GBT: Two-stage transformer framework for non-stationary time series forecasting

Li Shen, Yuning Wei *, Yangzhu Wang

Beihang University, RM.807, 8th Dormitory, Dayuncun Residential Quarter, No. 29, Zhichun Road, Beijing 100191, PR China

ABSTRACT

This paper shows that time series forecasting Transformer (TSFT) suffers from severe over-fitting problem caused by improper initialization method of unknown decoder inputs, especially when handling non-stationary time series. Based on this observation, we propose **GBT**, a novel two-stage **T**ransformer framework with **G**ood **B**eginning. It decouples the prediction process of TSFT into two stages, including *Auto-Regression* stage and *Self-Regression* stage to tackle the problem of different statistical properties between input and prediction sequences. Prediction results of *Auto-Regression* stage serve as a 'Good Beginning', i.e., a better initialization for inputs of *Self-Regression* stage. We also propose the **E**rror **S**core **M**odification module to further enhance the forecasting capability of the *Self-Regression* stage in GBT. Extensive experiments on seven benchmark datasets demonstrate that GBT outperforms SOTA TSFTs (FEDformer, Pyraformer, ETSformer, etc.) and many other forecasting models (SCINet, N-HiTS, etc.) with only canonical attention and convolution while owning less time and space complexity. It is also general enough to couple with these models to strengthen their forecasting capability. The source code is available at: https://github.com/OrigamiSL/GBT

© 2023 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

Popular machine learning technology and neural networks (Durbin & Koopman, 2012; He et al., 2022; Lai et al., 2018; Wang et al., 2021; Weerakody et al., 2021) have been applied in time series forecasting of various fields (Ahmad et al., 2019; Qu et al., 2019; Zhao et al., 2021) more and more widely to tackle complicated forecasting situations. *T*ime *S*eries *F*orecasting *T*ransformer (*TSFT*), which benefits from long-term dependency capturing capability of attention mechanism (Vaswani et al., 2017), dominates long-horizon forecasting researches based on neural networks (Li et al., 2019; Liu et al., 2022b; Woo et al., 2022b; Zhou et al., 2021) in the past few years. However, some recent researches (Kim et al., 2022; Liu et al., 2022a) point out that TSFTs are weak in tackling non-stationary time series, which are universal in real-world, and will suffer severe over-fitting problem caused by different statistical dynamics of different local windows. Furthermore, some advanced deep forecasting models based on simple MLP (or Linear) (Challu et al., 2022; Wu et al., 2022; Zeng et al., 2022; Zhang et al., 2022) have reported state-of-the-art forecasting performances which can compete and outperform TSFTs. Therefore, it indicates that there must exist certain deficiencies in TSFT architectures. In this paper, we find

out that the *initialization method of decoder inputs* is responsible for it.

As a significant component of Transformer architecture, decoder plays the role of inferring target outputs (Devlin et al., 2019; Yang et al., 2021), which are normally unknown, via representations provided by encoder. In time series forecasting tasks, target outputs will be prediction sequences. As inputs of decoder in TSFTs locate at the unknown timespan, many TSFTs initialize them with zero vector/tensor (Li et al., 2019; Zhou et al., 2021). This initialization method is established upon the assumption that average values of unknown prediction sequences are zero in statistical manner after Z-score standardization (Woo et al., 2022b; Wu et al., 2021; Zhou et al., 2022b), which commonly serves as preprocessing procedure of time series. Z-score standardization forces global data distribution of time series into a specific distribution with average value of zero and variance of one. Therefore, if the statistics of time series are invariant with time, i.e., time series is stationary, average values of prediction sequences are supposed to be zero, which means that zero-initialization method is appropriate under stationary conditions. However, this assumption is no longer rational under non-stationary time series. Take a widely-used real-world electricity consumption dataset (Woo et al., 2022b; Zhou et al., 2022b, 2021), whose time series are non-stationary, as an example. It can be observed in Fig. 1(a) that the statistics of its sub-sequences keep changing with different timespans though after Z-score standardization so that rarely does sub-sequence
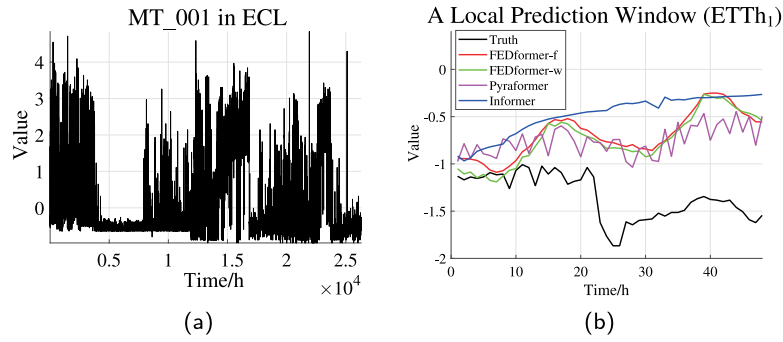
Fig. 1. (a) is the data distribution of 'MT_001' in ECL dataset after Z-score standardization. (b) are prediction results of some TSFTs when handling a local window in ETTh$_1$ dataset. It is obvious that they all suffer severe over-fitting problem.
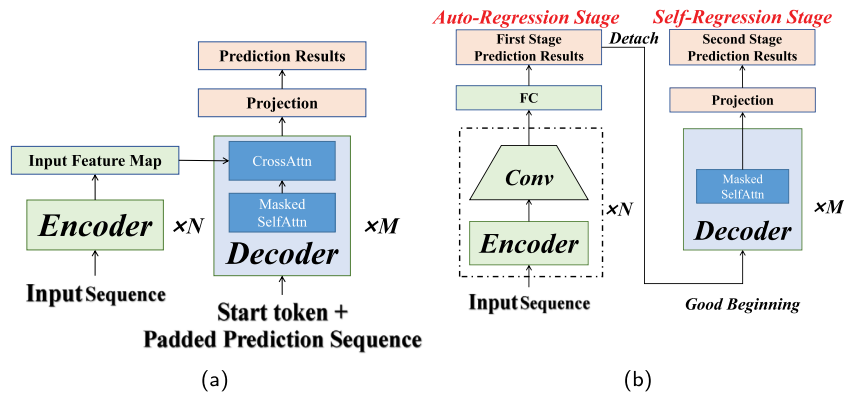
Fig. 2. (a) is the architecture of canonical TSFT with start token and (b) is the architecture of GBT which decouples the forecasting process of TSFT into two stages.

of non-stationary time series own the statistics of 0-mean and 1-variance. Therefore, if we initialize decoder inputs with zero vector/tensor, TSFTs will certainly suffer from the over-fitting problem (Fig. 1(b)), illustrating that zero-initialization method for decoder inputs is not a proper solution in practice.

To address the issue of zero-initialization of decoder inputs, some TSFTs additionally employ **start token** (Li et al., 2019; Zhou et al., 2021) or **decomposed trend terms** of input sequences from encoder (Woo et al., 2022b; Wu et al., 2021; Zhou et al., 2022b) to initialize inputs of decoders. As we have analyzed before, statistical properties of local sub-sequences within non-stationary time series are dynamic so that data distributions of input/output sequence are rarely identical in real-world practice. It means that these two methods are not always effective. Meanwhile, they will bring extra computation cost, which is expensive especially considering the quadratic complexity of attention mechanism in Transformer. We will provide more detailed analysis in later Section 4.

Thus, we propose **GBT**, a novel two-stage **T**ransformer framework with **G**ood **B**eginning. An overview of it is shown in Fig. 2(b) while the canonical architecture of TSFT is shown in Fig. 2(a) for comparison. The idea of GBT is simple: We divide the architecture of TSFT into two stages according to their different functions and outputs of the first stage are treated as inputs of the second stage. The first stage is the *Auto-Regression* stage. Only encoders are employed to capture features of input sequences and a FC layer is employed to obtain prediction sequences. It aims to acquire *better prediction elements initialization* for the second stage through features of input windows. As prediction results of the first stage are trained with the truth of prediction windows, these prediction results will have more similar statistical properties with those of truth compared with former initialization methods. Thus, they are 'Good Beginning' for the second stage input. Then

the second stage is the *Self-Regression* stage. As 'Good Beginning' is supposed to obtain auto-regressive results from input sequences, the second stage only needs to be concerned about relationships within prediction sequences. Therefore, it employs decoders which are only composed of masked self-attention modules and abandon modules including start token and cross-attention to reduce reduplicate computations. Networks of the second stage **do not participate** in the training phase of the first stage and parameters in the first stage will be **fixed** during the training phase of the second stage. It could be seen that compared with conventional TSFT framework, GBT not only alleviates the potential problem of different statistical properties of input/prediction windows when handling non-stationary time series by decoupled two-stage framework and 'Good Beginning' but also owns fewer parameters under the same conditions.

Though two-stage architecture can give inputs of decoder, i.e., the second stage, a better initialization, deviations of first-stage prediction results from ground truths are different in different time stamps (Lai et al., 2018; Taieb & Atiya, 2016). Hence, we employ the **E**rror **S**core **M**odification module (ESM) in the second stage to utilize prediction results of the first stage in a more rational way.

Leveraging from these mechanisms, GBT tackles non-stationary time series and the accompanied over-fitting problem well. Combining with experiments results, our main contributions are summarized as follows:

1. We propose **GBT**: a two-stage **T**ransformer framework with **G**ood **B**eginning to alleviate the over-fitting problem caused by bad initialization methods of decoder inputs in TSFTs, especially when forecasting non-stationary time series.
2. GBT decouples the forecasting process of TSFT into *Auto-Regression* and *Self-Regression* stages to solve the problem

of different statistical properties of input/prediction windows when handling non-stationary time series. Specifically, the second *Self-Regression* stage employs the *E*rror *S*core *M*odification module (ESM) to utilize the *Good Beginning* provided by *Auto-Regression* stage in a more rational way.

3. Experiments on seven benchmark datasets demonstrate that GBT outperforms SOTA TSFTs and other forecasting models with only canonical attention and convolution while owning less time and space complexity.

4. The framework of GBT is general and adaptive enough to couple with other forecasting models, especially TSFTs, to achieve more promising performances.

2. Related works

2.1. Time series forecasting

Time series forecasting becomes a critical ingredient in various fields, such as stock prediction (Stádník et al., 2016), traffic forecasting (Cirstea et al., 2022b; Shuvo et al., 2021), sensor-based recognition (Ahmad et al., 2019) and COVID-19 pandemic analysis (Alassafi et al., 2022; Kumar & Susan, 2020; Zhao et al., 2021). Thus, various models have recently been proposed to forecast time series. With the development of deep learning and the need of tackling Long Sequence Time series Forecasting (LSTF) problem, traditional models, such as ARIMA (Box et al., 1974, 2015) and SSM (Durbin & Koopman, 2012), fail to challenge deep learning based models, including models based on RNN, CNN and Transformer.

2.2. Deep time series forecasting methods

RNN based methods (Hochreiter & Schmidhuber, 1997; Lai et al., 2018; Salinas et al., 2020; Weerakody et al., 2021) predict time series through multi-step rolling procedures, which means that they could not capture long-term dependency well. CNN is another feasible approach (LIU et al., 2022; Oord et al., 2016; Wang et al., 2021; Wibawa et al., 2022), which leverages dilated causal convolution to do time series forecasting. Different from the above two models, Transformer (Vaswani et al., 2017) shows better potential in capturing long-term dependency thanks to its self-attention mechanism, which helps model equally available to any part of time series sequences regardless of temporal distance. Recently, popular corresponding models or methods have already formed an approximately mature research field for time series forecasting Transformers (TSFT) (Cirstea et al., 2022a; Li et al., 2019; Liu et al., 2022b; Woo et al., 2022b; Wu et al., 2020, 2021; Zhou et al., 2022b, 2021). After one-forward time series forecasting procedure was proposed by Informer (Zhou et al., 2021), one-forward TSFT becomes popular on account of its outstanding performance. The researches on enhancing performance of one-forward TSFT include solving its quadratic complexity problem (Cirstea et al., 2022a; Li et al., 2019; Liu et al., 2022b; Zhou et al., 2021) and decoupling trend and seasonal components by Fourier transform (Woo et al., 2022b; Wu et al., 2021; Zhou et al., 2022b). However, they all do not fundamentally modify the architecture of TSFT, which is different from this paper. That is also why GBT could be combined with them easily.

2.3. Two-stage forecasting methods

Some self-supervised time series forecasting methods based on representation learning (Woo et al., 2022a; Yue et al., 2022) use two stage forecasting architectures. Their first stage is to seek the representation of input sequence through convolutions and obtain the universal feature map of input sequence through contrastive learning (Woo et al., 2022a; Yue et al., 2022); then their second stage is able to acquire prediction results though representations outputted by the first stage and Regressors (Woo et al., 2022a; Yue et al., 2022).

Additionally, some forecasting models are involved with diverse reconstruction strategies (Challu et al., 2022; Zhou et al., 2022a), which makes them analogous to 'two-stage model'. Though most of them own only one training/inference phase, their networks can be divided into reconstruction and forecasting parts. The participation of the reconstruction part renders them able to be more interpretably and precisely extracting feature maps of input sequences.

Obviously, either models with representation learning or with reconstruction have a chance to tackle distribution shifts of input sequences quite well. However, as we point out in Section 1, the distribution shift between input sequences and prediction sequences are more formidable to get around and it cannot be solved by these two techniques as they are only concerned with the features of input sequences.

The architecture of GBT is motivated from conventional two-stage models, but usages of two-stage architecture of GBT proposed in this paper are different from them in three points: (1) Outputs of both two stages of GBT are prediction results rather than representations. (2) The intention of using two-stage architecture in GBT is to tackle non-stationarity instead of strengthening feature extraction capability of networks. (3) GBT is built upon Transformer while these self-supervised methods are mainly based on CNN/MLP.

2.4. Non-stationarity handling technique

Negative influences to time series forecasting originated from non-stationarity have already caught attentions in recent years (Kim et al., 2022; Liu et al., 2022a). However, existing solutions are only involved of diverse normalization/de-normalization methods. RevIN (Kim et al., 2022) employs instance normalization and its reverse version at the beginning and ending of every single window. Non-stationary Transformer (Liu et al., 2022a) applies similar methods within attention mechanism. It could be seen that they do not essentially change the framework of TSFT unlike GBT.

3. Preliminary

We provide the definition of time series forecasting problem and name origins of GBT's two stages.

3.1. Time series forecasting problem

Given input window $\{z_{i,1:t_0}\}_{i=1}^N$, the task is to obtain prediction window $\{z_{i,t_0+1:T}\}_{i=1}^N$. N is the number of variates. t_0 denotes the input window size and $T - t_0$ is the prediction window size. If $N = 1$, we call it univariate time series forecasting. Otherwise, we call it multivariate time series forecasting.

One-forward/Direct (Taieb & Atiya, 2016) forecasting strategy, which predicts the whole prediction window simultaneously, becomes the mainstream forecasting formula in that it avoids possible error accumulation brought by recursive forecasting strategy (Lai et al., 2018; Salinas et al., 2020). Nearly all forecasting methods discussed in this paper employ direct forecasting strategy.

3.2. Time series forecasting transformer

Time series forecasting Transformer (TSFT) is developed from the Vanilla Transformer (Vaswani et al., 2017). Similar with the
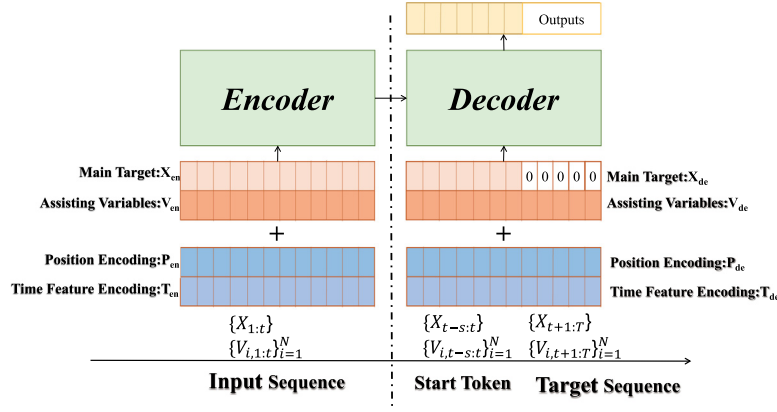
Fig. 3. Overview of the canonical architecture of time series forecasting Transformer. In this figure, we assume that the number of the main target to predict is one. t is the input window size and $T - t$ is the prediction window size. s stands for the length of the start token and N refers to the number of the associating variables.

canonical one, time series forecasting Transformer has an encoder–decoder architecture. Here, we only introduce the architecture of TSFT with direct forecasting procedure and zero-initialization of decoder inputs, which is majorly analyzed in the main text. As shown in Fig. 3, the encoder receives the value embedding of the past input sequence added with its position and time encoding features. Each layer of the encoder mainly contains a self-attention block and a feed-forward block. Decoder receives similar format of the following target sequence information but the unknown parts are padded into zero. Note that this is quite different from the canonical Transformer step-by-step 'dynamic decoding' process, for decoder now could obtain the whole outputs through one forward procedure, which avoids potential error accumulation brought by recursive decoding. Decoder extra receives the feature map from the encoder for cross-attention. Each layer of the decoder mainly contains a masked self-attention block, a cross-attention block and a feed-forward block. Start token is a popular technique to be employed to give decoder inputs a better start (Devlin et al., 2019; Zhou et al., 2021). However, it has been proved in our paper that it is not useful enough when tackling real-world non-stationary time series.

3.3. Origin of stage names in GBT

Canonical TSFTs are composed of encoders and decoders. The encoder and cross-attention module in decoder are *Auto-Regression* processes which aim to seek connections between input and prediction windows. Meanwhile, masked self-attention module in decoder belongs to *Self-Regression* process which is meant to deduce prediction elements from themselves. In fact, these two processes are both auto-regressive. We give them different names to distinguish them by their ***different auto-regressive inputs***. They also respectively correspond to two stages of GBT.

3.4. Meanings of abbreviations and phrases

The meanings of mentioned abbreviations and phrases used in this paper are shown in Table 1.

4. Methodology

4.1. Framework of GBT

4.1.1. Motivation

We have briefly introduced the motivation of the GBT framework in Section 1, i.e., current two existing solutions to help initialize decoder inputs are not enough to handle the over-fitting

Table 1
Meanings of abbreviations and phrases.

Abbr./Phrase	Meaning
TSFT	*T*ime *S*eries *F*orecasting *T*ransformer
Auto-Regression	Inference process of prediction elements by input sequences
Self-Regression	Inference process of prediction elements by themselves

problem of TSFTs caused by zero-initialized inputs in decoder and the non-stationarity of time series. We provide a more detailed analysis here to validate above statement.

The first solution is using start token in decoders to help initialize prediction elements (Zhou et al., 2021). This is the most common condition. Then let us check the inference process of the first masked self-attention within the decoder. Its scaled dot-product could be expressed as Eq. (1). $\{Q_1/K_1, Q_2/K_2\}$ refer to query/key of {start token, prediction elements} and d is the embedding dimension. Ignoring position embedding and bias, Q_2/K_2 will be zero tensors in that the linear projection of zero elements will also be zero elements. Then we can get query-key matching matrix as Eq. (2). It could be seen that attention scores allocated to prediction elements are all zero so start token has no effect on inference process of prediction elements and could not solve aforementioned problem.

$$Attn(Q, K, V) = Softmax(\frac{QK^\top}{\sqrt{d}})V, \quad Q = [Q_1, Q_2]^\top, \quad K = [K_1, K_2]^\top \quad (1)$$

$$QK^\top = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}\Bigg|_{Q_2=0} \times [K_1^\top, K_2^\top]|_{K_2=0} = \begin{bmatrix} Q_1 K_1^\top & 0 \\ 0 & 0 \end{bmatrix} \quad (2)$$

Some TSFTs additionally employ decomposed trend terms of input sequences from encoder (Wu et al., 2021; Zhou et al., 2022b) to initialize inputs of decoders. However, currently Q_1, K_1, Q_2, K_2 are generated from components of input sequences in Eq. (2). This is a better initialization for TSFTs when dealing with stationary time series. However, when facing non-stationary time series, it is no longer effective as trends of input/prediction sequence may be different. Therefore, we propose two-stage GBT, which decouples prediction processes of TSFT and transforms them into two different stages, i.e., the first *Auto-Regression* stage and the second *Self-Regression* stage, to solve this problem better. Its architecture is shown in Fig. 4.

4.1.2. Auto-regression stage

The first *Auto-Regression* stage is composed of N *Auto-Regression* Blocks to extract feature maps of input sequences. As Fig. 4
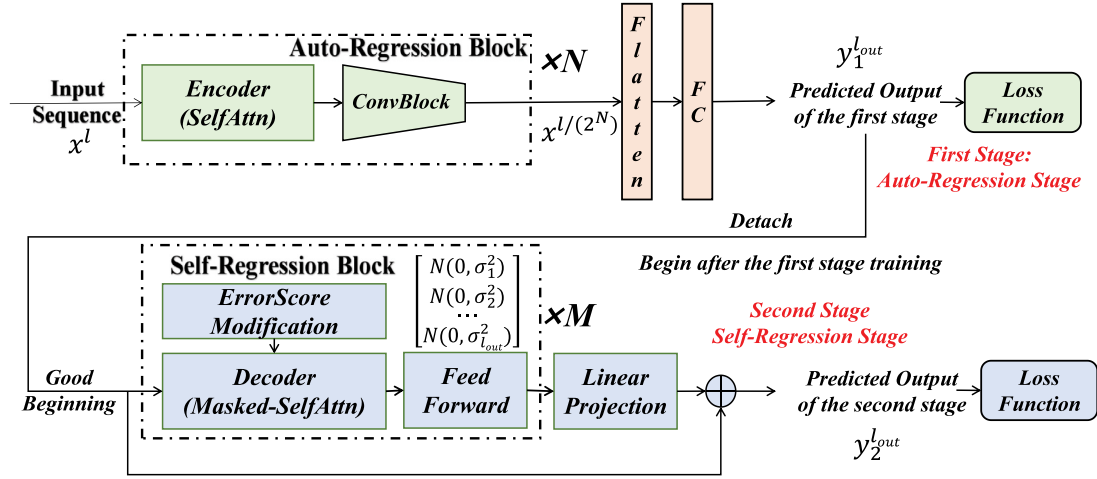
Fig. 4. An overview of whole GBT. The upper half is the *Auto-Regression* stage and the lower half is the *Self-Regression* stage. These two parts are trained separately. The concrete architecture of ConvBlock is shown in Fig. 5(a). We omit the pyramid architecture in *Auto-Regression* stage shown in Fig. 5(b).

shows, each *Auto-Regression* Block consists of one encoder and one ConvBlock. Its encoder is the same as the encoder of canonical Transformer but removing the feed-forward layer. Motivated by He et al. (2016) and Shen et al. (2022), ConvBlock replaces the feed-forward layer and uses convolutions to shrink the sequence length and double the hidden dimension shown in Fig. 5(a). In each ConvBlock, WN (Salimans & Kingma, 2016) is chosen as the normalization method and Gelu (Hendrycks & Gimpel, 2016) is employed as the activation function. ConvBlock is meant to enhance the locality of attention module (Kitaev et al., 2020) and smooth the turbulence (Woo et al., 2022b) brought by potential anomalies. We also employ similar pyramid network like Informer (Zhou et al., 2021) to extract hierarchical feature maps as shown in Fig. 5(b). Finally, we employ a FC layer to obtain prediction results at the first stage and to replace the cross-attention module of canonical TSFT which has the similar function of obtaining prediction sequence (feature map) through input sequence feature map in decoders. This replacement is established and rational in two reasons: (1) They share similar functions, i.e., *Auto-Regression* of prediction sequences through input sequences; (2) Currently the first stage only receives input sequences so that cross-attention module is unable to be applied. Consequently, prediction elements of the first stage are trained through the loss function and ground truths of prediction windows, so they will have more similar statistical properties with those of truth compared with former initialization methods. Thus, they are '*Good Beginning*' for the second stage input. Note that networks of the second stage *do not participate* in the training of the first stage.

4.1.3. Self-regression stage

After the training of the first stage, we have acquired *Auto-Regression* parts of prediction elements through input sequences and it is '*Good Beginning*' for the input of the second *Self-Regression* stage. For the second stage is an inference process of prediction elements by themselves, it contains M decoders composed of a masked self-attention module and a corresponding feed-forward layer. Compared with the decoder of canonical Transformer, it abandons cross-attention in that it is an *Auto-Regression* module which is redundant in *Self-Regression* process. Finally, we use a linear projection layer to obtain prediction results. Parameters of the first stage network will be *fixed* during the training phase of the second stage. Additionally, within the second stage, we employ the Error Score Modification module to manage the error

accumulation of '*Good Beginning*' provided by the first stage, which is shown as following.

4.2. Error score modification

4.2.1. Motivation

As '*Good Beginning*' provided by the first stage is the result of *Auto-Regression*, it can be deduced from former analysis in Section 1 and Fig. 1(b) that later prediction elements will be further from truth in the statistical sense. This phenomenon is elaborated in Taieb and Atiya (2016). Therefore, we need an additional mechanism specified for this problem to rationally deduce the prediction sequence from the auto-regressive results of the first stage. As the later prediction elements of the output sequence of the first stage is less reliable, it is reasonable to guide the second stage to allocate more scores to earlier elements, which are more conceivable to approach the real value, during the attention.

4.2.2. Architecture

We adopt a group of learnable Gauss distribution $N(0, \sigma_i^2)$, $i \in [1, l_{out}]$ (Xu et al., 2022) and add it into the masked query-key matching matrix in the second stage shown in Fig. 4. Its concrete formula is presented in Eq. (3). $QK^\top$ is the query-key matching matrix with size $l_{out} \times l_{out}$ where l_{out} is the prediction sequence length. $N(0, \sigma_i^2)$ is a learnable Gauss distribution where the center is zero and the scale σ is a learnable parameter. The center is fixed to zero in ESM so that more masked self-attention scores will be allocated to earlier and more trusted prediction elements, which means that earlier prediction elements will make more contributions for the *Self-Regression* of any prediction element during the second stage. The σ_i is learnable so that ESM will be more adaptive. The pseudo-code of ESM is given as Algorithm 1.

$$ESM(Q, K) = QK^\top + \begin{bmatrix} N(0, \sigma_1^2) \\ N(0, \sigma_2^2) \\ \cdots \\ N(0, \sigma_{l_{out}}^2) \end{bmatrix} \tag{3}$$

4.3. Complexity analysis

Essentially, GBT is a transformed and decoupled Transformer framework with less time and space complexity: (1) Feed-forward layers in encoders are replaced with ConvBlocks which shrink
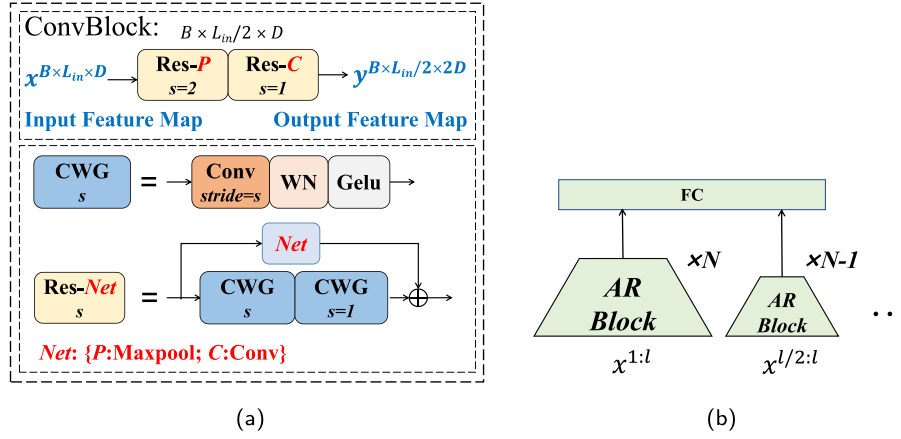
Fig. 5. ConvBlock shown in (a) is composed of Res-P and Res-C. Res-P is employed to shrink the sequence length by half and Res-C is employed to double the hidden dimension. (b) is the pyramid architecture similar with Informer. AR Block refers to the *Auto-Regression* Block. Extra pyramid networks own fewer AR Blocks with shorter input sequence.

Algorithm 1 Masked self-attention with ESM

Input: Tensor $x \in \mathbb{R}^{L \times d}$
Layer params: $Linear(x)$: Linear projection layer of x
1: $Q, K, V = Linear_1(x)$ $(Q, K, V \in \mathbb{R}^{L \times d})$
2: $\sigma = Linear_2(x)$ $(\sigma \in \mathbb{R}^{L \times 1})$
3: Calculate self-attention scores $A = QK^\top$ $(A \in \mathbb{R}^{L \times L})$
4: σ = Sigmoid(5σ) + 1e-5
5: $\sigma = 3^\sigma - 1$
6: σ = Repeat$(\sigma, \text{dim=1})$ $(\sigma \in \mathbb{R}^{L \times L})$
7: Generate Gauss distribution matrix $G \sim N(0, \sigma^2)$ $(G \in \mathbb{R}^{L \times L})$
8: Add self-attention scores with ESM: $A = A + G$
9: $A = mask(A)$
Output: Masked self-attention scores A

Table 2
Details of eight datasets.

Dataset	Size	Dimension	Frequency
ETTh$_1$	17 420	7	1 h
ETTm$_2$	69 680	7	15 min
WTH	35 064	7	1 h
ECL	26 304	321	1 h
Traffic	17 544	862	1 h
Exchange	7588	8	1 day
weather	52 696	21	10 min
ILI	966	8	7 days

input sequence length, thus mediately decreasing complexity of self-attention; (2) Due to two-stage architecture, start token is no longer needed so the sequence length within decoder of GBT is shorter than that of canonical TSFT; (3) A FC layer is employed to play the role of cross-attention in the *Auto-Regression* stage. Though these two modules both have $O(L^2)$ complexity, GBT gets rid of plenty of linear projection layers in cross attention modules which bring considerable computation. Note that GBT is only a framework so any modified TSFT could be introduced into GBT to acquire less complexity. We validate above analysis in later Section 5.8.

5. Experiment

5.1. Dataset

To evaluate the forecasting capability of our proposed GBT, we perform experiments on real-world datasets. Currently, there are dozens of benchmark time series forecasting datasets which are widely chosen by other state-of-the-art forecasting models (LIU et al., 2022; Wu et al., 2021; Yue et al., 2022; Zhou et al., 2022b). Among them, we choose {ETTh$_1$, ETTm$_2$} in {ETTh$_1$, ETTh$_2$, ETTm$_1$, ETTm$_2$} in that these two have different sampling intervals and are more commonly used. The rest of chosen datasets are {ECL, WTH, weather, Traffic, Exchange, ILI}. We briefly introduce these eight selected datasets[12] and their usages in this paper below and their numeral details (Size, Dimension and Frequency) are shown in Table 2:

ETT (Electricity Transformer Temperature) dataset (Zhou et al., 2021) is composed of ETT data collected from two separated counties in China lasting for almost 2 years. It contains four sub-sets: {ETTh$_1$, ETTh$_2$} are 1-hour-level datasets; {ETTm$_1$, ETTm$_2$} are 15-min-level datasets. Each data point is composed of the target value 'OT' (oil temperature) and other 6 power load features. For averting unnecessary experiments, we choose one 1-hour-level subset, ETTh$_1$, and one 15-min-level subset, ETTm$_2$, to do experiments on. The train/val/test is 12/4/4 months.

WTH dataset contains local climatological data from nearly 1,600 U.S. locations. It is a 1-hour-level dataset spanning 4 years from 2010 to 2013. Each data point consists of the target value 'Wet-BulbCelsius' and 11 climate features. The train/val/test is 60%/20%/20% according to settings of other baselines (Woo et al., 2022a; Zhou et al., 2021).

ECL (Electricity Consuming Load) dataset contains time series of electricity consumption (Kwh) from 321 clients. It is a 1-hour-level dataset which is converted into 2 years by Zhou et al. (2021) and 'MT_321' (Woo et al., 2022b; Wu et al., 2021; Zhou et al., 2022b) is the target value according to the most common settings (some papers use 'MT_001' (Yue et al., 2022)/'MT_320'(Zhou et al., 2021), but they are not the most commonly used). The train/val/test is 70%/10%/20% following Woo et al. (2022b), Wu et al. (2021) and Zhou et al. (2022b).

Traffic dataset is collected hourly from the California Department of Transportation to describe road occupancy rates in San Francisco Bay area freeways spanning 48 months (2015–2016). The

---

[1] {ETTh$_1$, ETTm$_2$, ECL, weather, Traffic, Exchange, ILI} datasets were acquired at:
https://drive.google.com/drive/folders/1ZOYpTUa82_jCcxIdTmyr0LXQfvaM9vIy?usp=sharing

[2] WTH dataset was acquired at: https://www.ncdc.noaa.gov/orders/qclcd/

Table 3
Details of hyper-parameters/settings.

Hyper-Parameters/Settings	Values/Mechanisms
The number of pyramid networks	3
The number of AR Blocks in the main pyramid network	3
Embedding dimensions of *Auto-Regression* stage	32
Embedding dimensions of *Self-Regression* stage	512
The kernel size of Conv layers	3
Input window size	96 (36 only for ILI)
Standardization	Z-score
Loss function	MSE
Optimizer	Adam
Dropout	0.1
Learning rate	1e$-$4
Learning rate decreasing rate	Half per epoch
Batch size	32
Random seed	4321 (if used)
Platform	Python 3.8.0; Pytorch 1.11.0
Device	A single NVIDIA GeForce RT $\times$ 3090 24 GB GPU

train/val/test is 70%/10%/20%. We set the variate name of target as 'OT' in {Traffic, Exchange, weather, ILI} for convenience according to Lai et al. (2018) and Zhou et al. (2022b).

Exchange dataset (Lai et al., 2018) is a 1-day-level dataset, including daily exchange rates of eight countries from 1990 to 2016. The train/val/test is 70%/10%/20%.

Weather dataset is a 10-min-level dataset which contains 21 meteorological indicators in Germany during 2020. The train/val/test is 70%/10%/20%.

ILI (Influenza-like Illness) dataset, is a 1-week-level dataset containing recorded influenza-like illness patients data in USA between 2002 and 2021. The train/val/test is 70%/10%/20%.

5.2. Baselines

As our proposed GBT aims to alleviate the over-fitting problem of TSFTs when facing non-stationary time series, we select six state-of-the-art TSFTs, i.e., {Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022b), Pyraformer (Liu et al., 2022b), ETSformer (Woo et al., 2022b), Airformer (Liang et al., 2023)}, as baselines. We additionally choose other six state-of-the-art forecasting models not based on Transformer, i.e., SCINet (LIU et al., 2022), TS2Vec (Yue et al., 2022), FiLM (Zhou et al., 2022a) N-BEATS (Oreshkin et al., 2020), N-HiTS (Challu et al., 2022) and DLinear (Zeng et al., 2022), as baselines for further comparison. Specially, Airformer, TS2Vec, FiLM and N-HiTS are involved with reconstruction or representation learning strategies. Brief introductions of these twelve baselines are shown in Appendix B.

5.3. Experiment details

Details of hyper-parameters/settings of experiments are shown in Table 3. Most of hyper-parameters are set identically to those commonly used ones. The number of pyramid networks and the number of AR Blocks in the main pyramid network are all set to 3 according to Shen et al. (2022) and Zhou et al. (2021) in that pyramid networks used in the first stage of GBT are motivated from them. In other experiments when GBT is compared with other methods or combined with other baselines, we reference

corresponding hyper-parameters/settings to ensure that experiment results are obtained under the same circumstances for persuasive comparison.

$MSE = \sum (x_i - \hat{x}_i)^2 / n$ and $MAE = \sum |x_i - \hat{x}_i| / n$ are used as metrics. All experiments are repeated for 5 times except special instruction and their average results are taken as final results. During multivariate forecasting of eight benchmark datasets, variates are treated as relevant variates only in WTH and are considered as different instances in other datasets (Yue et al., 2022). Most of results of these baselines are directly taken from their papers if existing experiment results with input sequence length of 96 and we supplement rest of experiments using their default settings except input sequence length is fixed as 96. Note that we would rather redo experiments of many chosen baselines in that their results in the original papers are produced by the models with different input sequence lengths. However, Wu et al. (2021) points out that input sequence length counts tremendously for the prediction performances. Therefore, we fix the input sequence length of all baselines as 96, which is the commonly agreed-upon number proposed by Wu et al. (2021) (36 for ILI dataset following the same research). '—' means that models fail for the out-of-memory (24 GB) even when the size of batch is 1. ***The best results are highlighted in bold and italic*** while *the second best results are highlighted in underline and italic* except specific instructions.

5.4. Main results

We compare forecasting capabilities of GBT with most of baselines and datasets aforementioned. We choose a commonly used prediction length group (Woo et al., 2022b; Wu et al., 2021; Zhou et al., 2022b), i.e., {96, 192, 336, 720}, to perform multivariate/univariate forecasting experiments. In these experiments, as GBT only employs the canonical self-attention selected from Vanilla Transformer (Vaswani et al., 2017), it is named as 'GBT-Vanilla'. The results of FEDformer and DLinear are separately the results of FEDformer-f and DLinear-S, which are their default and relatively better versions in general. The full results of their multiple versions, i.e., {FEDformer-f, FEDformer-w} and {DLinear-S, DLinear-I}, are presented in Appendix D.6. The results of GBT-Vanilla in this section are also presented there for further comparison. Note that DLinear-I and DLinear-S are identical under univariate forecasting conditions (Zeng et al., 2022).

We remove results of few datasets and baselines in this section to avoid excessive stacking of data. Concrete reasons are shown as below:

1. We choose WTH in {WTH, weather} in that they are all datasets about climate and WTH is broader. However, we still provide forecasting results of all baselines on weather in Appendix D.1.

2. We abandon ILI here in that it is a quite small dataset compared with others and has clear seasonal terms for all variates (Challu et al., 2022), which means that non-stationarity is not apparent in ILI. Note that other datasets all contain non-stationary time series. Thus, GBT will not be very effective in improving forecasting capabilities of TSFTs on ILI so that performances of GBT-Vanilla is not satisfactory. However, we will show that GBT combined with frequence-enhanced methods, e.g., Autoformer (Wu et al., 2021), can still achieve state-of-the-art performances in Appendix D.2.

3. Results of Informer, Autoformer and N-BEATS are removed in this section as their forecasting capabilities have been proved to be worse than at least half of the rest baselines. However, they are still very typical models, hence we will

Table 4
Results of multivariate forecasting.

Methods	Metrics	ETTh$_1$				ETTm$_2$				ECL			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	**0.398**	**0.448**	**0.497**	**0.538**	**0.189**	**0.249**	*0.324*	**0.395**	**0.143**	**0.175**	**0.197**	*0.235*
	MAE	**0.418**	**0.442**	**0.470**	**0.505**	**0.276**	*0.324*	0.368	0.419	**0.246**	**0.277**	**0.298**	**0.336**
FEDformer	MSE	0.415	0.474	0.535	0.680	0.203	0.269	0.325	0.421	0.193	*0.201*	0.214	0.246
	MAE	0.453	0.493	0.524	0.593	0.287	0.328	*0.366*	*0.415*	0.308	0.315	0.329	0.355
Pyraformer	MSE	0.662	0.791	0.902	0.974	0.378	1.192	1.176	6.720	0.418	0.408	0.410	0.407
	MAE	0.611	0.683	0.734	0.780	0.456	0.870	1.033	2.077	0.460	0.454	0.457	0.456
ETSformer	MSE	0.511	0.561	0.599	0.588	*0.189*	*0.253*	**0.314**	*0.414*	*0.187*	0.199	*0.212*	**0.233**
	MAE	0.487	0.513	0.529	*0.541*	0.280	**0.319**	**0.357**	**0.413**	0.304	0.315	0.329	0.345
AirFormer	MSE	1.061	1.108	1.130	1.179	0.812	0.884	1.263	4.174	0.382	0.381	0.383	0.390
	MAE	0.813	0.836	0.846	0.886	0.734	0.766	0.873	1.660	0.441	0.443	0.445	0.452
SCINet	MSE	0.531	0.535	0.584	0.685	0.312	0.573	1.870	3.462	0.210	0.234	0.227	0.269
	MAE	0.503	0.513	0.560	0.634	0.415	0.591	1.078	1.753	0.333	0.345	0.340	0.373
TS2Vec	MSE	0.670	0.781	0.911	1.059	0.360	0.534	0.833	1.906	0.336	0.337	0.350	0.375
	MAE	0.588	0.651	0.718	0.794	0.426	0.537	0.694	1.054	0.412	0.415	0.426	0.438
DLinear	MSE	0.431	0.474	0.518	*0.560*	0.199	0.287	0.387	0.544	0.246	0.246	0.260	0.294
	MAE	0.451	0.479	0.508	0.559	0.395	0.363	0.429	0.506	0.345	0.348	0.361	0.388
N-HiTS	MSE	*0.411*	*0.465*	*0.523*	0.592	0.192	0.284	0.346	0.520	0.198	0.205	0.218	0.254
	MAE	*0.419*	*0.453*	*0.492*	0.556	*0.277*	0.350	0.376	0.485	*0.291*	*0.301*	*0.314*	*0.342*
FiLM	MSE	0.700	0.718	0.721	0.722	0.230	0.285	0.339	0.433	0.845	0.849	0.861	0.891
	MAE	0.555	0.570	0.579	0.604	0.307	0.338	0.370	0.420	0.761	0.761	0.764	0.774

Methods	Metrics	WTH				Traffic				Exchange			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	**0.434**	**0.481**	**0.514**	**0.523**	**0.509**	**0.520**	**0.535**	**0.575**	0.110	**0.179**	0.358	**0.756**
	MAE	**0.466**	**0.506**	**0.527**	**0.532**	**0.282**	**0.293**	**0.307**	**0.317**	0.249	*0.312*	0.446	**0.655**
FEDformer	MSE	0.509	0.581	0.630	0.580	0.587	0.604	0.621	0.626	0.148	0.271	0.460	1.195
	MAE	0.513	0.557	0.636	0.586	*0.366*	0.373	0.383	0.382	0.278	0.380	0.500	0.841
Pyraformer	MSE	0.540	0.575	0.593	0.623	0.938	0.939	0.948	–	1.489	1.642	1.744	2.080
	MAE	0.546	0.567	0.578	0.599	0.490	0.488	0.488	–	1.018	1.075	1.107	1.197
ETSformer	MSE	0.538	0.615	0.655	0.719	0.607	0.621	0.622	0.632	**0.085**	*0.182*	**0.348**	1.025
	MAE	0.521	0.566	0.589	0.624	0.392	0.399	0.396	0.396	**0.204**	**0.303**	**0.428**	0.774
AirFormer	MSE	0.504	0.562	0.580	0.619	0.849	0.856	0.866	1.030	0.938	1.000	1.164	1.720
	MAE	0.520	0.560	0.570	0.595	0.476	0.478	0.480	0.613	0.840	0.870	0.932	1.115
SCINet	MSE	0.489	0.526	0.572	0.617	0.581	0.595	–	–	0.221	0.323	0.661	2.691
	MAE	0.495	0.524	0.562	0.586	0.423	0.429	–	–	0.365	0.442	0.564	1.320
TS2Vec	MSE	*0.450*	*0.505*	*0.532*	*0.566*	0.941	–	–	–	0.184	0.373	0.666	2.941
	MAE	*0.472*	*0.515*	*0.533*	*0.557*	0.550	–	–	–	0.315	0.452	0.612	1.313
DLinear	MSE	0.539	0.592	0.610	0.653	0.725	0.665	0.674	0.716	0.219	0.350	0.563	1.076
	MAE	0.522	0.557	0.571	0.600	0.460	0.438	0.441	0.457	0.387	0.481	0.606	0.799
N-HiTS	MSE	0.488	0.539	0.565	0.628	*0.561*	*0.543*	*0.554*	*0.616*	*0.099*	0.297	0.576	1.288
	MAE	0.496	0.536	0.555	0.596	0.369	*0.352*	*0.357*	*0.377*	*0.225*	0.390	0.550	0.847
FiLM	MSE	0.515	0.585	0.619	0.688	1.409	1.412	1.428	1.451	0.141	0.216	*0.351*	*0.938*
	MAE	0.497	0.540	0.563	0.602	0.799	0.802	0.806	0.809	0.272	0.342	*0.434*	*0.736*

perform other types of experiments with them in later sections.

It could be observed from Table 4/5 that GBT outperforms other baselines in most of univariate/multivariate forecasting situations even when it only employs the canonical attention mechanism. When compared with FED-former/Pyraformer/ETSformer/ AirFormer/SCINet/TS2Vec/DLinear/N-HiTS/FiLM, GBT-Vanilla yields 15.1%/52.6%/ 10.6%/61.1%/32.3%/ 42.1%/21.5%/12.6%/46.5% relative MSE reduction during multivariate forecasting and yields 23.4%/58.0%/27.4%/48.1%/28.9%/33.5%/31.8%/37.9%/67.5% relative MSE reduction during univariate forecasting in general.

5.5. Ablation study

In this sub-section, ablation study is conducted to examine functions of two-stage framework, ESM, ConvBlock and pyramid architecture in GBT-Vanilla (GBT for short in the following). Six ablation variants of GBT are tested: 1) GBT-first: GBT only with the first stage; 2) GBT-second: GBT-Vanilla; 3) GBT-simul: GBT-Vanilla with simultaneously training of two stages. 4) GBT-wo-ESM: GBT without ESM; 5) GBT-wo-CB: GBT with feed-forward layers instead of ConvBlock in the first stage; 6) GBT-wo-Pyra: GBT without pyramid networks. Results of the second stage are shown except GBT-first in Table 6.

As Table 6 shows, GBT-second performs better than other methods when forecasting under ETTh$_1$. GBT-Vanilla acquires 7.0%/5.2%/10.1% relative MSE increase when we remove the second stage/ESM module/pyramid architecture, 17.9% relative MSE increase when we replace ConvBlock with feed-forward layer and 27.6% relative MSE increase when we simultaneously training both two stages of GBT. It means that without any of these four necessary components, GBT will suffer from worse performance due to diverse negative effects caused by them. This proves benefits of all components of GBT proposed by us. Specially, GBT-simul

Table 5
Results of univariate forecasting.

Methods	Metrics	ETTh$_1$				ETTm$_2$				ECL			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	**0.051**	**0.074**	**0.080**	_0.119_	**0.068**	**0.091**	**0.109**	**0.163**	_0.254_	**0.282**	**0.324**	_0.359_
	MAE	**0.173**	**0.206**	**0.221**	_0.276_	**0.194**	**0.229**	**0.257**	**0.316**	**0.363**	**0.386**	**0.417**	**0.444**
FEDformer	MSE	0.103	0.129	0.132	0.134	0.072	_0.102_	0.130	0.178	**0.253**	_0.282_	0.346	0.422
	MAE	0.252	0.285	0.291	0.293	0.206	_0.245_	_0.279_	_0.325_	_0.370_	_0.386_	0.431	0.484
Pyraformer	MSE	0.143	0.159	0.196	0.230	0.461	0.781	1.372	5.780	0.347	0.436	0.493	0.614
	MAE	0.309	0.322	0.372	0.410	0.527	0.683	0.913	1.878	0.432	0.493	0.526	0.605
ETSformer	MSE	_0.060_	_0.081_	0.098	0.119	0.080	0.110	0.136	0.185	0.726	0.667	0.770	0.766
	MAE	_0.190_	_0.221_	0.248	0.282	0.213	0.252	0.283	0.333	0.656	0.625	0.677	0.674
AirFormer	MSE	0.169	0.215	0.252	0.336	0.094	0.129	0.167	0.246	0.508	0.506	0.531	0.583
	MAE	0.337	0.385	0.423	0.509	0.236	0.279	0.322	0.395	0.543	0.534	0.548	0.583
SCINet	MSE	0.119	0.129	0.160	0.243	0.076	0.102	_0.129_	_0.176_	0.312	0.314	_0.332_	0.364
	MAE	0.269	0.280	0.322	0.414	0.210	0.248	0.280	0.328	0.411	0.416	_0.427_	0.451
TS2Vec	MSE	0.098	0.153	0.169	0.164	0.088	0.122	0.158	0.200	0.315	0.333	0.347	**0.350**
	MAE	0.241	0.302	0.326	0.327	0.224	0.271	0.314	0.357	0.419	0.430	0.440	_0.447_
DLinear	MSE	0.111	0.136	0.166	0.280	0.094	0.130	0.164	0.223	0.411	0.385	0.410	0.447
	MAE	0.258	0.286	0.325	0.453	0.237	0.278	0.316	0.369	0.473	0.455	0.470	0.502
N-HiTS	MSE	0.144	0.172	0.178	0.291	_0.071_	0.113	0.164	0.226	0.328	0.343	0.395	0.449
	MAE	0.308	0.338	0.342	0.463	_0.195_	0.251	0.314	0.374	0.405	0.412	0.449	0.489
FiLM	MSE	0.066	0.083	_0.097_	**0.102**	0.152	0.175	0.197	0.238	0.967	0.958	0.992	1.038
	MAE	0.199	0.225	_0.248_	**0.252**	0.304	0.325	0.347	0.387	0.795	0.788	0.799	0.818

Methods	Metrics	WTH				Traffic				Exchange			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	_0.188_	**0.221**	**0.239**	**0.218**	**0.133**	**0.140**	**0.138**	**0.174**	**0.100**	**0.186**	_0.408_	_0.925_
	MAE	_0.318_	**0.348**	**0.372**	**0.349**	**0.222**	**0.228**	**0.234**	**0.268**	**0.249**	**0.343**	0.522	_0.743_
FEDformer	MSE	0.233	0.291	0.318	0.331	_0.207_	_0.205_	_0.219_	_0.244_	0.154	0.286	0.511	1.301
	MAE	0.353	0.406	0.422	0.432	_0.312_	_0.312_	_0.323_	_0.344_	0.304	0.420	0.555	0.879
Pyraformer	MSE	0.213	0.262	0.303	0.398	0.501	0.541	0.557	0.596	0.627	1.010	1.227	1.742
	MAE	0.342	0.383	0.415	0.483	0.512	0.532	0.541	0.561	0.639	0.820	0.915	1.134
ETSformer	MSE	0.243	0.296	0.339	0.432	0.243	0.241	0.240	0.252	_0.100_	0.226	0.434	0.990
	MAE	0.363	0.400	0.430	0.492	0.355	0.352	0.353	0.362	_0.252_	_0.353_	**0.500**	0.821
AirFormer	MSE	0.226	0.272	0.308	0.396	0.540	0.504	0.532	0.554	0.246	0.508	0.915	0.977
	MAE	0.358	0.394	0.425	0.488	0.537	0.516	0.531	0.544	0.400	0.577	0.777	0.786
SCINet	MSE	0.213	0.255	0.287	0.352	0.217	0.299	0.259	0.278	0.209	0.347	0.575	1.378
	MAE	0.341	0.375	0.399	0.449	0.330	0.397	0.365	0.379	0.366	0.475	0.604	0.939
TS2Vec	MSE	0.199	0.240	_0.262_	_0.281_	0.357	0.359	0.368	0.380	0.184	0.373	0.666	2.941
	MAE	0.323	0.361	_0.384_	_0.405_	0.431	0.433	0.440	0.447	0.315	0.452	0.612	1.313
DLinear	MSE	0.207	0.257	0.293	0.378	0.361	0.309	0.305	0.351	0.118	_0.222_	**0.400**	**0.837**
	MAE	0.336	0.376	0.402	0.470	0.442	0.395	0.392	0.425	0.277	0.382	_0.506_	**0.722**
N-HiTS	MSE	**0.183**	_0.227_	0.265	0.359	0.284	0.264	0.269	0.298	0.241	0.870	1.809	2.144
	MAE	**0.307**	_0.352_	0.384	0.462	0.369	0.354	0.361	0.384	0.372	0.700	1.061	1.152
FiLM	MSE	0.206	0.264	0.309	0.410	1.861	1.845	1.836	1.828	0.152	0.256	0.463	1.002
	MAE	0.322	0.370	0.405	0.475	1.173	1.169	1.167	1.163	0.308	0.406	0.531	0.774

Table 6
Results of ablation study on GBT-Vanilla.

Methods	Metrics	ETTh$_1$ (Univariate)				ETTh$_1$ (Multivariate)			
		96	192	336	720	96	192	336	720
GBT-first	MSE	_0.060_	_0.077_	0.096	_0.128_	0.406	0.472	0.515	0.555
	MAE	_0.186_	0.210	0.240	_0.282_	0.419	0.461	0.471	_0.512_
GBT-second	MSE	**0.051**	**0.074**	**0.080**	**0.119**	0.398	_0.448_	0.497	**0.538**
	MAE	**0.173**	**0.206**	**0.221**	**0.276**	_0.418_	_0.442_	0.470	**0.505**
GBT-wo-simul	MSE	0.092	0.167	0.189	0.284	0.412	0.449	0.500	_0.543_
	MAE	0.237	0.331	0.342	0.455	0.431	0.448	0.476	0.521
GBT-wo-ESM	MSE	0.061	0.076	_0.083_	0.134	0.407	0.467	0.483	0.564
	MAE	0.186	_0.209_	_0.223_	0.304	0.420	0.458	_0.459_	0.512
GBT-wo-CB	MSE	0.085	0.079	0.149	0.246	**0.381**	**0.431**	**0.474**	0.614
	MAE	0.223	0.213	0.317	0.420	**0.401**	**0.422**	**0.452**	0.523
GBT-wo-Pyra	MSE	0.064	0.078	0.098	0.156	0.416	0.478	_0.488_	0.563
	MAE	0.193	0.212	0.243	0.326	0.431	0.468	0.468	0.516

has the worse performance among all variants, illustrating that two-stage architecture is the core of GBT.

Apart from component contained in GBT, we extra examine whether cross-attention module and start token are redundant

Table 7
Results of GBT with cross-attention or start token.

Methods	Metrics	ETTh$_1$ (Univariate)				ETTh$_1$ (Multivariate)			
		96	192	336	720	96	192	336	720
GBT-second	MSE	***0.051***	0.074	*0.080*	***0.119***	***0.398***	***0.448***	*0.497*	***0.538***
	MAE	***0.173***	*0.206*	*0.221*	***0.276***	***0.418***	***0.442***	0.470	***0.505***
GBT-w-cross	MSE	*0.061*	***0.073***	0.095	0.235	0.403	*0.449*	***0.465***	0.584
	MAE	*0.187*	***0.204***	0.238	0.408	0.420	*0.442*	***0.442***	*0.515*
GBT-w-st	MSE	0.063	*0.074*	***0.076***	*0.166*	0.404	0.463	0.498	*0.582*
	MAE	0.192	0.207	***0.221***	*0.331*	0.420	0.460	*0.465*	0.534

Table 8
Results of GBT combined with other baselines.

Methods	Metrics	WTH (Univariate)				WTH (Multivariate)			
		96	192	336	720	96	192	336	720
GBT + FEDformer	MSE	***0.171***	***0.200***	***0.204***	***0.202***	***0.449***	***0.489***	***0.504***	***0.508***
	MAE	***0.303***	***0.331***	***0.334***	***0.332***	***0.473***	***0.500***	***0.510***	***0.515***
GBT + ETSformer	MSE	***0.202***	***0.227***	***0.234***	***0.231***	***0.444***	***0.494***	***0.506***	***0.532***
	MAE	***0.329***	***0.359***	***0.350***	***0.369***	***0.474***	***0.515***	***0.520***	***0.544***
GBT + SCINet	MSE	***0.200***	***0.230***	***0.232***	***0.212***	***0.436***	***0.484***	***0.507***	***0.509***
	MAE	***0.325***	***0.351***	***0.352***	***0.339***	***0.471***	***0.504***	***0.522***	***0.519***
GBT + DLinear	MSE	***0.196***	***0.235***	***0.237***	***0.225***	***0.497***	***0.547***	***0.577***	***0.627***
	MAE	***0.324***	***0.358***	***0.359***	***0.352***	***0.509***	***0.543***	***0.568***	***0.595***

Table 9
Comparison experiment with RevIN.

Methods	Metrics	ETTh$_1$ (Multivariate)				ETTm$_1$ (Multivariate)			
		168	336	720	960	48	96	288	672
Informer	MSE	1.138	1.278	1.357	1.470	0.499	0.605	0.906	0.943
	MAE	0.853	0.909	0.945	0.990	0.486	0.554	0.738	0.760
Informer + RevIN	MSE	*0.655*	*1.058*	*0.926*	*0.902*	*0.390*	*0.405*	*0.563*	*0.663*
	MAE	*0.561*	*0.758*	*0.717*	*0.715*	*0.391*	*0.411*	*0.502*	*0.550*
Informer + GBT	MSE	***0.443***	***0.511***	***0.555***	***0.596***	***0.268***	***0.308***	***0.363***	***0.429***
	MAE	***0.440***	***0.473***	***0.512***	***0.528***	***0.337***	***0.366***	***0.397***	***0.442***

in *Self-Regression* stage of GBT. To employ cross-attention module, we extra employ a network which is the same as the network of the first stage into the second stage to acquire key and value generated from input sequences as parts of inputs of cross-attention module during the second stage. To employ start token, we concatenate start token with outputs of the first stage and treat the whole of them as inputs of the second stage. Therefore, two additional variants of GBT are experimented: 1) 'GBT-w-cross' refers to GBT with cross-attention module; 2) 'GBT-w-st' denotes GBT with start token; in Table 7.

As Table 7 shows, after we employ cross-attention module/start token into the second stage, forecasting results get worse in most of conditions. In general, GBT-second acquires 12.3%/6.8% relative MSE increase after we employ cross-attention module/start token. Therefore, cross-attention and start token are redundant and even do harm to performance of GBT in most of situations.

5.6. Generality and adaptivity analysis

We combine GBT with FEDformer and ETSformer separately and perform experiments on WTH to estimate the generality and adaptivity of GBT with TSFTs. We also combine GBT with DLinear and SCINet to further illustrate that its combination is not limited to TSFTs. Their combination formats are shown in Appendix C. **The results which outperform corresponding ones shown in Table 4/Table 5 are highlighted in bold and italic.**

It could be observed from Table 8 that forecasting results of these three models are greatly improved under both univariate and multivariate situations of WTH after combined with GBT. Though they could not challenge TS2Vec, whose performances rank second under the forecasting of WTH in Table 4/Table 5 (GBT-Vanilla ranks first), in their initial formulas, they surpass TS2Vec after combined with GBT in most of forecasting situations under WTH. FEDformer/ETSformer/SCINet/DLinear separately gain 28.6%/25.4%/15.3%/10.8% relative MSE reductions after combined with GBT in general. Meanwhile, GBT+FEDformer even outperforms GBT-Vanilla, demonstrating that GBT is general and adaptive to be combined with other SOTA modified TSFTs to achieve superior forecasting accuracy in the future.

5.7. Ability to handle non-stationary time series

We conduct comparison experiments with RevIN (Kim et al., 2022) and Non-stationary Transformer (Liu et al., 2022a), which are other two advanced methods to help forecasting models deal with non-stationary time series. Multivariate forecasting conditions of {ETTh$_1$, ETTm$_1$} and {Exchange, weather} are separately chosen to perform comparison experiments with RevIN and Non-stationary Transformer. We choose different datasets for these two methods in that chosen datasets are their respective experiment circumstances in their original papers. Informer is selected as the baseline and experiment settings are chosen according to their original papers for fair comparison. Comparison results are respectively shown in Tables 9 and 10. As Table 9

Table 10

Comparison experiment with Non-stationary Transformer.

Methods	Metrics	Exchange (Multivariate)				Weather (Multivariate)			
		96	192	336	720	96	192	336	720
Informer	MSE	0.847	1.204	1.672	2.478	0.300	0.598	0.578	1.059
	MAE	0.752	0.895	1.036	1.310	0.384	0.544	0.523	0.741
Informer + Non-stationary Transformer	MSE	*0.129*	*0.251*	*0.373*	*1.229*	*0.186*	*0.259*	*0.295*	*0.361*
	MAE	*0.258*	*0.354*	*0.434*	*0.795*	*0.235*	*0.292*	*0.317*	*0.362*
Informer + GBT	MSE	***0.090***	***0.188***	***0.294***	***0.776***	***0.169***	***0.209***	***0.268***	***0.331***
	MAE	***0.218***	***0.320***	***0.406***	***0.661***	***0.226***	***0.265***	***0.313***	***0.358***

Table 11

The training/inference time and GPU memory consumption during univariate forecasting under $ETTh_1$.

Methods	Metric	$ETTh_1$ (Univariate) Batch_size = 16			
		96	192	336	720
FEDformer	GPU Memory/MB	6758	6844	7216	14 025
	Train time per epoch/s	217.350	215.727	280.154	598.435
	Inference Time/s	27.500	28.483	30.131	41.691
GBT + FEDformer	GPU Memory/MB	***2922***	***3528***	***3992***	***5341***
	Train time per epoch/s	***21.077 + 7.344***	***21.037 + 9.233***	***20.981 + 12.885***	***20.642 + 26.844***
	Inference Time/s	***1.400 + 0.367***	***1.442 + 0.560***	***1.420 + 1.010***	***1.308 + 3.103***
Autoformer	GPU Memory/MB	3358	3920	4560	5938
	Train time per epoch/s	46.326	50.998	56.738	76.022
	Inference Time/s	24.229	25.214	26.755	30.282
GBT + Autoformer	GPU Memory/MB	***3186***	***3520***	***3992***	***5342***
	Train time per epoch/s	***13.271 + 7.599***	***13.496 + 9.287***	***13.668 + 12.993***	***13.922 + 27.340***
	Inference Time/s	***1.939 + 0.277***	***1.926 + 0.583***	***1.794 + 1.010***	***1.691 + 3.031***

Table 12

Means/Stds of MSEs under $ETTm_2$.

Methods	Metric	96		192		336		720	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std
FEDformer	MSE	0.085	0.005	0.135	0.009	0.223	0.035	0.306	0.015
	MAE	0.224	0.006	0.285	0.009	0.366	0.028	0.439	0.012
GBT + FEDformer	MSE	***0.069***	***0.001***	***0.105***	***0.001***	***0.139***	***0.002***	***0.177***	***0.004***
	MAE	***0.189***	***0.001***	***0.240***	***0.002***	***0.283***	***0.002***	***0.326***	***0.008***
Autoformer	MSE	0.213	0.059	0.348	0.227	0.435	0.337	0.44	0.119
	MAE	0.355	0.053	0.444	0.144	0.522	0.223	0.547	0.083
GBT + Autoformer	MSE	***0.080***	***0.002***	***0.113***	***0.002***	***0.138***	***0.001***	***0.185***	***0.002***
	MAE	***0.214***	***0.003***	***0.255***	***0.002***	***0.288***	***0.003***	***0.334***	***0.002***

shows, Informer+GBT and Informer+RevIN both outperform Informer. The general MSE of Informer reduces 31.4% after RevIN is employed. However, it reduces 57.0% after we employ GBT, which improves more. Meanwhile, it could be observed from Table 10 that Informer+GBT and Informer+Non-stationary Transformer both surpass Informer. Though the general MSE of Informer reduces 63.1% after combined with Non-stationary Transformer, Informer with GBT obtains MSE reduction in a larger margin, i.e., 70.5%. Therefore, GBT is a better solution for TSFTs when handling non-stationary time series.

5.8. Complexity analysis

We choose FEDformer and Autoformer, as baselines to examine the computation efficiency of GBT. The training time of each epoch and the whole inference time of the test dataset during univariate forecasting under $ETTh_1$ are shown in Table 11. The batch size is set to 16. The training/inference time and GPU memory consumption of models combined with GBT are reported by two stages respectively. ***The results which outperform the corresponding ones are highlighted in bold and italic.*** Apparently, the training time/inference time/GPU memory consumption of FED-

former and Autoformer are decreased by 88.2%/92.0%/53.0% and 52.3%/88.7%/9.5% separately after combined with GBT. It means that GBT could tremendously reduce time and space complexity of FEDformer/Autoformer.

5.9. Robustness analysis

Here we define that a model is more robust only when its forecasting result is less influenced by weight initialization. According to former analysis, TSFTs are easier to suffer from the overfitting when handling non-stationary time series, i.e., they are less robust. So, we conduct this sub-experiment to check whether GBT could ameliorate this situation under univariate forecasting condition of $ETTm_2$. The prediction length is chosen within {96, 192, 336, 720} and each experiment is conducted for 20 times while means and standard deviations (stds) of their prediction MSEs are shown in Table 12. ***The results which outperform the corresponding ones are highlighted in bold and italic.*** As Table 12 shows, MSEs of FEDformer/Autoformer own large stds/means. It means that these baselines are easy to be affected by random weight initialization. In contrast, stds/means of them drop sharply and are more steady after combined with GBT. It shows that

GBT could enhance the robustness of TSFTs. Meanwhile, it mediately illustrates the seriousness of zero-initialization problem and necessity to solve it.

6. Conclusion

In this paper, we point out and analyze the over-fitting problem of time series forecasting Transformer caused by zero-initialization of decoder inputs especially when handling non-stationary time series. We propose GBT as a feasible and efficient solution. The two-stage framework of GBT decouples *Auto-Regression* and *Self-Regression* components of TSFTs to handle the problem brought by different statistical properties of input/prediction windows and zero-initialization of decoder inputs. We also propose the *E*rror *S*core *M*odification module to deal with the '*Good Beginning*' of the second stage in a more appropriate way. Extensive experiments on real-world datasets illustrate the promising forecasting capability of GBT. Furthermore, GBT is adaptive enough to easily couple with most of state-of-the-art forecasting models, especially TSFTs.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Yangzhu Wang reports financial support was provided by National Natural Science Foundation of China. Yangzhu Wang reports financial support was provided by National Key Research and Development Program of China.

Data availability

Data will be made available on request

Acknowledgments

Appendix A. Zero-initialization considering position embedding

We analyze zero-initialization of prediction elements in decoders ignoring position embedding and bias in the main text to prove that start token could not help initialize prediction elements. Even if considering position embedding of TSFTs, this will not change this condition and we prove it as below.

Here we check the first masked self-attention module in decoders again as Eq. (A.1). $Q_i = Q_{i1} + Q_{i2}, K_i = K_{i1} + K_{i2}(i = 1, 2)$, where Q_{i1}/K_{i1} denote the latent representations of {start token, prediction elements} in query/key while Q_{i2}/K_{i2} are the latent representations of position embedding in them. Similarity, Q_{21}/K_{21} will be zero tensors. Then we get query-key matching matrix as Eq. (A.2). Though zero tensors do not exist in query-key matching matrix, models obtain representations just through start token and position embedding. Therefore, models could only learn position differences of prediction elements but could not acquire the information of statistical properties of prediction elements by using start token, i.e., start token has limited effects on inference process of prediction elements when handling non-stationary time series.

$$Attn = (Q, K, V) = \frac{QK^\top}{\sqrt{d}}V, \quad Q = [Q_1, Q_2]^\top, \quad K = [K_1, K_2]^\top \quad (A.1)$$

$$
\begin{aligned}
QK^\top &= \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} \times \begin{bmatrix} K_1 \\ K_2 \end{bmatrix}^\top = \begin{bmatrix} Q_{11} + Q_{12} \\ Q_{21} + Q_{22} \end{bmatrix} \times \begin{bmatrix} K_{11} + K_{12} \\ K_{21} + K_{22} \end{bmatrix}^\top \\
&= \begin{bmatrix} Q_1 \\ Q_{22} \end{bmatrix} \times \begin{bmatrix} K_1 \\ K_{22} \end{bmatrix}^\top = \begin{bmatrix} Q_1 K_1^\top & Q_1 K_{22}^\top \\ Q_{22} K_1^\top & Q_{22} K_{22}^\top \end{bmatrix}
\end{aligned}
\quad (A.2)
$$

Appendix B. Brief introductions to baselines

Informer (Zhou et al., 2021): Time series forecasting Transformer (TSFT) with ProbSparse attention, distilling operation and one-forward decoder.

Autoformer (Wu et al., 2021): TSFT with Auto-Correlation attention and deep season-trend input sequence decomposition.

FEDformer (Zhou et al., 2022b): TSFT with frequency enhanced decomposition based on both Fourier and Wavelet form.

Pyraformer (Liu et al., 2022b): TSFT with pyramid architecture and low-complexity attention.

ETSformer (Woo et al., 2022b): TSFT with Exponential Smoothing (ES) and season-level-growth decomposition.

Airformer (Liang et al., 2023): TSFT with dartboard spatial attention, causal temporal attention and variational autoencoder (VAE).

TS2Vec (Yue et al., 2022): Self-supervised forecasting method with dilated CNN to extract universal feature maps.

SCINet (LIU et al., 2022): Temporal convolution network with binary tree architecture for efficient forecasting.

N-BEATS (Oreshkin et al., 2020): Simple forecasting method based on Linear Projection and season-trend decomposition.

N-HiTS (Challu et al., 2022): Simple forecasting method based on Linear Projection and temporal interpolation method.

DLinear (Zeng et al., 2022): Simple forecasting method based on Linear Projection and moving average method.

FiLM (Zhou et al., 2022a): Frequency enhanced forecasting model with linear projection layers and Legendre Polynomials projection for reconstruction.

### Appendix C. Combination of GBT with TSFTs

Here we introduce combinations of GBT with TSFTs and other SOTA time series forecasting models. As we mainly perform experiments on FEDformer, ETSformer and SCINet combined with GBT within Section 5.6 in the main text, we present combinations of GBT with these three models. Combinations of GBT with other TSFTs or SOTA time series forecasting models are similar with them.

Following the seasonal-trend decomposition of FEDformer, the output of *Auto-Regression* stage is also composed of season and trend components when GBT is combined with FEDformer as Fig. C.1 shows. Trend and season components are both decomposed by MOE modules. In virtue of different latent dimensions, season/trend representations are projected through a FC/linear projection layer to obtain outputs of the *Auto-Regression* stage. During *Self-Regression* stage, the '*Good Beginning*' is composed of season/trend components and we replace decoders of FEDformer with *Self-Regression* Blocks used in GBT-Vanilla.

Combinations of GBT with ETSformer/SCINet are different in that these two baselines do not own any *Self-Regression* module like masked self-attention mechanism. Therefore, whole networks of ETSformer/SCINet could be directly treated as networks in *Auto-Regression* stage and we employ *Self-Regression* Blocks used in GBT-Vanilla after their networks to combine them with GBT. Their architectures are shown in Fig. C.2.

If models (Wu et al., 2021; Zhou et al., 2021) own *Self-Regression* modules, e.g., masked self-attention mechanism, we choose the former format to combine them with GBT like combination of GBT with FEDformer. Otherwise, we choose another
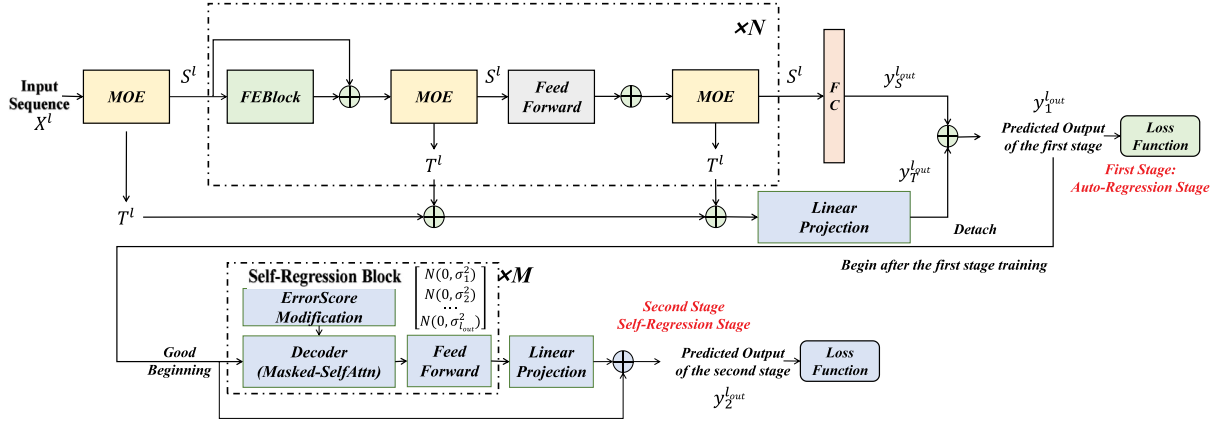
Fig. C.1. The architecture of FEDformer+GBT. The upper half is the *Auto-Regression* stage including *N* encoders of FEDformer and the lower half is the *Self-Regression* stage including *M Self-Regression* Blocks. MOE are used to decompose season and trend component from the input data and FEBlock are used to perform representation learning in frequency domain.
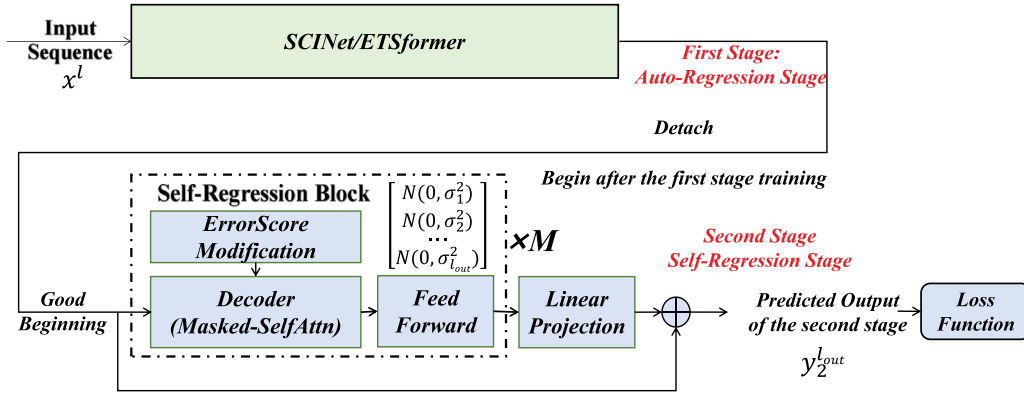
Fig. C.2. The architecture of ETSformer+GBT/SCINet+GBT. The upper half is the *Auto-Regression* stage including networks of ETSformer/SCINet and the lower half is the *Self-Regression* stage including *M Self-Regression* Blocks.

format to combine them with GBT like combinations of GBT with ETSformer/SCINet.

Appendix D. Supplementary experiment

D.1. Supplementary experiment on weather dataset

Here we extra conduct multivariate/univariate forecasting experiments under weather dataset for more comprehensive comparison as Table D.1 shows. It is obvious that GBT-Vanilla outperforms other baselines in most of situations. N-HiTS surpasses GBT-Vanilla during univariate forecasting when input length is 96. However, when compared with FEDformer/Pyraformer/ETSformer/SCINet/TS2Vec/DLinear/N-HiTS, GBT-Vanilla yields 18.9%/16.3%/22.5%/ 11.3%/4.9%/16.1%/11.9% relative MSE reduction during multivariate forecasting and 26.6%/23.4%/31.8%/20.0%/ 11.1%/21.0%/12.3% relative MSE reduction during univariate forecasting in general.

D.2. Supplementary experiment on ILI dataset

Though non-stationarity of time series in ILI is not distinct according to Section 5.4, we additionally combine GBT with Autoformer/N-BEATS, which are separately frequence-enhanced

TSFT/simple model, during multivariate/univariate forecasting under ILI dataset to show that GBT is still useful in relatively stationary forecasting conditions if combined with appropriate methods. Results are shown in Table D.2. In this experiment, input length is set to 36 and prediction length group is {24, 36, 48, 60} according to Woo et al. (2022b), Wu et al. (2021) and Zhou et al. (2022b). **The results which outperform the corresponding original ones are highlighted in bold and italic**. It could be observed from Table D.2 that forecasting performances of N-BEATS and Autoformer are improved in all of situations. N-BEATS/Autoformer separately obtain 10.0%/5.5% relative MSE reduction during multivariate forecasting and 32.0%/16.6% relative MSE reduction during univariate forecasting in general.

D.3. Comprehensive comparison with dlinear under various settings

We conduct a comprehensive comparison of GBT-Vanilla and DLinear-I, DLinear-S with four ETT datasets under multivariate forecasting conditions. DLinear (Zeng et al., 2022) empirically shows that its performance can be better than time series forecasting Transformers with different settings from the commonly agreed-upon one chosen by Woo et al. (2022b), Wu et al. (2021) and Zhou et al. (2022b). This experiment is intended for showing that GBT still excels in handling these forecasting conditions, unlike other TSFTs. They are experimented with different input sequence lengths (in {96, 336, 720}) and different learning rates

Table D.1
Forecasting results under weather.

Methods	Metrics	Weather (Multivariate)				Weather (Univariate)			
		96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	**0.434**	**0.481**	**0.514**	**0.523**	<u>0.188</u>	**0.221**	**0.239**	**0.218**
	MAE	**0.466**	**0.506**	**0.527**	**0.532**	<u>0.318</u>	**0.348**	**0.372**	**0.349**
FEDformer	MSE	0.531	0.601	0.646	0.631	0.236	0.289	0.332	0.335
	MAE	0.525	0.564	0.618	0.597	0.358	0.407	0.431	0.437
Pyraformer	MSE	0.540	0.575	0.593	0.623	0.213	0.262	0.303	0.398
	MAE	0.546	0.567	0.578	0.599	0.342	0.383	0.415	0.483
ETSformer	MSE	0.538	0.615	0.655	0.719	0.243	0.296	0.339	0.432
	MAE	0.521	0.566	0.589	0.624	0.363	0.400	0.430	0.492
SCINet	MSE	0.489	0.526	0.572	0.617	0.213	0.255	0.287	0.352
	MAE	0.495	0.524	0.562	0.586	0.341	0.375	0.399	0.449
TS2Vec	MSE	<u>0.450</u>	<u>0.505</u>	<u>0.532</u>	<u>0.566</u>	0.199	0.240	0.262	<u>0.281</u>
	MAE	<u>0.472</u>	<u>0.515</u>	<u>0.533</u>	<u>0.557</u>	0.323	0.361	0.384	<u>0.405</u>
DLinear	MSE	0.514	0.572	0.597	0.646	0.207	0.257	0.293	0.378
	MAE	0.514	0.553	0.570	0.603	0.336	0.376	0.402	0.470
N-HiTS	MSE	0.488	0.539	0.565	0.628	**0.183**	<u>0.227</u>	<u>0.265</u>	0.359
	MAE	0.496	0.536	0.555	0.596	**0.307**	<u>0.352</u>	<u>0.384</u>	0.462

Table D.2
Forecasting results under ILI.

Methods	Metrics	ILI (Multivariate)				ILI (Univariate)			
		24	36	48	60	24	36	48	60
N-BEATS	MSE	5.294	5.435	6.145	5.976	2.010	2.705	2.059	2.657
	MAE	1.582	1.684	1.797	1.780	1.039	1.259	1.130	1.243
GBT + N-BEATS	MSE	**5.060**	**4.807**	**5.614**	**5.050**	**1.515**	**1.820**	**1.531**	**1.464**
	MAE	**1.523**	**1.539**	**1.681**	**1.576**	**0.908**	**1.007**	**0.954**	**0.946**
Autoformer	MSE	4.971	3.971	4.060	3.951	1.469	0.953	1.085	1.320
	MAE	1.596	1.414	1.382	1.399	1.015	0.819	0.885	0.966
GBT + Autoformer	MSE	**4.351**	**3.857**	**3.935**	**3.817**	**1.087**	**0.845**	**0.918**	**1.141**
	MAE	**1.465**	**1.362**	**1.334**	**1.336**	**0.844**	**0.776**	**0.811**	**0.897**

(in {0.001, 0.0005, 0.0001}). Ultimately, there are $3^2 = 9$ settings for each forecasting condition. Only MSE is used as the evaluation metrics, otherwise the data would be exceedingly tremendous. The results are shown in Table D.3. The average and best results of the forecasting conditions with different prediction lengths and datasets for different baselines are also presented.

Although DLinear own two versions, either the best or the average performances of GBT-Vanilla are unequivocally better than those of DLinear in the major forecasting conditions (Best performances: 14/16; Average performances: 15/16), demonstrating that the GBT architecture is more accurate and general. These phenomena justifiably verify that Transformer with proper usage literally pertains to time series forecasting, which is opposed to the opinion of DLinear.

D.4. Visualization of error accumulation

We visualize the forecasting error for each time step in Fig. D.1 to compare extents of error accumulation among TSFTs with/without GBT. FEDformer and Autoformer are chosen as baselines and are experimented under the univariate forecasting of ETTm$_2$. We take the longest prediction length within main results, i.e., 720. The error at the t-th time step is computed as $MSE - t$ which is the MSE of t-th time step among all prediction windows. Fig. D.1 illustrates that the $MSE - t$ of GBT+FEDformer/GBT+Autoformer is much lower than $MSE - t$ of FEDformer/Autoformer. Specially, $MSE - 1$ of Autoformer is even bigger than the $MSE - 720$ of GBT+Autoformer. This demonstrates that GBT can credibly give TSFT a 'Good Beginning'. Moreover, error accumulations of baselines are slower and more steady after combined with GBT while error accumulations of initial
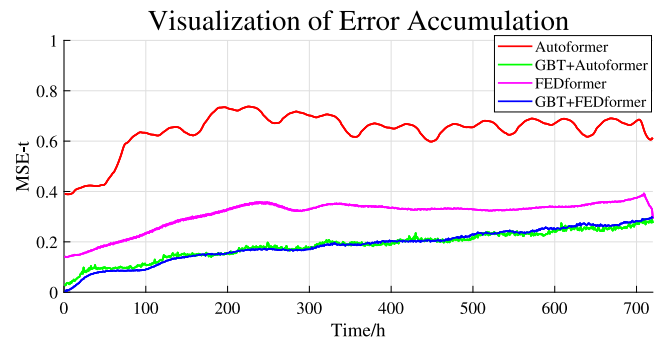
Fig. D.1. Forecasting error for each time step. We visualize the $MSE - t$ of baselines with/without GBT under univariate forecasting of ETTm$_2$. The prediction length is 720.

Autoformer and FEDformer are periodic or turbulent at the beginning/ending. These phenomena demonstrate that GBT could greatly mitigate error accumulation of TSFT.

D.5. Showcases of GBT with TSFTs

To more vividly illustrate how GBT helps alleviate the overfitting problem of TSFTs, the specific prediction curves, originating from ETTm$_2$, of several TSFTs with/without GBT are sketched in Fig. D.2. These forecasting conditions all own distribution shifts appearing between the input sequences and the prediction sequences or within the input sequences. It can be observed that these time series forecasting Transformers are prone to the overfitting problem, barring combined with GBT. When dramatical

Table D.3

Forecasting results of comprehensive comparison. All numbers are MSE results.

Input length	Learning rate	Methods	ETTh₁ 96	192	336	720	ETTh₂ 96	192	336	720	ETTm₁ 96	192	336	720	ETTm₂ 96	192	336	720
96	1e−4	GBT-Vanilla	0.398	0.448	0.497	0.538	0.328	0.468	0.561	0.691	0.329	0.366	0.413	0.454	0.189	0.249	0.324	0.395
		DLinear-I	0.441	0.493	0.534	0.553	0.651	0.894	1.081	1.291	0.337	0.373	0.405	0.467	0.254	0.432	0.648	0.986
		DLinear-S	0.431	0.474	0.518	0.560	0.381	0.508	0.621	0.856	0.350	0.387	0.417	0.477	0.199	0.287	0.387	0.544
	5e−4	GBT-Vanilla	0.380	0.422	0.461	0.495	0.304	0.389	0.446	0.497	0.327	0.356	0.405	0.446	0.183	0.266	0.340	0.450
		DLinear-I	0.395	0.450	0.496	0.511	0.567	0.801	1.010	1.196	0.332	0.370	0.401	0.461	0.272	0.388	0.674	0.935
		DLinear-S	0.392	0.443	0.491	0.519	0.347	0.470	0.591	0.791	0.346	0.383	0.417	0.475	0.194	0.284	0.376	0.506
	1e−3	GBT-Vanilla	0.390	0.429	0.468	0.524	0.322	0.410	0.436	0.535	0.337	0.370	0.407	0.447	0.185	0.271	0.356	0.491
		DLinear-I	0.396	0.453	0.497	0.514	0.564	0.780	0.958	1.295	0.334	0.372	0.399	0.467	0.249	0.406	0.543	0.957
		DLinear-S	0.392	0.444	0.485	0.520	0.332	0.454	0.570	0.800	0.346	0.383	0.416	0.477	0.188	0.264	0.381	0.524
336	1e−4	GBT-Vanilla	0.395	0.443	0.476	0.502	0.300	0.360	0.406	0.532	0.288	0.323	0.367	0.415	0.175	0.229	0.286	0.403
		DLinear-I	0.419	0.454	0.482	0.512	0.459	0.618	0.749	1.070	0.293	0.332	0.371	0.438	0.198	0.316	0.528	0.803
		DLinear-S	0.398	0.439	0.463	0.504	0.326	0.414	0.509	0.771	0.303	0.339	0.377	0.432	0.175	0.236	0.321	0.448
	5e−4	GBT-Vanilla	0.396	0.436	0.460	0.490	0.299	0.366	0.408	0.568	0.292	0.325	0.368	0.416	0.170	0.223	0.280	0.399
		DLinear-I	0.388	0.436	0.461	0.503	0.430	0.574	0.697	0.995	0.292	0.333	0.373	0.432	0.215	0.299	0.547	0.790
		DLinear-S	0.388	0.423	0.451	0.495	0.301	0.399	0.469	0.758	0.305	0.338	0.375	0.431	0.172	0.244	0.298	0.420
	1e−3	GBT-Vanilla	0.402	0.434	0.474	0.534	0.316	0.405	0.444	0.563	0.299	0.337	0.373	0.426	0.179	0.231	0.287	0.420
		DLinear-I	0.382	0.433	0.462	0.516	0.403	0.537	0.677	1.015	0.302	0.344	0.373	0.436	0.216	0.319	0.440	0.739
		DLinear-S	0.387	0.425	0.457	0.492	0.305	0.389	0.458	0.669	0.309	0.337	0.377	0.441	0.174	0.242	0.295	0.409
720	1e−4	GBT-Vanilla	0.399	0.431	0.474	0.535	0.293	0.359	0.426	0.459	0.301	0.320	0.358	0.423	0.191	0.244	0.293	0.392
		DLinear-I	0.417	0.453	0.478	0.537	0.455	0.657	0.794	1.293	0.300	0.341	0.381	0.432	0.201	0.274	0.495	0.651
		DLinear-S	0.391	0.431	0.461	0.506	0.332	0.442	0.547	0.938	0.312	0.341	0.375	0.422	0.175	0.233	0.292	0.418
	5e−4	GBT-Vanilla	0.401	0.447	0.475	0.539	0.340	0.404	0.411	0.498	0.302	0.330	0.359	0.424	0.187	0.241	0.308	0.433
		DLinear-I	0.414	0.458	0.485	0.531	0.420	0.622	0.725	1.334	0.307	0.342	0.387	0.445	0.199	0.275	0.369	0.579
		DLinear-S	0.392	0.426	0.461	0.510	0.309	0.428	0.502	0.824	0.315	0.345	0.373	0.429	0.167	0.227	0.292	0.410
	1e−3	GBT-Vanilla	0.404	0.450	0.475	0.538	0.325	0.417	0.425	0.558	0.278	0.313	0.356	0.417	0.171	0.219	0.272	0.457
		DLinear-I	0.406	0.448	0.511	0.547	0.421	0.549	0.890	1.615	0.324	0.353	0.394	0.469	0.232	0.319	0.446	0.641
		DLinear-S	0.393	0.454	0.475	0.519	0.301	0.371	0.561	0.792	0.322	0.354	0.380	0.427	0.182	0.238	0.299	0.464
Best results		GBT-Vanilla	0.380	0.422	0.460	0.490	0.293	0.359	0.406	0.459	0.278	0.313	0.356	0.415	0.170	0.219	0.272	0.392
		DLinear-I	0.382	0.433	0.461	0.503	0.403	0.537	0.677	0.995	0.292	0.332	0.371	0.432	0.198	0.274	0.369	0.579
		DLinear-S	0.387	0.423	0.451	0.492	0.301	0.371	0.458	0.669	0.303	0.337	0.373	0.422	0.167	0.227	0.292	0.409
Average results		GBT-Vanilla	0.396	0.438	0.473	0.522	0.314	0.398	0.440	0.544	0.306	0.338	0.378	0.430	0.181	0.241	0.305	0.427
		DLinear-I	0.406	0.453	0.490	0.525	0.485	0.670	0.842	1.234	0.313	0.351	0.387	0.450	0.226	0.336	0.521	0.787
		DLinear-S	0.396	0.440	0.474	0.514	0.326	0.431	0.536	0.800	0.323	0.356	0.390	0.446	0.181	0.251	0.327	0.460

Prediction Curves of four TSFTs with/without GBT
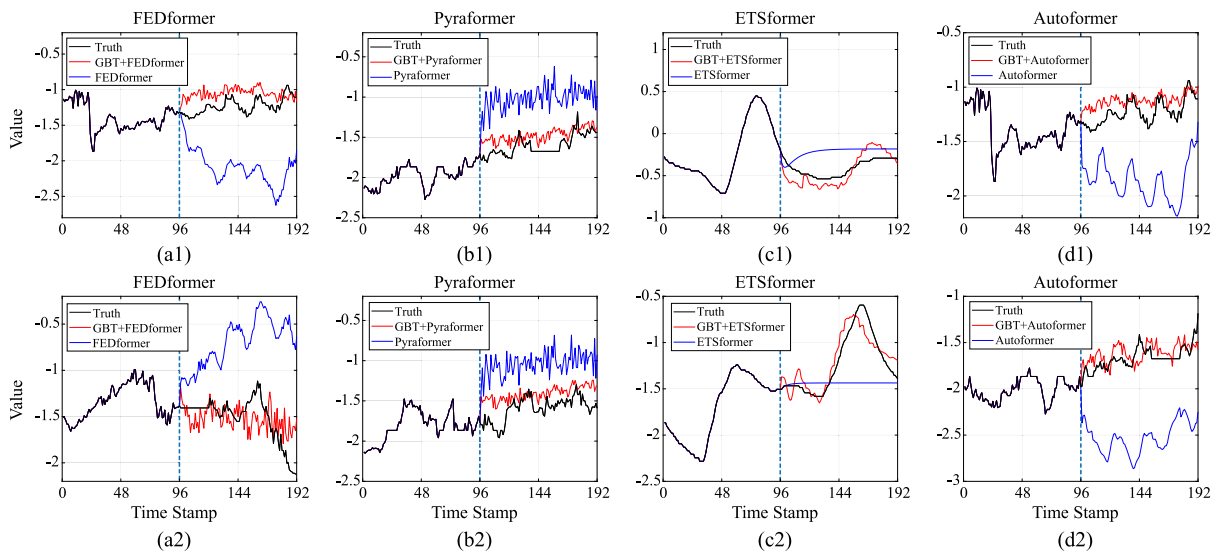
Fig. D.2. Several prediction curves of four TSFTs with/without GBT. Left sequences of blue dashed lines are input sequence while right ones are prediction sequences of TSFTs with/without GBT (red/blue lines) and corresponding truths (black lines).

distribution shifts occur, neither of them can even provide the prediction results whose statistics fit those of the ground truths. However, with the help of GBT, their performances become much better.

D.6. Full results of fedformer and dlinear

The full results of {DLinear-I, DLinear-S} and {FEDformer-f, FEDformer-w} are shown in Tables D.4 and D.5. Results of GBT-Vanilla is also presented for a vivid comparison. Obviously, the

Table D.4
Full results of FEDformer and DLinear during multivariate forecasting.

Methods	Metrics	ETTh$_1$				ETTm$_2$				ECL			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	*0.398*	*0.448*	*0.497*	*0.538*	*0.189*	*0.249*	*0.324*	*0.395*	*0.143*	*0.175*	*0.197*	*0.235*
	MAE	*0.418*	*0.442*	*0.470*	*0.505*	*0.276*	*0.324*	*0.368*	*0.419*	*0.246*	*0.277*	*0.298*	*0.336*
DLinear-I	MSE	0.441	0.492	0.534	*0.553*	0.253	0.435	0.608	0.977	0.220	0.219	0.232	0.268
	MAE	*0.443*	*0.472*	*0.493*	*0.527*	0.334	0.436	0.516	0.658	0.317	0.320	0.333	0.362
DLinear-S	MSE	0.431	0.474	*0.518*	0.560	*0.199*	0.287	0.387	0.544	0.246	0.246	0.260	0.294
	MAE	0.451	0.479	0.508	0.559	0.395	0.363	0.429	0.506	0.345	0.348	0.361	0.388
FEDformer-f	MSE	*0.415*	0.474	0.535	0.680	0.203	*0.269*	*0.325*	*0.421*	0.193	0.201	0.214	0.246
	MAE	0.453	0.493	0.524	0.593	*0.287*	*0.328*	**0.366**	**0.415**	0.308	0.315	0.329	0.355
FEDformer-w	MSE	0.423	*0.448*	0.525	0.691	0.204	0.316	0.359	0.433	*0.183*	*0.195*	*0.212*	**0.231**
	MAE	0.464	0.473	0.522	0.618	0.288	0.363	0.387	0.432	*0.297*	*0.308*	*0.313*	*0.343*

Methods	Metrics	WTH				Traffic				Exchange			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	*0.434*	*0.481*	*0.514*	*0.523*	*0.509*	*0.520*	*0.535*	*0.575*	*0.110*	*0.179*	*0.358*	**0.756**
	MAE	*0.466*	*0.506*	*0.527*	*0.532*	*0.282*	*0.293*	*0.307*	*0.317*	*0.249*	*0.312*	*0.446*	**0.655**
DLinear-I	MSE	*0.490*	*0.552*	*0.583*	0.639	0.794	0.750	0.756	0.799	*0.115*	*0.189*	**0.279**	*0.774*
	MAE	*0.506*	*0.548*	*0.570*	0.606	0.491	0.471	0.474	0.491	*0.258*	*0.335*	**0.412**	*0.693*
DLinear-S	MSE	0.539	0.592	0.610	0.653	0.725	0.665	0.674	0.716	0.219	0.350	0.563	1.076
	MAE	0.522	0.557	0.571	0.600	0.460	0.438	0.441	0.457	0.387	0.481	0.606	0.799
FEDformer-f	MSE	0.509	0.581	0.630	*0.580*	0.587	0.604	0.621	0.626	0.148	0.271	0.460	1.195
	MAE	0.513	0.557	0.636	*0.586*	0.366	0.373	0.383	0.382	0.278	0.380	0.500	0.841
FEDformer-w	MSE	0.553	0.620	0.661	0.681	*0.562*	*0.562*	*0.570*	*0.596*	0.139	0.256	0.426	1.090
	MAE	0.537	0.571	0.599	0.607	*0.349*	*0.346*	*0.323*	*0.368*	0.276	0.369	0.464	0.800

Table D.5
Full results of FEDformer and DLinear during univariate forecasting.

Methods	Metrics	ETTh$_1$				ETTm$_2$				ECL			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	*0.051*	*0.074*	*0.080*	*0.119*	*0.068*	*0.091*	*0.109*	*0.163*	*0.254*	*0.282*	*0.324*	*0.359*
	MAE	*0.173*	*0.206*	*0.221*	*0.276*	*0.194*	*0.229*	*0.257*	*0.316*	*0.363*	*0.386*	*0.417*	*0.444*
DLinear-I	MSE	0.111	0.136	0.166	0.280	0.094	0.130	0.164	0.223	0.411	0.385	0.410	0.447
	MAE	0.258	0.286	0.325	0.453	0.237	0.278	0.316	0.369	0.473	0.455	0.470	0.502
DLinear-S	MSE	0.111	0.136	0.166	0.280	0.094	0.130	0.164	0.223	0.411	0.385	0.410	0.447
	MAE	0.258	0.286	0.325	0.453	0.237	0.278	0.316	0.369	0.473	0.455	0.470	0.502
FEDformer-f	MSE	*0.103*	*0.129*	*0.132*	*0.134*	0.072	*0.102*	*0.130*	*0.178*	*0.253*	*0.282*	*0.346*	*0.422*
	MAE	*0.252*	*0.285*	*0.291*	*0.293*	0.206	*0.245*	*0.279*	*0.325*	*0.370*	*0.386*	*0.431*	*0.484*
FEDformer-w	MSE	0.126	0.144	0.151	0.154	**0.063**	0.110	0.147	0.219	0.262	0.316	0.361	0.448
	MAE	0.279	0.298	0.299	0.311	**0.189**	0.252	0.301	0.368	0.378	0.410	0.445	0.501

Methods	Metrics	WTH				Traffic				Exchange			
		96	192	336	720	96	192	336	720	96	192	336	720
GBT-Vanilla	MSE	*0.188*	*0.221*	*0.239*	*0.218*	*0.133*	*0.140*	*0.138*	*0.174*	*0.100*	*0.186*	*0.408*	*0.925*
	MAE	*0.318*	*0.348*	*0.372*	*0.349*	*0.222*	*0.228*	*0.234*	*0.268*	*0.249*	*0.343*	*0.522*	*0.743*
DLinear-I	MSE	*0.207*	*0.257*	*0.293*	0.378	0.419	0.367	0.366	0.411	0.187	0.315	0.524	*0.940*
	MAE	*0.336*	*0.376*	*0.402*	0.470	0.472	0.434	0.433	0.461	0.361	0.459	0.578	*0.772*
DLinear-S	MSE	0.207	0.257	0.293	0.378	0.419	0.367	0.366	0.411	0.187	0.315	0.524	0.940
	MAE	0.336	0.376	0.402	0.470	0.472	0.434	0.433	0.461	0.361	0.459	0.578	0.772
FEDformer-f	MSE	0.233	0.291	0.318	*0.331*	0.207	0.205	0.219	0.244	0.154	0.286	0.511	1.301
	MAE	0.353	0.406	0.422	*0.432*	0.312	0.312	0.323	0.344	0.304	0.420	0.555	0.879
FEDformer-w	MSE	0.238	0.287	0.345	0.339	*0.170*	*0.173*	*0.178*	*0.187*	*0.131*	*0.277*	*0.426*	1.162
	MAE	0.362	0.407	0.440	0.442	*0.263*	*0.265*	*0.266*	*0.286*	*0.284*	*0.420*	*0.511*	0.832

best results of their two versions are better than the single ones, however, GBT-Vanilla still outperforms them in the major conditions. GBT-Vanilla yields relatively 22.6%/21.5%/15.5%/15.1% MSE reduction during multivariate forecasting and 33.8%/33.8%/23.4%/21.4% MSE reduction during univariate forecasting, when compared with DLinear-I/DLinear-S/FEDformer-f/ FEDformer-w.

References

Ahmad, W., Kazmi, B. M., & Ali, H. (2019). Human activity recognition using multi-head CNN followed by LSTM. In *2019 15th international conference on emerging technologies* (pp. 1–6). http://dx.doi.org/10.1109/ICET48972.2019.8994412.

Alassafi, M. O., Jarrah, M., & Alotaibi, R. (2022). Time series predicting of COVID-19 based on deep learning. *Neurocomputing, 468*, 335–344. http://dx.doi.org/

10.1016/j.neucom.2021.10.035, URL: https://www.sciencedirect.com/science/article/pii/S0925231221015150.

Box, G. E. P., Jenkins, G. M., & MacGregor, J. F. (1974). Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C. Applied Statistics*, 23(2), 158–179. http://dx.doi.org/10.2307/2346997, URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.2307/2346997.

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

Challu, C., Olivares, K. G., Oreshkin, B. N., Garza, F., Mergenthaler-Canseco, M., & Dubrawski, A. W. (2022). N-HiTS: Neural hierarchical interpolation for time series forecasting. arXiv, arXiv:2201.12886.

Cirstea, R.-G., Guo, C., Yang, B., Kieu, T., Dong, X., & Pan, S. (2022). Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting. In L. D. Raedt (Ed.), *Proceedings of the thirty-first international joint conference on artificial intelligence* (pp. 1994–2001). International Joint Conferences on Artificial Intelligence Organization, http://dx.doi.org/10.24963/ijcai.2022/277, Main Track.

Cirstea, R.-G., Yang, B., Guo, C., Kieu, T., & Pan, S. (2022). Towards spatio-temporal aware traffic time series forecasting. In *2022 IEEE 38th international conference on data engineering* (pp. 2900–2913). http://dx.doi.org/10.1109/ICDE53745.2022.00262.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/N19-1423, URL: https://aclanthology.org/N19-1423.

Durbin, J., & Koopman, S. J. (2012). *Time series analysis by state space methods*. Oxford University Press, http://dx.doi.org/10.1093/acprof:oso/9780199641178.001.0001.

He, X., Shi, S., Geng, X., & Xu, L. (2022). Information-aware attention dynamic synergetic network for multivariate time series long-term forecasting. *Neurocomputing*, 500, 143–154. http://dx.doi.org/10.1016/j.neucom.2022.04.124, URL: https://www.sciencedirect.com/science/article/pii/S0925231222005197.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition* (pp. 770–778). http://dx.doi.org/10.1109/CVPR.2016.90.

Hendrycks, D., & Gimpel, K. (2016). Bridging nonlinearities and stochastic regularizers with Gaussian error linear units. CoRR abs/1606.08415, arXiv:1606.08415.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H., & Choo, J. (2022). Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*. URL: https://openreview.net/forum?id=cGDAkQo1C0p.

Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The efficient transformer. In *International conference on learning representations*. URL: https://openreview.net/forum?id=rkgNKkHtvB.

Kumar, N., & Susan, S. (2020). COVID-19 pandemic prediction using time series forecasting models. In *2020 11th international conference on computing, communication and networking technologies* (pp. 1–7). http://dx.doi.org/10.1109/ICCCNT49239.2020.9225319.

Lai, G., Chang, W.-C., Yang, Y., & Liu, H. (2018). Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval* (pp. 95–104). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3209978.3210006.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., & Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems, vol. 32*. Curran Associates, Inc., URL: https://proceedings.neurips.cc/paper/2019/file/6775a0635c302542da2c32aa19d86be0-Paper.pdf.

Liang, Y., Xia, Y., Ke, S., Wang, Y., Wen, Q., Zhang, J., Zheng, Y., & Zimmermann, R. (2023). AirFormer: Predicting nationwide air quality in China with transformers. In *Thirty-seventh AAAI conference on artificial intelligence*.

Liu, Y., Wu, H., Wang, J., & Long, M. (2022). Non-stationary transformers: Exploring the stationarity in time series forecasting. In A. H. Oh, A. Agarwal, D. Belgrave, & K. Cho (Eds.), *Advances in neural information processing systems*. URL: https://openreview.net/forum?id=ucNDIDRNjjv.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., & Dustdar, S. (2022). Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*. URL: https://openreview.net/forum?id=0EXmFzUn5I.

LIU, M., Zeng, A., Chen, M., Xu, Z., LAI, Q., Ma, L., & Xu, Q. (2022). SCINet: Time series modeling and forecasting with sample convolution and interaction. In A. H. Oh, A. Agarwal, D. Belgrave, & K. Cho (Eds.), *Advances in neural information processing systems*. URL: https://openreview.net/forum?id=AyajSjTAzmg.

Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.

Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2020). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International conference on learning representations*. URL: https://openreview.net/forum?id=r1ecqn4YwB.

Qu, L., Li, W., Li, W., Ma, D., & Wang, Y. (2019). Daily long-term traffic flow forecasting based on a deep neural network. *Expert Systems with Applications*, 121, 304–312. http://dx.doi.org/10.1016/j.eswa.2018.12.031, URL: https://www.sciencedirect.com/science/article/pii/S0957417418308017.

Salimans, T., & Kingma, D. P. (2016). Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in neural information processing systems, vol. 29*. Curran Associates, Inc., URL: https://proceedings.neurips.cc/paper/2016/file/ed265bc903a5a097f61d3ec064d96d2e-Paper.pdf.

Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3), 1181–1191. http://dx.doi.org/10.1016/j.ijforecast.2019.07.001, URL: https://www.sciencedirect.com/science/article/pii/S0169207019301888.

Shen, L., Wei, Y., & Wang, Y. (2022). Respecting time series properties makes deep time series forecasting perfect. arXiv abs/2207.10941.

Shuvo, M. A. R., Zubair, M., Purnota, A. T., Hossain, S., & Hossain, M. I. (2021). Traffic forecasting using time-series analysis. In *2021 6th international conference on inventive computation technologies* (pp. 269–274). http://dx.doi.org/10.1109/ICICT50816.2021.9358682.

Stádník, B., Raudeliuniene, J., & Davidavičienė, V. (2016). Fourier analysis for stock price forecasting: Assumption and evidence. *Journal of Business Economics and Management*, 17, 365–380. http://dx.doi.org/10.3846/16111699.2016.1184180.

Taieb, S. B., & Atiya, A. F. (2016). A bias and variance analysis for multistep-ahead time series forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, 27(1), 62–76. http://dx.doi.org/10.1109/TNNLS.2015.2411629.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems, vol. 30*. Curran Associates, Inc., URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, D., Shiralkar, P., Lockard, C., Huang, B., Dong, X. L., & Jiang, M. (2021). TCN: Table convolutional network for web table interpretation. In *Proceedings of the web conference 2021* (pp. 4020–4032). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3442381.3450090.

Weerakody, P. B., Wong, K. W., Wang, G., & Ela, W. (2021). A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing*, 441, 161–178. http://dx.doi.org/10.1016/j.neucom.2021.02.046, URL: https://www.sciencedirect.com/science/article/pii/S0925231221003003.

Wibawa, A. P., Utama, A. B. P., Elmunsyah, H., Pujianto, U., Dwiyanto, F. A., & Hernandez, L. (2022). Time-series analysis with smoothed convolutional neural network. *Journal of Big Data*, 9(1), 44. http://dx.doi.org/10.1186/s40537-022-00599-y.

Woo, G., Liu, C., Sahoo, D., Kumar, A., & Hoi, S. (2022). CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International conference on learning representations*. URL: https://openreview.net/forum?id=PilZY3omXV2.

Woo, G., Liu, C., Sahoo, D., Kumar, A., & Hoi, S. C. H. (2022). ETSformer: Exponential smoothing transformers for time-series forecasting. arXiv abs/2202.01381.

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., & Long, M. (2022). TimesNet: Temporal 2D-variation modeling for general time series analysis. arXiv abs/2210.02186.

Wu, X., Huang, C., Zhang, C., & Chawla, N. V. (2020). Hierarchically structured transformer networks for fine-grained spatial event forecasting. In *Proceedings of the web conference 2020* (pp. 2320–2330). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3366423.3380296.

Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems, vol. 34* (pp. 22419–22430). Curran Associates, Inc., URL: https://proceedings.neurips.cc/paper/2021/file/bcc0d400288793e8bdcd7c19a8ac0c2b-Paper.pdf.

Xu, J., Wu, H., Wang, J., & Long, M. (2022). Anomaly transformer: Time series anomaly detection with association discrepancy. In *International conference on learning representations*. URL: https://openreview.net/forum?id=LzQQ89U1qm_.

Yang, J., Li, C., Zhang, P., Dai, X., Xiao, B., Yuan, L., & Gao, J. (2021). Focal attention for long-range interactions in vision transformers. In A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing ssystems*. URL: https://openreview.net/forum?id=2zCRcTafea.

Yue, Z., Wang, Y., Duan, J., Yang, T., Huang, C., Tong, Y., & Xu, B. (2022). TS2Vec: Towards universal representation of time series. In *AAAI*. http://dx.doi.org/10.1609/aaai.v36i8.20881.

Zeng, A., Chen, M.-H., Zhang, L., & Xu, Q. (2022). Are transformers effective for time series forecasting? arXiv abs/2205.13504.

Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., Zheng, S., & Li, J. (2022). Less is more: Fast multivariate time series forecasting with light sampling-oriented MLP structures. arXiv abs/2207.01186.

Zhao, Y., Wang, Y., Liu, J., Xia, H., Xu, Z., Hong, Q., Zhou, Z., & Petzold, L. (2021). Empirical quantitative analysis of COVID-19 forecasting models. In *2021 international conference on data mining workshops* (pp. 517–526). http://dx.doi.org/10.1109/ICDMW53433.2021.00069.

Zhou, T., Ma, Z., wang, x., Wen, Q., Sun, L., Yao, T., Yin, W., & Jin, R. (2022). FiLM: Frequency improved Legendre memory model for long-term time series forecasting. In A. H. Oh, A. Agarwal, D. Belgrave, & K. Cho (Eds.), *Advances in neural information processing systems*. URL: https://openreview.net/forum?id=zTQdHSQUQWc.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022). FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, & S. Sabato (Eds.), *Proceedings of machine learning research*: *vol. 162, Proceedings of the 39th international conference on machine learning* (pp. 27268–27286). PMLR, URL: https://proceedings.mlr.press/v162/zhou22g.html.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The thirty-fifth AAAI conference on artificial intelligence, AAAI 2021, virtual conference, vol. 35, no. 12* (pp. 11106–11115). AAAI Press, http://dx.doi.org/10.1609/aaai.v35i12.17325.
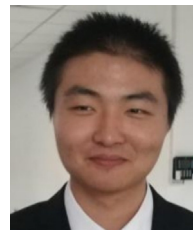
Li Shen received his B.S. degree in Navigation and Control from Beihang University in China at the School of Automation Science and Electrical Engineering. He is currently completing a Doctor degree in Electronic Information Engineering at the Beihang Institute of Unmanned System. His research interests include Computer Vision and Time Series Forecasting.

Yuning Wei received her B.S. degree in Agricultural Electrification from Nanjing Agricultural University in China at the School of Engineering. She is currently completing a MA.Eng in Electronic Information Engineering at the Beihang Institute of Unmanned System. Her research interests include Deep Learning and Time Series Forecasting.

Yangzhu Wang received his B.S, MA.Eng and Ph.D degrees in Measurement and Control Technology and Instrumentation Program from Beihang University in China at the Beihang Institute of Unmanned System. He is currently a researcher fellow of the Flying College of Beihang University and Beihang Institute of Unmanned System. His research interests include Computer Vision, Unmanned System and Measurement and Control Technology.