A Dataset

We evaluate the performance of TFPS on eight widely used datasets, including four ETT datasets (ETTh1, ETTh2, ETTm1 and ETTm2), Exchange, Weather, Electricity, and ILI. This subsection provides a summary of the datasets:

- **ETT** [1] (Zhou et al., 2021) (Electricity Transformer Temperature) dataset contains two electric transformers, ETT1 and ETT2, collected from two separate counties. Each of them has two versions of sampling resolutions (15min & 1h). Thus, there are four ETT datasets: **ETTm1**, **ETTm2**, **ETTh1**, and **ETTh2**.
- **Exchange-Rate** [2] (Lai et al., 2018) the exchange-rate dataset contains the daily exchange rates of eight foreign countries including Australia, British, Canada, Switzerland, China, Japan, New Zealand, and Singapore ranging from 1990 to 2016.
- **Weather** [3] (Wu et al., 2021) dataset contains 21 meteorological indicators in Germany, such as humidity and air temperature.
- **Electricity** [4] (Wu et al., 2021) is a dataset that describes 321 customers' hourly electricity consumption.
- **ILI** [5] (Wu et al., 2021) dataset collects the number of patients and influenza-like illness ratio in a weekly frequency.

For the data split, we follow Zeng et al. (2023) and split the data into training, validation, and testing by a ratio of 6:2:2 for the ETT datasets and 7:1:2 for the others. Details are shown in Table 5. The best parameters are selected based on the lowest validation loss and then applied to the test set for performance evaluation.

Table 5: The statistics of the datasets.

Datasets	Variates	Prediction Length	Timesteps	Granularity	Average MMD* (Time Domain)	Average MMD* (Frequency Domain)
ETTh1	7	{96, 192, 336, 720}	17,420	1 hour	0.938	0.340
ETTh2	7	{96, 192, 336, 720}	17,420	1 hour	0.582	0.635
ETTm1	7	{96, 192, 336, 720}	69,680	15 min	1.371	0.328
ETTm2	7	{96, 192, 336, 720}	69,680	15 min	1.213	0.815
Exchange-Rate	8	{96, 192, 336, 720}	7,588	1 day	0.805	0.485
Weather	21	{96, 192, 336, 720}	52,696	10 min	0.129	0.236
Electricity	321	{96, 192, 336, 720}	26,304	1 hour	0.026	0.005
ILI	7	{24, 36, 48, 60}	966	1 week	0.125	0.234

* A large MMD indicates a more severe drift.

B Maximum Mean Discrepancy

Maximum mean discrepancy (MMD) is a kernel-based statistical test used to determine whether given two distribution are the same. Given an X, the feature map ϕ transforms X to an another space $\mathcal{H}$ such that $\phi(X) \in \mathcal{H}$. $\mathcal{H}$ is Reproducing Kernel Hilbert Space (RKHS) and we can leverage the kernel trick to compute inner products in $\mathcal{H}$:

$$X, Y \quad \text{such that} \quad k(X, Y) = \langle \phi(X), \phi(Y) \rangle_{\mathcal{H}}. \tag{13}$$

Feature means. The mean embeddings of a probability distribution P is a feature map that transforms $\phi(X)$ into the mean of each coordinate of $\phi(X)$:

$$\mu_P(\phi(X)) = [\mathbb{E}[\phi(X_1)], \cdots, \mathbb{E}[\phi(X_m)]]^T. \tag{14}$$

[1] https://github.com/zhouhaoyi/ETDataset
[2] https://github.com/laiguokun/multivariate-time-series-data
[3] https://www.bgc-jena.mpg.de/wetter/
[4] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[5] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

The inner product of the mean embeddings of $X \sim P$ and $Y \sim Q$ can be written in terms of kernel function:

$$\langle \mu_P(\phi(X)), \mu_Q(\phi(Y)) \rangle_{\mathcal{H}} = \mathbb{E}_{P,Q}[\langle \phi(X), \phi(Y) \rangle_{\mathcal{H}}] = \mathbb{E}_{P,Q}[k(X,Y)]. \tag{15}$$

Maximum mean discrepancy. The MMD measures the distance between the mean embeddings of two samples, X and Y, in the RKHS:

$$\mathrm{MMD}^2(P,Q) = \|\mu_P - \mu_Q\|^2_{\mathcal{H}}, \tag{16}$$

For convenience we omit the $\phi(\cdot)$ terms. If we use the norm induced by the inner product such that $\|x\| = \sqrt{\langle x, x \rangle}$, the Eq. 16 becomes:

$$\mathrm{MMD}^2(P,Q) = \langle \mu_p - \mu_Q, \mu_p - \mu_Q \rangle = \langle \mu_p, \mu_p \rangle - 2\langle \mu_p, \mu_Q \rangle + \langle \mu_Q, \mu_Q \rangle. \tag{17}$$

Using the Eq. 15, finally above expression becomes:

$$\mathrm{MMD}^2(P,Q) = \mathbb{E}_P[k(X,X)] - 2\mathbb{E}_{P,Q}[k(X,Y)] + \mathbb{E}_Q[k(Y,Y)]. \tag{18}$$

Empirical estimation of MMD. In real-world applications, the underlying distribution are usually unknown. Thus, an empirical estimate of Eq. 18 can be used:

$$\mathrm{MMD}^2(X,Y) = \frac{1}{m(m-1)} \sum_{i \neq j} k\left(x_i, x_j\right) - \frac{2}{mn} \sum_{i,j} k\left(x_i, x_j\right) + \frac{1}{n(n-1)} \sum_{i \neq j} k\left(y_i, y_j\right), \tag{19}$$

where x_i and x_j are samples from P, y_i and y_j are samples from Q, and $k(x,y)$ is the kernel function, often the Gaussian (RBF) kernel.

C Distribution shifts in both time and frequency domains

The time series $\mathcal{X}$ is segmented into N patches, where each patch $\mathcal{P}_n = \{x_{n1}, x_{n2}, \ldots, x_{nP}\}$ consists of P consecutive timesteps for $n = 1, 2, \cdots, N$. For the frequency domain, we apply a Fourier transform $\mathcal{F}$ to each patch $\mathcal{P}_n$, obtaining its frequency-domain representation as $\hat{\mathcal{P}}_n = \mathcal{F}(\mathcal{P}_n)$.

Each patch's probability distribution in the time domain is denoted as $p_t(\mathcal{P}_n)$, representing the statistical properties of $\mathcal{P}_n$, while its frequency domain distribution, denoted as $p_f(\hat{\mathcal{P}}_n)$, captures its spectral characteristics.

The distribution shifts between two patches $\mathcal{P}_i$ and $\mathcal{P}_j$ are characterized by the comparing their probability distributions in both time and frequency domains. These shifts are defined as:

$$\mathcal{D}_t(\mathcal{P}_i, \mathcal{P}_j) = |d(p_t(\mathcal{P}_i), p_t(\mathcal{P}_j))| > \theta, \tag{20}$$

$$\mathcal{D}_f(\hat{\mathcal{P}}_i, \hat{\mathcal{P}}_j) = |d(p_f(\hat{\mathcal{P}}_i), p_f(\hat{\mathcal{P}}_j))| > \theta, \tag{21}$$

where d is a distance metric, such as MMD values or Kullback-Leibler divergence, and θ is a threshold indicating a significant distribution shift. If $\mathcal{D}_t(\mathcal{P}_i, \mathcal{P}_j)$ or $\mathcal{D}_f(\hat{\mathcal{P}}_i, \hat{\mathcal{P}}_j)$ exceeds θ, this implies a significant distribution shift between the two patches in either domain.

D Related Work

Mixture-of-Experts. Mixture-of-Experts (MoE) models have gained attention for their ability to scale efficiently by activating only a subset of experts for each input, as first introduced by Shazeer et al. (2017). Despite their success, challenges such as training instability, expert redundancy, and limited expert specialization have been identified (Puigcerver et al., 2023; Dai et al., 2024). These issues hinder the full potential of MoE models in real-world tasks.

Recent advances have integrated MoE with Transformers to improve scalability and efficiency. For example, GLaM (Du et al., 2022) and Switch Transformer (Fedus et al., 2022) interleave MoE layers with Transformer blocks, reducing computational costs. Other models like state space models

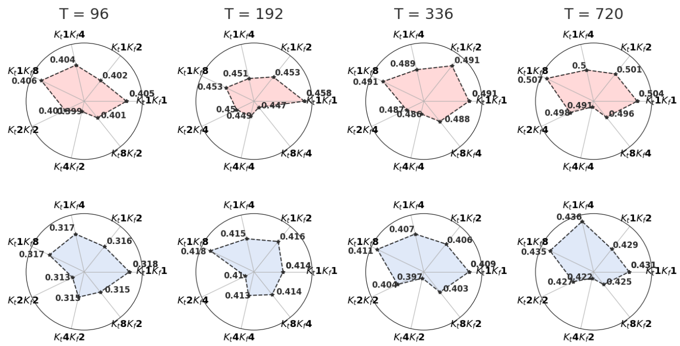


Figure 7: Results of expert number experiments for ETTh1 and ETTh2.

(SSMs) (Pióro et al., 2024; Anthony et al., 2024), (Alkilane et al., 2024) combines MoE with alternative architectures for enhanced scalability and inference speed.

In contrast, our approach introduces MoE into time series forecasting by assigning experts to specific time-frequency patterns, enabling more effective, patch-level adaptation. This approach represents a significant innovation in time series forecasting, offering a more targeted and effective way to handle varying patterns across both time and frequency domains.

E More Model Analysis

E.1 Analysis of experts

Detailed Results on the Number of Experts.

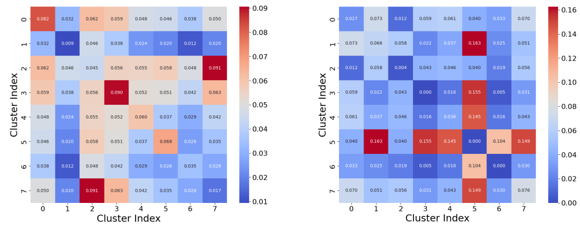
We provide the full results on the number of experts for the ETTh1 and ETTh2 dataset in Figure 7.

In Figure 6, we set the learning rate to 0.0001 and conducted four sets of experiments on the ETTh1 and ETTh2 datasets, $K_t = 1$, $K_f = \{1, 2, 4, 8\}$, to explore the effect of the number of frequency experts on the results. For example, $K_t 1 K_f 4$ means that the TFPS contains 1 time experts and 4 frequency experts. We observed that $K_t 1 K_f 2$ outperformed $K_t 1 K_f 4$ in both cases, suggesting that increasing the number of experts does not always lead to better performance.

In addition, we conducted three experiments based on the optimal number of frequency experts to verify the impact of varying the number of time experts on the results. As shown in Figure 7, the best results for ETTh1 were obtained with $K_t 4 K_f 2$, $K_t 8 K_f 4$, $K_t 4 K_f 4$, $K_t 4 K_f 4$, while for ETTh2, the optimal results were achieved with $K_t 2 K_f 2$, $K_t 2 K_f 4$, $K_t 4 K_f 2$ and $K_t 4 K_f 2$. Combined with the average MMD in Table 5, we attribute this to the fact that, in cases where concept drift is more severe, such as ETTh1 in the time domain, more experts are needed, whereas fewer experts are sufficient when the drift is less severe.

Comparing Inter- and Intra-Cluster Differences via MMD.

We present the heatmaps of inter-cluster and intra-cluster MMD values obtained using linear layers and PI in Figure 8. The diagonal elements represent the average MMD values of patches within the same clusters. If these values are small, it indicates that the difference of patches within the same cluster is relatively similar. The off-diagonal elements represent the average MMD values between patches from different clusters, where larger values mean significant differences between the clusters. We observe that when using PI, the intra-cluster drift is smaller, while the inter-cluster shift is more pronounced compared to the linear layer. This indicates that our identifier effectively classifies and distinguishes between different patterns.



(a) Linear layer (b) Pattern Identifier

Figure 8: Heatmap showing the MMD values of inter- and intra-cluster patches on ETTh1.

Table 6: Detailed results of the comparison between TFPS and normalization methods. The best results are highlighted in **bold** and the second best are underlined.

Model		IMP.	TFPS (Our)		FEDformer + SIN (2024a)		FEDformer + SAN (2023b)		FEDformer + Dish-TS (2023)		FEDformer + NST (2022)		FEDformer + RevIN (2021)	
Metric		MSE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	-1.0%	0.398	0.413	0.413	**0.372**	**0.383**	0.409	0.390	0.424	0.394	0.414	0.392	0.413
	192	3.8%	**0.423**	0.423	0.443	**0.417**	0.431	0.438	0.441	0.458	0.441	0.442	0.443	0.444
	336	-0.3%	0.484	0.461	**0.465**	**0.448**	0.471	0.456	0.495	0.486	0.485	0.466	0.495	0.467
	720	4.5%	**0.488**	**0.476**	0.509	0.490	0.504	0.488	0.519	0.509	0.505	0.496	0.520	0.498
ETTh2	96	31.3%	0.313	**0.355**	0.412	0.357	**0.300**	0.355	0.806	0.589	0.381	0.403	0.380	0.402
	192	26.0%	0.405	**0.410**	0.472	0.453	**0.392**	0.413	0.936	0.659	0.478	0.453	0.457	0.443
	336	36.7%	**0.392**	**0.415**	0.527	0.527	0.459	0.462	1.039	0.702	0.561	0.499	0.515	0.479
	720	37.9%	**0.410**	**0.433**	0.593	0.639	0.462	0.472	1.237	0.759	0.502	0.481	0.507	0.487
ETTm1	96	4.1%	0.327	0.367	0.373	**0.320**	**0.311**	0.355	0.348	0.397	0.336	0.382	0.340	0.385
	192	2.9%	0.374	0.395	0.394	**0.366**	**0.351**	0.383	0.406	0.428	0.386	0.409	0.390	0.411
	336	5.3%	0.401	0.408	0.418	**0.405**	**0.390**	0.407	0.438	0.450	0.438	0.441	0.432	0.436
	720	-0.5%	0.479	0.456	**0.451**	0.475	0.456	**0.444**	0.497	0.481	0.483	0.460	0.497	0.466
ETTm2	96	33.5%	**0.170**	0.255	0.326	**0.211**	0.175	0.266	0.394	0.395	0.191	0.272	0.192	0.272
	192	32.3%	**0.235**	**0.296**	0.402	0.316	0.246	0.315	0.552	0.472	0.270	0.321	0.270	0.320
	336	35.0%	**0.297**	**0.335**	0.465	0.399	0.315	0.362	0.808	0.601	0.353	0.371	0.348	0.367
	720	35.9%	**0.401**	**0.397**	0.555	0.547	0.412	0.422	1.282	0.771	0.445	0.422	0.430	0.415
Weather	96	28.4%	**0.154**	**0.202**	0.280	0.215	0.179	0.239	0.244	0.317	0.187	0.234	0.187	0.234
	192	23.3%	**0.205**	**0.249**	0.314	0.264	0.234	0.296	0.320	0.380	0.235	0.272	0.235	0.272
	336	19.8%	**0.262**	**0.289**	0.329	0.293	0.304	0.348	0.424	0.452	0.289	0.308	0.287	0.307
	720	18.4%	**0.344**	**0.342**	0.382	0.370	0.400	0.404	0.604	0.553	0.359	0.352	0.361	0.353
1st (2nd) Count			24 (8)		9 (4)		7 (24)		0 (1)		0 (1)		0 (2)	

E.2 Results of the Comparison between TFPS and Normalization Methods

In this section, we provide the detailed experimental results of the comparison between TFPS and five state-of-the-art normalization methods for non-stationary time series forecasting: SIN (Han et al., 2024a), SAN (Liu et al., 2023b), Dish-TS (Fan et al., 2023), Non-Stationary Transformers (NST) (Liu et al., 2022), and RevIN (Kim et al., 2021). The results of SIN are from Han et al. (2024a), other results are from Liu et al. (2023b). We report the evaluation of FEDformer over all the forecasting lengths for each dataset and the relative improvements in Table 6. It can be concluded that TFPS achieves the best performance among existing methods in most cases. The improvement is significant with an average MSE decrease of 18.9%. We attribute this improvement to the accurate identification of pattern groups and the provision of specialized experts for each group, thereby avoiding the over-stationarization problem often associated with normalization methods.

F Metric Illustration

We use mean square error (MSE) and mean absolute error (MAE) as our metrics for evaluation of all forecasting models. Then calculation of MSE and MAE can be described as:

$$\mathrm{MSE} = \frac{1}{H}\sum_{i=L+1}^{L+H}(\hat{Y}_i - Y_i)^2, \tag{22}$$

$$\mathrm{MAE} = \frac{1}{H}\sum_{i=L+1}^{L+H}\left|\hat{Y}_i - Y_i\right|, \tag{23}$$

where $\hat{Y}$ is predicted vector with H future values, while Y is the ground truth.

G Algorithm of TFPS

We provide the pseudo-code of TFPS in Algorithm 1.

H Broader Impact

Real-world applications. TFPS addresses the crucial challenge of time series forecasting, which is a valuable and urgent demand in extensive applications. Our method achieves consistent state-of-the-art performance in four real-world applications: electricity, weather, exchange rate, illness. Researchers in these fields stand to benefit significantly from the enhanced forecasting capabilities of TFPS. We believe that improved time series forecasting holds the potential to empower decision-making and proactively manage risks in a wide array of societal domains.

Academic research. TFPS draws inspiration from classical time series analysis and stochastic process theory, contributing to the field by introducing a novel framework with the assistance pattern recognition. This innovative architecture and its associated methodologies represent significant advancements in the field of time series forecasting, enhancing the model's ability to address distribution shifts and complex patterns effectively.

Model Robustness. Extensive experimentation with TFPS reveals robust performance without exceptional failure cases. Notably, TFPS exhibits impressive results and maintains robustness in datasets with distribution shifts. The pattern identifier structure within TFPS groups the time series into distinct patterns and adopts a mixture of pattern experts for further prediction, thereby alleviating prediction difficulties. However, it is essential to note that, like any model, TFPS may face challenges when dealing with unpredictable patterns, where predictability is inherently limited. Understanding these nuances is crucial for appropriately applying and interpreting TFPS's outcomes.

Our work only focuses on the scientific problem, so there is no potential ethical risk.

I Limitations

Though TFPS demonstrates promising performance on the benchmark dataset, there are still some limitations of this method. First, the patch length is primarily chosen heuristically, and the current design struggles with handling indivisible lengths or multi-period characteristics in time series. While this approach works well in experiments, it lacks generalizability for real-world applications. Second, the real-world time series data undergo expansion, implying that the new patterns continually emerge over time, such as an epidemic or outbreak that had not occurred before. Therefore, future work will focus on developing a more flexible and automatic patch length selection mechanism, as well as an extensible solution to address these evolving distribution shifts.

Algorithm 1 Time-Frequency Pattern-Specific architecture - Overall Architecture.

Input: Input lookback time series $X \in \mathbb{R}^{L \times C}$; input length L; predicted length H; variables number C; patch length P; feature dimension D; encoder layers number n; random Gaussian distribution-initialized subspace $\mathbf{D} = [\mathbf{D}^{(1)}, \mathbf{D}^{(2)}, \cdots, \mathbf{D}^{(K)}]$, each $\mathbf{D}^{(j)} \in \mathbf{R}^{q \times d}$, where $q = C \times D$ and $d = q/K$. Technically, we set D as 512, n as 2.

Output: The prediction result $\hat{Y}$.

$X = X.\mathtt{transpose}$ ▷ $X \in \mathbb{R}^{C \times L}$

$X_{PE} = \mathtt{Patch}\,(X) + \mathtt{Position\ Embedding}$ ▷ $X_t^0 \in \mathbb{R}^{C \times N \times D}$

▷ Time Encoder.

$X_t^0 = X_{PE}$

for l **in** $\{1, \dots, n\}$**:**

  $X_t^{l-1} = \mathtt{LayerNorm}\,(X_t^{l-1} + \mathtt{Self\text{-}Attn}\,(X_t^{l-1}))$. ▷ $X_t^{l-1} \in \mathbb{R}^{C \times N \times D}$

  $X_t^{l} = \mathtt{LayerNorm}\,(X_t^{l-1} + \mathtt{Feed\text{-}Forward}\,(X_t^{l-1}))$. ▷ $X_t^{l} \in \mathbb{R}^{C \times N \times D}$

End for

$z_t = X_t^l$ ▷ $z_t^l \in \mathbb{R}^{C \times N \times D}$

▷ Pattern Identifier for Time Domain.

$s_t = \mathtt{Subspace\ affinity}\,(z_t, \mathbf{D})$ ▷ Eq. 6 of the paper $s_t \in \mathbb{R}^{C \times N \times D}$

$\widetilde{s}_t = \mathtt{Subspace\ refinement}\,(s_t)$ ▷ Eq. 7 of the paper $\widetilde{s}_t \in \mathbb{R}^{C \times N \times D}$

▷ Mixture of Temporal Pattern Experts.

$G(s) = \mathtt{Softmax}\,(\mathtt{TopK}\,(s_t))$

$h_t = \sum_{k=1}^{K} G(s)\mathtt{MLP}_k(z_t)$ ▷ Eq. 10 and Eq. 11 of the paper $h_t \in \mathbb{R}^{C \times N \times D}$

▷ Frequency Encoder.

$X_f^0 = X_{PE}$ ▷ Eq. 2 of the paper $X_f^0 \in \mathbb{R}^{C \times N \times P}$

for l **in** $\{1, \dots, n\}$**:**

  $X_f^{l-1} = \mathtt{LayerNorm}\,(X_f^{l-1} + \mathtt{Fourier}\,(X_f^{l-1}))$. ▷ $X_f^{l-1} \in \mathbb{R}^{C \times N \times D}$

  $X_f^{l} = \mathtt{LayerNorm}\,(X_f^{l-1} + \mathtt{Feed\text{-}Forward}\,(X_f^{l-1}))$. ▷ $X_f^{l} \in \mathbb{R}^{C \times N \times D}$

End for

$z_f = X_f^l$ ▷ $z_f^n \in \mathbb{R}^{C \times N \times D}$

▷ Pattern Identifier for Frequency Domain.

$s_f = \mathtt{Subspace\ affinity}\,(z_f, \mathbf{D})$ ▷ Eq. 6 of the paper $s_f \in \mathbb{R}^{C \times N \times D}$

$\widetilde{s}_f = \mathtt{Subspace\ refinement}\,(s_f)$ ▷ Eq. 7 of the paper $\widetilde{s}_f \in \mathbb{R}^{C \times N \times D}$

▷ Mixture of Frequency Pattern Experts.

$G(s) = \mathtt{Softmax}\,(\mathtt{TopK}\,(s_f))$

$h_f = \sum_{k=1}^{K} G(s)\mathtt{MLP}_k(z_f)$ ▷ Eq. 10 and Eq. 11 of the paper $h_f \in \mathbb{R}^{C \times N \times D}$

$h = \text{Concat}(h_t, h_f)$ ▷ $h \in \mathbb{R}^{C \times N \times 2*D}$

for c **in** $\{1, \dots, C\}$**:**

  $\hat{Y} = \mathtt{Linear}\,(\mathtt{Flatten}\,(h))$. ▷ Project tokens back to predicted series $\hat{Y} \in \mathbb{R}^{C \times H}$

End for

$\hat{Y} = \hat{Y}.\mathtt{transpose}$ ▷ $\hat{Y} \in \mathbb{R}^{H \times C}$

Return $\hat{Y}$ ▷ Output the final prediction $\hat{Y} \in \mathbb{R}^{H \times C}$

Table 7: Multivariate long-term forecasting results for Traffic. The input lengths is $L = 96$. The best results are highlighted in **bold** and the second best are underlined.

Model		IMP.	TFPS (Our)		TSLANet (2024)		FITS (2024)		iTransformer (2024a)		TFDNet-IK (2023)		PatchTST (2023)		TimesNet (2023a)		DLinear (2023)		FEDformer (2022)	
Metric		MSE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	21.1%	**0.427**	0.296	0.475	0.307	0.651	0.388	0.428	**0.295**	0.519	0.314	0.446	0.284	0.586	0.316	0.650	0.397	0.575	0.357
	192	17.7%	**0.445**	**0.298**	0.478	0.306	0.603	0.364	0.448	0.302	0.513	0.314	0.453	0.285	0.618	0.323	0.600	0.372	0.613	0.381
	336	17.0%	**0.459**	**0.307**	0.494	0.312	0.610	0.366	0.465	0.311	0.525	0.319	0.467	0.291	0.634	0.337	0.606	0.374	0.622	0.380
	720	15.1%	**0.496**	**0.313**	0.528	0.331	0.648	0.387	0.501	0.333	0.561	0.336	0.501	0.492	0.659	0.349	0.646	0.396	0.630	0.383
1st Count			7		0		0		1		0		0		0		0		0	

Table 8: Experiment results under hyperparameter searching for the long-term forecasting task. The best results are highlighted in **bold** and the second best are underlined.

Model		IMP.	TFPS (Our)		TSLANet (2024)		FITS (2024)		iTransformer (2024a)		TFDNet-IK (2023)		PatchTST (2023)		TimesNet (2023a)		Dlinear (2023)		FEDformer (2022)	
Metric		MSE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	1.5%	0.372	0.404	0.368	0.394	0.374	0.395	0.387	0.405	**0.360**	**0.387**	0.375	0.400	0.389	0.412	0.384	0.405	0.385	0.425
	192	5.7%	**0.401**	**0.410**	0.413	0.418	0.407	0.414	0.441	0.436	0.403	0.412	0.414	0.421	0.441	0.442	0.443	0.450	0.441	0.461
	336	9.8%	**0.409**	**0.402**	0.412	0.416	0.429	0.428	0.491	0.463	0.434	0.429	0.432	0.436	0.491	0.467	0.447	0.448	0.491	0.473
	720	11.2%	**0.423**	**0.433**	0.473	0.477	0.425	0.446	0.509	0.494	0.437	0.452	0.450	0.466	0.512	0.491	0.504	0.515	0.501	0.499
ETTh2	96	9.3%	**0.268**	**0.325**	0.283	0.344	0.274	0.337	0.301	0.350	0.271	0.329	0.278	0.336	0.324	0.368	0.290	0.353	0.342	0.383
	192	10.4%	**0.329**	0.376	0.331	0.378	0.337	0.377	0.380	0.399	0.333	**0.372**	0.339	0.380	0.393	0.410	0.388	0.422	0.434	0.440
	336	17.7%	0.329	0.401	**0.319**	**0.377**	0.360	0.398	0.424	0.432	0.361	0.396	0.336	0.380	0.429	0.437	0.463	0.473	0.512	0.497
	720	9.0%	0.412	0.441	0.407	0.449	0.386	0.423	0.430	0.447	0.382	**0.418**	**0.382**	0.421	0.433	0.448	0.733	0.606	0.467	0.476
ETTm1	96	10.2%	**0.281**	**0.329**	0.291	0.353	0.303	0.345	0.342	0.377	0.283	0.330	0.288	0.342	0.337	0.377	0.301	0.345	0.360	0.406
	192	8.5%	**0.324**	**0.354**	0.329	0.372	0.337	0.365	0.383	0.396	0.327	0.356	0.334	0.372	0.395	0.406	0.336	0.366	0.395	0.427
	336	8.2%	0.359	0.404	**0.357**	0.392	0.372	0.385	0.418	0.418	0.361	**0.375**	0.367	0.393	0.433	0.432	0.372	0.389	0.448	0.458
	720	8.2%	**0.409**	**0.408**	0.423	0.425	0.428	0.416	0.487	0.457	0.411	0.409	0.417	0.422	0.484	0.458	0.427	0.423	0.491	0.479
ETTm2	96	8.9%	**0.158**	**0.243**	0.167	0.256	0.165	0.255	0.186	0.272	0.158	0.244	0.164	0.253	0.182	0.262	0.172	0.267	0.193	0.285
	192	5.7%	0.222	0.302	0.221	0.294	0.220	0.291	0.254	0.314	**0.219**	**0.282**	0.221	0.292	0.252	0.307	0.237	0.314	0.256	0.324
	336	8.5%	**0.268**	**0.316**	0.277	0.329	0.274	0.326	0.316	0.351	0.273	0.317	0.277	0.329	0.312	0.346	0.295	0.359	0.321	0.364
	720	12.0%	**0.344**	**0.373**	0.356	0.382	0.367	0.383	0.414	0.407	0.346	0.374	0.365	0.384	0.417	0.404	0.427	0.439	0.434	0.426
1st Count			21		3		0		0		7		1		0		0		0	

J TRAFFIC RESULTS

We conducted addition experiments on high-dimensional Traffic dataset to further evaluate the performance and generalizability of TFPS, as shown in Table 7.

K HYPERPARAMETER-SEARCH RESULTS

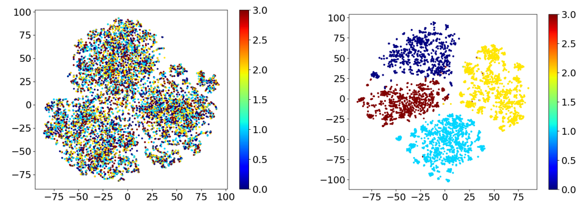
To ensure a fair comparison between models, we conducted experiments using unified parameters $L = 96$ and reported results in the main text.

In addition, considering that the reported results in different papers are mostly obtained through hyperparameter search, we provide the experiment results with the full version of the parameter search. We searched for input length among 96, 192, 336, and 512. The results are included in Table 8. All baselines are reproduced by their official code.

We can find that the relative promotion of TFPS over TFDNet is smaller under comprehensive hyperparameter search than the unified hyperparameter setting. It is worth noticing that TFPS runs much faster than TFDNet according to the efficiency comparison in Table 11. Therefore, considering performance, hyperparameter-search cost and efficiency, we believe TFPS is a practical model in real-world applications and is valuable to deep time series forecasting community.

L VISUALIZATION OF CLUSTERING

Figure 9 presents the t-SNE visualization of the learned embedded representation on the ETTh1. In the Figure 9 (a), where the pattern identifier is replaced with a linear layer, the representation lacks clear clustering structures, resulting in scattered and indistinct groupings. In contrast, Figure 9 (b) shows the visualization of the representation learned by the proposed method, which effectively captures discriminative features and reveals significantly clearer clustering patterns.



(a) Linear (b) Pattern Identifier

Figure 9: Visualization of the embedded representations with t-SNE on ETTh1. The left figure shows the visualization when the Patch Identifier is replaced with a Linear Layer for comparison, while the right figure shows the visualization of the proposed method.

Table 9: Comparison between TFPS and MoE-based methods. The best results are highlighted in **bold** and the second best are underlined.

Model		IMP.	TFPS (Our)		MoLE 2024		MoU 2024		KAN4TSF 2024b	
Metric		MSE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	-4.3%	0.398	0.413	0.383	**0.392**	**0.381**	0.403	0.382	0.400
	192	1.7%	**0.423**	**0.423**	0.434	0.426	0.429	0.430	0.430	0.426
	336	1.6%	**0.484**	**0.461**	0.489	0.478	0.488	0.463	0.498	0.467
	720	8.2%	**0.488**	**0.476**	0.602	0.545	0.499	0.484	0.494	0.479
ETTh2	96	10.4%	**0.313**	**0.355**	0.413	0.360	0.317	0.358	0.318	0.358
	192	10.3%	**0.405**	**0.410**	0.525	0.416	0.409	0.414	0.419	0.414
	336	7.1%	**0.392**	**0.415**	0.423	0.434	0.397	0.420	0.447	0.452
	720	8.4%	**0.410**	**0.433**	0.453	0.458	0.412	0.434	0.477	0.476
ETTm1	96	13.5%	**0.327**	**0.367**	0.338	0.380	0.465	0.442	0.333	0.371
	192	10.6%	**0.374**	**0.395**	0.388	0.403	0.483	0.455	0.384	0.399
	336	11.8%	**0.401**	**0.408**	0.417	0.431	0.540	0.488	0.407	0.413
	720	7.3%	**0.479**	**0.456**	0.486	0.472	0.583	0.509	0.483	0.469
ETTm2	96	13.9%	**0.170**	**0.255**	0.238	0.271	0.179	0.263	0.175	0.260
	192	3.8%	**0.235**	**0.296**	0.247	0.305	0.243	0.303	0.244	0.305
	336	3.3%	**0.297**	**0.335**	0.308	0.343	0.306	0.343	0.308	0.347
	720	13.7%	**0.401**	**0.397**	0.583	0.419	0.405	0.404	0.405	0.404
1^{st} Count			30		1		1		0	

M COMPARED WITH MOE-BASED METHODS

As shown in Table 9, unlike MoE-based methods that rely on the Softmax function as a gating mechanism, our approach constructs a pattern recognizer to assign different experts to handle distinct patterns. This results in TFPS achieving relative improvements of 2.3%, 9.0%, 10.6%, and 9.1% across the four datasets, respectively.

N COMPARED WITH DISTRIBUTION SHIFT METHODS

As shown in Table 10, we compare with the methods for distribution shift. This results in TFPS achieving relative improvements of 6.7%, 6.6%, 4.8%, and 5.9% across the four datasets, respectively.

O EFFICIENCY ANALYSIS

To make this clearer, we present the results of ETTh1 for a prediction length of 192 from Table 2 and include additional results on runtime and computational complexity in Table 11. Due to the sparsity of MoPE, TFPS achieves a remarkable balance between performance and efficiency:

Performance Superiority: TFPS achieves an MSE of 0.423, outperforming TSLANet (0.448), FITS (0.445), PatchTST (0.460), and FEDformer (0.441). This represents a 5.6% improvement over TSLANet and a 8.0% improvement over PatchTST, highlighting its significant accuracy gains.

Table 10: Comparison between TFPS and methods for Distribution Shift. The best results are highlighted in **bold** and the second best are underlined.

Model		IMP.	TFPS (Our)		Koopa 2024b		SOLID 2024a		OneNet 2024	
Metric		MSE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	7.9%	0.398	0.413	**0.385**	0.407	0.440	0.439	0.425	**0.402**
	192	10.3%	**0.423**	**0.423**	0.445	0.434	0.492	0.466	0.452	0.443
	336	4.9%	**0.484**	**0.461**	0.489	0.460	0.525	0.481	0.492	0.482
	720	4.4%	**0.488**	**0.476**	0.497	0.480	0.517	0.496	0.504	0.496
ETTh2	96	10.6%	**0.313**	**0.355**	0.318	0.360	0.318	0.359	0.382	0.362
	192	4.7%	0.405	0.410	**0.378**	**0.398**	0.414	0.418	0.435	0.426
	336	4.8%	**0.392**	**0.415**	0.415	0.430	0.398	0.421	0.426	0.419
	720	6.8%	**0.410**	**0.433**	0.445	0.456	0.424	0.441	0.456	0.437
ETTm1	96	6.8%	**0.327**	**0.367**	0.329	0.359	0.329	0.370	0.374	0.392
	192	2.0%	**0.374**	0.395	0.380	**0.393**	0.379	0.400	0.385	0.435
	336	8.7%	**0.401**	**0.408**	0.401	0.411	0.405	0.412	0.473	0.458
	720	2.0%	0.479	0.456	**0.475**	**0.448**	0.482	0.464	0.496	0.483
ETTm2	96	5.3%	**0.170**	**0.255**	0.179	0.261	0.175	0.258	0.184	0.274
	192	3.8%	**0.235**	**0.296**	0.246	0.305	0.241	0.302	0.248	0.384
	336	3.4%	**0.297**	**0.335**	0.310	0.348	0.303	0.342	0.313	0.374
	720	9.0%	**0.401**	**0.397**	0.405	0.402	0.456	0.436	0.425	0.438
1st Count			25		6		0		1	

Table 11: The GPU memory (MB) and speed (inference time) of each model.

	TFPS	TSLANet	FITS	iTransformer	TFDNet-IK	PatchTST	TimesNet	DLinear	FEDformer
MSE	**0.423**	0.448	0.445	0.441	0.458	0.460	0.441	0.434	0.441
GPU Memory (MB)	9.643	0.481	0.019	3.304	0.246	0.205	2.345	0.142	62.191
Average Inference Time (ms)	6.457	2.100	1.202	2.949	407.853	17.851	72.196	0.789	259.001

While DLinear achieves an MSE of 0.434, TFPS still demonstrates a 2.5% relative improvement, making it the most accurate model among all baselines.

Efficiency Gains: TFPS maintains competitive runtime and memory efficiency.

- Runtime: TFPS runs in 6.457 ms, making it 2.8× faster than PatchTST (17.851 ms) and 11.2× faster than TimesNet (72.196 ms).
- Memory Usage: TFPS uses 9.643 MB of GPU memory, significantly less than FEDformer (62.191 MB) and comparable to iTransformer (3.304 MB). This makes TFPS suitable for resource-constrained applications while maintaining superior performance.

Balancing Trade-offs: While lightweight models like DLinear (0.434 MSE, 0.789 ms runtime) are slightly more efficient, TFPS delivers a substantial performance improvement of 2.5%, providing a well-rounded solution that balances accuracy and efficiency effectively.

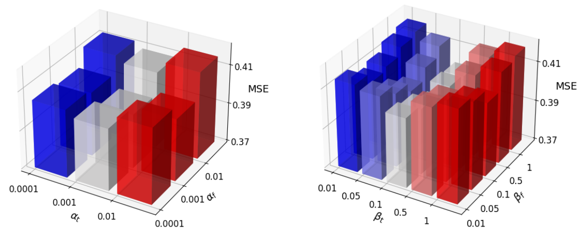
P HYPERPARAMETER SENSITIVITY

In this section, we analysis the impact of the hyperparameters α and β on the performance.

Specifically, we performed a grid search to optimize the hyperparameters $\alpha_t = \{0.0001, 0.001, 0.01\}$ and $\alpha_f = \{0.0001, 0.001, 0.01\}$, as shown in Figure 10 (a). After extensive testing, we ultimately fixed at $\alpha_t = \alpha_f = 10^{-3}$ in our experiments.

In addition, we conducted a grid search to optimize the balance factors $\beta_t = \{0.01, 0.05, 0.1, 0.5, 1\}$ and $\beta_f = \{0.01, 0.05, 0.1, 0.5, 1\}$. The performance under different parameter values is displayed in Figure 10 (b), from which we have the following observations:

- Firstly, the performance is affected when the value of β is too low, indicating that the proposed clustering objective plays a crucial role in distinguishing patterns.



(a) α (b) β

Figure 10: Parameter sensitivity of α and β of the proposed method on the ETTh1-96 dataset.

Table 12: In the table, w/ Imaginary indicates that we incorporate both the real and imaginary parts into the network.

	ETTh1				ETTh2			
	96	192	336	720	96	192	336	720
TFPS	0.398	**0.423**	**0.484**	0.488	0.313	**0.405**	0.392	0.410
w/ Imaginary	**0.397**	0.424	0.487	**0.486**	**0.312**	0.406	**0.391**	**0.399**

- Second, an excessive β also has a negative on the performance. One plausible explanation is that the excessive value influences the learning of the inherent structure of original data, resulting in a perturbation of the embedding space.
- Overall, we recommend setting β around 0.1 for optimal performance.

Q Full Ablation

Q.1 Impacts of Real/Imaginary Parts

To further validate the robustness of our approach, we adopted similar operations in FreTS to conduct experiments incorporating both the real and imaginary parts. The results in the Table 12 show that the performance of TFPS with the real part only is very similar to that when both parts are included, while requiring fewer parameters. This further reinforces the conclusion that TFPS remains highly effective even when focusing solely on the real part of the Fourier transform.

Q.2 Ablation on PI

The PI module plays a crucial role in identifying and characterizing distinct patterns within the time series data, while the gating network dynamically selects the most relevant experts for each segment. This collaborative mechanism allows the model to specialize in handling different patterns and adapt effectively to distribution shifts, thus mitigating the overfitting risks that arise from treating all data equally.

To validate the importance of PI empirically, we have conducted the ablation experiments comparing the model's performance by replacing the PI module with a linear layer in the Table 3 of main text. In addition, we supplement some ablation experiments in Table 13 to further verify the effectiveness of PI.

Q.3 Ablation on R_1 and R_2

We conducted ablation experiments to further verify the important roles of R_1 and R_2, as shown in Table 14.

R Replace MoPE with alternative designs

Here we provide the complete results of alternative designs for TFPS.

Table 13: Ablation study of PI components. The model variants in our ablation study include the following configurations across both time and frequency branches: (a) inclusion of the Time PI; (b) inclusion of the Frequency PI; (c) exclusion of both. The best results are in **bold**.

Time PI	Frequency PI	ETTh1				ETTh2			
		96	192	336	720	96	192	336	720
✓	✓	**0.398**	**0.423**	**0.484**	**0.488**	**0.313**	**0.405**	**0.392**	**0.410**
✓	✗	0.404	0.454	0.490	0.503	0.322	0.413	0.410	0.425
✗	✓	0.405	0.456	0.493	0.509	0.324	0.415	0.412	0.430
✗	✗	0.407	0.458	0.497	0.513	0.328	0.418	0.419	0.435

Table 14: Ablation study of Loss Constraint. The model variants in our ablation study include the following configurations across both time and frequency branches: (a) inclusion of the R_1; (b) inclusion of the R_2; (c) exclusion of both. The best results are in **bold**.

R_1	R_2	ETTh1				ETTh2			
		96	192	336	720	96	192	336	720
✓	✓	**0.398**	**0.423**	**0.484**	**0.488**	**0.313**	**0.405**	**0.392**	**0.410**
✓	✗	0.408	0.449	0.500	0.498	0.320	0.418	0.415	0.429
✗	✓	0.403	0.434	0.493	0.491	0.316	0.413	0.405	0.418
✗	✗	0.412	0.456	0.509	0.503	0.328	0.425	0.420	0.435

Table 15: Multi-output predictor and a stacked attention layer are used to replace MoPE in ETTh1 and ETTh2 datasets.

	ETTh1				ETTh2			
	96	192	336	720	96	192	336	720
TFPS	**0.398**	**0.423**	**0.484**	**0.488**	**0.313**	**0.405**	**0.392**	**0.410**
Multi-output Predictor	0.403	0.435	0.492	0.491	0.317	0.407	0.399	0.425
Attention Layers	0.399	0.452	0.492	0.508	0.334	0.407	0.409	0.451

As show in Table 15, we have conducted addition experiments where we replaced the MoPE module with weighted multi-output predictor and stacked self-attention layers, keeping all other components and configurations identical. The results demonstrate that our proposed method significantly outperforms them, which validates the importance of the Top-K selection and pattern-aware design in enhancing the model's representation capacity. In contrast, multi-output predictor and self-attention typically treats all data points uniformly, which may limit its ability to capture subtle distribution shifts or evolving patterns across patches.