速習！時系列予測Transformer属

NTTコミュニケーションズ　イノベーションセンター 木村大地

2024-06-26 Group Reading @DL連絡会

Introduction

- NLP・CVにおけるTransformerの成功により時系列解析（予測、分類、異常検知…）においても注目されている
  - この発表では、特に多変量時系列予測を扱う
- さまざまな時系列〇〇formerというタイトルの論文が発表され、それぞれがSOTAを主張
  - Informer, Autoformer, Crossformer, ….
- そこで、これらの手法をまとめてみる
  - 論文の結果は紹介しない（いずれもSOTAを主張しており面白味がないため）
  - どんなアイデアなのかを紹介
  - 2024-06-28 現在の被引用数（Google scholar調べ）も掲載

準備 -多変量時系列予測-

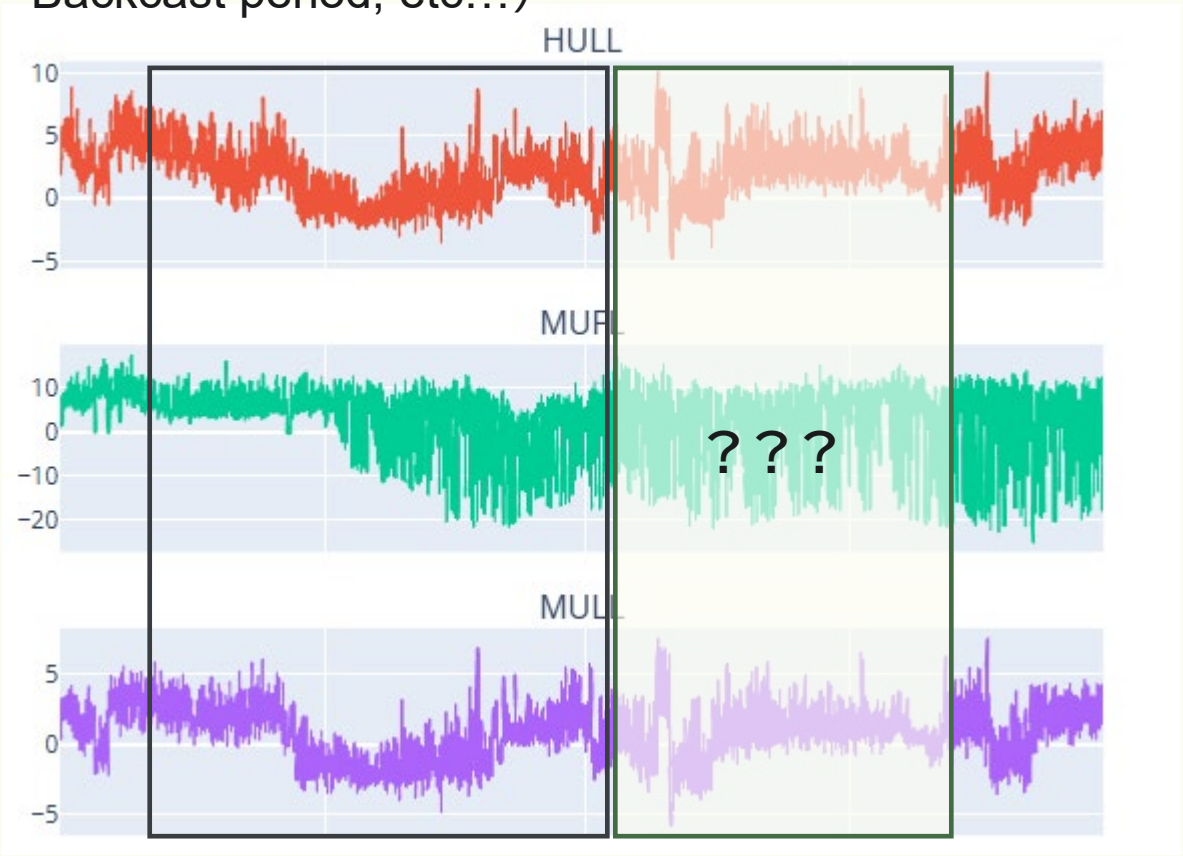
多変量時系列
- 複数の次元(変量)をもつ時系列

時系列予測
- 時系列の過去の値を入力として、未来の値を予測
- 予測対象が単変量の場合もありうる（Multi-Input/Single-Output, MISO）

典型的な手法
- 統計: AR, ARIMA, VAR, VARMA, etc…
- MLP系: DLinear,
- RNN系: LSTM, DeepAR
- CNN系: TCN

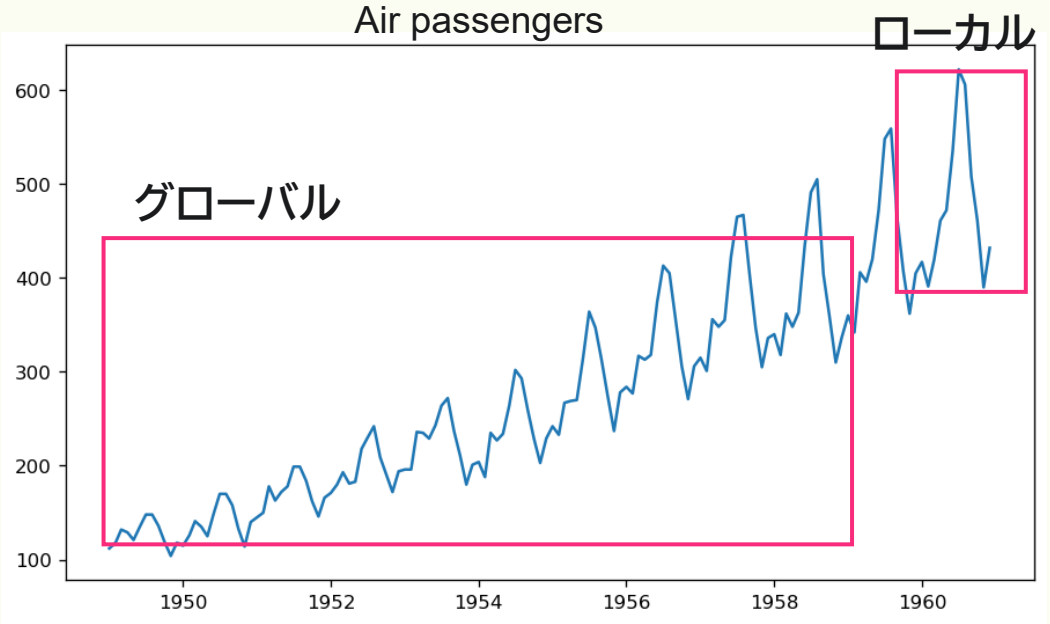
入力
（窓枠, lookback window, Backcast period, etc…）

予測対象
（Forecast Period, etc…）

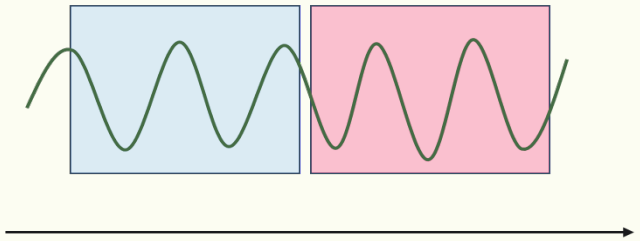


時系列予測の課題

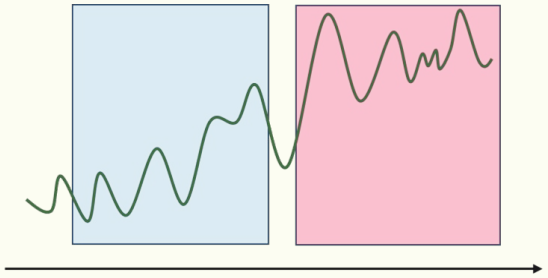
- 時間的依存関係
  - 過去現れたパターンを正しく捉える
- 変数間依存関係
  - 変数間の関係を正しく捉える
  - 例: 天気, 気温, 湿度の関係
- グローバル / ローカル
  - 全体的な傾向、局所的な変動
- 非定常性
  - 学習時とテスト時で分布が異なる
  - 例: トレンド成分



定常: 統計量(平均分散自己相関)
がどこでも同じ



非定常



Vanilla Transformer

- いわずとしれたTransformer（詳細な解説は略）
- Self-attention を積層したEncoder-Decoder構造
- 計算コスト: 系列長 L に対して, $O(L^2)$

(Self-) Attention※

Attention weight

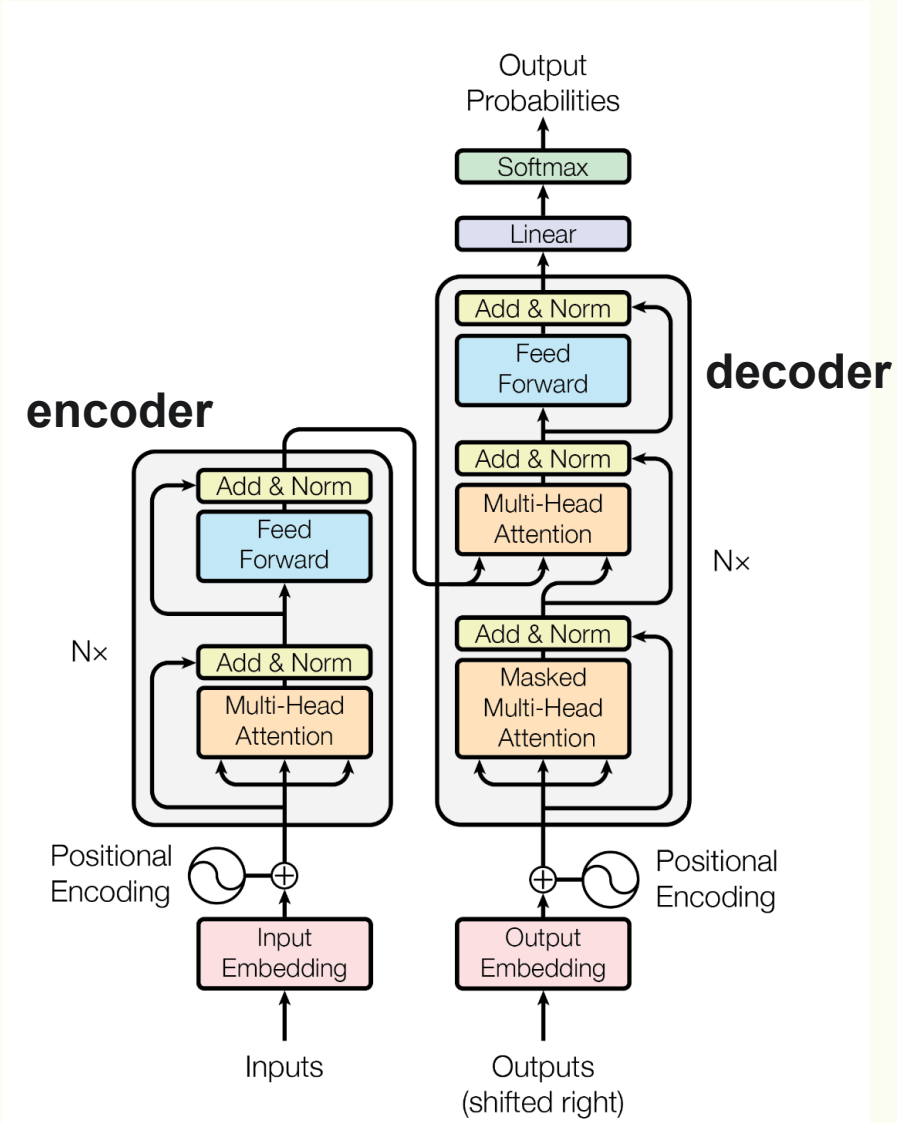
$$\mathrm{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{D}}\right)\mathbf{V}$$

query $\mathbf{Q} = \boldsymbol{x}\boldsymbol{w}^{\mathrm{Q}}$　key $\mathbf{K} = \boldsymbol{x}\boldsymbol{w}^{\mathrm{K}}$　value $\mathbf{V} = \boldsymbol{x}\boldsymbol{w}^{\mathrm{V}}$　$\mathbf{Q},\mathbf{K},\mathbf{V} \in \mathbb{R}^{L\times D}$

入力　$\boldsymbol{x} \in \mathbb{R}^{L\times N}$

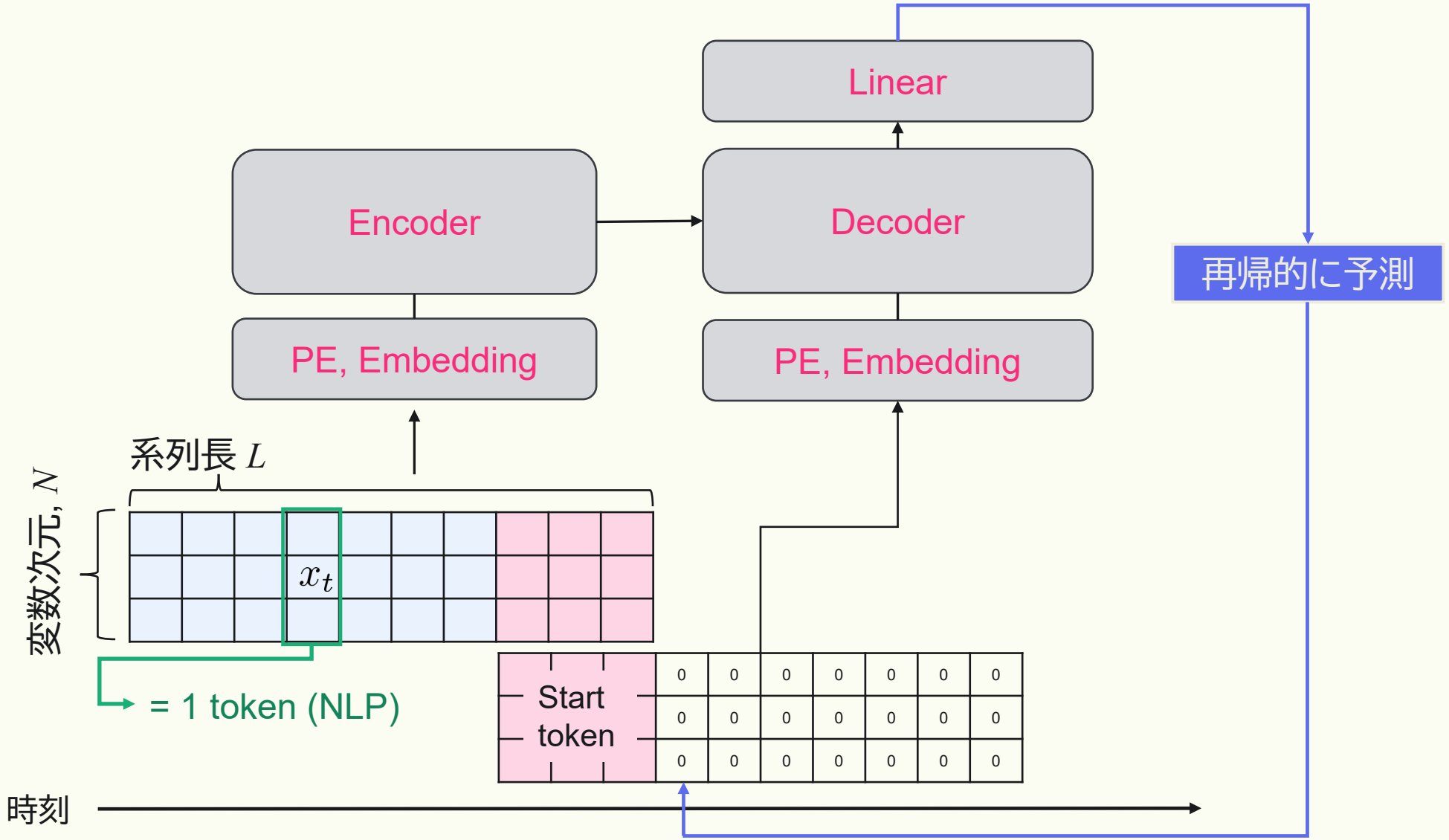
重み(学習)　$\boldsymbol{w}^{\mathrm{Q}}, \boldsymbol{w}^{\mathrm{K}}, \boldsymbol{w}^{\mathrm{V}} \in \mathbb{R}^{N\times D}$

※ 横着してSelf-Attention の場合のみ



Vaswani, Ashish et al. “Attention is All you Need.” *NeurIPS* (2017).

Vanilla Transformer



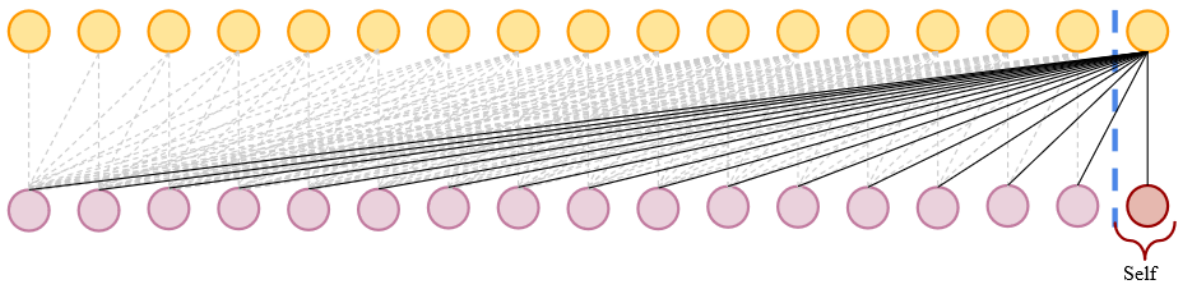
LogTrans（NeurIPS 2019, 1428 citations）

LI, SHIYANG et al. “Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting.” *NeurIPS* (2019)

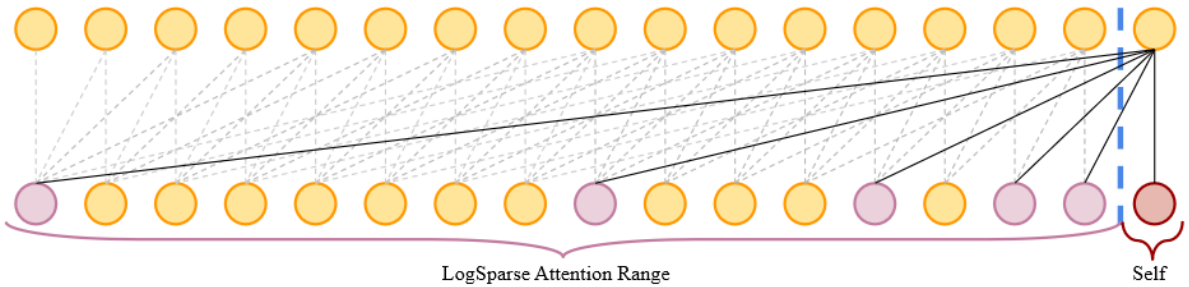
【主な工夫】計算量削減, 局所情報の取り込み

LosSparse Self Attention

- Attentionにおいて、時間的に遠い点を指数的に疎にサンプリング
- 遠い過去より直近を重視
- 時間・空間計算量: $O(L(\log L)^2)$



(a). Full Self Attention

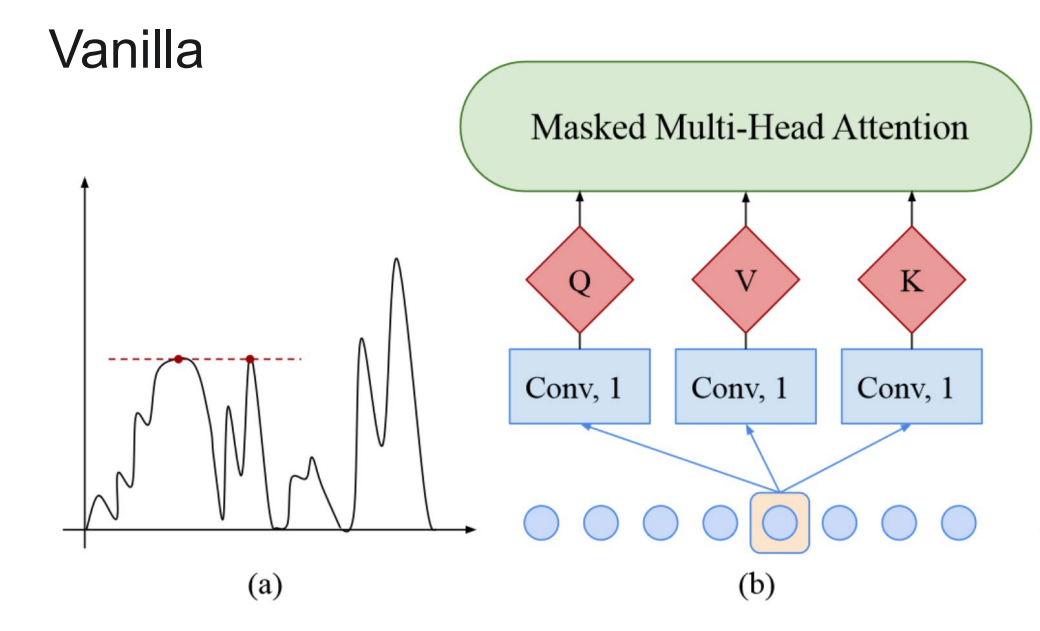


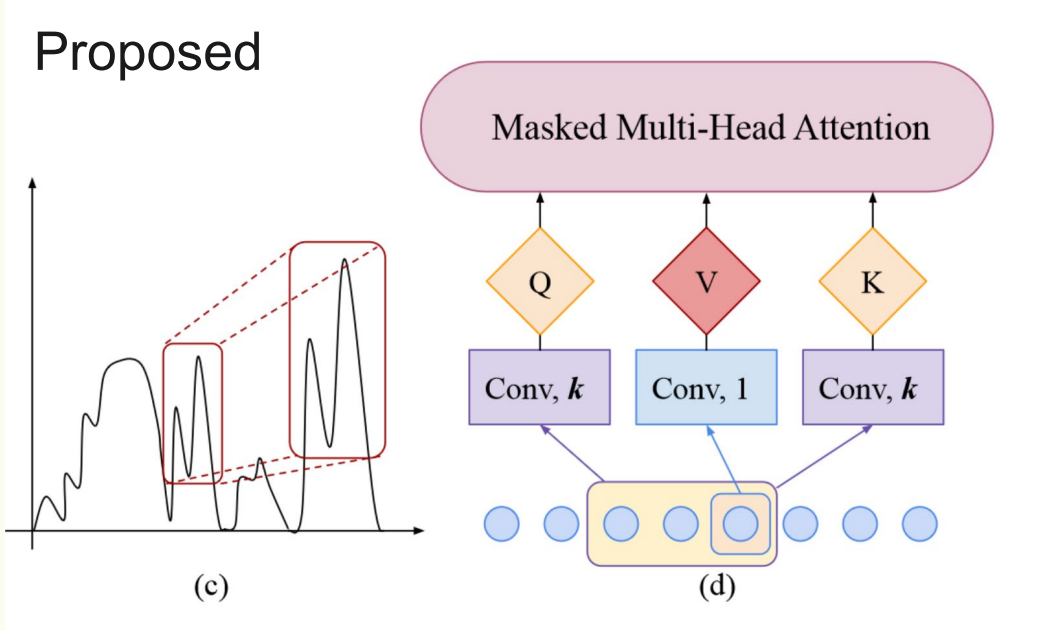
(b). LogSparse Self Attention

LogTrans（NeurIPS 2019）

局所情報の取り込み

- ある時刻点だけでなく、局所的なコンテキストを考慮するために1D Convで周辺の情報を取り込む
- 単純に1点だけみるより、形状を見たほうがよさそう



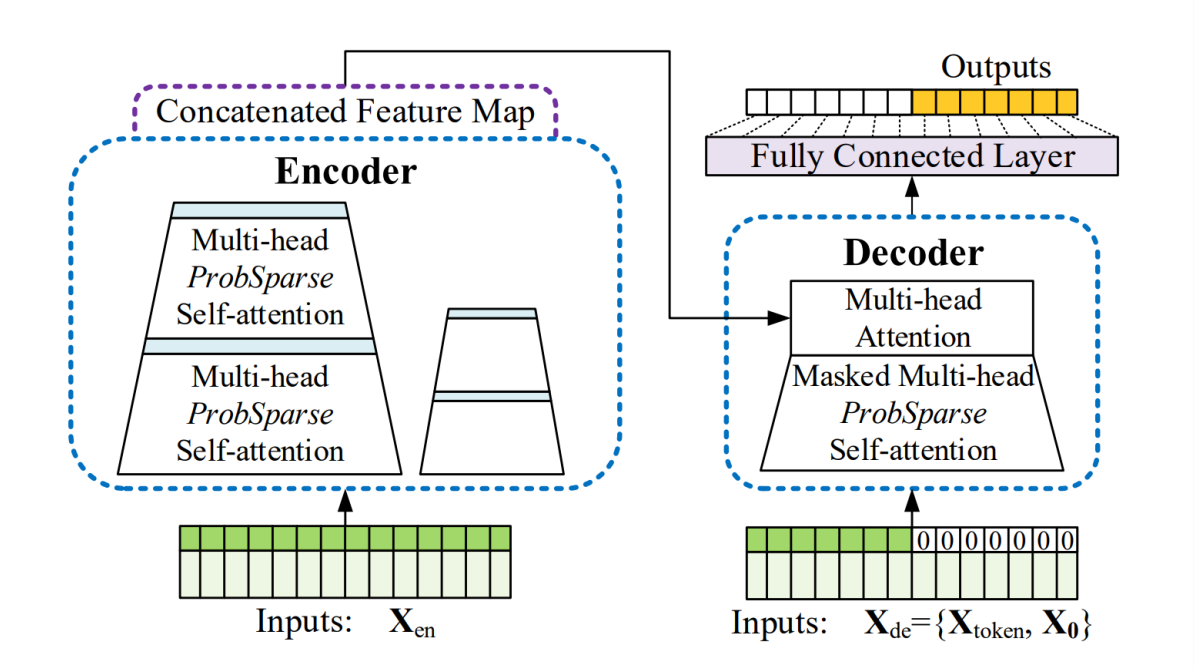


Informer（AAAI 2021, 2939 citations）

Zhou, Haoyi et al. “Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting.” *AAAI* (2021).

【主な工夫】計算量削減, 再帰的予測の廃止, 時刻情報の埋め込み

- **ProbSparse self-attention:**
  - 特に重要な特徴を含んでいそうなAttentionの部分だけを選択し計算（$O(L \log L)$）
- **Self-attention distilling(蒸留)：**
  - self-attention層の出力に対して、Conv1D+Maxpooling1Dを作用させる。冗長な表現を削減し、メモリ使用量削減
- **生成的decoder:**
  - 一度にまとめて長期系列を出力。再帰構造がなくなるため誤差の蓄積を防ぎ、計算コストも削減
- **Time stamp embedding**
  - PEだけでなく、時系列に特有の時刻情報を埋め込む

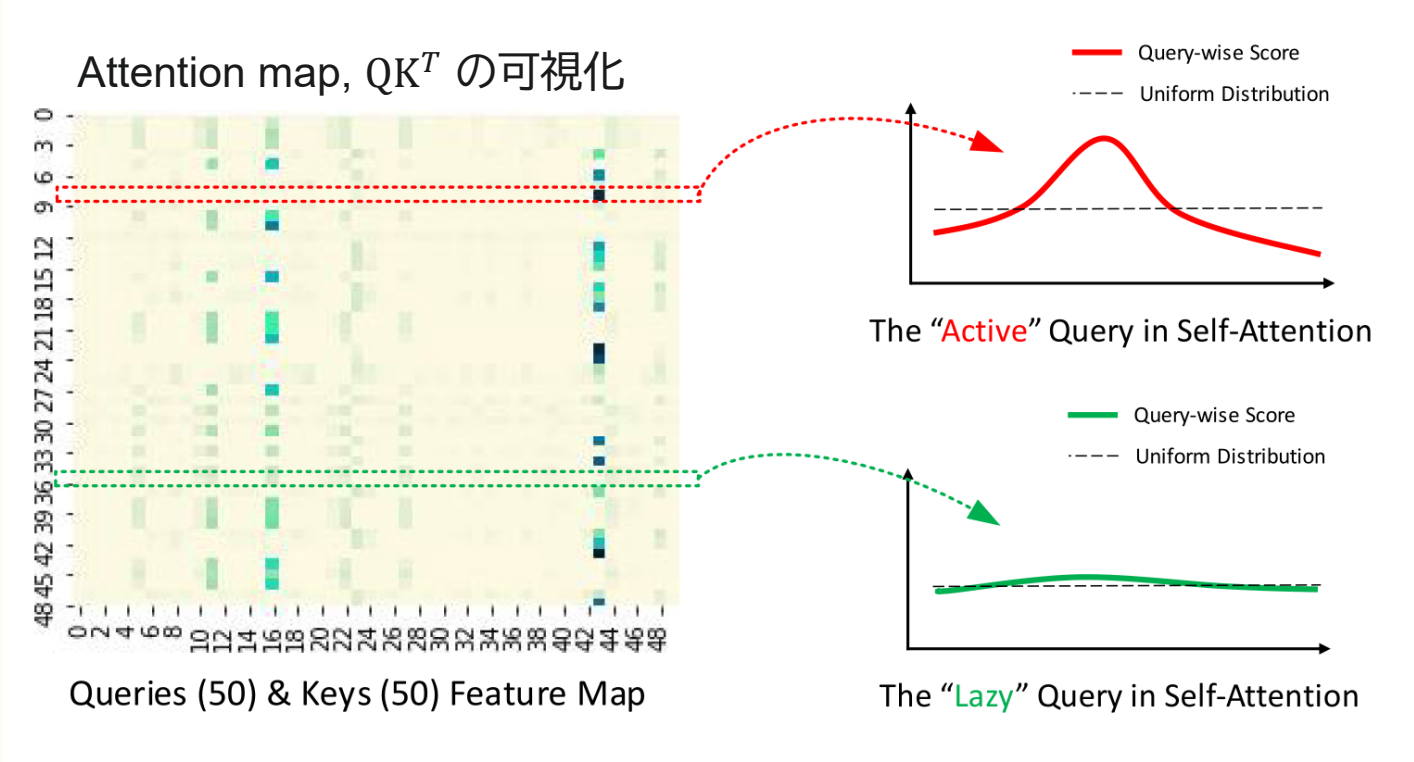


詳細は Informer (AAAI2021 Best Paper) — ごちきか (ntt.com)

Informer（AAAI 2021）

ProbSparse self-attention

- Attention mapで値が大きいTop-u個のみを用いる＝重要なAttentionのみを使う
  - 愚直にTop-uを決めるためにすべて計算すると意味がないので、一様分布からの逸脱度（KL-Div）を用いた指標で決定



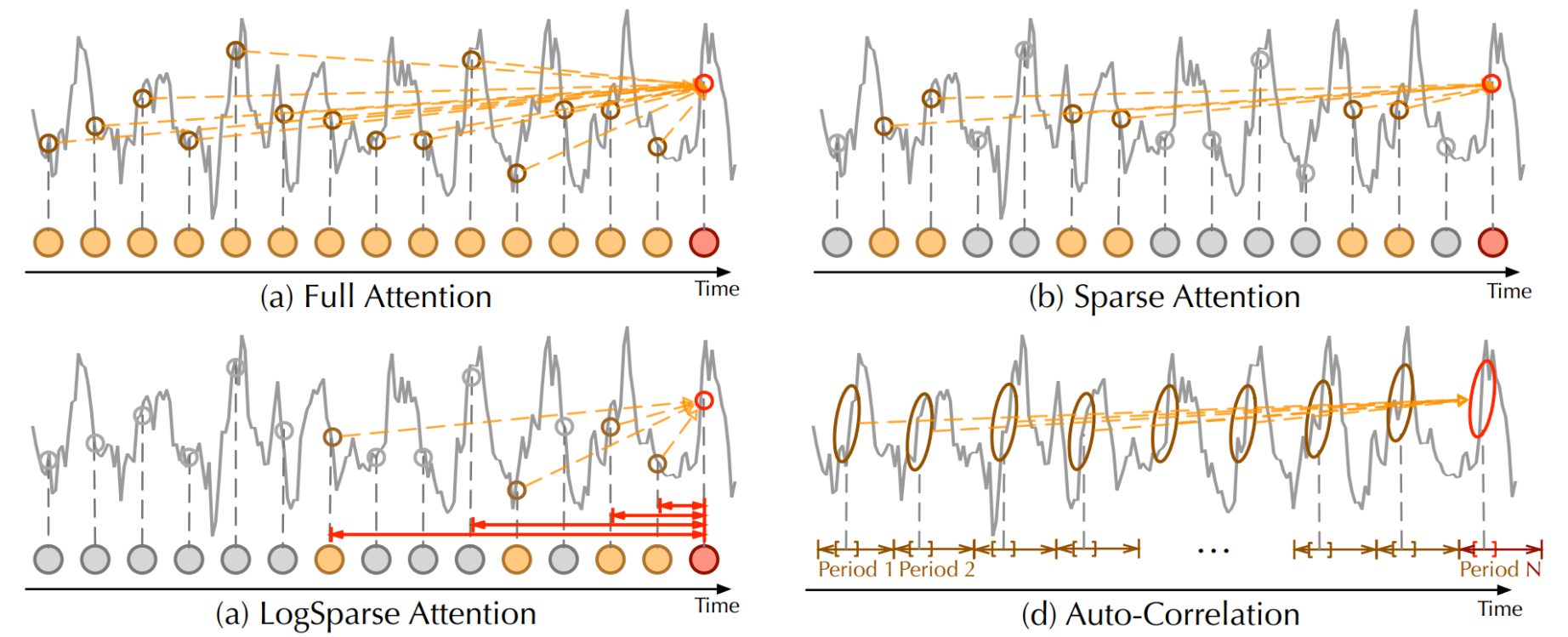
Autoformer（NeurIPS 2021, 1317 citations）

Wu, Haixu et al. “Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting.” *NeurIPS*(2021).

【工夫点】計算コスト改善, 季節-トレンド分解の活用

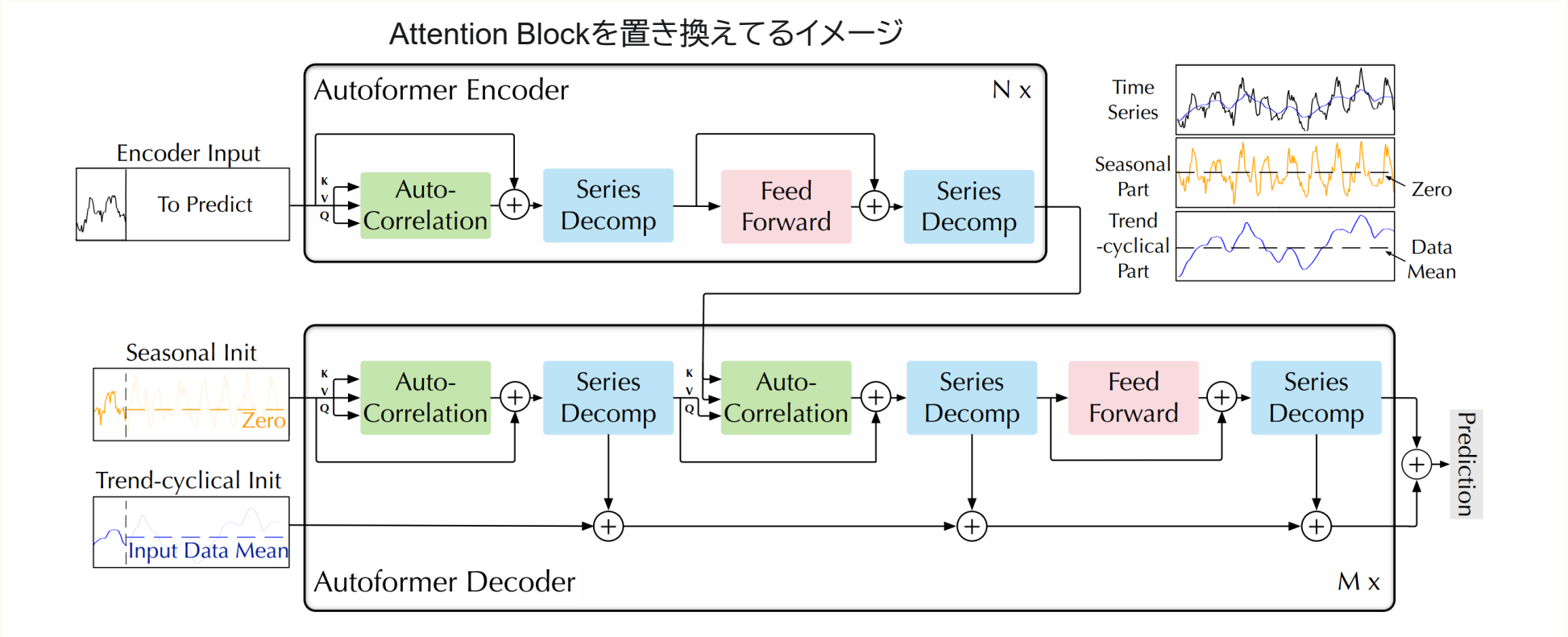
これまでの手法: ポイントワイズなダウンサンプリング＝情報損失が大きい

➡ 自己相関をもちいて似たパターン（意味のある情報）を集約



Autoformer（NeurIPS 2021）

- **Series Decomposition**
  - 季節-トレンド分解を行い、それぞれの成分について予測
- **Auto-Correlation Block**
  - 季節成分について、繰り返し現れるような部分系列を集約



Autoformer（NeurIPS 2021）

Series Decomp

Average Poolingで移動平均をかける

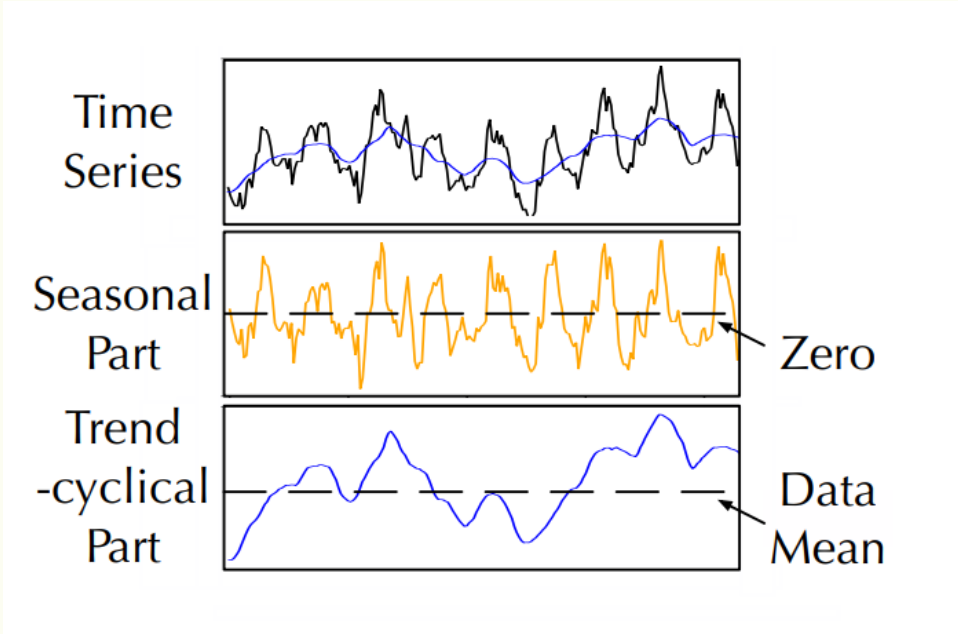
$$\mathcal{X}_{\text{trend}} = \text{AvgPool}(\text{Padding}(\mathcal{X}))$$

$$\mathcal{X}_{\text{seasonal}} = \mathcal{X} - \mathcal{X}_{\text{trend}}$$

それぞれ分けてモデリングを行うことで、
パターンをつかみやすくなり、予測性能向上が期待

季節-トレンド分解

季節性: 周期的な変動
トレンド性: 長期的な傾向



Autoformer（NeurIPS 2021）

Auto-Correlation Block:　系列レベルでの依存関係発見と情報集約

自己相関: 系列自身の過去のデータとどのぐらい類似してるか

$$\mathcal{R}_{\mathcal{X}\mathcal{X}}(\tau) = \lim_{L\to\infty} \frac{1}{L}\sum_{t=1}^{L} \mathcal{X}_t \mathcal{X}_{t-\tau}.$$

元の系列　ずらした系列　ラグ $\tau = \{1, \cdots, L\}$

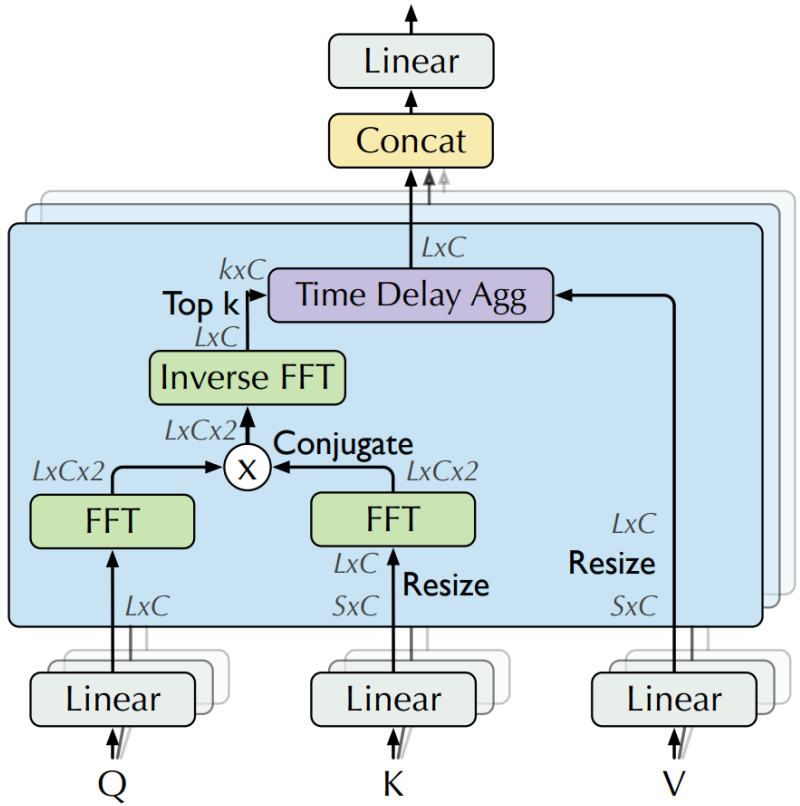
➡ **これをAttention weight みたいに使いたい**

自己相関の計算は高速フーリエ変換で効率的に計算可能
（Wiener-Khinchinの定理）

$$\mathcal{S}_{\mathcal{X}\mathcal{X}}(f) = \mathcal{F}(\mathcal{X}_t)\mathcal{F}^*(\mathcal{X}_t) = \int_{-\infty}^{\infty} \mathcal{X}_t e^{-i2\pi tf}\mathrm{d}t \overline{\int_{-\infty}^{\infty} \mathcal{X}_t e^{-i2\pi tf}\mathrm{d}t}$$

$$\mathcal{R}_{\mathcal{X}\mathcal{X}}(\tau) = \mathcal{F}^{-1}(\mathcal{S}_{\mathcal{X}\mathcal{X}}(f)) = \int_{-\infty}^{\infty} \mathcal{S}_{\mathcal{X}\mathcal{X}}(f) e^{i2\pi f\tau}\mathrm{d}f,$$

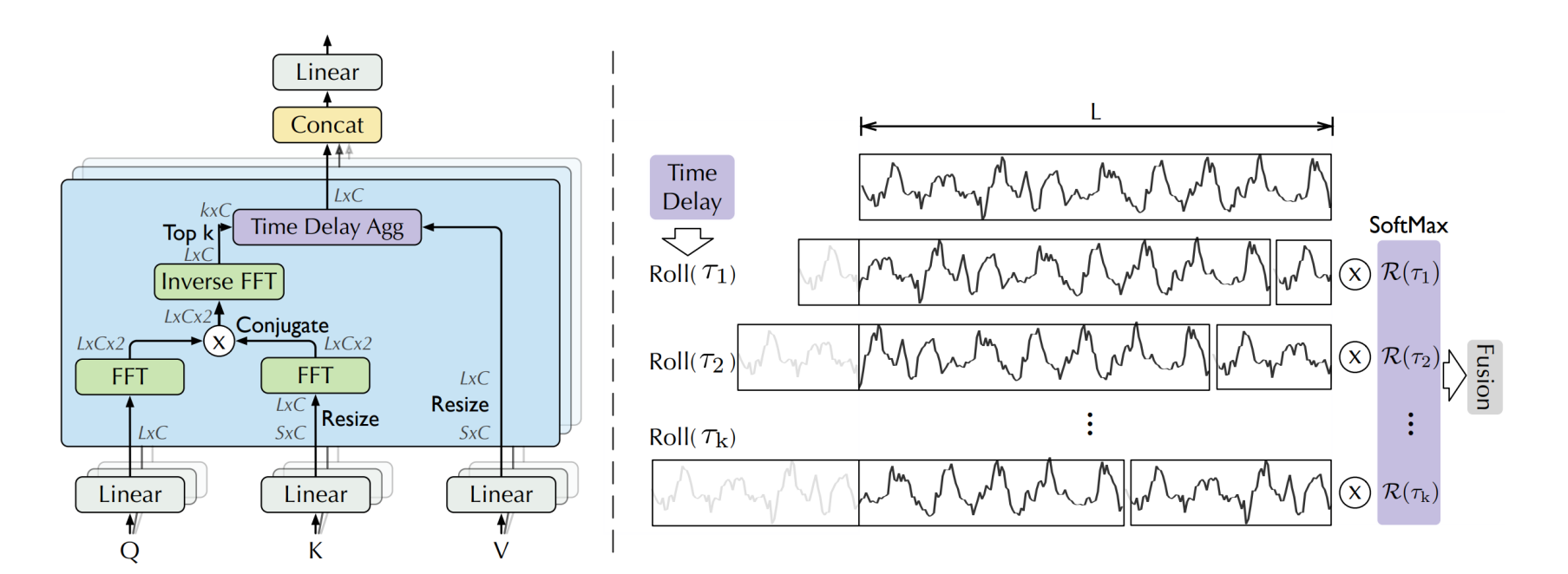
ここで計算した自己相関(と対応するラグ)のうち、
Top-k 個を使用し計算量削減



Autoformer（NeurIPS 2021）

Time Delay Aggregation

- 自己相関が大きい = 対応するラグτ だけ時間的にずれたところに似たパターン
  - それぞれのラグだけずらして、似た部分を集約
  - 余った部分は先頭に持ってくる
  - 最後に相関を重みとして乗算し、系列を合成する

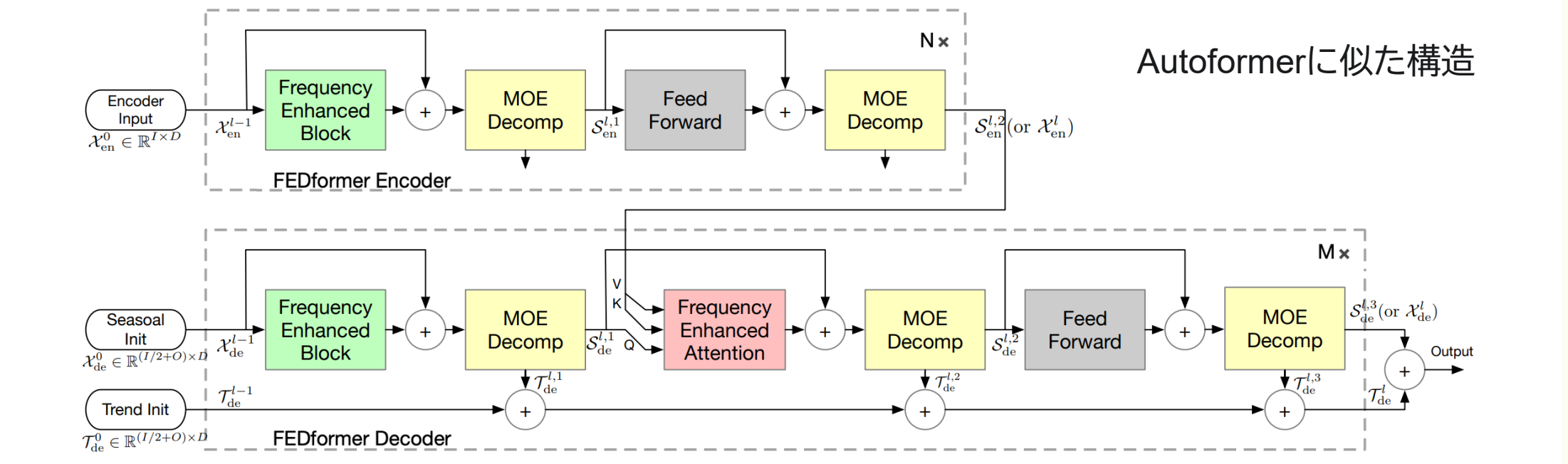


FEDformer（ICML 2022, 872 citations）

Zhou, Tian et al. “FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting.” *ICML* (2022)

【主な工夫】周波数領域の活用（時系列特徴抽出・計算量削減）

- **Mixture Of Expert Decomposition**
  - 季節-トレンド分解, さまざまなサイズの平均フィルターをもちいる
- **Frequency Enhanced Block/Attention**
  - 周波数領域での表現獲得, 計算量削減 $O(L)$



FEDformer（ICML 2022）

周波数領域でのモデリング

高周波成分だけ：ノイズが多すぎて表現が劣る
低周波成分だけ：重要なトレンド変化を見逃す

➡ ランダムに周波数成分を選択
ランダムでも十分にコンパクトな表現が得られる

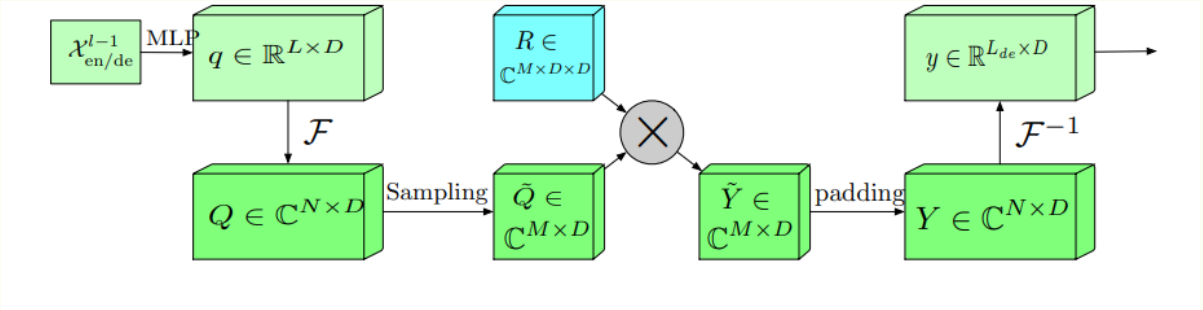
FFTで周波数領域に変換し、ランダム選択

$$\tilde{\boldsymbol{Q}} = \mathrm{Select}(\boldsymbol{Q}) = \mathrm{Select}(\mathcal{F}(\boldsymbol{q}))$$

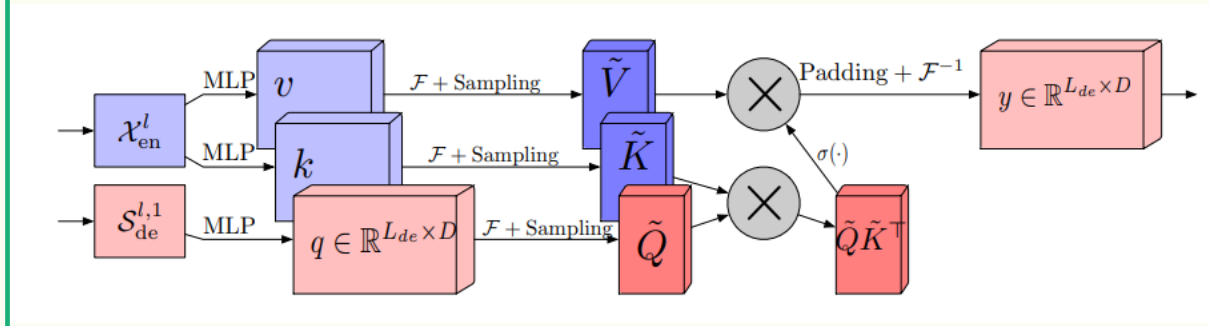
選択成分に重みをかけ、逆FFTで時間領域に戻す

$$\mathrm{FEB\text{-}f}(\boldsymbol{q}) = \mathcal{F}^{-1}(\mathrm{Padding}(\tilde{\boldsymbol{Q}} \odot \boldsymbol{R}))$$

Frequency Enhanced Block



Frequency Enhanced Attention



※Wavelet変換をもちいた場合も同様に提案されているが割愛

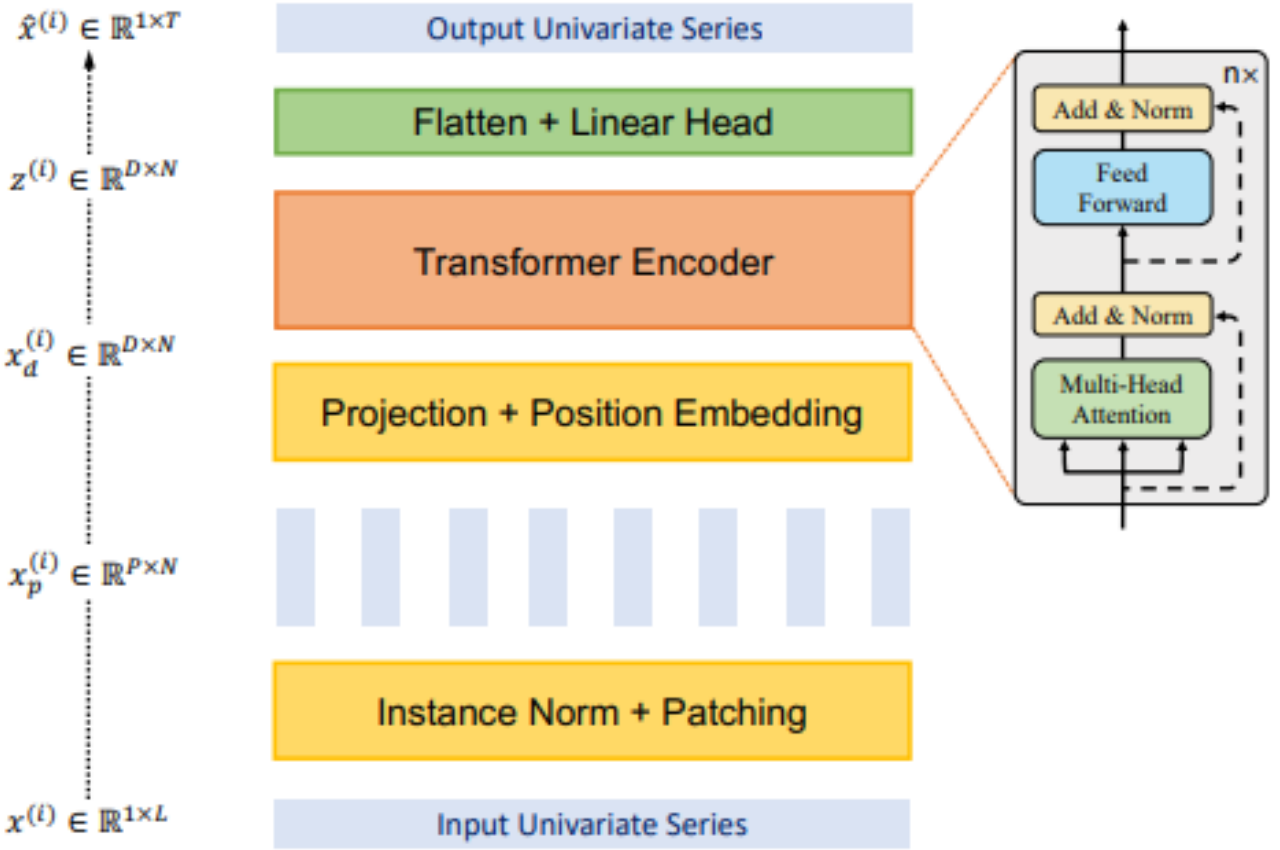
PatchTST (ICLR 2023, 495 citations)

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

【主な工夫】入力方法改善. パッチとして部分系列を切り出し、データの間引きなしで超長期時系列を扱う

Patching: Vision Transformerで提案

- まずは単変量時系列 $x \in \mathbb{R}^{1\times L}$を考える
- ストライド S で長さ P の部分系列を切り出す
- パッチ長方向をトークン次元だと思いこんでモデルに入力, $x_p \in \mathbb{R}^{P\times N}$
- 入力長を$N = \left\lfloor \frac{L-P}{S} \right\rfloor + 2$ に削減
- 加えてパッチが局所コンテキストになる

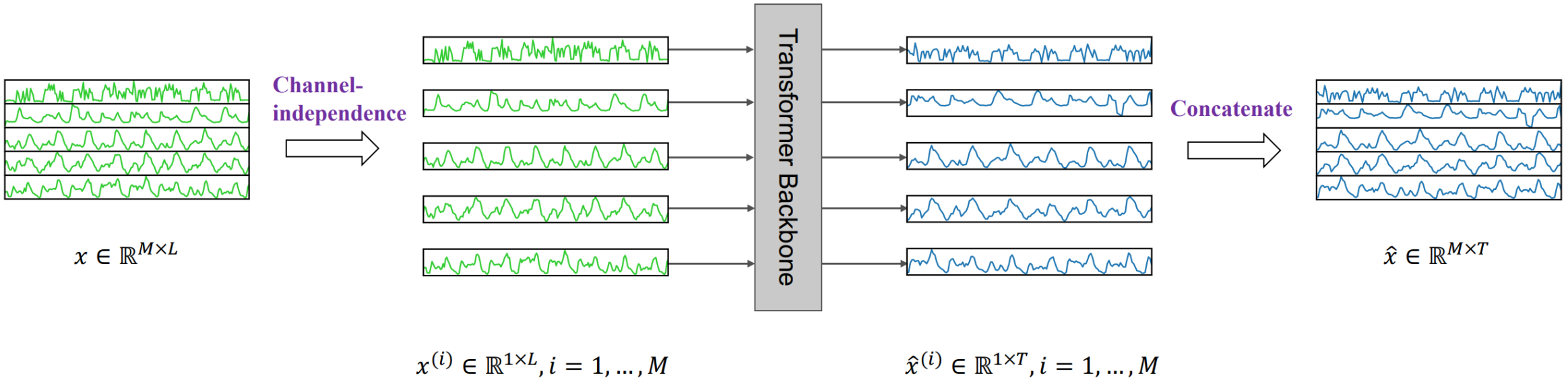


※ Vanilla Transformerの説明スライドと系列長と次元長が逆になってるため注意

PatchTST (ICLR 2023)

多変量の場合

- 各変量を単変量(Channel-independence)と考え、それぞれTransformerに入力
  - Transformerは共通のものを使用
- 最後に単変量の予測結果を結合して多変量とする
  - これで意外と性能が出るらしい [Zhao et al., 24]



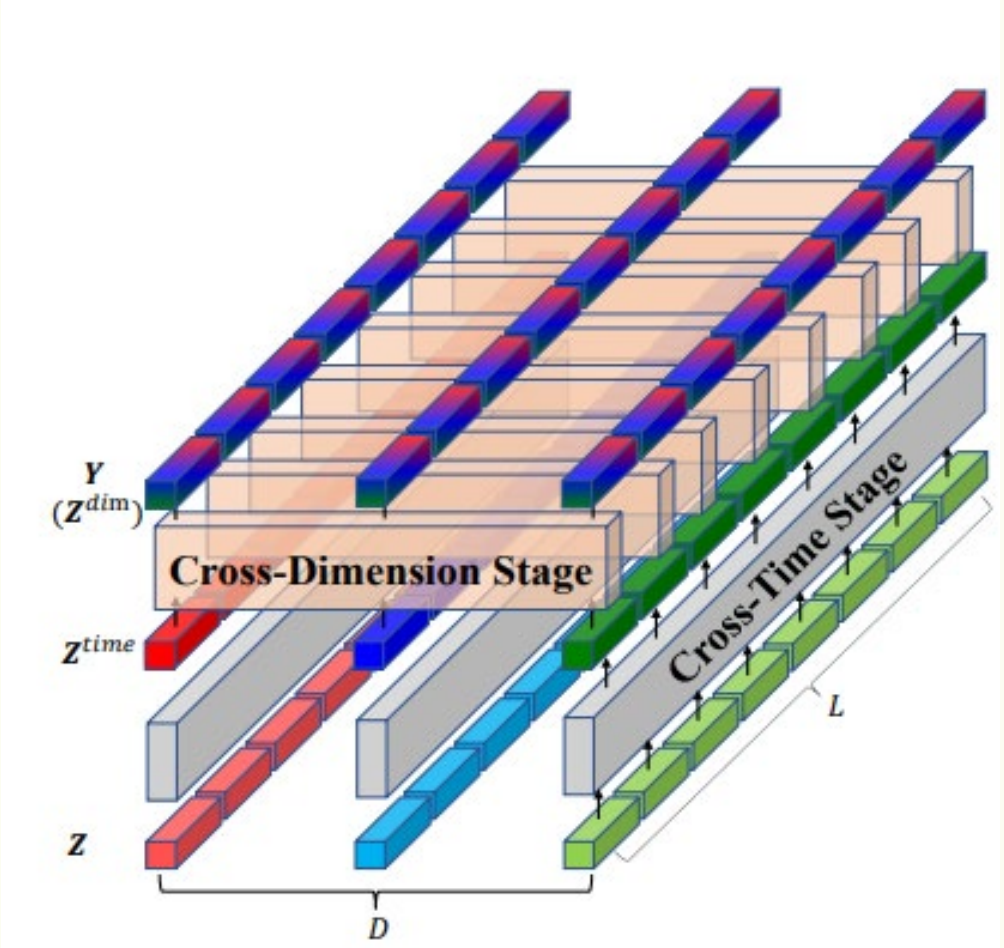
Zhao, Lifan and Yanyan Shen. "Rethinking Channel Dependence for Multivariate Time Series Forecasting: Learning from Leading Indicators." *ICLR* (2024)

Crossformer（ICLR 2023, 281 citations）

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

【主な工夫】変量間依存関係の考慮

- 時間方向/変数方向の2段階でSelf-Attention
- **Cross-Time Stage**
  - 入力をパッチに区切りSelf-Attention
  - 変量ごとにの時間方向の依存関係を獲得
- **Cross-Dimension Stage**
  - 前段で得た表現を入力として、変量方向にSelf-Attention
  - 変量間依存関係の獲得



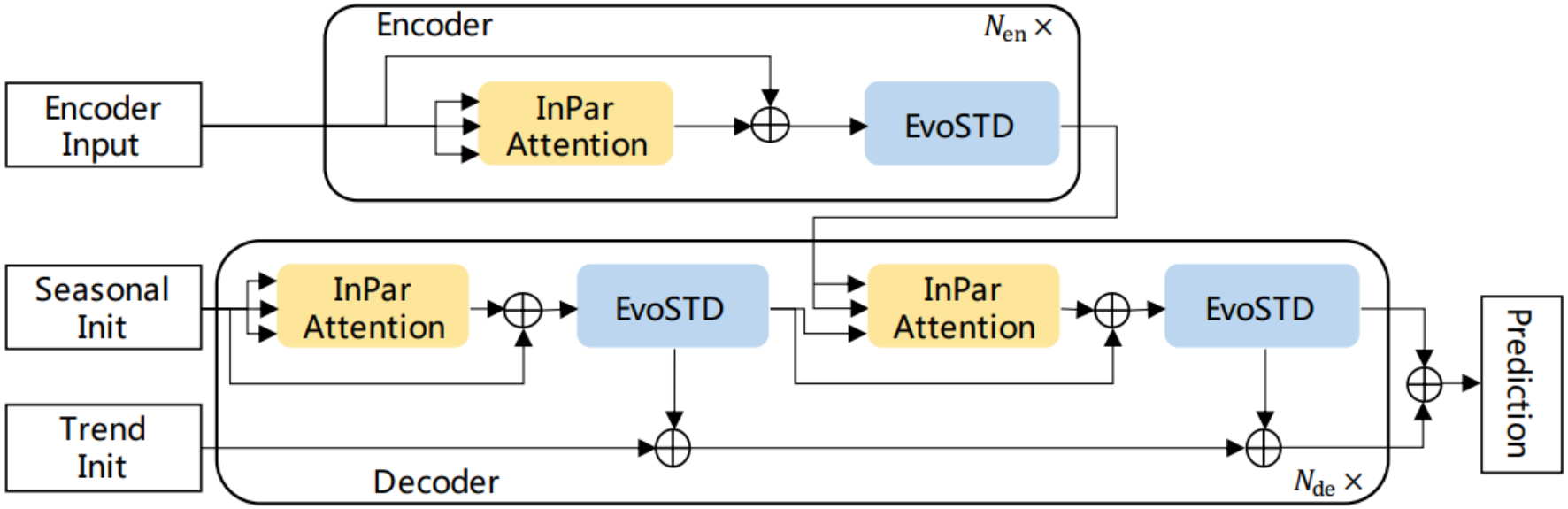
※論文中ではPatchTSTに言及がないが、同じ会議で発表のためだと思われる

InParformer（AAAI 2023, 8 citations）

Cao, Haizhou et al. “InParformer: Evolutionary Decomposition Transformers with Interactive Parallel Attention for Long-Term Time Series Forecasting.” *AAAI* (2023).

【主な工夫】周波数・時間領域両方の考慮, 季節-トレンド分解

- **Interactive parallel (InPar) attention**
  - 周波数領域・時間領域それぞれに対してAttentionの設計
- **Evolutionary seasonal-trend decomposition**
  - 季節-トレンド分解に加えて、季節成分から高周波/低周波情報の抽出



InParformer（AAAI 2023）

InPar Attention

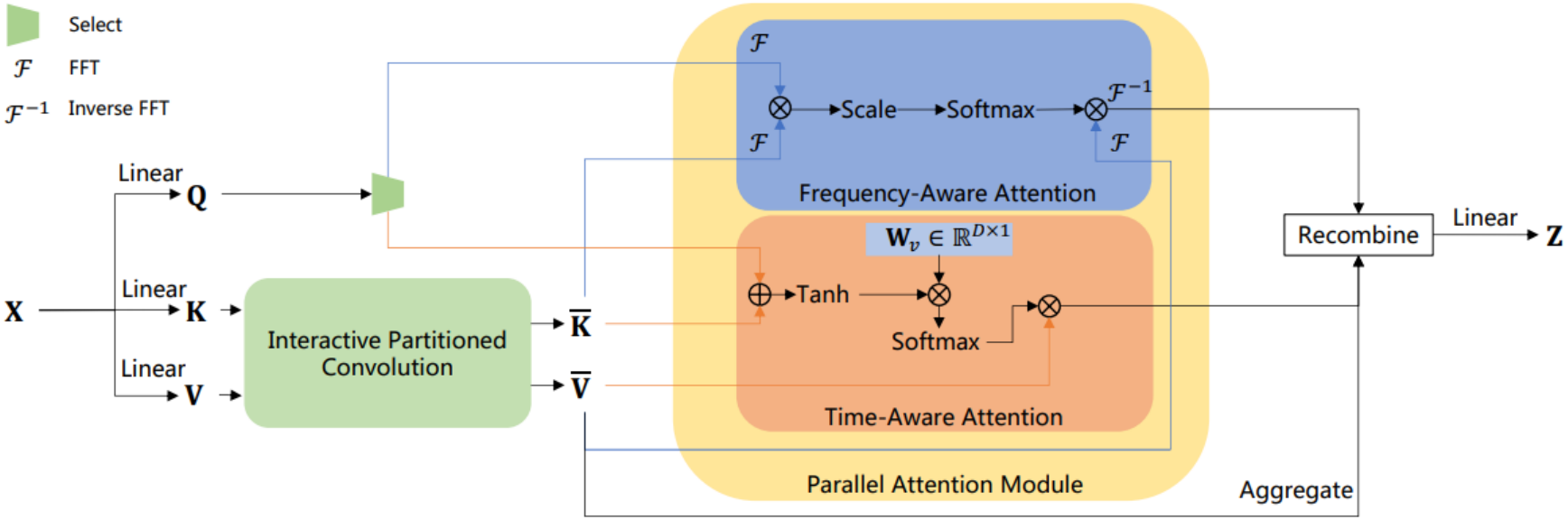
Frequency-Aware Attention（周波数領域）

$$\mathrm{FAA}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathcal{F}^{-1}(\mathrm{softmax}(\frac{\mathcal{F}(\mathbf{Q})\mathcal{F}(\mathbf{K})^{\top}}{\sqrt{D_K}})\mathcal{F}(\mathbf{V})),$$

Time-Aware Attention（時間領域）

$$\mathrm{TAA}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{softmax}(\tanh(\mathbf{Q} +^{*} \mathbf{K})\mathbf{w}_v)\mathbf{V},$$

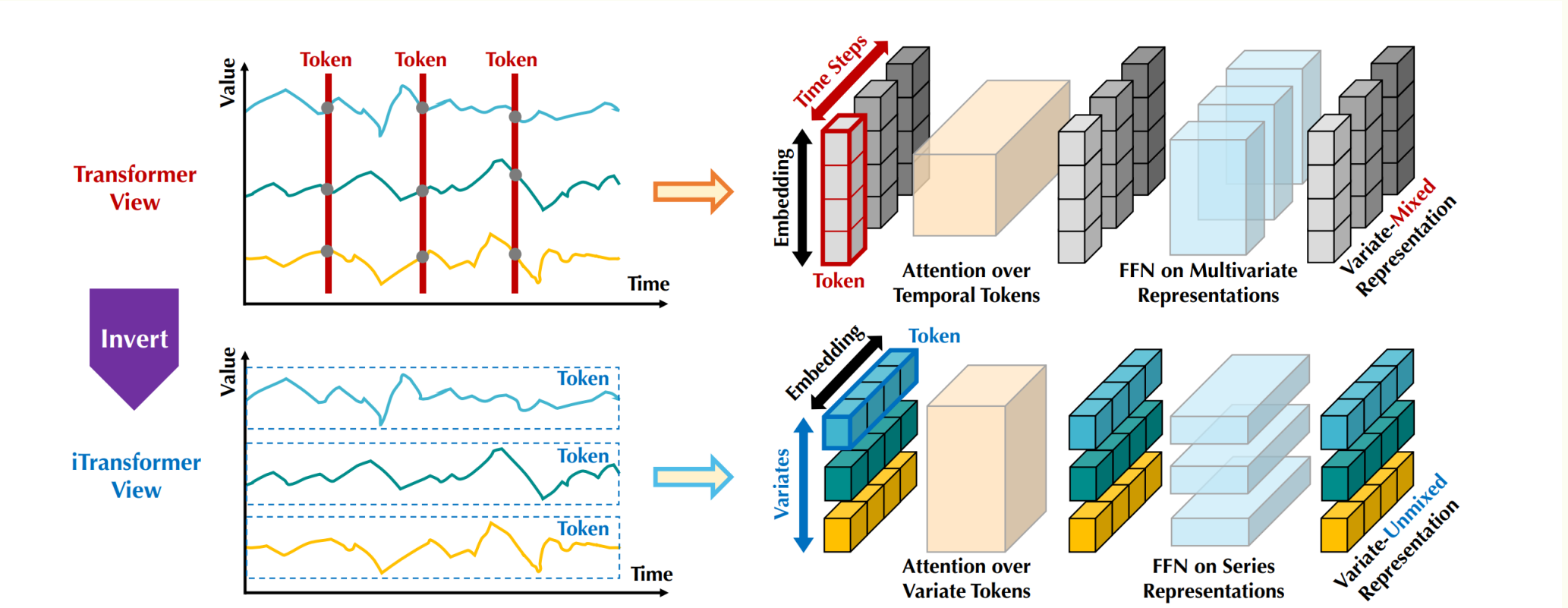
単純に2倍の計算量になるので、
Queryのランダム選択や、
Key-Valueに対して畳み込みを
行い圧縮する（InPar Conv）



iTransformer（ICLR 2024, 138 citations）

Liu, Yong et al. “iTransformer: Inverted Transformers Are Effective for Time Series Forecasting.” *ICLR* (2024)

- Transformerの入力次元を反転し、時刻方向を1トークンだとしてモデルに入力
  - PatchTSTは単変量パッチだったが、iTransformerはそれぞれのパッチが多変量時系列のイメージ
  - これにより、時系列のグローバルな表現を学習しつつ、変数間の関係も捉えられるようになる
- Encoderのみのアーキテクチャ



よく使われるテクニック

Reversible Instance Normalization（RevIN）

非定常性を緩和するための正規化手法

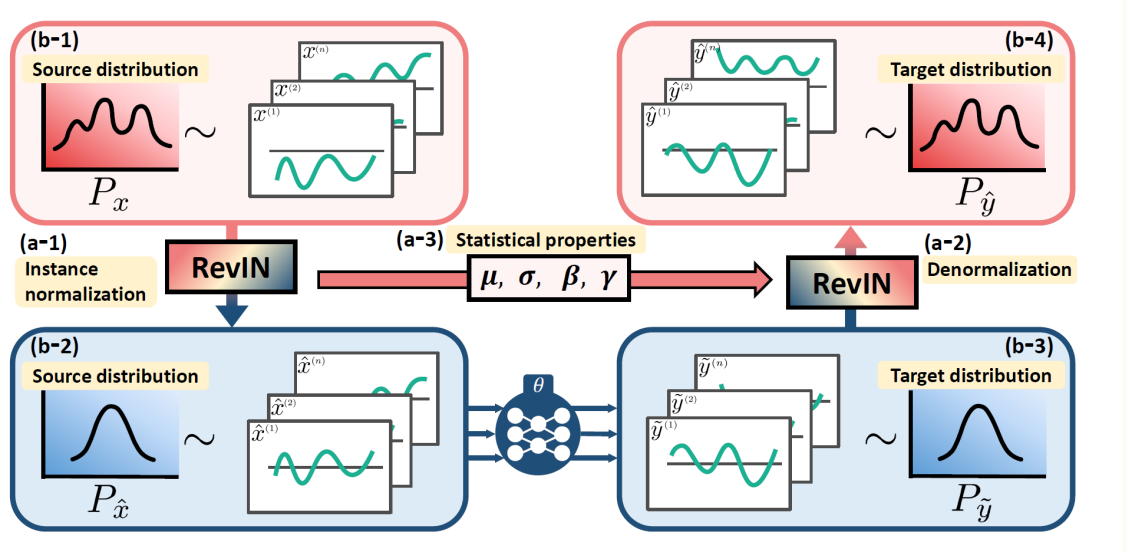
一般的な時系列解析

- 学習データの平均 μ_{train} と分散 σ^2_{train} で正規化
- テストデータは学習時の平均と分散で正規化

$$\hat{x}_{\text{train}} = \frac{x_{\text{train}} - \mu_{\text{train}}}{\sigma_{\text{train}}} \qquad \hat{x}_{\text{test}} = \frac{x_{\text{test}} - \mu_{\text{train}}}{\sigma_{\text{train}}}$$

RevIN

- モデルに入力されるデータごとに毎回正規化
- 元論文では、学習可能なパラメーターで変換がかかっているが、なくても動作する



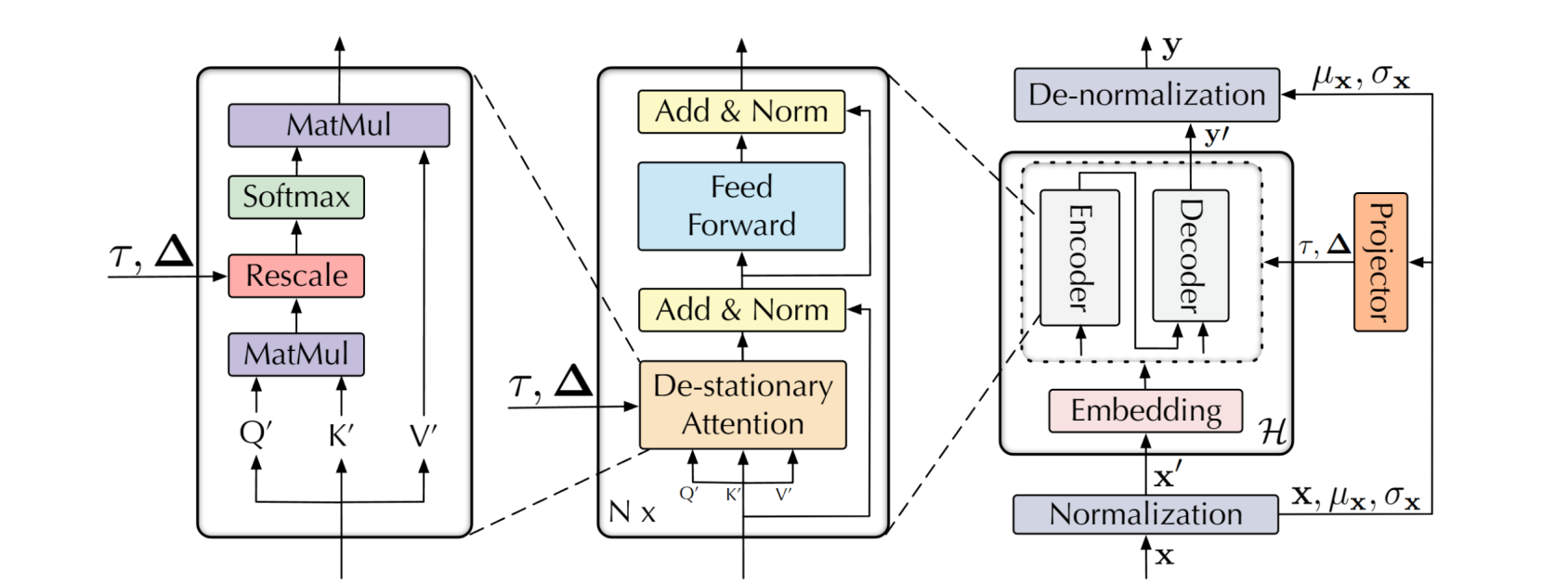
Kim, Taesung, et al. "Reversible instance normalization for accurate time-series forecasting against distribution shift." *ICLR* (2021)

Non-stationary Transformer (NeurIPS 2022, 251 citations)

Liu, Yong et al. “Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting.” *NeurIPS* (2022).

【主な工夫】非定常性への対処（系列の定常化+非定常Attention）

- RevINと同様のアイデアで系列を定常化（正規化）
- 元の系列が持っている非定常性も考慮したいので、Attentionの中では非定常に戻す（逆正規化）
  - 非定常化には、入力で計算した平均と分散を活用



まとめ

- 時系列Transformerモデルについて簡単に紹介
- 初期段階ではモデルの計算コスト削減が大きな課題としてとらえられていた
  - これは素のTransformerが $O(L^2)$ の計算コストであり、遠い過去のデータをまとめて入力したい場合のボトルネックになっているため
- それが、徐々に時系列性の考慮に注目が移ってきていた
  - 時間領域/周波数領域
  - 変数間の依存関係を捉える（相関）
  - 特に最近はパッチングが流行りな気がしている（個人の感想）
- 合わせてRevINはデフォルトの正規化としてよく用いられている
- 今回紹介した論文の著者では、Mingsheng Long（清華大学）のグループが一番多かった（Autoformer, iTransformer, Non-stationary Transformers）

参考文献

- Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. *Neural Information Processing Systems*.
- Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y. X., & Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, *32*.
- Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021, May). Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 35, No. 12, pp. 11106-11115).
- Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, *34*, 22419-22430.
- Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022, June). Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning* (pp. 27268-27286). PMLR.
- Zhao, L., & Shen, Y. (2024). Rethinking Channel Dependence for Multivariate Time Series Forecasting: Learning from Leading Indicators. ICLR 2024

参考文献

- Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2022). A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.
- Zhang, Y., & Yan, J. (2023, May). Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*.
- Cao, H., Huang, Z., Yao, T., Wang, J., He, H., & Wang, Y. (2023, June). Inparformer: Evolutionary decomposition transformers with interactive parallel attention for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 37, No. 6, pp. 6906-6915).
- Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., & Long, M. (2023). itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*.
- Kim, T., Kim, J., Tae, Y., Park, C., Choi, J. H., & Choo, J. (2021, May). Reversible instance normalization for accurate time-series forecasting against distribution shift. *In International Conference on Learning Representations*.
- Liu, Y., Wu, H., Wang, J., & Long, M. (2022). Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, *35*, 9881-9893.