Efficient Time Series Prediction Model for Embedded or Mobile System

Mobile and Ubiquitous Computing Term Project

Dongjun Lee(2023-22406), Minjun Park (2023-20349)

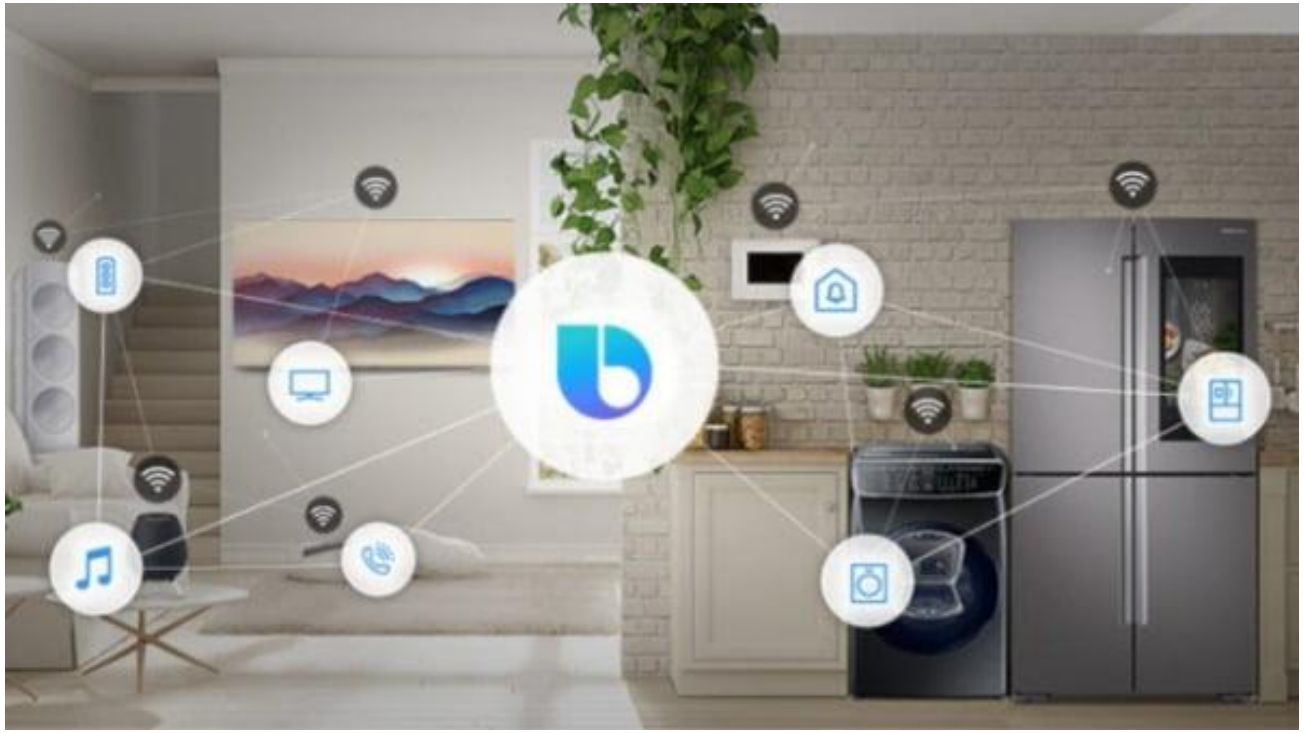
Seoul National University

Data Science & Artificial Intelligence Laboratory

DSAIL
Data Science & AI Laboratory

Motivation

Embedded systems are the most pervasive AI platform.



- 세탁기
- 냉장고
- 에어컨
- 조명
- 식기세척기
- AI 스피커

Motivation

Key features of embedded systems

- Multiple sensors: Multivariate

- Streaming data (real-time, continuous inference)

  ➔ Time series ML

- Low power & Limited resource

  ➔ Efficient model

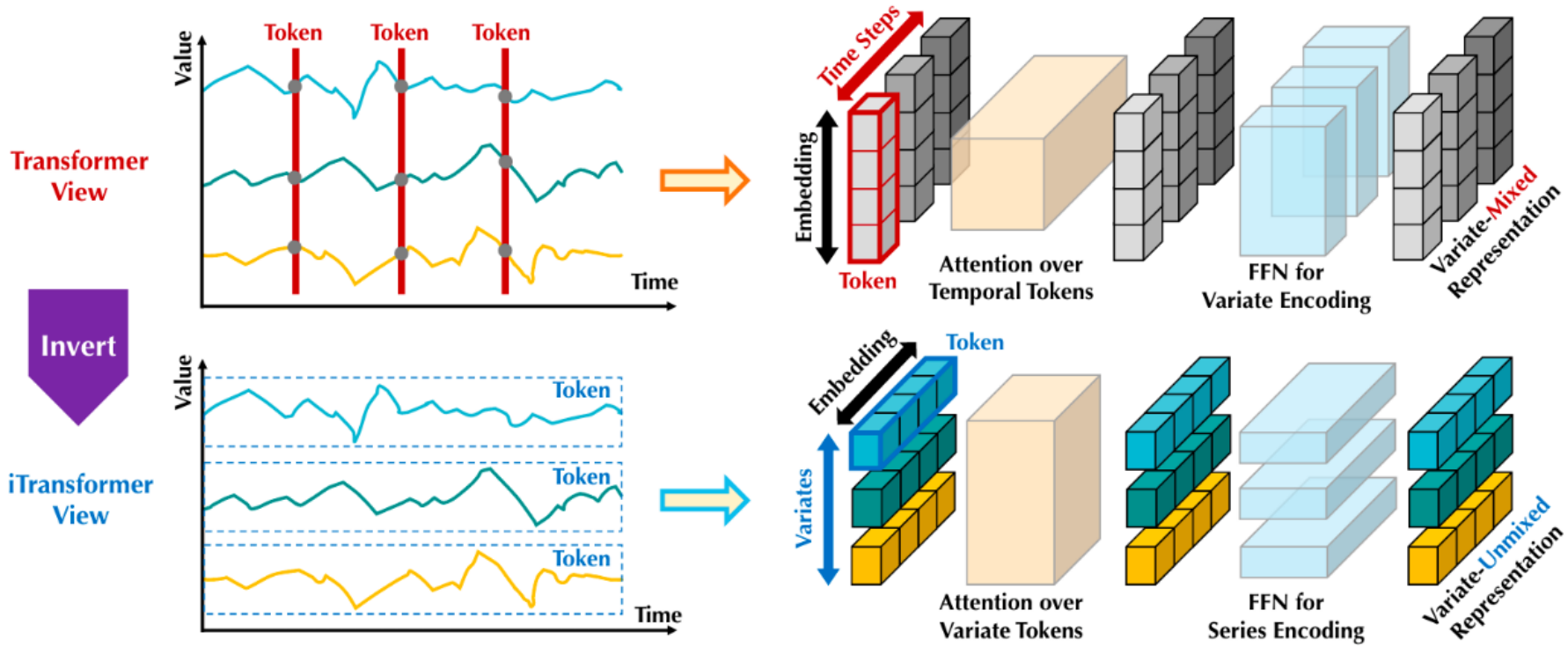
Motivation

- 1. Cross-variable information
  2. Temporal Information → Our focus!

- Long-term Temporal Information은 Forecasting Acc.를 위한 필수적인 요소

- Mobile & Embedded System
  - High sampling rate & memory and latency constraint → More vulnerable!

Motivation



Implicit Modeling; iTransformer

- Temporal information is encoded as a single token with a single linear layer.
- Channel-wise attention only

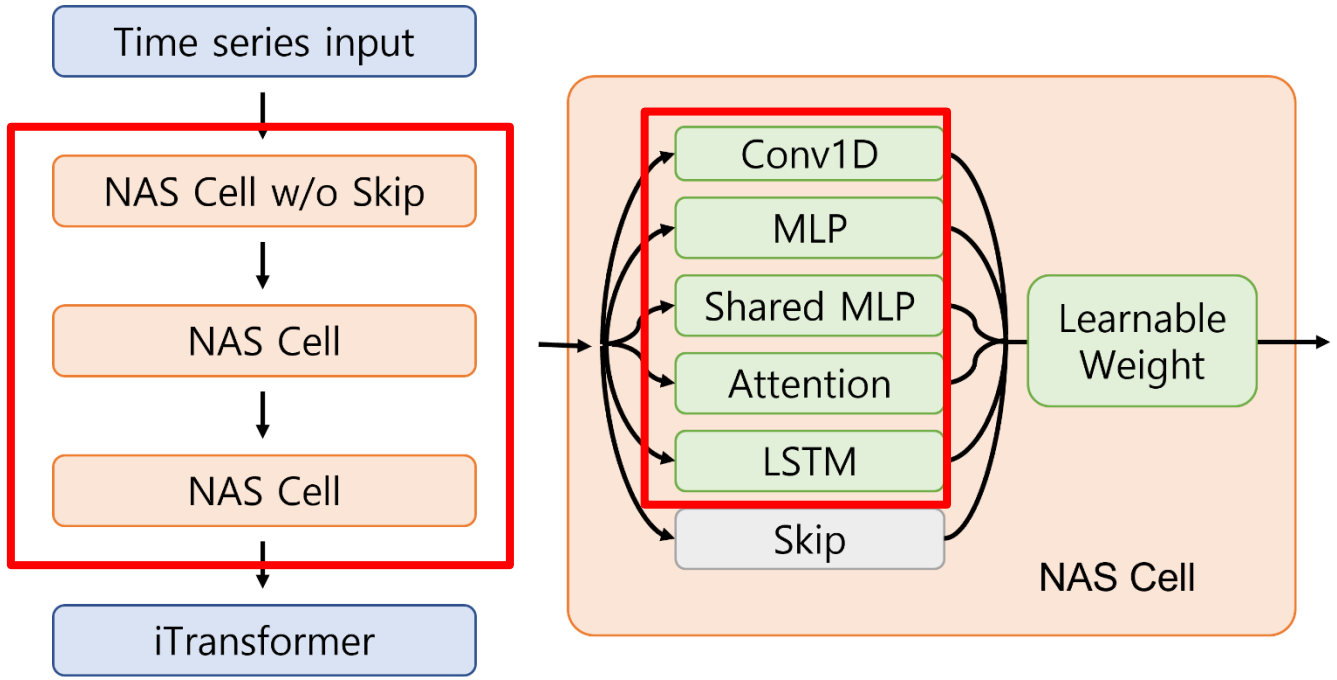
Motivation

iTransformer is efficient! But…

- Assumption: temporal dependency can be encoded with a linear model
  - ➔ lacks expressiveness
  - ➔ following transformer blocks should be powerful
  - ➔ Enhanced embedding layer → light transformer layers (NAS)

- Lack of modeling Long-range dependency
  - ➔ Longest dependency == look-back window size
  - ➔ Larger window == more cost & does not guarantee performance increase
  - ➔ Efficient long-range encoding with minimal input window size (Spectral Attention)

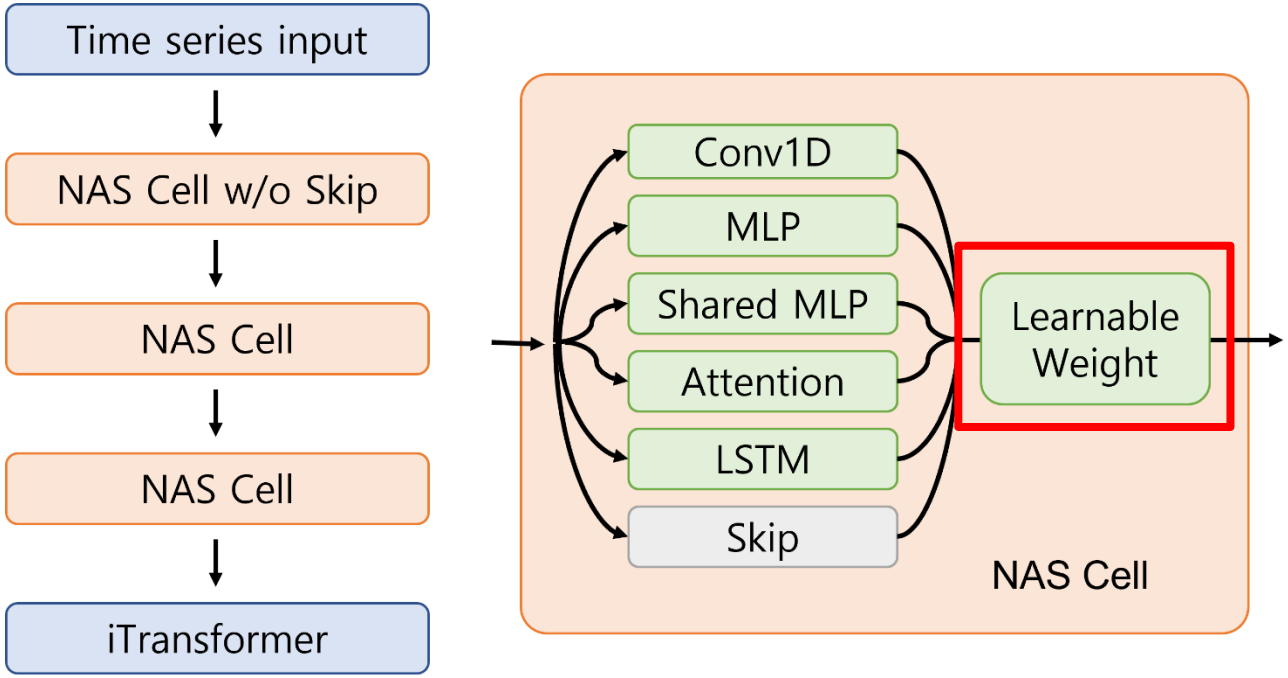
Method (1) NAS for Temporal Encoder



Cell-level search space

- Traditional temporal encoder
  - Conv1D
  - LSTM
- MLP Based
  - Shared MLP (iTransformer default) different channels share single MLP
  - MLP: diff. channel → independent
- Patch-based attention
  - Time slice → patchify
  - Self attention to encode

Method (1) NAS for Temporal Encoder



Gradient search strategy

- All components can be loaded on a single GPU (😁 iTransformer 👍)
- Consider all combinations with a single training

Performance Estimation Metric

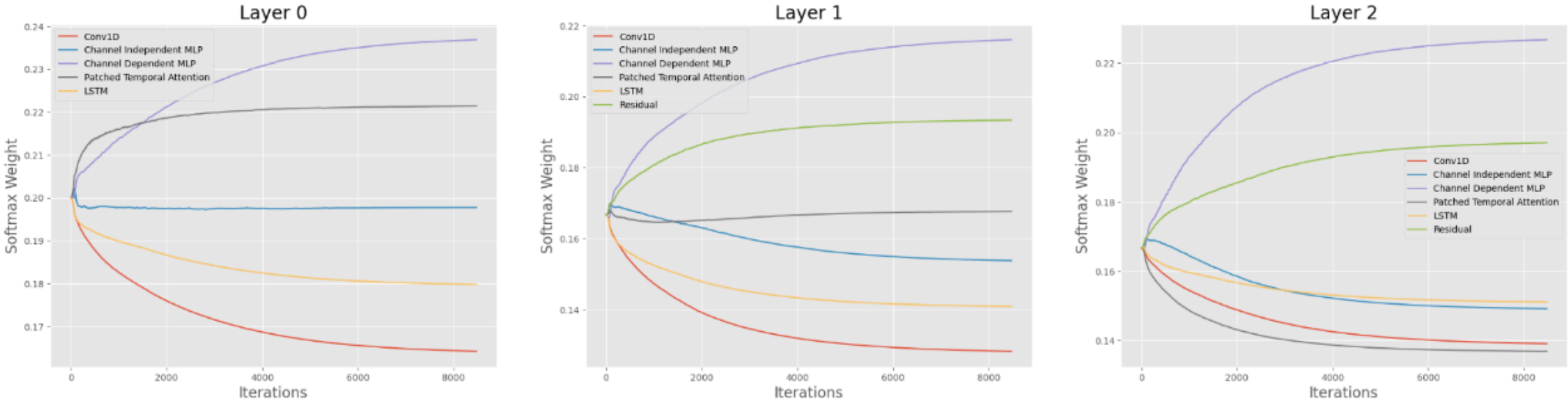
- Prediction performance
  - MAE / MSE
- Efficiency metric
  - MACs

Evaluation

Table 1: Performance of NAS searched model after finetune.

		Energy Data		Weather	
		MAE	MSE	MAE	MSE
Original iTransformer		0.4736	0.4825	0.2163	0.1761
LASSO	NAS Search	0.4499	0.4465	0.2047	0.1552
	top-1 module only	0.4481	0.4437	0.2073	0.161
	top-2 modules	0.4487	0.446	0.2037	0.1554
Softmax	NAS Search	0.4521	0.4527	0.2054	0.1558
	top-1 module only	0.4631	0.468	0.2082	0.161
	top-2 modules	0.4564	0.4611	0.2032	0.1546

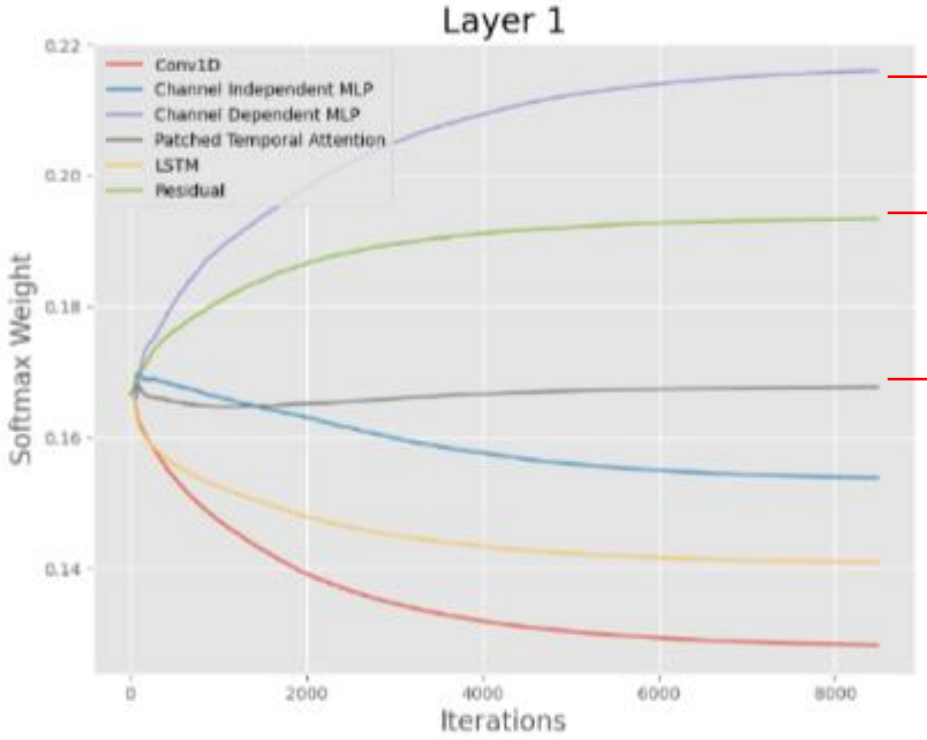
Discussion



Residual module (Skip) is the second dominant module in layer 1 & 2

We can adopt less layers for much harsh HW constraints

Discussion



MLP

Residual

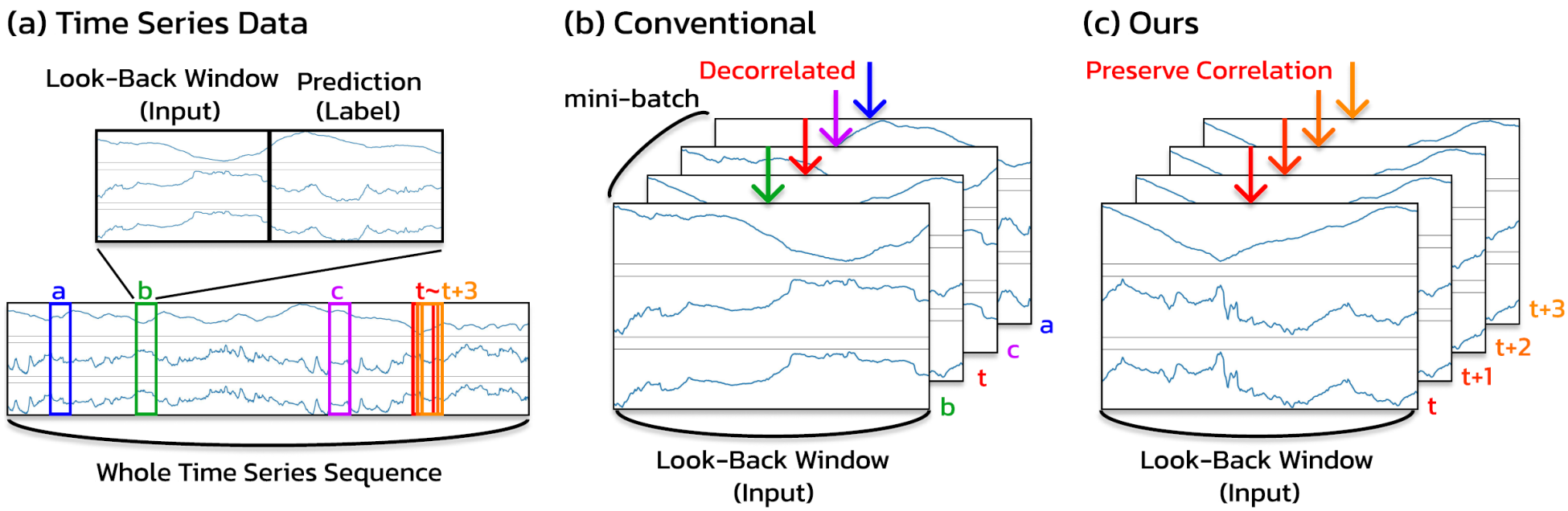
Patched Temporal Attention

The most expressive ➔ Patched temporal attention

The lightest ➔ Residual

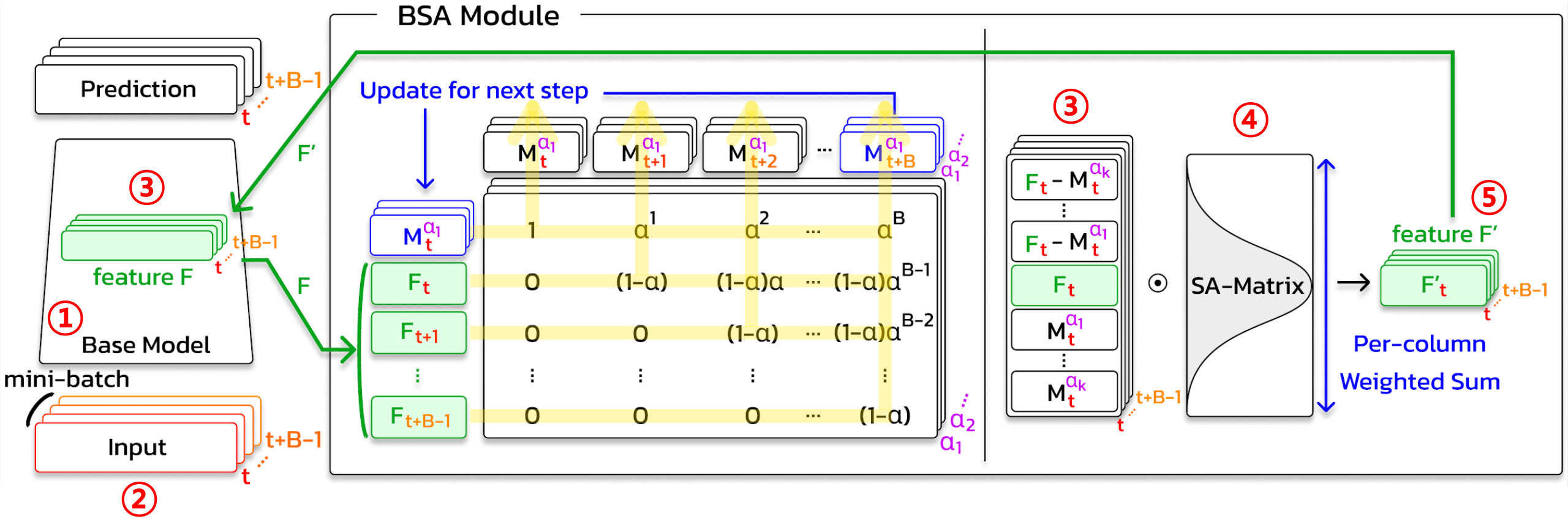
NAS selected MLP which stands in between

Method (2) Spectral Attention



- Utilize neglected information from highly correlated sequential inputs!

Method (2) Spectral Attention



Evaluation

Table 2: Experiment results on Weather dataset. All experiments are conducted with three random seeds and the average value of the seeds is reported.

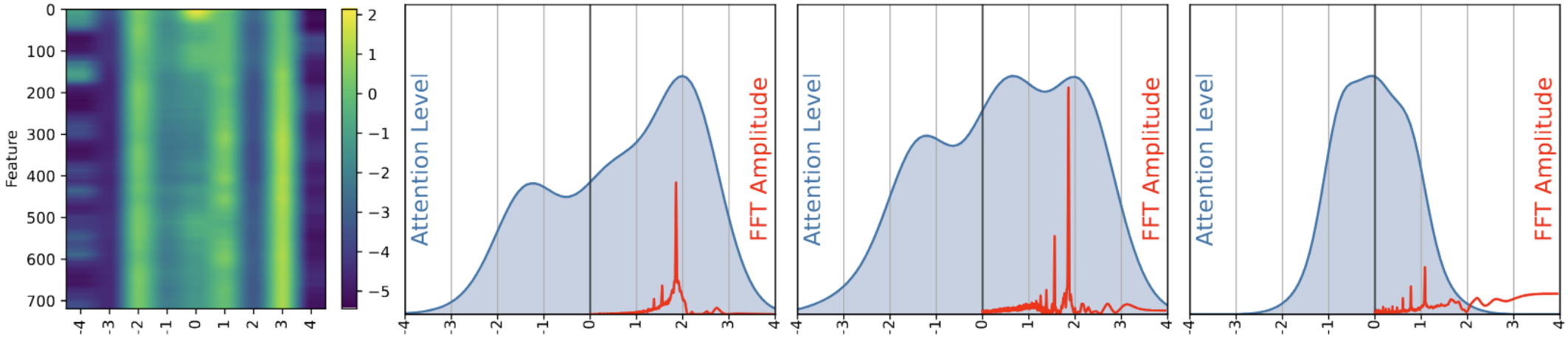
	BSA	input	output	e_layer	d_model	d_ff	# Param.	MACs	MSE	MAE
Baseline	X	96	96	3	512	512	4,833,888	42,656,352	0.17064	0.20957
Light	X	48	96	2	64	64	59,936	459,296	0.20470	0.23127
	O	**48**	**96**	**2**	**64**	**64**	**66,992**		**0.17042**	**0.21255**
	X	48	96	1	16	16	4,144	17,200	0.20951	0.23958
	O	**48**	**96**	**1**	**16**	**16**	**11,200**		**0.17685**	**0.22035**

Table 3: Experiment results on EnergyData dataset. All experiments are conducted with three random seeds and the average value of the seeds is reported.

	BSA	input	output	e_layer	d_model	d_ff	# Param.	MACs	MSE	MAE
Baseline	X	96	96	3	512	512	4,833,888	53,687,904	0.48430	0.46995
Light	X	48	96	1	16	16	4,144	21,008	0.60844	0.52195
	O	**48**	**96**	**1**	**16**	**16**	**13,552**		**0.48109**	**0.46742**

- BSA has user-defined non-typical operations so we couldn't get the exact MACs for Light + BSA

Model Inspection



(a) Heat map (b) Temp.(°C) (c) SWDR(W/m) (d) ETTh1-HULL

Figure 6: This figure illustrates the analysis of the SA-matrix trained on the prediction task for the Weather dataset. Panel (a) shows the heatmap of the SA-matrix, and (b)-(d) show the attention and FFT graphs.

Conclusion

- Improving Temporal Information Embedding in MTS Forecasting
  - NAS for embedding architecture search
  - Spectral Attention for learning long-term information over the input window

- Both leads to more smaller / efficient model while keeping the performance

Appendix

Method (1) NAS for Temporal Encoder

Rationale behind MACs

- MACs is NOT a good proxy metric for latency & efficiency
- We assume embedded systems
  - No sophisticated compute cores
  - Don’t support massive parallelism

  ➔ # computations is linearly correlated with latency
- MACs are weight-independent
  - Can be precomputed
  - Well-integrated with gradient-based NAS

Evaluation

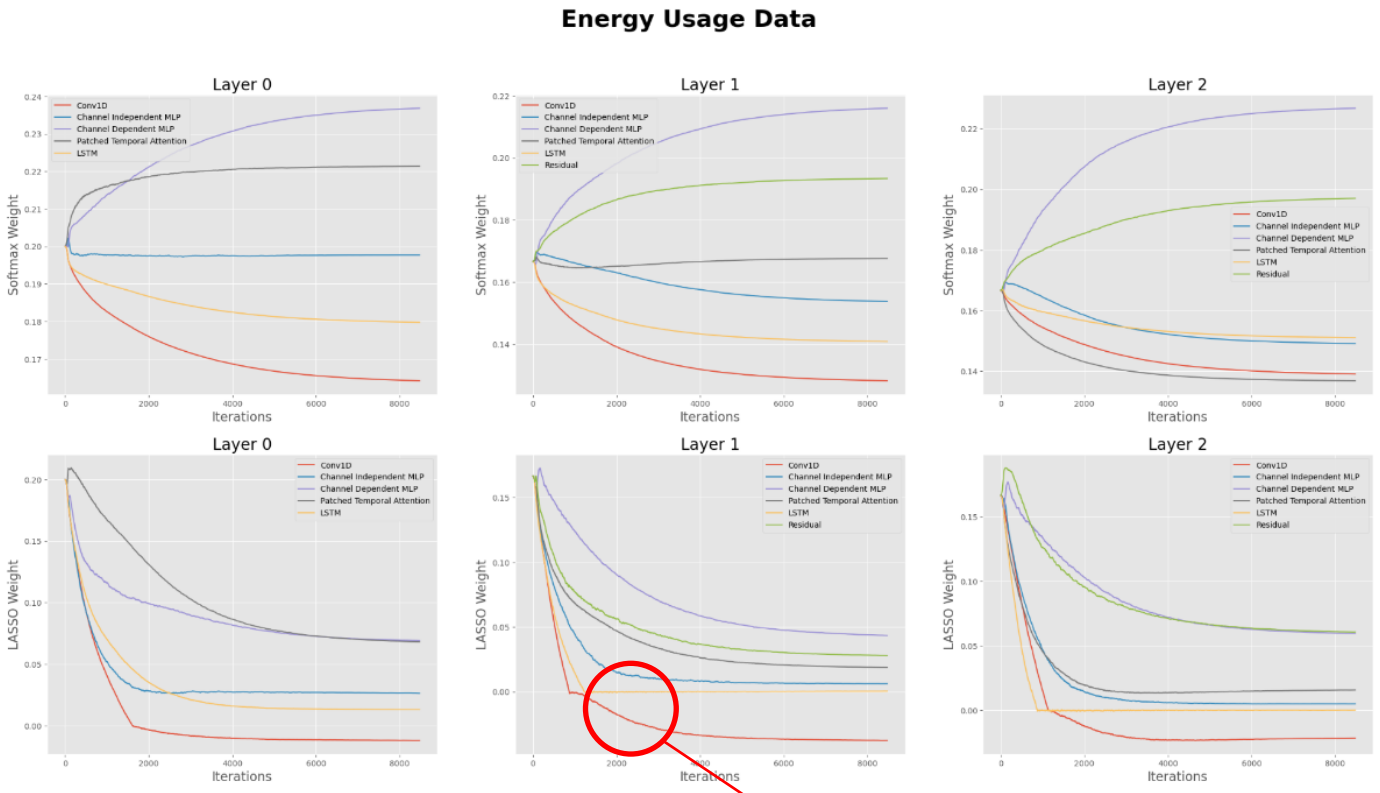


Figure 5: Trained weights of the modules via NAS. The upper row applied softmax to the learned weights. The lower row used the weights directly and employed a LASSO regularizer to enforce sparsity.

➔Apply softmax

- Probabilistic interpretation
- No sparsity regularization

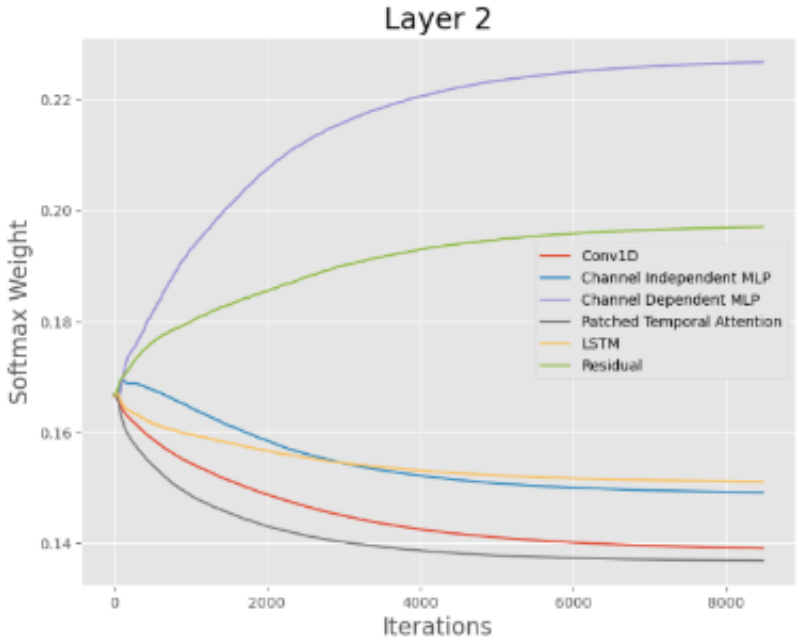
➔ Directly use weights

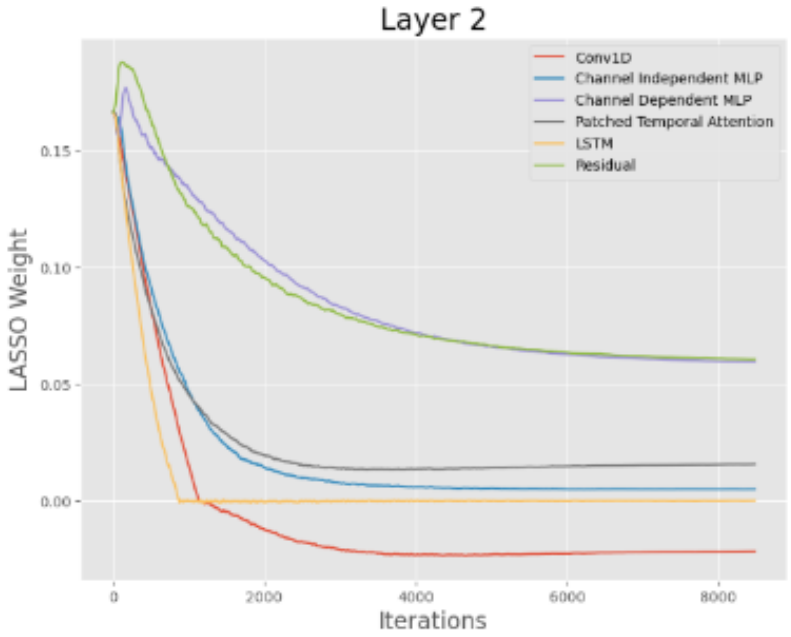
- LASSO for sparse feature selection
- No Probabilistic interpretation

Sudden architecture change!

But resultant models converge to the same performance

Discussion





LASSO & softmax gave different weights

BUT top-1 & top-2 modules are the same


Thank You!