathcal{X}, \Phi_1)$ and linear head $\mathbf{W}_{\text{head1}} \in \mathbb{R}^{d\times h}$. $\mathcal{X}$ is an input time series and Φ_1 are the parameters of the pre-trained LTSM backbone. d and h are the output sizes of the backbone and linear head.

For simplicity, we define the parameters of **auxiliary** LTSM are $\Theta_2 = [\Phi_2, \mathbf{W}_{\text{head2}}]$, including LTSM backbone and its head. Given an coefficient α, we can obtain the new model weights $\Theta_3 = [\Phi_3, \mathbf{W}_{\text{head3}}]$ through linear interpolation between parameters of **auxiliary** LTSM (Θ_2) and **pre-trained** LTSM (Θ_1). As a result, the formula of full fine-tuning, linear probing, and loss function can be formulated as Eq. 6, Eq. 7, and Eq. 8 respectively:

$$f(X, \Theta_3) = G(\mathcal{X}, \alpha\Phi_1 + (1-\alpha)\Phi_2)^T(\alpha\mathbf{W}_{\text{head1}} + (1-\alpha)\mathbf{W}_{\text{head2}}) \tag{6}$$

$$f(\mathcal{X}, \Theta_3) = G(\mathcal{X}, \alpha\Phi_1 + (1-\alpha)\Phi_2)^T_{\textbf{frozen}}(\alpha\mathbf{W}_{\text{head1}} + (1-\alpha)\mathbf{W}_{\text{head2}}) \quad (7)$$

$$\arg \min_{\alpha\Theta_1+(1-\alpha)\Theta_2} \sum_{(\mathcal{X}_i, Y_i)\in\mathcal{D}} \mathcal{L}\left(f(\mathcal{X}_i, \alpha\Theta_1 + (1-\alpha)\Theta_2), Y_i\right) \quad (8)$$

where α is the interpolation coefficient and controls the proportion of pre-trained knowledge retained. The larger α is, the more pre-trained knowledge is preserved.

Smoothing the loss landscape can be implemented in a few lines of PyTorch, and we provide example code in the Appendix Algorithm 1.

4 EXPERIMENTAL SETTINGS

LTSM Baselines. We use the eight popular LTSMs as baselines with diverse architectures, including **encoder-only** (Moirai Woo et al. (2024a) and MOMENT Goswami et al. (2024)), **decoder-only** (Sundial Liu et al. (2025), Timer Liu et al. (2024), and TimesFM Das et al. (2024)), **encoder-decoder** (Chronos Ansari et al. (2024) and UniTs Gao et al. (2024)), and **light-weight MLP model** (TTMs Ekambaram et al. (2024)). They also include models of different sizes, ranging from larger TimesFM (3.8GB) to smaller TTMs (3MB). We verify that our method can enhance their performance. We download these pre-trained models from the official links for experiments, e.g., with model sizes being 851MB, 2.6GB, and 3.8GB for Timer, MOMENT, and TimesFM, respectively.

Fine-tuning baselines. In addition to comparing with typical fine-tuning baselines, e.g., full fine-tuning (FF), linear probing (LP), and linear probing first and then full fine-tuning (LP-FF). We also incorporated various optimization strategies employed during model training, such as label smoothing, SAM (Foret et al., 2020), SWA (Izmailov et al., 2018), Mixout (Lee et al.), and L2-SP (Xuhong et al., 2018). Comparison results are shown in the Appendix Table 10 due to limited space. More details about datasets, evaluations, and implementation details are shown in Appendix A.5.

5 EXPERIMENTAL RESULTS

We conduct extensive experiments to validate the effectiveness of the proposed smoothed fine-tuning, including 8 TSF datasets and 250 anomaly detection datasets, also involving 8 popular LTSMs. To ensure the effectiveness of our method is not a random occurrence, we have conducted experiments with multiple random seeds under varying available data proportions. Our method also achieves improvements on the imputation task, as shown in Appendix Figure 10 due to limited space.

We also evaluate different initialization schemes and random seeds on SFF in Appendix Section A.7.1, Tables 11 and 12. The results show that mainstream initializations (e.g., Kaiming, Xavier) consistently yield stable gains, and SFF shows no noticeable sensitivity to the initialization seed.

Table 1: MSE of fine-tuning LTSM Timer for time series forecasting under different proportions of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively. Full standard deviations are shown in the Appendix Table 17. The results, improvements, and standard deviations under the available data proportion 1% to 20% are shown in Appendix Table 15 and Table 16. Similarly, MAE results are shown in Tables 18, 19, and 20.

Data proportion	25%			50%			75%			100%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange Standard deviation	**0.0805** ±4.5e-4	<u>0.0865</u> ±1.9e-4	0.1441 ±2.0e-3	**0.0802** ±5.4e-4	<u>0.0891</u> ±2.3e-3	0.114 ±9.9e-4	**0.0802** ±1.2e-3	<u>0.0914</u> ±1.6e-3	0.1026 ±8.8e-4	**0.08** ±7.6e-4	<u>0.091</u> ±1.3e-4	0.0981 ±1.2e-3
ETTh1	**0.3506**	<u>0.355</u>	0.3788	**0.3494**	<u>0.3573</u>	0.367	**0.3493**	<u>0.358</u>	0.3593	**0.3547**	0.3709	<u>0.36</u>
ETTh2	**0.271**	<u>0.2866</u>	0.2891	**0.273**	0.2905	<u>0.2775</u>	**0.2772**	0.3032	<u>0.2796</u>	**0.2737**	0.3047	<u>0.2777</u>
ETTm1	**0.298**	<u>0.3049</u>	0.333	**0.2955**	<u>0.3069</u>	0.3189	**0.2956**	<u>0.3092</u>	0.3116	**0.2954**	0.3128	<u>0.3093</u>
ETTm2	**0.1594**	<u>0.1707</u>	0.1741	**0.1605**	0.1718	<u>0.1627</u>	**0.1623**	0.1838	<u>0.1651</u>	**0.16**	0.1784	<u>0.1644</u>
Weather	**0.144**	<u>0.1472</u>	0.1627	**0.1441**	<u>0.1523</u>	0.1538	**0.1466**	0.1665	<u>0.1559</u>	**0.1443**	0.1612	<u>0.1526</u>
Electricity	**0.1303**	<u>0.1344</u>	0.1365	**0.1301**	0.1347	<u>0.1327</u>	**0.13**	0.1367	<u>0.1326</u>	**0.1304**	0.1344	<u>0.1324</u>
Traffic	**0.3488**	<u>0.3582</u>	0.3688	**0.3497**	0.3586	<u>0.3552</u>	**0.3478**	0.361	<u>0.3606</u>	**0.3551**	<u>0.3599</u>	0.3609

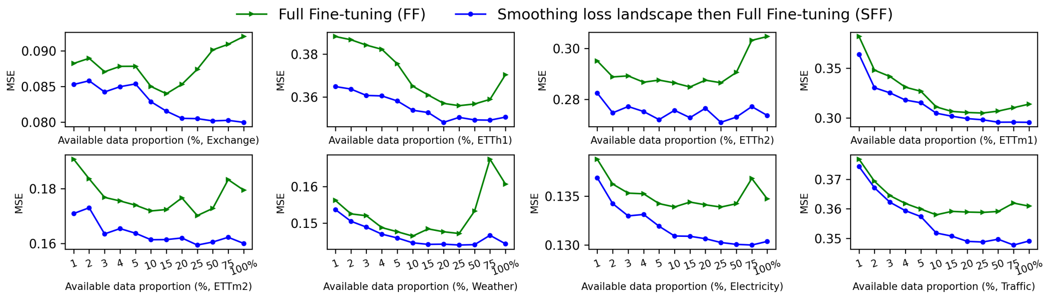


Figure 3: MSE of different fine-tuning strategies on the test set of various datasets with varied available data proportions (1% to 100%).

5.1 Fine-tuning Timer for multivariate time series forecasting (TSF) and anomaly detection

Experiments show that training from scratch (TFS) requires substantially more data to achieve good accuracy, while pretrained LSTMs reach—or even surpass—TFS's full-data performance using only 10–25% of the data. However, pretrained LSTMs often exhibit poor trainability due to overfitting, resulting in steep and unsmooth loss landscapes that weaken fine-tuning. As a result, full fine-tuning (FF) may struggle to improve—and can even degrade—as data increases (Figure 3). In contrast, our Smoothed Full Fine-tuning (SFF) consistently outperforms FF across all data proportions, with performance improving as more data is used. Across nine public datasets, SFF reduces MSE by an average of 3% and up to 6.5% over FF (Table 1). These results demonstrate that SFF effectively smooths the loss landscape, enhances trainability, and better leverages pretrained knowledge—without any additional memory or computation.

Similarly, in the anomaly detection task, we observe that the TFS overall performs better than FF. When using MSE as a confidence measure for anomaly detection, the higher the predicted MSE for anomalous segments, the better. As shown in Table 2, the average MSE for anomalous segments obtained through TFS is overall higher than that of FF. In contrast, the SFF achieves significantly higher MSE predictions for anomalous segments compared to both FF and TFS, which demonstrates the superiority of our SFF.

Table 2: Results on anomaly detection. We report the predicted MSE of the anomalous segments, and the higher is better. There are a total of 250 datasets. Due to space limitations, we sequentially report the average MSE and Wins in six groups under four random seeds. The MSE and standard deviation of each dataset are shown in Appendix Table 21 and Table 22.

Group 1 (41 datasets)						Group 2 (41 datasets)						Group 3 (41 datasets)					
SFF		FF		TFS		SFF		FF		TFS		SFF		FF		TFS	
MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins
0.136	**31.3**	0.072	4.3	0.073	5.3	**0.206**	**32.0**	0.098	5.3	0.089	3.7	**0.209**	**30.3**	0.104	5.0	0.112	5.7
±1.4e-2	±1.7	±1.3e-2	±1.9	±1.6e-2	±0.47	7.7e-3	0.82	4.0e-3	0.94	1.0e-2	0.47	2.9e-2	1.7	1.0e-2	0.0	2.1e-2	1.7
Group 4 (41 datasets)						Group 5 (41 datasets)						Group 6 (45 datasets)					
SFF		FF		TFS		SFF		FF		TFS		SFF		FF		TFS	
MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins	MSE	Wins
0.201	**33.7**	0.084	3.3	0.087	4.0	**0.163**	**29.0**	0.078	3.3	0.09	8.7	**0.157**	**34.3**	0.085	5.0	0.09	5.7
±2.9e-2	±0.47	±9.9e-3	±0.47	±8.1e-3	±0.82	3.4e-2	2.2	6.2e-3	0.47	2.4e-2	1.9	6.7e-3	1.2	9.8e-3	2.4	3.6e-3	2.4

5.2 Applying smoothed full fine-tuning (SFF) for other LTSMs

As shown in Table 3, for TimesFM and MOMENT, FF also performs worse than TFS in some cases, which indicates that the pre-trained LTSM may indeed suffer from overfitting issues. However, our SFF fine-tuning strategy outperforms both FF and TFS. Compared to FF, SFF achieves average improvements of 11.45% for TimesFM and 8.31% for MOMENT under different data proportions.

Moreover, as shown in Table 4, the experiments on more LTSMs, including UniTS, MOIRAI, Chronos, TTMs, and Sundial, SFF consistently outperforms FF, which indicates that the interpolation-based smoothing strategy of SFF can indeed improve the fine-tuning effect of LTSM. This is because

Table 3: MSE of applying our *smoothed fine-tuning* (SFF) on other LTSMs TimesFM and MOMENT. Full standard deviations and MAE results are shown in Appendix Table 23 and Table 24.

Data proportion	25% (TimesFM)			100% (TimesFM)			25% (MOMENT)			100% (MOMENT)		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.1139**	0.1276	0.1209	**0.1149**	0.1452	0.1199	**0.1502**	0.2648	0.1564	**0.1064**	0.1448	0.1091
Standard deviation	2.0e-3	4.2e-3	2.9e-4	6.3e-4	1.7e-2	2.3e-3	2.4e-3	4.6e-4	3.6e-3	5.8e-4	1.4e-4	2.6e-4
ETTh1	**0.3955**	0.4382	0.4638	**0.406**	0.5101	0.4358	**0.4287**	0.4454	0.454	**0.3757**	0.3951	0.387
ETTh2	**0.3232**	0.3384	0.3325	**0.3198**	0.3483	0.347	**0.3199**	0.3328	0.3326	**0.2818**	0.2936	0.2979
ETTm1	**0.3429**	0.4001	0.3903	**0.3478**	0.3756	0.3926	**0.3457**	0.3587	0.3538	**0.3139**	0.3148	0.3272
ETTm2	**0.1983**	0.2061	0.2091	**0.2026**	0.2122	0.225	**0.1793**	0.192	0.1846	**0.1692**	0.172	0.1736
Weather	**0.0865**	0.0885	0.1995	**0.082**	0.1184	0.1902	**0.1673**	0.1682	0.169	**0.1548**	0.1558	0.161

interpolation perturbs sharp minima without obviously harming originally flat regions, thereby aiding the escape of sharp minima to better and smoother basins.

Overall, our method works across various architectures of LTSMs, including encoder-only, decoder-only, encoder-decoder, and MLP-only, showing universality and generalizability.

Table 4: MSE of fine-tuning more LTSMs for the TSF task with prediction length 96. "-" indicates that the preprocessed version of the dataset is not provided in the official codes (Chronos) or that it runs out of memory (Sundial). The results on length 720 are shown in Appendix Table 13.

	UniTS-SFF	UniTS-FF	MOIRAI-SFF	MOIRAI-FF	Chronos-SFF	Chronos-FF	TTMs-SFF	TTMs-FF	Sundial-SFF	Sundial-FF
ETTh1	**0.656**	0.678	**0.448**	0.501	**0.773**	0.799	**0.367**	0.371	**0.368**	0.372
ETTh2	**0.364**	0.374	**0.32**	0.321	-	-	**0.275**	0.278	**0.293**	0.31
ETTm1	**0.355**	0.365	**0.308**	0.348	**0.687**	0.724	**0.309**	0.308	**0.419**	0.428
ETTm2	**0.179**	0.184	**0.181**	0.184	-	-	**0.17**	0.178	**0.182**	0.195
Weather	**0.171**	0.198	**0.166**	0.173	**1.137**	1.276	**0.157**	0.159	**0.179**	0.186
Elect.	**0.309**	0.476	**0.221**	0.226	**0.861**	0.885	**0.149**	0.151	-	-
Traffic	**0.877**	1.195	**0.476**	0.497	**0.831**	0.834	**0.453**	0.462	-	-

Table 5: MSE of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) Kumar et al.. Full standard deviations and MAE results are shown in Table 25 and Table 26. The average performance is reported for each group of three different data proportions, e.g., "Avg. on 1%, 2%, 3%". The MSE, MAE and standard deviations under each proportion are shown in Appendix Tables 27, 28, 29, 30, 31 and 32.

Data proportion	Avg. on 1%, 2%, 3%			Avg. on 4%, 5%, 10%			Avg. on 15%, 20%, 25%			Avg. on 50%, 75%, 100%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.0856**	0.5943	0.4801	**0.0848**	0.5906	0.4186	**0.0816**	0.563	0.1743	**0.0812**	0.474	0.0962
Standard deviation	4.4e-4	7.3e-3	3.9e-3	3.7e-4	7.2e-3	6.7e-3	6.1e-4	6.6e-3	7.4e-3	8.3e-4	4.7e-3	1.9e-3
ETTh1	**0.3722**	0.8806	0.7171	**0.3641**	0.8594	0.6367	**0.3523**	0.7955	0.4127	**0.3529**	0.6356	0.3731
ETTh2	**0.28**	0.4427	0.4026	**0.278**	0.4375	0.3707	**0.2768**	0.4234	0.3113	**0.2758**	0.3849	0.3001
ETTm1	**0.3448**	1.046	0.7038	**0.3162**	1.0043	0.4772	**0.301**	0.8975	0.3245	**0.2976**	0.6608	0.3124
ETTm2	**0.1723**	0.3555	0.3024	**0.1663**	0.3499	0.2559	**0.1623**	0.3297	0.1847	**0.1616**	0.2771	0.1804
Weather	**0.1515**	0.324	0.2478	**0.146**	0.3082	0.1741	**0.1441**	0.2699	0.1481	**0.1453**	0.2013	0.1565
Electricity	**0.1346**	0.6069	0.181	**0.132**	0.3242	0.1398	**0.1305**	0.2023	0.132	**0.1301**	0.1561	0.1335
Traffic	**0.3678**	0.9577	0.4081	**0.3562**	0.5999	0.3638	**0.3494**	0.4529	0.3572	**0.3516**	0.4079	0.3575

5.3 Compare SFF with other fine-tuning strategies

We compare SFF with the popular fine-tuning strategies, linear-probing (LP), and linear-probing then full fine-tuning (LPFF) Kumar et al.. The results in Table 5 show that SFF outperforms LP and LPFF under multiple proportions of finetuning data, with average MSE improvements of 7.17% to

41.57% compared to the best competitor LPFF. This demonstrates that SFF is indeed an effective new fine-tuning strategy. It smooths the sharp regions of the loss landscape first, and achieves better trainability and fine-tuning.

5.4 Discussion of retained pretraining knowledge after smoothing the loss landscape

Impact of smoothing the loss landscape on convergence speed. We record the test loss of different fine-tuning strategies at each epoch, as shown in Figure 4. Pre-trained model possess general knowledge and can quickly extract universal features from time series, thereby only requiring a few fine-tunings to converge. In Figure 4, both SFF and FF converge within the first epoch (enclosed by the red rectangles). This indicates that SFF-based LTSM also effectively retains the pre-trained knowledge after smoothing the loss landscape through model interpolation. Moreover, SFF converges to a lower MSE than FF, indicating that the smoothing process enhances the trainability of the LTSM, consistent with our design motivation.

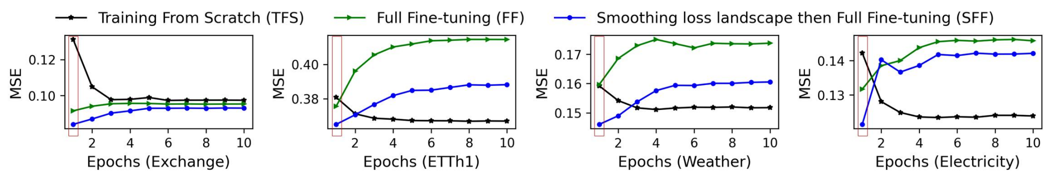


Figure 4: Different methods of forecasting on the test set of various datasets at each epoch.

Impact of smoothing the loss landscape on zero-shot forecasting. Since the pre-trained knowledge is still well-preserved after smoothing, what is its impact on zero-shot prediction? As shown in Table 6, the smoothed LTSM improves zero-shot accuracy on seven datasets, with average gains of 6.13% (Timer) and 35.75% (TimesFM). These results, averaged over multiple seeds, confirm that smoothing guides the pretrained model to a better local optimum. Besides, Figure 5 shows that an interpolation coefficient $\alpha \approx 0.85$ yields the best zero-shot performance, achieving sufficient smoothing with enough preservation of pretrained knowledge. Moreover, as shown in Table 7, the smoothing process also generally brought about an increase in the accuracy of zero-shot prediction on more LTSMs, including MOIRAI, Chronos, TTMs, and Sundial, showing our method's generalizability.

Table 6: Smoothing the loss landscape then perform zero-shot forecasting. MAE results are shown in Table 33. The result of random initialization is ignored since it shows significantly poor performance due to a lack of pre-trained knowledge, as "Re-initialize" shown in Figure 5.

	ETTh1		ETTh2		ETTm1		ETTm2		Weather		Electricity		Traffic	
	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth
MSE	0.454	**0.399**	0.316	**0.289**	0.816	**0.794**	0.225	**0.209**	0.19	**0.179**	0.210	**0.203**	0.479	**0.463**
Std.	±0	±1.3e-3	±0	±2.2e-3	±0	±1.2e-2	±0	±1.6e-3	±0	±9.0e-4	±0	±6.2e-4	±0	±6.6e-4
	TimesFM	+Smooth	TimesFM	+Smooth	TimesFM	+Smooth	Tim.FM	+Smooth	Time.FM	+Smooth	Tim.FM	+Smooth	Tim.FM	+Smooth
MSE	0.782	**0.741**	1.865	**0.382**	1.359	**0.993**	1.375	**0.256**	0.397	**0.227**	0.94	**0.886**	1.665	**1.521**
Std.	±0	±7.9e-3	±0	±4.8e-4	±0	±2.4e-2	±0	±5.8e-3	±0	±4.3e-4	±0	±3.6e-3	±0	±1.3e-2

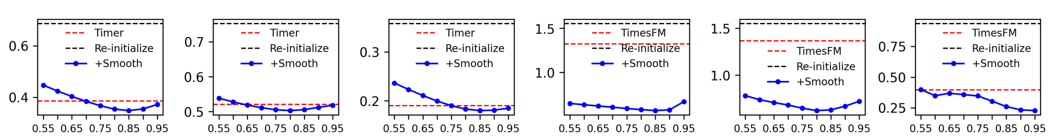


Figure 5: Impact of interpolation coefficient α on zero-shot forecasting.

5.5 Hyperparameter sensitivity analysis

Influence of the interpolation coefficient α. As shown in Figure 6, although the coefficient α in SFF may affect performance, across most α values (0.1–0.75), SFF achieves lower MSE than standard full fine-tuning (red dashed line). This also confirms that smoothing improves trainability and enables pretrained LTSMs to better leverage knowledge for higher fine-tuning accuracy.

Table 7: Zero-shot forecasting MSE of more LTSMs with prediction length 96. UniTS is not included since it focuses on few-shot learning. The results on length 720 are shown in Appendix Table 14.

	MOIRAI	+Smooth	Chronos	+Smooth	TTMs	+Smooth	Sundial	+Smooth
ETTh1	0.419	**0.405**	0.816	**0.779**	0.364	**0.362**	0.394	**0.385**
ETTh2	0.305	**0.295**	-	-	0.277	**0.275**	0.306	**0.303**
ETTm1	0.557	**0.552**	0.697	**0.655**	0.322	**0.315**	0.365	**0.362**
ETTm2	0.227	**0.219**	-	-	**0.171**	0.172	0.2	**0.191**
Weather	0.192	**0.189**	1.259	**1.087**	0.158	**0.158**	0.175	**0.173**
Elect.	0.21	**0.197**	0.823	**0.822**	**0.166**	0.167	-	-
Traffic	0.555	**0.544**	0.854	**0.836**	**0.514**	0.516	-	-

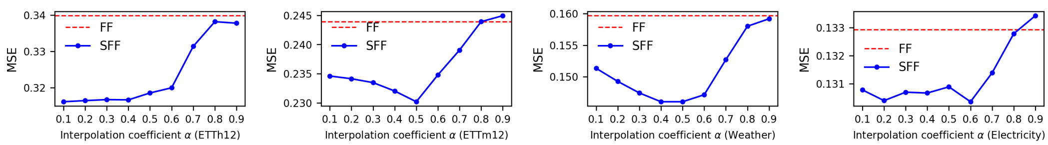


Figure 6: TSF of different interpolation coefficients α on the test set of various datasets. The available data proportion is 100%. ETTh12 denotes that the MSE is average on the ETTh1 and ETTh2 datasets.

Table 8: MSE performance of adding randomly initialized parameters to the pre-trained UniTS. The percentages in the first column indicate the proportion of parameters subject to weight perturbation.

	ETTh1	ETTh2	ETTm1	ETTm2	Weather	Electricity	Traffic
Proportion (17.91%)-96	0.678	0.373	0.359	0.181	0.192	0.474	1.194
Proportion (35.82%)-96	0.665	0.366	0.356	0.181	0.182	0.464	1.135
Proportion (53.73%)-96	0.656	0.364	0.36	0.181	0.183	0.462	1.138
Proportion (100%)-96	0.662	0.367	0.355	0.179	0.171	0.309	0.877
Proportion (17.91%)-720	0.735	0.431	0.626	0.419	0.339	0.492	1.305
Proportion (35.82%)-720	0.711	0.434	0.627	0.416	0.337	0.466	1.27
Proportion (53.73%)-720	0.704	0.431	0.595	0.418	0.334	0.431	1.238
Proportion (100%)-720	0.707	0.436	0.496	0.419	0.324	0.355	1.01

Influence of the interpolation proportion of model parameters. Notably, SFF operates at the parameter level, not the layer level (e.g., LayerNorm, FFN), without favoring specific layers. The key factor is the proportion of interpolated parameters. To verify this, we start from the first UniTS block and gradually increase the proportion of parameters undergoing weight interpolation (perturbation), observing fine-tuning performance across datasets. As Table 8 shows, larger datasets benefit from more interpolation (MSE: 100% < 53.73% < 35.82% < 17.91% for Electricity and Traffic), whereas smaller datasets benefit from less (MSE: 53.73% < 100% for ETTh1 and ETTh2). This is reasonable because larger datasets support broader parameter updates, exploring more non-convex regions and escaping sharp minima, while smaller datasets may under-train if too many parameters are interpolated, harming performance.

6 Conclusion

In this work, we identify a key challenge: pretrained LSTMs often suffer from poor trainability during fine-tuning due to a steep, unsmoothed loss landscape, which limits the benefits of pretraining. We address this with a lightweight strategy that first smooths the loss landscape—without additional memory or computational cost—and then performs downstream fine-tuning. This improves trainability while preserving pretrained knowledge, yielding consistently stronger performance. Theoretically, SFF works by perturbing sharp minima without affecting inherently flat regions, allowing the sharp minima to escape unfavorable basins. Extensive experiments on eight public datasets using eight pretrained LSTMs of varying architectures and sizes confirm that the improvements are robust across seeds and data regimes. We believe these insights may also benefit fine-tuning in broader pretrained-model settings.

7 Ethics statement

This work adheres to the ICLR Code of Ethics. In this study, no human subjects or animal experimentation were involved. All datasets used in this paper were sourced in compliance with relevant usage guidelines, ensuring no violation of privacy. We have taken care to avoid any biases or discriminatory outcomes in our research process. No personally identifiable information was used, and no experiments were conducted that could raise privacy or security concerns. We are committed to maintaining transparency and integrity throughout the research process.

8 Reproducibility statement

The efforts we have made to ensure the reproducibility of our algorithms and experimental results are as follows:

- We ran four random seeds in our experiments and reported the mean and standard deviation of the results to enhance reproducibility.
- We added an anonymous code link to the ABSTRACT for convenient download. The codes include the hyperparameter settings used in the experiments. By providing the code and detailed hyperparameter settings, we ensure that our algorithms and experimental results are reproducible.

References

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

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.

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

Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. *arXiv preprint arXiv:2402.05956*, 2024.

Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, pp. 10148–10167. PMLR, 2024.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In Ambuj K. Singh, Yizhou Sun, Leman Akoglu, Dimitrios Gunopulos, Xifeng Yan, Ravi Kumar, Fatma Ozcan, and Jieping Ye (eds.), *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pp. 459–469. ACM, 2023. doi: 10.1145/3580305.3599533. URL `https://doi.org/10.1145/3580305.3599533`.

Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam Nguyen, Wesley M Gifford, Chandra Reddy, and Jayant Kalagnanam. Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. *Advances in Neural Information Processing Systems*, 37:74147–74181, 2024.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Stanislav Fort and Adam Scherlis. The goldilocks zone: Towards better understanding of neural network loss landscapes. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pp. 3574–3581, 2019.

Milton Friedman. The interpolation of time series by related series. *Journal of the American Statistical Association*, 57(300):729–757, 1962.

Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. Units: Building a unified time series model. *arXiv*, 2024. URL `https://arxiv.org/pdf/2403.00131.pdf`.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: a family of open time-series foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 16115–16152, 2024.

Trevor Hastie. The elements of statistical learning: data mining, inference, and prediction, 2009.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

P Izmailov, AG Wilson, D Podoprikhin, D Vetrov, and T Garipov. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pp. 876–885, 2018.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

D Kim, J Park, J Lee, and H Kim. Are self-attentions effective for time series forecasting? In *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

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

Jędrzej Kozal, Jan Wasilewski, Bartosz Krawczyk, and Michał Woźniak. Continual learning with weight interpolation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4187–4195, 2024.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

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.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: generative pre-trained transformers are large time series models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 32369–32399, 2024.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. *arXiv preprint arXiv:2502.00816*, 2025.

Donghao Luo and Xue Wang. Deformabletst: Transformer for time series forecasting without over-reliance on patching. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a.

Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *The twelfth international conference on learning representations*, pp. 1–43, 2024b.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Hansheng Ren, Bixiong Xu, Yujing Wang, Chao Yi, Congrui Huang, Xiaoyu Kou, Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. Time-series anomaly detection service at microsoft. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3009–3017, 2019.

Tiffany J Vlaar and Jonathan Frankle. What can linear interpolation of neural network loss landscapes tell us? In *International Conference on Machine Learning*, pp. 22325–22341. PMLR, 2022.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. Timemixer++: A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv:2410.16032*, 2024a.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. Timemixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024b.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. 2024a.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *International Conference on Machine Learning*, pp. 53140–53164. PMLR, 2024b.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

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.

Renjie Wu and Eamonn J Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE transactions on knowledge and data engineering*, 35(3): 2421–2429, 2021.

LI Xuhong, Yves Grandvalet, and Franck Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *International conference on machine learning*, pp. 2825–2834. PMLR, 2018.

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

Xu Zhang, Zhengang Huang, Yunzhi Wu, Xun Lu, Erpeng Qi, Yunkai Chen, Zhongya Xue, Qitong Wang, Peng Wang, and Wei Wang. Multi-period learning for financial time series forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 2848–2859, 2025a.

Xu Zhang, Qitong Wang, Peng Wang, and Wei Wang. A lightweight sparse interaction network for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 13304–13312, 2025b.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2023.

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.

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

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023.

A Technical Appendices and Supplementary Material

A.1 The Use of Large Language Models (LLMs)

Large Language Models (LLMs) were used to aid in the writing and polishing of the manuscript. Specifically, we used an LLM to assist in refining the language, improving readability, and ensuring clarity in various sections of the paper. The model helped with tasks such as sentence rephrasing and enhancing the overall flow of the text.

It is important to note that the LLM was not involved in the ideation, research methodology, or experimental design. All research concepts, ideas, and analyses were developed and conducted by the authors. The contributions of the LLM were solely focused on improving the linguistic quality of the paper, with no involvement in the scientific content or data analysis.

The authors take full responsibility for the content of the manuscript, including any text generated or polished by the LLM. We have ensured that the LLM-generated text adheres to ethical guidelines and does not contribute to plagiarism or scientific misconduct.

A.2 Proof for Theorems about Sharp minimum and Flat minimum

We start by performing a second-order Taylor expansion of the loss function $\mathcal{L}(\Theta^*)$ at the minimum point Θ^*:

$$\begin{aligned} \mathcal{L}(\Theta^* + \delta) &\approx \mathcal{L}(\Theta^*) + \frac{1}{2}\delta^\top \nabla^2 \mathcal{L}(\Theta^*)\delta \\ &\approx \frac{1}{2}\delta^\top \nabla^2 \mathcal{L}(\Theta^*)\delta \end{aligned} \tag{9}$$

where δ is a small parameter perturbation (e.g., $|\delta| \ll 1$).

The loss change after perturbation can be formulated as:

$$\Delta\mathcal{L} = \mathcal{L}(\Theta^* + \delta) - \mathcal{L}(\Theta^*) \approx \frac{1}{2}\delta^\top \nabla^2 \mathcal{L}(\Theta^*)\delta \approx \frac{1}{2}\delta^\top H\delta \tag{10}$$

The largest $\Delta\mathcal{L}$ is governed by the λ_{max} of the Hessian $H = \nabla^2\mathcal{L}(\Theta^*)$.

Let unit vector v_{max} be the eigenvector corresponding to eigenvalue λ_{max}. Perturbing in the direction of v_{max} yields the largest $\Delta\mathcal{L}$ because v_{max} represents the largest curvature (Foret et al., 2020), i.e., steepest direction, in the loss landscape. Let $\delta = \epsilon \cdot v_{\text{max}}$ ($|\epsilon| \ll 1$), the Eq. 10 can be reformulated as:

$$\Delta\mathcal{L} \approx \frac{1}{2}\epsilon \cdot v_{\text{max}}^\top H\epsilon \cdot v_{\text{max}} \tag{11}$$

According to eigen equation, $Hv_{\text{max}} = \lambda_{\text{max}}v_{\text{max}}$, we can further obtain

$$\Delta\mathcal{L} \approx \frac{1}{2}\epsilon^2 v_{\text{max}}^\top \lambda_{\text{max}} v_{\text{max}} \approx \frac{1}{2}\epsilon^2 v_{\text{max}}^\top v_{\text{max}}\lambda_{\text{max}} \approx \frac{1}{2}\epsilon^2\lambda_{\text{max}} \tag{12}$$

Hence, $\Delta\mathcal{L}$ is closely related to λ_{max}, the larger λ_{max} is, the sharper the loss landscape, so parameter perturbations in many directions can noticeably increase the loss. Conversely, the smaller λ_{max} is, the flatter the landscape remains, and perturbations won't obviously raise the loss (Hochreiter & Schmidhuber, 1997).

A.3 Related works

Fine-tuning large time series models. Most works focus on designing pre-training architecture and collecting large-scale time series data Das et al. (2024); Goswami et al. (2024); Woo et al. (2024b); Liu et al. (2024) to enrich and improve the foundations of LTSM. The fine-tuning technique has received relatively less attention in the field of large time series models. The traditional fine-tuning strategies are either *full fine-tuning* (adjusting all model parameters) or only fine-tuning the prediction head (also called *linear probing*). In the computer vision domain, Kumar et al. points out that full fine-tuning can achieve worse accuracy than linear probing in the condition of meeting out-of-distribution (OOD) data when the pretrained features are good and the distribution shift is large. To address this, they propose to linear probing first and then full fine-tuning (LP-FF) to improve the fine-tuning performance of the pre-trained model in the OOD condition. In our study, we also apply LP-FF to fine-tune the LTSM as a baseline to compare with our method.

Various optimization strategies can also be used for fine-tuning. Specifically, SAM (Sharpness-Aware Minimization) (Foret et al., 2020) updates parameters not only by the loss at the current point, but also by the flatness in a small neighborhood, so the optimizer is less likely to fall into high-curvature sharp minima. Note, however, that SAM needs one extra forward–backward pass, so its training cost is twice that of ordinary training. SWA (Stochastic Weight Averaging) (Izmailov et al., 2018) is an ensemble-like strategy: it saves several weight snapshots taken after the model has converged and averages them to produce the final weights, reducing randomness and over-fitting. Mixout (Lee et al.) randomly replaces a subset of the fine-tuned weights with their pre-trained counterparts during training, mitigating catastrophic forgetting and over-fitting. L2-SP (Xuhong et al., 2018) adds an L2 penalty between the current weights and the pre-trained weights to the loss, preventing the fine-tuned model from drifting too far away from the pre-trained solution and thus preserving pre-trained knowledge while curbing over-fitting.

Small time series models. Recently, many small end-to-end models for time series analysis (e.g., forecasting) have been proposed. At first, transformer-based methods Wu et al. (2021); Zhou et al. (2022); Liu et al. (2021); Chen et al. (2024); Nie et al. (2023); Zhou et al. (2021); Zhang et al. (2025a); Zhang & Yan (2023); Kim et al. (2024) greatly promoted the development of the field. Lately, methods based on simple linear layers Wang et al. (2024b); Zhang et al. (2025b); Wang et al. (2024a); Zeng et al. (2023); Ekambaram et al. (2023); Chen et al. (2023) and CNNs Bai et al. (2018); Luo & Wang (2024a;b) have also become popular, thanks to their high efficiency and competitive performance. However, these methods also have some limitations, such as the inability to flexibly handle context length, and they need to retrain the model when using historical inputs of different lengths. They also lack generalizability across different tasks. Moreover, due to the lack of pre-trained knowledge, these methods typically require longer training times to converge. In contrast, pretrained Large time series models (LTSM) exhibit characteristics similar to large language models, such as flexible context length, scalability, and task generality, showing potential to outperform the task-specific models Liu et al. (2024); Das et al. (2024); Goswami et al. (2024).

Large time series models (LTSM). Existing efforts toward LTSMs can be categorized into two groups, with one being large language models for time series. FPT Zhou et al. (2023) partially fine-tunes GPT-2 on different downstream tasks. LLM4TS Chang et al. (2023) encodes time series into numerical tokens to utilize LLMs for time series forecasting. TimeLLM Jin et al. aligns the text prompt with time series to enhance prediction. These methods demonstrate the potential of LLMs for time series analysis. **Another category** includes pre-trained models on large-scale time series. Moirai Woo et al. (2024b), an encoder-only architecture, transforms time series into varied token sizes for better handling varied frequencies and then performs the pre-training strategy of mask modeling for time series forecasting. MOMENT Goswami et al. (2024), an encoder-decoder architecture, adopts a BERT-style mask modeling pre-training strategy and supports various downstream time series tasks. TimesFM Das et al. (2024) is a decoder-only Transformer pre-trained on Google Trends for forecasting, exhibiting notable zero-shot ability. Timer Liu et al. (2024) conducts GPT-style pre-training on the carefully processed and collected UTSD dataset and has achieved advanced accuracy on various tasks, including forecasting, imputation, and anomaly detection.

A.4 Pytorch codes to smooth the loss landscape of the pretrained LTSM

We show the example codes to use the randomly initialized LTSM to smooth the loss landscape of the pretrained one in Algorithm 1. By combining the strengths of the randomly initialized LTSM (good trainability with a smoother loss landscape) and the pretrained LTSM (good pretrained knowledge), the convergence of the smoothed LTSM can be improved during fine-tuning.

Algorithm 1: Smoothing the loss landscape of the pre-trained LTSM for fine-tuning

```
def Smoothing_Landscape (model1,model2):
    """model1: pre-trained LTSM, model2: randomly initialized LTSM"""
    for param1, param2 in zip(model1.parameters(), model2.parameters()):
        # Smoothing the loss landscape of model1 through interpolation
        model1.copy_((model1 * alpha + model2 * (1 - alpha)))
    # Next, the smoothed model1 is applied for fine-tuning
    # without increasing memory and computational overhead
    return model1
```

A.5 More details about reproducing paper results

Datasets. In forecasting and imputation, we conduct extensive experiments on eight well-known datasets, including Exchange rate, Weather, Electricity, Traffic, and four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2). Details can be seen in Appendix A.6 and Table 9. **In the anomaly detection task,** following previous work Liu et al. (2024); Wu & Keogh (2021), we use UCR Anomaly Archive (containing 250 datasets) Wu & Keogh (2021) for anomaly detection.

Evaluations. Following Timer Liu et al. (2024), we uniformly use MSE (Mean Squared Error) and MAE (Mean Absolute Error) to evaluate the performance of methods on forecasting, imputation, and anomaly detection tasks. **In forecasting**, we investigate the performance of the proposed *smoothed fine-tuning* under different proportions of available fine-tuning data. The proportions range from 1% to 100%. **In anomaly detection**, similar to Timer Liu et al. (2024), MSE is used as a confidence level to evaluate the effectiveness of anomaly detection. The higher the predicted MSE of the anomalous segments, the better, as this reduces the risk of normal segments being misjudged as anomalies.

Implementation details. The interpolation coefficient α is selected from 0.3, 0.5, 0.7, 0.9 for all tasks. For fairness, we use the source codes of each baseline and follow their recommended settings. Within each baseline, the configurations for "baseline" and "baseline-finetuning" are kept identical, ensuring that any performance change is attributable solely to the fine-tuning method (e.g., direct full FT vs. smoothed full FT). Settings may differ across baselines due to their public implementations, so accuracies between baselines are not directly comparable. When the recommended input length causes out-of-memory issues (e.g., Sundial), we reduce it while still keeping "baseline" and "baseline-finetuning" consistent. All experiments run four random seeds with NVIDIA 3090 GPUs using PyTorch, reporting the mean and standard deviation.

Following Liu et al. (2024), the input and prediction lengths of Timer are fixed at 672 and 96. Based on the limited computing resources and settings supported by each model, the input lengths for MOMENT and TimesFM are 512 and 256, while the forecast lengths are 96, and 128, respectively. Following Timer Liu et al. (2024), the fine-tuning epochs are fixed at 10, and we report the best metric in all epochs. The learning rate is 3e-5 and the optimizer is Adam. More details can be found in our source code.

When implementing the baseline LTSMs, we download the weights from their official links, e.g., which are listed as follows:

- Timer's codes and pre-trained weights can be downloaded from the links[1][2].
- TimesFM's codes and pre-trained weights can be downloaded from the links[3][4]. The model weight path on Hugging Face is "google/timesfm-2.0-500m-pytorch". We adopt the latest 2.0 version.
- MOMENT's codes and pre-trained weights can be downloaded from the links[5][6]. The model weight path on Hugging Face is "AutonLab/MOMENT-1-large".

[1] https://github.com/thuml/Large-Time-Series-Model?tab=readme-ov-file
[2] https://drive.google.com/drive/folders/15oaiAl4OO5gFqZMJD2lOtX2fxHbpgcU8
[3] https://github.com/google-research/timesfm
[4] https://huggingface.co/google/timesfm-2.0-500m-pytorch
[5] https://github.com/moment-timeseries-foundation-model/moment-research
[6] https://huggingface.co/AutonLab/MOMENT-1-large

A.6 More details about public datasets

The statistics of the eight well-known datasets are shown in Table 9, covering a range of variables and sampling frequency. These datasets involve applications in industrial machines, energy, and weather domains. They have been widely employed in the literature for time series analysis tasks Nie et al. (2023); Liu et al. (2024); Goswami et al. (2024); Woo et al. (2024b); Zhou et al. (2023); Chang et al. (2023); Jin et al..

Table 9: Statistics of eleven public datasets. *Data size* denotes the number of samples in train, validation, and test set for the single **variable**. In the experiment, each variable is separately split to construct samples and then merged, so the total number of samples needs to be multiplied by the number of variables. *Frequency* denotes the sampling interval of time points.

Datasets	Variable	Data size (single variable)	Frequency
ETTh1,ETTh2	7	(8545, 2881, 2881)	Hourly
ETTm1,ETTm2	7	(34465, 11521, 11521)	15min
Weather	21	(36792, 5271, 10540)	10min
Exchange rate	9	(5120, 665, 1422)	Daily
Electricity	321	(18317, 2633, 5261)	Hourly
Traffic	862	(12185, 1757, 3509)	Hourly

These datasets used in this paper are extensively used for TSF algorithm evaluation, including exchange rate forecasting in the financial field, electricity consumption forecasting in the energy field, climate parameter forecasting in the weather domain, and machine parameter (e.g., loads and oil temperature) forecasting in the industrial field:

- Electricity dataset[7] collects the electricity consumption (kWh) every 15 minutes of 321 clients from 2012 to 2014.
- ETT dataset[8] comprises two sub-datasets, ETT1 and ETT2, collected from two separate counties. Each sub-dataset offers two versions with varying sampling resolutions (15 minutes and 1 hour). ETT dataset includes multiple time series of electrical loads and a single time sequence of oil temperature.
- Weather dataset[9] contains 21 meteorological indicators, such as air temperature, humidity, etc, recorded every 10 minutes for the entirety of 2020.
- Traffic[10] dataset contains the occupation rate of freeway systems in California, USA. 5).

All datasets can be downloaded from the link[11].

A.7 Additional experiment results and discussions

The experiments in the appendix serve as supplements to those in the main paper, including complete standard deviations, MAE results, and interpolation experiments. All figures and tables in the appendix have been appropriately linked and referenced in the main paper. They can be located by clicking the hyperlinks in the main paper while reading it.

We hope that these extensive experiments can help demonstrate that directly fine-tuning pre-trained LTSMs may indeed lead to limited performance, as they may overfit during pre-training, resulting in a steep and unsmoothed loss landscape and poor trainability, thereby degrading and limiting the fine-tuning performance of pre-trained LTSMs on downstream tasks. Meanwhile, Our proposed *smoothed finetuning* can indeed help pre-trained LTSMs achieve better fine-tuning performance on downstream tasks.

[7] https://archive.ics.uci.edu/dataset/321/electricity
[8] https://github.com/zhouhaoyi/Informer2020
[9] https://www.bgc-jena.mpg.de/wetter/
[10] http://pems.dot.ca.gov
[11] https://drive.google.com/drive/folders/1ZOYpTUa82_jCcxIdTmyr0LXQfvaM9vIy

A.7.1 Comparisons between SFF and more baselines

Comparison with LoRA. We additionally add LoRA fine-tuning as a baseline to highlight the contribution of our method. Since LTSM is mostly < 1 B parameters, we follow the official recommendation and set the low-rank factor r = 8. As reported in Table 10, our approach outperforms LoRA. This is reasonable: LoRA trades full fine-tuning for a low-rank constraint, achieving appealing parameter efficiency, yet this restriction can limit the model's fine-tuning capacity.

Comparison with popular optimization strategies. We further compare our method with several widely used optimization strategies during training. As shown in Table 10, while some of these approaches offer modest improvements over standard full fine-tuning, their performance still falls far short of that achieved by our proposed SFF fine-tuning. The key limitation is that they don't address the underlying problem—namely, the highly steep and non-smooth loss landscape of the pre-trained model. In contrast, SFF explicitly mitigates this problem by first smoothing the landscape and then fine-tuning, resulting in substantially stronger fine-tuning and adaptation performance.

Influence of different parameter initialization schemes. We have conducted ablations with several perturbation-based smoothing strategies: standard Gaussian noise (mean = 0, variance = 1), Xavier Gaussian, Xavier uniform, Kaiming Gaussian, and Kaiming uniform. Table 11 shows that Xavier- and Kaiming-based schemes maintain stable performance improvements. Because they consider the stable gradient variance and can supply a flat loss landscape (demonstrated by (Fort & Scherlis, 2019)) that is used to smooth the sharp landscape of the pre-trained model for better fine-tuning effect, which is also aligned with our Theoretical analysis. In contrast, standard Gaussian initialization—lacking variance control—often pushes parameters into sharper regions of the landscape, resulting in weaker smoothing and noticeably degraded downstream performance. These findings provide strong empirical evidence supporting our design choice.

Influence of random seeds on parameter initializations. To assess sensitivity, we further evaluate SFF under different random seeds while using widely adopted initialization schemes (e.g., Kaiming uniform). The results in Table 12 indicate that improved performance remains highly stable across seeds. This suggests that SFF does not rely on a carefully engineered initialization. Instead, the mainstream initialization strategy suffices to obtain consistent smoothing and fine-tuning gains. This is reasonable because the prior work (Fort & Scherlis, 2019) has proven that the underlying design of mainstream initialization methods ensures the initialized parameters indeed lie in the flat region of the loss landscape, without being influenced by the random states (seeds). We believe this robustness is a desirable property for practical deployment.

Table 10: Comparison with more baselines on the LTSM Timer. We independently run four times with four random seeds to enhance the solidity of the results and report the mean value and standard deviation.

	Exchange	ETTh1	ETTh2	ETTm1	ETTm2	Weather
Original full fine-tuning	0.09±0.0007	0.367±0.0027	0.304±0.0049	0.312±0.0008	0.176±0.0013	0.158±0.0012
LoRA	0.122±0.0003	0.418±0.0005	0.304±0.0006	0.401±0.0019	0.197±0.0001	0.155±0.0004
Label-smoothing	0.09±0.0018	0.364±0.0036	0.303±0.0043	0.312±0.0011	0.177±0.0013	0.158±0.0008
SAM	0.088±0.0016	0.362±0.003	0.296±0.0027	0.309±0.0002	0.175±0.0017	0.157±0.0
SWA	0.094±0.0017	0.366±0.0024	0.304±0.0045	0.319±0.0009	0.178±0.0014	0.162±0.0005
MixOut	0.09±0.0003	0.376±0.0006	0.297±0.0001	0.348±0.0018	0.184±0.0006	0.16±0.0007
L2-SP	0.09±0.0007	0.368±0.0031	0.304±0.0051	0.315±0.0007	0.177±0.0013	0.16±0.0004
Ours (SFF)	**0.081**±0.0008	**0.355**±0.0013	**0.274**±0.0008	**0.297**±0.0016	**0.161**±0.0009	**0.145**±0.0006

Table 11: The effectiveness of our SFF (smoothed full fine-tuning) across different parameter initialization schemes on the LTSM Timer.

	Exchange	ETTh1	ETTh2	ETTm1	ETTm2	Weather
Original full fine-tuning	0.09±0.0007	0.367±0.0027	0.304±0.0049	0.312±0.0008	0.176±0.0013	0.158±0.0012
Standard Gaussian perturbation-SFF	5.986±0.261	0.723±0.003	1.879±0.275	3.876±0.128	19.031±0.963	0.447±0.03
Kaiming Normal Distribution-SFF	0.081±0.0006	0.353±0.0009	0.277±0.001	0.299±0.0011	0.164±0.0016	0.146±0.0008
Kaiming Uniform Distribution-SFF	0.081±0.0008	0.355±0.0013	0.274±0.0008	0.297±0.0016	0.161±0.0009	0.145±0.0006
Xavier Normal Distribution-SFF	0.081±0.0007	0.353±0.001	0.276±0.0012	0.3±0.0009	0.162±0.0001	0.145±0.0002
Xavier Uniform Distribution-SFF	0.082±0.0008	0.353±0.0007	0.277±0.0006	0.3±0.0003	0.162±0.0001	0.145±0.0002

Table 12: Experiments with four different random seeds (r1, r2,r3, and r4 here) on the LTSM Timer. The results show that our SFF (smoothed full fine-tuning) is insensitive to the choice of random initialization distribution. FF denotes original full fine-tuning.

	Exchange	ETTh1	ETTh2	ETTm1	ETTm2	Weather
SFF (Ours)-r1	0.07996	0.3547	0.27379	0.29542	0.16003	0.14605
FF-r1	0.08937	0.36941	0.31021	0.31134	0.17645	0.16067
SFF (Ours)-r2	0.08182	0.35772	0.27368	0.29902	0.16259	0.14432
FF-r2	0.09071	0.36245	0.3035	0.31282	0.17843	0.15912
SFF (Ours)-r3	0.08101	0.35766	0.27453	0.29824	0.16129	0.14481
FF-r3	0.09102	0.36848	0.29699	0.31167	0.17515	0.15851
SFF (Ours)-r4	0.08191	0.35588	0.27571	0.29938	0.16173	0.14525
FF-r4	0.08978	0.36815	0.30666	0.31057	0.17546	0.15649

A.7.2 Guidance for selecting α

In practice, we suggest the following guidance to select α:

(1) Empirically recommended values: As illustrated in Figures 5 and 6 of the manuscript, although different values introduce some variation, the sensitivity analysis demonstrates that SFF consistently outperforms vanilla fine-tuning across a broad range. Specifically, performs best for zero-shot prediction, while yields the strongest overall performance under full fine-tuning. These values can thus serve as reliable initial starting points.

(2) Validation-based tuning: We observe that the trend of test performance with respect to closely mirrors that on the validation set. Therefore, once a candidate search range is defined, selecting the that minimizes validation error provides a straightforward and computationally efficient strategy.

(3) Data-driven selection: Automatically learning is indeed a promising direction. In the current framework, however, the interpolation weights are fixed prior to fine-tuning, which makes adaptive selection non-trivial. Approaches such as meta-learning could potentially be explored to determine optimal values across models and datasets. We regard this as an important avenue for future research.

A.7.3 Influence of SFF on normalization or scale between layers

We discuss this influence in two scenarios:

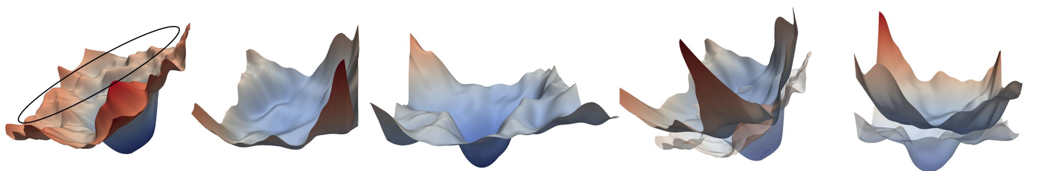
(1) Fine-tuning after loss landscape smoothing: When weights are smoothed, and the model is subsequently fine-tuned, the potential mismatch in normalization or scale is negligible. The model is free to update the relevant parameters during fine-tuning, effectively correcting any minor discrepancies introduced by interpolation.

(2) Zero-shot forecasting after loss landscape smoothing: First, the random initializations used for smoothing follow standard schemes (e.g., Kaiming, Xavier), whose typical scales are consistent with the learnable scale parameters in normalization layers. Second, the "flat" and "sharp" minima we analyze encompass the entire parameter space, including the weight matrices of normalization layers. Consequently, in theory, our method does not introduce significant scale or alignment inconsistencies. Instead, it smooths sharp minima located in suboptimal regions without harming flat minima, effectively relocating the sharp minima of the normalization layers to more favorable convergence points and thereby achieving improved and more generalizable performance. Moreover, we have included formal theoretical derivations and analyses demonstrating that the interpolation strategy improves sharp minima while not harming flat minima. For details, please refer to lines 182 to 257 and lines 772 to 804 of the revised manuscript.

Moreover, empirically, as shown in Tables 6 and 7 (after correcting the minor numerical ordering oversight), zero-shot forecasting following weight smoothing consistently demonstrates accuracy improvements, supporting the theoretical reasoning outlined above.

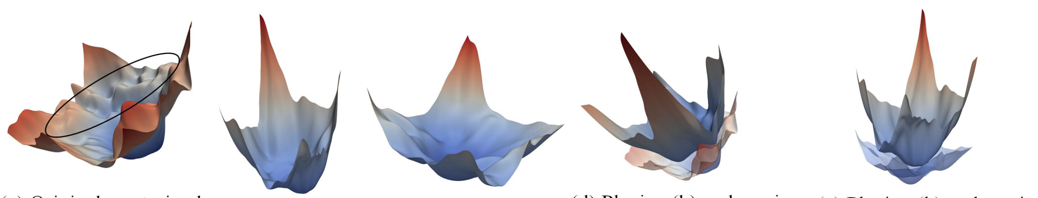
A.7.4 More cases about the loss landscape of the pretrained LTSM (Figure 10 and Figure 8)

As shown in Figure 10 in main paper and Figure 7 and Figure 8 in the Appendix, we visualize the loss landscape of different datasets and empirically find that LTSMs initialized randomly typically have a smooth loss landscape. In contrast, pre-trained LTSMs consistently exhibit steep and non-smooth loss landscapes, indicating that this is not a random occurrence. Hence, after pre-training, LTSMs may indeed show lower trainability, which can affect their fine-tuning performance on downstream tasks.



(a) Original pre-trained LTSM (b) Random Initialization (c) Utilizing (b) to smooth (a) (d) Placing (b) and semi-transparent (a) in the same coordinate system (e) Placing (b) and semi-transparent (c) in the same coordinate system

Figure 7: Loss landscape comparisons based on the LTSM Timer and weather dataset. The smoother the surface, the better.



(a) Original pre-trained LTSM (b) Random Initialization (c) Utilizing (b) to smooth (a) (d) Placing (b) and semi-transparent (a) in the same coordinate system (e) Placing (b) and semi-transparent (c) in the same coordinate system

Figure 8: Loss landscape comparisons based on the LTSM Timer and electricity dataset. The smoother the surface, the better.

A.7.5 Training loss and test loss during fine-tuning the pre-trained LTSM (Figure 9)

We also empirically observe severe overfitting during the fine-tuning of pre-trained LTSM on downstream tasks, which is consistent with our analysis of the loss landscape. A steep and non-smooth loss landscape may cause the model to fall into poor local optima, leading to severe overfitting Li et al. (2018). Specifically, as shown in Figure 9, the training loss of directly fine-tuning the Timer (green lines) is significantly the lowest. However, the test MSE of the fine-tuned Timer (green bars) is even significantly worse than that of training from scratch on the Timer (black bars) on the downstream datasets (e.g., ETTh1, ETTm1, and Weather) without pre-training. This suggests that directly fine-tuning Timer leads to severe overfitting Hastie (2009), which causes pre-trained knowledge of it not to be fully utilized for improving the accuracy of downstream tasks.

A.7.6 Applying *smoothed fine-tuning* for time series imputation task (Figure 10)

As shown in Figure 10, our method also shows improvement for the imputation task. This indicates that the poor trainability of the LTSM, caused by overfitting during the pre-training phase, may impact their performance on various downstream time series tasks, including forecasting, anomaly detection, and imputation tasks. Our proposed method offers a potential solution to address this issue and provides a new perspective for fine-tuning the pretrained LTSMs.

A.7.7 Summary of the content in the following sections

The experimental results in the following sections serve as supplements to the experiments in the main paper regarding complete standard deviations and MAE results under different available data proportions. Through multiple experiments with different random seeds, we ensure that our proposed fine-tuning method indeed helps the LTSM achieve better fine-tuning performance and that this is not a random occurrence. Moreover, our method consistently outperforms other fine-tuning methods in terms of the MAE metric and across different data proportions, which further demonstrates the effectiveness of our approach. Our work provides new insights for fine-tuning large models. In the

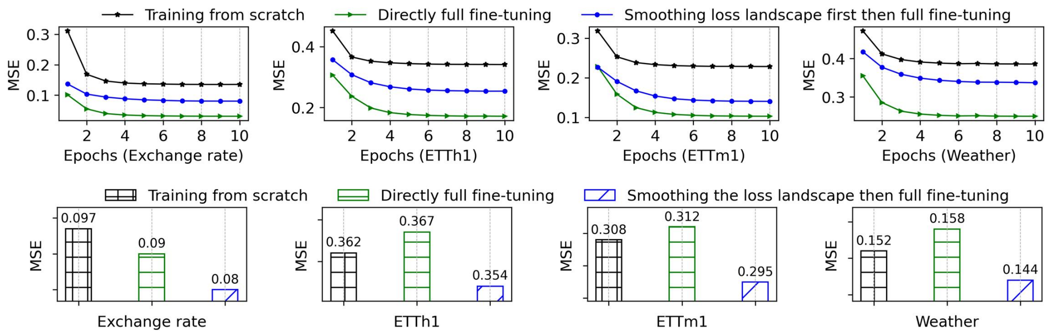


Figure 9: Time series forecasting of the LTSM Timer on various datasets with 100% available data proportion. We show the comparisons of training and testing losses for training from scratch on (black lines and bars), direct full fine-tuning (green lines and bars), and smoothing the loss landscape then full fine-tuning (blue lines and bars).

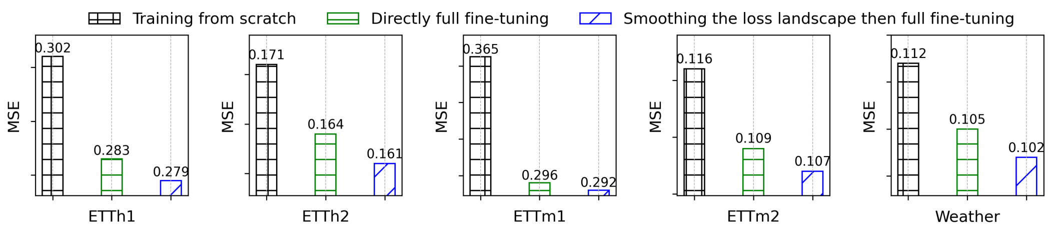


Figure 10: Time series imputation task on Timer with the mask ratio 25%. The experimental settings follow the Timer's codes: `https://github.com/thuml/Large-Time-Series-Model?tab=readme-ov-file`.

subsequent content, we provide relevant titles (in a table of contents format) for reference to the supplementary experimental results without further redundant explanations:

- Fintuning MSE with prediction length 720 for more LTSMs (Table 13).
- Zero-shot forecasting MSE with prediction length 720 for more LTSMs (Table 14).
- Forecasting MSE and standard deviations under 1%, 2%, 3% and 4% proportion of available data (Table 15).
- Forecasting MSE and standard deviations under 5%, 10%, 15% and 20% proportion of available data (Table 16).
- Forecasting MSE and standard deviations under 25%, 50%, 75% and 100% proportion of available data (Table 17).
- Forecasting MAE and standard deviations under 1%, 2%, 3% and 4% proportion of available data (Table 18).
- Forecasting MAE and standard deviations under 5%, 10%, 15% and 20% proportion of available data (Table 19).
- Forecasting MAE and standard deviations under 25%, 50%, 75% and 100% proportion of available data (Table 20).
- Complete MSE of anomaly detection results on 250 datasets (Table 21).
- Complete standard deviations of anomaly detection results on 250 datasets (Table 22).
- Complete MSE and standard deviations of applying the pretrained LTSMs TimesFM and MOMENT for time series forecasting (Table 23).
- Complete MAE and standard deviations of applying the pretrained LTSMs TimesFM and MOMENT for time series forecasting (Table 24).

- Complete MSE and standard deviations of applying other fine-tuning methods, LP and LPFF, under grouped data proportions for time series forecasting (Table 25).
- Complete MAE and standard deviations of applying other fine-tuning methods, LP and LPFF, under grouped data proportions for time series forecasting (Table 26).
- Complete MSE and standard deviations of applying other fine-tuning methods, LP and LPFF, under 1%, 2%, 3%, and 4% proportion of available data for time series forecasting (Table 27).
- Complete MSE and standard deviations of applying other fine-tuning methods, LP and LPFF, under 5%, 10%, 15%, and 20% proportion of available data for time series forecasting (Table 28).
- Complete MSE and standard deviations of applying other fine-tuning methods, LP and LPFF, under 25%, 50%, 75%, and 100% proportion of available data for time series forecasting (Table 29).
- Complete MAE and standard deviations of applying other fine-tuning methods, LP and LPFF, under 1%, 2%, 3%, and 4% proportion of available data for time series forecasting (Table 30).
- Complete MAE and standard deviations of applying other fine-tuning methods, LP and LPFF, under 5%, 10%, 15%, and 20% proportion of available data for time series forecasting (Table 31).
- Complete MAE and standard deviations of applying other fine-tuning methods, LP and LPFF, under 25%, 50%, 75%, and 100% proportion of available data for time series forecasting (Table 32).
- Complete MAE and standard deviations of zero-shot forecasting after smoothing the loss landscape (Table 33).

Table 13: MSE of fine-tuning more LTSMs for the TSF task with prediction length 720. SFF, and FF are *smoothed full fine-tuning* and *full fine-tuning*.

	UniTS-SFF	UniTS-FF	MOIRAI-SFF	MOIRAI-FF	Chronos-SFF	Chronos-FF	TTMs-SFF	TTMs-FF	Sundial-SFF	Sundial-FF
ETTh1	**0.704**	0.741	**0.582**	0.634	-	-	**0.421**	0.424	**0.485**	0.49
ETTh2	**0.431**	0.436	**0.582**	0.634	-	-	**0.402**	0.407	**0.404**	0.41
ETTm1	**0.496**	0.625	**0.402**	0.451	-	-	**0.433**	0.435	**0.552**	0.562
ETTm2	**0.416**	0.419	**0.361**	0.356	-	-	**0.374**	0.376	**0.376**	0.387
Weather	**0.324**	0.346	**0.319**	0.33	-	-	**0.328**	0.328	**0.36**	0.362
Elect.	**0.355**	0.495	**0.843**	0.999	-	-	**0.241**	0.24	-	-
Traffic	**1.01**	1.307	**0.554**	0.576	-	-	**0.612**	0.611	-	-

Table 14: Zero-shot forecasting MSE of more LTSMs with prediction length 720. "-" indicates that the preprocessed dataset is not included (Chronos) or out of memory (Sundial).

	MOIRAI	+Smooth	Chronos	+Smooth	TTMs	+Smooth	Sundial	+Smooth
ETTh1	**0.439**	0.447	-	-	**0.421**	0.424	**0.437**	0.453
ETTh2	**0.386**	0.4	-	-	**0.404**	0.408	**0.409**	0.417
ETTm1	**0.601**	0.603	-	-	**0.435**	0.439	**0.409**	0.417
ETTm2	**0.407**	0.438	-	-	**0.409**	0.41	**0.397**	0.408
Weather	**0.399**	0.424	-	-	**0.328**	0.328	**0.354**	0.357
Elect.	**0.253**	0.255	-	-	**0.257**	0.255	-	-
Traffic	**0.62**	0.622	-	-	**0.624**	0.622	-	-

Table 15: MSE of fine-tuning LTSM Timer for time series forecasting under **1%, 2%, 3% and 4%** proportion of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively.

Data proportion	1%			2%			3%			4%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.0853**	0.0887	0.2831	**0.0858**	0.0884	0.2808	**0.0842**	0.0876	0.2803	**0.085**	0.0879	0.2796
Standard deviation	±3.2e-4	±9.0e-4	±7.2e-3	±4.2e-4	±5.5e-4	±6.8e-3	±5.8e-4	±8.1e-4	±6.8e-3	±2.4e-4	±6.2e-4	±6.7e-3
ETTh1	**0.3649**	0.3873	0.5631	**0.3637**	0.3861	0.5614	**0.3608**	0.3846	0.5588	**0.3606**	0.3823	0.545
Standard deviation	±6.3e-3	±1.2e-4	±9.8e-3	±6.6e-3	±3.6e-4	±1.1e-2	±6.6e-3	±7.6e-4	±1.1e-2	±7.5e-3	±1.4e-3	±4.7e-3
ETTh2	**0.2825**	0.2945	0.363	**0.2747**	0.2893	0.362	**0.2772**	0.2896	0.3617	**0.2752**	0.2872	0.3434
Standard deviation	±1.0e-3	±4.4e-4	±2.0e-3	±1.7e-3	±1.5e-3	±2.3e-3	±1.9e-3	±7.0e-4	±2.7e-3	±2.7e-3	±2.1e-3	±2.3e-3
ETTm1	**0.364**	0.3826	0.5342	**0.3304**	0.3484	0.4193	**0.3252**	0.3418	0.4159	**0.3179**	0.3312	0.3884
Standard deviation	±4.4e-3	±1.3e-3	±1.1e-2	±2.7e-3	±1.0e-3	±3.0e-3	±3.4e-3	±8.0e-4	±3.2e-3	±3.5e-3	±7.9e-4	±2.3e-3
ETTm2	**0.1709**	0.1897	0.2531	**0.173**	0.184	0.2524	**0.1635**	0.1758	0.2157	**0.1654**	0.1755	0.216
Standard deviation	±2.4e-3	±1.7e-3	±3.2e-3	±2.1e-3	±7.4e-4	±3.4e-3	±2.4e-3	±3.4e-4	±1.7e-3	±1.4e-3	±6.1e-4	±1.5e-3
Weather	**0.1537**	0.1564	0.2403	**0.1505**	0.153	0.2259	**0.1489**	0.1525	0.2162	**0.147**	0.1492	0.2111
Standard deviation	±3.9e-4	±6.5e-4	±3.0e-3	±1.5e-4	±9.1e-4	±2.3e-3	±8.7e-4	±1.2e-3	±5.7e-4	±2.9e-4	±8.2e-4	±5.0e-4
Electricity	**0.1369**	0.1393	0.2285	**0.1342**	0.1367	0.2035	**0.133**	0.1356	0.1857	**0.1331**	0.1358	0.1708
Standard deviation	±7.8e-5	±7.6e-4	±1.5e-3	±1.9e-4	±7.1e-4	±1.4e-3	±3.3e-4	±6.9e-4	±1.3e-3	±2.5e-4	±7.4e-4	±6.4e-4
Traffic	**0.3743**	0.3768	0.5803	**0.3671**	0.3698	0.4794	**0.3622**	0.365	0.4388	**0.3594**	0.3623	0.4199
Standard deviation	±2.6e-4	±6.8e-4	±4.1e-3	±1.0e-4	±9.9e-4	±2.4e-3	±2.4e-4	±7.9e-4	±8.0e-4	±2.3e-4	±9.9e-4	±3.2e-4
Avg. Improvements	-	4.07%	37.9	-	3.64%	34.04	-	3.91%	31.47	-	3.4%	28.96
Max. Improvements	-	9.91%	69.87	-	5.98%	69.44	-	7.0%	69.96	-	5.75%	69.6

Table 16: MSE of fine-tuning LTSM Timer for time series forecasting under **5%, 10%, 15% and 20%** proportion of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively.

Data proportion	5%			10%			15%			20%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.0854**	0.0883	0.2721	**0.0829**	0.0854	0.1919	**0.0815**	0.0845	0.1715	**0.0805**	0.0858	0.157
Standard deviation	±1.6e-4	±8.4e-4	±6.4e-3	±7.2e-4	±6.9e-4	±1.7e-3	±3.1e-4	±9.4e-4	±9.1e-4	±1.1e-3	±1.4e-3	±2.5e-3
ETTh1	**0.3582**	0.3745	0.4509	**0.3539**	0.3654	0.4162	**0.3528**	0.3615	0.3963	**0.3483**	0.3565	0.382
Standard deviation	±4.6e-3	±7.7e-4	±3.2e-3	±2.1e-3	±1.0e-3	±1.2e-3	±1.7e-3	±1.2e-3	±1.3e-3	±1.9e-3	±4.6e-4	±1.2e-3
ETTh2	**0.272**	0.2866	0.3288	**0.2757**	0.2855	0.3157	**0.2728**	0.2854	0.3038	**0.2765**	0.2865	0.2929
Standard deviation	±3.5e-3	±1.4e-3	±1.2e-3	±2.2e-3	±6.2e-4	±7.0e-4	±3.8e-3	±1.4e-3	±1.3e-3	±1.5e-3	±1.1e-3	±3.7e-4
ETTm1	**0.3152**	0.3273	0.385	**0.3046**	0.3115	0.3532	**0.3016**	0.3067	0.3428	**0.2992**	0.3059	0.3378
Standard deviation	±2.6e-3	±9.4e-4	±2.7e-3	±1.7e-3	±8.7e-4	±1.3e-3	±1.0e-3	±6.1e-4	±1.1e-3	±1.1e-3	±7.1e-4	±1.1e-3
ETTm2	**0.1637**	0.1745	0.2151	**0.1614**	0.1714	0.1943	**0.1614**	0.1729	0.1846	**0.162**	0.1763	0.1793
Standard deviation	±2.9e-3	±9.4e-4	±1.4e-3	±1.8e-3	±4.9e-4	±5.6e-4	±1.3e-3	±1.1e-3	±2.9e-4	±7.8e-4	±6.7e-4	±3.7e-4
Weather	**0.1459**	0.1478	0.2024	**0.1446**	0.1464	0.1852	**0.1442**	0.1489	0.1739	**0.1442**	0.1466	0.166
Standard deviation	±1.6e-4	±6.5e-4	±3.6e-4	±1.2e-4	±4.9e-4	±2.0e-4	±1.2e-4	±1.2e-3	±8.3e-5	±2.9e-5	±2.1e-4	±2.6e-4
Electricity	**0.1319**	0.1346	0.1621	**0.1309**	0.1338	0.1461	**0.1309**	0.1342	0.1411	**0.1306**	0.1345	0.1378
Standard deviation	±2.6e-4	±7.0e-4	±4.4e-4	±1.8e-4	±5.2e-4	±2.0e-4	±2.3e-4	±5.5e-4	±1.4e-4	±1.3e-4	±7.8e-4	±8.5e-5
Traffic	**0.3574**	0.3604	0.4095	**0.3518**	0.3582	0.3874	**0.3508**	0.3596	0.3788	**0.349**	0.3579	0.373
Standard deviation	±1.2e-4	±9.4e-4	±2.7e-4	±8.5e-4	±7.1e-4	±1.6e-4	±1.1e-3	±8.6e-4	±1.0e-4	±1.7e-3	±4.4e-4	±1.8e-4
Avg. Improvements	-	3.34%	25.97	-	2.84%	19.58	-	3.34%	16.24	-	3.66%	13.63
Max. Improvements	-	6.19%	68.61	-	5.83%	56.8	-	6.65%	52.48	-	8.11%	48.73

Table 17: MSE of fine-tuning LTSM Timer for time series forecasting under **25%, 50%, 75% and 100%** proportion of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively.

Data proportion	25%			50%			75%			100%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.0805**	0.0865	0.1441	**0.0802**	0.0891	0.114	**0.0802**	0.0914	0.1026	**0.08**	0.091	0.0981
Standard deviation	±4.5e-4	±1.9e-4	±2.0e-3	±5.4e-4	±2.3e-3	±9.9e-4	±1.2e-3	±1.6e-3	±8.8e-4	±7.6e-4	±1.3e-4	±1.2e-3
ETTh1	**0.3506**	0.355	0.3788	**0.3494**	0.3573	0.367	**0.3493**	0.358	0.3593	**0.3547**	0.3709	0.36
Standard deviation	±6.1e-4	±5.8e-4	±1.2e-3	±1.1e-3	±1.3e-3	±8.4e-4	±1.4e-3	±9.3e-4	±1.1e-3	±1.4e-3	±3.6e-3	±1.2e-3
ETTh2	**0.271**	0.2866	0.2891	**0.273**	0.2905	0.2775	**0.2772**	0.3042	0.2796	**0.2737**	0.3117	0.2777
Standard deviation	±2.5e-3	±1.6e-3	±3.6e-4	±2.0e-3	±8.4e-4	±2.9e-4	±4.8e-4	±5.2e-4	±7.6e-4	±3.8e-4	±6.0e-3	±1.6e-3
ETTm1	**0.298**	0.3049	0.333	**0.2955**	0.3069	0.3189	**0.2956**	0.3092	0.3116	**0.2954**	0.3128	0.3093
Standard deviation	±1.1e-3	±5.6e-4	±9.3e-4	±1.6e-3	±1.1e-3	±7.1e-4	±1.3e-3	±7.3e-4	±1.2e-3	±1.5e-3	±5.5e-4	±1.1e-3
ETTm2	**0.1594**	0.1707	0.1741	**0.1605**	0.1718	0.1627	**0.1623**	0.1838	0.1651	**0.16**	0.1784	0.1644
Standard deviation	±9.7e-4	±1.3e-3	±2.8e-4	±3.8e-4	±5.9e-4	±1.8e-4	±4.3e-4	±2.5e-3	±9.3e-4	±1.0e-3	±1.4e-3	±1.1e-3
Weather	**0.144**	0.1472	0.1627	**0.1441**	0.1523	0.1538	**0.1466**	0.1665	0.1559	**0.1443**	0.1612	0.1526
Standard deviation	±5.2e-5	±6.1e-4	±5.7e-5	±2.1e-4	±7.4e-4	±4.2e-4	±1.3e-4	±1.5e-3	±1.1e-3	±7.3e-4	±1.6e-3	±9.2e-4
Electricity	**0.1303**	0.1344	0.1365	**0.1301**	0.1347	0.1327	**0.13**	0.1367	0.1326	**0.1304**	0.1344	0.1324
Standard deviation	±1.7e-4	±9.4e-4	±5.7e-5	±2.4e-4	±9.5e-4	±1.3e-4	±3.6e-4	±5.8e-4	±6.5e-4	±2.0e-4	±5.4e-4	±8.0e-4
Traffic	**0.3488**	0.3582	0.3688	**0.3497**	0.3586	0.3552	**0.3478**	0.361	0.3606	**0.3551**	0.3599	0.3609
Standard deviation	±2.0e-3	±9.2e-4	±2.0e-4	±1.5e-3	±5.8e-4	±1.5e-4	±3.2e-3	±1.1e-3	±3.0e-3	±2.7e-4	±2.2e-4	±2.5e-3
Avg. Improvements	-	3.79%	12.28	-	4.97%	6.82	-	7.52%	5.47	-	7.41%	4.64
Max. Improvements	-	6.94%	44.14	-	9.99%	29.65	-	12.25%	21.83	-	12.19%	18.45

Table 18: MAE of fine-tuning LTSM Timer for time series forecasting under **1%, 2%, 3% and 4%** proportion of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively.

Data proportion	1%			2%			3%			4%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.2048**	0.208	0.3946	**0.2045**	0.2083	0.3928	**0.2036**	0.2067	0.3923	**0.2038**	0.2071	0.3919
Standard deviation	±2.3e-4	±3.9e-4	±5.3e-3	±4.8e-4	±5.4e-4	±5.1e-3	±3.5e-4	±1.4e-4	±5.1e-3	±4.3e-4	±2.0e-4	±5.0e-3
ETTh1	**0.3973**	0.4101	0.5202	**0.3962**	0.4085	0.5184	**0.3943**	0.4072	0.5168	**0.3939**	0.4044	0.5055
Standard deviation	±3.3e-3	±1.6e-4	±4.5e-3	±3.4e-3	±4.5e-5	±5.2e-3	±3.5e-3	±6.2e-5	±5.1e-3	±5.3e-3	±4.7e-4	±4.1e-3
ETTh2	**0.3362**	0.3433	0.4109	**0.3319**	0.3401	0.41	**0.3322**	0.3394	0.4098	**0.3339**	0.3392	0.3944
Standard deviation	±2.0e-3	±1.1e-4	±1.8e-3	±3.2e-4	±5.4e-4	±2.0e-3	±5.2e-4	±6.5e-5	±2.4e-3	±5.2e-4	±1.8e-3	±2.7e-3
ETTm1	**0.4085**	0.4166	0.5138	**0.3838**	0.3952	0.4507	**0.3807**	0.3909	0.4488	**0.3755**	0.3824	0.4303
Standard deviation	±2.5e-3	±3.8e-4	±6.0e-3	±2.6e-3	±4.8e-4	±1.9e-3	±1.9e-3	±3.8e-4	±2.0e-3	±1.8e-3	±1.6e-4	±1.7e-3
ETTm2	**0.2586**	0.2738	0.3275	**0.2591**	0.2679	0.3271	**0.2523**	0.262	0.2979	**0.2541**	0.2609	0.2982
Standard deviation	±3.0e-3	±1.2e-3	±2.5e-3	±1.9e-3	±1.7e-4	±2.7e-3	±2.5e-3	±8.5e-4	±1.9e-3	±1.6e-3	±7.8e-5	±1.6e-3
Weather	**0.2028**	0.2062	0.2942	**0.1996**	0.2022	0.2808	**0.1978**	0.2015	0.2704	**0.1954**	0.1984	0.265
Standard deviation	±5.6e-4	±4.0e-4	±2.7e-3	±2.6e-4	±1.3e-4	±2.5e-3	±7.0e-4	±5.4e-4	±5.8e-4	±2.3e-4	±3.4e-5	±4.1e-4
Electricity	**0.2342**	0.2367	0.3243	**0.2307**	0.2334	0.2976	**0.229**	0.2315	0.2817	**0.2292**	0.2318	0.2702
Standard deviation	±8.3e-5	±6.8e-5	±1.4e-3	±3.4e-4	±8.8e-5	±1.2e-3	±5.0e-4	±5.1e-5	±9.6e-4	±2.7e-4	±8.6e-5	±5.5e-4
Traffic	**0.2672**	0.27	0.3921	**0.2611**	0.2636	0.3452	**0.2575**	0.26	0.3179	**0.2559**	0.2585	0.3037
Standard deviation	±2.0e-4	±2.4e-4	±1.7e-3	±2.1e-4	±1.9e-4	±1.4e-3	±6.8e-5	±7.4e-5	±5.0e-4	±9.9e-5	±2.5e-4	±2.0e-4
Avg. Improvements	-	2.25%	27.77	-	2.1%	25.24	-	2.12%	23.22	-	1.72%	21.26
Max. Improvements	-	5.55%	48.1	-	3.28%	47.94	-	3.7%	48.1	-	2.61%	48.0

Table 19: MAE of fine-tuning LTSM Timer for time series forecasting under **5%, 10%, 15% and 20%** proportion of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively.

Data proportion	5%			10%			15%			20%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.2044**	0.2078	0.3867	**0.2019**	0.2059	0.3252	**0.2009**	0.2047	0.3085	**0.2008**	0.2072	0.2936
Standard deviation	±2.0e-4	±1.7e-4	±4.9e-3	±9.0e-4	±1.8e-4	±1.7e-3	±6.1e-4	±1.8e-4	±1.2e-3	±1.0e-3	±1.5e-3	±2.9e-3
ETTh1	**0.3923**	0.4011	0.462	**0.3882**	0.3952	0.4422	**0.3884**	0.3946	0.4274	**0.3856**	0.3915	0.4173
Standard deviation	±3.4e-3	±3.8e-4	±1.5e-3	±1.8e-3	±1.8e-4	±6.8e-4	±1.5e-3	±5.2e-4	±1.3e-3	±1.4e-3	±6.1e-5	±1.1e-3
ETTh2	**0.3325**	0.3399	0.3826	**0.3327**	0.339	0.3686	**0.3348**	0.3385	0.3583	**0.3332**	0.3382	0.3512
Standard deviation	±1.2e-3	±9.5e-4	±1.9e-3	±2.3e-3	±9.5e-4	±9.4e-4	±1.7e-3	±8.9e-4	±2.1e-4	±2.6e-3	±7.9e-4	±1.9e-4
ETTm1	**0.3735**	0.3801	0.4283	**0.3663**	0.3706	0.4054	**0.3644**	0.3678	0.3967	**0.3631**	0.3679	0.3926
Standard deviation	±1.3e-3	±1.9e-4	±2.0e-3	±8.2e-4	±4.5e-5	±1.1e-3	±3.1e-4	±7.6e-5	±9.8e-4	±8.2e-4	±2.5e-5	±1.0e-3
ETTm2	**0.2522**	0.2599	0.2976	**0.2516**	0.2584	0.2787	**0.2532**	0.2576	0.2707	**0.2527**	0.2622	0.2672
Standard deviation	±1.6e-3	±8.6e-5	±1.6e-3	±1.3e-3	±5.0e-4	±6.1e-4	±5.0e-4	±2.9e-4	±2.0e-4	±4.1e-4	±4.0e-4	±2.5e-4
Weather	**0.1936**	0.1968	0.257	**0.1928**	0.1967	0.2385	**0.1928**	0.1978	0.2271	**0.1932**	0.1988	0.2198
Standard deviation	±2.3e-4	±2.2e-5	±5.3e-4	±1.4e-4	±5.3e-4	±7.3e-5	±1.3e-4	±6.9e-4	±2.9e-5	±3.0e-4	±1.2e-3	±3.8e-4
Electricity	**0.2273**	0.2303	0.2616	**0.2263**	0.2319	0.2444	**0.226**	0.2312	0.2384	**0.225**	0.23	0.2346
Standard deviation	±9.1e-5	±5.4e-5	±4.5e-4	±3.2e-4	±1.4e-3	±3.0e-4	±1.8e-4	±1.1e-3	±2.1e-4	±4.5e-4	±5.0e-4	±1.4e-4
Traffic	**0.2537**	0.2571	0.2948	**0.2501**	0.2554	0.2754	**0.2488**	0.2573	0.2678	**0.2473**	0.2573	0.2631
Standard deviation	±6.5e-5	±1.8e-4	±2.9e-4	±9.1e-4	±9.1e-5	±2.3e-4	±1.3e-3	±1.9e-3	±1.1e-4	±1.5e-3	±1.2e-3	±1.8e-4
Avg. Improvements	-	1.87%	19.39	-	1.98%	14.37	-	1.9%	11.57	-	2.48%	9.94
Max. Improvements	-	2.96%	47.14	-	2.63%	37.92	-	3.3%	34.88	-	3.89%	31.61

Table 20: MAE of fine-tuning LTSM Timer for time series forecasting under **25%, 50%, 75% and 100%** proportion of available data. SFF, FF, and TFS are *smoothed full fine-tuning*, *full fine-tuning*, and *training from scratch*, respectively.

Data proportion	25%			50%			75%			100%		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.1997**	0.2095	0.2782	**0.1993**	0.2122	0.2421	**0.2002**	0.2136	0.2266	**0.2006**	0.2153	0.2209
Standard deviation	±5.3e-4	±5.3e-4	±2.6e-3	±1.0e-3	±2.6e-3	±1.2e-3	±1.7e-3	±7.8e-4	±1.2e-4	±5.3e-4	±1.1e-3	±4.0e-4
ETTh1	**0.3879**	0.3902	0.4145	**0.388**	0.3905	0.404	**0.3858**	0.3907	0.3956	**0.3921**	0.3955	0.399
Standard deviation	±4.5e-4	±5.3e-4	±9.8e-4	±3.0e-4	±2.4e-4	±6.6e-4	±2.4e-3	±5.0e-4	±5.5e-4	±1.4e-3	±1.1e-3	±7.6e-4
ETTh2	**0.3325**	0.337	0.3467	**0.3327**	0.3421	0.3378	**0.3387**	0.3516	0.3435	**0.3353**	0.3531	0.3436
Standard deviation	±1.3e-3	±4.7e-4	±1.4e-4	±1.4e-3	±7.4e-4	±1.9e-4	±1.8e-4	±1.1e-3	±7.8e-4	±5.7e-4	±1.2e-3	±2.4e-3
ETTm1	**0.3599**	0.3656	0.3877	**0.3576**	0.3667	0.374	**0.3566**	0.3698	0.3696	**0.3558**	0.3703	0.3669
Standard deviation	±1.7e-3	±1.7e-4	±8.7e-4	±1.8e-3	±1.0e-4	±6.1e-4	±2.0e-3	±5.1e-4	±4.9e-4	±1.6e-3	±2.8e-4	±4.9e-4
ETTm2	**0.249**	0.2562	0.2626	**0.2497**	0.2566	0.253	**0.2534**	0.2635	0.2571	**0.2472**	0.2591	0.2533
Standard deviation	±6.4e-4	±4.6e-4	±2.4e-4	±5.8e-4	±5.6e-4	±1.6e-4	±3.8e-4	±1.5e-3	±6.4e-4	±1.5e-3	±8.1e-4	±5.7e-4
Weather	**0.1921**	0.1961	0.2146	**0.1929**	0.199	0.2037	**0.1971**	0.217	0.2098	**0.192**	0.2046	0.203
Standard deviation	±2.9e-4	±4.3e-4	±9.0e-5	±2.1e-4	±1.5e-3	±1.5e-4	±2.9e-4	±1.5e-3	±7.3e-4	±1.5e-3	±1.1e-3	±6.1e-4
Electricity	**0.2245**	0.2289	0.2328	**0.2247**	0.2273	0.228	**0.2245**	0.2292	0.2274	**0.2239**	0.2273	0.2268
Standard deviation	±4.2e-4	±8.4e-4	±1.5e-4	±2.7e-4	±2.3e-4	±9.0e-5	±4.3e-4	±4.4e-4	±2.1e-5	±4.2e-4	±3.3e-4	±2.2e-4
Traffic	**0.2463**	0.2545	0.2599	**0.2486**	0.2527	0.2512	**0.2485**	0.2538	0.2573	**0.2444**	0.2517	0.2553
Standard deviation	±1.9e-3	±1.8e-3	±1.5e-4	±1.4e-3	±1.3e-3	±1.6e-4	±4.6e-4	±9.0e-4	±2.6e-3	±3.0e-4	±2.0e-3	±1.3e-3
Avg. Improvements	-	2.27%	8.8	-	2.56%	4.58	-	3.99%	3.9	-	3.97%	3.72
Max. Improvements	-	4.68%	28.22	-	6.08%	17.68	-	9.17%	11.65	-	6.83%	9.19

Table 21: The complete anomaly detection results on 250 datasets, reporting the average MSE values of anomalous segments in each dataset under four random seeds, where higher values are better, because this reduces the risk of normal segments being misjudged as anomalies. The standard deviation of each dataset is shown in Table 22.

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1 (SFF)	**0.051**	**0.01**	**0.004**	0.435	**0.011**	**0.09**	**0.112**	**0.016**	**0.046**	**0.166**	0.129	**0.045**	**1.112**	**0.042**	**0.13**
1 (FF)	0.031	0.003	0.002	**0.706**	0.004	0.002	0.039	0.006	0.005	0.003	0.055	0.033	0.071	0.024	0.096
1 (TFS)	0.019	0.005	0.003	0.262	0.005	0.003	0.032	0.009	0.028	0.002	**0.139**	0.008	0.652	0.034	0.1
2 (SFF)	**0.107**	**0.077**	**0.002**	**0.164**	0.005	**0.002**	**0.093**	**0.004**	**0.082**	**0.065**	**0.071**	**0.053**	**0.27**	**0.324**	**0.043**
2 (FF)	0.012	0.024	0.0	0.11	0.005	0.0	0.036	0.002	0.024	0.038	0.069	0.013	0.231	0.213	0.023
2 (TFS)	0.012	0.026	0.001	0.103	**0.006**	0.001	0.055	0.002	0.043	0.049	0.02	0.02	0.13	0.232	0.028
3 (SFF)	**0.024**	**0.059**	**0.007**	**0.138**	**0.203**	**0.583**	**0.085**	**0.04**	**0.314**	**0.238**	**0.118**	**0.24**	**0.004**	**0.07**	**0.029**
3 (FF)	0.001	0.014	0.003	0.092	0.114	0.206	0.038	0.01	0.297	0.195	0.1	0.096	0.002	0.05	0.011
3 (TFS)	0.001	0.013	0.004	0.094	0.124	0.329	0.05	0.026	0.081	0.117	0.108	0.144	0.001	0.05	0.0
4 (SFF)	**0.121**	**0.32**	0.106	**0.085**	**0.092**	**0.348**	**0.512**	**0.129**	**0.54**	**0.122**	**0.027**	**0.413**	**0.36**	**0.033**	**0.006**
4 (FF)	0.026	0.095	0.073	0.013	0.016	0.062	0.259	0.003	0.223	0.013	0.026	0.396	0.352	0.005	0.003
4 (TFS)	0.087	0.184	**0.175**	0.031	0.079	0.095	0.323	0.005	0.213	0.012	0.015	0.212	0.218	0.004	0.003
5 (SFF)	**0.912**	**1.184**	**0.043**	**0.255**	**0.017**	**0.072**	**0.093**	**0.042**	**0.074**	**0.032**	**0.062**	**0.331**	**0.255**	0.058	**0.021**
5 (FF)	0.618	0.078	0.014	0.114	0.001	0.038	0.029	0.014	0.026	0.014	0.007	0.224	0.007	**0.063**	0.003
5 (TFS)	0.602	0.133	0.036	0.113	0.0	0.029	0.053	0.029	0.019	0.015	0.02	0.238	0.004	0.019	0.013
6 (SFF)	0.29	**0.015**	**0.034**	**0.142**	**0.105**	**0.497**	**0.352**	**0.134**	**0.013**	**0.021**	**0.141**	**0.002**	**0.14**	**0.011**	**0.068**
6 (FF)	**0.334**	0.004	0.011	0.12	0.007	0.432	0.156	0.002	0.005	0.006	0.14	0.001	0.135	0.006	0.002
6 (TFS)	0.158	0.005	0.007	0.074	0.007	0.169	0.051	0.001	0.002	0.001	0.081	0.0	0.104	0.007	0.003
7 (SFF)	**0.345**	**0.003**	0.013	**0.039**	**1.293**	**0.131**	**0.055**	0.318	**0.06**	**0.111**	**0.097**	**0.026**	**0.82**	**0.198**	**0.314**
7 (FF)	0.028	0.001	0.006	0.025	0.779	0.063	0.005	0.065	0.016	0.11	0.031	0.006	0.583	0.085	0.055
7 (TFS)	0.13	0.001	**0.014**	0.037	0.159	0.114	0.036	**0.332**	0.015	0.073	0.026	0.006	0.267	0.078	0.094
8 (SFF)	**0.182**	0.007	**0.119**	0.135	0.182	**0.004**	**0.172**	**0.233**	**0.073**	0.005	**0.117**	**0.081**	**0.045**	**0.518**	**0.014**
8 (FF)	0.155	0.004	0.036	0.058	**0.201**	0.003	0.07	0.157	0.051	**0.007**	0.04	0.04	0.038	0.41	0.008
8 (TFS)	0.105	**0.009**	0.111	**0.142**	0.145	0.003	0.135	0.073	0.046	0.001	0.105	0.039	0.036	0.402	0.014
9 (SFF)	**2.133**	**0.067**	**0.126**	**0.118**	**0.151**	**0.728**	**0.213**	**0.174**	**0.141**	**0.035**	**0.07**	**0.082**	**0.05**	0.471	**0.008**
9 (FF)	0.795	0.03	0.009	0.026	0.072	0.458	0.067	0.151	0.077	0.008	0.035	0.029	0.006	**0.478**	0.002
9 (TFS)	1.539	0.031	0.096	0.03	0.144	0.437	0.079	0.089	0.08	0.024	0.043	0.055	0.027	0.203	0.002
10 (SFF)	0.066	**0.031**	**0.478**	**0.029**	0.055	**0.004**	**0.087**	**0.069**	**0.202**	**0.423**	**0.835**	**0.103**	**0.08**	**0.981**	**0.085**
10 (FF)	0.028	0.015	0.094	0.004	0.014	0.001	0.043	0.02	0.017	0.135	0.431	0.008	0.056	0.255	0.008
10 (TFS)	**0.07**	0.027	0.167	0.005	**0.06**	0.001	0.045	0.014	0.043	0.273	0.462	0.008	0.012	0.382	0.005
11 (SFF)	**0.054**	**0.073**	**0.485**	**0.005**	**0.143**	0.005	**0.054**	**0.176**	**0.003**	**1.351**	**0.048**	**0.007**	0.003	**0.046**	**0.596**
11 (FF)	0.014	0.057	0.453	0.003	0.049	**0.023**	0.031	0.143	0.002	0.077	0.009	0.002	0.001	0.02	0.158
11 (TFS)	0.049	0.05	0.35	0.001	0.017	0.001	0.027	0.108	0.002	0.115	0.007	0.003	**0.005**	0.039	0.371
12 (SFF)	**0.511**	**0.026**	**0.184**	**0.082**	0.085	0.26	**0.637**	**0.221**	**0.092**	0.096	0.13	**0.014**	**0.002**	**0.011**	**0.137**
12 (FF)	0.349	0.016	0.137	0.016	0.006	**0.281**	0.091	0.085	0.026	**0.102**	**0.132**	0.013	0.001	0.003	0.059
12 (TFS)	0.126	0.025	0.107	0.022	**0.159**	0.034	0.297	0.071	0.04	0.057	0.07	0.012	0.002	0.004	0.091
13 (SFF)	**0.045**	**0.067**	**0.003**	**0.005**	**0.075**	0.001	**0.005**	**0.045**	**0.16**	**0.172**	**0.034**	**0.022**	**0.204**	**0.023**	**0.033**
13 (FF)	0.026	0.025	0.001	0.002	0.01	**0.002**	0.002	0.028	0.01	0.015	0.021	0.018	0.118	0.001	0.015
13 (TFS)	0.019	0.038	0.001	0.004	0.006	0.002	0.002	0.021	0.014	0.005	0.034	0.016	0.162	0.005	0.03
14 (SFF)	**0.98**	**0.579**	**0.012**	**0.022**	**0.014**	**0.003**	0.728	**0.02**	**0.002**	0.354	**0.042**	**0.023**	**0.055**	**0.068**	**0.132**
14 (FF)	0.248	0.226	0.003	0.009	0.006	0.001	0.683	0.009	0.0	0.244	0.041	0.018	0.032	0.059	0.074
14 (TFS)	0.166	0.406	0.01	0.009	0.011	0.002	**0.803**	0.008	0.0	**0.43**	0.023	0.009	0.053	0.041	0.039
15 (SFF)	**0.007**	**0.052**	**0.094**	**0.471**	**0.19**	**0.003**	**0.182**	**0.522**	**0.193**	**0.003**	**0.028**	**0.31**	**0.005**	**0.006**	**0.356**
15 (FF)	0.004	0.013	0.072	0.263	0.015	0.002	0.098	0.315	0.038	0.001	0.016	0.183	0.001	0.0	0.073
15 (TFS)	0.007	0.014	0.001	0.447	0.035	0.001	0.114	0.186	0.182	0.003	0.023	0.222	0.001	0.0	0.222
16 (SFF)	**0.003**	**0.283**	**0.335**	0.013	0.141	**0.185**	**0.165**	**0.638**	**1.156**	**0.14**	0.009	**0.043**	**0.105**	**0.015**	**0.013**
16 (FF)	0.001	0.175	0.275	**0.03**	0.049	0.129	0.016	0.299	0.694	0.048	0.006	0.031	0.073	0.009	0.004
16 (TFS)	0.001	0.102	0.335	0.009	**0.142**	0.109	0.1	0.153	0.774	0.004	**0.012**	0.033	0.026	0.009	0.008
17 (SFF)	**0.067**	**0.049**	0.358	**0.008**	**0.031**	**0.107**	**0.09**	**0.321**	**0.024**	**0.005**					
17 (FF)	0.031	0.02	**0.405**	0.004	0.023	0.091	0.044	0.051	0.015	0.001					
17 (TFS)	0.024	0.032	0.211	0.002	0.026	0.073	0.084	0.122	0.022	0.002					

Table 22: Standard deviations on 250 anomaly detection datasets under four random seeds.

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1 (SFF)	±6.8e-3	±3.0e-3	±1.3e-3	±5.0e-2	±1.2e-3	±0.12	1.0e-2	1.1e-3	3.6e-3	0.11	1.6e-2	2.9e-2	0.68	9.2e-3	6.8e-3
1 (FF)	±1.2e-2	±2.3e-3	±7.9e-4	±0.43	±1.7e-3	±1.9e-3	5.0e-2	1.3e-3	2.2e-5	1.3e-3	7.5e-4	1.8e-2	5.8e-2	1.0e-3	8.2e-4
1 (TFS)	±1.1e-2	±1.4e-3	±2.4e-4	±0.15	±2.7e-3	±1.7e-3	2.0e-2	4.6e-4	1.8e-2	1.4e-3	6.3e-2	3.9e-3	0.74	5.3e-3	3.4e-2
2 (SFF)	±5.6e-2	±3.9e-2	±1.4e-3	±1.3e-2	±1.6e-3	±6.9e-4	1.5e-2	3.8e-4	2.8e-2	2.1e-2	1.0e-2	2.0e-2	2.7e-2	4.1e-2	3.1e-3
2 (FF)	±5.5e-3	±4.4e-3	±3.9e-4	±2.1e-4	±2.1e-3	±3.2e-4	2.6e-2	8.7e-4	1.7e-2	3.7e-3	1.7e-2	7.3e-3	4.2e-2	6.3e-2	4.0e-3
2 (TFS)	±1.1e-2	±4.8e-4	±8.5e-4	±7.9e-2	±4.5e-3	±4.7e-4	6.3e-3	3.0e-4	3.0e-2	4.7e-3	4.7e-3	1.5e-2	5.7e-2	0.12	2.4e-3
3 (SFF)	±1.1e-2	±2.6e-2	±5.1e-4	±1.4e-2	±2.4e-2	±9.6e-2	2.4e-2	6.9e-3	2.3e-2	3.9e-2	4.9e-3	5.3e-2	1.1e-3	1.3e-2	1.6e-2
3 (FF)	±2.6e-4	±6.2e-3	±3.9e-4	±7.7e-4	±3.2e-2	±2.2e-4	1.2e-2	7.3e-4	5.5e-2	1.9e-2	3.5e-2	5.4e-2	2.7e-5	2.3e-2	8.4e-3
3 (TFS)	±8.4e-4	±6.9e-3	±2.2e-3	±2.2e-2	±6.4e-2	±0.18	1.0e-2	4.5e-3	1.5e-2	4.1e-2	4.5e-3	7.5e-2	1.2e-4	2.3e-3	1.4e-4
4 (SFF)	±4.2e-2	±3.4e-2	±1.3e-2	±3.0e-2	±1.7e-2	±7.8e-3	0.1	9.1e-2	0.23	7.7e-2	3.8e-3	3.6e-2	0.11	3.1e-2	3.4e-4
4 (FF)	±5.8e-3	±1.5e-2	±7.7e-4	±2.4e-3	±3.0e-3	±2.2e-2	0.18	1.9e-3	1.9e-3	5.2e-3	5.7e-5	5.5e-2	2.6e-2	4.3e-3	3.4e-4
4 (TFS)	±5.8e-3	±5.2e-2	±7.8e-2	±2.1e-2	±5.3e-3	±1.1e-2	0.22	9.8e-4	3.9e-2	5.7e-3	3.5e-3	2.5e-2	5.5e-2	2.5e-3	9.9e-4
5 (SFF)	±0.14	±0.47	±3.1e-3	±3.3e-3	±2.2e-2	±1.1e-2	3.4e-2	1.5e-2	1.9e-2	8.7e-3	4.6e-2	2.4e-2	0.35	2.0e-3	1.0e-3
5 (FF)	±0.15	±3.9e-2	±5.1e-3	±5.0e-2	±5.7e-4	±1.8e-2	1.0e-2	1.6e-2	1.3e-2	9.3e-3	4.6e-4	6.9e-2	4.8e-3	1.2e-3	2.6e-3
5 (TFS)	±0.1	±0.11	±2.4e-2	±4.6e-2	±7.5e-5	±7.2e-3	1.5e-3	4.7e-3	6.3e-3	3.0e-3	4.6e-3	8.8e-2	2.0e-3	4.4e-3	5.9e-3
6 (SFF)	±4.1e-2	±4.6e-3	±1.7e-2	±1.6e-2	±5.4e-2	±1.2e-2	0.2	9.4e-2	1.6e-2	1.3e-2	5.1e-3	7.5e-5	4.0e-3	1.0e-3	9.2e-2
6 (FF)	±6.8e-3	±2.9e-3	±8.2e-3	±6.9e-3	±6.2e-3	±7.6e-2	5.6e-2	1.7e-3	6.0e-3	5.4e-3	4.7e-3	9.2e-5	3.3e-4	3.4e-5	1.8e-3
6 (TFS)	±6.8e-2	±5.1e-5	±2.3e-3	±4.3e-3	±5.2e-3	±2.1e-2	7.8e-3	4.9e-4	3.9e-4	3.0e-4	1.1e-2	2.9e-4	1.4e-2	1.3e-3	1.1e-3
7 (SFF)	±0.28	±6.8e-4	±3.5e-3	±4.4e-3	±1.0	±9.3e-3	1.1e-2	4.9e-2	1.9e-2	4.4e-3	6.1e-3	6.7e-3	0.16	1.8e-2	6.2e-3
7 (FF)	±2.0e-3	±7.6e-4	±3.1e-3	±2.7e-3	±0.41	±2.4e-2	2.1e-4	5.1e-2	1.9e-4	2.7e-2	3.4e-2	2.1e-3	0.1	5.4e-2	2.2e-2
7 (TFS)	±3.7e-3	±8.3e-4	±9.1e-3	±1.9e-3	±1.8e-2	±7.0e-3	1.4e-2	2.7e-2	3.0e-3	2.5e-2	2.1e-2	1.5e-3	4.0e-2	5.1e-2	1.1e-2
8 (SFF)	±1.6e-3	±2.3e-4	±5.8e-3	±2.8e-2	±3.2e-2	±1.7e-4	2.8e-2	1.2e-2	2.5e-2	1.5e-3	1.5e-2	5.1e-2	5.4e-3	4.9e-2	4.5e-4
8 (FF)	±4.8e-2	±9.8e-5	±1.2e-3	±1.1e-2	±2.1e-2	±1.5e-3	1.0e-2	2.6e-2	3.8e-3	5.6e-3	1.3e-2	2.0e-2	8.4e-4	1.1e-2	1.4e-3
8 (TFS)	±5.9e-3	±5.5e-3	±1.6e-3	±1.0e-2	±2.0e-2	±9.2e-4	2.0e-2	4.9e-2	1.0e-2	1.2e-4	2.7e-2	2.2e-2	1.6e-3	8.6e-2	3.8e-3
9 (SFF)	±0.36	±5.6e-2	±0.1	±4.5e-2	±2.9e-2	±7.5e-2	1.5e-2	2.4e-2	3.1e-2	1.2e-2	1.8e-2	3.4e-2	3.3e-2	1.6e-3	5.5e-3
9 (FF)	±0.46	±1.3e-2	±2.0e-3	±3.4e-2	±2.3e-2	±8.5e-4	7.9e-2	3.2e-2	2.8e-2	9.5e-4	5.4e-4	4.8e-3	1.9e-3	1.3e-3	6.1e-4
9 (TFS)	±0.96	±2.3e-2	±0.12	±4.0e-2	±9.2e-3	±8.7e-2	8.7e-2	2.6e-2	2.4e-2	1.7e-2	1.5e-2	6.9e-3	1.8e-2	4.7e-2	6.4e-4
10 (SFF)	±6.5e-3	±7.5e-3	±0.27	±2.3e-2	±2.9e-2	±3.4e-4	2.1e-2	2.2e-2	5.0e-2	2.4e-2	0.13	6.3e-2	4.9e-3	0.16	9.7e-2
10 (FF)	±2.9e-3	±3.3e-3	±7.9e-3	±5.2e-4	±9.2e-4	±2.0e-4	5.4e-3	2.2e-2	3.5e-3	2.9e-3	4.8e-2	4.3e-3	9.0e-3	0.14	3.3e-3
10 (TFS)	±1.5e-2	±1.8e-3	±0.16	±1.1e-3	±5.3e-2	±2.9e-4	6.8e-3	7.0e-3	8.6e-3	0.15	0.24	2.8e-3	2.2e-3	0.18	1.3e-3
11 (SFF)	±5.4e-2	±1.6e-2	±3.2e-2	±6.6e-4	±5.0e-2	±1.7e-3	3.6e-3	2.4e-2	1.3e-3	0.86	4.7e-2	3.6e-3	7.6e-4	6.4e-3	0.11
11 (FF)	±1.2e-2	±6.8e-4	±3.1e-2	±1.5e-3	±3.7e-2	±1.5e-2	4.7e-3	6.1e-3	1.1e-3	7.5e-2	4.6e-3	9.3e-4	2.9e-4	1.3e-2	0.19
11 (TFS)	±6.3e-2	±1.1e-2	±3.3e-2	±3.4e-4	±9.3e-3	±2.6e-4	7.4e-3	2.5e-2	8.6e-4	2.9e-2	3.1e-3	1.0e-3	3.6e-3	9.9e-3	8.9e-2
12 (SFF)	±5.2e-2	±1.4e-3	±2.2e-3	±4.1e-2	±5.8e-2	±2.6e-2	0.39	1.5e-2	2.7e-2	4.5e-3	1.2e-2	1.5e-3	2.3e-4	6.3e-4	2.0e-2
12 (FF)	±1.7e-2	±5.2e-3	±1.2e-2	±3.8e-3	±2.0e-3	±3.3e-2	7.9e-3	5.2e-2	9.2e-3	4.8e-2	8.3e-3	7.4e-5	6.3e-5	1.1e-4	2.8e-2
12 (TFS)	±0.15	±8.7e-3	±1.8e-2	±3.8e-3	±5.6e-2	±4.1e-3	0.35	7.3e-2	2.0e-2	2.0e-2	1.4e-2	1.1e-3	6.9e-4	2.1e-4	1.1e-2
13 (SFF)	±8.6e-3	±3.9e-2	±6.5e-4	±1.0e-3	±3.8e-2	±2.2e-4	7.7e-4	2.5e-3	0.1	0.19	3.5e-3	2.1e-3	3.9e-2	1.6e-2	6.8e-3
13 (FF)	±1.6e-2	±7.7e-3	±3.6e-4	±4.5e-4	±2.7e-3	±2.8e-4	4.6e-4	1.4e-2	1.4e-3	5.5e-3	1.4e-3	7.4e-3	1.6e-2	7.6e-4	3.1e-3
13 (TFS)	±1.8e-2	±3.3e-2	±3.1e-4	±2.5e-3	±2.2e-3	±6.2e-4	2.2e-4	5.4e-3	2.3e-3	6.1e-4	2.1e-3	5.9e-3	5.9e-2	5.4e-3	5.8e-3
14 (SFF)	±0.91	±0.15	±3.1e-3	±3.6e-3	±4.6e-3	±5.6e-4	8.4e-2	7.6e-3	5.1e-4	2.9e-2	6.6e-3	2.8e-3	1.0e-2	6.5e-3	2.4e-2
14 (FF)	±5.4e-2	±4.4e-2	±2.1e-3	±2.9e-5	±2.3e-3	±3.7e-4	2.4e-3	8.8e-4	2.3e-4	8.8e-2	2.5e-2	1.0e-2	5.6e-3	1.3e-2	4.3e-3
14 (TFS)	±3.1e-2	±0.19	±5.9e-3	±9.0e-3	±3.2e-3	±5.3e-4	1.3e-2	1.3e-3	1.0e-4	0.4	1.6e-2	5.5e-3	9.4e-3	1.4e-2	2.5e-2
15 (SFF)	±5.2e-4	±3.9e-2	±8.6e-3	±1.7e-2	±6.7e-2	±1.4e-3	9.9e-3	0.11	5.7e-3	8.1e-4	8.0e-4	3.6e-2	2.0e-3	3.1e-3	9.7e-2
15 (FF)	±8.1e-5	±7.1e-3	±5.3e-3	±7.0e-4	±1.2e-3	±1.2e-3	3.6e-2	0.13	3.2e-2	1.5e-4	3.5e-3	9.7e-2	3.0e-4	1.8e-4	1.9e-2
15 (TFS)	±3.3e-3	±5.9e-3	±5.1e-4	±3.7e-2	±2.0e-2	±3.6e-4	5.1e-2	9.3e-2	1.5e-2	1.4e-3	8.2e-3	5.1e-2	1.2e-3	1.6e-4	7.8e-2
16 (SFF)	±3.6e-4	±4.2e-2	±1.5e-2	±7.2e-4	±2.2e-2	±8.0e-3	1.9e-2	0.13	0.12	5.8e-4	1.8e-3	8.4e-4	1.0e-2	2.5e-3	1.8e-3
16 (FF)	±6.4e-4	±1.7e-2	±5.5e-2	±3.7e-2	±1.3e-2	±1.9e-2	4.5e-3	0.2	8.6e-2	6.5e-2	2.0e-3	1.8e-2	4.8e-2	1.1e-2	4.6e-3
16 (TFS)	±5.4e-4	±2.4e-2	±0.13	±1.7e-3	±1.8e-2	±6.8e-2	5.7e-3	0.18	3.8e-2	1.1e-3	6.6e-3	1.1e-2	2.1e-2	5.5e-4	1.1e-2
17 (SFF)	±1.8e-2	±1.2e-2	±7.7e-2	±3.0e-3	±3.4e-3	±6.8e-3	4.2e-3	0.13	1.4e-3	4.2e-3					
17 (FF)	±1.8e-2	±3.0e-3	±3.4e-3	±2.1e-3	±4.1e-3	±1.5e-3	1.8e-2	5.7e-3	5.6e-4	8.5e-5					
17 (TFS)	±1.1e-2	±1.5e-3	±7.7e-2	±3.8e-4	±3.0e-3	±6.5e-3	4.7e-3	4.9e-3	7.9e-3	1.4e-3					

Table 23: Complete standard deviation and MSE of applying our *smoothed fine-tuning* (SFF) on other LTSMs TimesFM and MOMENT.

Data proportion	25% (TimesFM)			100% (TimesFM)			25% (MOMENT)			100% (MOMENT)		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.1139**	0.1276	0.1209	**0.1149**	0.1452	0.1199	**0.1502**	0.2648	0.1564	**0.1064**	0.1448	0.1091
Standard deviation	2.0e-3	4.2e-3	2.9e-4	6.3e-4	1.7e-2	2.3e-3	2.4e-3	4.6e-4	3.6e-3	5.8e-4	1.4e-4	2.6e-4
ETTh1	**0.3955**	0.4382	0.4638	**0.406**	0.5101	0.4358	**0.4287**	0.4454	0.454	**0.3757**	0.3951	0.387
Standard deviation	1.9e-3	2.4e-2	6.0e-4	3.6e-3	8.8e-3	2.0e-3	2.1e-3	9.9e-4	1.8e-3	4.0e-4	6.5e-5	1.4e-3
ETTh2	**0.3232**	0.3384	0.3325	**0.3198**	0.3483	0.347	**0.3199**	0.3328	0.3326	**0.2818**	0.2936	0.2979
Standard deviation	2.9e-3	9.0e-3	9.5e-4	2.0e-3	3.3e-3	4.7e-3	1.4e-3	2.6e-4	1.5e-3	7.8e-4	4.0e-5	1.6e-3
ETTm1	**0.3429**	0.4001	0.3903	**0.3478**	0.3756	0.3926	**0.3457**	0.3587	0.3538	**0.3139**	0.3148	0.3272
Standard deviation	3.6e-3	7.3e-3	7.2e-4	2.9e-3	3.0e-2	4.8e-4	1.2e-3	1.2e-4	6.6e-4	1.0e-4	3.1e-5	2.0e-3
ETTm2	**0.1983**	0.2061	0.2091	**0.2026**	0.2122	0.225	**0.1793**	0.192	0.1846	**0.1692**	0.172	0.1736
Standard deviation	3.7e-3	2.6e-3	4.0e-4	1.5e-3	6.8e-3	3.8e-3	2.1e-4	5.0e-4	6.1e-4	3.7e-4	3.7e-5	9.5e-4
Weather	**0.0865**	0.0885	0.1995	**0.082**	0.1184	0.1902	**0.1673**	0.1682	0.169	**0.1548**	0.1558	0.161
Standard deviation	4.7e-3	5.6e-3	4.5e-3	1.1e-2	3.0e-2	4.2e-3	1.1e-4	1.4e-4	1.7e-4	1.9e-4	2.2e-4	1.4e-4
Avg. Improvements	-	7.55%	16.21	-	15.35%	16.18	-	10.28%	3.25	-	6.34%	3.54
Max. Improvements	-	14.3%	56.64	-	30.74%	56.89	-	43.28%	5.57	-	26.52%	5.4

Table 24: Complete standard deviation and MAE of applying our *smoothed fine-tuning* (SFF) on other LTSMs TimesFM and MOMENT.

Data proportion	25% (TimesFM)			100% (TimesFM)			25% (MOMENT)			100% (MOMENT)		
Methods	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS	SFF	FF	TFS
Exchange	**0.2414**	0.2519	0.2497	**0.2422**	0.2703	0.2472	**0.282**	0.3844	0.2894	**0.2322**	0.2751	0.2369
Standard deviation	9.9e-4	4.3e-3	4.2e-4	8.4e-4	1.8e-2	1.7e-3	2.3e-3	3.3e-4	3.4e-3	5.8e-4	1.0e-4	3.1e-4
ETTh1	**0.405**	0.4226	0.4526	**0.4149**	0.4567	0.4344	**0.4386**	0.4455	0.4559	**0.4022**	0.4144	0.4112
Standard deviation	4.3e-3	8.0e-3	4.5e-4	3.0e-3	2.7e-3	5.3e-4	1.2e-3	7.4e-4	8.3e-4	4.2e-4	2.6e-5	1.3e-3
ETTh2	**0.3731**	0.3742	0.3803	**0.3704**	0.3782	0.391	**0.3695**	0.3797	0.3784	**0.3404**	0.35	0.3514
Standard deviation	1.2e-3	3.9e-3	9.3e-4	8.3e-4	2.6e-3	1.4e-3	1.4e-3	3.5e-4	9.3e-4	3.3e-4	4.1e-5	8.5e-4
ETTm1	**0.3851**	0.4119	0.4223	**0.3892**	0.3992	0.4227	**0.3938**	0.4054	0.4013	**0.3783**	0.3787	0.3854
Standard deviation	2.1e-3	3.8e-3	4.3e-4	3.2e-3	1.5e-2	7.6e-4	1.3e-3	1.3e-4	4.6e-4	1.6e-4	2.4e-5	7.5e-4
ETTm2	**0.2823**	0.2847	0.2925	**0.2748**	0.2851	0.3119	**0.2671**	0.2769	0.2724	**0.2587**	0.2613	0.2638
Standard deviation	1.0e-3	1.4e-3	3.8e-4	1.9e-3	5.1e-3	4.4e-3	6.3e-5	4.8e-4	4.9e-4	1.4e-3	1.2e-5	7.5e-4
Weather	**0.1135**	0.1161	0.2523	**0.1025**	0.152	0.2424	**0.2213**	0.2247	0.2231	**0.2107**	0.2114	0.2142
Standard deviation	3.3e-3	6.7e-3	4.2e-3	1.4e-2	4.0e-2	4.2e-3	1.1e-3	1.2e-4	3.6e-4	1.1e-4	4.0e-4	2.5e-4
Avg. Improvements	-	3.04%	13.84	-	10.05%	14.88	-	6.46%	2.22	-	3.78%	2.12
Max. Improvements	-	6.51%	55.01	-	32.57%	57.71	-	26.64%	3.79	-	15.59%	3.13

Table 25: Full standard deviation and MSE of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF). The average performance is reported for each group of three different data proportions, e.g., “Avg. on 1%, 2%, 3%”.

Data proportion	Avg. on 1%, 2%, 3%			Avg. on 4%, 5%, 10%			Avg. on 15%, 20%, 25%			Avg. on 50%, 75%, 100%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.0856**	0.5943	0.4801	**0.0848**	0.5906	0.4186	**0.0816**	0.563	0.1743	**0.0812**	0.474	0.0962
Standard deviation	4.4e-4	7.3e-3	3.9e-3	3.7e-4	7.2e-3	6.7e-3	6.1e-4	6.6e-3	7.4e-3	8.3e-4	4.7e-3	1.9e-3
ETTh1	**0.3722**	0.8806	0.7171	**0.3641**	0.8594	0.6367	**0.3523**	0.7955	0.4127	**0.3529**	0.6356	0.3731
Standard deviation	6.5e-3	9.2e-3	3.6e-3	4.7e-3	8.2e-3	9.2e-3	1.4e-3	6.7e-3	3.4e-3	1.3e-3	4.5e-3	1.8e-3
ETTh2	**0.28**	0.4427	0.4026	**0.278**	0.4375	0.3707	**0.2768**	0.4234	0.3113	**0.2758**	0.3849	0.3001
Standard deviation	1.5e-3	7.3e-3	3.8e-3	2.8e-3	6.8e-3	1.6e-3	2.6e-3	5.8e-3	1.5e-3	9.7e-4	3.1e-3	1.7e-3
ETTm1	**0.3448**	1.046	0.7038	**0.3162**	1.0043	0.4772	**0.301**	0.8975	0.3245	**0.2976**	0.6608	0.3124
Standard deviation	3.5e-3	2.6e-2	7.3e-3	2.6e-3	2.3e-2	4.5e-3	1.0e-3	1.8e-2	6.8e-4	1.5e-3	9.1e-3	1.2e-3
ETTm2	**0.1723**	0.3555	0.3024	**0.1663**	0.3499	0.2559	**0.1623**	0.3297	0.1847	**0.1616**	0.2771	0.1804
Standard deviation	2.3e-3	6.2e-3	3.1e-3	2.1e-3	5.9e-3	2.0e-3	1.0e-3	4.9e-3	9.8e-4	6.2e-4	2.7e-3	1.6e-3
Weather	**0.1515**	0.324	0.2478	**0.146**	0.3082	0.1741	**0.1441**	0.2699	0.1481	**0.1453**	0.2013	0.1565
Standard deviation	4.7e-4	4.2e-3	4.1e-3	1.9e-4	3.4e-3	3.5e-3	6.6e-5	1.9e-3	5.6e-4	3.6e-4	8.2e-4	1.1e-3
Electricity	**0.1346**	0.6069	0.181	**0.132**	0.3242	0.1398	**0.1305**	0.2023	0.132	**0.1301**	0.1561	0.1335
Standard deviation	2.0e-4	1.1e-3	8.6e-4	2.3e-4	4.3e-4	2.6e-4	1.8e-4	1.9e-4	2.6e-4	2.7e-4	8.9e-5	5.8e-4
Traffic	**0.3678**	0.9577	0.4081	**0.3562**	0.5999	0.3638	**0.3494**	0.4529	0.3572	**0.3516**	0.4079	0.3575
Standard deviation	1.3e-4	2.2e-3	4.4e-4	4.2e-4	1.9e-3	3.7e-4	1.8e-3	4.4e-4	7.4e-4	1.4e-3	1.5e-4	5.4e-4
Avg. Improvements	-	-	41.14%	-	-	30.02%	-	-	13.04%	-	-	6.95%
Max. Improvements	-	-	82.17%	-	-	79.74%	-	-	53.18%	-	-	15.59%

Table 26: Full standard deviation and MAE of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF). The average performance is reported for each group of three different data proportions, e.g., “Avg. on 1%, 2%, 3%”.

Data proportion	Avg. on 1%, 2%, 3%			Avg. on 4%, 5%, 10%			Avg. on 15%, 20%, 25%			Avg. on 50%, 75%, 100%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.2047**	0.5867	0.5287	**0.2039**	0.5848	0.4886	**0.2013**	0.5714	0.3048	**0.2014**	0.5244	0.2202
Standard deviation	3.5e-4	2.6e-3	1.4e-3	5.1e-4	2.6e-3	3.9e-3	7.2e-4	2.5e-3	6.6e-3	1.1e-3	2.1e-3	2.8e-3
ETTh1	**0.4006**	0.644	0.5865	**0.3963**	0.6364	0.5505	**0.3886**	0.6134	0.4376	**0.3905**	0.5504	0.4097
Standard deviation	3.4e-3	4.8e-3	1.3e-3	3.5e-3	4.4e-3	3.9e-3	1.1e-3	3.8e-3	2.1e-3	1.4e-3	2.8e-3	1.0e-3
ETTh2	**0.3345**	0.46	0.4366	**0.3347**	0.457	0.4154	**0.3358**	0.4488	0.3671	**0.3365**	0.4248	0.3558
Standard deviation	9.5e-4	4.1e-3	2.1e-3	1.4e-3	3.9e-3	9.5e-4	1.8e-3	3.3e-3	1.3e-3	7.3e-4	1.9e-3	1.2e-3
ETTm1	**0.3942**	0.7198	0.597	**0.3735**	0.7061	0.4873	**0.3637**	0.6695	0.3874	**0.3592**	0.5751	0.3759
Standard deviation	2.3e-3	8.9e-3	2.1e-3	1.3e-3	8.3e-3	1.2e-3	9.5e-4	6.7e-3	3.5e-4	1.8e-3	3.6e-3	6.5e-4
ETTm2	**0.2601**	0.3958	0.3636	**0.2547**	0.3926	0.3328	**0.2523**	0.381	0.2778	**0.2512**	0.349	0.2711
Standard deviation	2.5e-3	3.5e-3	1.7e-3	1.5e-3	3.4e-3	1.1e-3	5.2e-4	2.9e-3	1.3e-3	8.2e-4	1.7e-3	1.5e-3
Weather	**0.2005**	0.3584	0.299	**0.1941**	0.3472	0.2298	**0.1929**	0.3189	0.2007	**0.1946**	0.2598	0.2072
Standard deviation	5.1e-4	2.8e-3	2.9e-3	2.0e-4	2.3e-3	3.1e-3	2.4e-4	1.3e-3	6.5e-4	6.8e-4	6.2e-4	1.7e-3
Electricity	**0.2312**	0.6057	0.2824	**0.2275**	0.411	0.2387	**0.2251**	0.3048	0.2269	**0.2243**	0.2599	0.227
Standard deviation	3.1e-4	2.3e-3	6.4e-4	2.2e-4	8.3e-4	2.9e-4	3.5e-4	6.6e-5	1.2e-4	3.7e-4	7.8e-5	6.9e-4
Traffic	**0.2619**	0.6062	0.3052	**0.2532**	0.4331	0.263	**0.2474**	0.3423	0.2519	**0.2474**	0.2972	0.2486
Standard deviation	3.9e-5	4.8e-4	2.2e-4	4.0e-4	9.6e-4	1.0e-4	1.8e-3	2.0e-4	1.1e-3	6.2e-4	1.6e-4	3.3e-4
Avg. Improvements	-	-	30.51%	-	-	22.06%	-	-	9.43%	-	-	4.77%
Max. Improvements	-	-	61.28%	-	-	58.27%	-	-	33.96%	-	-	8.54%

Table 27: Full standard deviation and **MSE** of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) under **1%, 2%, 3%, and 4%** proportion of available data.

Data proportion	1%			2%			3%			4%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.0857**	0.5944	0.4841	**0.0863**	0.5943	0.4788	**0.085**	0.5943	0.4773	**0.0852**	0.5943	0.4763
Standard deviation	3.2e-4	7.3e-3	4.5e-3	4.2e-4	7.3e-3	3.0e-3	5.8e-4	7.3e-3	4.2e-3	2.4e-4	7.3e-3	3.9e-3
ETTh1	**0.3737**	0.8809	0.7219	**0.3731**	0.8806	0.7159	**0.37**	0.8804	0.7134	**0.371**	0.8731	0.7677
Standard deviation	6.3e-3	9.1e-3	3.0e-3	6.6e-3	9.2e-3	3.4e-3	6.6e-3	9.2e-3	4.3e-3	7.5e-3	7.7e-3	5.3e-3
ETTh2	**0.2839**	0.4428	0.4047	**0.2769**	0.4427	0.402	**0.2797**	0.4427	0.4012	**0.279**	0.4394	0.3874
Standard deviation	1.0e-3	7.3e-3	4.1e-3	1.7e-3	7.3e-3	3.7e-3	1.9e-3	7.3e-3	3.7e-3	2.7e-3	6.9e-3	2.5e-3
ETTm1	**0.3702**	1.0585	0.8141	**0.3343**	1.0401	0.6548	**0.3301**	1.0393	0.6424	**0.3229**	1.0215	0.5353
Standard deviation	4.4e-3	2.7e-2	8.4e-3	2.7e-3	2.6e-2	6.6e-3	3.4e-3	2.6e-2	6.7e-3	3.5e-3	2.4e-2	5.0e-3
ETTm2	**0.1744**	0.3569	0.3154	**0.1759**	0.3568	0.3139	**0.1668**	0.3528	0.2779	**0.1674**	0.3527	0.2757
Standard deviation	2.4e-3	6.3e-3	3.5e-3	2.1e-3	6.3e-3	3.0e-3	2.4e-3	6.0e-3	2.6e-3	1.4e-3	6.1e-3	2.1e-3
Weather	**0.1541**	0.3273	0.2726	**0.1507**	0.324	0.2477	**0.1499**	0.3207	0.223	**0.1474**	0.3175	0.2
Standard deviation	3.9e-4	4.4e-3	2.8e-3	1.5e-4	4.2e-3	4.5e-3	8.7e-4	4.0e-3	4.9e-3	2.9e-4	3.8e-3	6.5e-3
Electricity	**0.1369**	0.7664	0.2197	**0.1342**	0.5866	0.1698	**0.133**	0.4678	0.1535	**0.1331**	0.3911	0.1464
Standard deviation	7.8e-5	7.9e-4	9.6e-4	1.9e-4	1.4e-3	9.8e-4	3.3e-4	1.2e-3	6.5e-4	2.5e-4	5.6e-4	4.6e-4
Traffic	**0.3743**	1.1903	0.4536	**0.3671**	0.917	0.3941	**0.3622**	0.7658	0.3767	**0.3594**	0.6768	0.3688
Standard deviation	1.0e-4	2.4e-3	1.1e-3	1.4e-5	1.7e-3	1.7e-4	2.7e-4	2.4e-3	3.3e-5	2.6e-4	2.6e-3	2.2e-4
Avg. Improvements	-	62.35%	44.78	-	61.21%	40.11	-	59.87%	37.4	-	58.38%	34.83
Max. Improvements	-	85.58%	82.3	-	85.48%	81.98	-	85.7%	82.19	-	85.66%	82.11

Table 28: Full standard deviation and **MSE** of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) under **5%, 10%, 15%, and 20%** proportion of available data.

Data proportion	5%			10%			15%			20%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.0856**	0.5936	0.468	**0.0838**	0.5838	0.3114	**0.082**	0.5737	0.2205	**0.0818**	0.5613	0.1646
Standard deviation	1.6e-4	7.4e-3	3.0e-3	7.2e-4	7.0e-3	1.3e-2	3.1e-4	6.8e-3	1.1e-2	1.1e-3	6.7e-3	6.2e-3
ETTh1	**0.3646**	0.8618	0.6158	**0.3568**	0.8434	0.5265	**0.3551**	0.8139	0.4433	**0.3509**	0.7926	0.4029
Standard deviation	4.6e-3	8.7e-3	1.2e-2	2.1e-3	8.2e-3	1.1e-2	1.7e-3	6.6e-3	4.2e-3	1.9e-3	6.9e-3	3.5e-3
ETTh2	**0.277**	0.4386	0.3743	**0.2782**	0.4346	0.3503	**0.2776**	0.4279	0.3223	**0.2785**	0.423	0.3116
Standard deviation	3.5e-3	7.0e-3	1.7e-3	2.2e-3	6.6e-3	6.9e-4	3.8e-3	6.2e-3	1.6e-3	1.5e-3	5.7e-3	1.5e-3
ETTm1	**0.3189**	1.0209	0.5224	**0.307**	0.9706	0.374	**0.303**	0.9274	0.3353	**0.3006**	0.8982	0.3229
Standard deviation	2.6e-3	2.4e-2	5.2e-3	1.7e-3	2.1e-2	3.2e-3	1.0e-3	1.9e-2	5.1e-4	1.1e-3	1.8e-2	8.8e-4
ETTm2	**0.1679**	0.3525	0.2738	**0.164**	0.3444	0.2182	**0.1633**	0.3368	0.1942	**0.1631**	0.3297	0.1862
Standard deviation	2.9e-3	6.1e-3	2.2e-3	1.8e-3	5.6e-3	1.6e-3	1.3e-3	5.3e-3	1.3e-3	7.8e-4	4.9e-3	8.5e-4
Weather	**0.1461**	0.3116	0.1721	**0.1447**	0.2956	0.1501	**0.1443**	0.2823	0.1476	**0.1443**	0.2654	0.1488
Standard deviation	1.6e-4	3.5e-3	3.2e-3	1.2e-4	2.8e-3	6.2e-4	1.2e-4	2.2e-3	7.4e-4	2.9e-5	2.0e-3	7.5e-4
Electricity	**0.1319**	0.3355	0.1402	**0.1309**	0.246	0.1329	**0.1309**	0.2179	0.1321	**0.1306**	0.1989	0.1321
Standard deviation	2.6e-4	1.4e-4	2.9e-4	1.8e-4	5.8e-4	1.9e-5	2.3e-4	2.7e-4	2.6e-4	1.3e-4	1.5e-4	3.2e-4
Traffic	**0.3576**	0.6198	0.3639	**0.3518**	0.503	0.3588	**0.3508**	0.4671	0.3578	**0.349**	0.4505	0.3571
Standard deviation	2.8e-5	2.3e-3	7.1e-5	9.8e-4	8.0e-4	8.3e-4	1.2e-3	4.4e-4	5.8e-4	1.9e-3	4.2e-4	1.2e-3
Avg. Improvements	-	57.17%	31.11	-	53.5%	21.96	-	51.22%	15.9	-	49.36%	12.45
Max. Improvements	-	85.58%	81.71	-	85.65%	73.09	-	85.71%	62.81	-	85.43%	50.3

Table 29: Full standard deviation and **MSE** of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) under **25%, 50%, 75%, and 100%** proportion of available data.

Data proportion	25%			50%			75%			100%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.0811**	0.5541	0.1377	**0.0809**	0.5163	0.1011	**0.0819**	0.4667	0.0958	**0.0809**	0.439	0.0917
Standard deviation	4.5e-4	6.1e-3	4.7e-3	5.4e-4	5.4e-3	2.7e-3	1.2e-3	4.8e-3	2.0e-3	7.6e-4	3.8e-3	9.2e-4
ETTh1	**0.351**	0.7801	0.3918	**0.3509**	0.695	0.3724	**0.3513**	0.623	0.3769	**0.3567**	0.5888	0.37
Standard deviation	6.1e-4	6.6e-3	2.4e-3	1.1e-3	5.1e-3	1.2e-3	1.4e-3	4.4e-3	1.5e-3	1.4e-3	4.0e-3	2.8e-3
ETTh2	**0.2745**	0.4192	0.3001	**0.2757**	0.3985	0.2977	**0.2778**	0.3843	0.3034	**0.274**	0.372	0.2991
Standard deviation	2.5e-3	5.5e-3	1.4e-3	2.0e-3	4.0e-3	1.6e-3	4.8e-4	3.0e-3	1.9e-3	3.8e-4	2.4e-3	1.7e-3
ETTm1	**0.2995**	0.8669	0.3152	**0.2979**	0.7422	0.3103	**0.2974**	0.6467	0.3136	**0.2976**	0.5935	0.3132
Standard deviation	1.1e-3	1.6e-2	6.4e-4	1.6e-3	1.1e-2	8.8e-4	1.3e-3	8.7e-3	1.5e-3	1.5e-3	7.2e-3	1.3e-3
ETTm2	**0.1608**	0.3226	0.1738	**0.1609**	0.2947	0.1751	**0.1627**	0.2771	0.1863	**0.1613**	0.2595	0.1798
Standard deviation	9.7e-4	4.6e-3	7.9e-4	3.8e-4	3.4e-3	8.4e-4	4.3e-4	2.7e-3	1.2e-3	1.0e-3	2.1e-3	2.9e-3
Weather	**0.144**	0.2619	0.1479	**0.1443**	0.2248	0.1497	**0.1468**	0.192	0.1624	**0.1451**	0.1871	0.1573
Standard deviation	5.2e-5	1.5e-3	1.9e-4	2.1e-4	1.0e-3	7.2e-4	1.3e-4	7.9e-4	1.7e-3	7.3e-4	6.5e-4	9.1e-4
Electricity	**0.1303**	0.1902	0.1317	**0.1301**	0.1654	0.1324	**0.13**	0.1538	0.1345	**0.1304**	0.1491	0.1335
Standard deviation	1.7e-4	1.5e-4	2.0e-4	2.4e-4	1.0e-4	1.2e-4	3.6e-4	9.5e-5	9.7e-4	2.0e-4	6.9e-5	6.4e-4
Traffic	**0.3488**	0.441	0.3567	**0.3497**	0.4164	0.3564	**0.3502**	0.406	0.3587	**0.3551**	0.4014	0.3574
Standard deviation	2.3e-3	4.5e-4	4.8e-4	1.7e-3	2.5e-4	7.4e-4	2.5e-3	1.2e-4	3.3e-4	7.1e-5	7.5e-5	5.7e-4
Avg. Improvements	-	48.49%	9.8	-	42.89%	6.56	-	37.73%	7.86	-	35.19%	6.22
Max. Improvements	-	85.36%	41.1	-	84.33%	19.98	-	82.45%	14.51	-	81.57%	11.78

Table 30: Full standard deviation and **MAE** of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) under **1%, 2%, 3%, and 4%** proportion of available data.

Data proportion	1%			2%			3%			4%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.2051**	0.5867	0.5309	**0.2052**	0.5867	0.528	**0.204**	0.5867	0.5272	**0.2041**	0.5866	0.5267
Standard deviation	2.3e-4	2.6e-3	1.5e-3	4.8e-4	2.6e-3	1.1e-3	3.5e-4	2.6e-3	1.5e-3	4.3e-4	2.6e-3	1.4e-3
ETTh1	**0.4019**	0.6441	0.5884	**0.401**	0.644	0.5861	**0.3992**	0.6439	0.5851	**0.4012**	0.6409	0.6001
Standard deviation	3.3e-3	4.8e-3	1.2e-3	3.4e-3	4.8e-3	1.2e-3	3.5e-3	4.8e-3	1.4e-3	5.3e-3	4.4e-3	1.5e-3
ETTh2	**0.3389**	0.4601	0.4379	**0.3323**	0.46	0.4362	**0.3326**	0.46	0.4356	**0.3344**	0.4581	0.4267
Standard deviation	2.0e-3	4.1e-3	2.3e-3	3.2e-4	4.1e-3	2.0e-3	5.2e-4	4.1e-3	2.0e-3	5.2e-4	3.9e-3	1.5e-3
ETTm1	**0.412**	0.7239	0.6419	**0.3875**	0.7179	0.5773	**0.3835**	0.7176	0.5717	**0.378**	0.7119	0.5213
Standard deviation	2.5e-3	9.1e-3	3.6e-3	2.6e-3	8.8e-3	1.4e-3	1.9e-3	8.9e-3	1.3e-3	1.8e-3	8.6e-3	8.9e-4
ETTm2	**0.2628**	0.3966	0.3717	**0.2617**	0.3965	0.3709	**0.2559**	0.3942	0.3483	**0.2564**	0.3942	0.347
Standard deviation	3.0e-3	3.6e-3	2.0e-3	1.9e-3	3.6e-3	1.7e-3	2.5e-3	3.5e-3	1.5e-3	1.6e-3	3.5e-3	1.1e-3
Weather	**0.2034**	0.3607	0.3198	**0.1999**	0.3584	0.2994	**0.1987**	0.3562	0.2779	**0.1957**	0.3539	0.2567
Standard deviation	5.6e-4	3.0e-3	1.7e-3	2.6e-4	2.8e-3	3.2e-3	7.0e-4	2.7e-3	3.7e-3	2.3e-4	2.6e-3	5.8e-3
Electricity	**0.2342**	0.6951	0.3184	**0.2307**	0.5986	0.2729	**0.229**	0.5233	0.256	**0.2292**	0.4676	0.2476
Standard deviation	8.3e-5	2.5e-3	5.1e-4	3.4e-4	2.3e-3	8.5e-4	5.0e-4	2.0e-3	5.6e-4	2.7e-4	1.4e-3	4.4e-4
Traffic	**0.2672**	0.7112	0.3433	**0.2611**	0.59	0.2946	**0.2575**	0.5175	0.2776	**0.2559**	0.4736	0.2694
Standard deviation	1.1e-4	5.1e-4	4.6e-4	4.7e-6	1.5e-4	1.5e-4	4.7e-6	7.8e-4	7.1e-5	9.4e-5	1.2e-3	8.5e-5
Avg. Improvements	-	47.27%	33.23	-	46.5%	29.86	-	45.41%	27.81	-	44.14%	25.73
Max. Improvements	-	66.31%	61.37	-	65.02%	61.14	-	65.23%	61.31	-	65.21%	61.25

Table 31: Full standard deviation and **MAE** of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) under **5%, 10%, 15%, and 20%** proportion of available data.

Data proportion	5%			10%			15%			20%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.2046**	0.5863	0.5221	**0.2031**	0.5816	0.4169	**0.2017**	0.5766	0.3476	**0.2019**	0.5705	0.298
Standard deviation	2.0e-4	2.6e-3	1.0e-3	9.0e-4	2.6e-3	9.3e-3	6.1e-4	2.6e-3	8.9e-3	1.0e-3	2.5e-3	5.7e-3
ETTh1	**0.3971**	0.6374	0.5467	**0.3907**	0.6308	0.5046	**0.3904**	0.6203	0.4575	**0.3874**	0.6122	0.4314
Standard deviation	3.4e-3	4.6e-3	4.9e-3	1.8e-3	4.4e-3	5.1e-3	1.5e-3	3.8e-3	2.8e-3	1.4e-3	3.9e-3	2.1e-3
ETTh2	**0.3339**	0.4576	0.4186	**0.3359**	0.4554	0.4008	**0.3365**	0.4515	0.3769	**0.3367**	0.4485	0.3674
Standard deviation	1.2e-3	3.9e-3	8.9e-4	2.3e-3	3.7e-3	4.2e-4	1.7e-3	3.6e-3	1.3e-3	2.6e-3	3.2e-3	1.3e-3
ETTm1	**0.3753**	0.7116	0.5149	**0.3674**	0.6949	0.4256	**0.3646**	0.6801	0.3959	**0.3642**	0.6701	0.3869
Standard deviation	1.3e-3	8.6e-3	8.9e-4	8.2e-4	7.8e-3	1.8e-3	3.1e-4	7.1e-3	1.5e-4	8.2e-4	6.8e-3	5.4e-4
ETTm2	**0.2544**	0.3941	0.3458	**0.2534**	0.3896	0.3057	**0.2539**	0.3852	0.2866	**0.2533**	0.3811	0.2807
Standard deviation	1.6e-3	3.5e-3	1.1e-3	1.3e-3	3.3e-3	1.0e-3	5.0e-4	3.1e-3	1.4e-3	4.1e-4	2.9e-3	1.4e-3
Weather	**0.1939**	0.3497	0.2289	**0.193**	0.3381	0.2038	**0.193**	0.3282	0.1996	**0.1935**	0.3161	0.2025
Standard deviation	2.3e-4	2.4e-3	3.0e-3	1.4e-4	1.9e-3	6.9e-4	1.3e-4	1.5e-3	4.3e-4	3.0e-4	1.4e-3	9.5e-4
Electricity	**0.2273**	0.4226	0.2394	**0.2263**	0.3427	0.2291	**0.226**	0.3184	0.2274	**0.225**	0.3021	0.2266
Standard deviation	9.1e-5	7.8e-4	3.2e-4	3.2e-4	3.1e-4	1.2e-4	1.8e-4	1.5e-4	1.6e-4	4.5e-4	4.5e-5	2.6e-5
Traffic	**0.2537**	0.4449	0.2639	**0.2501**	0.3808	0.2557	**0.2488**	0.3548	0.2515	**0.2473**	0.3404	0.2511
Standard deviation	7.5e-5	1.3e-3	2.8e-5	1.0e-3	3.1e-4	1.9e-4	1.4e-3	9.9e-5	3.3e-5	1.6e-3	2.0e-4	1.8e-3
Avg. Improvements	-	43.28%	23.27	-	40.34%	16.19	-	38.52%	11.46	-	37.14%	9.14
Max. Improvements	-	65.1%	60.81	-	65.08%	51.28	-	65.02%	41.97	-	64.61%	32.25

Table 32: Full standard deviation and **MAE** of comparing our smoothed full fine-tuning (SFF) with linear-probing (LP) and linear-probing then full fine-tuning (LPFF) under **25%, 50%, 75%, and 100%** proportion of available data.

Data proportion	25%			50%			75%			100%		
Methods	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF	SFF	LP	LPFF
Exchange	**0.2004**	0.567	0.2689	**0.2003**	0.5476	0.2251	**0.2026**	0.5206	0.2204	**0.2013**	0.505	0.2151
Standard deviation	5.3e-4	2.3e-3	5.3e-3	1.0e-3	2.3e-3	3.7e-3	1.7e-3	2.2e-3	2.5e-3	5.3e-4	2.0e-3	2.0e-3
ETTh1	**0.3883**	0.6078	0.4238	**0.3883**	0.5753	0.4091	**0.3892**	0.5457	0.4127	**0.394**	0.5301	0.4074
Standard deviation	4.5e-4	3.7e-3	1.3e-3	3.0e-4	3.0e-3	1.5e-3	2.4e-3	2.8e-3	9.0e-4	1.4e-3	2.7e-3	6.8e-4
ETTh2	**0.3343**	0.4463	0.3571	**0.3346**	0.4337	0.3536	**0.339**	0.4245	0.3599	**0.3359**	0.4163	0.3539
Standard deviation	1.3e-3	3.1e-3	1.3e-3	1.4e-3	2.4e-3	1.4e-3	1.8e-4	1.8e-3	1.4e-3	5.7e-4	1.5e-3	8.3e-4
ETTm1	**0.3623**	0.6584	0.3794	**0.3602**	0.6101	0.3731	**0.3595**	0.5709	0.3797	**0.358**	0.5444	0.3748
Standard deviation	1.7e-3	6.2e-3	3.5e-4	1.8e-3	4.4e-3	5.6e-4	2.0e-3	3.4e-3	7.2e-4	1.6e-3	2.9e-3	6.7e-4
ETTm2	**0.2499**	0.3768	0.2661	**0.2504**	0.3601	0.2653	**0.2539**	0.3494	0.2797	**0.2493**	0.3375	0.2683
Standard deviation	6.4e-4	2.7e-3	1.2e-3	5.8e-4	2.1e-3	9.2e-4	3.8e-4	1.7e-3	1.3e-3	1.5e-3	1.4e-3	2.3e-3
Weather	**0.1925**	0.3125	0.1999	**0.193**	0.2815	0.2002	**0.1975**	0.252	0.2164	**0.1934**	0.2459	0.2049
Standard deviation	2.9e-4	1.0e-3	5.9e-4	2.1e-4	6.9e-4	1.4e-3	2.9e-4	6.3e-4	2.3e-3	1.5e-3	5.2e-4	1.5e-3
Electricity	**0.2245**	0.2939	0.2268	**0.2247**	0.2704	0.2266	**0.2245**	0.2578	0.2278	**0.2239**	0.2516	0.2266
Standard deviation	4.2e-4	4.7e-6	1.7e-4	2.7e-4	5.4e-5	1.1e-3	4.3e-4	1.0e-4	6.0e-4	4.2e-4	7.9e-5	3.9e-4
Traffic	**0.2463**	0.3316	0.2532	**0.2486**	0.3062	0.2465	**0.2485**	0.2951	0.2498	**0.245**	0.2902	0.2494
Standard deviation	2.2e-3	2.9e-4	1.4e-3	1.4e-3	2.5e-4	4.7e-4	4.6e-4	1.4e-4	3.5e-4	0.0e+00	8.5e-5	1.8e-4
Avg. Improvements	-	36.53%	7.28	-	32.17%	4.27	-	28.07%	5.6	-	26.68%	4.36
Max. Improvements	-	64.66%	25.47	-	63.42%	11.02	-	61.08%	9.22	-	60.14%	7.08

Table 33: MAE of Smoothing the loss landscape then perform zero-shot forecasting.

	ETTh1		ETTh2		ETTm1		ETTm2		Weather		Electricity		Traffic	
	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth	Timer	+Smooth
MAE	0.434	**0.418**	0.359	**0.352**	0.61	**0.607**	0.3	**0.294**	0.236	**0.231**	0.312	**0.308**	0.343	**0.335**
Std.	±0	±5.0e-4	±0	±1.1e-3	±0	±2.7e-3	±0	±1.2e-3	±0	±1.1e-3	±0	±5.4e-4	±0	±5.2e-4
Imp.	-	3.69%	-	1.95%	-	0.49%	-	2.0%	-	2.12%	-	1.28%	-	2.33%
	TimesFM	+Smooth	TimesFM	+Smooth	TimesFM	+Smooth	Tim.FM	+Smooth	Time.FM	+Smooth	Tim.FM	+Smooth	Tim.FM	+Smooth
MAE	0.559	**0.55**	0.541	**0.419**	0.749	**0.682**	0.404	**0.335**	0.278	**0.262**	0.756	**0.747**	0.867	**0.834**
Std.	±0	±1.4e-3	±0	±2.1e-3	±0	±3.0e-3	±0	±8.8e-4	±0	±3.8e-3	±0	±4.0e-3	±0	±4.0e-3
Imp.	-	1.61%	-	22.55%	-	8.95%	-	17.08%	-	5.76%	-	1.19%	-	3.81%