PPT: Patch Order Do Matters In Time Series Pretext Task

Jaeho Kim, Kwangryeol Park, Sukmin Yun, Seulki Lee

Index

1. Introduction

- Challenges in Self-Supervised Time Series.

2. Methodology

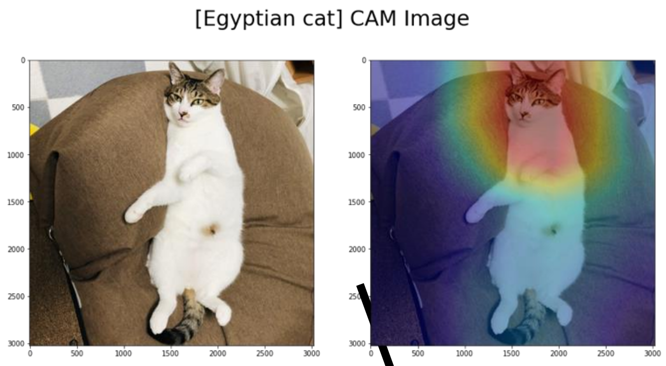
- PPT: Patch order do matters for time series.
- Consistency and contrastive order learning.
- ACF-CoS metric.

3. Results

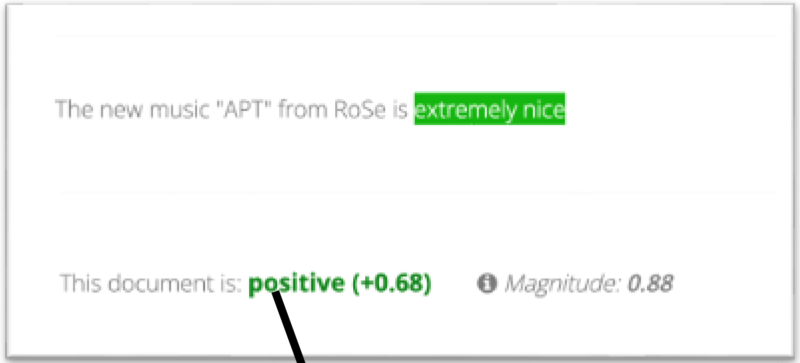
4. Conclusion

Introduction

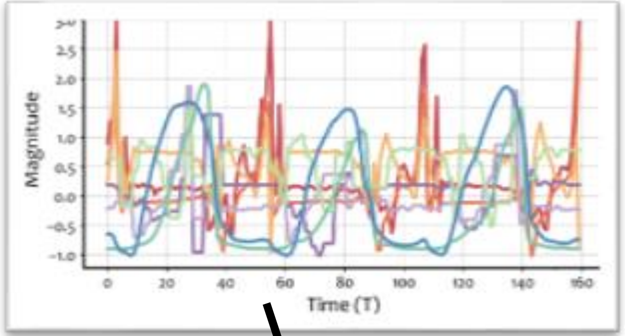
Time Series are Hard to Understand



- This is an image of a **cat.**
- It has **furs!**



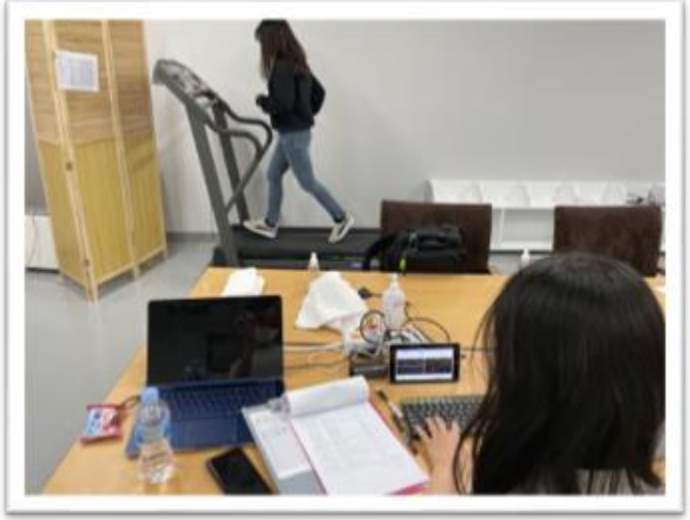
- The sentiment here is **positive.**
- It says **"extremely nice"**



- This is a time series of a ...?

- Unlike images or natural languages, time series are **hard to understand** as they are.
- They contain **signal information (lack semantics)** which is hard to interpret.
- They exhibit **temporal dependency** and **multi-channel characteristics.**

Overabundance of Unlabeled Time Series



- **Collecting labeled data for time series is expensive.**
  - As time series are **non-interpretable**, a labeler needs to be present during data collection.
  - Crowdsourced data labeling is hard for time series.
  - Most deep learning methodology assumes labeled data.

[2] Kim, Jaeho, et al. "Multitask Deep Learning for Human Activity, Speed, and Body Weight Estimation Using Commercial Smart Insoles." IEEE Internet of Things Journal 10.18 (2023): 16121-16133.

Self-Supervised Learning

- **What is Self-Supervised Learning (SSL)?**
  - The model learns from **unlabeled data** by creating its own supervisory signal.
  - This is done by creating a **pretext task.**
- **Pretext Task**
  - A pre-designed task for a network to solve in a self-supervised manner.

https://datasciencedojo.com/blog/data-science-memes/

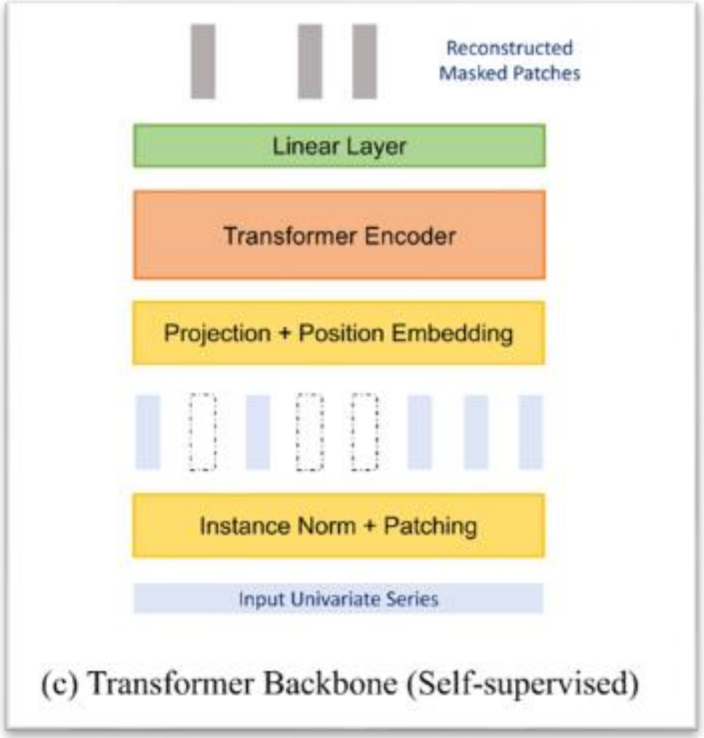
Research Questions

- **Research Points in Time Series SSL**
  - How can we design tasks that leverage the **temporal characteristics** of time series?
  - How can we design tasks that leverage the **inter-channel relationship** in time series?
  - How can we leverage the recent **patch-wise methodology** in time series?
  - How can we design better **pretext task** designed for time series?

Methodology

Motivation

- **Patch-based Representation in Time Series**
  - Gaining popularity in time series analysis.
    - Local semantic information is preserved.
    - Computation and memory usage is reduced.
  - PatchTST [2], PITS [3], Time-LLM [4], etc.,
- **Pretext Tasks: Mask and Reconstruction**
  - Predominant approach in time series analysis
  - Limitations
    - **Does not explicitly model the temporal and channel relationships in time series.**



[Figure] PatchTST SSL architecture

[2] A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. (ICLR 2023)
[3] Learning to Embed Time Series Patches Independently (ICLR 2024)
[4] Time-LLM: Time Series Forecasting by Reprogramming Large Language Models (ICLR 2024)

Previous Works on Time Series SSL

- **Mask-based Approaches.**
  - The mainstream approach to self-supervised time series pretext task is mask-modeling.
  - We randomly mask partial segments of time series and predict the masked-values.
  - PatchTST [5], PITS [6], VQ-MTM[7] all rely on mask-based pretext tasks.
- **Contrastive Approaches.**
  - Place similar instances close together in the representation space, while dissimilar far apart.
  - Previous approaches (e.g., TS-TCC [8], CA-TCC [9], TS-GAC [10]) augment the time instance using weak and strong augmentation, making weak and strong close to each other in representation space.

[5] Nie, Yuqi, et al. "A Time Series is Worth 64 Words: Long-term Forecasting with Transformers." The Eleventh International Conference on Learning Representations.
[6] Lee, Seunghan, Taeyoung Park, and Kibok Lee. "Learning to Embed Time Series Patches Independently." The Twelfth International Conference on Learning Representations.
[7] Gui, Haokun, Xiucheng Li, and Xinyang Chen. "Vector quantization pretraining for eeg time series with random projection and phase alignment." International Conference on Machine Learning. PMLR, 2024.
[8] Eldele, Emadeldeen, et al. "Time-Series Representation Learning via Temporal and Contextual Contrasting."
[9] Eldele, Emadeldeen, et al. "Self-supervised contrastive representation learning for semi-supervised time-series classification." IEEE Transactions on Pattern Analysis and Machine Intelligence (2023).
[10] Wang, Yucheng, et al. "Graph-Aware Contrasting for Multivariate Time-Series Classification." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 38. No. 14. 2024.

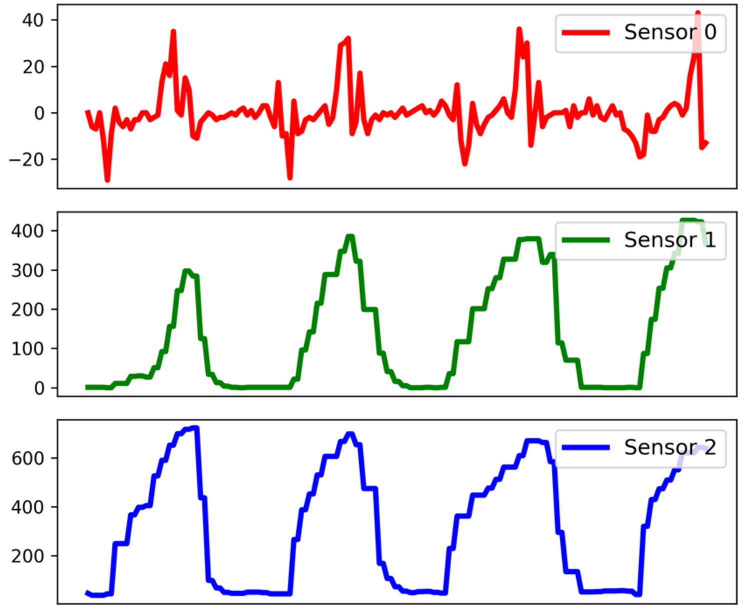
PPT: Patch Order Aware Pretext Task

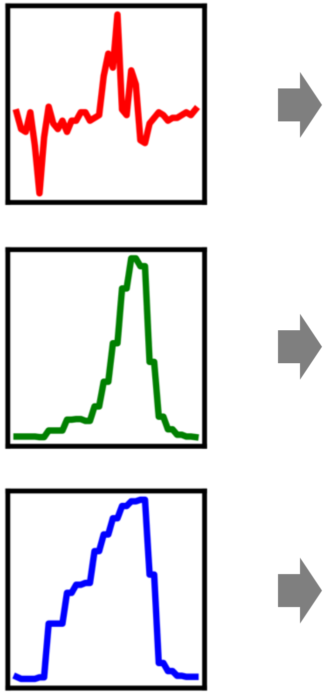
- **Overview of our methodology**

1. We propose an **order-aware pretext task** for patch-based time series learning.
   - PPT is applied and assessed on two state-of-the-art patch based models: PatchTST and PITS
2. PPT consists of **two order-aware learning methods**.
   - Consistency Learning and Contrastive Learning.
   - PPT is applicable to both self-supervised and supervised learning.
3. We also propose a metric **ACF-CoS**.
   - ACF-CoS can pre-examine whether a dataset could benefit from PPT.

Understanding Time Series Characteristics

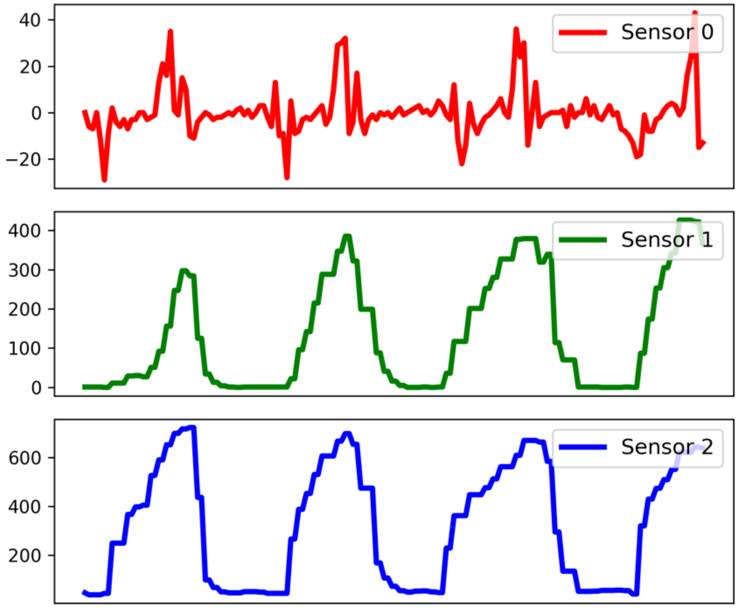
- **Can we leverage time series order characteristics?**
  - From the below, we can predict a certain order of signals in the Temporal axis.
  - The orders of patterns can be predicted.

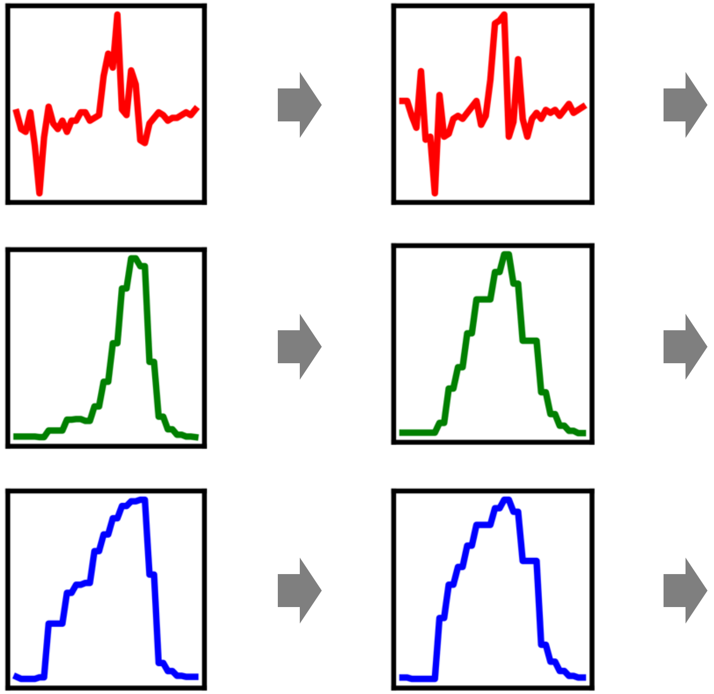




Understanding Time Series Characteristics

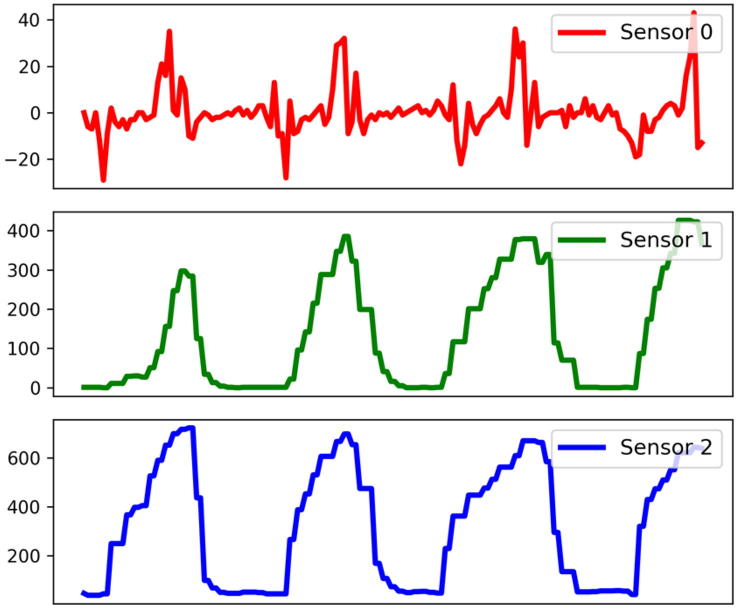
- **Can we leverage time series order characteristics?**
  - From the below, we can predict a certain order of signals in the **Temporal** axis.
  - The orders of patterns can be predicted.

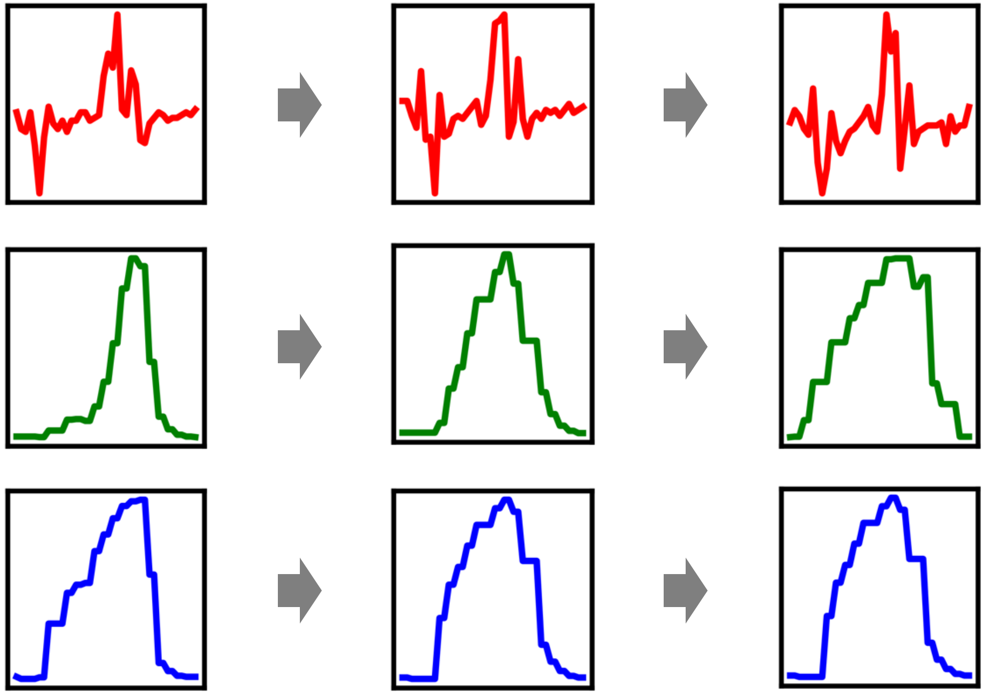




Understanding Time Series Characteristics

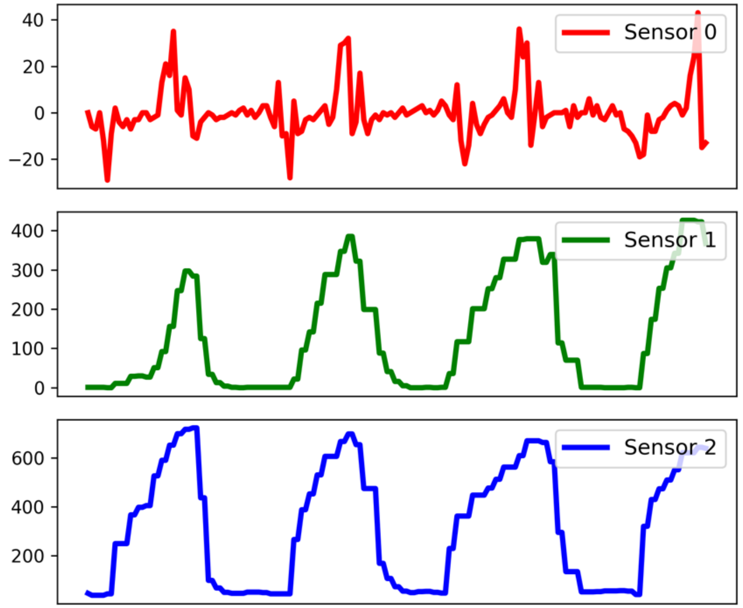
- **Can we leverage time series order characteristics?**
  - From the below, we can predict a certain order of signals in the **Temporal** axis.
  - The orders of patterns can be predicted.

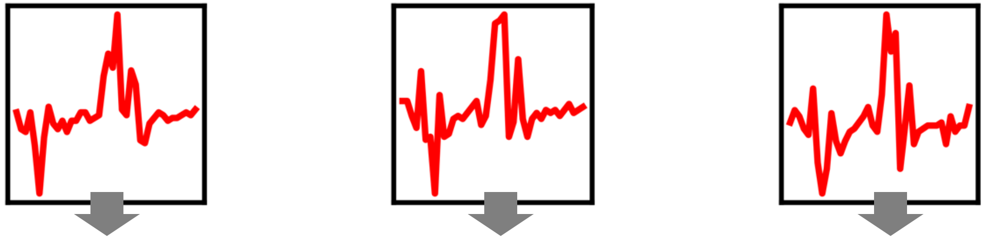




Understanding Time Series Characteristics

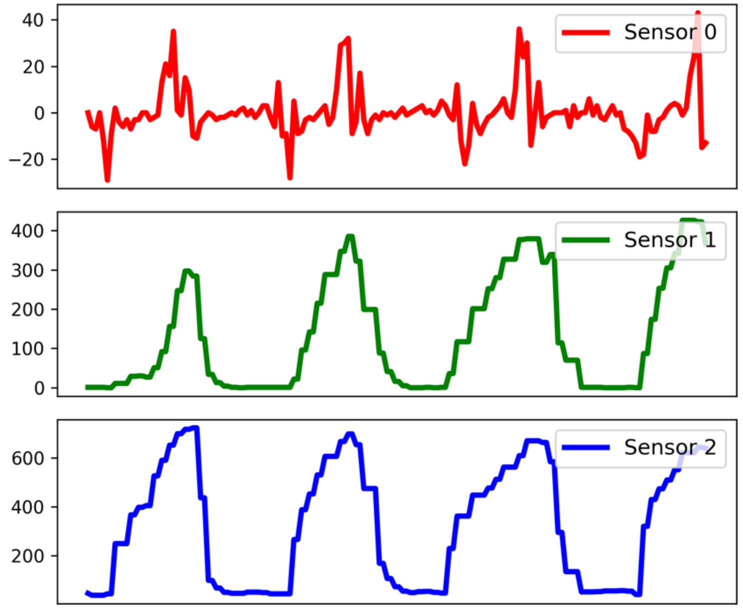
- **Can we leverage time series order characteristics?**
  - From the below, we can also predict a certain order of signals in the **Channel** axis.
  - The orders of patterns can be predicted.

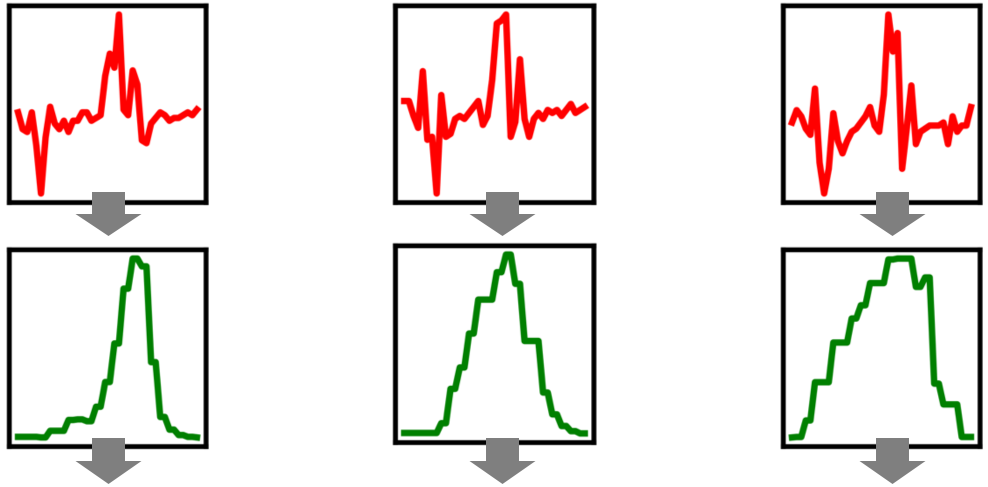




Understanding Time Series Characteristics

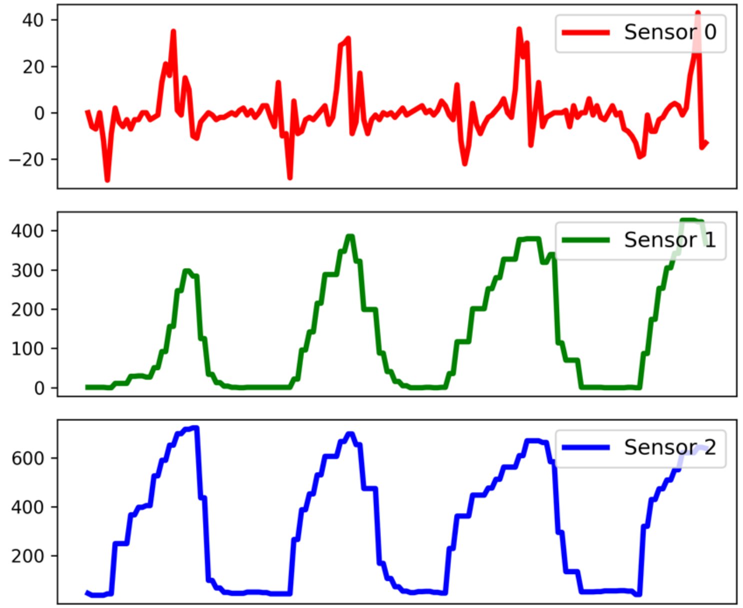
- **Can we leverage time series order characteristics?**
  - From the below, we can also predict a certain order of signals in the **Channel** axis.
  - The orders of patterns can be predicted.

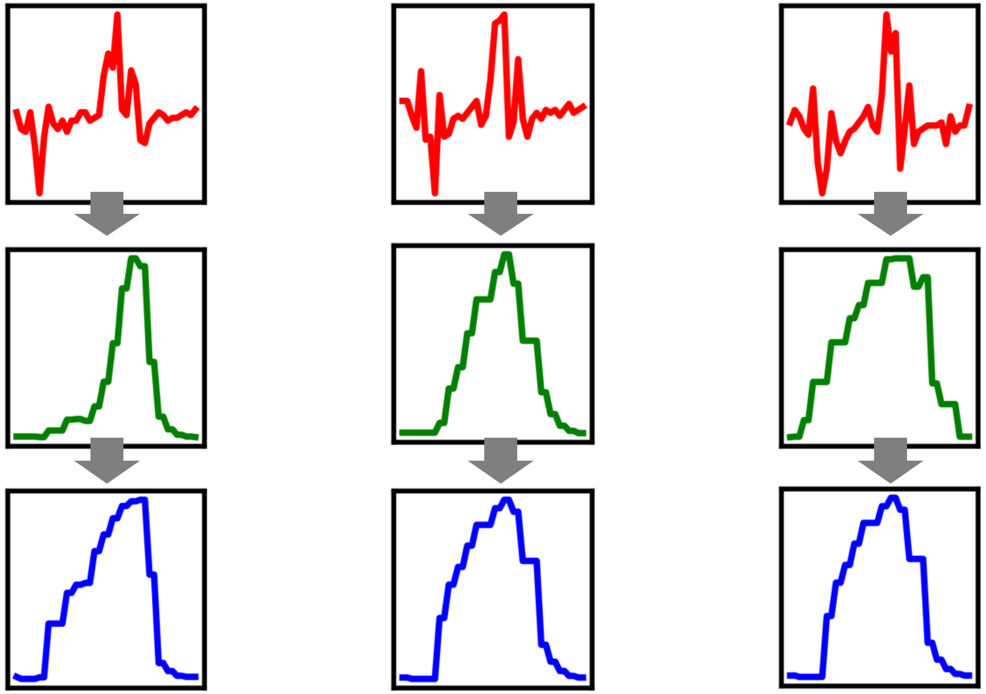




Understanding Time Series Characteristics

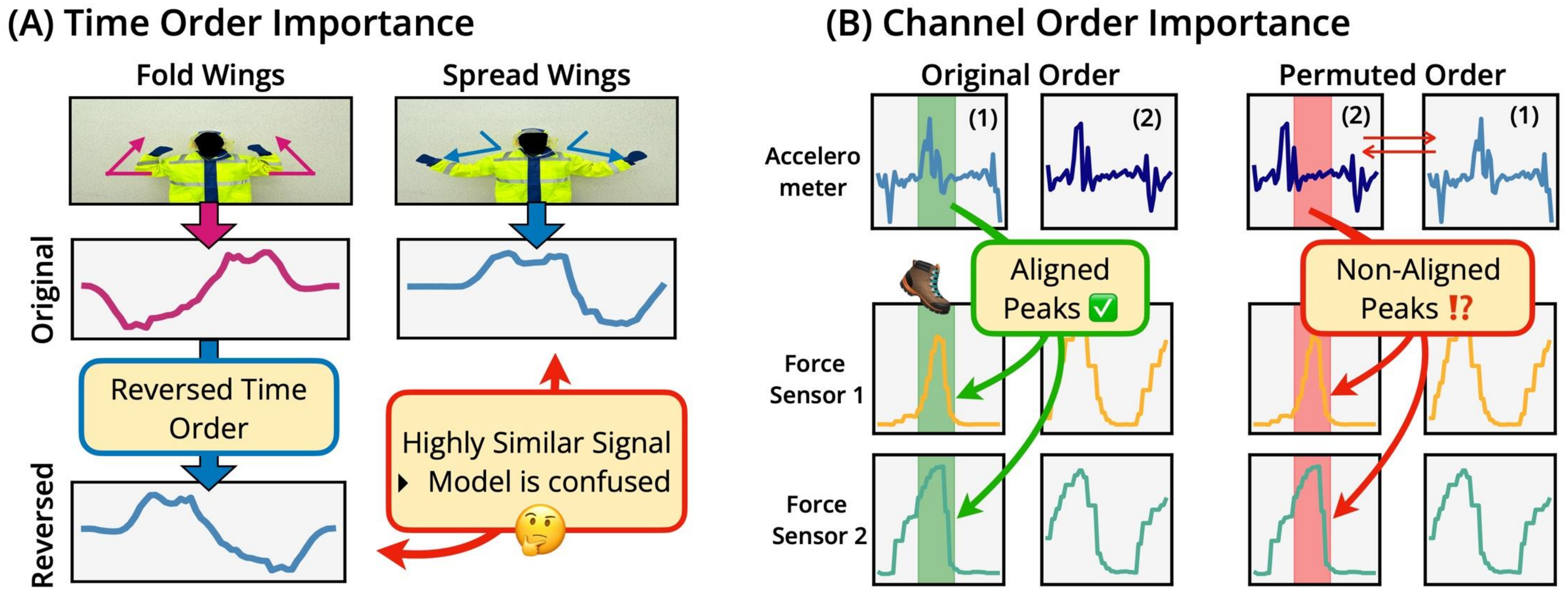
- **Can we leverage time series order characteristics?**
  - From the below, we can also predict a certain order of signals in the **Channel** axis.
  - The orders of patterns can be predicted.





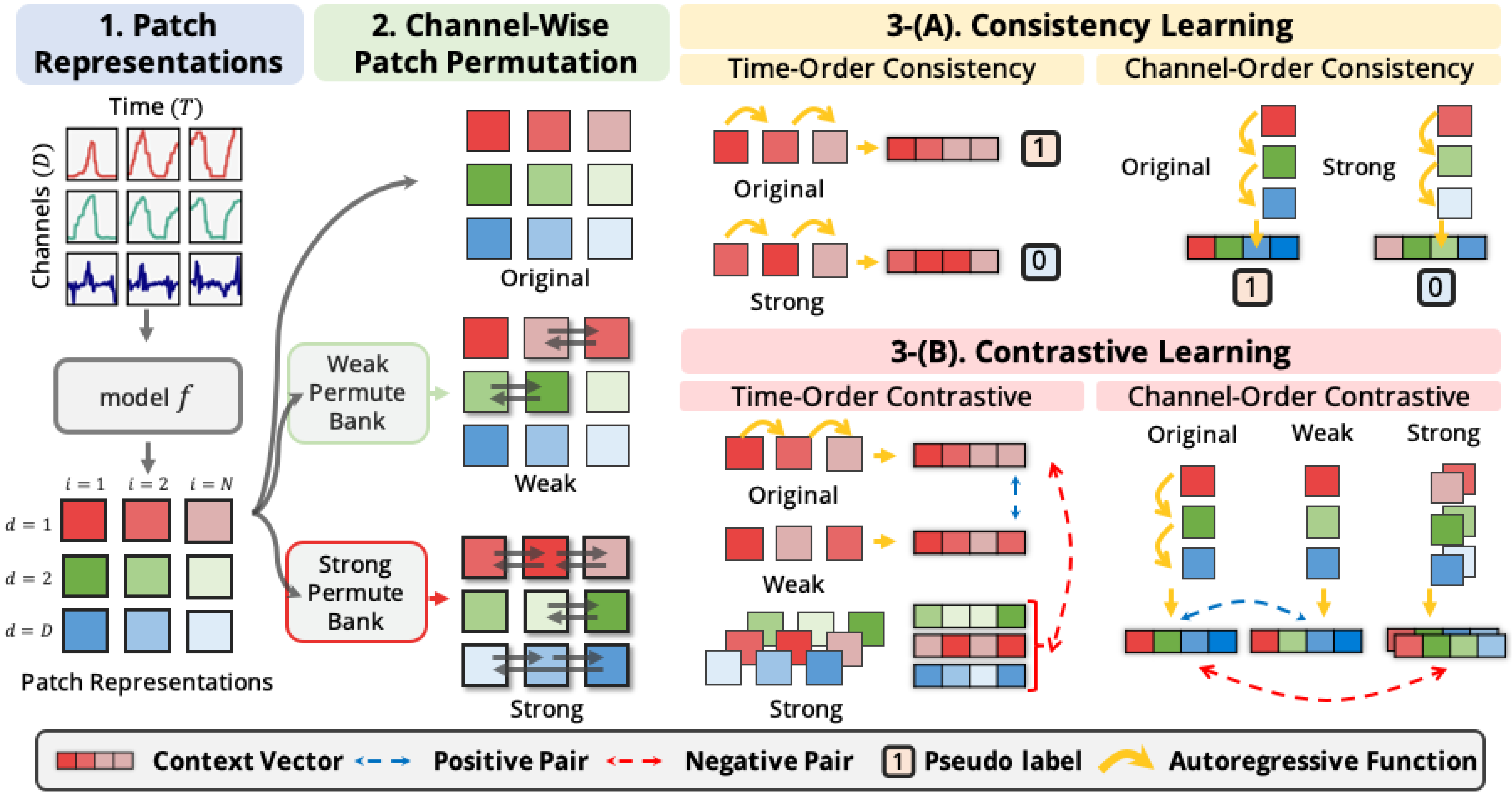
How can we supervise Order?

- **Order-Awareness of Patches**
  - Time series patches present **a natural order relationship**, both in time and channel order.



Methodology: PPT

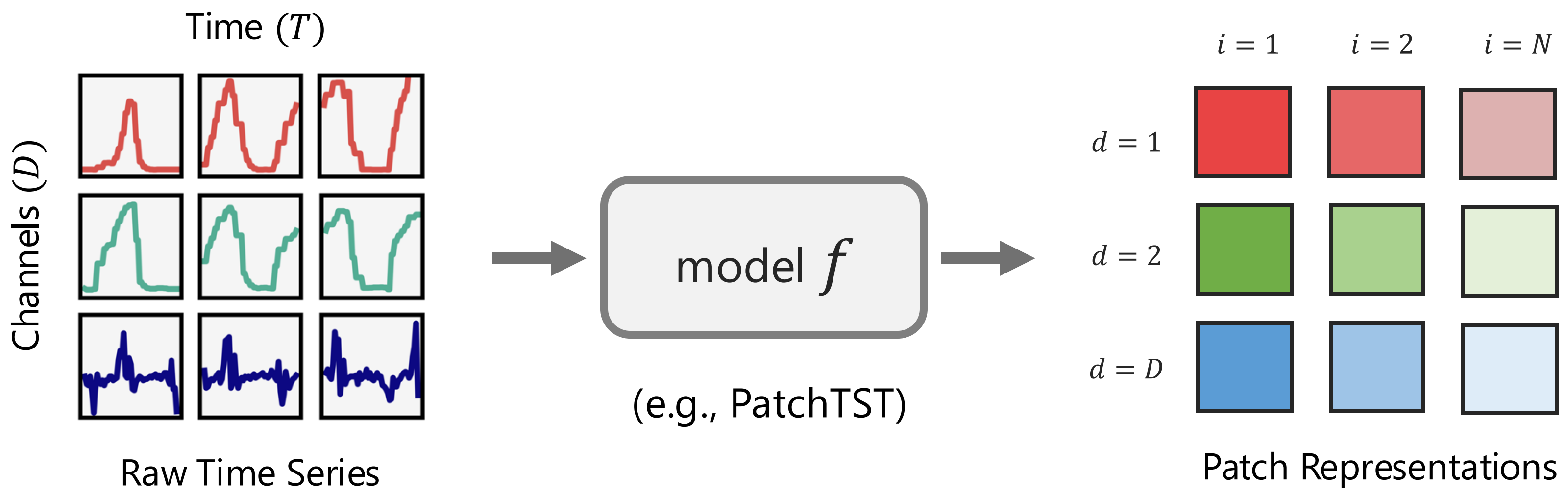
- **PPT Overview**



Methodology: PPT

- **Step 1. Encoding Time Series into Patches**

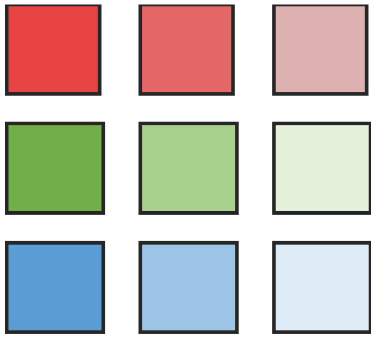
1. We first reshape time series into patches.
2. Then, encode each patches into representations using patch-based models.
   - E.g., PatchTST, PITS



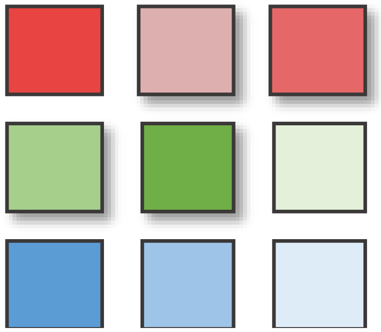
Methodology: PPT

- **Step 2. Channel-Wise Patch Permutation**

1. We construct three different sets of patches using permutation banks.
    - **Original**: The original sequence of patches.
    - **Weak**: The weakly permuted sequence of patches.
    - **Strong**: The strongly permuted sequence of patches.



Original



Weak

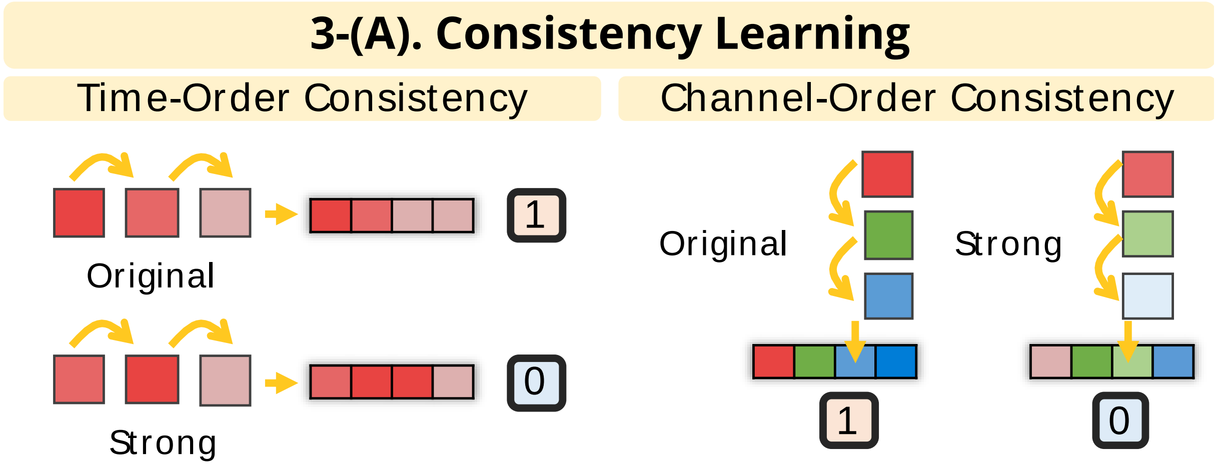
Strong

Methodology: PPT

- **Step 3-1. Consistency Learning**

1. Supervision Intuition: Is the given patch sequence order correct?
   - We perform **time-order** and **channel-order** consistency learning.
   - We utilize autoregressive models to supervise order consistency.

$$\mathcal{L}_{\text{Time}}^{\text{CS}} \text{ or } \mathcal{L}_{\text{Feature}}^{\text{CS}} = -\frac{1}{m}\sum_{i=1}^{m}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

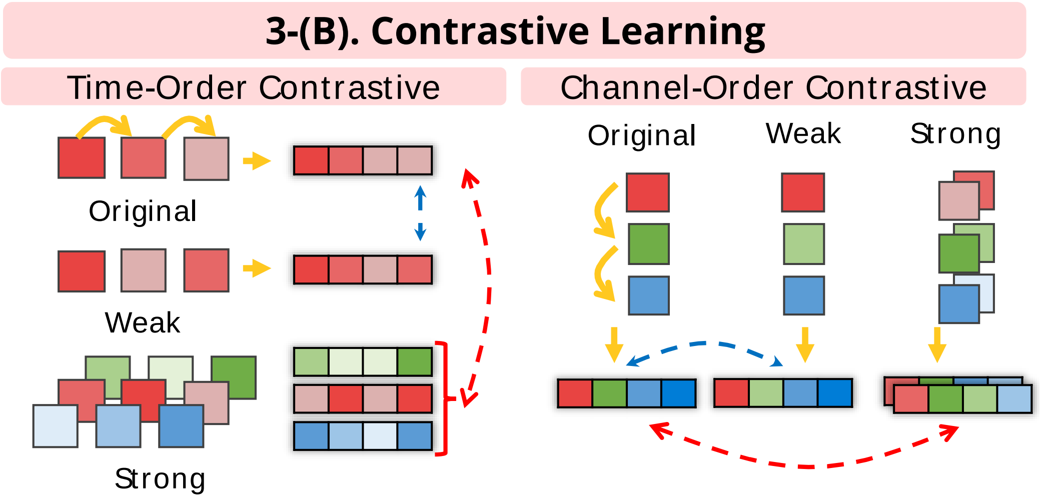


Methodology: PPT

- **Step 3-2. Contrastive Learning**

1. Supervision Intuition: Weak and Original are similar. But Strong is significantly different.
   - We set the Original and Weak as **Positive Pairs**, and the Strong as **Negative**.

$$\mathcal{L}_{\text{Time}}^{\text{CT}} \text{ or } \mathcal{L}_{\text{Feature}}^{\text{CT}} = -\frac{1}{D}\sum_{d=1}^{D}\log\left(\frac{\exp(sim\left(c_d^{original}, c_d^{weak}\right)/\tau)}{\exp(sim\left(c_d^{original}, c_d^{weak}\right)/\tau)) + \sum_{k=1}^{D}\exp(sim\left(c_d^{original}, c_d^{strong}\right)/\tau))}\right)$$



Methodology: PPT

- **Overall Loss Setup**

1. Self-Supervised Loss

   - We optimize the **consistency** and **contrastive** loss terms only.

$$\mathcal{L}_{\mathrm{Self-Supervised}} = \lambda_1 \mathcal{L}_{\mathrm{Sum}}^{\mathrm{CS}} + \lambda_1 \mathcal{L}_{\mathrm{Sum}}^{\mathrm{CT}}$$

2. Supervised Loss

   - We optimize the two terms along with the **task-specific** loss $\mathcal{L}_{\mathrm{T}}$

$$\mathcal{L}_{\mathrm{Supervised}} = \mathcal{L}_{\mathrm{T}} + \lambda_1 \mathcal{L}_{\mathrm{Sum}}^{\mathrm{CS}} + \lambda_1 \mathcal{L}_{\mathrm{Sum}}^{\mathrm{CT}}$$

ACF-CoS: Measuring Order

- **Autocorrelation function and Cosine Similarity**

1. Not all time series benefit from order-awareness
   - Can we **pre-assess the effect of PPT** prior to model training?
2. We propose ACF-CoS
   - We measure the cosine similarity between the autocorrelation of the Original and Strong.
   - If the autocorrelations **are similar → Structural order is absent.**

$$\mathrm{ACF-CoS} = 1 - \frac{\mathbf{a} \cdot \mathbf{a}'}{\|\mathbf{a}\| \|\mathbf{a}'\|}$$

$\mathbf{a}$: Autocorrelation of Original
$\mathbf{a}'$: Autocorrelation of Strong

Results

Self-Supervised Learning

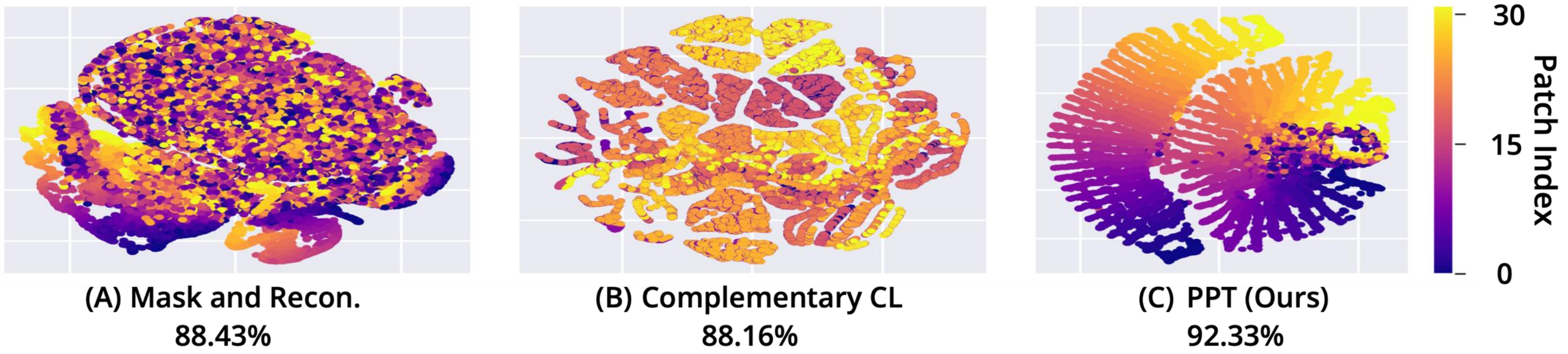
- **Linear-Probing**
  - The model is learned self-supervised, and linear probing is performed.
  - Linear probing fine-tunes only a single linear layer to obtain representation performance.
  - We obtain strong performance in all three tasks: EMO, Gilon, PTB.

Dataset	Models	Accuracy	F1 score	AUROC	AUPRC	Precision	Recall
EMO	Mixing-up	74.48±2.93	49.30±2.53	71.80±7.11	52.22±2.71	51.66±1.81	49.36±4.01
	SimCLR	74.42±4.38	44.64±6.34	72.71±7.29	48.34±6.76	47.83±7.07	47.19±8.02
	TS2Vec	78.08±2.93	49.07±3.00	78.00±3.58	50.95±3.83	48.97±3.52	49.87±2.88
	TF-C	77.63±6.18	53.30±6.90	82.22±5.31	54.84±7.18	56.14±8.79	**56.00±9.07**
	TS-TCC	75.61±3.49	48.47±3.59	73.82±7.40	53.98±3.77	54.75±0.67	49.02±4.05
	SimMTM	81.75±3.33	53.08±3.68	81.35±7.69	58.70±4.03	57.75±4.84	51.62±4.17
	TimeMAE*	73.97±2.28	42.44±2.07	70.11±4.43	43.25±2.14	42.81±2.37	42.43±2.09
	TS-GAC*	73.75±1.66	46.42±1.29	75.92±2.30	49.29±0.62	46.04±0.87	48.86±1.69
	PatchTST*	78.70±0.73	45.81±2.07	82.60±1.39	55.23±2.21	59.40±5.32	46.35±1.31
	PatchTST (+PPT)*	**81.92±0.58**	**54.19±2.33**	**84.74±1.55**	**62.51±3.09**	**62.96±2.49**	53.41±2.42
	PITS*	69.63±2.04	43.73±1.06	68.84±2.39	43.90±1.05	44.07±0.82	45.68±1.31
	PITS (+PPT)*	75.55±2.84	45.75±2.43	68.63±3.30	45.59±2.28	45.05±2.65	47.53±2.06

Self-Supervised Learning

- **Linear-Probing**
  - t-SNE visualization of representations **based on patch indexes**.
  - We observe better patch index alignment with **PPT**.

t-SNE Visualization of Self-Supervised Learning



Self-Supervised Learning

- **Semi-Supervised learning**
  - Perform self-supervised training, then perform supervised fine-tuning.
  - With limited labeled data: **10%** and **1%.**

Fraction	Models	Accuracy	F1 score	AUROC	AUPRC	Precision	Recall
10%	Mixing-up	92.85±0.69	90.44±0.91	98.74±0.26	94.42±1.14	90.59±0.65	90.69±0.90
	SimCLR	84.55±0.78	83.60±1.21	98.47±0.20	91.74±0.83	86.78±0.62	82.56±1.16
	TS2Vec	88.12±1.58	85.51±1.13	96.46±0.78	89.37±1.87	85.86±0.69	85.97±1.71
	TF-C	83.35±0.48	82.73±0.48	97.96±0.09	87.87±0.36	83.95±0.33	82.10±0.59
	TS-TCC	**93.69±1.05**	92.11±0.84	99.41±0.19	97.36±0.76	93.69±0.38	91.83±0.83
	TimeMAE	90.06±2.95	91.10±2.54	98.77±0.43	94.65±2.04	91.50±2.46	90.83±2.63
	SimMTM	91.94±0.58	91.35±0.53	98.95±0.35	95.65±0.62	91.41±0.44	91.40±0.65
	PatchTST	91.61±0.82	92.33±0.89	99.35±0.11	97.10±0.47	92.88±0.80	92.47±0.75
	PatchTST (+PPT)	93.26±1.57	**93.97±1.40**	**99.50±0.09**	**97.79±0.47**	**94.74±1.23**	**94.27±1.34**
	PITS	85.11±3.78	85.67±2.21	98.18±0.43	89.51±2.63	84.61±2.35	84.60±2.65
	PITS (+PPT)	92.47±1.06	93.32±0.60	99.48±0.12	97.28±0.78	93.17±0.69	93.07±0.46

Self-Supervised Learning

- **Semi-Supervised learning**
  - Perform self-supervised training, then perform supervised fine-tuning.
  - With limited labeled data: **10%** and **1%.**

1%	**Mixup**	84.82±2.17	82.08±2.85	97.27±0.53	87.48±1.81	83.76±2.52	81.53±3.34
	SimCLR	62.61±1.89	47.28±4.56	90.88±2.03	66.05±4.28	63.15±9.38	51.63±2.92
	TS2Vec	77.41±1.33	75.17±2.85	96.17±0.45	82.84±1.67	79.04±1.10	74.64±3.01
	TF-C	65.34±2.50	52.88±4.98	91.19±1.59	71.15±3.38	71.92±4.11	52.95±3.65
	TS-TCC	**85.77±1.08**	83.02±1.16	97.82±0.25	89.85±1.19	86.31±2.00	83.04±1.46
	TimeMAE	76.09±2.01	74.63±3.30	96.24±0.53	80.35±3.35	77.58±3.69	73.57±3.61
	SimMTM	78.44±2.20	79.48±1.95	94.93±0.87	82.75±1.34	80.66±2.40	79.31±1.85
	PatchTST	80.55±2.29	83.26±2.11	96.77±1.04	86.44±2.83	81.50±3.58	81.52±3.58
	PatchTST (+PPT)	84.80±1.68	**86.92±1.48**	**98.08±0.38**	**90.64±1.90**	**86.88±1.65**	**86.75±1.57**
	PITS	72.41±2.05	72.81±4.76	95.40±0.60	75.92±3.23	69.83±3.77	70.45±4.71
	PITS (+PPT)	81.04±1.86	83.71±0.95	97.68±0.30	87.26±1.62	81.25±1.45	82.05±1.30

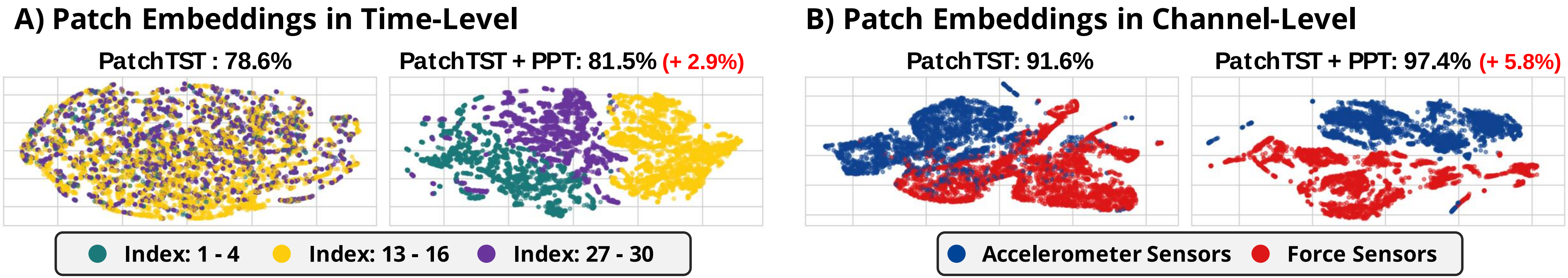
Supervised Training

- **Supervised training**
  - We perform supervised training.
  - We also perform ablation on each of the loss terms.
  - Each of the term contributes to model performance, and has synergistic effects.

Dataset Name			GL HAR			SleepEEG			PTB ECG		
Models	$\mathcal{L}^{CS}$	$\mathcal{L}^{CT}$	**Original ↑**	**Permuted ↓**	**Diff ↑**	**Original ↑**	**Permuted ↓**	**Diff ↑**	**Original ↑**	**Permuted ↓**	**Diff ↑**
PatchTST (2022)	✗	✗	91.6±3.35	88.8±6.01	2.76	61.6±1.57	58.5±1.51	3.03	78.6±2.16	76.3±2.79	2.28
	✗	✓	96.6±1.00	89.2±2.76	7.33	61.8±1.18	58.0±1.03	3.79	78.8±2.79	73.7±1.38	5.17
	✓	✗	97.2±0.40	89.0±4.13	8.17	61.5±0.61	**57.9±0.94**	3.56	**81.8±2.48**	73.5±2.29	8.33
	✓	✓	**97.4±0.46**	**88.7±2.59**	**8.65**	**63.5±0.79**	58.7±0.59	**4.69**	81.4±2.51	**72.7±0.91**	**8.71**
PITS (2023)	✗	✗	91.6±3.32	85.3±3.34	6.30	55.4±1.87	**55.1±1.85**	0.32	82.0±6.67	71.6±0.66	10.4
	✗	✓	92.8±4.63	81.0±8.30	11.8	56.3±2.34	55.6±2.55	0.65	85.1±2.98	68.9±6.02	16.2
	✓	✗	94.0±0.68	87.6±5.15	6.40	57.4±1.22	56.8±1.10	0.66	84.0±5.61	71.3±0.69	12.7
	✓	✓	**96.3±1.19**	**73.0±5.29**	**23.3**	**59.3±0.87**	57.3±1.03	**1.95**	**89.5±1.96**	**65.0±6.46**	**24.6**

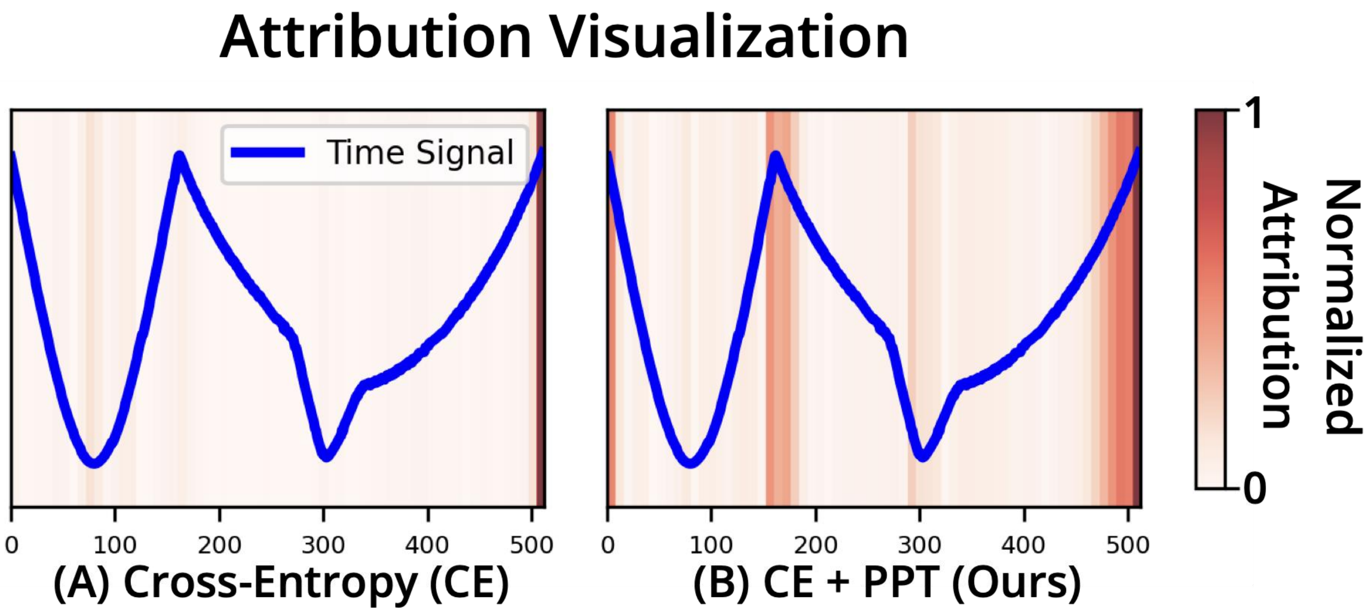
Supervised Training

- **Supervised training**
  - t-SNE visualization of patches in **both time and channel level.**



Supervised Training

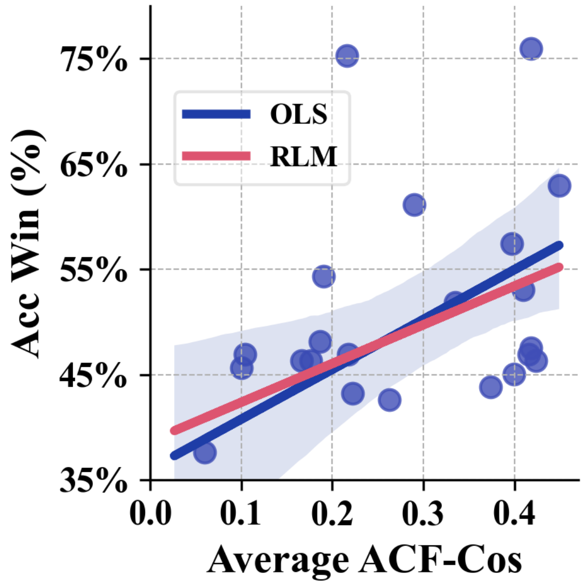
- **Supervised training**
  - We compare and visualize the importance of time series patches in model training.
  - We observe that incorporating **PPT better captures the inflection points** in time series.



ACF-CoS

- **Correlation between ACF-CoS and PPT**
  - We measured the performance gain obtained from **PPT** with 27 tasks from the UEA repository.
  - We observe positive correlation between **ACF-CoS** and **PPT**.

UEA Datasets	ACF-CoS ↑	Acc. Win% (Wins) ↑	Max CE / Max PPT (Acc) ↑
Step Function (Order ↑)	0.902	-	-
Cricket	0.418	75.9% (123/162)	69.4 / **72.7**
EigenWorms	0.289	61.1% (99/162)	47.1 / **54.5**
NATOPS	0.216	75.3% (122/162)	70.0 / **71.7**
LargeKitchen.	0.190	54.3% (88/162)	64.1 / **65.0**
GestureMidAirD1	0.186	48.1% (78/162)	26.2 / **31.3**
GestureMidAirD3	0.060	37.7% (61/162)	**18.2** / 16.9
White Noise (Order ↓)	0.001	-	-



PPT Conclusion

- PPT is an **order-aware self-supervised method** for time series
  - Supervises the order of patches in both time and channel dimension.
- PPT is a **plug-in method** for any patch-based models
  - PPT works with any patch-based models that can represent each patches independently.
- PPT shows **strong performance**
  - We show that incorporating order-awareness can enhance model performance.
  - We show ways to identify which time series tasks can benefit from PPT.

Thank you! Any questions?

Jaeho Kim, Ph.D. Student
kjh3690@unist.ac.kr

Artificial Intelligence Graduate School (AIGS)
Ulsan National Institute of Science and Technology (UNIST)