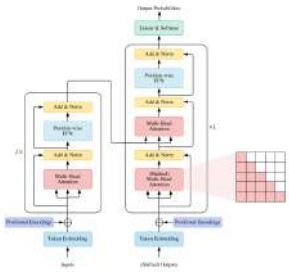
Time series forecasting papers

Attention Is All You Need (https://arxiv.org/pdf/1706.03762.pdf)



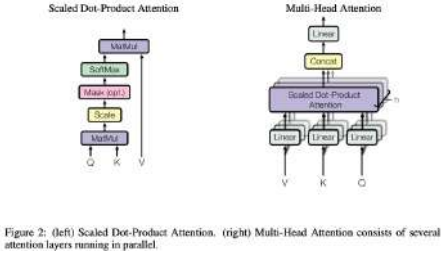


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

A Transformer is generally made of a collection of attention mechanisms, embeddings to encode some positional information, feed-forward blocks and a residual path (typically referred to as pre- or post- layer norm).

Temporal Fusion Transformers (https://arxiv.org/pdf/1912.09363.pdf)

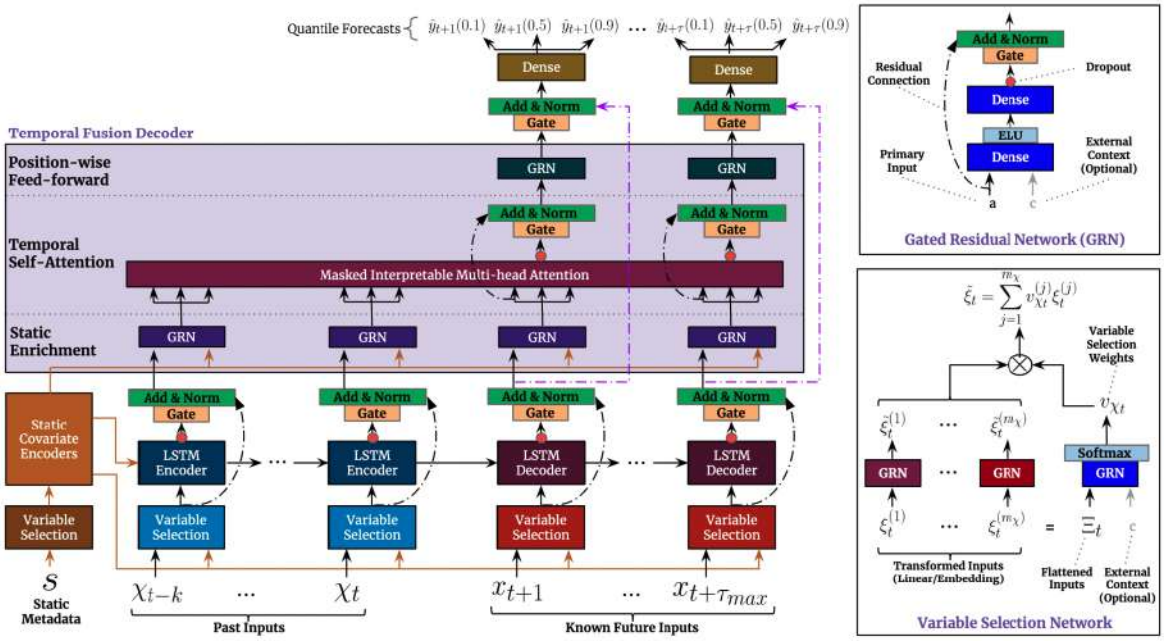


Figure 2: TFT architecture. TFT inputs static metadata, time-varying past inputs and time-varying a priori known future inputs. Variable Selection is used for judicious selection of the most salient features based on the input. Gated Residual Network blocks enable efficient information flow with skip connections and gating layers. Time-dependent processing is based on LSTMs for local processing, and multi-head attention for integrating information from any time step.

Hopfield Networks is All You Need (https://arxiv.org/pdf/2008.02217.pdf)

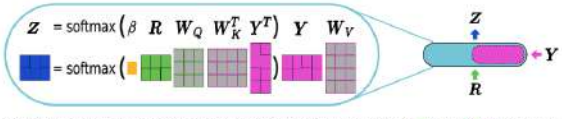


Figure 3: The layer `Hopfield` allows the association of two sets R (■) and Y (■). It can be integrated into deep networks that propagate sets of vectors. The Hopfield memory is filled with a set from either the input or previous layers. The output is a set of vectors Z (■).

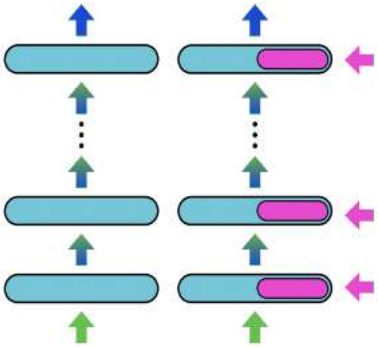


Figure 2: Left: A standard deep network with layers (■) propagates either a vector or a set of vectors from the input to the output. Right: A deep network, where layers (■) are equipped with associative memories via Hopfield layers (■).

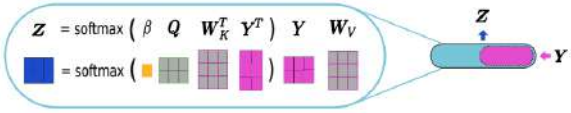


Figure 4: The layer `HopfieldPooling` enables pooling or summarization of sets, which are obtained from the input or from previous layers. The input Y (■) can be either a set or a sequence. The query patterns of each layer are static and can be learned. The output is a set of vectors Z (■), where the number of vectors equals the number of query patterns. The layer `HopfieldPooling` can realize multiple instance learning.

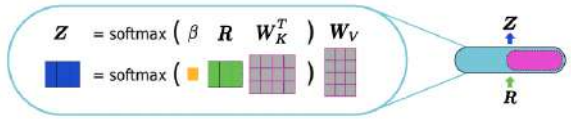


Figure 5: The layer `HopfieldLayer` enables multiple queries of the training set, a reference set, prototype set, or a learned set (a learned matrix). The queries for each layer are computed from the results of previous layers. The input is a set of vectors R (■). The output is also a set of vectors Z (■), where the number of output vectors equals the number of input vectors. The layer `HopfieldLayer` can realize SVM models, k-nearest neighbor, and LVQ.

SAnD (Simply Attend and Diagnose) (https://arxiv.org/pdf/1711.03905.pdf)

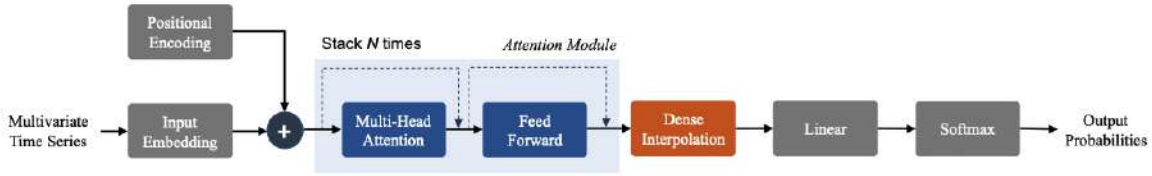


Figure 1: An overview of the proposed approach for clinical time-series analysis. In contrast to state-of-the-art approaches, this does not utilize any recurrence or convolutions for sequence modeling. Instead, it employs a simple self-attention mechanism coupled with a dense interpolation strategy to enable sequence modeling. The attention module is comprised of N identical layers, which in turn contain the attention mechanism and a feed-forward sub-layer, along with residue connections.

SCINet (https://arxiv.org/pdf/2106.09305.pdf)

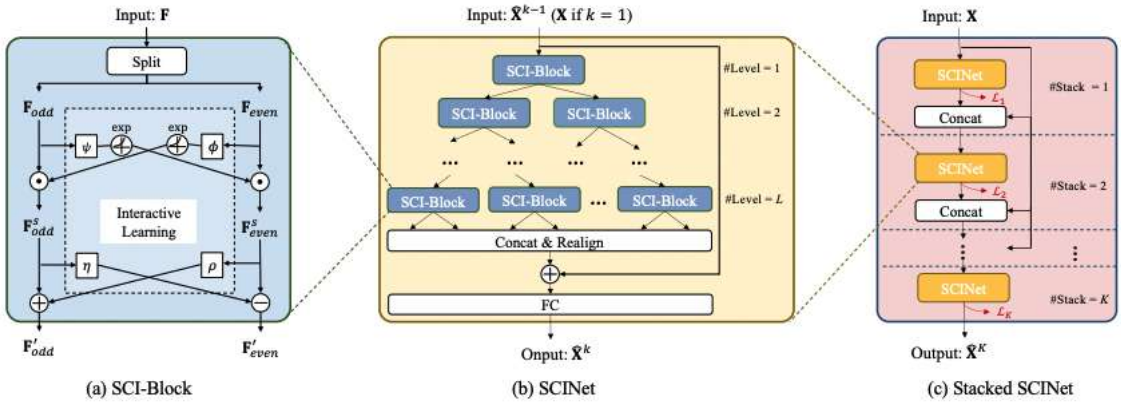


Figure 2: The overall architecture of Sample Convolution and Interaction Network (SCINet).

DEPTS (https://openreview.net/forum?id=AJAR-JgNw__)

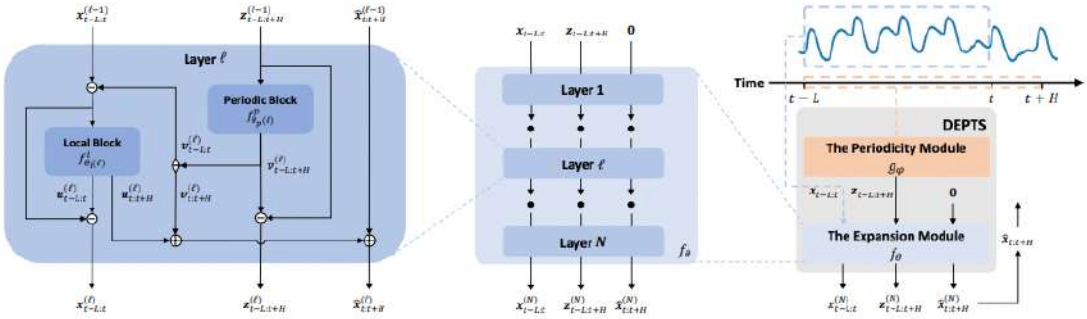


Figure 2: In the right part, we visualize the overall data flows for our framework, DEPTS. In the middle part, we plot the integral structure of three layer-by-layer expansion branches in the expansion module f_θ. In the left part, we depict the detailed residual connections within a single layer.

S4 (https://srush.github.io/annotated-s4/)

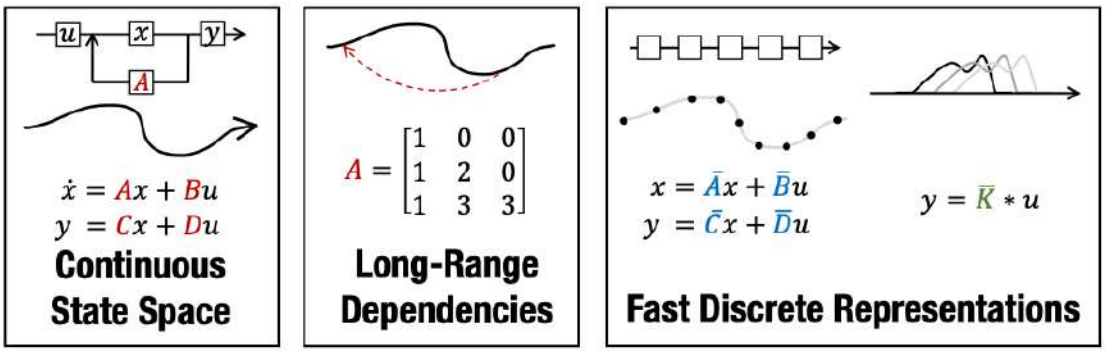
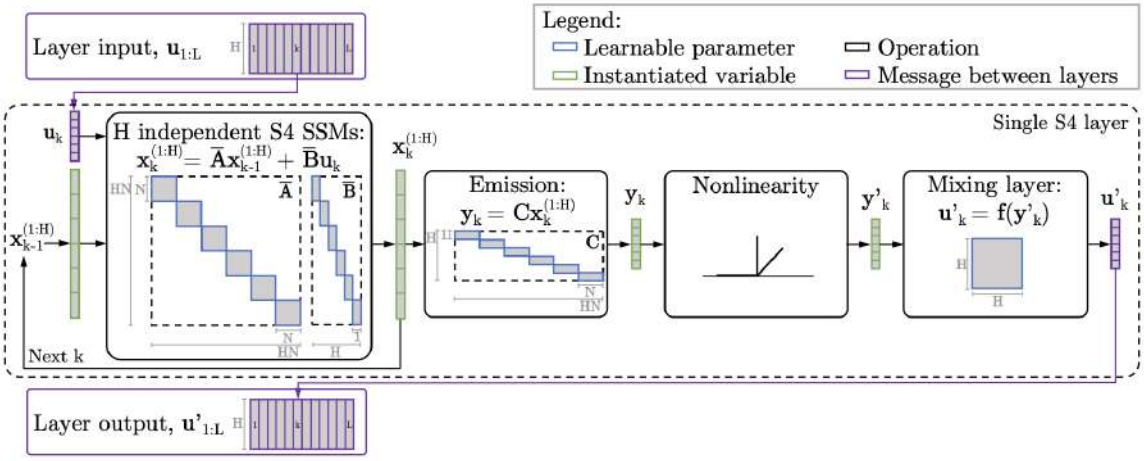


Figure 1: (**Left**) State Space Models (SSM) parameterized by matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}$ map an input signal $u(t)$ to output $y(t)$ through a latent state $x(t)$. (**Center**) Recent theory on continuous-time memorization derives special $\boldsymbol{A}$ matrices that allow SSMs to capture LRDs mathematically and empirically. (**Right**) SSMs can be computed either as a recurrence (left) or convolution (right). However, materializing these conceptual views requires utilizing different representations of its parameters (red, blue, green) which are very expensive to compute. S4 introduces a novel parameterization that efficiently swaps between these representations, allowing it to handle a wide range of tasks, be efficient at both training and inference, and excel at long sequences.



ETSformer (https://arxiv.org/abs/2202.01381)

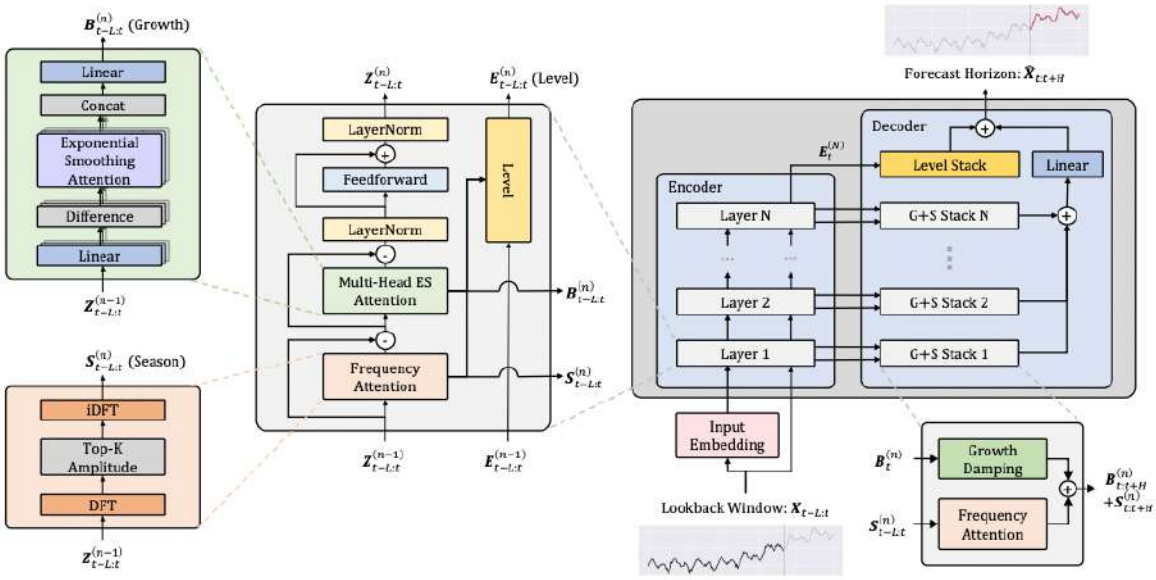


Figure 2. ETSformer model architecture.

Pyraformer (https://openreview.net/pdf?id=0EXmFzUn5I)

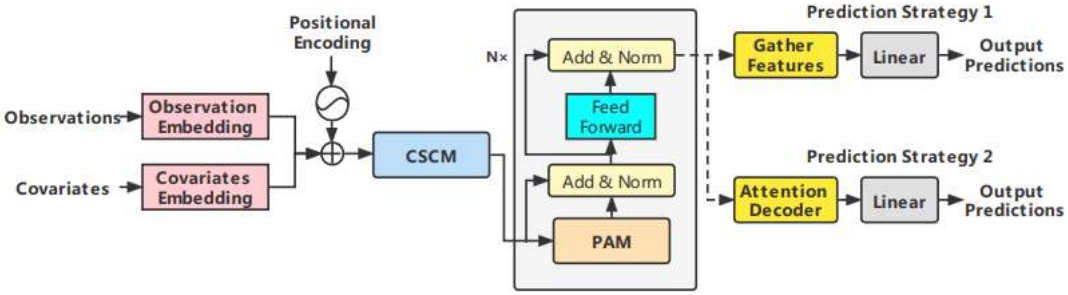


Figure 2: The architecture of Pyraformer: The CSCM summarizes the embedded sequence at different scales and builds a multiresolution tree structure. Then the PAM is used to exchange information between nodes efficiently.

Informer (https://arxiv.org/abs/2012.07436)

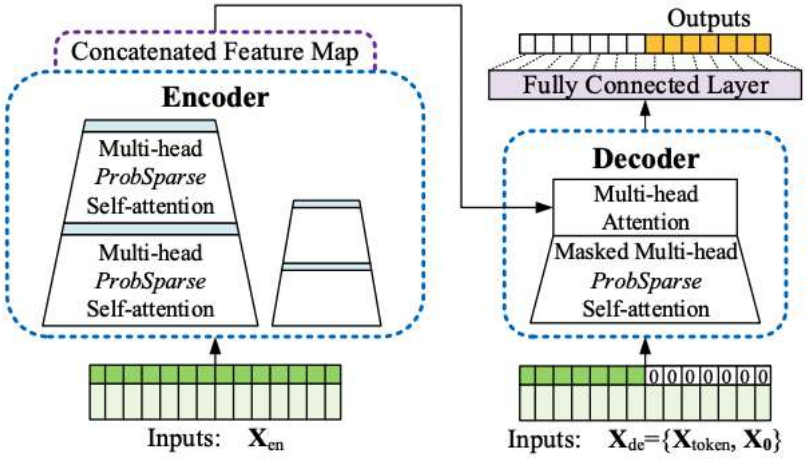


Figure 2: Informer model overview. Left: The encoder receives massive long sequence inputs (green series). We replace canonical self-attention with the proposed *ProbSparse* self-attention. The blue trapezoid is the self-attention distilling operation to extract dominating attention, reducing the network size sharply. The layer stacking replicas increase robustness. Right: The decoder receives long sequence inputs, pads the target elements into zero, measures the weighted attention composition of the feature map, and instantly predicts output elements (orange series) in a generative style.

Reformer (https://arxiv.org/pdf/2001.04451.pdf)

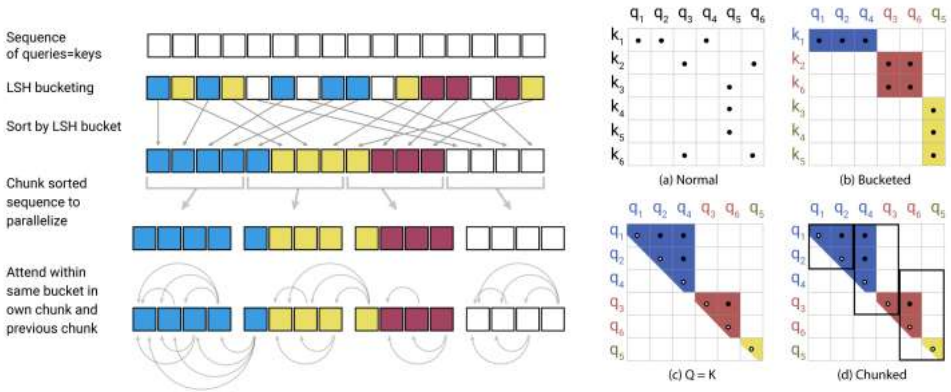


Figure 2: Simplified depiction of LSH Attention showing the hash-bucketing, sorting, and chunking steps and the resulting causal attentions. (a-d) Attention matrices for these varieties of attention.

N-HiTS (https://arxiv.org/pdf/2201.12886.pdf)

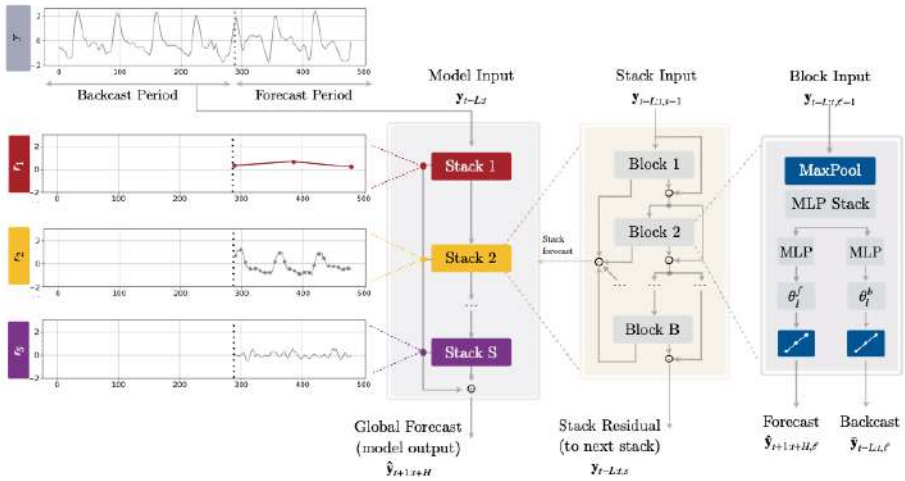


Figure 2. N-HiTS architecture. The model is composed of several MLPs with ReLU nonlinearities. Blocks are connected via doubly residual stacking principle with the backcast $\tilde{y}_{t-L:t,\ell}$ and forecast $\hat{y}_{t+1:t+H,\ell}$ outputs of the ℓ-th block. Multi-rate input pooling, hierarchical interpolation and backcast residual connections together induce the specialization of the additive predictions in different signal bands, reducing memory footprint and compute time, improving architecture parsimony and accuracy.

Autoformer (https://arxiv.org/pdf/2106.13008.pdf)

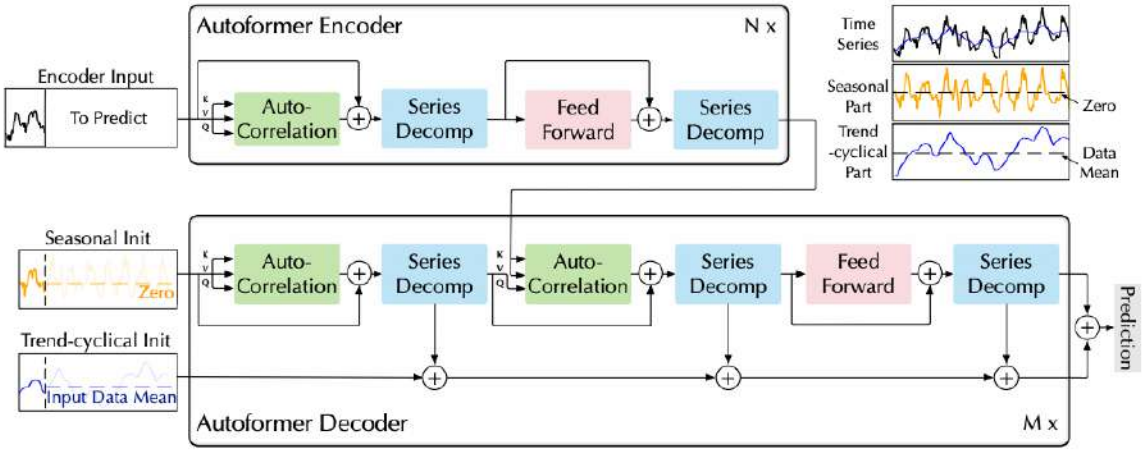


Figure 1: Autoformer architecture. The encoder eliminates the long-term trend-cyclical part by series decomposition blocks (blue blocks) and focuses on seasonal patterns modeling. The decoder accumulates the trend part extracted from hidden variables progressively. The past seasonal information from encoder is utilized by the encoder-decoder Auto-Correlation (center green block in decoder).

LogTrans (https://arxiv.org/pdf/1907.00235.pdf)

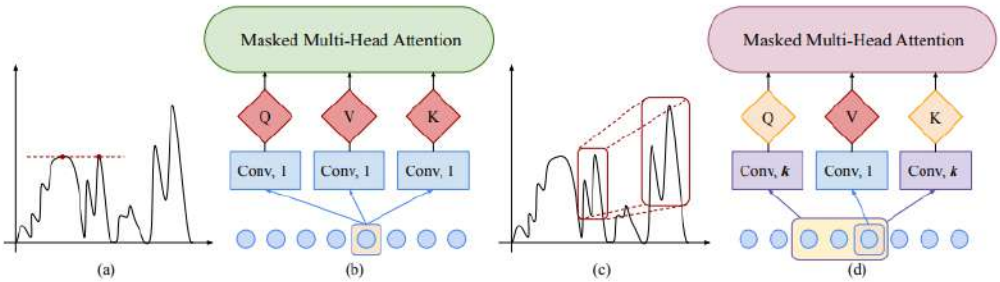


Figure 1: The comparison between canonical and our convolutional self-attention layers. "Conv, 1" and "Conv, k" mean convolution of kernel size $\{1, k\}$ with stride 1, respectively. Canonical self-attention as used in Transformer is shown in (b), may wrongly match point-wise inputs as shown in (a). Convolutional self-attention is shown in (d), which uses convolutional layers of kernel size k with stride 1 to transform inputs (with proper paddings) into queries/keys. Such locality awareness can correctly match the most relevant features based on shape matching in (c).

GLR local global ts representations (https://arxiv.org/pdf/2202.02262.pdf)

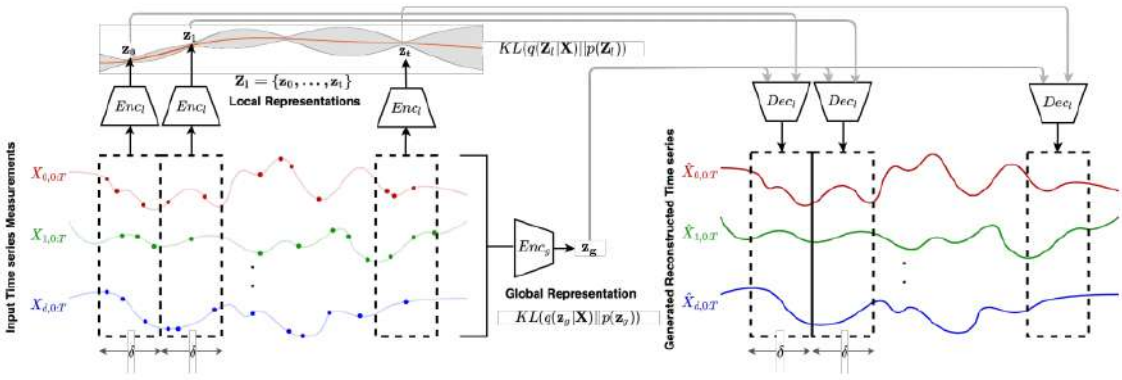


Figure 1: Overview of our method for learning global and local representations. The proposed method learns the distribution of the local representations of windows of samples over time using the local encoder Enc_l. The global encoder Enc_g models the posterior of the global representation, and the decoder Dec generates the time series using samples from the posterior of the local and global representations.

TACTiS (https://arxiv.org/pdf/2202.03528.pdf)

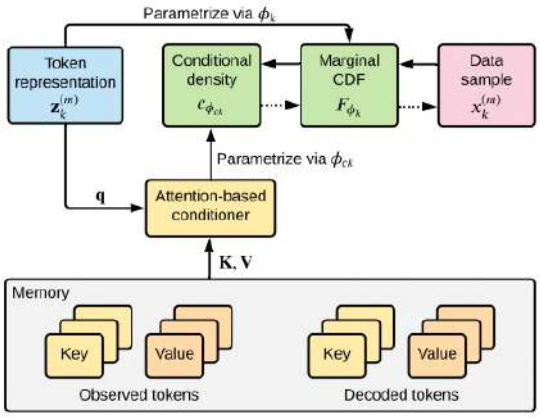


Figure 2: Overview of the TACTiS decoder architecture. Dotted arrows indicate the flow of information during sampling.

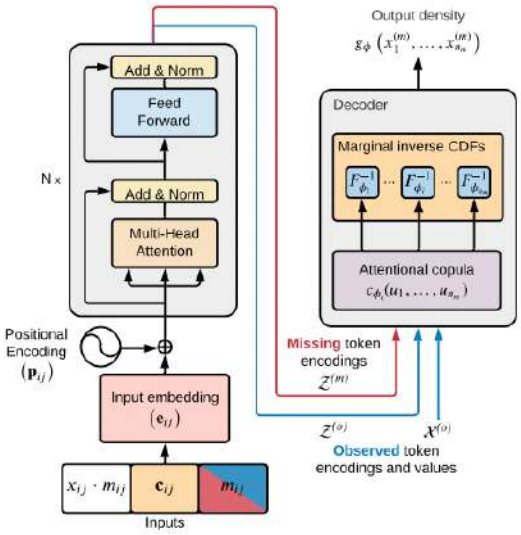


Figure 1: Model overview. **(Left)** The TACTiS encoder is very similar to that of standard transformers. The key difference is that both observed and missing tokens are encoded simultaneously. **(Right)** The decoder, based on an attentional copula, learns the output density given representations of the observed and missing tokens.

MQTransformer (https://arxiv.org/pdf/2009.14799.pdf)

ProTran (https://proceedings.neurips.cc/paper/2021/file/c68bd9055776bf38d8fc43c0ed283Paper.pdf)

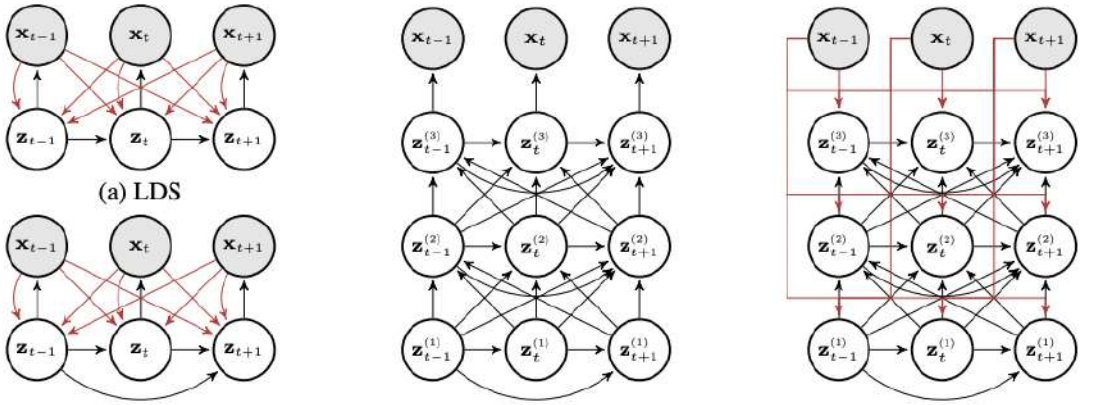


Figure 1: Graphical model representations of linear dynamical systems (LDSs) in (a), and our proposed models (ProTran) in (b), (c), and (d). Black arrows denote the generative mechanism and red arrows the inference procedure. The separation of generation and inference in (c) and (d) is for readability. While traditional SSMs such as LDSs are limited to Markovian dynamics and linear dependencies, our models allow for non-Markovian and non-linear interactions between time steps via attention mechanism. A multi-layer extension of our models further increases expressiveness without compromising the tractable inference procedure.

Preformer (https://arxiv.org/pdf/2202.11356.pdf)

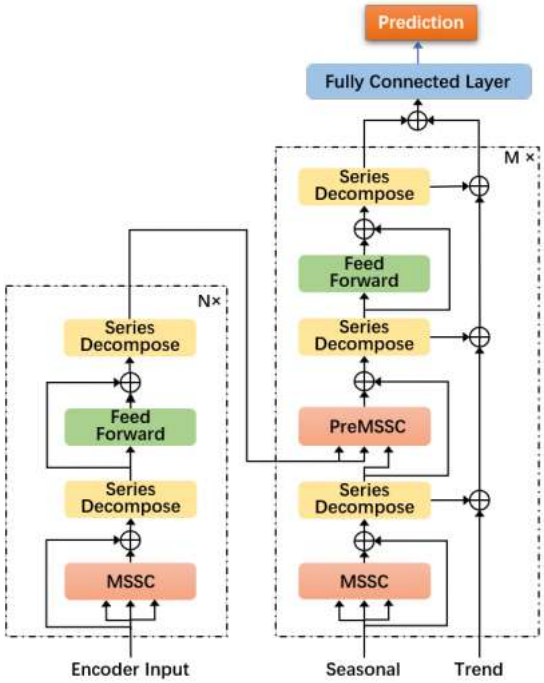


Figure 2. The overall framework of the Preformer model.

Spacetimeformer (https://arxiv.org/pdf/2109.12218.pdf)

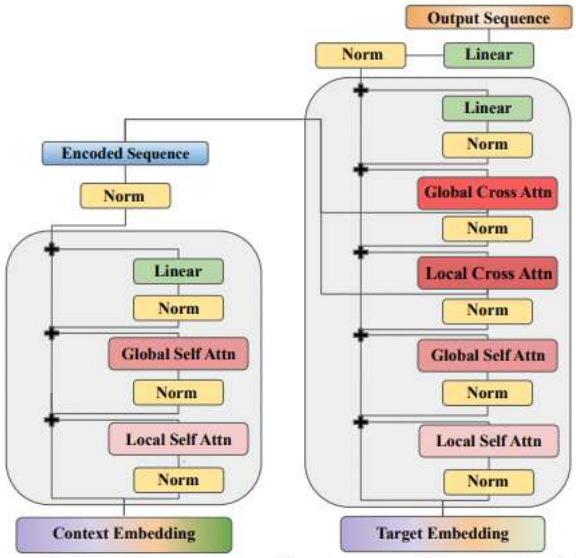


Figure 3: The `Spacetimeformer` architecture for joint spatiotemporal sequence learning.

Differential Attention Fusion (https://arxiv.org/pdf/2202.11402.pdf)

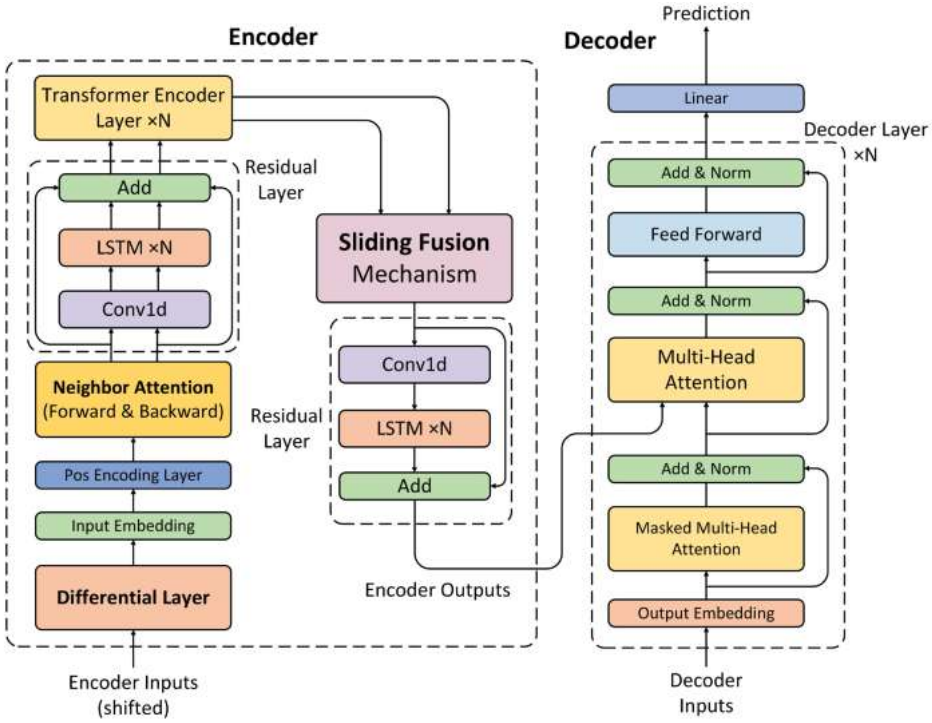


Fig. 1. The architecture of the proposed model

Block-Recurrent Transformers (https://arxiv.org/pdf/2203.07852.pdf)

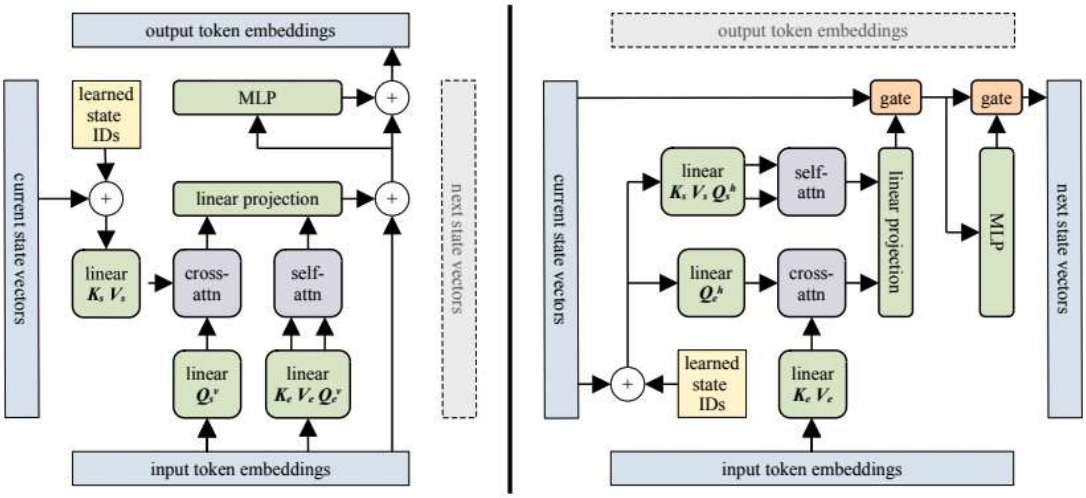


Figure 1: Illustration of our recurrent cell. The left side depicts the vertical direction (layers stacked in the usual way) and the right side depicts the horizontal direction (recurrence). Notice that the horizontal direction merely rotates a conventional transformer layer by 90°, and replaces the residual connections with gates.

Memorizing Transformers (https://arxiv.org/pdf/2203.08913.pdf)

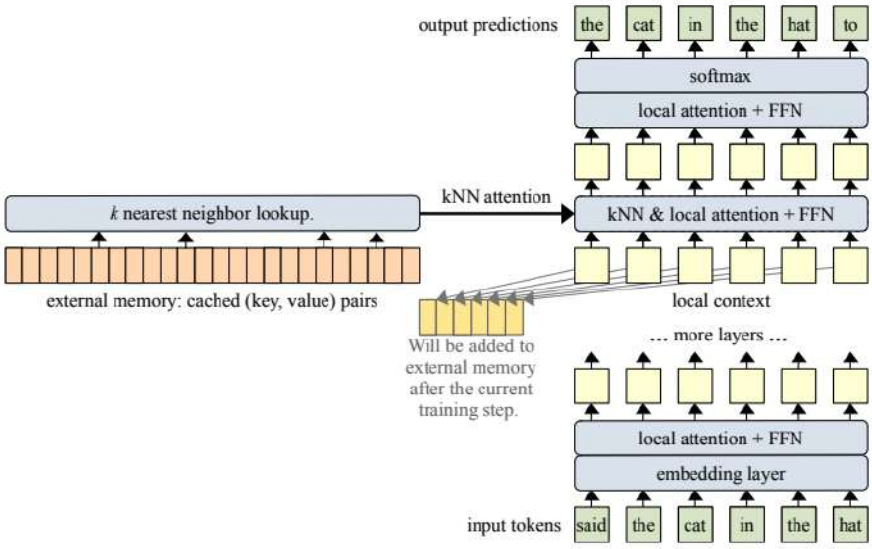
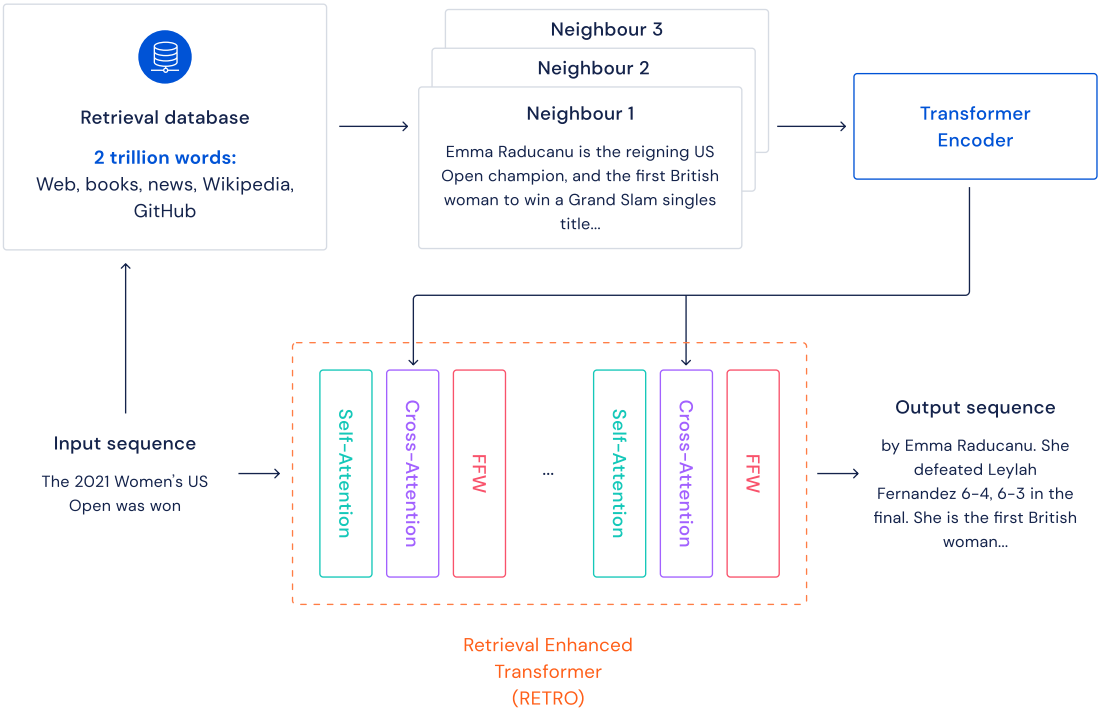


Figure 2: We extend Transformers with access to (key, value) pairs of previously seen subsequences.

Retrieval-Enhanced Transformer (RETRO) (https://arxiv.org/pdf/2112.04426.pdf)



Linformer (https://arxiv.org/pdf/2006.04768.pdf)

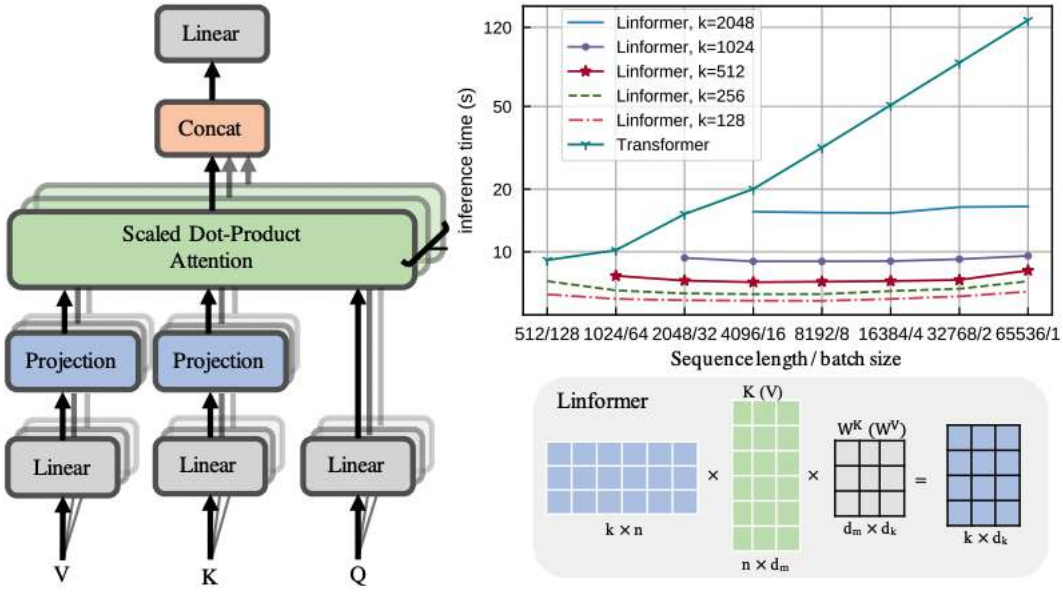
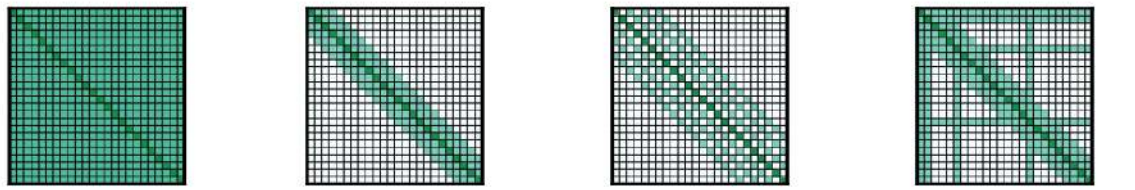


Figure 2: Left and bottom-right show architecture and example of our proposed multihead linear self-attention. Top right shows inference time vs. sequence length for various Linformer models.

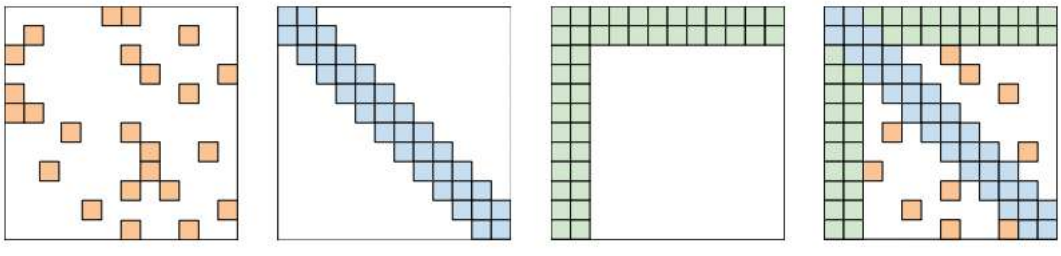
Longformer(https://arxiv.org/pdf/2004.05150.pdf)



(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

Figure 2: Comparing the full self-attention pattern and the configuration of attention patterns in our Longformer.

BigBird (https://arxiv.org/pdf/2007.14062.pdf)



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

Figure 1: Building blocks of the attention mechanism used in BIGBIRD. White color indicates absence of attention. (a) random attention with $r = 2$, (b) sliding window attention with $w = 3$ (c) global attention with $g = 2$. (d) the combined BIGBIRD model.

Nyströmformer (https://arxiv.org/pdf/2102.03902.pdf)

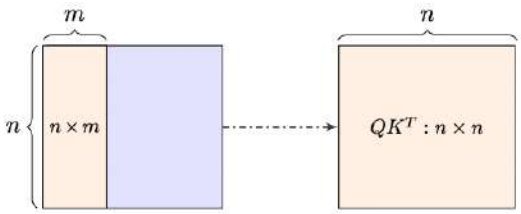


Figure 1: A key challenge of Nyström approximation. The orange block on the left shows a $n \times m$ sub-matrix of S used by Nyström matrix approximation in (4). Computing the sub-matrix, however, requires all entries in the $n \times n$ matrix before the softmax function (QK^T). Therefore, a direct application of Nyström approximation is problematic.

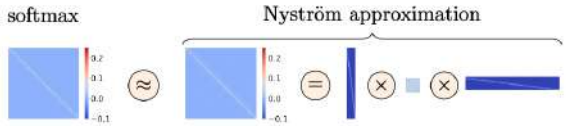


Figure 2: Illustration of a Nyström approximation of softmax matrix in self-attention. The left image shows the true softmax matrix used in self-attention and the right images show its Nyström approximation. Our approximation is computed via multiplication of three matrices.

Performers (https://arxiv.org/pdf/2009.14794v1.pdf)

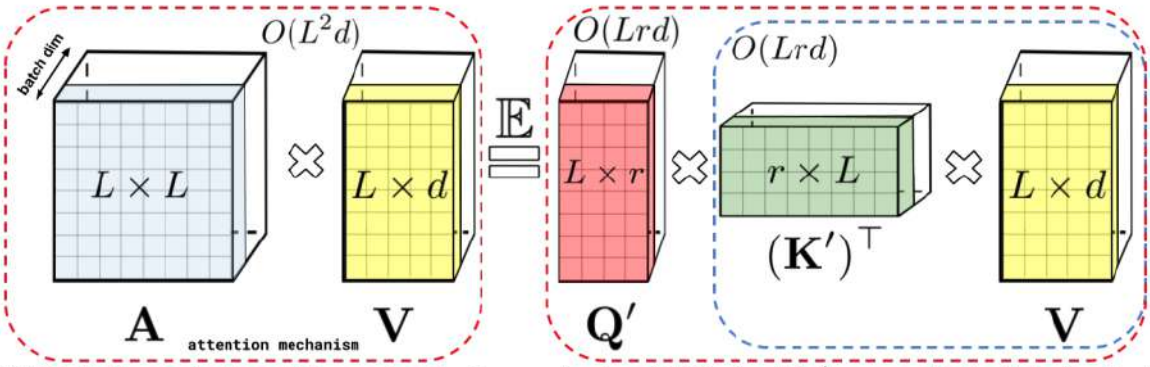


Figure 1: Approximation of the regular attention mechanism **AV** (before $\mathbf{D}^{-1}$-renormalization) via (random) feature maps. Dashed-blocks indicate order of computation with corresponding time complexities attached.

ConvTTLSTM (https://arxiv.org/pdf/2002.09131.pdf)

Axial Transformers (https://arxiv.org/pdf/1912.12180.pdf)

Cost (https://openreview.net/pdf?id=PilZY3omXV2)

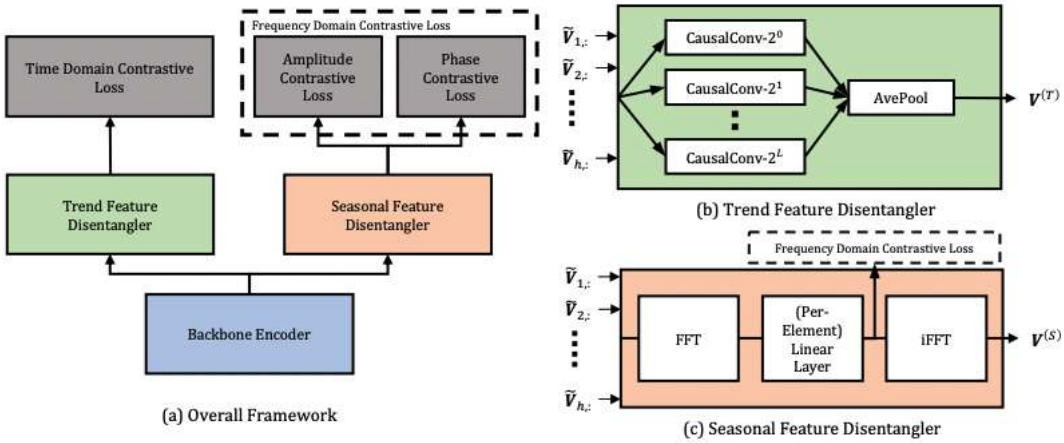
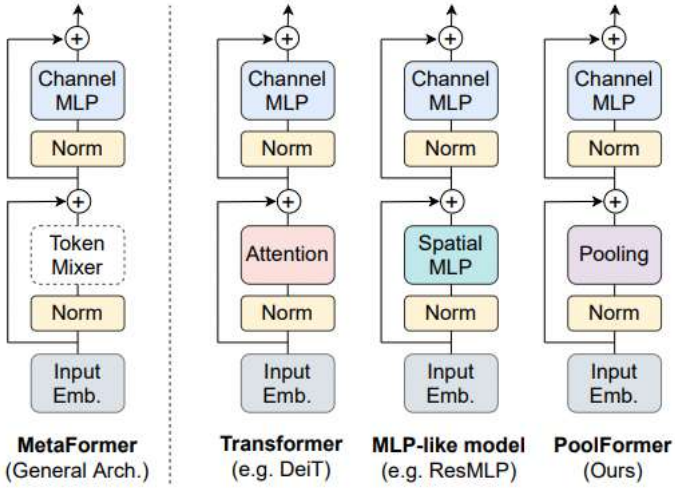


Figure 3: (a) Overall Framework. Given intermediate representations from the backbone encoder, $\tilde{\boldsymbol{V}} = f_b(\boldsymbol{X})$, the TFD and SFD produce the trend features, $\boldsymbol{V}^{(T)} = f_T(\tilde{\boldsymbol{V}})$, and seasonal features, $\boldsymbol{V}^{(S)} = f_S(\tilde{\boldsymbol{V}})$, respectively. (b) Trend Feature Disentangler. Composition of a mixture of auto-regressive experts, instantiated as 1d-causal convolutions with kernel size of $2^i, \forall i = 0, \ldots, L$, where L is a hyper-parameter. Followed by average-pool over the $L+1$ representations. (c) Seasonal Feature Disentangler. After transforming the intermediate representations into frequency domain via the FFT, the SFD applies a (complex-valued) linear layer with unique weights for each frequency. Then, an inverse FFT is performed to map the representations back to time domain, to form the seasonal representations, $\boldsymbol{V}^{(S)}$.

MetaFormer (https://arxiv.org/pdf/2111.11418.pdf)



CKConv (https://arxiv.org/pdf/2102.02611.pdf)

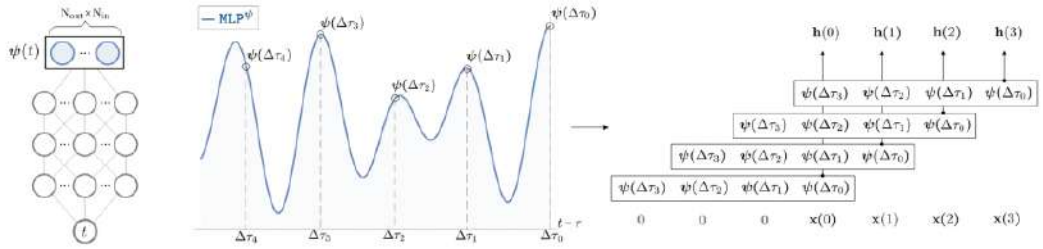


Figure 1: Continuous Kernel Convolution (CKConv). CKConv views a convolutional kernel as a vector-valued continuous function $\boldsymbol{\psi} : \mathbb{R} \rightarrow \mathbb{R}^{\mathrm{N_{out}} \times \mathrm{N_{in}}}$ parameterized by a small neural network $\texttt{MLP}^{\boldsymbol{\psi}}$. $\texttt{MLP}^{\boldsymbol{\psi}}$ receives a time-step and outputs the value of the convolutional kernel at that position. We sample convolutional kernels by passing a set of relative positions $\{\Delta\tau_i\}$ to $\texttt{MLP}^{\boldsymbol{\psi}}$, and perform convolution with the sampled kernel next. Since $\texttt{MLP}^{\boldsymbol{\psi}}$ is a continuous function, CKConvs can (*i*) construct arbitrarily large kernels, (*ii*) generate kernels at different resolutions, and (*iii*) handle irregular data.

UnICORNN (https://arxiv.org/pdf/2103.05487.pdf)

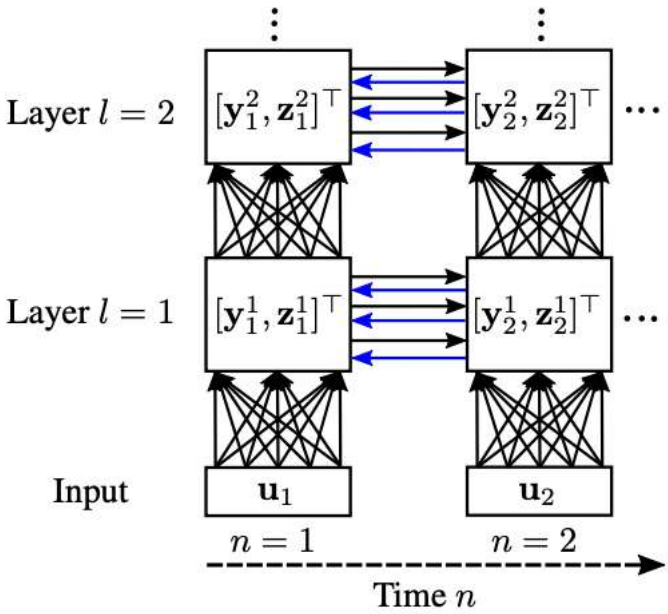


Figure 1. Schematic diagram of the multi-layer UnICORNN architecture, where the layers (respectively the input) are densely connected and the hidden states evolve independently in time. The invertibility of UnICORNN is visualized with blue arrows, emphasizing that the hidden states can be reconstructed during the backward pass and do not need to be stored.

Switch Transformers (https://arxiv.org/pdf/2101.03961.pdf)

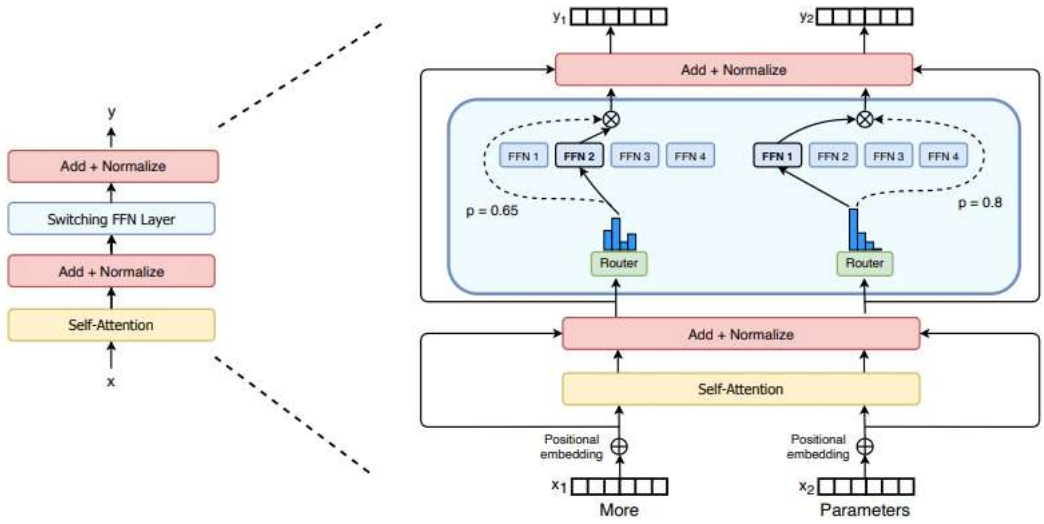


Figure 2: Illustration of a Switch Transformer encoder block. We replace the dense feed forward network (FFN) layer present in the Transformer with a sparse Switch FFN layer (light blue). The layer operates independently on the tokens in the sequence. We diagram two tokens (x_1 = "More" and x_2 = "Parameters" below) being routed (solid lines) across four FFN experts, where the router independently routes each token. The switch FFN layer returns the output of the selected FFN multiplied by the router gate value (dotted-line).

Non-stationary Transformer (https://arxiv.org/pdf/2205.14415.pdf)

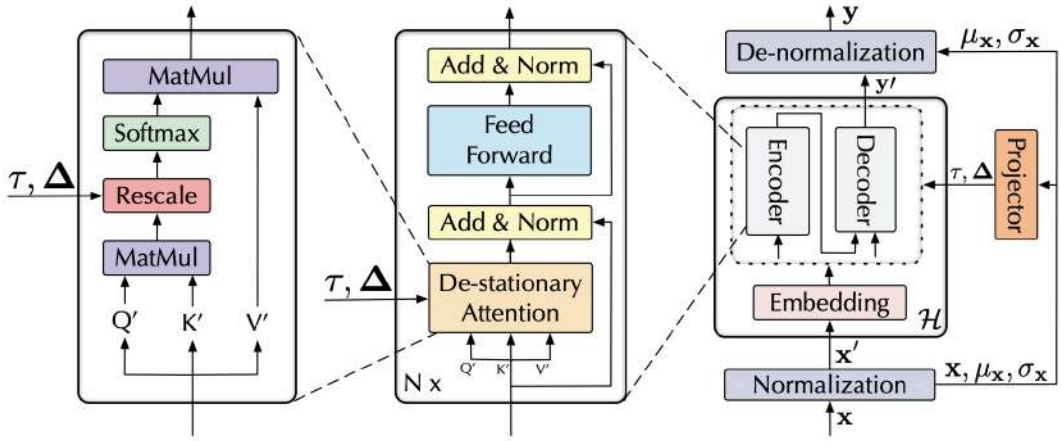


Figure 2: Non-stationary Transformers. Series Stationarization is adopted as a wrapper on the base model to normalize each incoming series and de-normalize the output. De-stationary Attention replaces the original Attention mechanism to approximate attention learned from unstationarized series, which rescales current temporal dependency weights with learned de-stationary factors τ, Δ.

FEDformer (https://arxiv.org/pdf/2201.12740.pdf)

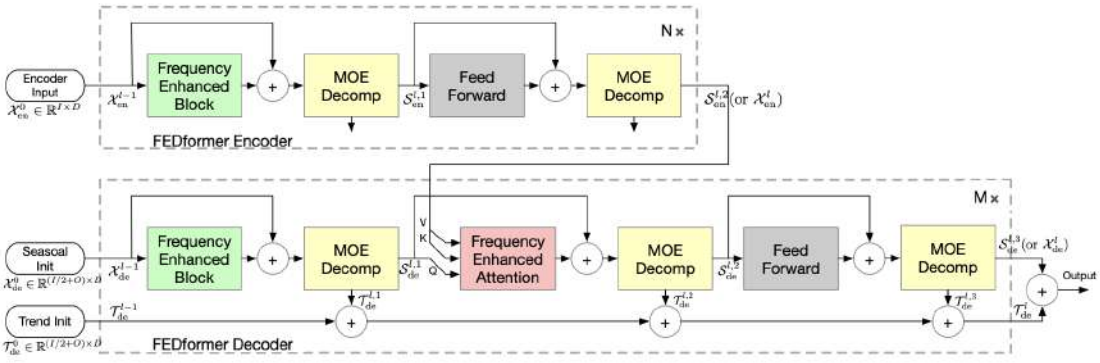
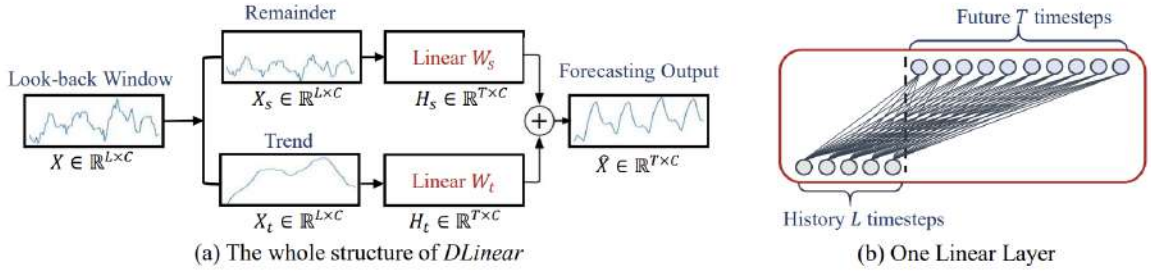


Figure 2. FEDformer Structure. The FEDformer consists of N encoders and M decoders. The Frequency Enhanced Block (FEB, green blocks) and Frequency Enhanced Attention (FEA, red blocks) are used to perform representation learning in frequency domain. Either FEB or FEA has two subversions (FEB-f & FEB-w or FEA-f & FEA-w), where '-f' means using Fourier basis and '-w' means using Wavelet basis. The Mixture Of Expert Decomposition Blocks (MOEDecomp, yellow blocks) are used to extract seasonal-trend patterns from the input data.

DLinear (https://arxiv.org/pdf/2205.13504.pdf)



(a) The whole structure of *DLinear* (b) One Linear Layer

Figure 2: Illustration of the Decomposition Linear Model.

Scaleformer (https://arxiv.org/pdf/2206.04038.pdf)

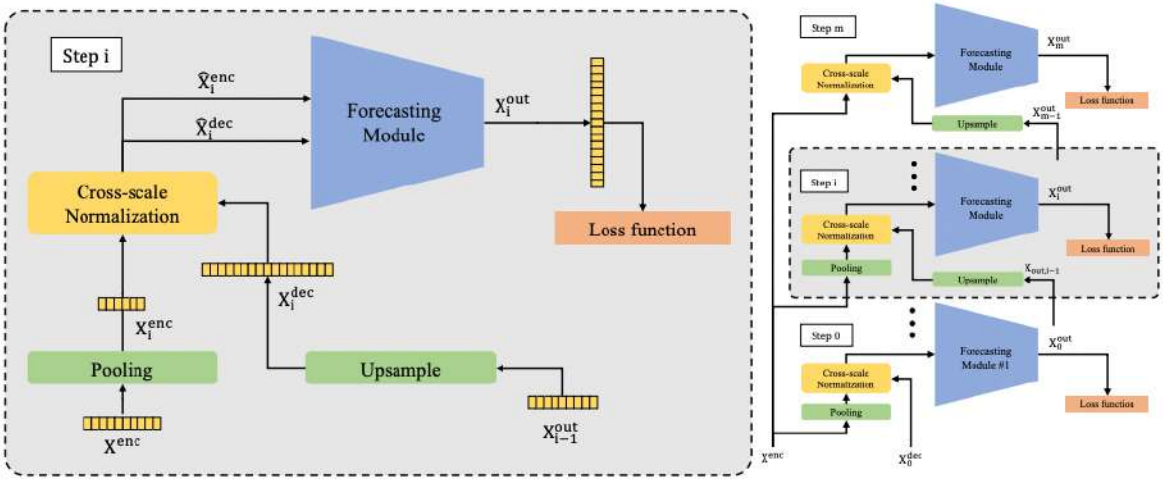


Figure 2: Overview of the proposed framework. (Left) Representation of a single scaling block. In each step, we pass the normalized upsampled version of the output from previous step along with the normalized downsampled version of encoder as the input. (Right) Representation of the full architecture. We process the input in a multi-scale manner iteratively from the smallest scale to the original scale.

Probabilistic Transformer (https://arxiv.org/pdf/2205.13927.pdf)

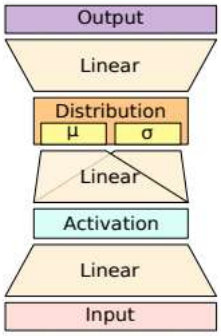
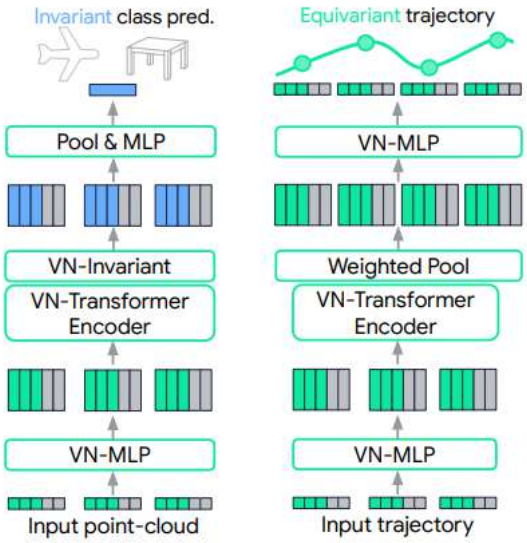


Figure 1: Probabilistic Feed-Forward layer.

VN-Transformer (https://arxiv.org/pdf/2206.04176.pdf)



(a) Rotation-invariant classification model.

(b) Rotation-equivariant trajectory forecasting model.

Figure 1: VN-Transformer ("early fusion") models. Legend: (■) SO(3)-equivariant features; (■) SO(3)-invariant features; (■) Non-spatial features.

EfficientFormer (https://arxiv.org/pdf/2206.01191.pdf)

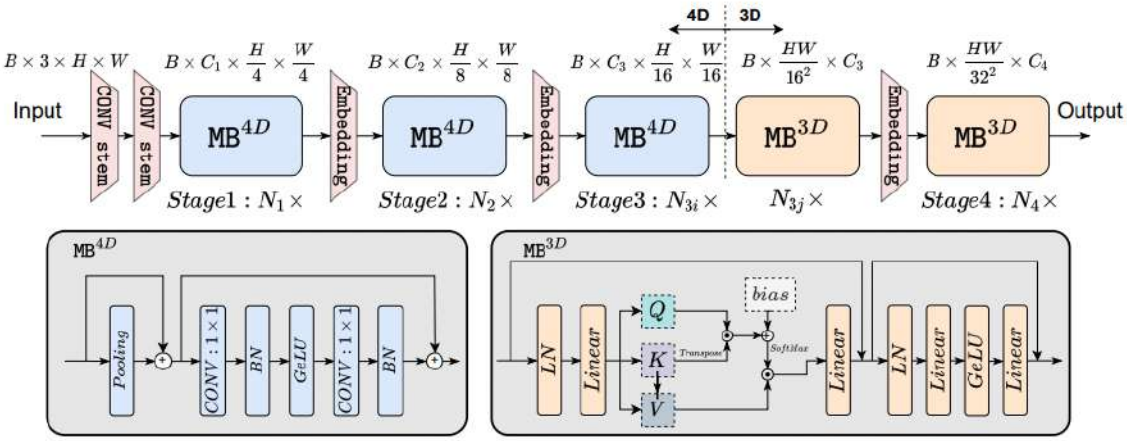


Figure 3: **Overview of EfficientFormer.** The network starts with a convolution stem as patch embedding, followed by MetaBlock (MB). The $\texttt{MB}^{4D}$ and $\texttt{MB}^{3D}$ contain different layer configurations with the token mixer, *i.e.*, pooling and multi-head self-attention, arranged in a dimension-consistent manner.

RoFormer (https://arxiv.org/pdf/2104.09864.pdf)

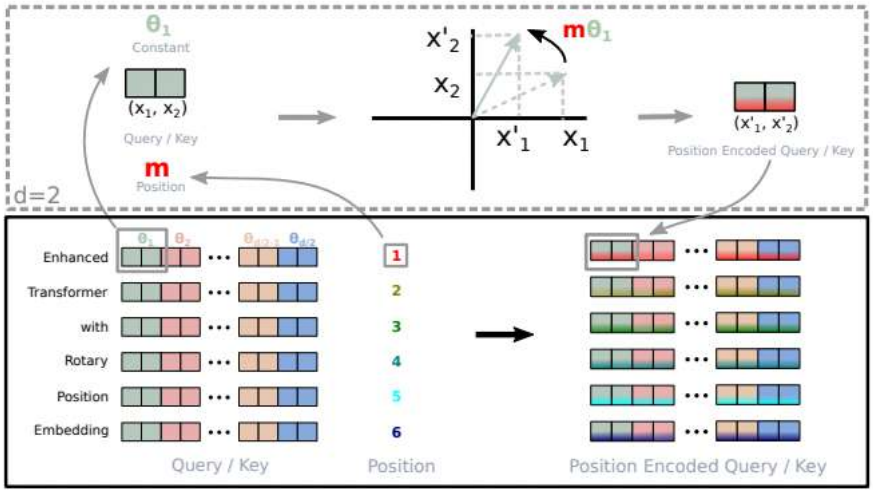


Figure 1: Implementation of Rotary Position Embedding(RoPE).

Perceiver AR (https://arxiv.org/pdf/2202.07765.pdf)

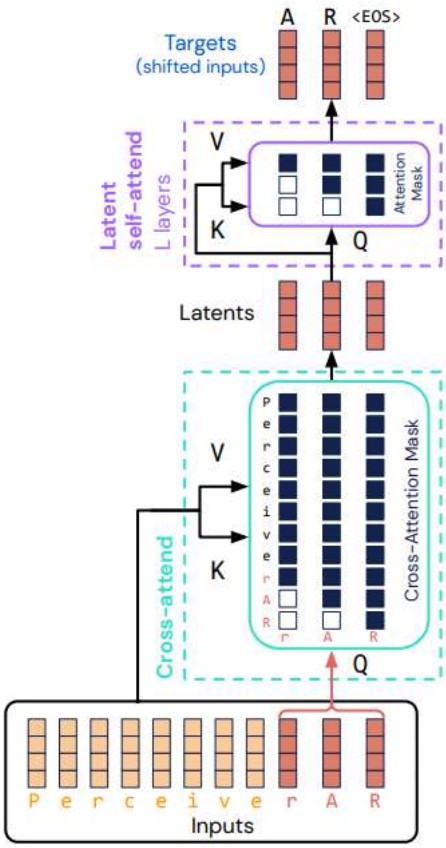


Figure 1. Perceiver AR maps inputs ($X \in \mathcal{R}^{M \times C}$; $M = 11$ shown) to a small latent ($Z \in \mathcal{R}^{N \times C}$; $N = 3$ shown) by cross-attention, querying with the N most recent inputs to produce one latent for each target. Latents subsequently interact via a deep stack of L self-attention layers to produce estimates for each target. Causal masking is used in both cross- and self-attention to maintain end-to-end autoregressive ordering.

FastFormer (https://arxiv.org/pdf/2108.09084.pdf)

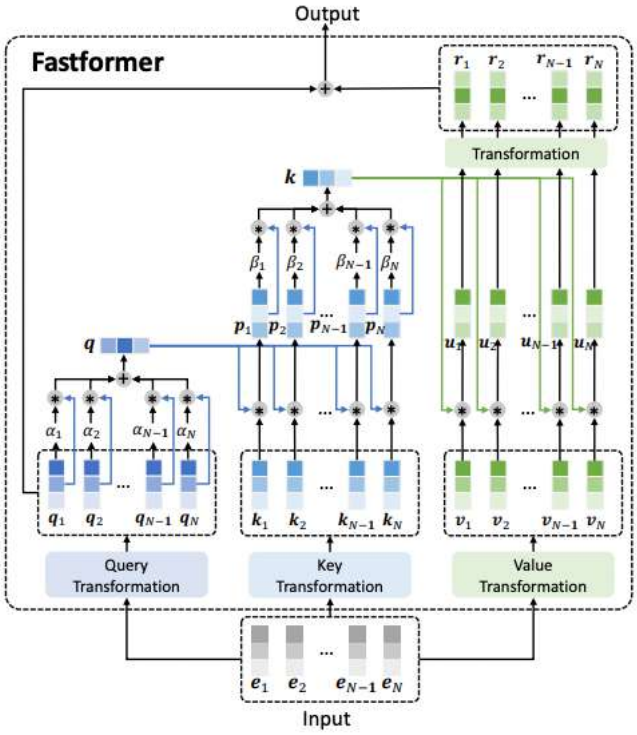
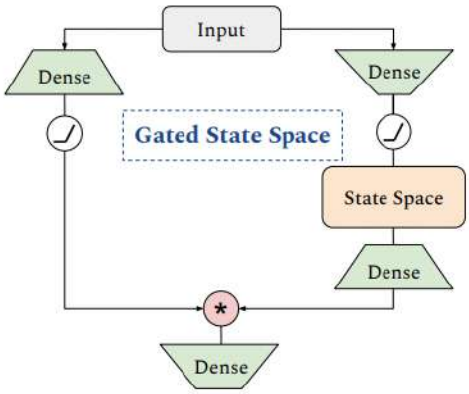


Figure 1: The architecture of *Fastformer*.

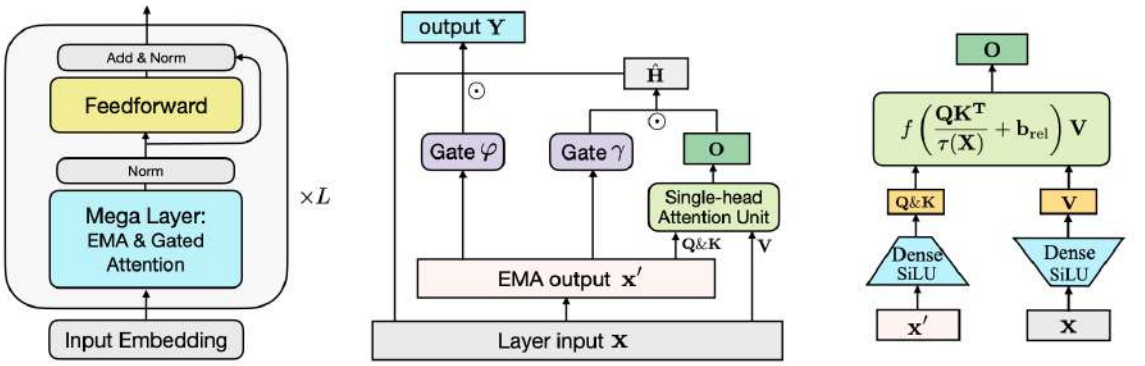
Gated State-Space (https://arxiv.org/pdf/2206.13947.pdf)




```
def gss(x, F=4096, L=4096, E=1024, H=256):
    shortcut, x = x, norm(x)
    v = dense(x, F, activation='gelu')
    u = dense(x, H, activation='gelu')
    y = dss(u, H, L)
    # yh1,..,yhL are linear in uh1,..,uhL
    uc = dense(y, F)
    o = dense(uc * v, E)
    return o + shortcut
```

Figure 1: (a) Our proposed Gated State Space (GSS) layer, (b) Pseudocode for GSS (full implementation in §A.2).

Mega (https://arxiv.org/pdf/2209.10655.pdf)



(a) Mega architecture. (b) Mage layer. (c) Single-head attention unit.

Figure 2: MEGA – model architecture. Figure (a) shows the overall architecture of each MEGA block. Figure (b) illustrates the gated attention sub-layer equipped with EMA, while Figure (c) displays the details of a single-head attention unit.

S4ND (https://arxiv.org/pdf/2210.06583.pdf)

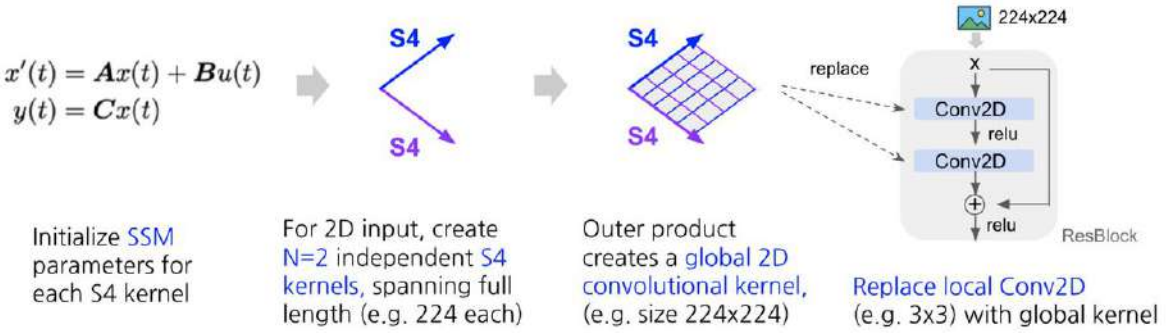


Figure 2: **(Flowchart of S4ND for images: 2D example.)** S4ND can process images as 2D inputs by initializing an SSM per spatial dimension x and y of the input. Two independent S4 kernels are then instantiated that span the entire input lengths of each dimension (e.g., 224 as shown above). Computing an outer product of the two 1D kernels produces a global convolutional kernel (e.g., 224x224). This global kernel can replace standard local Conv2D layers where ever they are used, such as ResNet or ConvNeXt blocks. A similar procedure can be done in 3D (with 3 S4 kernels) to create 3D global kernels for videos.

SGConv (https://arxiv.org/pdf/2210.09298.pdf)

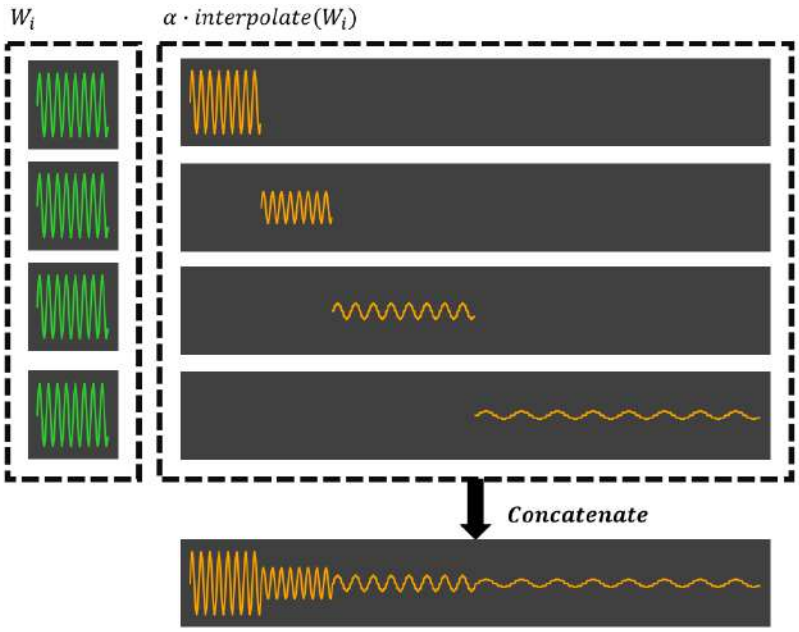


Figure 1: Illustration of the parameterization used in `SGConv` (Eq. (1)). The convolution kernel is composed of multi-scale sub-kernels. **Parameterization Efficiency.** Every larger sub-kernel doubles the size of the previous sub-kernel while the same number of parameters are used for every scale, ensuring a logarithmic dependency of the number of parameters to the input length. **Decaying.** We use a weighted combination of sub-kernels where the weights are decaying, and smaller weights are assigned to larger scales.

Non-stationary Transformers (https://arxiv.org/pdf/2205.14415.pdf)

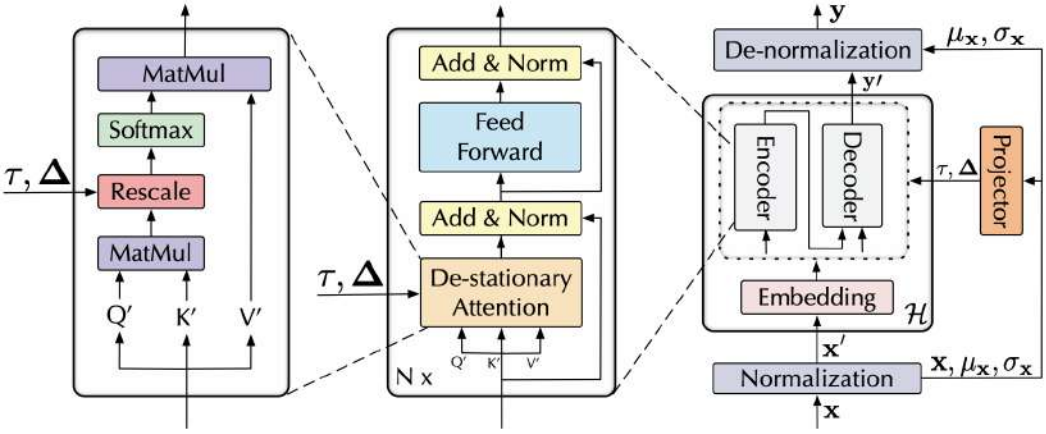
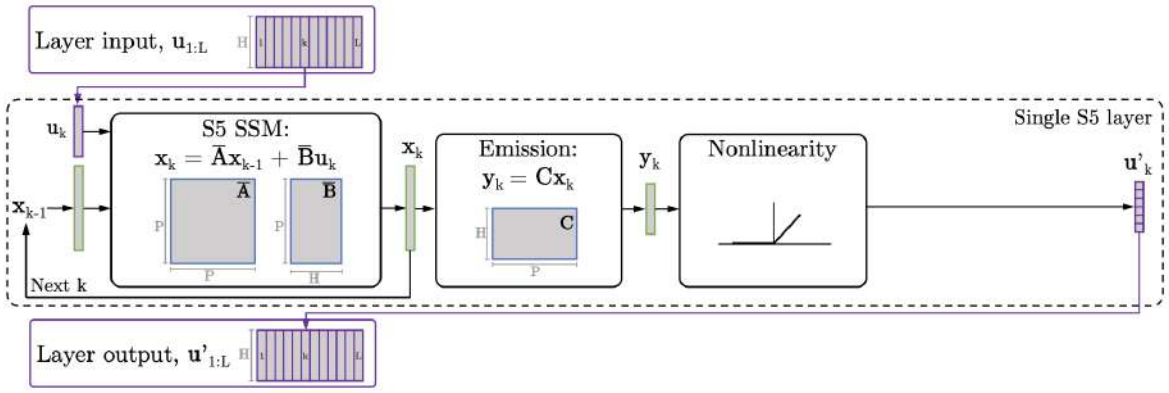


Figure 2: Non-stationary Transformers. Series Stationarization is adopted as a wrapper on the base model to normalize each incoming series and de-normalize the output. De-stationary Attention replaces the original Attention mechanism to approximate attention learned from unstationarized series, which rescales current temporal dependency weights with learned de-stationary factors $\tau, \boldsymbol{\Delta}$.

S5 (https://arxiv.org/pdf/2208.04933.pdf)



(b) Internal structure of a single S5 layer.

Figure 2: Schematic of the internal structure of a discretized S4 layer [17] (top) and S5 layer (bottom). Note $\mathbf{D}$ is omitted for simplicity. We view An S4 layer as a single block-diagonal SSM with a latent state of size HN, followed by a nonlinearity and mixing layer to mix the independent features. (b) In contrast, the S5 layer uses a dense, MIMO linear SSM with latent size $P \ll HN$.

Temporal Latent Bottleneck (https://openreview.net/pdf?id=mq-8p5pUnEX)

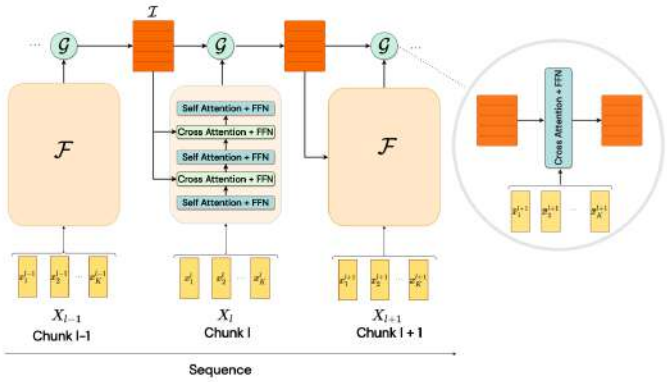


Figure 1: **Perceptual module + Temporal Latent Bottleneck Model**. $\mathcal{F}$ denotes the perceptual module or the fast stream which is a Transformer. $\mathcal{I}$ represents the temporal latent bottleneck state (consisting of a set of vectors) that are updated using a recurrent function denoted by $\mathcal{G}$. The given sequence is first divided into chunks of size K and each chunk X_l is processed by $\mathcal{F}$ which consists of interleaved SELF ATTENTION + FFN (denoted in blue) and CROSS ATTENTION + FFN (denoted in green) layers. The CROSS ATTENTION + FFN layers allow the representation of $\mathcal{F}$ to be conditioned on top-down information from $\mathcal{I}$. The representations of the temporal latent bottleneck state is updated using the outputs of $\mathcal{F}$ by a recurrent function $\mathcal{G}$, which consists of a CROSS ATTENTION + FFN layer as shown in the circle.

Encoding Recurrence into Transformers (https://openreview.net/pdf?id=7YfHla7IxBJ)

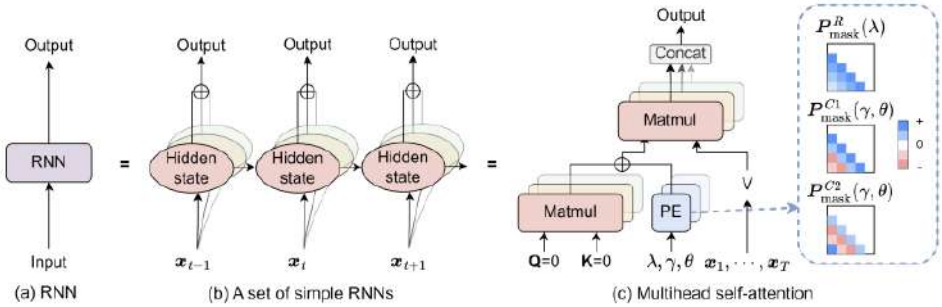


Figure 2: Illustration on how an RNN layer can be equivalently represented by a set of simple RNNs and further by a multihead self-attention.

MTS-Mixers (https://arxiv.org/pdf/2302.04501.pdf)

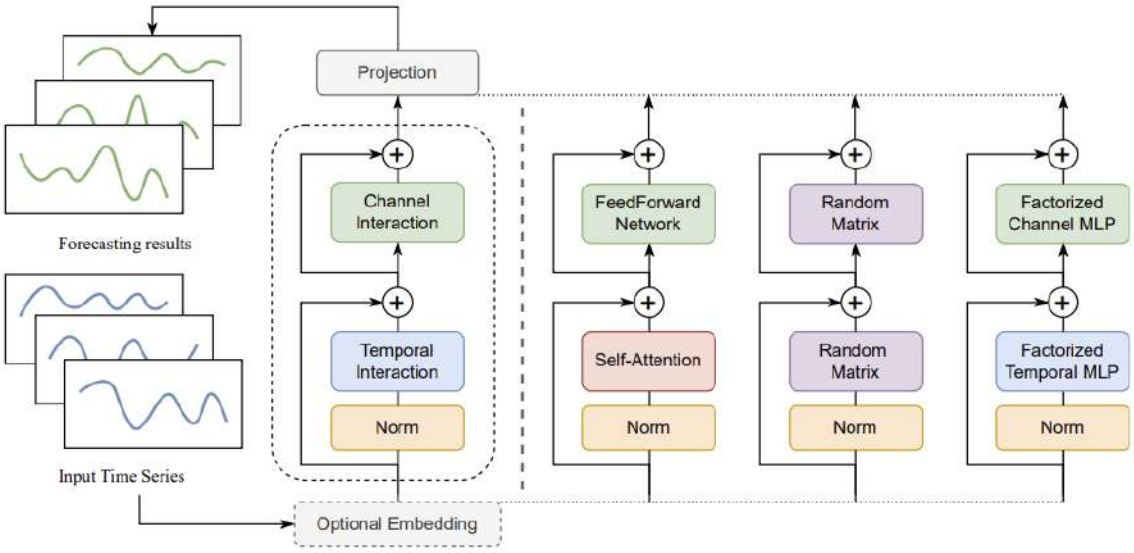


Figure 4. The overall architecture of MTS-Mixers. **Left**: the modules in the dashed box describe the general framework. **Right**: three specific implementations, where we can use attention, random matrix, or factorized MLP to capture dependencies.

TimesNet (https://arxiv.org/pdf/2210.02186.pdf)

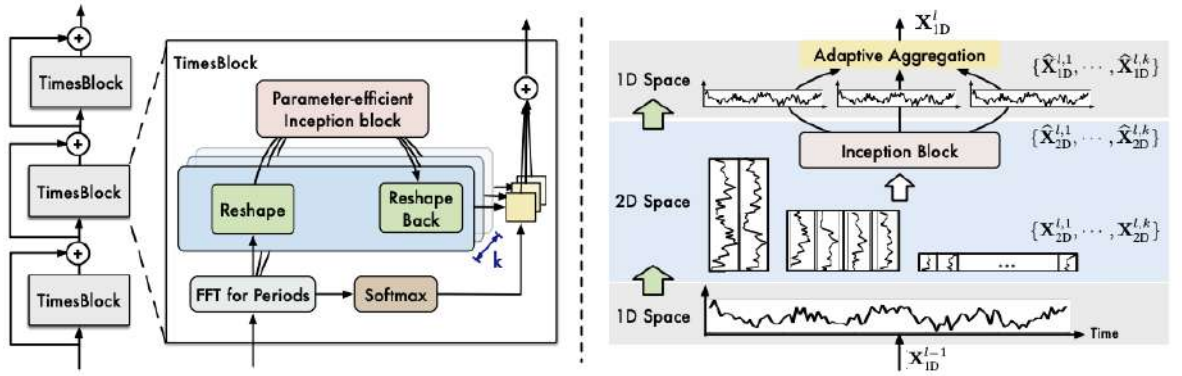
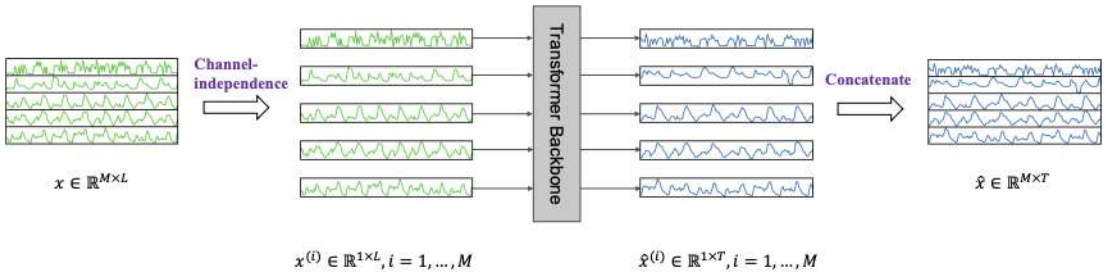
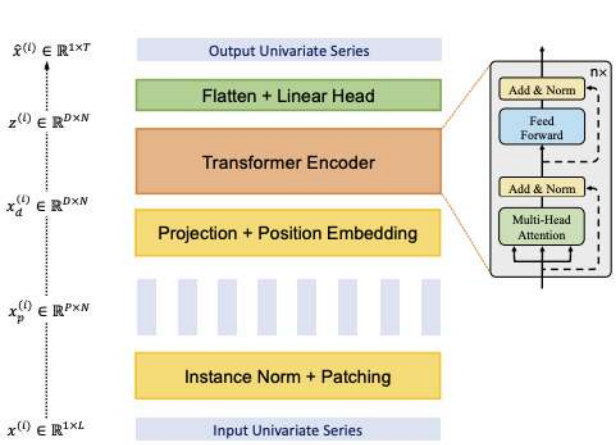


Figure 3: Overall architecture of TimesNet. TimesNet is stacked by TimesBlocks in a residual way. TimesBlocks can capture various temporal 2D-variations from k different reshaped tensors by a parameter-efficient inception block in 2D space and fuse them based on normalized amplitude values.

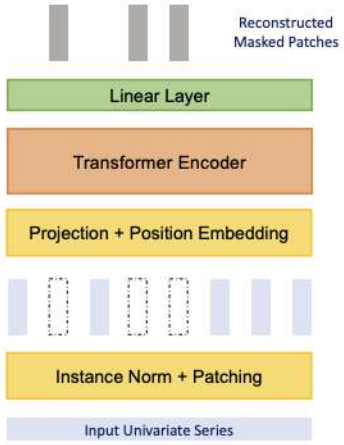
PatchTST (https://arxiv.org/pdf/2211.14730.pdf)



(a) PatchTST Model Overview



(b) Transformer Backbone (Supervised)



(c) Transformer Backbone (Self-supervised)

Figure 1: PatchTST architecture. (a) Multivariate time series data is divided into different channels. They share the same Transformer backbone, but the forward processes are independent. (b) Each channel univariate series is passed through instance normalization operator and segmented into patches. These patches are used as Transformer input tokens. (c) Masked self-supervised representation learning with PatchTST where patches are randomly selected and set to zero. The model will reconstruct the masked patches.

TSMixer (https://arxiv.org/pdf/2303.06053v1.pdf)

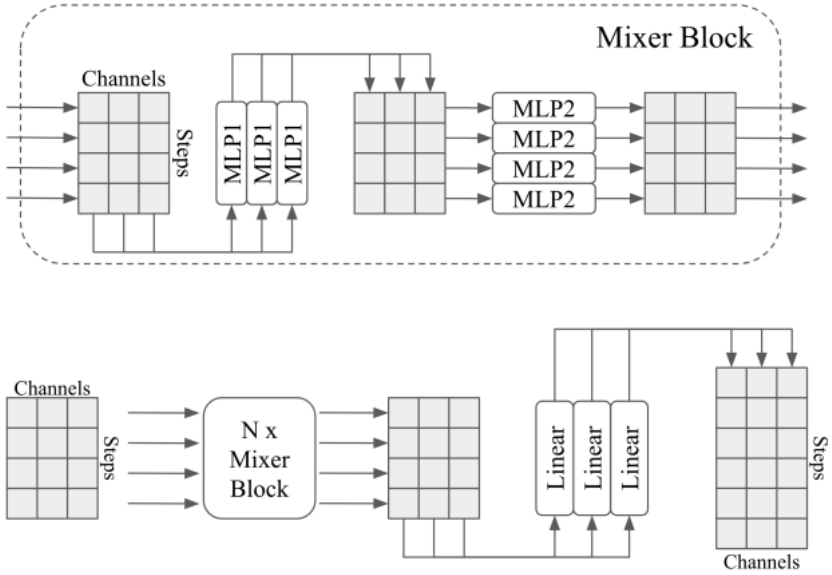


Figure 1. TSMixer for multivariate time series forecasting. TSMixer contains interleaving time-mixing and feature-mixing MLPs to aggregate information. The design balances the use of a long lookback window (as supported by our theoretical analysis on linear models) with limited number of parameters for superior generalization. The extension with auxiliary information is also explored in this paper.

SpaceTime (https://arxiv.org/pdf/2303.09489.pdf)

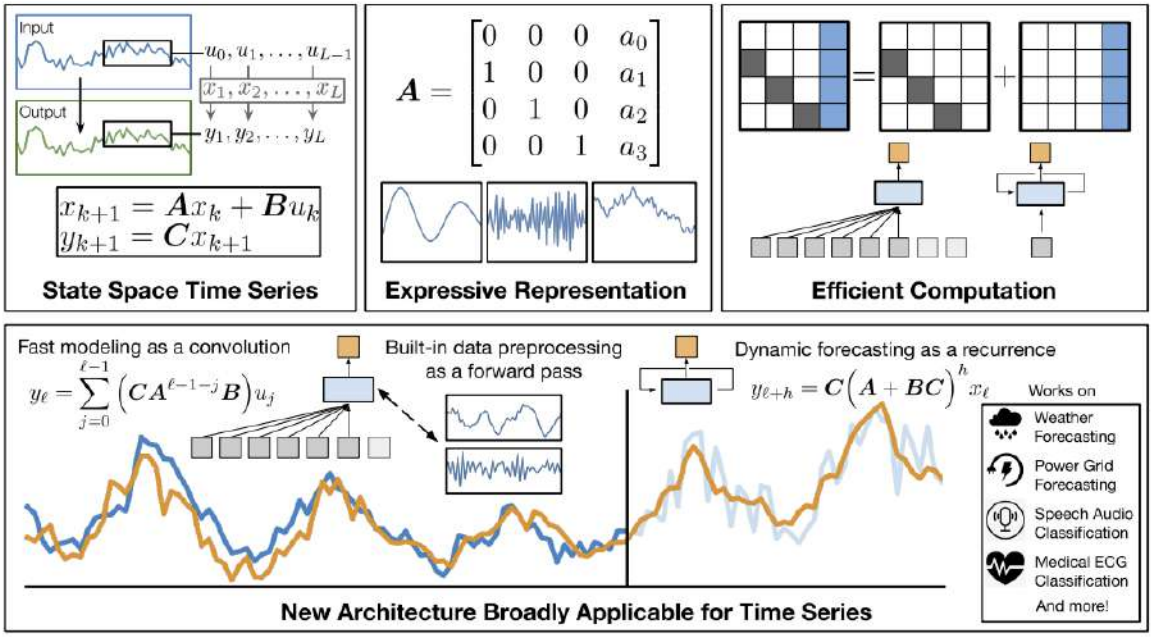


Figure 1: We learn time series processes as state-space models (SSMs) (**top left**). We represent SSMs with the *companion matrix*, which is a highly expressive representation for discrete time series (**top middle**), and compute such SSMs efficiently as convolutions or recurrences via a shift + low-rank decomposition (**top right**). We use these SSMs to build SpaceTime, a new time series architecture broadly effective across tasks and domains (**bottom**).

Conformal Prediction for Time Series with Modern Hopfield Networks (https://arxiv.org/pdf/2303.12783.pdf)

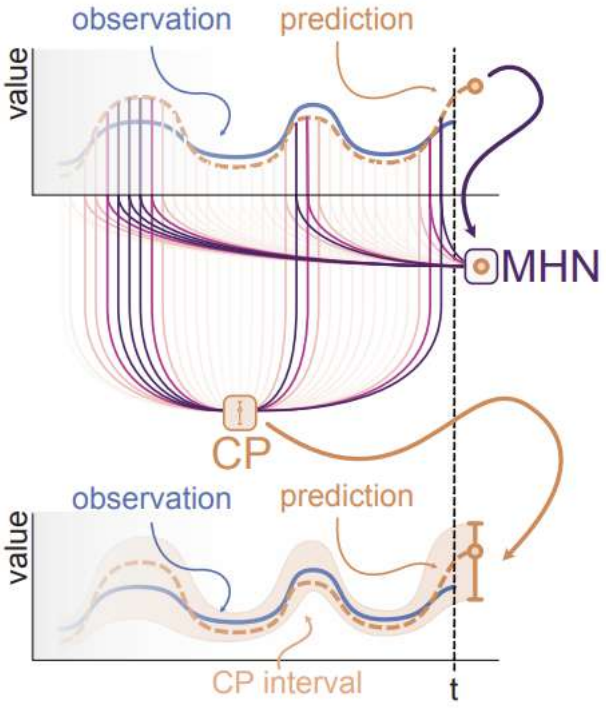


Figure 1. Schematic illustration of HopCPT. The Modern Hopfield Network (MHN) identifies regimes similar to the current one in the time series and weights them (indicated by the colored lines). The weighted information enriches the conformal prediction procedure so that prediction intervals can be derived.