

时序预测系列（二）

如何利用 Transformer-based 模型对价格进行“天气预报”

投资咨询业务资格：

证监许可【2012】669号

报告要点

本次报告分别介绍了 Transformer、Informer、Autoformer 和 FEDformer 四个模型并进行测试，发现 FEDformer 模型利用傅里叶变换将序列从时域转换到频率进行预测能捕捉到全局特征，提高模型预测精度。此外若利用小波变换提取序列特征能捕捉到频率的时间信息，增强 FEDformer 模型对相对较短序列的预测能力。

摘要：

在上一篇报告中，介绍了如何利用 RNN 系列算法对价格进行单元单步预测，本文围绕多元长序列预测展开，尝试实现类似于天气预报的预测效果。RNN 系列算法具有较强**长期依赖性**不适用于长序列预测，因此本文选择具有超强信息提取能力的 Transformer 模型为落脚点展开讨论。然而由于 Transformer 模型存在**过高计算复杂度**这一瓶颈，因此本文从近两年顶会论文中挑选出三个基于 Transformer 的改良模型(Informer、Autoformer 和 FEDformer)分别进行了介绍和测试对比。

为测试模型的有效性和泛化能力，本文在 CU、IF 和 T 三个期货品种的主力合约上进行了测试。在测试中分别利用过去 20/40/60 个交易日的交易数据(开盘价、最低价、最高价、成交量和收盘价)对三个合约接下来 5/10/20/40/60 个交易日的收盘价进行预测。经预测发现，FEDformer 模型的预测结果几乎在所有情况中都是最优的，MSE 和 MAE 最低达到 0.028 和 0.129，价格时序信噪比较低，利用傅里叶变换将价格从时域转换到频域进行预测不仅能更好地捕捉**全局信息**，还能过滤掉信号中的噪音。同时，使用小波增强模块能捕捉到**频率出现的时间信息**，再次加强 FEDformer 模型对短序列的预测能力

此外，我们还发现在输入信息和输出信息较短的情况下，过高计算复杂度对 Transformer 模型的负面影响并不大，在很多测试中，Transformer 的预测精度甚至高于 Autoformer 和 Informer 模型，但在稳定性和普适性上还是逊于 FEDformer 模型。

商品量化组

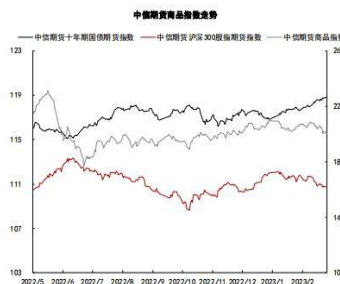
研究员：

蒋可欣 FRM

jiangkexin@citicfs.com

从业资格号 F03098078

投资咨询号 Z0018262



风险提示：本报告中所涉及的算法和模型应用仅为回溯举例，并不构成推荐建议。

目录

摘要:	1
一、引言	4
二、Transformer 的结构和原理	4
(一) 注意力机制	6
(二) Transformer 结构详解	10
三、Informer 的结构和原理	12
四、Autoformer 的结构和原理	15
五、FEDformer 的结构和原理	20
(一) 傅里叶增强结构	20
(二) 小波变换增强结构	22
六、预测结果分析	25
(一) 数据选择和处理	25
(二) 预测结果评价指标	25
(三) 模型参数设置	25
(四) 预测结果展示和分析	26
七、结论与展望	29
参考文献	30
免责声明	31

图表目录

图表 1: Transformer 模型的结构	5
图表 2: 抽象的 Encoder-Decoder 框架	6
图表 3: 引入注意力模型的 Encoder-Decoder 框架	7
图表 4: Attention 计算过程	7
图表 5: 缩放点积注意力模块(Scaled Dot-Product Attention)的计算流程	8
图表 6: 多头注意力机制(Multi-Head Attention)的计算流程	9
图表 7: Informer 模型的结构	13
图表 8: Informer-Encoder 部分侧面视角	14
图表 9: Autoformer 模型的结构	15
图表 10: 自注意力机制 VS 自相关机制	18
图表 11: Autoformer 自相关机制的计算流程	19
图表 12: FEDformer 模型的结构	20
图表 13: FEB-f 结构	21
图表 14: FEA-f 结构	22
图表 15: FEB-w 和 FEA-w 结构	23
图表 16: 前 20 个交易日为输入长度的模型测试结果	26
图表 17: 前 40 个交易日为输入长度的模型测试结果	27
图表 18: 前 60 个交易日为输入长度的模型预测结果	28

图表 19：模型复杂度对比 28

一、引言

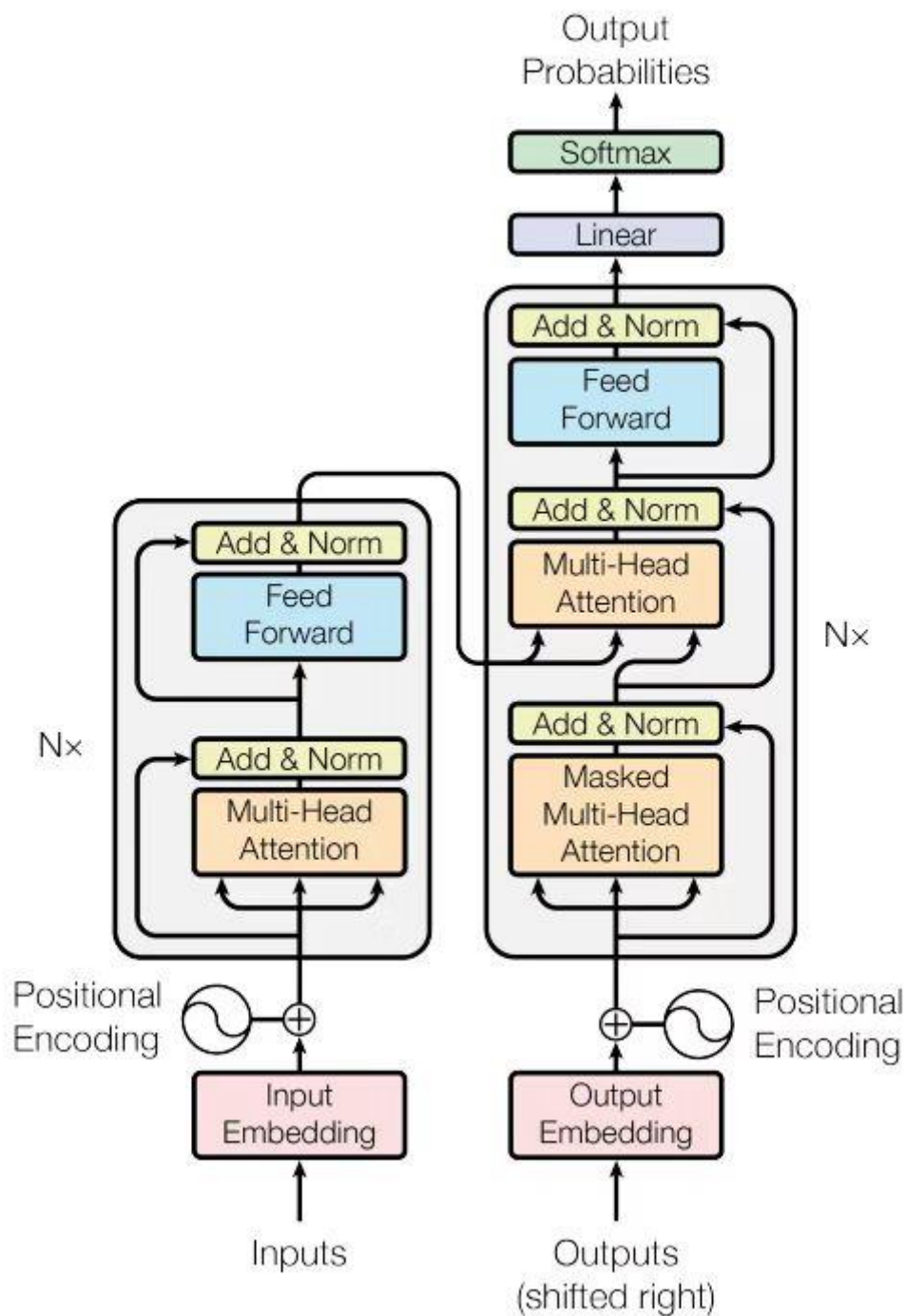
在上篇报告中，我们讨论了短期时序预测的方法，优化后的预测模型在短期预测中取得较好的拟合效果。然而，在现实生活中，短期价格预测对策略帮助有限，需得对未来价格进行长周期预测才能更充分地对行情进行判断。在上篇报告中介绍的 RNN 系列网络使用的是递归结构，因此只能进行单向依次计算，限制了模型的并行能力。同时，RNN 系列网络存在长期依赖问题，随着时间推移，RNN 会忘记较长时间之前的信息，从而造成**梯度消失和梯度爆炸**现象。因此，RNN 等网络只能进行短期预测。近几年越来越多的学者尝试将 Transformer 运用在长序列预测中，Transformer 自 2017 年被提出后在 NLP 和 CV 领域取得巨大的成功，是**第一个完全依赖自注意力机制来捕捉输入与输出信息关系的传导模型**。自注意力机制保留该时刻信息与先前所有时刻信息的直接连接，能够缓解梯度消失和梯度爆炸的问题，允许信息在更长的序列上传播。同时，Transformer 不是类似 RNN 的顺序结构使用，具有更好的并行性，符合现有的 GPU 框架。然而，Transformer 存在三大挑战：**二次时间复杂度、高内存使用量和 encoder-decoder 架构局限性**，使其不能直接适用于长期时序预测问题。本文罗列了 3 个近两年改良比较成功的模型，它们针对 Transformer 存在的问题提出了不同改进手段，分别为 1) **Informer 模型**，来自发表于 AAAI21 的一篇最佳论文《Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting》，提出了 ProbSparse 自注意机制、自注意蒸馏操作和生成式解码器这三大改进，极大地降低了模型的计算复杂性；2) **Autoformer 模型**，由清华大学软件学院机器学习实验室在 2021 年提出，包含深度分解架构，和自相关机制两大创新点，能有效提升了长序列模型的预测性能；3) **FEDformer 模型**，由阿里达摩院在 2022 年提出，使用了低秩近似 (low-rank approximation) 将计算复杂性降为线性，并进一步提升了模型的预测精度。本篇为本系列的第二篇，分为三大部分，第一部分将介绍 Transformer 模型的要点和结构原理；第二部分将介绍三个变形模型各自的结构原理；第三部分将利用 Transformer 和三个变形模型对期货价格分别进行长序列预测并对比预测结果。

二、Transformer 的结构和原理

Transformer 模型和上篇报告中介绍的 RNN 系列模型差别较大，如图表 1 所示，Transformer 模型宏观上使用了 **Encoder-Decoder 框架**，并且在

Encoder (编码器) 和 Decoder (解码器) 部分都使用了多头注意力机制 (Multi-Head Attention), 因此本部分先对注意力机制进行介绍, 再对 Transformer 结构进行详细介绍。

图表 1: Transformer 模型的结构

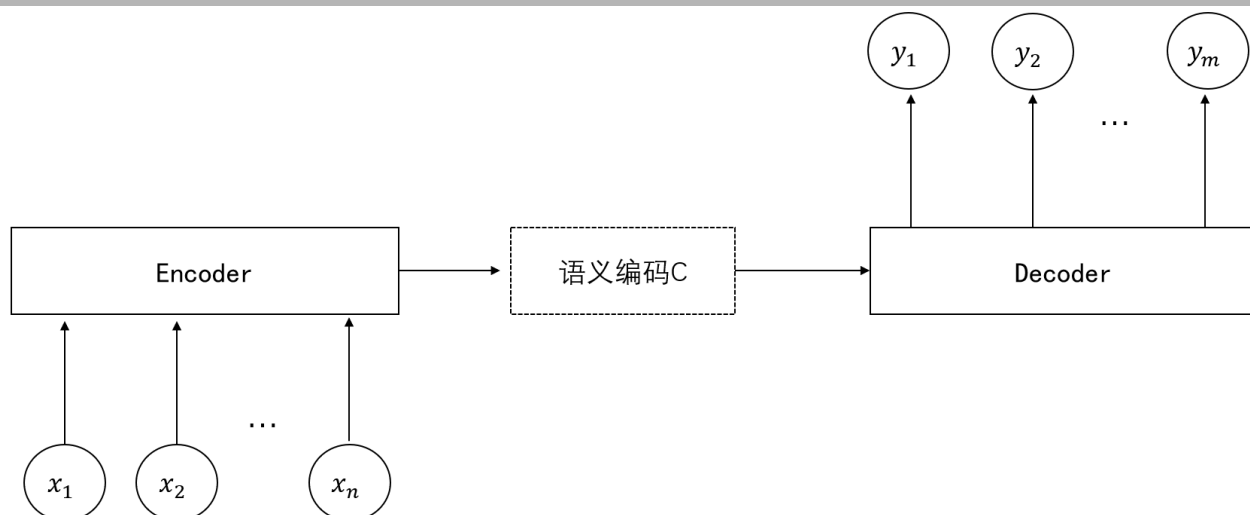


资料来源:《Attention is all you need》

（一）注意力机制

注意力机制最早由 Treisman 和 Gelade 提出，是一种模拟人脑注意力机制的模型，其核心目标是通过计算注意力的概率分布来从众多信息中突出某关键输入信息对输出的影响。目前大多数注意力机制均基于 Encoder-Decoder 框架，但需要注意的是注意力机制可以看作一种通用思想，本身是不依赖于特定框架的，而 Encoder-Decoder 框架可以看作是一种深度学习领域的研究模式，拥有广泛的应用场景，图表 2 展示了在 NLP 领域中常用的 Encoder-Decoder 抽象框架。这个框架中共含有三个部分分别为：Encoder（编码器）、Decoder（解码器）和语义表示 C。其中，Encoder 把一个变长的输入序列 $X(x_1, x_2, \dots, x_n)$ ，通过非线性变换转化为一个中间的语义表示 C： $C = f(x_1, x_2, \dots, x_n)$ ；Decoder 是根据输入序列 X 的中间语义表示 C 和先前已经生成的 y_1, y_2, \dots, y_{i-1} 来预测并生成 i 时刻的输出 $y_i = g(y_1, y_2, \dots, y_{i-1}, C)$ ， $f()$ 和 $g()$ 均为非线性转化函数。在这种传统框架中，输入信息被全部保存在语义表示 C 中，因此模型精度受输入句子长度的影响严重，从而在模型中引入了注意力机制，如图表 3 所示。

图表 2：抽象的 Encoder-Decoder 框架



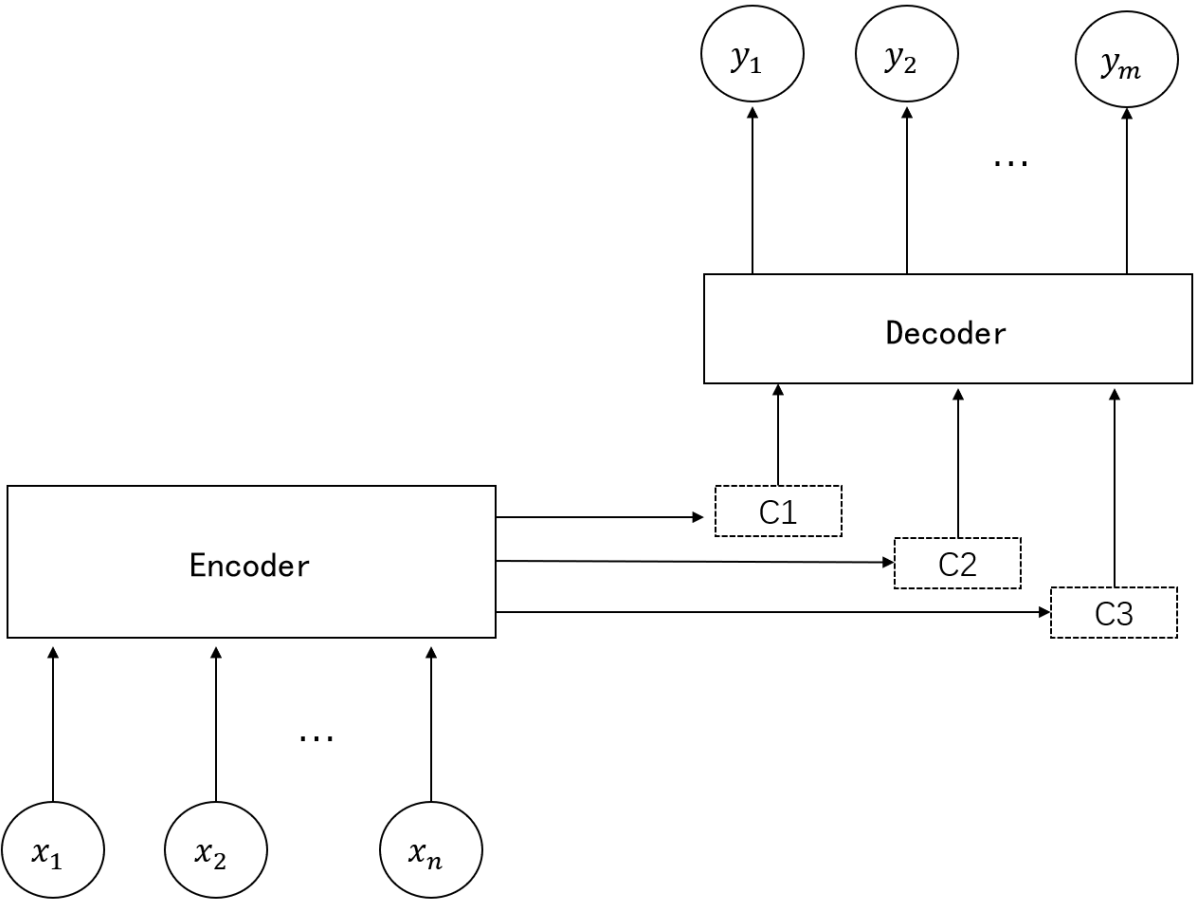
资料来源：中信期货研究所

如图表 3 所示，和抽象的 Encoder-Decoder 框架相比，引入注意力机制后改变最大的部分在于语义表示 C，原先针对不同输出值的语义表示 C 都是一致的，现在语义表示 C 会根据输出值对输入值做相应的权重分配。在面对一个任务时，可以把输入内容作为 Source，生成目标作为 Target，Source 可以看成由多个 <Key, Value> 对组成，而 Target 则由不同的 Query 组成，因此 Attention 机制的本质就是计算每一个 Query 在 Source 中所对应的值。

$$Attention(Query, Source) = \sum_{i=1}^{L_X} Similarity(Query, key_i) * value_i \quad (\text{公式 1})$$

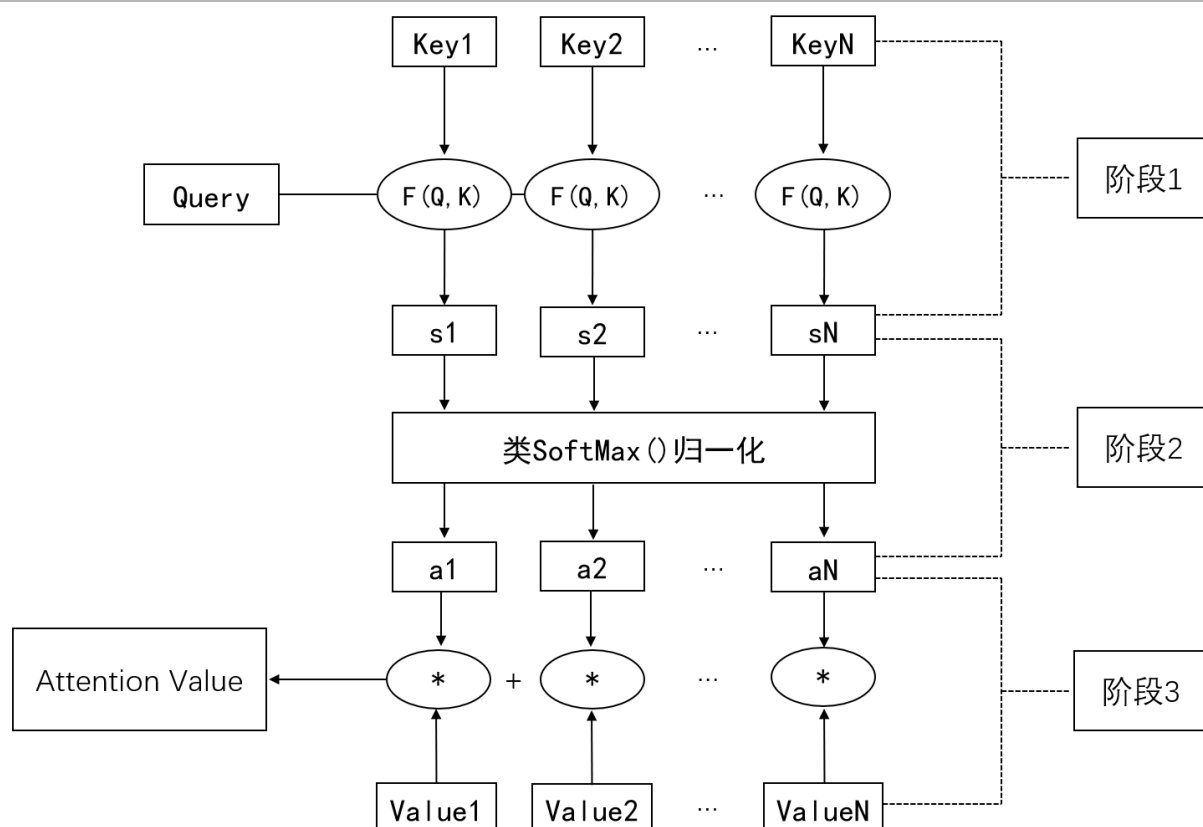
如图表 4 所示，Attention 机制的整体计算过程分为两步。首先，通过计算 Source 中所有 Key 与 Query 的相似性并对得出的相似性做 Softmax 归一化处理，从而得出相应权重。然后，根据得出的权重，对 Value 进行加权求和。

图表 3：引入注意力模型的 Encoder-Decoder 框架



资料来源：中信期货研究所

图表 4：Attention 计算过程



资料来源：中信期货研究所

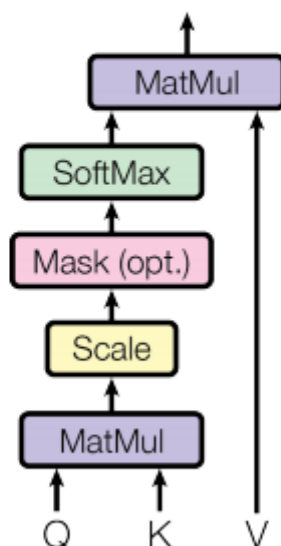
1. 自注意力机制(Self-Attention)

梳理完 Attention 机制后，将目光转向 Transformer 中使用的 Self-Attention 机制。和 Attention 机制相比 Self-Attention 机制最大的区别在于，Self-Attention 机制中 Target 和 Source 是一致的，所以 Self-Attention 机制是 Source 内部元素之间或者 Target 内部元素之间发生的 Attention 机制。Self-Attention 在计算过程中会直接将输入信息中任意两个信息直接联系起来，所以能有效缓解 RNN 系列算法自带的长期依赖问题。图表 5 展示了 Self-Attention 的计算流程，把 Query、Key 和 Value 分别装入 Q、K 和 V 三个矩阵中，再将三个矩阵带入以下公式中(公式 2)就可以求得 Value 的权重，这种计算方法被称为缩放点积注意力(Scaled Dot-Product Attention)，其中， d_k 为 Key 的维度。

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (\text{公式 2})$$

图表 5：缩放点积注意力模块(Scaled Dot-Product Attention)的计算流程

Scaled Dot-Product Attention



资料来源:《Attention is all you need》

2. 多头注意力机制 (Multi-Head Attention)

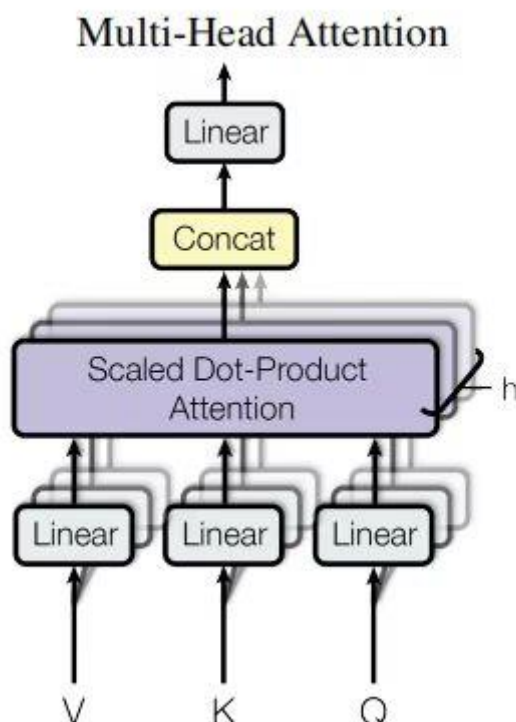
Vaswani 发现使用多头注意力机制 (Multi-Head Attention) 对输入信息进行多角度关联比使用单一注意力机制更有效, 因此在 Transformer 模型中采用了 Multi-Head Attention 机制, 图表 6 展示了 Multi-Head Attention 机制的计算流程, 计算公式如下。

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_n)W^O \quad (\text{公式 3})$$

$$where \ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

公式中 $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ 和 $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ 。通过多头自注意力机制, 可以考虑到不同序列位置的不同子空间的表征信息, 而在单一注意力机制中, 这些表征信息由于取均值操作的存在, 会被模型丢弃。

图表 6: 多头注意力机制 (Multi-Head Attention) 的计算流程



资料来源：《Attention is all you need》

（二）Transformer 结构详解

图表 1 展示了 Transformer 模型的整体结构，由 Input、Encoder、Decoder 和 Output 四部分组成，我们根据模型处理流程进行一一理解。

1. 输入编码

输入信息在进入 Encoder 和 Decoder 部分前会经历两种编码转换，分别为 Input/Output Embedding 和 Positional Encoding。由于是运用在时序预测中，输入信息大都为结构数据，因此在 Embedding 部分仅需对输入信息进行维度转换。Position Encoding 部分是利用输入信息的位置进行编码，因此可以提取出序列的顺序特征，具体公式如下：

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (\text{公式 4})$$

$$PE(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (\text{公式 5})$$

然后，将两种编码转换后的 Tensor 相加就可以带入相应的 Decoder 和

Encoder 部分。

2. Encoder 部分

在 Vaswani 发表的论文中，Encoder 部分共含有 6 层，前一层的输出会作为输入传入下一层中，最后一层 Encoder 的输出会输入到 Decoder 中。每一层 Encoder 中都含有两小层，分别为：一层 Multi-Head Attention 和一层 Feed Forward，每一小层后都带有一个 Add & Norm 层。首先来看一下 Add & Norm 层的作用，add 是指残差连接，将上一层网络的输入和输出相加，即 $F(x)+x$ ，相当于每层求导时都加上一个常数项 1，能缓解梯度消失的问题；Norm 是指归一化，需要注意的是 Transformer 中使用的归一化方式都是 Layer Normalization，即在每一个样本上计算均值和方差来进行归一，公式如下：

$$LN(x_i) = \alpha * \frac{x_i - \mu_L}{\sqrt{\sigma_L^2 + \varepsilon}} + \beta \text{ (公式 6)}$$

然后是 FeedForward 层，由两个线性变换和一个 ReLU 激活函数组成，公式如下：

$$FFN(x) = \max(0, xW_1 + b_1) W_2 + b_2 \text{ (公式 7)}$$

FFN 层通过第一层线性变换，将每个位置的注意结果映射到一个较大维的特征空间，再利用 ReLU 激活函数进行过滤，最后通过再次线性变化将输出结果恢复到原始维度。Multi-Head Attention 已在上文中进行介绍，此处就不详说了。

3. Decoder 部分

Decoder 部分和 Encoder 部分一样，都含有 6 层，每层包含三小层，分别为：一层 Masked Multi-Head Attention、一层 Multi-Head Attention 和一层 FeedForward，Decoder 部分的 FeedForward 层和 Encoder 是一样的，主要看一下 Masked Multi-Head Attention 层和 Multi-Head Attention 层。Masked Multi-Head Attention 层在 self-attention 机制上使用了 Mask 掩盖，来防止 Decoder 依据未来信息进行预测。Multi-Head Attention 层和 Encoder 中的 Multi-Head Attention 层相比，计算结构是一样的，但输入 Q、K 和 V 来源不同，Q 是来源于 Decoder 中上一层的输出，K 和 V 来源于 Encoder 的输出。

4. 输出部分

输出部分比较简单，由一层线性转换和一层 SoftMax 层组成，Decoder 的输出向量经过线性转换改变维度，再进行 SoftMax 计算就能得到最终的概率矩阵。

三、Informer 的结构和原理

在引言中有提到 Transformer 模型的三大挑战：二次时间复杂度、高内存使用量和 Encoder-Decoder 架构局限性，这三大挑战使 Transformer 模型不能被直接运用在长序列预测中，需要对算法和结构进行调整，Informer 模型就是改良较为成功的一种模型，图表 7 展示了 Informer 模型的结构，橘色、绿色和蓝色标志标出来的部分都是 Informer 主要改变的地方。针对平方级的计算复杂度这一大挑战，Informer 提出了 **ProbSparse Self-Attention 机制**，也就是图表 7 中橘色圈出的部分，通过筛选出重要的 Query 来降低计算复杂度。现在，我们来看一下 ProbSparse Self-Attention 机制的细节。Tsai 在 2019 年提出，第 i 个 Query 的 Attention 公式可以从概率的角度改写成如下公式 8 形式：

$$\mathcal{A}(q_i, K, V) = \sum_j \frac{k(q_i, k_j)}{\sum_{\ell} k(q_i, k_{\ell})} v_j = \mathbb{E}_{p(k_j|q_i)}[v_j] \quad (\text{公式 8})$$

$$p(k_j|q_i) = \frac{k(q_i, k_j)}{\sum_{\ell} k(q_i, k_{\ell})} \quad (\text{公式 9})$$

$$q(k_j|q_i) = \frac{1}{L_k} \quad (\text{公式 10})$$

其中， $p(k_j|q_i)$ 也就是公式 9，定义了一个概率的形式，即在给定第 i 个 Query 条件下 key 的分布。公式 10 是公式 9 的特殊形式，代表了均匀分布，若此 Query 分布接近均匀分布则说明此 Query 是“Lazy”的，反之说明 Query 是“Active”的。“Lazy”的 Query 对自注意力贡献较少，因此通过筛去“Lazy”的 Query 来降低 Transformer 模型的计算复杂度。具体规则中，使用了 KL 散度来衡量 $p(k_j|q_i)$ 和 $q(k_j|q_i)$ 的相似性，也就是 Query 的稀疏性，具体计算如公式 11 所示：

$$KL(q||p) = \ln \sum_{\ell=1}^{L_K} e^{\frac{q_i k_{\ell}^T}{\sqrt{d}}} - \frac{1}{L_K} \sum_{j=1}^{L_K} q_i k_j^T / \sqrt{d} - \ln L_K \quad (\text{公式 11})$$

$$M(q_i, K) = \ln \sum_{j=1}^{L_K} e^{\frac{q_i k_j^T}{\sqrt{d}}} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}} \quad (\text{公式 12})$$

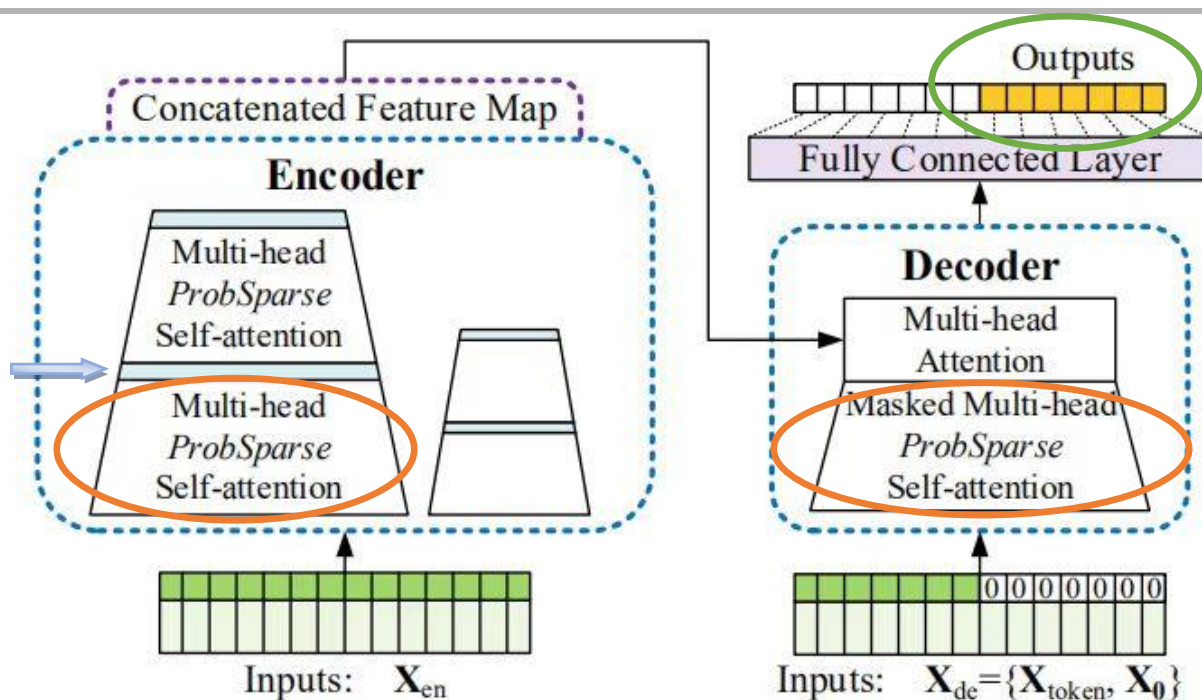
通过忽略 KL 散度中的常数项，就得出了 Query 稀疏度的衡量指标 $M(q_i, K)$ (公式 12)，该值越大说明此 Query 的概率分布与均匀分布差异越大，该 query 越活跃。通过计算每个 Query 的稀疏度，挑选出 u 个占主导地位的 Query，就可以得到一个和 Q 形状相同的稀疏矩阵 \bar{Q} ，此矩阵中除了挑选出的 Query 外，其他都是 0。此处 u 可以被定义为 $c * \ln Q$, c 为超参数。在公式 12 的第一部分中使用 Log-Sum-Exp (LSE) 操作，此操作存在潜在的数值稳定性问题，因此对 $M(q_i, K)$ 的计算流程进行了进一步的优化，如公式 13 所示。

$$\bar{M}(q_i, K) = \max_j \left\{ \frac{q_i k_j^T}{\sqrt{d}} \right\} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}} \quad (\text{公式 13})$$

和之前一样的步骤，通过计算每一个 Query 的 \bar{M} 值，找出 \bar{M} 值最大的 Query 形成稀疏矩阵 \bar{Q} ，最终 ProbSparse Self-Attention 机制的计算公式就调整成公式 14 所示。

$$\mathcal{A}(Q, K, V) = \text{Softmax} \left(\frac{\bar{Q} K^T}{\sqrt{d}} \right) V \quad (\text{公式 14})$$

图表 7：Informer 模型的结构

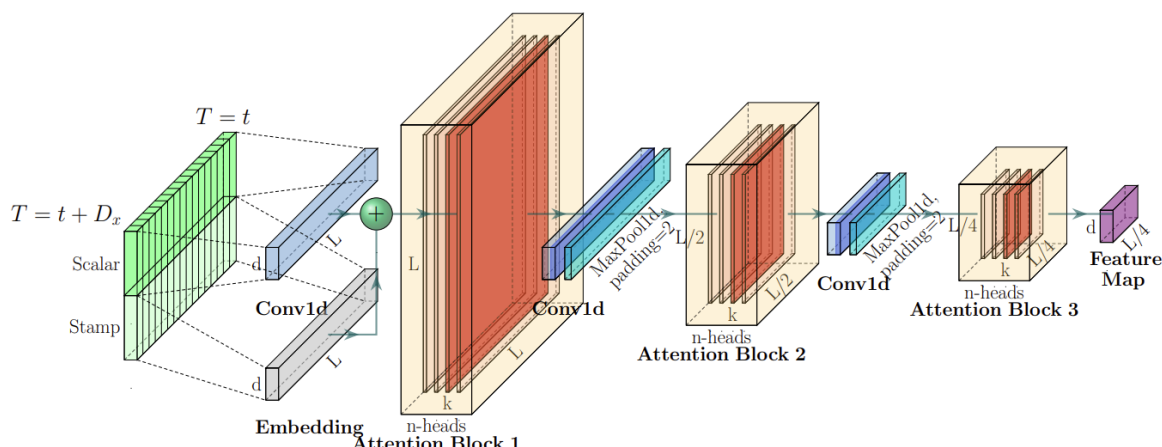


资料来源：《Informer: Beyond efficient transformer for long sequence time-series forecasting》

为了解决第二大挑战高内存使用量的问题，Informer 在 Encoder 的每层 Multi-head ProbSparse Self-Attention 层后都加入了一层自注意力蒸馏 (**Self-Attention Distilling**) 的操作，来对上一层的输出结果进行降维，图表 7 中蓝色箭头指出的部分。图表 8 和公式 15 共同展示了 Distilling 的计算过程，先利用卷积层对输入信息进行特征提取，再对提取后的信息进行一个 ELU 激活过滤，最后通过最大池化层进行降维，以此来增强特征信息的鲁棒性。

$$X_{j+1}^t = \text{MaxPool}(\text{ELU}(\text{Convld}[X_j^t]_{AB})) \quad (\text{公式 15})$$

图表 8：Informer-Encoder 部分侧面视角



资料来源：《Informer: Beyond efficient transformer for long sequence time-series forecasting》

最后一大挑战就是 Encoder-Decoder 结构自带的局限性，Transformer 模型中的 Decoder 部分采用的是 step-by-step 式的动态解码，因此解码速度很慢，Informer 提出了 **Generative Style Decoder**，图表 7 中绿圈圈出的部分。Informer 将 Start token 技术扩展到生成式的方法中，不再像以往选择一个额定的标志作为标记，而是从输入序列中抽取一个相对更短的长度作为输入，因此可以一步得出同样长度的输出结果，具体计算流程见公式 16。

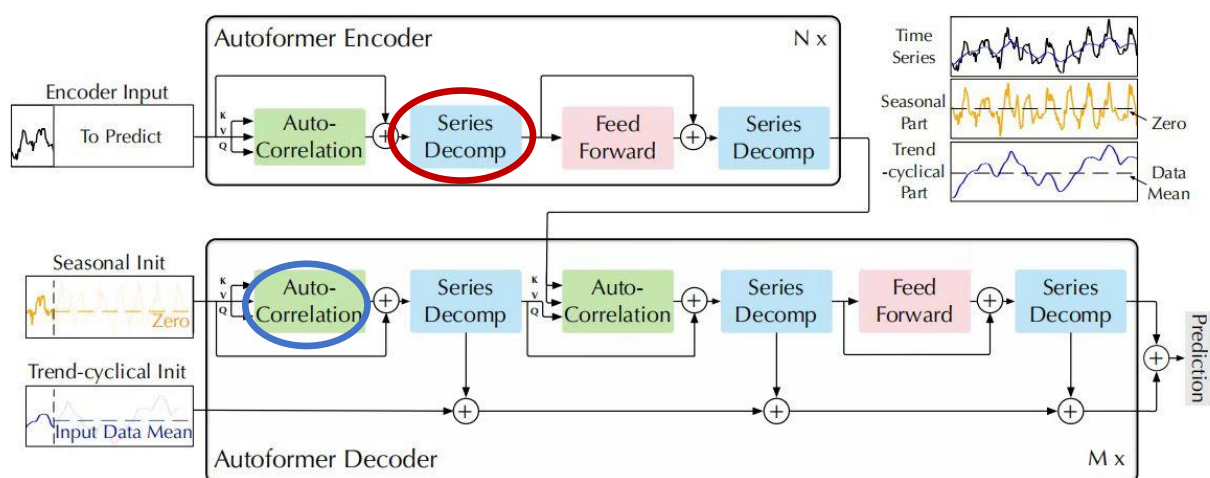
$$X_{de}^t = \text{Concat}(X_{token}^t, X_0^t) \in \mathbb{R}^{(L_{token}+L_y) \times d_{model}} \quad (\text{公式 16})$$

以上就是 Informer 模型对 Transformer 模型提出的三大改造，其他部分还是保持了 Transformer 模型原本的设置。这三大改造将模型的时间复杂度从 $O(L^2)$ 降到 $O(L \log(L))$ ，相关研究也证明 Informer 模型预测结果的拟合程度远高于 Transformer 模型。

四、Autoformer 的结构和原理

Informer 通过使用稀疏形式的注意力机制来缓解二次复杂度的问题，虽然在解决复杂度的问题上起到了效果，但也引起了信息未充分利用的问题，因此 Autoformer 模型被提出，针对 Transformer 的瓶颈进行了新一轮改良，Autoformer 模型主要含有两大革新点：**深度分解架构**和**自相关机制**。深度分解架构是 Autoformer 模型的整体架构，如图表 9 所示，包含三大部分：序列分解单元、自相关机制以及相应的 Encoder-Decoder 结构。

图表 9：Autoformer 模型的结构



资料来源：《Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting》

以往的序列分解方法包括上篇报告中介绍的 EMD 分解都是先对过去序列进行分解，再分别进行预测，这样会使预测结果受限于分解效果，且忽视未来各分解序列的相互作用。为了克服这个缺陷，Autoformer 将序列分解融合到整体模型中，遵循传统的分解手段将序列分解为季节项 (Seasonal) 和趋势项 (Trend-cyclical)，从而实现渐进式分解。序列分解单元 (Series Decomp Block) 的具体分解计算流程，如公式所示：

$$X_t = \text{AvgPool}(\text{Padding}(X)) \quad (\text{公式 17})$$

$$X_s = X - X_t \quad (\text{公式 18})$$

$$X_s, X_t = \text{SeriesDecomp}(X) \quad (\text{公式 19})$$

以上公式中 X 代表了长度为 L 的输入序列， X_s 和 X_t 分别代表了经均线分解后的季节项和趋势项，公式 19 就是 Series Decomp Block 的综合表达式。了解了解分解手段后，来细看一下模型结构。首先是模型输入，Encoder 的输入就是简单的历史序列 $X_{en} \in \mathbb{R}^{l \times d}$ ， l 为过去的时间长度；Decoder 的输入分为季节项 ($X_{des} \in \mathbb{R}^{(\frac{l}{2}+o) \times d}$) 和趋势项 ($X_{det} \in \mathbb{R}^{(\frac{l}{2}+o) \times d}$) 两部分，需要注意的是前 $\frac{l}{2}$ 部分是由 X_{en} 分解得到的，后面 o 长度部分分别由 0 和均值补齐。具体计算公式如下：

$$X_{ens}, X_{ent} = \text{SeriesDecomp}\left(X_{en_{\frac{l}{2}}}\right) \quad (\text{公式 20})$$

$$X_{des} = \text{Concat}(X_{ens}, X_0) \quad (\text{公式 21})$$

$$\mathcal{X}_{det} = \text{Concat}(\mathcal{X}_{ent}, \mathcal{X}_{mean}) \text{ (公式 22)}$$

然后是 Encoder 和 Decoder 的结构，在 Encoder 部分中，通过层层分解，逐步消除趋势项，只保留季节项，每一大层主要由一层自相关机制和一层 Feedforward 层组成。具体计算公式如下：

$$\mathcal{S}_{en}^{\ell,1}, _ = \text{SeriesDecomp}(\text{Auto} - \text{Correlation}(\mathcal{X}_{en}^{\ell-1}) + \mathcal{X}_{en}^{\ell-1}) \text{ (公式 23)}$$

$$\mathcal{S}_{en}^{\ell,2}, _ = \text{SeriesDecomp}(\text{FeedForward}(\mathcal{S}_{en}^{\ell,1}) + \mathcal{S}_{en}^{\ell,1}) \text{ (公式 24)}$$

$$\mathcal{X}_{en}^{\ell} = \text{Encoder}(\mathcal{X}_{en}^{\ell-1}) \text{ (公式 25)}$$

若 Encoder 部分含有 N 大层，那第 ℓ 层的输出可以归纳写成公式 25 的形式。在 Decoder 中，因为序列是被分为季节项和趋势项两部分分别解码，所以采用双路处理模式。在上层对季节项的解码中，先通过自相关机制提取出未来季节波动中的时间依赖特性，再在自相关机制中融合 Encoder 部分提取出的历史依赖关系，最后通过一层 FeedForward 层输出结果。而下层对趋势项的解码则采用了加权法，将上分支每个子层的输出进行加合。具体计算公式如下：

$$\mathcal{S}_{de}^{\ell,1}, \mathcal{T}_{de}^{\ell,1} = \text{SeriesDecomp}(\text{AutoCorrelation}(\mathcal{X}_{de}^{\ell-1}) + \mathcal{X}_{de}^{\ell-1}) \text{ (公式 26)}$$

$$\mathcal{S}_{de}^{\ell,2}, \mathcal{T}_{de}^{\ell,2} = \text{SeriesDecomp}(\text{AutoCorrelation}(\mathcal{S}_{de}^{\ell,1}, \mathcal{X}_{en}^N) + \mathcal{S}_{de}^{\ell,1}) \text{ (公式 27)}$$

$$\mathcal{S}_{de}^{\ell,3}, \mathcal{T}_{de}^{\ell,3} = \text{SeriesDecomp}(\text{FeedForward}(\mathcal{S}_{de}^{\ell,2}) + \mathcal{S}_{de}^{\ell,2}) \text{ (公式 28)}$$

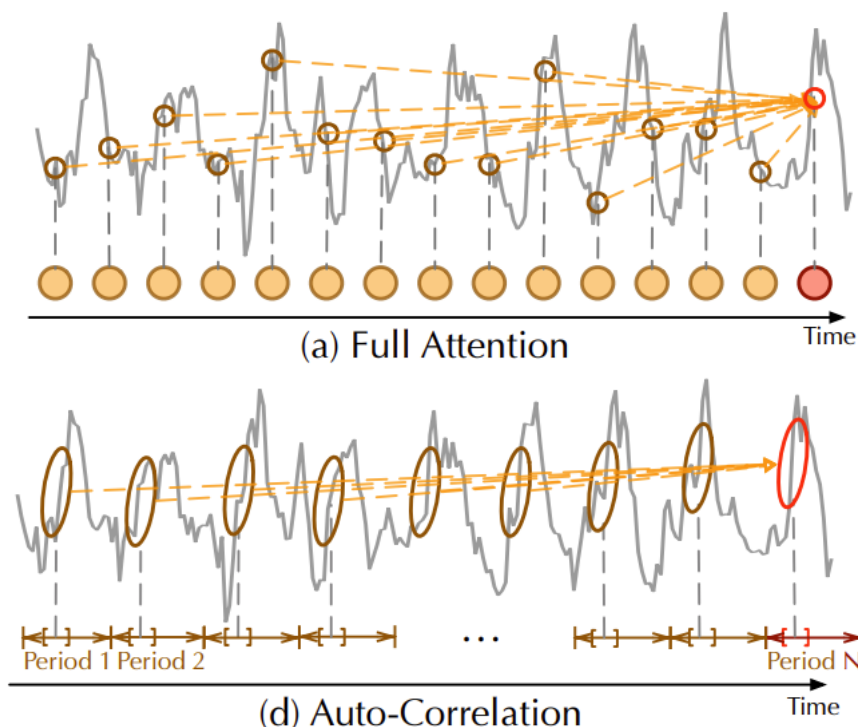
$$\mathcal{T}_{de}^{\ell} = \mathcal{T}_{de}^{\ell-1} + W_{\ell,1}\mathcal{T}_{de}^{\ell,1} + W_{\ell,2}\mathcal{T}_{de}^{\ell,2} + W_{\ell,3}\mathcal{T}_{de}^{\ell,3} \text{ (公式 29)}$$

$$\mathcal{X}_{de}^{\ell} = \mathcal{S}_{de}^{\ell,3} \text{ (公式 30)}$$

假设 Decoder 部分共有 M 层，那模型最终的输出结果为 $W_S \mathcal{X}_{de}^M + W_T \mathcal{T}_{de}^M$ 。这样的一个框架就是 Autoformer 模型的深度分解框架，刚刚有提到自相关机制，也是图表 9 中蓝圈圈出来的部分，现在我们着重来解释一下。

和自注意力机制相比，自相关机制 (Auto-Correlation) 从点对点连接调整为序列对序列连接，因此扩大了对信息的使用率，图表 10 形象地展示了这一变化。自相关机制是依靠基于周期的依赖性 (Period-based dependencies) 和时延信息聚合 (Time delay aggregation) 两大部分实现的，逐一来看。

图表 10：自注意力机制 VS 自相关机制



资料来源：《Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting》

基于周期的依赖性是指周期之间相同相位位置会自然地提供相似的子过程，因此基于随机过程理论，可以根据公式 31 计算出自相关系数 $R_{xx}(\mathcal{T})$ ，公式如下：

$$R_{xx}(\mathcal{T}) = \lim_{L \rightarrow \infty} \frac{1}{L} \sum_{t=0}^{L-1} x_t x_{t-\mathcal{T}} \quad (\text{公式 31})$$

$R_{xx}(\mathcal{T})$ 反映了序列 $\{x_t\}$ 与它的 \mathcal{T} 延迟序列 $\{x_{t-\mathcal{T}}\}$ 之间的相似性，可以被看成是未归一化的预测周期长度为 \mathcal{T} 的置信度 $R(\mathcal{T})$ 。而时延信息聚合是指基于周期间的依赖性连接预测周期中的子序列。如图表 11 右边部分所示，在时延聚合块中，根据选定的时间延迟 \mathcal{T}_1 、...、 \mathcal{T}_k 滚动，对齐在预测周期中相同相位位置的相似子序列。最后，再通过 Softmax 函数归一化置信度来聚合子系列。

了解了两个部分的含义后，来看一下自相关机制的整体计算流程。首先根据 Wiener-Khinchin 定理可知平稳随机过程 $x(t)$ 的功率谱密度 $S_x(\omega)$ 是其自相关函数 $R_x(\mathcal{T})$ 的傅里叶变换，公式如下：

$$S_{xx}(f) = \mathcal{F}(x_t) \mathcal{F}^*(x_t) \quad (\text{公式 32})$$

$$\mathcal{R}_{xx}(\mathcal{T}) = \mathcal{F}^{-1}(\mathcal{S}_{xx}(f)) \text{ (公式 33)}$$

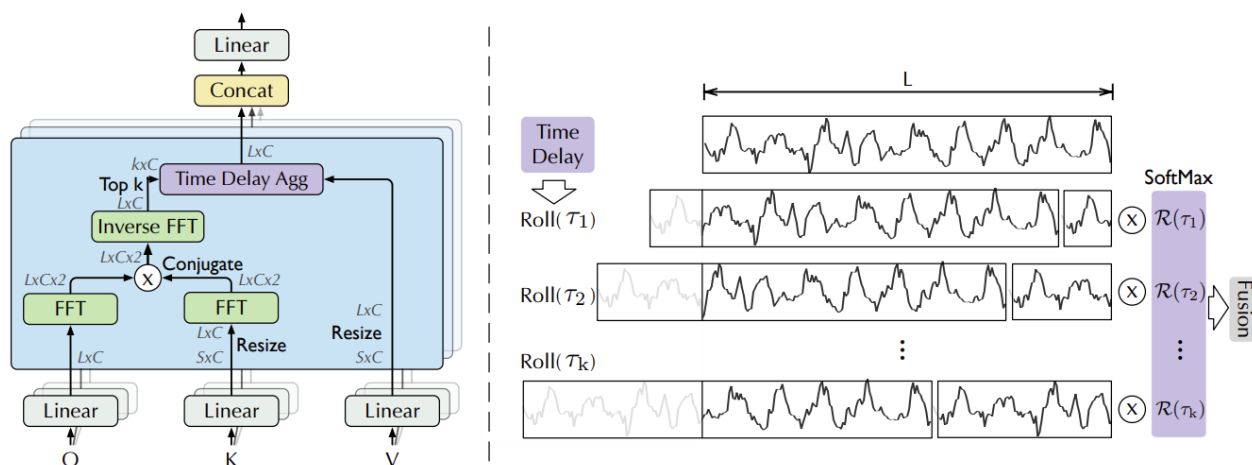
因此对 K 和 Q 分别执行快速傅里叶变换操作 (FFT), 如图表 11 左边部分所示, 以此求得 \mathcal{T} 从 1 到 $L-1$ 的所有 $\mathcal{R}_{xx}(\mathcal{T})$, 取出最大的 k 个 $\mathcal{R}_{xx}(\mathcal{T})$ 对其进行 SoftMax 函数转换得出 k 个概率, 最后将求得的概率与通过 Roll 函数对其的 V 相乘就可以得出最终结果。公式如下:

$$\mathcal{T}_1, \dots, \mathcal{T}_k = \text{argTopk}(\mathcal{R}_{Q,K}(\mathcal{T})) (\mathcal{T} \in \{1, \dots, L\}) \text{ (公式 34)}$$

$$\hat{\mathcal{R}}_{Q,K}(\mathcal{T}_1), \dots, \hat{\mathcal{R}}_{Q,K}(\mathcal{T}_k) = \text{SoftMax}(\mathcal{R}_{Q,K}(\mathcal{T}_1), \dots, \mathcal{R}_{Q,K}(\mathcal{T}_k)) \text{ (公式 35)}$$

$$\text{AutoCorrelation}(Q, K, V) = \sum_{i=1}^k \text{Roll}(V, \mathcal{T}_k) \hat{\mathcal{R}}_{Q,K}(\mathcal{T}_k) \text{ (公式 36)}$$

图表 11: Autoformer 自相关机制的计算流程



资料来源:《Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting》

和 Transformer 一样 Autoformer 也使用了多头机制, 具体计算公式如下:

$$\text{MultiHead}(Q, K, V) = \mathcal{W}_{\text{output}} * \text{Concat}(\text{head}_1, \dots, \text{head}_h) \text{ (公式 37)}$$

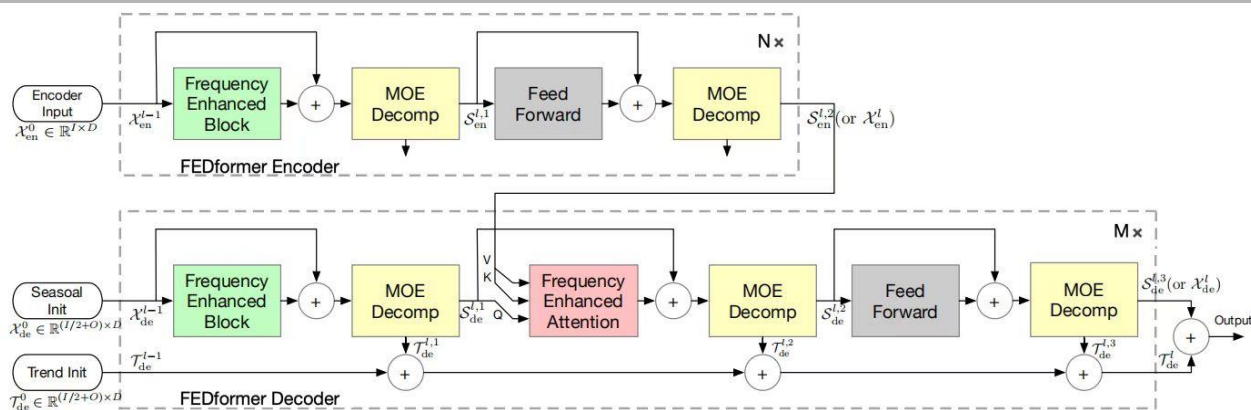
$$\text{where } \text{head}_i = \text{Auto-Correlation}(Q_i, K_i, V_i)$$

Autoformer 的整体结构和两大创新点就是这些, 经实验证明 Autoformer 和 Transformer 相比, 不仅能提高计算效率, 拟合程度也能大幅提高。

五、FEDformer 的结构和原理

FEDformer 模型是在 2022 年由阿里达摩院提出的，之前提出的 Transformer 变形虽然都已经在初始 Transformer 的基础上实现了优化，但仍难以捕捉时序的总体特征和分布，因此 FEDformer 提出了两大革新：1. 提出频率增强分解的 Transformer 结构，融入季节趋势分解手段，来更好地捕捉时序全局特性；2. 在 Transformer 结构中提出了傅里叶增强模块和小波增强模块，来替代原本的自注意力机制。图表 12 展示了 FEDformer 的整体结构，主要包括四大子模块：频域增强模块 FEB(Frequency Enhanced Block)、分解模块(MOE Decomp)、频域增强注意力模块 FEA(Frequency Enhanced Attention)和前向传播模块(Feed Forward)。FEDformer 的整体架构和 Autoformer 非常相似，但除了 Feed Forward 块，其他子模块的具体设计均不一致。同时，在 FEB 和 FEA 模块中都分别使用了离散傅里叶变换(DFT)和离散小波变换(DWT)两种不同手段，从而存在两种不同版本，因此我们分傅里叶增强结构和小波增强结构两部分进行讲解。

图表 12：FEDformer 模型的结构



资料来源：《FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting》

(一) 傅里叶增强结构

一般信号在频域上具有稀疏性，因此使用傅里叶变换将信号从时域转换到频域，在频域中只需保留少部分点位，就能近似还原出时域信号。此结构使用了离散傅里叶变换(DFT)，DFT 的时间复杂度为 $O(L^2)$ ，此结构通过随机选择一个傅里叶变换的子集极大地降低了输入向量的长度从而降低了计算复杂度。

我们先来看一下使用傅里叶变换的频率增强模块 (FEB-f)，该模块在结构

中的 Encoder 和 Decoder 部分均有涉及。输入信息 x 先通过一次线性变化转换成 q ，再对 q 进行傅里叶变换得到 Q ，在 Q 中随机挑选出 M 个子序列，再将挑选出的子序列与 R （一个随机初始化的参数化内核）进行点乘得到 Y ，对 Y 进行填充（零的部分填充为 $\mathbb{C}^{N \times D}$ ）后再进行逆傅里叶变换就可以得出最终的模块输出，具体公式如下，结构如图表 13 所示。

$$q = x \cdot w \text{ (公式 38)}$$

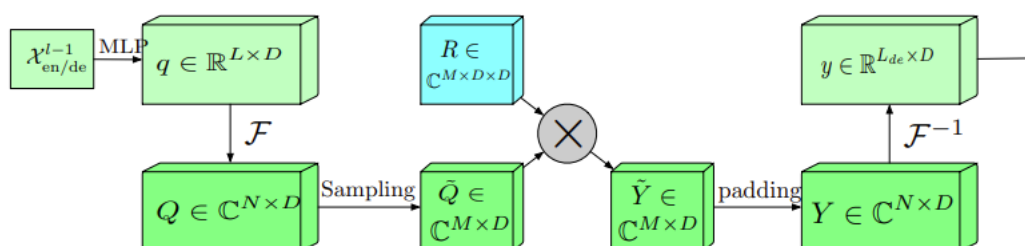
$$\tilde{Q} = \text{Select}(Q) = \text{Select}(\mathcal{F}(q)) \text{ (公式 39)}$$

$$\text{where } Q \in \mathbb{C}^{N \times D}, \tilde{Q} \in \mathbb{C}^{M \times D}, M \ll N$$

$$\text{FEB} - f(q) = \mathcal{F}^{-1}(\text{Padding}(\tilde{Q} \odot R)) \text{ (公式 40)}$$

$$\text{where } R \in \mathbb{C}^{D \times D \times M}, Y = Q \odot R \Rightarrow Y_{m,d_o} = \sum_{d_i=0}^D Q_{m,d_i} \cdot R_{d_i,d_o,m}$$

图表 13: FEB-f 结构



资料来源：《FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting》

然后是使用傅里叶变换的频率增强注意力模块。首先，对由 Queries、Keys 和 Values 得出的矩阵 q 、 k 和 v 进行傅里叶变换，挑选出 M 个子序列就得出了相应的 \tilde{Q} 、 \tilde{K} 和 \tilde{V} ，将 \tilde{Q} 和 \tilde{K} 相乘并用激活函数进行过滤，此处激活函数可以使用 SoftMax 函数或者 Tanh 函数，对过滤后的值和 \tilde{V} 相乘并对结果进行填充（零的部分填充为 $\mathbb{C}^{L \times D}$ ），最后进行一次逆傅里叶变换就可以得到最终输出结果，具体公式如下，结构如图表 14 所示。

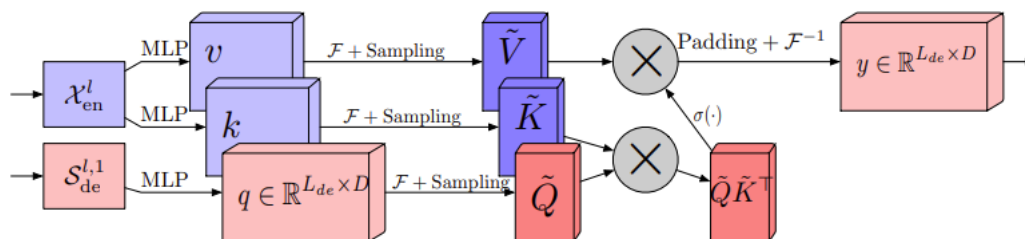
$$\tilde{Q} = \text{Select}(\mathcal{F}(q)) \text{ (公式 41)}$$

$$\tilde{K} = \text{Select}(\mathcal{F}(k)) \text{ (公式 42)}$$

$$\tilde{V} = \text{Select}(\mathcal{F}(v)) \text{ (公式 43)}$$

$$FEA - f(q, k, v) = \mathcal{F}^{-1}(\text{Padding}(\sigma(\tilde{Q} \cdot \tilde{K}^T) \cdot \tilde{V})) \quad (\text{公式 44})$$

图表 14: FEA-f 结构



资料来源:《FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting》

(二) 小波变换增强结构

小波变换是在傅里叶变换的基础上发展而来的, 傅里叶变化把信号从时域转换到频率能够更好地捕捉全局信息, 非常适合转换平稳序列。然而傅里叶变换只包含频率分析, 没有时域分析, 因此无法获取各频率出现的时间信息, 无法捕捉局部特征, 在面对突变和非平稳信号时效果就很差, 由此提出了小波变换的概念, 小波变换能够提供一个随频率改变的“时间-频率”窗口, 能考虑到时域和频率两方面, 因此缓解了傅里叶变换的瓶颈。对于给定的 $f(x)$, n 尺度下多小波系数可以被分别定义为 $s_l^n = [\langle f, \phi_{il}^n \rangle_{\mu_n}]_{i=0}^{k-1}$, $d_l^n = [\langle f, \psi_{il}^n \rangle_{\mu_n}]_{i=0}^{k-1}$, 其中 ϕ_{il}^n 为分段多项式小波的标准正交基。跨尺度的分解/重构可以定义为:

$$s_l^n = H^{(0)} s_{2l}^{n+1} + H^{(1)} s_{2l+1}^{n+1} \quad (\text{公式 45})$$

$$s_{2l}^{n+1} = \sum^{(0)} (H^{(0)T} s_l^n + G^{(0)T} d_l^n) \quad (\text{公式 46})$$

$$d_l^n = G^{(0)} s_{2l}^{n+1} + H^{(1)} s_{2l+1}^{n+1} \quad (\text{公式 47})$$

$$s_{2l+1}^{n+1} = \sum^{(1)} (H^{(1)T} s_l^n + G^{(1)T} d_l^n) \quad (\text{公式 48})$$

其中, $(H^{(0)}, H^{(1)}, G^{(0)}, G^{(1)})$ 是多小波分解滤波器的线性系数, 是被用于小波分解的混合矩阵。一个信号的多小波展示可以通过多尺度和多小波基的张量积得出。因为不同尺度下的基是由张量乘积结合得出的, 所以需要解开它, 本模型采用了非标准小波表示来解决此问题。对于一个映射函数 $F(x) = x'$, 此映射在多小波域下可以写成:

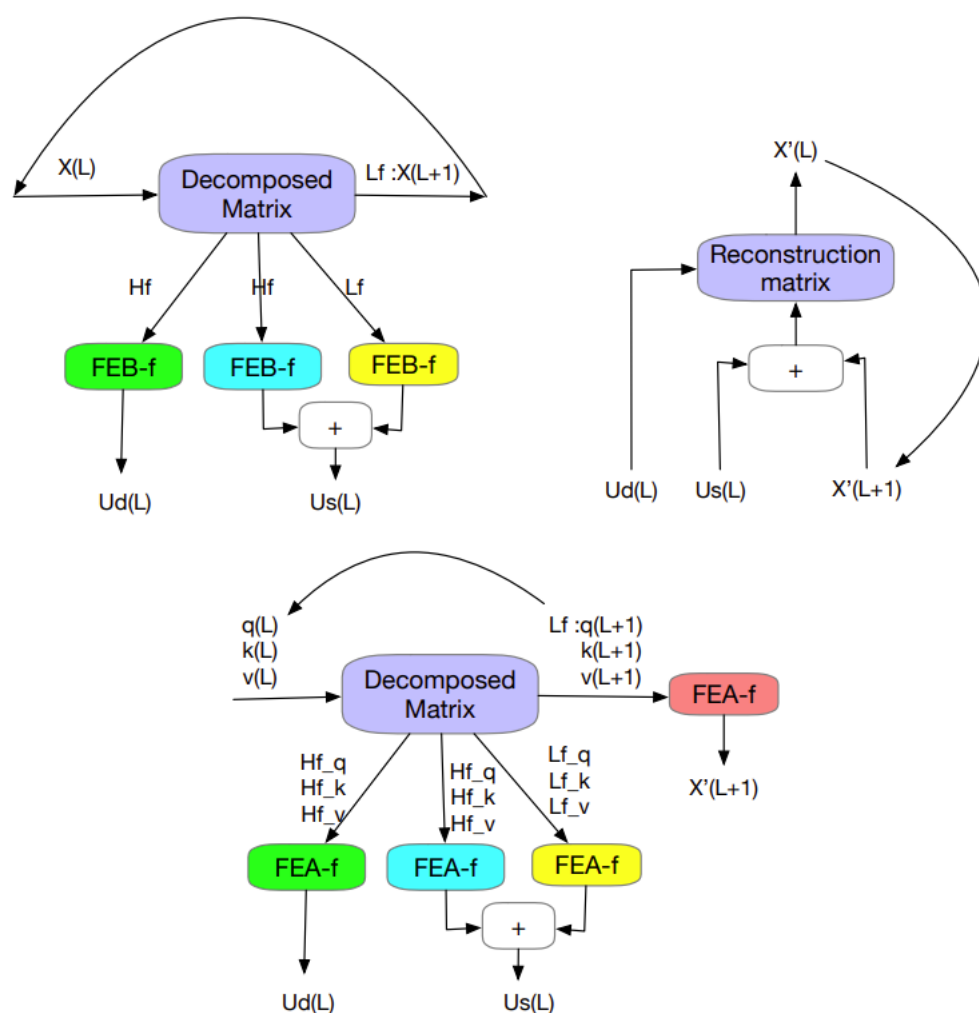
$$U_{dl}^n = A_n d_l^n + B_n s_l^n \quad (\text{公式 49})$$

$$U_{sl}^n = C_n d_l^n \text{ (公式 50)}$$

$$U_{sl}^L = \bar{F} s_l^L \text{ (公式 51)}$$

其中, $(U_{sl}^n, U_{dl}^n, s_l^n, d_l^n)$ 是多尺度、多小波系数, L 是递归分解下最粗的尺度, (A_n, B_n, C_n) 是在分解和重建过程中用于处理不同信号的三个独立的 FEB-f 模块, \bar{F} 是用来处理 L 分解后遗留的最粗信号的单层感知器。了解完基础概念, 来看一下 FEB-w 和 FEA-w 的板块结构, 图表 15 的左上部分展示了 FEB-w 的分解步骤, 下半部分展示了 FEA-w 的分解步骤, 右上部分展示了 FEB-w 和 FEA-w 均使用的重构步骤。

图表 15: FEB-w 和 FEA-w 结构



资料来源:《FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting》

FEB-w 不同于 FEB-f 的递归机制, 输入信息会被循环分解成 3 个部分, 独立

地进行运作。在小波分解部分中，使用了固定勒让德小波分解基分解矩阵。承接的三个 FEB-f 模块被分别用来处理产生的高频部分、低频部分和小波分解后遗留下来的部分。对于每一个周期 L ，它会产生一个处理过的高频张量 $U_d(L)$ ，一个处理过的低频张量 $U_s(L)$ 和原始低频张量 $X(L+1)$ 。这是一种阶梯向下的方法，分解阶段通过 $1/2$ 因子对信号进行抽取，运行最多 L 个循环，其中 $L < \log_2(M)$ 对于大小为 M 的给定输入序列。通常， L 被设置成一个固定参数，在不同分解周期 L 中，这三个 FEB-f 模块是共享的。而在小波重建部分中，会同样递归式的构建输出张量。在每一个循环 L 中，我们合并分解部分得出 $U_d(L)$ ， $U_s(L)$ 和 $X(L+1)$ 生成 $X(L)$ 。在每一个循环中，信号张量的长度维度会被提高两倍。

FEA-w 和 FEB-w 一样都包含了分解和重构阶段，FEA-w 和 FEB-w 的重构阶段是一致的，唯一的区别在于分解阶段。使用相同的分解矩阵对 q ， k ， v 信号分别进行分解，同时， q ， k ， v 也使用了相同的处理模块。上文中提及的 FEB-w 模块包含了三个用于信号处理的 FEB-f 模块，此处的 FEB-f 模块可以看作是一种自我注意力机制的替代，同时在 FEA-w 中使用了三个 FEA-f 模块。此外，另加了一个 FEA-f 模块来处理剩余最粗的 $q(L)$ ， $k(L)$ ， $v(L)$ 信号。

理解完 FEB 和 FEA 模块后，我们来看一下另一个改变较大的模块：分解模块 (MOE Decomp)。因为通常观察到的复杂周期部分是与真实世界中的趋势相融合的，所以凭借固定窗口的平均池化来提取趋势会非常困难。本模型为克服这个问题，提出了混合专家分解模块，它包含一组大小不同的平均过滤器来从输入信号中提取多个趋势成分和权重，并将它们组成最终趋势。具体公式如下：

$$X_{trend} = Softmax(L(x)) * (F(x)) \text{ (公式 52)}$$

其中， $F(x)$ 是一组平均池化过滤器， $Softmax(L(x))$ 是混合被提取出的趋势的权重。

FEDformer 的主要结构就是这些，它和 Autoformer 模型最大的区别点在于：Autoformer 是将序列分解为多个时域子序列进行特征提取，而 FEDformer 是采用频率变换将序列分解为多个频域模态来提取特征，所有的频率特征都是从整个序列中得出，因此具备更好的全局特性。本文接下来将在 CU、IF 和 T 三个品种的主力合约上测试 Transformer、Informer、Autoformer 和 FEDformer 四大类模型的预测效果。

六、预测结果分析

（一）数据选择和处理

本文选取沪铜期货、沪深 300 股指期货和十年期国债期货主力合约作为测试标的。考虑到上市时间，沪铜期货和沪深 300 股指期货数据集选用的时间范围均为 2011 年 1 月 1 日至 2022 年 12 月 31 日，每个合约共 2917 行数据，十年期国债期货数据集选用时间为 2015 年 3 月 20 日至 2022 年 12 月 31 日，共 1898 行数据，将数据集划分为训练集、验证集和测试集，其中切割比率为 0.8:0.1:0.1。所有测试均使用日频数据，选取各主力合约的开盘价、最高价、最低价、成交量和收盘价作为模型的输入特征。为缓解不同量纲带来的影响，提高模型训练效率，对数据进行 Z-score 标准化处理，转换公式如下：

$$x_{normalization} = \frac{x - \mu}{\sigma} \quad (\text{公式 53})$$

（二）预测结果评价指标

因为本次测试结果涉及多维，因此选用平均绝对误差 (MAE) 和均方误差 (MSE) 两种误差评价指标对模型预测性进行评估，两种指标的计算公式如下：

$$MAE = \frac{1}{m} \sum_{i=1}^m |y_i - \hat{y}_i| \quad (\text{公式 55})$$

$$MSE = \frac{1}{m} \sum_{i=1}^m (y_i - \hat{y}_i)^2 \quad (\text{公式 56})$$

其中， y_i 为真实值， \hat{y}_i 为预测值， \bar{y} 为真实值的平均值， m 为序列长度。MSE 和 MAE 用来衡量真实值和预测值间的偏差，值越小表明预测误差越小。

（三）模型参数设置

为探究参数设置对测试结果的影响，分别使用了过去 20/40/60 个交易日的日频数据对未来 5/10/20/40/60 个交易日的收盘价进行预测。参考相关论文，Encoder 和 Decoder 部分网络均设置为两层，多头机制部分多头设置为 8，Batch-size 设置为 32，迭代次数设置为 20，并使用早停机制，当连续 5 次迭代

轮次的损失值出现增加时停止训练。因为是对时序预测精度进行判断，因此选取均方误差 (MSE) 作为损失函数，采用 Adam 作为优化器进行训练，超参数使用默认参数，所有预测结果为随机训练三次结果的均值。最后，本文是基于 python 语言环境，以 PyTorch 作为深度学习框架进行训练和预测。

(四) 预测结果展示和分析

首先，将输入长度设置为 20 利用各个模型对 CU、IF 和 T 三个品种的主力合约收盘价分别进行预测，测试结果见图表 16。我们有以下几点发现，第一：几乎在所有情况下，FEDformer 的预测精度都是显著优于其他三个模型的，当输出长度较短时使用小波增强模块的 FEDformer 模型的预测精度高于使用傅里叶增强模块的 FEDformer 模型，反之亦然。因此，我们认为使用傅里叶变换将时序从时域转换到频域进行判断能更好地消除噪音，捕捉时序的全局特征，提高模型预测拟合精度。同时，在对短序列进行预测时，通过小波增强模块捕捉价格位置对时序变化的影响能再次强化模型的预测能力。第二：存在一次 Transformer 的预测精度优于其他模型的情况。我们发现在预测较短长度的序列时，Transformer 模型的预测结果经常优于 Autoformer 和 Informer 模型。此外，由于本次测试是随机训练三次，因此发现除 FEDformer 外其他三个模型的预测结果都非常不稳定，三次预测的结果差别较大。因此，我们认为仅凭时域关系对价格进行预测存在极大偶然性，虽然可能出现某次预测精度较优的情况，但预测能力不稳定，普适性较差。同时，Transformer 模型的计算复杂度虽然高，但在预测较短价格序列时，这一缺点并不明显，并不会对预测精度造成过多负面影响。第三：预测精度会随着预测序列长度逐步削弱，尤其在预测长度高于输入长度后，精度会发生更明显滑落，因此我们尝试通过扩大输入序列的长度来再次优化模型的预测能力，预测结果见图表 17 和图表 18。

图表 16：前 20 个交易日为输入长度的模型测试结果

Models		FEDformer-w		FEDformer-f		Autoformer		Informer		Transformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
CU	5	<u>0.033</u>	<u>0.134</u>	0.038	0.145	0.089	0.230	0.124	0.264	0.072	0.182
	10	<u>0.061</u>	<u>0.175</u>	0.066	0.185	0.105	0.254	0.190	0.323	0.117	0.237
	20	<u>0.126</u>	<u>0.230</u>	0.129	0.236	0.215	0.326	0.220	0.318	0.238	0.342
	40	0.350	0.380	<u>0.268</u>	<u>0.323</u>	0.360	0.398	0.372	0.436	0.400	0.480
	60	0.416	0.436	<u>0.374</u>	<u>0.408</u>	0.458	0.479	0.442	0.516	0.570	0.581
IF	5	<u>0.020</u>	<u>0.105</u>	0.022	0.111	0.079	0.223	0.060	0.362	0.037	0.153
	10	<u>0.033</u>	<u>0.135</u>	0.035	0.138	0.105	0.260	0.070	0.218	0.046	0.170

T	20	<u>0.061</u>	<u>0.191</u>	0.063	0.193	0.161	0.314	0.102	0.265	0.079	0.227
	40	0.160	0.318	<u>0.145</u>	<u>0.299</u>	0.244	0.398	0.181	0.352	0.160	0.333
	60	0.268	0.410	0.250	0.384	0.366	0.487	0.218	0.377	<u>0.135</u>	<u>0.305</u>
	5	<u>0.026</u>	<u>0.122</u>	0.030	0.136	0.050	0.174	0.044	0.165	0.033	0.143
	10	0.044	0.161	<u>0.040</u>	<u>0.157</u>	0.059	0.195	0.064	0.204	0.058	0.191
	20	0.081	0.219	<u>0.077</u>	<u>0.211</u>	0.106	0.256	0.086	0.229	0.094	0.245
	40	0.094	0.241	<u>0.086</u>	<u>0.228</u>	0.100	0.249	0.116	0.264	0.154	0.313
	60	0.110	0.260	<u>0.107</u>	<u>0.258</u>	0.123	0.280	0.220	0.371	0.319	0.450

资料来源：同花顺 iFind、中信期货研究所

经过测试发现，预测结果并未与预期一致，输入长度扩大后，不管是预测多久的序列，不同模型的预测精度都被削弱了。同时，除 FEDformer 外的三个模型在对短序列进行预测时，这种削弱影响会更为明显。我们仔细考虑后得出以下可能性分析，首先价格序列中噪音成分通常较大，输入序列越长噪音影响越大。其次，Autoformer 等依靠序列自相关性进行预测的模型通常抗噪能力较差，在预测短序列此类趋势不明显的序列时，更易受到噪音干扰，因此预测能力下降明显。其他方面，不管输入序列和输出序列长度如何设置，和之前预测结果一致，FEDformer 模型的预测结果几乎都是最优且最稳定的，这一点也再次证明了 FEDformer 模型具有更强的普适性。

图表 17：前 40 个交易日为输入长度的模型测试结果

Models		FEDformer-w		FEDformer-f		Autoformer		Informer		Transformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
CU	5	<u>0.054</u>	<u>0.174</u>	0.063	0.188	0.165	0.309	0.174	0.327	0.101	0.229
	10	<u>0.086</u>	<u>0.211</u>	0.097	0.228	0.199	0.325	0.259	0.363	0.133	0.237
	20	0.183	0.298	<u>0.183</u>	<u>0.297</u>	0.253	0.366	0.361	0.437	0.245	0.329
	40	0.366	0.406	<u>0.287</u>	<u>0.362</u>	0.400	0.453	0.458	0.558	0.387	0.450
	60	0.442	0.477	<u>0.366</u>	<u>0.433</u>	0.470	0.510	0.526	0.599	0.497	0.538
IF	5	<u>0.028</u>	<u>0.129</u>	0.038	0.151	0.194	0.341	0.120	0.271	0.040	0.158
	10	<u>0.046</u>	<u>0.164</u>	0.055	0.178	0.278	0.405	0.138	0.309	0.067	0.210
	20	<u>0.074</u>	<u>0.210</u>	0.082	0.225	0.278	0.420	0.226	0.385	0.091	0.253
	40	0.204	0.352	<u>0.173</u>	<u>0.324</u>	0.343	0.457	0.302	0.443	0.206	0.371
	60	0.420	0.506	<u>0.274</u>	<u>0.398</u>	0.485	0.554	0.412	0.507	0.298	0.431
T	5	<u>0.051</u>	<u>0.177</u>	0.060	0.191	0.158	0.317	0.057	0.195	0.060	0.203
	10	0.067	0.202	0.082	0.222	0.110	0.264	0.070	0.213	<u>0.057</u>	<u>0.195</u>
	20	0.111	0.260	0.112	0.264	0.139	0.291	0.111	0.269	<u>0.094</u>	<u>0.248</u>
	40	0.132	0.289	<u>0.124</u>	<u>0.266</u>	0.131	0.286	0.267	0.433	0.327	0.484
	60	0.112	0.273	<u>0.103</u>	<u>0.265</u>	0.268	0.395	0.344	0.498	0.416	0.512

资料来源：同花顺 iFind、中信期货研究所

图表 18：前 60 个交易日为输入长度的模型预测结果

Models		FEDformer-w		FEDformer-f		Autoformer		Informer		Transformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
CU	5	0.061	0.184	0.096	0.227	0.405	0.467	0.211	0.372	0.141	0.275
	10	0.106	0.234	0.168	0.298	0.350	0.440	0.189	0.334	0.139	0.257
	20	0.313	0.380	0.232	0.365	0.304	0.275	0.327	0.452	0.264	0.370
	40	0.394	0.454	0.346	0.443	0.393	0.476	0.633	0.668	0.360	0.469
	60	0.438	0.489	0.402	0.485	0.444	0.541	1.620	1.125	0.539	0.597
IF	5	0.041	0.160	0.069	0.203	0.245	0.376	0.141	0.308	0.063	0.193
	10	0.095	0.243	0.083	0.222	0.264	0.402	0.265	0.430	0.091	0.243
	20	0.107	0.257	0.124	0.278	0.268	0.409	0.371	0.497	0.155	0.322
	40	0.401	0.510	0.241	0.391	0.380	0.473	0.499	0.575	0.316	0.439
	60	0.511	0.607	0.406	0.512	0.872	0.771	0.495	0.578	0.505	0.569
T	5	0.063	0.199	0.104	0.256	0.226	0.367	0.133	0.300	0.074	0.229
	10	0.098	0.240	0.119	0.273	0.177	0.327	0.129	0.300	0.124	0.298
	20	0.142	0.292	0.145	0.302	0.173	0.318	0.215	0.393	0.176	0.353
	40	0.167	0.317	0.157	0.296	0.240	0.387	0.378	0.553	0.415	0.571
	60	0.139	0.285	0.107	0.272	0.119	0.283	0.439	0.587	0.679	0.752

资料来源：同花顺 iFind、中信期货研究所

最后，我们结合模型的计算复杂度来综合比较一下这四个模型，计算复杂度见图表 19。虽然 FEDformer 模型公式推导后的计算复杂度是最低的，但它单次迭代所需时间是最久的，其他三个模型的迭代时间较为相近，FEDformer 单次迭代所需时间是其他三个模型的三倍多，采用小波增强模块的 FEDformer 单次迭代所需时间比采用傅里叶增强模块的 FEDformer 更久。因此我们认为在输入长度和输出长度较短时，Transformer 过高计算复杂度带来的影响并不明显，FEDformer 模型的优势更多体现于全局信息的捕捉和降噪能力。

图表 19：模型复杂度对比

模型	时间复杂度	记忆复杂度
FEDformer	$\mathcal{O}(L)$	$\mathcal{O}(L)$
Autoformer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$
Informer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$
Transformer	$\mathcal{O}(L^2)$	$\mathcal{O}(L^2)$

资料来源：中信期货研究所

七、结论与展望

本文详细介绍了 Transformer 模型的层结构和具体算法，并从近两年顶刊论文中挑选出三个模型：Informer、Autoformer 和 FEDformer 进行针对性介绍，最后在 CU、IF 和 T 三个期货品种的主力合约上进行测试，发现：

1. 不管输入序列和输出序列长度如何设置，FEDformer 模型都取得了更优地预测结果。同时，通过使用小波增强模块，可以获取时序位置对价格变动的的影响，以此优化 FEDformer 模型对短序列的预测能力。
2. Autoformer、Informer 和 Transformer 模型在多次测试中虽然存在较优的预测结果，但在不同轮次中预测精度跳跃较大，预测结果存在较强偶然性，预测能力不稳定，普适性较差。
3. 所有模型的预测精度并不会随着输入长度增加而提高，输入长度越大，预测精度反而越低。同时，预测精度会随着输出长度的增加而削弱。
4. 在对较短序列进行预测时，Transformer 过高计算复杂度对预测精度和迭代耗时的负面影响并不明显。相反 FEDformer 模型单次迭代所需时间最高，是其他三个模型所需时间的三倍多。在很多情况中，Transformer 模型的预测精度也是优于 Autoformer 和 Informer 模型的。

价格时序中信噪比较低，去噪能力和信息提取能力同样重要，综合来看，在这四个模型中 FEDformer 会更适用于价格的长序列预测。本系列更侧重于对算法本身的介绍和使用方法的讨论，之后会在其他系列中运用此系列介绍的算法进行具体因子和交易策略的构建。此篇为本系列的第二篇，结合上一篇 RNN 系列的算法尝试为投资者提供从量化的角度对价格进行长短期多维度判断的方法，同时，本次测试并未对神经网络层数、神经元个数和迭代次数进行过多讨论，后续可以通过更改参数来对模型进行更深地挖掘。

参考文献

- 1) Cleveland, R. B. , Cleveland, W. S. , McRae, J. E. , and Terpenning, I. Stl:A seasonal-trend decomposition. Journal of official statistics, 6(1):3-73, 1990.
- 2) Gupta, G. , Xiao, X. , and Bogdan, P. Multiwavelet-based operator learning for differential equations, 2021.
- 3) 任欢, 王旭光. 注意力机制综述[J]. 计算机应用. 2021, 41 (S1).
- 4) Vaswani, A. , Shazeer, N. , Parmar, N. , Uszkoreit, J. , Jones, L. , Gomez, A. N. , Kaiser, L. , and Polosukhin, I. Attention is you need. CoRR, abs/1706.03762, 2017.
- 5) Wen, Q. , Gao, J. , Song, X. , Sun, L. , Xu, H. , and Zhu, S. RobustSTL:A robust seasonal-trend decomposition algorithm for long time series. In Proceedings of the AAAI Conference on Artificial Intelligence, V(33), pp. 5409-5416, 2019.
- 6) Wu, H. , Xu, J. , Wang, J. , and Long, M. Autoformer:Decomposition transformers with auto-correlation for long-term series forecasting. In Proceedings of the Advances in Neural Information Proceeding System(NeurIPS), pp. 101-112, 2021.
- 7) Zhou, T. , Ma, Z. , Wen, q. , Wang, X. , Sun, L. , and Jin, R. FEDformer:Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In Proceedings of the 39th International Conference on Machine Learning(ICML), 2022
- 8) Zhou, H. , Zhang, S. , Peng, J. , Zhang, S. , Li, J. , Xiong, H. , and Zhang, W. Informer:Beyond efficient transformer for long sequence time-series forecasting. In The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI2021, Virtual Conference, volume 35, pp. 11106-11115. AAAIPress, 2021.
- 9) 朱张莉, 饶元, 吴渊, 祁江楠, 张钰. 注意力机制在深度学习中的研究进展[J]. 中文信息学报. 2019, 33(6).

免责声明

除非另有说明，中信期货有限公司拥有本报告的版权和/或其他相关知识产权。未经中信期货有限公司事先书面许可，任何单位或个人不得以任何方式复制、转载、引用、刊登、发表、发行、修改、翻译此报告的全部或部分材料、内容。除非另有说明，本报告中使用的所有商标、服务标记及标记均为中信期货有限公司所有或经合法授权被许可使用的商标、服务标记及标记。未经中信期货有限公司或商标所有权人的书面许可，任何单位或个人不得使用该商标、服务标记及标记。

如果在任何国家或地区管辖范围内，本报告内容或其适用与任何政府机构、监管机构、自律组织或者清算机构的法律、规则或规定内容相抵触，或者中信期货有限公司未被授权在当地提供这种信息或服务，那么本报告的内容并不意图提供给这些地区的个人或组织，任何个人或组织也不得在当地查看或使用本报告。本报告所载的内容并非适用于所有国家或地区或者适用于所有人。

此报告所载的全部内容仅作参考之用。此报告的内容不构成对任何人的投资建议，且中信期货有限公司不会因接收人收到此报告而视其为客户。

尽管本报告中所包含的信息是我们于发布之时从我们认为可靠的渠道获得，但中信期货有限公司对于本报告所载的信息、观点以及数据的准确性、可靠性、时效性以及完整性不作任何明确或隐含的保证。因此任何人不得对本报告所载的信息、观点以及数据的准确性、可靠性、时效性及完整性产生任何依赖，且中信期货有限公司不对因使用此报告及所载材料而造成的损失承担任何责任。本报告不应取代个人的独立判断。本报告仅反映编写人的不同设想、见解及分析方法。本报告所载的观点并不代表中信期货有限公司或任何其附属或联营公司的立场。

此报告中所指的投资及服务可能不适合阁下。我们建议阁下如有任何疑问应咨询独立投资顾问。此报告不构成任何投资、法律、会计或税务建议，且不担保任何投资及策略适合阁下。此报告并不构成中信期货有限公司给予阁下的任何私人咨询建议。

深圳总部

地址：深圳市福田区中心三路8号卓越时代广场（二期）北座13层1301-1305、14层

邮编：518048

电话：400-990-8826

传真：(0755) 83241191

网址：<http://www.citicsf.com>