全面扩展 A　绿色数据中心可持续发展的自动化 s

Shiyu Wang、Yinbo Sun、XiaomingShi、Zhu Shiyi、Lin-Tao Ma、James Zhang、YangFei Zheng和Liu Jian蚂蚁集团

{weiming.wsy, yinbo.syb, peter.sxm, zhushiyi.zsy, lintao.mlt, james.z, yangfei.zyf, basti.lj}@antgroup.com

arXiv:2305.00706v2 [cs.DC] 1 Mar 2025

摘要

云计算的迅速发展导致数据中心的碳排放量惊人地增加，目前已占全球温室气体排放量的 >3%，因此亟需采取行动，应对其对全球气候日益加剧的影响。这项工作的一个重要重点是提高资源利用，以减少电力消耗。我们提出的 Full Scaling Automation（FSA）机制能够动态调整资源，以适应大规模云计算集群中不断变化的工作负载，从而使数据中心的集群维持预期的 CPU 利用率目标，进而提高能源效率。FSA 利用深度表征学习的能力，准确预测每项服务的未来工作负载，并自动稳定相应的目标 CPU 使用率；而 Autopilot 或 FIRM 等以往的自动伸缩方法则需要借助统计模型和专家知识来调整计算资源。与现有研究相比，我们的方法在真实世界数据集上取得了显著的性能提升。我们还在工业数据中心的大规模云计算集群上部署了 FSA。根据中国环境联合认证中心（CEC）的认证，在 2022 年双十一购物节期间，该方案减少了 947 吨二氧化碳排放，相当于节省 1538,000 千瓦时电力；这标志着我们公司朝着 2030年实现碳中和的战略目标迈出了关键一步。

1 引言

随着对数据和云计算服务的需求持续激增，数据中心行业正以惊人的速度扩张，成为现代世界经济、商业和社会生活的支柱。然而，数据中心也是全球耗电量最大的行业之一，其不断增长的碳排放更是加剧了全球变暖。目前，来自数据中心的排放量占全球温室气体排放总量的3.7%，超过了商业航班以及其他支撑经济运行的活动所产生的排放量。这引起了许多全球组织的关注，其中包括聚焦数据中心可持续发展的联合国气候大会。作为全球迈向碳中和（或净零碳排放）努力的一部分，提升数据中心计算资源的利用效率，正日益成为降低能耗的热门研究领域。

数据中心大规模云计算集群的碳排放主要来自设备负载的能耗。多项研究表明，长期处于低 CPU 利用率状态的服务器是造成能源浪费的主要原因之一。通过减少分配给低负载服务的资源，并将节省下来的服务器重新分配给高负载服务，可以缓解这一问题[Krieger等人, 2017；Abdullah等人， 2020]。因此，高效的资源分配机制在这项工作中发挥着重要作用。此外，一些系统支持灵活关闭不需要的服务器，并通过适当的调度和工程机制进一步降低能耗。因此，大多数云计算服务提供商都尝试利用自动伸缩系统（autoscaling），将所配置系统的 CPU 利用率稳定在目标水平，以确保服务满足严格的服务等级目标（SLOs），同时防止 CPU 处于低利用率状态[Qiu 等人， 2020]。

具体而言，自动伸缩会根据某些性能指标（例如资源利用率），通过水平扩展（即改变所分配的虚拟容器（Pod）数量）或垂直扩展（即调整 CPU 和内存预留）弹性地扩缩资源，以匹配不断变化的工作负载。以往关于云自动伸缩的研究可分为基于规则的方案和基于学习的方案。基于规则的机制根据专家经验设置指标（例如 CPU 利用率和工作负载）的阈值，并监控指标变化，以促进资源调度。基于学习的自动伸缩机制采用统计方法或机器学习对历史模式进行建模，从而生成扩缩容策略。然而，现有研究面临以下挑战：

• **挑战1**：**预测**时间序列（**TS**）**工作负载的复杂周期性和长期时间依赖关系**。大多数自动伸缩研究关注的是伸缩的决策过程，而非服务器工作负载预测。

……，未能解决准确预测未来工作负载这一关键任务（通常是具有多个周期和复杂长时序依赖关系的高频非平稳时间序列），而现有采用经典统计方法（如 ARIMA）或简单神经网络（如循环神经网络）的研究无法很好地处理这一问题。

• **挑战2**：**在决策过程中维持稳定的 CPU 利用率并刻画不确定性**。 大多数方法旨在充分利用资源以节约成本，却忽视了服务稳定性指标，例如鲁棒性和服务等级目标保障，而稳定的 CPU 利用率至关重要。此外，受服务器的各种因素（如温度）影响，决策过程中需要对 CPU 利用率与工作负载之间相关性的不确定性进行建模，以平衡收益与风险。

• **挑战3**：**兼顾绿色数据中心的可持续发展**。 现有研究只优化资源和成本，而不考虑碳排放；然而，降低碳排放是绿色数据中心可持续发展的重要任务之一。

为此，本文提出一种新颖的预测式水平自动伸缩方法 Full Scaling Automation（FSA），以应对上述难题。具体而言，**首先**，我们开发了一种基于表征学习的全新工作负载预测框架，即从历史工作负载时间序列中学习多尺度表征，作为外部记忆，以刻画长期时间依赖关系以及多种复杂周期性（如天、周和月）。随后，我们构建了一种表征增强型深度自回归模型，通过多头自注意力机制整合多尺度表征与近邻观测时间序列。**其次**，我们提出一种任务条件化贝叶斯神经模型，用于学习 CPU 利用率与工作负载之间的关系，从而刻画二者之间的不确定性，同时提供收益与风险的上下界。此外，借助任务条件超网络[Mahabadi 等，2021；He等， 2022], 我们的方法可以轻松适应各种在线场景。**第**三，基于上述工作负载预测和 CPU 利用率估计，我们可以合理地预先分配资源，并将未来工作负载分配至各台机器，以达到特定的 CPU 利用率水平并最大限度地提高能源效率。

我们已在大规模云计算集群上部署 FSA 机制，并借助中国环境联合认证中心（CEC），实现节能与碳减排之间的转换，旨在促进数据中心的可持续发展。据我们所知，我们是业内首家为实现这一目标而采用基于人工智能的资源扩缩容的企业。

我们的贡献可总结如下：

• 一种新颖的工作负载时间序列预测框架，采用增强表征的深度自回归模型，并整合多尺度时间序列表示。

• 一种任务条件贝叶斯神经模型，用于估计 CPU 利用率与工作负载之间的不确定性。

• 一种高效的预测式水平自动扩缩容机制

稳定 CPU 利用率，节省成本和能源，助力可持续数据中心。

2 背景与相关工作

2.1 水平自动伸缩

水平自动伸缩是一种实用的资源管理方法，可在水平方向弹性伸缩资源（即改变所分配的虚拟容器（Pod）数量）[Nguyen 等人， 2020]。现有研究大多侧重于避免服务异常，例如，FIRM [Qiu 等人， 2020] 采用机器学习检测服务性能异常（例如，微服务的响应时间异常过长），并在发生此类异常时分配更多 Pod；Autopilot [Rzadca等人， 2020] 以微服务 CPU 利用率的时间序列作为输入，并采用一种简单的启发式机制来确定目标 CPU 利用率，再据此通过线性函数计算所需的 Pod 数量。通过增加或减少 Pod，尽可能降低微服务发生异常的风险。这些方法存在以下局限：（1）大多数方法采用响应式伸缩，即只有在发生性能异常后才进行自动伸缩。（2）由于缺乏工作负载预测机制，难以对未来资源伸缩做出精准决策。（3）决策时未考虑 CPU 利用率的不确定性，导致资源伸缩存在未知风险。

2.2 工作负载预测

我们的方法的一个关键组成部分是工作负载预测。然而，工作负载通常具有高频、非平稳的特点，并包含各种周期和复杂的长期时间依赖关系，使大多数经典时间序列预测模型难以应对。因此，我们提出使用时间序列表征来记忆长期时间依赖关系，以应对这种复杂性。

时间序列 (TS) 预测。 基于循环神经网络或 Transformer 的深度预测方法，包括 N-BEATS、Informer 和 DeepAR 等，已广泛应用于时间序列预测，其表现优于 ARIMA 和向量自回归等经典模型。N-BEATS 是一种可解释的时间序列预测模型，其深度神经网络架构基于反向和前向残差连接，并采用非常深的全连接层堆栈[Oreshkin等， 2019]。Informer 是一种基于概率稀疏自注意力机制的模型，旨在提升长序列时间序列预测问题中的预测能力；该模型验证了类 Transformer 模型在捕捉长序列时间序列输出与输入之间的单个长程依赖关系方面的潜在价值 [Zhou 等， 2021]。DeepAR 是一种基于循环神经网络的概率自回归预测模型，广泛应用于现实世界的工业场景 [Salinas 等， 2020]。

时间序列（TS）表征。表示学习近期在推动时间序列研究方面取得了巨大成功，它基于对比学习方法刻画长期时间依赖关系和复杂的周期性。TS-TCC 通过应用强弱数据增强为每个样本创建两个视图，然后使用时间对比模块，通过应用跨视图预测任务来学习鲁棒的时间特征 [Eldele 等人， 2021]。TS2Vec

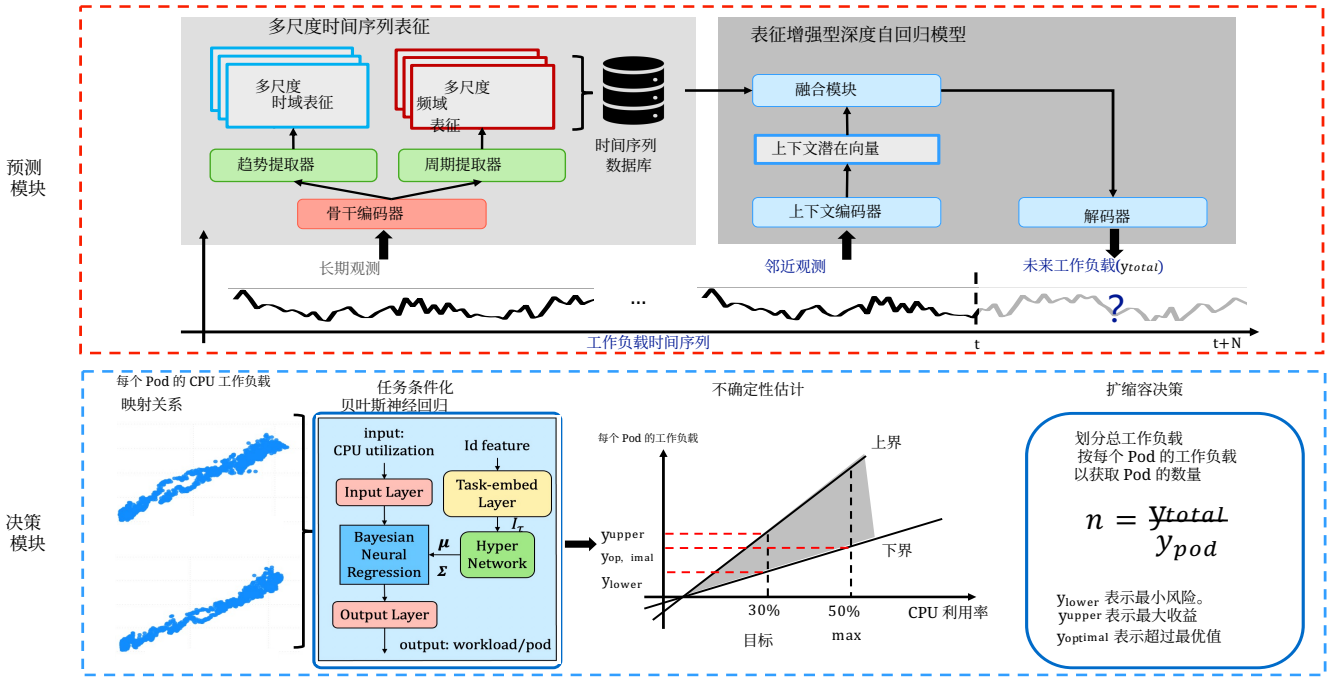


图1：**模型架构**。红色虚线表示工作负载预测模块，包括多尺度时间序列表征和表征增强型深度自回归模型。蓝色虚线突出显示了通过任务条件贝叶斯神经回归实现的扩缩容决策模块。根据对响应时间*(rt)*的服务容忍度，可在上下界之间确定最优值。

最近提出了一种通用框架，通过在增强的上下文视图上采用分层损失进行对比学习，从而学习时间序列表征 [Yue 等,2022]。CoST 提出了一种用于长序列时间序列预测的新型时间序列表征学习框架，该框架采用对比学习方法来学习解耦的季节性-趋势表示 [Woo 等，2022]。

2.3 贝叶斯方法

贝叶斯方法常用于对变量的不确定性建模，并被引入深度学习模型以提升模型的鲁棒性 [Bishop and Nasrabadi,2006]。在本工作中，我们将任务条件化超网络 [Mahabadi 等， 2021] 引入贝叶斯回归，以估计 CPU 利用率与工作负载之间的相关性。

3 拟议方法

本节将详细介绍我们的框架，包括预测模块和决策模块，如图1所示。

3.1 问题定义

设工作负载时间序列的值为 $y_t \in \mathbb{R}^F$，其中 t 表示时间范围 $t \in \{1, 2, ..., T\}$ 上的时间索引， F 表示特征的维度，例如时间特征和 ID特征。

给定应用程序 y_{t-L} 在时间 t的历史工作负载时间序列，我们旨在确定在未来时间范围 N内分配的最优 Pod 数量，使应用程序能够在目标 CPU 利用率附近稳定运行。我们假设一个应用程序的总工作负载通过负载均衡均匀分配到所有 Pod 上，并且我们可以将它们之间的关系表示为

$$y_t^{pod} = y_t^{total}/n_t \quad \Leftrightarrow \quad n_t = y_t^{total}/y_t^{pod}, \qquad (1)$$

其中， y_t^{total} 表示总工作负载， y_t^{pod} 表示每个 Pod 的工作负载， n_t 表示时间 t 时 Pod 的数量。请注意，该假设由数据中心的软件和硬件工程平台保证。

3.2 工作负载预测

考虑到工作负载时间序列具有非平稳性和高频特征（分钟级），且存在各种周期以及复杂的长期时间依赖关系（例如每日、每周、每月甚至季度周期性），而现有工作需要将长期历史时间窗口的回溯预测作为上下文输入。庞大的数据量使得记忆历史数据并学习这些时间依赖关系变得困难。因此，我们提出了一种时间序列表示方法，将具有各种周期性的复杂长期历史时间序列表征为压缩表示向量，并将其存储在时间序列数据库中。随后，我们设计了一种深度自回归融合模型，将长期历史时间序列表示与短期观测（邻近窗口）相结合，以实现准确的预测结果。

多尺度时间序列表示

给定回溯窗口 $y_{t-h} \in \mathbb{R}^F$，我们的目标是学习一个非线性嵌入函数 f_θ，将 $\{y_{t-h} ... y_t\}$ 映射为其表示 $r_t = [r_t^T, r_t^F]$，其中 $r_t \in$

$\mathbb{R}^K$ 表示每个时间戳 t， $r_t^T \in \mathbb{R}^{K_T}$ 表示时域表征，$r_t^F \in \mathbb{R}^{K_F}$ 表示频域表征， $K = K_T + K_F$ 表示表示向量的维度。在编码表征阶段，通过使用不同长度的回溯窗口，我们可以获得不同尺度的表征。

首先，从输入时间序列中随机采样两个重叠子序列，然后分别进行数据增强。接着，使用多层感知机（MLP）作为输入投影层，将原始输入 y_t 映射为高维潜在向量 z_t。我们使用时间戳掩码，在随机选取的时间戳处遮蔽潜在向量，以生成增强的上下文视图。随后，使用 CovnTrans 作为骨干编码器，提取每个时间戳处的上下文嵌入。接下来，分别使用 CausalConv 和快速傅里叶变换（FFT），从上下文嵌入中提取时域趋势和频域周期。最后，我们在时域和频域中进行对比学习。以下各节将详细介绍这些组件。

随机裁剪 是一种常用于对比学习的数据增强方法，用于生成新的上下文视图。我们可以从时间序列 $y \in \mathbb{R}^{T \times F}$ 中随机采样两个满足$0 < a_1 < b_1 < a_2 < b_2 \leq T$的重叠时间片段 $[a_1, a_2]$和 $[b_1, b_2]$。请注意，重叠片段 $[b1, a2]$ 上的上下文表示可确保两个上下文视图保持一致。

时间戳遮蔽 旨在通过随机遮蔽时间序列的时间戳，生成增强的上下文视图。我们可以在输入投影层之后，沿时间轴使用二进制掩码$m \in \{0, 1\}^T$遮蔽潜在向量 $z = \{z_t\}$，其中各元素均独立地从参数为 $p = 0.5$的伯努利分布中采样。

骨干编码器用于提取每个时间戳处的上下文表征。我们使用 1 层因果卷积 Transformer（ConvTrans）作为骨干编码器，其卷积多头自注意力机制能够捕捉长期和短期依赖关系。

具体而言，给定时间序列 $y \in \mathbb{R}^{T \times F}$，ConvTrans 通过扩张因果卷积层将$y$ （作为输入）转换为维度为 l 的表示，如下所示：

$$
\begin{aligned}
Q &= DilatedConv(y) \\
K &= DilatedConv(y) \,, \\
V &= DilatedConv(y)
\end{aligned}
$$

其中， $\boldsymbol{Q} \in \mathbb{R}^{dl \times dh}$、 $\boldsymbol{K} \in \mathbb{R}^{dl \times dh}$和 $\boldsymbol{V} \in \mathbb{R}^{dl \times dh}$ （我们用 dh表示时间步长度）。经过这些变换后，缩放点积注意力通过以下方式计算输出向量序列：

$$
S = \text{Attention}(Q,\, K,\, V) = \text{softmax}\left(QK^T / \sqrt{d_K} \cdot M\right) V \,,
$$

其中，掩码矩阵 M 可通过将其上三角元素设为 $-\infty$ 来屏蔽右侧注意力（或防止未来信息泄漏），归一化因子d_K 是 W_h^K矩阵的维度。最后，将所有输出 S 拼接起来，再次进行线性投影，输入下一层。经过上述一系列操作后，我们使用该骨干网络 f_θ提取每个时间戳处的上下文嵌入（中间表示），记为 $\tilde{r} = f_\theta\,(y)$。

时域对比学习。为提取时间序列的潜在趋势，一种直接的方法是使用一组具有不同卷积核大小的一维因果卷积层（CasualConv），并通过平均池化操作提取表示，具体如下：

$$
\tilde{r}^{(T,i)} = CausalConv(\tilde{r}, 2^i)
$$

$$
r^T = AvgPool(\tilde{r}^{(T,1)}, \tilde{r}^{(T,2)} ..., \tilde{r}^{(T,L)}) \,,
$$

其中， L 是表示 Casu- alConv 数量的超参数， 2^i （$i = 0, ..., L$）是每个 Casual- Conv 的核大小， $\tilde{r}$ 是骨干编码器输出的中间表征，随后对 L 个表征进行平均池化，以获得时域表征 $\tilde{r}^{(T)}$。为了学习具有判别性的时间表征，我们采用时域对比损失，将输入时间序列两个视图中同一时间戳处的表征作为正样本（$r_{i,t}^T, \hat{r}_{i,t}^T$），而将同一时间序列中不同时间戳处的表征作为负样本，定义如下：

$$
\mathcal{L}_{time} = -log \frac{exp(r_{i,t}^T \cdot \hat{r}_{i,t}^T)}{\sum_{t' \in \mathcal{T}} (exp(r_{i,t}^T \cdot \hat{r}_{i,t'}^T) + \mathbb{I}(t \neq t') exp(r_{i,t}^T \cdot r_{i,t'}^T)}
$$

，其中， $\mathcal{T}$是两个子序列重叠部分内的时间戳集合，下标 i 是输入时间序列样本的索引， t是时间戳。

频域对比学习。 鉴于频域中的频谱分析已广泛用于周期检测，我们使用快速傅里叶变换（FFT）将上述中间表征映射到频域，以捕捉不同的周期模式。因此，我们可以构建一个周期提取器，包括FFT和多层感知机，用于从上述上下文嵌入中提取频谱，并将其映射为基于频率的表示 r_t^F。为了学习能够区分不同周期模式的表征，我们采用沿批次中实例计算的、由（i， t）索引的频域对比损失，其形式为

$$
\mathcal{L}_{Freq} = -log \frac{exp(r_{i,t}^F \cdot \hat{r}_{i,t}^F)}{\sum_{j \in \mathcal{D}} (exp(r_{i,t}^F \cdot \hat{r}_{i,t'}^F) + \mathbb{I}(i \neq j) exp(r_{i,t}^F \cdot r_{i,t'}^F)}
$$

其中， $\mathcal{D}$ 定义为一批时间序列。我们将同一批次中时间戳 t 处其他时间序列的基于频率的表示用作负样本。

对比损失由两个相互补充的损失组成，定义如下：

$$
\mathcal{L} = \frac{1}{|\mathcal{D}|T} (\mathcal{L}_{time} + \mathcal{L}_{Freq}) \,, \tag{2}
$$

其中， $\mathcal{D}$ 表示一批时间序列。如前所述，我们对时间序列表征模型进行预训练；在编码表征阶段，通过使用不同长度的回溯窗口，可以获得不同尺度的表征。本文对原始高频时间序列（分钟级）进行编码，以生成长期历史时间序列的日、周、月和季度表征。

表示增强型深度自回归模型

现在，我们详细介绍表示增强型深度自回归模型。设

时间序列（TS）的值为 $y_t \in \mathbb{R}$，其中 t 表示时间范围 $t \in \{1, 2, ..., T\}$上的时间索引。请注意，我们将 x_t 定义为时间步 t 处未来已知的协变量，例如时间特征和ID特征。给定最近的 L 个观测值 y_{t-L}，……，y_t，时间序列预测任务旨在预测未来的N个观测值 y_{t+1}，……，y_{t+N}。我们使用 r_t（长期历史时间序列的表征）以及 h_t（由短期观测值（附近窗口）得到的上下文潜在向量）来预测未来观测值。

具体而言，我们首先从时间序列数据库中加载多尺度时间序列表征 r_t（包括日、周、月甚至季度尺度），以刻画不同周期和复杂的长期时间依赖关系。其次，我们使用循环神经网络（RNN）将短期观测（临近窗口）编码为上下文潜在向量h_t，以捕捉工作负载时间序列的日内变化模式。然后，我们使用多层感知机将它们整合到相同维度。第三，受 Transformer架构启发，我们自然地使用多头自注意力作为融合模块，将长期历史时间序列表征与短期观测（临近窗口）相结合，从而获得更好的预测结果。这种架构不仅能够捕捉时间序列的长期依赖关系和周期，还能捕捉日内的短期变化。最后，在循环神经网络解码器的帮助下，我们执行自回归解码，以实现多步预测。

在训练过程中，给定 $\mathcal{D}$（定义为一批时间序列 $Y := \{y_1, y_2, ..., y_T\}$）、长期历史时间序列的表示 r_t 以及相关协变量 $X := \{x_1, x_2, ..., x_T\}$，我们可以推导出似然如下：

$$\mathcal{L} = \frac{1}{|\mathcal{D}|T} \prod_{x1:T, y1:T \in \mathcal{D}} \prod_{t=1}^{T} p(y_t | y_{1:t-1}; x_{1:t}, r_t, \Phi) , \quad (3)$$

其中 Φ 是该模型的可学习参数 el.

3.3 扩缩容决策

如前所述，我们旨在找出未来时间范围 N 内的最优 Pod 分配方案，使应用程序在目标 CPU 利用率附近稳定运行。我们假设应用程序的总工作负载通过负载均衡均匀分配到所有 Pod 上，即未来总工作负载 y_t^{total}均匀分布在所有 Pod 上。因此，为了确定扩缩容 Pod 的数量，根据上文的公式 1，我们当前的关键任务是学习 CPU 利用率与工作负载之间的关系。为此，我们提出了任务条件贝叶斯神经回归。

任务条件贝叶斯神经回归。我们提出一种新颖的方法，将任务条件超网络与贝叶斯方法相结合，构建一个能够在线适应各种场景的统一模型。该模型旨在解决挑战2中所述的 CPU 利用率与工作负载之间的不确定性问题。具体而言，我们的模型以目标 CPU 利用率 x_{cpu} 和应用的 ID特征x_{id} 作为输入 X，并输出每个Pod的工作负载y。我们的回归模型（$y = w \cdot x_{cpu} + b$）采用贝叶斯神经网络 [Blundell 等人，2015] 来估计不确定性。参数 β 包括权重 w和偏置 b。我们假设参数 β服从概率分布，其中先验分布为 $p_{prior}(\beta) \sim \mathcal{N}(0, I)$，后验分布为 $p_{posterior}(\beta|X, y)$。借助变分贝叶斯，我们可以使用可学习模型 $q_\phi(\beta|X, y) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$，近似难以计算的真实后验分布，其中 ϕ 是由任务条件信息给出的模型参数。

我们首先使用任务投影网络 $h_I(\cdot)$，为应用程序的每种场景计算一个任务嵌入 I_τ。该网络是由两个前馈层和一个 ReLU 非线性函数组成的多层感知机：

$$\boldsymbol{I_\tau} = h_I(x_{id}) , \quad (4)$$

其中，x_{id}是上述任务特征，用于表示应用程序的不同场景；任务投影网络 $h_I(\cdot)$ 则从输入任务特征中学习合适的压缩任务嵌入。

然后，我们定义一个简单的超网络线性层 $g_A^l(\cdot)$，以任务嵌入 I_τ 为输入，为上述 $q_\phi(\beta|X, y)$ 任务条件贝叶斯模型的近似分布生成参数 $\boldsymbol{\mu}$（均值）和 $\boldsymbol{\Sigma}$（方差）。通过这种方式，我们可以为每次应用程序预测提供不同的模型参数。

在训练阶段，给定定义为一批数据的 $\mathcal{D}$，我们使用随机梯度变分贝叶斯（SGVB）估计器和重参数化技巧 [（Kingma 和 Welling，2013],）训练任务条件贝叶斯神经模型，使其遵循证据下界（ELBO）：

$$\mathcal{L} = \frac{1}{|\mathcal{D}|}(KL[q_\phi(\beta|X, y)||p(\beta)] - \mathbb{E}_{\beta \sim q_\phi(\beta|X,y)} logP(X, y|\beta)) . \quad (5)$$

模型训练完成后，我们多次运行模型以生成一组样本，并计算结果的均值和方差。此外，通过固定目标 CPU 利用率，我们可以确定每个 Pod 所分配工作负载的上下界。根据上述公式（1），给定总工作负载 y_t^{total} 和每个 Pod 工作负载的上下界 y_t^{pod}，即可得出扩缩容 Pod 数量的决策范围，其中上界对应于收益最大化，下界对应于风险最小化。如图1所示，我们可以获得兼顾收益与风险的整体最优决策方案（依据对响应时间的服务容忍度）。在实际应用中，我们每五分钟扩缩容一次 Pod，以确保应用程序在目标 CPU 利用率附近稳定运行。

4 实验

在本节中，我们使用从本公司应用程序服务器收集的真实工业数据集进行了大量实证评估。我们的实验由三部分组成：**工作负载时间序列预测、CPU利用率与工作负载估算**以及**扩缩容 Pod 决策**。

4.1 设置

数据集。我们从云服务中收集了 3000 个不同在线微服务的 10 分钟频率运行数据，例如 Web服务、数据服务和 AI推理服务，包括工作负载、CPU 利用率和 Pod 数量。工作负载时间序列由应用程序每 10 分钟的最大每秒请求数（每秒请求数）组成。此外，

方法	MAE	RMSE
N-BEATS	1.851(0.071)	41.681(0.533)
DeepAR	1.734(0.030)	31.315(0.246)
Transformer	1.698(0.018)	30.359(0.136)
LogTrans	1.634(0.013)	29.531(0.375)
Informer	1.655(0.031)	30.121(0.679)
Autoformer	1.482(0.043)	27.489(0.371)
Fedformer	1.427(0.075)	26.761(0.752)
我们的模型采用表示	**1.257(0.058)**	**22.763(0.911)**
我们的模型不采用表示	1.873(0.059)	31.997(0.319)

表1：**5 次运行的平均性能（越低越好）**。我们对不含表示的模型进行了消融研究。

我们的数据还包括每 10 分钟对应的 CPU 利用率和运行中实例（Pod）数量。

实现环境。所有实验均在 Linux 服务器（Ubuntu 16.04）上运行，服务器配备 Intel® Xeon® Silver 4214 2.20GHz CPU、512GB 内存和 8 张 Nvidia® A100 GPU。

4.2 评估

工作负载时间序列预测

我们首先使用工作负载时间序列的历史数据预训练多尺度时间序列表示模型。接下来，我们使用多尺度时间序列表示评估预测模型。

基线方法与指标。我们将性能与最先进的基线方法进行比较，包括*N-BEATS*、*DeepAR*、*Transformer*、*LogTrans*、*Informer*、*Autoformer*和*Fedformer*，详见附录A。我们通过平均绝对误差（MAE）和均方根误差（RMSE）评估工作负载时间序列预测性能。

实验结果。我们进行了5次实验，并报告了平均绝对误差和均方根误差的均值与标准差。工作负载时间序列预测结果如表1所示，可以看出，我们提出的方法取得了全部最佳结果，在真实世界的高频数据集上显著提升了准确性。

此外，实验结果表明，对于高频时间序列数据，经典深度方法的效果远不及多尺度表示融合模型。

消融研究。为进一步证明第3.2节所述设计的有效性，我们使用模型变体和相关模型开展了消融研究。显然，表示增强模型的表现比原始深度模型高出约30%。

如图4所示，我们的时间序列表征能够成功刻画不同周期之间的长期时间依赖关系。我们可以直观地从该表征中看到周周期之间的依赖关系。此外，时间序列表征还挖掘出了复杂的嵌套周期，例如，如图所示，周周期包含日周期。总而言之，得益于强大的时间序列表征，我们能够捕获具有不同周期以及复杂长期时间依赖关系的高频、非平稳工作负载时间序列。此外，借助基于自注意力的融合模块，我们的模型整合了长期历史时间序列表征和短期观测值（邻近窗口），从而获得更高的准确率。

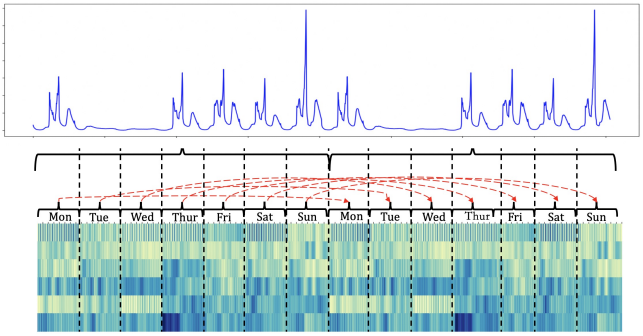


图2：**两周时间序列表征的可视化**（每10分钟）。每日和每周周期的变化清晰可见。

方法	MAE	RMSE
线性回归	0.712(0.011)	1.315(0.023)
MLP	0.781(0.071)	1.152(0.533)
XGBoost	0.628(0.002)	1.240(0.004)
高斯过程	0.677(0.018)	1.271(0.136)
神经过程	0.641(0.013)	1.275(0.375)
我们的模型	**0.365(0.005)**	**0.799(0.016)**

表2：**性能（越低越好），为5次运行的平均值**。

CPU利用率和工作负载估算

我们将应用的目标 CPU 利用率和 ID特征作为输入，并输出每个Pod的工作负载。

基线与指标。我们将性能与先进方法进行比较，包括 线性回归（*LR*）、多层感知机、*XGBoost*（*XGB*）、高斯过程（*GP*）、神经过程（*NP*） [Garnelo 等人， 2018]。请注意，对于 LR、MLP、XGB 和 GP，我们为每种场景的数据分别构建独立模型；相比之下，我们的方法使用全局统一模型（理论上更难），而 NP（通常被视为一种元学习方法）则用于适应跨场景数据，详情见附录B。我们通过平均绝对误差和均方根误差评估这些实验。

实验结果。 我们进行了5次实验，并报告了平均绝对误差（MAE）和均方根误差（RMSE）的均值和标准差。表2展示了不同方法在真实世界数据集上的性能，可以观察到，我们提出的方法（任务条件贝叶斯神经回归）取得了所有最佳结果，准确率显著提高（约40%）。特别是，通过充分利用深度贝叶斯方法的优势，我们的方法刻画了 CPU 利用率与工作负载之间的不确定性，提供了收益和风险的上下界。此外，我们利用任务条件化超网络，为每组应用数据生成不同的贝叶斯模型参数，从而能够在不同数据场景下构建一个全局统一的模型，与上述传统方法相比，大幅提高了建模效率。

扩缩容 Pod 决策

完成上述预测和估算后，我们得到总工作负载 y_t^{total}以及每个 Pod 工作负载的

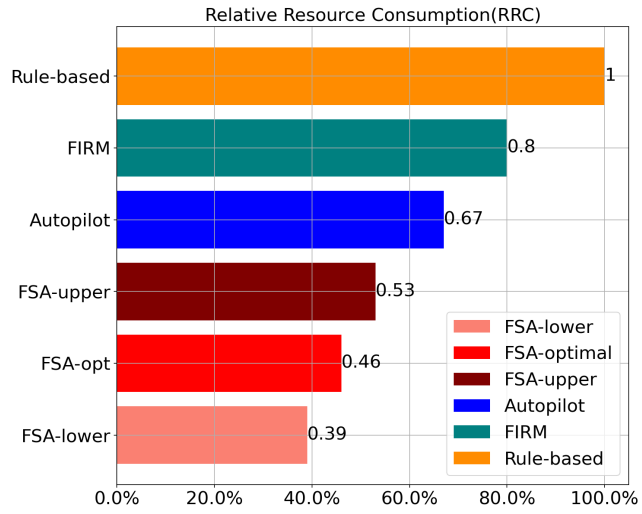


图 3：**不同自动伸缩方法的性能（越低越好）**。纵轴表示“自动伸缩方法”，横轴表示“相对资源消耗-RRC%”。

Pod 工作负载 y_t^{pod} 。因此，根据上述公式（1），我们可以得出扩缩 Pod 数量的决策范围，其中上界对应于收益最大化，下界对应于风险最小化。

基线与指标。我们还将自动伸缩方法与先进方法进行性能比较，包括基于规则的自动伸缩（基于规则）、*Autopilot* [Rzadca 等。、 2020], *FIRM* [Qiu 等。、 2020]。我们在附录 C 中详细介绍上述基线方法。

我们引入相对资源消耗（**RRC**）作为衡量自动伸缩性能的指标，定义如下：

$$RRC = c/c_r\,, \tag{6}$$

其中， c 是评估方法分配的 Pod 数量， c_r 是基线基于规则的自动伸缩方法设定的 Pod 数量。RRC 越低，效率越高（即资源消耗越低）。

实验结果。我们在 3000 个在线微服务应用上运行实验，以调整 Pod，并在图 3 中报告 RRC 和扩缩容 Pod 数量。

如图 3 所示，与其他方法相比，FSA 的性能显著提升。在确保服务等级目标稳定并维持目标 CPU 利用率的前提下，FSA 使用的资源最少。具体而言，FSA 为扩缩容 Pod 提供决策范围，其中上下界分别代表收益和风险。我们还可以根据对响应时间 (RT)的服务容忍度，在上界和下界之间取得最优值。图 5(b) 展示了各方法运行一天后的扩缩容 Pod 数量，体现了 FSA 的有效性和鲁棒性。我们对决策过程中的不确定性进行建模，为扩缩容提供上下界，使我们的方法能够确保服务等级目标稳定。

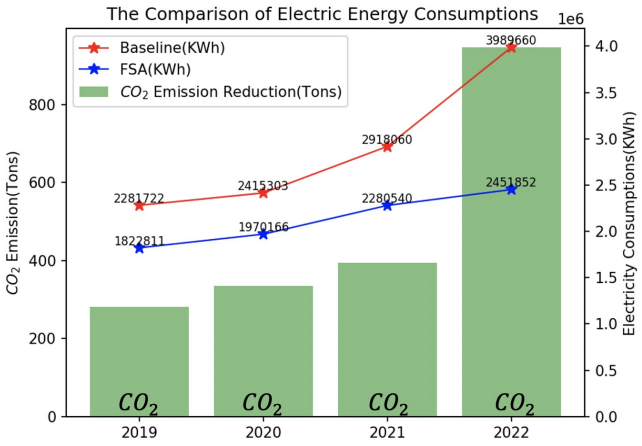


图4：**电能消耗对比**。红色线表示基准方案的用电量，蓝色线表示 FSA 的用电成本。绿色柱状图表示过去四年采用 FSA 所减少的碳排放量。

5 部署

我们的方法已成功部署在公司数据中心的服务器集群上，为 3000 多项在线应用程序服务提供资源调度支持。如图 6 所示，我们报告了过去四年中该方法在提升数据中心可持续性方面的成效，从而提高资源效率并节省电能。更多详情请参阅附录 D。

过去双十一购物节的**4 年**期间，我们应用了 FSA。根据中国环境联合认证中心（CEC）的认证，购物节期间分别减排**282、335、394 和 947 吨二氧化碳**，并节省用电**459000、545000、640000 和 1538000 千瓦时**（=）。

6 结论

本文提出了一种 Full Scaling Automation（FSA）机制，以提高数据中心中大规模云计算集群的能源效率和资源利用率。FSA 机制利用对未来工作负载需求的预测，自动调整可用资源，以满足预期的资源目标水平。在真实数据上进行的实证测试表明，与先进方法相比，该方法具有竞争力。此外，FSA 在我们自有数据中心的成功部署降低了能源消耗和碳排放，标志着我们朝着 2030 年实现碳中和迈出了重要一步。

贡献声明

设计、实施和部署主要由 Shiyu Wang、Yinbo Sun 和 Xiaoming Shi 完成——三位作者在这项工作中发挥了关键作用。

参考文献

[Abdullah等人， 2020] Muhammad Abdullah、Waheed Iqbal、Josep Lluis Berral、Jorda Polo 和 David Carrera。面向容器化微服务的突发感知预测性自动扩缩容。IEEE服务计算汇刊，15(3):1448–1460，2020。[Bishop 和 Nasrabadi， 2006] Christopher M Bishop 和 Nasser M Nasrabadi。模式识别与机器学习，第4卷。Springer，2006。[Blundell等人， 2015] Charles Blundell、Julien Cornebise、Koray Kavukcuoglu 和 Daan Wierstra。神经网络中的权重不确定性。载于国际机器学习会议，页码1613–1622。PMLR，2015。[Eldele等人， 2021] Emadeldeen Eldele、Mohamed Ragab、Zhenghua Chen、Min Wu、Chee Keong Kwoh、Xiaoli Li 和 Cuntai Guan。通过时间与上下文对比进行时间序列表示学习。arXiv预印本 arXiv:2106.14112，2021。[Garnelo等人， 2018] Marta Garnelo、Jonathan Schwarz、Dan Rosenbaum、Fabio Viola、Danilo J Rezende、SM Eslami 和 Yee Whye Teh。神经过程。arXiv预印本 arXiv:1807.01622，2018。[He 等人， 2022] Yun He、Steven Zheng、Yi Tay、Jai Gupta、Yu Du、Vamsi Aribandi、Zhe Zhao、YaGuang Li、Zhao Chen、Donald Metzler 等。Hyperprompt：基于提示的 Transformer 任务条件化。载于国际机器学习会议，页码8678–8690。PMLR，2022。[Kingma 和 Welling， 2013] Diederik P Kingma 和 Max Welling。自编码变分贝叶斯。arXiv预印本 arXiv:1312.6114，2013。[Krieger等人， 2017] Michael T Krieger、Oscar Torreno、Oswaldo Trelles 和 Dieter Kranzlm¨uller。构建具有自动扩缩容资源的开源云环境，以执行生物信息学和生物医学工作流。未来一代计算机系统，67:329–340，2017。[Mahabadi 等人， 2021] Rabeeh Karimi Mahabadi、Sebastian Ruder、Mostafa Dehghani 和 James Henderson。通过共享超网络实现 Transformer 的参数高效多任务微调。arXiv预印本 arXiv:2106.04489，2021。[Nguyen等人， 2020] Thanh-Tung Nguyen、Yu-Jin Yeom、Taehong Kim、Dae-Heon Park 和 Sehan Kim。面向弹性容器编排的 Kubernetes 水平Pod自动扩缩容。Sensors，20(16):4621，2020。[Oreshkin等人， 2019] Boris N Oreshkin、Dmitri Carpov、Nicolas Chapados 和 Yoshua Bengio。N-BEATS：用于可解释时间序列预测的神经基扩展分析。arXiv预印本 arXiv:1905.10437，2019。[Qiu等人， 2020] Haoran Qiu、Subho S Banerjee、Saurabh Jha、Zbigniew T Kalbarczyk 和 Ravishankar K Iyer。{FIRM}：面向 {SLO的} 微服务智能细粒度资源管理框架。载于第14届USENIX操作系统设计与实现研讨会（OSDI 20），页码805–825，2020。

[Rzadca 等, 2020] Krzysztof Rzadca, Pawel Findeisen, Jacek Swiderski, Przemyslaw Zych, Przemyslaw Broniek, Jarek Kusmierek, Pawel Nowak, Beata Strack, Piotr Witu- sowski, Steven Hand 等。Autopilot：Google 的工作负载自动扩缩容。发表于第十五届欧洲计算机系统会议，第 1–16 页，2020 年。[Salinas等， 2020] David Salinas、Valentin Flunkert、Jan Gasthaus 和 Tim Januschowski。DeepAR：利用自回归循环网络进行概率预测。国际预测期刊，36(3):1181–1191，2020 年。[Woo 等， 2022] Gerald Woo、Chenghao Liu、Doyen Sahoo、Akshat Kumar 和 Steven Hoi。CoST：用于时间序列预测的解耦季节性-趋势表示的对比学习。*arXiv* 预印本 *arXiv:2202.01575*，2022 年。[Yue 等， 2022] Zhihan Yue、Yujing Wang、Juanyong Duan、Tianmeng Yang、Congrui Huang、Yunhai Tong 和 Bixiong Xu。TS2Vec：迈向时间序列的通用表示。发表于 *AAAI* 人工智能会议论文集，第 36 卷，第 8980–8987 页，2022 年。[Zhou 等， 2021] Haoyi Zhou、Shanghang Zhang、Jieqi Peng、Shuai Zhang、Jianxin Li、Hui Xiong 和 Wancai Zhang。Informer：超越高效 Transformer 的长序列时间序列预测。发表于*AAAI* 论文集，2021 年。