Informer

Overall Network Structure

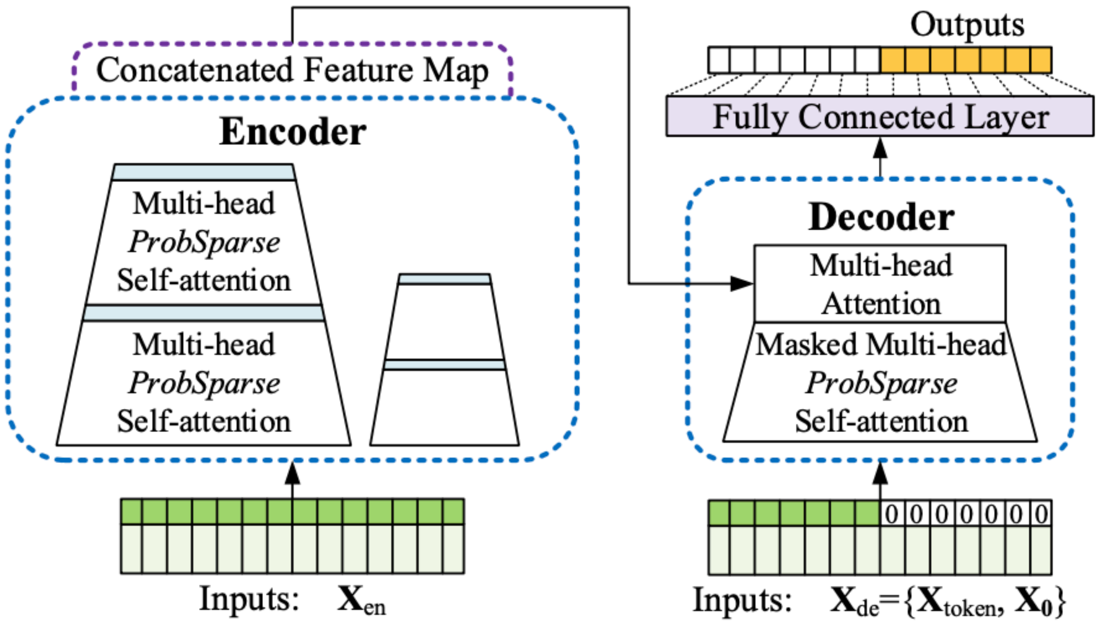


Figure 2: Informer model overview. Left: The encoder receives massive long sequence inputs (green series). We replace canonical self-attention with the proposed *ProbSparse* self-attention. The blue trapezoid is the self-attention distilling operation to extract dominating attention, reducing the network 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.

Inspiration Fact: Query Sparsity

- Long tail distribution of self-attention weights

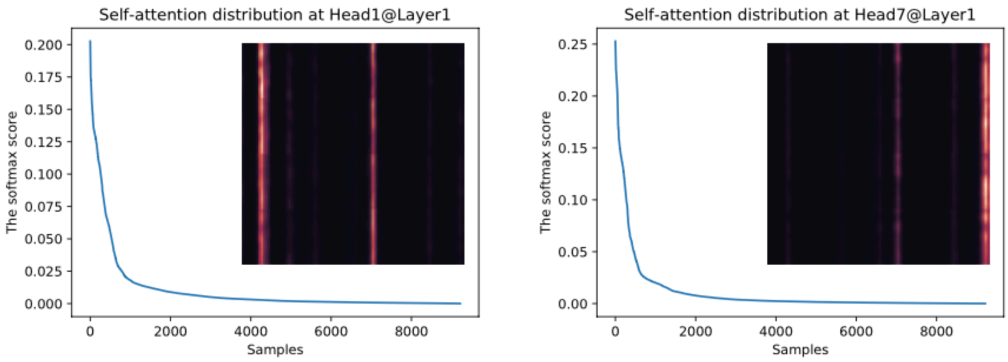
- 

Figure 7: The Softmax scores in the self-attention from a 4-layer canonical Transformer trained on **ETTh**$_1$ dataset.

Scenario: Long Sequence Time-series Forecasting(LSTF)

- horizon > 300
- Problem Setting:

  We first provide the LSTF problem definition. Under the rolling forecasting setting with a fixed size window, we have the input $\mathcal{X}^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_{L_x}^t \mid \mathbf{x}_i^t \in \mathbb{R}^{d_x}\}$ at time t, and the output is to predict corresponding sequence $\mathcal{Y}^t = \{\mathbf{y}_1^t, \ldots, \mathbf{y}_{L_y}^t \mid \mathbf{y}_i^t \in \mathbb{R}^{d_y}\}$. The LSTF problem encourages

Target:

1. reduce attention operation speed

   $O(L^2)->O(L*logL)$
2. reduce memory usage per attention layer

   O(L^2) -> O(LlogL)$

Improvements

1. ProbSparse Self-attention

1.1 **Query sparse measurements**

- the i_th query's attention on all keys:

$p(k_j|q_i) = \frac{k(q_i,k_j)}{\Sigma_l k(q_i,k_l)}$,

where k = asymmetric exponetial kernel $exp(q_i k_j^T/\sqrt{d})$

- $q(k_j|q_i) = 1/L_k$

- $KL(q||p) = ln\Sigma_{l=1}^{L_k} e^{q_i k_i^T/\sqrt{d}} - \frac{1}{L_k}\Sigma_{l=1}^{L_k} q_i k_i^T/\sqrt{d} - lnL_k$
- the i_th query's sparsity measurements:

  $$\mathbf{M}(q_i, K) = ln\Sigma_{l=1}^{L_k} e^{\frac{q_i k_i^T}{\sqrt{d}}} - \frac{1}{L_k}\Sigma_{l=1}^{L_k} \frac{q_i k_i^T}{\sqrt{d}}$$

1.2 **ProbSparse Self-attention**

$$A(Q, K, V) = Softmax(\frac{\overline{Q}K^T}{\sqrt{d}}V)$$

where Q contains **top-u** queries under **M**

1.3 **Sampling**

- $u = c * lnL_Q$, where c is a constant sampling factor
- to obtain top-u queries in $O(LlnL)$:
  - Approximation: max-mean measurements
    - $\overline{M}(q_i, K) = max_j\{\frac{q_i k_j^T}{\sqrt{d}}\} - \frac{1}{L_k}\Sigma_{l=1}^{L_k} \frac{q_i k_i^T}{\sqrt{d}}$
  - Prove: the range of top-u holds in the boundary relaxation
  - *???* **then -> Under the long tail distribution: randomly sample** $U = L_k lnL_Q$ to calculate $\overline{M}(q_i, K)$ (fill others with zero)

Algorithm 1 ProbSparse self-attention

Require: Tensor $\mathbf{Q} \in \mathbb{R}^{m \times d}$, $\mathbf{K} \in \mathbb{R}^{n \times d}$, $\mathbf{V} \in \mathbb{R}^{n \times d}$

print set hyperparameter c, $u = c \ln m$ and $U = m \ln n$

randomly select U dot-product pairs from $\mathbf{K}$ as $\bar{\mathbf{K}}$

set the sample score $\bar{\mathbf{S}} = \mathbf{Q}\bar{\mathbf{K}}^\top$

compute the measurement $M = \max(\bar{\mathbf{S}}) - \text{mean}(\bar{\mathbf{S}})$ by row

set Top-u queries under M as $\bar{\mathbf{Q}}$

set $\mathbf{S}_1 = \text{softmax}(\bar{\mathbf{Q}}\mathbf{K}^\top/\sqrt{d}) \cdot \mathbf{V}$

set $\mathbf{S}_0 = \text{mean}(\mathbf{V})$

set $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_0\}$ by their original rows accordingly

Ensure: self-attention feature map $\mathbf{S}$.

2. Self-attention Distilling

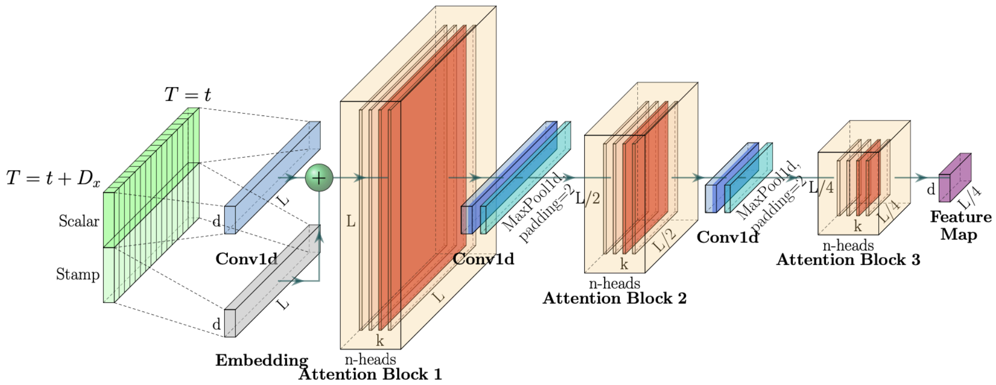


Figure 3: The single stack in Informer's encoder. (1) The horizontal stack stands for an individual one of the encoder replicas in Fig.(2). (2) The presented one is the main stack receiving the whole input sequence. Then the second stack takes half slices of the input, and the subsequent stacks repeat. (3) The red layers are dot-product matrixes, and they get cascade decrease by applying self-attention distilling on each layer. (4) Concatenate all stacks' feature maps as the encoder's output.

$$X^t_{t=j+1} = MaxPool(ELU(Conv1d[X^t_j]_{AB}))$$

reduce memory to $O((2-\epsilon)LlogL)$

3. Generative Decoder

- Start Token:
  - $X^t_{de} = Concat(X^t_{token}, X^t_0) \in \mathbb{R}^{(L_{token}+L_y)\times d_{model}}$
- Pro:
  - Non-autoregressive
  - Only one forward pass
  - Allow inputs with timestamps/offsets
- Limits:
  - Pre-Fixed Length Output Sequence
  - Start Token: previous known labels

Appendix:

Table 7: The Informer network components in details

Encoder:			N
Inputs	1x3 Conv1d	Embedding ($d = 512$)	4
ProbSparse Self-attention Block	Multi-head ProbSparse Attention ($h = 16$, $d = 32$)		
	Add, LayerNorm, Dropout ($p = 0.1$)		
	Pos-wise FFN ($d_{\text{inner}} = 2048$), GELU		
	Add, LayerNorm, Dropout ($p = 0.1$)		
Distilling	1x3 conv1d, ELU		
	Max pooling (stride $= 2$)		
Decoder:			N
Inputs	1x3 Conv1d	Embedding ($d = 512$)	2
Masked PSB	add Mask on Attention Block		
Self-attention Block	Multi-head Attention ($h = 8$, $d = 64$)		
	Add, LayerNorm, Dropout ($p = 0.1$)		
	Pos-wise FFN ($d_{\text{inner}} = 2048$), GELU		
	Add, LayerNorm, Dropout ($p = 0.1$)		
Final:			
Outputs	FCN ($d = d_{\text{out}}$)		