北京工業大學
BEIJING UNIVERSITY OF TECHNOLOGY

# 毕 业 论 文

题　　目　基于时频域特征耦合的
　　时间序列长期预测框架

姓　　名　周子渔

学　　号　20074214

指导教师　吕庚育

日　　期　2024 年 6 月 1 日

# 北京工业大学

# 毕业设计（论文）任务书

| 课目 | 基于时频域特征耦合的时间序列长期预测框架 | | | | |
|---|---|---|---|---|---|
| 专业 | 计算机科学与技术 | 学号 | 20074214 | 姓名 | 周子渔 |

【主要内容】

本课题拟提出基于 iTransformer 模型的时间序列长期预测模型。该模型结合时间序列在时频域上的特征，并设计特殊的注意力机制，实现对其时序信息的有效捕捉，从而提高预测的准确性。

【基本要求】

知识与理解能力：学生必须展示对机器学习和深度学习领域的基本理解，并对当前该领域的前沿进展有一定的认识。

技术技能：需要具备高级 Python 编程技能，对 PyTorch 框架和 Linux 操作环境有熟练的掌握和应用能力。

个人素质：学生应具备成熟的心态，能在面对学术研究过程中出现的挑战和困难时，表现出坚持不懈的态度。

文献能力：应具有较好的文献处理能力，包括阅读和撰写中英文学术论文的能力，以及有效地整理和呈现研究成果的技巧。

【主要参考资料】

[1] Haixu Wu et al. “TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis”. In: International Conference on Learning Representations.2023.

[2] Ashish Vaswani et al. “Attention is all you need”. In:Advances in neural information processing systems (2017).

完成期限：2024-06-01

指导教师签章：

专业负责人签字：　　2023 年 12 月 22 日

# 毕业设计（论文）诚信声明书

本人郑重声明：在毕业设计（论文）工作中严格遵守学校有关规定，恪守学术规范；所提交的论文是我个人在导师指导下独立研究、撰写的成果，毕业设计（论文）中所引用他人的文字、研究成果，均已在毕业设计（论文）中加以说明；在本人的毕业生设计（论文）中未剽窃、抄袭他人的学术观点、思想和成果，未篡改实验数据。

本毕业设计（论文）和资料若有不实之处，本人愿承担一切相关责任。

学生签名：_____ 日期：_____

# 关于论文使用授权的说明

本人完全了解北京工业大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。

（保密的论文在解密后应遵守此规定）

签名：_____ 导师签名：_____ 日期：2024 年 6 月 1 日

# 摘 要

时间序列广泛存在于人类的生产生活，一直以来是数据挖掘领域的重点研究对象。在时间序列分析的下游任务中，长期预测被认为是最难、最有价值的一类任务。准确的长期预测可以有效帮助人类把握未来，进而做出合理的决策。在基于深度学习的时间序列预测方案中，Transformer 因其自注意力机制在捕捉长期依赖性方面的卓越能力而被广泛应用于时间序列建模。然而，在处理具有众多变量的多元时间序列数据时，原始自注意力机制倾向于均匀且分散地分配注意力权重，出现注意力图中的行同质化现象，阻碍了注意力机制挖掘变量之间相关性，进而影响了模型的预测性能。

为了解决这一问题，本文提出了一种“基于时频域特征耦合的时间序列长期预测框架”，命名为 SDformer。该模型包括两个新颖的模块：谱滤波变换和动态定向注意力，并将它们整合到 Transformer 的编码器中实现更集中的注意力分配。具体来说，谱滤波变换模块利用快速傅里叶变换依幅值选取并保留最突出的频率，在频域内实现有效的去噪，并结合汉明窗口在时域内平滑处理频域内去噪过滤后的序列数据；动态定向注意力将一个特殊的核函数应用于从去噪数据中投影出的查询和键向量，更有效地将注意力集中到最具信息量的变量上，以获得更准确的注意力分布。这两个模块共同使得注意力权重的分布差异更加显著，从而增强了注意力捕捉多变量相关性的能力，有效提高了传统 Transformer 模型的预测性能。

SDformer 在 7 个公开数据集上的广泛实验证明了其相较于其他最先进模型的优越性能，充分的消融实验和超参数敏感性分析证明了模型中模块化设计的有效性。此外，本文从定性和定量两个角度证明了 SDformer 可以有效缓解传统 Transformer 的自注意力机制中的注意力分布过于均匀的问题。对模型复杂度的分析也说明了 SDformer 的轻量性，其计算开销低于现有的先进模型。SDformer 凭借其强大的性能，为多变量时间序列的长期预测任务提供了新的解决方案。

**关键词：**时间序列分析；Transformer 模型；傅里叶变换；注意力机制

# Abstract

Time series extensively pervade human productive activities and daily life, consistently standing as a pivotal area of research within the data mining domain. Among the myriad tasks linked to time series analysis, long-term forecasting emerges as the most challenging yet valuable. Precise long-term forecasts can significantly aid humans in grasping future trends, thus facilitating informed decision-making.

In the realm of deep learning-based approaches to time series forecasting, the Transformer model, with its unparalleled proficiency in capturing long-term dependencies via its self-attention mechanism, has seen widespread application in time series modeling. Nonetheless, when addressing multivariate time series data encompassing numerous variables, the original self-attention mechanism tends to distribute attention weights in a uniform and dispersed manner, resulting in a homogenization phenomenon within the attention maps. This obstructs the mechanism's capacity to unearth inter-variable correlations, adversely impacting the predictive performance of the model.

To address this challenge, this manuscript introduces an innovative long-term forecasting framework for time series, predicated on the synergistic coupling of temporal and frequency domain features, dubbed SDformer. This framework encompasses two novel components: the Spectral Filter Transform (SFT) and Dynamic Directional Attention (DDA), seamlessly woven into the Transformer's encoder to ensure a more targeted allocation of attention. The SFT module leverages the Fast Fourier Transform to select and preserve dominant frequencies, facilitating effective denoising in the frequency domain, while simultaneously employing a Hamming window for refined processing of the denoised sequence data in the temporal domain. The DDA module employs a unique kernel function on the query and key vectors derived from the denoised data, enabling a more efficient focus of attention on the variables rich in information for a more accurate distribution of attention weights. Collectively, these modules significantly diversify the attention weight distribution, enhancing the model's proficiency in discerning correlations among multiple variables and substantially elevating the traditional Transformer model's forecasting accuracy.

Extensive empirical evaluation across seven public datasets attests to SDformer's superior performance relative to other leading-edge models, with comprehensive ablation studies and hyperparameter sensitivity analysis affirming the modular design's efficacy. Furthermore, both qualitative and quantitative assessments verify SDformer's capability to alleviate the conventional Transformer self-attention mechanism's tendency towards an overly uniform distribution of attention. Analysis concerning the model's complexity also reveals SDformer's streamlined nature, showcasing a lower computational expenditure compared to current advanced models. Through its formidable performance, SDformer presents a novel avenue for long-term forecasting tasks within multivariate time series.

**Keywords:** Time Series Analysis, Transformer, Fourier Transform, Attention Mechanism

# 目 录

# 第 1 章　绪论

本章首先在1.1节介绍时间序列预测问题的研究背景和意义。然后在1.2节分别介绍基于深度学习的三类时间序列预测模型，并对研究现状进行总结。在此基础上分析现有模型的不足之处，并在1.3节阐述本文的研究动机和具体的实现方案。最后介绍文本总体的组织结构。

## 1.1　课题研究背景及意义

时间序列（Time Series）是指一组按照事件发生先后顺序排列而成的数据点序列[1]。如图1-1所示，人们的日常生活中充斥着大量的时间序列数据，例如基于电力负荷的时间序列就是电力系统中的观测指标（如用电功率）随着时间变化的过程[2] [3]。此外，近年来随着物联网和大数据的迅速发展以及可穿戴设备的大范围普及，各种各样的时间序列数据大量涌现，因此如何理解并分析这些海量的、杂乱无章的时序数据成为了数据挖掘领域重要的议题之一[4]。在数据挖掘领域，传统意义上的时间序列分析包括四个下游任务：预测（长期、短期)[5]、分类[6]、缺失值填充[7]和异常检测[8]，其中时间序列长期预测被认为是最重要、最困难的一类任务。


图 1-1 现实世界中广泛存在的时间序列数据

具体来讲，时间序列预测是指在某个特定的时间点，利用某个时间序列的历史观测以及该数据所在领域内的先验知识，挖掘和分析该时间序列的周期性、波动性等时序模式，从而预测未来一定长度的序列值。有效的时间序列预测不仅能让人们预知未来并做出合理决策，同时也能增强人们对历史时间序列所蕴含的时序信息的理解。例如在医疗

领域，医生可以基于在连续时间观测且规则采样的功能磁共振成像（fMRI）数据对病人进行诊断[9]，判断病人是否患病的同时掌握该疾病在人体内的具体表征。此外，在气象研究领域，研究者也可以根据历史的气象数据分析并判断未来的气象变化，为人们安排合理出行提供参考[10]。

近年来，深度学习技术在时间序列预测任务上取得了卓越的成果，越来越多的深度模型从多种不同的视角建模时间序列，其中基于 Transformer[11] 及其改进版本的模型占据了主导地位[12] [13] [14] [15]。但传统的 Transformer 模型在时序分析中也存在着一些局限性。首先，传统的注意力机制具有较高的时间和空间复杂度，计算资源的受限导致传统 Transformer 模型难以建模序列长度较大的时间序列。此外，本文还发现传统的自注意力机制在建模变量数较多的时间序列时会出现注意力分布过于均匀的现象，意味着注意力机制没有能够有效关注到重要的时间点或变量，进而影响了模型对时序特征（Temporal Patterns）和变量相关性（Inter-variate Correlations）的有效识别。

为了解决这些问题，本文提出了基于时频域特征耦合的时间序列长期预测框架，命名为 SDformer。模型设计方面，SDformer 通过仅使用 Transformer 的 Encoder 结构编码历史信息，从而摒弃了传统 Transformer 中复杂的编码器-解码器（Encoder-Decoder）结构，大大降低了模型的计算开销；此外通过设计去噪平滑模块“谱滤波变换”（Spectral-Filter-Transform，SFT）以及创新的注意力机制“动态定向注意力”（Dynamic-Directional-Attention，DDA），有效解决了传统注意力机制中注意力分布过于均匀的问题，有效提高了注意力机制的表征学习能力，促进了模型对多变量时间序列变量间相关性的有效挖掘，进而提高了模型的预测准确性。

## 1.2 基于深度学习的时间序列预测模型研究现状

随着深度学习技术的快速发展，基于深度学习的时间序列预测模型已经成为研究的热点。下面对这些机器学习和深度学习模型在时间序列预测任务上的重要应用进行总结。自 1951 年自回归积分滑动平均（ARIMA）模型的提出以来，各种时间序列预测模型相继被开发和改进[16] [17]。ARIMA 模型通过结合自回归（AR）、移动平均（MA）以及差分（I）操作，将非平稳时间序列数据转换为平稳时间序列，实现对数据的趋势和季节性变化的有效捕捉。随后，神经网络（Neural Network）[18] 为处理非线性和高维度的时间序列数据提供了新的可能性。神经网络通过模拟人脑的处理方式，能够自动识别数据中的复杂模式和特征。循环神经网络（RNN）[19] 被特别设计来处理序列数据。RNN 通过其内部状态的循环连接，能够保持对历史信息的记忆，这使其格外适合处理时间序列数据。1997 年，两个重要的模型相继出现：支持向量回归（SVR）[20] 和长短期记忆网络（LSTM）[21]。SVR 是基于统计学习理论的一种强大的回归工具，适用于高维和非线性问题。而 LSTM 作为 RNN 的一个变种，通过引入“门”机制解决了传统 RNN 在处理长序列时梯度消失与梯度爆炸的问题，极大地提高了模型对长期依赖关系的捕捉能力。进入

21 世纪后，极限学习机（ELM）[22]的提出为单隐藏层前馈神经网络提供了快速训练的方法。2014 年，门控循环单元（GRU）[23]作为 LSTM 的简化版被引入时间序列预测，同样能够有效处理长期依赖问题，但该模型的参数更少且训练更快。编码器-解码器[24]架构适用于将一个序列转换成另一个序列（seq2seq）的任务，如机器翻译。该架构通常与注意力机制结合使用，允许模型在生成输出时自适应地关注输入序列的不同部分。图神经网络（GNN/GCN）[25]于 2016 年被提出，它通过在图结构数据捕捉节点间的复杂关系，为建模时间序列数据提供了新的视角。最后，Transformer[11]通过自注意力机制解决了 RNN 和 LSTM 在处理长距离依赖时的低效和误差累积的问题，成为了自然语言处理（NLP）领域的一个重要突破，同时也为时间序列预测提供了新的解决方案。这些模型的发展不仅体现了时间序列预测技术的进步，也反映了深度学习领域对于模型结构、算法优化和数据处理技术不断的探索和创新。

### 1.2.1 基于频域的基础预测模型

在时间序列预测领域，基于频域的分析方法为探究数据的周期性和趋势特征提供了一个独特视角，这类方法的核心假设是时间序列可以被分解为不同频率的正弦和余弦波的组合。基于此思想，Schuster 在 1898 年引入的周期图法。周期图最初设计用于在噪声背景下检测和估计已知频率的正弦波分量的幅度[26]。随后，该方法被进一步应用于分析时间序列（尤其是在特定模型拟合后的残差序列），以评估序列的随机性，并探寻可能存在于序列中的、具有未知频率的周期性分量。因而后续的研究中常基于频域内的方法识别时间序列的周期性，并进行时序分量的分解。本节将详细介绍几种利用频域策略的创新方法，并评述其在时间序列预测中的应用和表现。

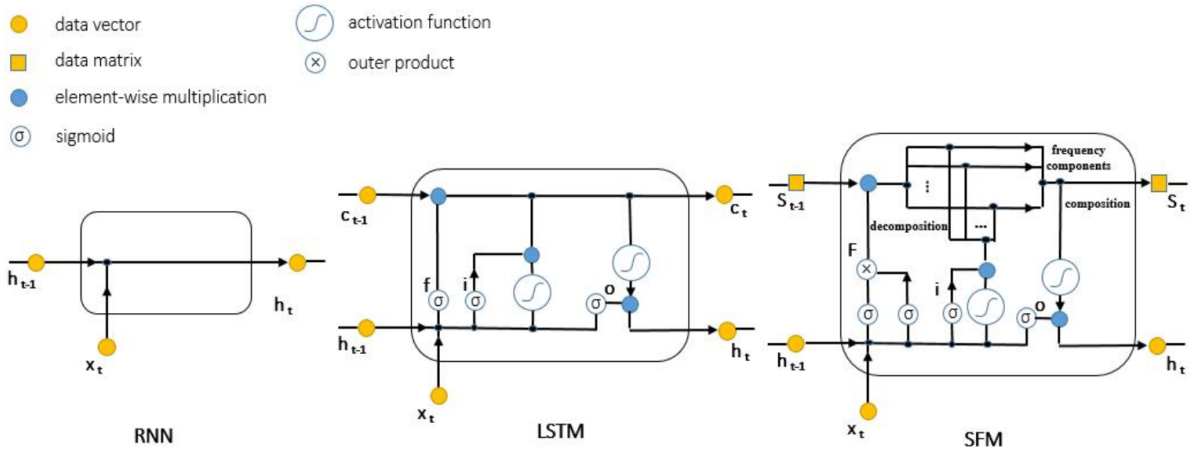



图 1-2 SFM 循环神经网络结构与传统时序建模基础模型 RNN 和 LSTM 的对比图

首先，一种可行的方法是将频域内的时间序列分量分解思想引入传统的经典时间序列预测模型如 RNN[19]和 LSTM[21]中。如图1-2所示，在一篇代表性的工作中，SFM（State Frequency Memory）循环神经网络模型[27]采用了一种独特的方法来处理时间序列

数据，它通过分解隐藏状态为不同的频率组分，揭示和模拟股票价格波动背后的复杂交易模式。这种方法致力于捕捉交易数据的多频率特征，旨在对股价的短期及长期走势做出准确预测。SFM 模型的创新之处在于，它基于离散傅立叶变换（DFT）的原理，对时间序列的隐藏状态进行多频率层面的分析。通过这种分析，每个频率分量都被视为股价波动中某种潜在交易模式的体现。接下来，模型利用逆傅立叶变换（IFT）对这些频率分量进行非线性整合，从而构建出对未来股价变动的预测模型。SFM 的应用于实际股市数据的实验，证明了其在捕获交易模式多频率动态及预测股价方面相比当下其他模型具有明显优势。

使用图结构建模时间序列数据的有效性同样得到了广泛的验证[28] [29]，图神经网络的兴起拓宽了时间序列的建模思路。其中一个代表性的模型是 StemGNN（Spectral Temporal Graph Neural Network)[30]，该模型提出了一种结合图傅立叶变换（GFT）和离散傅立叶变换（DFT）的方法，以同时捕捉时间序列的内部时间依赖性和序列间的相关性。StemGNN 模型是通过堆叠多个 StemGNN 块并使用跳跃连接来构建的，每个 StemGNN 块内嵌了一个谱序列（Spe-Seq）单元到一个谱图卷积模块中，以捕获时间序列的内部时间依赖性和序列间的相关性。StemGNN 块利用图傅立叶变换（GFT）在频谱域实现时间序列的映射，通过特定的图卷积操作来建模序列间的相关性。具体而言，StemGNN 块的构造如下：

$$Z_j = \mathrm{GFT}^{-1}\left(\sum_i \theta_{ij}\mathrm{GFT}(X_i) \odot \mathrm{S}(\Lambda_i)\right) \tag{1-1}$$

其中，GFT 和 $\mathrm{GFT}^{-1}$ 分别代表图傅立叶变换及其逆变换，$\theta_{ij}$ 是针对第 $i$ 个输入和第 $j$ 个输出通道的图卷积核，$\Lambda_i$ 是归一化拉普拉斯矩阵的特征值矩阵，$\mathrm{S}(\Lambda_i)$ 是基于特征值的滤波函数。通过这种方式，StemGNN 块能够为每个输入通道 $X_i$ 生成相应的输出通道 $Z_j$。此外，StemGNN 模型通过在多个通道上应用 GFT 和 Spe-Seq 单元，进一步提高了时间序列预测的精确度。Spe-Seq 单元通过 DFT 将时间序列信号分解为频率域中的实部和虚部，然后分别进行处理。Spe-Seq 单元的运算表示为：

$$M^*(X_i^*) = \mathrm{GLU}(\theta^*(X_i^*), \theta^*(X_i^*)) \odot \sigma^*(\theta^*(X_i^*)) \tag{1-2}$$

其中，$\theta^*$ 是卷积核，$\sigma^*$ 是 sigmoid 激活函数，$\odot$ 表示哈达玛积（逐元素乘法）。这一操作允许模型处理 DFT 输出的实部 $X_i^*$ 和虚部 $X_i^{i*}$，以捕获序列的周期性模式。在 StemGNN 模型中，图卷积是通过三个步骤实现的：首先，多变量时间序列输入通过 GFT 投影到频谱域；其次，通过学习到的核对频谱表示进行滤波；最后，使用逆图傅立叶变换（IGFT）生成最终输出。StemGNN 利用正则化图拉普拉斯矩阵的特征向量构建正交基，从而完成 GFT 的操作。StemGNN 巧妙地将时间序列分析的时域和频域特征结合起来，通过在频谱域内建模这些依赖性，使用卷积和序列学习模块有效地预测清晰的频谱表示。

在频域进行时间序列特征解耦也是一种流行的思路。其中一种方法，CoST（Con-

trastive learning of Seasonal-Trend representation)[31]，采用对比学习方法学习解耦的季节性-趋势表示。CoST 通过时间域和频域对比损失学习区分趋势和季节性表示，实验结果显示 CoST 在多变量基准数据集上显著优于最先进方法，并在各种编码器和回归器选择上表现出鲁棒性。ATFN（Adaptive Temporal-Frequency Network)[32]提出了一种适应性时频网络，结合深度学习网络和频率模式，用于中长期时间序列预测。ATFN 内部采用增强的序列到序列模型学习复杂非平稳时间序列的趋势特征，频域块捕获动态和复杂的周期模式，实验结果表明 ATFN 在长期时间序列预测方面展现出强大的能力。

除了上文提到的结构复杂的深度模型，越来越多的研究者开始将目光集中在轻量且高效的简单结构上，例如 N-beats[33]、LightTS[34]以及 DLinear[35]等。这里介绍 FreTS（Frequency-domain MLP)[36]，该模型探索了在频域应用多层感知机（MLP）进行时间序列预测的新尝试。FreTS 发现：从全局视角，频谱使得多层感知机对信号有一个完整的视角，更容易学习全局依赖关系；从能量压缩的视角，频域多层感知机集中关注具有紧凑信号能量的较小关键频率成分。如图1-3所示，FreTS 采用了两阶段的设计思路，第一阶段堆叠多个频域通道学习器（Frequency Channel Learner）通过提出的频域 MLP 在通道维度上来捕捉序列内的关联性，第二阶段则是堆叠多个频域时序学习器（Frequency Temporal Learner）通过频域 MLP 在时序维度上学习时序依赖关系，进一步促进了通道维度和时间维度两方面的表征学习。在短期和长期预测的 13 个真实世界基准数据集上的广泛实验验证了 FreTS 相较于最先进方法的一致优越性。

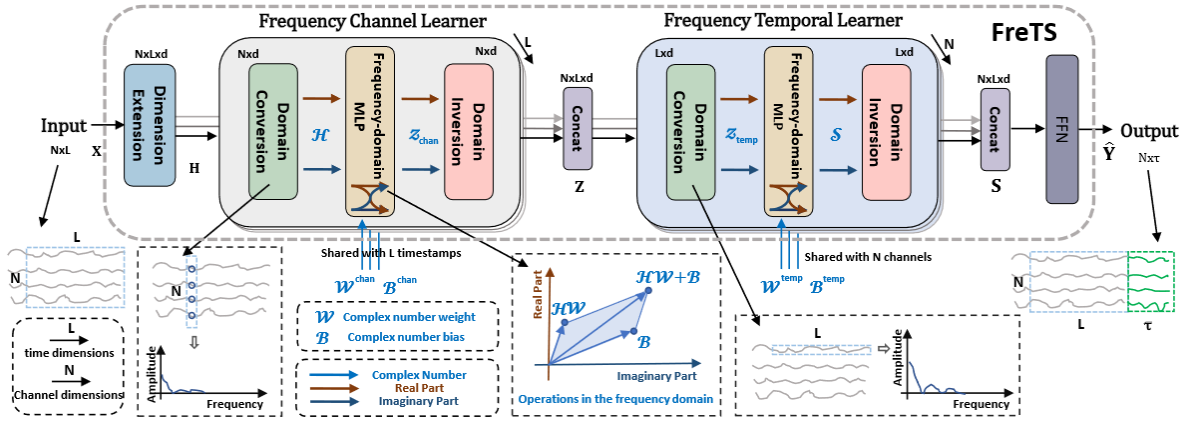



图 1-3 FreTS 结构示意图，包括频域通道学习器和频域时序学习器两个模块

通过上述方法的介绍，不难看到基于频域的时间序列预测模型通过不同的技术策略，有效地提高了预测的准确性和模型的适用性。一系列深度基础模型，如 GNN 和 MLP，都可以通过建模时间序列中的频域特性实现更有效的时序建模。此外，在频域中还可以实现特征分解，通过分离分别代表周期性或趋势性的子序列，深度模型可以有针对性地建模不同特征。频域方法不仅揭示了时间序列数据中潜在的复杂频率成分，也为未来的研究提供了新的方向和思路。

### 1.2.2 基于创新型注意力机制的类 Transformer 预测模型

近年来，Transformer[11]得益于其注意力机制对序列数据长程依赖的捕捉能力，在时间序列预测任务上取得了极大的成功[37]。但是传统的注意力机制在时间和空间复杂度上都是 $O(N^2)$，其中 $N$ 是输入的时间序列长度，这种二次复杂度在建模长度较大的序列时会出现算力瓶颈，很大程度上限制了模型的能力。因此，为了解决计算成本的问题，很多模型专注于通过创新的注意力机制规避传统注意力机制的二次复杂度，进而降低计算开销，增强预测性能。表1-1总结了近年来类 Transformer 模型在时间序列预测任务上的代表性工作[37]，并对比了其计算复杂度，不难看出近年来的代表性工作都在轻量化注意力机制的思路上取得了瞩目的成果。

表 1-1 不同的类 Transformer 时间序列预测模型复杂度对比

| 模型 | 训练 | | 测试 |
|---|---|---|---|
| | 时间 | 空间 | 步数 |
| Transformer[11] | $\mathcal{O}(N^2)$ | $\mathcal{O}(N^2)$ | $N$ |
| LogTrans[38] | $\mathcal{O}(N\log N)$ | $\mathcal{O}(N\log N)$ | 1 |
| Informer[39] | $\mathcal{O}(N\log N)$ | $\mathcal{O}(N\log N)$ | 1 |
| Autoformer[40] | $\mathcal{O}(N\log N)$ | $\mathcal{O}(N\log N)$ | 1 |
| Pyraformer[41] | $\mathcal{O}(N)$ | $\mathcal{O}(N)$ | 1 |
| Quatformer[42] | $\mathcal{O}(2cN)$ | $\mathcal{O}(2cN)$ | 1 |
| FEDformer[43] | $\mathcal{O}(N)$ | $\mathcal{O}(N)$ | 1 |
| Crossformer[14] | $\mathcal{O}(\frac{D}{T_{seq}}N^2)$ | $\mathcal{O}(N)$ | 1 |

Informer 模型[39]引入了一种创新的高效 Transformer 架构，旨在优化长序列时间序列预测的性能。此模型的关键革新之处在于其采用了 ProbSparse 自注意力机制，该机制通过达到 $O(N\log N)$（$N$ 是序列长度）的时间和空间效率的同时，精准地捕捉了序列间的依赖关系。此外，模型通过自注意力蒸馏技术，即通过减少一半的输入层数据量来增强关键信息的关注度，从而有效应对极长序列的处理需求。更进一步，Informer 采用了一种高效的生成式解码器策略，它允许模型通过一次性的前向计算来预测整个长序列，显著提升了推理过程的速度，这些创新点共同提升了模型在长序列预测任务的准确性。

Reformer[44] 通过引入两种技术提升了 Transformer 的效率：一是采用局部敏感哈希来替代点积注意力，将复杂度从 $O(N^2)$ 改进为 $O(N\log N)$；二是使用可逆残差层代替标准残差，仅在训练过程中存储一次激活，而不是 $N$ 次。这使 Reformer 在处理长序列时具有更高的内存效率和速度，同时保持了与 Transformer 模型相媲美的性能。

Pyraformer[41] 提出了一种基于金字塔注意力的新模型，通过探索时间序列的多分辨率表示来实现精确的预测。金字塔注意力模块（PAM）利用跨尺度树结构总结不同分辨率的特征，并通过内尺度邻接连接来模拟不同范围的时间依赖性。Pyraformer 的时间和空间复杂度随序列长度线性增长，而信号遍历路径的最大长度为常数 O(1)，在单步和

长范围多步预测任务中以最少的时间和内存消耗实现了最高的预测准确性。

ContiFormer[45] 针对不规则时间序列的连续时间动态建模提出了一种新颖的连续时间 Transformer 模型。通过将传统 Transformer 的关系建模能力扩展到连续时间领域，并显式整合神经常微分方程（Neural ODEs）的连续动态建模能力与注意力机制，ContiFormer 在多个不规则时间序列建模任务上展现了卓越的建模能力和预测性能。与现有方法相比，ContiFormer 因其放弃了对底层动态的显式封闭形式假设，而具有更好的泛化能力，并能生成连续平滑的输出结果。

综上所述，这些模型通过引入创新的注意力机制和结合频域方法，不仅提高了时间序列预测的准确性和效率，也为处理长序列数据和不规则时间序列提供了新的思路和解决方案。

### 1.2.3 结合频域方法以及创新型注意力的时间序列预测模型

在时间序列预测的研究中，将创新性的注意力机制与频域分析相结合的方法取得了显著的进展。例如，Autoformer[40] 提出了一种新型的分解架构，并引入了自相关机制以实现创新的时间模式识别。如图1-4所示，Autoformer 采取了编码器-解码器结构。该模型不同于传统的 Transformer，通过将序列分解操作作为深度模型的内部嵌入模块，赋予了模型逐步分解复杂时间序列的能力。受随机过程理论的启发，设计的自相关机制基于序列的周期性，在子序列级别进行依赖性发现和信息聚合，其在长期预测方面相比自注意力机制在效率和准确性上都有显著提升，通过六个基准测试数据集在多个实际应用中实现了 38% 的相对性能提升。 FEDformer[43] 则提出了将 Transformer 与季节性趋势分

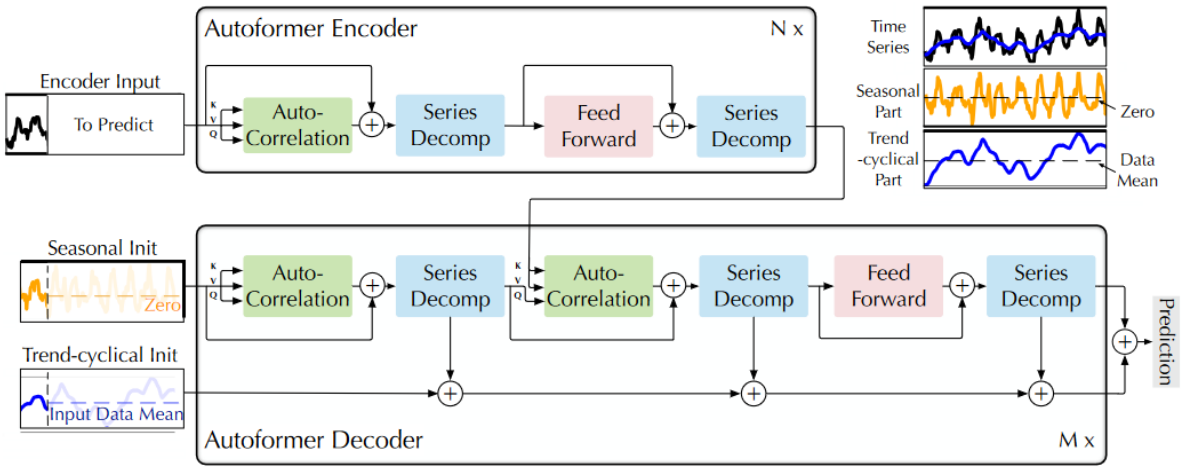



图 1-4 Autoformer 结构图，其中 Encoder 关注季节特征建模，Decoder 则关注趋势特征建模

解方法相结合的模型。为了进一步增强 Transformer 在长期预测中的性能，作者利用了时间序列在傅立叶变换等基础上通常具有稀疏表示的先验知识设计了一种频率增强型 Transformer。与传统的 Transformer 相比，FEDformer 不仅更有效，而且具有线性的序列长度计算复杂度。实验表明，与最先进的方法相比，FEDformer 在多元和单变量时间序

列预测中分别减少了 14.8% 和 22.6% 的预测误差。

此外，ETSformer[46] 利用指数平滑的原理来改进时间序列预测中传统的 Transformer 模型。作者借鉴了时间序列预测中的经典指数平滑方法，提出了新颖的指数平滑注意力（ESA）和频率注意力（FA）机制来替换传统 Transformer 中的自注意力机制，从而提高了准确性和效率。基于这些创新，作者重新设计了 Transformer 架构，加入了模块化的分解块，使其能够学习将时间序列数据分解为可解释的时间序列组成部分。广泛的实验验证了该方法的有效性。

综上，这些方法不仅通过引入频域分析和创新的注意力机制来提高时间序列预测的准确性和效率，而且还展示了众多基于 Transformer 的模型在长期预测和复杂时间序列分解方面的独特优势。本章的后续内容会对上文提到的工作进行总结，并给出本文的时序预测解决方案。

### 1.2.4 研究现状总结

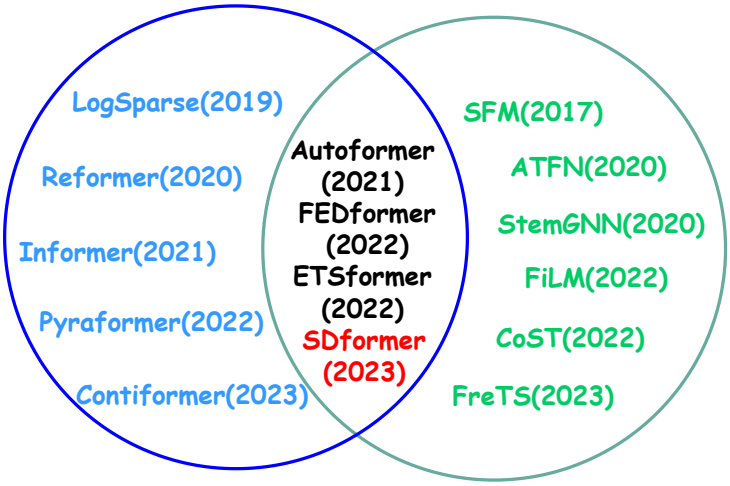



图 1-5 频域方法和创新注意力机制代表性工作概括图

图1-5总结了截至目前在频域方法和创新注意力两个思路的代表性工作，蓝色圈和绿色圈对应了在1.2.1节和1.2.2节介绍了基于频域的基础预测模型和基于创新型注意力机制的类 Transformer 预测模型。其中频域方法使用不同的基础模型挖掘时间序列在频域内的潜在特征，进而帮助模型在时域内进行预测，例如 SFM 将频域内的分量分解结合 LSTM[27]，以及 FreTS 实现了频域 MLP[36]；而创新型注意力方法则是重点集中在解决传统注意力机制中的二次复杂度问题，从而突破传统 Transformer 模型在序列建模过程中的计算资源瓶颈，最大程度发挥其建模时间序列数据的强大能力，例如 Informer[39] 和 Reformer[44] 都在平衡复杂度和预测精度的过程中展现了优越的效果。此外，蓝色圈和绿色圈重合的部分代表了1.2.3节总结的同时兼顾1.2.1和1.2.2中方法的模型，例如 Autoformer[40] 和 FEDformer[43] 都结合了频域内的操作以及特殊的注意力机制来实现时序特征的挖掘。

以上提到的这三种建模思路都在时间序列的研究中起到了至关重要的作用，其中频域内的操作关注挖掘时间序列数据本身的特征，而创新的注意力机制则从 Transformer 模型本身的角度出发，提供了更丰富的时序建模思路。但是这些方法也面临着一些挑战，首先，虽然利用频域建模的方法结合了众多的基础模型，提高了传统基础模型建模时间序列的能力，但它们往往没有考虑潜在噪声的情况下直接利用增强后的数据进行后续学习，从而导致时序特征会因为噪声的存在而变得难以识别，也可能导致时序数据在频率分解过程中不经意地放大了噪声成分，影响了准确的预测。此外，注意力机制的设计都是从平衡计算复杂性和预测性能的角度出发的，现有的注意力机制尚未有效地解决时间序列分析中过于平滑的注意力分布问题，进而忽略了注意力机制无法有效发挥作用导致模型失效的后果。而本文提出的基于时频域特征耦合的时间序列长期预测框架 SDformer 解决了这些被忽视的问题，该模型通过将谱滤波变换和动态定向注意力集成进 Transformer 架构，有效实现了数据去噪与平滑处理，并且增强了注意力机制的表征学习能力，显著增强了多变量时间序列预测的准确性。

## 1.3 本文研究内容及创新点

为了解决1.2.3提到的问题，本文提出了一种创新的基于 Transformer 的模型，名为 **SD**former，它整合了两个特别的模块设计，谱滤波变换（**S**FT）和动态定向注意力（**D**DA），以重新分配注意力权重，增加注意力图的异质性，从而进一步提升预测性能。在技术细节上，在谱滤波变换模块中，本文使用快速傅里叶变换实现滤波，过滤掉包括无意义噪声和波动在内的不重要频率，从而保留时间序列的基本特性，如连续性、周期性和趋势。然后应用一个钟形汉明窗口到过滤后的数据上，在时域内通过其谱特性来最小化边缘效应并增强序列的平滑性。这两个操作有效地通过减少噪声显著提高了数据质量，进一步帮助后续模块实现更有效的表示学习。动态定向注意力模块引入了一个带有动态参数和方向性权重的新颖核函数，同时作用于查询和键上（其中查询和键是从谱滤波变换中平滑序列中线性投射得到的），增加相似的查询-键对的相似度，降低不相似的查询-键对的相似度，从而获得更有区分度的的注意力分数分布。这样的操作使得注意力权重分布在众多变量中更加尖锐，因而使得自注意力机制更能够识别和优先考虑关键的变量间模式，并有效缓解注意力分数分布的“平滑”问题。

如图1-6所示，与 PatchTST、iTransformer 和传统 Transformer 相比，SDformer 的注意力图在一些区域内显示出更集中的颜色分布，表明注意力分数的分布更不均匀，即注意力机制对重要的 Token（本文为变量）具有更强的聚焦能力。此外，本文还计算了注意力矩阵的基尼系数，这为注意力分的集中程度提供了一个清晰的指标[47]。观察到 iTransformer 和传统 Transformer 分别得分 **0.078** 和 **0.081**，而 SDformer 达到了更高的值 **0.154**。这一数值上的差异表明 SDformer 具有更集中的注意力分布，表明其在时间序列中增强对关键变量聚焦的能力，从而缓解了“平滑”问题。总体而言，谱滤波变换和动态

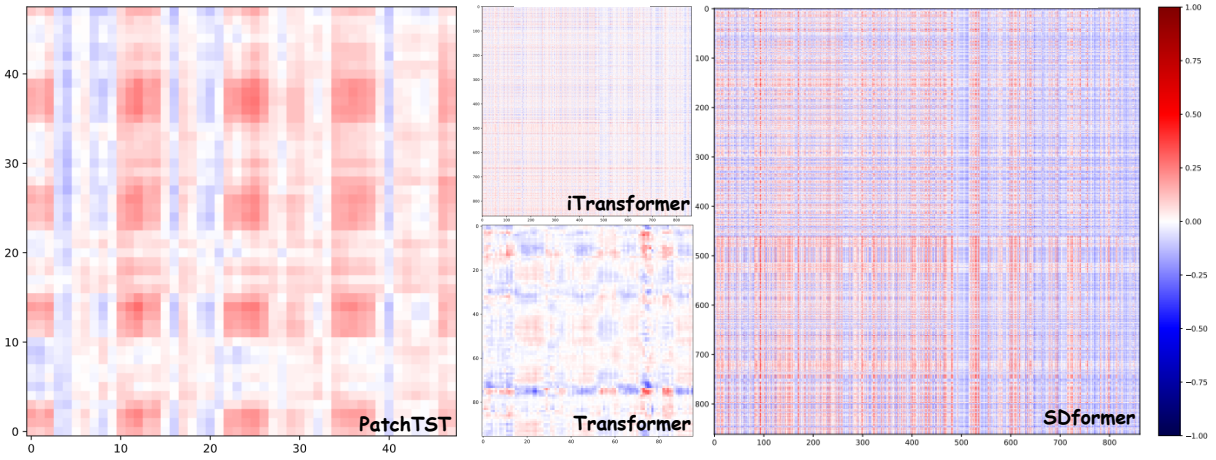



图 1-6 PatchTST、iTransformer、Transformer 和 SDformer 编码器中的注意力图可视化

定向注意力共同克服了传统注意力机制在建模具有大量变量的时间序列时出现的问题，使得该模型对多变量时间序列数据的预测能力更强。此外，本文提出的 SDformer 采用传统 Transformer 模型的编码器-解码器（Encoder-Decoder）结构，但不同的是，SDformer 的解码器仅为单层 MLP，所以可以说 SDformer 是一个编码器（Encoder-only）的结构，这种设计有效降低了传统 Transformer 模型的计算复杂度，大大提高了模型的计算效率。此外，在实现轻量化设计的同时，SDformer 同样兼顾了在时间序列长期预测任务上的卓越表现，在多个公开数据集上的表现超过了现有的最优模型如 iTransformer[12] 和 DLinear[35]。得益于 SDformer 的轻量性设计以及优异的预测表现，该模型对于现实世界中的多变量时间序列预测任务（如气象数据预测[10]等）以及硬件资源受限的场景（如移动端设备等）有较高的应用价值。本文的核心贡献可以归纳为：

(1) 本文提出了一种新颖的 Transformer 架构（命名为 SDformer）用于长期时间序列预测。截至目前，这是**首次**针对建模具有大量变量的时间序列数据时平滑注意力分布问题提出解决方案。

(2) 谱滤波变换和动态定向注意力模块旨在实现滤波效果从而保留关键频率并锐化注意力分布，它们共同作用使注意力权重在变量之间更加突出，进一步提高注意力机制捕获多变量相关性的能力，并提高最终的预测性能。

(3) 在多个数据集上的广泛实验证明了 SDformer 相比其他最先进方法的优越性。特别在一些具有众多变量的数据集上体现了显著的优势，例如在 Traffic 数据集（862 变量）上相较于目前的最优模型实现了 11.6% 的预测误差降低。

## 1.4 论文组织结构

如图1-7所示，本文共计四个章节，组织结构如下：

**第一章 绪论** 第一章介绍本文的研究背景和意义，以及在频域操作和创新注意力机制等方向的代表性工作，总结本文的研究动机、核心创新点并说明论文的组织结构。

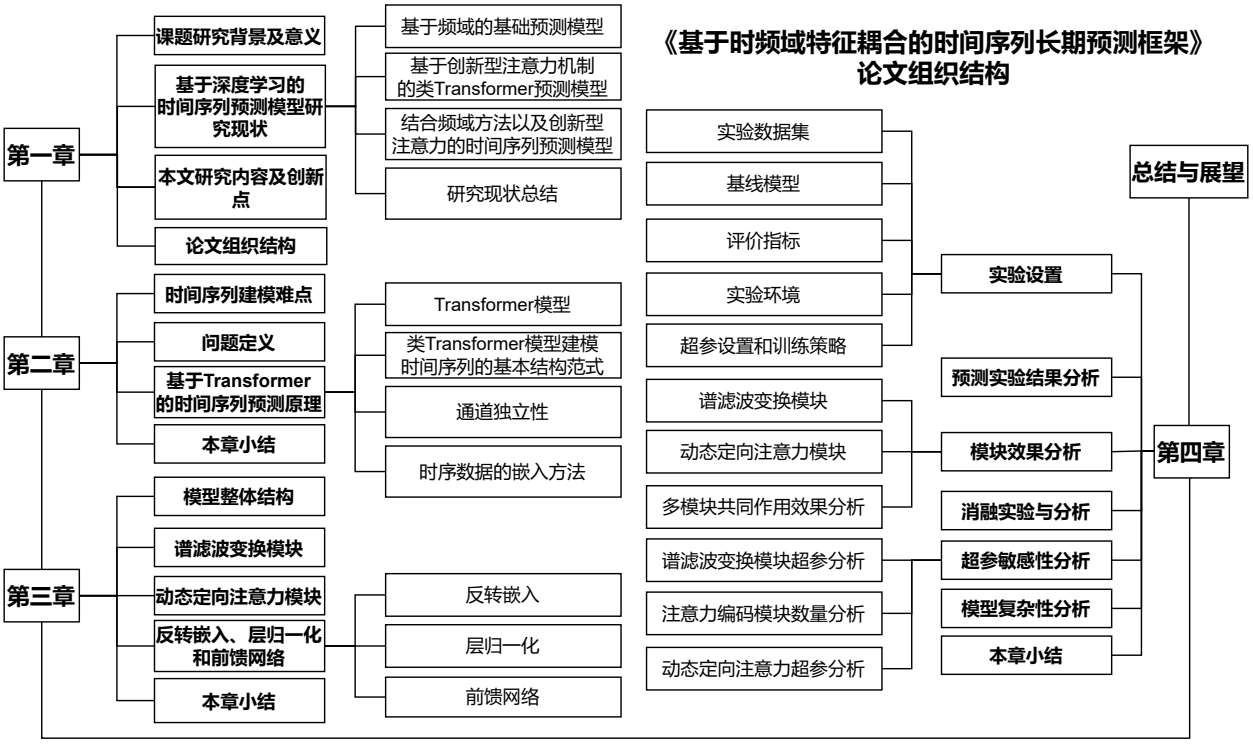



图 1-7 全文组织结构

**第二章 时间序列预测相关理论与方法** 第二章首先介绍时间序列的建模难点，其次介绍时间序列预测的问题定义，最后介绍类 Transformer 模型的结构范式。这些技术将作为重要的理论基础，在本文的后续模型构建过程中提供帮助。

**第三章 基于时频域特征耦合的时间序列长期预测框架** 第三章详细介绍本文提出的基于时频域特征耦合的时间序列长期预测框架的原理与设计思路，其中模块化结构分为谱滤波变换和动态定向注意力，配套的操作包括反转嵌入、层归一化和前馈网络。

**第四章 实验结果与分析** 第四章介绍了本文在时间序列长期预测任务使用的数据集与评价指标，以及验证本文提出模型所对比的其他基线模型。介绍了长期预测实验和消融实验的实验设置，并对结果进行细致分析，还分析了模型中的重要超参数及其对模型效果的影响。最后对模型的轻量性进行了验证。

最后的**总结与展望**部分归纳总结了本文的全部工作内容，分析了本文提出的模型的不足之处和可改进的方向，并对未来工作进行了展望。

# 第 2 章　时间序列预测相关理论与方法

时间序列作为数据挖掘领域最重要的研究方向之一，随着近年来机器学习和深度学习技术的涌现重新吸引了人们广泛的研究兴趣。本章内容集中于介绍时间序列预测问题的相关理论和方法，首先在2.1节介绍时间序列建模过程的难点，接着在2.2节说明时间序列预测的问题定义，其次在2.3节介绍 Transformer 模型的基础架构以及 Transformer 模型建模时间序列的基本范式，最后介绍“通道独立”概念以及使用 Transformer 的不同嵌入方法。

## 2.1　时间序列建模难点

时间序列数据是按特定时间间隔组织的序列，这些数据在早期的统计分析中常用于预测。对于那些具有稳定统计特性的序列，这些序列的均值、方差和自相关结构等在时间上保持不变，传统的统计方法如 ARIMA 取得了显著的效果[16] [17]。但对于动态变化的非平稳序列，就需要采用更复杂的处理方法。这类序列通常需要根据季节性、周期性等因素进行分解，并依据不同的理论假设来构建模型。在实际情况中通常遇到的是非平稳序列，它们的统计属性随时间发生变化。在机器学习领域，处理非平稳时间序列的预测问题被视为一种分布漂移（Distribution Shift）问题，该问题特别被定义为**时序协方差漂移（Temporal Covariant Shift）**[48]。如图2-1所示，在不同的时间段 $A,B,C$ 内，时序数据的分布 $P(x)$ 完全不同，而模型所预测的时间间隔 $D$ 同样具有与过往分布完全不同的分布 $P(D)$，给模型的准确预测造成了影响。

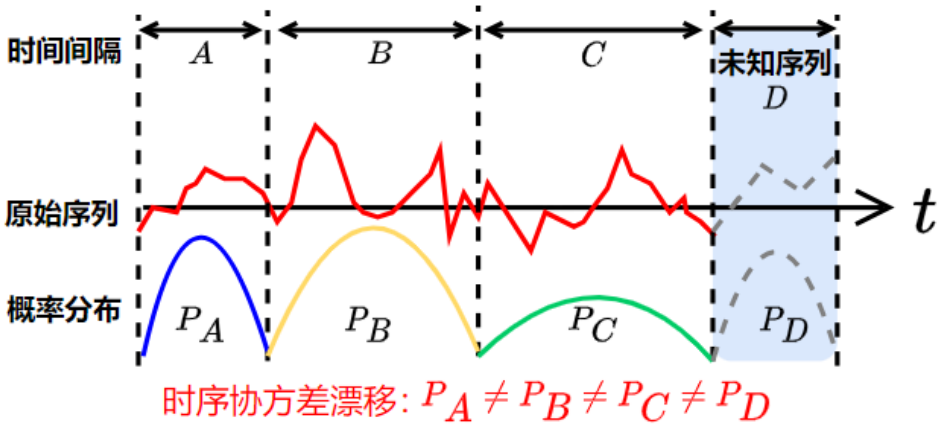



图 2-1 时序协方差漂移问题示意图

对于模型来说，处理那些它在训练期间未曾遇见过的数据分布总是一个挑战，特别是当这些测试数据的分布与训练数据有所不同。模型由缺少处理这类分布的经验，其预测效果可能因此变得不理想。然而在某些情况下，可以假定条件概率 $P(y|x)$ 保持稳定，这在众多应用场景中其实是一个可接受的前提。以股市预测为例，市场的变动可能引起

财经指标 $P(x)$ 的变动，而潜在的经济规则 $P(y|x)$ 即使在这些波动中仍旧被视为恒定不变（这里的 $x$ 和 $y$ 分别代表样本和预测值）。这一假设的变化打破了传统的独立同分布（I.I.D）原则，导致现有的机器学习算法在性能及其泛化能力上的表现不尽人意[49]。

此外，如图2-2中蓝色虚线框所示，对于计算机视觉领域主要建模的数据类型图像来说，某一个局部或区域（Patch）内所包含的语义信息非常丰富，机器学习模型可以通过识别某个局部的特征来捕捉整张图片的关键信息，这种局部信息的丰富性是卷积神经网络在计算机视觉领域发挥功效的基础[50]。对于自然语言，一个句子中的某个词所包含的信息同样十分丰富，甚至对于一个结构复杂的句子只需要关注其中几个单词就可以知道句子所表达的含义。但是对于时间序列数据来说，**某一个点所包含的语义信息非常有限**，仅代表了系统在这一时刻的观测值，而更丰富的信息则包含在较长时间维度中数据随时间变化的过程（Temporal Variation）中[51]。

但是，这种变化的过程同样非常复杂，原因在于时序特征具有**高耦合性**。如图2-2中红色虚线框所示，在图中红色虚线框标出的三个时序片段中，第一个片段总体展现出上升的趋势，但上升的过程中还包含了波动以及一小段下降，第三个片段中总体呈现下降趋势，但下降的过程中还包含了波动和一小段上升的片段。因此，多种时序变化特征耦合在一起使得对于时序数据的建模变得非常困难。

近年来，Transformer 模型在处理时间序列数据方面展现出了显著的潜力。原本设计用于处理自然语言处理任务的 Transformer 通过其自注意力机制能够有效地捕捉序列中的长程依赖关系（Long-term Dependencies），这一特性对于理解和建模时间序列数据中的复杂动态尤为重要。本章的后续内容会详细介绍 Transformer 如何建模时间序列。

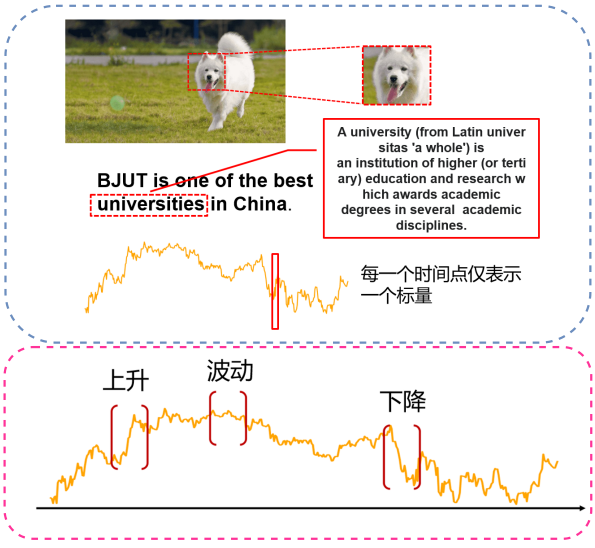



图 2-2 时序数据与图像和自然语言的对比以及时序数据特征的高耦合性

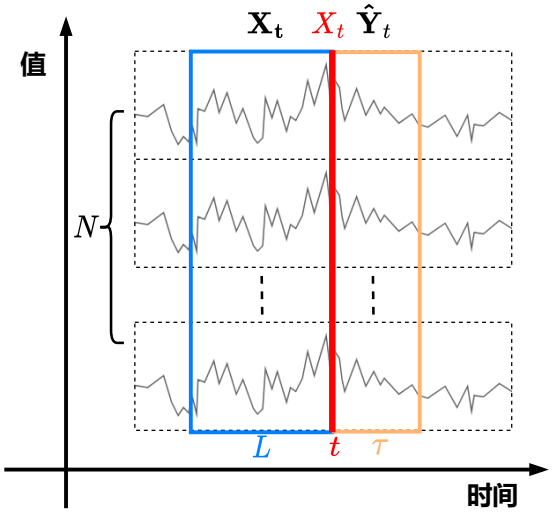



图 2-3 使用滑动窗口进行时间序列预测示意图

## 2.2 问题定义

如图2-3所示，传统时间序列的建模通常使用固定窗口大小的滑动窗口法进行滚动式预测。通常来说，用 $[X_1, X_2, \ldots, X_T] \in \mathbb{R}^{N\times T}$ 来表示一个多变量时间序列，其中每个 $X_t \in \mathbb{R}^N$ 对应于第 $t$ 个时间点的 $N$ 个变量的观测值，共 $T$ 个时间点。对于任何给定的时间 $t$，模型的输入是包括时间点 $t$ 及其前 $L$ 个观测值的时间窗口，表示为 $\mathbf{X_t} = [X_{t-L+1}, X_{t-L+2}, \ldots, X_t] \in \mathbb{R}^{N\times L}$。在时间 $t$ 的时间序列预测目标旨在预测后续的 $\tau$ 个连续的序列值，表示为 $\mathbf{Y_t} = [X_{t+1}, X_{t+2}, \ldots, X_{t+\tau}] \in \mathbb{R}^{N\times \tau}$。预测模型表示为一个映射关系 $f_\theta$，利用历史数据 $\mathbf{X_t}$ 估计未来的值 $\hat{\mathbf{Y}}_t$，预测过程由 $\hat{\mathbf{Y}}_t = f_\theta(\mathbf{X_t})$ 定义。

## 2.3 基于 Transformer 的时间序列预测原理

### 2.3.1 Transformer 模型

Transformer 模型作为近年来自然语言处理领域的一个重大突破，首次由 Vaswani 等人于 2017 年提出[11]。这一模型主要用于处理序列到序列的任务，如机器翻译，其最大的创新之处在于完全依靠自注意力（Self-Attention）机制，摒弃了传统的循环网络和卷积网络结构。这种设计使得 Transformer 模型在处理长距离依赖关系和并行处理方面表现出色，而且 Transformer 格外适合在 GPU 上训练，因而极大地提高了处理效率，使得参数在万亿级别的大型深度神经网络的训练成为可能[52]。

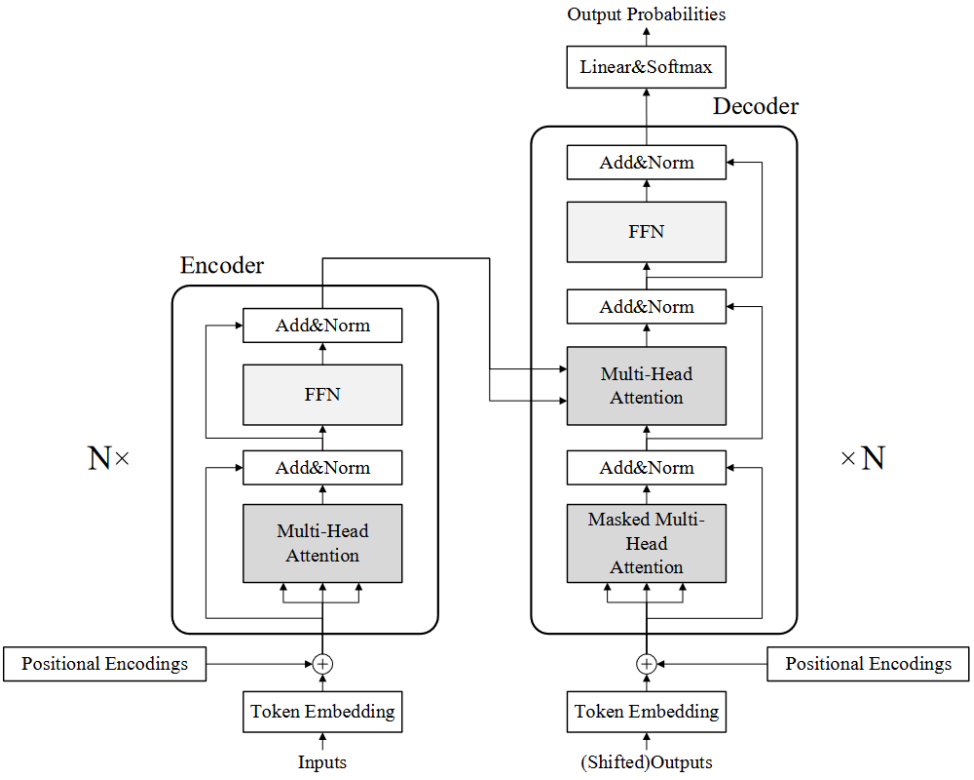



图 2-4 原始 Transformer 模型结构图，由位置编码、编码器和解码器构成

如图2-4所示，Transformer 模型架构主要由两部分构成：一是编码器（Encoder），二是解码器（Decoder），每个部分均由多个一致的模块堆叠而成。在每一个编码器层中，可以观察到两个主要的组成部分：首先是多头自注意力机制（Multi-Head Self-Attention）；其次是位置全连接前馈网络（Position-wise Feed-Forward Network）。多头自注意力机制使得模型能够在不同的子空间中并行处理信息，而位置全连接前馈网络则对每个位置的表示进行独立处理。这种层级结构赋予了 Transformer 模型强大的表示能力。解码器层在编码器层的基础上增加了第三个子层，用于从编码器接收信息并生成最终的输出。通过这种方式，解码器能够有效地集成并利用编码器处理过的信息，从而在序列生成任务中实现更准确的预测。

自注意力机制（Self-Attention Mechanism）是 Transformer 的核心，同时也是传统注意力机制（Attention Mechanism）的变种。注意力机制的本质是模拟人脑注意力的生物学行为，即模型可以关注到对任务更重要的信息[53]，该思想最早被提出用在机器翻译领域来提升 RNN 的性能[54]。随后，注意力机制的特殊能力被重新审视，于是诞生了 Transformer 这个仅使用注意力机制而不借助其他基础模型（如 RNN 的循环单元）的结构。注意力机制的核心是实现模型在处理每个序列的不同位置时能够考虑到整个序列的信息，注意力分数是通过计算查询（$\mathbf{Q}$）、键（$\mathbf{K}$）和值（$\mathbf{V}$）的点积来实现的，具体公式如下：

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D_k}}\right)\mathbf{V} \tag{2-1}$$

其中 $\mathbf{Q}$、$\mathbf{K}$ 和 $\mathbf{V}$ 在自注意力机制中都是由一系列注意力机制的 Token 输入 $\mathbf{X}$ 通过线性变换得到的。$D_k$ 是键的维度，代表了注意力机制中的缩放因子，避免了点积过大导致 Softmax 函数进入饱和区。注意力机制的引入使模型在处理序列时能够灵活地关注到关键信息，有效提高模型的表征学习能力。而多头自注意力机制（Multi-head Self-Attention，MHSA）则进一步扩展了这一概念，它引入了多个并行的“头”来捕捉更全面的信息，通过将查询、键和值映射到多个不同的表示子空间，使模型能够同时从多个角度捕捉信息。这种机制的具体实现如下：

$$\begin{aligned} \text{MHSA}(\mathbf{Q},\mathbf{K},\mathbf{V}) &= \text{Concat}(\text{head}_1,\ldots,\text{head}_h)\mathbf{W}^O \\ head_i &= \text{Attention}(\mathbf{X}\mathbf{W}_i^Q,\mathbf{X}\mathbf{W}_i^K,\mathbf{X}\mathbf{W}_i^V) \end{aligned} \tag{2-2}$$

其中每个头 $\text{head}_i$ 执行独立的注意力运算，而 $\mathbf{W}_i^Q$、$\mathbf{W}_i^K$、$\mathbf{W}_i^V$ 是输入 Token 进行线性变换以调整维度的随机初始化矩阵，$\mathbf{W}^O$ 用来对多个 head 拼接后的结果进行线性变换。

位置编码是 Transformer 模型中的另一个关键组件。由于模型中缺乏对序列顺序的内在理解，位置编码通过向每个输入元素添加一个与其位置相关的信号来解决这个问题，公式如下：

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}}) \tag{2-3}$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}}) \tag{2-4}$$

其中，$pos$ 表示位置，$i$ 表示维度，$d_{\text{model}}$ 是模型中所有层的输出维度。这使得模型能够识别并利用输入序列中各个元素的位置信息。在 Transformer 中，位置编码会直接加在输入向量上，这个直接的操作并不会破坏输入向量的信息，从而并使得网络的任务变得更加困难。这是因为在高维空间中，向量间的干扰和重叠的可能性远小于在低维空间中。因此尽管位置编码被添加到了输入向量中，但神经网络仍然能够通过学习分辨不同的信息类型，有效地处理序列数据，这对于理解语言结构和语义非常关键。

此外，在模型的更深层中，前馈网络和残差网络[55]起到了至关重要的作用。前馈网络是一个全连接的模块，表示为：

$$\text{FFN}(\mathbf{H}') = \text{ReLU}(\mathbf{H}'\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2 \tag{2-5}$$

其中，$\mathbf{H}'$ 是前一层的输出，$\mathbf{W}_1 \in R^{D_m \times D_f}$，$\mathbf{W}_2 \in R^{D_f \times D_m}$，$b_1 \in R^{D_f}$，$b_2 \in R^{D_m}$ 是可训练的参数。而残差连接和层归一化模块被应用于每个子层，以增强模型的学习能力和稳定性，具体表示为：

$$\mathbf{H}' = \text{LayerNorm}(\mathit{SelfAttn}(\mathbf{X}) + \mathbf{X}) \tag{2-6}$$

$$\mathbf{H} = \text{LayerNorm}(\mathit{FFN}(\mathbf{H}') + \mathbf{H}') \tag{2-7}$$

其中，$\mathit{SelfAttn}(.)$ 表示自注意力模块，$\mathit{LayerNorm}(.)$ 表示层归一化操作，该操作可以有效提高模型训练效率[56]。

总而言之，Transformer 模型通过其独特的多头自注意力机制，综合前馈网络、残差连接和位置编码，实现了对序列数据的高效和精确处理。这些模块的有机结合使得 Transformer 在处理复杂的序列到序列任务时，不仅在性能上具有显著优势，同时在模型训练和并行处理方面也显示出极高的效率。

### 2.3.2 类 Transformer 模型建模时间序列的基本结构范式

Transformer 得益于其自身独特的结构，有效地建模长距离依赖问题，在序列建模任务上显示了优越的性能。在自然语言领域，BERT、GPT 等一众基于 Transformer 的杰出工作的不断涌现，一次又一次地证明了 Transformer 在序列建模领域的统治地位[57] [58] [59] [60]。时间序列和自然语言类似的一点是二者均为序列数据，因此 Transformer 在近年来也被研究者广泛应用于时间序列分析领域。图2-5展示了类 Transformer 模型建

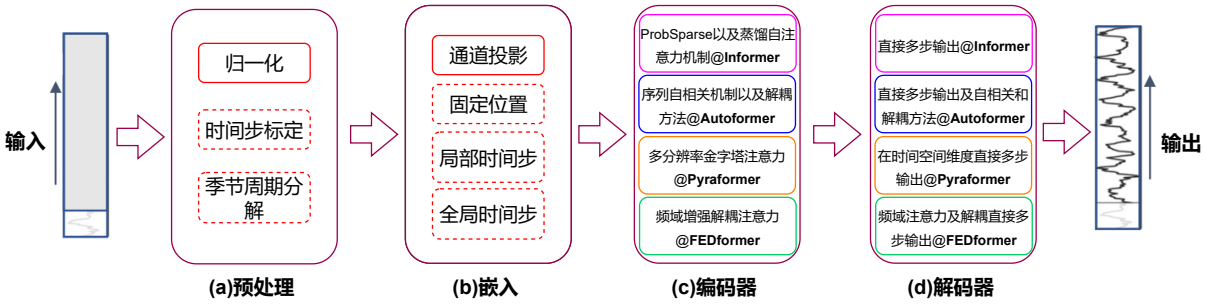



图 2-5 类 Transformer 时序模型结构图：在（a）和（b）模块中，实线框表示基础操作，虚线框则是不同模型可选择性的操作，此外每个模型都有自己具体的（c）和（d）中的实现方法

模时间序列的通用基础范式，从输入序列开始，经过预处理（Preprocessing）、嵌入（Embedding）、编码器（Encoder）和解码器（Decoder），最终得到预期的输出序列。基于2.2节的定义，类 Transformer 模型的输入是一个时间窗口内的观测值，可表示为：

$$\mathbf{X_t} = [X_{t-L+1}, X_{t-L+2}, \ldots, X_t] \in \mathbb{R}^{N \times L} \tag{2-8}$$

其中，$L$ 是窗口长度，$N$ 是变量的数量。模型的目标是基于这个输入窗口预测未来 $\tau$ 个时间步长的值：

$$\mathbf{Y_t} = [X_{t+1}, X_{t+2}, \ldots, X_{t+\tau}] \in \mathbb{R}^{N \times \tau} \tag{2-9}$$

首先，预处理阶段对输入窗口 $\mathbf{X_t}$ 进行预处理:

$$\mathbf{X'_t} = Preprocessing(\mathbf{X_t}) \tag{2-10}$$

最为广泛使用的一种预处理方法是零均值化（zero-mean），从而将数据标准化到均值为 0，标准差为 1 的正态分布。这种方法既可以加快模型通过梯度下降求取最优解的速度，又可以消除模型训练中由于数据的量纲不同、自身变异或者数值相差较大所引起的误差。对于时间序列数据，这种方法最直观的效果是减轻序列的非平稳性[61]，有助于后续的模块捕获时间序列的显著时序特征，如周期性和趋势等。此外，将时间序列数据进行分解也是一种常用的预处理手段，其核心思想是基于一定的先验知识，将时间序列数据分解为两个或多个子序列。例如 Autoformer[40]提出在将数据输入模型之前进行季节性-趋势分解，即在输入序列上使用移动平均窗口来提取时间序列的趋势-周期分量，这两种分量会在模型中经过不同的处理，最后再将两种分量得到的预测结果求和作为模型的输出，以提高数据的可预测性。

然后，嵌入阶段将预处理后的数据转换为一个高维稠密向量，并注入位置和时间信息，以便更好地表示时间序列的特征：

$$\mathbf{E_t} = Embedding(\mathbf{X'_t}) \tag{2-11}$$

Transformer 架构中的自注意力层无法保留时间序列的位置信息[35]。对于传统的时间序列来说，时间戳之间的前后关系可以被认为是局部的位置信息。此外，全局时间信息，

如层次化的时间信息（周、月、年）和一些未知的不固定的时间戳（假期和事件），同样具有很强的信息量，所以传统的时间序列数据嵌入的过程往往会将按“年月日”标定的时间戳进行编码，作为变量添加到原始时间序列数据中。为了更好地保留时间戳之间的先后关系，不破坏时序数据最重要的时序信息，类 Transformer 模型会采用一些行之有效的嵌入方式，如固定位置编码、通道投影嵌入和可学习的时间嵌入到输入序列中。经过嵌入的时序数据会被投影到一个更高的维度，成为类似于自然语言处理中的词元（Token），以便后续的模块学习有意义的表征。最近，一种非常新颖的嵌入方式引起了广泛的关注[12]，该模型采取了一种“反转嵌入”的设计，将多变量时间序列的每一个变量整体嵌入到高维空间，而不是像过往方法一样将每一个时间戳的多个变量值进行嵌入。这种方法能够最大程度上保留变量在时序维度上完整性，同时有助于挖掘变量之间的相关性，在多变量时间序列预测任务上取得了最优效果。

接下来，编码器阶段使用自注意力机制处理嵌入向量 $\mathbf{E_t}$ 来提取时间序列的关键特征，并通过改进自注意力机制优化计算复杂度：

$$\mathbf{C_t} = Encoder(\mathbf{E_t}) \tag{2-12}$$

Transformer 能够建模时序长程依赖主要得益于其特殊的自注意力机制。但是由于自注意力机制是没有依赖关系的，即输入元素会同等地和历史序列中所有元素进行交互，这便导致了 Transformer 模型的时间复杂度和空间复杂度均为 $O(L^2)$。优化的自注意力机制通过引入稀疏性和降低计算复杂度的方法，已成功缓解了计算复杂度过高的问题。一种方法是引入稀疏模式，减少在自注意力计算中涉及的元素对，从而降低了时间和空间复杂度。例如 Pyraformer[41] 通过采用金字塔形式的注意力机制，有效地捕获了多尺度的时间依赖性，同时将复杂度降低到线性级别，即 $O(L)$。FEDformer[43] 则侧重于利用自注意力矩阵的低秩特性，通过引入频域增强的自注意力模块，以进一步减少计算负担，实现了 $O(L)$ 的计算复杂度。总而言之，模型可以通过编码器中的自注意力机制模块挖掘时间序列数据的长程依赖关系，同时也可以通过改进注意力的实现方法规避传统 Transformer 中的二次复杂度问题，提高模型的计算效率。

最后，解码器基于编码器提取的特征来预测未来的时间序列值，采用高效的解码策略以减少推理时间和误差累积：

$$\mathbf{Y_t} = Decoder(\mathbf{C_t}) \tag{2-13}$$

原始 Transformer 解码器以自回归方式输出序列，导致推理速度慢和误差累积效应。其他 Transformer 变体采用类似的直接多步输出策略。例如，Pyraformer[41] 使用一个全连接层连接空间-时间轴作为解码器，而 Autoformer[40] 通过叠加两个趋势-周期分量和堆叠的自相关机制来获取季节分量，以得到最终预测。此外，最近的研究也发现仅使用单层 MLP 作为解码器同样可以达到较好的效果[12]，即仅有编码器的 Transformer 结构同样可以有效地完成时间序列预测任务。

综上，类 Transformer 模型借助经典 Transformer[11]架构的独特设计，结合时间序列数据本身的特点进行有针对性的改进，在建模时间序列的过程中取得了引人瞩目的优越效果。

### 2.3.3 通道独立性

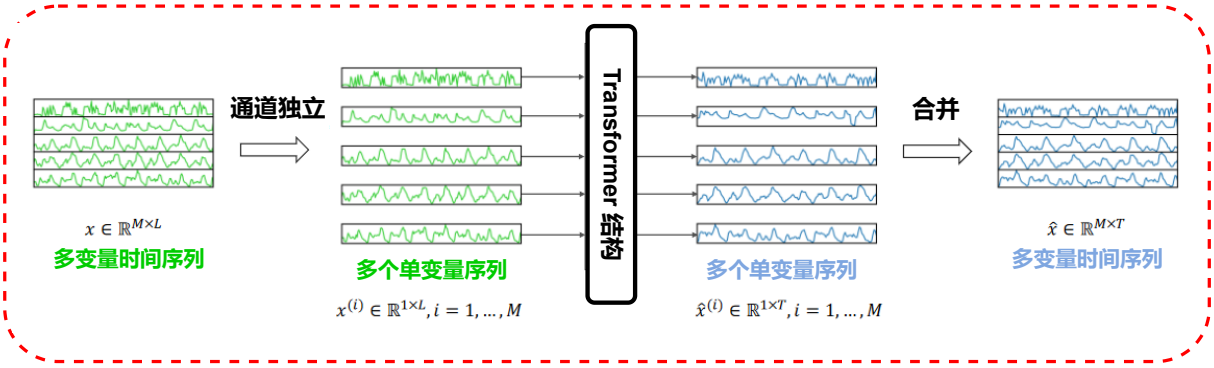



图 2-6 通道独立设计示意图

通道独立设计(Channel-independence)区别于以往的通道融合设计(Channel-mixing)，前者将一个多变量时间序列当作若干个单变量时间序列，单变量时间序列（通道）之间彼此独立，而后者则将多变量时间序列当成一个整体，通道间存在交互关系。通道独立的设计在时间序列领域曾被证明在其他基础模型如 CNN[62]、MLP[35]和 Transformer[13]上行之有效。如图2-6所示，具体来说将一个起始时间索引为 1 的长度为 $L$ 的单变量序列表示为：

$$x_{1:L}^{(i)} = (x_1^{(i)}, \ldots, x_L^{(i)}) \quad i = 1, \ldots, M \tag{2-14}$$

输入 $(x_1, \ldots, x_L)$ 被分割为 $M$ 个单变量序列 $x^{(i)} \in \mathbb{R}^{1 \times L}$，每个序列独立地输入到 Transformer 主干中进行处理。在通道独立设置中，Transformer 主干结构相应地提供预测结果：

$$\hat{x}^{(i)} = (\hat{x}_{L+1}^{(i)}, \ldots, \hat{x}_{L+T}^{(i)}) \in \mathbb{R}^{1 \times T} \tag{2-15}$$

这种通道独立的设计放弃了探索多变量时间序列中变量之间的相关性，而将更多的注意力放在探索每个通道内独特的时序特征，使得每个通道拥有不同的注意力矩阵，在有监督和无监督任务上取得了良好的效果[63]。

### 2.3.4 时序数据的嵌入方式

当前的类 Transformer 模型主要采用三种嵌入的方式[12]，如图2-7所示。首先，传统的 Transformer 模型使用“时间戳嵌入（Temporal Embedding）”，将多变量时间序列的所有变量值在某一时刻的观测嵌入到高维空间；iTransformer 采用“反转时间戳嵌入（Invert Embedding）”，将多变量时间序列的其中一个变量整体嵌入到高维空间，通过扩展感受

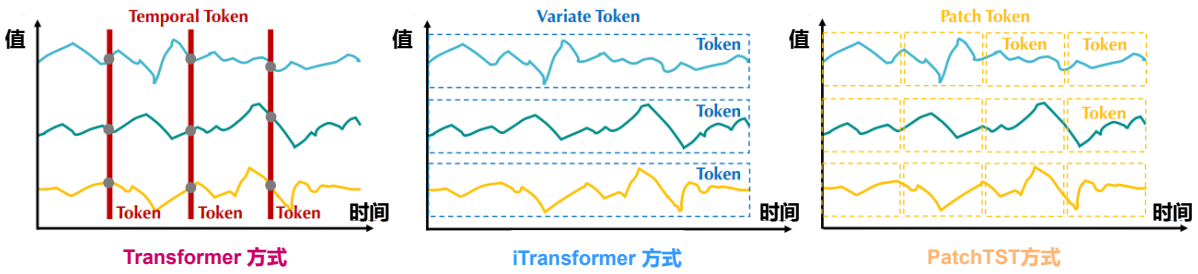



图 2-7 三种主流的嵌入方式示意图

野来应对多维序列中时间对齐嵌入可能引起的问题；此外，Patching 思想[13] [14]所引入的“分块嵌入（Patch Embedding）”是将每个单变量时间序列划分为不同的区域（Patch），该方法源自于计算机视觉领域将图片按 Patch 划分的思路[64]，将局部的一段子序列编码为 Token，进而为模型提供更精细的局部特征捕获能力。综合考虑这三种嵌入方式的优缺点后，本文选择了“反转时间戳嵌入”的策略，后文称其为“反转嵌入”，将时间序列的每一个变量编码为单独的 Token，利用注意力机制挖掘变量间的关联关系，进而提高时序预测的准确率。

## 2.4 本章小结

本章介绍了时间序列预测的基本原理和相关知识。首先在2.1节介绍了时间序列数据建模的难点，以及时间序列数据和计算机视觉领域和自然语言处理领域研究的数据的区别；随后在2.2节介绍了多变量时间序列预测问题的具体定义；接着在2.3介绍了传统 Transformer 模型结构以及基于 Transformer 的时间序列预测模型的基本范式，此外还介绍了“通道独立”的概念以及三种主流的嵌入方式。

# 第 3 章　基于时频域特征耦合的时间序列长期预测框架

本章主要介绍基于时频域特征耦合的时间序列长期预测框架 SDformer，该模型的设计基于传统 Transformer 的编码器-解码器（Encoder-Decoder）结构，但不同的是解码器部分仅使用多层感知机（MLP），在保证模型对于多变量时间序列数据的强大建模能力的同时也兼顾了模型尽可能轻量化的设计原则。本章首先在3.1节介绍模型的整体结构和各个组成模块所实现的功能；然后在第3.2和第3.3节详细说明谱滤波变换模块和动态定向注意力模块；接着在第3.4节介绍模型中引入的反转嵌入策略、层归一化方法以及用于表征学习的前馈网络。

## 3.1　模型整体结构

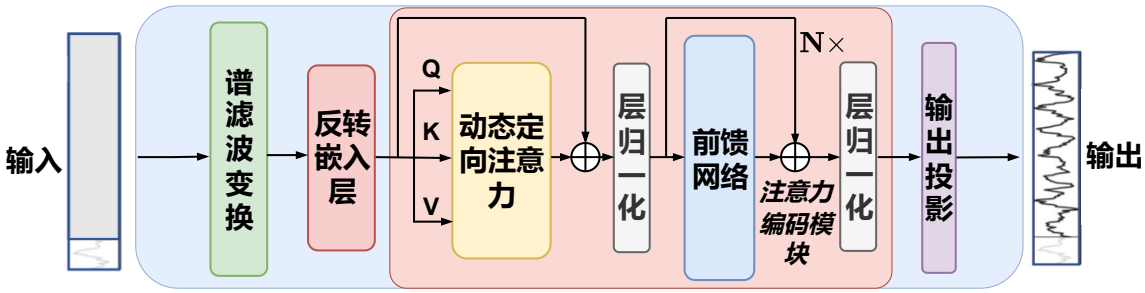



图 3-1 SDformer 整体结构图，在 Transformer 的 Encoder 基础上整合了创新的模块设计

图3-1展示了 SDformer 的整体框架，而算法3-1则宏观地给出了 SDformer 的模块间执行流程，包括谱滤波变换（Spectral-Filter-Transform）模块、反转嵌入、堆叠的动态定向注意力（Dynamic-Directional-Attention）编码器模块（DDAEncoder）以及最后的输出投影操作。整体上看，SDformer 中每个预测序列 $\hat{\mathbf{Y}}_{:,n}$ 的预测过程基于回顾序列 $\mathbf{X}_{:,n}$，可以简单地表示为：

$$\begin{aligned}
\mathbf{X}f_{:,n} &= \text{Spectral-Filter-Transform}(\mathbf{X}_{:,n}), \\
\mathbf{H}_n^0 &= \text{Invert Embedding}(\mathbf{X}f_{:,n}), \\
\mathbf{H}^{l+1} &= \text{DDAEncoder}(\mathbf{H}^l), \qquad\qquad l = 0, \ldots, L-1, \\
\hat{\mathbf{Y}}_{:,n} &= \text{Projection}(\mathbf{H}_n^L),
\end{aligned} \tag{3-1}$$

其中 $\mathbf{H} = \{h_1, \ldots, h_N\} \in \mathbb{R}^{N \times D}$ 包含 $N$ 个嵌入的 Token，每个 Token 的维度为 $D$，$\mathbf{H}$ 的上标表示层索引。反转嵌入（Invert Embedding）：$\mathbb{R}^T \to \mathbb{R}^D$ 和 投影（Projection）：$\mathbb{R}^D \to \mathbb{R}^S$ 都是通过多层感知机 (MLP) 实现的。所获得的变量 Token 通过每个注意力编码模块中的共享前馈网络相互作用，并且是独立处理的。具体来说，序列中元素的时间先后顺序隐式地存储在每个前馈网络的神经元排列中，因此在 SDformer 不再需要位置编码嵌入。

算法 3-1: SDformer 整体流程

输入： 固定长度的回看窗口 $\mathbf{X} \in \mathbb{R}^{T \times N}$；窗口长度为 $T$；时间序列变量个数 $N$；预测长度 $S$；嵌入维度 $D$; 注意力编码模块数量 $L$。

输入序列 $\mathbf{X}$ 首先进入谱滤波变换模块得到降噪平滑后的序列 $\mathbf{X}_f$

$\mathbf{X}_f = \text{Spectral-Filter-Transform}(\mathbf{X})$ $\{\mathbf{X}_f \in \mathbb{R}^{T \times N}\}$

降噪平滑后的序列 $\mathbf{X}_f$ 进行反转嵌入得到注意力编码模块的输入 $\mathbf{X}'$：

$\mathbf{X}' = \mathbf{X}_f^{\mathsf{T}}$ $\{\mathbf{X}' \in \mathbb{R}^{N \times T}\}$

使用 MLP 将 $\mathbf{X}'$ 的最后一维投影到 $D$，获得 $D$ 个变量 Token：

$\mathbf{H}^0 = \text{MLP}(\mathbf{X}')$ $\{\mathbf{H}^0 \in \mathbb{R}^{N \times D}\}$

循环堆叠 $L$ 个注意力编码模块，$\mathbf{H}^0$ 通过注意力编码模块进行表征学习：

**for** $l = 1$ **to** $L$ **do**

  动态定向注意力挖掘变量 Token 间的相关性，层归一化增强数据平稳性：

  $\mathbf{H}^{l-1} = \text{LayerNorm}(\mathbf{H}^{l-1} + \text{Dyn-Directional-Attention}(\mathbf{H}^{l-1}))$ $\{\mathbf{H}^{l-1} \in \mathbb{R}^{N \times D}\}$

  使用前馈网络获取时序数据的表征，通过层归一化减少数据分布差异：

  $\mathbf{H}^{l} = \text{LayerNorm}(\mathbf{H}^{l-1} + \text{Feed-Forward}(\mathbf{H}^{l-1}))$ $\{\mathbf{H}^{l} \in \mathbb{R}^{N \times D}\}$

  得到注意力编码模块的输出 $\mathbf{H}^l$。

**end for**

使用简单的 MLP 作为解码器，将编码器得到的表征投影到预测长度 $S$：

$\hat{\mathbf{Y}} = \text{MLP}(\mathbf{H}^L)$ $\{\hat{\mathbf{Y}} \in \mathbb{R}^{N \times S}\}$

将 $\hat{\mathbf{Y}}$ 转置以匹配模型正确的输出形状：

$\hat{\mathbf{Y}} = \hat{\mathbf{Y}}^{\mathsf{T}}$ $\{\hat{\mathbf{Y}} \in \mathbb{R}^{S \times N}\}$

**return** $\hat{\mathbf{Y}}$

SDformer 的具体框架如算法3-1所示。具体实现方面，输入的时间序列 $\mathbf{X}$ 首先在谱滤波变换模块（Spectral-Filter-Transform）中去噪并增加平滑性，得到处理后的序列 $\mathbf{X}_f$。之后 $\mathbf{X}_f$ 的 $N$ 个变量会被独立嵌入到 $N$ 个独立的 Token 中，同时其时间维度从 $T$ 被投影到 $D$。这些 Token 会被送入堆叠的注意力编码器块（DDAEncoder）中进行初步的表征学习，其中动态定向注意力通过其独特的核函数更有效地探索变量间的相关性，同时利用层归一化（LayerNorm）和带有残差连接的前馈网络（Feed-Forward）学习时间依赖性并减轻序列的非平稳性。最后一层注意力编码模块会得到输出 $\mathbf{H}^l$，这个输出是由 $L$ 个 $\mathbf{H}^{l-1}$ 更新得到的。最后，SDformer 通过一个简单的线性投影操作（MLP）对编码器提取到的表征 $\mathbf{H}^l$ 进行解码，得到最终结果 $\hat{\mathbf{Y}}$。根据预测长度的不同，最后的线性投影操作拥有不同的投影维度。

本章的后续内容将详细介绍两个创新的模块化设计：谱滤波变换和动态定向注意力，以及类 Transformer 建模时间序列的常用设计，如层归一化和前馈网络，并解释它们如何有效地帮助建模多变量时间序列。此外，SDformer 的设计动机一方面是希望拓展 Transformer 模型在时间序列长期预测任务的上限，成为最新的效果最好的 SOTA 模

型，另一方面则是致力于解决传统自注意力机制中注意力分数分配过于均匀的问题，即注意力矩阵的同质化。正是解决了这个之前没有被关注的问题，SDformer 才可以在多个数据集，尤其是变量数非常多的数据集上取得优异的表现，成为该领域内的最新 SOTA 模型。因此本章在介绍模块具体实现方法的过程中也会具体分析 SDformer 是如何解决该问题，从而实现更有效的时间序列建模。

## 3.2 谱滤波变换模块

谱滤波变换模块在模型开始阶段对输入时间序列数据进行去噪和平滑处理中扮演了关键角色。它通过两阶段跨域过程实现这种数据增强，如算法3-2所示，这两阶段分别是：频域去噪（步骤 7、9 和 11）和时域平滑（步骤 13 和 15）。值得注意的是，在进行频域去噪之前，本文将多变量时间序列视为多个单变量时间序列（受到 PatchTST 通道独立设计的启发[13]），在谱滤波变换中分别独立地处理它们，并最终将所有处理过的单变量序列合并起来用于后续操作。

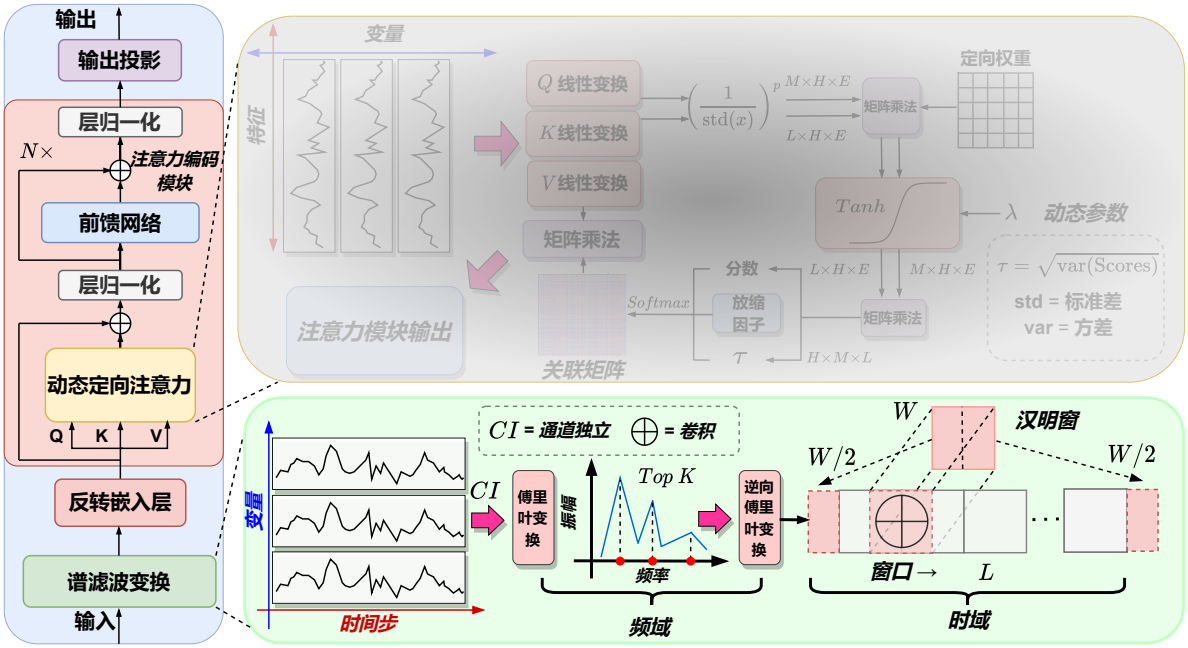



图 3-2 谱滤波变换模块结构图

在时间序列分析中，去噪是提高预测准确性的关键，往往是通过减轻噪声对周期或趋势模式识别的影响来实现。本文提出了一种新的去噪策略，将时间序列转换到其对应的频域，通过滤波的方式减轻时间序列中的随机环境噪声和白噪声。具体来说，谱滤波变换首先利用快速傅立叶变换（FFT）将长度 $T$ 的单变量时间序列 $x \in \mathbb{R}^T$ 转换到其对应频域 $X_f$，即：

$$X_f = \sum_{t=0}^{T-1} x[t] e^{-\frac{2\pi i}{T} kt}, \quad k = 0, \ldots, T-1 \tag{3-2}$$

算法 3-2: 谱滤波变换模块

**输入:** 多变量时间序列 $\mathbf{X} \in \mathbb{R}^{T\times N}$, 输入长度 $T$, 变量数为 $N$。

**输出:** 降噪并平滑后的序列 $\mathbf{X}_h \in \mathbb{R}^{T\times N}$。

**初始化:** 大小为 $w$ 的汉明窗 $w_n$，最显著的频率数量为 $k$。

将变量数为 $N$ 的多变量时间序列考虑为 $N$ 个单变量时间序列，每个单变量时间序列执行相同的操作（通道独立）:

**for** $n = 1$ **to** $N$ **do**

  将时间序列数据 $(x_n)$ 从时域转化到频域 $\mathbf{X}_{f_n}$:

  $\mathbf{X}_{f_n} = \mathrm{FFT}(x_n)$ $\{\mathbf{X}_{f_n} \in \mathbb{R}^{T\times 1}\}$

  在频域内可以获得幅值与频率的准确对应关系，实现滤波:

  $\mathbf{X}_{f_{kn}} = \mathrm{TopK}(\mathbf{X}_{f_n}, k)$ $\{\mathbf{X}_{f_{kn}} \in \mathbb{R}^{T\times 1}\}$

  将时间序列数据 $(\mathbf{X}_{f_{kn}})$ 从频域转化到时域 $x_{if_n}$:

  $x_{if_n} = \mathrm{IFFT}(\mathbf{X}_{f_{kn}})$ $\{x_{if_n} \in \mathbb{R}^{T\times 1}\}$

  在时域中通过加窗操作增强数据的平滑性:

  $x_{p_n} = \text{Reflective Padding}(x_{if_n}, w_n)$ $\{x_{p_n} \in \mathbb{R}^{(T+w)\times 1}\}$

  窗函数作为固定尺度为 $w$ 的卷积核对序列数据进行卷积:

  $x_{h_n} = \text{Applying Window}(x_{p_n}, w_n)$ $\{x_{h_n} \in \mathbb{R}^{T\times 1}\}$

**end for**

$\mathbf{X}_h = \mathrm{Concat}(x_{h_n})$ {拼接 $N$ 个单变量时间序列得到 $\mathbf{X}_h \in \mathbb{R}^{T\times N}$}

**Return:** $\mathbf{X}_h$

然后保留最高 $k$ 频率分量来过滤掉不显著的频率，即 $X_{f_k} = \mathrm{TopK}(X_f, k)$，其中 $\mathrm{TopK}(X_f, k)$ 表示选择 $k$ 个最大振幅，$k$ 是超参数。之后，通过采用逆快速傅立叶变换（IFFT）将滤波后的频谱 $X_{f_k}$ 转换回其对应的时域 $x_{if}$，方便后续进一步的分析，即:

$$x_{if} = \frac{1}{T}\sum_{k=0}^{T-1} X_{f_k} e^{\frac{2\pi i}{T}kt}, \quad t = 0, \ldots, T-1 \tag{3-3}$$

以上简单而有效的操作保留了输入时间序列中的主要时序模式，如连续性或趋势，从而在不被噪声影响的情况下帮助后续操作更好地挖掘变量间关联性。

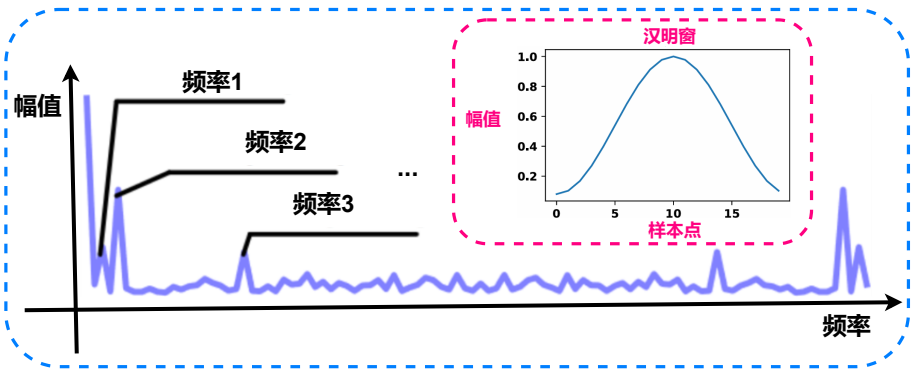



图 3-3 频域内幅值-频率对应关系以及汉明窗函数示意图

通过频域中的滤波消除噪声后，SDformer 通过在时域实施平滑操作进一步处理时间序列。此处，本文使用汉明窗口在时域实现对序列的平滑操作以减轻频谱泄露效应[65]。通过窗函数的卷积操作使得时间序列在边界处的连接变平滑，以减少不连续性。具体来说，首先定义一个大小为 $w$ 的钟形汉明窗口（如图3-3所示），该窗函数的具体定义为：

$$w[n] = 0.54 - 0.46\cos\left(\frac{2\pi nw}{w}\right), \quad n = 1, \ldots, w, \tag{3-4}$$

窗大小 $w$ 设置为偶数以便于后续实现等长的对半切分，$n$ 表示窗口内样本点的索引。随后，根据公式3-3得到的去噪序列 $x_{if}$ 会被窗函数反向填充以确保其长度与基于窗口大小 $w$ 的后续操作匹配：

$$x_p[n] = \begin{cases} x_{if}\left[\frac{w}{2} - n\right], & 1 \le n \le \frac{w}{2} \\ x_{if}[n - \frac{w}{2}], & \frac{w}{2} < n \le T + \frac{w}{2} \\ x_{if}[T + w - n], & T + \frac{w}{2} < n \le T + w \end{cases} \tag{3-5}$$

根据公式3-5，汉明窗的前 $\frac{w}{2}$ 和后 $\frac{w}{2}$ 窗口会被填充在过滤序列 $x_{if}$ 上，作为其最开始的 $\frac{w}{2}$ 长度和最后的 $\frac{w}{2}$ 长度。这样的操作将序列 $x_p[n]$ 扩展到新的长度 $T+w$，其中 $T$ 是去噪序列 $x_{if}$ 的总长度，从而适应后续的卷积平滑操作，即：

$$x_h[t] = \frac{\sum_{n=1}^{w} x_p[t+n] \cdot w[n]}{\sum_{n=1}^{w} w[n]}, \quad t = 1, \ldots, T, \tag{3-6}$$

通过在每个时间点 $t$ 取序列的加权平均来计算平滑序列 $x_h[t]$，可以理解成使用一个大小为 $w$ 的一维卷积核对长度为 $T+w$ 的单变量时间序列进行卷积。经过上述操作后，谱滤波变换在最后将所有 $N$ 个单变量时间序列 $x_h$ 在变量维度拼接起来，得到一个多变量时间序列 $\mathbf{X}_h \in \mathbb{R}^{T \times N}$，其长度为 $T$ 变量数为 $N$，其中 $\mathbf{X}_h$ 保持与输入谱滤波变换模块相同的形状（$T \times N$），这种设计有效避免了因为张量形状的显著变化导致的数据信息损失，同时有效地实现了时序数据的去噪和平滑。

综上所述，谱滤波变换通过在“频域内滤波”和“时域内平滑”两步骤操作，在保留输入序列主要时序特征的同时减少了复杂的波动，还通过加窗的操作进一步增强了数据的平滑性，让时间序列的连续性和趋势更加显著。本文将在4.3节详细展示并分析该模块的具体效果。

## 3.3 动态定向注意力模块

注意力机制是 Transformer 的核心，传统的自注意力机制（Vanilla Self-Attention）中注意力分数的定义如下：

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{3-7}$$

其中 **Q**、**K** 和 **V** 分别代表查询、键和值矩阵，$d_k$ 表示键向量的维度。Softmax 操作应用于结果矩阵的最后一个维度。该机制计算查询和键之间的点积，随后通过键维度的逆平方根进行缩放，并应用 Softmax 函数获得注意力权重，用这些权重对值矩阵进行加权。然而，当 Token 的数量非常多时，本文发现传统的自注意力机制有时不能有效地挖掘 Token 之间的潜在关联关系，对于 Transformer 来说，即模型无法准确识别历史观测中的哪个时间戳是重要的，进而影响了模型对长程依赖的建模。这种能力的缺陷在注意力机制中表现为注意力矩阵出现了严重的行同质化现象，即注意力分数的分布过于均匀，而基于本文选择的反转嵌入设计，传统的注意力机制在这种情况下便不能充分挖掘变量间的关联性。

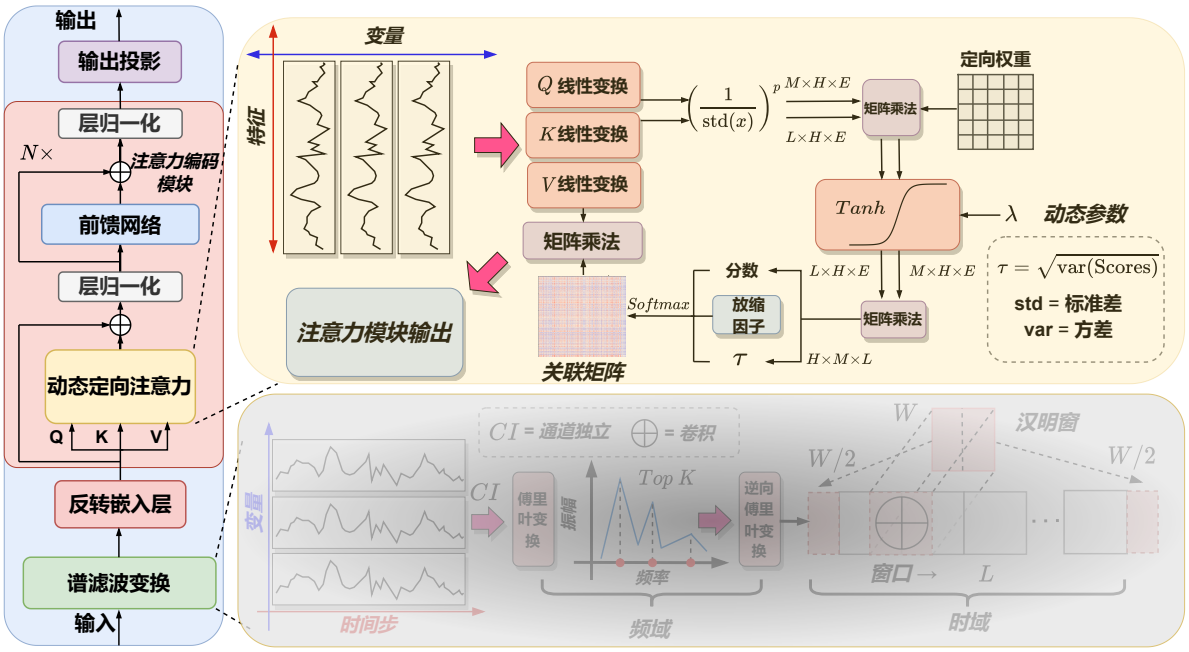



图 3-4 动态定向注意力模块结构图

为了捕获更准确的多变量相关性，本文引入了一种新的动态定向注意力机制来实现更准确、更有区分度的注意力分布。如图3-4所示，动态定向注意力机制的原理是同时作用一个核函数在查询和键向量上，实现动态的重定向并缩放查询 $\mathbf{Q} \in \mathbb{R}^{M\times H\times E}$ 和键 $\mathbf{K} \in \mathbb{R}^{L\times H\times E}$，其中 $M$ 和 $L$ 是序列长度，$H$ 是头的数量，$E$ 表示每个头的维度。动态定向注意力中每个头的注意力得分 $Score$ 表示为：

$$Score(\mathbf{Q}_i, \mathbf{K}_j) = \phi_p(\mathbf{Q}_i)\phi_p(\mathbf{K}_j)^T, \tag{3-8}$$

其中 $\phi_p(x) = f_p(\tan(x))$ 是同时应用于查询和键的特别设计的函数。这里，$\phi_p(x)$ 由非线性映射 $tan(x)$ 和特殊核函数 $f_p$：

$$f_p(x) = x \cdot w_{\text{dir}} \cdot (\text{std}(x))^{-p} \cdot \lambda_{\text{dyn}} \tag{3-9}$$

定义，其中定向系数 $p$ 是元素级的幂，$w_{\text{dir}}$ 和 $\lambda_{\text{dyn}}$ 是代表方向权重和动态参数的可学习

参数，std($x$) 代表输入 $x$ 的标准差。此外，动态定向注意力引入了动态缩放因子 $\tau$ 来计算注意力权重 $\mathbf{A}$：

$$\mathbf{A} = \text{Dropout}\left(\text{Softmax}\left(\frac{scale \cdot Score}{\tau}\right)\right), \tag{3-10}$$

其中 $\tau$ 可以被理解为 Softmax 运算中常引入的温度系数，调节 Softmax 曲线的平滑程度，其计算方法为 $\tau = \sqrt{\text{var}(Score)}$，var($Score$) 是计算得分（$Score$）的方差，$scale$ 是作用 Softmax 前的缩放因子，这个超参数统一定义为 $\frac{1}{\sqrt{E}}$，$E$ 是查询和键矩阵的最后一个维度值。作用 Softmax 后跟随一个 Dropout 过程，通过在训练阶段随机将一部分权重归零来正则化注意力权重 $\mathbf{A}$，减少过拟合。最后，该模块的输出 **Output** 由注意力分数对值矩阵的加权和计算得到：

$$\mathbf{Output} = \sum_{s} \mathbf{A} \cdot \mathbf{V}. \tag{3-11}$$

动态定向注意力的核心在于其特殊的核函数 $f_p$，该核函数通过同时作用在查询向量和键向量上，进而直接改变了二者点积计算注意力分数的值，起到的作用是有效地增强了所有键值对的注意力分布对比度，从而缓解了注意力分数分布平滑的问题。

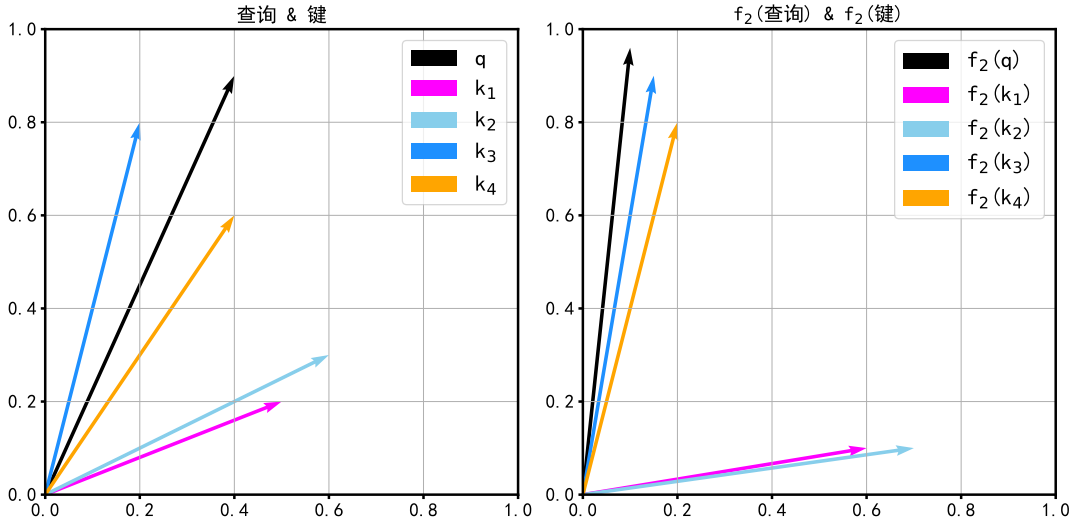



图 3-5 核函数对查询和键的作用示意图

图3-5提供了一个 $f_p$ 的效果示意图，展示了应用核函数前的查询$q$，以及应用核函数后的查询$f_2(q)$，应用核函数前的键$k_1$ 到$k_4$，以及应用其后的键$f_2(k_1)$ 到$f_2(k_4)$，其中 $p$ 为 2。本质上，$f_p$ 将每个向量拉近到其最近的坐标轴，但是除了改变其方向外还改变了其模长。定向系数 $p$ 在此处至关重要，因为它决定了这种向量重新定向的程度。这一过程有助于根据向量与特定轴的接近度将向量分类为不同的组，如图 3-5中以对角线区分的左右两组向量。结果上，同一组内的“查询-键”的相似性得到了增强，因为以坐标轴为标定物那么同一组内的“查询-键”由于同时更靠近相同的坐标轴而更接近，而不同组内的相似度则同理更远离，相似性降低。这里的‘相似性’代表注意力得分；因此，放大的相似性分布导致更加尖锐的注意力权重分布。例如图3-5左子图中的注意力分数

分布为 [0.32,0.24,0.18,0.25]，则作用核函数后的注意力分布变为 [0.68,0.17,0.02,0.13]。

## 3.4 反转嵌入、层归一化和前馈网络

除了3.2节和3.3节提到的谱滤波变换模块以及动态定向注意力机制，SDformer 还引入了三个有效的操作：反转嵌入、层归一化和前馈网络。这三个操作的有效性在过往的众多工作中得到了验证，本节将会对这些操作详细介绍。

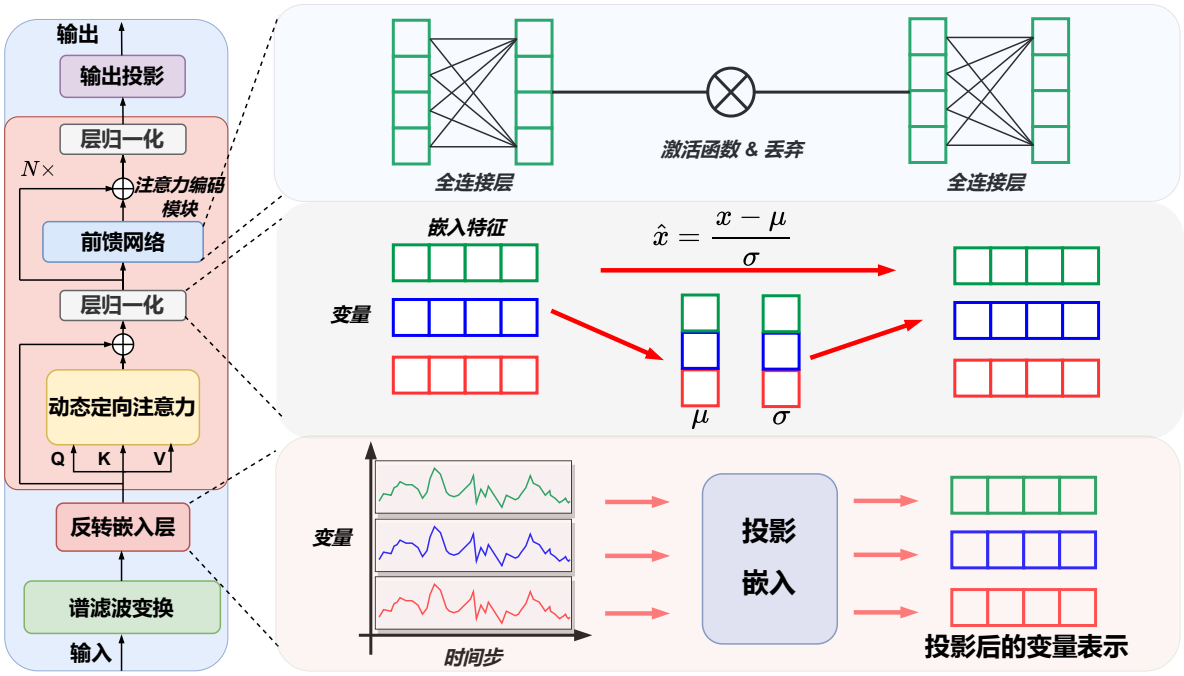



图 3-6 反转嵌入、层归一化和前馈网络的结构示意图

### 3.4.1 反转嵌入

如图3-6所示，在本文中通过引入 iTransformer[12] 提出了“反转嵌入”操作，SDformer 可以获得时间序列数据在 Transformer 模型中的一种特殊的表示：“变量 Token”。传统的类 Transformer 预测模型通常将同一时间的多个变量值视为一个“时序 Token”，即时间序列某一时刻所有变量的观测。然而当处理时间序列数据时，这种方法对于学习准确的注意力分数分布往往不够有效。为了解决这一问题，一些以“Patching”技术为基础的技术应运而生[13] [64]，这些技术通过扩大感受野来增强模型性能，从而避免模型对局部特征的过度关注。而“反转嵌入”则是“Patching”以外的另一种新兴的解决办法。

“反转嵌入”操作首先将输入时间序列的批次（Batch）、变量（Variate）和时间轴（Time）进行重排，以便更加有效地捕捉时间点上的多变量依赖性。具体实现上，给定一个批次的时间序列数据 $\mathbf{X} \in \mathbb{R}^{B \times T \times N}$，其中 $B$ 是批次大小，$T$ 是时间序列长度，$N$ 是变量数目，首先对输入序列进行转置：

$$\mathbf{X}' = \text{Transpose}(\mathbf{X}) \tag{3-12}$$

其中 Transpose(·) 函数是对数据的维度进行重新排列，使得输入序列形状从 $\mathbf{X} \in \mathbb{R}^{B\times T\times N}$ 变为 $\mathbf{X} \in \mathbb{R}^{B\times N\times T}$。接下来，如果存在额外的时间标记 $\mathbf{X}_{\text{mark}}$，即输入序列的时间索引由“yy-mm-dd-hh”构成，例如（2002-7-17-12），则将其编码后与 $\mathbf{X}'$ 在变量维度上进行拼接（Concat）：

$$\mathbf{X}_{\text{combined}} = \begin{cases} \mathbf{X}' & \mathbf{X}_{\text{mark}}\text{存在} \\ \text{Concat}(\mathbf{X}', \text{Transpose}(\mathbf{X}_{\text{mark}})) & \mathbf{X}_{\text{mark}}\text{不存在} \end{cases} \tag{3-13}$$

然后通过一个线性嵌入层将时间维度从 $T$ 映射到嵌入维度 $D$，再应用一个丢弃层（Dropout）来减少过拟合，得到最终的嵌入向量表示 $\mathbf{X}_{\text{h}} \in \mathbb{R}^{B\times N\times D}$：

$$\begin{aligned} \mathbf{X}_{\text{embedded}} &= \mathbf{X}_{\text{combined}} \cdot \mathbf{W} + \mathbf{b}, \\ \mathbf{X}_{\text{h}} &= \text{Dropout}(\mathbf{X}_{\text{embedded}}) \end{aligned} \tag{3-14}$$

其中 $\mathbf{W} \in \mathbb{R}^{T\times D}$ 是线性层的权重矩阵，$\mathbf{b} \in \mathbb{R}^{D}$ 是偏置项。

在本文提出的 SDformer 框架中，“反转嵌入”操作使得模型不仅能够捕捉多个变量之间的关联性，而且还能扩大局部的感受野，从而聚合全局的序列表征，这在预测复杂时间序列时尤其重要。过往的工作已经证明这种嵌入方法的有效性[12]，即该操作能够提高 Transformer 时间序列预测的准确性以及可解释性。

此外，经过“反转嵌入”后，每个 Token 都被认为是“变量 Token”并赋予了和以往的“时间戳 Token”截然不同的意义，因此注意力机制的作用也需要被重新审视[12]。当 $\mathbf{q}_i, \mathbf{k}_j \in \mathbb{R}^{d_k}$ 分别被定义为一个变量 Token 的查询和键时，那么在作用 Softmax 之前的注意力分数可以被描述为:

$$\mathbf{A}_{i,j} = \frac{(f(\mathbf{Q})f(\mathbf{K})^T)}{\sqrt{d_k}}_{i,j} \propto \mathbf{q}_i^T \mathbf{k}_j. \tag{3-15}$$

这个分数揭示了变量间的相关性，而整个注意力分数 $\mathbf{A} \in \mathbb{R}^{N\times N}$ 则代表着成对变量 Token 之间的多变量相关性。因此，高度相关的变量将会在与值向量 $\mathbf{V}$ 的交互中被赋予更多的权重，进一步深化了多变量关联性在时间序列预测中的作用。

### 3.4.2 层归一化

如图3-6所示，本文还引入了层归一化（Layer Normalization）。这是一种对神经网络中各层的激活值进行标准化的技术[56]，在深度学习中尤其对提高模型训练的效率和稳定性发挥着重要作用。与批量归一化（Batch Normalization）不同[66]，层归一化是在单个样本的层内进行，从而有助于处理变长输入和增强模型的泛化能力。在时间序列预测的应用场景中，层归一化对于调节模型对时间步长敏感度十分有效，因为时间序列数据的非平稳性——即其统计特性随时间的变化，对模型的预测能力是一个重大的挑战[61] [67]。这种方法不仅能够稳定梯度，还能减少训练过程中的内部协变量偏移现象（Covariant

Shift）。如公式3-16所示，SDformer 通过将每个变量的序列表示进行层归一化来平衡同一变量内不同时间步之间的激活分布，使模型在各个 Token 的处理上更为一致，有助于捕捉时间序列中的动态变化和潜在的复杂关系。

$$\text{LayerNorm}(\mathbf{H}) = \left\{ \frac{\mathbf{h}_n - \text{Mean}(\mathbf{h}_n)}{\sqrt{\text{Var}(\mathbf{h}_n)}} \right\}_{n=1,\ldots,N} \tag{3-16}$$

### 3.4.3 前馈网络

与前文中提到的 RNN 或 Transformer 不同，前馈神经网络（Feedforward Neural Network, FNN）是一种最基础、最直接、最容易理解的人工神经网络结构[68]，它的各个网络节点之间的连接不包括任何循环[69]。多层感知器（MLP）就是一种前馈人工神经网络，2023 年，基于 MLP 的时序预测模型大量涌现，挑战了 Transformer 在时序预测领域的统治地位[35] [36] [70] [71]。依据泛逼近定理[72]，一种解释是 MLP 的神经元被训练为描绘任何时间序列的内在特性，例如振幅、周期性，甚至频谱（将神经元视为滤波器），这使它们成为比 Transformer 更优的预测表示学习器[12]。如图3-6所示，SDformer 使用类 MLP 的结构提取每个 Token 的序列表征，并添加残差连接解决梯度消失和梯度爆炸问题，从而使得深层网络的训练变得更加可行和稳定[55]，其具体实现方法为：

$$\begin{aligned} \mathbf{y} &= \mathbf{x} + \text{Dropout}\left(\text{Activation}\left(\text{Conv1d}_{d_\text{m},d_\text{f}}(\mathbf{x})\right)\right) \\ \mathbf{z} &= \mathbf{y} + \text{Dropout}\left(\text{Conv1d}_{d_\text{f},d_\text{m}}(\mathbf{y})\right) \end{aligned} \tag{3-17}$$

其中，$\mathbf{x}$ 是输入的 Token 表示，$\mathbf{y}$ 是第一层卷积操作后得出的结果，$\mathbf{z}$ 是第二层卷积操作后的结果，即前馈网络的输出，$d_\text{m}$ 是反转嵌入的维度，$d_\text{f}$ 是前馈层内部的维度，数值上为 $d_\text{m}$ 的四倍。这两个卷积层，$\text{Conv1d}_{d_\text{m},d_\text{f}}$ 和 $\text{Conv1d}_{d_\text{f},d_\text{m}}$，分别用于扩展和压缩特征维度，并且依靠卷积神经网络的表征学习能力实现该前馈网络对时序特征的学习，Activation 是非线性激活函数 ReLU。通过堆叠注意力编码模块，SDformer 可以有效地通过多层 MLP 编码作为模型输入的时间序列窗口，并在后续的模块中同样使用单层 MLP 解码该时序表征，实现有效的预测。

## 3.5 本章小结

本章详细介绍了本文提出的基于时频域特征耦合的时序预测长期预测框架 SDformer。模型的编码器部分包括两个模块：谱滤波变换和动态定向注意力，以及三个配套操作：反转嵌入、层归一化和前馈网络。谱滤波变换可以在频域进行滤波并且在时域进行平滑，动态定向注意力则是改进了传统的自注意力机制，实现了一个基于核函数的新颖注意力机制增强注意力分数的有效分配，缓解传统注意力机制在建模多变量时间序列过程的能力退化。此外，反转嵌入将每个变量整体投影到高维空间，层归一化增强时序数据的平稳性并加速模型收敛，前馈网络实现对时序特征的提取。最终 SDformer 使用单层 MLP 作为解码器，完成特定长度的预测。

# 第 4 章　实验结果与分析

本章主要介绍实验设置与实验结果分析。首先在4.1节介绍了实验中采用的 7 个公开的时间序列数据集，详细说明了本文实验中用于对比的 11 个深度学习模型，以及用于评估预测效果和注意力分数分布情况的评价指标，实验环境和超参设置等关键信息。4.2节详细说明了本文提出的 SDformer 模型在时间序列长期预测任务上的具体实验结果，展示了本文模型在该任务上的优越性。4.3节详细分析了实验中两个关键模块的具体效果，以及多模块共同作用时的效果。4.4节针对模型中的模块化结构设计了两组消融实验，通过对消融实验结果的分析证明了模块的不可或缺性。4.5节着重讨论了模块中的 4 个重要的超参数如何对实验结果产生影响。4.6节分析了模型的复杂性，证明了其轻量性。4.7节对本章内容进行了总结。

## 4.1　实验设置

### 4.1.1　实验数据集

本文在 7 个公开的现实世界数据集上进行实验，评估本文提出的 SDformer 的性能，这些数据集的详细信息总结在表4-1中[1]。特征维度表示每个数据集的变量数；预测长度表示要预测的未来时间点数量，每个数据集包含四个预测长度；数据集大小表示按固定比例划分（训练，验证，测试）的总时间点数量；采样频率表示时间点的采样间隔。

表 4-1 本文预测实验采用的数据集信息汇总

| 数据集 | 特征维度 | 预测长度 | 数据集大小 | 来源（采样频率） |
|---|---|---|---|---|
| ETTm2 | 7 | {96, 192, 336, 720} | (34465, 11521, 11521) | 电力（15 分钟） |
| ETTh2 | 7 | {96, 192, 336, 720} | (8545, 2881, 2881) | 电力（小时） |
| Electricity | 321 | {96, 192, 336, 720} | (18317, 2633, 5261) | 电力（小时） |
| Traffic | 862 | {96, 192, 336, 720} | (12185, 1757, 3509) | 交通（小时） |
| Weather | 21 | {96, 192, 336, 720} | (36792, 5271, 10540) | 气象（10 分钟） |
| Exchange | 8 | {96, 192, 336, 720} | (5120, 665, 1422) | 汇率（每日） |
| ILI | 7 | {24, 36, 48, 60} | (617, 74, 170) | 疾病（每周） |

ETT 包含了 2016 年 7 月至 2018 年 7 月电力变压器温度的 7 个指标。本文使用了其中的两个子集，其中 ETTh2 每小时记录一次，ETTm2 每 15 分钟记录一次。Exchange 收集了 1990 年至 2016 年 8 个国家的每日汇率面板数据。Weather 包括了马克斯 · 普朗克生物地球化学研究所气象站在 2020 年每 10 分钟收集的 21 个气象特征。Electricity 记录了 321 个客户的每小时电力消耗数据。Traffic 收集了 2015 年 1 月至 2016 年 12 月旧金

[1]所有的数据集来自清华大学软件学院龙明盛教授课题组维护的开源时序深度模型库 Time Series Library[51]。

山湾区高速公路 862 个传感器测量的每小时道路占用率。ILI 描述了 2002 年至 2021 年美国疾病控制和预防中心记录的患有流感疾病的患者与患者数量的比率。如表4-1所示，在数据集划分上本文实验遵循了 TimesNet[51] 中使用的相同数据处理和训练-验证-测试集切分方式，其中训练、验证和测试数据集按 7：2：1 划分。预测设置方面，本文将回顾窗口的长度固定为 96，而在 ETTm2、ETTh2、Weather、ECL、Exchange 和 Traffic 数据集中，预测长度为 {96,192,336,720}。对于 ILI 数据集，预测长度为 {24,36,48,60}。

### 4.1.2 基线模型

为了保证本文实验部分的对比试验是充分且全面的，本文广泛采纳了近年来在时间序列分析，尤其是时间序列长期预测任务上取得优异表现的 11 个模型。这些模型按照结构可划分为如下几类：以 iTransformer[12] 和 PatchTST[13] 为代表的 Transformer 类时序模型，以 TimesNet[51] 为代表的时序卷积（TCN）类模型，以 DLinear[35] 为代表的 MLP 类模型[2]。这 11 个基线模型分别是：

**iTransformer**[12]：iTransformer 是一种 Transformer 类的新型时间序列预测模型，它通过将注意力和前馈网络应用于倒置的维度来捕获多变量间的相关性并学习非线性表示。在多个数据集上取得了全面领先的结果。

**DLinear**[35]：DLinear 属于 LSTF-Linear 模型家族，这类模型提出了一套简单的单层线性 MLP 的设计思路，用于学习输入和输出序列之间的固定的映射关系，该模型的提出引领了使用 MLP 进行时间序列预测的潮流。

**PatchTST**[13]：PatchTST 提出了一种基于 Transformer 的时序模型，通过引入两个关键组件：Patching 和通道独立结构，该模型可以有效挖掘单变量序列的局部特征进行有效的时序建模。本文实验将其作为长期预测任务重要的基准对比模型之一，并遵循统一的设置作为实验配置重新训练了该模型。

**TimesNet**[51]：TimesNet 是一种用于时间序列分析的任务通用框架，它通过将一维时间序列转换为基于多周期的二维张量来处理复杂的时间变化。在多个主流时间序列分析任务中取得一致的最先进性能。

**FEDformer**[43]：FEDformer 提出了一种在频率上具有低秩近似的注意力机制，以及一种控制分布移动的混合分解。本文遵循推荐的设置作为实验配置。

**Autoformer**[40]：Autoformer 提出了一种分解架构，通过将序列分解块作为内部模块嵌入到编码器-解码器的结构中，可以逐步聚合中间预测的长期趋势部分。并且该模型提出了 Auto-correlation 机制，进而降低了模型的计算复杂度。

**Non-stationary Transformer**[61]：Non-stationary Transformer 主要专注于设计规范化机制。它探索了时间序列预测任务中的平稳化问题，通过相对简单的插件系列平稳化和

[2]值得说明的是，本文实验部分对比的基线模型包括了 2023 年 10 月在 arXiv 放出的 iTransformer（arXiv:2310.06625），而本文的实验部分是在 2023 年 11 月开展的，因此可以说本文对比了最新和最优的 SOTA 模型 iTransformer，证明了本文实验部分的充分性和严谨性。

去平稳化模块来提升注意力模块的性能。

**Crossformer**[14]：Crossformer 是一种基于 Transformer 的时间序列预测模型，它显式地利用跨维度依赖性和跨时间步依赖性进行多变量时间序列预测，同时提出两阶段注意力层进行依赖学习，并且该模型实现了线性复杂度。

**LightTS**[34]：LightTS 是一种仅基于简单 MLP 结构的轻量深度学习架构。其关键思想是在两种精细的下采样策略之上应用 MLP，包括间隔采样和连续采样。连续采样侧重于捕获短期局部模式，而间隔采样侧重于捕获长期依赖性。

**TiDE**[70]：TiDE 是一种基于 MLP 的编码器-解码器模型，它享有线性模型的轻量性和训练速度，同时也能够处理协变量和非线性依赖关系。

**Informer**[39]：Informer 利用高效的自注意力蒸馏机制增强模型中具有主导作用的注意力机制，有效处理较长的输入序列。使用并行生成式解码器结构，对长时间序列进行一次前向计算输出所有预测结果而不是逐步的方式进行预测，提高了模型的推理速度。

### 4.1.3 评价指标

对于时序模型预测效果的评估，本文考虑了两个基本指标：均方误差（MSE）和平均绝对误差（MAE）。这两个指标广泛用于量化回归问题，尤其是衡量模型的预测准确性。MSE 计算为预测值和实际值差异平方的平均值，提供了预测的方差度量。MAE 测量一组预测中错误的平均幅度，不考虑它们的方向。它们定义如下：

$$MSE = \frac{1}{H}\sum_{i=1}^{H}(X_i - \hat{X}_i)^2, \tag{4-1}$$

$$MAE = \frac{1}{H}\sum_{i=1}^{H}|X_i - \hat{X}_i|, \tag{4-2}$$

其中，$H$ 是样本数量，$X_i$ 是实际值，$\hat{X}_i$ 是预测值。通过这两个常用的指标，本文实验部分实现了对时序预测效果的有效衡量，直观展示了不同预测模型的性能差异。

除此之外，为了验证 SDformer 可以有效解决注意力机制中出现的注意力矩阵行同质化的现象，本文还引入了基尼系数作为衡量模型注意力分配均匀程度的指标，以及矩阵的秩来衡量注意力矩阵的信息丰富度。针对注意力分数矩阵 $M$，基尼系数 $G$ 被用来衡量其中注意力分数的分布情况。基尼系数越接近 0，表示注意力分配越均匀；越接近 1，则表示分配越不均匀。基尼系数的计算方法如下：

$$G = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}|m_i - m_j|}{2N^2\bar{m}}, \tag{4-3}$$

其中，$N$ 是矩阵 $M$ 中所有元素的数量，$m_i$ 和 $m_j$ 分别是矩阵中的元素，而 $\bar{m}$ 是矩阵中所有元素的平均值。此外，矩阵的秩用于衡量注意力矩阵的信息丰富度。秩越高，表示矩阵包含的信息越多，注意力分布越不平均。通过引入基尼系数，可以实现定量地评估

模型注意力机制中的注意力分数分配的均匀程度，而矩阵的秩则显示了模型注意力机制所包含信息量的丰富程度。

### 4.1.4 实验环境

由于本文实验内容饱满，需要使用 GPU 计算才可以实现在有限的时间内开展尽可能多的实验，以保证对比试验的充分和完整。此外，使用 GPU 进行训练还可以大大提高网格搜索超参的效率，最大程度上探索 SDformer 的能力上限。软硬件实验环境的具体配置如下：

**硬件环境：**

- 课题组服务器
- GPU：RTX3090 (24GB)
- CPU：AMD EPYC 7763
- CPU 核心数：每个插槽 64 核心，共 2 个插槽
- 总内存：251 GB

**软件环境：**

- 操作系统：Ubuntu 22.04.2 LTS
- 编码语言：Python 3.8
- CUDA 版本：12.2
- 深度学习框架：PyTorch 1.13.1
- 包管理工具：conda + pip
- 编码工具：VSCode

### 4.1.5 超参设置与训练策略

本文实验使用 PyTorch 框架[73]进行，每个实验都在单独的 NVIDIA RTX3090 24GB GPU 上运行。本文使用 ADAM 算法[74]进行优化，初始学习率从 $\{10^{-3}, 5\times10^{-4}, 10^{-4}\}$ 中选取。损失函数采用时序预测任务中常用的 L2 损失（MSE）。批处理大小（Batch Size）从 $\{16, 32\}$ 中选择，训练周期固定为 10 epochs，模型中全连接层的维度从 $\{256, 512\}$ 中选择。此外，用于比较的基准模型是参考 TimesNet[51]的统一标准实现的，确保遵守每个模型的原始论文或官方代码中设定的配置。此外，本文采用了两种关键的训练策略来优化模型的性能：动态调整学习率和早期停止机制。

为了适应训练过程中的变化，本文采用了动态调整优化器，包含两种类型的学习率调整方法：第一种方法是学习率在每个周期按固定比例（50%）下降。这种连续的减少有助于模型在整个训练过程中平稳地调整学习率，从而稳定地向最优解收敛。第二种方法是在特定的训练周期进行显著的学习率调整。具体来说，学习率在第 2 个周期调整到 $5\times10^{-5}$，在第 4 个周期调整到 $10^{-5}$，在第 6 个周期调整到 $5\times10^{-6}$，在第 8 个周期调整到 $10^{-6}$，在第 10 个周期调整到 $5\times10^{-7}$。每次学习率更新时，系统会打印出新的学习率值，以便于监控和调试。这种动态调整帮助模型在早期快速学习，同时在后期训练中优化其性能，确保训练过程中学习率始终保持在最优水平。

此外，为了避免过拟合并保证训练的高效性，本文采用了一个基于验证集损失的早期停止机制。这一机制监测验证集的损失，并在连续几个训练周期（默认为 7 个周期）

内损失没有显著改善时触发停止训练的条件。若在这些周期内新的损失分数没有比当前最佳分数更好（考虑一个较小的容忍度），则计数器递增；否则，计数器重置为 0，并更新最佳分数。此外，每当发现比之前更低的验证集损失时，系统会自动保存当前的模型状态，以确保即使训练提前终止，也能保留性能最优的模型。一旦计数器达到设定的耐心阈值，早停机制将被激活，训练过程便会停止。这种早停机制有效地防止了过拟合的发生，同时提高了模型训练的整体效率。

## 4.2 预测实验结果分析

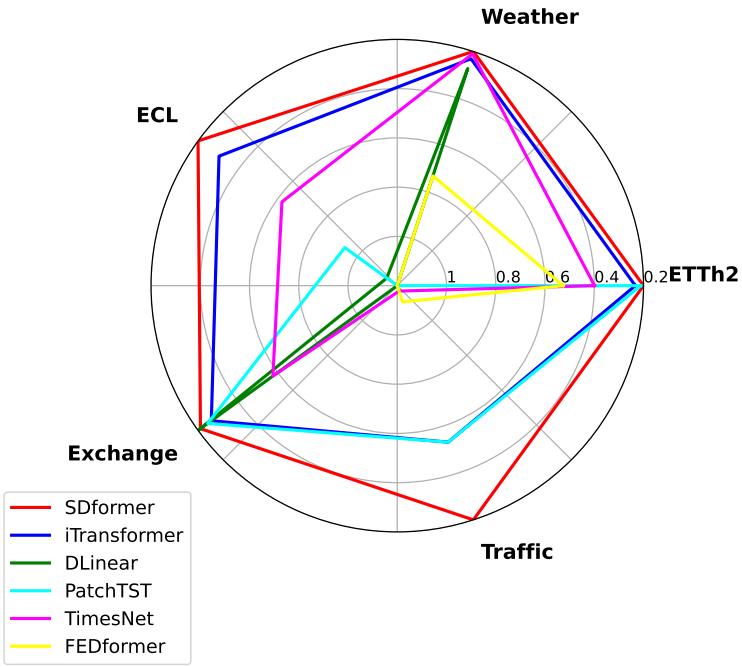



图 4-1 不同模型的长期预测任务效果对比图，图中的数值为 MSE 标准化后的结果

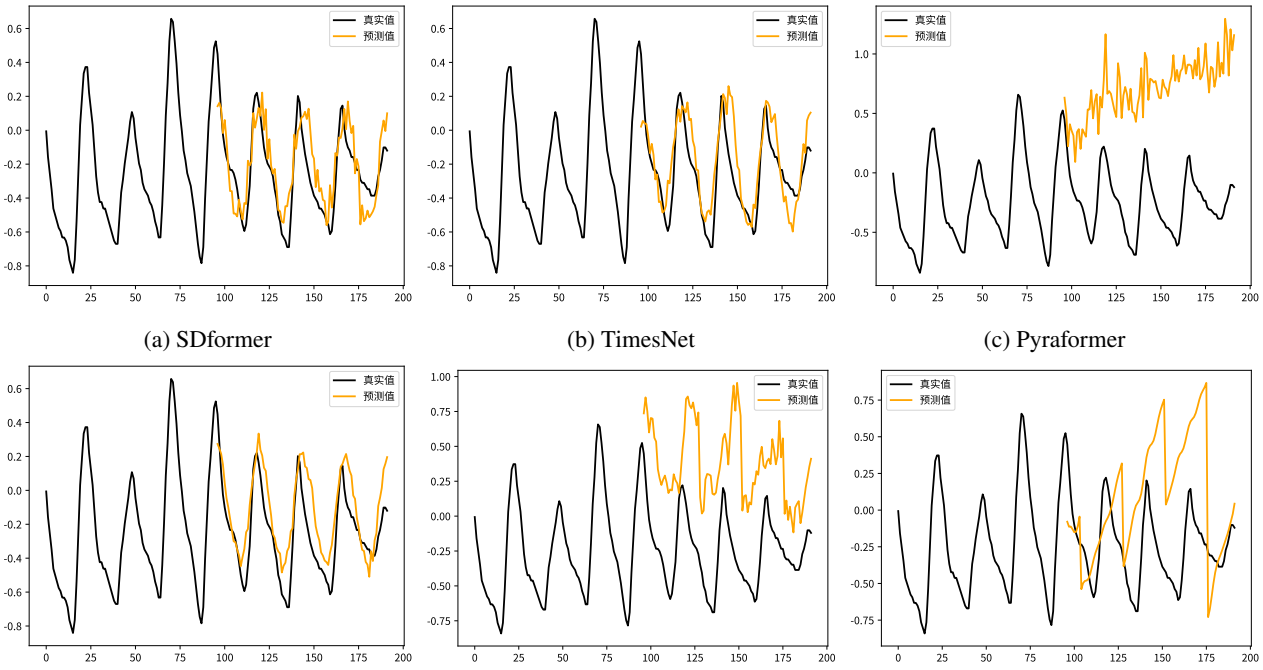
为了验证本文提出的 SDformer 在时间序列长期预测任务上的优越性能，本文在 7 个公开数据集上对比 11 个近年来的最优模型进行了广泛的实验。特别说明的是，由于 PatchTST[13] 扩大了回看窗口的长度，其原文中使用了可调节的回看长度为 336 和 512，为了保证对比的公平，本文在实验过程中使用固定长度为 96 的一致回看长度重新训练了该模型。除此之外，所有其他模型的结果均来自 iTransformer[12] 和 TimesNet[51]。从图4-1中不难看到，SDformer 在多个数据集上表现优于截至目前的最优模型，如 iTransformer[12] 或 PatchTST[13]，完整的结果总结在了表 4-2中。表 4-2直观地展示了 SDformer 在长期预测方面的显著优越性，其中较低的均方误差（MSE）和平均绝对误差（MAE）值表明性能更优。具体来说，与之前的最佳模型 iTransformer[12] 相比，在 Traffic 数据集上，SDformer 的 MSE 和 MAE 分别实现了 15%和 8.5%的降低，凸显了其在建模高维时间序列数据以及捕获可靠多变量相关性方面的强大能力。在 ETTm2 数据集上，它在 MSE 和 MAE 方面比另一个最优模型 DLinear[35] 误差降低了 17%。从表4-2中还可以观

察到 PatchTST[13]在 ETTm2 和 ILI 数据集上都表现出了最优的效果，说明该模型得益于其独特的通道独立设计和 Patch 建模思想，通过有效建模时间序列的局部特征，扩展了 Transformer 在时序建模领域的使用思路。此外，在时间序列预测模型的比较可视化中，可以观察到 SDformer 模型在各类数据集上的表现具有显著优势。对于平稳时间序列数据集，如图4-2和图4-3，SDformer 的预测曲线（橙色线条）与实际值（黑色线条）之间的匹配度非常高，表明其在平稳时间序列的预测上准确性较高。同样，在非平稳时间序列数据集，如图4-4和图4-5，SDformer 同样展现出了较其他模型更为精确的预测能力，其预测曲线同样紧密跟随实际值的波动。不同于 SDformer，其他模型如 Pyraformer[41]和 DLinear[35]等，其预测结果与实际值存在较明显的偏差。这些模型的预测曲线在与真实值的曲线的峰值和谷值处通常出现滞后或过度平滑的现象，例如 DLinear，基于该模型通过单层 MLP 学习一批固定的权重和偏置的原理，DLinear 只能根据历史窗口拟合未来窗口的大致趋势，但无法捕捉细微的模式化波动，这表明它们在捕捉时间序列中的快速变化或复杂模式上不如 SDformer。综上所述，SDformer 在七个数据集中的五个都取得了最优的预测效果，在剩余的两个数据集上也取得了仅次于 PatchTST[13]的次优的效果，对预测结果的可视化也佐证了 SDformer 的强大预测性能，因此可以说 SDformer 已经跻身 SOTA（state-of-the-art）模型行列。

表 4-2 长期预测实验完整结果汇总，Avg 表示四个预测长度结果的均值，“Stationary”表示 Non-stationary Transformer，越低的 MSE 和 MAE 表示效果越好，表中**红色**：最佳，**蓝色**：次佳

| 模型 | | **SDformer** | | iTransformer | | DLinear | | PatchTST | | TimesNet | | FEDformer | | Autoformer | | Stationary | | Crossformer | | TiDE | | LightTS | | Informer | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 评价指标 | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| *ETTm2* | 96 | 0.183 | 0.268 | 0.184 | 0.268 | 0.193 | 0.292 | **0.177** | **0.261** | 0.187 | 0.267 | 0.203 | 0.287 | 0.255 | 0.339 | 0.192 | 0.274 | 0.287 | 0.366 | 0.207 | 0.305 | 0.308 | 0.365 | 0.453 | 0.596 |
| | 192 | 0.249 | 0.309 | 0.253 | 0.313 | 0.284 | 0.362 | **0.244** | **0.303** | 0.249 | 0.309 | 0.269 | 0.328 | 0.281 | 0.340 | 0.280 | 0.339 | 0.414 | 0.492 | 0.290 | 0.364 | 0.311 | 0.382 | 0.311 | 0.382 |
| | 336 | 0.313 | 0.348 | 0.312 | 0.350 | 0.369 | 0.427 | **0.309** | **0.346** | 0.321 | 0.351 | 0.325 | 0.366 | 0.339 | 0.372 | 0.334 | 0.361 | 0.597 | 0.542 | 0.377 | 0.422 | 0.442 | 0.466 | 1.363 | 0.887 |
| | 720 | 0.407 | 0.402 | 0.412 | 0.406 | 0.554 | 0.522 | **0.400** | **0.398** | 0.408 | 0.403 | 0.421 | 0.415 | 0.433 | 0.432 | 0.417 | 0.413 | 1.730 | 1.042 | 0.558 | 0.524 | 0.675 | 0.587 | 3.379 | 1.338 |
| | Avg | 0.288 | 0.332 | 0.291 | 0.334 | 0.350 | 0.401 | **0.255** | **0.327** | 0.291 | 0.333 | 0.305 | 0.349 | 0.327 | 0.371 | 0.306 | 0.347 | 0.757 | 0.610 | 0.358 | 0.404 | 0.409 | 0.436 | 1.410 | 0.810 |
| *ETTh2* | 96 | 0.298 | 0.345 | 0.299 | 0.350 | 0.333 | 0.387 | **0.295** | **0.344** | 0.340 | 0.374 | 0.358 | 0.397 | 0.346 | 0.388 | 0.476 | 0.458 | 0.745 | 0.584 | 0.400 | 0.440 | 0.397 | 0.437 | 3.755 | 1.525 |
| | 192 | 0.378 | 0.394 | 0.381 | 0.400 | 0.477 | 0.476 | **0.367** | **0.391** | 0.402 | 0.414 | 0.429 | 0.439 | 0.456 | 0.452 | 0.512 | 0.493 | 0.877 | 0.656 | 0.528 | 0.509 | 0.520 | 0.504 | 5.602 | 1.931 |
| | 336 | **0.419** | **0.427** | 0.424 | 0.433 | 0.549 | 0.541 | 0.434 | 0.443 | 0.452 | 0.452 | 0.496 | 0.487 | 0.482 | 0.486 | 0.552 | 0.551 | 1.041 | 0.731 | 0.377 | 0.422 | 0.626 | 0.559 | 4.721 | 1.835 |
| | 720 | **0.418** | **0.437** | 0.430 | 0.446 | 0.831 | 0.657 | 0.423 | 0.445 | 0.462 | 0.468 | 0.463 | 0.474 | 0.515 | 0.511 | 0.562 | 0.560 | 1.104 | 0.763 | 0.874 | 0.679 | 0.863 | 0.672 | 3.647 | 1.625 |
| | Avg | **0.378** | **0.401** | 0.384 | 0.407 | 0.559 | 0.515 | 0.380 | 0.406 | 0.414 | 0.427 | 0.437 | 0.449 | 0.450 | 0.459 | 0.526 | 0.516 | 0.942 | 0.684 | 0.611 | 0.550 | 0.543 | 0.548 | 4.431 | 1.729 |
| *Weather* | 96 | 0.171 | **0.210** | 0.176 | 0.216 | 0.196 | 0.255 | 0.177 | 0.218 | 0.172 | 0.220 | 0.217 | 0.296 | 0.266 | 0.336 | 0.173 | 0.223 | **0.158** | 0.230 | 0.202 | 0.261 | 0.182 | 0.242 | 0.300 | 0.384 |
| | 192 | 0.222 | **0.255** | 0.225 | 0.257 | 0.237 | 0.296 | 0.225 | 0.259 | 0.219 | 0.261 | 0.276 | 0.336 | 0.307 | 0.367 | 0.245 | 0.285 | **0.206** | 0.277 | 0.242 | 0.298 | 0.227 | 0.278 | 0.598 | 0.544 |
| | 336 | 0.278 | **0.297** | 0.281 | 0.299 | 0.283 | 0.335 | 0.279 | 0.297 | 0.280 | 0.306 | 0.339 | 0.380 | 0.359 | 0.395 | 0.321 | 0.338 | **0.272** | 0.335 | 0.287 | 0.335 | 0.282 | 0.334 | 0.578 | 0.523 |
| | 720 | 0.358 | **0.348** | 0.360 | 0.352 | **0.345** | 0.381 | 0.447 | 0.466 | 0.365 | 0.359 | 0.403 | 0.428 | 0.419 | 0.428 | 0.414 | 0.410 | 0.398 | 0.418 | 0.351 | 0.386 | 0.352 | 0.386 | 1.059 | 0.741 |
| | Avg | **0.258** | **0.278** | 0.261 | 0.281 | 0.265 | 0.317 | 0.354 | 0.348 | 0.259 | 0.287 | 0.309 | 0.360 | 0.338 | 0.382 | 0.288 | 0.314 | 0.259 | 0.315 | 0.271 | 0.320 | 0.261 | 0.312 | 0.634 | 0.548 |
| *ECL* | 96 | 0.150 | 0.243 | **0.149** | **0.240** | 0.197 | 0.282 | 0.181 | 0.271 | 0.168 | 0.272 | 0.193 | 0.308 | 0.201 | 0.317 | 0.169 | 0.273 | 0.219 | 0.314 | 0.237 | 0.329 | 0.207 | 0.307 | 0.274 | 0.368 |
| | 192 | **0.164** | 0.258 | 0.165 | **0.257** | 0.196 | 0.285 | 0.187 | 0.276 | 0.184 | 0.289 | 0.201 | 0.315 | 0.222 | 0.334 | 0.182 | 0.286 | 0.231 | 0.322 | 0.236 | 0.330 | 0.213 | 0.316 | 0.296 | 0.386 |
| | 336 | 0.180 | 0.274 | **0.178** | **0.271** | 0.209 | 0.301 | 0.203 | 0.292 | 0.198 | 0.300 | 0.214 | 0.329 | 0.231 | 0.338 | 0.200 | 0.304 | 0.246 | 0.337 | 0.249 | 0.344 | 0.230 | 0.333 | 0.300 | 0.394 |
| | 720 | **0.211** | **0.302** | 0.228 | 0.312 | 0.245 | 0.333 | 0.245 | 0.325 | 0.220 | 0.320 | 0.246 | 0.355 | 0.254 | 0.361 | 0.222 | 0.321 | 0.280 | 0.363 | 0.284 | 0.373 | 0.265 | 0.360 | 0.373 | 0.439 |
| | Avg | **0.176** | 0.269 | 0.180 | **0.261** | 0.212 | 0.300 | 0.204 | 0.291 | 0.192 | 0.295 | 0.214 | 0.327 | 0.227 | 0.338 | 0.193 | 0.296 | 0.244 | 0.334 | 0.251 | 0.344 | 0.229 | 0.329 | 0.311 | 0.397 |
| *Exchange* | 96 | **0.087** | 0.208 | 0.087 | 0.207 | 0.088 | 0.218 | 0.089 | **0.206** | 0.107 | 0.234 | 0.148 | 0.278 | 0.197 | 0.323 | 0.111 | 0.237 | 0.256 | 0.367 | 0.094 | 0.218 | 0.116 | 0.262 | 0.847 | 0.752 |
| | 192 | 0.177 | **0.300** | 0.181 | 0.304 | **0.176** | 0.315 | 0.186 | 0.307 | 0.226 | 0.344 | 0.271 | 0.315 | 0.300 | 0.369 | 0.219 | 0.335 | 0.470 | 0.509 | 0.184 | 0.307 | 0.215 | 0.359 | 1.204 | 0.895 |
| | 336 | 0.331 | 0.418 | 0.338 | 0.422 | 0.313 | 0.427 | **0.310** | **0.403** | 0.367 | 0.448 | 0.460 | 0.427 | 0.509 | 0.524 | 0.421 | 0.476 | 1.268 | 0.883 | 0.349 | 0.431 | 0.377 | 0.466 | 1.672 | 1.036 |
| | 720 | **0.829** | **0.688** | 0.853 | 0.696 | 0.839 | 0.695 | 0.864 | 0.701 | 0.964 | 0.746 | 1.195 | 0.695 | 1.447 | 0.941 | 1.092 | 0.769 | 1.767 | 1.068 | 0.852 | 0.698 | 0.831 | 0.699 | 1.941 | 1.127 |
| | Avg | 0.356 | **0.404** | 0.365 | 0.407 | **0.354** | 0.414 | 0.362 | 0.404 | 0.416 | 0.443 | 0.519 | 0.429 | 0.613 | 0.539 | 0.461 | 0.454 | 0.940 | 0.707 | 0.370 | 0.413 | 0.385 | 0.447 | 1.550 | 0.998 |
| *Traffic* | 96 | **0.377** | **0.262** | 0.393 | 0.268 | 0.650 | 0.396 | 0.460 | 0.295 | 0.593 | 0.321 | 0.587 | 0.366 | 0.613 | 0.388 | 0.612 | 0.338 | 0.522 | 0.290 | 0.805 | 0.493 | 0.615 | 0.391 | 0.719 | 0.391 |
| | 192 | **0.396** | **0.272** | 0.413 | 0.277 | 0.598 | 0.370 | 0.464 | 0.296 | 0.617 | 0.336 | 0.604 | 0.373 | 0.616 | 0.382 | 0.613 | 0.340 | 0.530 | 0.293 | 0.756 | 0.474 | 0.601 | 0.382 | 0.696 | 0.379 |
| | 336 | **0.413** | **0.281** | 0.424 | 0.283 | 0.605 | 0.373 | 0.480 | 0.303 | 0.629 | 0.336 | 0.621 | 0.383 | 0.622 | 0.337 | 0.618 | 0.328 | 0.558 | 0.305 | 0.762 | 0.477 | 0.613 | 0.386 | 0.777 | 0.420 |
| | 720 | **0.447** | **0.295** | 0.460 | 0.301 | 0.645 | 0.394 | 0.514 | 0.322 | 0.640 | 0.350 | 0.626 | 0.382 | 0.660 | 0.408 | 0.653 | 0.355 | 0.589 | 0.328 | 0.719 | 0.449 | 0.658 | 0.407 | 0.864 | 0.472 |
| | Avg | **0.408** | **0.278** | 0.423 | 0.282 | 0.625 | 0.383 | 0.480 | 0.304 | 0.620 | 0.336 | 0.610 | 0.376 | 0.628 | 0.379 | 0.624 | 0.340 | 0.550 | 0.304 | 0.760 | 0.473 | 0.622 | 0.392 | 0.767 | 0.416 |
| *ILI* | 24 | 2.079 | 0.900 | 2.567 | 0.949 | 4.830 | 1.167 | **1.319** | **0.754** | 2.317 | 0.934 | 3.228 | 1.260 | 3.483 | 1.287 | 2.294 | 0.945 | 2.527 | 1.020 | 8.313 | 2.144 | 8.313 | 2.144 | 5.764 | 1.677 |
| | 36 | 2.219 | 0.933 | 2.082 | 0.919 | 4.454 | 1.563 | **1.430** | **0.834** | 1.972 | 0.920 | 2.679 | 1.080 | 3.103 | 1.148 | 1.825 | 0.848 | 2.615 | 1.007 | 6.631 | 1.902 | 6.631 | 1.902 | 4.755 | 1.467 |
| | 48 | 1.986 | 0.881 | 2.146 | 0.921 | 4.099 | 1.476 | **1.553** | **0.815** | 2.238 | 0.940 | 2.622 | 1.078 | 2.669 | 1.085 | 2.010 | 0.900 | 2.359 | 0.972 | 7.299 | 1.982 | 7.299 | 1.982 | 4.763 | 1.469 |
| | 60 | 2.031 | 0.915 | 2.052 | 0.931 | 4.207 | 1.482 | **1.470** | **0.788** | 2.027 | 0.928 | 2.857 | 1.157 | 2.770 | 1.125 | 2.178 | 0.963 | 2.487 | 1.016 | 7.283 | 1.985 | 7.283 | 1.985 | 5.264 | 1.564 |
| | Avg | 2.066 | 0.915 | 2.212 | 0.930 | 4.398 | 1.422 | **1.443** | **0.797** | 2.139 | 0.931 | 2.847 | 1.144 | 3.006 | 1.161 | 2.077 | 0.914 | 2.497 | 1.004 | 7.382 | 2.003 | 7.382 | 2.003 | 5.137 | 1.544 |

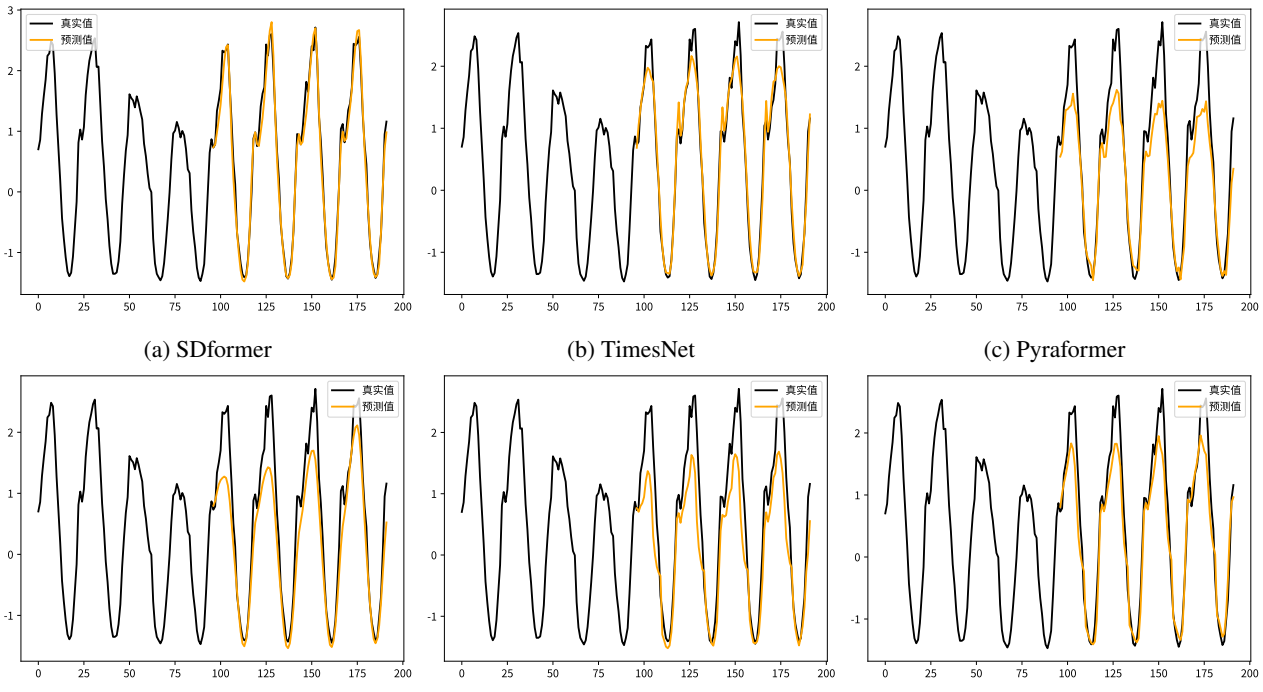
本文在预测实验的可视化部分引入了两个表4-2中没有记录的模型：Reformer[44]和Transformer[11]。本文在四个数据集上各随机选择了一组实验，以曲线图的形式绘制了真实值和预测值，以直观地体现模型的预测效果。结果如图4-2、4-3、4-4、4-5所示。




(a) SDformer (b) TimesNet (c) Pyraformer

(d) DLinear (e) Reformer (f) Transformer

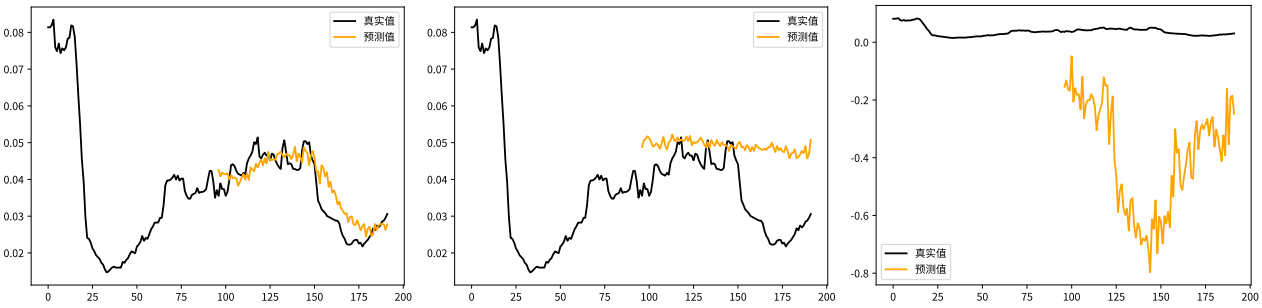
**图 4-2 在 ETTh2 数据集上，输入长度 96、预测长度 96 结果可视化**



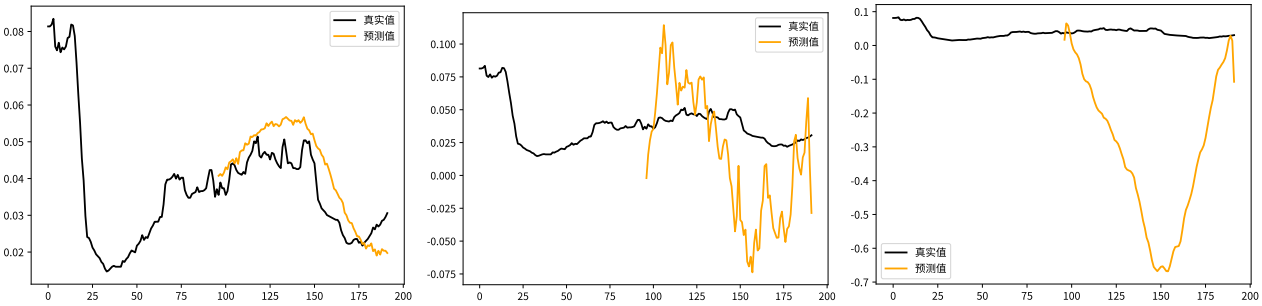

(a) SDformer (b) TimesNet (c) Pyraformer

(d) DLinear (e) Reformer (f) Transformer

**图 4-3 在 Traffic 数据集上，输入长度 96、预测长度 96 结果可视化**

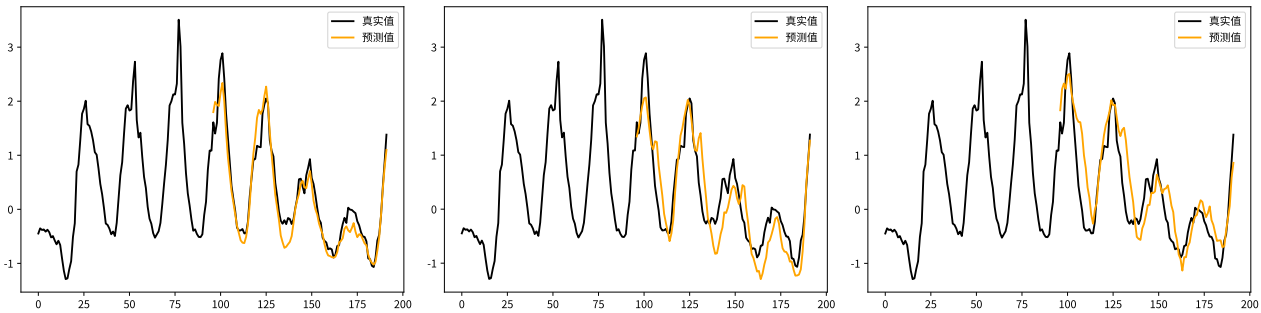



(a) SDformer (b) TimesNet (c) Pyraformer

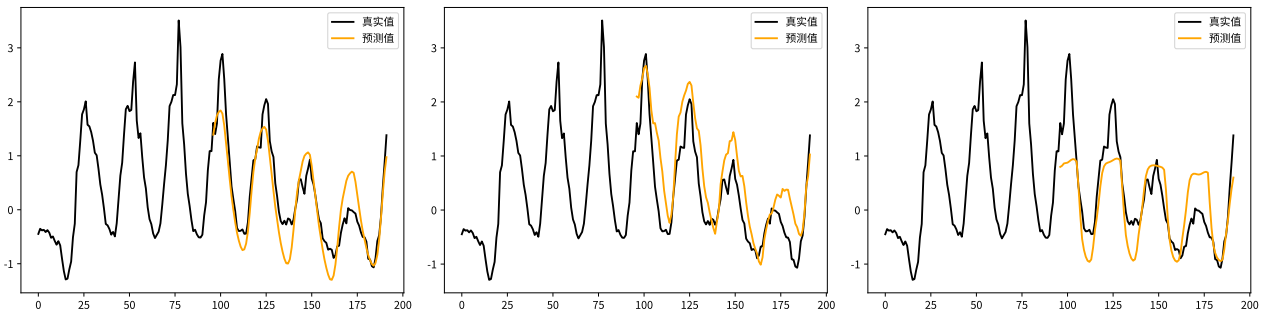



(d) DLinear (e) Reformer (f) Transformer

**图 4-4 在 Weather 数据集上，输入长度 96、预测长度 96 结果可视化**




(a) SDformer (b) TimesNet (c) Pyraformer




(d) DLinear (e) Reformer (f) Transformer

**图 4-5 在 ECL 数据集上，输入长度 96、预测长度 96 结果可视化**

## 4.3 模块效果分析

本文的核心创新点在于提出了谱滤波变换模块和动态定向注意力，并将二者整合到 Transformer 的编码器中，有效解决了传统 Transformer 的自注意力机制在建模多变量时

间序列时出现的注意力分布过于均匀，导致注意力能力退化的问题。为了证明 SDformer 确实有效解决了该问题，本节将会从三个角度对 SDformer 中的模块化设计进行有效性分析。具体地，本节将会分别细致解释和分析谱滤波变换和动态定向注意力的作用和效果，还会从两者联合的角度从定量和定性两个角度证明 SDformer 对该问题的解决情况。

### 4.3.1 谱滤波变换模块

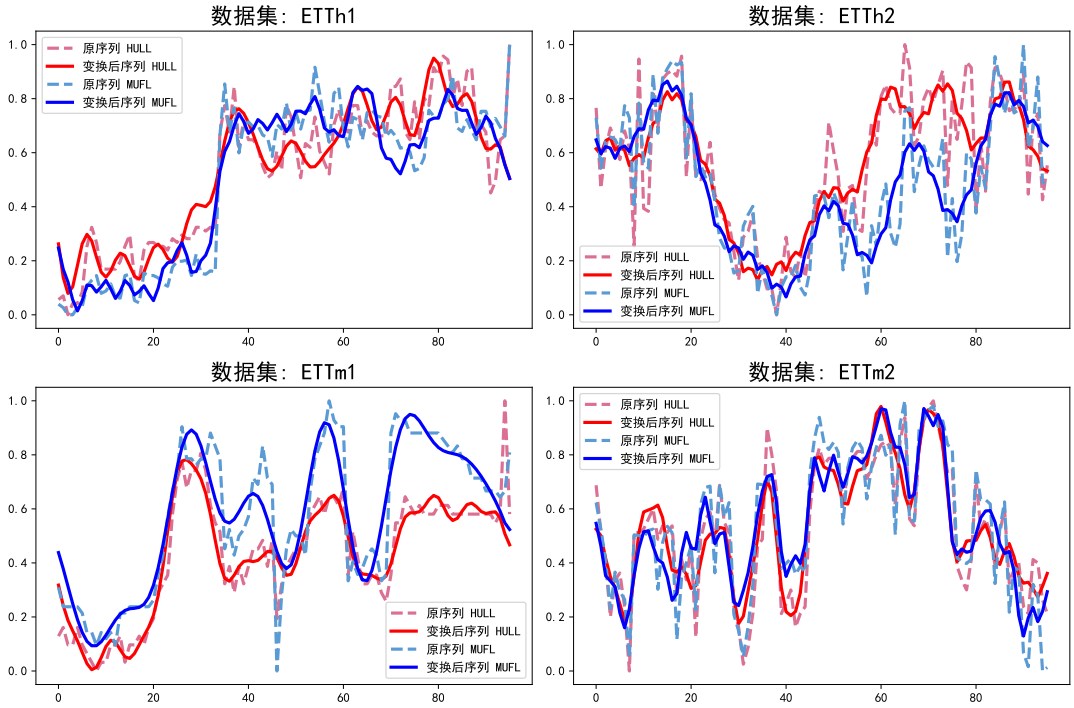



图 4-6 作用谱滤波变换模块前后的时间序列对比

图4-6 展示了对谱滤波变换模块的作用分析，其中实线代表了作用这个模块后的序列，而对应颜色的虚线表示了作用这个模块前的序列。本文选择了 ETT 数据集的四个子集（ETTh1、ETTh2、ETTm1、ETTm2）中的其中两个变量（HULL、MUFL）进行绘制，固定输入数据选择数据集前 96 个连续样本点，滤波选择最显著的 8 个频率，汉明窗口大小为 3。从图中可以分析得到以下结论：

(1) **平滑性增强**：四个数据集上都展现了实线比虚线更平滑，说明谱滤波变换有效地去除了数据的高频噪声，这种平滑性增强是通过滤波器保留显著频率成分来实现的。

(2) **复杂波动减少**：实线在整个时间序列中的波动比虚线小，这表明谱滤波变换抑制了非主要频率成分，使得主要趋势更加突出。

(3) **主要趋势提取**：从实线可以更清晰地观察到序列的主要趋势，谱滤波变换有助于更好地分析时间序列的基本模式，如周期性或趋势性变化。

(4) **序列连续性强化**：经过谱滤波变换的序列显示出更好的连续性，有助于模型在窗口数据中识别出更加直观的时间模式。

这种去噪和平滑效果对于后续提取语义时间模式（如趋势和连续性）至关重要，同时对于识别多变量相关性也是必不可少的，进而有助于精确的序列预测。本文认为，由于长期预测的原理是通过一个固定大小的窗口对未来一个窗口的时序数据进行预测，在考虑窗口大小为 96 的情况下，由于窗口大小相对完整序列实在是过小，模型其实很难捕捉到隐藏在细微波动中的隐式时序信息，而对于长期预测模型来说更重要的则是确保对整体宏观特征的捕捉（趋势和连续性等）。本文认为这也是为什么过去一年中基于 MLP 的时序长期预测模型能够取得很好的效果：一批固定的权重和偏置可以对历史的一小段序列进行统一编码，保留了其宏观的时序特征（趋势和连续性等），进而能够确保模型预测的未来窗口能够保持和输入窗口序列相似的时序模式。考虑到领域内常用的数据集的预处理十分完善，数据的突变和异常值较少，导致相邻窗口数据分布差异不大，复杂的、隐式的时序特征在这个过程中难以发挥作用，使得宏观时序特征更有利于时序预测。因此，本文设计了谱滤波变换模块实现对宏观时序信息的有效识别，进而帮助模型实现直接、有效的时序预测。

### 4.3.2 动态定向注意力模块

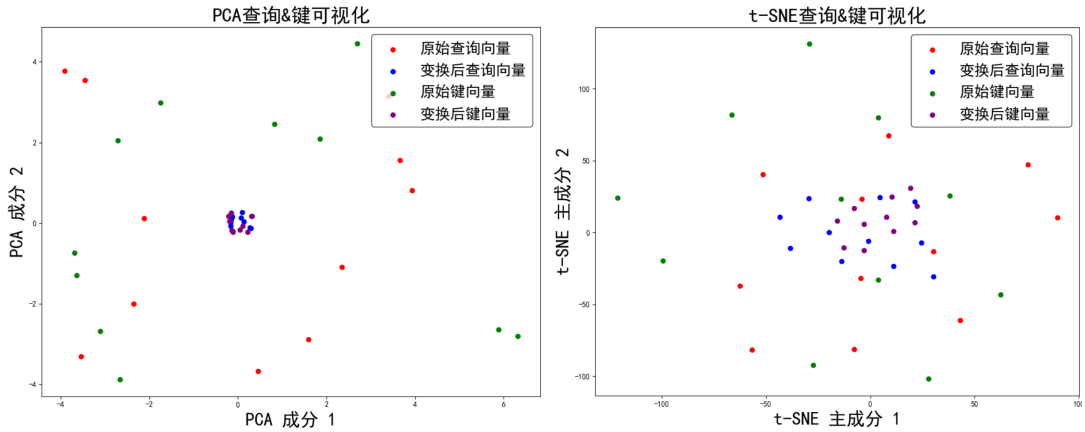



图 4-7 查询向量和键向量在作用核函数之后的相似性可视化

为了验证动态定向注意力模块，尤其是其特殊核函数 $f_p$ 的有效性，图4-7以散点形式展示了作用核函数前后的查询（Query）和键（Key）向量的相似程度。之所以探究相似程度的变化是因为考虑到注意力分数的计算是通过计算查询和键的点积实现的，其实也就是查询和键的余弦相似度。相似度越高，则注意力分数越高。考虑到查询和键是高维张量，所以本文使用两种常见的降维方法 PCA 和 t-SNE（这两种降维方法通过 scikit-learn 实现[75]）将查询和键映射到二维空间变成散点以便直观理解。在图4-7的左侧子图中，作用该核函数后的查询向量和键向量的散点显示出比作用核函数之前更密集的分组，这一现象在右侧子图中得到了印证，其中作用核函数后的散点图同样更加集中。这种密集性表明核函数有效地拉近了相似的查询和键，使得原本距离靠近的查询和键计算得到的注意力分数变得更高，进而实现了变量之间更为显著的注意力分布。值得

说明的是，本文给出的例子是查询和键距离拉近的例子，实际的实验过程中还有很大一部分查询和键的散点分布变化不显著甚至趋于离散，此时注意力分数相应降低。

### 4.3.3 多模块共同作用效果分析

本文还从两者联合的角度评估了提出的谱滤波变换和动态定向注意力机制，探究了二者在解决注意力分数分布均匀问题上的作用。图4-8展示了两张注意力图（Attention Map）以及两组时间序列。由于注意力图的尺寸较大（即 866×866），本文截取了整个图的一个局部区域，以便更清楚地展示每个图中各个“像素”的颜色差异。具体来说，本文随机选择了两组时间序列 Case 1 和 Case 2，其中显然 Case 1 中两个序列之间的相似性高于 Case 2。随后可以比较对应于 Case 1 的 SDformer 和 iTransformer 的注意力图区域，结果显示 SDformer 对于相似序列分配了比 iTransformer 更高的注意力分数（“像素”颜色更深），而对 Case 2 中不相似的序列，SDformer 的注意力分数相对较低（“像素”颜色更浅）。这种差异表明谱滤波变换和动态定向注意力机制共同提高了相似查询-键对的注意力分数，同时降低了不相似对的分数。这一结果展示了 SDformer 在众多变量中更为独特和不均匀的注意力分配能力，使其能更有效地挖掘变量间的相关性。

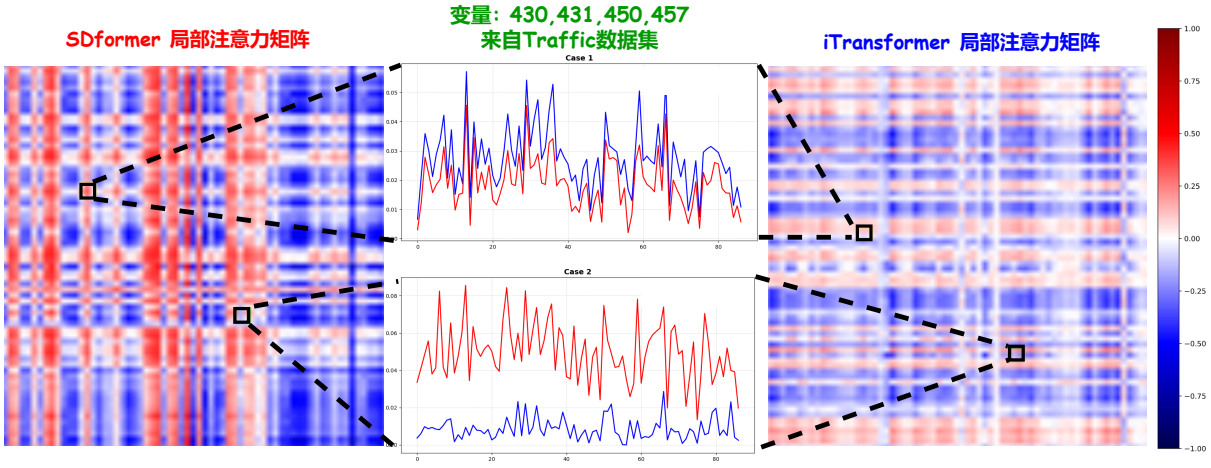



图 4-8 Traffic 数据集中两组时间序列的注意力图可视化

为了进一步说明本文提出的谱滤波变换和动态定向注意力机制在解决平滑注意力分布方面的有效性，本文分别对拥有四个编码器层的两个模型进行训练，计算了 SDformer 和 iTransformer 每一层的注意力分数矩阵的基尼系数和秩，如表 4-3 所总结。这两个指标用于评估模型在分析注意力权重分布时是否具有优先考虑重要变量间相关性的能力，其中更高的基尼系数和秩表明分布更不均匀。结果显示，SDformer 在所有层中的基尼系数一致高于 iTransformer，显示了更集中的注意力权重分布。不难看到，SDformer 在编码器第四层的注意力矩阵的基尼系数达到了 **0.268**，反观 iTransformer 的基尼系数仅为 **0.095**。此外，SDformer 在第四层的秩达到 **459**，超过了 iTransformer 的 **296**，这表明了动态定向注意力的特征表示更具有多样性。这些数值上的差异验证了谱滤波变换和

动态定向注意力机制在克服注意力矩阵同质化问题的有效性，彰显了 SDformer 在识别关键变量以捕获多变量相关性方面的能力。

表 4-3 iTransformer 和 SDformer 在每个编码器层的注意力矩阵的基尼系数和秩

| 模型 | iTransformer | | SDformer | |
|---|---|---|---|---|
| 评价指标 | 基尼系数 | 秩 | 基尼系数 | 秩 |
| 第 1 层 | 0.078 | 260 | **0.154** | **344** |
| 第 2 层 | 0.086 | 281 | **0.244** | **375** |
| 第 3 层 | 0.104 | 302 | **0.223** | **365** |
| 第 4 层 | 0.095 | 296 | **0.268** | **459** |

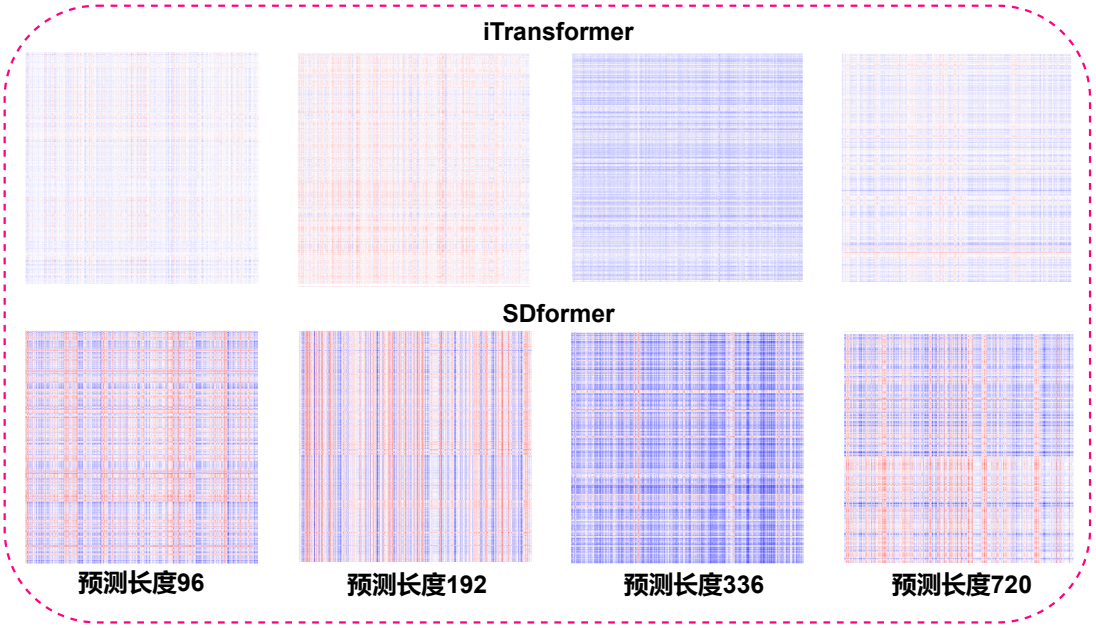



图 4-9 Traffic 数据集上不同预测长度的注意力图可视化

此外，为了更直观地展示 SDformer 在解决注意力分数分布过于均匀问题的有效性，本文在 Traffic 数据集上重新训练了具有四个编码器层的 iTransformer 和 SDformer 模型，并在不同预测长度的设置下绘制了 iTransformer 和 SDformer 同一编码器层的相同头的注意力图，即图4-9。不难看出，iTransformer 所使用的传统自注意力机制容易引起注意力图的行同质化，直观体现为注意力图的颜色较浅且分布过于均匀，说明此时的注意力分数集中在 0 附近的一个较小邻域内（经过归一化后），此时注意力机制没有能够捕捉 Token 之间的相对重要性关系。而 SDformer 的注意力图能够明显看出更深的颜色，以及更有区分度的色块分布，说明动态定向注意力机制能够改善行同质化的问题，即注意力机制能够挖掘到 Token 之间的相对重要性关系。综上所述，得益于创新性的谱滤波变换和动态定向注意力机制，SDformer 可以有效突破传统自注意力机制在建模多变量时间序列时的能力瓶颈，为使用 Transformer 模型进行时间序列预测提供新的解决方案。

## 4.4 消融实验与分析

消融实验可以有效验证模型中的模块化设计是否真正对整体模型带来正向的效果，验证模块的有效性，从而更全面地探究模型整体的创新型。本文在 Weather、ETTm2 和 Exchange 数据集上进行了消融实验，从两个方面验证本文提出的谱滤波变换模块和动态定向注意力模块的有效性和必要性：（1）去除动态定向注意力：SDformer 中的动态定向注意力被替换为普通的自注意力；（2）去除谱滤波变换模块：即移除谱滤波变换模块。消融实验的实验设计均与4.2节的主要实验一致。实验结果上，表4-4总结了三个数据集上消融实验的结果，不难发现其中移除任何模块组件都会导致性能下降（更高的 MSE）。

表 4-4 SDformer 中创新性模块的消融实验结果

| 模型类别 | | 本文完整方法 | | 去除动态定向注意力 | | 去除谱滤波变换模块 | |
|---|---|---|---|---|---|---|---|
| 评价指标 | | MSE | MAE | MSE | MAE | MSE | MAE |
| *Weather* | 96 | 0.171 | 0.210 | 0.182 | 0.224 | 0.170 | 0.211 |
| | 192 | 0.222 | 0.255 | 0.226 | 0.260 | 0.224 | 0.255 |
| | 336 | 0.278 | 0.297 | 0.281 | 0.298 | 0.280 | 0.298 |
| | 720 | 0.358 | 0.348 | 0.359 | 0.351 | 0.360 | 0.350 |
| | Avg | **0.258** | **0.278** | 0.262 | 0.281 | 0.259 | 0.279 |
| *ETTm2* | 96 | 0.183 | 0.268 | 0.183 | 0.269 | 0.184 | 0.268 |
| | 192 | 0.249 | 0.309 | 0.253 | 0.312 | 0.251 | 0.311 |
| | 336 | 0.313 | 0.348 | 0.313 | 0.349 | 0.314 | 0.350 |
| | 720 | 0.407 | 0.402 | 0.413 | 0.405 | 0.408 | 0.404 |
| | Avg | **0.288** | **0.332** | 0.291 | 0.334 | 0.289 | 0.333 |
| *Exchange* | 96 | 0.087 | 0.208 | 0.087 | 0.206 | 0.088 | 0.209 |
| | 192 | 0.177 | 0.300 | 0.180 | 0.303 | 0.178 | 0.301 |
| | 336 | 0.331 | 0.418 | 0.337 | 0.421 | 0.332 | 0.419 |
| | 720 | 0.829 | 0.688 | 0.849 | 0.696 | 0.843 | 0.694 |
| | Avg | **0.356** | **0.404** | 0.363 | 0.407 | 0.360 | 0.406 |

这样的结果表明，在没有谱滤波变换模块的情况下，模型更容易受到环境或白噪声的干扰，难以准确捕捉时间序列数据中的时序信息，从而对预测准确性产生负面影响。此外，不难观察到用普通的自注意力替换动态定向注意力也会导致更高的预测误差，这表明普通自注意力在建模具有众多变量的多变量时间序列方面不如本文提出的动态定向注意力有效。总之，这些结果共同表明了这两个模块化组成部分的重要性，它们共同提高了模型在多变量时间序列预测中的性能。

## 4.5 超参敏感性分析

超参数作为模型中重要的初始化参数，很大程度上直接决定了模型的性能，超参数的正确设置能够最大程度地挖掘模型的性能上限。在本节中，谱滤波模块、动态定向注意力模块以及注意力编码器中的几个超参数将会被细致分析。除了分析这些超参数会如何影响模型在预测任务上的性能，本文还将根据模块的实现原理分析不同超参数对模型实际的影响机制，实现对模型原理的深入解析。

### 4.5.1 谱滤波变换模块超参分析

在谱滤波变换模块中，滤波部分的 Top-$k$ 值直接决定了模型对主要时序信息的保留程度，表示为频域内保留的最高幅值数量。时域内作用窗函数的窗口大小则直接决定了加窗卷积所带来的平滑效果。表4-5中的超参数分析研究了谱滤波变换模块中 $k$ 值和窗口大小在 Exchange 数据集上对模型效果的影响。可以发现更大的 $k$ 值可以使模型获得更低的 MSE，尤其有利于像预测未来 720 长度这样的长期预测。这样的结果表明，在长期预测中保留更多的频率成分对于捕捉复杂的时间动态变化至关重要。相反，较小的 $k$ 值选择可能会扭曲并破坏序列的关键时间特征，例如周期性和趋势。

表 4-5 谱滤波变换模块超参敏感性分析

| 窗口大小 | | 2 | | 10 | | 18 | | 26 | |
|---|---|---|---|---|---|---|---|---|---|
| 评价指标 | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| 幅值数为 8 | 96 | 0.089 | 0.210 | 0.089 | 0.210 | 0.088 | 0.210 | 0.088 | 0.210 |
| | 192 | 0.179 | 0.302 | 0.179 | 0.302 | 0.180 | 0.303 | 0.181 | 0.304 |
| | 336 | 0.335 | 0.421 | 0.337 | 0.422 | 0.335 | 0.421 | 0.334 | 0.421 |
| | 720 | 0.839 | 0.693 | 0.832 | 0.690 | 0.845 | 0.693 | 0.843 | 0.693 |
| | Avg | 0.361 | 0.407 | **0.359** | **0.406** | 0.362 | 0.407 | 0.362 | 0.407 |
| 最高幅值 | | 4 | | 12 | | 20 | | 28 | |
| 评价指标 | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| 窗口大小为 4 | 96 | 0.091 | 0.214 | 0.088 | 0.209 | 0.089 | 0.210 | 0.087 | 0.208 |
| | 192 | 0.189 | 0.313 | 0.179 | 0.301 | 0.178 | 0.301 | 0.178 | 0.301 |
| | 336 | 0.341 | 0.426 | 0.335 | 0.421 | 0.333 | 0.420 | 0.333 | 0.420 |
| | 720 | 0.854 | 0.698 | 0.835 | 0.691 | 0.835 | 0.691 | 0.837 | 0.692 |
| | Avg | 0.369 | 0.413 | **0.359** | 0.406 | 0.359 | 0.406 | 0.359 | **0.405** |

此外，表4-5还显示谱滤波变换模块中汉明窗口的较大窗口会导致更高的 MSE，这意味着较大的窗口可能会影响输入时间序列的固有特征。例如对于本文实验设置的固定长度为 96 回顾窗口，若汉明窗的大小超过 10，则此时窗函数已经占据了超过 10% 的输入长度，这在一定程度上破坏了输入序列的原有特征。但是，观察到谱滤波变换模块中 $k$ 值和窗口大小变化对 MSE 的变化并不显著，所以可以说 SDformer 表现出了超参上的鲁棒性，即尽管这些超参数导致模型的表现发生了一定变化，但总体仍然保持稳定的性能。因此本文的全部实验都进行了大范围的超参搜索，确保模型达到最优的表现。

### 4.5.2 注意力编码模块数量分析

注意力编码模块作为 SDformer 中最重要的模块化结构，起到了编码时间序列历史信息的作用，挖掘了历史序列中蕴含的时序信息。堆叠的注意力编码模块可以迭代地挖掘更深层次的时序信息，提高了网络的深度。图4-10展示了 SDformer 中注意力编码模块数量对模型性能的影响。作为模型中重要的超参数之一，注意力编码模块数量直接决定了模型的结构以及复杂度，进而决定了模型能够捕获有效的时序信息的能力。从图中可

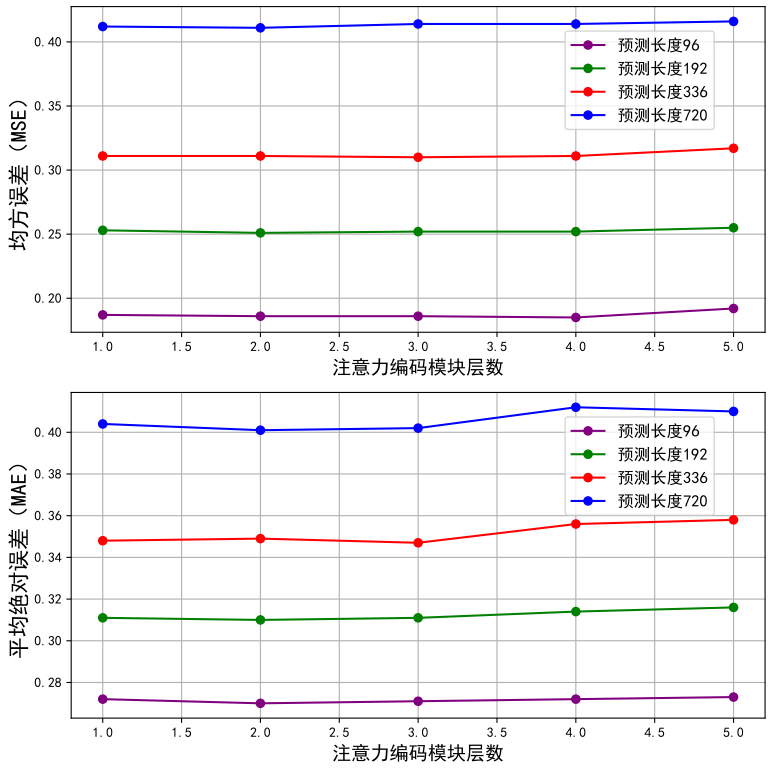



图 4-10 注意力编码模块数量对模型预测效果的影响

以看到，在基于注意力机制的模型中，注意力编码模块的数量对模型性能有显著影响。实验表明，当注意力编码模块的数量为 2 时，模型性能最佳，而数量较少（如 1）或较多（如 4 或 5）时，性能反而下降。当注意力编码模块的数量较少时，如仅有 1 层，模型的学习能力受限。在这种情况下，模型可能无法充分捕捉到数据中的复杂特征和深层关系，导致无法形成足够丰富和精准的数据表示，从而影响最终的性能。相反，当模块数量过多时，如 4 或 5 层，可能导致模型过于复杂，引起过拟合。在这种情况下，模型可能会过度适应训练数据中的特定特征和噪声，而在新的或未见数据上表现不佳。此外，过多的层数还可能导致梯度消失或爆炸问题，使得训练过程变得困难。经过多次实验，本文发现当使用 2 层注意力编码模块时模型达到了一种“平衡点”，既能够有效提取和利用数据的关键特征，又避免了过度复杂化的问题。这种平衡使得模型可以更好地泛化到新数据，从而在多种任务中表现出良好的性能。

### 4.5.3 动态定向注意力超参分析

作为动态注意力机制中的重要参数，定向系数 $p$ 对该模块的性能产生了显著影响。定向系数 $p$ 在动态定向注意力机制中决定了模块“定向”的尺度，同时调整了查询向量和键向量的长度和方向，进而起到调节模型关注度的作用，不同的 $p$ 值决定了模型在处理输入数据时关注的重点和范围。从表4-6中可以看出，当 $p=2$ 时，对于两种不同的数据集（Exchange 和 ETTh2），模型在多个时间序列预测长度（96、192、336、720）上的

平均 MSE 和 MAE 均达到最低，表明此时模型性能最优。这意味着在 $p=2$ 的设置下，模型达到了对 Token 之间的关联性挖掘达到了最优效果，最大程度上识别并聚拢了相似的“查询-键”向量。当 $p$ 值较低（如 1）或较高（如 4 或 5）时，模型的 MSE 和 MAE 普遍高于 $p=2$ 的情况，此时较低的 $p$ 值导致模型无法充分捕获数据中的复杂关系，即无法有效定向查询向量和键向量，而过高的 $p$ 值则导致向量本身的分布被破坏，进而使得计算得到的注意力分数无法有效表征变量之间的重要性关系。此外可以发现两个数据集对 $p$ 值的敏感度略有不同，表明定向系数 $p$ 的最优值与具体数据集中的数据分布特点相关。因此本文在实验中针对具体的数据集动态调整了 $p$ 值，以获得最佳性能。

表 4-6 动态定向注意力中定向系数 $p$ 对模型效果的影响分析

| p 值 | | 1 | | 2 | | 3 | | 4 | | 5 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 评价指标 | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Exchange | 96 | 0.087 | 0.208 | 0.087 | 0.208 | 0.088 | 0.208 | 0.087 | 0.207 | 0.087 | 0.207 |
| | 192 | 0.178 | 0.301 | 0.178 | 0.301 | 0.178 | 0.301 | 0.178 | 0.301 | 0.180 | 0.303 |
| | 336 | 0.335 | 0.421 | 0.333 | 0.420 | 0.332 | 0.418 | 0.333 | 0.419 | 0.334 | 0.420 |
| | 720 | 0.840 | 0.693 | 0.837 | 0.692 | 0.842 | 0.694 | 0.845 | 0.695 | 0.853 | 0.697 |
| | Avg | 0.360 | 0.406 | **0.359** | **0.405** | 0.360 | 0.405 | 0.361 | 0.405 | 0.363 | 0.407 |
| p 值 | | 1 | | 2 | | 3 | | 4 | | 5 | |
| 评价指标 | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh2 | 96 | 0.305 | 0.349 | 0.305 | 0.349 | 0.305 | 0.350 | 0.305 | 0.350 | 0.305 | 0.350 |
| | 192 | 0.379 | 0.395 | 0.378 | 0.394 | 0.383 | 0.396 | 0.383 | 0.397 | 0.383 | 0.397 |
| | 336 | 0.436 | 0.439 | 0.419 | 0.427 | 0.420 | 0.428 | 0.422 | 0.430 | 0.421 | 0.429 |
| | 720 | 0.422 | 0.439 | 0.418 | 0.438 | 0.422 | 0.440 | 0.428 | 0.444 | 0.424 | 0.441 |
| | Avg | 0.385 | 0.406 | **0.380** | **0.402** | 0.382 | 0.403 | 0.384 | 0.405 | 0.383 | 0.404 |

## 4.6 模型复杂性分析

模型的复杂性也是本文重点关注的问题。前文曾经提到，传统 Transformer 模型在建模时间序列时常受限于自注意力机制的二次复杂度，导致模型受限于计算资源瓶颈。因此 SDformer 通过仅保留编码器结构以及反转嵌入的思路分别实现了模型轻量化设计以及 Token 数量的有效减少（Token 数量和时间序列变量数量相同），使得 SDformer 可以在 Token 数量大大降低的前提下接受二次复杂度。表4-7记录了在 Traffic 数据集上与

表 4-7 SDformer 与其他最优模型的计算开销对比

| 模型 | 参数量 | 时间 | 乘积累加运算 | 浮点运算 |
|---|---|---|---|---|
| SDformer | 6.41M | 165ms | 44.18G | 88.38G |
| iTransformer | 6.41M | 107ms | 44.19G | 88.38G |
| PatchTST | 6.91M | 353ms | 526.89G | 1053.80G |
| Crossformer | 54.46M | 492ms | 1597.86G | 3195.72G |
| TimesNet | 304.25M | 1039ms | 3122.55G | 6245.02G |

近年来效果最优的时序预测模型的复杂性比较。复杂性对比的实验设置包括 96 长度的

回顾窗口和 96 长度的预测。时间指的是 3 次迭代的平均值，所有实验都在相同的设备上进行以确保公平性。具体来说，SDformer 在资源利用和计算效率之间实现了最佳的均衡。与当前的轻量级 iTransformer[12] 相比，SDformer 展示了更优越的预测性能，同时具有相似的参数大小，这表明了 SDformer 实现了轻量设计与增强的表征能力之间的兼顾。此外，与诸如 PatchTST[13] 或 TimesNet[51] 等其他在时间序列长期预测任务上表现最优的模型相比，SDformer 在 MACs（乘积累加运算）和 FLOPs（浮点运算）上都显示出显著的减少，进一步证明了该模型拥有更低的计算开销。与 TimesNet[51] 相比，SDformer 的轻量性设计更为显著，SDformer 不仅在参数量上减少了将近 50 倍，而且在乘积累加运算上也减少了超过 70 倍，这再次证明了其在保持预测准确性的同时，极大地优化了资源消耗。总体来说，凭借其适度的内存需求和较高的模型推理速度，SDformer 即使作为一种基于 Transformer 的时序模型，也能够在模型复杂性和内存使用之间取得平衡，并同时保证了卓越的预测结果。

## 4.7 本章小结

本章围绕实验展开。首先在4.1节介绍了实验中采用的领域内公认的时间序列数据集，介绍了 11 个用于对比的基线模型，阐述了评价指标 MSE 和 MAE，以及实验环境和超参设置等关键信息；4.2节详细说明了本文提出的 SDformer 在时间序列长期预测任务上的完整实验结果，展示了本文模型在该任务上的优越性；4.3节详细分析了 SDformer 模型中两个关键模块的具体效果，以及多模块共同作用的效果；4.4节针对模型中的模块化结构设计了多组消融实验，通过对消融实验结果的分析证明了该模块的不可或缺性；4.5节着重讨论了模块中的 4 个重要的超参数如何对实验结果产生影响；4.6节分析了模型的复杂性，证明了其轻量性。

# 总结与展望

时间序列在人类生活中广泛存在，准确的时间序列预测对于人类的生产生活至关重要。本文回顾了时间序列预测问题的研究背景和意义，介绍了机器学习模型以及以 Transformer 为代表的深度学习模型在该领域的应用。同时发现 Transformer 在处理多变量时间序列数据时仍面临挑战，特别是在处理具有大量变量的时间序列数据时，传统 Transformer 模型注意力权重分布过于分散，导致注意力机制失效，进而影响预测结果。

本文针对现有模型在时间序列预测的挑战提出了一种新颖的类 Transformer 结构，命名为 SDformer。通过引入谱滤波变换（Spectral-Filter-Transform）和动态定向注意力（Dynamic-Directional-Attention）两个创新模块，SDformer 在增强时间序列数据的去噪和平滑处理的同时实现了对注意力权重分布的尖锐化，有效提升了模型对多变量时间序列中关键变量的识别能力。具体实现方面，谱滤波变换模块在频域内通过滤波对时序数据进行有效的去噪，在时域内通过引入窗函数进行平滑处理，以保留数据的基本时序特征。同时，动态定向注意力模块引入了一种新颖的注意力分配策略，通过动态调整注意力权重集中于最具信息量的变量，显著增强了模型的表征学习能力和预测精度。SDformer 在多个公开数据集上的实验展现了其在长期预测任务中的优异性能，相比现有的先进模型实现了显著的性能提升。这些实验结果不仅证明了 SDformer 在提高时间序列预测精度方面的有效性，同时也展示了其在处理具有大量变量的复杂时间序列数据时的强大能力。此外，SDformer 有效解决了 Transformer 模型在多变量时间序列预测任务上出现的注意力分布过于均匀的问题，为处理模式复杂的多变量时间序列数据提供了新的解决方案。

尽管 SDformer 在预测任务上取得了优异的性能，但该模型仍具有一些可改进的空间。首先，SDformer 对通用时序特征的挖掘效果并不显著，尽管本文已经从多个角度证明 SDformer 的注意力编码模块可以有效提取时序表征，但仅在预测任务上证明了该表征的有效性。但是，在分类、缺失值填充和异常值检测等其他时间序列分析的子任务上，SDformer 并没有相较于其他 SOTA 模型展现出明显的优势。未来工作将探索 SDformer 在更大规模的真实世界数据集上的应用，例如人流量预测、出租车需求预测等方面，以及进一步优化模型结构以支持其在移动设备的部署。此外还需要进一步深化理论分析，为基于 Transformer 的时间序列分析提供更高效、准确的解决方案。

# 参考文献


[1] 张旭. 基于循环神经网络的时间序列预测方法研究[D]. 南京大学, 2019.

[2] Zhou Z, Chen H, Wang W. Short-term power load forecasting based on similar day selection and improved lstm[C]. EEBDA. 2023: 1610-1615.

[3] 田楚杰. 神经网络在时间序列与时空序列流量预测中的应用与研究[D]. 北京邮电大学, 2021.

[4] Yang Q, Wu X. 10 challenging problems in data mining research[J]. International Journal of Information Technology & Decision Making, 2006, 5:597-604.

[5] 陈璐. 基于 LSTM 模型的金融时间序列预测算法研究[D]. 哈尔滨工业大学, 2019.

[6] 原继东. 时间序列分类算法研究[D]. 北京交通大学, 2016.

[7] 王善辉. 双向循环神经网络在 GNSS 坐标时间序列插值中的研究[D]. 太原理工大学, 2019.

[8] 刘媛媛. 基于时间模式注意力机制的多维时间序列异常检测方法[D]. 西南财经大学, 2022.

[9] Zhou Z, Huang Y, Wang Y, et al. Stfm: Enhancing autism spectrum disorder classification through ensemble learning-based fusion of temporal and spatial fmri patterns[C]. PRICAI. 2023: 409-421.

[10] Huang Y, Zhou Z, Wang Z, et al. Timesnet-pm2. 5: Interpretable timesnet for disentangling intraperiod and interperiod variations in pm2. 5 prediction[J]. Atmosphere, 2023, 14(11):1604.

[11] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. NeurIPS, 2017, 30:6000–6010.

[12] Liu, Hu T, Zhang H, et al. itransformer: Inverted transformers are effective for time series forecasting [C]. ICLR. 2024: 1-24.

[13] Nie Y, H. Nguyen N, Sinthong P, et al. A time series is worth 64 words: Long-term forecasting with transformers[C]. ICLR. 2023: 1-24.

[14] Zhang Y, Yan J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting[C]. ICLR. 2022: 1-21.

[15] Ni Z, Yu H, Liu S, et al. Basisforme: Attention-based time series forecasting with learnable and interpretable basis[C]. NeurIPS. 2023: 1-20.

[16] Box G E, Jenkins G M. Some statistical aspects of adaptive optimization and control[J]. Journal of the Royal Statistical Society Series B: Statistical Methodology, 1962, 24(2):297-331.

[17] Moran P A. Hypothesis Testing in Time Series Analysis[J]. Royal Statistical Society. Journal. Series A: General, 2018, 114(4):579-579.

[18] Smith J W, Everhart J E, Dickson W, et al. Using the adap learning algorithm to forecast the onset of diabetes mellitus[C]. AMIA Annual Symposium Proceedings. 1988: 1-261.

[19] Williams R J, Zipser D. A learning algorithm for continually running fully recurrent neural networks [J]. Neural computation, 1989, 1(2):270-280.

[20] Müller K R, Smola A J, Rätsch G, et al. Predicting time series with support vector machines[C]. ICANN. 1997: 999-1004.

[21] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural Computation, 1997, 9(8):1735-1780.

[22] Singh R, Balasundaram S. Application of extreme learning machine method for time series analysis[J]. International Journal of Computer and Information Engineering, 2007, 1(11):3407-3413.

[23] Cho K, van Merrienboer B, Gülçehre Ç, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[C]. EMNLP. 2014: 1724-1734.

[24] Chorowski J, Bahdanau D, Cho K, et al. End-to-end continuous speech recognition using attention-based recurrent nn: First results[C]. NeurIPS Workshop on Deep Learning. 2014: 1-10.

[25] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[C]. ICLR. 2017: 1-14.

[26] Box G E, Jenkins G M. Time series analysis forecasting and control[J]. Technical Report, 1970:1-720.

[27] Zhang L, Aggarwal C, Qi G J. Stock price prediction via discovering multi-frequency trading patterns [C]. SIGKDD. 2017: 2141-2149.

[28] Bai L, Yao L, Li C, et al. Adaptive graph convolutional recurrent network for traffic forecasting[J]. NeurIPS, 2020, 33:17804-17815.

[29] Wu Z, Pan S, Long G, et al. Connecting the dots: Multivariate time series forecasting with graph neural networks[C]. SIGKDD. 2020: 753–763.

[30] Cao D, Wang Y, Duan J, et al. Spectral temporal graph neural network for multivariate time-series forecasting[J]. NeurIPS, 2020, 33:17766-17778.

[31] Woo G, Liu C, Sahoo D, et al. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting[C]. ICLR. 2022: 1-18.

[32] Yang Z, Yan W, Huang X, et al. Adaptive temporal-frequency network for time-series forecasting[J]. TKDE, 2020, 34(4):1576-1587.

[33] Oreshkin B N, Carpov D, Chapados N, et al. N-BEATS: neural basis expansion analysis for interpretable time series forecasting[C]. ICLR. 2020: 1-31.

[34] Zhang T, Zhang Y, Cao W, et al. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures[Z]. 2022: 1-11.

[35] Zeng A, Chen M, Zhang L, et al. Are transformers effective for time series forecasting?[J]. AAAI, 2023:11121–11128.

[36] Yi K, Zhang Q, Fan W, et al. Frequency-domain MLPs are more effective learners in time series forecasting[C]. NeurIPS. 2023: 1-24.

[37] Wen Q, Zhou T, Zhang C, et al. Transformers in time series: A survey[C]. IJCAI. 2023: 6778-6786.

[38] Li S, Jin X, Xuan Y, et al. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting[J]. NeurIPS, 2019, 32:1-11.

[39] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]. AAAI: volume 35. 2021: 11106-11115.

[40] Wu H, Xu J, Wang J, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting[J]. NeurIPS, 2021, 34:22419-22430.

[41] Liu S, Yu H, Liao C, et al. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting[C]. ICLR. 2021: 1-20.

[42] Chen W, Wang W, Peng B, et al. Learning to rotate: Quaternion transformer for complicated periodical time series forecasting[C]. SIGKDD. 2022: 146–156.

[43] Zhou T, Ma Z, Wen Q, et al. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting[C]. ICML. 2022: 27268-27286.

[44] Kitaev N, Kaiser L, Levskaya A. Reformer: The efficient transformer[C]. ICLR. 2020: 1-12.

[45] Chen Y, Ren K, Wang Y, et al. Contiformer: Continuous-time transformer for irregular time series modeling[J]. NeurIPS, 2023:1-33.

[46] Woo G, Liu C, Sahoo D, et al. Etsformer: Exponential smoothing transformers for time-series forecasting [J]. arXiv preprint arXiv:2202.01381, 2022:1-18.

[47] Hurley N, Rickard S. Comparing measures of sparsity[J]. IEEE Transactions on Information Theory, 2009, 55(10):4723-4741.

[48] Du Y, Wang J, Feng W, et al. Adarnn: Adaptive learning and forecasting of time series[C]. CIKM. 2021: 402–411.

[49] Kuznetsov V, Mohri M. Generalization bounds for time series prediction with non-stationary processes [C]. Auer P, Clark A, Zeugmann T, et al. Algorithmic Learning Theory. 2014: 260-274.

[50] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11):2278-2324.

[51] Wu H, Hu T, Liu Y, et al. Timesnet: Temporal 2d-variation modeling for general time series analysis [C]. ICLR. 2023: 1-23.

[52] Bishop C M, Bishop H. Deep learning: Foundations and concepts[M]. Springer, 2024.

[53] 辛洲扬. 基于 LSTM 与 Transformer 模型的股价预测[D]. 山东大学, 2022.

[54] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate [C]. ICLR. 2015: 1-15.

[55] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. CVPR. 2016: 770-778.

[56] Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016:1-14.

[57] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018:1-16.

[58] Radford A, Narasimhan K, Salimans T, et al. Improving language understanding by generative pre-training[J]. OpenAI blog, 2018:1-12.

[59] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[J]. OpenAI blog, 2019, 1(8):9.

[60] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners[J]. NeurIPS, 2020, 33: 1877-1901.

[61] Liu Y, Wu H, Wang J, et al. Non-stationary transformers: Exploring the stationarity in time series forecasting[J]. NeurIPS, 2022, 35:9881-9893.

[62] Zheng Y, Liu Q, Chen E, et al. Time series classification using multi-channels deep convolutional neural networks[C]. WAIM. 2014: 298-310.

[63] Miller J A, Aldosari M, Saeed F, et al. A survey of deep learning and foundation models for time series forecasting[J]. arXiv preprint arXiv:2401.13912, 2024:1-35.

[64] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: Transformers for image recognition at scale[C]. ICLR. 2021: 1-22.

[65] Mottaghi-Kashtiban M, Shayesteh M. New efficient window function, replacement for the hamming window[J]. IET Signal Processing, 2011, 5(5):499-505.

[66] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[C]. ICML. 2015: 448-456.

[67] Kim T, Kim J, Tae Y, et al. Reversible instance normalization for accurate time-series forecasting against distribution shift[C]. ICLR. 2021: 1-25.

[68] 梁宇轩. 基于多层注意力神经网络的地理传感器时间序列预测[D]. 西安电子科技大学, 2019.

[69] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks[C]. AISTATS. JMLR Workshop and Conference Proceedings, 2010: 249-256.

[70] Das A, Kong W, Leach A, et al. Long-term forecasting with tide: Time-series dense encoder[J]. arXiv preprint arXiv:2304.08424, 2023:1-18.

[71] Ekambaram V, Jati A, Nguyen N, et al. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting[C]. SIGKDD. 2023: 459–469.

[72] Hornik K. Approximation capabilities of multilayer feedforward networks[J]. Neural Networks, 1991, 4(2):251-257.

[73] Paszke A, Gross S, Massa F, et al. Pytorch: An imperative style, high-performance deep learning library [J]. NeurIPS, 2019, 32.

[74] Kingma D P, Ba J. Adam: A method for stochastic optimization[C]. ICLR. 2015: 1-15.

[75] Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: Machine learning in Python[J]. JMLR, 2011, 12:2825-2830.