

# InjectTST: A Transformer Method of Injecting Global Information into Independent Channels for Long Time Series Forecasting

Ce Chi<sup>1</sup>, Xing Wang<sup>1</sup>, Kexin Yang<sup>1</sup>, Zhiyan Song<sup>1</sup>, Di Jin<sup>1</sup>, Lin Zhu<sup>1</sup>,  
Chao Deng<sup>1</sup> and Junlan Feng<sup>1</sup>

<sup>1</sup>China Mobile Research Institute

{chice, wangxing, yangkexin, songzhiyan, jindi, zhulinyj, dengchao, fengjunlan}@chinamobile.com

## Abstract

Transformer has become one of the most popular architectures for multivariate time series (MTS) forecasting. Recent Transformer-based MTS models generally prefer channel-independent structures with the observation that channel independence can alleviate noise and distribution drift issues, leading to more robustness. Nevertheless, it is essential to note that channel dependency remains an inherent characteristic of MTS, carrying valuable information. Designing a model that incorporates merits of both channel-independent and channel-mixing structures is a key to further improvement of MTS forecasting, which poses a challenging conundrum. To address the problem, an injection method for global information into channel-independent Transformer, InjectTST, is proposed in this paper. Instead of designing a channel-mixing model directly, we retain the channel-independent backbone and gradually inject global information into individual channels in a selective way. A channel identifier, a global mixing module and a self-contextual attention module are devised in InjectTST. The channel identifier can help Transformer distinguish channels for better representation. The global mixing module produces cross-channel global information. Through the self-contextual attention module, the independent channels can selectively concentrate on useful global information without robustness degradation, and channel mixing is achieved implicitly. Experiments indicate that InjectTST can achieve stable improvement compared with state-of-the-art models.

## 1 Introduction

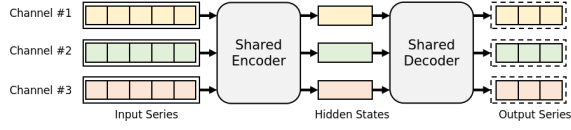
Unprecedented advances have been made in multivariate time series (MTS) forecasting with the rapid growth of deep learning [Benidis *et al.*, 2022], benefiting various regions, such as finance [Patton, 2013], weather [Angryk *et al.*, 2020], traffic [Cai *et al.*, 2020], and networking [Liu *et al.*, 2021]. Recently, Transformer-based MTS models have been widely explored [Wen *et al.*, 2022] and gradually become one of the most

popular architectures for MTS modeling [Cao *et al.*, 2023; Chang *et al.*, 2023; Zhang *et al.*, 2023].

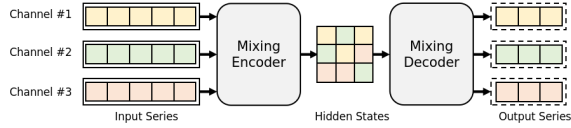
However, recent MTS forecasting models generally prefer a channel-independent structure [Nie *et al.*, 2023; Cao *et al.*, 2023; Zhang *et al.*, 2023; Zhou *et al.*, 2023], as the channel independence usually demonstrates superior performance than channel-mixing structures [Nie *et al.*, 2023; Han *et al.*, 2023]. In detail, channel independence (as shown in Fig. 1(a)) has two merits. (1) **noise mitigation**: channel-independent models can focus on individual channel forecasting without being disturbed by noise from other channels [Nie *et al.*, 2023]. (2) **distribution drift mitigation**: channel independence can alleviate the distribution drift problem of MTS [Han *et al.*, 2023]. In contrast, channel mixing proves less effective in dealing with the channel noise and distribution drift issues, leading to inferior performance. Nevertheless, channel mixing (as shown in Fig 1(b)) possesses some unique advantages. (1) **high information capacity**: channel-mixing models excel in capturing channel dependencies and can bring more information to the forecasting [Wu *et al.*, 2019; Zhang and Yan, 2022]. (2) **channel specificity**: the optimization of multiple channels in channel-mixing models is carried out simultaneously, enabling the model to fully capture the distinct characteristics of each channel. In contrast, as channel independence treats channels with a shared model, the model cannot distinguish the channels and mainly learns the general patterns of multiple channels, leading to the loss of channel specificity and potential impact on forecasting. Hence, designing an effective model with merits of both channel independence and channel mixing, i.e., noise mitigation, distribution drift mitigation, high information capacity and channel specificity, is a key to further enhancing the MTS forecasting performance.

However, designing a model that embodies the merits of both channel independence and channel mixing poses a challenging conundrum. First, from the perspective of channel independence, channel-independent models are inherently contradictory to channel dependencies. Although fine-tuning the shared model for each channel can solve the problem of channel specificity, it comes at the expense of substantial training costs. Second, from the perspective of channel mixing, existing denoising methods and distribution drift-solving methods still struggle to make channel-mixing frameworks as robust as channel independence. How to transfer the strong noise

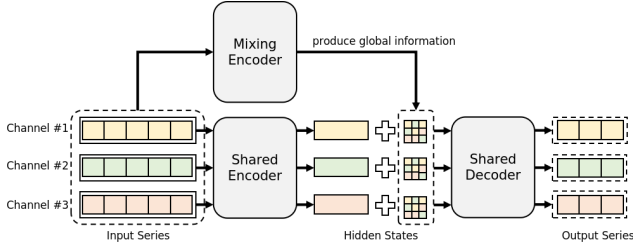
and distribution drift mitigation capabilities of channel independence to channel-mixing frameworks has not been investigated in previous works.



(a) Channel-independent MTS forecasting model framework.



(b) Channel-mixing MTS forecasting model framework.



(c) InjectTST MTS forecasting model framework.

Figure 1: Different types of MTS forecasting frameworks. The decoder can be replaced by a simple prediction head. (a) In a channel-independent framework, the prediction of a channel is irrelevant to other channels. The channels share the same model. (b) In a channel-mixing framework, the channels are mixed for a unified representation, and then the decoder produces the prediction for all channels at the same time. (c) In our proposed InjectTST framework, the channel-independent structure is used as a backbone. Each channel receives additional global information so as to achieve channel-mixing implicitly.

In order to solve these problems, an MTS injection method for global information into individual channels, InjectTST, is proposed in this paper. As shown in Fig. 1(c), in sharp contrast to existing channel-mixing works [Wu *et al.*, 2022; Zhang and Yan, 2022], we avoid to explicitly model the channel dependencies for forecasting. Instead, our solution is retaining the channel-independent structure as a backbone, and global (channel-mixing) information is injected into each channel selectively so as to achieve channel mixing in an implicit way. Each individual channel can selectively receive useful global information and avoid noisy one, so high information capacity and noise mitigation can be compatibly maintained. As channel independence is retained as the backbone, the distribution drift can be alleviated as well. We introduce a channel identifier into InjectTST to solve the channel specificity problem, which is elaborated in the following.

Specifically, as shown in Fig. 2, a channel identifier, a global mixing module and a self-contextual attention (SCA)

module are devised in InjectTST. (1) First, we assign each channel with a trainable embedding, named channel identifier. From the perspective of channels, the channel identifier represents the distinct features of each channel after optimization, which can facilitate the model capability of extracting unique representation of channels. Moreover, the channel identifier serves the crucial function of distinguishing channels for the Transformer in the injection period, which is similar to positional encoding. (2) Second, a global mixing module is designed to produce global information for subsequent injection. Two kinds of global mixing module design are presented in this paper, where a Transformer encoder is used for a high-level global representation. (3) Third, a self-contextual attention module is designed for harmless information injection. In our design, the obtained global information is viewed as context, where the injection is achieved via a modified cross attention design. By selectively concentrating on valuable global information, each channel absorbs the global information with minimal noise disturb. Hence, channel mixing is implicitly achieved while the robustness is not ruined. Experiments based on widely used real-world datasets demonstrate that InjectTST can achieve state-of-the-art performance compared with the latest MTS forecasting methods. More importantly, InjectTST reveals a promising combination solution for MTS modeling, bridging the gap between channel-mixing and channel-independent models.

The contributions of this paper are concluded as follows:

- A MTS modeling framework with merits of both channel mixing and channel independence is proposed. In this framework, channel-independent structures are used as backbones, and channel-mixing information is viewed as context and is injected into individual channels in a selective way.
- We propose InjectTST, an injection MTS forecasting method for global information into channel-independent Transformer models. In InjectTST, a channel identifier is proposed to identify each channel for better representation. Two kinds of global mixing modules are proposed, which can mix channel information effectively. Viewing global information as context, a cross attention-based SCA module is proposed so as to inject valuable global information into individual channels.
- Experimental results show that InjectTST can achieve state-of-the-art in multiple datasets. Extensive analysis indicates that InjectTST is a promising framework for future improvement of both channel-mixing and channel-independent MTS models.

## 2 Related Works

Transformer has shown great potential in MTS forecasting. In Informer [Zhou *et al.*, 2021], the long tail distribution issue in MTS Transformer is observed, and a ProbSparse self-attention mechanism is proposed to improve Transformer efficiency. Autoformer [Wu *et al.*, 2021] utilizes MTS decomposition modules to capture the complicated temporal pattern, where an Auto-Correlation mechanism is proposed. FEDformer [Zhou *et al.*, 2022] utilizes decomposition mod-

ules as global profile extractors and frequency-based Transformer as detail extractors, where a linear-complexity model is proposed for MTS forecasting. In Triformer [Cirstea *et al.*, 2022], the authors propose a triangular structure to optimize the Transformer computation complexity, with which a variable-specific parameter scheme is proposed to improve the accuracy of Transformer models. In Crossformer [Zhang and Yan, 2022], a two-stage attention mechanism is proposed to capture the cross-channel dependency of MTS. However, Crossformer is a pure channel-mixing framework, and thus still faces the risk of robustness loss. In PatchTST [Nie *et al.*, 2023], local semantic information is emphasized and a patching mechanism is proposed. Besides, channel independence is found effective for MTS forecasting. However, channel independence ignores the channel dependency information, so there is still room for improvement. iTransformer [Liu *et al.*, 2023] applies Transformer on inverted dimensions of MTS, where the time series in a channel is viewed as a token. Therefore, iTransformer focuses on capturing the channel dependency of MTS, and also may result in loss of robustness.

Apart from Transformer-based methods, linear models also show comparable performance for MTS forecasting. DLinear [Zeng *et al.*, 2023] applies a simple linear model with decomposition modules and is validated to be effective for MTS forecasting. TSMixer [Ekambaram *et al.*, 2023] introduces the patching mechanism into MLP-based MTS frameworks, with which an online reconciliation approach is proposed to improve the learning capacity for MLP models. In FreTS [Yi *et al.*, 2023], the authors explore a novel combination of MLP architectures and frequency domain for time series forecasting. The frequency domain provides a complete view of global dependencies and can make the model concentrate on key parts of frequency components in time series. However, the contradiction between channel independence and channel mixing remains unresolved in these works.

### 3 Methodology

We consider the MTS forecasting problem. The input is historical  $L$  time steps of MTS  $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_L)$ , where each time step is a vector with dimension of  $M$ , i.e.,  $\mathbf{x}_t = (x_{t,1}, x_{t,2}, \dots, x_{t,M})$ ,  $t = 1, 2, \dots, L$ . The target is to forecast the values in future  $T$  time steps based on the given history  $\mathbf{X}$ , i.e.,  $\mathbf{Y} = (\mathbf{x}_{L+1}, \dots, \mathbf{x}_{L+T})$ . To solve the problem and incorporate merits of both channel mixing and independence, InjectTST is proposed in this paper. The architecture of InjectTST is illustrated in Fig. 2.

#### 3.1 Channel-Independent Backbone

##### Patching and Projection

Patching mechanism is proposed in [Nie *et al.*, 2023], which can improve the local semantic information and reduce calculation complexity. We adopt the patching mechanism in InjectTST. Specifically, in the channel-independent backbone, the input  $\mathbf{X} \in \mathbb{R}^{L \times M}$  passes through the patching module and produces patches  $\mathbf{x}^{patch} \in \mathbb{R}^{M \times PN \times PL}$ , where  $PL$  denotes the length of each patch and  $PN$  denotes the number of patches in each channel.  $PN = \lfloor \frac{L-PL}{S} \rfloor + 2$ , where  $S$  denotes the stride of patching. After that, a linear projection  $\mathbf{W} \in \mathbb{R}^{PL \times D}$  with a learnable positional encoding

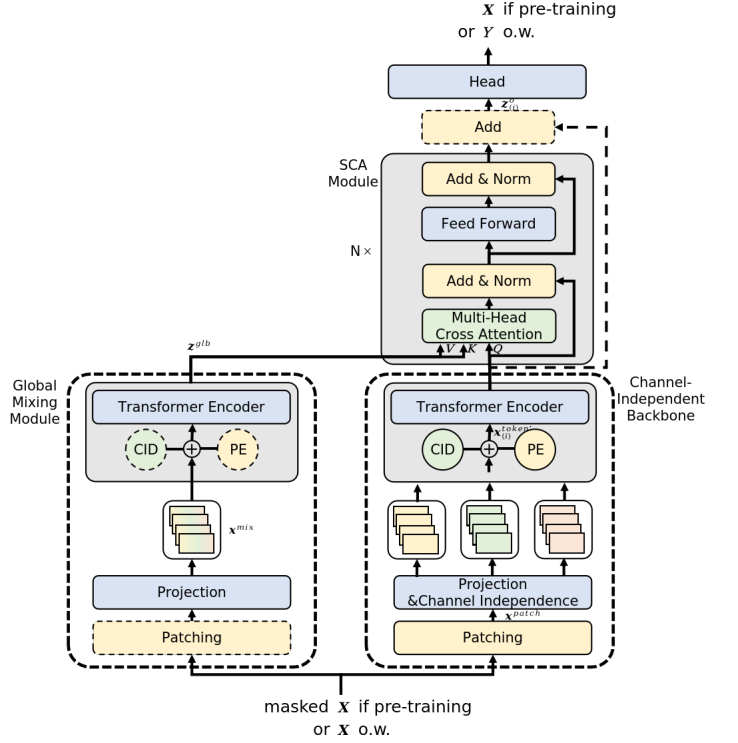


Figure 2: InjectTST architecture. In the channel-independent backbone, the patches of a channel are added with a positional encoding as well as a channel identifier (CID for short). In the global mixing module, the channels are mixed for the global information. Finally, in the self-contextual attention (SCA) module, the global information is injected into each channel via a cross attention design.

$\mathbf{U} \in \mathbb{R}^{PN \times D}$  is used to project patches into initial tokens, where  $D$  denotes the dimension of the Transformer model. The process can be represented as

$$\mathbf{x}^{token} = \mathbf{x}^{patch} \mathbf{W} + \mathbf{U}, \quad (1)$$

where the output  $\mathbf{x}^{token} \in \mathbb{R}^{M \times PN \times D}$ .

##### Channel Identifier

The channel-independent framework treats channels with a shared model. As a result, the model cannot distinguish the channels and mainly learns general patterns of the channels, lacking channel specificity.

To solve the problem, we introduce channel identifier into InjectTST, which is a learnable tensor  $\mathbf{V} \in \mathbb{R}^{M \times D}$ . After the linear projection of patches, the tokens are added with both positional encoding and channel identifier, i.e.,

$$\mathbf{x}^{token'} = \mathbf{x}^{token} + \mathbf{V} = \mathbf{x}^{patch} \mathbf{W} + \mathbf{U} + \mathbf{V}. \quad (2)$$

Finally,  $\mathbf{x}^{token'}$  is input into the Transformer encoder for high-level representation in a channel-independent way:

$$\mathbf{z}_{(i)} = \text{Encoder}^{ci}(\mathbf{x}_{(i)}^{token'}), i = 1, 2, \dots, M. \quad (3)$$

The channel identifier represents the distinct features of each channel, which enables the model to distinguish channels and capture unique representation of channels.

### 3.2 Global Mixing Module

In the channel-mixing route, the input sequence  $\mathbf{X}$  first pass through a global mixing module to produce global information. The major target of InjectTST is to inject global information into each channel, so how to obtain global information is a critical problem. Two kinds of effective global mixing modules are presented in this paper, named CaT (channel as a token) and PaT (patch as a token) respectively.

#### CaT Global Mixing Module

Specifically, as shown in Fig. 3(a), the CaT module directly projects each channel into a token. In detail, a linear project  $\mathbf{W}_{mix} \in \mathbb{R}^{L \times D}$  is applied on the whole channel values:

$$\mathbf{x}^{mix} = \mathbf{X}^T \mathbf{W}_{mix}, \quad (4)$$

where  $\mathbf{x}^{mix} \in \mathbb{R}^{M \times D}$ . Then  $\mathbf{x}^{mix}$  is input into a Transformer encoder with our proposed channel identifier  $\mathbf{V}$  to produce the final global information:

$$\mathbf{z}^{glb} = \text{Encoder}^{mix}(\mathbf{x}^{mix} + \mathbf{V}). \quad (5)$$

#### PaT Global Mixing Module

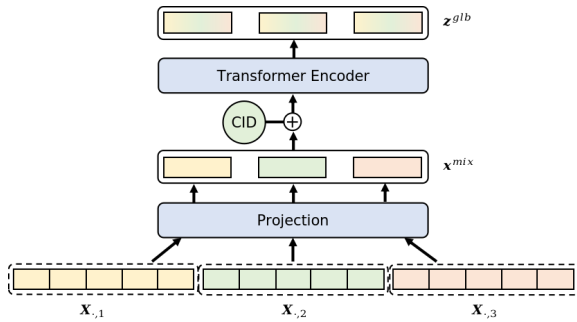
PaT global mixing module is shown in Fig. 3(b). PaT takes patches  $\mathbf{x}^{patch}$  as inputs. It first reshapes the patches into dimensions of  $PN \times (M \cdot PL)$  so that the patches in the same relative position are grouped. Then, a linear projection  $\mathbf{W}_{mix} \in \mathbb{R}^{(M \cdot PL) \times D}$  is applied on the grouped patches, i.e.,

$$\mathbf{x}^{mix} = \mathbf{x}^{patch} \mathbf{W}_{mix}, \quad (6)$$

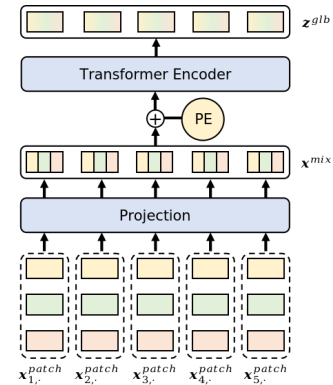
where  $\mathbf{x}^{mix} \in \mathbb{R}^{PN \times D}$ . The linear projection mainly fuses information within the patch level. After that, a Transformer encoder is applied on  $\mathbf{x}^{mix}$  with positional encoding  $\mathbf{U}$  to further fuse information cross patches, and the global information  $\mathbf{z}^{glb}$  is obtained:

$$\mathbf{z}^{glb} = \text{Encoder}^{mix}(\mathbf{x}^{mix} + \mathbf{U}). \quad (7)$$

Experiments illustrate that PaT is more stable while CaT is outstanding in some special datasets. Details are presented in Section 4.4.



(a) CaT global mixing module.



(b) PaT global mixing module.

Figure 3: Two types of design for the global mixing module. (a) CaT global mixing module directly projects each channel into tokens, and utilizes a Transformer encoder for cross-channel global mixing. (b) PaT global mixing module groups the patches in the same position and applies a linear projection for local mixing. Then a Transformer encoder is used for global mixing.

### 3.3 Self-Contextual Attention Module

One of the challenges in our framework is that we must inject global information into each channel with minimal impact on the robustness. In the vanilla Transformer, cross attention makes the target sequence selectively and freely focus on context information from another source according to relevance. With this insight, the cross attention design could be suitable for the MTS injection as well.

To be specific, the global information mixed from channels  $\mathbf{z}^{glb}$  can be viewed as context. We use a cross attention design to inject the global information into each channel, as shown in Fig.2. A residual connection is optional in our design, which is marked with dotted lines in the SCA module in Fig.2. In general cases, the residual connection makes the model slightly unstable. However, as depicted in Section 4.4, the residual connection can achieve outstanding improvement in some special datasets.

Overall, the global information  $\mathbf{z}^{glb}$  is input into the SCA module as keys and values. And the channel information  $\mathbf{z}_{(i)}$  is input as queries:

$$\mathbf{z}_{(i)}^o = \text{SCA}(\mathbf{z}_{(i)}, \mathbf{z}^{glb}, \mathbf{z}^{glb}). \quad (8)$$

After the cross attention, the output  $\mathbf{z}_{(i)}^o$  is improved by global information. Finally, a linear head is appended to produce the prediction. We adopt a self-supervised training method same as [Nie *et al.*, 2023]. In the pre-training stage, the head is a pre-training head, which predicts the masked patches. In the finetuning stage, the head is a prediction head, which predicts the future series.

### 3.4 Self-Supervised Training and Normalization

The training process consists of 3 stages. First, in the pre-training stage, the input time series are randomly masked and the target is to predict the masked parts. Second, in the head-finetuning stage, the pre-training head of InjectTST is replaced by a prediction head and we finetune the prediction head while the rest of the network is frozen. Third, in the finetuning stage, the entire network of InjectTST is finetuned.

Moreover, we adopt a simple normalization introduced in DLinear[Zeng *et al.*, 2023; Han *et al.*, 2023] to further alleviate the distribution drift problem. The method modifies the input by subtracting the last values of the input sequence. After prediction, the last values are added back to the prediction. Therefore, the prediction is the difference from the latest value, and the distribution drift problem can be alleviated.

## 4 Experiments

### 4.1 Experimental Setup

#### Datasets

Multiple widely used real-world MTS datasets are used in our experiments, including (1) Weather, (2) Electricity, (3) Traffic, (4) ETTh1, (5) ETTh2, (6) ETTm1, (7) ETTm2. Please refer to [Wu *et al.*, 2021] for dataset details.

#### Baselines

Various well known latest MTS forecasting baselines are compared, including (1) Transformer-based models: PatchTST [Nie *et al.*, 2023], iTransformer, [Liu *et al.*, 2023], Autoformer [Wu *et al.*, 2021], FEDformer [Zhou *et al.*, 2022]; (2) a MLP-based model: DLinear [Zeng *et al.*, 2023]; (3) a CNN-based model: TimesNet [Wu *et al.*, 2022].

### Implementation Details

Our Experiments are conducted in a server equipped with Intel(R) Xeon(R) Gold 5118 CPU @ 2.30GHz, 376 GB of RAM, and 2 Tesla V100-PCIE-32GB GPUs. The experiment is implemented with PyTorch 1.11.0 [Paszke *et al.*, 2019] based on the code framework of PatchTST [Nie *et al.*, 2023].

The input historical sequence length is set to 512. The patch length is set to 12 and stride is set to 12. The mask ratio of self-supervised learning is set to 50%. There are 3 steps for the training. First, we pre-train InjectTST for 20 epochs. Second, we finetune the prediction head of InjectTST for 10 epochs. Finally, the entire network of InjectTST is finetuned for 100 epochs. Mean Square Error (MSE) and Mean Absolute Error (MAE) are used as evaluation metrics.

#### Model Variants

As different kinds of global mixing modules are designed and the residual connection in the SCA module is optional, InjectTST is a set of models, including InjectTST-CaT, InjectTST-CaT-RC, InjectTST-PaT, InjectTST-PaT-RC, respectively. RC means the model adopts the optional residual connection in SCA. In the main results, InjectTST-PaT is used as the major model, and unless otherwise specified, InjectTST refers to InjectTST-PaT.

Models		InjectTST		PatchTST		iTransformer		DLinear		TimesNet		FEDformer		Autoformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	<b>0.140</b>	<b>0.192</b>	0.144	0.193	0.176	0.216	0.176	0.237	0.170	0.220	0.238	0.314	0.249	0.329
	192	<b>0.184</b>	<b>0.235</b>	0.193	0.239	0.225	0.257	0.220	0.282	0.224	0.263	0.275	0.329	0.325	0.370
	336	<b>0.236</b>	<b>0.278</b>	0.244	0.280	0.281	0.299	0.265	0.319	0.282	0.304	0.339	0.377	0.351	0.391
	720	<b>0.308</b>	<b>0.331</b>	0.318	0.334	0.358	0.350	0.323	0.362	0.357	0.353	0.389	0.409	0.415	0.426
Traffic	96	<b>0.360</b>	<b>0.250</b>	0.376	0.261	0.393	0.268	0.410	0.282	0.589	0.315	0.576	0.359	0.597	0.371
	192	<b>0.387</b>	<b>0.259</b>	0.390	0.267	0.413	0.277	0.423	0.287	0.618	0.324	0.610	0.607	0.382	0.2
	336	<b>0.399</b>	<b>0.265</b>	<b>0.399</b>	0.271	0.425	0.283	0.436	0.296	0.635	0.341	0.608	0.375	0.623	0.387
	720	0.442	0.303	<b>0.436</b>	<b>0.293</b>	0.458	0.300	0.466	0.315	0.664	0.352	0.621	0.375	0.639	0.395
Electricity	96	<b>0.125</b>	<b>0.219</b>	0.126	0.221	0.148	0.239	0.140	0.237	0.168	0.271	0.186	0.302	0.196	0.313
	192	<b>0.145</b>	0.239	<b>0.145</b>	<b>0.238</b>	0.167	0.258	0.153	0.249	0.191	0.293	0.197	0.311	0.211	0.324
	336	<b>0.161</b>	<b>0.256</b>	0.164	<b>0.256</b>	0.179	0.272	0.169	0.267	0.197	0.299	0.213	0.328	0.214	0.327
	720	<b>0.198</b>	0.291	<b>0.198</b>	<b>0.289</b>	0.208	0.297	0.203	0.301	0.262	0.345	0.233	0.344	0.236	0.342
ETTh1	96	0.367	<b>0.394</b>	<b>0.366</b>	0.397	0.393	0.408	0.375	0.399	0.389	0.412	0.376	0.415	0.435	0.446
	192	<b>0.400</b>	<b>0.415</b>	0.401	0.416	0.446	0.440	0.405	0.416	0.440	0.442	0.423	0.446	0.456	0.457
	336	<b>0.432</b>	<b>0.438</b>	0.450	0.456	0.494	0.467	0.439	0.443	0.495	0.471	0.444	0.462	0.486	0.487
	720	<b>0.468</b>	<b>0.482</b>	0.472	0.484	0.517	0.501	0.472	0.490	0.518	0.495	0.469	0.492	0.515	0.517
ETTh2	96	<b>0.275</b>	<b>0.337</b>	0.284	0.343	0.301	0.350	0.289	0.353	0.332	0.370	0.332	0.374	0.332	0.368
	192	<b>0.344</b>	<b>0.385</b>	0.357	0.387	0.380	0.399	0.383	0.418	0.397	0.410	0.407	0.446	0.426	0.434
	336	<b>0.370</b>	<b>0.406</b>	0.377	0.410	0.424	0.432	0.448	0.465	0.453	0.451	0.400	0.447	0.477	0.479
	720	<b>0.398</b>	0.436	0.400	<b>0.435</b>	0.430	0.447	0.605	0.551	0.438	0.450	0.412	0.469	0.453	0.490
ETTh1	96	<b>0.285</b>	<b>0.343</b>	0.289	0.344	0.342	0.377	0.299	0.343	0.335	0.377	0.326	0.390	0.510	0.492
	192	<b>0.323</b>	<b>0.365</b>	0.330	0.369	0.383	0.396	0.335	<b>0.365</b>	0.405	0.411	0.365	0.415	0.514	0.495
	336	0.354	<b>0.383</b>	<b>0.353</b>	0.387	0.418	0.418	0.369	0.386	0.416	0.422	0.392	0.425	0.510	0.492
	720	<b>0.403</b>	<b>0.413</b>	0.407	0.415	0.487	0.457	0.425	0.421	0.479	0.459	0.446	0.458	0.527	0.493
ETTh2	96	<b>0.165</b>	<b>0.254</b>	0.170	0.257	0.186	0.272	0.167	0.260	0.188	0.266	0.180	0.271	0.205	0.293
	192	<b>0.219</b>	<b>0.293</b>	0.221	0.295	0.254	0.314	0.224	0.303	0.263	0.311	0.252	0.318	0.278	0.336
	336	<b>0.268</b>	<b>0.324</b>	0.269	0.325	0.316	0.351	0.281	0.342	0.322	0.349	0.324	0.364	0.343	0.379
	720	<b>0.354</b>	<b>0.382</b>	0.365	0.388	0.414	0.407	0.397	0.421	0.424	0.408	0.410	0.420	0.414	0.419

Table 1: Multivariate time series forecasting results of various models. InjectTST refers to InjectTST-PaT for specification. PatchTST here is the self-supervised version which generally outperforms its supervised one so as to present a strong baseline. Prediction length is set in {96, 192, 336, 720}. The best results are in bold.



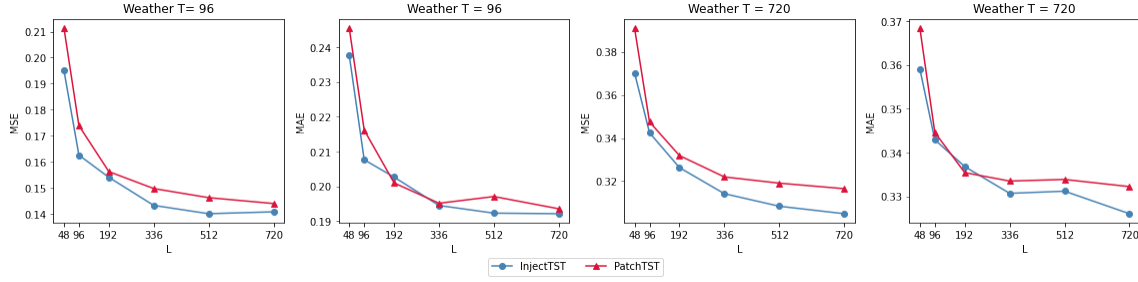


Figure 4: Forecasting performance of InjectTST and PatchTST with varying historical sequence lengths. The historical sequence lengths  $L$  are set in  $\{48, 96, 192, 336, 512, 720\}$ . The prediction lengths  $T$  are selected in  $\{96, 720\}$ .

## 4.2 Main Results

As shown in Table 1, InjectTST achieves SOTA in most of datasets. The improvement in Weather, ETTh1, ETTh2 and ETTm2 is significant. On average, InjectTST achieves 1.63% reduction on MSE and 0.89% on MAE compared with PatchTST, one of the strongest baselines. It indicates that InjectTST can effectively inject global information into individual channels and improve the forecasting performance with very little impact on the robustness of channel-independent structures.

## 4.3 Ablation Study

### Effect of Each Component

The importance of the channel identifier and global information is analyzed in Table. 2. The global mixing and SCA modules are closely related, so we directly present a model without global information, named InjectTST w/o GI. It can be seen that the model performance degrades significantly without global information, indicating the absolute importance of global information in InjectTST.

On the other hand, according to the results of PatchTST and InjectTST w/o GI, where the channel identifier plays a major role, it can be observed that the channel identifier is also effective for MTS forecasting. Although InjectTST w/o CID achieves similar performance to InjectTST, the stability becomes inferior without the channel identifier. In summary, the channel identifier plays a role in making the overall performance more stable and has a slight positive effect on performance improvement.

### Varying Historical Sequence Lengths

The performance of InjectTST with different historical sequence lengths is tested. As shown in Fig. 4, with the benefits of the patching mechanism, the performance of InjectTST can be improved with the increase of input length. Moreover, InjectTST can generally outperform PatchTST with different input lengths, validating the effectiveness of InjectTST.

## 4.4 Analysis

### Different Design of Global Mixing Module

We change the component design of InjectTST to further validate the effectiveness of the InjectTST framework. First, we change the global mixing module from PaT to CaT, as shown in Table. 3. It can be seen that InjectTST-CaT can achieve comparable improvement compared with InjectTST.

Although InjectTST-CaT shows worse stability, the improvement of InjectTST-CaT compared with PatchTST can still validate the effectiveness of the injection framework design.

### Effect of Residual Connection

An optional residual connection in SCA can be added so that the selection freedom of channels can be further improved. We try this design and the results are presented in Table. 3. Two observations are worth noting. First, InjectTST-PaT-RC and InjectTST-CaT-RC achieve much better series performance compared with ones without RC in the ETTh1 dataset. Second, InjectTST-PaT-RC and InjectTST-CaT-RC become more unstable in the cases of longer series forecasting, such as prediction 720 steps in Weather, ETTh2, and ETTm1 datasets. The reason may come from the different characteristics of the datasets. If a dataset needs less channel dependencies, a little global information can make great improvement so that residual connection is needed to mitigate redundant global information. More importantly, all of the InjectTST variants present considerable improvement. It indicates that the injection design is effective and potential for MTS forecasting.

## 5 Conclusion and Future Work

Designing a model that can incorporate the merits of both channel independence and channel mixing is a key to further improvement of MTS forecasting. In this paper, we provide a different way to achieve channel-mixing time series forecasting. That is, channel-independent structures are used as backbones and cross-channel global information is injected into individual channels so as to improve the information content of each channel. In our proposed InjectTST, a channel identifier, a global mixing module, and a self-contextual attention module are devised to achieve a selective and harmless injection framework. Experiments demonstrate that InjectTST can effectively improve forecasting performance with negligible impact on robustness.

InjectTST not only presents an effective MTS forecasting method but also a new framework for channel mixing. For injection framework design, the global mixing module can be further explored for more effectiveness and efficiency. The self-contextual attention module for injection can also be replaced by a more effective design so that noise can be further reduced. In summary, InjectTST can be a potential framework to bridge channel-independent and channel-mixing models for MTS forecasting.

Models		InjectTST		InjectTST w/o CID		InjectTST w/o GI		PatchTST	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	<b>0.140</b>	<b>0.192</b>	0.144	0.194	<u>0.143</u>	0.197	0.144	<u>0.193</u>
	192	<b>0.184</b>	<b>0.235</b>	<u>0.186</u>	<u>0.238</u>	<u>0.186</u>	0.238	0.193	0.239
	336	<b>0.236</b>	<u>0.278</u>	<b>0.236</b>	<b>0.275</b>	0.237	0.281	0.244	0.280
	720	<b>0.308</b>	<u>0.331</u>	<b>0.308</b>	<b>0.328</b>	0.316	0.335	0.318	0.334
	Avg	<b>0.217</b>	<b>0.259</b>	<u>0.218</u>	<b>0.259</b>	0.220	0.263	0.225	0.262
ETTh2	96	0.275	0.337	<b>0.274</b>	<b>0.335</b>	0.276	0.338	0.284	0.343
	192	0.344	<b>0.385</b>	<b>0.343</b>	0.386	0.351	0.386	0.357	0.387
	336	0.370	0.406	<u>0.369</u>	<b>0.403</b>	<b>0.365</b>	<u>0.405</u>	0.377	0.410
	720	<b>0.398</b>	0.436	0.402	<b>0.434</b>	0.420	0.455	<u>0.400</u>	<u>0.435</u>
	Avg	<b>0.347</b>	<u>0.391</u>	<b>0.347</b>	<b>0.389</b>	0.353	0.396	0.354	0.394
ETTm1	96	0.285	0.343	<b>0.283</b>	<b>0.338</b>	<u>0.284</u>	<u>0.340</u>	0.289	0.344
	192	<b>0.323</b>	0.365	0.326	<b>0.363</b>	<u>0.325</u>	0.367	0.330	0.369
	336	<u>0.354</u>	<b>0.383</b>	<u>0.354</u>	<u>0.385</u>	0.357	0.387	<b>0.353</b>	0.387
	720	<b>0.403</b>	<b>0.413</b>	<u>0.406</u>	0.419	<u>0.406</u>	0.417	0.407	<u>0.415</u>
	Avg	<b>0.341</b>	<b>0.376</b>	<u>0.342</u>	<b>0.376</b>	0.343	0.378	0.345	0.379

Table 2: Ablation Study of channel identifier (CID) and global information (GI). InjectTST w/o CID means that the channel identifier is removed from InjectTST, so global information plays a major role. InjectTST w/o GI means that the global information, including the global mixing module and self-contextual attention module, is removed from InjectTST, so the channel identifier plays a major role. The best results are marked in bold and the second best are underlined.

Models		InjectTST		InjectTST-CaT		InjectTST-PaT-RC		InjectTST-CaT-RC		PatchTST	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	<b>0.140</b>	<b>0.192</b>	0.142	0.194	0.141	<u>0.193</u>	0.141	0.194	0.144	<u>0.193</u>
	192	<b>0.184</b>	0.235	0.186	<b>0.234</b>	0.186	0.239	<b>0.184</b>	0.236	0.193	0.239
	336	0.236	<u>0.278</u>	<b>0.235</b>	<b>0.275</b>	0.240	0.279	<b>0.235</b>	0.278	0.244	0.280
	720	<b>0.308</b>	<u>0.331</u>	<u>0.310</u>	<b>0.329</b>	0.320	0.339	0.311	0.333	0.318	0.334
	Avg	<b>0.217</b>	<u>0.259</u>	<u>0.218</u>	<b>0.258</b>	0.222	0.263	0.218	0.260	0.225	0.262
ETTh1	96	0.367	0.394	<b>0.360</b>	<b>0.392</b>	<u>0.365</u>	0.394	<u>0.365</u>	<u>0.393</u>	0.366	0.397
	192	0.400	<b>0.415</b>	<b>0.396</b>	0.416	0.401	0.417	0.406	0.418	0.401	<u>0.416</u>
	336	0.432	0.438	0.455	<u>0.462</u>	<b>0.422</b>	<u>0.429</u>	0.423	<b>0.428</b>	0.450	<u>0.456</u>
	720	0.468	0.482	0.465	0.479	<u>0.439</u>	<u>0.453</u>	<b>0.426</b>	<b>0.444</b>	0.472	0.484
	Avg	0.417	0.432	0.419	0.437	<u>0.407</u>	<u>0.423</u>	<b>0.405</b>	<b>0.421</b>	0.422	0.438
ETTh2	96	0.275	<u>0.337</u>	0.273	<b>0.336</b>	<u>0.272</u>	0.338	<b>0.271</b>	0.337	0.284	0.343
	192	0.344	0.385	0.343	0.387	<b>0.336</b>	<b>0.381</b>	<b>0.336</b>	<u>0.382</u>	0.357	0.387
	336	0.370	0.406	<u>0.367</u>	<b>0.403</b>	<b>0.364</b>	0.407	<u>0.367</u>	<u>0.405</u>	0.377	0.410
	720	0.398	0.436	<b>0.396</b>	<b>0.434</b>	0.414	0.448	0.409	0.445	0.400	<u>0.435</u>
	Avg	0.347	<u>0.391</u>	<b>0.345</b>	<b>0.390</b>	0.347	0.394	<u>0.346</u>	0.392	0.354	0.394
ETTm1	96	0.285	0.343	<u>0.282</u>	<u>0.338</u>	<u>0.282</u>	0.341	<b>0.280</b>	<b>0.337</b>	0.289	0.344
	192	<b>0.323</b>	<b>0.365</b>	<b>0.323</b>	<b>0.365</b>	0.326	0.367	<b>0.323</b>	0.366	0.330	0.369
	336	<u>0.354</u>	<b>0.383</b>	0.355	0.386	0.356	0.386	<u>0.354</u>	<u>0.385</u>	<b>0.353</b>	0.387
	720	<b>0.403</b>	<b>0.413</b>	0.407	0.418	0.414	0.422	<u>0.404</u>	0.416	0.407	<u>0.415</u>
	Avg	0.341	<b>0.376</b>	0.342	0.377	0.345	0.379	<b>0.340</b>	<b>0.376</b>	0.345	0.379
ETTm2	96	0.165	<b>0.254</b>	0.166	0.256	<b>0.164</b>	<b>0.254</b>	<b>0.164</b>	<b>0.254</b>	0.170	0.257
	192	<b>0.219</b>	<b>0.293</b>	0.222	0.296	0.220	<b>0.293</b>	<b>0.219</b>	<b>0.293</b>	0.221	0.295
	336	<b>0.268</b>	<b>0.324</b>	0.272	0.329	0.276	0.332	<u>0.269</u>	<u>0.325</u>	<u>0.269</u>	<u>0.325</u>
	720	<u>0.354</u>	<b>0.382</b>	0.357	0.386	<b>0.351</b>	<b>0.382</b>	0.356	0.388	0.365	0.388
	Avg	<b>0.251</b>	<b>0.313</b>	0.254	0.317	0.253	<u>0.315</u>	<u>0.252</u>	<u>0.315</u>	0.256	0.316

Table 3: Variants of InjectTST. InjectTST refers to InjectTST-PaT for specification, which utilizes PaT global mixing design. InjectTST-CaT adopts a CaT global mixing design. InjectTST-PaT-RC and InjectTST-CaT-RC add an additional residual connection in the self-contextual attention module. The best results are in bold and the second best are underlined.

## References

- [Angryk *et al.*, 2020] Rafal A Angryk, Petrus C Martens, Berkay Aydin, Dustin Kempton, Sushant S Mahajan, Sunitha Basodi, Azim Ahmadzadeh, Xumin Cai, Soukaina Filali Boubrahimi, Shah Muhammad Hamdi, et al. Multivariate time series dataset for space weather data analytics. *Scientific data*, 7(1):227, 2020.
- [Benidis *et al.*, 2022] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6):1–36, 2022.
- [Cai *et al.*, 2020] Ling Cai, Krzysztof Janowicz, Gengchen Mai, Bo Yan, and Rui Zhu. Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. *Transactions in GIS*, 24(3):736–755, 2020.
- [Cao *et al.*, 2023] Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *arXiv preprint arXiv:2310.04948*, 2023.
- [Chang *et al.*, 2023] Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.
- [Cirstea *et al.*, 2022] Razvan-Gabriel Cirstea, Chenjuan Guo, Bin Yang, Tung Kieu, Xuanyi Dong, and Shirui Pan. Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting. In Lud De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 1994–2001. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Main Track.
- [Ekambaram *et al.*, 2023] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. *arXiv preprint arXiv:2306.09364*, 2023.
- [Han *et al.*, 2023] Lu Han, Han-Jia Ye, and De-Chuan Zhan. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *arXiv preprint arXiv:2304.05206*, 2023.
- [Liu *et al.*, 2021] Qingyao Liu, Jianwu Li, and Zhaoming Lu. St-tran: Spatial-temporal transformer for cellular traffic prediction. *IEEE Communications Letters*, 25(10):3325–3329, 2021.
- [Liu *et al.*, 2023] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.
- [Nie *et al.*, 2023] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.
- [Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.
- [Patton, 2013] Andrew Patton. Copula methods for forecasting multivariate time series. *Handbook of economic forecasting*, 2:899–960, 2013.
- [Wen *et al.*, 2022] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.
- [Wu *et al.*, 2019] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 1907–1913. International Joint Conferences on Artificial Intelligence Organization, 7 2019.
- [Wu *et al.*, 2021] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.
- [Wu *et al.*, 2022] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.
- [Yi *et al.*, 2023] Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Defu Lian, Ning An, Longbing Cao, and Zhendong Niu. Frequency-domain mlps are more effective learners in time series forecasting. *arXiv preprint arXiv:2311.06184*, 2023.
- [Zeng *et al.*, 2023] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.
- [Zhang and Yan, 2022] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.
- [Zhang *et al.*, 2023] Yitian Zhang, Liheng Ma, Soumyasundar Pal, Yingxue Zhang, and Mark Coates. Multi-resolution time-series transformer for long-term forecasting. *arXiv preprint arXiv:2311.04147*, 2023.
- [Zhou *et al.*, 2021] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.



- [Zhou *et al.*, 2022] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, pages 27268–27286. PMLR, 2022.
- [Zhou *et al.*, 2023] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*, 2023.