学号	202130490229
姓名	张勃文
班级	软件工程卓越班

Autoformer: 基于自相关机制的分解 Transformer 用于长期序列预测

摘 要

扩展预测时间是实际应用中的关键需求，例如极端天气的早期预警和长期能源消耗规划。本文研究了时间序列的长期预测问题。先前基于 Transformer 的模型采用了各种自注意力机制来发现长距离依赖。然而，长期未来的复杂时间模式使得模型难以找到可靠的依赖关系。此外，为了处理长序列的效率问题，Transformers 必须采用稀疏版本的点对点自注意力，这导致了信息利用的瓶颈。超越 Transformers，我们设计了一种基于自相关机制的新型分解架构 *Autoformer*。我们打破了传统的序列分解预处理方式，并将其革新为深度模型的基本内部模块。这个设计使 Autoformer 具备了对复杂时间序列进行逐步分解的能力。此外，受随机过程理论的启发，我们基于序列的周期性设计了自相关机制，该机制在子序列级别进行依赖关系发现和表示聚合。自相关在效率和准确性方面均优于自注意力。在长期预测中，Autoformer 在六个基准上实现了最先进的准确性，相对提升 38%，涵盖了五个实际应用：能源、交通、经济、天气和疾病。代码可在此仓库获取：`https://github.com/thuml/Autoformer`。本论文复现论文为 Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting [NeurIPS2021]。

关键词：时间序列预测，Transformer，深度分解模型，自相关

目录

一、引言

时间序列预测已广泛应用于能源消耗、交通和经济规划、天气和疾病传播预测。在这些实际应用中，一个紧迫的需求是将预测时间延长至遥远的未来，这对于长期规划和早期预警非常有意义。因此，本文研究了时间序列的长期预测问题，其特点是预测的时间序列长度较长。近期的深度预测模型[15;17;18;21;25;26;32;37] 取得了巨大的进展，特别是基于 Transformer 的模型。得益于自注意力机制，Transformer 在为顺序数据建模长期依赖关系方面具有很大优势，使得更强大的大模型成为可能[7;11]。

然而，在长期设置下预测任务极具挑战性。首先，从长期时间序列中直接发现时间依赖关系是不可靠的，因为这些依赖关系可能被复杂的时间模式所掩盖。其次，具有自注意力机制的传统 Transformer 在长期预测中计算成本过高，因为其序列长度的复杂度是平方级的。先前基于 Transformer 的预测模型[15;18;37] 主要集中于将自注意力改进为稀疏版本。虽然性能显著提升，但这些模型仍然使用点对点表示聚合。因此，在提高效率的过程中，由于稀疏点对点连接，它们会牺牲信息利用率，导致时间序列长期预测的瓶颈。

为了推理复杂的时间模式，我们尝试采用分解的思想，这是一种时间序列分析中的标准方法[1;24]。它可以用于处理复杂的时间序列并提取更可预测的成分。然而，在预测背景下，它只能用作过去序列的预处理，因为未来是未知的[14]。这种常见的用法限制了分解的能力，并忽视了分解成分之间潜在的未来互动。因此，我们尝试超越分解的预处理使用，并提出一种通用架构，使深度预测模型具有渐进分解的内在能力。此外，分解可以解开复杂的时间模式并突出时间序列的内在特性[14]。受益于此，我们尝试利用序列的周期性来革新自注意力中的点对点连接。我们观察到，在各周期中处于相同相位位置的子序列通常呈现出相似的时间过程。因此，我们尝试基于序列周期性导出的过程相似性来构建序列级连接。

基于上述动机，我们提出了一种原始的 **Autoformer**，用于替代 Transformer 进行长期时间序列预测。Autoformer 仍然遵循残差和编码器-解码器结构，但将 Transformer 革新为分解预测架构。通过嵌入我们提出的分解块作为内部操作，Autoformer 可以逐步从预测的隐藏变量中分离出长期趋势信息。这个设计使我们的模型能够在预测过程中交替分解和细化中间结果。受随机过程理论[8;22] 启发，Autoformer 引入了一种**自相关**机制来替代自注意力机制，该机制基于序列的周期性发现子序列的相似性，并从底层周期中聚合相似的子序列。这种序列级机制实现了长度为 L 的序列的 $O(L \log L)$ 复杂度，并通过将点对点表示聚合扩展到子序列级别，突破了信息利用瓶颈。Autoformer 在六个基准上实现了最先进的准确性。总结起来，我们的主要贡献有：

- 面对复杂的长期未来时间模式，我们提出了 *Autoformer* 作为一种分解架构，并设计了内部分解块，赋予深度预测模型渐进分解的内在能力。

- 我们提出了一种 *Auto-Correlation* 机制，通过在序列级别进行依赖关系发现和信息聚合。我们的机制超越了以往的自注意力家族，可以同时提升计算效率和信息利用率。

- 在长期设置下，Autoformer 在六个基准上相对提升了 38%，涵盖了能源、交通、经济、天气和疾病等五个实际应用领域。

二、相关工作

2.1 时间序列预测模型

由于时间序列预测的重要性，已经开发了各种模型。

许多时间序列预测方法从经典工具开始[9;29]。ARIMA[5;6] 通过差分将非平稳过程转换为平稳过程来解决预测问题。还引入了滤波方法用于序列预测[10;16]。

此外，循环神经网络（RNN）模型用于为时间序列建模时间依赖性[20;23;33;36]。DeepAR[25] 结合自回归方法和 RNN 来建模未来序列的概率分布。LSTNet[17] 引入了卷积神经网络（CNN）和循环跳跃连接，以捕捉短期和长期的时间模式。基于注意力机制的 RNN[27;28;35] 引入了时间注意力机制以探索预测的长程依赖性。

此外，许多基于时间卷积网络（TCN）[3;4;26;31] 的工作尝试使用因果卷积建模时间因果关系。

这些深度预测模型主要通过循环连接、时间注意力或因果卷积来聚焦于时间关系建模。

最近，基于自注意力机制的 Transformer[32;34] 在序列数据中显示出强大的能力，例如自然语言处理[7;11]、音频处理[13] 甚至计算机视觉[12;19]。然而，将自注意力应用于长期时间序列预测在计算上是不可行的，因为序列长度 L 在内存和时间上的二次复杂度。LogTrans[18] 向 Transformer 引入局部卷积并提出 LogSparse 注意力，通过选择按指数增长间隔的时间步来将复杂度降低到 $O(L(\log L)^2)$。Reformer[15] 提出了局部敏感哈希（LSH）注意力，将复杂度降低到 $O(L \log L)$。Informer[37] 使用基于 KL 散度的 ProbSparse 注意力扩展了 Transformer，也达到了 $O(L \log L)$ 复杂度。需要注意的是，这些方法基于原始 Transformer 并尝试将自注意力机制改进为稀疏版本，仍然遵循逐点依赖和聚合。在本文中，我们提出的自动相关机制基于时间序列的内在周期性，可以提供序列级的连接。

2.2 时间序列分解

作为时间序列分析的标准方法，时间序列分解[1;24] 将时间序列分解为几个分量，每个分量代表一个更易预测的底层模式类别。它主要用于探索历史变化。对于预测任务，分解总是作为预测未来序列之前的历史序列的预处理[2;14]，例如带有趋势-季节性分解的 Prophet[30] 和带有基扩展的 N-BEATS[21] 以及带有矩阵分解的 DeepGLO[26]。然而，这种预处理受限于历史序列的简单分解效果，忽略了长期未来中底层模式之间的层次交互。本文从一个新的渐进维度引入分解思想。我们的 Autoformer 将分解作为深度模型的内部块，可以在整个预测过程中逐步分解隐藏的序列，包括过去的序列和预测的中间结果。

三、模型方法

3.1 问题定义

定义 1.（时间序列分类 (TSC)）。给定一个时间序列实例 $X = \{X_1, \ldots, X_n\} \in \mathbb{R}^{n \times m}$，其中 m 表示变量的数量，n 表示时间步的数量，TSC 的目标是从 c 个类别中准确预测类别标签 $y \in \{1, \ldots, c\}$。当 m 等于 1 时，TSC 是单变量的，否则它是多变量的。

定义 2.（时间序列 (TS) 训练集）。一个训练集 $D = \{(X^{(1)}, y^{(1)}), \ldots, (X^{(N)}, y^{(N)})\}$ 由 N 个时间序列实例组成，其中 $X^{(i)} \in \mathbb{R}^{n \times m}$ 可以是单变量或多变量的时间序列实例，其对应的标签为 $y^{(i)} \in \{1, \ldots, c\}$。注意，在我们的情况下，所有实例在一个 TS 数据集中具有相同的时间步数。在不失一般性的情况下，给定一个训练集，我们旨在训练一个具有自适应和各种接收域（RF）的 CNN 分类器，且资源成本较低（例如内存和计算）以完成 TSC 任务。

3.2 具有自适应接收域的动态稀疏 CNN 层

一种覆盖各种 RF 的直接策略是在每个 CNN 层中应用多尺寸内核，但这有几个限制。首先，不同 TS 数据集的 TS 实例在大多数情况下长度和周期不同，这使得即使有先验知识也很难为所有数据集设置固定的内核配置。其次，获得大接收域通常需要大内核或堆叠更多层，这会引入更多参数，从而增加存储和计算成本。

为了解决这些挑战，提出的动态稀疏 CNN 层具有大但稀疏的内核，这些内核是可学习的，以捕获自适应 RF。具体来说，给定第 l 层中的输入特征图 $x^l \in \mathbb{R}^{c_{l-1} \times h \times w}$（对于单变量 TSC，$h$ 等于 1，且 c_{l-1} 是输入通道的数量），以及内核权重 $\Theta^l \in \mathbb{R}^{c_{l-1} \times c_l \times 1 \times k}$（$k$ 是内核大小，c_l 是输出通道的数量），我们提出的动态稀疏 CNN 层的步幅为 1 的卷积操作和填充方式如下公式所示：

$$O_j = \sum_{0 < i \leqslant c_{l-1}, i \in \mathbb{Z}} (\mathbb{I}^l(\Theta^l)_{i,j} \odot \Theta^l_{i,j}) \cdot x^l_i, \tag{1}$$

其中 $O_j \in \mathbb{R}^{h \times w}$ 表示第 j 个输出通道的输出特征表示，$\mathbb{Z}$ 表示整数集合，$\mathbb{I}^l(\cdot) : \mathbb{R}^{c_{l-1} \times c_l \times 1 \times k} \to \{0, 1\}^{c_{l-1} \times c_l \times 1 \times k}$ 是一个指示函数，$\mathbb{I}^l(\Theta^l)_{i,j}$ 表示在第 (i, j) 通道中激活的权重 $\Theta^l_{i,j} \in \mathbb{R}^{1 \times k}$ 是内核，$\odot$ 表示逐元素乘积，$\cdot$ 表示卷积算子。指示函数 $\mathbb{I}^l(\cdot)$ 在提出的 DSN 训练过程中被学习，并满足 $\|\mathbb{I}^l(\cdot)\|_0 \leqslant (1 - S)N_l$，其中 $0 \leqslant S < 1$ 是稀疏率，$\|\cdot\|_0$ 表示 L_0 范数，$N_l = c_{l-1} \times c_l \times 1 \times k$。当 $S > 0$ 时，内核是稀疏的，我们可以在动态稀疏 CNN 层中使用大的 k 以获得大 RF，且计算成本较低。

注释（动态稀疏 CNN 层中的有效邻域接收域（eNRF）大小） 接收域被定义为 CNN 模型在输入中关注的区域。我们将邻域接收域（NRF）定义为每个特征在连续层中考虑的区域。具体来说，在标准 CNN 层中，NRF 大小等于内核大小（考虑扩张率等于 1 的情况）。不同的是，当内核中的第一个或最后一个权重未激活时，例如，存在 $i \in \{1, \ldots, c_{l-1}\}$ 和 $j \in \{1, \ldots, c_l\}$，使得 $\mathbb{I}^l(\Theta^l)_{i,j,1,1} = 0$ 或 $\mathbb{I}^l(\Theta^l)_{i,j,1,k} = 0$，则提出的动态稀疏 CNN 层的 NRF 大小小于内核大小。如图 2 所示，我们定义了第 l 层 CNN 中内核 $\Theta^l_{i,j}$ 的 eNRF 大

小 $f_{i,j}^{l}$ 为在第 l 层中第一个激活权重和最后一个激活权重之间的距离：

$$f_{i,j}^{l} := \begin{cases} \max(\mathrm{Ind}_{i,j}^{l}) - \min(\mathrm{Ind}_{i,j}^{l}) + 1, & \text{if } \mathrm{Ind}_{i,j}^{l} \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

其中 $\mathrm{Ind}_{i,j}^{l}$ 表示内核 $\Theta_{i,j}^{l}$ 中对应于非零权重的索引集合。显然，我们有 $0 \leqslant f_{i,j}^{l} \leqslant k$。

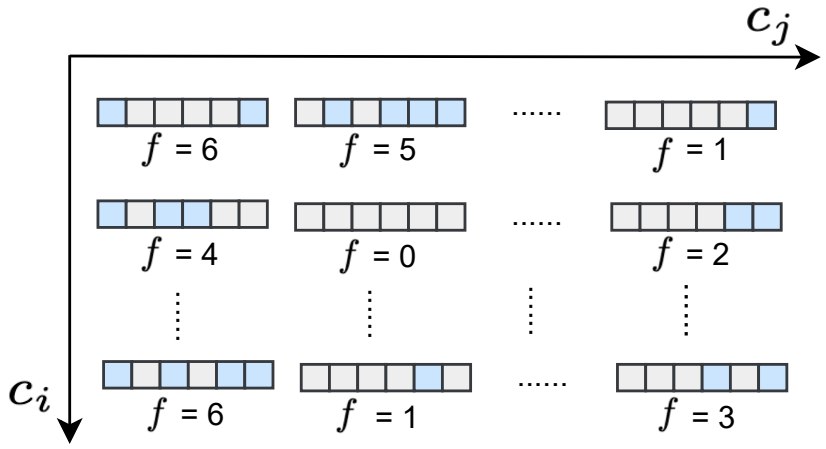

图 1 展示了 eNRF 的大小。蓝色的权重是已激活的。

第 l 层动态稀疏 CNN 层的 eNRF 大小集合记为 $F^{(l)}$，其满足 $0 \leqslant \min(F^{(l)})$ 且 $\max(F^{(l)}) \leqslant k$。以图 2 中的第 l 层为简单例子，则 $F^{(l)} = \{0, 1, 2, 3, 4, 5, 6\}$。每个动态稀疏 CNN 层可能覆盖从 1 到 k 的各种 eNRF 大小。当需要全局信息时，$\mathbb{I}^{l}(\cdot)$ 可以激活更加分散的权重以获得更大的 eNRF，并有选择地利用输入特征。为了捕捉局部上下文，预期 eNRF 较小，因此激活的权重往往集中。以 $k = 5$ 为例，$\mathbb{I}^{l}(\Theta^{l})_{i,j}$ 对于全局上下文可能为 $[1, 0, 1, 0, 1]$，而对于局部上下文可能为 $[0, 0, 1, 1, 0]$。通过这种方式，eNRF 可以自适应地调整。

3.3 DSN 结构

提出的 DSN 模型由三个稀疏 CNN 模块组成，每个模块由一个动态稀疏 CNN 层和一个 1×1 CNN 层组成。在堆叠的稀疏 CNN 模块之后，还有一个附加的动态稀疏 CNN 层、两个自适应平均池化层和一个 1×1 卷积层作为 DSN 模型中的分类器。整体结构如图 3 所示。

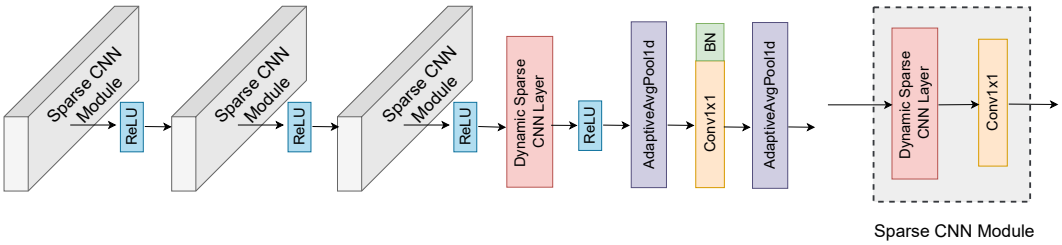

图 2 提出的 DSN 模型（左图），其中包含若干稀疏 CNN 模块（右图），后接一个动态稀疏 CNN 层、两个自适应平均池化层和一个 1×1 卷积层。

第 l 个稀疏 CNN 模块中的 eNRF 大小集合 $S^{(l)}$ 等于动态稀疏 CNN 层中的 eNRF 大小集合，因为 1×1 卷积的 eNRF 大小始终等于 1。$S^{(l)}$ 满足 $0 \leqslant \min(S^{(l)})$ 且 $\max(S^{(l)}) \leqslant k$。那么，三个连续堆叠的稀疏 CNN 模块的 eNRF 大小集合可以用 RF 描述为：

$$RF = \max(0, s^{(1)} + s^{(2)} + s^{(3)} - 2) \quad | \quad s^{(i)} \in S^{(i)}, i \in \{1, 2, 3\}. \tag{3}$$

根据公式 (3)，我们可以看到，堆叠多个稀疏 CNN 模块可以线性地增加 RF 的大小，其中第 l 个动态稀疏 CNN 层增加的大小为 $S^{(l)}$。为简化起见，我们的研究中每个动态稀疏 CNN 层的内核大小一致设置为 k。然后，RF 满足 $\max(RF) \leqslant 3k-2$ 和 $0 \leqslant \min(RF)$。因此，稀疏 CNN 模块中的大 k 可以增加堆叠的稀疏 CNN 模块的 eNRF 大小范围。

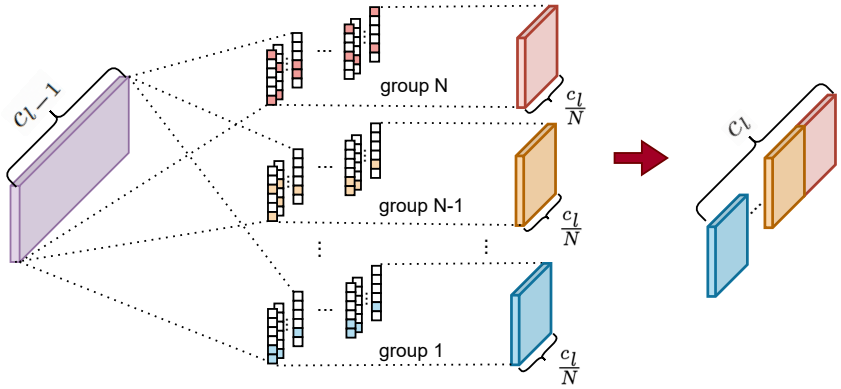

图 3 动态稀疏 CNN 层的示意图。在动态稀疏 CNN 层中，卷积核被分成 N 组，每组通过动态稀疏训练在特定约束区域内学习到稀疏连接。有颜色的连接表示非零值。

3.4 DSN 模型的动态稀疏训练

在本节中，我们提出了训练策略，以发现需要激活的权重，从而确保良好的 RF 性能。也就是说，我们需要研究如何在提出的 DSN 训练过程中更新指示函数 $\mathbb{I}^l(\cdot)$。

我们从头开始稀疏地训练提出的 DSN 模型，以保持内核的稀疏性，遵循 DST 方法的主要思想 [40]。在设计上，在训练阶段，激活的权重总数不能超过 $N_l(1-S)$。然而，我们观察到，微小的 eNRF 很难被 DST 方法以逐层探索的方式捕捉到，也就是说，激活的权重是逐层发现的，特别是当稀疏率 S 较小时（更多分析见第 4.5 节）。受到这一观察的启发，每个动态稀疏 CNN 层中的内核被分为不同的组，其对应的探索区域大小不

同，如图 3 所示。与 DST 方法相反，在 DSN 中，激活权重的探索在不同的内核组中单独进行，这是一种更精细的策略。

具体来说，第 l 层中的内核权重 $\Theta^l \in \mathbb{R}^{c_{l-1} \times c_l \times 1 \times k}$ 被沿输出通道方向分成 N 组，即 $\Theta_1^l, \ldots, \Theta_N^l \in \mathbb{R}^{c_{l-1} \times \frac{c_l}{N} \times 1 \times k}$ 并具有对应的探索区域 $R_1^l, \ldots, R_N^l$。第 l 层第 i 组的探索区域 R_i^l 定义为每个内核中前 $\frac{i \times k}{N}$ 个位置。例如，取 $k = 6$ 和 $N = 3$，第一组中的激活权重仅在 Θ_1^l 中每个内核的前两个位置，而最后一组中的激活权重在整个内核中（如图 4 所示）。通过这种方式，可以覆盖不同的 eNRF，并减少探索空间（更多细节见附录 A.1），以提高探索效率。给定权重探索区域 $R_1^l, \ldots, R_N^l$ 和稀疏比率 S，我们按照算法 1 所示训练提

Algorithm 1 DSN 模型的动态稀疏训练

Require: 数据集 D，网络 f_Θ，稀疏率 S
Require: 第 l 层的探索区域：$\{R_l^1, ..., R_l^N\}$
Require: 探索计划：$T, \Delta T, \alpha, f_{decay}$

$\theta_l^i \leftarrow$ 使用 S 和 R_l^i 初始化 Θ_l^i 中的激活权重
for $t = 1$ to T **do**
  从 D 中采样一个小批量 B_t
  $L_t = \sum_{i \in B_t} L((f_\theta(x_i), y_i))$
  if $(t \mod \Delta T) == 0$ **then**
    for 每一层 $l = 1$ 到 L **do**
      for 每一组 $i = 1$ 到 N **do**
        $u = n(R_l^i) f_{decay}(t; \alpha, T)(1 - S)$
        $I_{prune} = \text{ArgTopK}\left(-\left|\theta_i^l\right|, u\right)$
        $I_{grow} = \text{RandomK}\left(R_i^l \backslash \theta_i^l, u\right)$
        $I_l(\cdot) \leftarrow$ 使用 I_{prune} 和 I_{grow} 更新 $I_l(\cdot)$
        $\theta_l^i \leftarrow$ 更新激活权重 $I_l(\Theta_l^i) \odot \Theta_l^i$
      end for
    end for
  else
    $\theta = \theta - \alpha \nabla_\theta L_t$
  end if
end for

出的 DSN 模型。激活的权重在探索区域内被探索并每 Δt 次迭代更新一次。更新权重的比例随时间按照函数 $f_{\text{decay}}(t; \alpha, T)$ 逐渐减少，该函数遵循余弦退火 [11]：

$$f_{\text{decay}}(t; \alpha, T) = \frac{\alpha}{2}\left(1 + \cos\frac{t\pi}{T}\right), \tag{4}$$

其中 α 是初始更新的激活权重比例，t 是当前训练迭代次数，T 是训练迭代总次数。因此，在第 t 次迭代期间，第 l 层第 i 组的更新激活权重的数量为 $n(R_i^l) f_{\text{decay}}(t; \alpha, T)(1 - S)$，其中 $n(R_i^l)$ 是在区域 R_i^l 中可以被探索的权重数量。在激活权重更新过程中，我们首先通

过 ArgTopK $\left(-\left|\theta_i^l\right|, u\right)$ 剪枝激活权重，然后通过 RandomK $\left(R_i^l \backslash \theta_i^l, u\right)$ 随机增长新的激活权重，其中 ArgTopK(v,u) 返回向量 v 中取值最大的 u 个元素的索引，RandomK(v,u) 返回向量 v 中随机的 u 个元素的索引，$R_i^l \setminus \theta_i^l$ 表示 R_i^l 中除了 θ_i^l 之外的权重。

激活权重的探索是直接的。首先，剪枝小幅度的权重是直观的，因为小幅度权重的贡献微不足道甚至可以忽略不计。考虑到剪枝的可恢复性，我们随机重新增长与剪枝数量相同的新权重，以实现更好的激活权重探索。通过这种方式，与训练前后剪枝的方法相比，权重的探索是动态且具有可塑性的（关于剪枝和动态稀疏训练的更多解释可见[22, 39, 32]）。

四、实验

在本节中，我们评估了所提出的 DSN 模型在单变量和多变量 TSC 数据上的资源成本和准确性，并将其与最近的基线方法进行了比较。

4.1 数据集和基线方法

每种数据集的详细信息如下：

- **来自 UCR 85 档案的单变量 TS 数据集**[?]：该档案包含 85 个单变量 TS 数据集，这些数据集来自各个领域（例如健康监测和遥感），具有不同的特征和复杂性。训练集中实例数量从 16 到 8926 不等，而时间步长分辨率从 24 到 2709 不等。
- **来自 UCI 的三个多变量 TS 数据集**[?]：EEG2 数据集包含 1200 个实例，分为 2 类，有 64 个变量。人类活动识别（HAR）数据集包含 10,299 个实例，分为 6 类，有 9 个变量。日常运动数据集包含 9,120 个实例，分为 19 类，有 45 个变量。这些数据集的处理方式与[?]相同。

单变量 TSC 基线方法：报告了以下三种基线方法的性能：OS-CNN[?]，Inception-Time[?]，以及 ResNet[?]，因为它们在单变量 TSC 中被广泛使用。**多变量 TSC 基线方法：**与[?]相似，选择了以下三种多变量基线方法：OS-CNN，TapNet[?]，MLSTM-FCN[?]。为了避免不公平的比较，我们排除了集成方法。然而，基于我们的最佳知识，我们的性能分析涵盖了大多数最先进的方法。

4.2 实验设置和实施细节

为了优化，我们使用初始学习率为 3×10^{-4} 的 Adam 优化器，并以余弦衰减至 10^{-4}。我们的模型训练了 1000 轮，mini-batch 大小为 16。如[? ?]中所述，使用对应于最小训练损失的最佳模型来评估测试集的性能。受[?]的启发，每个动态稀疏 CNN 层中的核大小 k 设置为 39。算法 **??** 中的核组数量 N 设置为 3，以覆盖小、中、大 eNRF。每个设置重复五次，并报告平均结果。

五、结论

本文提出了一种动态稀疏网络（DSN），用于覆盖时间序列分类（TSC）中多样化的有效邻域感受野（eNRF），无需繁琐的超参数调优。提出的 DSN 在内存和计算成本方面可实现约 2 倍的减少，同时在单变量和多变量 TSC 数据集上的准确性方面表现优异。此外，DSN 以更细粒度的策略探索激活权重，可以轻松与现有的 DST 技术相结合，显示出结合先进技术时进一步改进的潜力。DSN 提供了一种在资源感知和各种 eNRF 覆盖之间搭桥的可行解决方案，希望能够启发其他领域的研究人员。将 DSN 扩展到其他研究领域（如时间序列预测和计算机视觉）是可行的。我们的 DSN 模型尚未在真实的边缘设备上执行，尚未研究在边缘设备上关于准确性和内存占用的性能。在未来，我们希望将我们的工作扩展到其他领域，并在边缘设备上实现去中心化训练。natbib

参考文献

[1] O. Anderson and M. Kendall. Time-series. 2nd edn. *J. R. Stat. Soc. (Series D)*, 1976.

[2] Reza Asadi and Amelia C Regan. A spatio-temporal decomposition based deep neural network for time series forecasting. *Appl. Soft Comput.*, 2020.

[3] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[4] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[5] G. E. P. Box and Gwilym M. Jenkins. Time series analysis, forecasting and control. 1970.

[6] George EP Box and Gwilym M Jenkins. Some recent advances in forecasting and control. *J. R. Stat. Soc. (Series-C)*, 1968.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

[8] Chris Chatfield. *The analysis of time series: an introduction*. 1981.

[9] Renyi Chen and Molei Tao. Data-driven prediction of general hamiltonian dynamics via learning exactly-symplectic maps. *ICML*, 2021.

[10] Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. Normalizing kalman filters for multivariate time series analysis. In *NeurIPS*, 2020.

[11] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

[13] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. In *ICLR*, 2019. URL `https://openreview.net/forum?id=rJe4ShAcF7`.

[14] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. 2018.

[15] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020. URL `https://openreview.net/forum?id=rkgNKkHtvB`.

[16] Richard Kurle, Syama Sundar Rangapuram, Emmanuel de Bézenac, Stephan Günnemann, and Jan Gasthaus. Deep rao-blackwellised particle filters for time series forecasting. In *NeurIPS*, 2020.

[17] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, 2018.

[18] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *NeurIPS*, 2019. URL `https://proceedings.neurips.cc/paper/2019/file/6775a0635c302542da2c32aa19d86be0-Paper.pdf`.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[20] Danielle C Maddix, Yuyang Wang, and Alex Smola. Deep factors with gaussian processes for forecasting. *arXiv preprint arXiv:1812.00098*, 2018.

[21] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *ICLR*, 2019.

[22] Athanasios Papoulis and H Saunders. Probability, random variables and stochastic processes. 1989.

[23] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *NeurIPS*, 2018.

[24] Cleveland Robert, C William, and Terpenning Irma. STL: A seasonal-trend decomposition procedure based on loess. *J. Off. Stat*, 1990.

[25] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *Int. J. Forecast.*, 2020.

[26] Rajat Sen, Hsiang-Fu Yu, and Inderjit S. Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *NeurIPS*, 2019.

[27] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. Temporal pattern attention for multivariate time series forecasting. *Mach. Learn.*, 2019.

[28] Huan Song, Deepta Rajan, Jayaraman Thiagarajan, and Andreas Spanias. Attend and diagnose: Clinical time series analysis using attention models. In *AAAI*, 2018.

[29] Antti Sorjamaa, Jin Hao, Nima Reyhani, Yongnan Ji, and Amaury Lendasse. Methodology for long-term prediction of time series. *Neurocomputing*, 2007.

[30] Sean J Taylor and Benjamin Letham. Forecasting at scale. *Am. Stat.*, 2018.

[31] Aäron van den Oord, S. Dieleman, H. Zen, K. Simonyan, Oriol Vinyals, A. Graves, Nal Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *SSW*, 2016.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[33] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *NeurIPS*, 2017.

[34] Sifan Wu, Xi Xiao, Qianggang Ding, Peilin Zhao, Ying Wei, and Junzhou Huang. Adversarial sparse transformer for time series forecasting. In *NeurIPS*, 2020.

[35] Q. Yao, D. Song, H. Chen, C. Wei, and G. W. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *IJCAI*, 2017.

[36] Rose Yu, Stephan Zheng, Anima Anandkumar, and Yisong Yue. Long-term forecasting using tensor-train rnns. *arXiv preprint arXiv:1711.00073*, 2017.

[37] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

附 录