Beyond Channel Independence: New Paradigms for Time Series Analysis

2025.05.16

Data Mining & Quality Analytics Lab.

조광은

발표자 소개

❖ 조광은 (Kwangeun Cho)

- 고려대학교 산업경영공학과 Data Mining & Quality Analytics Lab.
- M.S. Student (2024.03 ~ Present)
- 지도교수: 김성범 교수님

❖ Research Interest

- Multivariate Time Series Forecasting
- Multivariate Time Series Anomaly Detection

❖ Contact

- E-mail: chopol98@korea.ac.kr

Contents

Background
Research Trend

❖ **Multivariate Time Series Forecasting using Transformer**

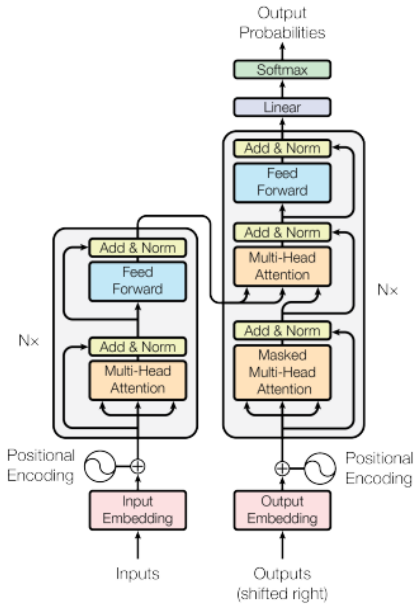
- 초기에는 RNN based model로 다변량 시계열 데이터 예측을 수행
- Transformer의 등장 이후, 자연어와 이미지 처리에서 뛰어난 성능을 보이는 Transformer 활용 방법론들이 등장

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

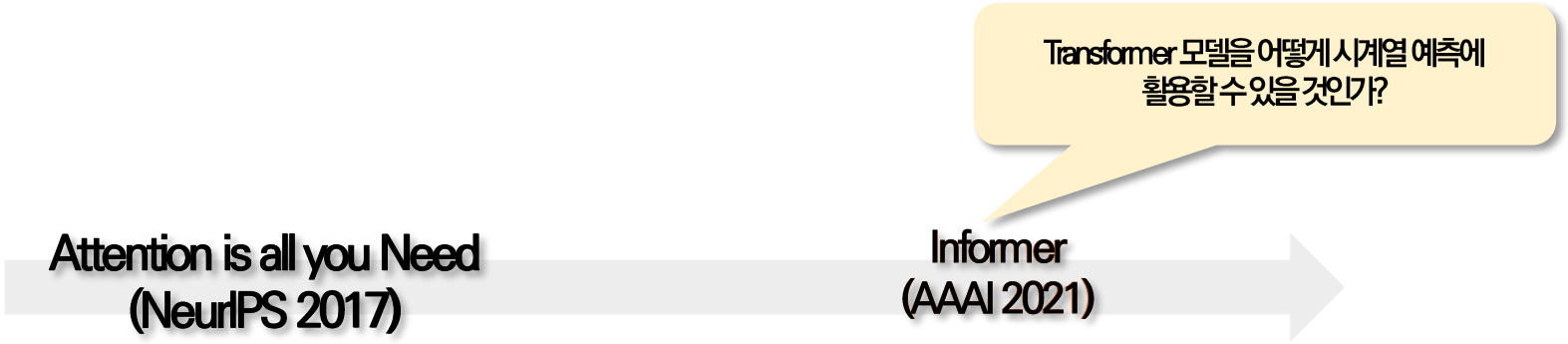
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.



Vaswani, A. (2017). Attention is all you need. Advances in Neural Information Processing Systems.

Background
Research Trend

❖ Research Trend after Transfomer

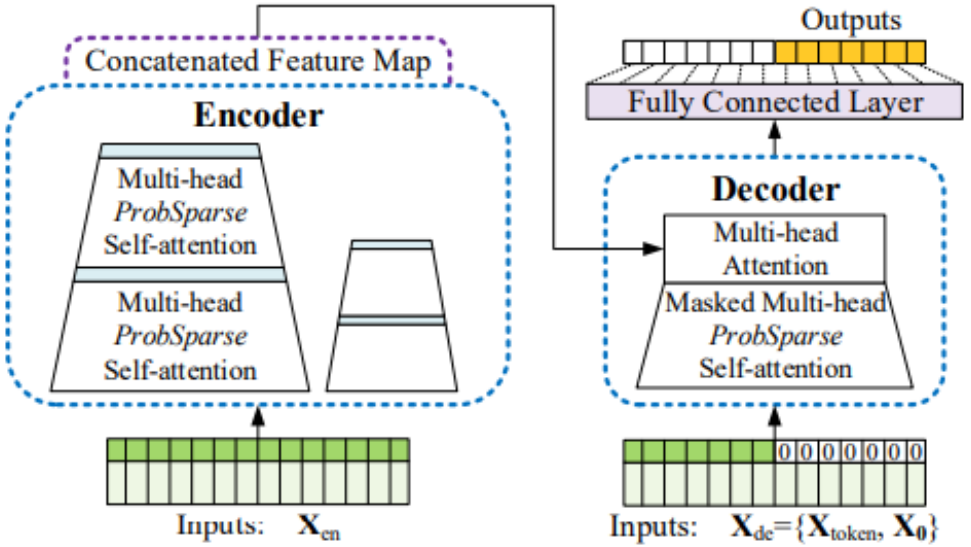


Background
Research Trend

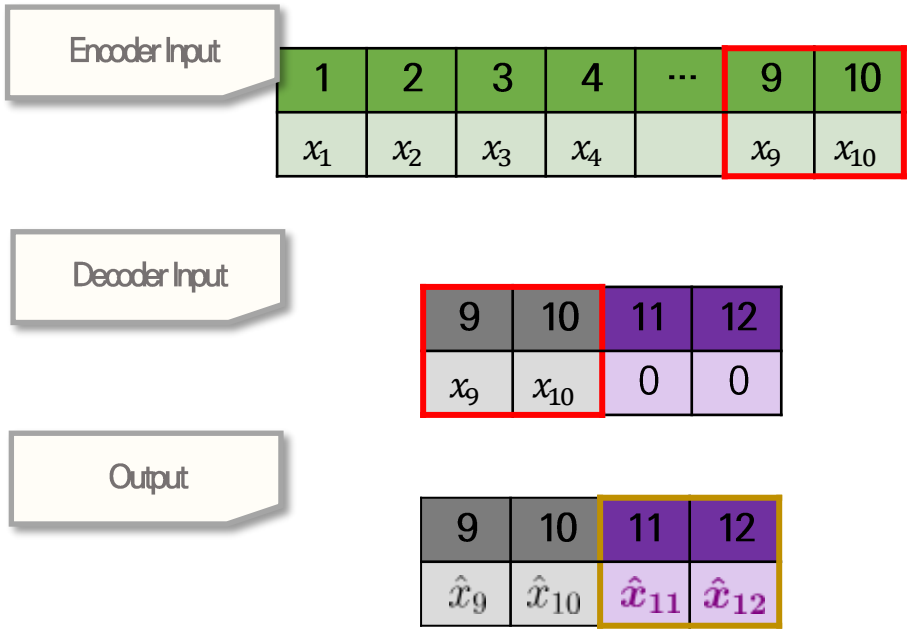
Transformer 모델을 어떻게 시계열 예측에 활용할 수 있을 것인가?

❖ **Informer: Beyond Efficient Transformer for Long-sequence Time Series Forecasting**

- 다변량 시계열 예측에 특화된 최초의 Transformer 모델
- 시계열 예측 task에 특화된 ProbSparse Attention 제안, Generative Style Decoder를 통한 Encoder-Decoder Architecture 제안

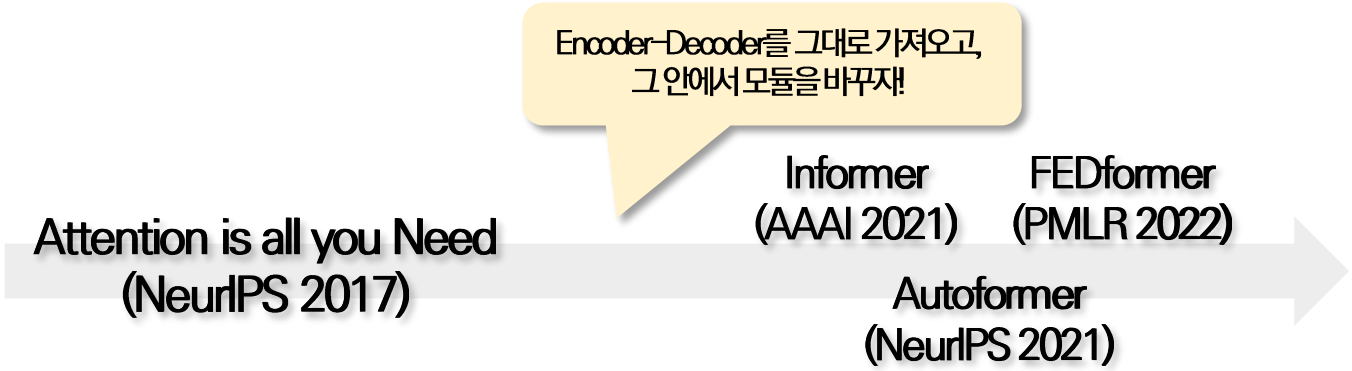


E..g. Input window 10, Label Length 2, Predict Length 2



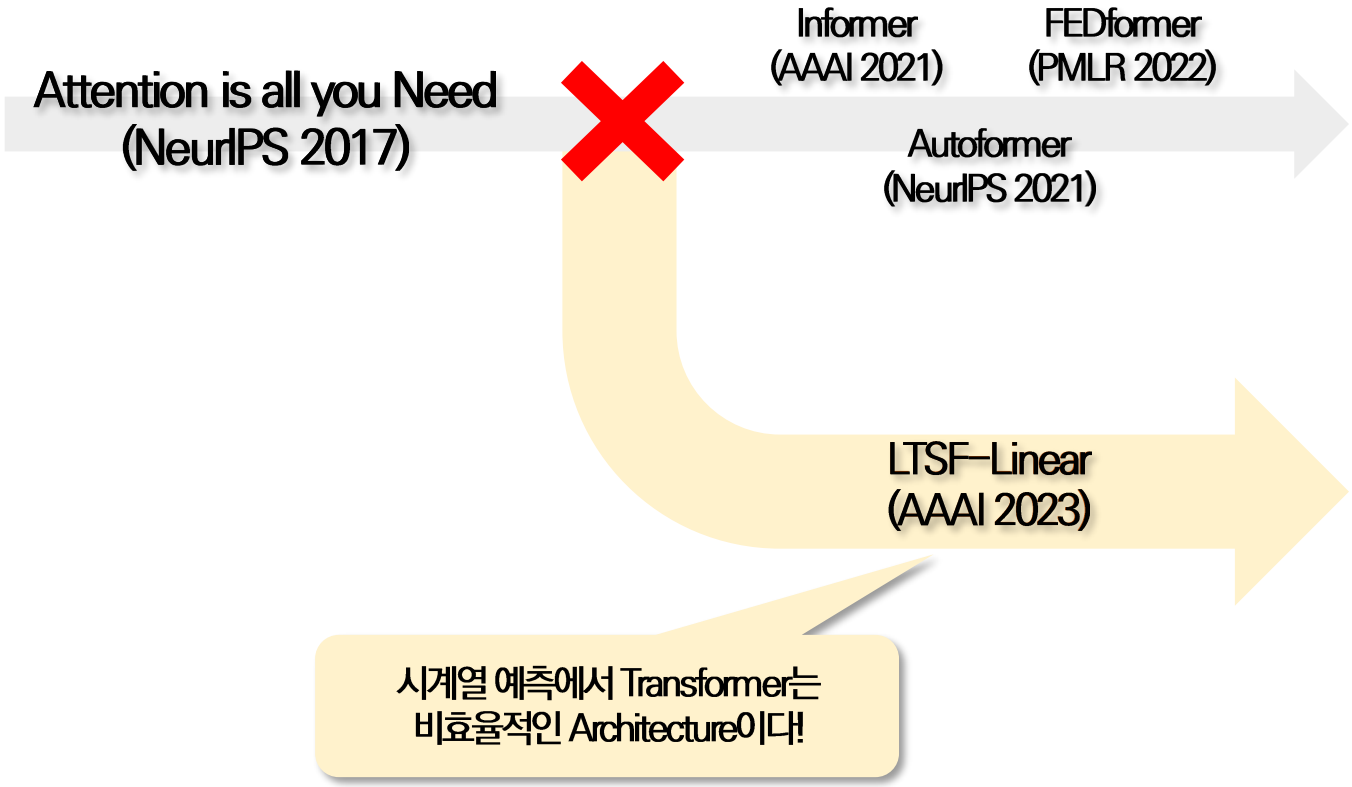
Background
Research Trend

❖ Research Trend after Transfomer



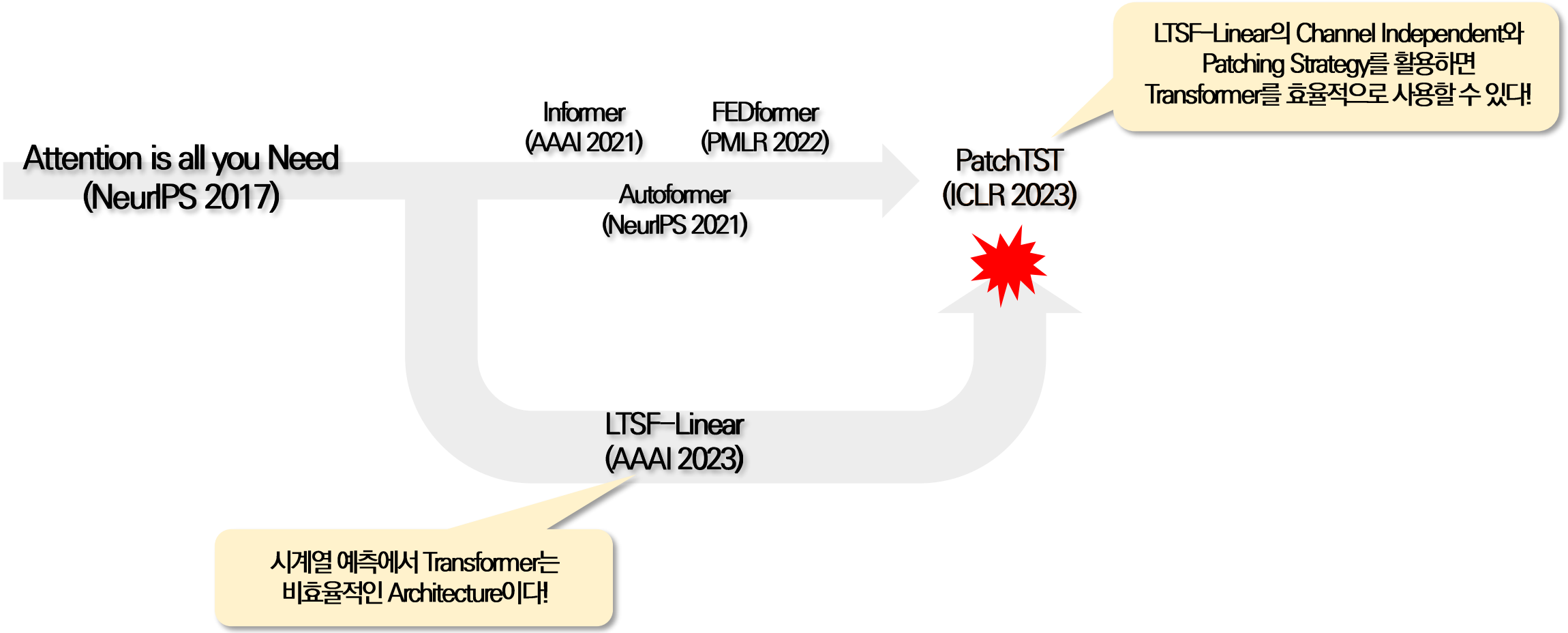
Background
Research Trend

❖ Research Trend after Transfomer



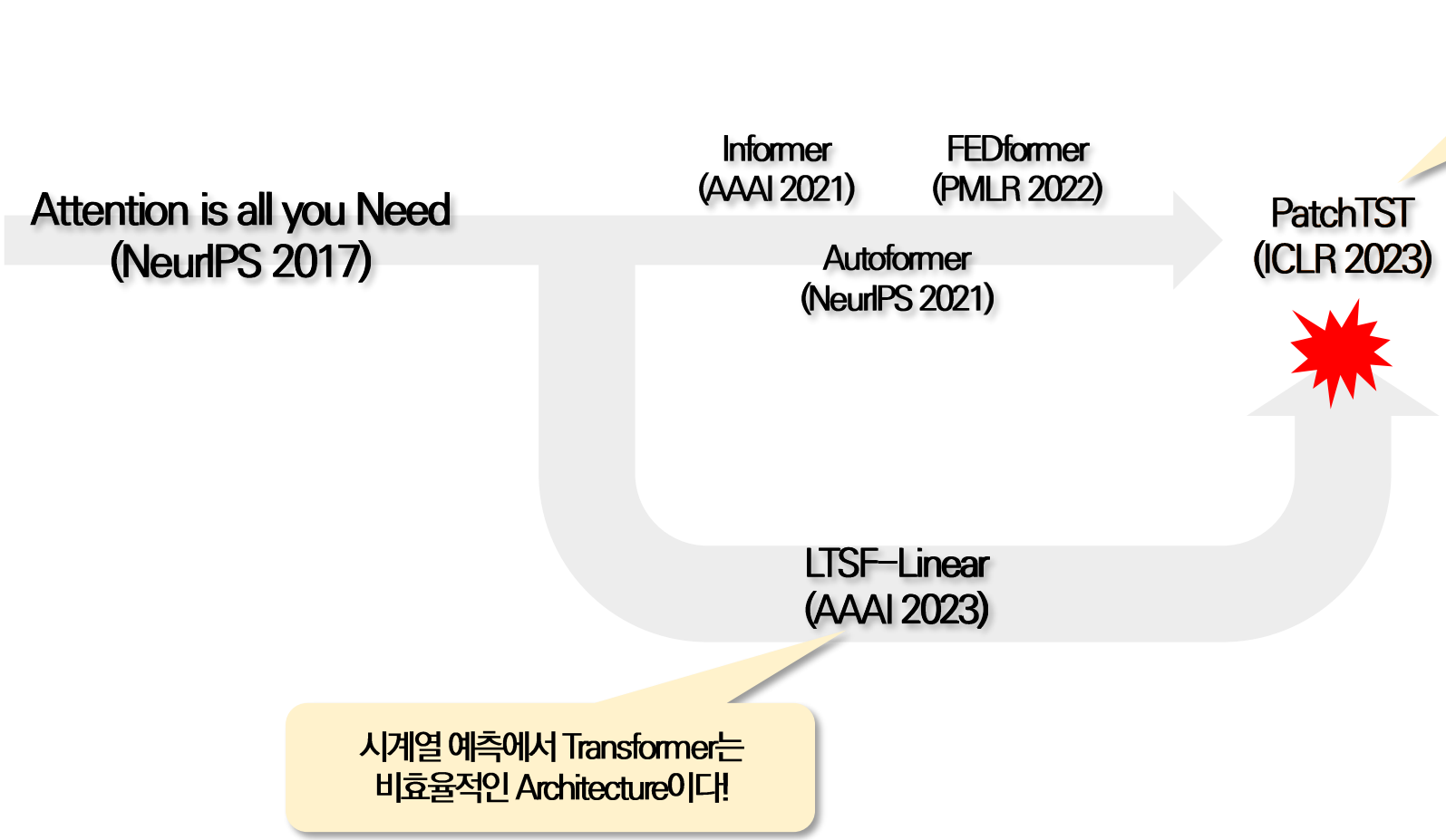
Background
Research Trend

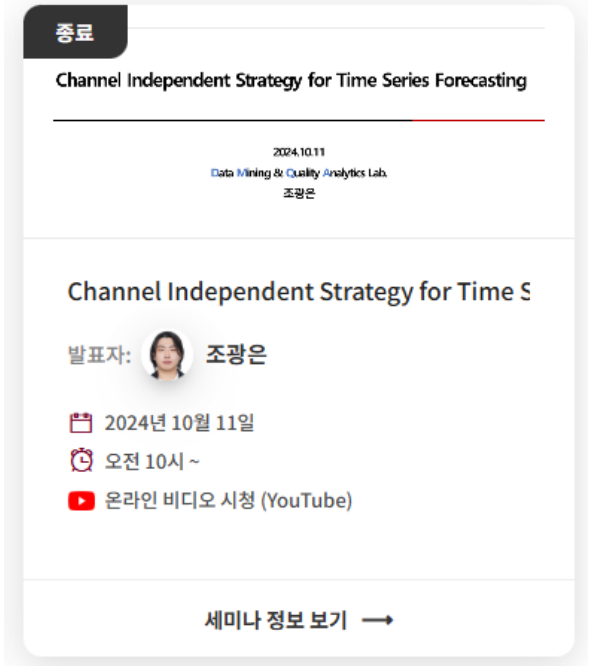
❖ Research Trend after Transfomer



Background
Research Trend

❖ Research Trend after Transfomer



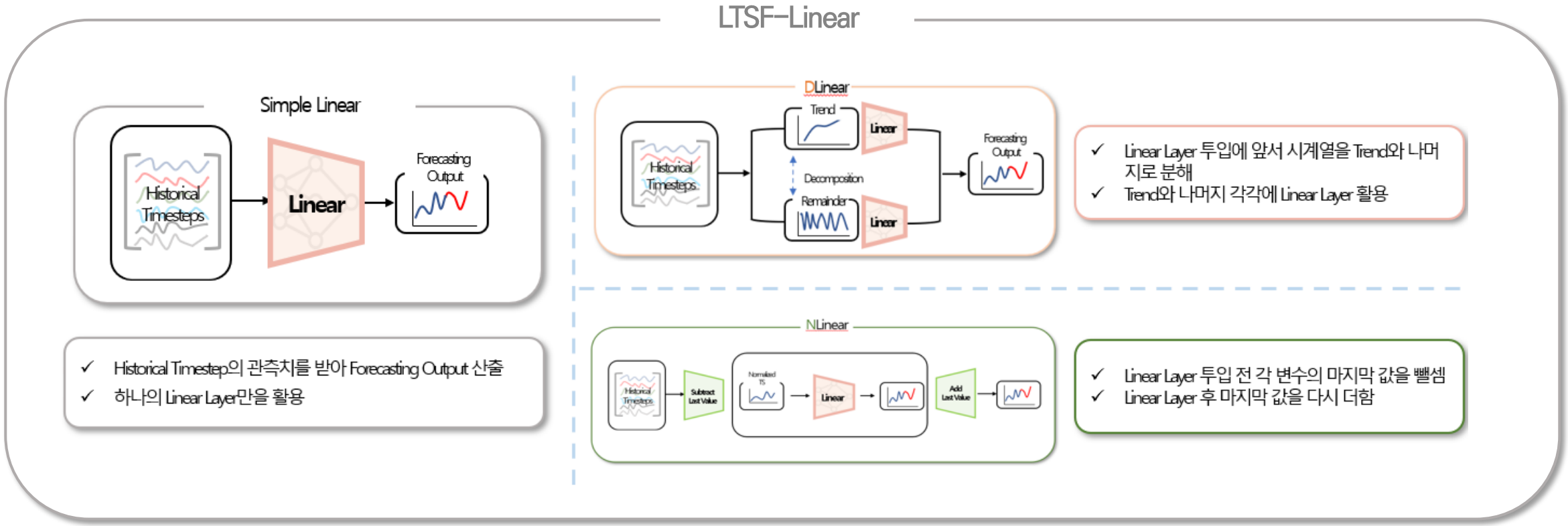


Background
Research Trend

시계열 예측에서 Transformer는 비효율적인 Architecture이다!

❖ **Are Transformers Effective for Time Series Forecasting?**

- 간단한 선형 레이어 모델이 복잡한 Transformer 구조의 성능을 넘음을 실험적으로 증명
- Linear Layer를 기반으로 한 방법론 제안, 다변량 시계열 예측에서 유의미한 예측 정확도 차이



Methods		IMP.	Linear*		NLinear*		DLinear*		FEDformer		Autoformer		Informer	
Metric		MSE	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
	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
	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
	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
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
	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
	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
	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
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
	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
	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
	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
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
	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
	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
	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
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
	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
	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
	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
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
	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
	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
	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
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
	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
	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
	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
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
	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
	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
	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
ETTm2	96	17.73%	0.168	0.262	**0.167**	**0.255**	**0.167**	0.260	0.203	0.287	0.255	0.339	0.365	0.453
	192	17.84%	0.232	0.308	**0.221**	**0.293**	0.224	0.303	0.269	0.328	0.281	0.340	0.533	0.563
	336	15.69%	0.320	0.373	**0.274**	**0.327**	0.281	0.342	0.325	0.366	0.339	0.372	1.363	0.887
	720	12.58%	0.413	0.435	**0.368**	**0.384**	0.397	0.421	0.421	0.415	0.433	0.432	3.379	1.338

시계열 예측에서 Transformer는
비효율적인 Architecture이다!

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2023, June). Are transformers effective for time series forecasting?. In Proceedings of the AAAI conference on artificial intelligence (Vol. 37, No. 9, pp. 11121-11128).

Background
Research Trend

LTSF-Linear의 Channel Independent와
Patching Strategy를 활용하면
Transformer를 효율적으로 사용할 수 있다!

❖ **A Time Series is Worth 64 Words: Long-term Forecasting with Transformers**

- Transformer Encoder만을 사용하여 다변량 시계열 예측에서 SOTA 성능 달성
- Channel Independent, Patching Strategy를 활용하여 Transformer를 fully utilize

Channel Independent

	변수1	변수2	···	변수K-1	변수K
시점1	12	15	···	24	26
시점2	26	22	···	45	56
···	···	···	···	···	···
시점L-1	36	45	···	48	55
시점L	65	62	···	34	24

Input Time Series $\in \mathbb{R}^{L \times K}$

변수 1	변수 2
12	15
26	22
···	···
36	45
65	62

변수 K-1	변수 K
24	26
45	56
···	···
48	55
34	24

Input Time Series $\in \mathbb{R}^{K \times L \times 1}$

Input을 만들 때 한 번에 다변량 시계열을 넣지 말고,
K개의 단변량 시계열처럼 취급하자!

Background

Research Trend

LTSF-Linear의 Channel Independent와
Patching Strategy를 활용하면
Transformer를 효율적으로 사용할 수 있다!

❖ A Time Series is Worth 64 Words: Long-term Forecasting with Transformers

- Transformer Encoder만을 사용하여 다변량 시계열 예측에서 SOTA 성능 달성
- Channel Independent, Patching Strategy를 활용하여 Transformer를 fully utilize

Channel Independent

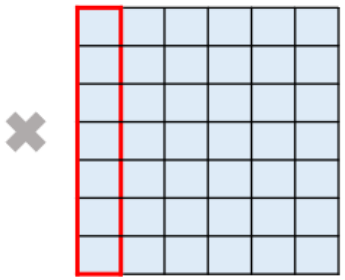
Why? LTSF-Linear에서도 변수들은 따로 고려되었기 때문!

시점 L개를 받아 H개를 예측

Simple Linear

	시점 1	시점2	...	시점 L-1	시점 L
변수1	12	15	...	24	26
변수2	26	22	...	45	56
변수3	65	100	...	85	82
...	...	...	...	...	...
변수 K-1	36	45	...	48	55
변수 K	65	62	...	34	24

Input Time Series $\in \mathbb{R}^{K \times L}$



Weight Matrix $\in \mathbb{R}^{L \times H}$

	시점 L+1	...	시점 L+H-1	시점 L+H
변수1	24	...	27	32
변수2	55	...	45	53
변수3	79	...	80	85
...	...	...	...	...
변수 K-1	52	...	43	40
변수 K	18	...	57	60

Output Time Series $\in \mathbb{R}^{K \times H}$

Input을 K개의 단변량 시계열처럼 취급하자!

Background
Research Trend

LTSF-Linear의 Channel Independent와
Patching Strategy를 활용하면
Transformer를 효율적으로 사용할 수 있다!

❖ **A Time Series is Worth 64 Words: Long-term Forecasting with Transformers**

- Transformer Encoder만을 사용하여 다변량 시계열 예측에서 SOTA 성능 달성
- Channel Independent, Patching Strategy를 활용하여 Transformer를 fully utilize

Channel Independent

	변수1	변수2	···	변수K-1	변수K
시점1	12	15	···	24	26
시점2	26	22	···	45	56
···	···	···	···	···	···
시점L-1	36	45	···	48	55
시점L	65	62	···	34	24

Input Time Series $\in \mathbb{R}^{L \times K}$

변수 1	변수 2
12	15
26	22
···	···
36	45
65	62

변수 K-1	변수 K
24	26
45	56
···	···
48	55
34	24

Input Time Series $\in \mathbb{R}^{K \times L \times 1}$

Patch

시점	1	2	3	4	5	6	7	8
값	T1	T2	T3	T4	T5	T6	T7	T8

T1	T3	T5	T7
T2	T4	T6	T8
T3	T5	T7	T8
T4	T6	T8	T8

Input을 만들 때 한 번에 다변량 시계열을 넣지 말고,
K개의 단변량 시계열처럼 취급하자!

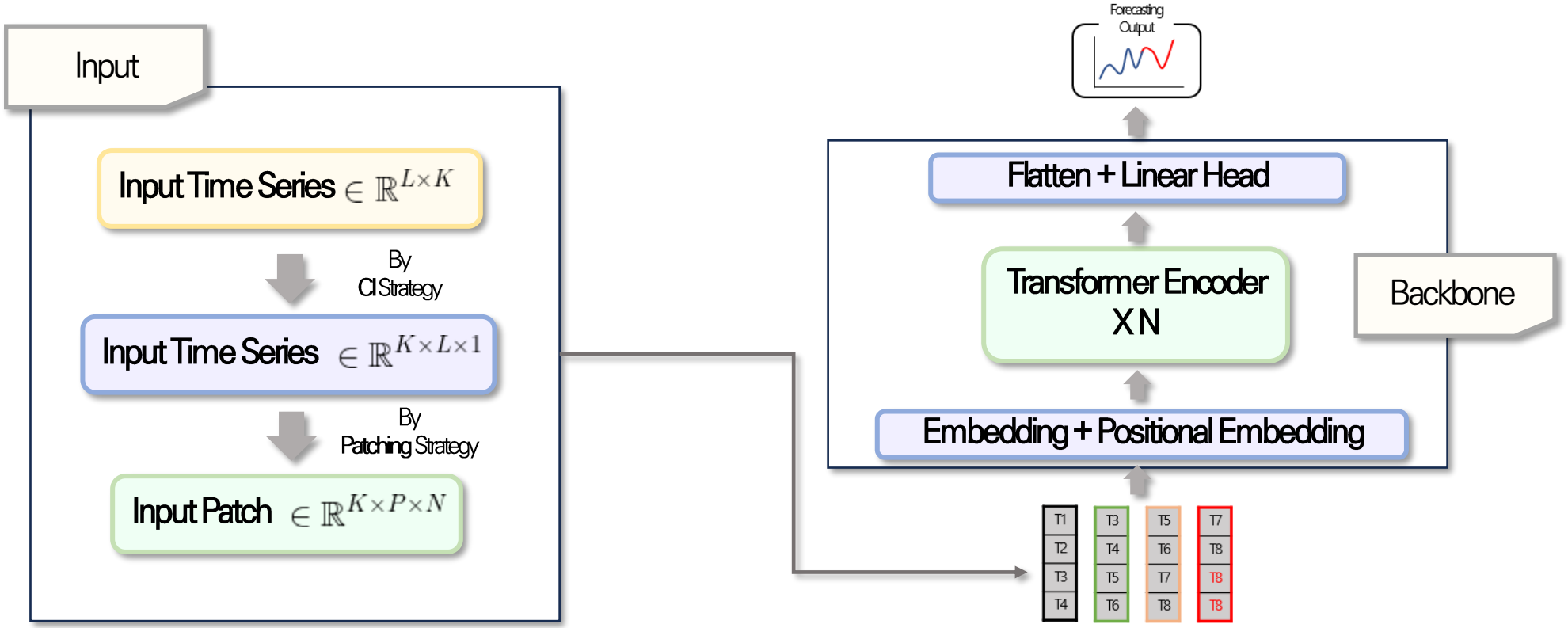
여러 개의 단변량 시계열을 지역적인 정보를 가지고 있는
Patch로 분할하자!

Background
Research Trend

LTSF-Linear의 Channel Independent와 Patching Strategy를 활용하면 Transformer를 효율적으로 사용할 수 있다!

❖ **A Time Series is Worth 64 Words: Long-term Forecasting with Transformers**

- Transformer Encoder만을 사용하여 다변량 시계열 예측에서 SOTA 성능 달성
- Channel Independent, Patching Strategy를 활용하여 Transformer를 fully utilize -> Transformer Encoder만을 사용!



Background

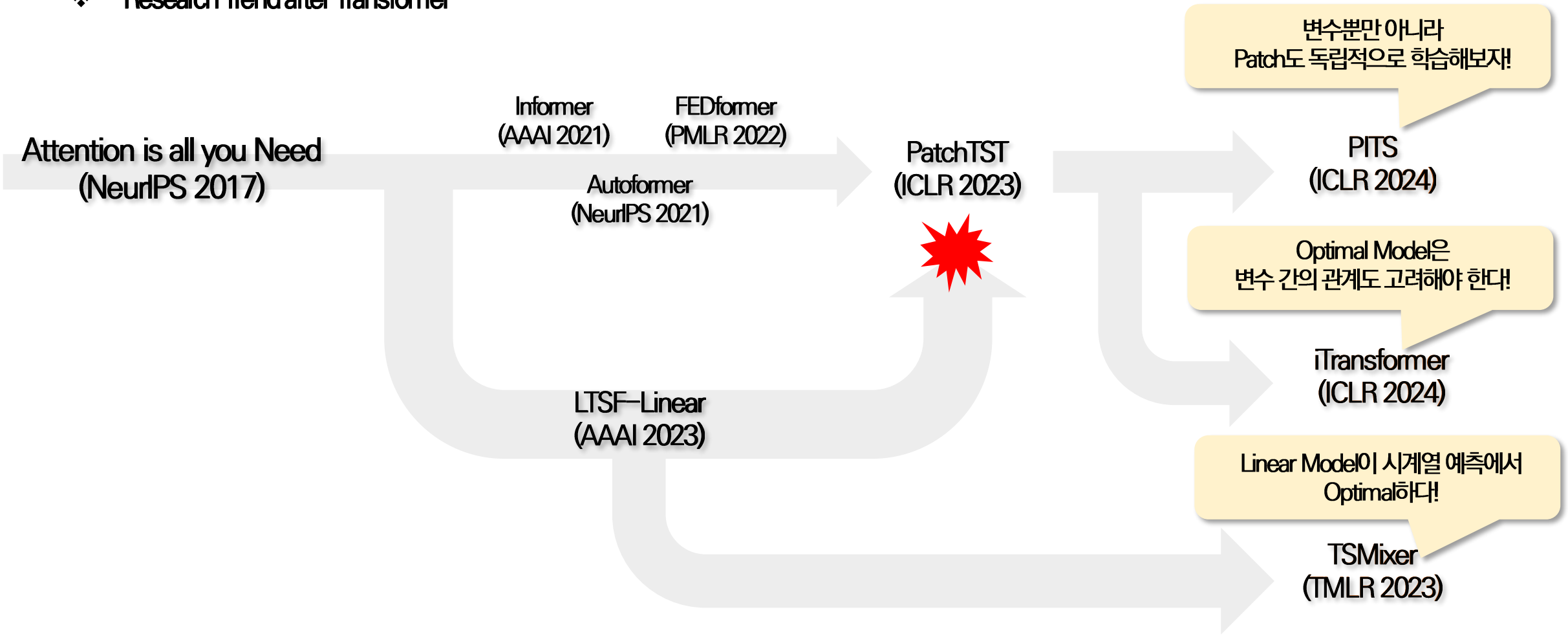
Research Trend

Models		PatchTST/64		PatchTST/42		DLinear		FEDformer		Autoformer		Informer		Pyraformer		LogTrans	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.149**	**0.198**	0.152	0.199	0.176	0.237	0.238	0.314	0.249	0.329	0.354	0.405	0.896	0.556	0.458	0.490
	192	**0.194**	**0.241**	0.197	0.243	0.220	0.282	0.275	0.329	0.325	0.370	0.419	0.434	0.622	0.624	0.658	0.589
	336	**0.245**	**0.282**	0.249	0.283	0.265	0.319	0.339	0.377	0.351	0.391	0.583	0.543	0.739	0.753	0.797	0.652
	720	**0.314**	**0.334**	0.320	0.335	0.323	0.362	0.389	0.409	0.415	0.426	0.916	0.705	1.004	0.934	0.869	0.675
Traffic	96	**0.360**	**0.249**	0.367	0.251	0.410	0.282	0.576	0.359	0.597	0.371	0.733	0.410	2.085	0.468	0.684	0.384
	192	**0.379**	**0.256**	0.385	0.259	0.423	0.287	0.610	0.380	0.607	0.382	0.777	0.435	0.867	0.467	0.685	0.390
	336	**0.392**	**0.264**	0.398	0.265	0.436	0.296	0.608	0.375	0.623	0.387	0.776	0.434	0.869	0.469	0.734	0.408
	720	**0.432**	**0.286**	0.434	0.287	0.466	0.315	0.621	0.375	0.639	0.395	0.827	0.466	0.881	0.473	0.717	0.396
Electricity	96	**0.129**	**0.222**	0.130	**0.222**	0.140	0.237	0.186	0.302	0.196	0.313	0.304	0.393	0.386	0.449	0.258	0.357
	192	**0.147**	**0.240**	0.148	**0.240**	0.153	0.249	0.197	0.311	0.211	0.324	0.327	0.417	0.386	0.443	0.266	0.368
	336	**0.163**	**0.259**	0.167	0.261	0.169	0.267	0.213	0.328	0.214	0.327	0.333	0.422	0.378	0.443	0.280	0.380
	720	**0.197**	**0.290**	0.202	0.291	0.203	0.301	0.233	0.344	0.236	0.342	0.351	0.427	0.376	0.445	0.283	0.376
ILI	24	**1.319**	**0.754**	1.522	0.814	2.215	1.081	2.624	1.095	2.906	1.182	4.657	1.449	1.420	2.012	4.480	1.444
	36	1.579	0.870	**1.430**	**0.834**	1.963	0.963	2.516	1.021	2.585	1.038	4.650	1.463	7.394	2.031	4.799	1.467
	48	**1.553**	**0.815**	1.673	0.854	2.130	1.024	2.505	1.041	3.024	1.145	5.004	1.542	7.551	2.057	4.800	1.468
	60	**1.470**	**0.788**	1.529	0.862	2.368	1.096	2.742	1.122	2.761	1.114	5.071	1.543	7.662	2.100	5.278	1.560
ETTh1	96	**0.370**	0.400	0.375	**0.399**	0.375	**0.399**	0.376	0.415	0.435	0.446	0.941	0.769	0.664	0.612	0.878	0.740
	192	0.413	0.429	0.414	0.421	**0.405**	**0.416**	0.423	0.446	0.456	0.457	1.007	0.786	0.790	0.681	1.037	0.824
	336	**0.422**	0.440	0.431	**0.436**	0.439	0.443	0.444	0.462	0.486	0.487	1.038	0.784	0.891	0.738	1.238	0.932
	720	**0.447**	0.468	0.449	**0.466**	0.472	0.490	0.469	0.492	0.515	0.517	1.144	0.857	0.963	0.782	1.135	0.852
ETTh2	96	**0.274**	0.337	**0.274**	**0.336**	0.289	0.353	0.332	0.374	0.332	0.368	1.549	0.952	0.645	0.597	2.116	1.197
	192	0.341	0.382	**0.339**	**0.379**	0.383	0.418	0.407	0.446	0.426	0.434	3.792	1.542	0.788	0.683	4.315	1.635
	336	**0.329**	0.384	0.331	**0.380**	0.448	0.465	0.400	0.447	0.477	0.479	4.215	1.642	0.907	0.747	1.124	1.604
	720	**0.379**	**0.422**	**0.379**	**0.422**	0.605	0.551	0.412	0.469	0.453	0.490	3.656	1.619	0.963	0.783	3.188	1.540
ETTm1	96	0.293	0.346	**0.290**	**0.342**	0.299	0.343	0.326	0.390	0.510	0.492	0.626	0.560	0.543	0.510	0.600	0.546
	192	0.333	0.370	**0.332**	0.369	0.335	**0.365**	0.365	0.415	0.514	0.495	0.725	0.619	0.557	0.537	0.837	0.700
	336	0.369	0.392	**0.366**	0.392	0.369	**0.386**	0.392	0.425	0.510	0.492	1.005	0.741	0.754	0.655	1.124	0.832
	720	**0.416**	**0.420**	0.420	0.424	0.425	0.421	0.446	0.458	0.527	0.493	1.133	0.845	0.908	0.724	1.153	0.820
ETTm2	96	0.166	0.256	**0.165**	**0.255**	0.167	0.260	0.180	0.271	0.205	0.293	0.355	0.462	0.435	0.507	0.768	0.642
	192	0.223	0.296	**0.220**	**0.292**	0.224	0.303	0.252	0.318	0.278	0.336	0.595	0.586	0.730	0.673	0.989	0.757
	336	**0.274**	**0.329**	0.278	**0.329**	0.281	0.342	0.324	0.364	0.343	0.379	1.270	0.871	1.201	0.845	1.334	0.872
	720	**0.362**	**0.385**	0.367	**0.385**	0.397	0.421	0.410	0.420	0.414	0.419	3.001	1.267	3.625	1.451	3.048	1.328

LTSF-Linear의 Channel Independent와
Patching Strategy를 활용하면
Transformer를 효율적으로 사용할 수 있다!

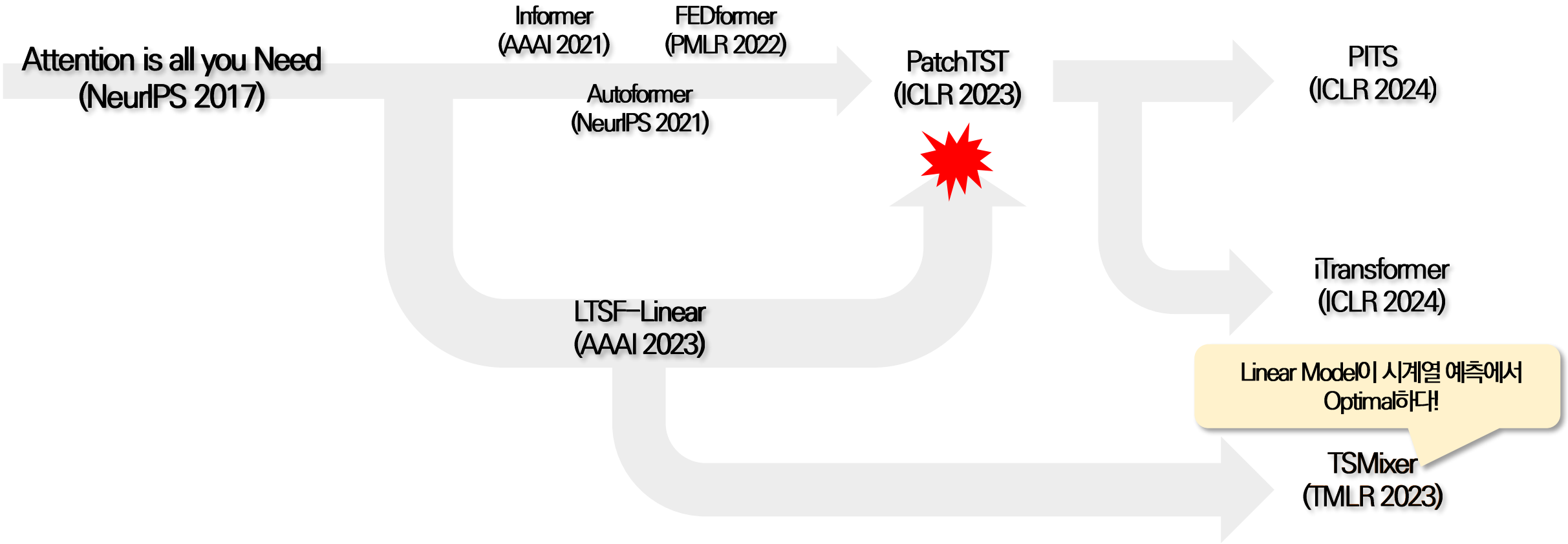
Background
Research Trend

❖ Research Trend after Transfomer



Background
Research Trend

❖ Research Trend after Transfomer



Method

TSMixer: An all-mlp architecture for time series forecasting

❖ **TSMixer: An all-mlp architecture for time series forecasting(TMLR 2023)**

- 2025년 5월 기준 289회 인용
- 시간 정보를 포착하는 데 Linear model이 효율적임을 보여줌으로써, 새로운 MLP 아키텍처로 SOTA 성능 달성

TSMixer: An All-MLP Architecture for Time Series Forecasting

Si-An Chen d09922007@ntu.edu.tw
National Taiwan University
Google Cloud AI Research

Chun-Liang Li chunliang@google.com
Google Cloud AI Research

Nathanael C. Yoder nyoder@google.com
Google Cloud AI Research

Sercan Ö. Arık soarik@google.com
Google Cloud AI Research

Tomas Pfister tpfister@google.com
Google Cloud AI Research

Reviewed on OpenReview: *https://openreview.net/forum?id=wbpzTuIgm0*

Abstract

Real-world time-series datasets are often multivariate with complex dynamics. To capture this complexity, high capacity architectures like recurrent- or attention-based sequential deep learning models have become popular. However, recent work demonstrates that simple univariate linear models can outperform such deep learning models on several commonly used academic benchmarks. Extending them, in this paper, we investigate the capabilities of linear models for time-series forecasting and present Time-Series Mixer (TSMixer), a novel architecture designed by stacking multi-layer perceptrons (MLPs). TSMixer is based on mixing operations along both the time and feature dimensions to extract information efficiently. On popular academic benchmarks, the simple-to-implement TSMixer is comparable to specialized state-of-the-art models that leverage the inductive biases of specific benchmarks. On the challenging and large scale M5 benchmark, a real-world retail dataset, TSMixer demonstrates superior performance compared to the state-of-the-art alternatives. Our results underline the importance of efficiently utilizing cross-variate and auxiliary information for improving the performance of time series forecasting. We present various analyses to shed light into the capabilities of TSMixer. The design paradigms utilized in TSMixer are expected to open new horizons for deep learning-based time series forecasting. The implementation is available at: https://github.com/google-research/google-research/tree/master/tsmixer.

LTSF-Linear

실험적으로 봤더니 Linear Model이 잘 나오더라?

TSMixer

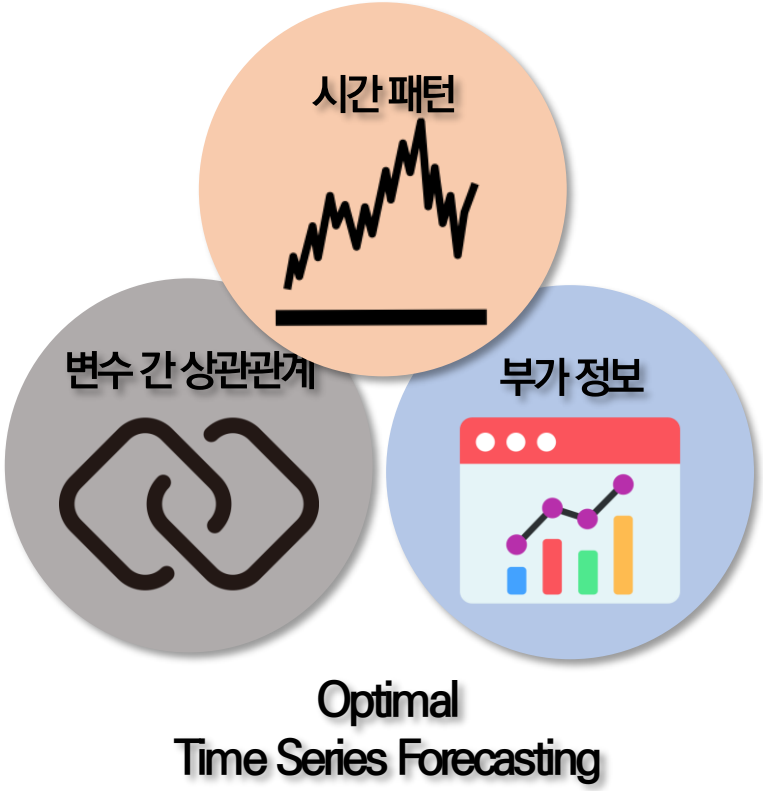
그러면 왜 잘되고, 어떻게 Linear 모델의 표현력을 극대화할 수 있을까?

Method

TSMixer: An all-mlp architecture for time series forecasting

❖ **Three key aspects of Time Series Predictability**

- 시계열 예측 가능성을 결정 짓는 요인으로 세 가지 측면(Temporal pattern, Cross-variate Information, Auxiliary features)을 제시
- Temporal pattern: 추세와 계절 패턴 / Cross-variate information: 변수들 간의 상관 관계 / Auxiliary feature: 부가 정보들

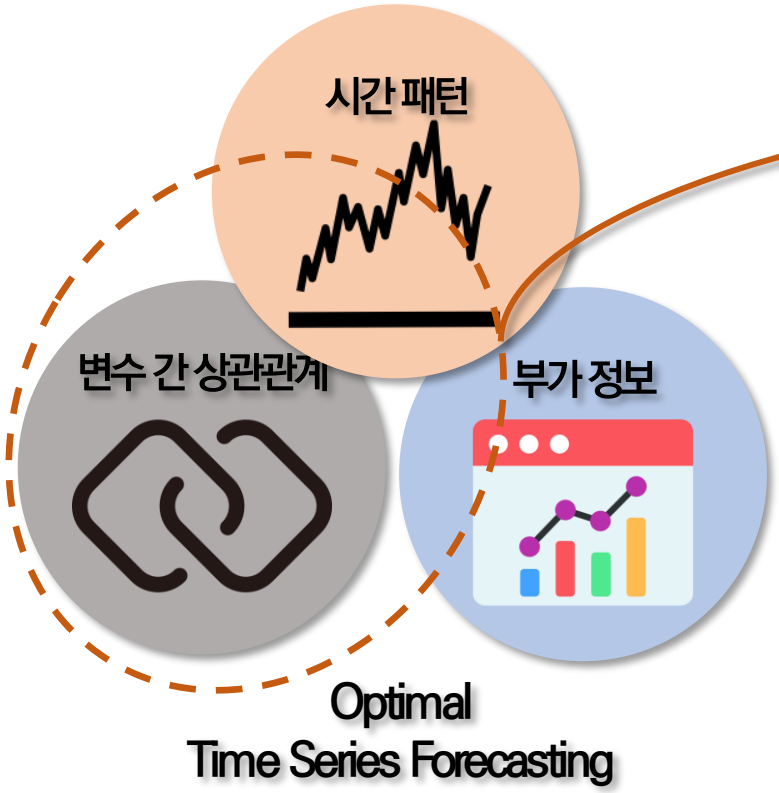


Method

TSMixer: An all-mlp architecture for time series forecasting

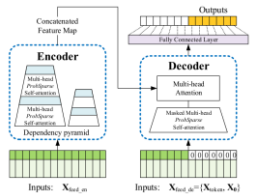
❖ **Three key aspects of Time Series Predictability**

- 시계열 예측 가능성을 결정 짓는 요인으로 세 가지 측면(Temporal pattern, Cross-variate Information, Auxiliary features)을 제시
- Temporal pattern: 추세와 계절 패턴 / Cross-variate information: 변수들 간의 상관 관계 / Auxiliary feature: 부가 정보들



Transformer-based Model

변수 간 상관 관계를 고려하나
Temporal Information 포착이 떨어짐 -> 성능 하락



Method

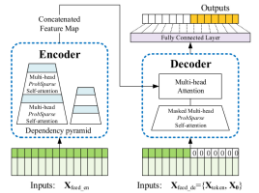
TSMixer: An all-mlp architecture for time series forecasting

❖ **Three key aspects of Time Series Predictability**

- 시계열 예측 가능성을 결정 짓는 요인으로 세 가지 측면(Temporal pattern, Cross-variate Information, Auxiliary features)을 제시
- Temporal pattern: 추세와 계절 패턴 / Cross-variate information: 변수들 간의 상관 관계 / Auxiliary feature: 부가 정보들

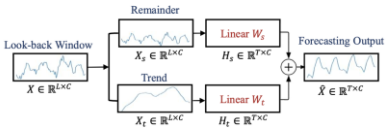
Transformer-based Model

변수 간 상관 관계를 고려하나
Temporal Information 포착이 떨어짐 -> 성능 하락



Linear Model + CI

Linear model은 **Temporal Information 포착에서 뛰어남**

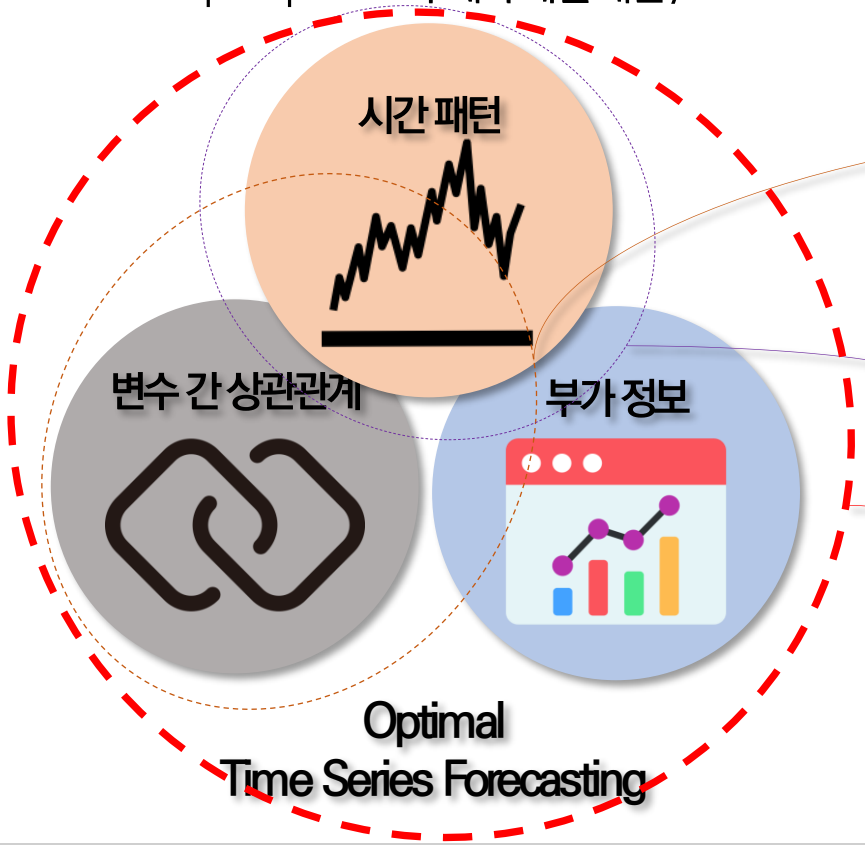


Method

TSMixer: An all-mlp architecture for time series forecasting

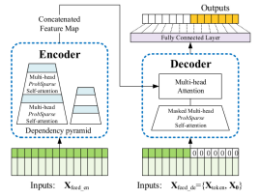
❖ **Three key aspects of Time Series Predictability**

- 시계열 예측 가능성을 결정 짓는 요인으로 세 가지 측면(Temporal pattern, Cross-variate Information, Auxiliary features)을 제시
- Temporal pattern: 추세와 계절 패턴 / Cross-variate information: 변수들 간의 상관 관계 / Auxiliary feature: 부가 정보들



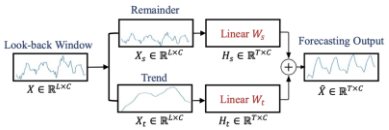
Transformer-based Model

변수 간 상관 관계를 고려하나
Temporal Information 포착이 떨어짐 -> 성능 하락



Linear Model + CI

Linear model은 **Temporal Information 포착**에서 **뛰어남**



TSMixer

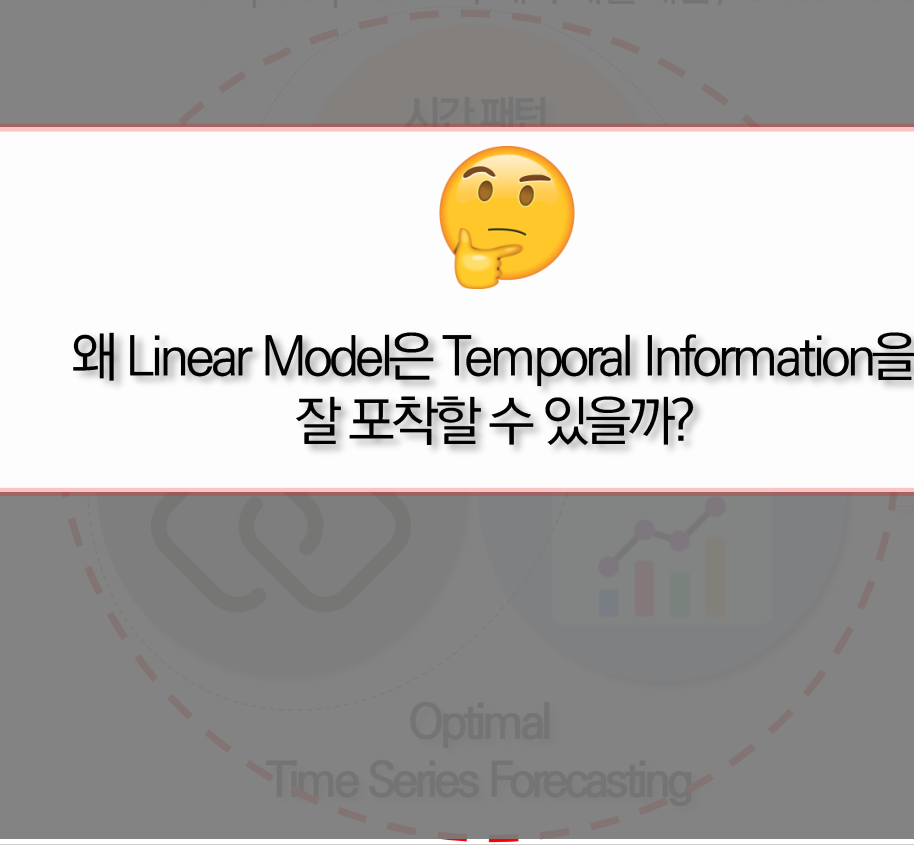
Temporal Information을 잘 포착하는 Linear Model을 기반으로
변수 간 상관 관계와 부가 정보까지 고려

Method

TSMixer: An all-mlp architecture for time series forecasting

❖ Three key aspects of Time Series Predictability

- 시계열 예측 가능성을 결정 짓는 요인으로 세 가지 측면(Temporal pattern, Cross-variate Information, Auxiliary features)을 제시
- Temporal pattern: 추세와 계절 패턴 / Cross-variate information: 변수들 간의 상관 관계 / Auxiliary feature: 부가 정보들



🤔

왜 Linear Model은 Temporal Information을
잘 포착할 수 있을까?

Transformer-based Model

변수 간 상관 관계를 고려하나
Temporal Information 포착이 떨어짐 -> 성능 하락

Linear Model

Linear model은 Temporal Information 포착에서
뛰어남

TSMixer

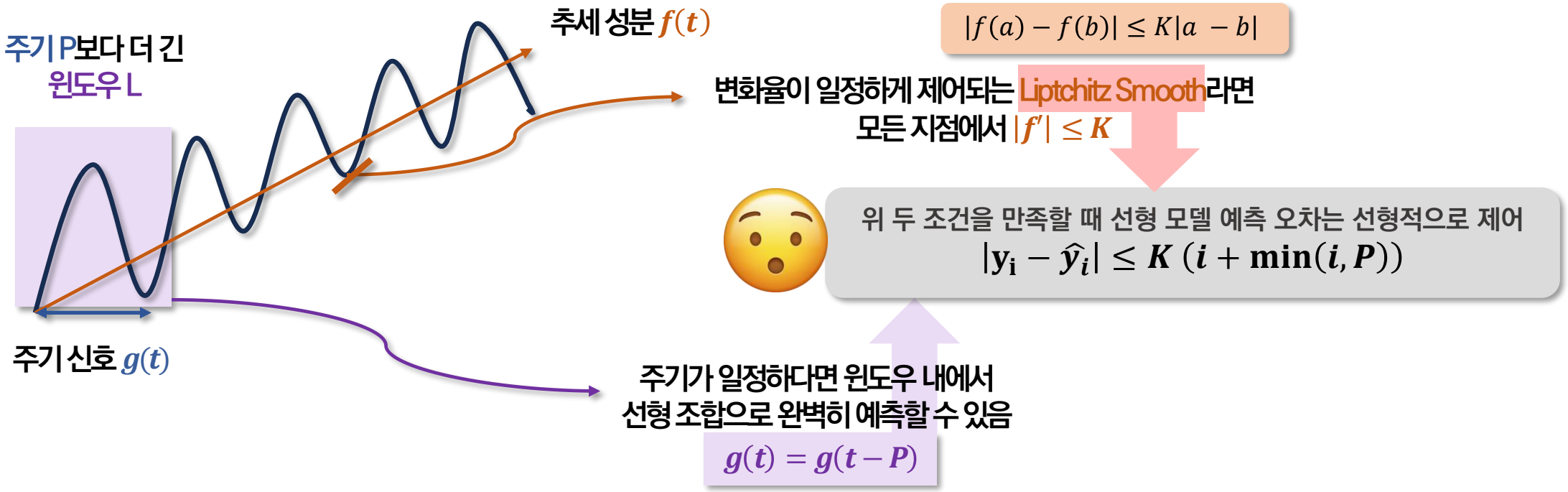
Temporal Information을 잘 포착하는 Linear Model을 기반으로
변수 간 상관 관계와 부가 정보까지 고려

Method

TSMixer: An all-mlp architecture for time series forecasting

❖ **Linear Model's advantages of capturing temporal information**

- Lipschitz smoothness가 보장되는 시계열과 충분한 window size 하에서 모델의 오차는 **선형적으로 제어**
- 시계열의 추세가 부드럽게 변한다는 일반적 가정 하에, **Linear Model은 가장 안정적인 솔루션임**



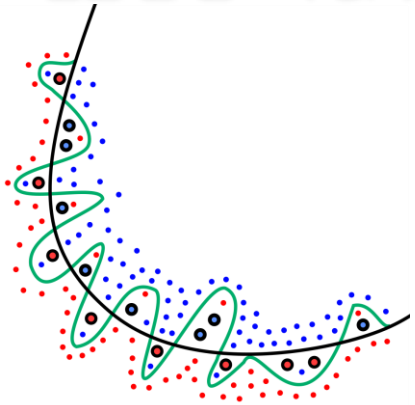
Method

TSMixer: An all-mlp architecture for time series forecasting

❖ **Linear Model's advantages of capturing temporal information**

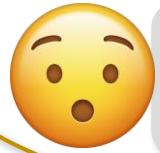
- Lipschitz smoothness가 보장되는 시계열과 충분한 window size 하에서 모델의 오차는 **선형적으로 제어**
- 시계열의 추세가 부드럽게 변한다는 일반적 가정 하에, **Linear Model은 가장 안정적인 솔루션임**

굳이 Temporal Information을 포착하기 위해서 복잡한 모델을 쓸 필요가 없다!



오히려 Overfitting 문제(오차가 안정적으로 제어 X)로 성능 하락할 수 있음!

Why? 일반적인 가정에서 Linear Model을 써도 오차가 제어가 되니까…



위 두 조건을 만족할 때 선형 모델 예측 오차는 선형적으로 제어

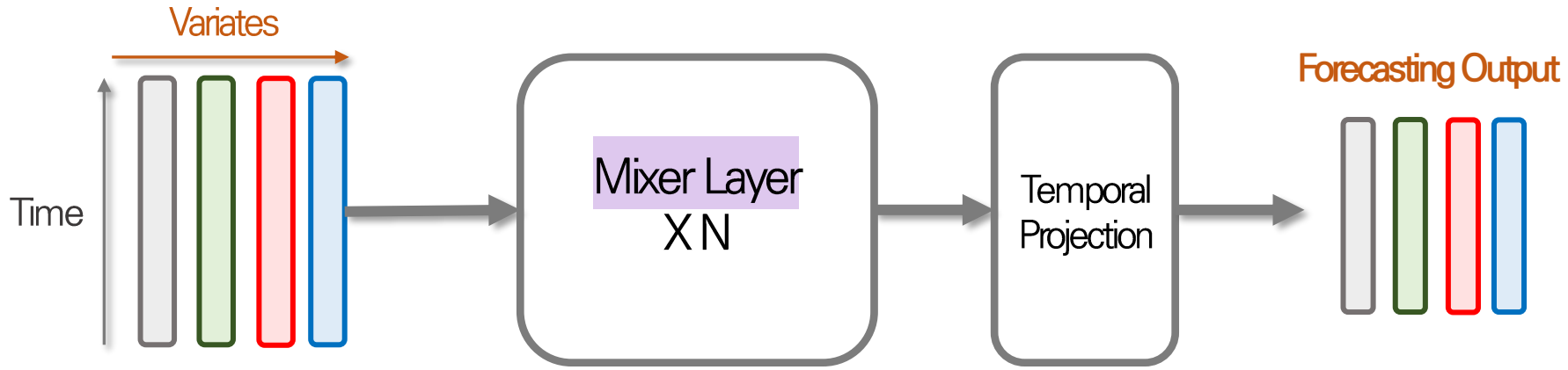
$$|\mathbf{y_i} - \hat{y}_i| \leq K\,(i + \min(i, P))$$

Method

TSMixer: An all-mlp architecture for time series forecasting

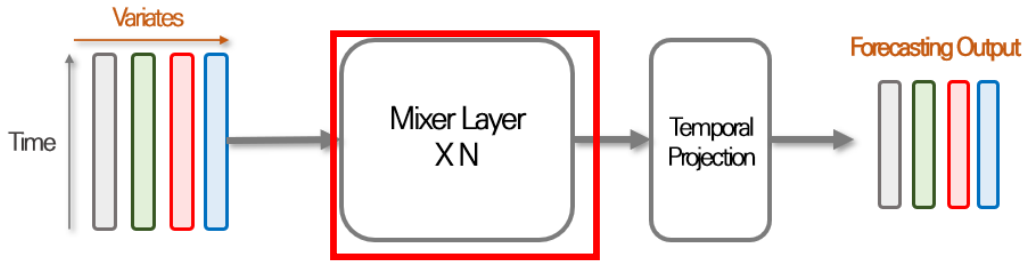
❖ TSMixer Architecture: Overall

- 이론적으로 Linear Model은 temporal information을 잡는 데 능하다 + 변수 간의 관계도 고려해야 한다 = **MLP-Mixer 사용**
- Temporal Information을 잡아내는 **Time-Mixer, 변수 간 상관 관계를 포착하는 Feature-Mixer 사용**



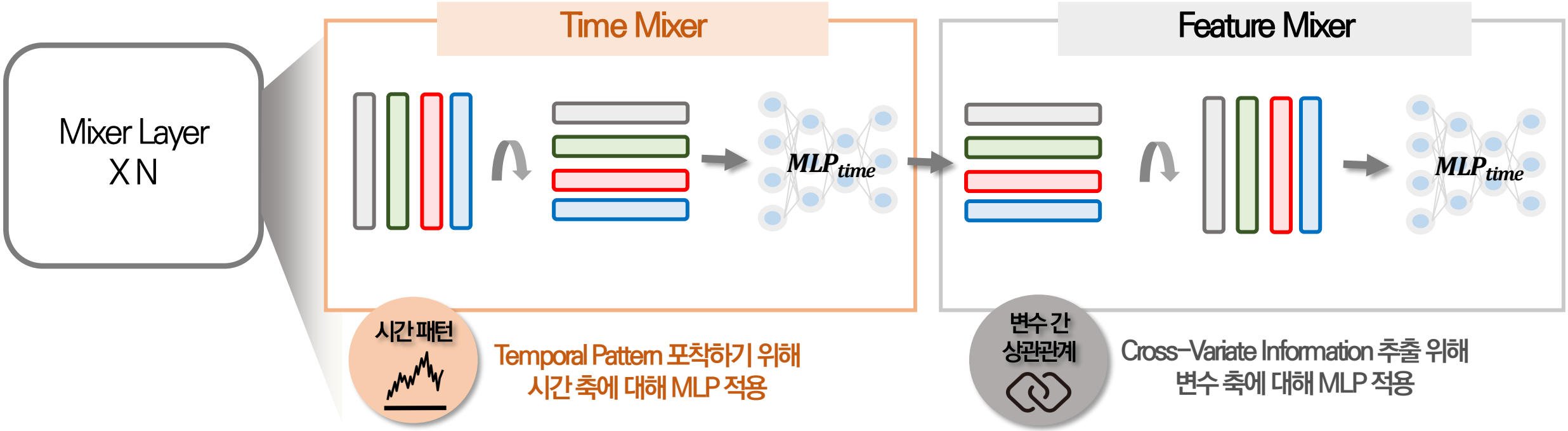
Method

TSMixer: An all-mlp architecture for time series forecasting



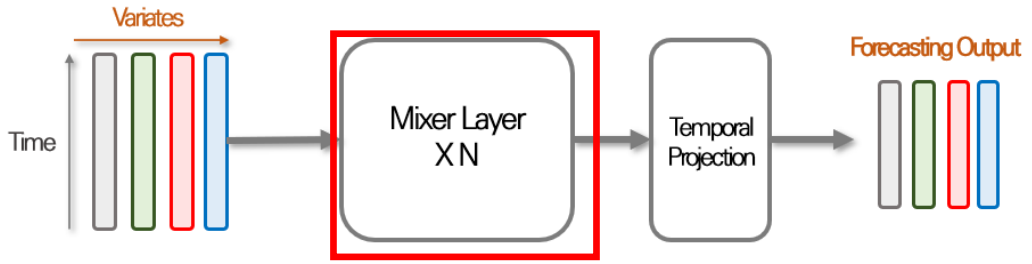
❖ **TSMixer Architecture: Mixer Layer**

- 이론적으로 Linear Model은 temporal information을 잡는 데 능하다 + 변수 간의 관계도 고려해야 한다 = **MLP-Mixer 사용**
- Temporal Information을 잡아내는 **Time-Mixer, 변수 간 상관 관계를 포착하는 Feature-Mixer 사용**



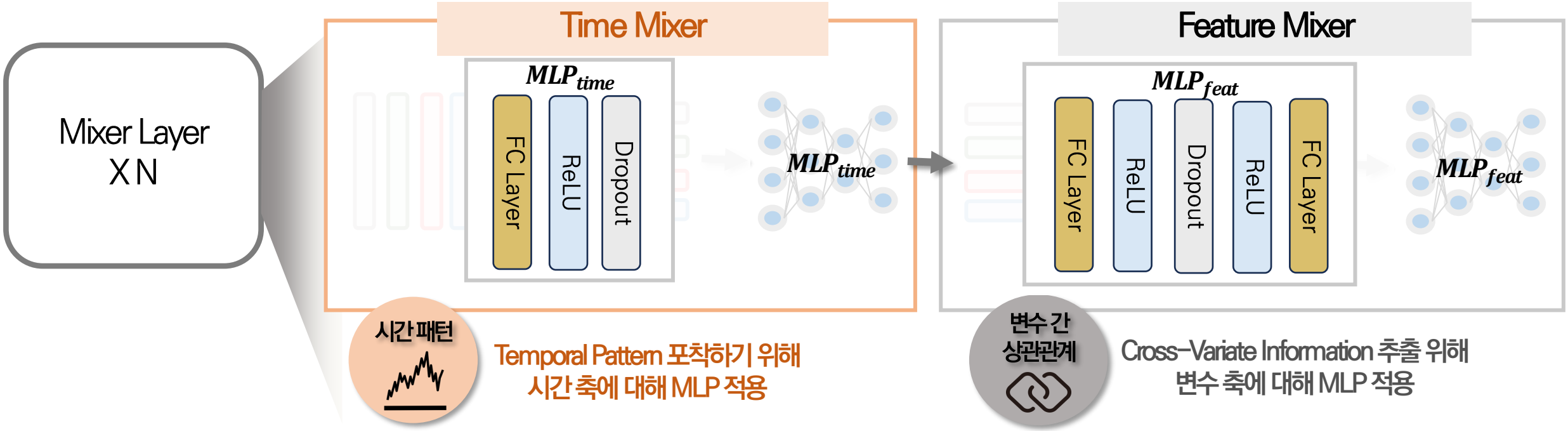
Method

TSMixer: An all-mlp architecture for time series forecasting



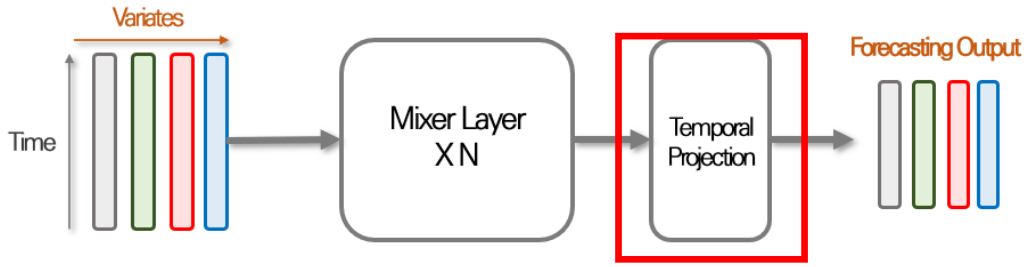
❖ **TSMixer Architecture: Mixer Layer**

- 이론적으로 Linear Model은 temporal information을 잡는 데 능하다 + 변수 간의 관계도 고려해야 한다 = **MLP-Mixer 사용**
- Temporal Information을 잡아내는 **Time-Mixer, 변수 간 상관 관계를 포착하는 Feature-Mixer 사용**



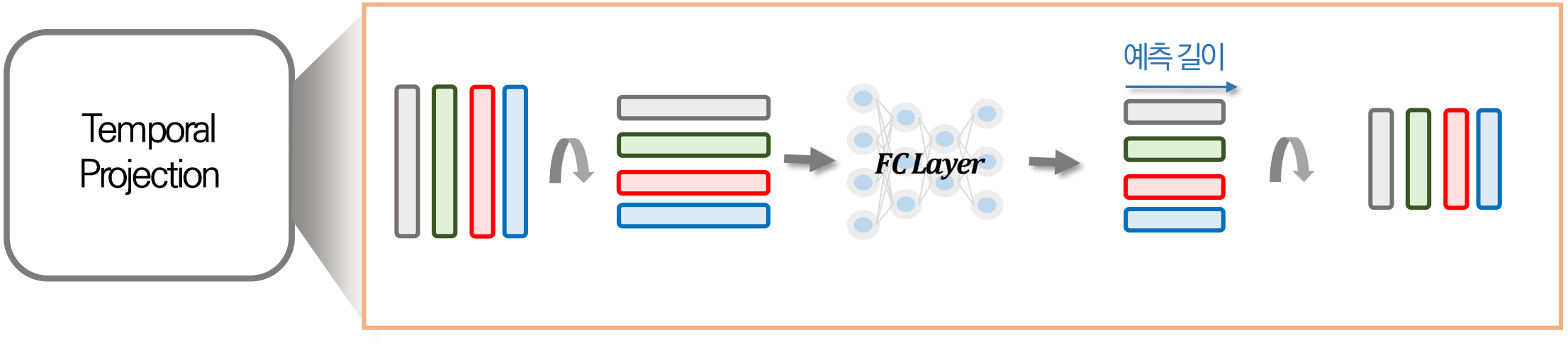
Method

TSMixer: An all-mlp architecture for time series forecasting



❖ **TSMixer Architecture: Temporal Projection**

- Mixer Layer의 output에 대해 Fully-Connected Layer 적용
- 이는 LTSF-Linear의 Simple Linear mapping과 비슷

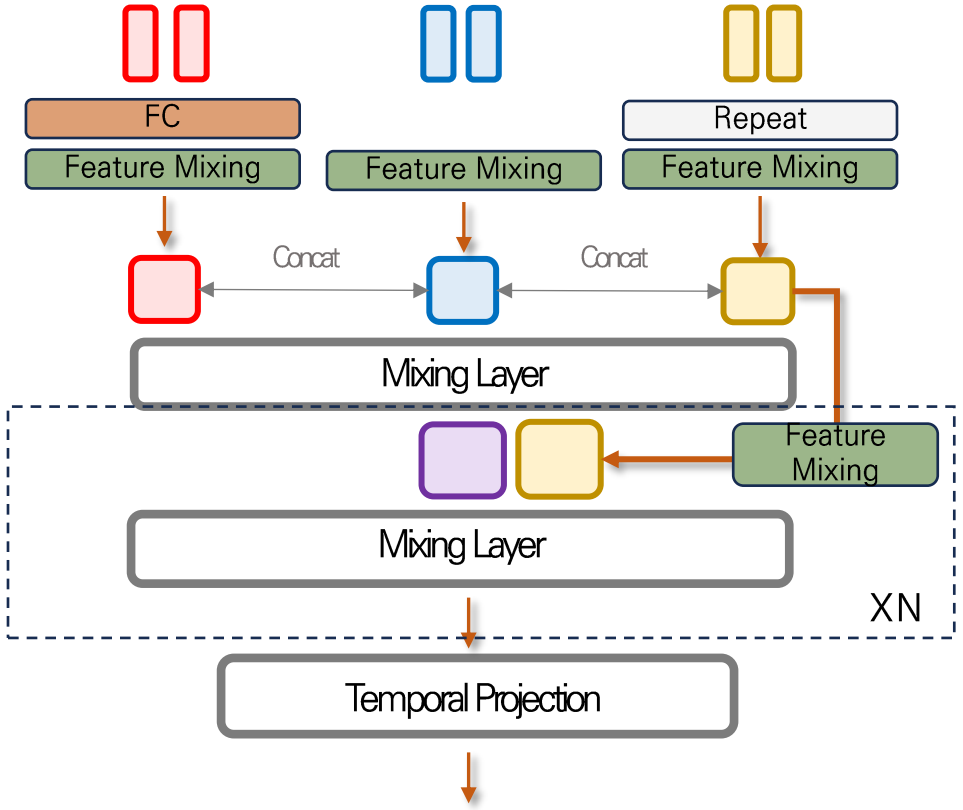


Method

TSMixer: An all-mlp architecture for time series forecasting

❖ **TSMixer Architecture: Extended**

- TSMixer는 Historical observations뿐만 아니라 **보조 정보를 활용하는 확장된 아키텍쳐**를 제공
- Align Stage: 각기 다른 정보들을 하나로 통합 / Mixing Stage: 기존 TSMixer Architecture



Historical Observation

기존 사용하던 이전 시계열의 정보

Future Features

미래의 시간에 따라 달라지는 정보
E.g.) 다음 주에 진행될 할인 정보

Static

시간 불변 특성
E.g) 매장 위치 정보 등

Method

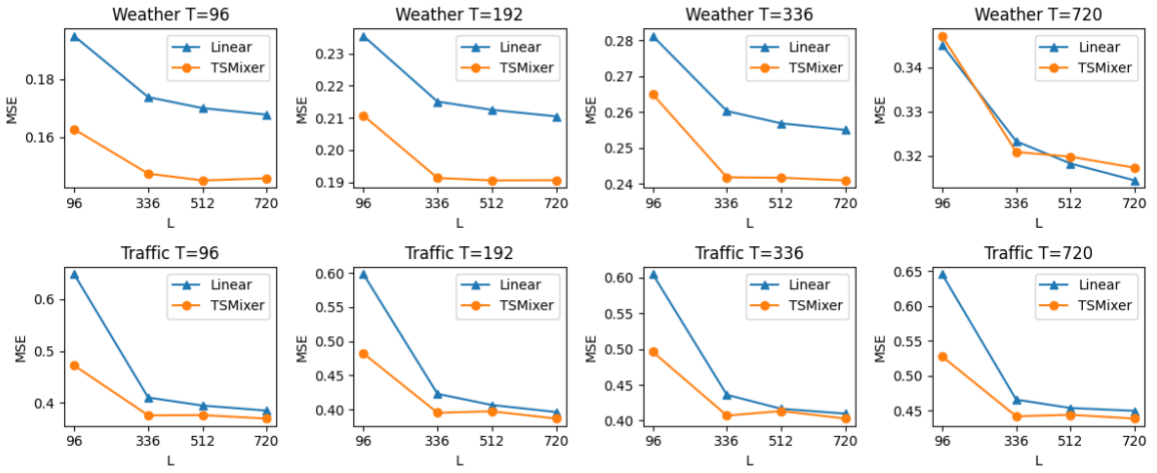
TSMixer: An all-mlp architecture for time series forecasting

❖ Experimental Results: Multivaraite Time Series Forecasting

- 7개의 Multivariate Time Series Forecasting Benchmark Dataset에 대해 실험, 비교 방법론 대비 좋은 성능
- Window size에 대해서 Linear 대비 강건한 모습: 적은 window size에서도 좋은 성능을 보여줌

Table 3: Evaluation results on the long-term forecasting datasets. The numbers of models marked with "*" are obtained from Nie et al. (2023). The best numbers in each row are shown in **bold** and the second best numbers are underlined. We skip TMix-Only in comparisons as it performs similar to TSMixer. The last row shows the average percentage of MSE improvement of TSMixer over other methods.

Models		Multivariate Model										Univariate Model					
		TSMixer		TFT		FEDformer*		Autoformer*		Informer*		**TMix-Only**		Linear		PatchTST*	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.361**	**0.392**	0.674	0.634	0.376	0.415	0.435	0.446	0.941	0.769	0.359	0.391	0.368	**0.392**	0.370	0.400
	192	**0.404**	0.418	0.858	0.704	0.423	0.446	0.456	0.457	1.007	0.786	0.402	0.415	**0.404**	**0.415**	0.413	0.429
	336	**0.420**	**0.431**	0.900	0.731	0.444	0.462	0.486	0.487	1.038	0.784	0.420	0.434	0.436	0.439	0.422	0.440
	720	0.463	0.472	0.745	0.666	0.469	0.492	0.515	0.517	1.144	0.857	0.453	0.467	0.481	0.495	**0.447**	**0.468**
ETTh2	96	**0.274**	0.341	0.409	0.505	0.332	0.374	0.332	0.368	1.549	0.952	0.275	0.342	0.297	0.363	**0.274**	**0.337**
	192	**0.339**	0.385	0.953	0.651	0.407	0.446	0.426	0.434	3.792	1.542	0.339	0.386	0.398	0.429	0.341	**0.382**
	336	0.361	0.406	1.006	0.709	0.400	0.447	0.477	0.479	4.215	1.642	0.366	0.413	0.500	0.491	**0.329**	**0.384**
	720	0.445	0.470	1.187	0.816	0.412	0.469	0.453	0.490	3.656	1.619	0.437	0.465	0.795	0.633	**0.379**	**0.422**
ETTm1	96	**0.285**	**0.339**	0.752	0.626	0.326	0.390	0.510	0.492	0.626	0.560	0.284	0.338	0.303	0.346	0.293	0.346
	192	**0.327**	**0.365**	0.752	0.649	0.365	0.415	0.514	0.495	0.725	0.619	0.324	0.362	0.335	**0.365**	0.333	0.370
	336	**0.356**	**0.382**	0.810	0.674	0.392	0.425	0.510	0.492	1.005	0.741	0.359	0.384	0.365	0.384	0.369	0.392
	720	0.419	**0.414**	0.849	0.695	0.446	0.458	0.527	0.493	1.133	0.845	0.419	0.414	0.419	0.415	**0.416**	0.420
ETTm2	96	**0.163**	**0.252**	0.386	0.472	0.180	0.271	0.205	0.293	0.355	0.462	0.162	0.249	0.170	0.266	0.166	0.256
	192	**0.216**	**0.290**	0.739	0.626	0.252	0.318	0.278	0.336	0.595	0.586	0.220	0.293	0.236	0.317	0.223	0.296
	336	**0.268**	**0.324**	0.477	0.494	0.324	0.364	0.343	0.379	1.270	0.871	0.269	0.326	0.308	0.369	0.274	0.329
	720	0.420	0.422	0.523	0.537	0.410	0.420	0.414	0.419	3.001	1.267	0.358	0.382	0.435	0.449	**0.362**	**0.385**
Weather	96	**0.145**	**0.198**	0.441	0.474	0.238	0.314	0.249	0.329	0.354	0.405	0.145	0.196	0.170	0.229	0.149	**0.198**
	192	**0.191**	0.242	0.699	0.599	0.275	0.329	0.325	0.370	0.419	0.434	0.190	0.240	0.213	0.268	0.194	**0.241**
	336	**0.242**	**0.280**	0.693	0.596	0.339	0.377	0.351	0.391	0.583	0.543	0.240	0.279	0.257	0.305	0.245	0.282
	720	0.320	0.336	1.038	0.753	0.389	0.409	0.415	0.426	0.916	0.705	0.325	0.339	0.318	0.356	**0.314**	**0.334**
Electricity	96	0.131	0.229	0.295	0.376	0.186	0.302	0.196	0.313	0.304	0.393	0.132	0.225	0.135	0.232	**0.129**	**0.222**
	192	0.151	0.246	0.327	0.397	0.197	0.311	0.211	0.324	0.327	0.417	0.152	0.243	0.149	0.246	**0.147**	**0.240**
	336	**0.161**	0.261	0.298	0.380	0.213	0.328	0.214	0.327	0.333	0.422	0.166	0.260	0.164	0.263	0.163	**0.259**
	720	**0.197**	0.293	0.338	0.412	0.233	0.344	0.236	0.342	0.351	0.427	0.200	0.291	0.199	0.297	**0.197**	**0.290**
Traffic	96	0.376	0.264	0.678	0.362	0.576	0.359	0.597	0.371	0.733	0.410	0.370	0.258	0.395	0.274	**0.360**	**0.249**
	192	0.397	0.277	0.664	0.355	0.610	0.380	0.607	0.382	0.777	0.435	0.390	0.268	0.406	0.279	**0.379**	**0.256**
	336	0.413	0.290	0.679	0.354	0.608	0.375	0.623	0.387	0.776	0.434	0.404	0.276	0.416	0.286	**0.392**	**0.264**
	720	0.444	0.306	0.610	0.326	0.621	0.375	0.639	0.395	0.827	0.466	0.443	0.297	0.454	0.308	**0.432**	**0.286**
TSMixer MSE Imp.				**51.94%**		**16.69%**		**24.51%**		**62.40%**		-0.66%		**6.77%**		-1.53%	



Method

TSMixer: An all-mlp architecture for time series forecasting

❖ **Experimental Results: Auxiliary feature**

- 보조 정보를 사용한(static, future information 등) 모델 중 가장 좋은 성능을 기록
- 시계열 예측 알고리즘에 효과적으로 보조 정보를 통합

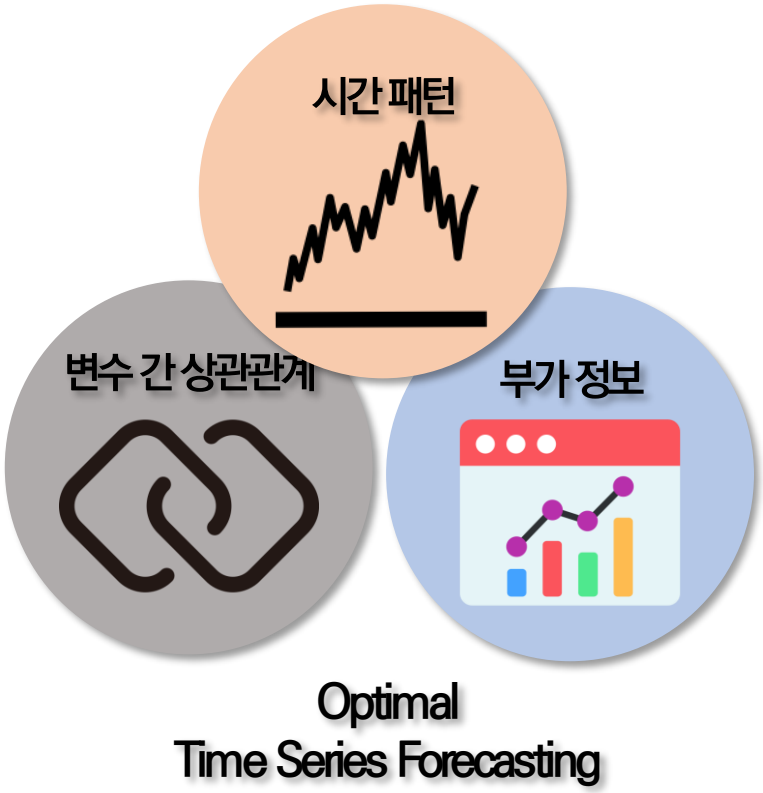
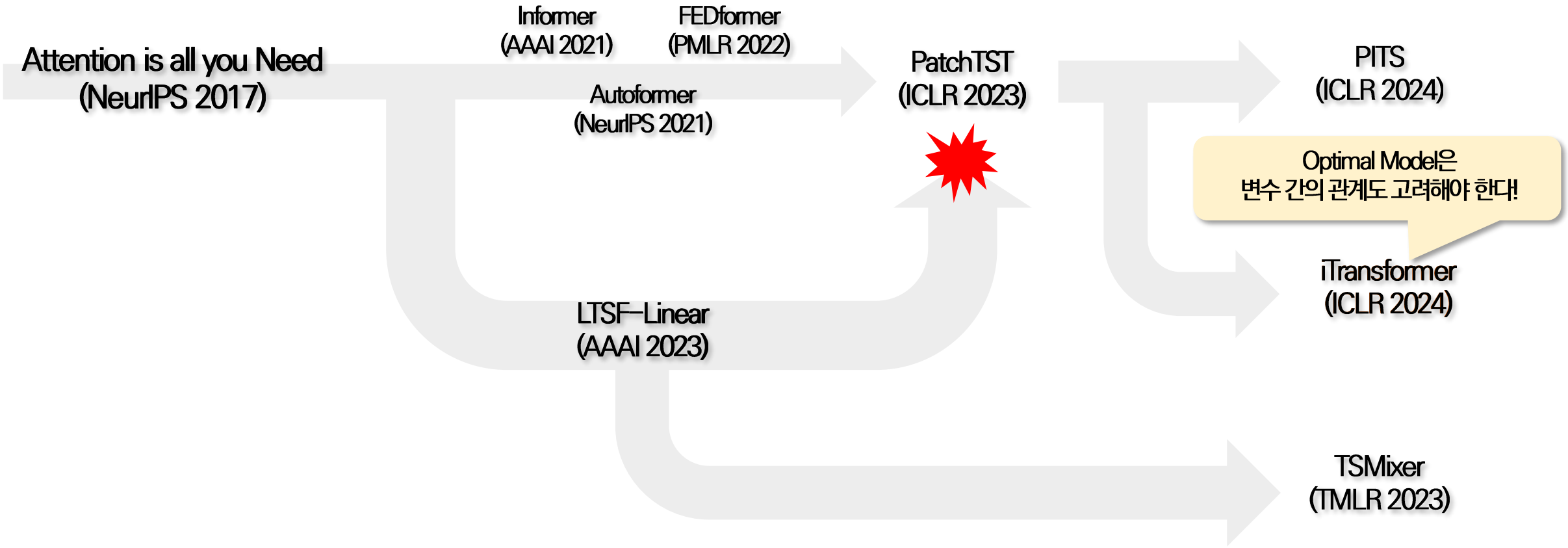


Table 5: Evaluation on M5 with auxiliary information.

Models	Auxiliary feature		Test WRMSSE	Val WRMSSE
	Static	Future		
DeepAR	✔	✔	0.789±0.025	0.611±0.007
TFT	✔	✔	0.670±0.020	0.579±0.011
TSMixer-Ext			0.737±0.033	0.000±0.000
	✔		0.657±0.046	0.000±0.000
		✔	0.697±0.028	0.000±0.000
	✔	✔	**0.640**±0.013	0.568±0.009

Background
Research Trend

❖ Research Trend after Transfomer

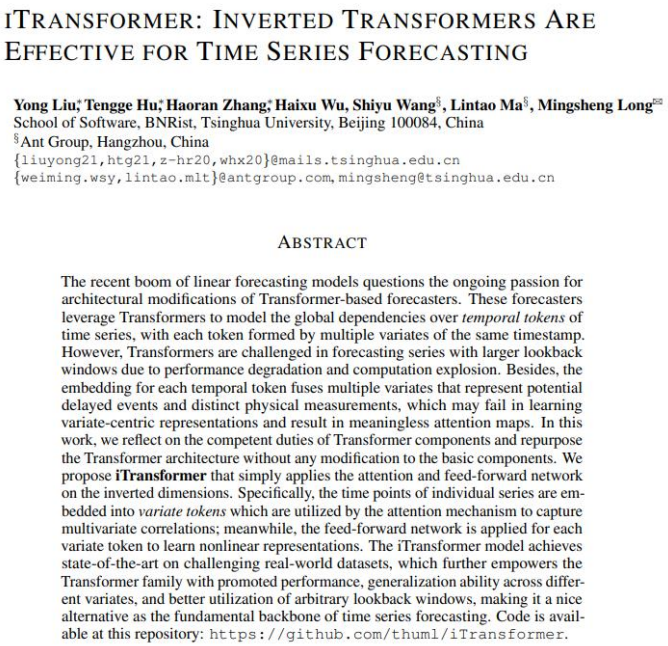


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ **iTransformer: Inverted Transformers are Effective for Time Series Forecasting(ICLR 2024)**

- 2025년 5월 기준 943회 인용
- Transformer의 input을 뒤집는 간단한 전략을 통해, 다변량 시계열 예측에 Transformer가 효율적임을 제시



iTRANSFORMER: INVERTED TRANSFORMERS ARE EFFECTIVE FOR TIME SERIES FORECASTING

Yong Liu[*], **Tengge Hu**[*], **Haoran Zhang**[*], **Haixu Wu**, **Shiyu Wang**[§], **Lintao Ma**[§], **Mingsheng Long**[✉]
School of Software, BNRist, Tsinghua University, Beijing 100084, China
[§]Ant Group, Hangzhou, China
{liuyong21,htg21,z-hr20,whx20}@mails.tsinghua.edu.cn
{weiming.wsy,lintao.mlt}@antgroup.com, mingsheng@tsinghua.edu.cn

ABSTRACT

The recent boom of linear forecasting models questions the ongoing passion for architectural modifications of Transformer-based forecasters. These forecasters leverage Transformers to model the global dependencies over *temporal tokens* of time series, with each token formed by multiple variates of the same timestamp. However, Transformers are challenged in forecasting series with larger lookback windows due to performance degradation and computation explosion. Besides, the embedding for each temporal token fuses multiple variates that represent potential delayed events and distinct physical measurements, which may fail in learning variate-centric representations and result in meaningless attention maps. In this work, we reflect on the competent duties of Transformer components and repurpose the Transformer architecture without any modification to the basic components. We propose **iTransformer** that simply applies the attention and feed-forward network on the inverted dimensions. Specifically, the time points of individual series are embedded into *variate tokens* which are utilized by the attention mechanism to capture multivariate correlations; meanwhile, the feed-forward network is applied for each variate token to learn nonlinear representations. The iTransformer model achieves state-of-the-art on challenging real-world datasets, which further empowers the Transformer family with promoted performance, generalization ability across different variates, and better utilization of arbitrary lookback windows, making it a nice alternative as the fundamental backbone of time series forecasting. Code is available at this repository: https://github.com/thuml/iTransformer.

LTSF-Linear의 Title

Are Transformers Effective for Time Series Forecasting?

iTransformer의 Title

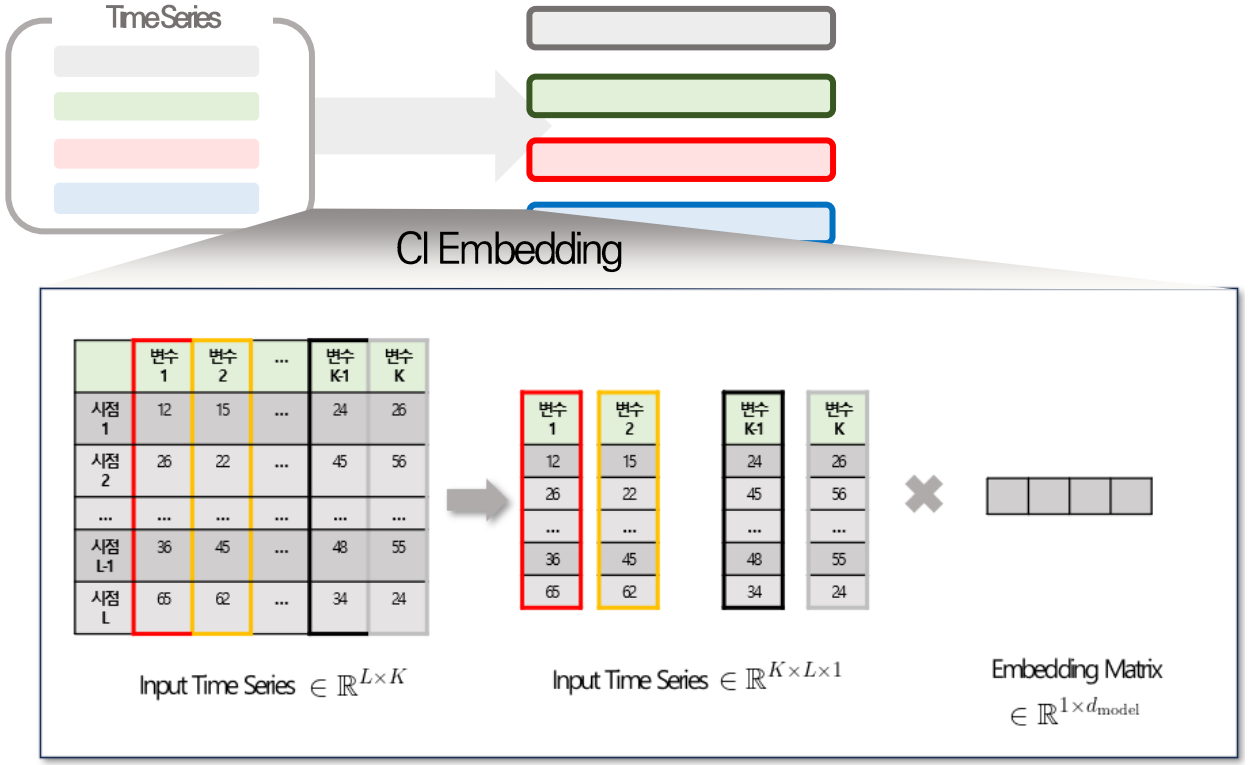
Inverted Transformers are Effective for Time Series Forecasting!

Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- 왜 CI Strategy가 잘 될까?: **이론적인 배경 설명 부족, 실험적인 결과만 존재**

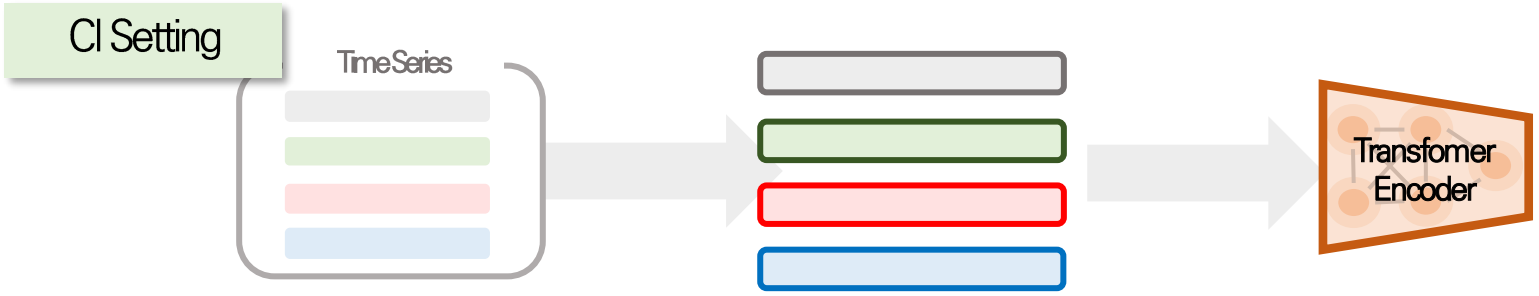


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

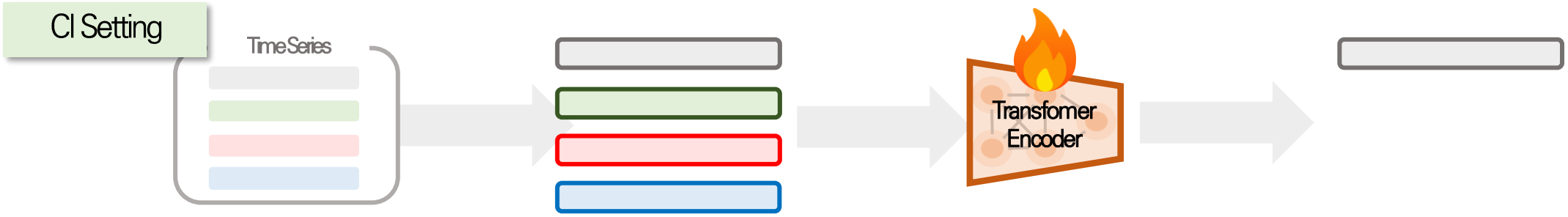


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

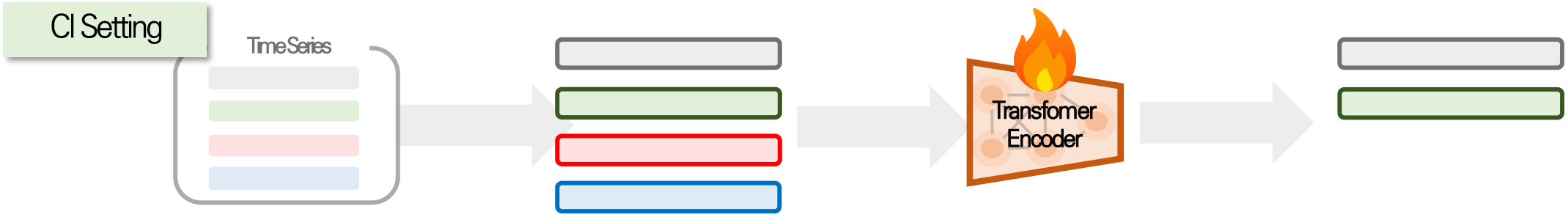


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

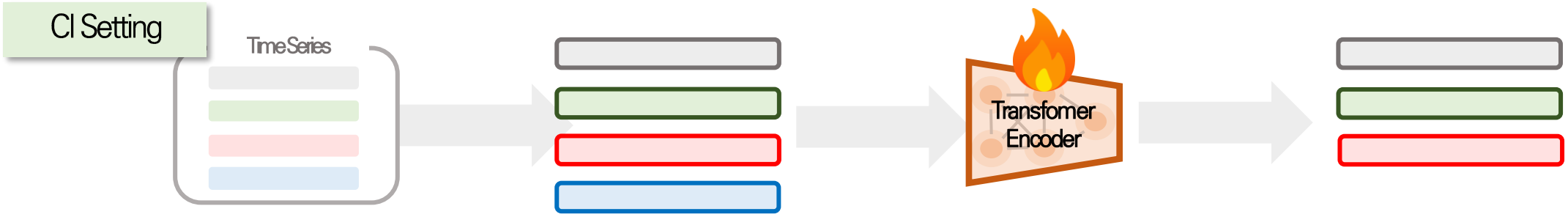


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

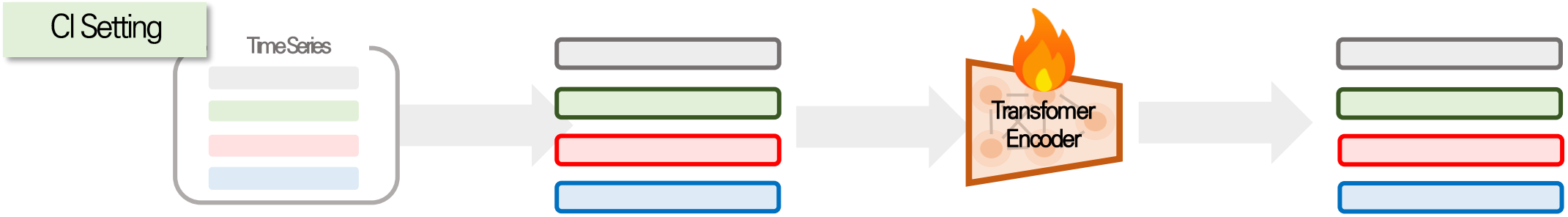


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

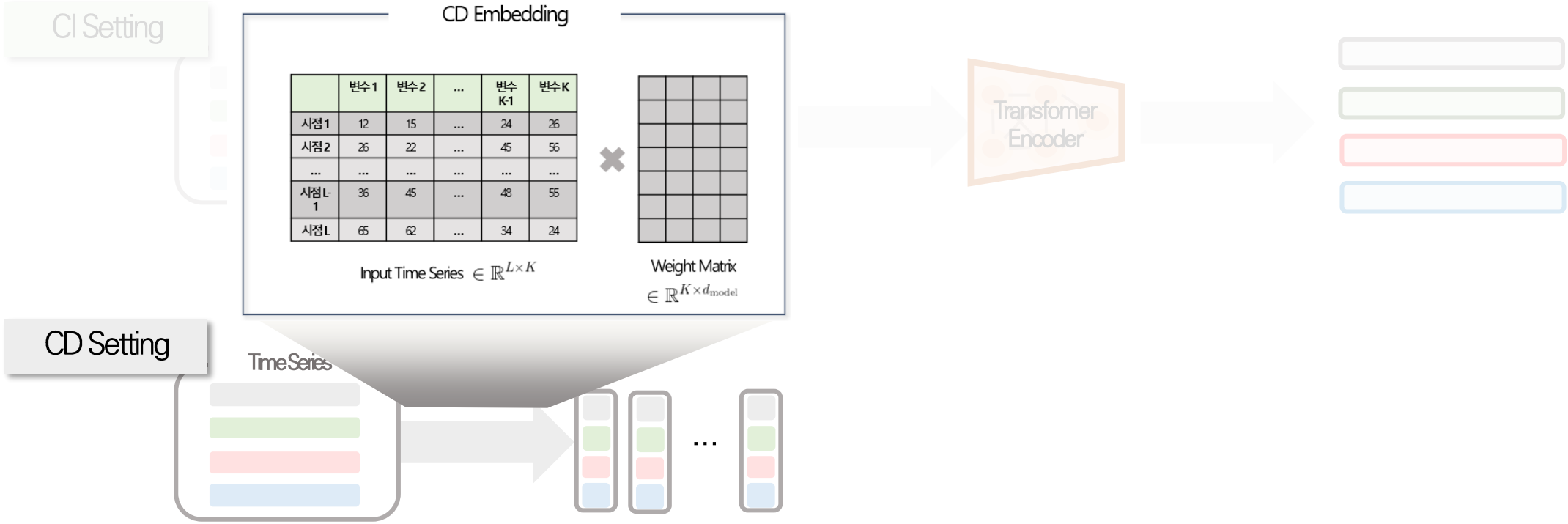


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

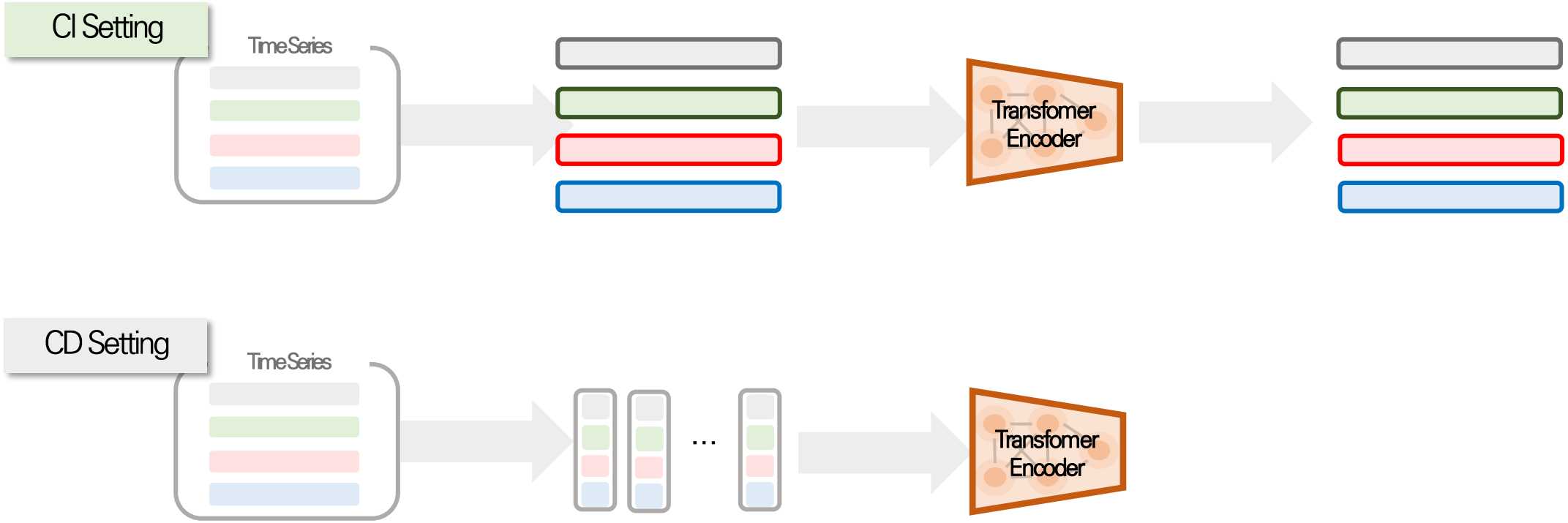


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

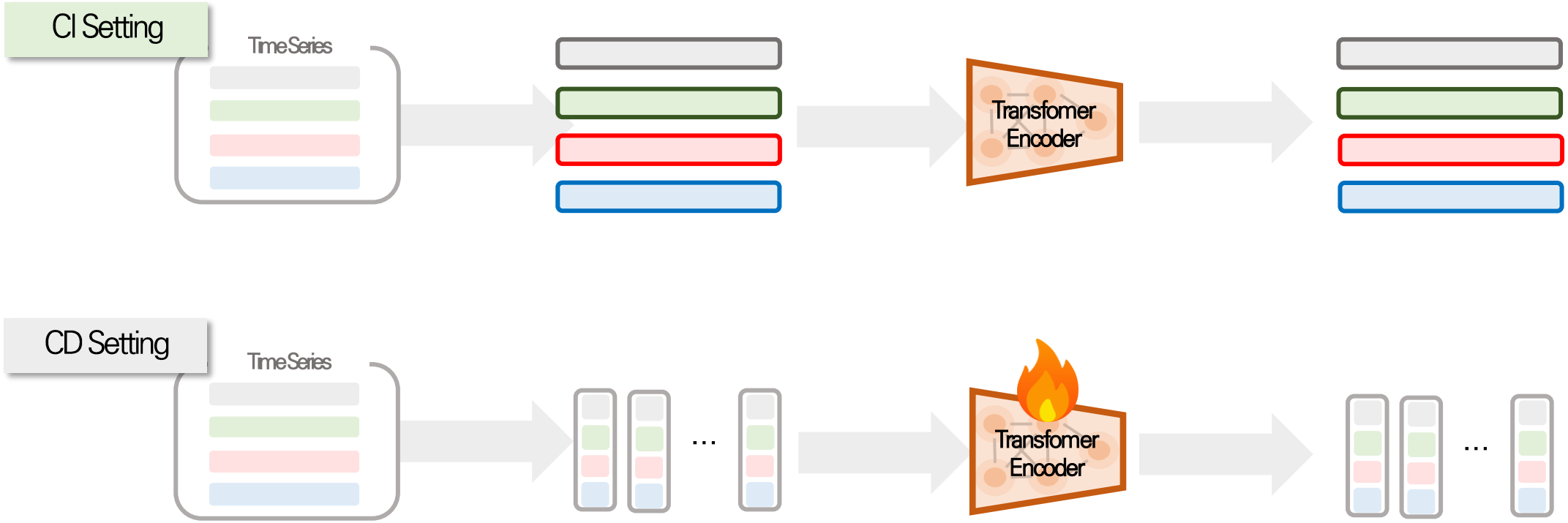


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**



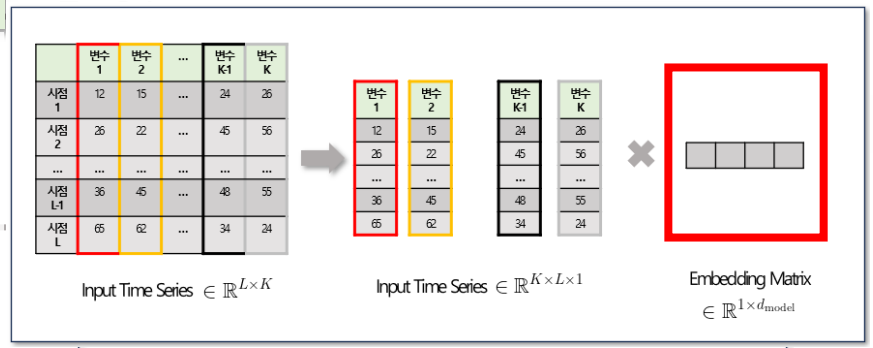
Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

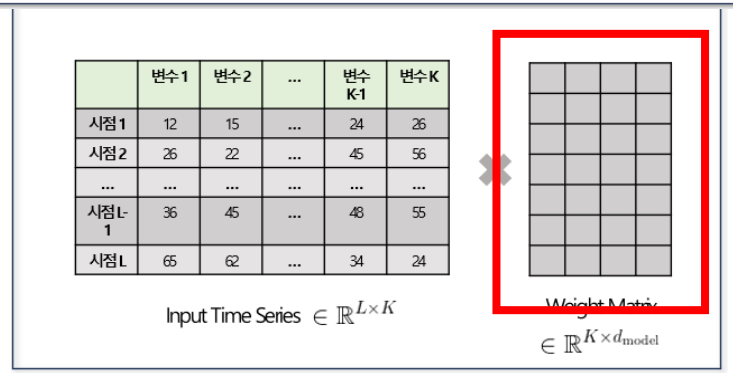
❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

CI Setting



CD Setting



왜 Parameter가 적어지는데 성능이 월등하지?

이론적으로는 모르겠는데,
실험적으로는 Distribution Shift가 있으면 잘하더라

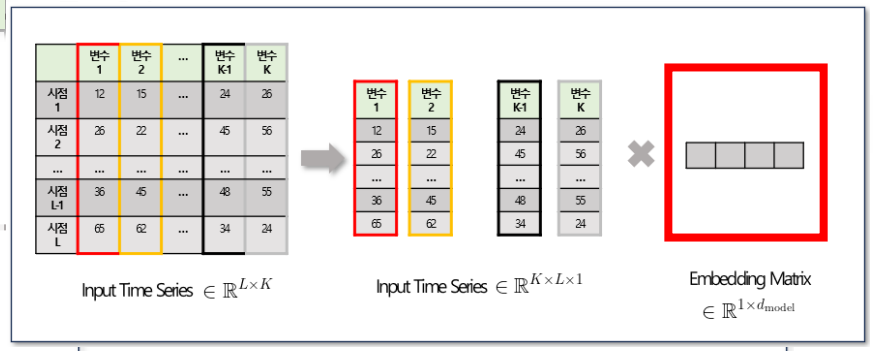
Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

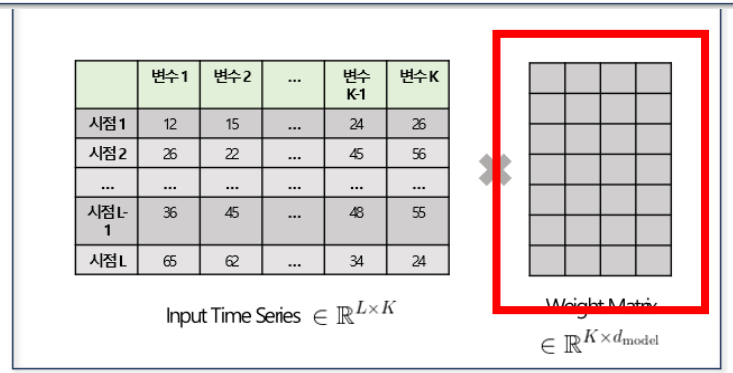
❖ Revisiting the Channel Independence Strategy

- Channel Independence Strategy의 핵심: **다변량 시계열을 단변량 시계열의 집합으로 인식하게 하자!**
- **왜 CI Strategy가 잘 될까?: 이론적인 배경 설명 부족, 실험적인 결과만 존재**

CI Setting



CD Setting



그럼 죽어

그럼 CI가 Optimal하지 않을 수도 있다는 거네?

이론적으로는 모르겠는데,
실험적으로는 Distribution Shift가 있으면 잘하더라

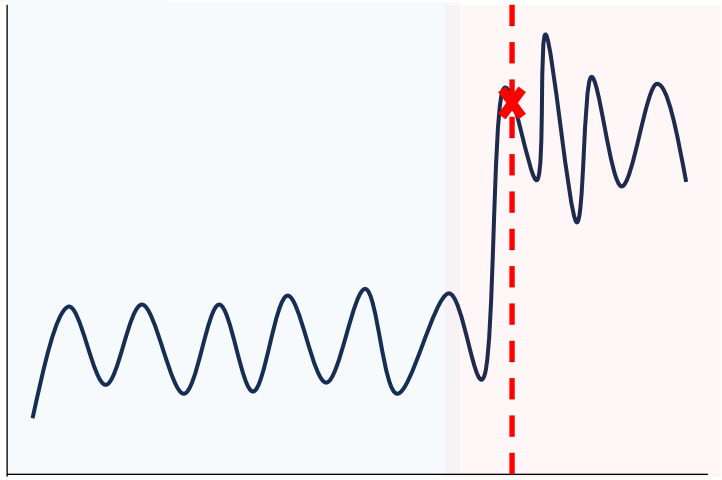
Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Inverted Transformer

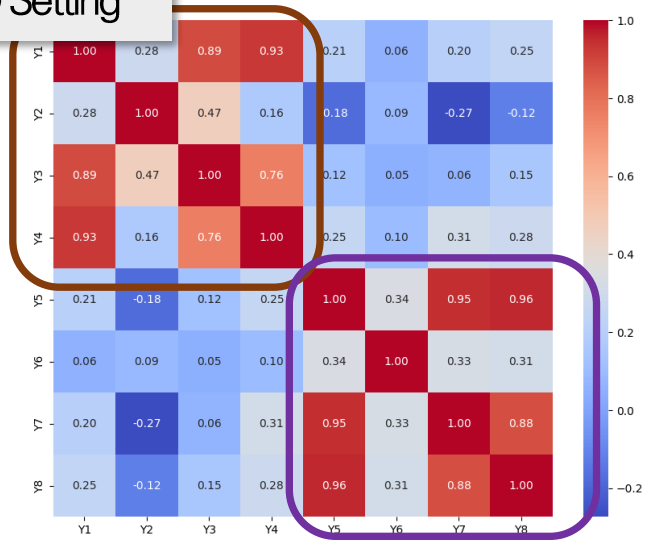
- CI와 CD Setting은 각각의 장단점을 가지고 있음
  - CI: 변수 별로 따로 학습하게 하여 Distribution Shift에 강건 but 모델의 표현력은 낮아짐
  - CD: 변수 간의 correlation 모델링하여 표현력이 높음 but Training과 다른 분포가 들어오면 성능 낮아짐

CI Setting



분포 변화가 있는 상황에서 강건

CD Setting



변수 간의 Correlation 고려 가능

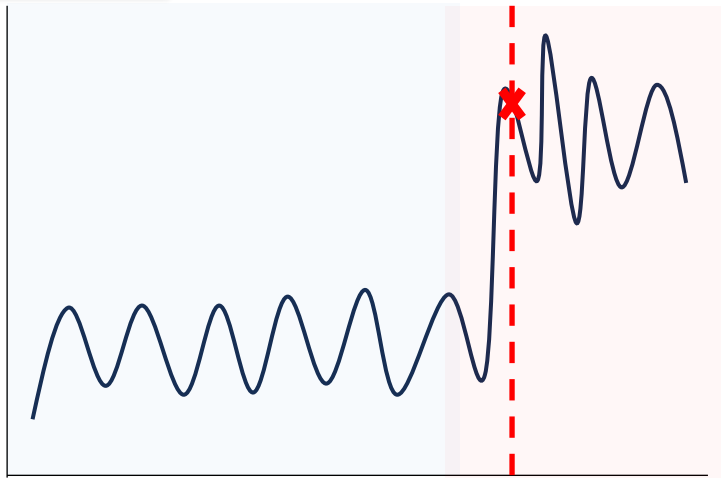
Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Inverted Transformer

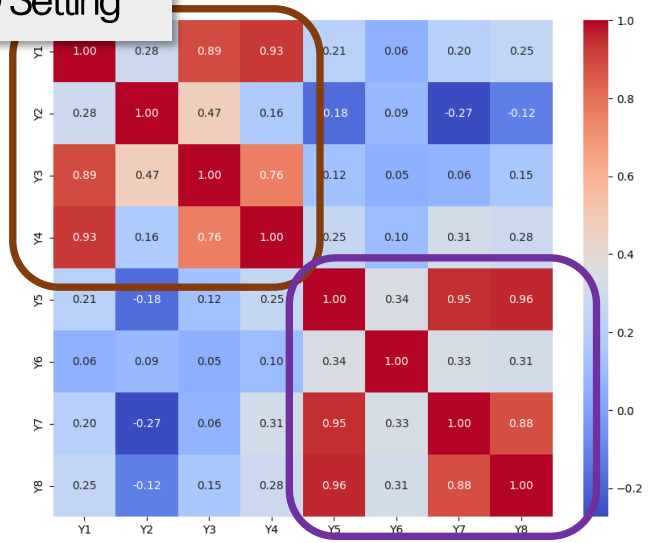
- CI와 CD Setting은 각각의 장단점을 가지고 있음 -> 두 개의 장점을 모두 포함할 수 있는 방법론 필요!
  - CI: 변수 별로 따로 학습하게 하여 Distribution Shift에 강건 but 모델의 표현력은 낮아짐
  - CD: 변수 간의 correlation 모델링하여 표현력이 높음 but Training과 다른 분포가 들어오면 성능 낮아짐

CI Setting



분포 변화가 있는 상황에서 강건

CD Setting



변수 간의 Correlation 고려 가능

Inverted Transformer

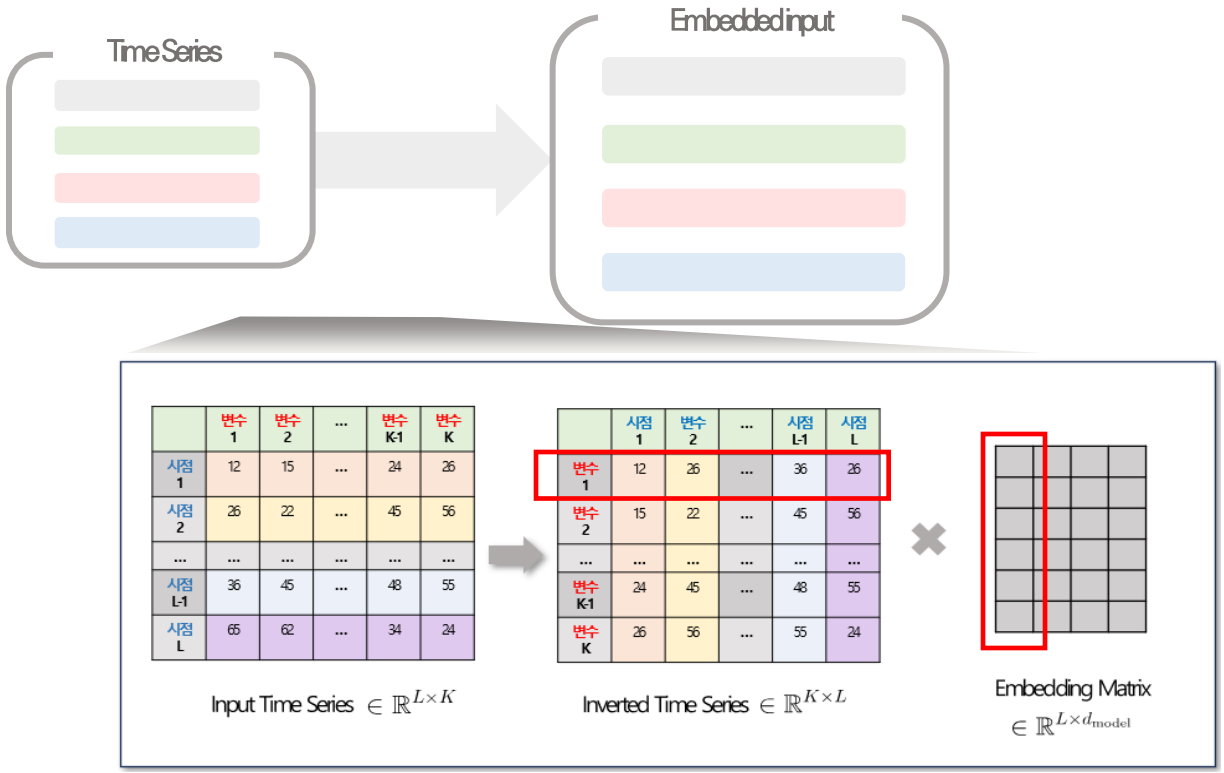
How?

Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Overall Structure

- 임베딩 시 각각의 변수를 **독립적으로** 임베딩 -> 시계열의 길이를 임베딩 차원으로 보냄
  - E.g) CI Setting: **시계열을 변수 단위로 쪼갠 후 임베딩** / CD Setting: 변수 길이를 임베딩 차원으로 보냄



변수 독립적으로 embedding vector 생성

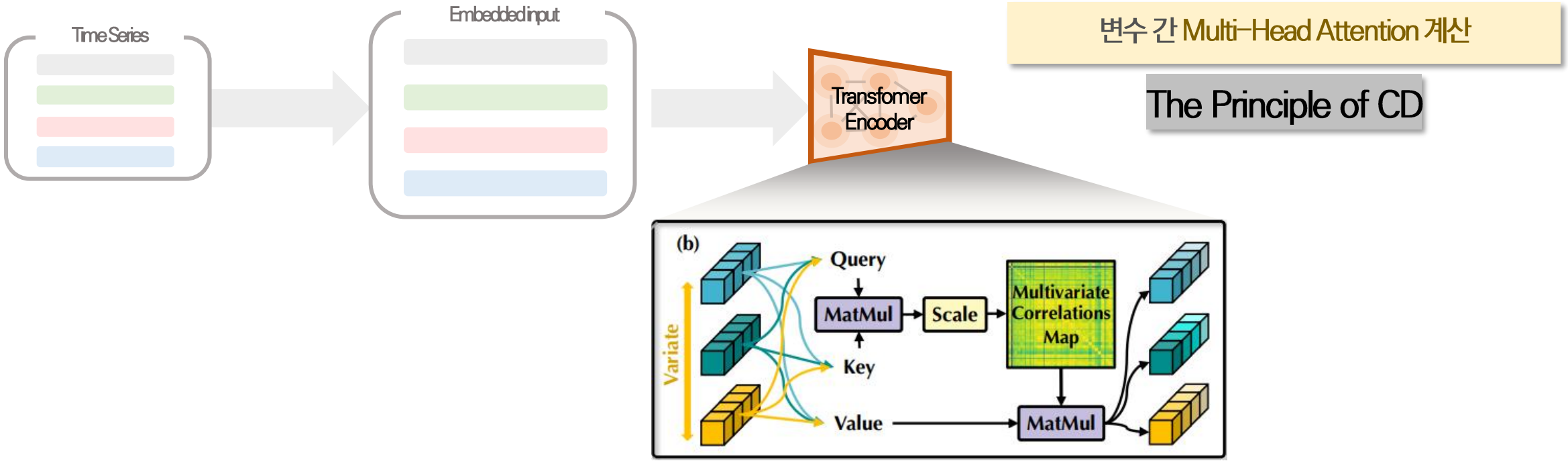
The Principle of CI

Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Overall Structure

- Transformer Encoder 내에서 변수 간의 Correlation을 모델링
  - E.g) CI Setting: 변수가 개별적으로 입력되어 변수 간 관계 고려 x / CD Setting: FFN에서 토큰 내부의 변수 간 관계 고려 가능

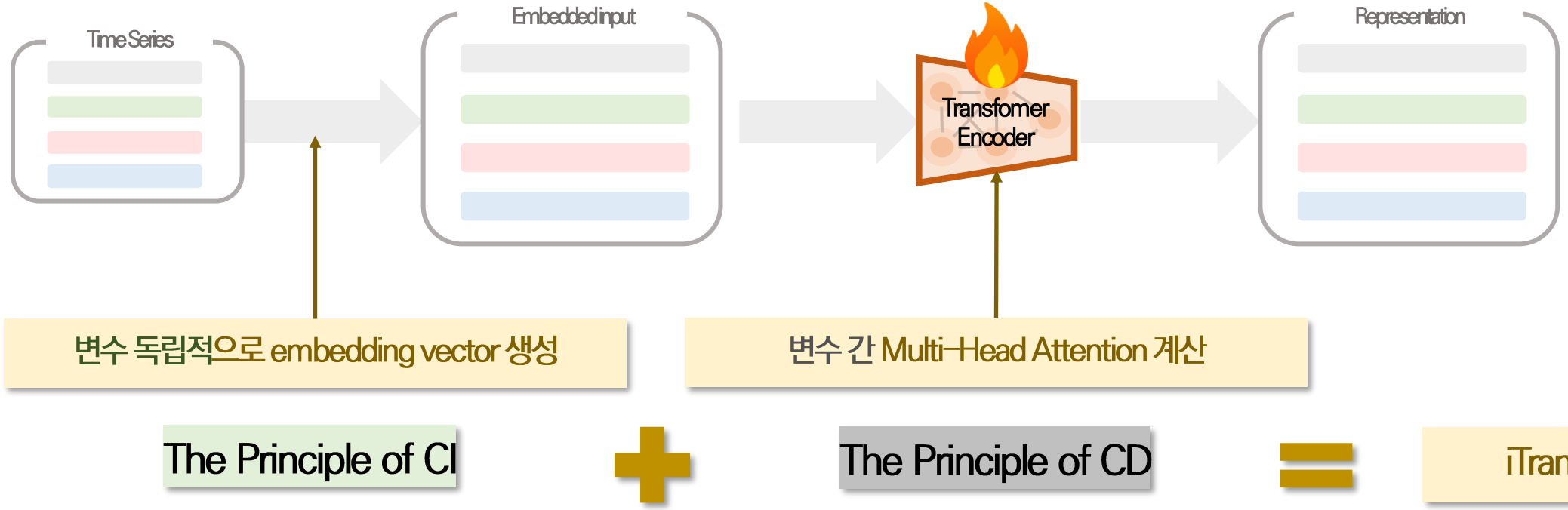


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Overall Structure

- 임베딩 과정에서 변수 독립적으로 입력, Multi-Head Attention 계산 과정에서 변수 간 관계 고려
- CI와 CD의 장점을 Input 차원을 Transpose하면 달성할 수 있다라고 주장

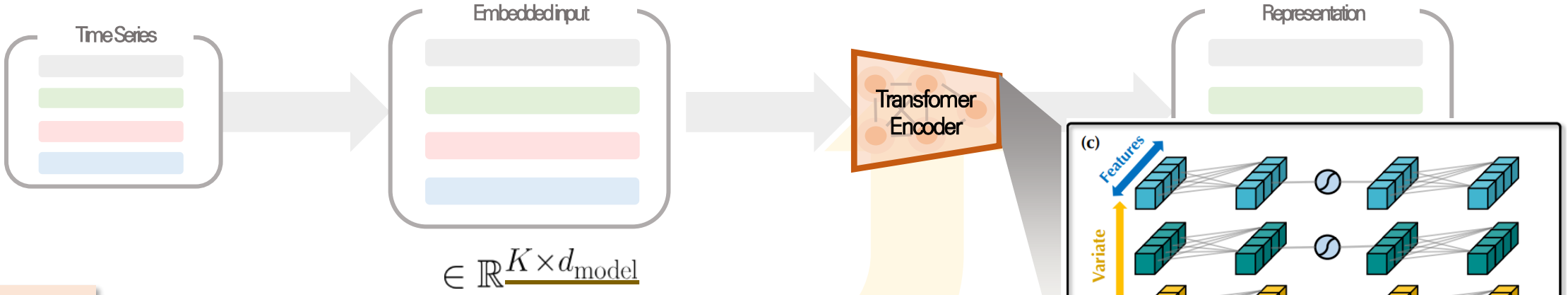


Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ **Postional Embedding is not Needed**

- 임베딩 시 시간 순서 정보는 임베딩 차원으로 투영
- (논문 주장) Transformer Encoder의 FFN에서 시계열의 순서가 보존되기 때문에 순서 정보를 주입하는 것은 불필요하다



Problems

시간 길이 정보가 embedding dimension으로 투영되기 때문에,
시계열 정보를 임베딩 후 주입할 수 없음

Paper says...

시계열 순서성이 FFN의 Neuron Permutation에 내재적으로 저장되기 때문에,
Positional Embedding은 필요 없다.

Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ **Experimental Results: Multivaraite Time Series Forecasting**

- 7개의 Multivariate Time Series Forecasting Benchmark Dataset에 대해 실험
- 이중 6개의 데이터셋에서 SOTA의 성능을 기록

Table 1: Multivariate forecasting results with prediction lengths $S \in \{12, 24, 36, 48\}$ for PEMS and $S \in \{96, 192, 336, 720\}$ for others and fixed lookback length $T = 96$. Results are averaged from all prediction lengths. *Avg* means further averaged by subsets. Full results are listed in Appendix F.4.

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	0.205	0.290	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	0.481	0.304	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**	**0.278**	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

Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Experimental Results: Generalization

- Transformer 구조를 사용한 기존 시계열 예측 모델에 iTransformer의 아이디어(input을 Transpose) 적용
- 16.8% ~ 38.9%의 높은 성능 향상을 보임

Table 2: Performance promotion obtained by our inverted framework. Flashformer means Transformer equipped with hardware-accelerated FlashAttention (Dao et al., 2022). We report the average performance and the relative MSE reduction (Promotion). Full results can be found in Appendix F.2.

Models		Transformer (2017)		Reformer (2020)		Informer (2021)		Flowformer (2022)		Flashformer (2022)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	Original	0.277	0.372	0.338	0.422	0.311	0.397	0.267	0.359	0.285	0.377
	+Inverted	**0.178**	**0.270**	**0.208**	**0.301**	**0.216**	**0.311**	**0.210**	**0.293**	**0.206**	**0.291**
	Promotion	35.6%	27.4%	38.4%	28.7%	30.5%	21.6%	21.3%	18.6%	27.8%	22.9%
Traffic	Original	0.665	0.363	0.741	0.422	0.764	0.416	0.750	0.421	0.658	0.356
	+Inverted	**0.428**	**0.282**	**0.647**	**0.370**	**0.662**	**0.380**	**0.524**	**0.355**	**0.492**	**0.333**
	Promotion	35.6%	22.3%	12.7%	12.3%	13.3%	8.6%	30.1%	15.6%	25.2%	6.4%
Weather	Original	0.657	0.572	0.803	0.656	0.634	0.548	0.286	0.308	0.659	0.574
	+Inverted	**0.258**	**0.279**	**0.248**	**0.292**	**0.271**	**0.330**	**0.266**	**0.285**	**0.262**	**0.282**
	Promotion	60.2%	50.8%	69.2%	55.5%	57.3%	39.8%	7.2%	7.7%	60.2%	50.8%

Method

iTransformer: Inverted Transformers Are Effective for Time Series Forecasting

❖ Experimental Results: Increasing Lookback Length

- Input Time Series의 Look back window를 다르게 하여 실험
- 다른 모델과 다르게, Look back window가 큰 상황에서 더 좋은 성능을 기록

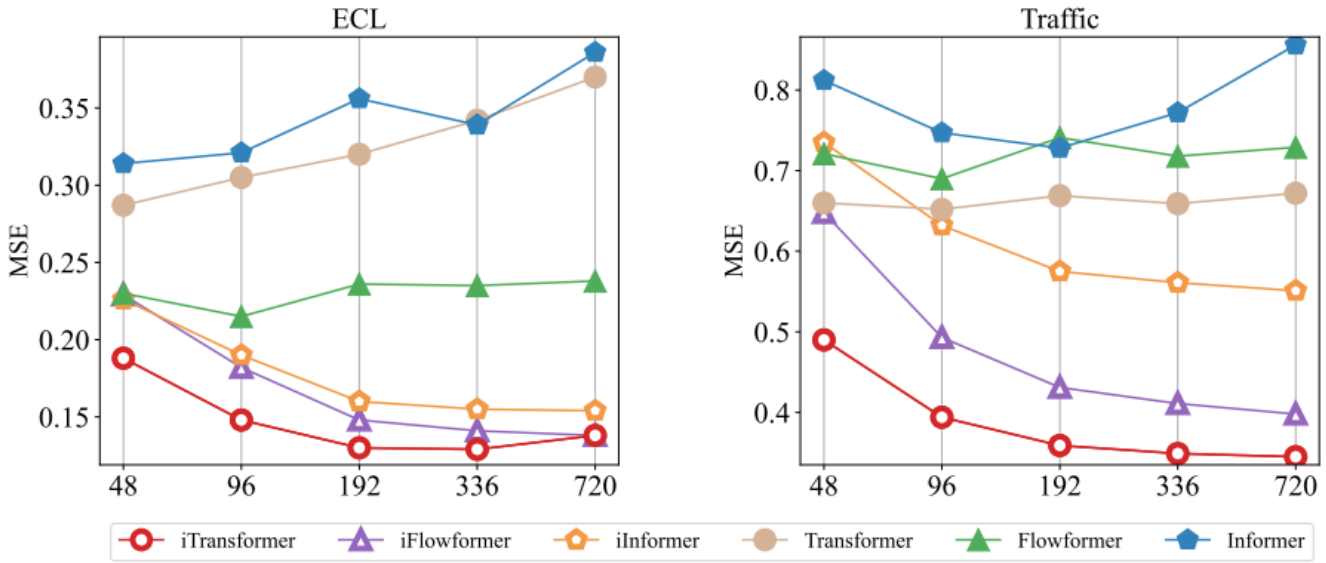
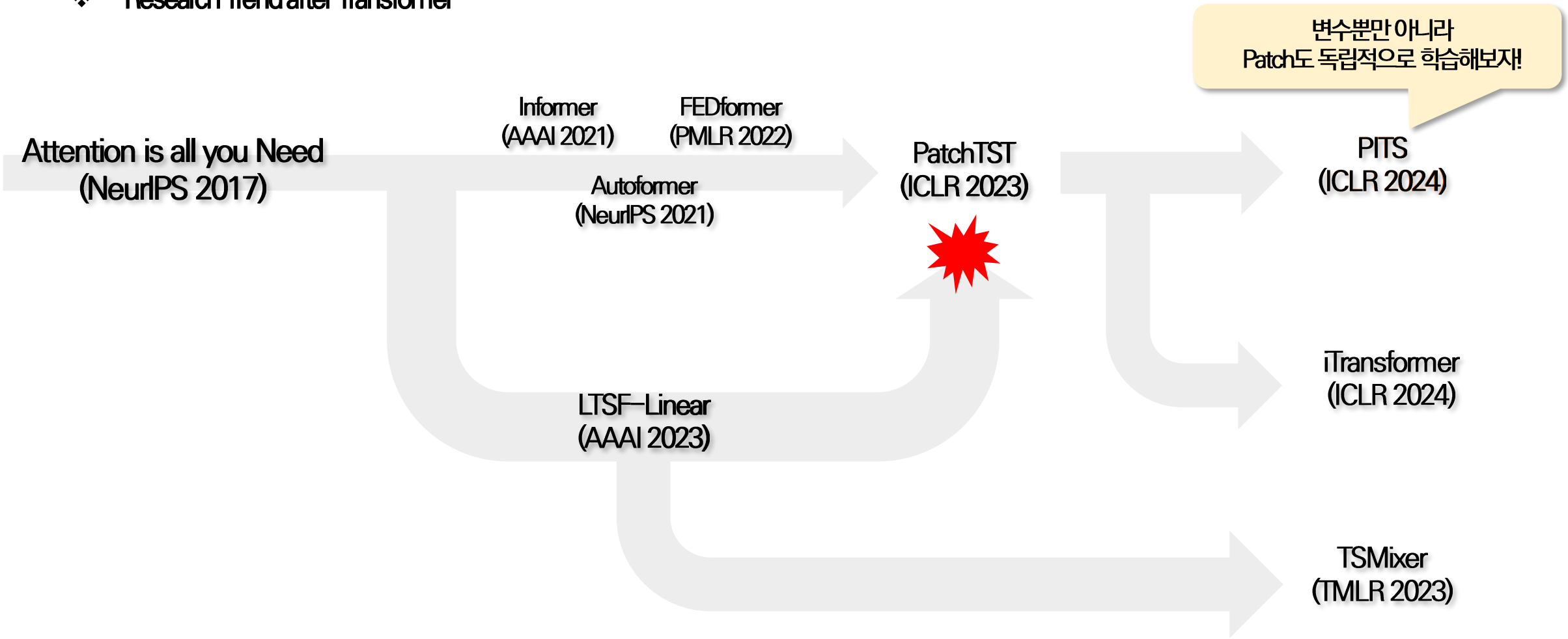


Figure 6: Forecasting performance with the lookback length $T \in \{48, 96, 192, 336, 720\}$ and fixed prediction length $S = 96$. While the performance of Transformer-based forecasters does not necessarily benefit from the increased lookback length, the inverted framework empowers the vanilla Transformer and its variants with improved performance on the enlarged lookback window.

Background
Research Trend

❖ Research Trend after Transfomer



Method

Learning to Embed Time Series Patches Independently

❖ **Learning to Embed Time Series Patches Independently(ICLR 2024)**

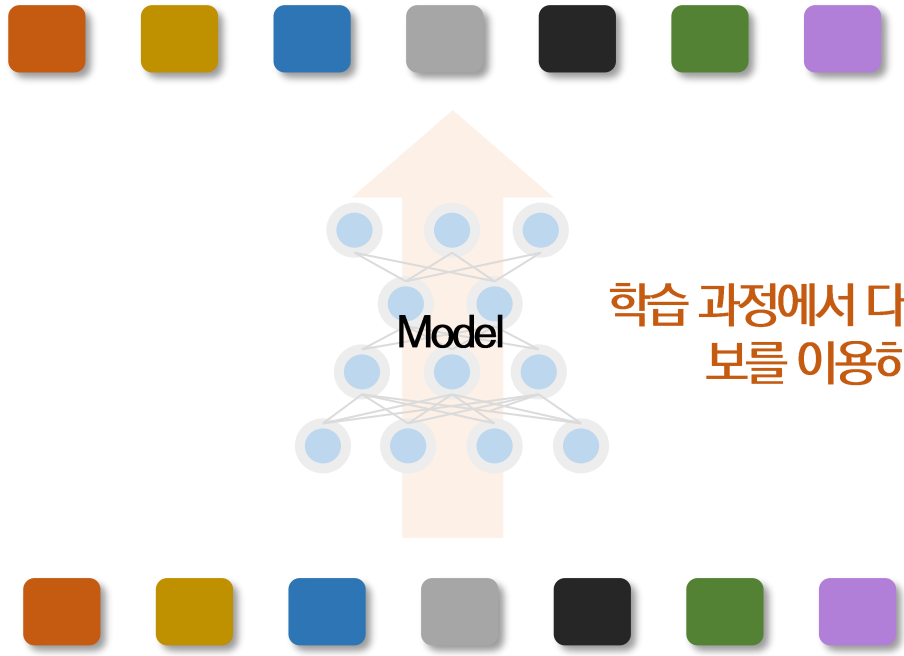
- 2025년 5월 기준 40회 인용
- Idea: 각 패치를 재구축하기 위해 다른 패치의 정보가 필요 없다 → **변수뿐만 아니라 패치도 독립적으로 학습하자!**

LEARNING TO EMBED TIME SERIES PATCHES INDEPENDENTLY

Seunghan Lee, Taeyoung Park, Kibok Lee
Department of Statistics and Data Science, Yonsei University
{seunghan9613,tpark,kibok}@yonsei.ac.kr

ABSTRACT

Masked time series modeling has recently gained much attention as a self-supervised representation learning strategy for time series. Inspired by masked image modeling in computer vision, recent works first patchify and partially mask out time series, and then train Transformers to capture the dependencies between patches by predicting masked patches from unmasked patches. However, we argue that capturing such patch dependencies might not be an optimal strategy for time series representation learning; rather, learning to embed patches independently results in better time series representations. Specifically, we propose to use 1) the simple patch reconstruction task, which autoencode each patch without looking at other patches, and 2) the simple patch-wise MLP that embeds each patch independently. In addition, we introduce complementary contrastive learning to hierarchically capture adjacent time series information efficiently. Our proposed method improves time series forecasting and classification performance compared to state-of-the-art Transformer-based models, while it is more efficient in terms of the number of parameters and training/inference time. Code is available at this repository: https://github.com/seunghan96/pits.



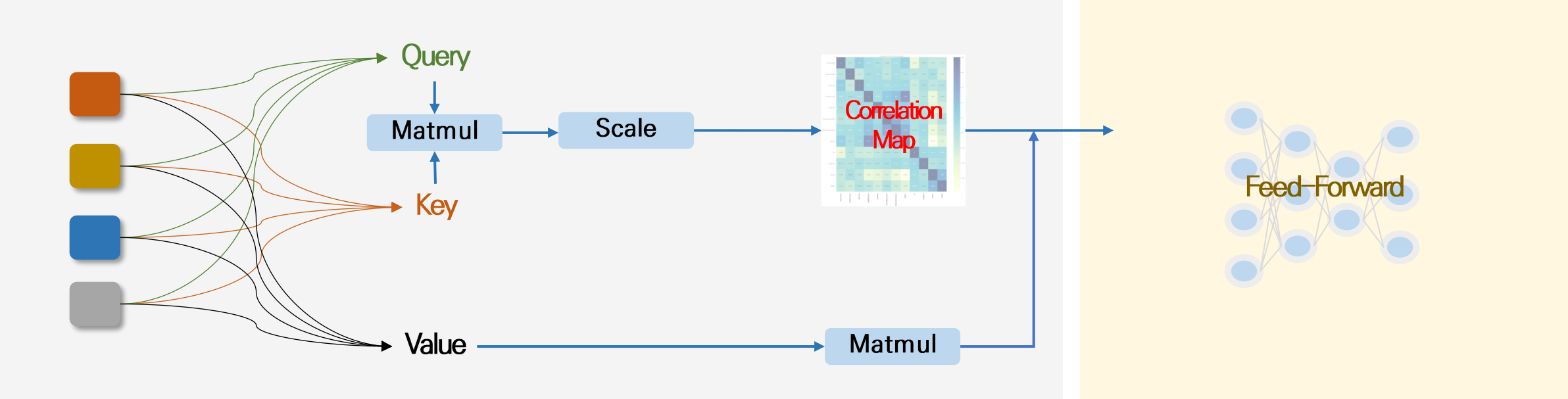
학습 과정에서 다른 Patch의 정보를 이용하지 않게!

Method

Learning to Embed Time Series Patches Independently

❖ **Revisit PatchTST: Modeling Inter-Patch Correlation**

- PatchTST: 변수는 독립적으로, Patch는 dependencies 고려
- 어떻게 상관관계를 고려하는가?: Transformer의 Attention이 입력 간의 상관 관계를 고려하기 때문

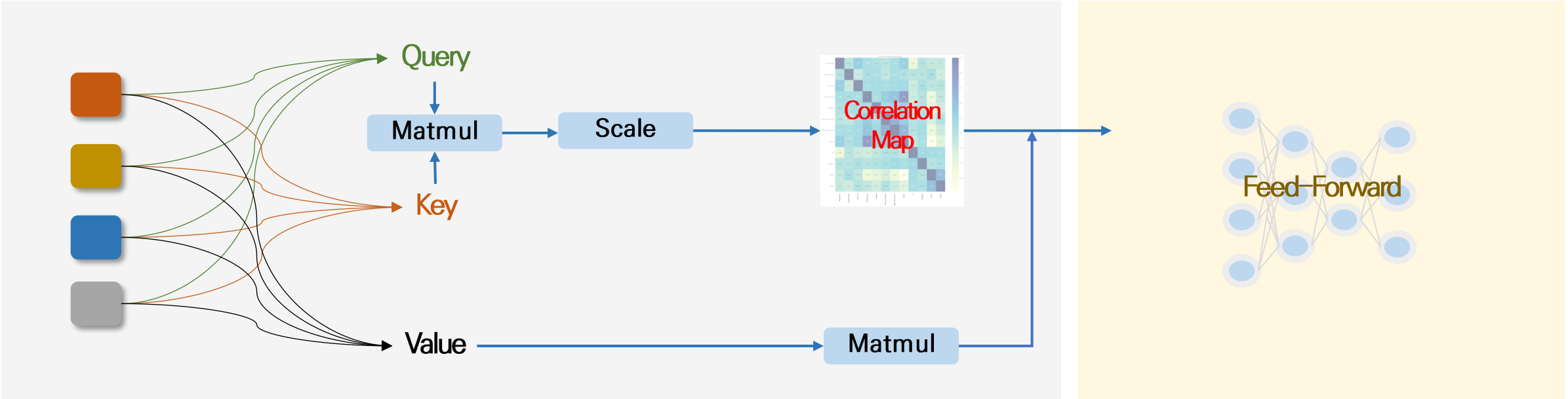


Method

Learning to Embed Time Series Patches Independently

❖ **Revisit PatchTST: Modeling Inter-Patch Correlation**

- PatchTST: 변수는 독립적으로, Patch는 dependencies 고려
- 어떻게 상관관계를 고려하는가?: Transformer의 Attention이 입력 간의 상관 관계를 고려하기 때문



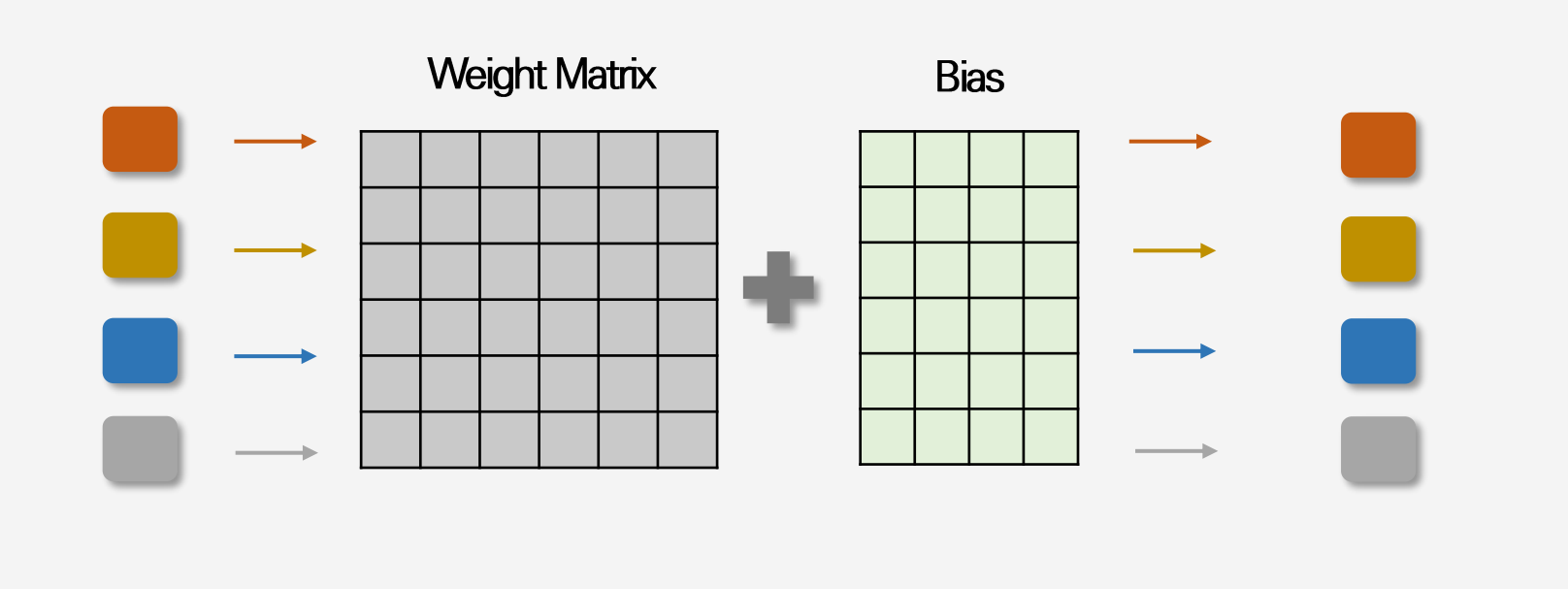
표현 학습에서 Patch를 모델링하는 데 Patch 간의 관계가 필요한가?

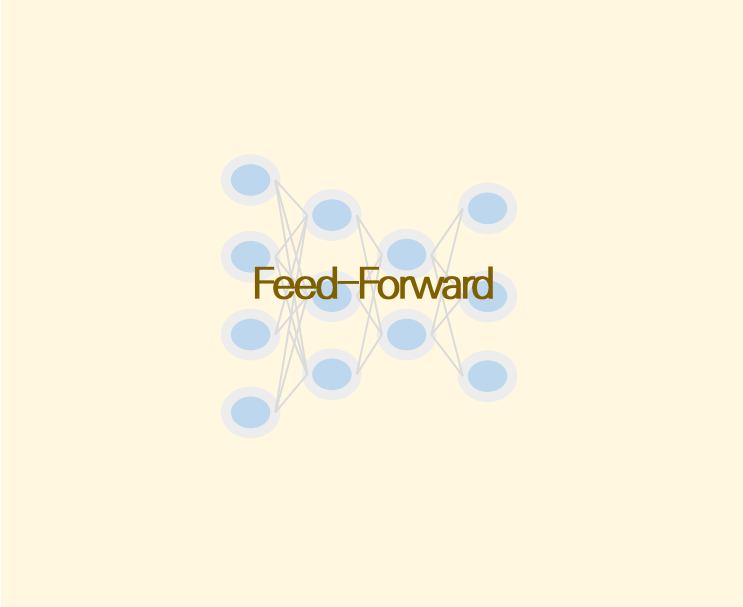
Method

Learning to Embed Time Series Patches Independently

❖ **PI(Patch Independent) Structure**

- Patch들의 dependencies를 고려하지 않고, patch 독립적으로 학습하게 하는 Patch Independent 구조 제안
- 기존 Patch 기반 모델들이 사용하던 Attention 기반 Transformer가 아닌 Simple MLP로 시계열 표현 학습



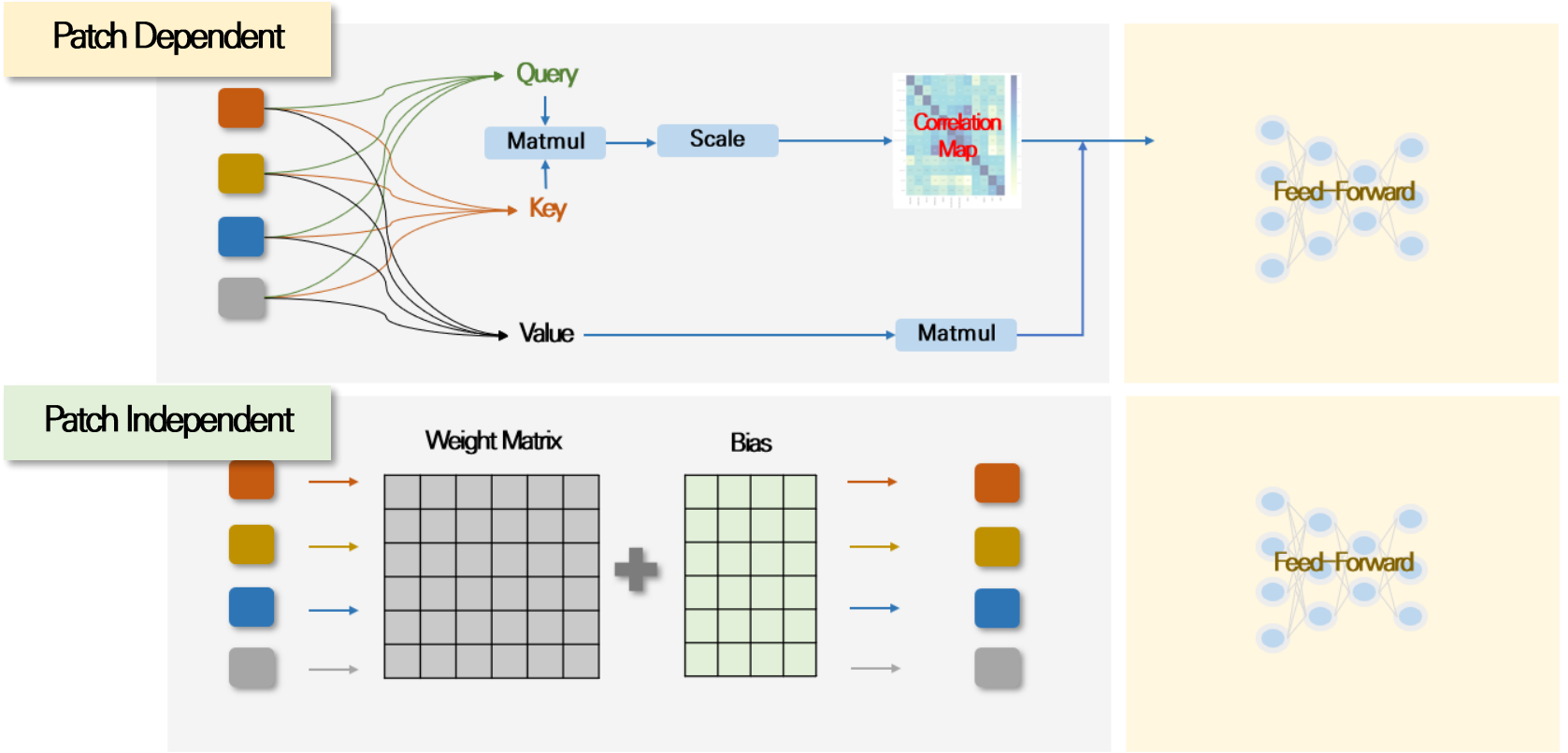


Method

Learning to Embed Time Series Patches Independently

❖ PI(Patch Independent) Structure

- Patch들의 dependencies를 고려하지 않고, patch 독립적으로 학습하게 하는 Patch Independent 구조 제안
- 기존 Patch 기반 모델들이 사용하던 Attention 기반 Transformer가 아닌 Simple MLP로 시계열 표현 학습



Patch 간의 유사도를 이용
Patch Dependence

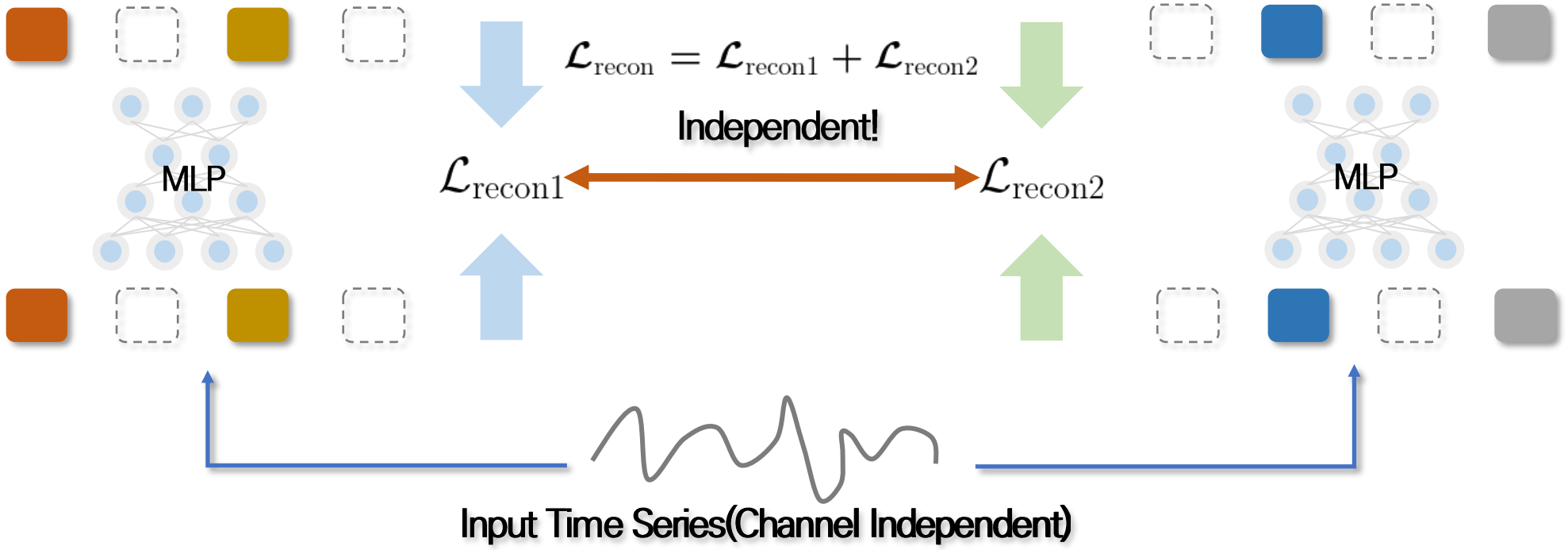
Patch 간의 관계 고려 X
Patch Independence

Method

Learning to Embed Time Series Patches Independently

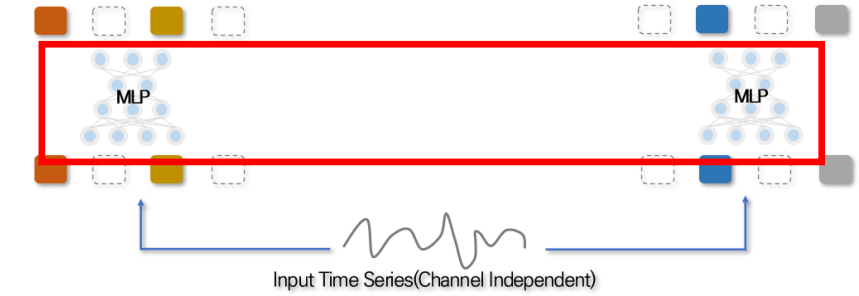
❖ PITS

- Unmasked Prediction: 일반적으로 Masking된 것을 예측하는 표현 학습 방법론과 달리, Masking되지 않은 것을 **현재 Patch만으로 복원**하도록 하는 Unmasked Prediction 제안
- Complementary Contrastive Learning: Mask로 만들어낸 두 view를 **추가적인 contrastive learning으로 학습**



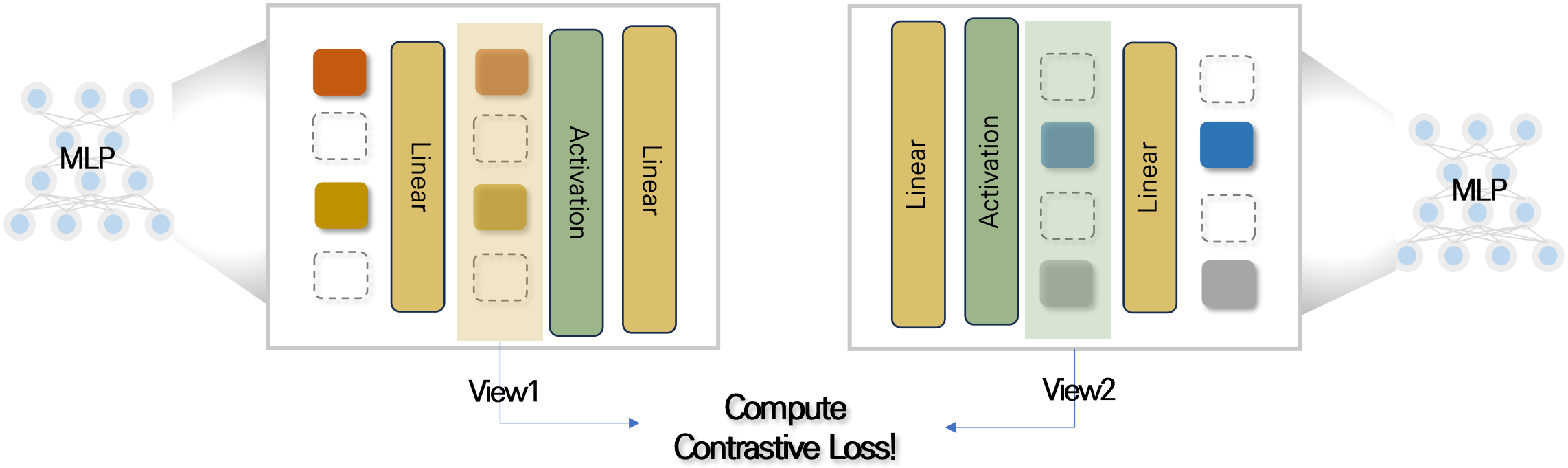
Method

Learning to Embed Time Series Patches Independently



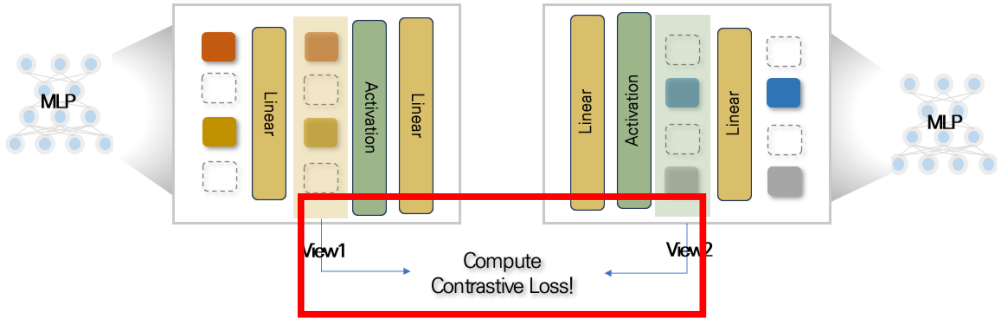
❖ PITS

- Unmasked Prediction: 일반적으로 Masking된 것을 예측하는 표현 학습 방법론과 달리, Masking되지 않은 것을 **현재 Patch만으로 복원**하도록 하는 Unmasked Prediction 제안
- Complementary Contrastive Learning: Mask로 만들어낸 두 view를 **추가적인 contrastive learning으로 학습**



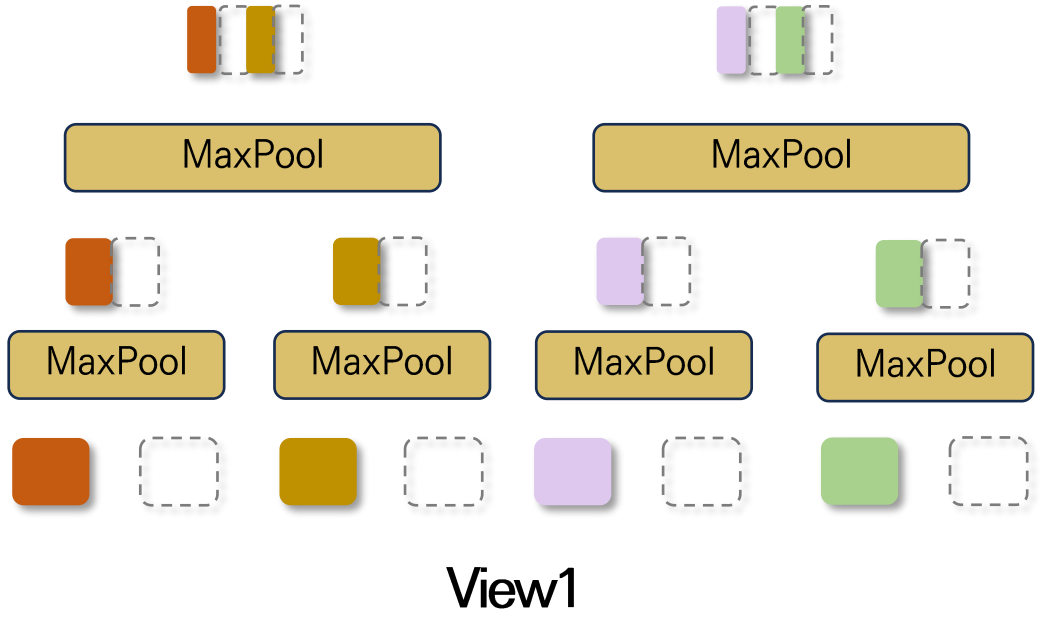
Method

Learning to Embed Time Series Patches Independently

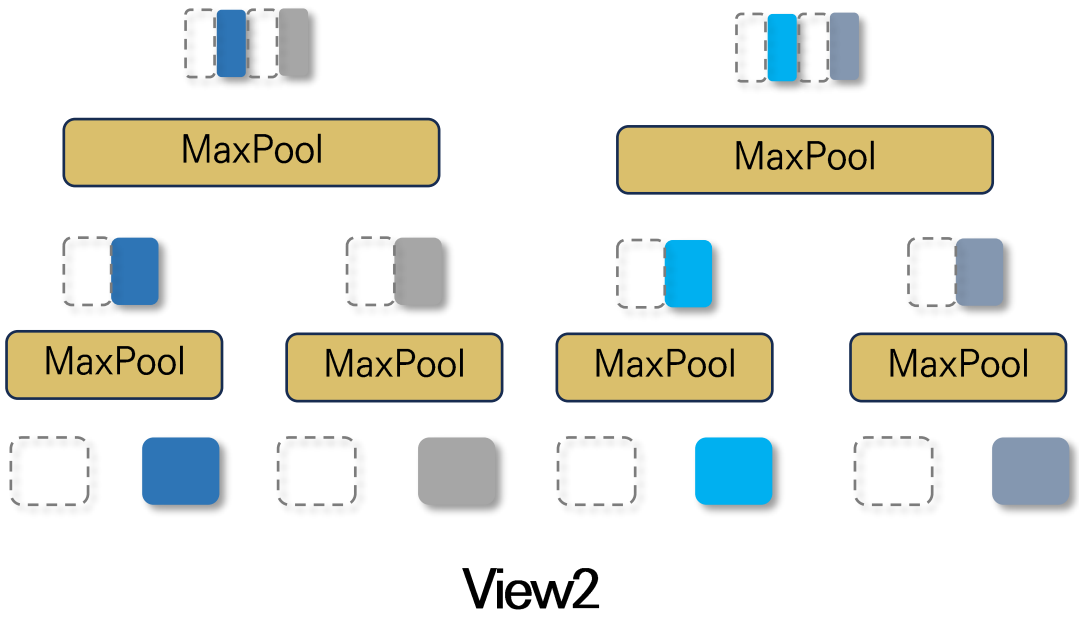


❖ Complementary Contrastive Loss

- Patch representation을 시간축에 대해서 max-pooling시키고, 각 단계에서 loss를 통합
- 이는 mask를 통해 생성된 두 view에 의해 이루어지며, 상대적 유사도를 softmax 확률로 계산



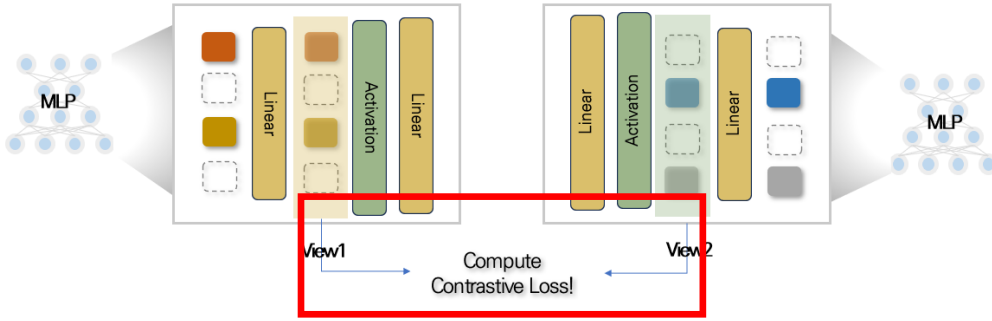
View1



View2

Method

Learning to Embed Time Series Patches Independently



❖ Complementary Contrastive Loss

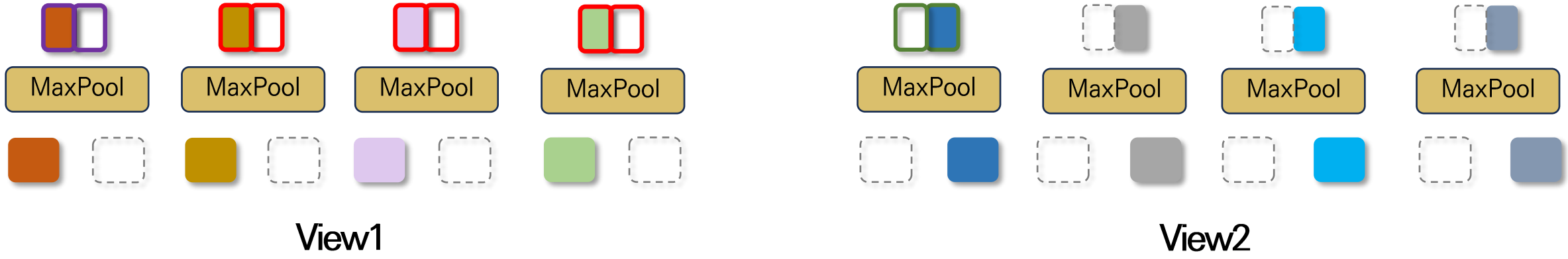
- Patch representation을 시간축에 대해서 max-pooling시키고, 각 단계에서 loss를 통합
- 이는 mask를 통해 생성된 두 view에 의해 이루어지며, 상대적 유사도를 softmax 확률로 계산

Anchor

Negative

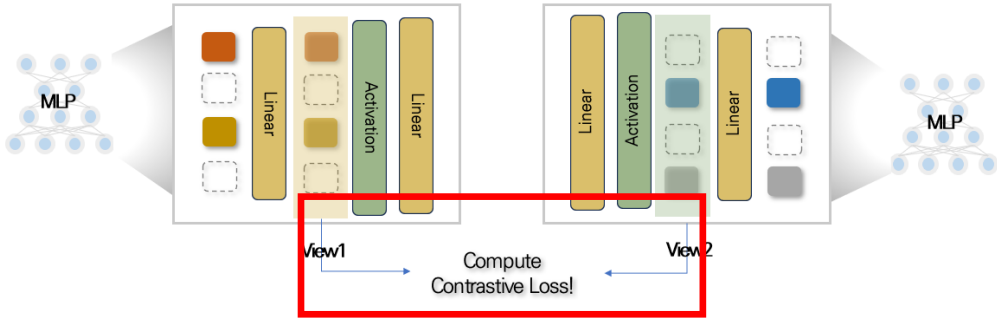
Positive

$$p(i,c,(n,n')) = \frac{\exp\left(z_1^{(i,c,n)} \circ z_1^{(i,c,n')}\right)}{\sum_{s=1,s\neq n}^{2N} \exp\left(z_1^{(i,c,n)} \circ z_1^{(i,c,s)}\right)},$$



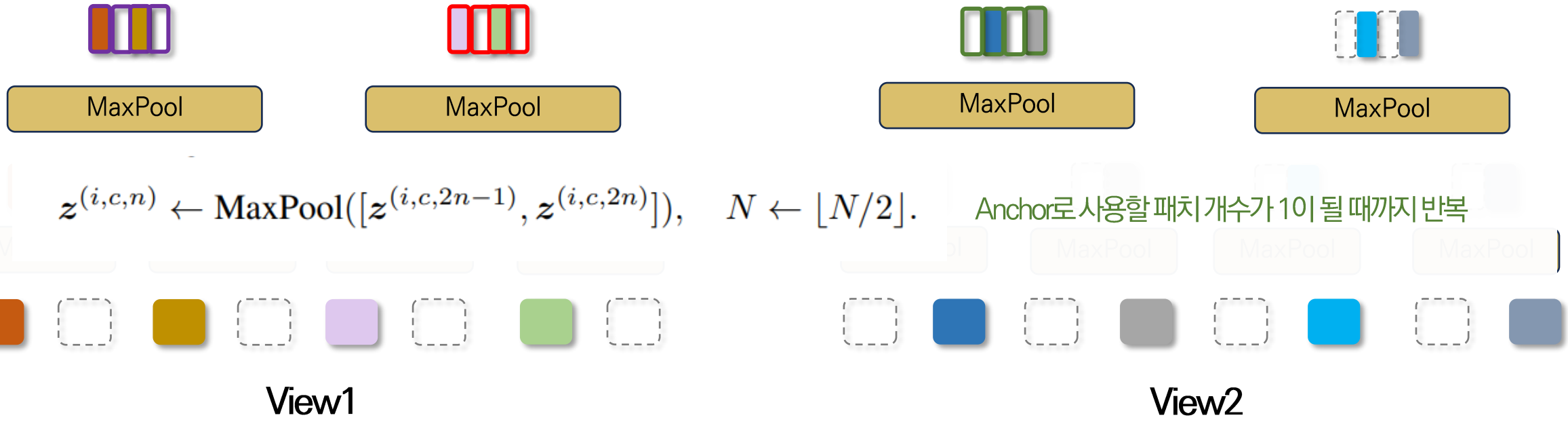
Method

Learning to Embed Time Series Patches Independently



❖ Complementary Contrastive Loss

- Patch representation을 시간축에 대해서 max-pooling시키고, 각 단계에서 loss를 통합
- 이는 mask를 통해 생성된 두 view에 의해 이루어지며, 상대적 유사도를 softmax 확률로 계산



$$z^{(i,c,n)} \leftarrow \text{MaxPool}([z^{(i,c,2n-1)}, z^{(i,c,2n)}]), \quad N \leftarrow \lfloor N/2 \rfloor.$$

Anchor로 사용할 패치 개수가 1이 될 때까지 반복

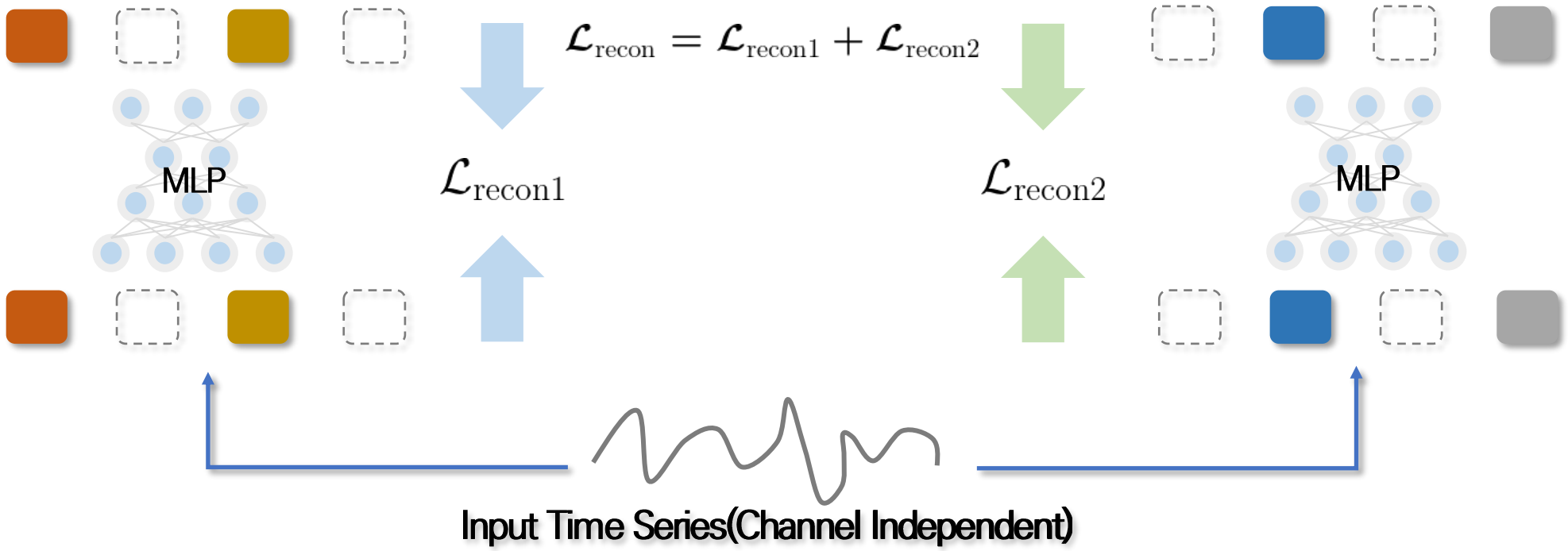
View1 View2

Method

Learning to Embed Time Series Patches Independently

❖ Loss function

- Patch reconstruction loss: 다른 패치를 보지 않고 패치를 복원하여 reconstruction loss 측정
- Contrastive loss: 다른 view에 있는 인접 시계열 정보를 계층적으로 반영

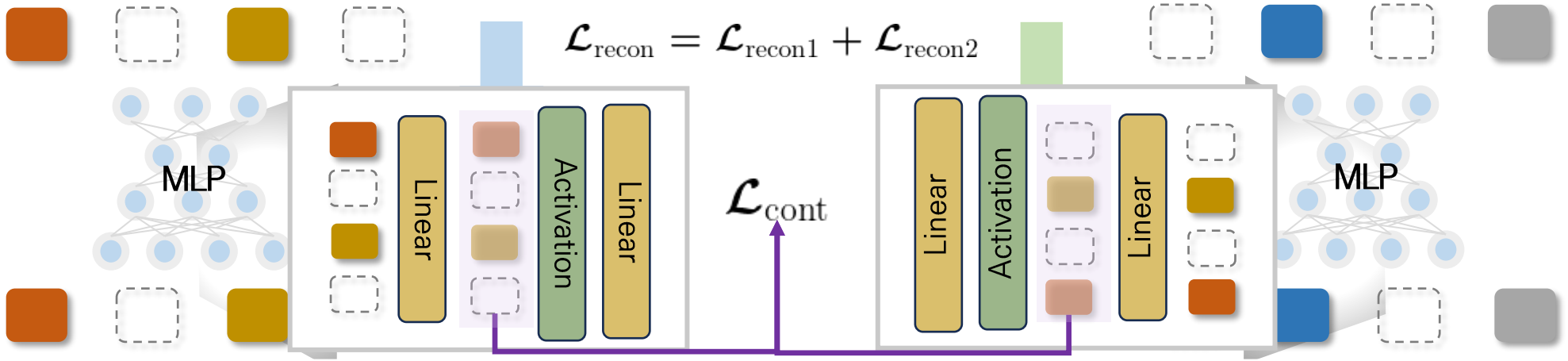


Method

Learning to Embed Time Series Patches Independently

❖ Loss function

- Patch reconstruction loss: 다른 패치를 보지 않고 패치를 복원하여 reconstruction loss 측정
- Contrastive loss: 다른 view에 있는 인접 시계열 정보를 계층적으로 반영
- 최종 Loss function은 reconstruction loss와 contrastive loss의 합으로 계산



최종 Self-supervised Loss $\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{cont}}$

Method

Learning to Embed Time Series Patches Independently

❖ **Experimental Results: Multivariate Time Series Forecasting**

- 어떤 세팅에서도 시계열 예측 SOTA 모델 대비 뛰어난 성능 기록
- 데이터셋이 바뀌는 Transfer Learning 시나리오에서도 타 self-supervised 방법론 대비 우수한 성능

Models		Self-supervised								Supervised													
		PITS		PITS w/o CL		PatchTST*		SimMTM†		PITS		PatchTST		SimMTM†		DLinear		TSMixer		FEDformer		Autoformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.367**	**0.393**	**0.367**	**0.393**	0.379	0.408	**0.367**	0.402	0.369	0.397	0.375	0.399	0.380	0.412	0.375	0.399	**0.361**	**0.392**	0.376	0.415	0.435	0.446
	192	0.401	0.416	**0.400**	**0.413**	0.414	0.428	0.403	0.425	**0.403**	**0.416**	0.414	0.421	0.416	0.434	0.405	**0.416**	0.404	0.418	0.423	0.446	0.456	0.457
	336	**0.415**	**0.428**	0.425	0.430	0.435	0.446	**0.415**	0.430	**0.409**	**0.426**	0.431	0.436	0.448	0.458	0.439	0.443	0.420	0.431	0.444	0.462	0.486	0.487
	720	**0.425**	**0.452**	0.444	0.459	0.468	0.474	0.430	0.453	0.456	**0.465**	**0.449**	0.466	0.481	0.469	0.472	0.490	0.463	0.472	0.469	0.492	0.515	0.517
ETTh2	96	**0.269**	**0.333**	**0.269**	0.334	0.306	0.351	0.288	0.347	0.281	0.343	**0.274**	**0.336**	0.325	0.374	0.289	0.353	**0.274**	0.341	0.332	0.374	0.332	0.368
	192	**0.329**	**0.371**	0.332	0.375	0.361	0.392	0.346	0.385	0.345	0.383	**0.339**	**0.379**	0.400	0.424	0.383	0.418	0.339	0.385	0.407	0.446	0.426	0.434
	336	**0.356**	**0.397**	0.362	0.400	0.405	0.427	0.363	0.401	0.334	0.389	**0.331**	**0.380**	0.405	0.433	0.448	0.465	0.361	0.406	0.400	0.447	0.477	0.479
	720	**0.383**	**0.425**	0.385	0.428	0.419	0.446	0.396	0.431	0.389	0.430	**0.379**	**0.422**	0.451	0.475	0.605	0.551	0.445	0.470	0.412	0.469	0.453	0.490
ETTm1	96	0.294	0.354	0.303	0.351	0.294	0.345	**0.289**	**0.343**	0.295	0.346	0.290	0.342	0.296	0.346	0.299	0.343	**0.285**	**0.339**	0.326	0.390	0.510	0.492
	192	**0.321**	0.373	0.338	0.371	0.327	**0.369**	0.323	**0.369**	0.330	0.369	0.332	0.369	0.333	0.374	0.335	0.365	**0.327**	0.365	0.365	0.415	0.514	0.495
	336	0.359	0.388	0.365	**0.384**	0.364	0.390	**0.349**	0.385	0.360	0.388	0.366	0.392	0.370	0.398	0.369	0.386	**0.356**	**0.382**	0.392	0.425	0.510	0.492
	720	**0.396**	**0.414**	0.420	0.415	0.409	0.415	0.399	0.418	**0.416**	0.421	0.420	0.424	0.427	0.431	0.425	0.421	0.419	**0.414**	0.446	0.458	0.527	0.493
ETTm2	96	0.165	0.260	**0.160**	**0.253**	0.167	0.256	0.166	0.257	**0.163**	0.255	0.165	0.255	0.175	0.268	0.167	0.260	**0.163**	**0.252**	0.180	0.271	0.205	0.293
	192	**0.213**	**0.291**	**0.213**	**0.291**	0.232	0.302	0.223	0.295	**0.215**	0.293	0.220	0.292	0.240	0.312	0.224	0.303	0.216	**0.290**	0.252	0.318	0.278	0.336
	336	**0.263**	**0.325**	**0.263**	**0.325**	0.291	0.342	0.282	0.334	**0.266**	0.329	0.278	0.329	0.298	0.351	0.281	0.342	0.268	**0.324**	0.324	0.364	0.343	0.379
	720	**0.337**	**0.373**	0.339	0.375	0.368	0.390	0.370	0.385	**0.342**	**0.380**	0.367	0.385	0.403	0.413	0.397	0.421	0.420	0.422	0.410	0.420	0.414	0.419
Weather	96	0.151	0.201	0.154	0.205	**0.146**	**0.194**	0.151	0.202	0.154	0.202	0.152	0.199	0.166	0.216	0.176	0.237	**0.145**	**0.198**	0.238	0.314	0.249	0.329
	192	0.195	0.242	0.200	0.247	**0.192**	**0.238**	0.223	0.295	**0.191**	**0.242**	0.197	0.243	0.208	0.254	0.220	0.282	**0.191**	**0.242**	0.275	0.329	0.325	0.370
	336	**0.244**	**0.280**	0.245	0.282	0.245	**0.280**	0.246	0.283	0.245	**0.280**	0.249	0.283	0.257	0.290	0.265	0.319	**0.242**	**0.280**	0.339	0.377	0.351	0.391
	720	0.314	**0.330**	**0.312**	**0.330**	0.320	0.336	0.320	0.338	**0.309**	0.330	0.320	0.335	0.326	0.338	0.323	0.362	0.320	0.336	0.389	0.409	0.415	0.426
Traffic	96	0.372	**0.258**	0.374	0.266	0.393	0.275	**0.368**	0.262	0.375	0.264	**0.367**	**0.251**	0.471	0.309	0.410	0.282	0.376	0.264	0.576	0.359	0.597	0.371
	192	0.396	0.271	0.395	0.270	0.376	0.254	**0.373**	**0.251**	0.389	0.270	**0.385**	**0.259**	0.475	0.308	0.423	0.287	0.397	0.264	0.610	0.380	0.607	0.382
	336	0.411	0.280	0.408	0.277	0.384	0.259	**0.395**	**0.254**	0.401	0.277	**0.398**	**0.265**	0.490	0.315	0.436	0.296	0.413	0.290	0.608	0.375	0.623	0.387
	720	0.436	**0.290**	0.436	**0.290**	0.446	0.306	**0.432**	**0.290**	0.437	0.294	**0.434**	**0.287**	0.524	0.332	0.466	0.315	0.444	0.306	0.621	0.375	0.639	0.395
Electricity	96	0.130	0.225	0.131	0.226	**0.126**	**0.221**	0.133	0.223	0.132	0.228	**0.130**	**0.222**	0.190	0.279	0.140	0.237	0.131	0.229	0.186	0.302	0.196	0.313
	192	**0.144**	0.240	0.145	0.240	0.145	0.238	0.147	**0.237**	**0.147**	0.242	0.148	**0.240**	0.195	0.285	0.153	0.249	0.151	0.246	0.197	0.311	0.211	0.324
	336	**0.160**	**0.256**	0.162	**0.256**	0.164	**0.256**	0.166	0.265	0.162	**0.261**	0.167	0.261	0.211	0.301	0.169	0.267	**0.161**	0.261	0.213	0.328	0.214	0.327
	720	**0.194**	**0.287**	0.201	0.290	0.200	0.290	0.203	0.297	0.199	**0.290**	0.202	0.291	0.253	0.333	0.203	0.301	**0.197**	0.293	0.233	0.344	0.236	0.342
Average		**0.301**	**0.327**	**0.304**	**0.328**	0.314	0.333	0.306	0.331	**0.304**	0.329	0.307	**0.327**	0.343	0.355	0.332	0.351	0.311	0.333	0.373	0.386	0.412	0.409

〈Multivariate Time Series Forecasting Results〉

	Source	Target	PITS FT	PITS LP	PatchTST FT	PatchTST LP	SimMTM	TimeMAE	TST	LaST	TF-C	CoST
In-domain	ETTh2	ETTh1	0.404	**0.403**	0.423	0.464	0.415	0.728	0.645	0.443	0.635	0.584
	ETTm2	ETTm1	**0.345**	0.354	0.348	0.411	0.351	0.682	0.480	0.414	0.758	0.354
	Average		**0.375**	0.378	0.386	0.406	0.383	0.705	0.563	0.429	0.697	0.469
Cross-domain	ETTm2	ETTh1	0.407	**0.405**	0.433	0.421	0.428	0.724	0.632	0.503	1.091	0.582
	ETTh2	ETTm1	**0.350**	0.357	0.363	0.378	0.365	0.688	0.472	0.475	0.750	0.377
	ETTm1	ETTh1	**0.409**	**0.409**	0.447	0.432	0.422	0.726	0.645	0.426	0.700	0.750
	ETTh1	ETTm1	0.352	0.357	0.348	0.374	**0.346**	0.666	0.482	0.353	0.746	0.359
	Weather	ETTh1	**0.406**	**0.406**	0.437	0.423	0.456	-	-	-	-	-
	Weather	ETTm1	0.350	0.356	**0.348**	0.355	0.358	-	-	-	-	-
	Average		**0.379**	0.382	0.396	0.397	0.396	-	-	-	-	-

〈Transfer Learning Results〉

Method

Learning to Embed Time Series Patches Independently

❖ Experimental Results: Time Series Classification

- Time Series Classification Task에서도 타 self-supervised 방법론 대비 좋은 성능
- 여러 task에서 transfer learning의 성능 입증 -> **도메인 변화에 강건한 방법론임을 입증**

	ACC.	PRE.	REC.	F_1
TS2Vec	92.17	93.84	81.19	85.71
CoST	88.07	91.58	66.05	69.11
LaST	92.11	93.12	81.47	85.74
TF-C	93.96	94.87	85.82	89.46
TST	80.21	40.11	50.00	44.51
TimeMAE	80.34	90.16	50.33	45.20
SimMTM	94.75	95.60	89.93	91.41
PITS w/o CL	95.27	95.35	95.27	95.30
PITS	**95.67**	**95.63**	**95.67**	**95.64**

Table 5: Results of TSC.

	In-domain transfer learning				Cross-domain transfer learning											
	SleepEEG → Epilepsy				SleepEEG → FD-B				SleepEEG → Gesture				SleepEEG → EMG			
	ACC.	PRE.	REC.	F_1	ACC.	PRE.	REC.	F_1	ACC.	PRE.	REC.	F_1	ACC.	PRE.	REC.	F_1
TS-SD	89.52	80.18	76.47	77.67	55.66	57.10	60.54	57.03	69.22	66.98	68.67	66.56	46.06	15.45	33.33	21.11
TS2Vec	93.95	90.59	90.39	90.45	47.90	43.39	48.42	43.89	69.17	65.45	68.54	65.70	78.54	80.40	67.85	67.66
CoST	88.40	88.20	72.34	76.88	47.06	38.79	38.42	34.79	68.33	65.30	68.33	66.42	53.65	49.07	42.10	35.27
LaST	86.46	90.77	66.35	70.67	46.67	43.90	47.71	45.17	64.17	70.36	64.17	58.76	66.34	79.34	63.33	72.55
Mixing-Up	80.21	40.11	50.00	44.51	67.89	71.46	76.13	72.73	69.33	67.19	69.33	64.97	30.24	10.99	25.83	15.41
TS-TCC	92.53	94.51	81.81	86.33	54.99	52.79	63.96	54.18	71.88	71.35	71.67	69.84	78.89	58.51	63.10	59.04
TF-C	94.95	94.56	89.08	91.49	69.38	75.59	72.02	74.87	76.42	77.31	74.29	75.72	81.71	72.65	81.59	76.83
TST	80.21	40.11	50.00	44.51	46.40	41.58	45.50	41.34	69.17	66.60	69.17	66.01	46.34	15.45	33.33	21.11
TimeMAE	89.71	72.36	67.47	68.55	70.88	66.98	68.94	66.56	71.88	70.35	76.75	68.37	69.99	70.25	63.44	70.89
SimMTM	95.49	93.36	92.28	92.81	69.40	74.18	76.41	75.11	80.00	79.03	80.00	78.67	97.56	98.33	98.04	98.14
PITS	**95.71**	**95.69**	**95.71**	**95.70**	**88.65**	**88.86**	**88.65**	**88.63**	**92.50**	**93.32**	**92.50**	**92.48**	**100.0**	**100.0**	**100.0**	**100.0**

Table 6: Results of TSC with transfer learning.

Conclusion

❖ **시계열 예측 딥러닝 모델의 패러다임 변화**

- Linear 모델을 사용한 **LTSF-Linear**, Patch 형태를 사용한 PatchTST가 시계열 예측 성능을 끌여올림
- 세부 요소들에 대해 발전시킨 방법론 출현

❖ **TSMixer: An All-MLP Architecture for Time Series Forecasting**

- Linear Model이 시계열의 일반적인 형태에서 안정적으로 작용할 수 있음을 이론적으로 증명
- 변수들 간의 상관 관계, 보조 정보를 활용한 MLP-only Architecture 제시

❖ **iTransformer: Inverted Transformers Are Effective for Time Series Forecasting**

- 시간 축에 대해서 임베딩 도입, Multi-Head Attention에서 명시적으로 변수 간 상관 관계 고려하여 예측 성능 향상

❖ **Learning to Embed Time Series Patches Independently**

- 패치 독립적으로 학습시키는 전략을 통해서 self-supervised learning 및 transfer learning에서 독립 전략 제시

고맙습니다