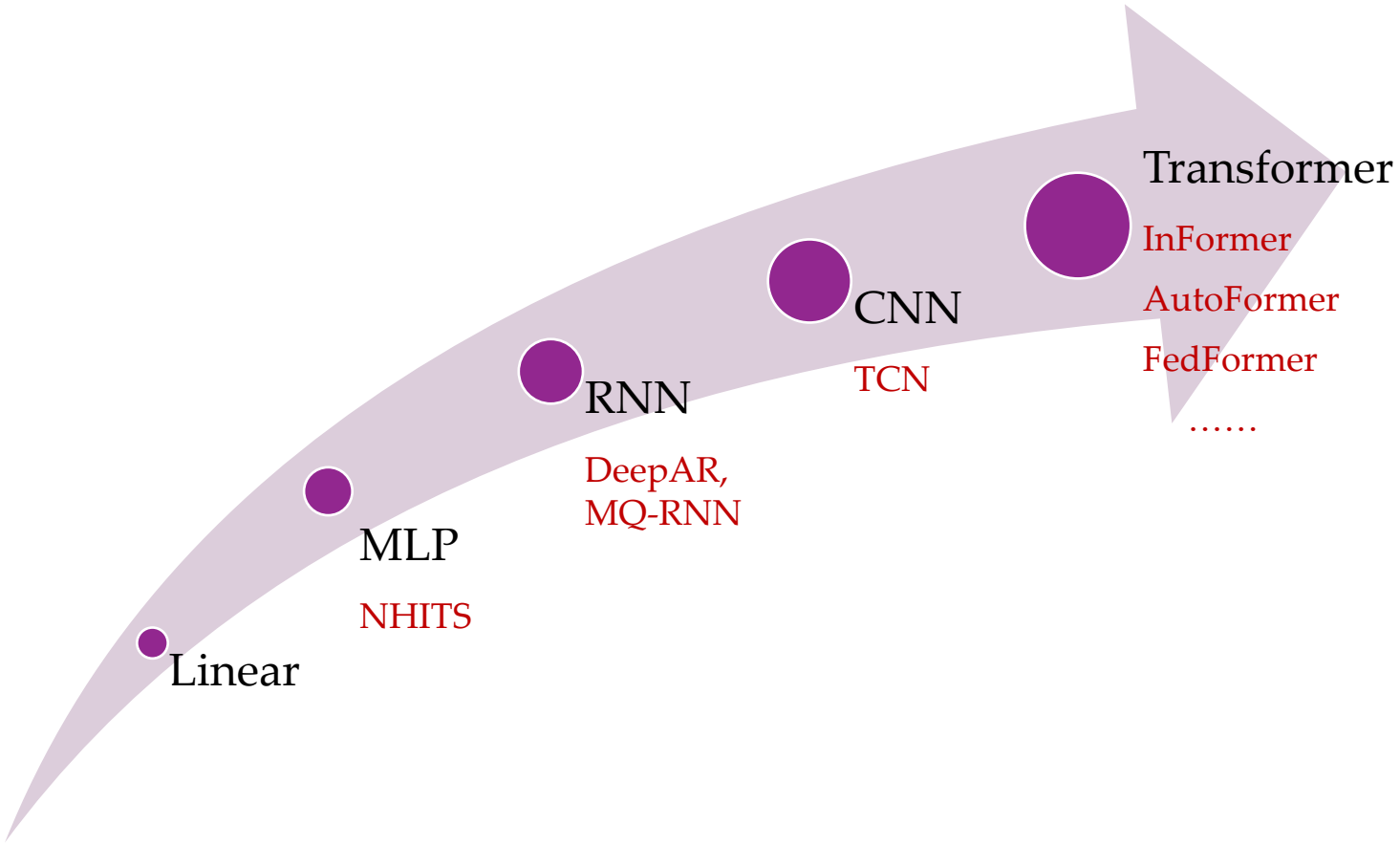
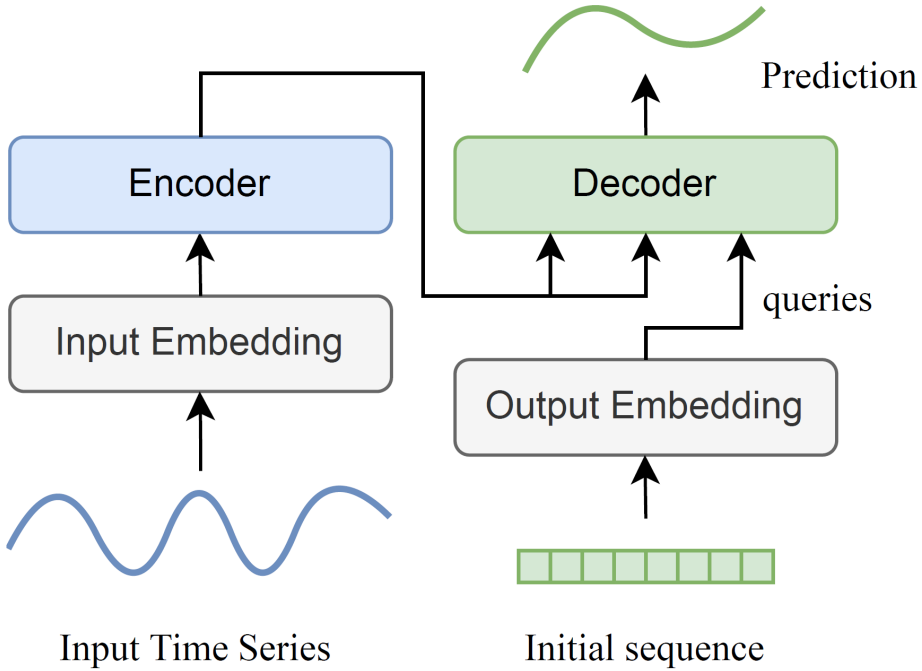
NJUAI 南京大学 人工智能学院
SCHOOL OF ARTIFICIAL INTELLIGENCE, NANJING UNIVERSITY

10 时间序列分析进阶深度学习方法

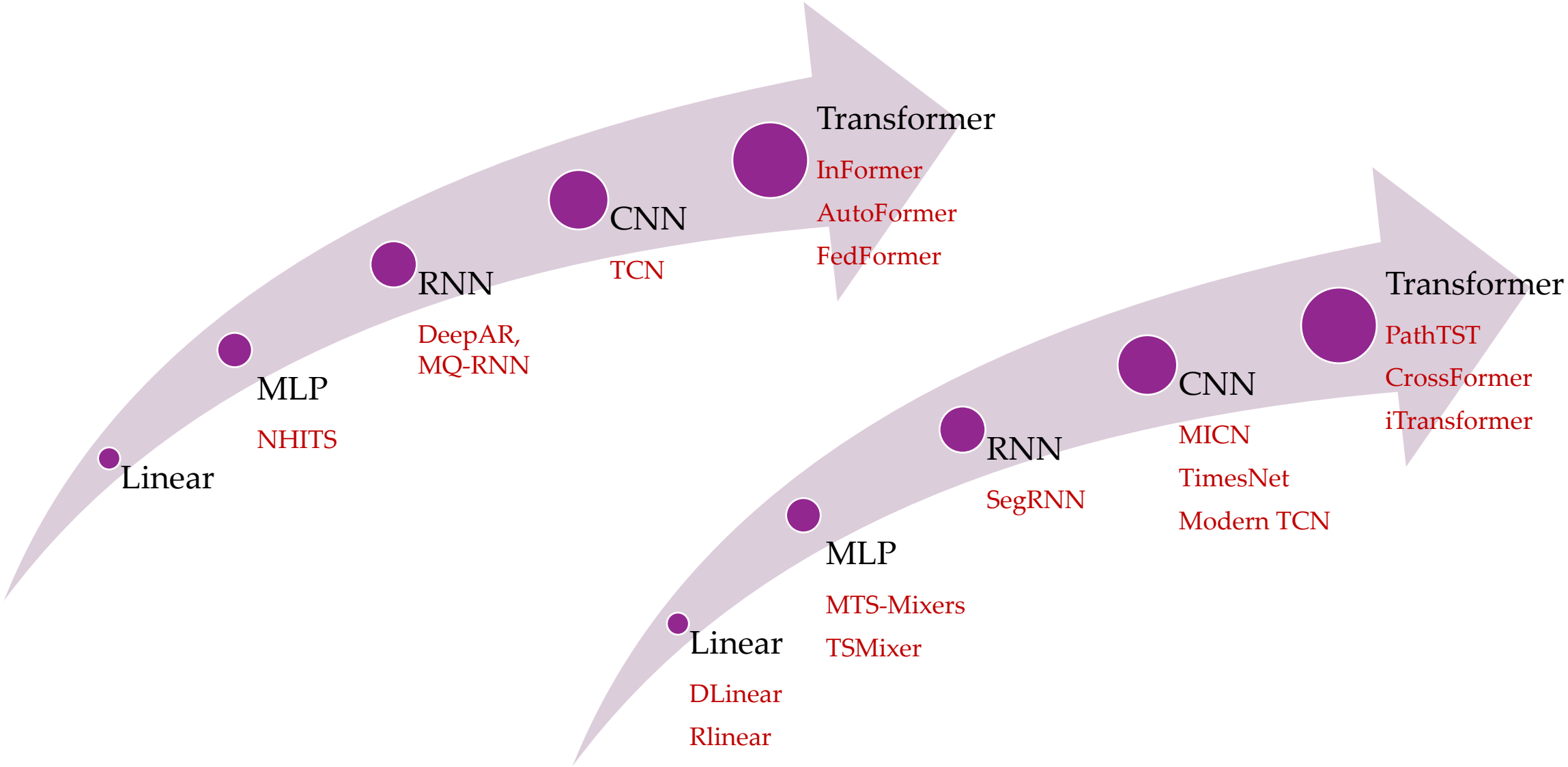
深度学习模块和训练方式的改进

神经网络时间序列预测模型





神经网络时间序列预测模型



概要

1. 时间序列模型的预处理

2. MLP模型

3. RNN模型

4. CNN模型

5. 注意力模型

……

概要

1.线性模型

2. Normalization

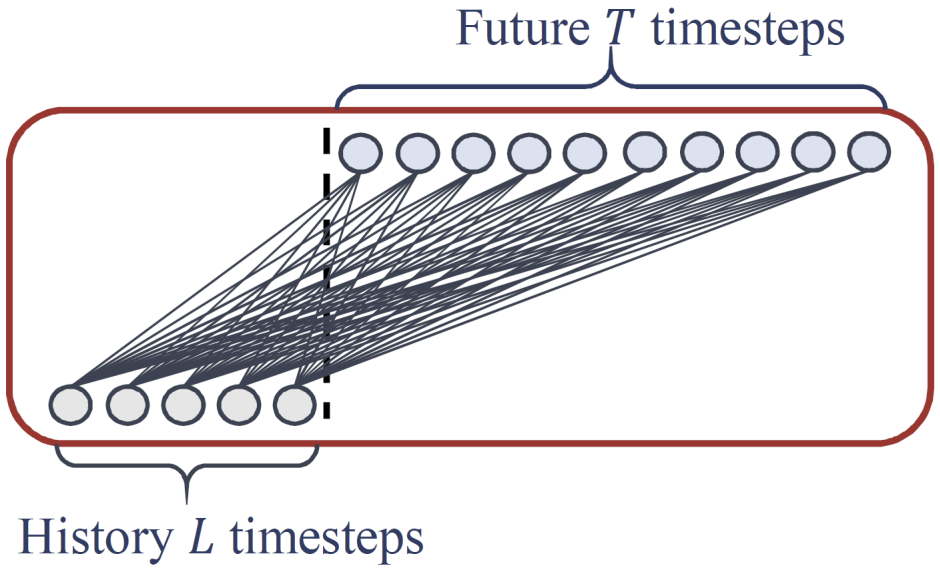
3. Channel Independent

DLinear

- 基于长为L历史数据，预测长为T未来数据
  - 历史数据：$\boldsymbol{Y}_{\text{old}} \in \mathbb{R}^{L \times d}$
  - 目标数据：$\boldsymbol{Y}_{new} \in \mathbb{R}^{T \times d}$
  - 模型：$\boldsymbol{W} \in \mathbb{R}^{T \times L}$

$$\boldsymbol{Y}_{new} = W\,\boldsymbol{Y}_{old}$$

- LTSF-Linear
- Dlinear：分解为Trend以及Remaining，分别做Linear
- Nlienar: 在Naive1的基础上做Linear



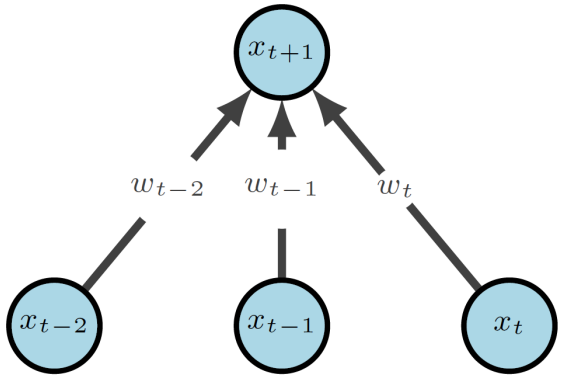
Ailing Zeng, Muxi Chen, Lei Zhang, Qiang Xu. **Are Transformers Effective for Time Series Forecasting**? AAAI 2023.

Methods		IMP.	Linear*		NLinear*		DLinear*		FEDformer		Autoformer		Informer		Pyraformer*		LogTrans		*Repeat**	
Metric		MSE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Electricity	96	27.40%	**0.140**	**0.237**	0.141	**0.237**	**0.140**	**0.237**	0.193	0.308	0.201	0.317	0.274	0.368	0.386	0.449	0.258	0.357	1.588	0.946
	192	23.88%	**0.153**	0.250	0.154	**0.248**	**0.153**	0.249	0.201	0.315	0.222	0.334	0.296	0.386	0.386	0.443	0.266	0.368	1.595	0.950
	336	21.02%	**0.169**	0.268	0.171	**0.265**	**0.169**	0.267	0.214	0.329	0.231	0.338	0.300	0.394	0.378	0.443	0.280	0.380	1.617	0.961
	720	17.47%	**0.203**	0.301	0.210	**0.297**	**0.203**	0.301	0.246	0.355	0.254	0.361	0.373	0.439	0.376	0.445	0.283	0.376	1.647	0.975
Exchange	96	45.27%	0.082	0.207	0.089	0.208	**0.081**	0.203	0.148	0.278	0.197	0.323	0.847	0.752	0.376	1.105	0.968	0.812	**0.081**	**0.196**
	192	42.06%	0.167	0.304	0.180	0.300	**0.157**	0.293	0.271	0.380	0.300	0.369	1.204	0.895	1.748	1.151	1.040	0.851	0.167	**0.289**
	336	33.69%	0.328	0.432	0.331	0.415	**0.305**	0.414	0.460	0.500	0.509	0.524	1.672	1.036	1.874	1.172	1.659	1.081	**0.305**	**0.396**
	720	46.19%	0.964	0.750	1.033	0.780	**0.643**	**0.601**	1.195	0.841	1.447	0.941	2.478	1.310	1.943	1.206	1.941	1.127	0.823	0.681
Traffic	96	30.15%	**0.410**	0.282	**0.410**	**0.279**	**0.410**	0.282	0.587	0.366	0.613	0.388	0.719	0.391	2.085	0.468	0.684	0.384	2.723	1.079
	192	29.96%	**0.423**	0.287	**0.423**	**0.284**	**0.423**	0.287	0.604	0.373	0.616	0.382	0.696	0.379	0.867	0.467	0.685	0.390	2.756	1.087
	336	29.95%	0.436	0.295	**0.435**	**0.290**	0.436	0.296	0.621	0.383	0.622	0.337	0.777	0.420	0.869	0.469	0.734	0.408	2.791	1.095
	720	25.87%	0.466	0.315	**0.464**	**0.307**	0.466	0.315	0.626	0.382	0.660	0.408	0.864	0.472	0.881	0.473	0.717	0.396	2.811	1.097
Weather	96	18.89%	**0.176**	0.236	0.182	**0.232**	**0.176**	0.237	0.217	0.296	0.266	0.336	0.300	0.384	0.896	0.556	0.458	0.490	0.259	0.254
	192	21.01%	**0.218**	0.276	0.225	**0.269**	0.220	0.282	0.276	0.336	0.307	0.367	0.598	0.544	0.622	0.624	0.658	0.589	0.309	0.292
	336	22.71%	**0.262**	0.312	0.271	**0.301**	0.265	0.319	0.339	0.380	0.359	0.395	0.578	0.523	0.739	0.753	0.797	0.652	0.377	0.338
	720	19.85%	0.326	0.365	0.338	**0.348**	**0.323**	0.362	0.403	0.428	0.419	0.428	1.059	0.741	1.004	0.934	0.869	0.675	0.465	0.394
ILI	24	47.86%	1.947	0.985	**1.683**	**0.858**	2.215	1.081	3.228	1.260	3.483	1.287	5.764	1.677	1.420	2.012	4.480	1.444	6.587	1.701
	36	36.43%	2.182	1.036	**1.703**	**0.859**	1.963	0.963	2.679	1.080	3.103	1.148	4.755	1.467	7.394	2.031	4.799	1.467	7.130	1.884
	48	34.43%	2.256	1.060	**1.719**	**0.884**	2.130	1.024	2.622	1.078	2.669	1.085	4.763	1.469	7.551	2.057	4.800	1.468	6.575	1.798
	60	34.33%	2.390	1.104	**1.819**	**0.917**	2.368	1.096	2.857	1.157	2.770	1.125	5.264	1.564	7.662	2.100	5.278	1.560	5.893	1.677
ETTh1	96	0.80%	0.375	0.397	**0.374**	**0.394**	0.375	0.399	0.376	0.419	0.449	0.459	0.865	0.713	0.664	0.612	0.878	0.740	1.295	0.713
	192	3.57%	0.418	0.429	0.408	**0.415**	**0.405**	0.416	0.420	0.448	0.500	0.482	1.008	0.792	0.790	0.681	1.037	0.824	1.325	0.733
	336	6.54%	0.479	0.476	**0.429**	**0.427**	0.439	0.443	0.459	0.465	0.521	0.496	1.107	0.809	0.891	0.738	1.238	0.932	1.323	0.744
	720	13.04%	0.624	0.592	**0.440**	**0.453**	0.472	0.490	0.506	0.507	0.514	0.512	1.181	0.865	0.963	0.782	1.135	0.852	1.339	0.756
ETTh2	96	19.94%	0.288	0.352	**0.277**	**0.338**	0.289	0.353	0.346	0.388	0.358	0.397	3.755	1.525	0.645	0.597	2.116	1.197	0.432	0.422
	192	19.81%	0.377	0.413	**0.344**	**0.381**	0.383	0.418	0.429	0.439	0.456	0.452	5.602	1.931	0.788	0.683	4.315	1.635	0.534	0.473
	336	25.93%	0.452	0.461	**0.357**	**0.400**	0.448	0.465	0.496	0.487	0.482	0.486	4.721	1.835	0.907	0.747	1.124	1.604	0.591	0.508
	720	14.25%	0.698	0.595	**0.394**	**0.436**	0.605	0.551	0.463	0.474	0.515	0.511	3.647	1.625	0.963	0.783	3.188	1.540	0.588	0.517
ETTm1	96	21.10%	0.308	0.352	0.306	0.348	**0.299**	**0.343**	0.379	0.419	0.505	0.475	0.672	0.571	0.543	0.510	0.600	0.546	1.214	0.665
	192	21.36%	0.340	0.369	0.349	0.375	**0.335**	**0.365**	0.426	0.441	0.553	0.496	0.795	0.669	0.557	0.537	0.837	0.700	1.261	0.690
	336	17.07%	0.376	0.393	0.375	0.388	**0.369**	**0.386**	0.445	0.459	0.621	0.537	1.212	0.871	0.754	0.655	1.124	0.832	1.283	0.707
	720	21.73%	0.440	0.435	0.433	0.422	**0.425**	**0.421**	0.543	0.490	0.671	0.561	1.166	0.823	0.908	0.724	1.153	0.820	1.319	0.729

Properties of The Linear Model

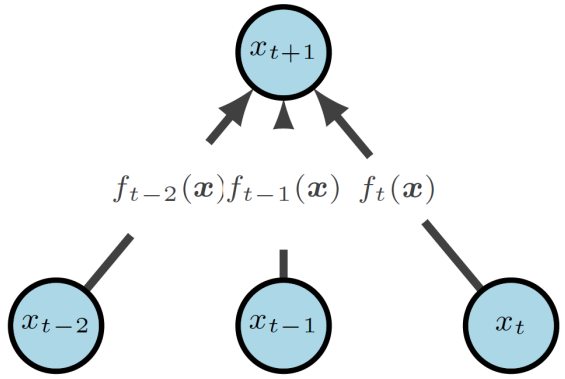
- The time-step-dependent linear model, despite its simplicity, proves to be highly effective in modeling temporal patterns.
- Conversely, even though recurrent or attention architectures have high representational capacity, achieving time-step independence is challenging for them. They usually overfit on the data instead of solely considering the positions.

$$x_{t+1} = \sum_{i=1}^{t} w_i x_i$$



Time-step-dependent

$$x_{t+1} = \sum_{i=1}^{t} f_i(\boldsymbol{x}) x_i$$



Data-dependent

Si-An Chen et al. **TSMixer: An all-MLP Architecture for Time Series Forecasting**. TMLR (2023)

Properties of The Linear Model

- A single linear layer can also effectively learn ***periodic*** patterns
- c channels and n time steps, predict next m steps

Theorem 1. *Given a seasonal time series satisfying $x(t) = s(t) = s(t-p)$ where $p \leq n$ is the period, there always exists an analytical solution for the linear model as*

$$[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \cdot \mathbf{W} + \mathbf{b} = [\boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2}, \ldots, \boldsymbol{x}_{n+m}], \tag{2}$$

$$\mathbf{W}_{ij}^{(k)} = \begin{cases} 1, & \textit{if } i = n - kp + (j \bmod p) \\ 0, & \textit{otherwise} \end{cases}, 1 \leq k \in \mathbb{Z} \leq \lfloor n/p \rfloor, b_i = 0. \tag{3}$$

linear mapping can predict periodic signals ***when the length of the input historical sequence is not less than the period****, but that is not a unique solution.*

Zhe Li et al. **Revisiting Long-term Time Series Forecasting: An Investigation on Linear Mapping**. CoRR abs/2305.10721 (2023)

Properties of The Linear Model

- A single linear layer can also effectively learn ***periodic*** patterns
- c channels and n time steps, predict next m steps

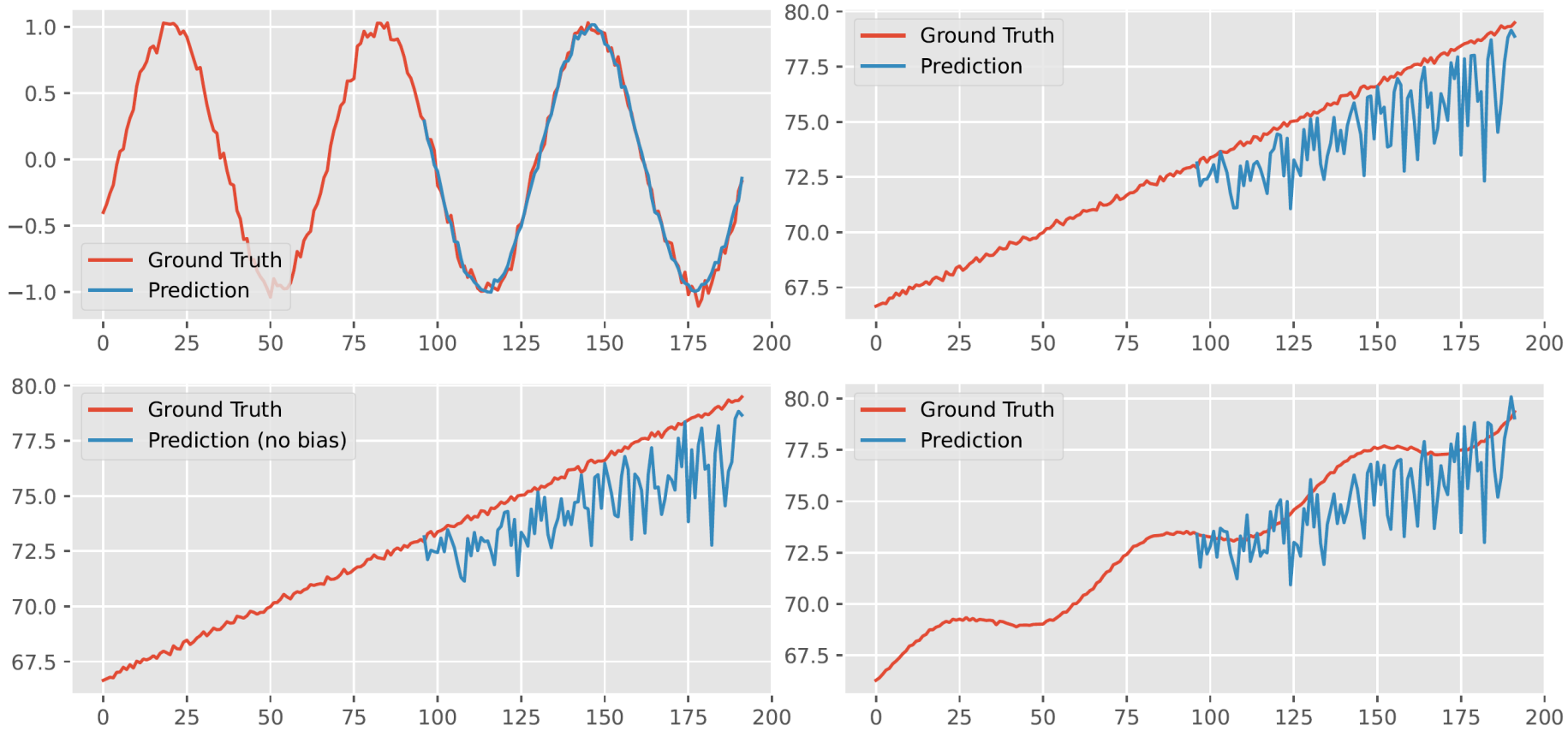
Corollary 1.1. *When the given time series satisfies $x(t) = ax(t-p) + c$ where a, c are scaling and translation factors, the linear model still has a closed-form solution to Equation 2 as*

$$\mathbf{W}_{ij}^{(k)} = \begin{cases} a^k, & \text{if } i = n - kp + (j \bmod p) \\ 0, & \text{otherwise} \end{cases}, 1 \le k \in \mathbb{Z} \le \lfloor n/p \rfloor, b_i = \sum_{l=0}^{k-1} a^l \cdot c. \tag{4}$$

the linear model ***fits seasonality well but performs poorly on the trend***

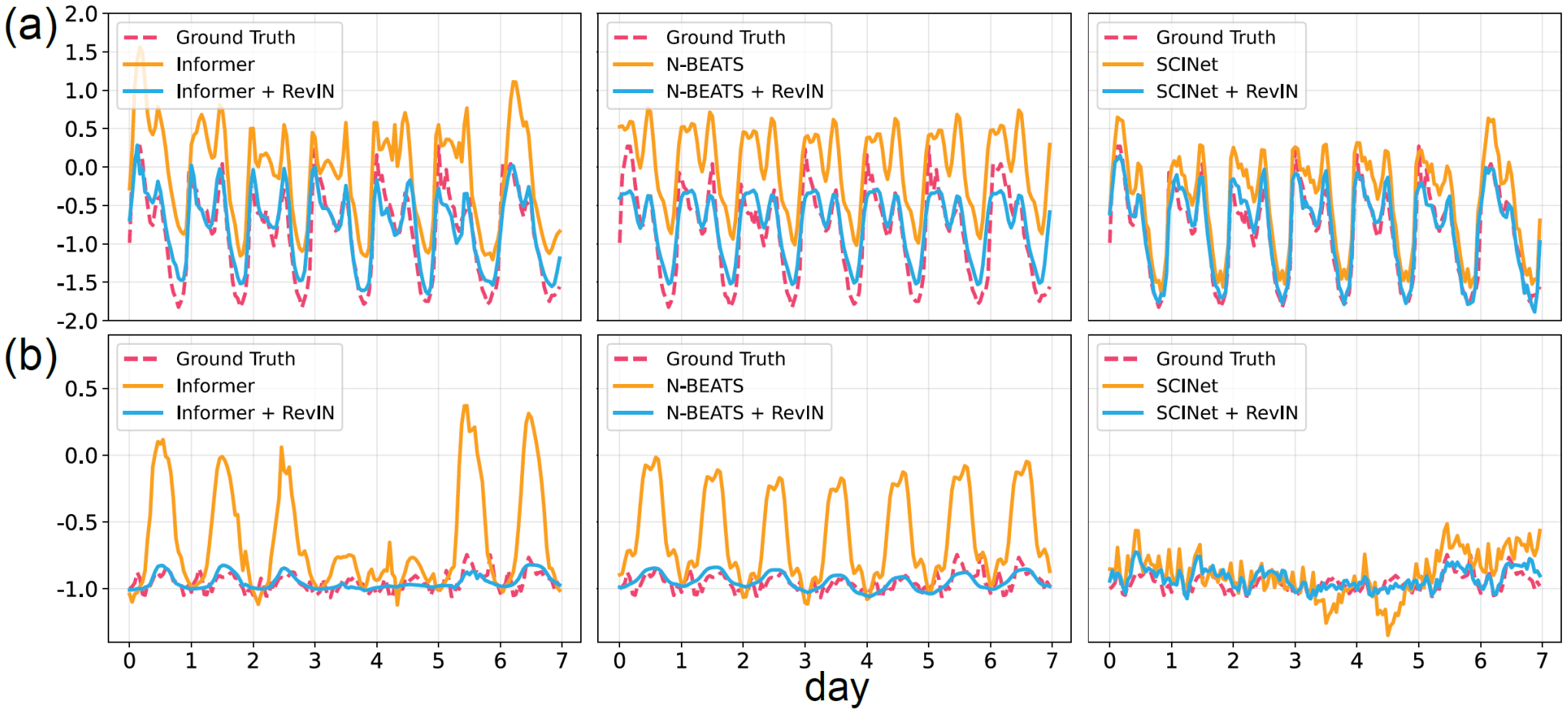
Properties of The Linear Model

- The linear model **fits seasonality well but performs poorly on the trend**



Reversible Instance Normalization (RevIN)

The predictions of the baselines are inaccurately *(a) shifted and (b) scaled*

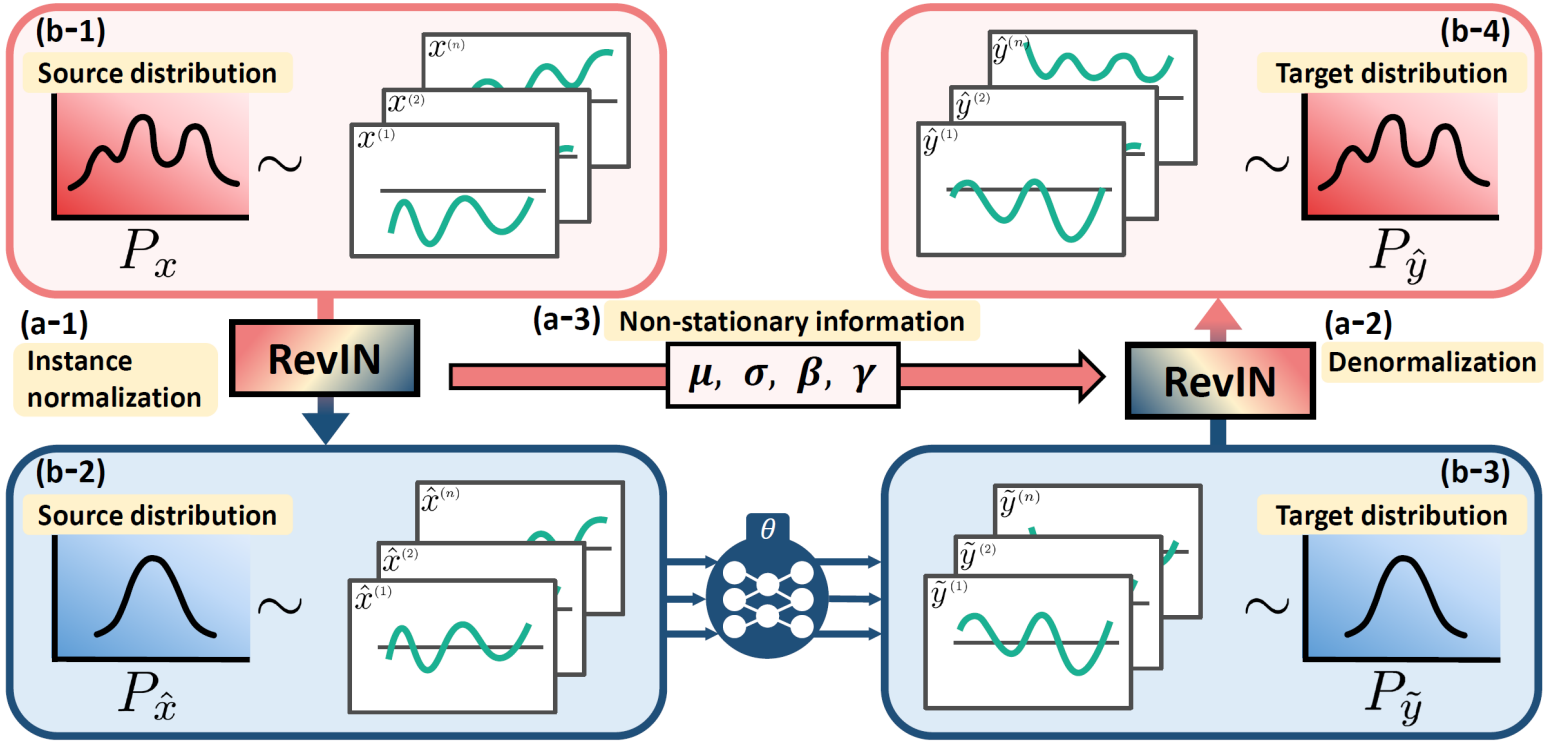


Taesung Kim et al. **Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift**. ICLR 2022

RevIN

The (a-3) non-stationary information includes statistical properties from the input data: mean μ, variance σ^2, and learnable affine parameters γ, β.

(a-1) and (a-2) are ***symmetrically*** structured to remove (a-3) nonstationary information from one layer and restore it on the other layer. Here, RevIN is applied to the input and output layers.



The normalization layer transforms the (b-1) original data distribution into a (b-2) mean-centered distribution, where the distribution discrepancy between different instances is reduced.

Using $\hat{x}$, the model predicts the future values $\tilde{y}$ following the (b-3) distribution where non-stationary information is eliminated.

Taesung Kim et al. **Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift**. ICLR 2022

RevIN

- Let K, T_x and T_y denote the number of variables, the input sequence length, and the model prediction length, $x^{(i)} \in \mathbb{R}^{K \times T_x} \rightarrow y^{(i)} \in \mathbb{R}^{K \times T_y}$
- For $x^{(i)}$,

$$\mathbb{E}_t\left[x_{kt}^{(i)}\right] = \frac{1}{T_x}\sum_{j=1}^{T_x} x_{kj}^{(i)} \text{ and } \mathrm{Var}\left[x_{kt}^{(i)}\right] = \frac{1}{T_x}\sum_{j=1}^{T_x}\left(x_{kj}^{(i)} - \mathbb{E}_t\left[x_{kt}^{(i)}\right]\right)^2$$

- Normalize the input data $x^{(i)}$ as

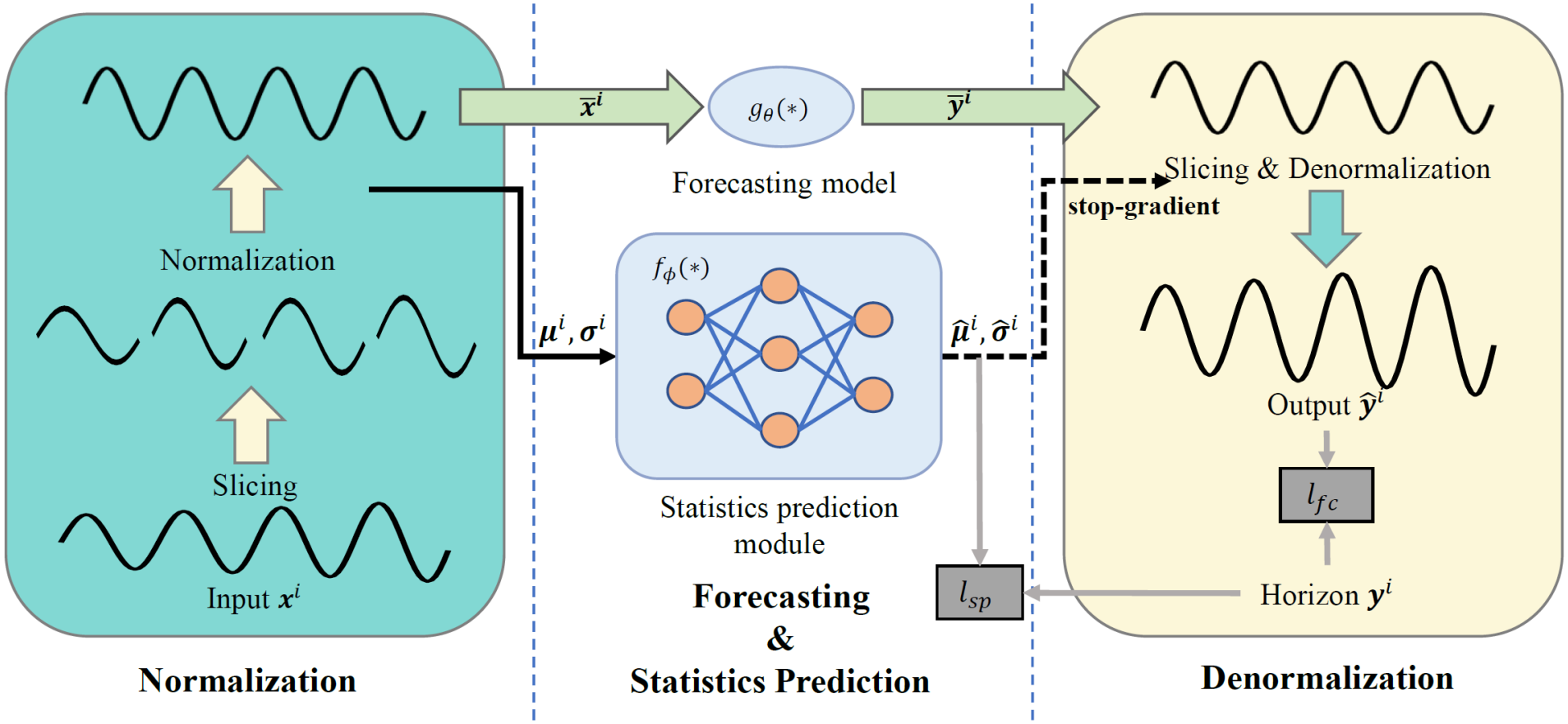
$$\hat{x}_{kt}^{(i)} = \gamma_k\left(\frac{x_{kt}^{(i)} - \mathbb{E}_t\left[x_{kt}^{(i)}\right]}{\sqrt{\mathrm{Var}\left[x_{kt}^{(i)}\right] + \epsilon}}\right) + \beta_k$$

- Denormalize the model output $\tilde{y}^{(i)}$

$$\hat{y}_{kt}^{(i)} = \sqrt{\mathrm{Var}\left[x_{kt}^{(i)}\right] + \epsilon} \cdot \left(\frac{\tilde{y}_{kt}^{(i)} - \beta_k}{\gamma_k}\right) + \mathbb{E}_t\left[x_{kt}^{(i)}\right]$$

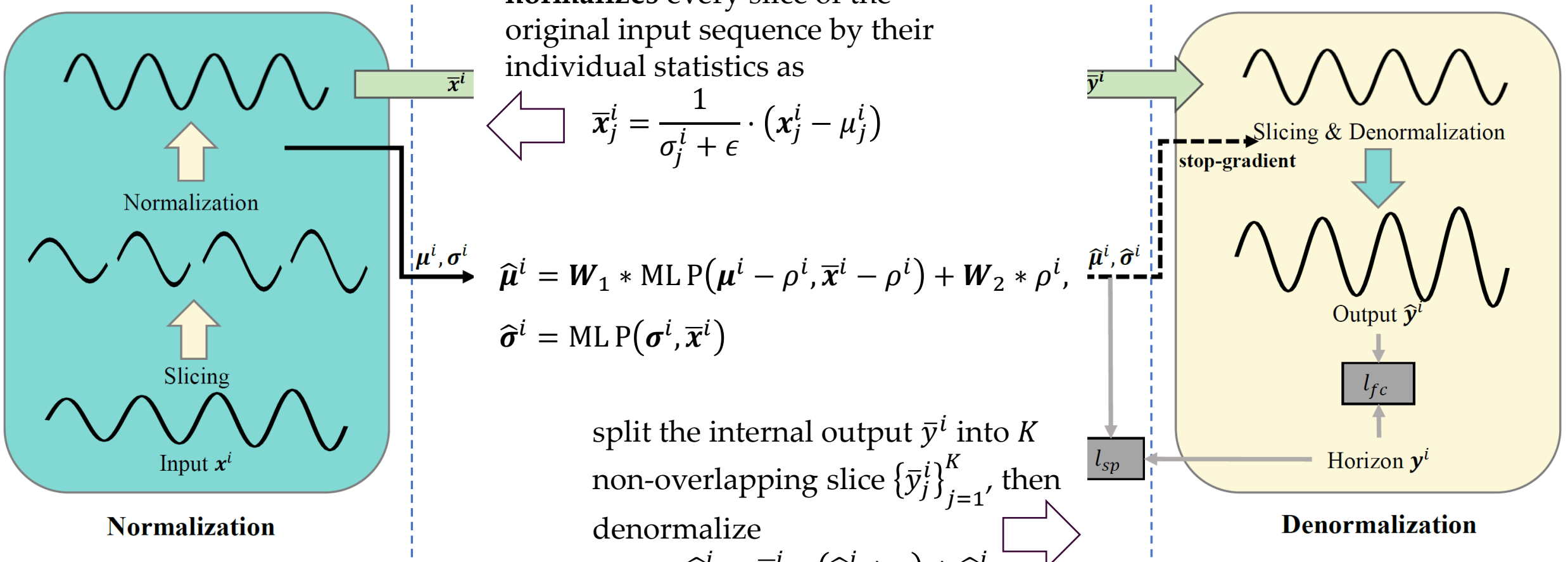
Method		Informer		**+ RevIN**		N-BEATS		**+ RevIN**		SCINet		**+ RevIN**	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh_1$	24	0.550	0.536	**0.504**	**0.472**	0.478	0.505	**0.330**	**0.373**	0.338	0.373	**0.308**	**0.347**
	48	0.772	0.668	**0.646**	**0.547**	0.536	0.542	**0.372**	**0.400**	0.436	0.459	**0.365**	**0.389**
	168	1.138	0.853	**0.655**	**0.561**	1.005	0.782	**0.466**	**0.452**	0.459	0.461	**0.406**	**0.416**
	336	1.278	0.909	**1.058**	**0.758**	0.932	0.743	**0.515**	**0.483**	0.527	0.513	**0.467**	**0.471**
	720	1.357	0.945	**0.926**	**0.717**	1.389	0.926	**0.576**	**0.534**	0.596	0.571	**0.507**	**0.505**
	960	1.470	0.990	**0.902**	**0.715**	1.383	0.932	**0.678**	**0.575**	0.604	0.574	**0.545**	**0.526**
$ETTh_2$	24	0.450	0.520	**0.238**	**0.325**	0.403	0.472	**0.192**	**0.276**	0.199	0.295	**0.180**	**0.263**
	48	2.171	1.200	**0.361**	**0.404**	1.330	0.918	**0.254**	**0.320**	0.350	0.422	**0.231**	**0.302**
	168	8.157	2.558	**0.859**	**0.649**	7.174	2.329	**0.410**	**0.418**	0.559	0.518	**0.337**	**0.378**
	336	4.746	1.844	**0.890**	**0.673**	4.859	1.863	**0.449**	**0.447**	0.664	0.583	**0.357**	**0.403**
	720	3.190	1.529	**0.576**	**0.546**	5.656	2.012	**0.496**	**0.482**	1.546	0.944	**0.411**	**0.445**
	960	2.972	1.441	**0.600**	**0.570**	6.408	2.077	**0.471**	**0.481**	1.862	1.066	**0.438**	**0.462**
$ETTm_1$	24	0.330	0.382	**0.309**	**0.352**	0.443	0.437	**0.403**	**0.392**	0.130	0.231	**0.106**	**0.196**
	48	0.499	0.486	**0.390**	**0.391**	0.453	0.472	**0.328**	**0.371**	0.155	0.262	**0.135**	**0.222**
	96	0.605	0.554	**0.405**	**0.411**	0.603	0.581	**0.379**	**0.406**	0.195	0.291	**0.162**	**0.247**
	288	0.906	0.738	**0.563**	**0.502**	0.849	0.702	**0.451**	**0.445**	0.361	0.419	**0.265**	**0.321**
	672	0.943	0.760	**0.663**	**0.550**	0.860	0.726	**0.555**	**0.511**	1.020	0.756	**0.357**	**0.380**
	1344	1.095	0.823	**0.824**	**0.632**	14.613	1.948	**0.631**	**0.556**	1.841	1.044	**0.412**	**0.422**
ECL	24	0.250	0.358	**0.148**	**0.257**	0.279	0.372	**0.176**	**0.285**	0.138	0.246	**0.112**	**0.207**
	48	0.300	0.386	**0.171**	**0.279**	0.309	0.388	**0.194**	**0.301**	0.163	0.265	**0.126**	**0.222**
	168	0.345	0.423	**0.261**	**0.354**	0.333	0.410	**0.218**	**0.320**	0.177	0.281	**0.153**	**0.249**
	336	0.429	0.473	**0.356**	**0.414**	0.326	0.406	**0.241**	**0.337**	0.202	0.308	**0.162**	**0.262**
	720	0.851	0.719	**0.834**	**0.700**	0.420	0.467	**0.303**	**0.383**	0.234	0.333	**0.183**	**0.281**
	960	0.930	0.750	**0.894**	**0.741**	0.399	0.455	**0.325**	**0.398**	0.235	0.330	**0.200**	**0.292**

Slice-level Adaptive Normalization (SAN)



Zhiding Liu et al. **Adaptive Normalization for Non-stationary Time Series Forecasting: A Temporal Slice Perspective**. NeurIPS 2023

Slice-level Adaptive Normalization (SAN)



normalizes every slice of the original input sequence by their individual statistics as

$$\bar{x}_j^i = \frac{1}{\sigma_j^i + \epsilon} \cdot \left(x_j^i - \mu_j^i\right)$$

$$\hat{\mu}^i = W_1 * \mathrm{MLP}\left(\mu^i - \rho^i, \bar{x}^i - \rho^i\right) + W_2 * \rho^i,$$

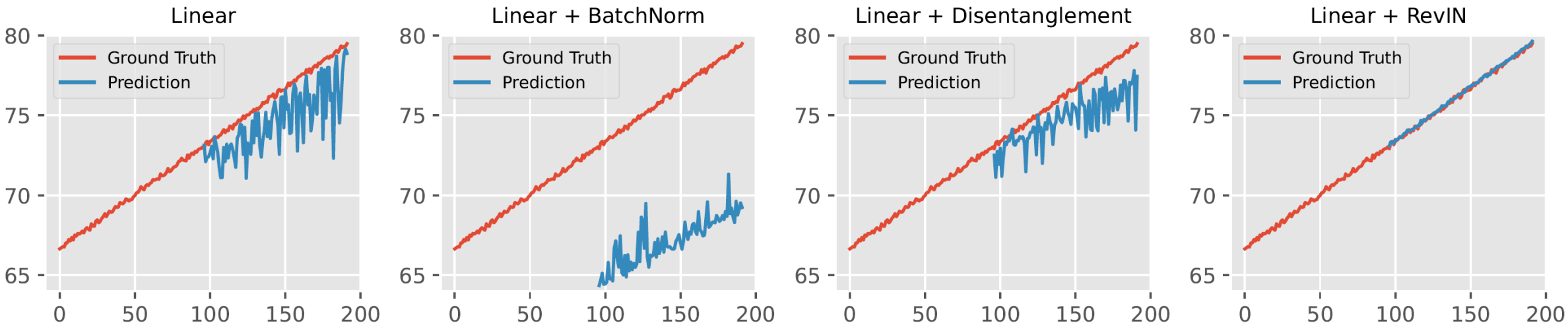
$$\hat{\sigma}^i = \mathrm{MLP}\left(\sigma^i, \bar{x}^i\right)$$

split the internal output $\bar{y}^i$ into K non-overlapping slice $\{\bar{y}_j^i\}_{j=1}^K$, then denormalize

$$\hat{y}_j^i = \bar{y}_j^i * \left(\hat{\sigma}_j^i + \epsilon\right) + \hat{\mu}_j^i.$$

RevIN and Linear Classifier

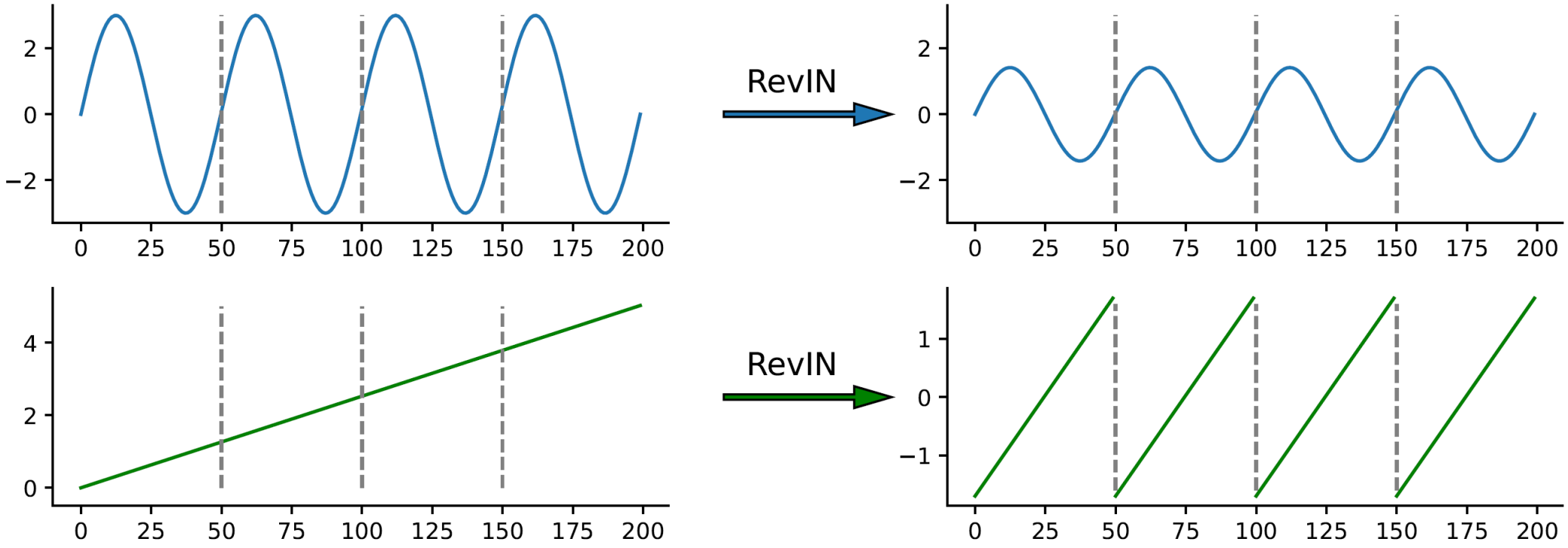
- Turning trend into seasonality



- Directly applying normalization to input data may erase this statistical information and lead to poor predictions;
- It is challenging to fit trend changes solely using a linear layer. Applying batch normalization even induces worse results. Disentangling the simulated time series also does not work.

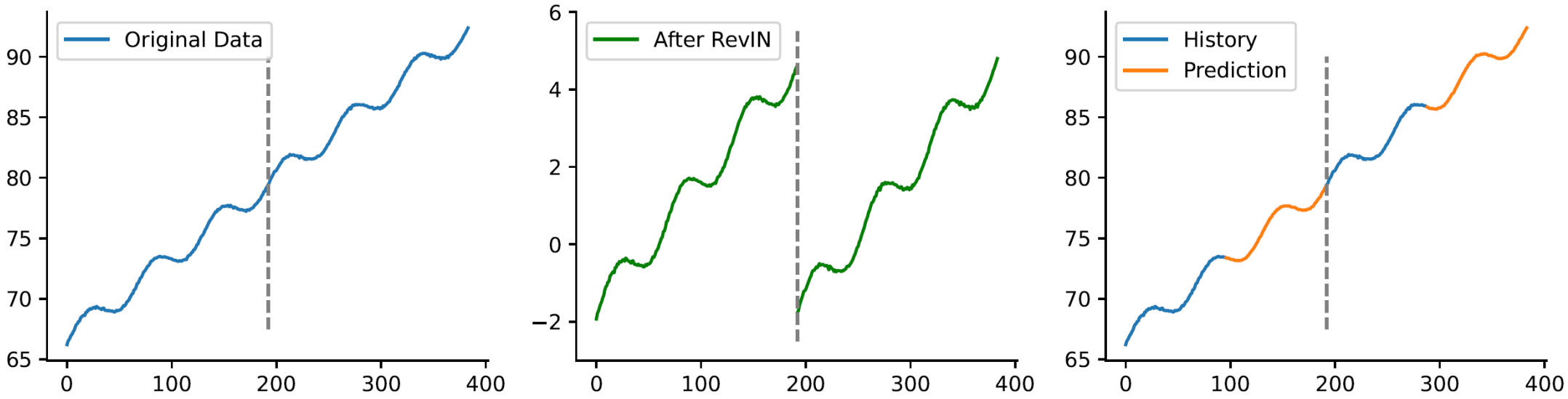
Zhe Li et al. **Revisiting Long-term Time Series Forecasting: An Investigation on Linear Mapping**. CoRR abs/2305.10721 (2023)

RevIN and Linear Classifier



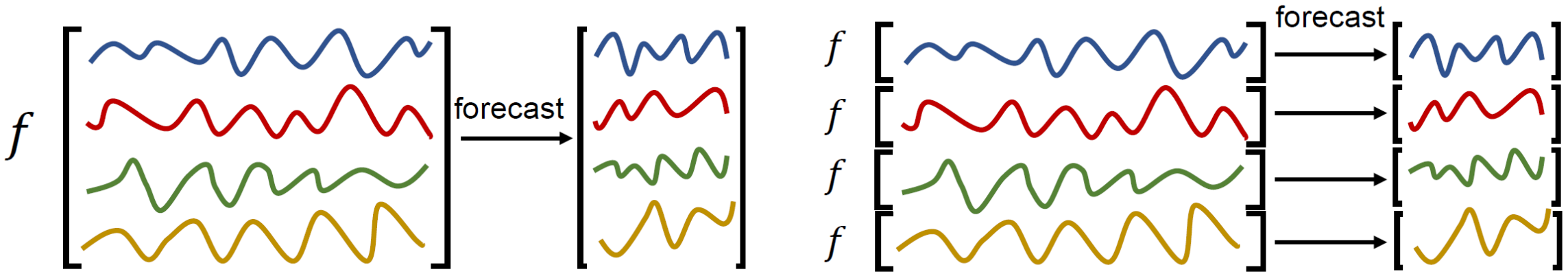
- For the ***seasonal*** signal, RevIN scales the range but does not change the periodicity.
- For the ***trend*** signal, RevIN scales each segment into the same range and exhibits **periodic** patterns. RevIN is capable of turning some trends into seasonality, making models better learn or memorize trend terms.

RevIN and Linear Classifier



- RevIN converts continuously changing trends into multiple segments with a fixed and similar trend, demonstrating periodic characteristics.
- As a result, errors in trend prediction caused by accumulated timesteps in the past can be alleviated, leading to more accurate forecasting results.

Channel Independent



(a) Channel Dependent (CD) Strategy

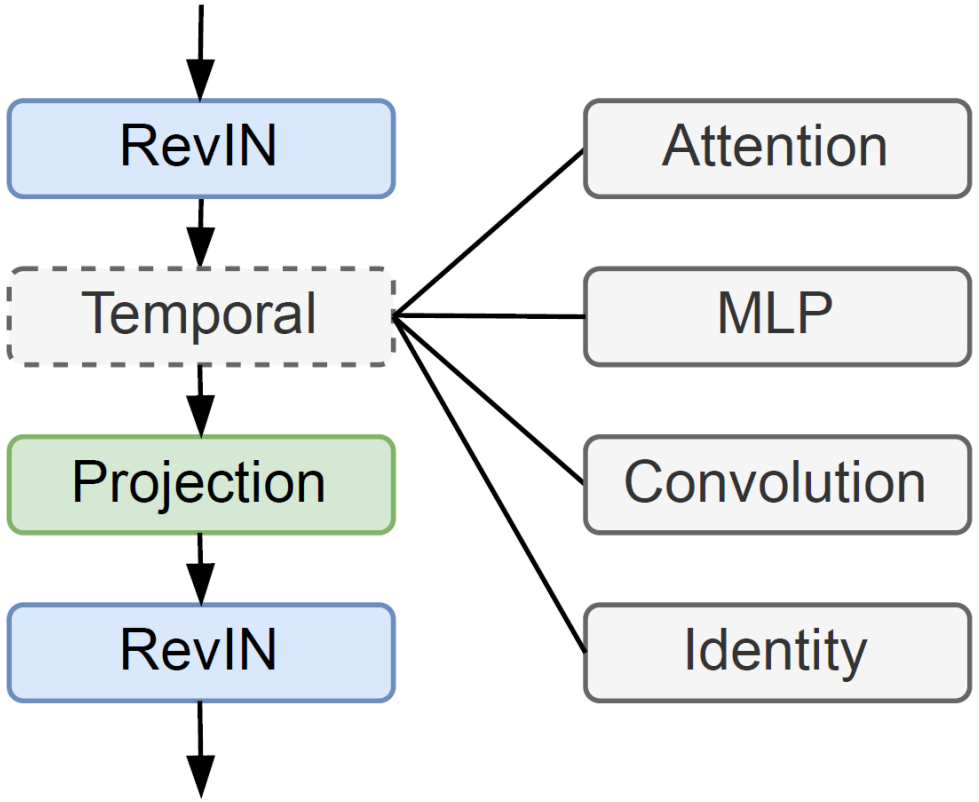
(b) Channel Independent (CI) Strategy

Lu Han, Han-Jia Ye, De-Chuan Zhan. **The Capacity and Robustness Trade-off: Revisiting the Channel Independent Strategy for Multivariate Time Series Forecasting**. CoRR abs/2304.05206 (2023)

MAE Comparison

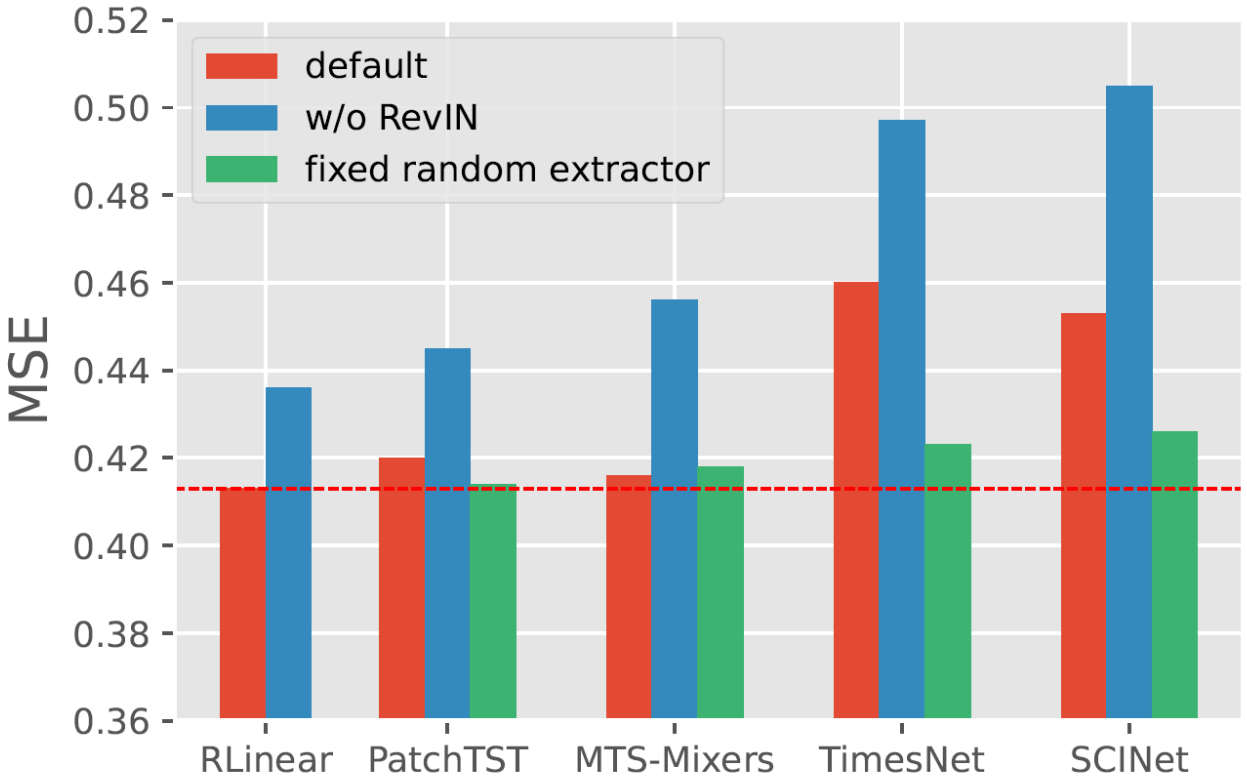
Dataset	Electricity		ETTh1		ETTh2		ETTm1		ETTm2		Exchange_Rate		Traffic		Weather		ILI		Mean
Horizon	48	96	48	96	48	96	48	96	48	96	48	96	48	96	48	96	24	36	
Linear (CD)	0.488	0.493	0.426	0.497	0.645	0.961	0.427	0.441	0.277	0.372	0.246	0.366	-	-	0.219	0.247	**0.955**	**0.945**	**11/16**
Linear (CI)	0.275	0.279	**0.374**	**0.398**	**0.302**	0.373	0.382	**0.377**	**0.251**	**0.285**	**0.164**	**0.218**	0.428	0.397	0.229	0.261	1.163	1.135	**2/16**
Improve (%)	**+43.57**	**+43.39**	**+12.26**	**+19.87**	**+53.28**	**+61.15**	**+10.49**	**+14.35**	+9.37	**+23.29**	**+33.29**	**+40.52**	-	-	-4.81	-5.36	**-21.75**	**-20.09**	**+19.55**
GBRT (CD)	-	-	0.499	0.560	0.732	0.936	0.424	0.477	0.404	0.517	0.719	0.912	-	-	0.236	0.268	1.597	1.554	**12/14**
GBRT (CI)	**0.249**	0.256	0.385	0.415	0.454	0.614	**0.360**	0.380	0.297	0.355	0.336	0.401	**0.282**	0.286	**0.190**	**0.232**	1.459	1.501	**0/14**
Improve (%)	-	-	**+22.87**	**+25.84**	**+37.92**	**+34.38**	**+15.00**	**+20.41**	**+26.49**	**+31.49**	**+53.19**	**+56.05**	-	-	**+19.49**	**+13.61**	+8.62	+3.44	**+26.34**
MLP (CD)	0.385	0.398	0.523	0.625	1.028	1.543	0.480	0.511	0.439	0.410	0.617	0.676	26.834	26.054	0.218	0.251	1.161	1.254	**12/18**
MLP (CI)	0.287	0.289	0.395	0.422	0.319	**0.365**	0.453	0.483	0.266	0.294	0.265	0.255	0.406	0.388	0.230	0.261	1.358	1.369	**1/18**
Improve (%)	**+25.43**	**+27.43**	**+24.45**	**+32.52**	**+68.93**	**+76.36**	+5.66	+5.38	**+39.39**	**+28.09**	**+57.07**	**+62.33**	**+98.49**	**+98.51**	-5.67	-4.00	**-16.89**	-9.14	**+34.13**
DeepAR (CD)	0.401	0.378	0.668	0.763	0.938	1.042	0.594	0.612	0.505	0.662	0.795	0.874	0.361	0.386	0.395	0.456	1.593	1.570	**13/18**
DeepAR (CI)	0.330	0.342	0.587	0.594	0.541	0.597	0.511	0.520	0.304	0.354	0.623	0.660	0.370	0.410	0.240	0.287	1.449	1.454	**0/18**
Improve (%)	**+17.72**	+9.51	**+12.14**	**+22.17**	**+42.28**	**+42.66**	**+14.03**	**+15.09**	**+39.76**	**+46.54**	**+21.65**	**+24.55**	-2.46	-6.08	**+39.29**	**+37.09**	+9.00	+7.38	**+21.80**
TCN (CD)	0.423	0.440	0.647	0.746	0.985	0.985	0.803	0.712	0.769	0.841	0.971	0.955	0.627	0.637	0.427	0.399	1.600	1.482	**12/18**
TCN (CI)	0.322	0.349	0.405	0.471	0.441	0.585	0.555	0.502	0.358	0.386	0.929	0.971	0.441	0.469	0.388	0.411	1.837	1.593	**1/18**
Improve (%)	**+23.75**	**+20.69**	**+37.42**	**+36.92**	**+55.20**	**+40.65**	**+30.83**	**+29.45**	**+53.44**	**+54.16**	+4.29	-1.69	**+29.63**	**+26.37**	+9.26	-2.89	**-14.81**	-7.49	**+23.62**
Informer (CD)	0.424	0.424	0.766	0.959	0.906	1.386	0.477	0.568	0.428	0.478	0.717	0.769	0.403	0.416	0.402	0.371	1.565	1.590	**15/18**
Informer (CI)	0.285	0.285	0.509	0.655	0.372	0.427	0.408	0.447	0.264	0.350	0.308	0.312	0.337	0.297	0.228	0.343	1.486	1.552	**0/18**
Improve (%)	**+32.90**	**+32.90**	**+33.56**	**+31.77**	**+58.89**	**+69.23**	**+14.54**	**+21.29**	**+38.41**	**+26.74**	**+56.99**	**+59.48**	**+16.31**	**+28.58**	**+43.27**	+7.31	+5.08	+2.40	**+32.20**
Transformer (CD)	0.352	0.357	0.734	0.774	0.829	1.111	0.458	0.533	0.404	0.547	0.571	0.769	0.364	0.359	0.343	0.452	1.508	1.555	**17/18**
Transformer (CI)	0.281	**0.255**	0.565	0.501	0.347	0.461	0.407	0.466	0.254	0.321	0.227	0.312	0.303	**0.273**	0.232	0.287	1.348	1.525	**0/18**
Improve (%)	**+20.06**	**+28.66**	**+23.03**	**+35.36**	**+58.18**	**+58.51**	**+11.30**	**+12.62**	**+37.15**	**+41.27**	**+60.25**	**+59.48**	**+16.70**	**+23.78**	**+32.57**	**+36.49**	**+10.62**	+1.88	**+31.55**

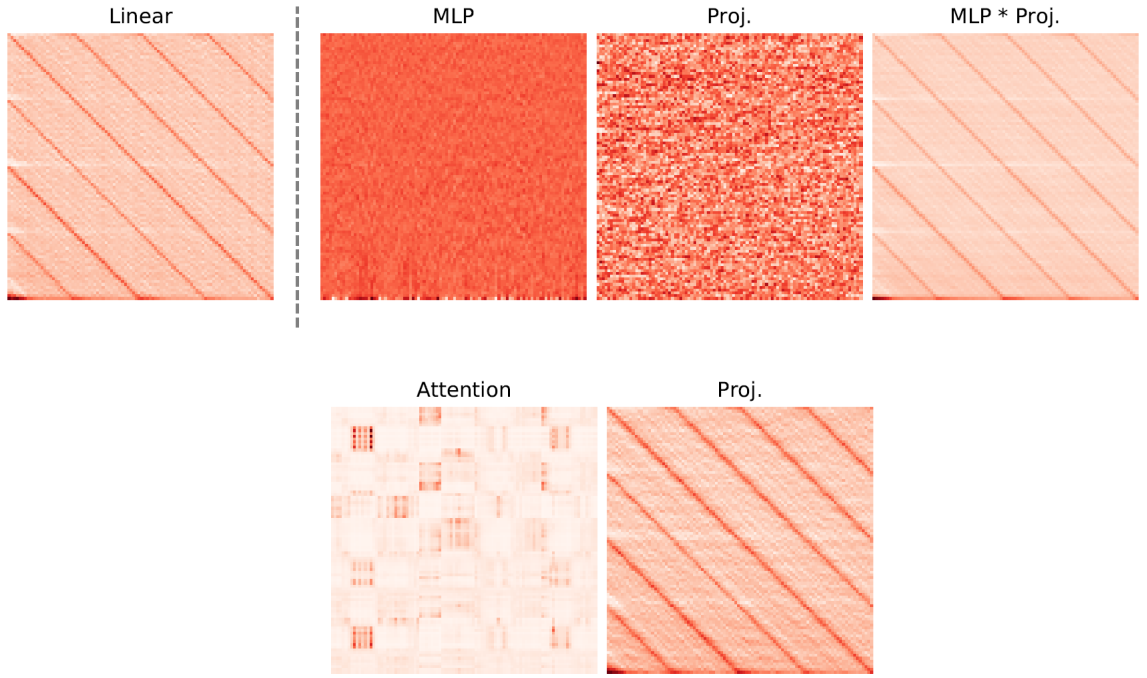
The Framework



- **Normalization**
- **Temporal Module**
- **Channel Independent Training**

The Influence of the Encoder





- Even using a randomly initialized temporal feature extractor with untrained parameters can induce competitive, even better forecasting results.

RLinear

- RLinear : RevIN + MLP + CI
- RMLP: RevIN + MLP + CI

Method		RLinear		RMLP		PatchTST		TimesNet		DLinear	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.366**	**0.391**	0.390	0.410	0.381	0.405	0.398	0.418	0.375	0.399
	192	**0.404**	**0.412**	0.430	0.432	0.416	0.423	0.447	0.449	0.405	0.416
	336	**0.420**	**0.423**	0.441	0.441	0.431	0.436	0.493	0.468	0.439	0.443
	720	**0.442**	**0.456**	0.506	0.495	0.450	0.466	0.518	0.504	0.472	0.490
ETTm1	96	0.301	**0.342**	0.298	0.345	**0.293**	0.345	0.335	0.380	0.306	0.349
	192	**0.335**	**0.363**	0.344	0.375	**0.335**	0.372	0.358	0.388	0.339	0.368
	336	0.370	**0.383**	0.390	0.410	**0.366**	0.392	0.406	0.418	0.374	0.390
	720	0.425	**0.414**	0.445	0.441	**0.420**	0.424	0.449	0.443	0.428	0.423
ETTh2	96	**0.262**	**0.331**	0.288	0.352	0.276	0.337	0.348	0.392	0.289	0.353
	192	**0.319**	**0.374**	0.343	0.387	0.339	0.379	0.362	0.404	0.383	0.418
	336	**0.325**	0.386	0.353	0.402	0.331	**0.380**	0.358	0.420	0.448	0.465
	720	**0.372**	**0.421**	0.410	0.440	0.379	0.422	0.442	0.463	0.605	0.551
ETTm2	96	**0.164**	**0.253**	0.174	0.259	0.165	0.256	0.188	0.267	0.167	0.260
	192	**0.219**	**0.290**	0.236	0.303	0.238	0.305	0.252	0.308	0.224	0.303
	336	**0.273**	**0.326**	0.291	0.338	0.276	0.332	0.304	0.353	0.281	0.342
	720	**0.366**	**0.385**	0.371	0.391	0.369	0.391	0.405	0.409	0.397	0.421
Weather	96	0.175	0.225	**0.149**	**0.202**	0.155	0.205	0.172	0.220	0.176	0.237
	192	0.218	0.260	**0.194**	**0.242**	0.199	0.245	0.219	0.261	0.220	0.282
	336	0.265	0.294	**0.243**	**0.282**	0.249	0.284	0.280	0.306	0.265	0.319
	720	0.329	0.339	**0.316**	**0.333**	0.319	0.335	0.365	0.359	0.326	0.363
ECL	96	0.140	0.235	**0.129**	**0.224**	0.133	0.226	0.168	0.272	0.140	0.237
	192	0.154	0.248	**0.147**	**0.240**	0.149	0.242	0.184	0.289	0.153	0.249
	336	0.171	0.264	**0.164**	**0.257**	0.167	0.260	0.198	0.300	0.169	0.267
	720	0.209	0.297	**0.203**	**0.291**	0.205	0.293	0.220	0.320	0.210	0.310

Zhe Li et al. **Revisiting Long-term Time Series Forecasting: An Investigation on Linear Mapping**. CoRR abs/2305.10721 (2023)

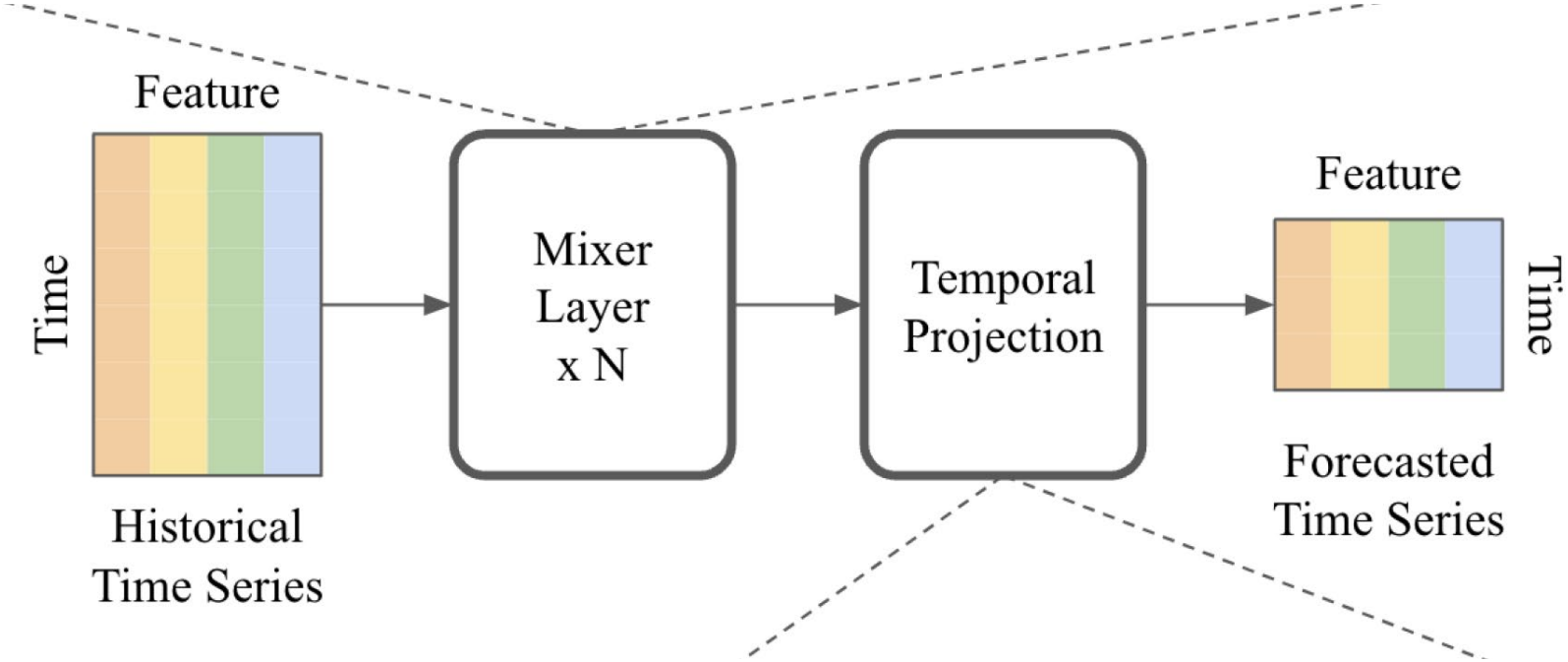
RLinear

Dataset	ETTh1		ETTm1		ETTh2		ETTm2	
Method	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
RLinear	*0.420*	*0.423*	*0.370*	*0.383*	*0.325*	*0.386*	*0.273*	*0.326*
PatchTST	0.431	0.436	**0.366**	0.392	0.331	**0.380**	**0.276**	0.332
†PatchTST	**0.429**	**0.435**	0.371	**0.389**	**0.328**	0.384	0.280	**0.331**
MTS-Mixers	**0.414**	0.425	0.378	0.399	0.353	0.407	0.291	0.337
†MTS-Mixers	0.423	**0.424**	**0.377**	**0.392**	**0.351**	**0.405**	**0.282**	**0.334**
TimesNet	0.493	0.468	0.406	0.418	0.358	0.420	**0.304**	0.353
†TimesNet	**0.428**	**0.439**	**0.384**	**0.400**	**0.342**	**0.406**	0.306	**0.351**
SCINet	0.467	0.469	0.404	0.423	0.365	0.414	0.329	0.369
†SCINet	**0.428**	**0.431**	**0.386**	**0.398**	**0.349**	**0.403**	**0.299**	**0.345**

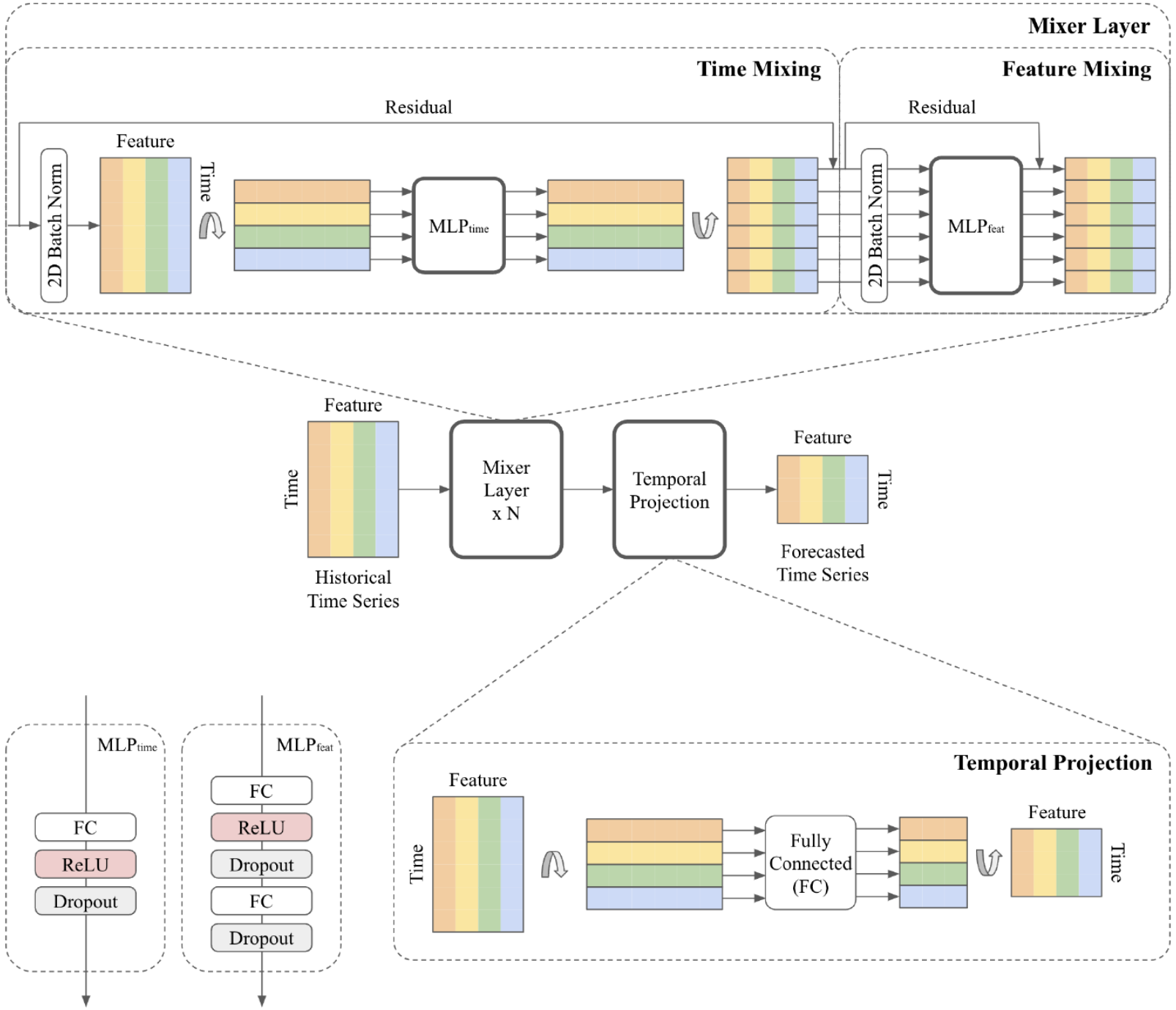
- † indicates the temporal feature extractor with fixed random weights

TSMixer

- To better leverage cross-variate information, TSMixer contains interleaving time-mixing and feature-mixing MLPs to aggregate information.
- The time-mixing MLPs are **shared** across all features and the feature-mixing MLPs are **shared** across all of the time steps.



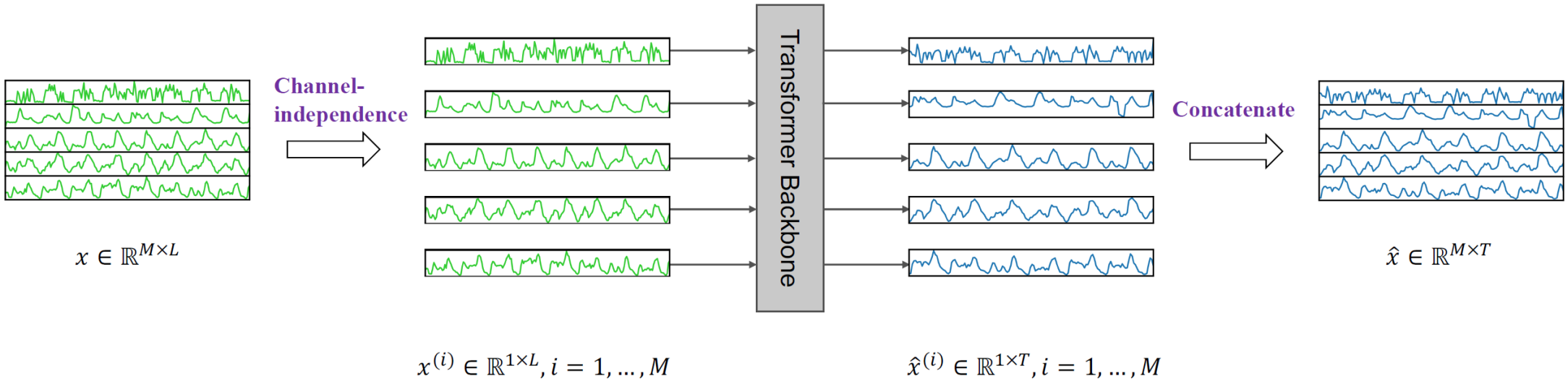
Si-An Chen et al. **TSMixer: An all-MLP Architecture for Time Series Forecasting**. TMLR (2023)



- Time-mixing MLP
- Feature-mixing MLP
- Temporal Projection
- Residual Connections
- Normalization

PathTST

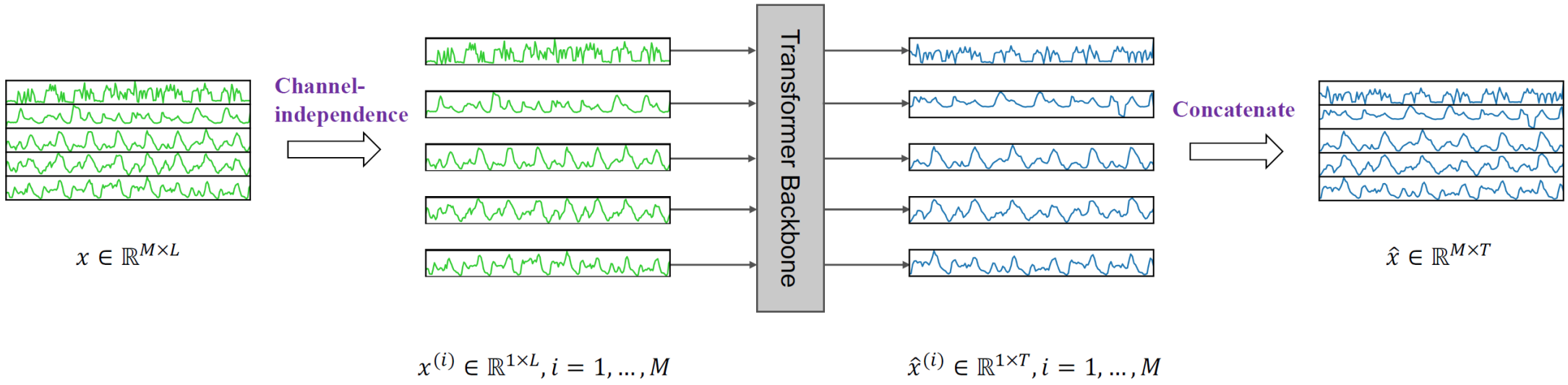
- Multivariate time series data is divided into different channels. They ***share the same Transformer backbone***, but the forward processes are independent



Forward Process. Denote a i-th univariate series of length L starting at time index 1 as $\boldsymbol{x}_{1:L}^{(i)} = \left(x_1^{(i)}, \ldots, x_L^{(i)}\right)$ where $i = 1, \ldots, M$. The input $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L)$ is split to M univariate series $\boldsymbol{x}^{(i)} \in \mathbb{R}^{1 \times L}$, where each of them is fed independently into the Transformer backbone. Then the Transformer backbone will provide prediction results $\hat{\boldsymbol{x}}^{(i)} = \left(\hat{x}_{L+1}^{(i)}, \ldots, \hat{x}_{L+T}^{(i)}\right) \in \mathbb{R}^{1 \times T}$.

Yuqi Nie et al. **A Time Series is Worth 64 Words: Long-term Forecasting with Transformers**. ICLR 2023

PathTST



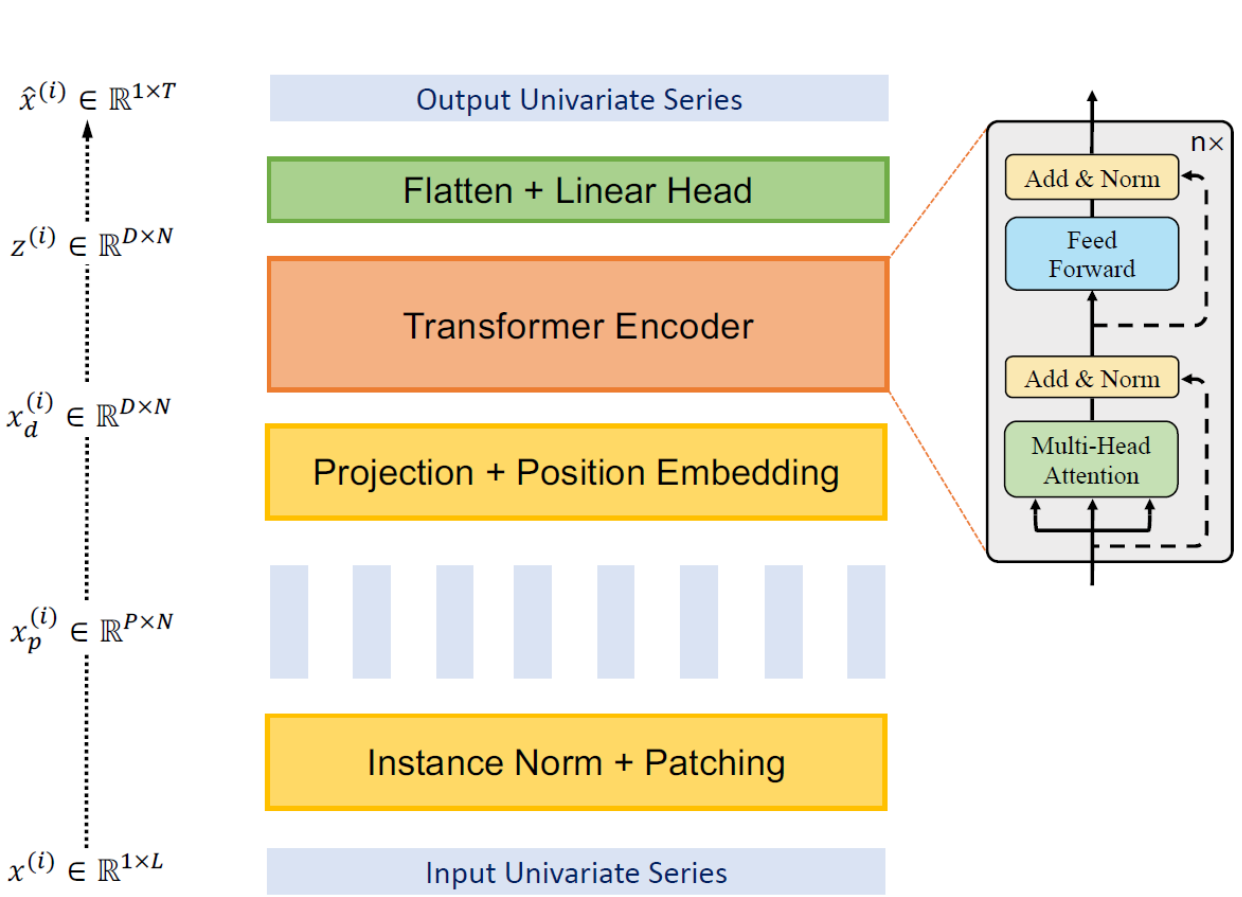
Patching. Each input univariate time series $\boldsymbol{x}^{(i)}$ is first divided into **patches** which can be either overlapped or non-overlapped. Denote the patch length as P and the stride, then the patching process will generate the a sequence of patches $\boldsymbol{x}_p^{(i)} \in \mathbb{R}^{P \times N}$ where N is the number of patches.

With the use of patches, the number of input tokens can be reduced.

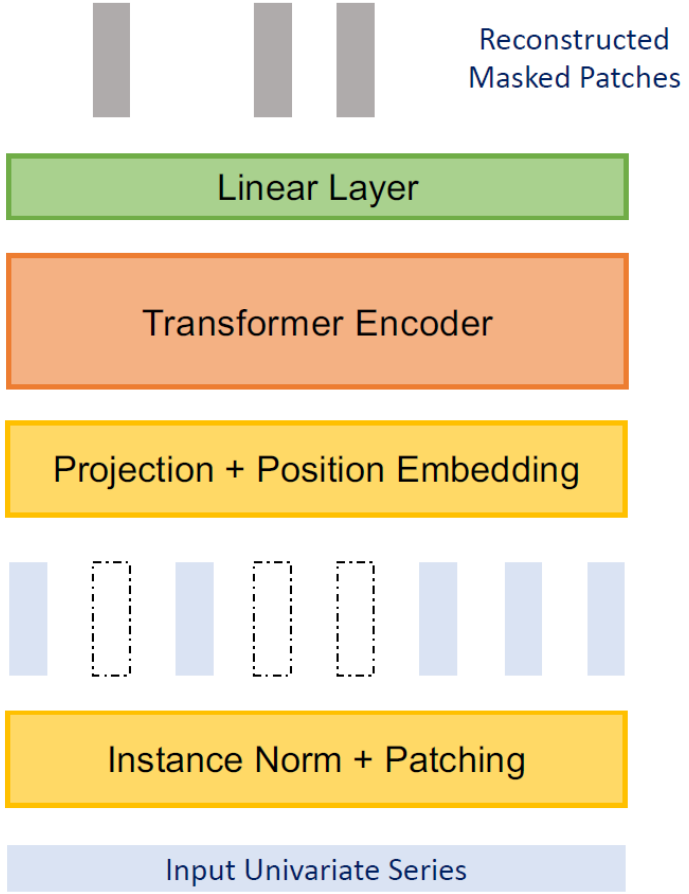
Positional Encoding. A learnable additive position encoding $W_{pos} \in \mathbb{R}^{D \times N}$ is applied to monitor the temporal order of patches.

Yuqi Nie et al. **A Time Series is Worth 64 Words: Long-term Forecasting with Transformers**. ICLR 2023

PathTST



(b) Transformer Backbone (Supervised)



(c) Transformer Backbone (Self-supervised)

PathTST

Models		PatchTST								FEDformer	
		P+CI		CI		P		Original			
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.152**	**0.199**	0.164	0.213	0.168	0.223	0.177	0.236	0.238	0.314
	192	**0.197**	**0.243**	0.205	0.250	0.213	0.262	0.221	0.270	0.275	0.329
	336	**0.249**	**0.283**	0.255	0.289	0.266	0.300	0.271	0.306	0.339	0.377
	720	**0.320**	**0.335**	0.327	0.343	0.351	0.359	0.340	0.353	0.389	0.409
Traffic	96	**0.367**	**0.251**	0.397	0.271	0.595	0.376	-	-	0.576	0.359
	192	**0.385**	**0.259**	0.411	0.276	0.612	0.387	-	-	0.610	0.380
	336	**0.398**	**0.265**	0.423	0.282	0.651	0.391	-	-	0.608	0.375
	720	**0.434**	**0.287**	0.457	0.309	-	-	-	-	0.621	0.375
Electricity	96	**0.130**	**0.222**	0.136	0.231	0.196	0.307	0.205	0.318	0.186	0.302
	192	**0.148**	**0.240**	0.164	0.263	0.215	0.323	-	-	0.197	0.311
	336	**0.167**	**0.261**	0.168	0.262	0.228	0.338	-	-	0.213	0.328
	720	**0.202**	**0.291**	0.219	0.312	0.244	0.345	-	-	0.233	0.344

(a) both patching and channel-independence are included in model (P+CI);

(b) only channel independence (CI);

(c) only patching (P);

(d) neither of them is included (Original TST model).

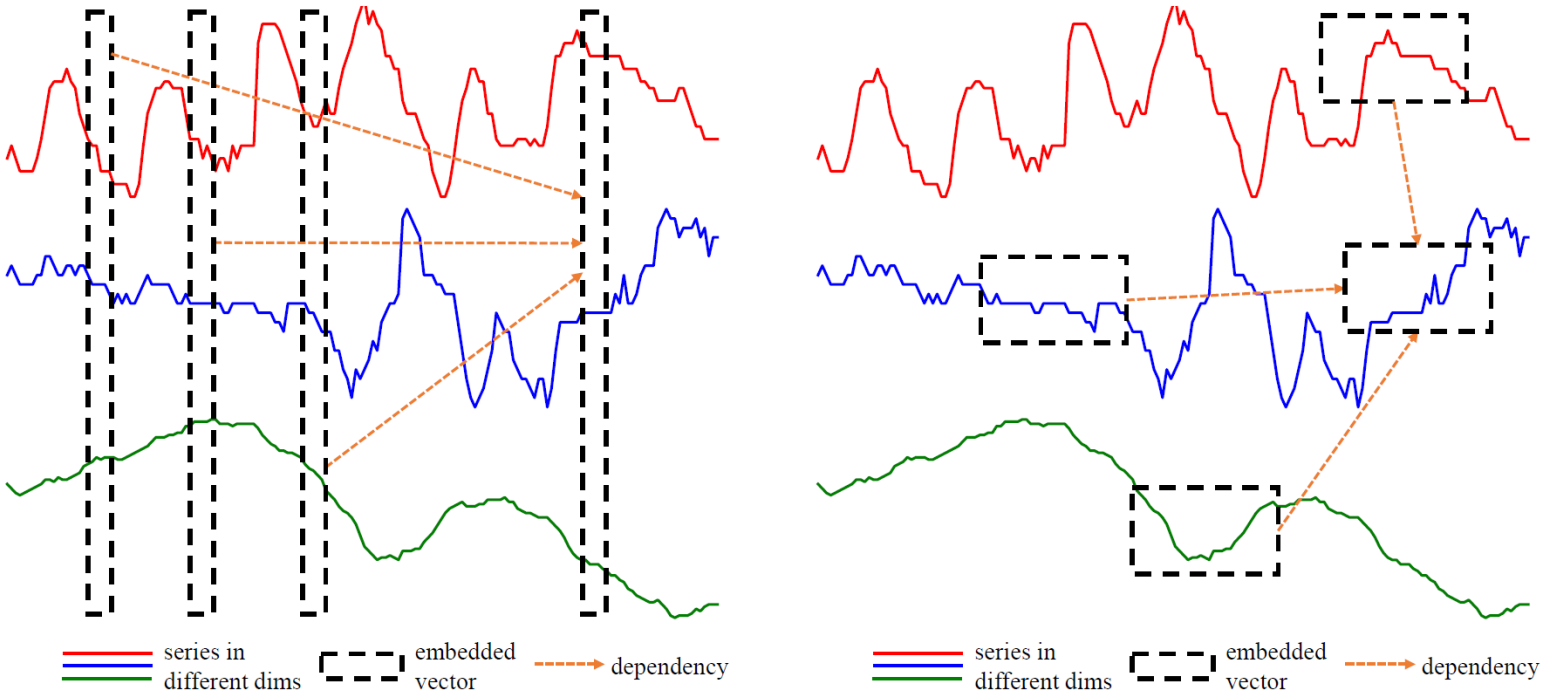
CrossFormer

- Existing embedding: $\mathbf{x}_t$ represents all the data points in D dimensions at step t

$$\mathbf{x}_t \in \mathbb{R}^D \rightarrow \mathbf{h}_t \in \mathbb{R}^{d_{model}}$$

- $\mathbf{x}_{1:T}$ is embedded into T vectors $\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_T\}$

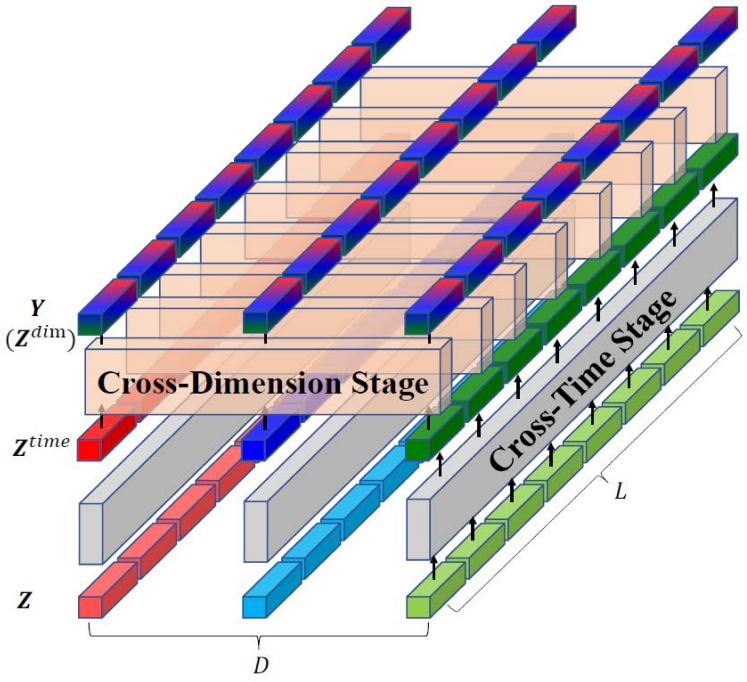
Dimension-Segment-Wise (DSW) embedding



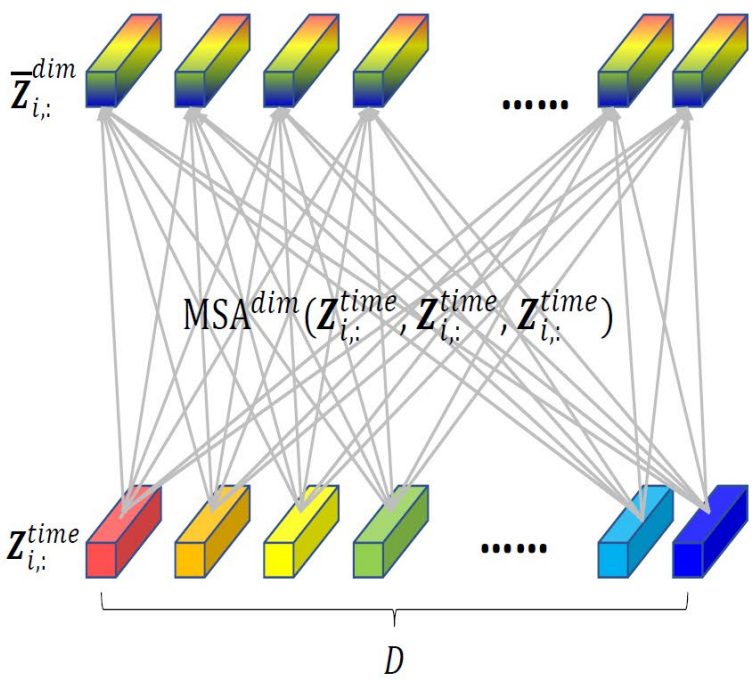
Yunhao Zhang, Junchi Yan. **Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting**. ICLR 2023

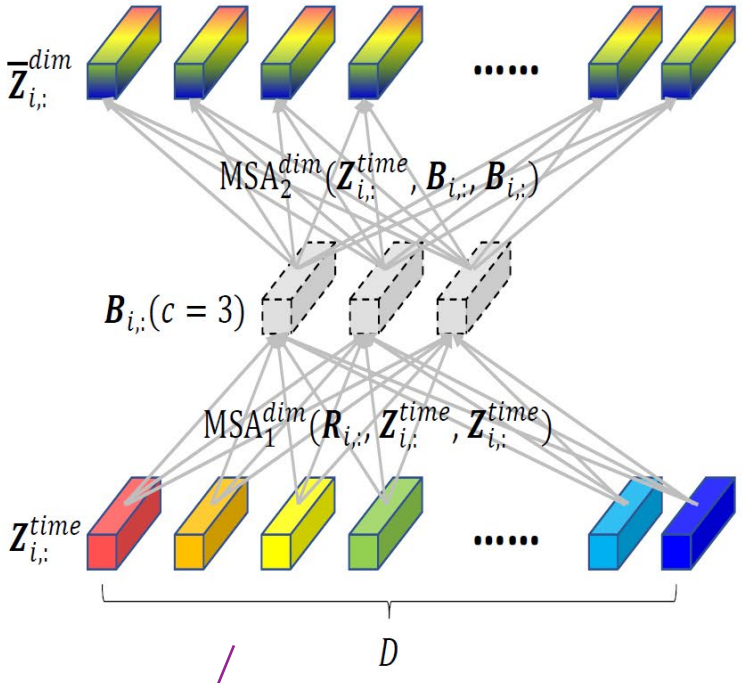
CrossFormer

Two-Stage Attention



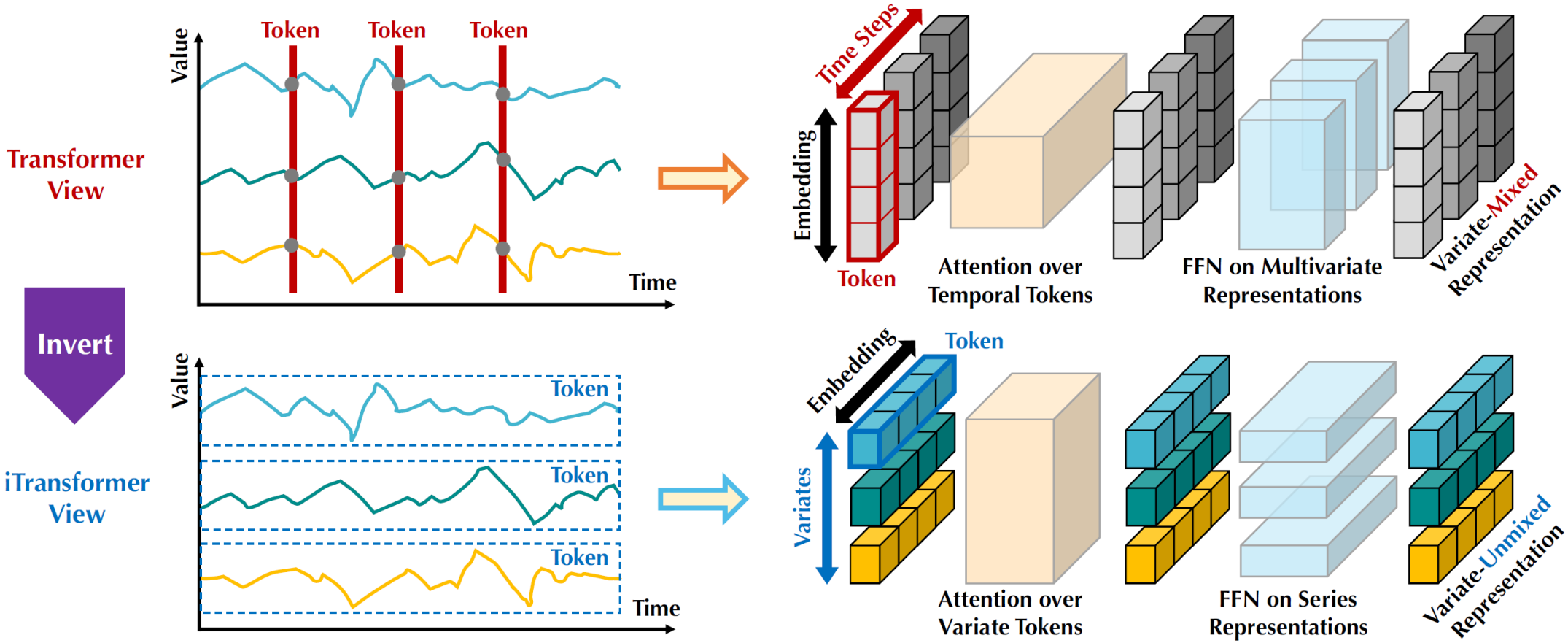
Directly using MSA in Cross-Dimension Stage to build the D-to-D connection results in $O(D^2)$ complexity





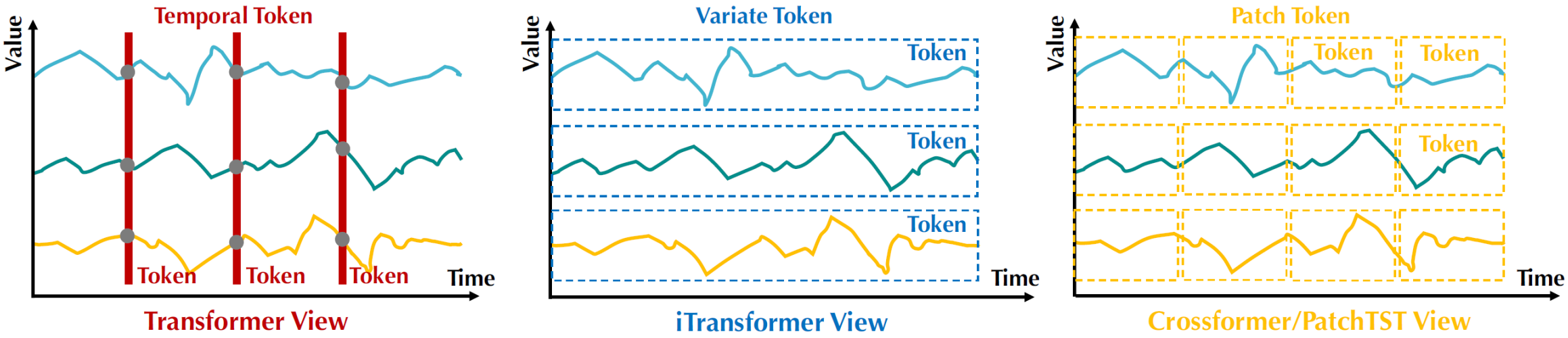
Router mechanism: a small fixed number (c) of "routers" gather information from all dimensions and then distribute the gathered information. The complexity is reduced to $O(2cD) = O(D)$.

iTransformer



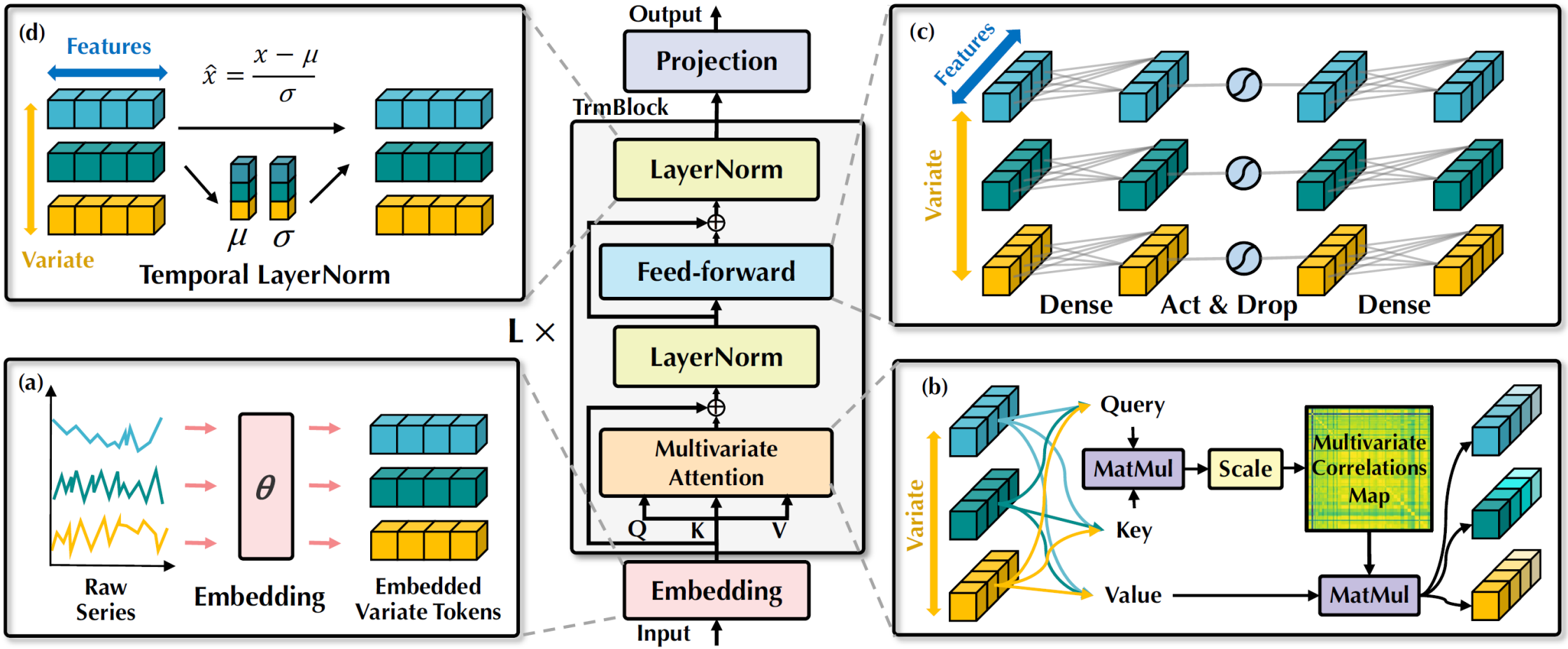
Yong Liu et al. **iTransformer: Inverted Transformers Are Effective for Time Series Forecasting**. CoRR abs/2310.06625 (2023)

iTransformer



- Transformer treats time series as the natural language but the time aligned embedding may bring about risks in multi-dimensional series. The problem can be alleviated by expanding the receptive field.
- Patching can be more fine-grained, it also brings higher computational complexity and the potential interaction noise between time-unaligned patches.

iTransformer



Cristian Challu et al. **NHITS: Neural Hierarchical Interpolation for Time Series Forecasting**. AAAI 2023: 6989-6997

iTransformer

Algorithm 1 iTransformer - Overall Architecture.

Require: Input lookback time series $\mathbf{X} \in \mathbb{R}^{T \times N}$; input Length T; predicted length S; variates number N; token dimension D; iTransformer block number L.

$\mathbf{X} = \mathbf{X}.\texttt{transpose}$ ▷ $\mathbf{X} \in \mathbb{R}^{N \times T}$

▷ Multi-layer Perceptron works on the last dimension to embed series into variate tokens.

$\mathbf{H}^0 = \texttt{MLP}(\mathbf{X})$ ▷ $\mathbf{H}^0 \in \mathbb{R}^{N \times D}$

for l **in** $\{1, \cdots, L\}$**:** ▷ Run through iTransformer blocks.

    ▷ Self-attention layer is applied on variate tokens.

    $\mathbf{H}^{l-1} = \texttt{LayerNorm}\big(\mathbf{H}^{l-1} + \texttt{Self-Attn}(\mathbf{H}^{l-1})\big)$ ▷ $\mathbf{H}^{l-1} \in \mathbb{R}^{N \times D}$

    ▷ Feed-forward network is utilized for series representations, broadcasting to each token.

    $\mathbf{H}^{l} = \texttt{LayerNorm}\big(\mathbf{H}^{l-1} + \texttt{Feed-Forward}(\mathbf{H}^{l-1})\big)$ ▷ $\mathbf{H}^{l} \in \mathbb{R}^{N \times D}$

    ▷ LayerNorm is adopted on series representations to reduce variates discrepancies.

End for

$\hat{\mathbf{Y}} = \texttt{MLP}(\mathbf{H}^L)$ ▷ Project tokens back to predicted series, $\hat{\mathbf{Y}} \in \mathbb{R}^{N \times S}$

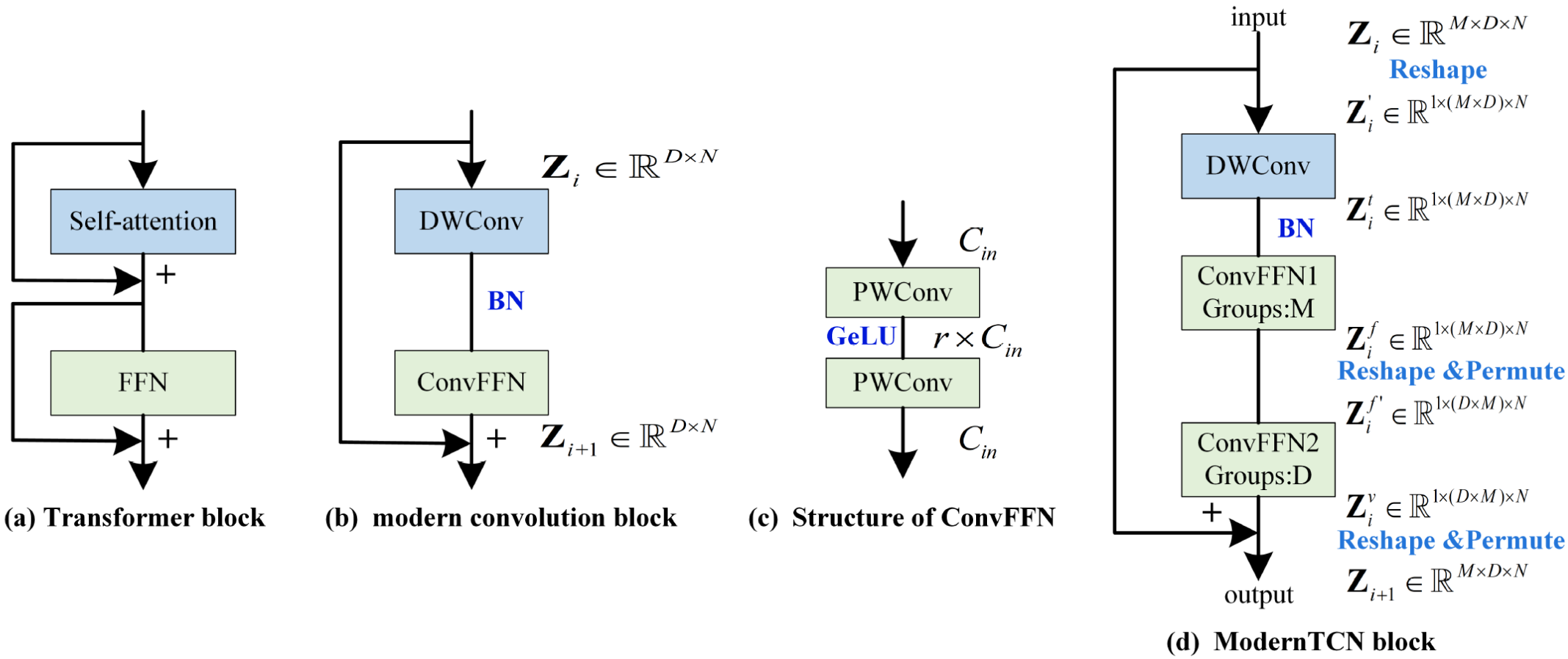
$\hat{\mathbf{Y}} = \hat{\mathbf{Y}}.\texttt{transpose}$ ▷ $\hat{\mathbf{Y}} \in \mathbb{R}^{S \times N}$

Return $\hat{\mathbf{Y}}$ ▷ Return the prediction result $\hat{\mathbf{Y}}$

iTransformer

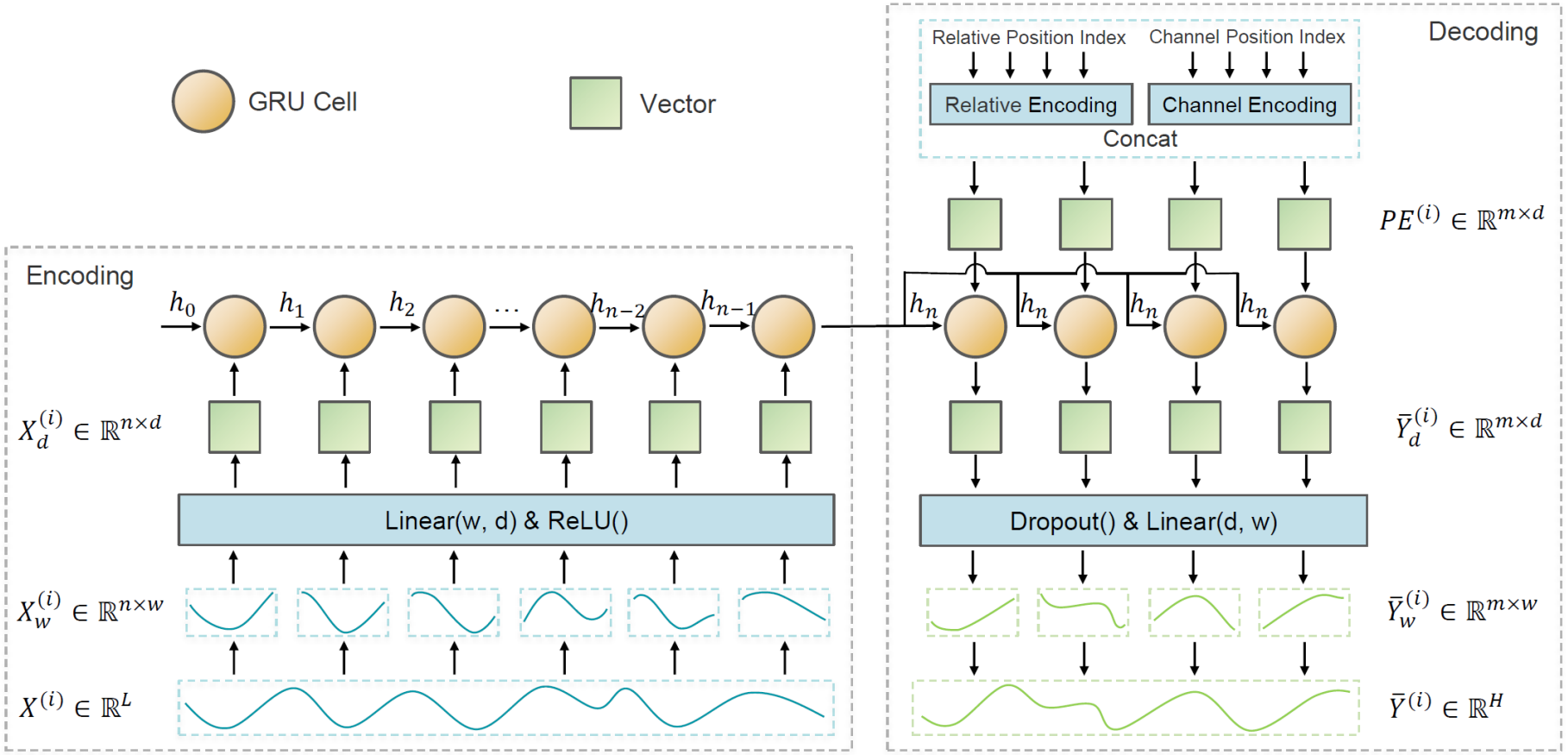
Models	**iTransformer (Ours)**		RLinear (2023)		PatchTST (2023)		Crossformer (2023)		TiDE (2023)		TimesNet (2023)		DLinear (2023)		SCINet (2022a)		FEDformer (2022)		Stationary (2022b)		Autoformer (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	**0.178**	**0.270**	0.219	0.298	0.216	0.304	0.244	0.334	0.251	0.344	0.192	0.295	0.212	0.300	0.268	0.365	0.214	0.327	0.193	0.296	0.227	0.338
ETT (Avg)	0.383	0.399	**0.380**	**0.392**	0.381	0.397	0.685	0.578	0.482	0.470	0.391	0.404	0.442	0.444	0.689	0.597	0.408	0.428	0.471	0.464	0.465	0.459
Exchange	0.360	**0.403**	0.378	0.417	0.367	0.404	0.940	0.707	0.370	0.413	0.416	0.443	**0.354**	0.414	0.750	0.626	0.519	0.429	0.461	0.454	0.613	0.539
Traffic	**0.428**	**0.282**	0.626	0.378	0.555	0.362	0.550	0.304	0.760	0.473	0.620	0.336	0.625	0.383	0.804	0.509	0.610	0.376	0.624	0.340	0.628	0.379
Weather	**0.258**	**0.279**	0.272	0.291	0.259	0.281	0.259	0.315	0.271	0.320	0.259	0.287	0.265	0.317	0.292	0.363	0.309	0.360	0.288	0.314	0.338	0.382
Solar-Energy	**0.233**	**0.262**	0.369	0.356	0.270	0.307	0.641	0.639	0.347	0.417	0.301	0.319	0.330	0.401	0.282	0.375	0.291	0.381	0.261	0.381	0.885	0.711
PEMS (Avg)	**0.119**	**0.218**	0.514	0.482	0.217	0.305	0.220	0.304	0.375	0.440	0.148	0.246	0.320	0.394	0.121	0.222	0.224	0.327	0.151	0.249	0.614	0.575

Modern TCN



ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis. ICLR 2024 submission.

SegRNN



Shengsheng Lin et al. **SegRNN: Segment Recurrent Neural Network for Long-Term Time Series Forecasting**. CoRR abs/2308.11200 (2023)