A How to deal with Non-Divisible Cases

In this section, we further elaborate on how to deal with the cases where the number of features D is not divisible by the number of groups N_G (or, the size of each group S_G). During training, we randomly drop some features to ensure divisibility. On the other hand, in the inference stage, we simply repeat some randomly chosen features and augment them to the original input time series, in order to make the total number of features divisible by N_G. After finishing the forecasting procedure with the augmented inputs, we drop augmented features from outputs. The details are delineated in Algorithm 1.

Algorithm 1: How to deal with the non-divisible case in ESSformer

Input: # of features D, # of groups N_G, # of ensembling N_E, Past observations $\mathbf{X} = \{\mathbf{X}_d\}_{d=1}^{D}$
$N_E = N_E$ **if** $is\_inference$ **then else** 1;
$\mathbf{F} = \{0, 1, ..., D-1\}$; $S_G = \lceil \frac{D}{N_G} \rceil$; $R = D \ \% \ S_G$;
for $i \leftarrow 1$ **to** N_E **do**
  if $R \neq 0$ **then**
    Randomly split $\mathbf{F}$ into $\mathbf{F}^+, \mathbf{F}^-$, where $|\mathbf{F}^-| = R, |\mathbf{F}^+| = D - R, \mathbf{F}^+ \cap \mathbf{F}^- = \phi$;
    $\mathbf{F} = \mathbf{F}^+$;
    if $is\_inference$ **then**
      $\mathbf{F}^{++} = \{f_i | f_i$is a random sample from $\mathbf{F}^+$ without replacement, $i = [0, S_G - R) \cap \mathbb{Z}\}$;
      $\mathbf{X}^{++} = \{\mathbf{X}_b\}_{b \in \mathbf{F}^{++}}$;
      Augment $\mathbf{X}^{++}$ to $\mathbf{X}$ as the (D)-th, $(D+1)$-th, ..., $(D + S_G - R - 1)$-th features;
    else
      Remove $\mathbf{F}^-$ features in $\mathbf{X}$;
    $\mathbf{G} = \{\mathcal{G}_g\}_{g=0}^{N_G - 2} =$ `Random_Partition`$(\mathbf{F})$;
  else
    $\mathbf{G} = \{\mathcal{G}_g\}_{g=0}^{N_G - 1} =$ `Random_Partition`$(\mathbf{F})$;
  if $R \neq 0$ *and* $is\_inference$ **then**
    $\mathbf{G} = \mathbf{G} \cup \{\mathbf{F}^- \cup \{D, D+1, ..., D + N_G - R - 1\}\}$;
    $\mathbf{Y}^i$ = **ESSformer**$(\mathbf{X}, \mathbf{G})$;
    Remove the (D)-th, $(D+1)$-th, ..., $(D + S_G - R - 1)$-th features from $\mathbf{Y}^i$;
  else
    $\mathbf{Y}^i$ = **ESSformer**$(\mathbf{X}, \mathbf{G})$;
$\mathbf{Y} = (\mathbf{Y}^1 + \mathbf{Y}^2 + ... + \mathbf{Y}^{N_E})/N_E$;
return Predicted future observations $\mathbf{Y}$;

B Code Implementation of PeriA and R-PartA in Python

This section provides how to implement the single-head case of PeriA and R-PartA with PyTorch in Python. With **rearrange** function from **einops** package, they are easily implemented. The implementation of PeriA and R-PartA are as follows:

```
from einpos import rearrange
def PeriA_RPartA(input_x, P, N_G, Linear_q_PeriA, Linear_q_RPartA,
                 Linear_k, Linear_v, FA):
    """
    input_x: segmented input tensor (size: batch_size * D * N_S * d_h)
    P: length of one period
    N_G: the number of groups
    MHSA: nn.Module for vanilla multi-head self-attention
    """

    # intra-period attention in PeriA
    x_PeriA_1 = rearrange(input_x, "b c (n p) d -> b c n p d", p = P)
    value     = MHSA(x_PeriA_1, x_PeriA_1, x_PeriA_1)

    # inter-period attention in PeriA
    x_PeriA_2 = rearrange(input_x, "b c (n p) d -> b c p n d", p = P)
```

```
    value     = rearrange(value,   "b c n p d -> b c p n d")
    value     = MHSA(x_PeriA_2, x_PeriA_2, value)
    value     = rearrange(value,   "b c p n d -> b c (n p) d")

    # R-PartA
    x_RPertA  = rearrange(input_x, "b (n g) s d -> b n s g d", n = N_G)
    value     = rearrange(value,   "b (n g) s d -> b n s g d", n = N_G)
    value     = MHSA(x_RPertA, x_RPertA, value)
    value     = rearrange(value,   "b n s g d -> b (n g) s d")
    return value # size : (batch_size * D * N_S * d_h)
```

C Details of Experimental Environments

C.1 Datasets

We evaluate ESSformer on 7 M-LTSF benchmark datasets. The normalization and train/val/test splits are also the same as that of the baseline segment-based Transformers (Zhang & Yan, 2023; Nie et al., 2023). The information of each dataset is as follows:

- **(1-2) ETTh1,2**[7] (Electricity Transformer Temperature-hourly): They have 7 indicators in the electric power long-term deployment, such as oil temperature and 6 power load features. This data is collected for 2 years and the granularity is 1 hour. Different numbers denote different counties in China. Train/val/test is 12/4/4 months and the number of time steps is 17,420.
- **(3-4) ETTm1,2** (Electricity Transformer Temperature-minutely): This dataset is exactly the same with **ETTh1,2**, except for granularity. The granularity of these cases is 15 minutes. The number of time steps is 69,680.
- **(5) Weather**[8]: It has 21 indicators of weather including temperature, humidity, precipitation, and air pressure. It was recorded for 2020, and the granularity is 10 minutes. The ratio of train/val/test is 0.7/0.1/0.2 and the number of time steps is 52,696.
- **(6) Electricity**[9]: In this dataset, information about hourly energy consumption from 2012 to 2014 is collected. Each feature means the electricity consumption of one client, and there are 321 clients in total. The ratio of train/val/test is 0.7/0.1/0.2 and the number of time steps is 26,304.
- **(7) Traffic**[10]: Traffic dataset pertains to road occupancy rates. It encompasses hourly data collected by 862 sensors deployed on San Francisco freeways during the period spanning from 2015 to 2016. The ratio of train/val/test is 0.7/0.1/0.2 and the number of time steps is 17,544.

C.2 Software and Hardware Environments

We conduct experiments on this software and hardware environments for M-LTSF: Python 3.7.12, PyTorch 2.0.1, and NVIDIA GeForce RTX 3090.

C.3 Baselines

We select 11 baselines considering diversity and their forecasting performance. For segment-based Transformers, we include two recent works, Crossformer and PatchTST, as baselines. As for observation-based Transformers, there are many candidates (Zhou et al., 2021; Liu et al., 2022b; Lim et al., 2020; Wu et al., 2022; Zhou et al., 2022; Li et al., 2020). Among them, our choices are FEDformer, Pyraformer, and Informer, considering their performance and meaning in M-LTSF tasks. Furthermore, we add a linear-based model as our baseline. This is because existing Transformers are proven to underperform a single linear in Zeng et al. (2022), so comparing Transformers

[7] https://github.com/zhouhaoyi/ETDataset
[8] https://www.bgc-jena.mpg.de/wetter/
[9] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[10] http://pems.dot.ca.gov

to linear-based models is of importance. We use TSMixer, NLinear, and NLinear-m as our baselines. NLinear-m is modified NLinear for multivariate cases, of which the weight matrix $W \in \mathbb{R}^{T \times \tau}$ is changed into $W \in \mathbb{R}^{T \cdot D \times \tau \cdot D}$. Because NLinear doesn't consider inter-feature dependencies, we add NLinear-m for a fair comparison to ESSformer which is multivariate. To make baselines more diverse, there are other types of baselines, methods based on convolution neural networks (CNNs) and implicit neural representations (INRs). We use recent two CNN-based methods, MICN and TimesNet, and one INR-based method, DeepTime.

- **(1) Crossformer** (Zhang & Yan, 2023): It uses segmentation with separate tokenization where different features are allocated to different segments, and two types of attention, one of which is self-attention for temporal dependencies and the other is for inter-feature relationships. It reduces the complexity of self-attention for inter-feature relationships using routers with low-rank approximation concepts.
- **(2) PatchTST** (Nie et al., 2023): It use similar segmentation to **Crossformer**. A distinct difference from it is that it doesn't consider any relationship between different features.
- **(3) FEDformer** (Zhou et al., 2022): Using the sparsity in frequency domains, it tries to reduce the quadratic complexity of self-attention layers to a linear one.
- **(4) Pyraformer** (Liu et al., 2022b): It has hierarchical structures with different resolutions, leading to linear complexity of self-attention.
- **(5) Informer** (Zhou et al., 2021): By estimating KL divergence between query-key distribution and uniform distribution, it discerns useful and useless information. By using only useful information, it achieves log-linear complexity. Also, a new type of decoder was proposed, which generates forecasting outputs at once.
- **(6) TSMixer** (Chen et al., 2023a): Using the concept of MLP-Mixer in vision domains (Tolstikhin et al., 2021), it was devised to explore the abilities of linear layers in M-LTSF.
- **(7) NLinear** (Zeng et al., 2022): A single linear layer mapping past observations into future observations with a normalization trick that subtracts the last value of input observations from input and adds the value to the output.
- **(8) NLinear-m**: a multivariate version of **NLinear**
- **(9) MICN** (Wang et al., 2023): To capture both local and global patterns from time series efficiently, it extracts patterns with down-sampled convolution and isometric convolution. Also, multi-scale structures are used to capture more diverse patterns.
- **(10) TimesNet** (Wu et al., 2023): Building upon the multi-periodicity of time series, it regards time series as not 1d but 2d structures and aims to figure out intra-period and inter-period relationships.
- **(11) DeepTime** (Woo et al., 2023): It solves the problem where INRs are hard to be generalized in time-series forecasting tasks, with a meta-optimization framework.

Furthermore, we find concurrent works which are Transformer-based methods for time-series forecasting and include them as our baselines. Among (Chen et al., 2023b; Zhao et al., 2023; Xue et al., 2023; Gao et al., 2023; Zhang et al., 2023; Shao et al., 2023; Yu et al., 2023; Lin et al., 2023a), we select JTFT, GCformer, CARD, Client, PETformer as our baselines because they have the same experimental settings with ours[11] or their executable codes are available to run models in our settings.

- **(1) JTFT** (Chen et al., 2023b): Similar to **Crossformer**, segmentation and two types of Transformers are employed. Before a Transformer takes input, it pre-processes input time series. It only encodes a fixed length of recent observations into tokens and sparse frequency information extracted from the whole input into tokens, rather than encodes the whole input directly. This leads to efficient self-attention for temporal dependencies. Also, with a low-rank approximation scheme, it reduces the complexity of self-attention for inter-feature dependencies.

[11] We decide that it has the same experimental setting with ours when the scores of some baselines are the same.

- **(2) GCformer** (Zhao et al., 2023): To overcome the limitations of Transformers that they cannot deal with long time series well, it combines a convolutional branch for global information and Transformer-based branch for local, recent information.
- **(3) CARD** (Xue et al., 2023): With a dual Transformer, it can capture various dependencies across temporal, feature, and hidden dimensions. On top of that, the author devised a robust loss function to relieve overfitting issues in M-LTSF.
- **(4) Client** (Gao et al., 2023): This method has two parts, one of which is a linear model to capture temporal trends and the other is self-attention for inter-feature dependencies.
- **(5) PETformer** (Lin et al., 2023a): Based on **Crossformer** architecture, it introduced placeholder enhancement technique (PET). Thanks to PET, PETformer can forecast with only encoders (i.e., without decoder).

As for evaluation metrics of baseline methods, we repeat the scores when the scores of the same experimental settings as ours are available. Otherwise, we measure evaluation scores with their official codes and best hyperparameters in our experimental environments. The scores of PatchTST, FEDformer, Pyraformer, and Informer are from Nie et al. (2023), and those of TSMixer and NLinear are from Chen et al. (2023a). For Crossformer[12], Linear-m, MICN, TimesNet, Client[13], and DeepTime[14], we measure new scores in the same experimental environments with ours, such as the range of T. When training Crossformer, we convert a Transformer-based encoder into a linear-based encoder for fair comparison to ESSformer, because the latter usually has better performance than the former. For JTFT, GCformer, CARD, and PETformer out of the concurrent works, we repeat the score reported in each paper while we re-evaluate the scores in the case of Client.

C.4 HyperParameters

The details of hyperparameters used in the ESSformer are delineated in this section. We use 7 hyperparameters. The first hyperparameter is the length of input time steps T. We regard it as hyperparameters which is common in recent literature for M-LTSF (Liu et al., 2022b; Zhang & Yan, 2023). The range of T is {512, 1024} similarly to Zhang & Yan (2023). Also, the number of segments N_S is in {8,16,32,64} and the dropout ratio r_{dropout} is in {0.1, 0.2, 0.3, 0.4, 0.7}. The hidden dimension d_h is in {32,64,128,256,512}. The number of heads in self-attention n_h is in {2,4,8,16} and the number of layers L is in {1,2,3}. d_{ff} is the hidden size of the last MLP of each ESSformer layer in equation 4 and in {32,64,128,256,512}. As for the size of each group S_G in random partitioning, we use 3, 7, 30, and 20 for ETT, Weather, Electricity, and Traffic datasets, respectively. Also, batch size is 128, 128, 16, and 12 for ETT, Weather, Electricity, and Traffic datasets, respectively. Finally, we set the learning rate and training epochs to 10^{-3} and 100, respectively. The selected best hyperparameters of ESSformer are in Table 5.

D Additional Complexity Analysis

D.1 Elaboration on Theoretical Analysis of ESSformer's Costs

This section provides how to derive the cost of two efficient attention modules, $\mathcal{O}(N_S^{1.5})$ of PeriA and $\mathcal{O}(DS_G)$ of R-PartA. Because our two attention modules can be formulated by block-diagonal self-attention, we first derive the computation cost of block-diagonal self-attention. Let's assume that B is the size of each block and N is the total number of tokens. Then, the attention cost of each block is $\mathcal{O}(B^2)$ and the number of blocks is N/B. Therefore, the computation cost of block-diagonal self-attention is $\mathcal{O}(NB)$.

PeriA can be formulated by two block-diagonal self-attention modules. Given a period of P and the number of segments (tokens) N_S, the block sizes of two modules are $B_1 = P$ and $B_2 = N_S/P$. As such, the cost of PeriA can be written as $\mathcal{O}(N_S P + N_S^2/P)$. If we set P to $\sqrt{N_S}$ for efficiency, the final cost becomes $\mathcal{O}(N_S^{1.5})$.

[12] https://github.com/Thinklab-SJTU/Crossformer

[13] For MICN, TimesNet, and Client, we use the same code from https://github.com/daxin007/Client/tree/main.

[14] https://github.com/salesforce/DeepTime

Table 5: Selected hyperparameters of ESSformer.

Data	τ	T	N_S	r_{dropout}	d_h	n_h	L	d_{ff}
ETTh1	96	512	64	0.7	128	4	1	256
	192	512	64	0.7	32	4	1	256
	336	512	64	0.7	64	8	1	64
	336	512	64	0.7	64	8	1	64
ETTh2	96	512	64	0.7	512	4	1	256
	192	1024	64	0.7	512	2	1	256
	336	1024	64	0.7	64	16	1	256
	720	512	64	0.7	64	16	1	128
ETTm1	96	512	64	0.2	256	2	2	256
	192	512	64	0.1	64	8	1	128
	336	512	64	0.2	64	2	2	64
	720	1024	64	0.7	64	4	1	128
ETTm2	96	1024	64	0.7	512	2	1	64
	192	1024	64	0.7	128	4	1	32
	336	1024	64	0.4	128	2	1	32
	720	1024	64	0.7	256	4	1	32
Weather	96	512	64	0.2	128	8	3	256
	192	512	64	0.2	128	16	3	256
	336	512	64	0.4	128	16	3	512
	720	512	64	0.4	128	2	1	256
Electricity	96	512	64	0.3	256	8	1	256
	192	512	64	0.2	256	4	2	256
	336	512	64	0.2	128	4	3	256
	720	512	64	0.2	256	4	3	256
Traffic	96	512	8	0.2	512	2	3	512
	192	512	8	0.1	256	4	3	512
	336	512	8	0.2	256	2	3	256
	720	512	8	0.2	512	4	3	512

Table 6: Complexity Comparison of ESSformer against all baselines.

Method	Theoretical Complexity	Avg. MSE	Avg. Rank
ESSformer	$\mathcal{O}(D\left(\frac{T}{S}\right)^{1.5})$	0.292	1.04
Crossformer	$\mathcal{O}(D\left(\frac{T}{S}\right)^{2})$	0.650	7.21
PatchTST	$\mathcal{O}(D\left(\frac{T}{S}\right)^{2})$	0.304	2.86
FEDformer	$\mathcal{O}(T)$	0.373	6.93
Pyraformer	$\mathcal{O}(T)$	0.888	10.29
Informer	$\mathcal{O}(T \log T)$	1.170	10.43
TSMixer	$\mathcal{O}(DT^2 + D^2T)$	0.310	2.79
NLinear	$\mathcal{O}(DT)$	0.319	4.61
NLinear-m	$\mathcal{O}(D^2T)$	N/A	N/A
MICN	$\mathcal{O}(D^2T)$	0.486	8.00
TimesNet	$\mathcal{O}(DT)$	0.387	7.25
DeepTime	$\mathcal{O}(DT)$	0.328	4.36

As for R-PartA, it can be formulated by one block-diagonal self-attention module. Given a group size of S_G and the number of entire features D, the block size is $B = S_G$ and the final cost of R-PartA is $\mathcal{O}(DS_G)$.

D.2 Complexity comparison between ESSformer and all baselines in Table 1

We provide theoretical analyses of all baselines in Table 1 about computational costs with an average of performance scores (MSE) and rank across all datasets. T is the input historical time length and D is the number of features. Also, $S = \frac{T}{N_S}$ is the size of each segment in segment-based Transformer, where N_S is the number of segments. Table 6 shows that our ESSformer gives the best forecasting performance with a quite low cost.

E Effect of Changing the Length of One Period Every Layer

In Section 3.1, we make P change per each layer. To explore the effectiveness of this design, we fix P to P_* every layer and compare the scores without changing P to the original one with changing

Table 7: Test MSE and MAE comparison between ESSformer with and without changing P per layer. 'N/A' denotes this experiment cannot be applicable to the setting because the number of layers $N_L = 1$.

Score	P of multiple layers	Electricity				Traffic			
		$\tau = 96$	192	336	720	96	192	336	720
MSE	$[\frac{1}{2}P_*, P_*, 2P_*]$	N/A	**0.142**	**0.154**	**0.176**	**0.345**	**0.370**	**0.385**	**0.426**
	$[P_*, P_*, P_*]$	N/A	0.144	0.156	0.179	0.346	0.372	0.387	**0.426**
MAE	$[\frac{1}{2}P_*, P_*, 2P_*]$	N/A	**0.240**	**0.256**	**0.278**	**0.245**	**0.255**	**0.265**	**0.287**
	$[P_*, P_*, P_*]$	N/A	0.242	**0.256**	0.280	0.246	0.256	0.266	**0.287**

Table 8: Test MSE of ESSformer compared to other concurrent models for M-LTSF.

Data		Ours ESSformer	Concurrent Model PITS	Concurrent Model ModernTCN	Concurrent Model FITS	Existing Baseline PatchTST	Existing Baseline TSMixer
ETTh1	96	**0.361**	0.364	0.368	0.368	0.370	**0.361**
	192	**0.396**	0.398	0.405	0.404	0.413	0.404
	336	0.400	0.415	**0.391**	0.405	0.422	0.420
	720	**0.412**	0.425	0.450	0.425	0.447	0.463
ETTh2	96	0.269	0.269	0.263	**0.255**	0.274	0.274
	192	0.323	0.326	0.320	**0.307**	0.341	0.339
	336	0.317	0.354	0.313	**0.306**	0.329	0.361
	720	0.370	0.378	0.392	**0.368**	0.379	0.445
ETTm1	96	**0.282**	0.296	0.292	0.305	0.293	0.285
	192	0.325	**0.321**	0.332	0.339	0.333	0.327
	336	**0.352**	0.353	0.365	0.366	0.369	0.356
	720	0.401	**0.395**	0.416	0.414	0.416	0.419
ETTm2	96	**0.160**	0.163	0.166	0.164	0.166	0.163
	192	**0.213**	**0.213**	0.222	0.217	0.223	0.216
	336	**0.262**	0.263	0.272	0.269	0.274	0.268
	720	**0.336**	0.337	0.351	0.347	0.361	0.420
Weather	96	**0.142**	0.149	0.149	0.145	0.149	0.145
	192	**0.185**	0.195	0.196	0.188	0.194	0.191
	336	**0.235**	0.244	0.238	0.236	0.245	0.242
	720	**0.305**	0.312	0.314	0.308	0.314	0.320
Electricity	96	**0.125**	0.129	0.129	0.135	0.129	0.131
	192	**0.142**	0.144	0.143	**0.142**	0.147	0.151
	336	**0.154**	0.160	0.161	0.163	0.163	0.161
	720	**0.176**	0.197	0.191	0.200	0.197	0.197
Traffic	96	**0.345**	0.373	0.368	0.381	0.360	0.376
	192	**0.370**	0.388	0.379	0.381	0.379	0.379
	336	**0.385**	0.401	0.397	0.404	0.392	0.413
	720	**0.426**	0.436	0.440	0.446	0.432	0.444
Avg. Rank		**1.357**	3.071	3.536	3.500	4.393	4.250

P, in Table 7. In almost all cases, ESSformer with changing P achieves better performance. In spite of the small margin, consistent improvement by changing P proves the efficacy of our design.

F Comparison to Other Concurrent Works in Main Forecasting Experiments

In addition to the concurrent models in Table 2, We found other concurrent models which have similar experimental settings to ours (Anonymous, 2023c;b;a). Accordingly, we provide Table 8 for comparison. Our ESSformer still outperforms the very recent works by a large margin.

G The Performance of ESSformer without The Test-Time Ensemble Method

In Table 9, we conduct the main M-LTSF task including ESSformer without the test-time ensemble method. By setting N_E to 1, we obtain ESSformer without the test-time ensemble method. In this experiment, we include some baselines showing decent forecasting performance. As Table 9 shows, although the test-time ensemble method is removed, ESSformer without the test-time ensemble method still gives better results than baselines.

Table 9: MSE scores of main forecasting results including ESSformer without the test-time ensemble method.

Data		ESSformer w/o ensembling	PatchTST	FEDformer	TSMixer	DeepTime
ETTh1	96	**0.361**	0.370	0.376	**0.361**	0.372
	192	**0.393**	0.413	0.423	0.404	0.405
	336	**0.404**	0.422	0.444	0.420	0.437
	720	**0.412**	0.447	0.469	0.463	0.477
ETTh2	96	**0.270**	0.274	0.332	0.274	0.291
	192	**0.321**	0.341	0.407	0.339	0.403
	336	**0.317**	0.329	0.400	0.361	0.466
	720	**0.371**	0.379	0.412	0.445	0.576
ETTm1	96	0.286	0.293	0.326	**0.285**	0.311
	192	0.328	0.333	0.365	**0.327**	0.339
	336	**0.354**	0.369	0.392	0.356	0.366
	720	0.403	0.416	0.446	0.419	**0.400**
ETTm2	96	**0.160**	0.166	0.180	0.163	0.165
	192	**0.213**	0.223	0.252	0.216	0.222
	336	**0.262**	0.274	0.324	0.268	0.278
	720	**0.336**	0.361	0.410	0.420	0.369
Weather	96	**0.142**	0.149	0.238	0.145	0.169
	192	**0.186**	0.194	0.275	0.191	0.211
	336	**0.236**	0.245	0.339	0.242	0.255
	720	**0.305**	0.314	0.389	0.320	0.318
Electricity	96	**0.127**	0.129	0.186	0.131	0.139
	192	**0.145**	0.147	0.197	0.151	0.154
	336	**0.158**	0.163	0.213	0.161	0.169
	720	**0.181**	0.197	0.233	0.197	0.201
Traffic	96	**0.347**	0.360	0.576	0.376	0.401
	192	**0.372**	0.379	0.610	0.397	0.413
	336	**0.387**	0.392	0.608	0.413	0.425
	720	**0.430**	0.432	0.621	0.444	0.462
Avg. Rank		**1.107**	2.679	4.821	2.500	3.786

H MORE ANALYSIS ON S_G AND N_E

Along the way we conduct additional experiments to analyze the behavior of R-PartA, we find three empirical rules of thumb which can give help to selecting S_G and N_E before training.

A large number of distinct partitions tends to perform well. As shown in Figure 4(b), we found that it is better to select the size of each group in R-PartA, S_G, as what makes the available number of distinct partitions N_P large because ESSformer benefits from the diversity of partitioning during training and inference. Note that N_P is formulated by S_G as follows:

$$N_P = \prod_{i=0}^{\lfloor \frac{D}{S_G} \rfloor} \binom{n - iS_G}{S_G} / \lfloor \frac{D}{S_G} \rfloor ! \tag{8}$$

In Figure 9, we further extend this experiment to four datasets (ETTm1, Weather, Electricity, and Traffic) where all datasets differ in feature size and originate from different domains. The experimental results well verify that S_G leading to large N_P tends to perform well.

A large number of instances for ensembling tends to give better results. As shown in in Figure 5 and Figure 8, the larger the number of instances for ensembling (N_E) is, the better ESSformer performs. Therefore, to achieve better forecasting results, large N_E is required.

The effect of N_E tends to be smaller, as S_G increases. We think that because ESSformer with small S_G can capture relationships within very small groups, it might need large N_E to recover the entire relationships from small ones. To prove this statement, we additionally conduct the experiments in Figure 10 where we measure performance gain of increasing N_E in various S_G. This figure shows that the effect of N_E tends to be smaller, as S_G increases, supporting our statement. Therefore, this fact helps to select efficient but effective N_E given S_G.

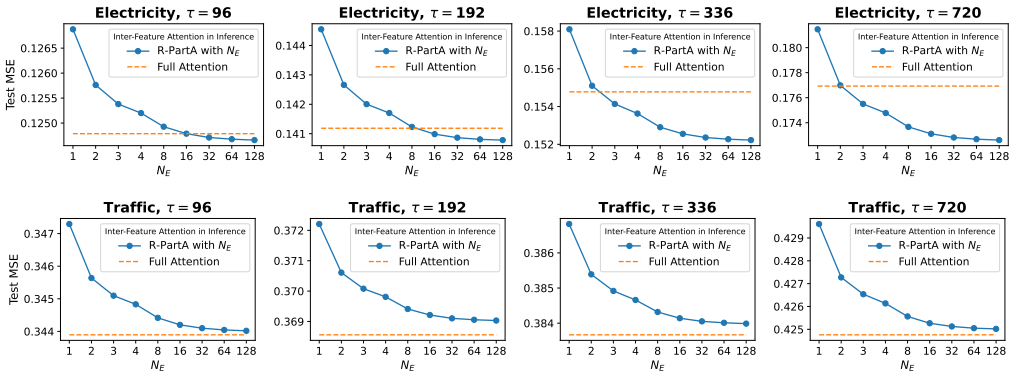

Figure 8: Sensitivity to N_E (additional results for Figure 5)

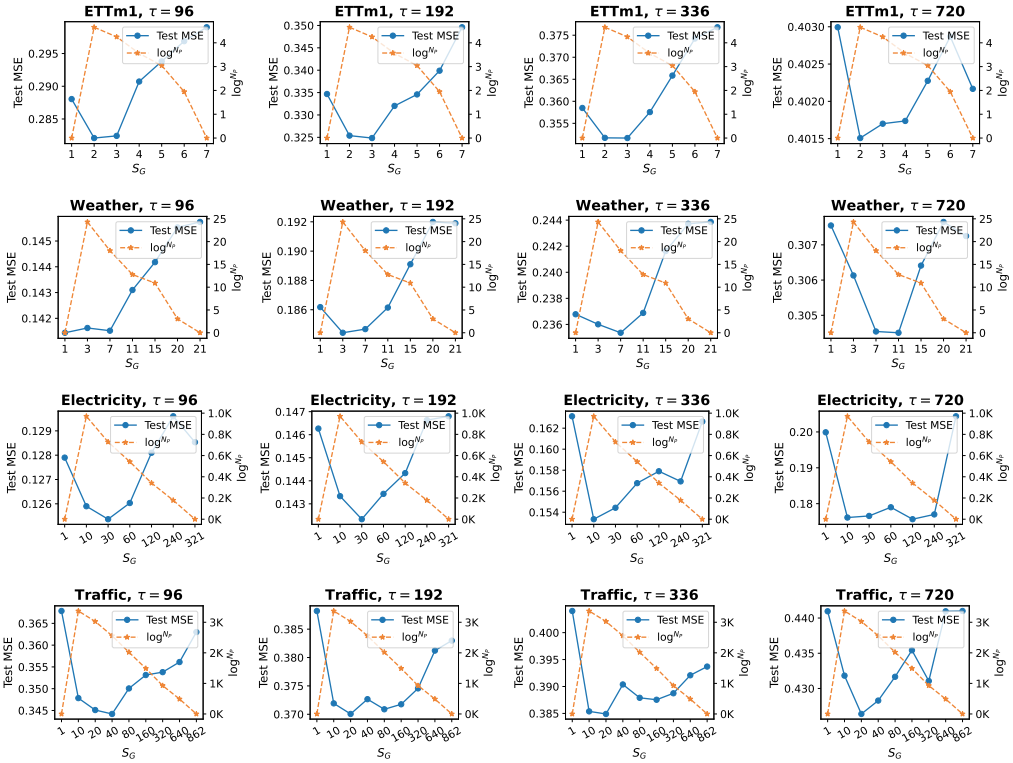

Figure 9: Sensitivity to S_G (additional results for Figure 4(b))

I Additional Experiments

I.1 Additional Experimental Results in Tabular Forms

In this section, we provide additional results for existing experiments, such as experiments with other datasets and MAE evaluation metrics. Table 11 and Table 10 are additional results for Table 1 and Table 3, respectively.

I.2 Additional Visualization

Like Appendix I.1, this section provides additional visualizations with other datasets or models for existing ones. Figure 11 is for Figure 2, Figure 12 and Figure 9 for Figure 4, Figure 8 for Figure 5,

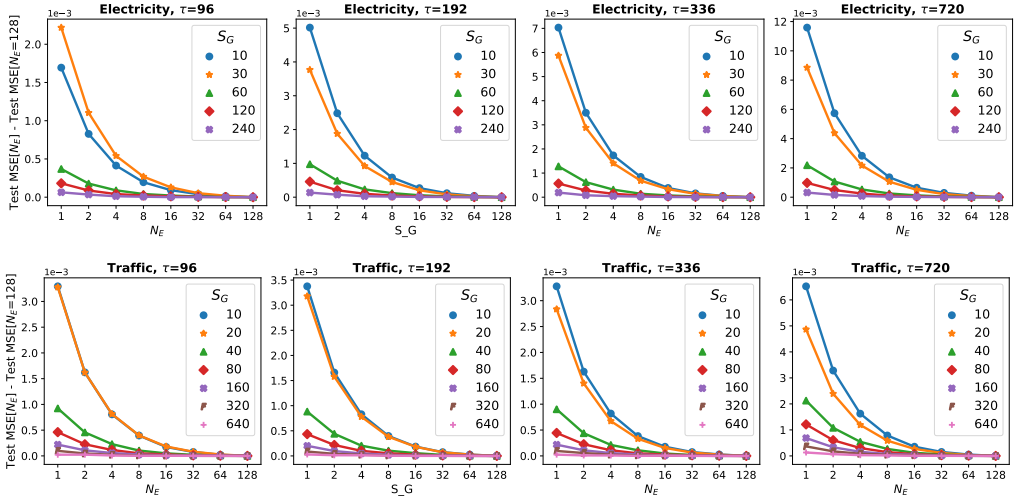

Figure 10: Changes in the effect of N_E when S_G increases. The y axis shows the performance difference between Test MSEs when $N_E \in \{1, 2, 4, 8, 16, 32, 64, 128\}$ and $N_E = 128$.

Table 10: Test MSE and MAE of ablation study for PeriA and R-PartA by substituting vanilla multi-head full self-attention (MHSA) for them. Also, the univariate ESSformer denotes a case where R-PartA is removed, not considering inter-feature dependencies. 'M' denotes multivariate cases and 'U' denotes univariate cases (additional results for Table 3)

Score	ESSformer Variants	PeriA	R-PartA	ETTh1 (7) 96	ETTh1 (7) 192	ETTh1 (7) 336	ETTh1 (7) 720	ETTh2 (7) 96	ETTh2 (7) 192	ETTh2 (7) 336	ETTh2 (7) 720	ETTm1 (7) 96	ETTm1 (7) 192	ETTm1 (7) 336	ETTm1 (7) 720	ETTm2 (7) 96	ETTm2 (7) 192	ETTm2 (7) 336	ETTm2 (7) 720
MSE	Original (M)	PeriA	R-PartA	**0.361**	0.396	**0.400**	**0.412**	**0.269**	**0.323**	**0.317**	**0.370**	**0.282**	**0.325**	**0.352**	**0.401**	**0.160**	**0.213**	**0.262**	**0.336**
	Ablated (M)	MHSA	R-PartA	**0.361**	**0.393**	0.404	0.421	0.273	0.328	0.319	0.373	0.288	0.333	0.359	0.403	0.161	0.214	0.263	0.337
		PeriA	MHSA	**0.361**	0.395	0.401	0.413	**0.269**	0.325	0.318	0.371	0.299	0.350	0.377	0.402	0.161	**0.213**	0.265	0.338
	Ablated (U)	PeriA	N/A	**0.361**	**0.393**	0.404	0.420	0.272	0.325	0.318	0.371	0.288	0.335	0.358	0.403	0.161	**0.213**	0.265	0.338
MAE	Original (M)	PeriA	R-PartA	**0.390**	0.414	0.421	**0.442**	**0.332**	**0.369**	**0.378**	**0.416**	**0.340**	**0.365**	**0.385**	**0.408**	**0.253**	**0.290**	**0.325**	**0.372**
	Ablated (M)	MHSA	R-PartA	**0.390**	**0.410**	**0.419**	0.446	0.334	0.374	0.380	0.418	0.343	0.367	0.387	0.409	**0.253**	0.291	0.326	0.374
		PeriA	MHSA	**0.390**	0.413	0.420	**0.442**	**0.332**	0.371	0.380	**0.416**	0.353	0.382	0.396	**0.408**	**0.253**	**0.290**	0.327	0.374
	Ablated (U)	PeriA	N/A	**0.390**	**0.410**	**0.419**	0.446	0.334	0.373	0.380	0.418	0.344	0.371	0.386	0.409	**0.253**	**0.290**	0.328	0.376

Score	ESSformer Variants	PeriA	R-PartA	Weather (21) 96	Weather (21) 192	Weather (21) 336	Weather (21) 720	Electricity (321) 96	Electricity (321) 192	Electricity (321) 336	Electricity (321) 720	Traffic (862) 96	Traffic (862) 192	Traffic (862) 336	Traffic (862) 720
MSE	Original (M)	PeriA	R-PartA	0.142	**0.185**	**0.235**	**0.305**	**0.125**	**0.142**	**0.154**	**0.176**	**0.345**	**0.370**	**0.385**	**0.426**
	Ablated (M)	MHSA	R-PartA	0.142	**0.185**	0.236	**0.305**	**0.125**	0.144	0.156	0.178	0.348	0.373	0.391	0.430
		PeriA	MHSA	0.146	0.192	0.244	0.307	0.129	0.147	0.163	0.204	0.363	0.383	0.394	0.441
	Ablated (U)	PeriA	N/A	**0.141**	0.186	0.237	0.308	0.128	0.146	0.163	0.204	0.368	0.388	0.404	0.441
MAE	Original (M)	PeriA	R-PartA	**0.193**	0.237	**0.277**	**0.328**	**0.222**	**0.240**	**0.256**	**0.278**	**0.245**	**0.255**	**0.265**	**0.287**
	Ablated (M)	MHSA	R-PartA	0.194	**0.236**	0.280	**0.328**	**0.222**	0.241	**0.256**	0.280	0.246	0.256	0.269	**0.287**
		PeriA	MHSA	0.199	0.243	0.286	0.331	0.228	0.246	0.259	0.297	0.257	0.269	0.276	0.303
	Ablated (U)	PeriA	N/A	0.195	0.239	0.279	0.333	0.223	0.242	0.260	0.297	0.257	0.265	0.277	0.299

and Figure 13 for Figure 7. Furthermore, Figure 14 shows the forecasting results of three segment-based models including ESSformer. Our method captures temporal dynamics better than baselines.

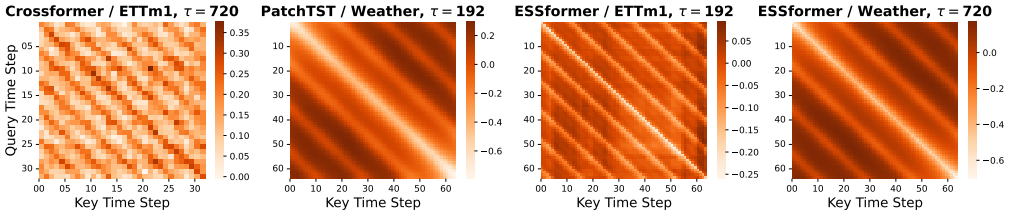

Figure 11: Periodic properties on attention matrices of self-attention layers capturing temporal dependencies in segment-based transformer. For ESSformer cases, we replace PeriA with full self-attention. (additional results for Figure 2)

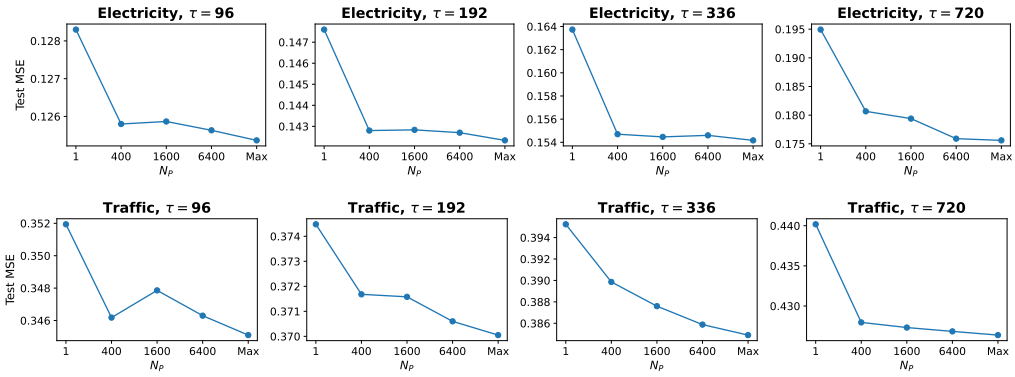

Figure 12: Sensitivity to N_P (additional results for Figure 4(a))

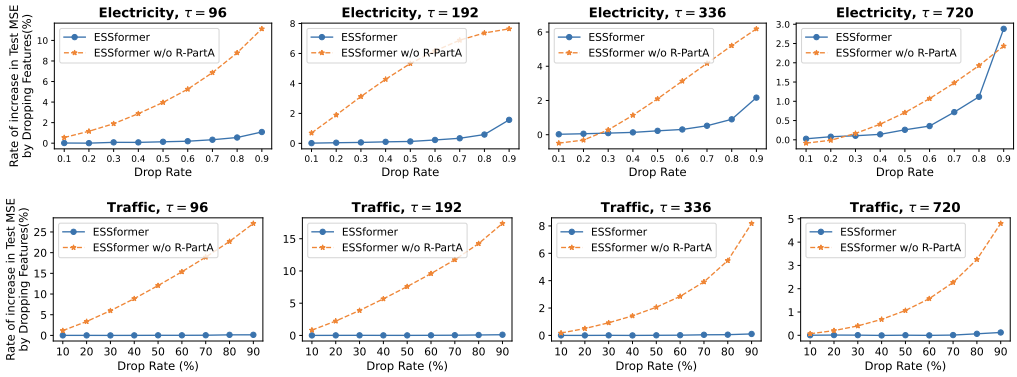

Figure 13: Increasing rate of test MSE by dropping $n\%$ features in ESSformer with or without R-PartA (additional results for Figure 7)

Table 11: MSE and MAE scores of main forecasting results. 'former' included in some model names is abbreviated to 'f.'. Seg.T and Obs.T denote segment-based and obeservation-based transformers, respectively. (addtional results for Table 1)

Scores	Data		Seg.T			Obs.T			Linear			Others		
			ESSf.	Crossf.	PatchTST	FEDf.	Pyraf.	Inf.	TSMixer	NLinear	NLinear-m	MICN	TimesNet	DeepTime
MSE	ETTh1	$\tau = 96$	**0.361**	0.427	0.370	0.376	0.664	0.941	**0.361**	0.374	0.463	0.828	0.465	0.372
		192	**0.396**	0.537	0.413	0.423	0.790	1.007	0.404	0.408	0.535	0.765	0.493	0.405
		336	**0.400**	0.651	0.422	0.444	0.891	1.038	0.420	0.429	0.531	0.904	0.456	0.437
		720	**0.412**	0.664	0.447	0.469	0.963	1.144	0.463	0.440	0.558	1.192	0.533	0.477
	ETTh2	96	**0.269**	0.720	0.274	0.332	0.645	1.549	0.274	0.277	0.347	0.452	0.381	0.291
		192	**0.323**	1.121	0.341	0.407	0.788	3.792	0.339	0.344	0.425	0.554	0.416	0.403
		336	**0.317**	1.524	0.329	0.400	0.907	4.215	0.361	0.357	0.414	0.582	0.363	0.466
		720	**0.370**	3.106	0.379	0.412	0.963	3.656	0.445	0.394	0.460	0.869	0.371	0.576
	ETTm1	96	**0.282**	0.336	0.293	0.326	0.543	0.626	0.285	0.306	0.322	0.406	0.343	0.311
		192	**0.325**	0.387	0.333	0.365	0.557	0.725	0.327	0.349	0.365	0.500	0.381	0.339
		336	**0.352**	0.431	0.369	0.392	0.754	1.005	0.356	0.375	0.392	0.580	0.436	0.366
		720	0.401	0.555	0.416	0.446	0.908	1.133	0.419	0.433	0.445	0.607	0.527	**0.400**
	ETTm2	96	**0.160**	0.338	0.166	0.180	0.435	0.355	0.163	0.167	0.191	0.238	0.218	0.165
		192	**0.213**	0.567	0.223	0.252	0.730	0.595	0.216	0.221	0.260	0.302	0.282	0.222
		336	**0.262**	1.050	0.274	0.324	1.201	1.270	0.268	0.274	0.330	0.447	0.378	0.278
		720	**0.336**	2.049	0.361	0.410	3.625	3.001	0.420	0.368	0.416	0.549	0.444	0.369
	Weather	96	**0.142**	0.150	0.149	0.238	0.896	0.354	0.145	0.182	0.162	0.188	0.179	0.169
		192	**0.185**	0.200	0.194	0.275	0.622	0.419	0.191	0.225	0.213	0.231	0.230	0.211
		336	**0.235**	0.263	0.245	0.339	0.739	0.583	0.242	0.271	0.267	0.280	0.276	0.255
		720	**0.305**	0.310	0.314	0.389	1.004	0.916	0.320	0.338	0.343	0.358	0.347	0.318
	Electricity	96	**0.125**	0.135	0.129	0.186	0.386	0.304	0.131	0.141	OOM	0.177	0.186	0.139
		192	**0.142**	0.158	0.147	0.197	0.386	0.327	0.151	0.154	OOM	0.195	0.208	0.154
		336	**0.154**	0.177	0.163	0.213	0.378	0.333	0.161	0.171	OOM	0.213	0.210	0.169
		720	**0.176**	0.222	0.197	0.233	0.376	0.351	0.197	0.210	OOM	0.204	0.231	0.201
	Traffic	96	**0.345**	0.481	0.360	0.576	2.085	0.733	0.376	0.410	OOM	0.489	0.599	0.401
		192	**0.370**	0.509	0.379	0.610	0.867	0.777	0.397	0.423	OOM	0.493	0.612	0.413
		336	**0.385**	0.534	0.392	0.608	0.869	0.776	0.413	0.435	OOM	0.496	0.618	0.425
		720	**0.426**	0.585	0.432	0.621	0.881	0.827	0.444	0.464	OOM	0.520	0.654	0.462
	Avg. Rank		**1.036**	7.214	2.857	6.929	10.286	10.429	2.786	4.607	N/A	8.000	7.25	4.357
MAE	ETTh1	96	**0.390**	0.448	0.400	0.415	0.612	0.769	0.392	0.394	0.486	0.607	0.466	0.398
		192	**0.414**	0.520	0.429	0.446	0.681	0.786	0.418	0.415	0.530	0.575	0.479	0.419
		336	**0.421**	0.588	0.440	0.462	0.738	0.784	0.431	0.427	0.533	0.621	0.473	0.442
		720	**0.442**	0.612	0.468	0.492	0.782	0.857	0.472	0.453	0.552	0.736	0.525	0.493
	ETTh2	96	**0.332**	0.615	0.337	0.374	0.597	0.952	0.341	0.338	0.418	0.462	0.423	0.350
		192	**0.369**	0.785	0.382	0.446	0.683	1.542	0.385	0.381	0.470	0.528	0.445	0.427
		336	**0.378**	0.980	0.384	0.447	0.747	1.642	0.406	0.400	0.473	0.556	0.422	0.475
		720	**0.416**	1.487	0.422	0.469	0.783	1.619	0.470	0.436	0.499	0.667	0.424	0.545
	ETTm1	96	0.340	0.387	0.346	0.390	0.510	0.560	**0.339**	0.348	0.375	0.434	0.381	0.353
		192	**0.365**	0.419	0.370	0.415	0.537	0.619	**0.365**	0.375	0.399	0.500	0.403	0.369
		336	0.385	0.449	0.392	0.425	0.655	0.741	**0.382**	0.388	0.414	0.549	0.438	0.391
		720	**0.408**	0.532	0.420	0.458	0.724	0.845	0.414	0.422	0.447	0.560	0.488	0.414
	ETTm2	96	0.253	0.393	0.256	0.271	0.507	0.462	**0.252**	0.255	0.293	0.331	0.307	0.259
		192	**0.290**	0.519	0.296	0.318	0.673	0.586	**0.290**	0.293	0.345	0.374	0.352	0.299
		336	0.325	0.732	0.329	0.364	0.845	0.871	**0.324**	0.327	0.390	0.478	0.407	0.338
		720	**0.372**	1.170	0.394	0.420	1.451	1.267	0.422	0.384	0.442	0.554	0.450	0.400
	Weather	96	**0.193**	0.224	0.198	0.314	0.556	0.405	0.198	0.232	0.239	0.258	0.237	0.227
		192	**0.237**	0.267	0.241	0.329	0.624	0.434	0.242	0.269	0.285	0.295	0.279	0.266
		336	**0.277**	0.328	0.282	0.377	0.753	0.543	0.280	0.301	0.328	0.337	0.310	0.304
		720	**0.328**	0.363	0.334	0.409	0.934	0.705	0.336	0.348	0.380	0.399	0.353	0.357
	Electricity	96	**0.222**	0.234	**0.222**	0.302	0.449	0.393	0.229	0.237	OOM	0.294	0.290	0.239
		192	**0.240**	0.262	**0.240**	0.311	0.443	0.417	0.246	0.248	OOM	0.306	0.301	0.253
		336	**0.256**	0.283	0.259	0.328	0.443	0.422	0.261	0.265	OOM	0.324	0.314	0.270
		720	**0.278**	0.328	0.290	0.344	0.445	0.427	0.293	0.297	OOM	0.317	0.329	0.300
	Traffic	96	**0.245**	0.265	0.249	0.359	0.468	0.410	0.264	0.279	OOM	0.317	0.325	0.280
		192	**0.255**	0.277	0.256	0.380	0.467	0.435	0.277	0.284	OOM	0.319	0.332	0.285
		336	0.265	0.291	**0.264**	0.375	0.469	0.434	0.290	0.290	OOM	0.317	0.332	0.292
		720	0.287	0.325	**0.286**	0.375	0.473	0.466	0.306	0.307	OOM	0.326	0.348	0.312
	Avg. Rank		**1.214**	7.214	2.786	7.321	10.393	10.464	2.786	3.679	N/A	8.071	6.786	5.000

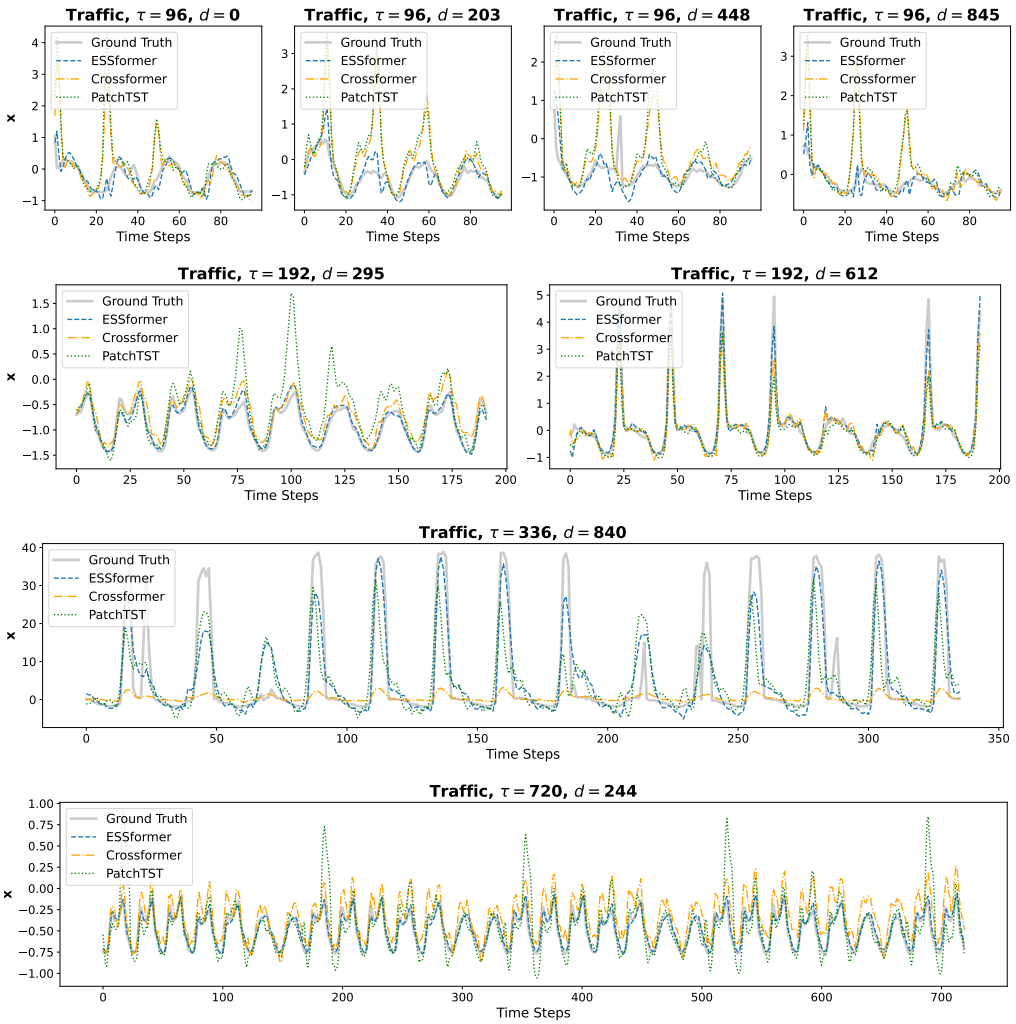

Figure 14: Forecasting results of various segment-based transformers (Crossformer, PatchTST, and ESSformer). Dotted lines and dotted-dashed lines denote baselines, dashed lines denote ESSformer, and solid lines denote ground truth. τ denotes the length of time steps in future outputs and d denotes a feature index.