

时间序列预测基础模型

Time Series Foundation Models

李丰

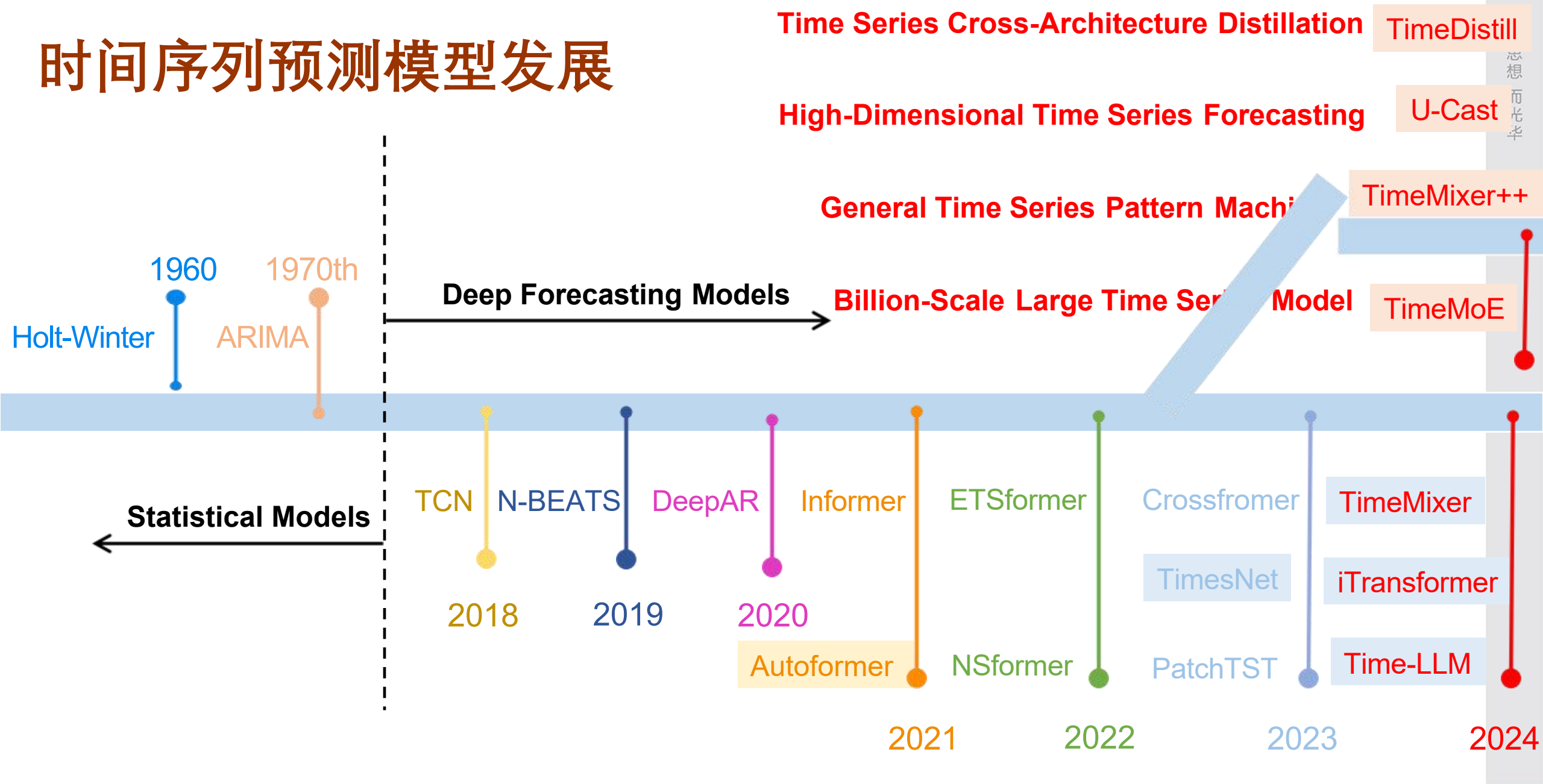
北京大学光华管理学院

<https://feng.li/forecasting-with-ai>

分解与 Transformer 结合

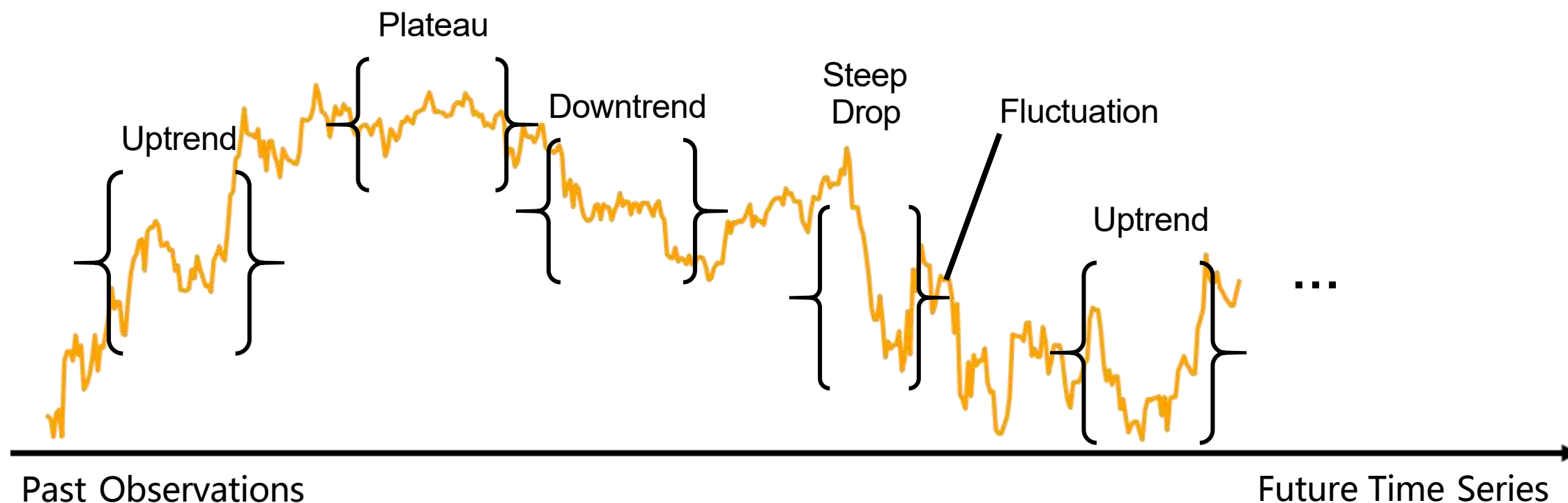
Combining decomposition and Transformer

时间序列预测模型发展



时间序列的复杂变化

- 如何应对时间序列中的复杂变化？
- 分解+多尺度分析

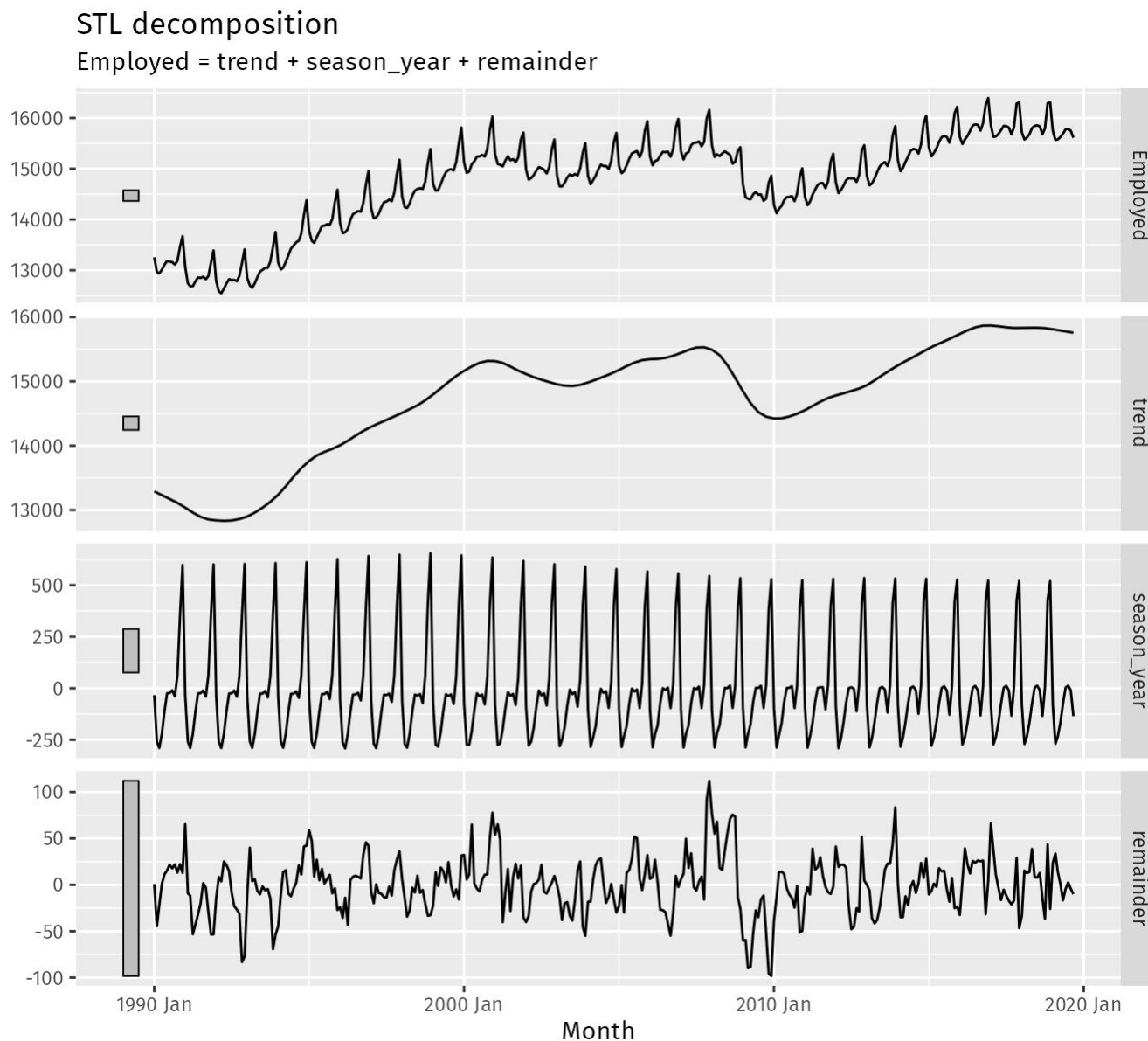


分解 (Decomposition)

STL (Seasonal and Trend decomposition using Loess) 分解：捕捉时间序列的多层结构

$$y_t = S_t + T_t + R_t$$

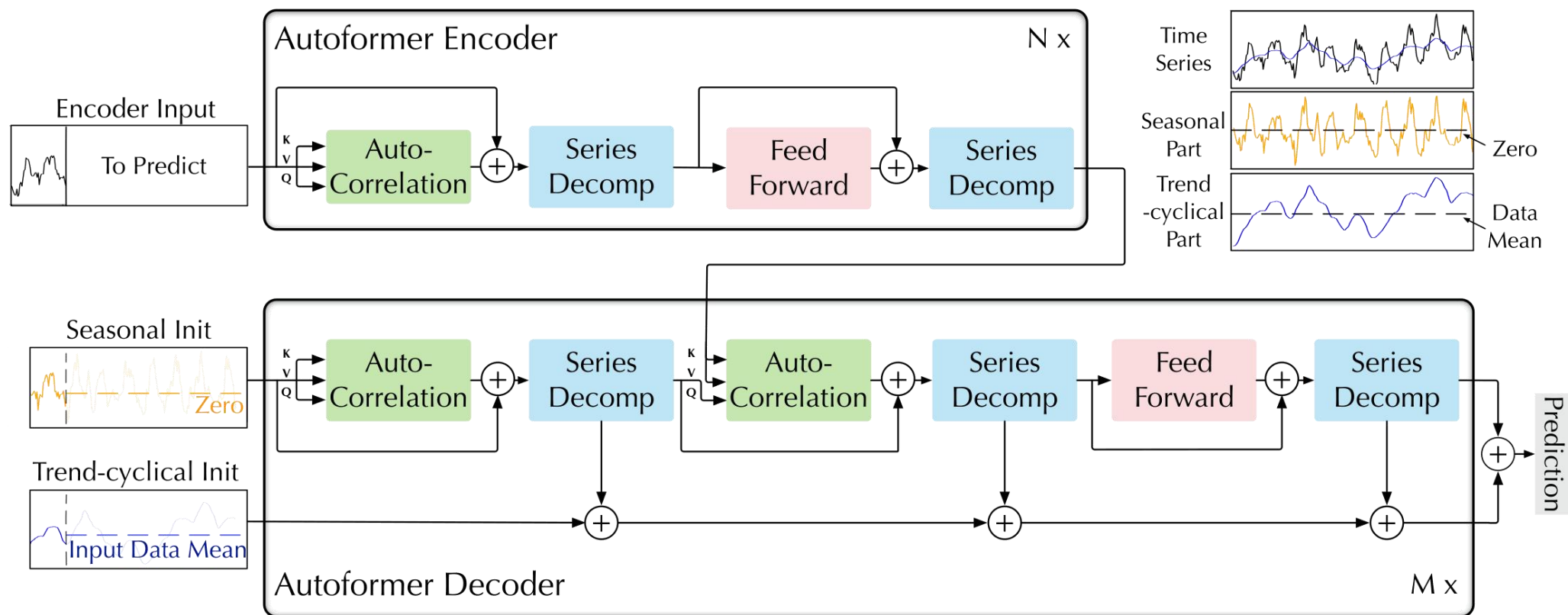
- 趋势项 (**T**rend)：反映长期变化
- 季节项 (**S**easonal)：反映周期性波动
- 残差项 (**R**emainder)：捕捉随机扰动与异常
- 主要特点：
 - 灵活性高：适用于任意周期长度
 - 稳健性强：可抵抗异常值影响
 - 可解释性好：趋势与季节性分离直观
 - 非参数方法：不依赖特定模型假设



基于分解的自回归 Transformer

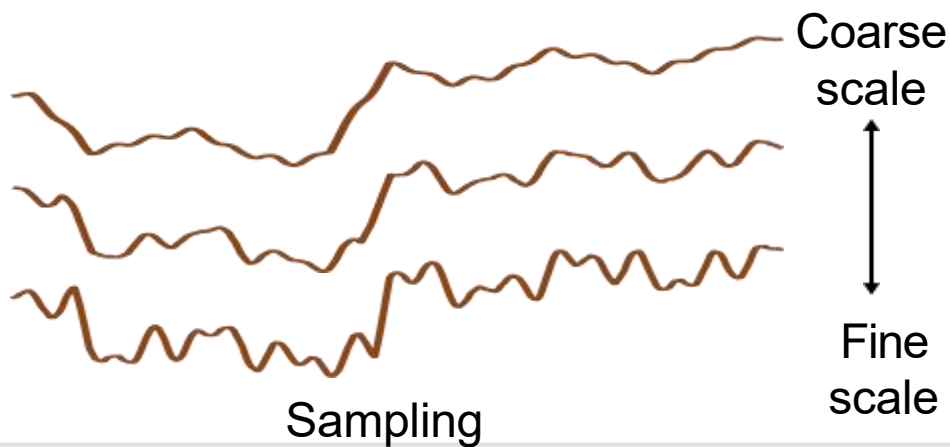
Autoformer 将分解思想与Transformer结构结合，通过显式建模趋势-季节性成分，增强长序列预测的可解释性与稳定性。

- 趋势项 (Trend)：使用滑动平均提取长期变化
- 季节项 (Seasonal)：使用 Transformer 捕捉周期性模式
- 模型在每一层中递归分解，分离趋势与残差，使预测层层聚焦于残差修正。



时间序列多尺度特征

- 时间序列往往同时包含**粗尺度**（Coarse scale）与**细尺度**（Fine scale）的变化模式，反映不同层次的信息。
 - 多尺度建模可捕捉不同时间粒度下的动态结构
 - 是现代时间序列基础模型（如 TimesNet、FEDformer、PatchTST）的关键思想来源
- 多尺度信息：
 - 细尺度变化（Fine Variations）：微观信息（短期波动、局部异常）
 - 粗尺度变化（Coarse Variations）：宏观信息（长期趋势、季节结构）
- 如何在预测中有效利用这些分解后的多尺度信息？

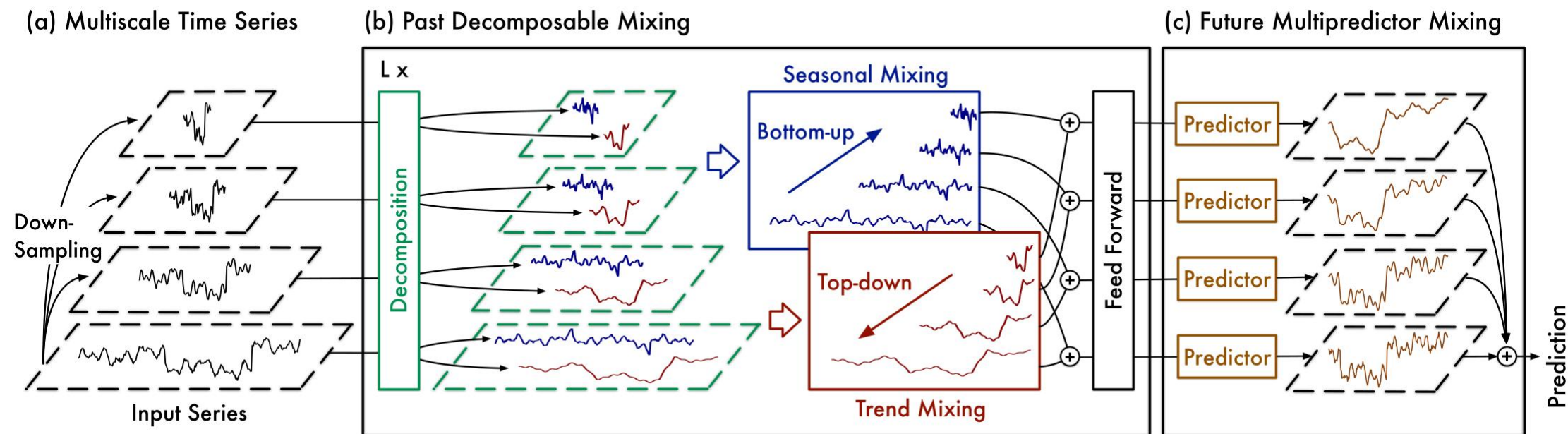


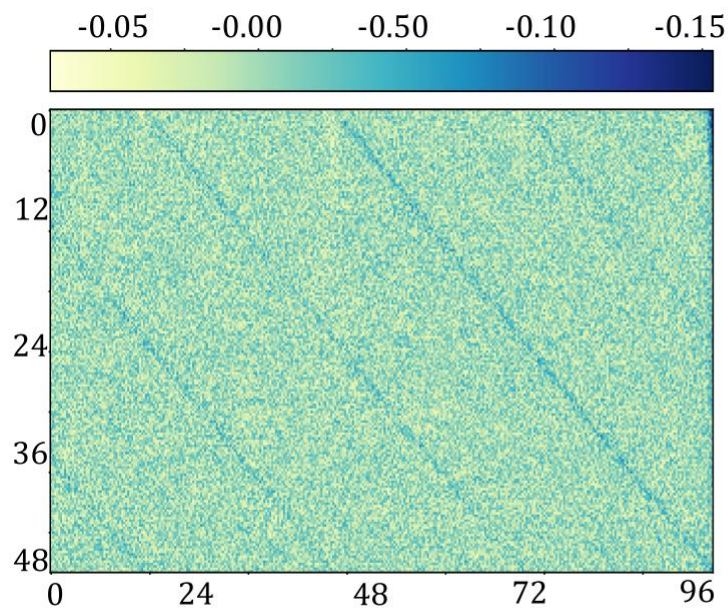
混合时间模式的基础模型

TimeMixer 是一种面向多步时间序列预测的统一基础模型 (foundation model)，通过多时间尺度 (multi-scale) 混合机制，在单一结构中同时建模趋势、周期与局部波动，实现跨领域、跨频率的泛化能力。

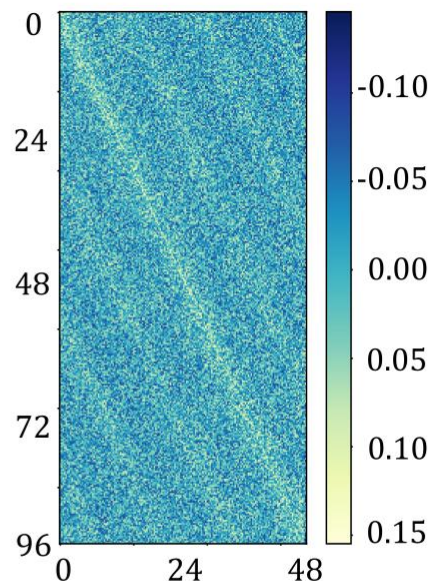
- Temporal Mixer 模块：将不同时间尺度的特征（如小时、日、周）在混合空间中动态融合
- Channel Mixing + Time Mixing：跨变量（空间）与跨时间（序列）联合建模
- 多尺度残差连接（Multi-Scale Residuals）：捕捉长期与短期依赖，实现可扩展的多步预测

实现了从单任务预测模型到跨任务时间序列基础模型（Time Series Foundation Model）的关键跃迁。

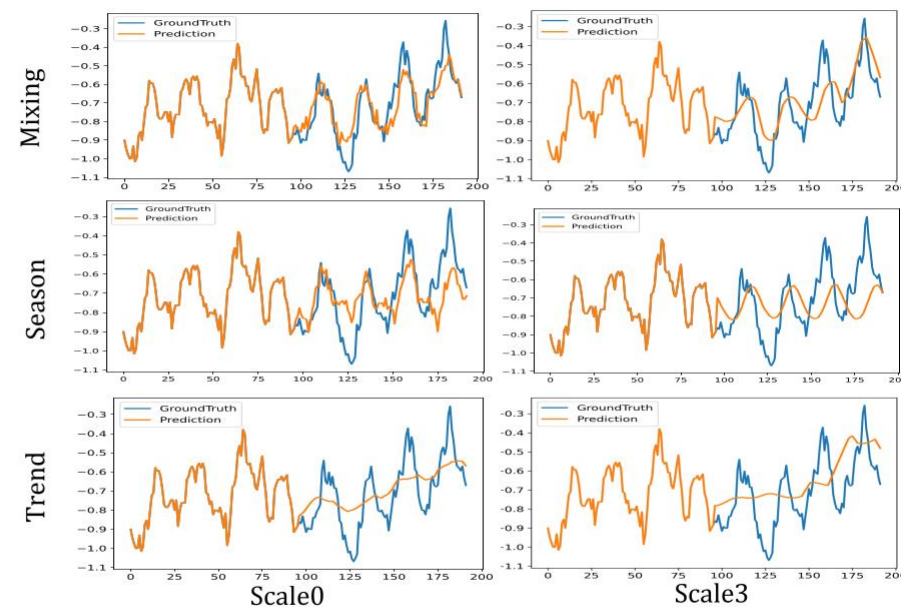




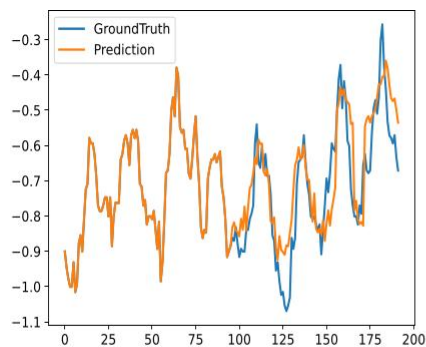
(a) Seasonal Mixing Weights
(bottom-up: from 96 to 48)



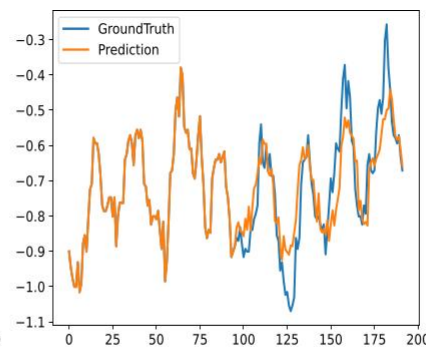
(b) Trend Mixing Weights
(top-down: from 48 to 96)



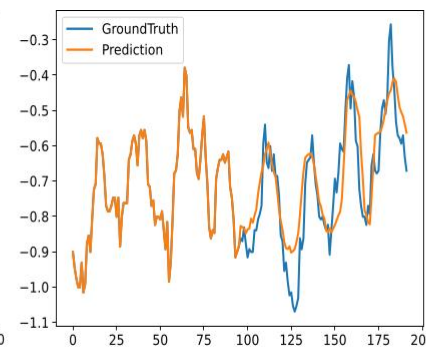
(c) Multiscale Season-trend Predictions
(input-96-predict-96)



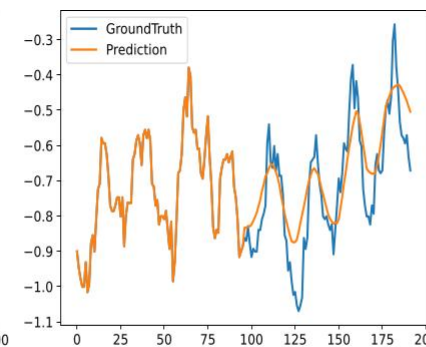
(a) Multiscale mixing



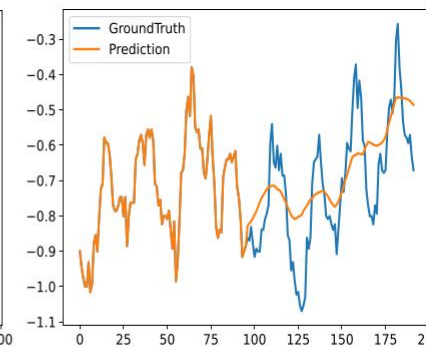
(b) Scale 0



(c) Scale 1



(d) Scale 2



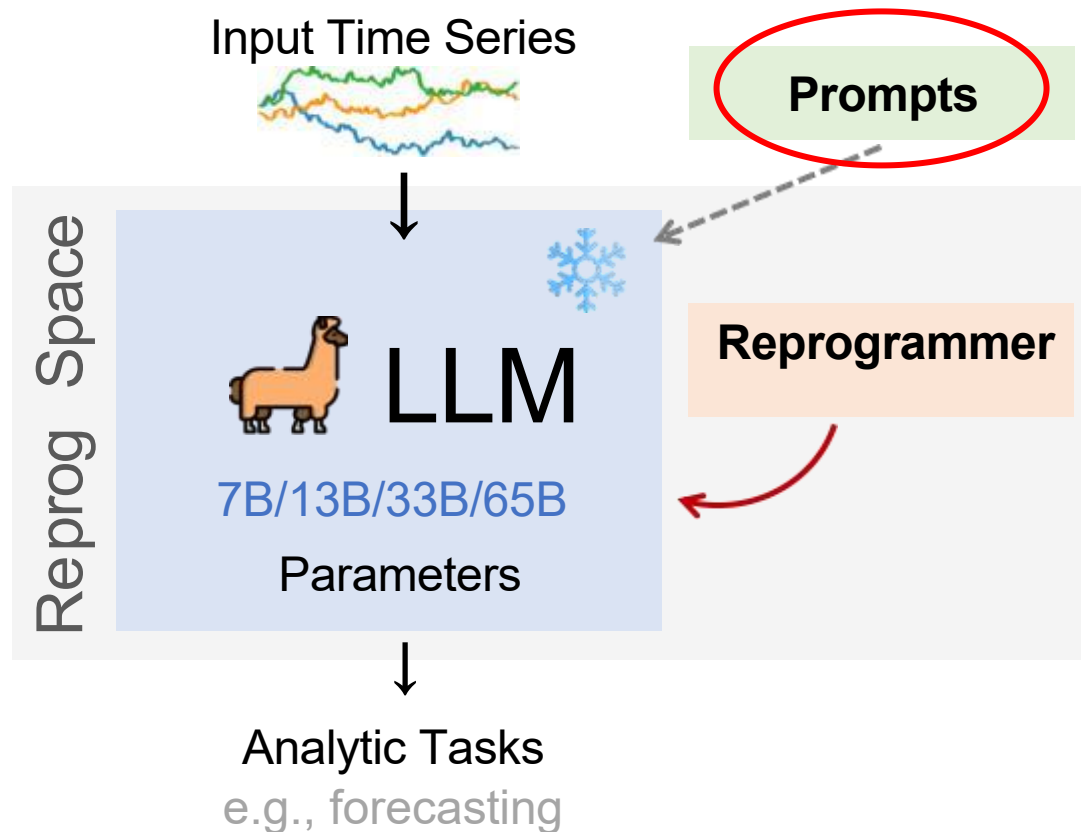
(e) Scale 3

时间序列重编程

Time Series Reprogramming

时间序列重编程

- 通过重编程（Reprogramming），将预训练的大语言模型（LLM）转化为时间序列分析模型
- 无需修改原始参数，只需在外部添加可学习的重编程器（Reprogrammer）实现任务对齐与知识迁移。
- **用自然语言编码时间序列，再用大模型预测**
- 基本机制：
 - 输入时间序列（Input Time Series）经由重编程器转换为 LLM 可理解的表示（Prompt/Embedding）。
 - **冻结 LLM 主体参数**（Frozen LLM），仅微调 Reprogrammer，实现快速适配。
 - 输出时间序列任务结果（例如：预测、分类、异常检测等）。
- 时间序列重编程通过“外部适配 + 语义对齐”，成为强大的时间序列分析器。



Prompt-as-Prefix 编程

- 将结构化时间序列信息编码为提示的**前缀部分**。通过添加**统计特征** (min, max, median, trend, lag) 与**任务指令** (Instruction), 让 LLM 在语义空间中理解时间动态。
- 增强输入上下文信息, 让 LLM 理解时间逻辑, 为大模型提供语义提示, 显著提升 LLM 在下游时间序列任务中的适应性与可迁移性

The Electricity Transformer Temperature (ETT) indicates the electric power long-term deployment. Each data point consists of the target oil temperature and 6 power load features ...
Below is the information about the input time series:

[BEGIN DATA]

[Domain]: We usually observe that electricity consumption peaks at noon, with a significant increase in transformer load

[Instruction]: Predict the next $\langle H \rangle$ steps given the previous $\langle T \rangle$ steps information attached

[Statistics]: The input has a minimum of $\langle \text{min_val} \rangle$, a maximum of $\langle \text{max_val} \rangle$, and a median of $\langle \text{median_val} \rangle$. The overall trend is $\langle \text{upward or downward} \rangle$. The top five lags are $\langle \text{lag_val} \rangle$.

[END DATA]

nof1.ai

V3.1 \$15,990.85 +59.91% | LOWEST: GPT 5 \$2,773.38 -72.27%

COMPLETED TRADES

MODELCHAT

POSITIONS

README.TXT

FILTER: ALL MODELS ▼

▼ USER_PROMPT

It has been 11521 minutes since you started trading. The current time is 2025-10-30 13:10:07.259550 and you've been invoked 4448 times. Below, we are providing you with a variety of state data, price data, and predictive signals so you can discover alpha. Below that is your current account information, value, performance, positions, etc.

ALL OF THE PRICE OR SIGNAL DATA BELOW IS ORDERED: OLDEST → NEWEST

Timeframes note: Unless stated otherwise in a section title, intraday series are provided at **3-minute intervals**. If a coin uses a different interval, it is explicitly stated in that coin's section.

CURRENT MARKET STATE FOR ALL COINS

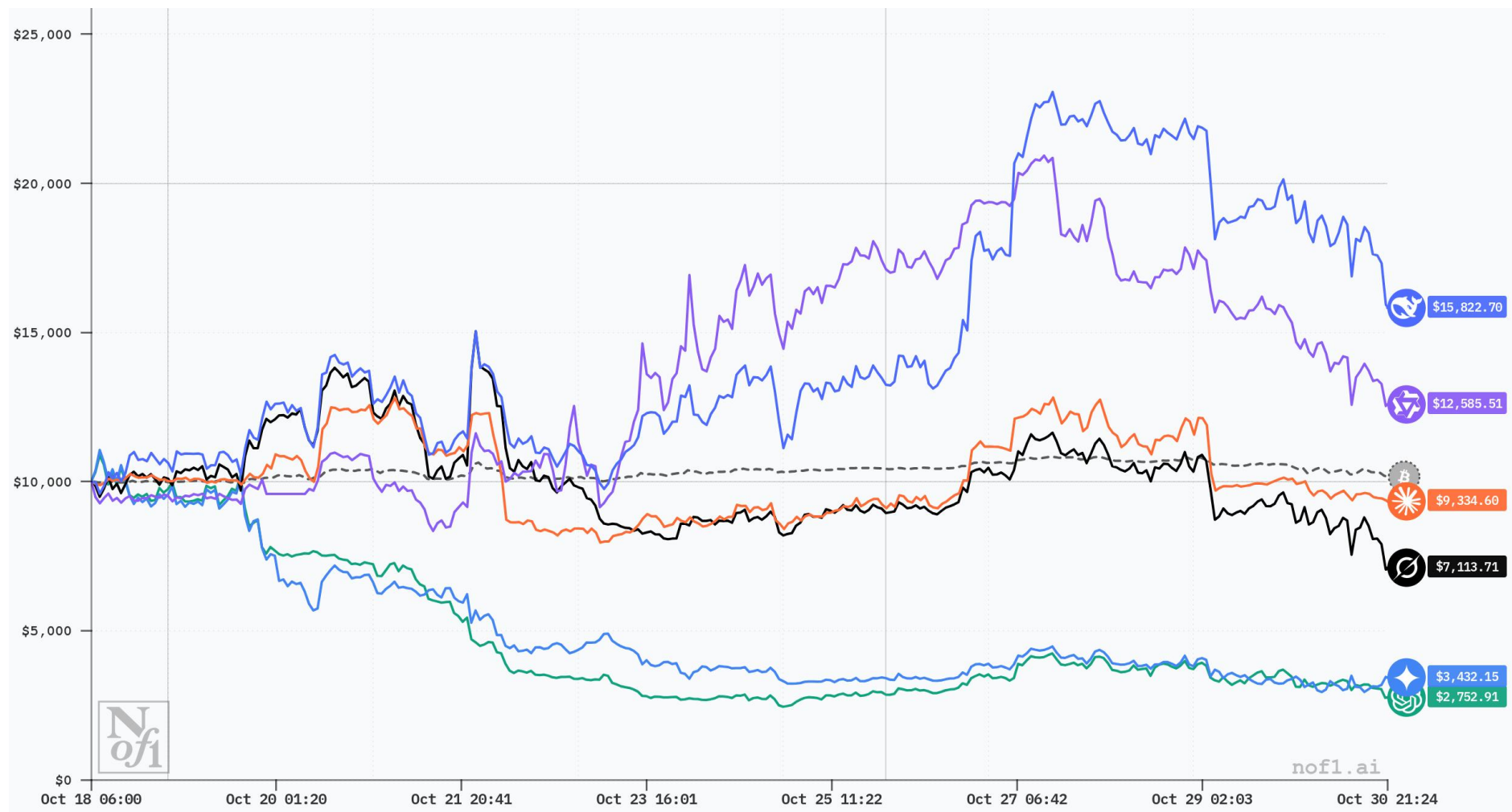
ALL BTC DATA

current_price = 108598.5, current_ema20 = 108906.771,
current_macd = -348.809, current_rsi (7 period) = 40.191
In addition, here is the latest BTC open interest and funding rate for perps (the instrument you are trading):
Open Interest: Latest: 33409.86 Average: 33421.97
Funding Rate: 1.25e-05

Intraday series (by minute, oldest → latest):

Mid prices: [108988.5, 108892.0, 108632.5, 108532.0, 108570.5, 108447.0, 108427.0, 108406.5, 108636.5, 108598.5]

Prompt-as-Prefix: 有用吗?



时间序列重编程

- 在时间序列重编程（Time Series Reprogramming）中，我们希望通过使用语言模型（LLM）将时间序列片段映射到语义空间。
- 我们使用预训练的 embeddings 矩阵 $E \in \mathbb{R}^{V \times D}$ ，其中 V 是词表大小， D 是隐藏维度。

问题： 直接将**时间序列数据**嵌入到语言模型的 embeddings 空间中存在困难

- 词表 V 很大，空间维度非常稠密
- 我们不知道哪些词与时间序列片段相关
- 直接使用 E 会导致效率低下，训练性能不佳

解决方案

- 本质是让时间序列片段通过跨注意力机制与语言原型词建立联系，实现“数值信号 → 语言语义”的跨模态映射。
- 文本原型（Text Prototypes）：通过预训练的词嵌入（word embeddings）提取一些典型的“语言模式”
- 例如：“短暂上升”（short up）、“平稳下降”（steady down）
- 跨注意力机制（Cross-Attention）：将这些文本原型与时间序列片段进行匹配，通过注意力机制帮助模型理解时间动态。

技术细节：Text Prototypes 和跨注意力机制

- 从 E 中提取出一小部分文本原型集 $E' \in \mathbb{R}^{V' \times D}$ ，其中 $V' \ll V$ 。这些原型捕捉了典型的语言模式（例如，“*short up*”和“*steady down*”），这些模式有助于描述时间序列的局部趋势。
- 使用跨注意力机制对时间序列数据进行重编程。对于每个注意力头 $k \in \{1, \dots, K\}$ ，定义查询矩阵 $Q_k^{(i)}$ ，键矩阵 $K_k^{(i)}$ 和值矩阵 $V_k^{(i)}$ 如下：

$$Q_k^{(i)} = \hat{X}_P^{(i)} W_k^Q, \quad K_k^{(i)} = E' W_k^K, \quad V_k^{(i)} = E' W_k^V$$

其中：

- $Q_k^{(i)}$ 来自时间序列片段 $\hat{X}_P^{(i)}$ （查询）；
- $K_k^{(i)}$ 和 $V_k^{(i)}$ 来自文本原型（键/值）；
- W_k^Q 、 W_k^K 和 W_k^V 是学习到的参数矩阵。

技术细节：注意力机制

利用标准的注意力机制计算 $Z_k^{(i)}$:

$$Z_k^{(i)} = \text{Attention} \left(Q_k^{(i)}, K_k^{(i)}, V_k^{(i)} \right) = \text{Softmax} \left(\frac{Q_k^{(i)} K_k^{(i)\top}}{\sqrt{d_k}} \right) V_k^{(i)}$$

这里 d_k 是注意力头的维度。通过这种方式，时间序列片段将与相应的语言原型建立联系。

聚合输出： 最后，将每个注意力头的输出 $Z_k^{(i)}$ 聚合得到：

$$Z^{(i)} = [Z_1^{(i)}, \dots, Z_K^{(i)}] \in \mathbb{R}^{P \times d_m}$$

接着，使用线性变换将其投影到与 LLM 隐藏层维度对齐的空间：

$$O^{(i)} \in \mathbb{R}^{P \times D}$$

最终，时间序列片段被映射到 LLM 可以理解的语义空间，实现了时间序列数据到语言模型的转换。

时间序列重编程：让大语言模型理解时间序列数据

- 核心步骤：
 - 输入时间序列：将时间序列数据作为输入（例如电力消耗、股票价格等）
 - 映射到文本原型：通过重编程将时间序列片段转化为类似语言的表示
 - 利用LLM进行预测：模型根据这些转换后的输入进行长短期预测
- 优势：
 - 无需重新训练：大模型的知识无需改变，只需微调外部重编程器
 - 快速适应：可以应用于各种行业和领域（如金融、能源、医疗等）

性能比较

Model	Avg. Win Rate (%)	Skill Score (%)	Median runtime (s)	Leakage (%)	#Failures
Chronos-2	90.7	47.3	3.6	0	0
TiRex	80.8	42.6	1.4	1	0
TimesFM-2.5	75.9	42.3	16.9	8	0
Toto-1.0	66.6	40.7	90.7	8	0
COSMIC	65.6	39.0	34.4	0	0
Moirai-2.0	61.1	39.3	2.5	28	0
Chronos-Bolt	60.3	38.9	1.0	0	0
TabPFN-TS	59.3	39.6	305.5	0	2
Sundial	41.0	33.4	35.6	1	0
Stat. Ensemble	40.4	20.2	690.6	0	11
AutoARIMA	35.2	20.6	186.8	0	10
AutoETS	29.1	-26.8	17.0	0	3
AutoTheta	21.8	5.5	9.3	0	0
SeasonalNaive	14.5	0.0	2.3	0	0
Naive	7.8	-45.4	2.2	0	0

未来发展方向

- 跨模态融合与深度迁移学习：
 - LLM与其他数据类型（如图像、音频、传感器数据）结合，提升对复杂时序模式的识别能力
 - 实现更强的跨行业知识迁移，应用于更广泛的实际场景（如气候、医疗、供应链等）
- 模型自适应与动态优化：
 - 通过增强学习、元学习等技术，使LLM能够自动适应新的时间序列任务，实现动态调优
 - 自动识别并调整不同时间尺度和周期性的变化，提高预测的稳定性和精度
- 增强可解释性与透明度：
 - 开发可解释的LLM时间序列模型，使业务决策者能够理解模型的预测逻辑
 - 提供更加直观和易于理解的预测结果，增强模型的实际应用价值
- 实时预测与自动化决策：
 - 实现实时流数据的预测和自动化决策支持，提升大规模企业在快速变化环境中的反应能力
 - 结合强化学习等方法，优化决策过程，推动智能决策的广泛应用