中图分类号：TP391 论文编号：
学科分类号：081200

硕士学位论文

基于双时序流量预测的自响应动态负载均衡技术的研究

研究生姓名 高自强
学科、专业 计算机技术
研 究 方 向 人工智能
指 导 教 师 顾晶晶 教授

南京航空航天大学
研究生院 计算机科学与技术学院

二O二三年四月

Nanjing University of Aeronautics and Astronautics
The Graduate School
College of Computer Science and Technology

Research of self-response dynamic load balancing technology based on dual time series traffic prediction

A Thesis in
Computer Science and Technology

by

Ziqiang Gao

Advised by

Prof. Jingjing Gu

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Master of Engineering

April, 2023

承诺书

本人声明所呈交的博/硕士学位论文是本人在导师指导下进行的研究工作及取得的研究成果。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得南京航空航天大学或其他教育机构的学位或证书而使用过的材料。

本人授权南京航空航天大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本承诺书）

作者签名：_____

日　　期：_____

摘　要

高可用、高可靠的服务器集群设计与实现可为用户提供准确、高效的服务，基于用户请求和集群负载双时序流量预测的自响应动态负载均衡集群系统建设对于提升集群全局任务分配和局部动态负载调度，进而实现集群整体高并发、低损耗具有重要意义。目前，基于用户请求和集群负载时序流量预测的集群系统存在以下挑战：1）用户请求历史数据少，可利用数据不足；2）用户请求和集群负载时序数据在周期性模式上均呈现短时和长时周期分化的特点，且长短时周期特性不同；3）现有负载均衡策略未能兼顾全局和局部均衡，在局部负载调度方面存在局部失衡问题。

本文对基于双时序流量预测的自响应动态负载均衡技术展开研究，针对上述挑战，主要研究内容如下：

（1）针对用户请求流量历史累积数据少，以及用户请求流量和集群负载流量共有的数据周期性差，预测无法同时兼顾短时、长时预测的问题，本文设计了一种基于加权长短时特征融合的双时序流量预测模型。首先分别利用基于查询路径优化的 DTW 用户请求时序数据聚类算法和多变量联合特征选择技术对用户请求和集群负载数据进行定向处理；然后利用基于注意力机制的加权长短时特征融合技术对两类时序数据进行短时与长时特征提取、长短时特征融合以及向量加权等处理，充分挖掘时序数据的长短时特征，实现高准确度的时序流量短期预测和长期预测。

（2）在集群综合负载均衡方面，本文分别对全局任务分配和局部动态负载调度两类工作建立不同的处理机制。针对全局任务分配，本文设计了基于预测自响应的全局任务分配模型，该模型在双时序流量预测的基础上，挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立合理的全局任务分配模型。实现准确、高效的全局任务分配，保证集群运行在较低负载均衡度；针对局部动态负载调度，本文提出了基于集群服务器自索取的局部动态负载调度模型，该模型借助基于集群服务器自索取的局部动态负载调度方法，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载实现了集群局部负载均衡，同时减少了服务器集群整体资源消耗。

（3）在公开数据 google-cluster-trace-v2011 和 alibaba-cluster-trace-v2018 以及自定义测试数据集上进行模型测试实验，验证了本文所提出算法的有效性。在上述研究的基础上，完成了基于双时序流量预测的自响应动态负载均衡系统的初步设计和实现。结果表明，本文设计的基于双时序流量预测的自响应动态负载均衡模型能够在保证用户请求准确、高效处理的同时，实现集群系统整体负载均衡，提高系统吞吐量。

关键词：集群，负载均衡，双时序流量，用户请求预测，负载预测，动态调度

ABSTRACT

The design and implementation of a highly available and reliable server cluster can provide users with accurate and efficient services. The construction of a self-response dynamic load balancing cluster system based on user requests and cluster load dual time-series traffic prediction is of great significance for improving the overall task allocation and local dynamic load scheduling of the cluster, and thus realizing the overall high concurrency and low loss of the cluster. At present, the cluster system based on user requests and cluster load time-series traffic prediction has the following challenges: 1) There is little historical data of user requests and insufficient available data; 2) Both user requests and cluster load timing data have the characteristics of short-term and long-term cycle differentiation in the periodic mode, and the characteristics of long-term and short-term cycles are different; 3) The existing load balancing strategies fail to give consideration to both global and local balancing, and there is a local imbalance problem in local load scheduling.

In this paper, the self-response dynamic load balancing technology based on dual time-series traffic prediction is studied. In view of the above challenges, the main research contents are as follows:

(1) In view of the problem that the historical cumulative data of user request traffic is few, and the periodicity of the data shared by user request traffic and cluster load traffic is poor, the prediction can not take into account both short-term and long-term prediction, this paper designs a dual time-series traffic prediction model based on weighted long-short-time feature fusion. Firstly, DTW user request time-series data clustering algorithm based on query path optimization and multivariable joint feature selection technology are used to preprocess user request and cluster load data directionally; Then, the weighted long-short-time feature fusion technology based on attention mechanism is used to process the two types of time-series data, such as short time and long time feature extraction, long time and short time feature fusion, and vector weighting, to fully mine the long time and short time features of time-series data, and to achieve high accuracy short-term prediction and long-term prediction of time series traffic.

(2) In the aspect of cluster integrated load balancing, this paper establishes different processing mechanisms for global task allocation and local dynamic load scheduling. For global task allocation, a global task allocation model based on predictive self-response is designed in this paper. On the basis of dual time series traffic prediction, this model mines the interaction and response relationship between real-time user requests, real-time cluster load and predictive user requests, predictive cluster load and server performance, and establishes a reasonable global task allocation model. Implement accurate and efficient global task allocation, and ensure that the cluster runs at a low load balance. For local dynamic load scheduling, this paper proposes a local dynamic load scheduling model based on cluster server self-

request. With the help of local dynamic load scheduling method based on cluster server self-request, this model coordinates the task distribution relationship between local adjacent server nodes, balances the load among server nodes to achieve local load balancing of the cluster, and reduces the overall resource consumption of the server cluster.

(3) Model test experiments were conducted on the public data google-cluster-trace-v2011 and alibaba-cluster-trace-v2018, as well as custom test data sets, to verify the effectiveness of the algorithm proposed in this paper. On the basis of the above research, the preliminary design and implementation of a self-response dynamic load balancing system based on dual time series traffic prediction is completed. The results show that the self-response dynamic load balancing model based on dual temporal traffic prediction designed in this paper can achieve the overall load balancing of the cluster system and improve the system throughput while ensuring the accurate and efficient processing of user requests.

Keywords: Cluster, Load balancing, Dual sequential traffic, User request prediction, Load prediction, Dynamic scheduling

目 录

图表清单

注释表

W	时序序列路径距离矩阵	M_{avg}	平均内存容量
D	时序序列整体路径距离	D_{avg}	平均磁盘容量
X_a	整数点 A	N_{avg}	平均网络带宽
X_c	整数点 C	c_i	服务器 i 的 CPU 频率
m	时序序列 m	m_i	服务器 i 的内存容量
n	时序序列 n	d_i	服务器 i 的磁盘容量
Ω_1	矩形	n_i	服务器 i 的网络带宽
X	时序负载数据	AL_i	实时负载因子
S	短时特征向量	L_{c_i}	服务器 i 的 CPU 使用率
C_{t-1}	前一时刻神经元的状态	L_{m_i}	服务器 i 的内存使用率
h_{t-1}	前一时刻神经元的输出	L_{d_i}	服务器 i 的磁盘使用率
x_t	当前时刻的输入	L_{n_i}	服务器 i 的网络带宽占用率
W_i	输入门的权重矩阵	SL_i	静态负载因子
b_i	输入门的偏置常数	PL_i	预测负载因子
W_f	遗忘门的权重矩阵	L_{pc_i}	服务器 i 预测得到的 CPU 利用
b_f	遗忘门的偏置常数	L_{pm_i}	服务器 i 预测得到的内存利用率
W_o	输出门的权重矩阵	L_{pd_i}	服务器 i 预测得到的磁盘利用率
b_o	输出门的偏置常数	L_{pn_i}	服务器 i 预测得到的网络带宽占
L	长时特征向量	RA_i	自响应实时负载因子
M_{sl}	长短时融合特征向量	$Load$	服务器实时负载
W_m	加权长短时融合特征向量	$LoadMax$	负载上限
C_i	注意力向量	$LoadMin$	负载下限
h_j	隐藏状态	ε	负载的均方差
D	解码向量	L_{avg}	服务器 i 的实时负载
RO_i	服务器资源占用因子	L_{multi}	平均实时负载
C_{avg}	平均 CPU 频率	ω_i	服务器 i 的综合权值

缩略词

缩略词	英文全称	中文全称
SVM	Support Vector Machine	支持向量机
SVR	Support Vector Regression	支持向量回归
ANN	Artificial Neural Network	人工神经网络
LSTM	Long Short Term Memory	长短期记忆网络
RNN	Recurrent Neural Networks	循环神经网络
GRU	Gated Recurrent Unit	门控循环单元
FCN	Fully Convolutional Network	全卷积网络
1D FCN	1D-Fully Convolutional Network	一维全卷积网络
TCN	Temporal Convolutional Network	时序卷积网络
DTW	Dynamic Time Warping	动态时序规整
MAE	Mean Absolute Error	平均绝对误差
RMSE	Root Mean Square Error	均方根误差
MAPE	Mean Absolute Percentage Error	平均绝对百分比误差
AR	Auto-regressive	自回归模型
MA	Moving Average	滑动平均模型
ARIMA	Autoregressive Integrated Moving Average	差分整合移动平均自回归
RO	Resource Occupancy	资源占用因子
AL	Real-time Load	实时负载因子
SL	Static Load	静态负载因子
PL	Prediction Load	预测负载因子
RA	Responce Actual Load	自响应实时负载因子
DL	Dynamic Load	动态负载因子
DLBLF	Research on a Dynamic Load Balancing model and algorithm based on Prediction	一种基于预测的动态负载均衡模型及算法研究
DLBDS	A Dynamic Load Balancing Strategy in Distributed Systems	一种分布式系统中动态负载均衡策略
UML	Unified Modeling Language	统一建模语言
SDLB-DS	Self-response Dynamic Load Balancing based on Double Time-series traffic prediction	基于双时序流量预测的自响应动态负载均衡

第一章 绪论

1.1 课题背景与研究意义

自互联网诞生以来，互联网技术取得迅猛发展，其中尤以移动互联网为甚。据统计，以手机、平板为代表的全球终端总数已经达到 62 亿余台[1]。因而，如何满足数以几十亿计的用户访问请求，并保证用户服务的高性能、高可用与高可拓展性，给提供用户服务的后端服务器结构设计带来了巨大的挑战。

为解决这一问题，为用户提供及时、可靠、高效的网络服务，同时控制后端服务器的资源消耗，很多服务器结构设计方案被提出。根据用户请求规模和实际业务场景的不同，主要有单一服务器和服务器集群这两种方案[2]。

为应对高并发用户访问请求，单一服务器方案主要通过升级服务器硬件配置，提升单台服务器性能这一方式来解决。例如，使用性能更高的 CPU、容量更大的存储设备、更高效的数据传输协议等。显然，通过升级硬件配置的方式来提升单台服务器设备的性能这一方法是存在局限性的[3]。一方面，在高并发用户请求场景中，该方法显然不能奏效；另一方面，对互联网厂商而言，如何在满足用户请求的前提下尽可能降低服务器消耗成本是其要考虑的核心问题。显然，提升单一服务器端的硬件性能无法有效解决高并发用户请求问题。

针对上述单一服务器方案存在的诸多弊端，服务器集群方案应运而生。服务器集群方案主要有以下几个特点[4]。第一，服务器集群对单台服务器的性能要求不高，不需要为每台服务器配置最佳的硬件性能；第二，服务器集群借助用户请求分配方案，将不同用户请求分发至不同的后端服务器，以满足不同用户请求；第三，服务器集群需要合理、高效的集群架构设计和管理方案，统一管理用户请求分发、服务器资源迁移等工作，以保证服务器集群的高性能、高可用与高可拓展性[5]。基于此，集群负载均衡是服务器集群方案中一项重要的功能和性能要求。负载均衡的核心思想是，通过一台中转路由服务器，根据后端集群中不同服务器的性能以及用户请求等信息，将最适合当前用户请求的后端服务器分配给该用户请求，从而实现集群中不同服务器之间的负载和性能均衡[6]。

但是，目前服务器集群负载均衡存在以下两方面的瓶颈：

第一，高并发用户请求导致网络链路频频发生拥堵，致使数据传输过程中发生数据包延迟甚至丢失，由此导致整体网络为用户提供的服务能力大大降低[7]。因而即使在服务器处理能力足够的情况下也可能因为网络链路拥塞的问题降低整体效率，甚至出现某些服务器空载的情况，造成服务器资源的严重浪费。

第二，当集群系统中服务器在计算速度、通信能力以及存储容量等自身性能方面存在较大差异时，不能充分考虑服务器性能对集群进行动态负载调度会产生不合理的任务分配，导致部分服务器和局部网络负载过重的同时，某些服务器和链路处于空载甚至空闲状态[8]。

针对上述瓶颈，目前主流解决方案分别从用户请求和集群负载两个方面着手，分别对用户请求流量和集群负载流量进行研究工作，以实现集群负载均衡。基于此，目前常用的负载均衡算法可以分为基于传统软件的方法和基于流量预测的方法。基于软件方法的负载均衡调度策略主要有基于随机选择任务移动节点的概率调度算法、根据负载变化差额而基于梯度模型的调度算法以及自适应的近邻契约算法等；基于流量预测的动态均衡调度策略主要有基于用户请求流量的负载均衡算法和基于服务器负载流量预测的负载均衡方法，这些方法根据预测结果制定负载均衡策略。

1.2 国内外研究现状

1.2.1 集群技术的发展

在以云计算为代表的集群高并发用户请求场景中，通过提高 CPU 主频、增加内存容量以及拓展总线带宽等方式提升单台服务器的性能显然无法应对大量、高频用户访问请求。服务器集群技术的发展，为高并发应用场景提供了一套更好的解决方案。

集群是一组相互独立的、通过网络互联的计算机，它们构成了一个组，并以单一系统的模式加以管理[9]。集群系统中的每一个服务器节点都是一台独立的物理设备，其他节点的状态变化不会影响该节点的正常运作。在集群系统运行过程中，若单个服务器节点出现宕机等故障导致其不能继续提供服务，集群系统会选出下一个节点作为该业务运行的替代载体，以保证集群服务的高可用和高可靠性。因此，用户与集群相互作用时，对于用户而言，一个集群相当于一台单独的服务器。一个可靠的服务器集群系统应具备高性能、高可用和高可拓展等特性。

集群技术是一种服务器架构技术，其通过硬件或软件技术将一些独立的服务器组织在一起，共同处理高并发用户请求[10]。集群技术可以有效提高集群系统的高性能、高可用和高可拓展等性能。集群技术解决了单个服务器存在的运算能力和 I/O 性能不足等问题，提高了集群服务的可靠性，使集群获得可拓展能力，降低集群整体的运维成本。根据组成集群系统的计算机之间的体系结构特征，集群可分为同构集群与异构集群；根据业务场景和技术点的不同，集群可分为三种类型，即高可用集群、高性能集群和负载均衡集群。每种集群的介绍如下。

高可用集群一般是指当集群中某个节点失效时，其上的任务会自动转移到其他正常的节点上；还指可以将集群中的某节点进行离线维护再上线，该过程并不影响整个集群的运行。当节点中运行任务的应用程序出现故障，或者系统硬件如网络出现故障时，集群可以将任务自动、快速地从一个节点切换到另一个节点，从而保证集群持续、不间断地对外提供服务[11]。

高性能计算集群将计算任务分配到集群的不同节点而提高计算能力，因而主要应用在科学计算领域。比较流行的高性能计算集群采用 Linux 操作系统和其他一些免费软件来完成并行运算。这一集群配置通常被称为 Beowulf 集群。这类集群通常运行特定的程序以发挥高性能计算集群的并行能力。这类程序一般应用特定的运行库，比如专为科学计算设计的 MPI 库。高性能计算集群适用于在计算过程中各计算节点之间发生大量数据通讯的计算作业，比如一个节点的中间结果或影响到其他节点计算结果的情况[12]。

负载均衡集群由两台或者两台以上服务器组成，分为前端负载调度和后端服务两个部分。负载调度部分负载把用户的请求按照不同的策略分配给后端服务节点，而后端节点是真正提供应用程序服务的部分[13]。与高可用集群不同的是，负载均衡集群中，所有的后端节点都处于活动状态，它们都对外提供服务，分摊系统的工作负载。负载均衡集群可以把一个高负载的用户请求分散到多个节点共同完成，适用于高并发、大负载访问的应用系统。但是它也有不足的地方：当一个节点出现故障时，前端调度系统并不知道此节点已经不能提供服务，仍然会把用户请求调度到故障节点上来，这样访问就会失败。为了解决这个问题，负载调度系统一般都引入了节点监控系统。节点监控系统位于前端负载调度机上，负责监控下面的服务节点。当某个节点出现故障后，节点监控系统会自动将故障节点从集群中剔除；当此节点恢复正常后，节点监控系统又会自动将其加入集群中，而这一切对用户来说是完全透明的。

综上所述，由于负载均衡集群在分发用户请求、调节服务器节点负载，保证集群整体负载均衡等方面存在的突出优势，负载均衡集群已经在高并发、高可用集群场景中取得了广泛应用，是目前高并发用户场景中主流应用集群架构之一，本文正是基于负载均衡系统进行了深入研究。

1.2.2 负载均衡技术及其研究现状

1）负载均衡技术

负载均衡技术于 1996 年由 Foundry 公司提出，是现代计算机领域的基础技术之一。其基本原理是通过运行在前端的负载均衡服务器，根据负载均衡算法，将用户请求分配到后端服务器节点上，从而提高整个系统的扩展能力，实现服务的并行扩展[14]。同时，负载均衡技术还可以起到对外网屏蔽内网服务器的作用，从而提高系统的可用性。一般来说，负载均衡技术具有两个方面的含义：一方面，负载均衡技术对用户请求进行了合理分配，后端多台服务器设备共同处理任务，使得整个集群系统的处理能力大大提高；另一方面，由单台服务器拓展为多台服务器处理，缩短了集群系统响应和用户等待的时间。

针对不同的分类标准，目前有多种不同的负载均衡技术以满足不同的用户请求。根据负载均衡所采用的设备对象、负载均衡的作用范围以及应用的网络层次等三个方面，负载均衡技术可以分为以下几类[15]。

（1）软/硬件负载均衡

软件负载均衡解决方案是指在一台或多台服务器的操作系统上安装一个或多个附加软件来实现负载均衡[16]。该种解决方案基于特定环境，配置简单、使用灵活、成本低廉，可以满足一般的负载均衡需求。

当然，软件解决方案存在较多缺点。因为每台服务器上安装的额外的软件运行会消耗系统不定量的资源，越是功能强大的模块，消耗得越多。所以当连接请求并发量特别大的时，软件本身会成为服务器工作成败的一个关键；受操作系统的限制，软件可扩展性欠佳；另外，操作系统本身存在的一些问题，往往会导致集群安全问题。

硬件负载均衡解决方案是直接在服务器和外部网络间安装负载均衡设备，该设备通常被称为负载均衡器。基于专门的设备完成专门的任务，硬件负载均衡器独立于操作系统，其整体性能得到极大提高。加上多样化的负载均衡策略、智能化的流量管理，硬件负载均衡解决方案可达到最佳的负载均衡效果[17]。

硬件负载均衡器有多种多样的形式，除了作为独立意义上的负载均衡器外，有些负载均衡器集成在交换设备中，置于服务器与公共网络之间；有些则通过两块网络适配器将这一功能集成到服务器中，一块连接到公共网络上，一块连接到后端服务器群的内部网络上。

整体而言，硬件负载均衡解决方案在功能、性能上优于软件方式，但成本昂贵。

（2）本地/全局负载均衡

本地负载均衡是指对本地的服务器集群做负载均衡，全局负载均衡是指对分别放置在不同的地理位置、有不同网络结构的服务器集群作负载均衡。

本地负载均衡能有效地解决数据流量过大、网络负载过重的问题，并且不需花费昂贵开支购置性能卓越的服务器，充分利用现有设备，避免服务器单点故障造成数据流量的损失。其通过灵活多样的负载均衡策略把用户请求流量合理地分配给集群后端服务器使其共同负担。若需要为现有服务器扩充升级，只需简单地增加一个新的服务器到服务集群中，而不需改变现有网络结构、停止现有的服务。

全局负载均衡适用于服务器节点分布在不同区域的集群[18]。该负载均衡方案可以使用户只以一个 IP 地址或域名就能访问到离自己最近的服务器，从而获得最快的访问速度。该方案也可用于子公司站点分散较广的大公司，通过企业内部互联网来达到资源统一合理分配的目的。

全局负载均衡有以下特点：第一，实现地理位置无关性，能够远距离为用户提供完全的透明服务；第二，除了能避免服务器、数据中心等的单点失效，也能避免由于 ISP 专线故障引起的单点失效；第三，解决网络拥塞问题，提高服务器响应速度，服务就近提供，实现更好的访问质量。

（3）不同网络层次的负载均衡

针对网络上负载过重导致的不同瓶颈，从网络的不同层次入手，可以采用相应的负载均衡

技术来解决现有问题。现代负载均衡技术通常操作于网络的第四层或第七层。

第四层负载均衡技术将一个公共网络上合法注册的 IP 地址映射为多个内部服务器的 IP 地址，对每次 TCP 连接请求动态使用其中一个内部 IP 地址，达到负载均衡的目的。在第四层交换机中，此种均衡技术得到广泛的应用，一个目标地址是服务器集群虚拟 IP 的连接请求的数据包流经交换机，交换机根据源端和目的 IP 地址、TCP 或 UDP 端口号和对应的负载均衡策略，在服务器 IP 和服务器集群虚拟 IP 间进行映射，选取服务器群中性能最佳的服务器来处理连接请求[19]。

第七层负载均衡技术控制应用层服务的内容，提供了一种对访问流量的高层控制方式，适合对 HTTP 等应用层协议服务器集群的应用。第七层负载均衡技术通过检查流经的 HTTP 等应用层传输报文的报头，根据报头内的信息来执行负载均衡任务。

从负载均衡技术的应用来看，基于集群负载均衡技术实现的高可用和高可靠特性，负载均衡技术的应用主要有以下几个方面。

第一，用于解决高并发问题，主要应用于高访问量的业务；

第二，根据业务发展扩展应用程序；

第三，可以在负载均衡集群下添加多台服务器实例，解决单点故障问题；

第四，在各地域部署多可用区，实现同城容灾；

第五，将域名解析到不同地域的负载均衡集群下，实现全局负载均衡，解决跨地域容灾问题。

2）负载均衡技术发展现状

互联网技术与应用的快速普及，伴随互联网终端用户的快速增长，国内外互联网市场均涌现出众多“头部”互联网厂商，其旗下产品的用户规模可达上亿甚至十亿级。不同企业产品其应用场景也呈现不同特点。为保证用户体验，为用户提供高可用、高可靠服务，服务器集群负载均衡技术的研究得到了众多科研工作者和互联网厂商的广泛关注，该技术也取得了极大的发展。根据负载均衡技术的应用场景，负载均衡技术覆盖了分布式计算、并行计算、网格计算以及云计算等众多应用和技术场景[20]。根据负载均衡技术的策略和所引用的系统规模，负载均衡技术的发展呈现从静态向动态、从集中式到分布式的发展趋势和特点。根据负载调节方式的不同，集群负载均衡策略可分为静态策略和动态策略；根据负载控制方式的不同，集群负载均衡策略可分为集中式策略和分布式策略。

国内外对集群负载均衡技术的研究侧重点略有不同，下面分别为国内外负载均衡技术的研究现状。

就国内研究现状而言，在负载均衡技术的理论研究层面，算法优化是主要研究方向；在负载均衡技术的应用研究层面，对已有负载均衡软件产品进行改进以提高负载均衡软件的可用性

是主要研究方向。在理论研究层面：文献[21]提出一种基于布谷鸟搜索的集群负载均衡多目标优化调度算法，该调度算法根据最优匹配集进行用户任务的调整与转发。文献[22]针对物联网中智能终端设备数量快速增长导致的移动网络拥塞问题，构建一种基于雾集群协作的云雾混合计算模型，在考虑集群负载均衡的同时引入权重因子以平衡任务运行时延和资源消耗，最终实现系统时延能耗加的权和最小。文献[23]提出一种基于负载反馈的分布式数字集群动态负载均衡算法，实现公网数字集群系统负载均衡，同时提高集群用户规模。文献[24]提出一类基于动态调节的闭环负载分配策略，该策略建立处理不同服务请求与负载均衡的内在动态映射关系,以优化静态页面缓存与调用方式；采用负载率偏差最小的任务权重最优分配模型,基于服务器负载率动态预测和均衡指标，制定服务器集群处理混合页面访问的负载均衡分配策略。在应用研究层面：文献[25]分析 Nginx 服务器负载均衡方案的体系架构，研究传统的加权轮询算法，通过实时收集负载信息,重新计算并分配权值等改进措施，构建出一种改进后的动态负载均衡算法。文献[26]为了减轻快速增长的网络负载压力,为 Web 后端服务器集群搭建了基于 Nginx 的负载均衡服务器,将其作为集群的反向代理服务器,使集群具备了负载均衡的功能；针对 Nginx 自带负载均衡策略的缺陷提出了一种动态自适应负载均衡算法--改进型加权最小连接数算法。

就国外研究现状而言，在负载均衡技术的理论研究层面，其主要研究方向在于云计算环境下的负载均衡算法优化；在负载均衡算法的应用研究层面同样侧重于对已有负载均衡软件产品进行改进以提高负载均衡软件的可用性。文献[27]提出一种基于云分区概念的负载平衡模型，该模型将博弈论应用于负载平衡策略，以提高云计算环境中的效率；借助切换机制，实现针对不同的负载场景选择不同的策略。文献[28]提出利用适应度函数和双寡头博弈理论将任务分配给能够处理传入任务的资源需求的物理机器，以优化数据中心的负载平衡，避免资源过载或利用不足，实现集群资源的有效利用。文献[29]提出一种基于双阈值的功率感知蜜蜂负载平衡算法，将传入的用户请求公平、均匀地分配给所有虚拟机。实现消耗最少的资源满足用户服务需求。

1.2.3 负载预测技术及其研究现状

在服务器集群系统中，负载均衡和资源分配是实现集群高可用和高可靠性的两项关键技术。其中资源管理和分配是集群系统中控制成本和合理分配服务器计算能力的重要算法，集群系统进行资源管理和分配时一项很重要的参考指标便是各服务器节点的负载。随着国内外研究人员对负载均衡和资源管理技术的深入研究，以及人工智能的快速发展，集群资源管理和分配方案性能参考指标由原来的服务器节点静态负载逐渐转向对服务器节点动态负载。服务器节点动态负载的获取一个重要的方式便是负载预测。实时性要求较高的用户请求，需要集群系统做出高效、准确的任务分配和调度，此时，准确的服务器节点负载预测起到至关重要的作用。基于负载预测的动态负载均衡对于集群高效任务分配和调度而言具有重要意义。因此，越来越多的集

群资源管理和分配方案都在利用负载预测技术来提升服务器节点的动态负载获取准确度，进而提升整个集群负载均衡和资源管理的质量和准确度。

集群服务器节点负载预测一直是国内外研究人员的研究热点，目前，国内外在该领域的研究主要集中在服务器节点负载时序流量的预测。根据研究方法的不同，对于时序数据的预测主要有三类方法，分别为基于传统线性回归模型的负载预测方法，基于传统机器学习的负载预测方法和基于深度学习的负载预测方法。

第一类基于传统线性回归模型的负载预测方法主要有自回归(Autoregressive,AR)、滑动平均(Moving Average,MA)、自回归移动平均(Autoregressive Moving Average, ARMA)以及差分整合移动平均自回归(Autoregressive Integrated Moving Average, ARIMA)等模型。文献[30]应用 AR 和 ARIMA 两种模型对软件定义网络 SDN 时序流量进行预测，从平均绝对百分比误差（MAPE）来看，ARIMA 的预测精度高于 AR。这些模型在复杂度低、线性关系较强的数据中可以实现较好的预测效果，因此此类方法存在对数据的限制较多，且需要人工调整模型参数等方面的不足。

第二类基于传统机器学习的负载预测方法主要有马尔科夫模型、贝叶斯模型、支持向量回归(Support Vector Regression,SVR)模型，以及决策树和传统人工神经网络(Artificial Neural Networks, ANN)等模型。文献[31]提出一种改进灰狼搜索算法优化支持向量机（SVM）的短期云计算资源负载预测模型。该模型能够更加准确地刻画云计算短期资源负载的复杂变化趋势, 从而有效提升云计算资源负载短期预测的精度。文献[32]讨论了人工神经网络（ANN）在负载预测中的应用和训练，以及使用人工神经网络进行短期负荷预测的可能性。此类方法能够提取时序数据中的短期非线性特征，但无法捕获数据中的长期依赖关系，因此其在长期预测方面存在较大不足。

第三类基于深度学习的预测方法在时序数据预测方面取得了较好的发展。文献[33]利用集群负载数据中不同特征之间的相互作用关系，使用多维特征进行负载预测。为了解决集群长期负载存在的模式转换和振幅波动问题，更好挖掘负载数据的不同周期模式，文献[34]提出一种多尺度注意力机制，根据不同的周期模式设置不同的注意力权重，提高集群负载的长期预测能力。为提高模型的短期预测能力，文献[35]提出将 TCN 时序神经网络用于集群负载预测，同时利用多维特征进行目标特征预测。该方法在短期预测方面具备较好表现，但是长期预测能力存在很大不足。

1.3 存在的挑战

互联网技术的快速普及使得互联网用户在过去十几年中实现了高速增长，随之而来的是海量用户请求。为保证用户服务质量，实现高可用、高可靠服务器集群系统，以实现集群动态负载均衡目标为代表的负载预测与负载均衡技术取得了长足发展。例如，文献[36]提出一种基于注

意机制的 LSTM 编码器-解码器机制，该方法基于集群负载预测，借助编码-解码特征提取和注意力机制，在云计算等混合工作负载预测中实现了较好的性能表现；文献[37]提出一种基于负载反馈的分布式数字集群动态负载均衡算法,该算法实现公网数字集群系统负载均衡,同时提高用户请求容量。实现更好的负载均衡度和更低的用户请求响应延迟。然而，目前实现服务器集群动态负载均衡存在以下两方面的瓶颈：

第一，集群负载呈现如下两个特点：1）短时间跨度内，负载变化呈现无周期性和波动性；2）长时间跨度内，负载变化呈现周期性特点，且不同时间跨度呈现不同的周期模式。因此，就基于负载预测实现集群均衡而言，如何提高负载预测的准确度，同时兼顾短期预测和长期预测，是负载预测中需要解决的一个关键问题。

第二，服务器集群将多台服务器节点连接到一起，在减轻单台服务器压力的同时为用户提供高质量的服务。单台服务器完成负载预测后，集群如何利用负载预测结果制定负载均衡策略，兼顾全局任务分配和局部负载调度，在实现为服务器节点合理分配任务的同时，兼顾局部节点负载均衡。从整体上实现集群全局任务分配和局部动态负载调度的协调，保证集群负载均衡解决方案的有效性和系统性是另一个很关键的问题。

1.4 主要研究工作

根据以上挑战，本文研究了基于双时序流量预测的自响应动态负载均衡技术。主要包括基于加权长短时特征融合的双时序流量预测方法、基于预测自响应的全局任务分配方法、基于集群服务器自索取的局部动态负载调度方法。并最终根据对上述技术的研究与分析，实现了一个基于双时序流量预测的自响应动态负载均衡集群系统。本文研究内容分为四个部分：

研究内容一，基于用户业务请求流量和集群负载流量共有的时序特征，对两类时序流量数据建立双时序预测模型，并分别预测出未来时刻的用户请求和集群负载；

研究内容二，将用户预测请求和集群预测负载作为本研究内容中全局任务分配的输入，通过用户请求、服务器负载和服务器性能之间的作用和响应模型计算出服务器实时自响应负载，然后根据集群服务器自响应实时负载序列，通过加权最小负载分配策略为用户请求选择目标服务器[38]，从而确定用户请求分配方案；

研究内容三，在局部动态负载调度方面，本研究内容在预测自响应的全局任务分配模型的基础上，建立基于接受者主动的服务器自索取动态任务调度方案，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载。其中研究内容二基于预测自响应的全局任务分配和本研究内容为相互协同关系，分别处理新用户请求全局分配和局部相邻服务器之间的任务调度重分配关系。

最后将三项研究内容进行整合，集成出一套基于双时序流量预测的自响应动态负载均衡集

群系统。

1.5 论文组织结构

本文主要解决了服务器集群系统中基于负载预测实现高可用集群负载均衡和高效资源管理的问题，对基于用户请求流量和服务器负载流量的双时序流量预测技术、基于预测自响应的全局任务分配技术以及基于集群服务器自索取的局部动态负载调度技术进行了深入研究。论文总体组织结构如图 1.1 所示，总共包括六个章节，每个章节的具体内容如下：

第一章，绪论。首先阐述了本文的研究背景和意义，包括移动互联网技术的发展、终端设备的快速普及以及集群应用的。然后介绍了集群和集群技术、负载均衡技术和基于负载预测的集群技术等的发展及其研究现状，总结分析现有算法的优势和不足，最后对本文的研究内容和组织架构进行概括说明。

第二章，基于双时序流量预测的自响应动态负载均衡集群系统的总体设计。本章首先分析了服务器集群综合负载均衡系统的功能需求和性能需求，然后根据用户请求流量、集群负载流量预测的模型特点和全局与局部任务调度算法的实现方式进行系统结构设计和流程设计，并详细讲解了系统实现所采用的关键算法。

第三章，基于用户请求流量和服务器负载流量的双时序流量预测模型。针对用户请求流量和集群负载流量的数据特点，兼顾用户请求流量预测与集群负载预测，提高用户请求和集群负载的预测准确度，本章提出了一种基于加权长短时特征融合的双时序流量预测模型，该模型第一部分对用户请求流量和集群负载流量进行时序特征提取前的预处理工作；第二部分利用基于注意力机制的加权长短时特征融合技术对时序数据进行短时与长时特征提取、长短时特征融合以及向量加权等处理，充分挖掘时序数据的长短时特征，实现高准确度的时序流量短期预测和长期预测。

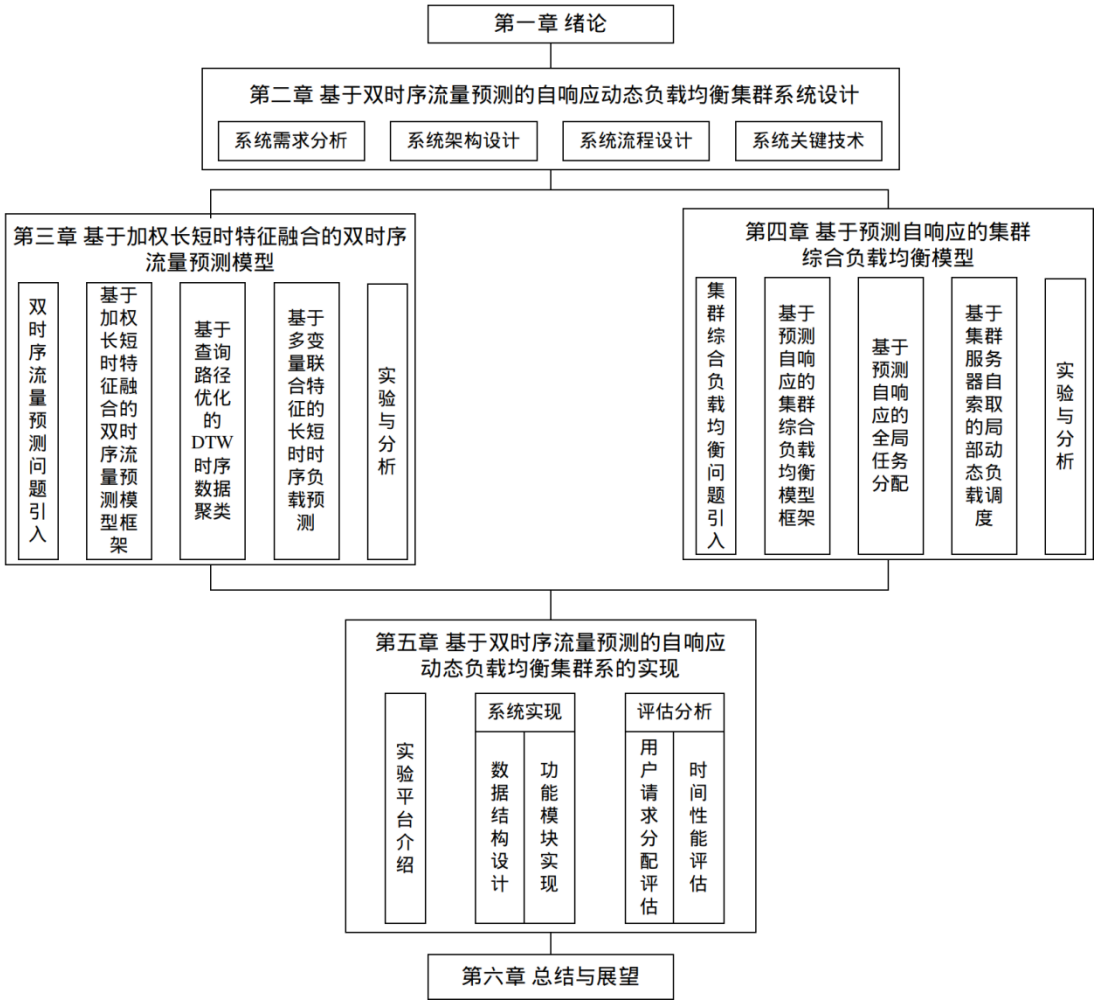


图 1.1 论文组织结构图

第四章，基于预测自响应的集群综合负载均衡模型。针对现有集群负载均衡策略存在实时性与准确性不足，以及集群负载均衡决策在局部负载调度方面存在的局部失衡等问题，本文从全局任务分配和局部动态负载调度两个层面，充分分析集群全局任务分配和局部负载调度机制并结合第三章提出的用户请求和集群负载预测技术，提出了基于预测自响应的集群综合负载均衡模型。从全局层面讲，该模型借助基于预测自响应的全局任务分配方法，充分利用第三章模型对用户请求和集群负载的准确预测，科学、准确挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立合理的全局任务分配模型。从局部层面讲，本文利用基于集群服务器自索取的局部动态负载调度方法，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载，实现了集群局部负载均衡。

第五章，基于双时序流量预测的自响应动态负载均衡集群系统的实现。首先介绍了系统实现的软硬件开发环境，并在文中详细地介绍了自响应动态负载均衡集群系统所需的数据结构和功能模块，完成了自响应动态负载均衡集群系统各个模块的代码编写，最后从用户请求分配的准确性和用户请求响应时延两方面对系统性能进行了相应的测试工作。

第六章，总结与展望。对本文已有研究工作进行总结归纳，分析了当前工作的不足之处，并对未来的研究方向进行展望。

第二章 基于双时序流量预测的自响应动态负载均衡集群系统设计

2.1 基于双时序流量预测的自响应动态负载均衡集群系统需求分析

在服务器集群负载均衡和资源管理过程中，由于用户服务的高可用和高可靠性以及集群系统的复杂性，如何统筹分析用户请求和集群负载信息，利用时序流量预测技术对用户请求和集群负载进行高效、准确的预测，保证集群系统全局任务分配和局部负载调度相统一，是检验集群系统高可用、高可靠性的重要评价标准[39]。为保证集群系统实现负载均衡与资源管理，本文提出了双时序流量预测模型，对用户请求流量和集群负载流量进行流量预测，借助预测自响应的全局任务分配方法和集群服务器自索取的局部动态负载调度方法保证集群系统全局任务分配和局部负载调度相统一。本章首先分析了系统在功能和性能方面需要实现的目标，阐明本系统的实际意义及价值;然后对系统的物理结构和逻辑结构进行设计,介绍了各个功能模块的作用，规划了系统实现流程；最后对系统中所用到的流量预测算法以及任务分配与负载调度算法进行了阐述说明。

2.1.1 系统功能需求分析

本系统从实现集群负载均衡与资源管理的功能角度可以划分为两个大的模块，分别为基于用户请求流量和集群工作负载的时序流量预测模块以及基于预测的全局任务分配和局部动态负载调度模块。由于用户请求的高并发性与集群系统的复杂性，为实现时序流量预测，需要相应的数据采集、存储与预处理单元为流量预测提供的数据准备工作；与此同时，还需要模型设计与训练、全局任务分配、局部动态负载调度等功能单元。为保证集群系统实现负载均衡与资源管理，本文设计的基于双时序流量预测的自响应动态负载均衡集群系统将从以下几个角度进行功能需求分析。

（1）数据采集、统计与存储

数据是时序流量预测模型的基础，因此，数据采集、统计与存储单元是必不可少的。

一个完整的服务器集群系统，其数据来源主要有用户请求数据和集群服务器节点的实时负载数据。从数据特征来看，用户请求数据和集群负载数据均为时序数据，是一种时间强相关的数据类型。

在数据统计与存储单元中，首先，系统需要统计用户请求和集群负载信息。一方面，系统需要实时记录与统计来自客户端的用户请求，准确记录用户请求对应的集群资源消耗量，例如CPU、内存、磁盘、网络 IO 等资源消耗情况；另一方面，系统需要实时记录和统计集群中各服务器节点的负载情况，同样包括服务器节点的 CPU、内存、磁盘以及网络等资源信息。其次，系统需要对上述用户请求和集群负载等数据进行合理存储。另外，为保证数据的安全性，系统

需要设置有效的数据备份机制。

（2）数据预处理

由于用户请求波动和集群中服务器节点故障，可能会存在部分数据丢失、记录时间不匹配、重复记录等问题，且进行数据统计与存储的服务器节点可能发生某些技术错误[40]，因此需要对冗余数据进行去重、剔除错误数据，填充缺失数据，经过处理后的数据才能用于用户请求与集群负载预测。

（3）双时序流量预测

双时序流量预测分为用户请求预测与集群负载预测，是本系统的核心功能之一。

本系统需要充分挖掘用户业务请求流量和集群服务器工作负载流量的时序特性，并分别对两种时序流量数据建立有效的流量预测模型，以对用户请求和集群负载进行准确预测，为集群全局任务分配和局部动态负载均衡调度提供可靠依据。

（4）集群综合负载均衡

集群综合负载均衡是本系统的另一核心功能。

一方面，在双时序流量预测的基础上，挖掘实时用户业务请求、实时服务器工作负载与预测用户请求、预测服务器工作负载以及服务器性能之间的相互作用与响应关系，并建立合适的全局任务分配模型，实现基于预测自响应的全局任务分配。

另一方面，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载，减轻负载均衡器压力，降低集群通信开销，减少服务器集群整体资源消耗，实现基于服务器自索取的局部动态负载调度。

2.1.2 系统性能需求分析

为了保证负载均衡和资源管理系统的实时性、准确性与低消耗性，本文将从以下四个方面进行系统性能分析：

（1）实时性

解决传统软件方法的负载均衡算法无法实时获取集群服务器工作负载导致负载均衡滞后效果明显，但频繁对服务器进行负载采样以获取实时负载会导致增加服务器压力这一矛盾。本文充分利用用户请求与集群负载预测得到的用户请求与集群负载信息，制定合理的全局任务分配和局部动态负载调度策略，以提高集群负载均衡调度的实时性。

（2）准确性

解决基于流量预测方法的负载均衡算法只针对用户求情或服务器负载中的某一流量做流量预测且未考虑用户请求和服务器负载之间的相互作用这一缺陷。本文在双时序流量预测的基础上，挖掘实时用户业务请求、实时服务器工作负载与预测用户请求、预测服务器工作负载以及

服务器性能之间的相互作用与响应关系，并建立合适的全局任务分配模型，实现基于预测自响应的全局任务分配，以提高集群负载均衡调度和资源管理的准确性。

（3）低消耗性

解决传统方法主要依赖负载均衡调度器实现集群服务器进行任务调度，导致用户请求响应慢、服务器通信开销大这一弊端，提高集群的用户响应速度，降低服务器集群的资源消耗。

（4）高可用性

高可用性是衡量集群系统实用价值的一个关键指标，良好的稳定性与高可用性可以为用户提供可靠的使用体验[41]，因此，提高基于双时序流量预测的自响应动态负载均衡集群系统的稳定性和高可用性是系统设计中需要考虑的重要一环。当发生用户请求流量波动、集群节点故障等异常情况时，集群系统仍然能够为用户提供连续、高质量服务，保证集群业务处理稳定，集群运行正常。

2.2 基于双时序流量预测的自响应动态负载均衡集群系统总体架构设计

2.2.1 基于双时序流量预测的自响应动态负载均衡集群系统架构设计

图2.1展示了基于双时序流量预测的自响应动态负载均衡集群系统架构，一共分为数据层、数据处理层、时序流量预测层、集群综合负载均衡层以及显示控制层等五个部分。

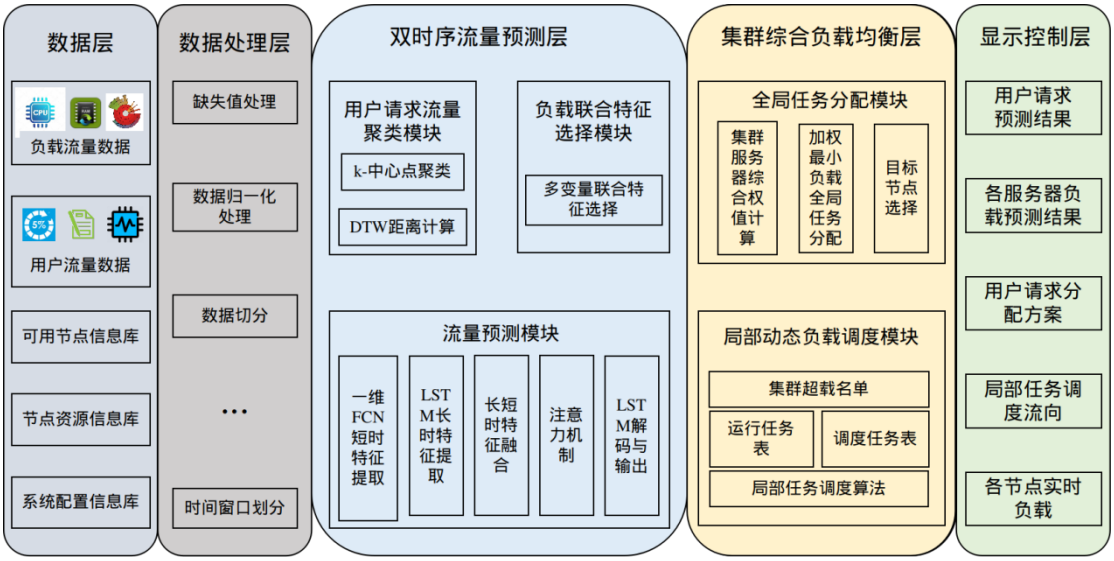


图 2.1 集群系统架构图

数据层主要负责集群系统的数据和资源管理工作，为系统提供数据和资源支持。该层由用户请求数据库、集群负载数据库、可用节点信息库、节点资源信息库以及系统配置信息库组成。用户请求数据库包含历史时间内访问集群系统的用户请求时序数据集，包括数据记录时间、任务编号、服务器节点编号、资源消耗率、响应时间等信息；集群负载数据库包含历史时间内各

个服务器节点的负载时序数据集，包括服务器节点编号、数据记录时间、资源消耗率等信息；可用节点信息库存储集群当前可用服务器节点的各项基本信息，包括节点编号、负载评分等信息；节点资源信息库存储集群中各服务器节点的资源配置信息，包括节点编号、资源利用率等信息；系统配置信息库存储集群系统的基础配置信息，包括集群架构、节点数量、网络拓扑、通信速率等信息。

数据处理层主要负责对数据层接收到的数据的处理工作，将其转换成时序流量预测层可直接使用的数据。该层包括用户请求和集群负载数据的预处理模块、数据切分模块和多变量联合特征选择模块。其中预处理模块将收集到的数据进行归纳整理，包括数据去重、异常值检查与删除、数据填充操作；数据切分模块对时序流量数据在时间维度上进行切分，将时序数据切分成一段历史训练窗口和未来的预测窗口，对于预测窗口中的每一条样本，基于训练窗口中的历史信息构建特征，转化为一个监督学习预测问题进行求解；多变量联合特征选择模块通过计算不同资源变量特征之间的相关性，为目的变量特征选择多个相关变量特征，将单一变量时序预测问题转化为多变量时序预测问题。

时序流量预测层是基于双时序流量预测的自响应动态负载均衡集群系统的核心层之一，负责用户请求和集群负载的预测任务。该层可分为两个子层，第一子层为用户请求数据和集群负载数据定向处理层，该子层对用户请求流量和集群负载流量进行特征提取前的定向处理，确定待预测的用户请求序列所属的用户请求类型以及确定用于流量预测的源特征和目的特征。第二子层为流量预测层，该子层负责对两类时序流量进行预测，包含基于一维全卷积的短时特征提取模块、基于 LSTM 的长时特征提取模块、加权长短时特征融合模块和解码模块。其中，短时特征提取模块在对时序数据进行 LSTM 编码之前，先使用一维全卷积神经网络对原时序数据进行一维全卷积操作，得到短时特征向量；长时特征提取模块将经一维全卷积短时特征提取后的短时特征数据输入 LSTM 进行长时特征提取，得到时序序列的长时特征向量；加权长短时特征融合模块先将经一维卷积后的短时特征向量与经 LSTM 长时特征提取模块得到的长时特征向量进行拼接融合，得到长短时融合特征向量，然后借助注意力机制对每个预测步的长短时融合特征向量进行注意力加权处理，得到每个预测步的加权长短时融合特征向量；解码模块依次读取加权长短时融合特征、更新其神经元状态和隐藏状态，输出当前时刻的预测值。

集群综合负载均衡层是集群系统的另一核心层，负责制定集群全局任务分配方案和协调局部相邻服务器节点之间的任务分配关系。该层可分为两个子层，第一子层为全局任务分配层，该子层负责建立合适的全局任务分配模型，为集群制定任务分配方案；第二子层为局部动态负载调度层，该子层基于服务器自索取机制，实现集群局部动态负载调度。全局任务分配子层包括集群服务器综合权值计算模块、加权最小负载全局任务分配模块和目标节点选择模块；局部动态负载调度子层包括服务器负载上下限比较模块、集群超载任务管理模块和集群转移任务管

理模块。

显示控制层用于实现集群的显示功能，包含结果用户请求列表模块、服务器节点列表模块以及用户请求分配和局部负载调度模块。用户请求列表模块用于显示集群系统当前等待处理和分配的用户请求列表；服务器节点列表模块用于显示集群后端当前服务器节点的负载状态；用户请求分配和局部负载调度模块负责为用户请求分配后端服务器节点，以及后端服务器节点之间的局部负载调度。

2.2.2 基于双时序流量预测的自响应动态负载均衡集群系统逻辑结构设计

图 2.2 展示了基于双时序流量预测的自响应动态负载均衡集群系统的逻辑结构，核心功能模块为时序流量预测模块和集群综合负载均衡模块，其中时序流量预测模块又分为用户请求预测模块和集群负载预测模块；集群综合负载均衡模块又分为全局任务分配模块和局部负载调度模块。除此之外，还有数据库、数据输入处理模块以及结果显示模块。

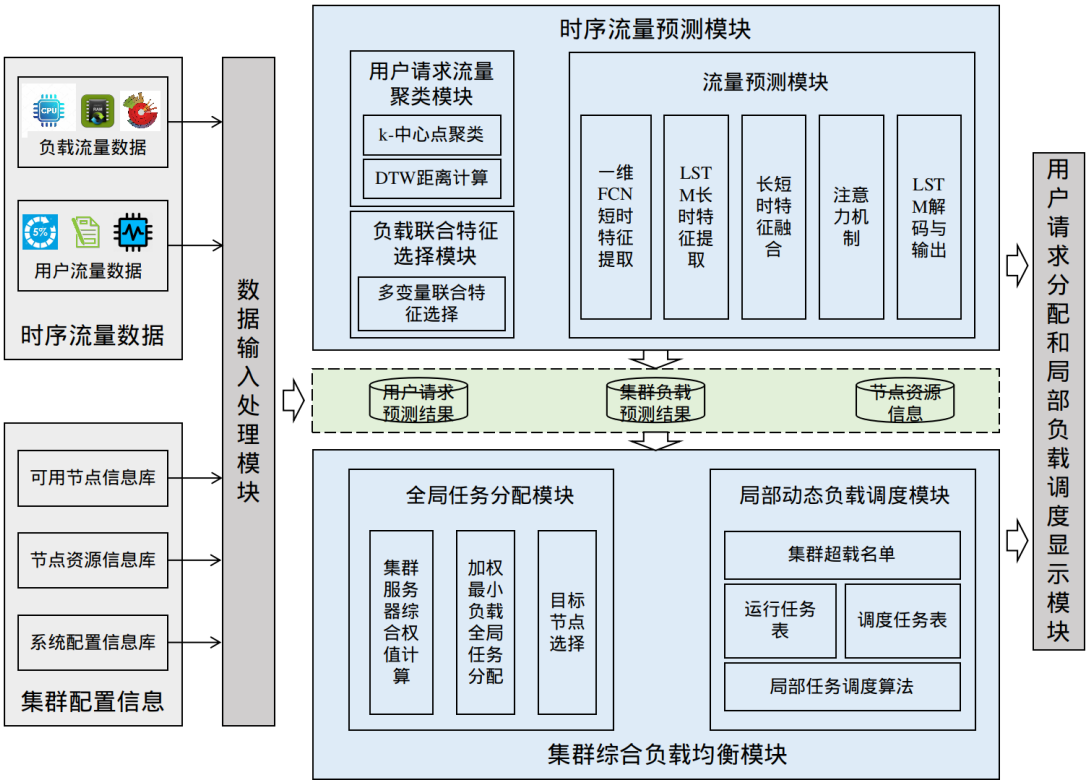


图 2.2 集群系统逻辑结构图

时序流量预测模块是基于双时序流量预测的自响应动态负载均衡集群系统的核心功能模块之一，具体又可划分为两个子模块，分别为用户请求数据和集群负载数据定向处理模块和流量预测模块。用户请求数据和集群负载数据定向处理模块对用户请求流量和集群负载流量进行特征提取前的定向处理，确定待预测的用户请求序列所属的用户请求类型以及确定用于流量预测的源特征和目的特征。流量预测模块负责对两种时序数据进行预测，包含联合特征选择、长短

时特征提取、负载预测等部分。具体而言，先使用一维全卷积神经网络对原时序数据进行一维全卷积操作，得到短时特征向量；然后，长时特征提取模块借助 LSTM 网络对得到的短时特征向量进行长时特征提取，得到长时特征向量；然后加权长短时特征融合模块结合注意力机制，将短时特征向量与长时特征向量进行拼接融合并做注意力加权处理加权长短时融合特征向量；最后，解码模块依次读取加权长短时融合特征、更新其神经元状态和隐藏状态，得到未来时刻的时序预测值。

基于双时序流量预测的自响应动态负载均衡集群系统的另一核心功能模块是集群综合负载均衡模块，该功能模块包含全局任务分配和局部动态负载调度两个子模块，分别负责制定集群任务分配方案和协调局部相邻服务器节点之间的任务分配关系。全局任务分配子模块包括集群服务器综合权值计算模块、加权最小负载全局任务分配模块和目标节点选择模块。局部动态负载调度子模块包括服务器负载上下限比较模块、集群超载任务管理模块和集群转移任务管理模块。

数据输入处理模块和结果显示模块共同组成了显示控制模块，主要负责对用户请求的接收、时序流量预测和任务分配结果的展示。数据输入处理模块需要实现用户和集群系统的交互，并实时记录并存储用户请求的资源信息。用户请求和集群负载预测完成后，时序流量预测结果显示模块显示未来一段时间内用户请求和集群负载的资源消耗情况，供用户查看中间预测结果；全局任务分配和局部负载调度完成后，结果显示模块会显示集群系统为特定用户请求制定的任务分配方案，以及局部负载调度的结果信息。

该集群系统的用户请求预测模块和集群负载预测模块中的模型训练和流量预测均为离线部分，模型训练会随着历史数据的累积周期性更新，时序流量预测可以根据用户请求的频次进行多次预测，并通过显示控制层对将评估与诊断结果进行可视化展示。

2.3 基于双时序流量预测的自响应动态负载均衡集群系统总体流程设计

本文设计的基于双时序流量预测的自响应动态负载均衡集群系统的流程分为数据预处理阶段、双时序流量预测阶段和集群综合负载均衡阶段。数据预处理阶段将采集到的数据进行缺失值处理、归一化处理、数据切分、多变量联合特征处理等操作。时序流量预测阶段进行用户请求和集群负载两种时序流量的预测任务，分析用户请求和集群负载两种时序数据的特征，预测未来一段时间内两种流量的变化趋势，为任务分配与负载调度阶段提供方案制定依据。集群综合负载均衡阶段为来自客户端的用户请求制定任务分配方案，同时负责协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载，减少服务器集群整体资源消耗。基于双时序流量预测的自响应动态负载均衡集群系统的流程如图 2.3 所示，主要分为数据预处理、双时序流量预测和多任务分配与动态负载调度三个部分，具体流程如下。

（1）数据预处理阶段

数据预处理阶段首先将集群系统中采集和存储的数据进行初步处理，去除重复记录和记录不全的数据；其次进行数据清洗工作，对数据进行缺失值和异常值检查，使用均值填充法对缺失数据进行补全，删除因集群故障导致的异常值；最后对时序流量数据在时间维度上进行切分，将时序数据切分成一段历史训练窗口和未来的预测窗口[42]。

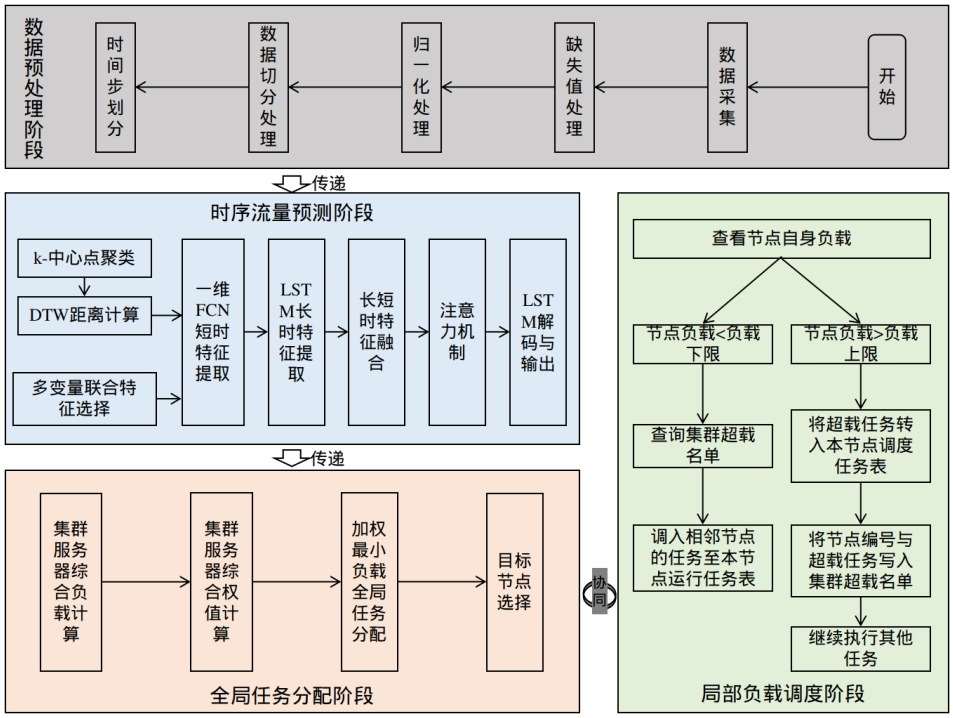


图 2.3 集群系统流程图

（2）双时序流量预测阶段

该阶段分为用户请求流量预测和集群负载流量预测两个任务，两个任务是同时进行的。

用户请求流量任务负责根据历史用户请求时序流量信息对未来一段时间内的用户请求流量进行预测，包含用户请求流量聚类和请求预测两个阶段。集群负载流量预测阶段负责对服务器节点未来一段时间内的负载信息进行预测，分为基于一维全卷积的短时特征提取、基于 LSTM 的长时特征提取、加权长短时特征融合和解码预测等阶段。

（3）集群综合负载均衡阶段

该阶段分为全局任务分配和局部动态负载调度两个任务，其中任务分配任务为触发性任务，其基于用户请求，有新用户请求时运行该任务；动态负载调度为实时任务，其在集群系统的整个生命周期运行。

全局任务分配任务包括集群服务器综合权值计算模块、加权最小负载全局任务分配模块和目标节点选择模块。局部动态负载调度任务包括服务器负载上下限比较模块、集群超载任务管

理模块和集群转移任务管理模块。

（4）结果显示

双时序流量预测阶段完成以后用户请求和集群负载的预测结果通过结果显示模块进行结果的显示；全局任务分配与局部动态负载调度的用户请求分配方案与局部动态负载调度的实时结果也会在结果显示模块进行显示。

2.4 基于双时序流量预测的自响应动态负载均衡集群系统关键算法

本文提出的基于双时序流量预测的自响应动态负载均衡集群系统基于时序流量预测技术，从全局任务分配和局部动态负载调度两个方面着手，实现集群系统的负载均衡和高效资源管理。

在双时序流量预测方面，本文对用户请求和集群负载两种时序流量建立时序预测模型。在进行时序流量预测之前，需要对用户请求数据和集群负载数据进行定向处理，针对用户请求数据的分类处理，本文提出了一个基于查询路径优化的 DTW 时序数据聚类模型。该模型在传统 DTW 算法的基础上，对路径查询方法进行优化，减少路径探索过程中不必要的检索，减少计算量，提高路径查询效率[43]。针对集群负载流量，本文提出了一个基于加权长短时特征融合的双时序流量预测模型。针对传统时序预测方法中短期预测方面存在的不足，同时增强模型长期预测能力，该模型借助注意力加权长短时特征融合方法，以兼顾长期预测和短期预测。具体而言，为解决现有方法在短期预测方面存在的不足，模型借助一维全卷积神经网络（FCN），在进行长期特征提取之前，对数据进行一维全卷积（1D FCN）处理，强化时序数据的短期依赖关系，得到短时特征向量。为增强模型长期预测能力，使用 LSTM 提取时序负载的长时特征，然后将短时特征与长时特征进行拼接融合，得到长短时融合特征向量；然后利用注意力机制，分别对每一时刻的长短时融合特征向量进行加权，得到对应时刻的加权长时融合特征向量。

在集群综合负载均衡方面，本文分别对全局任务分配和局部动态负载调度两类工作建立不同的处理机制。针对全局任务分配，本文提出了基于预测自响应的全局任务分配模型，该模型在双时序流量预测的基础上，挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立合理的全局任务分配模型。针对局部动态负载调度，本文提出了基于集群服务器自索取的局部动态负载调度模型，该模型借助基于集群服务器自索取的局部动态负载调度方法，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载。

2.5 本章小结

本章对基于双时序流量预测的自响应动态负载均衡集群系统进行了总体设计。首先，根据系统应用场景，对集群系统的功能需求和性能需求进行说明；然后，从系统结构设计和逻辑结构设计两个方面来介绍系统总体架构设计，详细解释了每个核心模块的功能；之后，展示了集

群系统实现的总体流程，对流程中的每一步都进行了详细说明；最后，介绍了系统中用于双时序流量预测和集群综合负载均衡的关键算法。

第三章 基于加权长短时特征融合的双时序流量预测模型

3.1 双时序流量预测问题描述

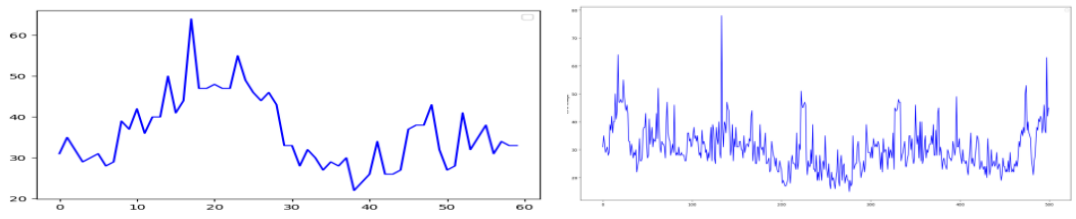
随着互联网终端的快速、大量普及，为满足日益普遍的高并发应用场景，我们对云计算、电网等服务器集群技术提出了更高的要求[44]。针对集群系统常用的适用场景，例如 B/S、C/S 架构场景，提高用户请求和资源分配准确度，保证任务分配的实时性，降低集群整体资源消耗，是实现高效集群系统的重要参考指标。实现高效的集群负载均衡和资源分配，可以从客户端用户请求和服务端集群负载两个方面着手。

就客户端而言，在服务器集群、云计算、电网等很多应用场景中，我们借助时序流量预测技术，对用户请求流量或网络流量进行预测，并以此为依据来调整和管理集群资源，以辅助运营商进行精细化运营，提高用户请求和资源分配准确度，降低系统资源消耗[45]。

在多数服务器集群的应用场景中，用户请求时序流量或网络时序流量呈现两个显著特点：1）存在大量的用户请求时序流量，某些会呈现一定的模式，但某些可能不会呈现周期性或表现出一定的趋势[46]；2）很多用户请求的持续的时间较短，积累的历史数据很少。

就服务端而言，在被动响应式集群系统中，资源管理是纯反映式的，系统根据负载变化对集群进行资源分配和调整。配置决策和资源调整的滞后性，容易导致资源配置不足或过度配置问题[47]。为提高系统资源配置的主动性，实现自适应资源分配和管理，我们需要对集群负载进行预测。通过提前预测集群未来一段时间内的负载变化，预先制定资源分配和调整方案，降低配置决策和资源调整滞后性的影响，同时提高集群系统的动态性，进而提高系统的资源利用率。

如图 3.1 所示，通过对现有公开集群负载数据集的分析，我们发现，集群负载时序流量呈现如下两个特点：1）短时间跨度内，负载变化呈现无周期性和波动性；2）长时间跨度内，负载变化呈现周期性特点，且不同时间跨度呈现不同的周期模式[48]。因此，如何提高负载预测的准确度，同时兼顾短期预测和长期预测，是负载预测中需要解决的关键问题。



（a）短时间跨度内集群负载变化　　（b）长时间跨度内集群负载变化

图 3.1 不同时间跨度内集群负载变化

综合来看，我们可以发现用户请求流量和集群负载流量共有的特性：二者同为服务器负载时序数据，数据特征均为 CPU、内存、磁盘等机器资源的利用率；二者在周期性模式上均呈现

短时和长时周期分化的特点，且长短时周期特性不同。

目前，对于时序数据的预测主要有三类方法，分别为基于传统线性回归模型的负载预测方法，基于传统机器学习的负载预测方法和基于深度学习的负载预测方法[49]。第一类基于传统线性回归模型的负载预测方法主要有自回归（Autoregressive,AR）、滑动平均（Moving Average,MA）、自回归移动平均（Autoregressive Moving Average, ARMA）以及差分整合移动平均自回归（Autoregressive Integrated Moving Average, ARIMA）等模型。这些模型在复杂度低、线性关系较强的数据中可以实现较好的预测效果，因此此类方法存在对数据的限制较多，且需要人工调整模型参数等方面的不足。第二类基于传统机器学习的负载预测方法主要有马尔科夫模型、贝叶斯模型、支持向量回归（Support Vector Regression,SVR）模型，以及决策树和传统人工神经网络（Artificial Neural Networks, ANN）等模型。此类方法能够提取时序数据中的短期非线性特征，但无法捕获数据中的长期依赖关系，因此其在长期预测方面存在较大不足。近年来，随着深度学习在非线性特征处理方面表现出的显著优势，基于深度学习的预测方法在时序数据预测方面取得了较好的发展。Fargana[50]等人利用 LSTM 的长期依赖挖掘能力，在此基础上使用 Encoder-Decoder 模型架构对时序数据进行特征提取和分解，提高了集群负载预测准确率；MinXian Xu[51]等人提出一种多特征负载预测模型，利用集群负载数据中不同特征之间的相互作用关系，使用多维特征进行负载预测。但是由于不同特征之间的相关性不均等且作用效果有限，不同目标特征的预测效果存在较大差异；为了解决集群长期负载存在的模式转换和振幅波动问题，更好挖掘负载数据的不同周期模式，Jiaming Huang[52]等人提出一种多尺度注意力机制，根据不同的周期模式设置不同的注意力权重，提高了集群负载的长期预测能力。但是，该模型在短期预测方面存在较大不足；为提高模型的短期预测能力，Wenyan Chen[53]等人提出将 TCN 时序神经网络用于集群负载预测，同时利用多维特征进行目标特征预测。该方法在短期预测方面具备较好表现，但是长期预测能力存在很大不足。

现有集群负载预测模型多为基于深度学习的预测模型，为提高模型的预测准确度，现有模型充分利用多特征、注意力机制等方法，并且在短期预测或长期预测的某一方面取得了较好效果。显然，现有模型未能兼顾负载短期预测和长期预测效果，使模型同时具备较好的短期和长期预测能力。

因此，为解决上述问题，兼顾用户请求流量预测与集群负载预测，提高用户请求和集群负载的预测准确度，本章提出了基于加权长短时特征融合的双时序流量预测模型。具体而言，该模型可分为两大部分，第一部分负责对用户请求流量和集群负载流量进行时序特征提取前的预处理工作，第二部分负责对两类时序数据进行加权长短时特征融合处理。其中，第一部分别对两类时序数据进行基于查询路径优化的 DTW 时序数据聚类分类和多变量联合特征选择处理[54]；第二部分分别对两类时序数据进行长短时特征提取与加权融合，并完成时序预测。首先通过一

维全卷积短时特征提取模块，利用一维全卷积神经网络（1D FCN）对输入时序负载进行短时特征提取，得到短时特征向量；接着将其输入 LSTM 长时特征提取模块，进行长时特征提取得到长时特征向量；然后借助注意力加权长短时特征融合模块，将短时特征向量与长时特征向量进行拼接融合，得到长短时融合特征向量；再利用注意力机制，分别对每一时刻的长短时融合特征向量进行加权，得到对应时刻的加权长时融合特征向量；最后利用 LSTM 解码模块，得到负载预测结果。

本章的主要研究内容如下：

1）针对集群中用户请求流量呈现出的特点，本章提出基于查询路径优化的 DTW 时序数据聚类分类方法，利用聚类将用户请求流量序列划分为不同的类别，弥补某些用户请求历史数据少，可利用数据不足的问题。同时对 DTW 进行查询路径优化，解决用户请求时序过长导致 DTW 聚类耗时长的问题。

2）针对用户请求和集群负载时序数据包含的特征繁多，且特征之间呈现一定关联关系这一数据特点，本章提出一种多变量联合特征选择机制，通过多变量联合特征选择，充分挖掘和利用不同特征之间的相关关系，同时结合注意力机制，提高模型的负载预测能力。

3）针对复杂场景下短期时序预测方面存在的不足，同时增强模型长期预测能力，本章提出一种基于加权长短时特征融合的双时序流量预测方法，在兼顾长期预测和短期预测的同时，能够同时适用用户请求和集群负载两种时序数据的预测工作。针对现有方法在短期预测方面存在的不足，本文借助一维全卷积神经网络（1D FCN），在进行长期特征提取之前，对数据进行一维全卷积处理，强化时序数据的短期依赖关系，得到短时特征向量。为增强模型长期预测能力，使用 LSTM 提取时序数据的长时特征，然后将短时特征与长时特征进行拼接融合，得到长短时融合特征向量；然后利用注意力机制，分别对每一时刻的长短时融合特征向量进行加权，得到对应时刻的加权长时融合特征向量。

3.2 基于加权长短时特征融合的双时序流量预测模型框架

本章深入研究了基于查询路径优化的DTW时序数据聚类技术和多变量联合特征选择技术，进而提出一种基于加权长短时特征融合的双时序流量预测模型。该模型首先分别利用基于查询路径优化的 DTW 时序数据聚类技术和多变量联合特征选择技术对用户请求流量和集群负载流量进行特征提取前的定向处理，确定出用户请求序列所属的用户请求类型以及确定源特征和目的特征。然后利用基于加权长短时特征融合的双时序流量预测技术分别对两类时序流量进行时序特征提取和分类。具体而言，首先通过一维全卷积短时特征提取模块，得到短时特征向量；接经过LSTM长时特征提取模块得到长时特征向量；然后借助注意力加权长短时特征融合模块，得到长短时融合特征向量；再利用注意力机制[55]，分别对每一时刻的长短时融合特征向量进行

加权，得到对应时刻的加权长时融合特征向量；最后利用 LSTM 解码模块，得到负载预测结果。

本章模型框架主要包括四个模块：用户请求流量聚类分类模块、集群负载多变量联合特征选择模块、长短时特征提取与加权融合处理模块以及 LSTM 解码预测模块，如图 3.2 所示：

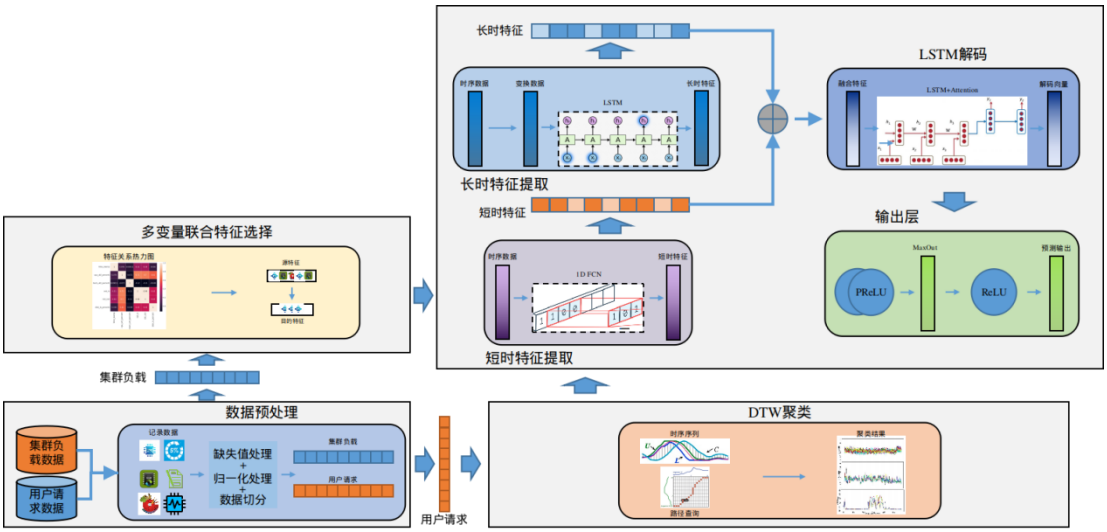


图 3.2 基于加权长短时特征融合的双时序流量预测模型框架

（1）在用户请求流量聚类分类的过程中，一方面，存在大量的用户请求时序流量，某些会呈现一定的模式，但某些可能不会呈现周期性或表现出一定的趋势；另一方面，很多用户请求的持续的时间较短，积累的历史数据很少。因此，本文提出在对用户请求时序数据进行特征提取之前，利用 DTW 聚类方法对用户请求时序数据进行聚类分类处理，按照其时序特点划分为不同的类别，由此解决部门用户请求的历史数据较少缺乏足够历史特征的问题。同时，针对较长的用户请求时序数据进行 DTW 聚类耗时过长的问题，本文对传统 DTW 聚类算法进行了查询路径优化的改进，提高对较长用户请求时序数据的聚类效率。

（2）在集群负载多变量联合特征选择过程中，为充分利用不同资源变量特征之间的相互作用关系，通盘考虑，本文对原时序负载数据进行多变量联合特征处理。通过计算不同资源变量特征之间的相关性，为目的变量特征选择多个相关变量特征，将单一变量时序预测问题转化为多变量时序预测问题。

（3）长短时特征提取与加权融合处理过程中，为了短期负载预测方面存在的不足，同时增强模型长期预测能力，使模型兼顾长期预测和短期预测，本文对时序数据分别进行短时和长时特征提取，并借助注意力机制，实现对时序数据的长短时特征提取与加权融合处理。首先通过一维全卷积短时特征提取模块，利用一维全卷积神经网络（1D FCN）对输入时序负载进行短时特征提取，得到短时特征向量；接着将其输入 LSTM 长时特征提取模块，进行长时特征提取得到长时特征向量；然后借助注意力加权长短时特征融合模块，将短时特征向量与长时特征向量进行拼接融合，得到长短时融合特征向量；再利用注意力机制，分别对每一时刻的长短时融合

特征向量进行加权，得到对应时刻的加权长时融合特征向量。

（4）最后，LSTM 解码预测负责对加权长短时融合特征向量进行解码和预测。LSTM 解码器将经特征融合处理得到的加权长短时融合特征向量依次输入解码器，依次得到每一时刻的解码值。加权长短时特征融合发生在解码的每一个时刻。对解码器而言，其每个时刻的输入由经注意力机制处理后的加权长时特征向量和经一维卷积得到的短时特征向量拼接融合得到。在解码过程中，解码器的神经元依次读取加权长短时融合特征、更新其神经元状态和隐藏状态，输出当前时刻的解码值。每一时刻的解码输出会作为解码器下一时刻的输入。然后通过预测输出层的三层激活函数对解码向量进行预测结果输出。

3.3 基于加权长短时特征融合的双时序流量预测模型实现

3.3.1 双时序数据预处理

1）基于查询路径优化的 DTW 用户请求时序数据聚类

（1）传统 DTW 算法

动态时序规整（Dynamic Time Warping）算法是一种比较两个不完全同步的序列（通常是时间序列）的有效算法，该算法基于动态规划思想，计算两个时序序列的最优匹配。该算法在语音识别、数据挖掘等场景中得到广泛应用，是一种计算两个时序序列之间距离的有效方法[56]。该算法定义如下。

两时序序列 X 和 Y 长度分别为 m 、 n ，其定义如下：

$$X = x_1, x_2, , , , x_i, ..., x_n \tag{1}$$

$$Y = y_1, y_2, , , , y_i, ..., y_n \tag{2}$$

由时序 X 和 Y 可构成一个 $m*n$ 的矩阵，其中点 (i, j) 代表点 x_i 与 y_j 的之间的对齐度。

该算法通过寻找一条序列 X 和 Y 之间的最优规整路径 W 以最小化两时序序列之间的距离，W 为矩阵中的点的集合。其计算过程如下。

$$D_{\min}(i_k, j_k) = \min D_{\min}(i_{k-1}, j_{k-1}) + d(i_k, j_k \mid i_{k-1}, j_{k-1}) \tag{3}$$

其中， d 为欧氏距离， $d(i, j) = \| f_1(i) - f_2(i) \|$ 。

整体路径距离为：

$$D = \sum_k d(i_k, j_k) \tag{4}$$

该规整路径 W 的计算基于动态规划思想，为提高路径优化效率，该规整路径 W 存在如下约束条件。

第一，边界约束，该条件约束确保扭曲路径从两个信号的起点开始，并以其端点结束。

$$i_1 = 1, i_k = n \quad \text{and} \quad j_1 = 1, j_k = m$$

第二，单调性约束，该条件保留点的时间顺序（不返回时间）。

$$i_{t-1} \le i_t \quad \text{and} \quad j_{t-1} \le j_t$$

第三，连续性约束，该条件将路径过渡限制到相邻时间点（而不是时间跳跃）。

$$i_t - i_{t-1} \le 1 \quad \text{and} \quad j_t - j_{t-1} \le 1$$

总结来说，一条合理的规整路径 W 需满足以下约束条件。

水平移动：$(i, j) \to (i, j+1)$

垂直移动：$(i, j) \to (i+1, j)$

对角移动：$(i, j) \to (i+1, j+1)$

（2）基于路径查询优化的 DTW 算法

针对 DTW 规整路径优化问题，很多文献做了大量工作。为了减少路径探索过程中不必要的检索，文献[57]提出将查询路径W限制在斜率为 1/2 到 2 之间的平行四边形内。如图 3.3（a）所示，m 和 n 为两个不同的时序序列，OABC 为平行四边形。已知 OA 和 OC 的斜率分别为 2、1/2，因此平行四边形 OABC 四条边的函数关系和四个点的坐标可以确定如下。

$$y_{OA} = 2x \tag{5}$$

$$y_{OC} = 0.5x \tag{6}$$

$$y_{AB} = 0.5x + m - 0.5n \tag{7}$$

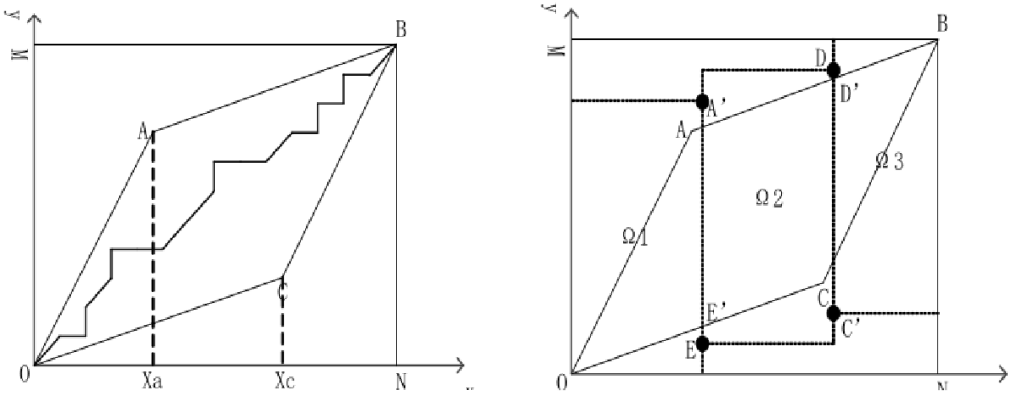
$$y_{CB} = 2x + m - 2n \tag{8}$$

其中，点 O 为 $(0,0)$，点 A 为 $(\frac{2m-n}{3}, \frac{2(2m-n)}{3})$，点 B 为 (n,m)，点 C 为 $(\frac{2(2n-m)}{3}, \frac{2(2n-m)}{3})$。

我们可以得出，X_a 与 X_c 为距离最近的两个整数点。因此，序列 m 和 n 的长度限制为：

$2m - n \le 3$；$2n - m \ge 2$

同时我们可以看到，距离矩阵 D 的计算量很大。当搜索路径被限制在平行四边形 OABC 时，不需要计算 OABC 外部晶格点的匹配距离。因此，计算量大大减少。



（a）传统 DTW 路径查询区域　　（b）三矩形 DTW 路径查询区域

图 3.3 DTW 路径查询区域优化

然而，目前仍存在一个问题，即如何判断两个时序序列限定的矩形 OMBN 内的点是否位于平行四边形 OABC。本文提出三矩形法以解决该问题。

三个矩形 Ω_1、Ω_2、Ω_3 如图 3.3（b）所示。矩形 Ω_1 由点 $(0,0)$ 与点 $A'(\left\lceil \frac{2m-n}{3} \right\rceil, \left\lceil \frac{2(2m-n)}{3} \right\rceil)$ 决定，其中 A' 为点 A 顶部最近的整数点；矩形 Ω_2 由点 E 与点 D 决定，其中点 $E(\left\lceil \frac{2m-n}{3} \right\rceil, \left\lceil \frac{2m-n}{6} \right\rceil)$ 为垂线 $A'E$ 与直线 OC 的交点 E' 底部最近的整数点，其中点 $D(\left\lceil \frac{2(2n-m)}{3} \right\rceil, \left\lceil \frac{4m+n}{6} \right\rceil)$ 为垂线 $C'D'$ 与直线 OC 的交点 D' 顶部最近的整数点；矩形 Ω_3 由点 D 与点 B 决定。

因此，在进行最优路径探索时，只需要考虑位于三个矩形 $\Omega_1+\Omega_2+\Omega_3$ 范围之内的点，而不需要考虑该范围之外的点。由此，最优路径探索问题转换为查找位于三个矩阵范围之内的点，该范围需满足如下条件：

$$y-2x\le 0 \tag{9}$$

$$y-0.5x\ge 0 \tag{10}$$

$$y-0.5x-m+0.5n\le 0 \tag{11}$$

$$y-2x-m+2n\ge 0 \tag{12}$$

通过对查找范围的精确限定，可以极大的减少最优路径探索过程中的计算量，提高路径查

询效率。两时序序列长度越长，效率提升越明显。

2）集群负载流量多变量联合特征选择

在集群运行场景中，负载数据是通过采集各服务器在不同时刻的负载得到的。由于云计算应用场景的不同，负载数据的时间间隔不等，但整体来看，相邻的负载数据在时间上是连续的。因此，集群负载数据为典型的时序数据。具体而言，在大多数集群运行场景中，每个服务器的负载主要通过 CPU、内存、磁盘、网络等四种资源的利用率来衡量。对于每个服务器，其每个时刻的负载都包含 CPU、内存、磁盘和网络等四种资源的利用率。本文将这四种资源的利用率作为影响负载预测效果的四个主要特征。同时，我们需要认识到，在集群运行场景中，不同资源的消耗情况呈现一定的相关性[58]。例如，当集群中内存消耗量增加时，CPU 的利用率往往也会提升。因此，挖掘不同特征之间的相关关系，有助于提高负载预测效果。

为了充分挖掘同一资源变量特征在不同时刻之间的影响以及同一时刻不同资源变量特征之间的相互作用关系，本文提出为负载时序数据添加滑动窗口以及使用多变量联合特征预测目标变量特征，以提高负载预测准确度。

（1）数据切分处理

在机器学习中，我们经常利用交叉验证，用一部分数据的训练特征预测另一部分数据。本文借鉴该交叉验证思想，对负载数据在时间维度上进行切分，将时序数据切分成一段历史训练窗口和未来的预测窗口。对于预测窗口中的每一条样本，基于训练窗口中的历史信息构建特征，转化为一个监督学习预测问题进行求解[59]。

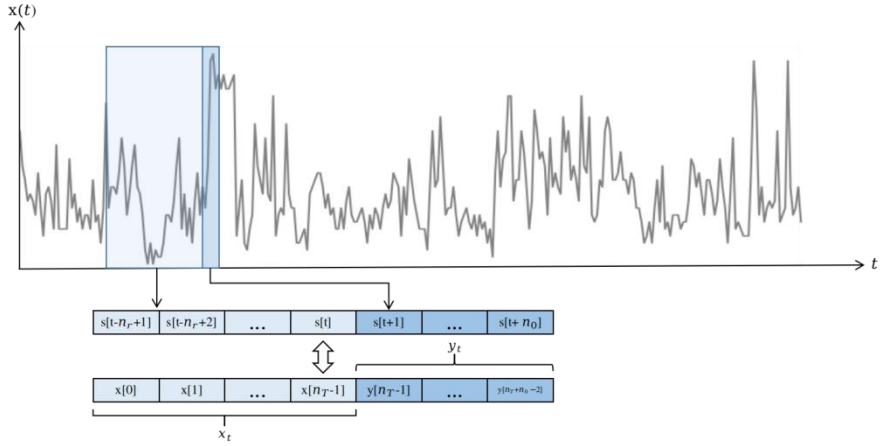


图 3.4 滑动窗口数据切分原理图

（2）多变量联合特征处理

在负载预测中，某一资源的时序变化波动往往受到其他资源变化的影响。例如，内存的使用往往与 CPU 相关，因此 CPU 的时序变化会受到内存的影响。反之，服务器在进行磁盘读取时，CPU 往往处于空闲状态，因此磁盘的时序变化对 CPU 的时序波动影响不大[60]。

为充分利用不同资源变量特征之间的相互作用关系，通盘考虑，本文对原时序负载数据进

行多变量联合特征处理。通过计算不同资源变量特征之间的相关性，为目的变量特征选择多个相关变量特征，将单一变量时序预测问题转化为多变量时序预测问题。

以 alibaba-cluster-trace-v2018 中的一段数据为例，表 3.1 为包含 CPU 和内存两种资源利用率的原数据。为利用内存使用率对 CPU 使用率时序变化的作用，我们将内存利用率也作为预测 CPU 利用率的训练特征。如表 3.1 所示，此时目标特征为 CPU 利用率，训练特征为 CPU 和内存利用率。

表 3.1 alibaba-cluster-trace-v2018 数据集：单一变量预测

时间	CPU 利用率	内存利用率
0	26	94
10	31	96
20	26	94
30	40	95
40	37	97
50	37	96
60	32	95
70	30	95

表 3.2 alibaba-cluster-trace-v2018 数据集：多变量预测

时间	变量 1	变量 2	变量 3	预测变量
0			26	94
10	26	94	31	96
20	31	96	26	94
30	26	94	40	95
40	40	95	37	97
50	37	97	37	96
60	37	96	32	95
70	32	95	30	95
80	30	95		

3.3.2 一维全卷积短时特征提取

近年来，以 RNN、LSTM、GRU 为代表的时序神经网络在时序问题上取得了很好的表现，包括语音文本识别、机器翻译、手写体识别、序列数据分析与预测等领域[61]。尤其是 LSTM 和 GRU 对 RNN 记忆范围的改进，使得时序神经网络可以将距离当前数据很远的历史信息利用起来，极大地提高了模型的长时预测能力。

然而，在短时预测方面，RNN 等时序神经网络存在一定不足。其原因在于，时序神经网络依赖各种门控机制，保存时序数据中的历史信息，其待预测的数据依赖于前面的历史信息。当时序网络中累积的历史信息不足时，其预测能力也会相应下降。

为进一步提高模型只利用少量历史信息便能完成短时负载预测的能力，同时提高模型的大规模并行处理能力，本文提出，在对负载数据进行 LSTM 编码之前，先使用一维全卷积神经网络（1D FCN）对原时序负载数据 $X=\left\{x_1^{'},x_2^{'},...x_n^{'}\right\}$ 进行一维全卷积操作，得到短时特征向量 $S=\left\{s_1,s_2,...s_n\right\}$。由于在全卷积神经网络中，信息计算不依赖于当前数据之前的历史信息，因此每个计算都是独立的。同时通过调节卷积核的大小，尽可能保留原时序数据的短期依赖关系。

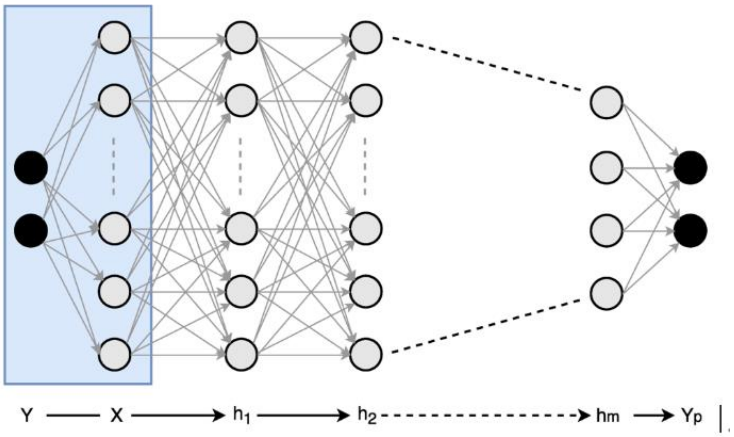


图 3.5 FCN 原理图

3.3.3 长时特征提取

1）LSTM 简介

RNN 在很多时序问题上都能取得良好的表现，例如语音文本识别、机器翻译、时序数据预测等问题。然而 RNN 在模型梯度传递过程中存在梯度消失问题，导致模型难以学习到远距离的依赖关系，进而导致其长时预测能力不足。另外，由于 RNN 在特征提取过程中需要保留每个时刻的信息，所以其在训练阶段需要很大的存储空间，且训练速度较慢。LSTM 继承了 RNN 能够对全局信息进行建模的优点，同时借助遗忘门，对历史数据中的重要信息进行保留，并传递下去，从而具备较好的长时预测能力[62]。同时，相较于 GRU，LSTM 模型的拟合和预测精度总体较高。基于以上原因，本文采用 RNN 的变体 LSTM 作为我们长时特征提取和特征解码的主网络。

LSTM 由多个循环单元组成，针对 RNN 在获取长时依赖方面存在的问题，LSTM 提出“门”这一机制控制历史信息的传输，主要有输入门、输出门和遗忘门。

LSTM 单元结构如下：

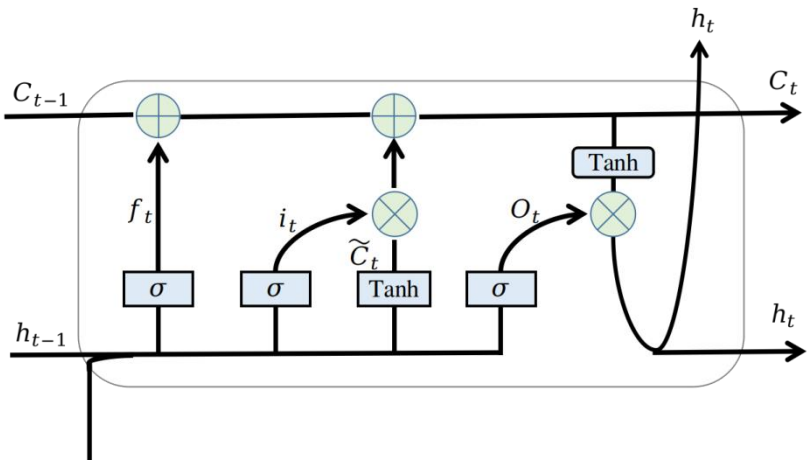


图 3.6 LSTM 神经单元

在图 3.6 中，每个单元的输入为C_{t-1}、h_{t-1}和x_t，其中C_{t-1}为前一时刻神经元的状态，h_{t-1}为前一时刻神经元的输出，x_t为当前时刻的输入。

输入门决定x_t中哪些新的输入可以存储在神经元中，其输入门控制信号如公式 13 所示，W_i为输入门的权重矩阵，b_i为偏置常数；遗忘门控制有多少上一时刻神经元的输出可以传递到当前时刻，其遗忘门控制信号如公式 14 所示，W_f为遗忘门的权重矩阵，b_f为偏置常数；输出门控制当前神经元状态的输出并将当前神经元状态转移到下一神经元，其输出门控制信号如公式 15 所示，W_o为输出门的权重矩阵，b_o为偏置常数。

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{13}$$

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{14}$$

$$\tilde{c}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{15}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{16}$$

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{17}$$

$$h_t = o_t \odot \tanh(c_t) \tag{18}$$

整体而言，当 LSTM 网络接收t时刻的输入x_t时，由公式 16 对本次输入进行激活处理，然后公式 17 通过输入门和遗忘门得到当前神经元的状态，最后公式 18 经输出门得到当前神经元的输出。

2）基于 LSTM 的长时特征提取

将经一维全卷积短时特征提取后的短时特征数据$S = \{s_1, s_2, ...s_n\}$输入 LSTM 进行长时特征提取。随着负载数据的输入，LSTM 神经元的隐藏层状态和神经元状态不断进行信息累积和更新，捕获负载数据中的长时依赖关系。最后，LSTM 编码器输出得到负载时序序列的长时特征向量$L = \{l_1, l_2, ...l_n\}$。

3.3.4 加权长短时特征融合

1）长短时特征融合

经LSTM长时特征提取模块得到的长时特征向量L能够很好的捕获时序负载中的长期依赖关系，为进一步增强模型的短时预测能力，本文将经一维卷积后的短时特征向量S与经LSTM长时特征提取模块得到的长时特征向量L进行拼接融合，得到长短时融合特征向量$M_{sl}=\{m_1,m_2,...m_n\}$。

2）基于注意力机制的加权长短时特征融合

在普通LSTM编码-解码模型中，编码器提取特征得到的特征向量作为输入直接传入解码器中。这种情况下，默认时序数据中的不同时刻的历史信息对当前时刻的影响是相同的，其影响权重相同。然而，在集群负载时序数据中，不同时刻的历史信息对当前时刻的影响是不同的。例如，峰顶负载与波谷负载对当前时刻的影响是不同的。因此，其历史信息对当前时刻的影响权重也应当是不同的。

为充分利用历史信息对当前时刻的影响，更细粒度地利用负载数据中的长时依赖关系，本文引入注意力机制，为不同时刻的历史信息对当前时刻的影响赋予不同的权重值，以表明不同时刻历史信息对当前时刻的影响不同。

本文注意力向量C_i的计算如式19所示，其中隐藏状态h_j的计算如式20，式中e_{ij}如式21。

$$C_i=\sum_{i=1}^{T}\alpha_{ij}h_j \tag{19}$$

$$\alpha_{ij}=\frac{\exp(e_{ij})}{\sum_{k=1}^{T}\exp(e_{ik})} \tag{20}$$

$$e_{ij}=a(S_{i-1},h_j) \tag{21}$$

在进行LSTM解码时，对每个预测步的长短时融合特征向量进行注意力加权处理，得到每个预测步的加权长短时融合特征向量$W_m=\{w_1,w_2,...w_n\}$。然后并将其作为输入，送入LSTM解码模块进行解码。

3.3.5 解码与预测

1）LSTM解码

LSTM解码器将经特征融合处理得到的加权长短时融合特征向量W_m依次输入解码器，依次得到每一时刻的解码输出值。如图中所示，加权长短时特征融合发生在解码的每一个时刻。对解码器而言，其每个时刻的输入$W_m=\{w_1,w_2,...w_n\}$由经注意力机制处理后的加权长时特征向量和经一维卷积得到的短时特征向量S拼接融合得到。在解码过程中，解码器的神经元依次读取加权长短时融合特征、更新其神经元状态和隐藏状态，输出当前时刻的解码值。每一时刻的

解码输出会作为解码器下一时刻的输入。最终得到解码向量 D 。

2）预测输出层

预测输出层负责将 LSTM 解码器得到的解码向量进行激活处理，得到对应时刻的预测值。

该层是一个三层感知网络，其中前两层的激活函数为 PReLU 函数，相对于 ReLu 和 LeakyReLU 函数，PReLU 函数具备更好的性能[63]，并且其模型参数量变化不大，因此不会增加训练过程中的过拟合风险。其计算公式如下所示。

$$f(x)=\max(\alpha x) \tag{22}$$

其中，参数 α 在模型训练过程中不断更新。

第三层的激活函数为 sigmoid 函数，本文使用该函数以保证预测输出值维持在 0 到 1 之间。其计算公式如下所示。

$$y=f(x)=\frac{1}{1+e^{-\theta x}} \tag{23}$$

其中， y 为模型最终的集群负载预测输出值。

3.4 算法描述

本章节所述算法模型主要分为以下 7 步：

步骤 1，获取用户请求流量数据和集群负载数据，并对数据进行预处理以消除噪声数据；

步骤 2，对用户请求流量和集群负载分别进行聚类分类和多变量联合特征选择处理。使用基于路经查询优化的 DTW 聚类算法确定用户请求时序序列的类型；根据不同资源变量特征之间的相关性，为目的变量特征选择多个相关变量特征，将单一变量时序预测问题转化为多变量时序预测问题；

步骤 3，对时序序列进行短时特征提取以强化短时特征，使用一维全卷积神经网络（1D FCN）对原时序负载数据 X 进行一维全卷积操作，得到短时特征向量 S ；

步骤 4，利用基于 LSTM 的长时特征编码器对时序数据进行长时特征提取，得到负载时序序列的长时特征向量 L ；

步骤 5，将经一维卷积后的短时特征向量 S 与经 LSTM 长时特征提取得到的长时特征向量 L 进行拼接融合，得到长短时融合特征向量 M_{sl} 。

步骤 6，通过基于 LSTM 的解码器进行解码。在进行 LSTM 解码时，对每个预测步的长短时融合特征向量 M_{sl} 进行注意力加权处理，得到每个预测步的加权长短时融合特征向量 W_m 。

步骤 7，将 LSTM 解码器得到的解码向量进行激活处理，得到对应时刻的预测值。

3.5 实验与分析

本小节通过大量实验对基于加权长短时特征融合的双时序流量预测模型进行研究分析，与多种基准模型进行实验对比，以验证模型性能。本章使用 google-cluster-trace-v2011[64]和 alibaba-cluster-trace-v2018[65]两个数据集分别对用户请求时序数据和集群负载时序数据进行实验，用来评估基于加权长短时特征融合的双时序流量预测模型的有效性和准确性，同时还评估了模型的不同参数设置对流量预测的影响。

3.5.1 实验环境与数据集

1）实验环境

本文模型基于 Pytorch1.4.0 实现，Python 版本为 3.6。

模型训练迭代次数为 200 轮，在训练阶段，选择 Adam 优化算法，设置训练批次大小为 20，学习率为 0.001，dropout 参数为 0.4，采用的损失函数为每类时序流量的预测值与真实值之间的 MSE 均方差损失。为验证多变量联合特征选择和一维全卷积短时特征提取的有效性，本章分别设计消融试验。分别使用单变量特征预测以及无短时特征强化的模型与原模型进行对比。

2）数据集简介

本章分别使用 google-cluster-trace-v2011 和 alibaba-cluster-trace-v2018 数据集作为用户请求预测和集群负载预测的性能评估数据集。

google-cluster-trace-v2011 数据集跟踪了自 2011 年 5 月以来，在一个约 12,500 台服务器的集群上，共计 29 天的服务器节点负载信息。该数据集共包含 6 种不同的数据表。本章使用其 task table 表作为用户请求预测模型的训练和测试集。

alibaba-cluster-trace-v2018 数据集为阿里云于 2018 年发布的集群公开数据集，该数据集包含余约 4000 台服务器在 8 天内的资源消耗情况。本文使用其中 10 台服务器在 8 天内的资源消耗情况，共计约 70000 条数据，其中前 80%为作为训练集，后 20%作为测试集。

两个数据集均包含服务器中 CPU、内存、磁盘和网络等资源的消耗情况，可以很好地表现集群环境中的机器负载特征。工作负载泛指集群的多种性能指标，不同的集群环境对于负载预测问题所关注的侧重点不同。由于集群中 CPU 利用率的波动大以及内存墙现象的存在，本文使用 CPU 消耗情况作为本文模型预测能力的关键评价指标。以 alibaba-cluster-trace-v2018 为例，如图 3.7 所示，为表现集群中服务器集群短期负载和长期负载变化的不同特点，本文选择其中一台服务器，分别展示其 CPU 资源在一天和一分钟的周期中负载消耗变化情况。

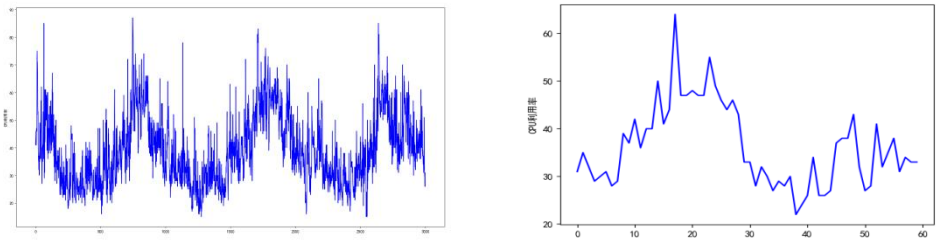


图 3.7 CPU 每日和每分钟利用率

3.5.2 评价指标与基准模型

1）评价指标

为验证模型的预测准确度，本文使用以下三种误差度量方式作为预测模型的评价指标[68]。其计算方式如下。

（1） MAE（Mean Absolute Error）

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\tilde{y}^{(i)} - y^{(i)}| \tag{24}$$

（2） RMSE（Root Mean Square Error）

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tilde{y}^{(i)} - y^{(i)})^2} \tag{25}$$

（3） MAPE（Mean Absolute Percentage Error）

$$MAPE = \frac{1}{n}\sum_{i=0}^{n}\left|\frac{\tilde{y}^{(i)} - y^{(i)}}{y^{(i)}}\right| \times 100\% \tag{26}$$

2）基准模型

为比较、验证本文流量预测模型的预测效果，本文采用 AR、MA、ARIMA[66]、LSTM、AutoEncoder[67]和 TCN 等模型作为本文的基准对比模型。

AR：Auto-Regressive，自回归模型。是统计学中的一种处理时间序列的方法，用变量的历史信息预测当前时刻信息，并假设它们为线性关系。自回归模型被广泛运用于经济学、信息学、自然现象等领域的预测工作。

MA：Moving Average，滑动平均模型。滑动平均法是一种简单平滑预测技术，其基本思想为：根据时间序列信息，逐项推移，依次计算包含一定项数的时序平均值，以反映长期趋势的方法。因此，当时间序列的数值由于受周期变动和随机波动的影响，起伏较大，不易显示出时间的发展趋势时，使用滑动平均模型可以消除这些因素的影响，显示出时间的发展方向与趋势，然后分析预测序列的长期趋势。

ARIMA：Autoregressive Integrated Moving Average，差分整合移动平均自回归模型。该模型

由 AR 和 MA 模型整合而成，充分利用两种模型的优势，将非平稳时间序列经差分处理转化为平稳时间序列，然后对因变量的滞后值以及随机误差项的现值和滞后值进行回归算建立模型。

LSTM：Long Short-Term Memory，长短期记忆网络。LSTM 是一种时间循环神经网络，是为了解决一般的 RNN 循环神经网络存在的长期依赖问题而专门设计出来的。LSTM 利用其独特的门控机制，有效解决了 RNN 存在的梯度消失问题，适合于处理和预测时间序列中间隔和延迟非常长的重要事件。

AutoEncoder：自编码器。AutoEncoder 是一类在半监督学习和非监督学习中使用的人工神经网络，其功能是通过将输入信息作为学习目标，对输入信息进行表征学习。自编码器包含编码器（encoder）和解码器（decoder）两部分，本文中 AutoEncoder 编码器和解码器均为 LSTM 网络。

TCN：Temporal Convolutional Network，时序卷积网络。TCN 基于卷积神经网络对时序问题进行建模，借助因果和膨胀卷积机制获取时序数据中的长期依赖信息，解决时序问题。

3.5.3 实验结果分析

1）用户请求流量预测结果

为验证基于加权长短时特征融合的双时序流量预测模型对用户请求的预测准确度，本章选择 AR、MA、ARIMA、LSTM、TCN、和 AutoEncoder 等模型作为对比实验的基准模型，并选择各模型的最好预测结果进行负载预测能力比较。各模型在 google-cluster-trace-v2011 数据集上的预测结果如表 3.3 所示。

通过表 3.3 我们可以看出，从 MAPE 指标来看，三个基于时序神经网络的模型--基础 LSTM、AutoEncoder 和本文模型--的表现要优于三个非时序神经网络模型--AR、MA 和 ARIMA。但从 MSE、MAE 和 RMSE 这三项指标来看，LSTM 时序神经网络模型和 TCN 卷积神经网络的表现并没有表现出对非三个非时序神经网络模型的优势。这表明，以长时特征提取见长的纯时序神经网络或以短时特征提取见长的纯卷积神经网络在用户请求流量预测中未能表现出相关优势，同时也表明仅仅对用户请求时序数据做长时或短时特征提取无法实现较好的时序预测效果。而 AutoEncoder 模型凭借其特有的编码-解码结构，能够对用户请求时序数据进行较充分的特征挖掘，进而实现相对较好的预测效果。这说明了编码-解码结构在时序特征提取方面存在的优势。而本章模型优于 AutoEncoder 模型，是因为本章模型中的一维卷积提供的短期特征和注意力机制进一步增强了解码器的长短期预测能力。

表 3.3 不同模型在 google-cluster-trace-v2011 数据集上的预测结果

Model	google-cluster-trace-v2011× 10^{-2}			
	MSE	MAE	RMSE	MAPE
AR	0.10	1.88	3.14	36.10%
MA	0.12	2.19	3.49	40.73%
ARIMA	0.12	1.98	3.48	36.48%
LSTM	0.95	6.96	9.78	16.83%
AutoEncoder	0.10	1.32	1.69	13.00%
TCN	9.11	9.90	3.75	16.90%
Our model	**0.03**	**1.20**	**1.62**	**12.55%**

2）集群负载流量预测结果

（1）验证模型的一般预测准确度

为验证基于加权长短时特征融合的双时序流量预测模型对集群负载的预测准确度，本章选择 AR、MA、ARIMA、LSTM、TCN、和 AutoEncoder 等模型作为对比实验的基准模型，并选择各模型的最好预测结果进行负载预测能力比较。各模型在 alibaba-cluster-trace-v2018 数据集上的预测结果如表 4 所示。

通过表 3.4 我们可以看出，三个基于时序神经网络的模型--基础 LSTM、AutoEncoder 和本文模型--的表现要优于三个非时序神经网络模型--AR、MA 和 ARIMA。这说明，在非静态、非线性、变化复杂的时序负载数据中，非时序神经网络模型 AR、MA 和 ARIMA 不能有效地进行负载预测。另外，我们可以看出，以提取时序数据短期依赖能力见长的时序卷积网络 TCN 的负载预测能力并没有体现出很大的优势。

同时，在基于时序神经网络的基础 LSTM、AutoEncoder 和本章模型中，其中 AutoEncoder 和本文模型两个使用编码-解码结构的模型的预测能力要优于基础 LSTM 模型。这说明了编码-解码结构在时序特征提取方面存在的优势。而本章模型优于 AutoEncoder 模型，是因为本章模型中的一维卷积提供的短期特征和注意力机制进一步增强了解码器的长短期预测能力。

表 3.4 不同模型在 alibaba-cluster-trace-v2018 数据集上的预测结果

Model	alibaba-cluster-trace-v2018× 10^{-2}			
	MSE	MAE	RMSE	MAPE
AR	0.66	5.82	8.38	13.75%
MA	0.76	6.46	8.73	15.19%
ARIMA	0.81	6.36	8.99	14.73%
LSTM	0.63	6.31	7.96	12.40%
AutoEncoder	0.50	5.80	7.08	13.46%
TCN	1.04	5.91	8.24	13.36%
Our model	**0.09**	**2.30**	**3.04**	**4.90%**

（2）验证模型的长短时预测能力

为验证基于加权长短时特征融合的双时序流量预测模型的长短期负载预测能力，本章将时间步长设置为 5 分钟，计算不同模型在长度为 12 的步长序列中不同预测长度的负载预测准确度。表 3.5 和图 3.8 所示为基础 LSTM、AutoEncoder、TCN 和本章模型在不同预测步长时的预测结果。

通过表 3.5 和图 3.8 我们可以看出，随着预测步长的增大，各基准模型和本章模型的 MAPE 都呈现逐渐增大的趋势。但就短期和长期预测能力而言，本章模型都要由于基础 LSTM、AutoEncoder 和 TCN 三个基准模型。尤其是在预测步长小于 4，即预测步长小于 20 分钟时，本章模型的短期负载预测准确度是明显优于其他三个基准模型的。在预测步长大于 20 分钟的长期预测中，本章模型仍表现出其预测优势。这说明本章模型长短期预测都能取得良好表现。当然，在预测步长为 4 时，LSTM 基准模型的 MAE 参数略优于本章模型 0.0001，该数据差值属于合理误差范围；预测步长为 6 时，AutoEncoder 基准模型在 MAE 方面的表现略优于本章模型；预测步长为 8 时，AutoEncoder 模型的 RMSE 参数优于本章模型。这是因为，在预测步长为 6 和 8 时，AutoEncoder 模型能够更好地挖掘历史负载中的时序特征，并通过解码器进行更好地负载预测。

表 3.5 不同模型在不同预测步长时的预测结果

Prediction	LSTM			AutoEncoder			TCN			Our model		
Step	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
2	0.0667	0.0905	0.1197	0.0535	0.0651	0.0989	0.1026	0.1040	0.1742	**0.0230**	**0.0305**	**0.0490**
4	**0.0505**	0.0735	0.1185	0.0622	0.0734	0.1148	0.1235	0.1303	0.1917	0.0506	**0.0698**	**0.0992**
6	0.0667	00905	0.1197	**0.0509**	0.0719	0.1137	0.1198	0.1339	0.2213	0.0545	**0.0698**	**0.1077**
8	0.0609	0.0787	0.1219	0.0646	**0.0759**	0.1207	0.1146	0.1383	0.2421	**0.0535**	0.0766	**0.1027**
10	0.0652	0.0771	0.1241	0.0667	0.0802	0.1279	0.1381	0.1444	0.2582	**0.0576**	**0.0758**	**0.1101**
12	0.0661	0.0823	0.1286	0.0711	0.0853	0.1378	0.1345	0.1708	0.2555	**0.0599**	**0.0744**	**0.1164**

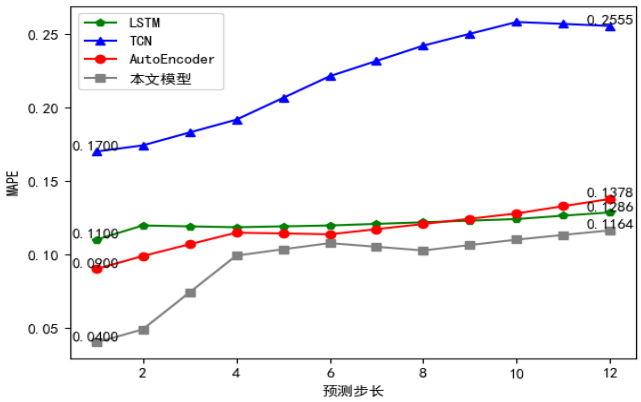
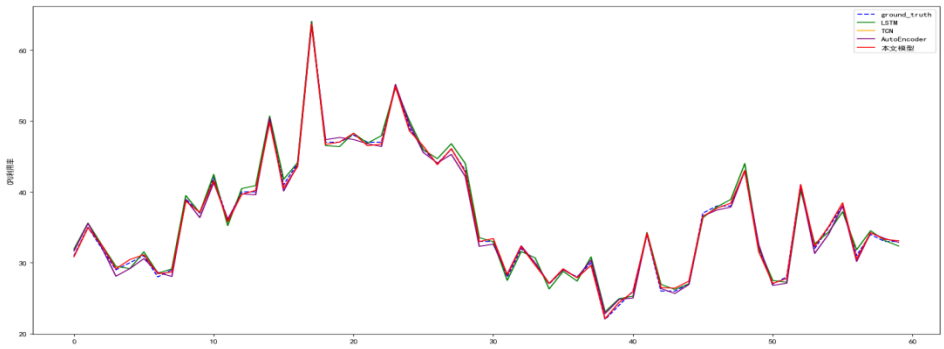


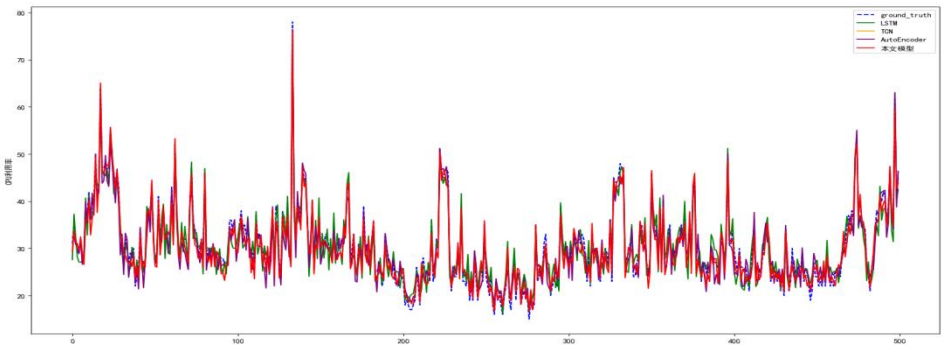
图 3.8 各模型不同预测步长的 MAPE

同时，通过图 3.9 我们可以看出，本章模型在以分钟为周期的短时预测和以天为周期的长

时预测方面都能取得较好的预测效果。



（a）CPU 每分钟预测



（b）CPU 每日预测

图 3.9 CPU 每分钟预测和每日预测

3.5.4 参数设置

本节对模型中的滑动窗口长度、多变量联合特征选择在不同配置时模型的性能表现进行研究。

1）滑动窗口长度

为从另一个角度验证本章模型的长短期预测能力，进一步检验其长短期历史负载的特征提取和依赖捕获能力，我们计算在对负载序列采用不同滑动窗口时各模型的负载预测准确度。滑动窗口序列为 1 到 140。图 3.10 所示为本章模型在不同滑动窗口时的预测结果。

通过图 3.10 我们可以看出，随着滑动窗口的不断增大，模型在滑动窗口大于等于 60 时，其预测的 MAE、EMSE 和 MAPE 等各项指标均逐渐趋于平稳，未呈现增大趋势。由此可见，在长负载序列中，本章模型能够较好地捕获序列中的长期负载依赖关系，且呈现较好的稳定性。

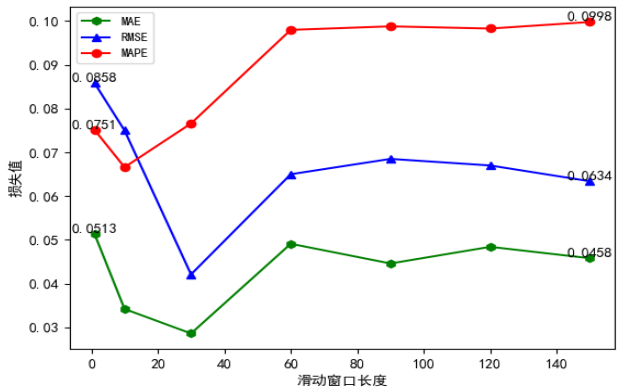


图 3.10 不同滑动窗口时的 MAE、RMSE 与 MAPE 的值

2）多变量联合特征选择

为验证不同资源变量的特征组合对 CPU 负载预测效果的影响，本章进行了验证实验，结果如图 3.11 所示。

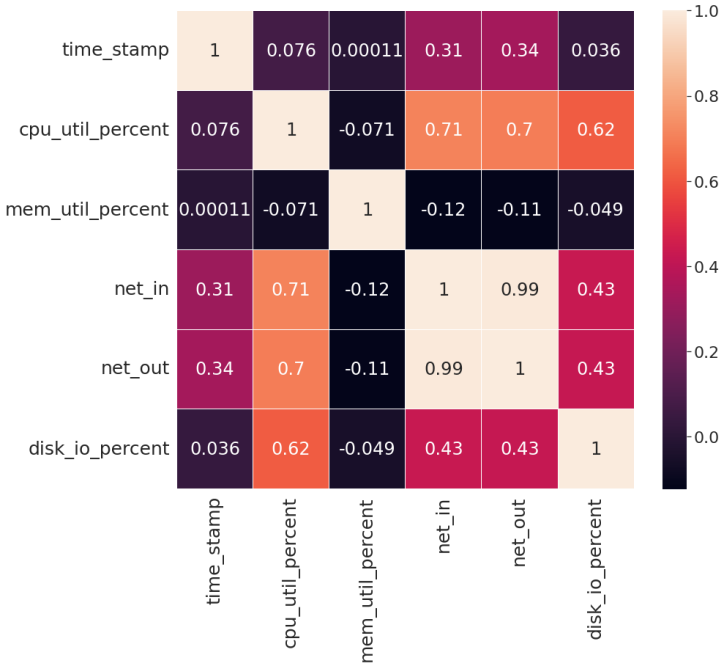


图 3.11 各项资源指标相关度

结合图 3.12 各项资源指标相关度，我们可以看出，尽管网络输入输出特征与 CPU 特征的相关度最高，但在实际预测结果来看，磁盘特征更有助于 CPU 负载预测。具体到各项指标，只利用 CPU 和磁盘资源进行预测时，其 MAE、RMSE 和 MAPE 等各项性能指标都是最好结果；当磁盘与其他资源结合时，CPU 的预测性能都能实现较好的表现。

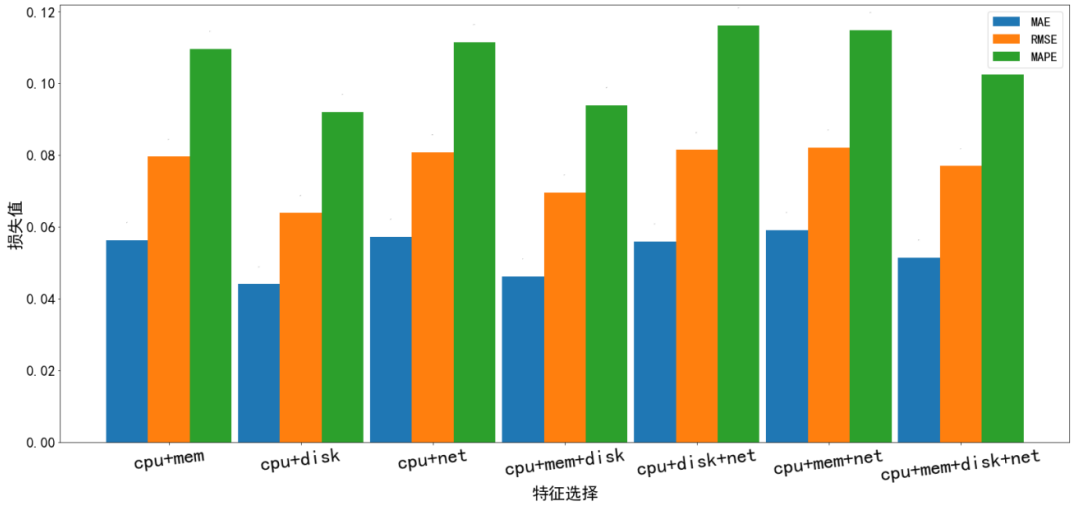


图 3.12 不同资源变量特征组合对 CPU 预测的影响

3.6 本章小结

本章针对用户请求流量历史累积数据少、时序数据周期性差以及集群负载流量预测无法同时兼顾短时、长时预测的问题，提出了一种基于加权长短时特征融合的双时序流量预测模型，分别利用基于查询路径优化的 DTW 用户请求时序数据聚类算法和多变量联合特征选择技术对用户请求流量和集群负载流量进行特征提取前的预处理。然后利用基于注意力机制的加权长短时特征融合技术对时序数据进行短时与长时特征提取、长短时特征融合以及向量加权等处理，充分挖掘时序数据的长短时特征，实现高准确度的时序流量短期预测和长期预测。最后通过 google-cluster-trace-v2011 和 alibaba-cluster-trace-v2018 真实公开数据集评估了该模型。

第四章 基于预测自响应的集群综合负载均衡模型

4.1 集群综合负载均衡问题描述

服务器集群通过负载均衡策略将来自客户端的用户请求分发到后端服务器，以实现分解用户请求流量，降低后端服务器压力的目的。目前多数负载均衡算法都是基于动态加权思想，根据服务器各项性能指标，为后端服务器设置不同权值，然后确定用户请求分配方案[69]。例如，加权轮询法、加权最小连接数法等常用集群负载均衡算法都是基于动态加权思想。

然而，集群系统中的服务器处于动态变化过程中，随着用户请求处理的推进，其自身各项性能指标均处于不断变化中。显然，只使用服务器当前的性能指标作为负载均衡分配方案的依据会使得负载决策的实时性不足。为改善这一问题，很多负载均衡方案借助负载预测技术，在进行负载均衡决策时参考预测得到的集群服务器负载，结合服务器当前的实时负载性能，以提高集群负载决策的实时性[70]。在利用负载预测得到的服务器负载进行负载均衡决策时，需要认识到用户请求对预测到的服务器负载的影响，即需要认识到二者之间的相互作用关系。用户请求作用到该服务器时，服务器在进行用户请求处理的过程中其自身负载性能指标会发生相应的变化，该过程为服务器针对该用户请求的响应过程。

集群局部性角度来看，目前的负载均衡方案多是针对单个用户请求任务的全局任务分配方案。在该方案中，用户请求分配是一次性的，负载均衡策略只负责为用户请求指定目标服务器。然而，在集群实际运行过程中，即使集群没有接收到新的用户请求任务，其自身任务处理过程中也会发生负载超载或负载不足等情形[71]。因此，从局部性角度来看，需要关注集群局部部分服务器之间的任务运行和负载情况，针对局部服务器进行动态负载调度，以实现集群局部高效运行，进而推动整个集群负载均衡。

概括来讲，当前负载均衡技术存在以下问题：

第一，负载均衡策略的实时性不够。传统软件方法的负载均衡算法无法实时获取集群服务器工作负载导致负载均衡滞后效果明显，但频繁对服务器进行负载采样以获取实时负载会导致增加服务器压力；

第二，负载均衡策略的准确性不足。基于流量预测方法的负载均衡算法只针对用户请求或集群负载中的某一流量做流量预测且未考虑用户请求和服务器负载之间的相互作用，导致负载均衡决策准确性不足[72]；

第三，负载均衡策略未能兼顾全局和局部均衡。目前多数负载均衡策略着眼于用户请求的一次性全局分配，未能充分考虑到集群局部服务器节点运行过程中出现的负载不足和负载超载情形，存在集群局部失衡问题。

基于此，本章针对上述集群负载均衡在全局和局部方面存在的问题，充分分析集群全局任

务分配和局部负载调度机制并结合第三章提出的用户请求和集群负载预测技术，提出基于预测自响应的集群综合负载均衡算法。该算法分为全局和局部两个层次，全局层面主要有基于预测自响应的全局任务分配方法;局部层面主要有基于集群服务器自索取的局部动态负载调度方法。具体来说，全局层面在双时序流量预测的基础上，挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立合理的全局任务分配模型;局部层面协调局部相邻服务器节点之间的任务分配关系,平衡各服务器节点之间的负载，减轻负载均衡器压力，降低集群通信开销，减少服务器集群整体资源消耗。

本章的主要研究内容如下：

1）针对集群负载均衡策略在全局任务分配和局部负载调度方面存在的问题，提出基于预测自响应的集群综合负载均衡算法。该算法从全局和局部两个层面可分为基于预测自响应的全局任务分配方法和基于集群服务器自索取的局部动态负载调度方法两个子方法。

2）针对集群负载均衡决策在全局任务分配方面存在的实时性和准确性不足的问题，提出一个基于预测自响应的全局任务分配方法。该方法在双时序流量预测的基础上，利用流量预测得到的用户请求和集群负载，挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立实时、准确的全局任务分配模型；

3）针对集群负载均衡决策在局部负载调度方面存在的局部失衡问题，提出基于集群服务器自索取的局部动态负载调度方法。该方法协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载，减轻负载均衡器压力，降低集群通信开销，减少服务器集群整体资源消耗。

4.2 基于预测自响应的集群动态负载均衡模型框架

本章深入研究了基于预测自响应的全局任务分配技术和基于集群服务器自索取的局部动态负载调度技术等，综合这两种技术，进而提出一种基于预测自响应的集群综合负载均衡模型。从全局层面看,该模型利用基于预测自响应的全局任务分配技术,在双时序流量预测的基础上，利用流量预测得到的用户请求和集群负载,挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立实时、准确的全局任务分配模型；从局部层面看，该模型利用基于集群服务器自索取的局部动态负载调度技术，调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载，减轻负载均衡器压力，降低集群通信开销，减少服务器集群整体资源消耗。

本章提出的基于预测自响应的集群综合负载均衡模型主要由基于预测自响应的全局任务分配模块和基于集群服务器自索取的局部动态负载调度模块组成，如图 4.1 所示：

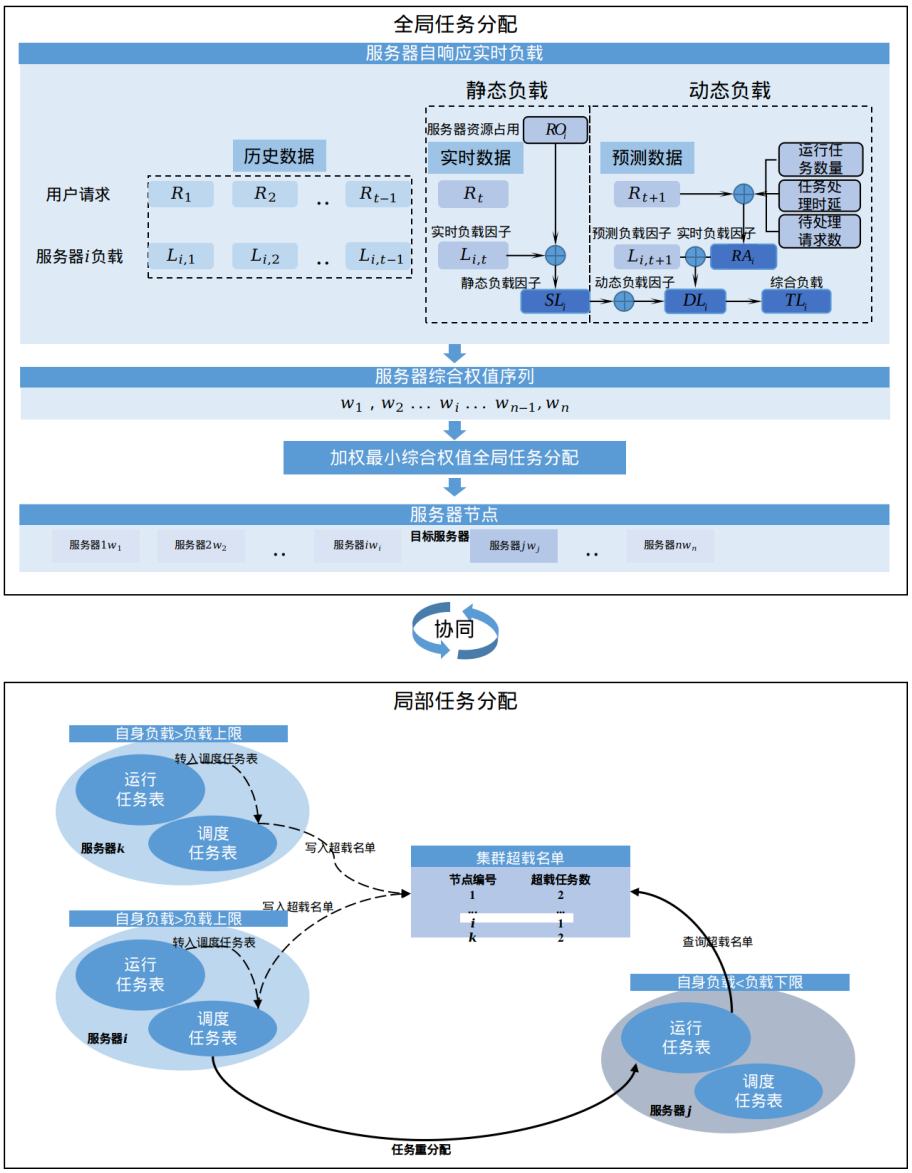


图 4.1 基于预测自响应的集群动态负载均衡模型框架

在全局任务分配模块，本模型首先利用第三章提出的用户请求和集群负载预测模型，分别对用户请求流量和集群负载流量建立时序预测模型，分别将用户请求历史序列和集群负载历史序列作为模型输入，经过多变量联合特征提取和 DTW 聚类、长短时特征提取、LSTM 解码以及预测输出，得到用户请求和集群负载的预测值。然后建立用户请求与集群负载之间的作用和反馈模型，并结合服务器自身性能参数，建立实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的作用和响应模型，评估服务器自响应预测负载，然后根据集群实时负载和自响应预测负载确定服务器自响应实时负载。根据集群服务器自响应实时负载序列建立加权最小负载全局任务分配策略，为用户请求分配合适的目标服务器。

在局部动态负载调度模块，对局部范围内的多个服务器建立接受者主动的服务自索取动态任务调度策略。服务器查看自身负载，若服务器 i 负载小于其负载下限，则查询集群超载名单

表，查看是否存在与该服务器i相邻的服务器j，若存在则按服务器i自身负载能力调用服务器j的任务至服务器i的工作任务表；否则服务器i继续执行自身的其他任务。若服务器i负载大于其负载上限，则将超出任务转入服务器i的转移任务表，并将服务器i的编号和超载任务数写入集群超载名单表，然后继续执行服务器i的其他任务。

4.3 基于预测自响应的集群动态负载均衡模型实现

本章模型从全局和局部两个层面可分为基于预测自响应的全局任务分配子模型和基于集群服务器自索取的局部动态负载调度子模型。

4.3.1 基于预测自响应的全局任务分配算法

1）集群服务器综合负载

集群服务器综合负载包含服务器静态负载和自响应动态负载两部分。其中静态负载表征服务器的资源占用情况和实时负载能力；自响应动态负载综合考虑预测得到的未来一段时间的服务器负载和当前任务运行情况，表征服务器运行过程中的剩余负载能力。

（1）静态负载

静态负载包含服务器资源占用和实时负载因子两部分。

首先定义服务器资源占用因子RO_i。

本文定义服务器的平均 CPU 频率C_{avg}、平均内存容量M_{avg}、平均磁盘容量D_{avg}和平均网络带宽N_{avg}分别为

$$C_{avg}=\frac{\sum_{i=1}^{n}c_i}{n} \tag{27}$$

$$M_{avg}=\frac{\sum_{i=1}^{n}m_i}{m} \tag{28}$$

$$D_{avg}=\frac{\sum_{i=1}^{n}d_i}{n} \tag{29}$$

$$N_{avg}=\frac{\sum_{i=1}^{n}n_i}{n} \tag{30}$$

其中，c_i、m_i、d_i、n_i分别为服务器i的 CPU 频率、内存容量、磁盘容量和网络带宽，n为集群中服务器总数。

因此，服务器资源占用因子RO_i为：

$$RO_i = \alpha_c \frac{c_i}{C_{avg}} + \alpha_m \frac{m_i}{M_{avg}} + \alpha_d \frac{d_i}{D_{avg}} + \alpha_n \frac{n_i}{D_{avg}} \tag{31}$$

其中，c_i、m_i、d_i、n_i 分别为服务器 i 的 CPU 频率、内存容量、磁盘容量和网络带宽，n 为集群中服务器总数，α_c、α_m、α_d、α_n 分别为 CPU 频率、内存容量、底盘容量和网络带宽的影响程度。

然后定义实时负载因子 AL_i。

服务器实时负载因子反映服务器当前时刻的实际负载，表征服务器当前运行情况。服务器实时负载因子 AL_i 定义为：

$$AL_i = \alpha_c L_{c_i} + \alpha_m L_{m_i} + \alpha_d L_{d_i} + \alpha_n L_{n_i} \tag{32}$$

$$\alpha_c + \alpha_m + \alpha_d + \alpha_n = 1 \tag{33}$$

其中，L_{c_i}、L_{m_i}、L_{d_i}、L_{n_i} 分别为服务器 i 的 CPU 使用率、内存使用率、磁盘使用率和网络带宽占用率，α_c、α_m、α_d、α_n 分别为 CPU 使用率、内存使用率、磁盘使用率和网络带宽占用率的影响程度。该影响程度根据上文中的层次分析法确定。

最后，服务器的静态负载因子 SL_i 可以定义为：

$$SL_i = \beta_{ro} RO_i + \beta_{al} AL_i \tag{34}$$

$$\beta_{ro} + \beta_{al} = 1 \tag{35}$$

其中，β_{ro} 和 β_{al} 分别为服务器资源占用因子 RO_i 和实时负载因子 AL_i 对服务器静态负载因子的影响程度。

（2）自响应动态负载

自响应动态负载包含服务器预测负载因子和自响应实时负载因子两部分。

首先定义预测负载因子 PL_i，服务器预测负载为第三章提出的用户请求和集群负载预测模型预测得到的服务器负载。

服务器预测负载因子 PL_i 定义为：

$$PL_i = \alpha_c L_{pc_i} + \alpha_m L_{pm_i} + \alpha_d L_{pd_i} + \alpha_n L_{pn_i} \tag{36}$$

$$\alpha_c + \alpha_m + \alpha_d + \alpha_n = 1 \tag{37}$$

其中，L_{pc_i}、L_{pm_i}、L_{pd_i}、L_{pn_i} 分别为服务器 i 预测得到的 CPU 利用率、内存利用率、磁盘利用率和网络带宽占用率，α_c、α_m、α_d、α_n 分别为预测得到的 CPU 利用率、内存利用率、磁盘利用率和网络带宽占用率对服务器预测负载因子 PLi 的影响程度。该影响程度根据上文中的层次分析法确定。

然后定义自响应实时负载因子 RA_i。服务器静态负载因子表征服务器的当前时刻的资源占用情况和实时负载能力，而无法表征服务器的在未来一段时间的剩余负载能力。因此，本章结合第三章提出的用户请求和集群负载预测模型，从用户端和集群服务器端两个角度综合评估服务器在未来一段时间的剩余负载能力。从用户端来看，主要参考因素为预测得到的用户请求对各项资源的需求 PR；从集群服务器端来看，参考因素有预测得到的集群服务器负载 PL，以及服务器节点当前的任务运行数 N_{now}、任务处理平均时延 T、待处理用户请求数 N_{to}。显然，这些因素之间存在相互作用与响应关系。由此，本章将这些因素的作用和响应结果定义为自响应实时负载因子。

因此，服务器自响应实时负载因子 RA_i 的定义为：

$$RA_i = \alpha_{pr} PR + \alpha_{pl} PL + \alpha_{now} PN_{now} + \alpha_t T + \alpha_{to} N_{to} \tag{38}$$

$$\alpha_{pr} + \alpha_{pl} + \alpha_{now} + \alpha_t + \alpha_{to} = 1 \tag{39}$$

其中，PR 为预测得到的用户请求对各项资源的需求，PL 预测得到的集群服务器负载，N_{now}、T、N_{to} 分别为服务器 i 当前的任务运行数、任务处理平均时延、待处理用户请求数；α_{pr}、α_{pl}、α_{now}、α_t、α_{to} 分别为上述各项因素对服务器自响应实时负载因子 RA_i 的影响程度。

最后，服务器的动态负载因子 DL_i 可以定义为：

$$DL_i = \beta_{pl} PL_i + \beta_{ra} RA_i \tag{40}$$

$$\beta_{pl} + \beta_{ra} = 1 \tag{41}$$

其中，β_{pl} 和 β_{ra} 分别为服务器预测负载因子 PL_i 和自响应实时负载因子 RA_i 对服务器动态负载因子的影响程度。

（3）服务器综合负载

服务器 i 综合负载 TL_i 为：

$$TL_i = \frac{\gamma_1 DL_i}{\gamma_2 SL_i} \tag{42}$$

$$\gamma_1 + \gamma_2 = 1 \tag{43}$$

其中，SL_i 为服务器 i 的静态负载因子，DL_i 为服务器 i 的动态负载因子，γ_1 和 γ_2 为静态负载因子和动态负载因子对服务器综合负载的影响程度。

由公式定义 43 可以看出，服务器综合负载与静态负载因子成反比，与动态负载因子成正比，即静态负载因子越大，服务器的初始承载能力越强，相应的未来负载越低；动态负载因素越大，服务器的负载越重，未来剩余承载能力越低。

2）集群服务器综合权值

服务器i的综合权值ω_i定义为：

$$\omega_i = \frac{\frac{1}{TL_i}}{\sum_{k=1}^{n}\frac{1}{TL_k}} \tag{44}$$

其中，TL_i为服务器i的综合负载，n为集群中服务器总数。

由公式定义 44 可以看出，服务器综合权值之比为服务器综合负载倒数之比,综合负载高的服务器权值占比较低,处理更少请求,综合负载低的服务器权值占比较高,处理更多请求。

3）用户请求分配方案

得到集群中各服务器节点的综合权值之后，根据加权轮询法的思想[73]，对集群后端的服务器节点进行排序，得到一个有序的服务器节点序列。然后将其上报至集群负载均衡器，由负载均衡器根据该服务器节点序列为各用户请求确定对应的目标服务器节点，由此确定用户请求分配方案。

4.3.2 基于服务器自索取的局部动态负载调度算法

1）相关参数

服务器实时负载$Load$：根据层次分析法计算得到的服务器节点当前负载。

负载上限$LoadMax$：服务器节点能够承载的最大任务负载上限。

负载下限$LoadMin$：服务器节点承载任务低于正常情况的负载下限。

运行用户任务数：服务器节点当前运行的用户任务数量。

服务器节点状态：根据服务器节点实时负载与其负载上下限，可将其状态分为正常、空载、超载三种状态。

2）局部动态负载调度算法

（1）运行任务表

该表记录当前服务器节点上运行的用户任务，用户任务记录其所属节点编号，同时记录每个任务占用的负载量。当服务器节点负载超出其自身负载上限$LoadMax$时，此时服务器已无法承受当前用户任务总量，需将运行任务表中的部分用户任务迁出，将其调度至集群的超出任务名单。同时，将自身状态设置为超载，等待其他负载不足的服务器节点到超载任务名单中自动索取并处理这些迁出任务。

（2）调度任务表

该表记录当前服务器节点上待迁出并调度至其他节点的用户任务。当服务器节点负载低于其自身负载下限$LoadMin$时，该节点会主动查询集群的超载任务名单，根据名单中用户任务所属的服务器节点，到对应节点中查看其调度任务表，选择适合自身负载条件的超载任务。然后

通过用户任务迁移技术，将对应任务迁移至自身节点，同时添加任务信息到该节点的运行任务表。同时更新迁出服务器节点的调度任务表以及集群的超载任务名单。

（3）超载任务名单

该表记录集群当前各服务器节点中发生超载的用户任务。当服务器节点负载超出其自身负载上限 $LoadMax$ 时，会将对应用户任务添加至超载任务名单；当空闲服务器节点完成用户任务迁移后，会将被迁移的任务从超载任务名单删除。

3）局部动态负载调度算法工作原理

如图 4.2 所示为基于服务器自索取的局部动态负载调度算法的工作机制，该算法模型可分为 4 步。

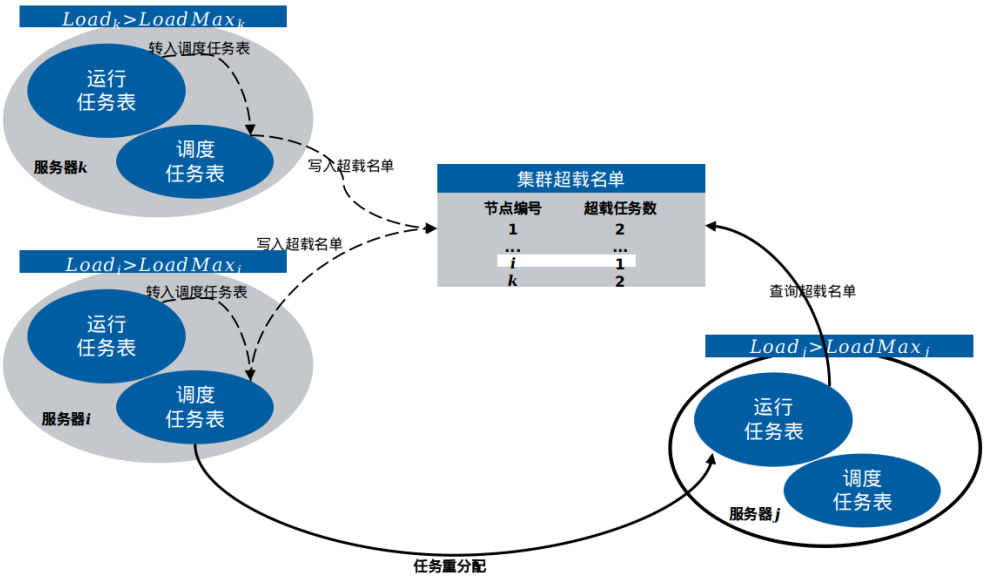


图 4.2 局部动态负载调度原理图

步骤 1，等待集群负载均衡器分配用户任务。若没有新的用户请求，则服务器节点的运行任务表和调度任务表均为空，服务器节点处于初始状态；若负载均衡器根据上一节提出的全局任务分配策略分配用户请求任务至当前服务器节点，则转到步骤 2；若没有新用户任务分配，则转到步骤 3；

步骤 2，服务器节点接管分配得到的用户请求任务，并将该任务添加至服务器节点的运行任务表，同时更新节点当前的实时服务器负载 $Load$；

步骤 3，检查服务器节点的节点运行状态标记，若节点运行状态标记不为空，则根据节点当前的负载上限 $LoadMax$ 和负载下限 $LoadMin$ 执行局部任务调度算法；否则，检查服务器节点的运行任务表和调度任务表，若两表均为空，则转到步骤 4；

局部任务调度算法：

if $LoadMin < Load < LoadMax$，表示当前节点处于正常状态。节点继续执行当前任务。返回步骤 3。

if $Load >= LoadMax$，表示当前节点处于超载状态。选择当前服务器节点中的用户任务，将该任务信息添加至节点的调度任务表，同时更新至集群的超载任务名单。然后继续执行节点的其他任务。返回步骤 3。

if $Load <= LoadMin$，表示当前节点处于空载状态。首先查询集群超载任务名单，若超载任务名单为空，则不进行任务调度，继续执行节点的其他任务。返回步骤 3；若超载名单不为空，则根据各超载任务所需的负载量选择适合自身的超载任务，然后找到该超载任务所在的服务器节点。然后根据用户任务迁移策略，将该任务迁移至自身节点。同时更新本节点的运行任务表、迁出节点的调度任务表以及集群超载任务名单。继续执行节点任务。返回步骤 3。

步骤 4，节点退出任务运行。

4.4 算法描述

本章节所述算法模型主要分为以下 6 步：

步骤 1，用户向集群发出访问请求，集群负载均衡器接收到用户请求；

步骤 2，负载均衡器利用基于加权长短时特征融合的双时序流量预测模型对用户请求和集群负载进行预测，得到未来一段时间的用户请求和集群服务器负载；

步骤 3，对用户请求进行全局任务分配。根据集群中各服务器的资源占用、实时负载、预测负载、自响应实时负载等信息，计算得到各服务器的综合负载，进而得到各服务器的综合权值；

步骤 4，根据用户请求的负载需求情况，为用户请求制定分配方案，确定目标服务器节点。完成全局任务分配；

步骤 5，对集群进行局部动态负载调度。查看各服务器节点的自身实时负载和自身负载上下限，执行局部任务调度算法，完成节点的任务调度；循环执行该步骤；

步骤 6，转步骤 1。

4.5 实验与分析

4.5.1 数据集

为验证本章所提出的全局任务分配模型和局部动态负载调度模型的性能，本章搭建了一个模拟集群环境，分别通过 Webbench 模拟用户并发请求和自定义函数库模拟运算任务对全局任务分配模型和局部动态负载调度模型进行性能测试[74]。

Webench 是一款轻量级的集群压力测试工具，能够测试处在相同硬件上、不同服务的性能以及不同硬件上同一个服务的运行状况。Webbench 最多可以对集群模拟 30,000 左右的并发请求，可以控制时间、是否使用缓存、是否等待服务器回复等测试设置。该压测工具对中小型集群有明显的效果，可以测出中小型网站的承受能力[75]。

本章使用质数求解函数模拟集群中的运算任务。质数求解函数中，其程序循环迭代相互独立，而且每个循环的执行时间不同，相对于矩阵相乘等函数更能达到分布式计算不均衡的效果。

4.5.2 实验设置

本章使用四台性能配置不同的主机作为服务器节点，结合其他配置的主机搭建了一个小型分布式集群，用以模拟集群测试环境。集群结构如图 4.3 所示。

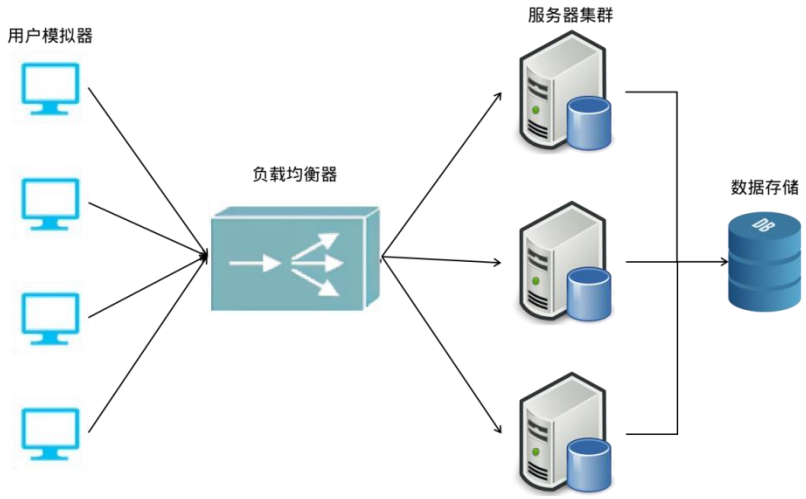


图 4.3 测试用模拟集群环境

用户模拟器运行 Webbench 压测工具模拟用户请求，向集群后端服务器发送访问请求；负载均衡器负责执行全局任务分配算法；后端服务器节点执行模拟运算任务函数，同时执行局部动态负载调度算法。

四台服务器节点的配置参数如表 4.1 所示。

表 4.1 测试服务器节点配置参数表

服务器	CPU 频率	内存容量	磁盘容量	网络带宽
用户模拟器	1G	256M	40G	8Mbps
负载均衡器	1G	256M	40G	8Mbps
后端服务器 1	450MHZ	128M	10G	1Mbps
后端服务器 2	450MHZ	256M	10*2G	1Mbps
后端服务器 3	1.7GHZ	256M	80G	2Mbps
后端服务器 4	1.3GHZ	128M	40G	3Mbps

4.5.3 评价指标与基准模型

1）评价指标

本章采用负载均衡度、用户请求响应时延和系统吞吐量三项指标，验证本章提出的基于预测自响应的集群综合负载均衡算法的性能。

（1）负载均衡度

集群负载均衡度用于衡量集群中各服务器的负载相差程度，其本质为集群中各服务器负载的均方差[76]，定义如下：

$$\varepsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(L_i - L_{avg})^2} \tag{45}$$

其中，L_i 为服务器 i 的实时负载，L_{avg} 为集群中所有服务器的平均实时负载，n 为集群中服务器总数。

（2）用户请求响应时延

集群负载均衡的最终目标是为用户提供高可用、高可靠和高拓展服务，高可用的一项重要体现便是用户请求响应时延的长短。因此，本章将用户请求响应时延作为衡量模型负载均衡效果的一项评价指标。本章取用户请求的平均响应时延以比较不同算法的性能体现。

（3）系统吞吐量

系统吞吐量为并发应用场景中另一项重要的性能评价指标，也是本章模型要解决的一个关键问题。本章统计在不同用户并发请求量下，各基准算法的系统吞吐量。

2）基准模型

为比较、验证本章模型的负载均衡效果，本文采用简单轮询法、加权轮询法、最小连接数法三种常用负载均衡算法以及以及 DLBLF[77]、DLBDS[78]两种动态负载均衡算法作为本文的基准对比模型。

4.5.4 实验结果对比

1）基于预测自响应的全局任务分配算法实验结果分析

本章选择以负载均衡度和用户请求响应时延两项评价指标来衡量基于预测自响应的全局任务分配算法的性能。

（1）负载均衡度

本章利用 Webbench 测试工具，以 30ms 的时间间隔向服务器集群发送用户请求，测试并记录 1 分钟内集群接收到 2000 条用户请求的负载均衡度。实验结果如图 4.4 所示。

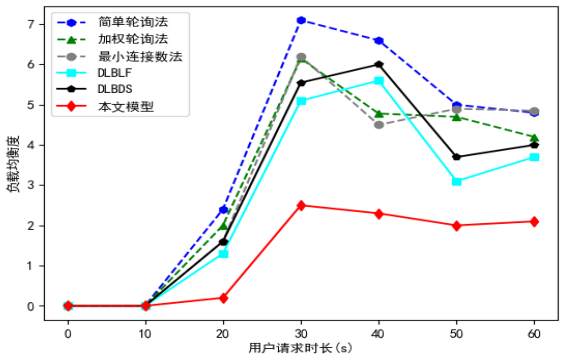


图 4.4 各模型的负载均衡度曲线图

从负载均衡效果来看，DLBLF 和 DLBDS 两种动态负载均衡算法整体而言是优于简单轮询法等传统负载均衡算法的，本文模型效果优于两种动态负载均衡算法。这是因为简单轮询、加权轮询算法其负载均衡策略主要基于服务器节点的顺序，最小连接数算法侧重服务器节点的用户请求连接数量，这些方法均未考虑节点负载的影响，所以其负载均衡效果不如 DLBLF 和 DLBDS 两种动态负载均衡算法。而本文模型在进行负载均衡决策时，基于用户请求和集群负载预测，同时兼顾节点性能和实时负载，所制定的用户请求分配决策更实时、精确。因此本文模型的负载均衡效果优于 DLBLF 和 DLBDS 两种动态负载均衡算法。

从负载均衡稳定性来看，随着用户请求数量的增长，简单轮询等传统方法与 DLBLF 等动态负载均衡算法的稳定性均不如本文模型。其原因在于，本文模型模型基于用户请求和集群负载预测，能够及时通过预测得到未来时刻的用户请求和服务器负载情况，并充分考虑二者之间的相互影响和响应关系，由此制定的分配方案更为精确、及时，减轻了负载迟滞效应造成的不稳定性。

（2）用户请求响应时延

本章利用 Webbench 测试工具，以 30ms 的时间间隔向服务器集群发送用户请求，测试并记录 1 分钟内集群接收到 2000 条用户请求的响应时延。统计分析所有用户请求的平均登录时延、最短登录时延和最长登录时延，实验结果如表 4.2 所示。

表 4.2 不同算法的用户请求响应时延结果

	平均响应时延（s）	最短响应时延（s）	最长响应时延（s）
简单轮询法	5.215	0.200	16.010
加权轮询法	5.178	0.205	15.880
最小连接数法	5.433	0.217	15.491
DLBLF	4.635	0.257	14.150
DLBDS	4.790	0.260	13.835
本文模型	4.114	0.276	12.126

从平均响应时延与最长响应时延来看，DLBLF 和 DLBDS 两种动态负载均衡算法以及本文模型均优于简单轮询法等传统负载均衡算法。其原因在于基于负载的均衡算法在进行用户请求分配时更多的基于集群负载，其分配策略的有效性更好。故，从整体来看，能够更好地协调分配集群负载的算法其用户请求分配更合理，集群负载分布更均衡。由此，其服务器任务运行效率更高，对新用户请求的接收速度也更快。

从最短响应时延来看，相对于简单轮询法等传统算法，DLBLF 和 DLBDS 两种动态负载均衡算法以及本文模型的最短响应延迟略长。其原因在于，最短响应时延一般发生在用户请求数量不多的测试初期。基于集群负载的算法其在制定用户请求分配方案时，需要调动一些负载计算方法，而这些是需要时间开销的。因此从最短响应时延来看，简单轮询等传统算法凭借其简单的用户请求分配策略，能够实现较好的最短响应延迟。但随着用户请求的增多，其最短响应

时延较短的优势也会消失。

2）基于集群服务器自索取的局部动态负载调度算法实验结果分析

本章采用四台不同配置的主机作为集群后端服务器节点，其性能参数如表 4.3 所示。使用质数求解函数库模拟服务器节点中运行的计算任务，通过函数的参数设置，模拟系统负载较重、负载正常和负载较轻等三种实验测试场景。测试结果如表 4.3 和图 4.5 所示。

表 4.3 各服务器节点在不同负载条件下的运行任务数

	负载较轻	负载正常	负载较重
服务器 1	20	38	45
服务器 2	25	40	50
服务器 3	80	150	190
服务器 4	40	95	120
任务总数	165	323	405

从表 4.3 和图 4.5 中可以看出，本章提出的基于集群服务器自索取的局部动态负载调度算法能够根据任务总数、集群负载和节点自身性能，根据集群动态调度方案，动态调整不同节点之间的任务运行数，实现各服务器节点任务分布均衡。

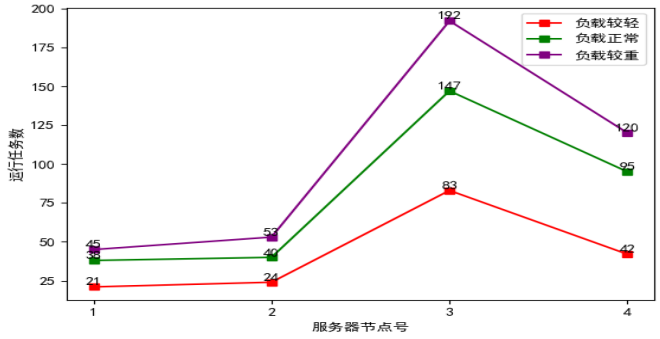


图 4.5 各节点任务数量分布

3）基于预测自响应的集群综合负载均衡算法实验结果分析

本章采用系统吞吐量作为衡量综合全局任务分配和局部负载调度的基于预测自响应的集群综合负载均衡算法性能的评价指标。实验中使用 Webbench 测试工具分别模拟 200、400、...1800 个并发用户请求，对集群进行请求测试，每次运行 10 分钟。实验结果如图 4.6 所示。

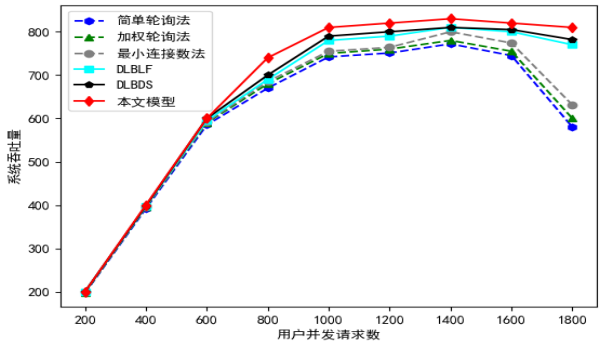


图 4.6 各模型系统吞吐量曲线图

从图 4.6 可以看出，当用户请求并发总数超过 1400 时，简单轮询法等传统负载均衡算法的系统吞吐量开始呈现明显下降趋势；但基于负载调度的 DLBLF 和 DLBDS 算法的系统吞吐量虽略有下降，但基本保持平缓趋势。整体来看，本章提出的基于预测自响应的集群综合负载均衡算法在系统吞吐量方面的性能是优于上述基准模型的。

4.5.5 参数设置

本章模型涉及到的参数较多，基于预测自响应的全局任务分配方法主要包括计算服务器自响应实时负载因子、服务器的动态负载因子和服务器综合负载等指标的权值参数，基于集群服务器自索取的局部动态负载调度方法主要包括服务器节点的负载上限和负载下限等参数。本节选取了其中对模型负载均衡效果影响较大的基于预测自响应的全局任务分配方法中计算服务器综合负载时动态负载和静态负载的权值以及基于集群服务器自索取的局部动态负载调度方法的服务器节点负载上下限参数进行实验分析。

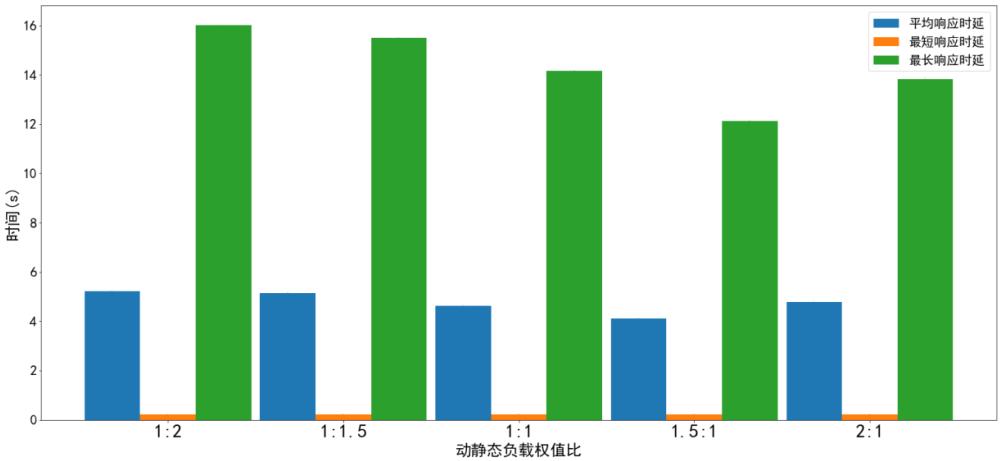


图 4.7 不同参数设置下用户请求响应时延

图 4.7 展示了基于预测自响应的全局任务分配方法中计算服务器综合负载时动态负载和静态负载的权值的不同设置对用户请求响应时延的影响。其中，动态负载和静态负载的权值比例

为 1.5:1 时用户请求响应的平均时延和最大时延效果最好。这表明该比例下，模型在进行全局任务分配时既能根据服务器节点的动态实时负载对用户请求做出合理分配，又能参考服务器节点的静态性能。

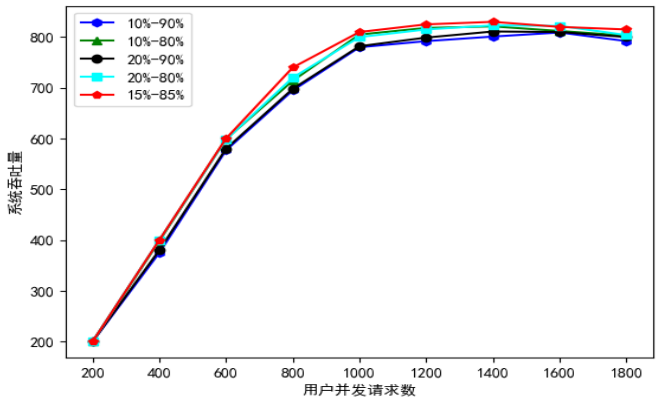


图 4.8 不同参数设置下系统吞吐量

图 4.8 展示了基于集群服务器自索取的局部动态负载调度方法中不同服务器节点负载上下限参数设置对系统整体吞吐量的影响。可以看到，集群负载调度方案将负载上下限分别设置为 15%和 85%时，系统整体吞吐量可以实现最大值。这意味着，该负载上下限设置可以保证系统在进行局部动态负载调度和任务迁移时，其任务迁移依据更符合集群服务器节点的负载现状，保证任务迁移的有效性和准确定。避免了资源浪费，同时防止服务器节点负载过重，实现集群整体较高的系统吞吐量。

4.6 本章小结

本章针从集群系统的全局任务分配和局部动态负载调度两个层面，分别就集群存在的负载均衡策略的实时性与准确性不足，以及集群负载均衡决策在局部负载调度方面存在的局部失衡等问题，充分分析集群全局任务分配和局部负载调度机制并结合第三章提出的用户请求和集群负载预测技术，提出了基于预测自响应的集群综合负载均衡算法。该算法在全局任务分配层面借助基于预测自响应的全局任务分配方法，双时序流量预测的基础上，挖掘实时用户请求、实时集群负载与预测用户请求、预测集群负载以及服务器性能之间的相互作用与响应关系，建立合理的全局任务分配模型。提高了全局任务分配的准确性和实时性。在局部负载调度层面，借助基于集群服务器自索取的局部动态负载调度方法，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载，实现了集群局部负载均衡，同时减少了服务器集群整体资源消耗。最后通过搭建集群环境模拟对本章模型的性能进行了实验验证。

第五章 基于双时序流量预测的自响应动态负载均衡集群系统实现

准确、高效、实用的集群负载均衡系统能够为高并发用户请求提供高可用、高可靠服务，一个完整、流畅的负载均衡集群系统可以极大提高用户使用体验。为保证用户服务质量的同时，改善并提高用户使用体验，本章设计并实现了负载均衡系统的软件应用端，并详细说明了该负载均衡系统的具体实现方式。通过集群负载均衡度实验和时间开销测试对系统性能进行评估。

该系统主要包括数据输入与预处理、用户请求和集群负载双时序流量预测、全局和局部集群综合负载均衡、任务分配和负载调度结果可视化等功能。在本系统的实现过程中，用户请求和集群负载双时序流量预测功能主要通过 Python 开发实现，全局和局部集群综合负载均衡功能主要通过 C++开发实现，可视化通过 Qt 开发完成。

5.1 系统开发环境介绍

该集群系统主要包括两个部分，分别为 Python 以及 C++后台和 Qt 前端，系统实现所需的软件、硬件环境具体如下。

（1）系统开发的硬件环境

操作系统：ubuntu18.04(64 bit)

处理器：Intel Core i7-8700

内存：32.00GB

硬盘：2.0T HDD

GPU：NVIDIA GeForce 2080 Ti

SSD：970 EVO Plus 500G

（2）系统开发的软件环境

开发环境：ubuntu18.04 (64 bit)

开发语言：Python3.7、C++11、Qt

开发工具：Pycharm、CLion、Qt Creator

5.2 数据结构设计

本节主要介绍该系统中双时序流量预测模块和集群综合负载均衡涉及的数据结构，这两个模块分别由 Python 和 C++实现。

5.2.1 双时序流量预测模块数据结构

双时序流量预测模块中涉及到的数据包括用户请求流量数据和集群负载数据，分别来自 google-cluster-trace-v2011 和 alibaba-cluster-trace-v2018 两个公开集群数据集。本系统在数据预处

理过程中将其中的数据进行整理，采用更高效的数据结构进行存储。

表 5.1 展示了用户请求数据的数据结构，其记录了用户请求任务在进行任务请求期间所需要的 CPU、内存、磁盘等资源负载数据。具体而言，该数据结构包括时间戳、用户请求任务 ID、服务器节点 ID、任务优先级、所需 CPU 负载、所需内存负载以及所需磁盘负载等等[64]。每个用户请求持续的时长不等，因此每个用户请求会产生多条数据记录，为区分不同时刻、不同用户的访问请求，本章使用时间戳和用户请求任务 ID 对数据样本进行标识。

表 5.1 用户请求数据结构

数据变量名	数据类型	数据项	长度
timestamp	String	时间戳	/
taskID	String	用户请求任务 ID	/
machineID	String	服务器节点 ID	/
schedulingClass	int	任务优先级	8
requestCPU	float	所需 CPU 负载	8
requestRAM	float	所需内存负载	8
requestDisk	float	所需磁盘负载	8
requestNet	float	所需网络带宽	8

表 5.2 为集群负载数据的数据结构，该数据结构对服务器集群中各节点的负载记录进行了详细说明。该数据结构服务器节点 ID、时间戳、CPU 利用率、内存利用率、磁盘利用率等等[65]。集群负载数据的时间窗口划分方式与数据处理方式与用户请求数据相同。

表 5.2 集群负载数据结构

数据变量名	数据类型	数据项	长度
machineID	String	服务器节点 ID	/
timestamp	String	时间戳	/
usageCPU	float	CPU 利用率	8
usageMem	float	内存利用率	8
usageDisk	float	磁盘利用率	8
usageNet	float	网络带宽利用率	8

5.2.2 集群综合负载均衡模块数据结构

集群综合负载均衡模块中局部动态负载调度算法涉及集群超载名单、节点运行任务表和节点调度任务表共三个表单，各表详细信息如下所示。

表 5.3 为集群超载名单，该表记录集群当前各服务器节点中发生超载的用户任务。具体而言，该数据结构包括记录 ID、超载任务 ID、当前负载和任务所属节点 ID 等信息。

表 5.3 集群超载名单数据结构

数据变量名	数据类型	数据项	长度
infoID	String	记录 ID	/
overloadTaskID	String	超载任务 ID	/
currentLoad	double	当前负载	8
machineID	double	任务所属节点 ID	8

表 5.4 为运行任务表，该表记录当前服务器节点上运行的用户任务，用户任务记录其所属节点编号，同时记录每个任务占用的负载量。具体而言，该数据结构包括节点 ID、用户任务 ID、当前消耗负载、调入时间和已运行时长等信息。

表 5.4 运行任务表数据结构

数据变量名	数据类型	数据项	长度
machineID	String	节点 ID	/
userTaskID	String	用户任务 ID	/
currentLoad	double	当前消耗负载	8
callInTime	String	调入时间	/
runTime	double	已运行时长	8

表 5.5 为调度任务表，该表记录当前服务器节点上待迁出并调度至其他节点的用户任务。具体而言，该数据结构包括节点 ID、用户任务 ID、当前消耗负载、调入时间和已等待时长。

表 5.5 调度任务表数据结构

数据变量名	数据类型	数据项	长度
machineID	String	节点 ID	/
userTaskID	String	用户任务 ID	/
currentLoad	double	当前消耗负载	8
callInTime	String	调入时间	/
waitedTime	double	已等待时长	8

5.3 核心功能模块的实现

基于双时序流量预测的自响应动态负载均衡集群系统由数据预处理模块、双时序流量预测模块、集群综合负载均衡模块、显示与控制模块构成。其中，双时序流量预测模块基于长短时特征融合算法实现、集群综合负载均衡模块基于全局任务分配与局部动态负载调度算法实现。对于数据预处理模块、双时序流量预测模块和集群综合负载均衡模块三个模块，本节分别展示了其对应的 UML 类图并进行了详细说明。

5.3.1 数据预处理模块

1）数据预处理模块模块类图

数据预处理模块实现的主要步骤有缺失值处理、数据归一化处理与数据切分处理等。下面

对各步骤的核心功能进行详细阐述，其 UML 类图如图 5.1 所示。

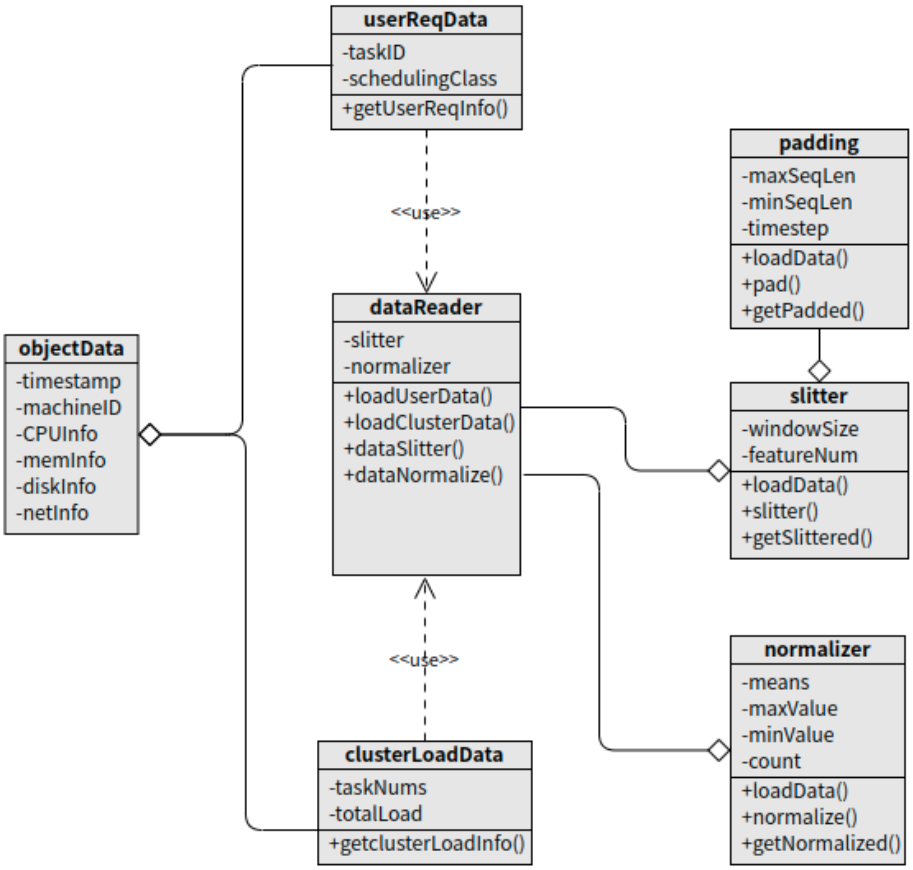
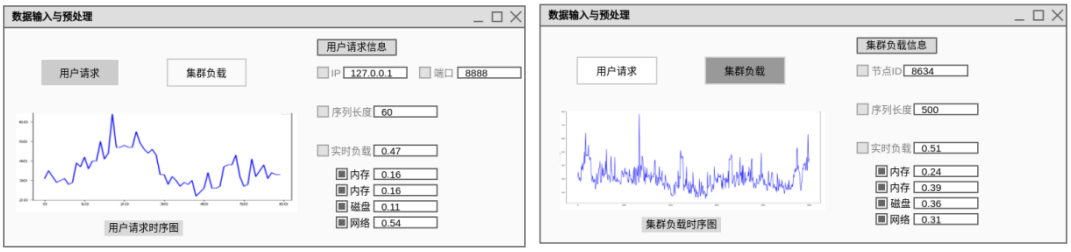


图 5.1 数据预处理模块 UML 类图

图 5.1 为数据预处理模块 UML 类图，其中核心类为 Padding 类、Slitter 类和 Normalizer 类，可以完成对数据的缺失值填充、归一化、数据切分等处理。objectData 类为用户请求数据和集群负载数据共有字段；userReqData 和 clusterLoadData 继承自 objectData 类，分别表示用户请求和集群负载数据；dataReader 类为输入数据读取类，它不仅对数据进行读取，并且调用数据处理类 Padding 类、Slitter 类和 Normalizer 类中的方法，对数据进行预处理。

2）数据预处理模块模块界面展示

数据载入与预处理功能的界面如图 5.2 所示。界面左侧为用户请求数据和集群负载数据的载入按钮。点击这两个按钮分别可以在界面左下方和右侧界面显示用户请求数据和集群负载数据的详情。具体而言，界面左下方可显示载入时序数据的时间-数值曲线图；界面右侧可显示载入时序数据的基础信息。以集群负载数据为例，包含节点 ID、实时负载、时序序列长度和时序数据特征总数等信息。完成数据载入和前端界面显示的同时，后端程序会同时启动数据预处理操作。



（a）用户请求数据输入　　（b）集群负载数据输入

图 5.2 数据输入与预处理模块界面图

5.3.2 双时序流量预测模块

1）双时序流量预测模块模块类图

双时序流量预测模块实现的主要步骤有用户请求数据聚类、集群负载多变量联合特征选择、短时特征提取、长时特征提取、特征加权融合、解码与输出等处理。下面对各步骤的核心功能进行详细阐述，其 UML 类图如图 5.3 所示。

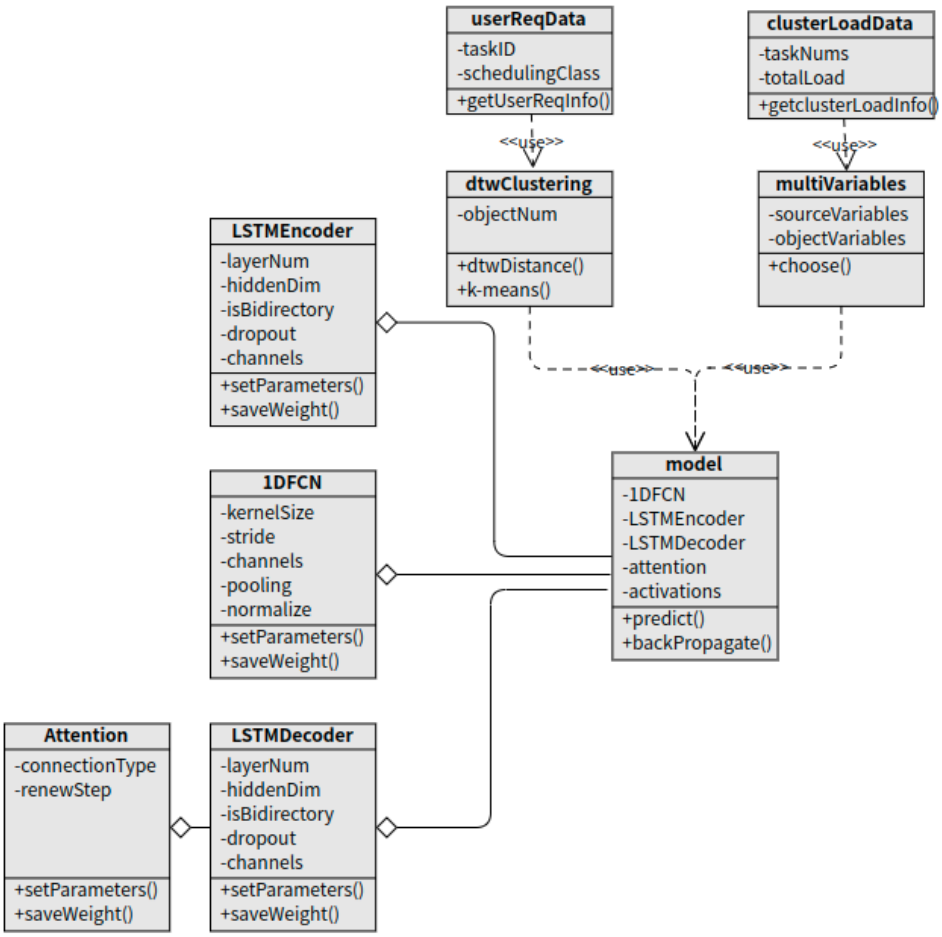
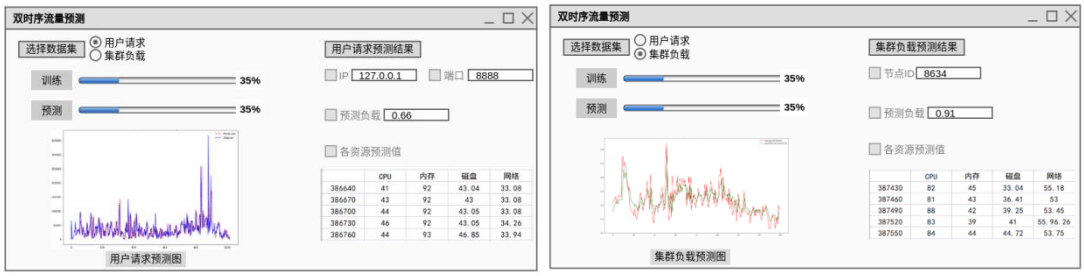


图 5.3 双时序流量预测模块 UML 类图

图 5.3 为双时序流量预测模块 UML 类图，其中核心类为 1DFCN 类、LSTMEncoder 类、LSTMDecoder 类、Attention 类和 model 类。在进行时序数据长短时特征提取之前，先由 dtwClustering 类和 multiVariables 类分别对用户请求数据和集群负载数据进行定向处理。1DFCN 类负责对时序数据提取短时特征，其中 kernelSize 设置卷积核大小，stride 参数设置卷积步长；LSTMEncoder 类对时序数据进行尝试特征提取和挖局，其中 channels 定义 LSTM 输入数据维度，layerNum 定义层数；LSTMDecoder 负责对加权长短时特征进行解码处理；Attention 辅助 LSTMDecoder 类对解码过程进行加权处理；model 类为核心类，其整合上述几个工具类，统一数据特征提取和处理流程。

2）双时序流量预测模块模块界面展示

双时序流量预测功能的界面如图 5.4 所示，该界面划分为左右两部分。界面左侧为模型训练和预测功能区，同时显示时序预测的曲线图；界面右侧为时序数据的基础信息和预测结果详情。具体而言，界面左上方为有模型训练和预测按钮，点击训练按钮，开始进行模型训练，同时训练进度条会实时更新，训练完毕会在进度条右侧提示训练完成；点击预测按钮，模型开始进行时序数据预测，同时将预测结果的曲线图显示在下方；同时界面右侧也会实时显示预测结果的响应，包括预测时刻，及其对应的 CPU、内存、磁盘等资源负载值。



（a）用户请求数据预测　　（b）集群负载数据预测

图 5.4 双时序流量预测模块界面图

5.3.3 集群综合负载均衡模块

1）集群综合负载均衡模块模块类图

集群综合负载均衡模块实现用户请求全局任务分配和局部动态负载调度两个子模块。其中局部动态负载调度子模块的主要步骤有集群超载任务管理、节点运行任务管理和节点调度任务管理。下面对局部动态负载调度子模块各步骤的核心功能进行详细阐述，其 UML 类图如图 5.5 所示。

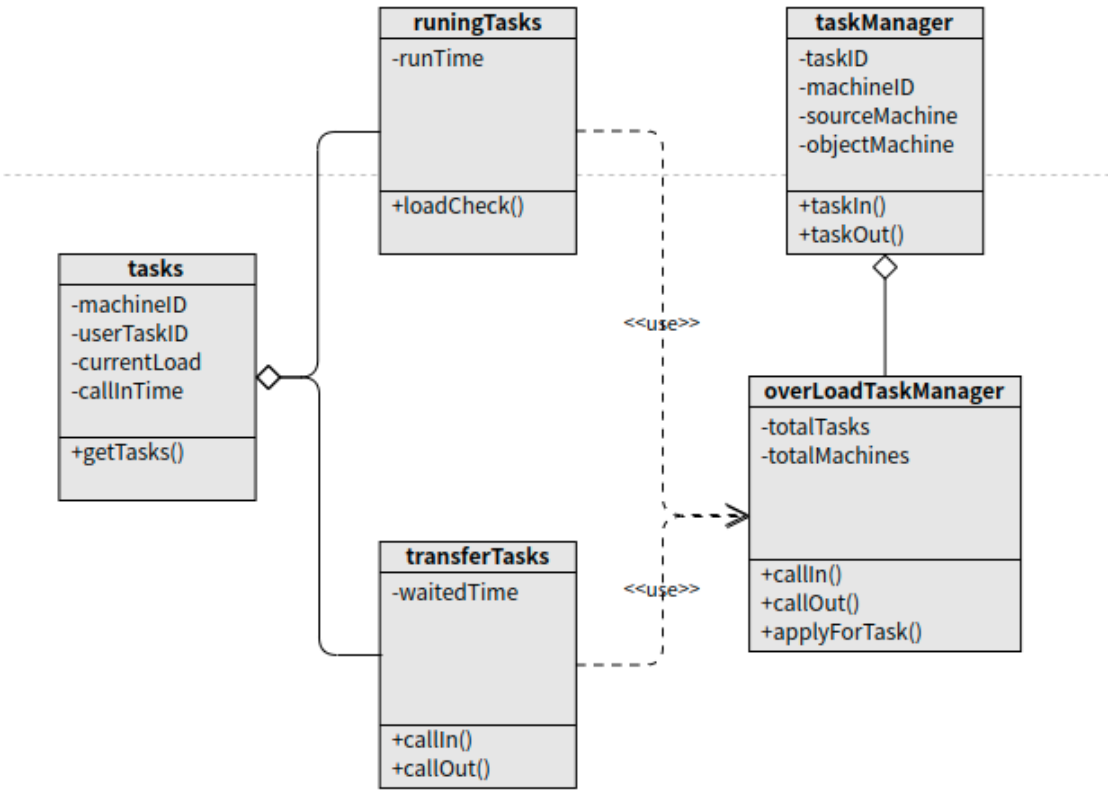


图 5.5 局部动态负载调度子模块 UML 类图

图 5.5 为局部动态负载调度子模块 UML 类图，其中核心类为 runningTasks 类、transferTasks 类和 overLoadTaskManager 类。runningTasks 类、transferTasks 类共同继承自 tasks 公共类，其含有的主要变量有 machineID、userTaskID、currentLoad 和 callInTime 等表示用户请求数据和集群负载数据的特征变量。runningTasks 类、transferTasks 类为两个运行在集群后端节点的动态工具类，其主要对节点中运行的用户任务和待调出的用户任务进行动态管理；overLoadTaskManager 类为核心类，其运行在集群全局，对所有后端节点中超载的任务进行统一管理和调度。其中，taskManager 类为 overLoadTaskManager 类的继承类，负责基础任务管理方法的细节实现。

2）集群综合负载均衡模块模块界面展示

双时序流量预测功能的界面如图 5.6 所示，该界面划分为左右两个功能分区。界面左侧为全局任务分配功能区，界面右侧为局部动态负载调度功能区。界面左上方包含节点综合负载计算和节点综合权值计算两个按钮，界面左侧中间位置为计算得到的集群综合权值序列，界面左下方为全局任务分配的目标节点。界面右侧主要是两个与局部负载调度相关的信息表格，右上方为集群超载名单表，下方为局部任务调度流向表。具体而言，在全局任务分配功能区中，依次点击节点综合负载计算和节点综合权值计算两个按钮，对集群中所有后端节点进行节点综合负载计算，在完成排序之后，会在左侧界面中间位置显示排序后的集群综合权值序列，同时在

左侧界面最下方显示用户请求的最终分配节点。在局部负载调度功能区中，不需要进行功能按钮的操作，右侧界面会实时显示后端局部任务调度程序的运行结果，具体结果以超载名单和任务调度流量表的形式展示。

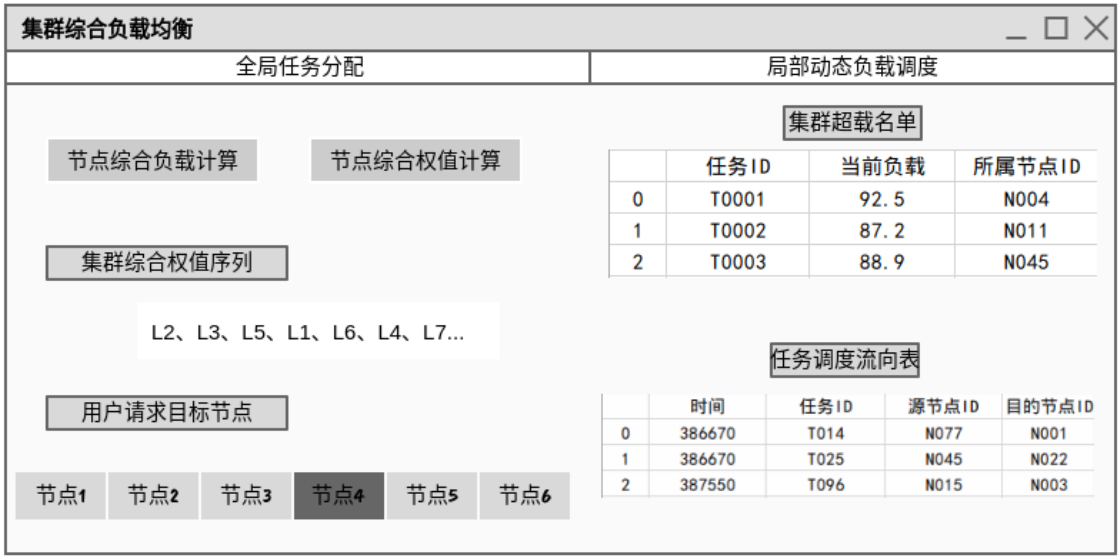


图 5.6 集群综合负载均衡模块界面图

5.4 系统评估与分析

本节在集群负载均衡准确度性和用户请求响应时延方面设计并完成了性能分析实验，确保系统在实验数据集上进行用户请求分配的测试结果满足用户要求。

5.4.1 集群负载均衡准确度分析

集群接收到用户请求后，首先对用户请求流量数据和集群负载流量数据进行缺失值处理、归一化处理、数据切分等数据预处理操作。然后由双时序流量预测阶段进行用户请求和集群负载两种时序流量的预测任务，分析用户请求和集群负载两种时序数据的特征，预测未来一段时间内两种流量的变化趋势，为任务分配与负载调度阶段提供方案制定依据。同时，集群综合负载调度模块根据用户请求和集群负载预测值为该用户请求制定任务分配方案，同时负责协调集群中局部相邻服务器节点之间现有任务分配关系，平衡各服务器节点之间的负载，减少服务器集群整体资源消耗。

为了评估集群系统的整体负载均衡度，避免特定用户造成的实验误差，本文对 Webbench 中所有的用户请求测试集样本进行了测试，计算集群的整体负载均衡度来评判系统性能，其计算公式为：

$$\varepsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(L_i - L_{avg})^2} \tag{46}$$

其中，L_i为服务器i的实时负载，L_{avg}为集群中所有服务器的平均实时负载，n为集群中服务器总数。

本实验采用不同长度的集群负载历史时长来测试系统的长短时预测鲁棒性[79]，实验结果如表 5.6 所示：

表 5.6 系统负载均衡度性能评估结果

时间长度 / 评价指标	3 天	2 天	1 天	12 小时	6 小时	3 小时	1 小时	0.5 小时	0.1 小时
负载均衡度	5.61	5.42	4.90	4.88	4.67	4.54	4.79	4.86	5.11

表 5.4 表明，当集群负载历史时长为最长 3 天时，集群综合负载均衡度为 5.61；当集群负载历史时长为最短 0.1 小时时，集群综合负载均衡度为 5.11；当集群负载历史时长为 3 小时时，集群综合负载均衡度为 4.54，实现最好负载效果。可以看出，最大负载均衡度与最低负载均衡度相差仅为 1.07，这说明本文模型使用较长和较短时间的数据都可以实现较好的用户任务分配和集群负载调度，表明本文模型具有很好的鲁棒性，在集群负载均衡度方面可以满足系统设计要求。

5.4.2 时间性能分析

本节从系统运行时间和全局任务分配模型运行时间两个方面对系统性能进行测试，图 5.7 为整个集群系统和基于预测自响应的全局任务分配模型的时间性能分析结果。从实验结果中可以看出，整个集群系统进行任务分配的效率要低于基于预测自响应的全局任务分配模型，这是因为系统包含接收与存储数据的功能，在进行模型诊断前，需要完成数据的预处理工作，包括缺失值填充、归一化和数据标准化等操作，需要额外的运行时间；同时，整个集群系统还需要负责所有后端服务器之间的负载调度。本文的时间性能测试采用 Webbench 模拟数据集，记录分别使用不同时间长度的集群负载历史数据进行全局用户请求分配所需时间。从图中曲线可知，随着时间的增加，系统和模型的计算量增大，运行时间变长。当使用时间长度为获得较高诊断准确度的 3 天，即 72 小时时，系统和模型进行疾病诊断的耗费时间分别为 2.05 秒和 1.84 秒，对用户来说处于可接受范围内，满足系统设计的要求。

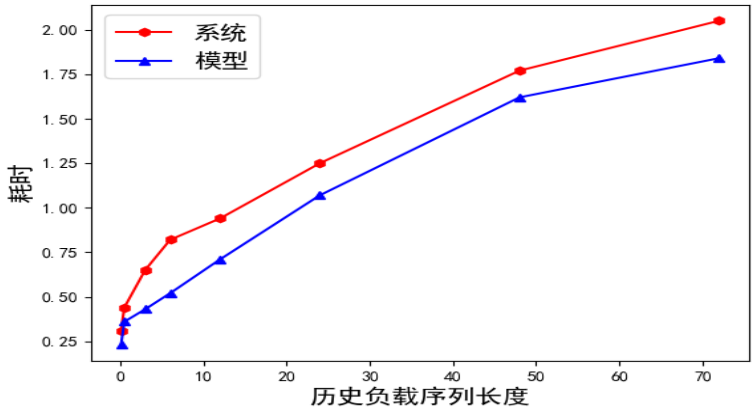


图 5.7 系统与模型时间性能评估结果

5.5 本章小结

本章详细介绍了基于双时序流量预测的自响应动态负载均衡集群系统的具体实现细节。首先介绍了系统开发环境，包括系统实现所需的硬件环境和软件环境；然后展示了系统中数据预处理模块、双时序流量预测模块和集群综合负载均衡模块的相关数据结构和前端界面；最后通过集群整体负载均衡度实验和时间开销测试对系统性能进行评估，完成系统性能测试。

第六章 总结与展望

6.1 论文研究工作总结

随着移动互联网技术的不断提高和移动终端设备的快速普及，移动端用户数量的剧增对服务器集群服务质量和性能提出了更高要求。众多科研工作者和互联网厂商对集群技术的广泛关注极大地推动了负载均衡等集群技术的发展。近年来，机器学习在流量预测方面的应用推动了集群负载均衡技术向着智能化、动态化方向发展。以网络流量和集群负载预测为代表的动态负载均衡技术取得了长足发展。然而，基于网络流量和集群负载等时序数据的多变性和集群服务的复杂性，仅仅借助流量预测技术无法实现准确、高效、完整的集群负载均衡服务。因此，如何更准确地实现网络流量和集群负载预测、兼顾集群整体均衡，保证集群服务的准确性、高效性和完整性，已成为当前研究中的热点问题。

本文从多个方面研究了基于双时序流量预测的自响应动态负载均衡技术，对基于加权长短时特征融合的双时序流量预测技术、基于预测自响应的全局任务分配算法和基于集群服务器自索取的局部动态负载调度算法进行调研，总结分析现有研究成果的优势和不足，在其基础上对集群负载均衡算法进行改进，并完成了集群动态负载均衡系统的初步实现。本文的具体研究内容如下：

（1）对基于双时序流量预测的自响应动态负载均衡技术进行了深层次的分析与研究，分别从集群应用、集群与集群技术和基于负载预测的集群技术发展等方面介绍了基于流量预测的集群负载均衡技术的国内外研究现状，对现有技术的优势和不足进行了分析。基于国内外研究现状的总结分析，确定了基于双时序流量预测的自响应动态负载均衡集群系统的功能和性能需求，并对系统物理结构、逻辑结构和实现流程进行设计。

（2）针对用户请求流量历史累积数据少，以及用户请求流量和集群负载流量共有的数据周期性差，预测无法同时兼顾短时、长时预测的问题，本文提出了一种基于加权长短时特征融合的双时序流量预测模型。首先对于用户请求数据，利用基于查询路径优化的 DTW 用户请求时序数据聚类算法对其进行初步分类，确定所属用户请求类别；对于集群负载数据，借助多变量联合特征选择技术其进行源特征和目的特征选择。然后利用基于注意力机制的加权长短时特征融合技术对时序数据进行短时与长时特征提取、长短时特征融合以及向量加权等处理，充分挖掘时序数据的长短时特征，实现高准确度的时序流量短期预测和长期预测。为基于预测自响应的全局任务分配模型提供可靠数据输入。

（3）在集群综合负载均衡方面，本文分别对全局任务分配和局部动态负载调度两类工作建立不同的处理机制。针对全局任务分配，本文提出了基于预测自响应的全局任务分配模型，该模型在双时序流量预测的基础上，挖掘实时用户请求、实时集群负载与预测用户请求、预测集

群负载以及服务器性能之间的相互作用与响应关系，建立合理的全局任务分配模型。实现准确、高效的全局任务分配，保证集群运行在较低负载均衡度。针对局部动态负载调度，本文提出了基于集群服务器自索取的局部动态负载调度模型，该模型借助基于集群服务器自索取的局部动态负载调度方法，协调局部相邻服务器节点之间的任务分配关系，平衡各服务器节点之间的负载实现了集群局部负载均衡，同时减少了服务器集群整体资源消耗。

（4）完成了基于双时序流量预测的自响应动态负载均衡系统的初步的实现和设计，系统采用了本文提出的基于加权长短时特征融合的双时序流量预测模型、基于预测自响应的全局任务分配模型和基于集群服务器自索取的局部动态负载调度模型，实现了集群动态负载均衡。保证用户请求准确、高效处理的同时，降低了集群系统整体负载均衡度，提高了系统吞吐量。

6.2 未来研究方向

本文初步实现了基于双时序流量预测的自响应动态负载均衡系统，本文提出的模型与现有其他集群负载均衡模型相比具有很好的性能。但是由于用户请求流量和集群负载等时序数据的多变性和集群服务的复杂性等特点，该模型在实际应用场景中的性能仍有待进一步研究。未来对于基于时序流量预测集群负载均衡技术的研究工作主要包括以下几个方面：

（1）由于实际应用场景中用户请求流量历史积累较少，数据周期性特征呈现不明显，本文针对用户请求流量分类提出的基于查询路径优化的 DTW 时序数据聚类分类方法通过聚类以期借助同一聚类类型的其他用户请求数据的特征，进而为本用户请求数据提供尽可能多的相关特征。基于用户请求类别的多样性和聚类效果不足，仅通过该聚类方法无法充分挖掘用户请求流量的类别信息。因此，如何更好地对聚类模型进行优化，充分挖掘实际应用场景中用户请求数据，是提高用户请求流量预测准确度的一个重要研究方向[80]。

（2）本文提出的基于加权长短时特征融合的双时序流量预测模型在保证长时预测能力的同时，也能实现较好的短时预测能力，兼顾了长短时负载预测能力。然而，LSTM 编码-解码器在进行特征提取和解码时，由于 LSTM 的多种门控机制，每个神经单元的计算量都很大；当 LSTM 的时间跨度较大，网络深度较长时，其特征提取耗时较大[81]。未来将针对这一问题，就在保证预测准确度的前提下提高特征提取和解码效率这一工作继续进行研究。

（3）本文初步实现的基于预测自响应的集群综合负载均衡模型，基于软件开发技术，对全局任务分配和局部动态负载进行了软件实现。但由于开发过程中数据结构设计、程序架构等方面存在的不足，模型运行过程中存在响应延迟等问题，降低了模型对用户请求的响应速度和集群系统运行效率。在后续工作中，可以通过优化程序数据解结构设计和代码流程，减少程序运行对模型运行效率的影响，实现实时响应与处理，提升用户体验。

参考文献

[1] 张利刚.移动互联网技术的发展现状及未来发展趋势[J]. 电子元器件与信息技术, 2020, 4(09): 47-48.

[2] Bano, Saima & Khan, Naeem. A Survey of Data Clustering Methods. International Journal of Advanced Science and Technology, 2018:113-14.

[3] Xu, D., Tian, Y. A Comprehensive Survey of Clustering Algorithms. Ann. Data. Sci. 2015, (2):165–193.

[4] Omran, Mahamed & Engelbrecht, Andries & Salman, Ayed. An overview of clustering methods. Intell. Data Anal, 2007, (11):583-605.

[5] Mann, Amandeep K. and Navneet Kaur. Survey Paper on Clustering Techniques, 2013.

[6] Werstein, Paul & Situ, Hailing & Huang, Zhiyi. Load Balancing in a Cluster Computer, 2007, (10):569 - 577.

[7] Ali, Syed Amjad and Cüneyt Sevgi. “Energy Load Balancing for Fixed Clustering in Wireless Sensor Networks.” 2012 5th International Conference on New Technologies, Mobility and Security (NTMS), 2012: 1-5.

[8] Shanbhog, Manjula & Kalpna. Load balancing research paper, 2020: 10-35.

[9] Imiełowski, Andrzej. Load balancing algorithms in cluster systems. ITM Web of Conferences，2018:10-15.

[10] Li, B., Shang, J., Dong, M., & He, Y. Research and Application of Server Cluster Load Balancing Technology. 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2020, (1):2622-2625.

[11] Khoulalene, N., Bouallouche-Medjkoune, L., Aissani, D. et al. Clustering with Load Balancing-Based Routing Protocol for Wireless Sensor Networks. Wireless Pers Commun, 2018, (103):2155–2175.

[12] Cheng, T., Chin, PJ., Cha, K. et al. Profiling the BLAST bioinformatics application for load balancing on high-performance computing clusters. BMC Bioinformatics, 2022:23-544.

[13] Enric, Fontdecaba., Antonio, González., Jesús, Labarta. Load Balancing in a Network Flow Optimization Code, 1995:214-222.

[14] Patil, Sharada & Gopal, A. Cluster performance evaluation using load balancing algorithm, 2013:104-108.

[15] Akperi, Brian & Matthews, Peter. Analysis of clustering techniques on load profiles for electrical

distribution, 2014:1142-1149.

[16] Alankar B, Sharma G, Kaur H, Valverde R, Chang V. Experimental Setup for Investigating the Efficient Load Balancing Algorithms on Virtual Cloud. Sensors (Basel), 2020(24):7342.

[17] Afzal, S., Kavitha, G. Load balancing in cloud computing – A hierarchical taxonomical classification. J Cloud Comp, 2019.

[18] Pattanaik, L.N., Gupta, A.K., Surin, S., Singh, A.P. Application of Clustering Algorithms for Locating Distribution Centers of Logistics System. In: Mahapatra, R.P., Panigrahi, B.K., Kaushik, B.K., Roy, S. (eds) Proceedings of 6th International Conference on Recent Trends in Computing. Lecture Notes in Networks and Systems, 2021(177).

[19] Satyanarayana KV, Rao NT, Bhattacharyya D, Hu YC. Identifying the presence of bacteria on digital images by using asymmetric distribution with k-means clustering algorithm. Multidimens Syst Signal Process, 2022, 33(2):301-326.

[20] Bosque, J.L., Toharia, P., Robles, O.D. et al. A load index and load balancing algorithm for heterogeneous clusters. J Supercomput, 2013(65):1104–1113.

[21] Jiang, Xiaoming et al. 'Cluster Load Balancing Algorithm Based on Dynamic Consistent Hash', 2021:4461–4468.

[22] H. Rai, S. K. Ojha and A. Nazarov, "Comparison Study of Load Balancing Algorithm," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020:852-856.

[23] S.B. Kshama and K.R. Shobha. Load balancing algorithms with cluster in cloud environment. Int. J. Commun. Netw. Distrib, 2022:679–703.

[24] Yu, A., Yang, S. Research on web server cluster load balancing algorithm in web education system, 2020(76):3364–3373.

[25] Kapoor, Surbhi & Dabas, Chetna. Cluster Based Load Balancing in Cloud Computing, 2015.

[26] Tasneem, R., Jabbar, M.A. An Insight into Load Balancing in Cloud Computing. In: Qian, Z., Jabbar, M., Li, X. (eds) Proceeding of 2021 International Conference on Wireless Communications, Networking and Applications. WCNA 2021. Lecture Notes in Electrical Engineering. Springer, Singapore, 2022.

[27] Mesbahi, Mohammad Reza and Amir Masoud Rahmani. "Load Balancing in Cloud Computing: A State of the Art Survey." International Journal of Modern Education and Computer Science, 2016(8):64-78.

[28] Dhurandher, Sanjay & Obaidat, Mohammad & Woungang, Isaac & Agarwal, Pragya & Gupta,

Abhishek & Gupta, Prateek. A cluster-based load balancing algorithm in cloud computing, 2014:2921-2925.

[29] S. Kapoor and C. Dabas, "Cluster based load balancing in cloud computing," 2015 Eighth International Conference on Contemporary Computing (IC3), 2015:76-81.

[30] Nyaramneni, S., Saifulla, M.A., Shareef, S.M. ARIMA for Traffic Load Prediction in Software Defined Networks. In: Suma, V., Bouhmala, N., Wang, H. (eds) Evolutionary Computing and Mobile Sustainable Networks. Lecture Notes on Data Engineering and Communications Technologies, vol 53. Springer, Singapore, 2021.

[31] Lv, Feng & Kang, Fengning & Sun, Hao. The Predictive Method of Power Load Based on SVM. TELKOMNIKA Indonesian Journal of Electrical Engineering, 2014, 12.

[32] Abhishek Gupta, Ravi Malik. ELECTRIC LOAD FORECASTING USING ANN, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT), 2013:1-2.

[33] Ren B, Huang C, Chen L, Mei S, An J, Liu X, Ma H. CLSTM-AR-Based Multi-Dimensional Feature Fusion for Multi-Energy Load Forecasting. Electronics, 2022, 11(21):3481.

[34] Zhu, Y., Zhang, W., Chen, Y. et al. A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment. J Wireless Com Network 2019, 2019, (274).

[35] Mrhari, Amine & Hadi, Youssef. Workload Prediction Using VMD and TCN in Cloud Computing. Journal of Advances in Information Technology, 2022, 13(3):284-289.

[36] Luo, Tao & Cao, Xudong & Li, Jin & Dong, Kun & Zhang, Rui & Wei, Xueliang. Multi-task prediction model based on ConvLSTM and encoder-decoder. Intelligent Data Analysis, 2021, 25:359-382.

[37] Shaik Dai Haleema. Short-Term Load Forecasting using Statistical Methods: A Case Study on Load Data, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT), 2020, 9.

[38] Guo M, Haque A, Huang D A, et al. Dynamic task prioritization for multitask learning[C]//Proceedings of the European Conference on Computer Vision (ECCV), 2018:270-287.

[39] Mesbahi, M.R., Rahmani, A.M. & Hosseinzadeh, M. Reliability and high availability in cloud computing environments: a reference roadmap. Hum. Cent. Comput. Inf. Sci, 2018:8-20.

[40] Snyder, B., Ringenberg, J., Green, R. et al. Evaluation and design of highly reliable and highly

utilized cloud computing systems, 2015:4-11.

[41] Shiv Shankar, Ashish Kumar Sharma, 2017, A Comparative Performance Analysis of Cloud, Cluster and Grid Computing over Network, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICADEMS, 2017, 5:3.

[42] Bhanu, L. & Narayana, Dr. Customer Loan Prediction Using Supervised Learning Technique. International Journal of Scientific and Research Publications (IJSRP), 2021, 11: 403-407.

[43] Lei Gao, Lu Wei, Jian Yang, Jinhong Li, "Trajectory Clustering in an Intersection by GDTW", Journal of Advanced Transportation, 2022: 23.

[44] Sparsh Verma , Dipankar Giri. Survey Paper of Cloud Computing, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT), 2022, 11:1.

[45] Ajay Kumar.A.H., Asha Gowda Karegowda. Applications of Cloud Computing: a Survey, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT), 2013:1-4.

[46] Sahi, S.K., & Dhaka, V. A survey paper on workload prediction requirements of cloud computing. 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2016:254-258.

[47] Huang, S., Zhang, Y., Peng, G. et al. MF-GCN-LSTM: a cloud-edge distributed framework for key positions prediction in grid projects, 2022, 11:9-15.

[48] Yadav, A., Kushwaha, S., Gupta, J., Saxena, D., Singh, A.K. A Survey of the Workload Forecasting Methods in Cloud Computing. In: Tomar, A., Malik, H., Kumar, P., Iqbal, A. (eds) Proceedings of 3rd International Conference on Machine Learning, Advances in Computing, Renewable Energy and Communication. Lecture Notes in Electrical Engineering, 2022:915.

[49] ManjunathC, R., Manaswini, C., & Bangar, N. A Survey on Load Prediction Techniques in Cloud Environment. International journal of engineering research and technology, 2018:2.

[50] Nguyen, Hoang Minh & Kalra, Gaurav & Kim, Daeyoung. Host load prediction in cloud computing using Long Short-Term Memory Encoder–Decoder. The Journal of Supercomputing, 2019:75.

[51] Song, B., Yu, Y., Zhou, Y. et al. Host load prediction with long short-term memory in cloud computing. J Supercomput, 2018, 74:6554–6568.

[52] Zhong, W., Zhuang, Y., Sun, J., & Gu, J. A load prediction model for cloud computing using PSO-based weighted wavelet support vector machine. Applied Intelligence, 2018, 48:4072 - 4083.

[53] Mrhari, Amine & Hadi, Youssef. (2022). Workload Prediction Using VMD and TCN in Cloud

Computing. Journal of Advances in Information Technology. 13. 284. 10.12720/jait.13.3.284-289.

[54] Mrhari, Amine & Hadi, Youssef. Workload Prediction Using VMD and TCN in Cloud Computing. Journal of Advances in Information Technology, 2022, 13:284-289.

[55] Javad Dogani, Farshad Khunjush, Mehdi Seydali,Host load prediction in cloud computing with Discrete Wavelet Transformation (DWT) and Bidirectional Gated Recurrent Unit (BiGRU) network,Computer Communications, 2023, 198:157-174.

[56] Arya, M.S., Deepa, R., Gandhi, J. Dynamic Time Warping-Based Technique for Predictive Analysis in Stock Market. In: Mahapatra, R.P., Panigrahi, B.K., Kaushik, B.K., Roy, S. (eds) Proceedings of 6th International Conference on Recent Trends in Computing. Lecture Notes in Networks and Systems, 2021, 177.

[57] Deriso, D., Boyd, S. A general optimization framework for dynamic time warping. Optim Eng, 2022.

[58] Javad Dogani, Farshad Khunjush, Mehdi Seydali,Host load prediction in cloud computing with Discrete Wavelet Transformation (DWT) and Bidirectional Gated Recurrent Unit (BiGRU) network,Computer Communications, 2023, 198:157-174.

[59] Zhang, Weishan & Li, Bo & Zhao, Dehai & Gong, Faming & Lu, Qinghua. Workload Prediction for Cloud Cluster Using a Recurrent Neural Network, 2016: 104-109.

[60] Nguyen, H.M., Kalra, G. & Kim, D. Host load prediction in cloud computing using Long Short-Term Memory Encoder–Decoder. J Supercomput, 2019, 75:7592–7605.

[61] Zheng Huang, Jiajun Peng, Huijuan Lian, Jie Guo, Weidong Qiu, "Deep Recurrent Model for Server Load and Performance Prediction in Data Center", Complexity, 2017: 10.

[62] Ciechulski, T.; Osowski, S. High Precision LSTM Model for Short-Time Load Forecasting in Power Systems. Energies 2021, 2021, 14:2983.

[63] Varshney, M., Singh, P. Optimizing nonlinear activation function for convolutional neural networks. SIViP, 2021, 15:1323–1330.

[64] Chen, Y., Ganapathi, A., Griffith, R., & Katz, R.H. Analysis and Lessons from a Publicly Available Google Cluster Trace, 2020.

[65] Congfeng, Jiang., Yitao, Qiu., Weisong, Shi., Ge, Zhefeng., Jiwei, Wang., Shenglei, Chen., Christophe, Cérin., Zujie, Ren., Guoyao, Xu., Jiangbin, Lin. Characterizing Co-located Workloads in Alibaba Cloud Datacenters. IEEE Transactions on Cloud Computing, 2021:1-1.

[66] Xin, L. Application of ARMA Time Series Model. Techniques of Automation and Applications, 2008.

[67] Abdullayeva, Fargana. Cloud Computing Virtual Machine Workload Prediction Method Based on Variational Autoencoder. International Journal of Systems and Software Security and Protection, 2021, 12:33-45.

[68] Chai, T., & Draxler, R.R. Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. Geoscientific Model Development, 2014, 7:1247-1250.

[69] Cheng, T., Chin, PJ., Cha, K. et al. Profiling the BLAST bioinformatics application for load balancing on high-performance computing clusters. BMC Bioinformatics, 2022, 23:544.

[70] Ruan, L., Bai, Y., Li, S. et al. Workload time series prediction in storage systems: a deep learning based approach. Cluster Comput, 2021.

[71] Renhai Feng, Yuanbiao Xue, Wei Wang, Meng Xiao. Saturated load forecasting based on clustering and logistic iterative regression. Electric Power Systems Research, 2022, 202:378-396.

[72] Zeqing Wu, Yunfei Mu, Shuai Deng, Yang Li,Spatial–temporal short-term load forecasting framework via K-shape time series clustering method and graph convolutional networks,Energy Reports, 2022, 8:752-766.

[73] A. Tokgöz and G. Ünal. A RNN based time series approach for forecasting turkish electricity load. 2018 26th Signal Processing and Communications Applications Conference (SIU), 2018:1-4.

[74] Zheng Huang, Jiajun Peng, Huijuan Lian, Jie Guo, Weidong Qiu. Deep Recurrent Model for Server Load and Performance Prediction in Data Center. Complexity, 2017:10.

[75] Sun, P., Zhang, Z., Jin, M., & Wang, K. Simulation of Switching Power Based on WEBENCH Tool. DEStech Transactions on Computer Science and Engineering, 2016.

[76] Camila Maione, Donald R. Nelson, Rommel Melgaço Barbosa. Research on social data by means of cluster analysis. Applied Computing and Informatics, 2019, 15:153-162.

[77] 曲乾聪,王俊. 基于负载反馈的分布式数字集群动态负载均衡算法[J]. 计算机应用研究, 2022, 39(02): 526-530.

[78] 陈志刚,许伟,曾志文. 一种基于预测的动态负载均衡模型及算法研究[J]. 计算机工程, 2004(23) :87-89.

[79] McLachlan GJ. Cluster analysis and related techniques in medical research. Stat Methods Med Res, 1992, 1(1):27-48.

[80] Camila Maione, Donald R. Nelson, Rommel Melgaço Barbosa. Research on social data by means of cluster analysis. AppliedComputing and Informatics, 2019, 15(2):153-162.

[81] Patel, E., Kushwaha, D.S. A hybrid CNN-LSTM model for predicting server load in cloud

computing. J Supercomput, 2022, 78:1–30.

致 谢

时光荏苒，岁月如梭。

美好的时光总是过得很快，不知不觉间已经度过三年，研究生生活即将划上句号。回首这三年，南京航空航天大学带给我很多很多。她传授我学识、技能与才华；开拓我的视野、思维与见识；给予我荣耀、掌声与自豪。在这里，在南航，在南京，我收获了前沿的科学知识，锻炼了科学的思维能力，培养了优秀的团队意识，也收获了许许多多的友情。在这里，我想对所有曾陪伴我、鼓励我、支持我、帮助我的人表示最真挚的感谢。

首先，我要感谢我的恩师顾晶晶老师。顾老师学识渊博、学术精湛、平易近人，是她带我走进机器学习的大门，给与我机会参与多项项目实践，开拓理论学习的视野，增强软件实践能力。在我实验和论文陷入瓶颈时，是她为我指点迷津；在我项目进展遇到困难时，是她为我耐心解答；在我生活中遇到囹圄时，是她为我排忧解难。是顾老师的谆谆教诲和传道解惑，我才得以在科研上顺利前行；是顾老师的平易近人和善良体贴，我才得以在一个温馨、活泼、快乐的实验室团队享受科研和学术生活之乐。感谢顾老师对我短暂研究生三年的教导和照顾，感谢顾老师的辛勤付出。

我要感谢南航的同学和朋友们。在这里，我结识了很多益友，他们在学习、工作和生活中给以我很多帮助。感谢 202 和 317 实验室的同门师兄弟，与你们一起度过的每一个日夜是我研究生学术生涯最难忘的时光。感谢同门冯晨在多个项目中对我及时雨般的帮助；感谢同门陈妍在实验中对我的开导和启发；感谢同门陈俊义在工作态度上对我的激励；感谢王秋红师姐、霍甜媛师姐、刘玉强师兄对我科研和工作的热心帮助；感谢文宝师弟同我在球场上度过的每一个午后、黄昏和夏夜；还要感谢郭小奉师弟和陈晨同门在秋招时对我的帮助。感谢我的室友刘京、张明在我研究生期间的陪伴与照顾。最后，还要感谢我的女朋友，给予我研究生期间生活和情感上的照料、鼓励和呵护，与我一起倾诉烦恼、分享快乐。

我要感谢我的家人。感谢你们的养育之恩，是你们一直以来为我提供学习、工作和生活上的一切帮助。感谢你们的爱护、教导、鼓励和支持。儿子不孝，在外求学多年，未能在二老身前尽奉养之孝。还望二老在今后的日子里开心工作、健康生活。

我还要感谢一下自己，感谢自己三年多来从未放弃。前路漫漫、任重道远，道阻且长、行则将至。

最后，由衷感谢参加论文评审和答辩的各位老师，感谢您提出的宝贵意见和建议。

在学期间的研究成果及发表的学术论文

攻读硕士学位期间发表（录用）论文情况

1.高自强，顾晶晶，A long-periodic and short-periodic time-series load forecasting method in highly-variable cloud computing scenarios ， International Academic Conference For Graduates,NUAA，2022.11，第一作者，已收录

攻读硕士学位期间专利申请情况

1.基于细粒度目标分类的多视角多目标识别方法，排名 2/2（已公开）

攻读硕士学位期间参加科研项目情况及获奖情况

1.2021 年全国研究生数学建模竞赛三等奖