Transformers for Time Series Forecasting

Marco Zanotti

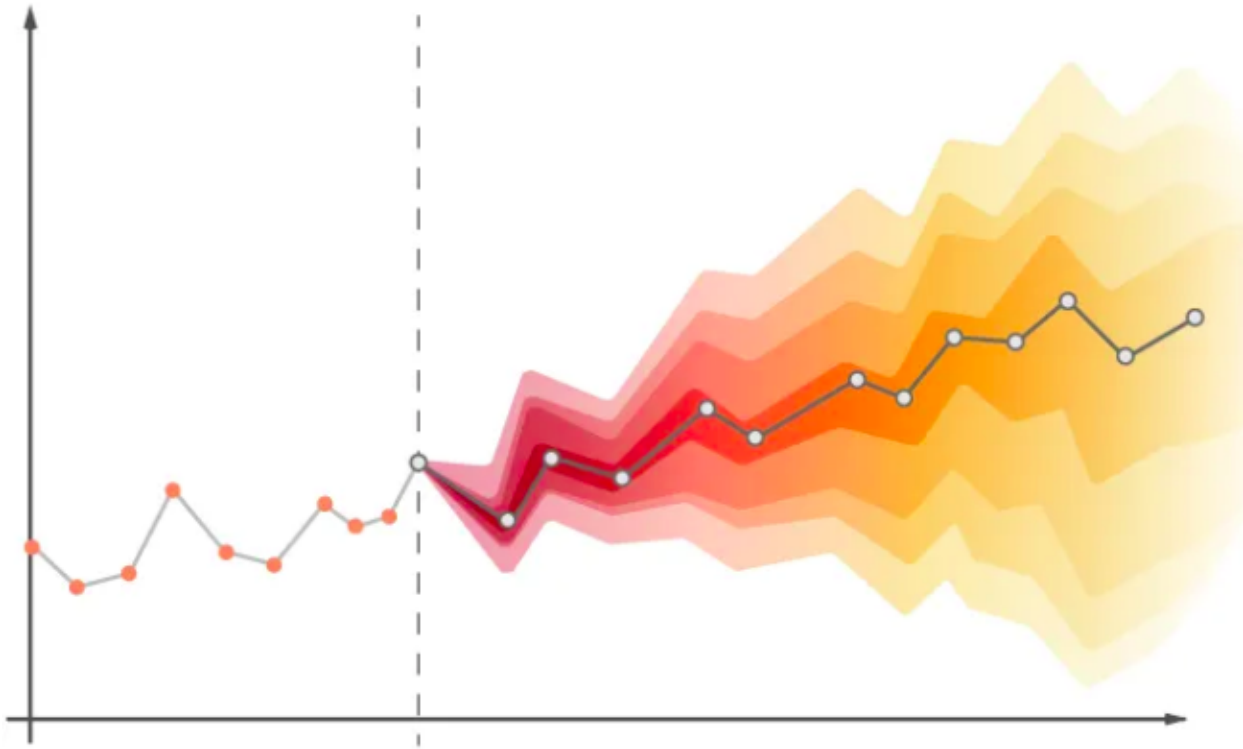
University Milano-Bicocca

Contents

1. The TSF Problem
2. Vanilla Transformer
3. TSF Transformers
4. Conclusions

1. The TSF Problem

Time series forecasting (TSF) is the task of predicting future values of a given sequence based on previously observed values.



The TSF problem may be essentially identified by the following aspects:

- **Prediction objective**: point forecasting vs probabilistic forecasting
- **Forecast horizon**: short-term vs long-term forecasting
- **Input-Output dimension**: univariate vs multivariate forecasting
- **Forecasting task**: single-step vs multi-step forecasting

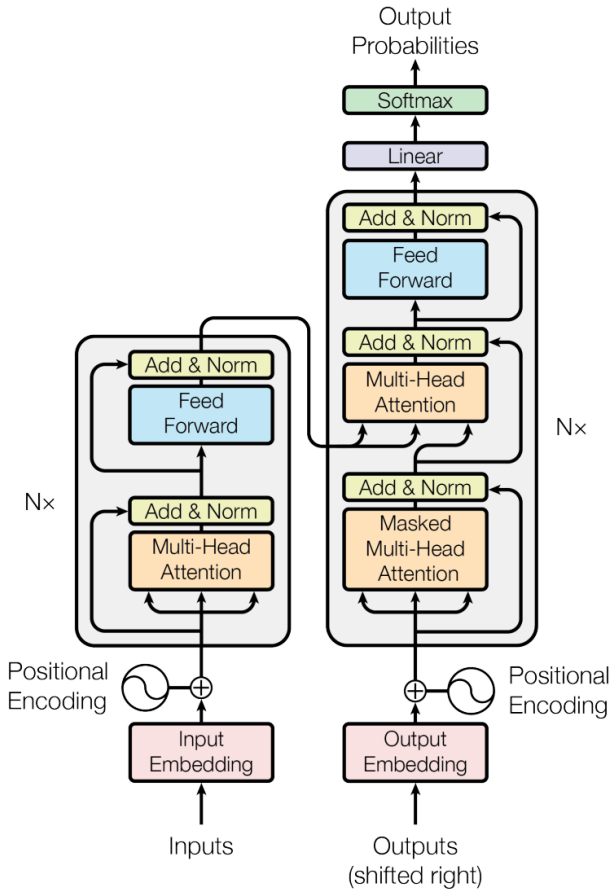
The TSF problem is usually faced with statistical models (ARIMA, ETS) or deep learning models (RNN, LSTM).

The main challenges of the TSF problem are:

- **uncertainty** increases as the forecast horizon increases
- difficulty in capturing **multiple complex patterns** over time
- difficulty in capturing **long-term dependencies** (critical for long-term forecasting)
- difficulty to handle **long input sequences**

2. Vanilla Transformer

- Based on Encoder-Decoder architecture
- Uses self-attention mechanism to access any part of the sequence history
- Positional encoding allows to account for element positions
- Residual connections and layer normalization help to stabilize the learning process
- Each encoder and decoder layer is composed of a self-attention layer and a feed-forward layer



Can vanilla Transformers be used for TSF?

The TSF problem can be seen as a sequence learning problem such as machine translation.

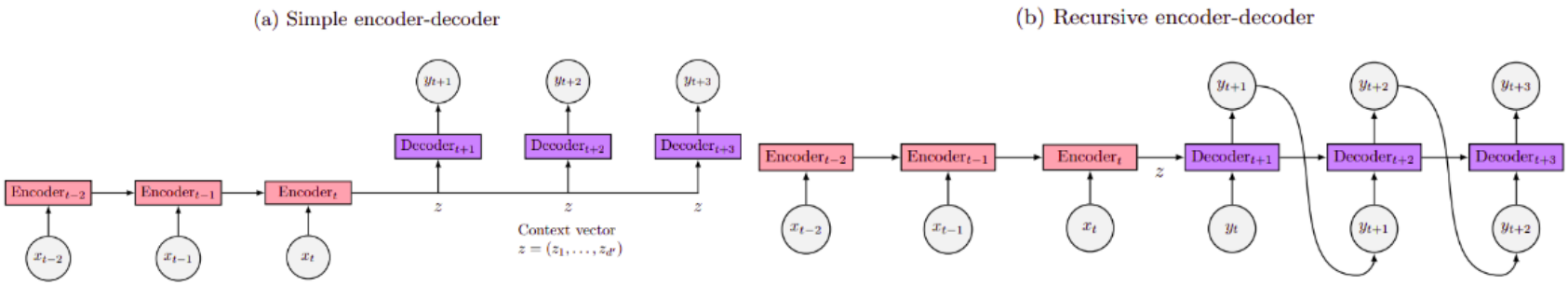
Main ingredients allowing to use vanilla Transformers for TSF:

- **Multi-head Self-attention** mechanism allows to access any part of the sequence history, capturing both short-term and long-term dependencies (but it is invariant to the order of elements in a sequence)
- **Positional encoding** allows to account for the sequence ordering
- **Masked self-attention** allows to avoid information leakage from future

Can vanilla Transformers be used for TSF?

Just few changes are needed to adapt Transformers to TSF:

- **Remove the final activation** function (softmax) from the output layer and set the dimension of the linear layer equal to the forecasting horizon
- **Adapt the structure** to the desired forecasting task (single-step or multi-step)



Problems with vanilla Transformers

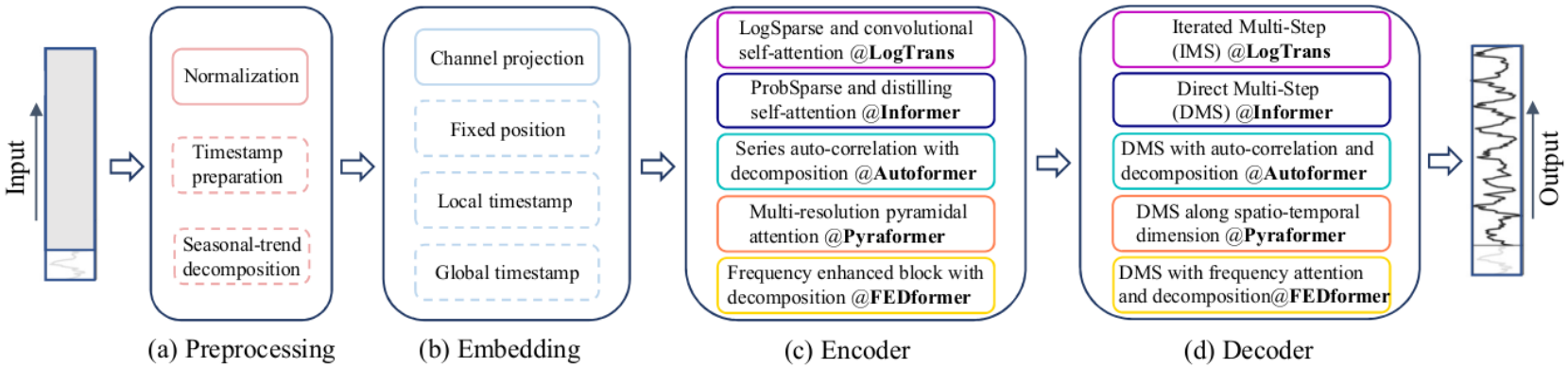
- **Locally agnostic**: the attention mechanism matches queries and keys without considering their local context being prone to temporal anomalies
- **Positional encoding**: only the order in which two elements occur is taken into account, but their temporal distance is not
- **Computational complexity**: given a sequence of length L, the time and memory burden is $O(L^2)$, making it difficult to learn patterns in long time series
- **Simple Architecture**: the architecture does not include any component of typical importance in TSF (e.g. autocorrelation, decomposition, recurrent layers, etc.)

3. TSF Transformers

Classification

Transformers are **very appealing for long-term TSF** due to their ability to learn long-range dependencies.

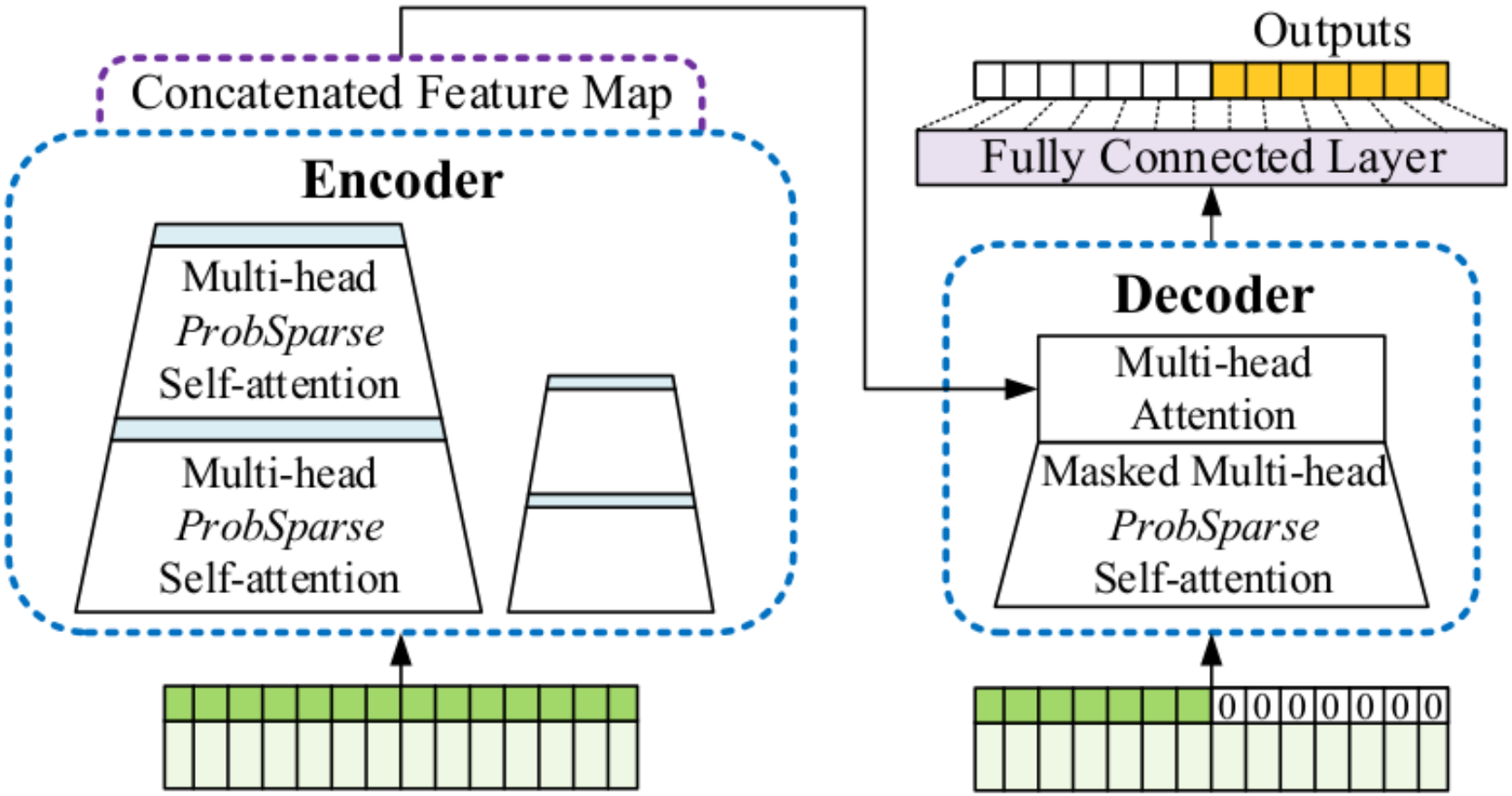
Many solutions have been proposed to adapt Transformers to TSF, mainly in the direction to improve the encoding, adopt more efficient attention and expand the architecture.



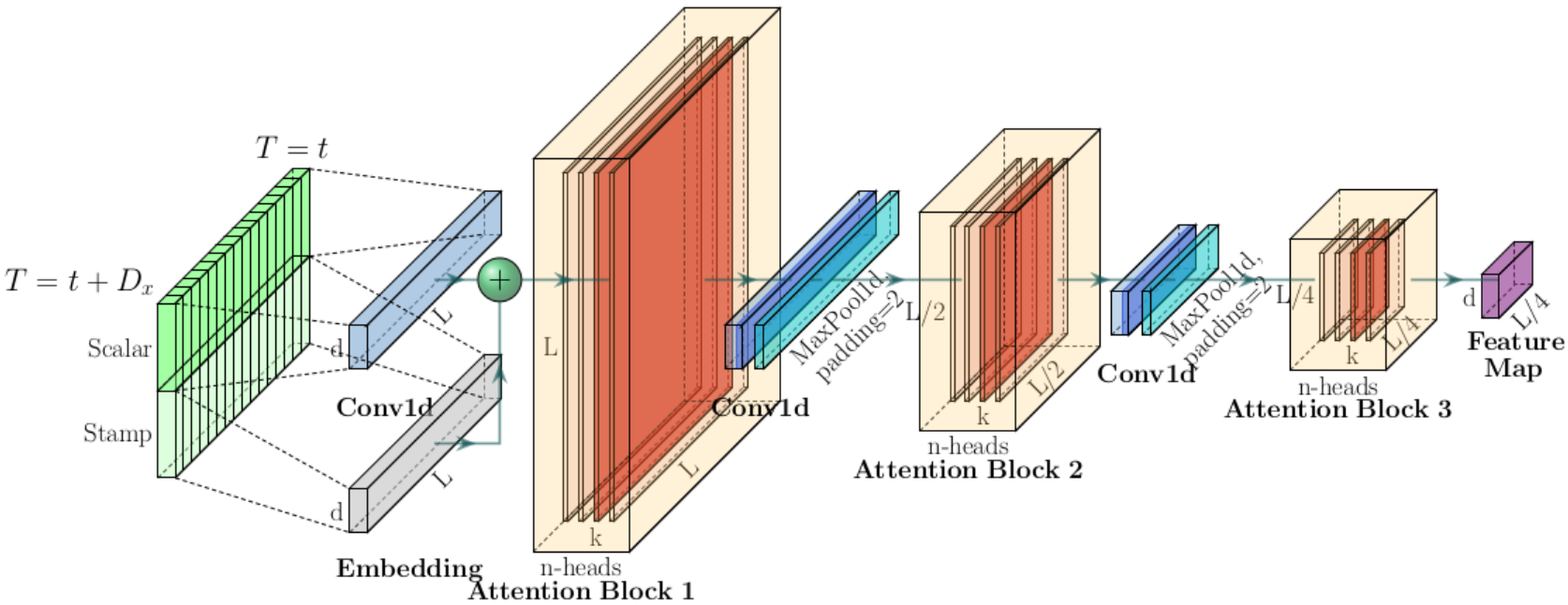
Informer

aa

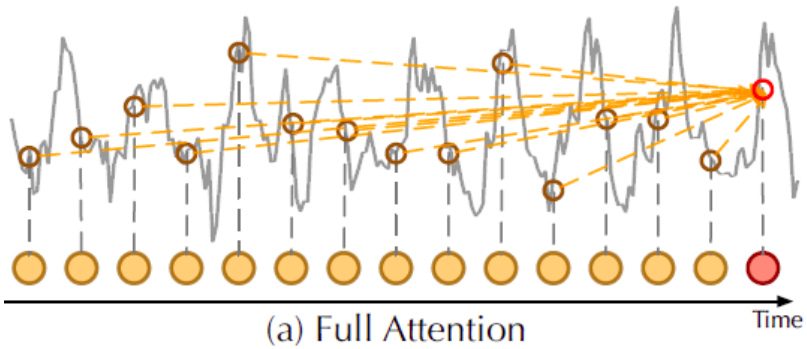
Informer - Architecture



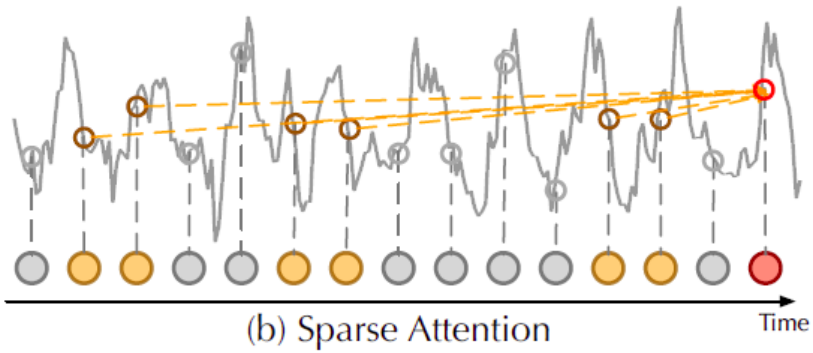
Informer - Causal Convolution Layers



Informer - ProbSparse Attention



(a) Full Attention



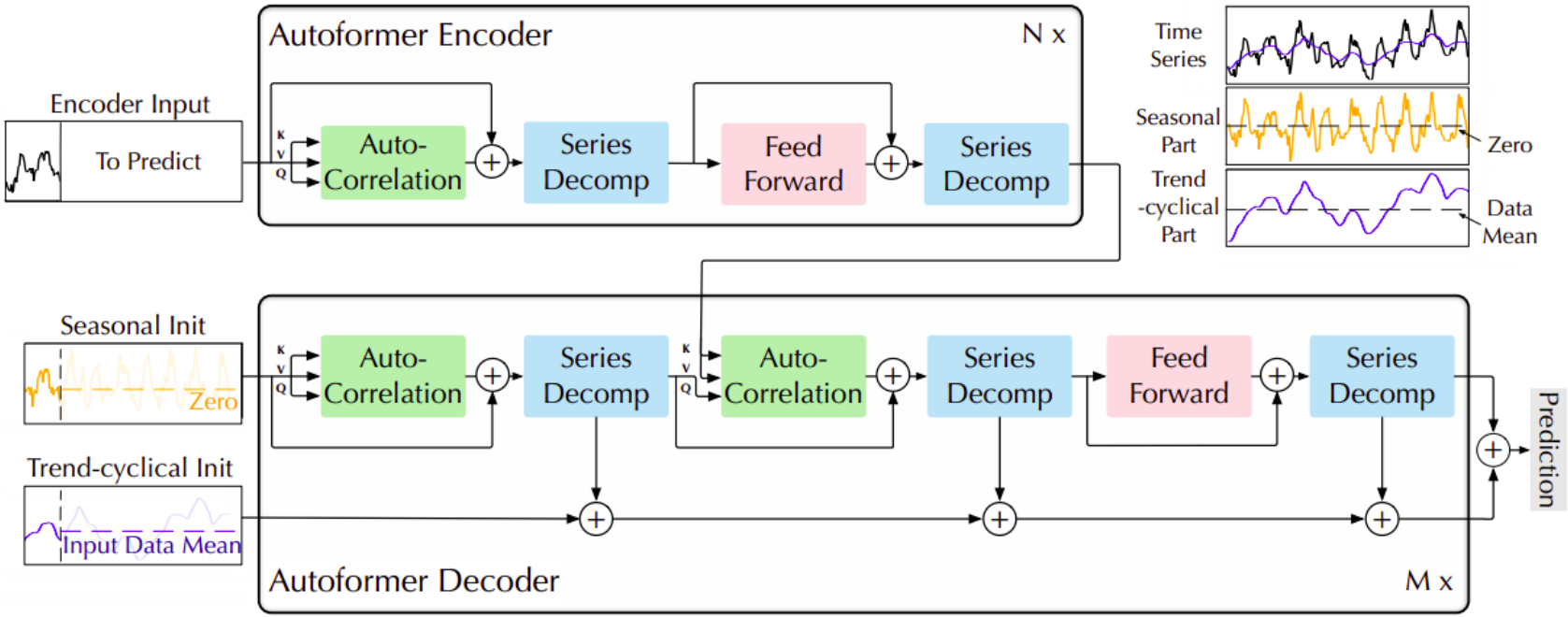
(b) Sparse Attention

Autoformer

Autoformer builds upon two traditional time series analysis methods:

- **Decomposition Layer**, which allows to decompose the time series into seasonality and trend-cycle components, enhancing the model's ability to capture these components accurately
- **Attention (Autocorrelation) Mechanism**, which replaces the standard self-attention used in the vanilla transformer with an autocorrelation mechanism, allowing to capture the temporal dependencies in the frequency domain

Autoformer - Architecture



Autoformer - Decomposition Layer

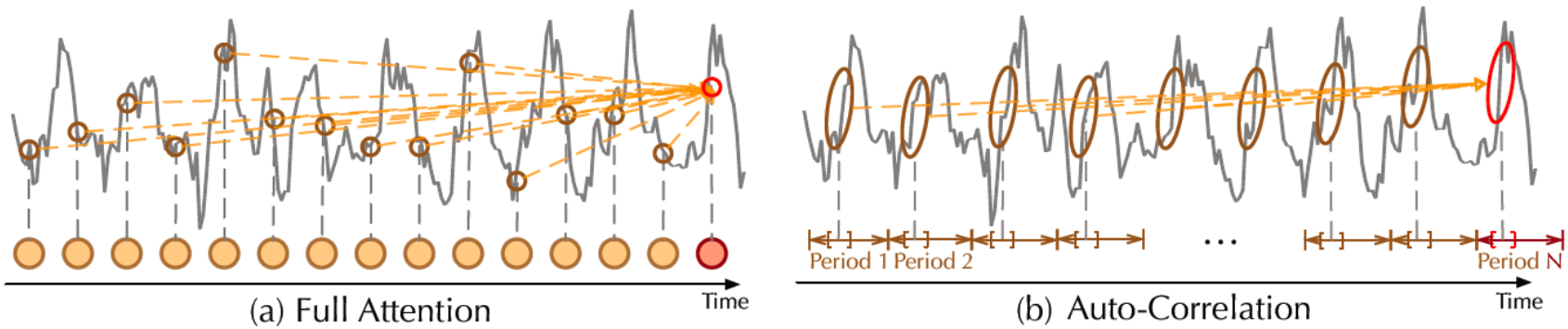
Autoformer incorporates **decomposition blocks** as an inner operation of the model.

Encoder and decoder use decomposition blocks to extract and aggregate the trend-cyclical and seasonal parts from the series progressively.

For an input series X_t with length L, the decomposition layer returns X_{trend} and $X_{seasonal}$, both of length L.

Autoformer - Attention Mechanism

Autoformer uses **autocorrelation within the self-attention** mechanism, extracting frequency-based dependencies from queries and keys (instead of the standard dot-product).



In practice, autocorrelation of the queries and keys for all lags is calculated at once by Fast Fourier Transform, so to achieve $O(LlogL)$ time complexity (similar to Informer's ProbSparse attention).

4. Conclusions

Conclusions

- aa

Bibliografy

Ailing Z., et al., 2023, 'Are Transformers Effective for Time Series Forecasting?', AAAI

Haixu W., et al., 2021, 'Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting', NeurIPS

Haoyi Z., et al., 2021, 'Informer: Beyond efficient transformer for long sequence time-series forecasting', AAAI

Lara-Benitez P., et al., 2021, 'Evaluation of the Transformer Architecture for Univariate Time Series Forecasting', Advances in Artificial Intelligence, CAEPIA

Qingsong W., et al., 2022, 'Transformers in Time Series: A Survey', AAAI

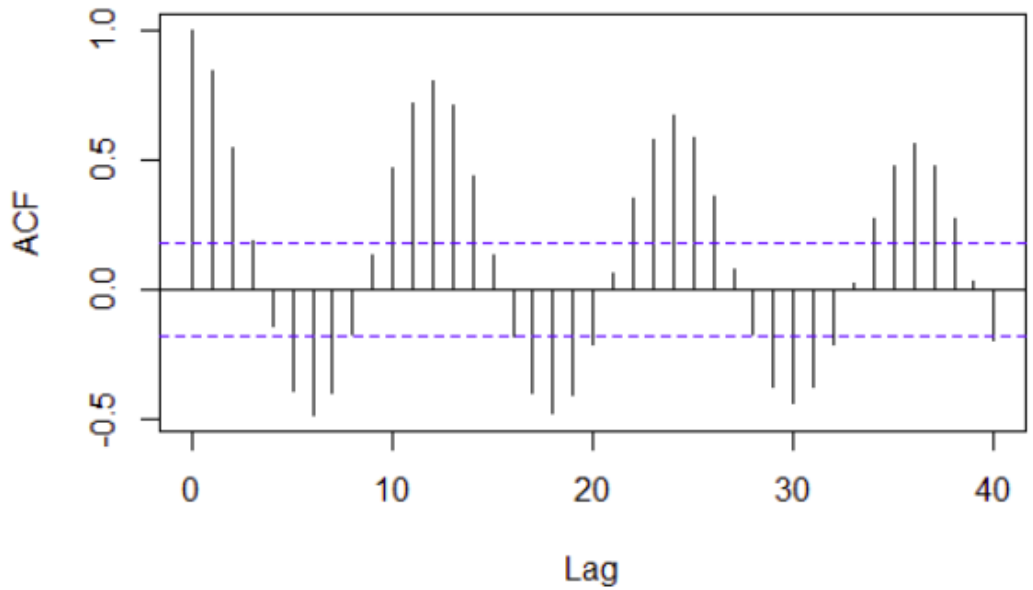
Thank you!

Appendix

Autocorrelation

In theory, given a time lag k, autocorrelation for a single discrete variable Y is used to measure the "relationship" (pearson correlation) between the variable's current value at time t to its past value at time $t - k$.

$$Autocorrelation(k) = Corr(Y_t, Y_{t-k})$$



Time Series Decomposition

In time series analysis, decomposition is a method of breaking down a time series into three systematic components: trend-cycle, seasonal variation, and random fluctuations.