南京大学

研 究 生 毕 业 论 文

（申请工程硕士学位）

论 文 题 目	面向新能源汽车BMS荷电状态估计的数据扩增技术
作 者 姓 名	倪烨
学科、专业名称	工程硕士（软件工程领域）
研 究 方 向	软件工程
指 导 教 师	陈振宇 教授

2022 年 05 月 20 日

学　　号：**MF20320106**

论文答辩日期：**2022 年 05 月 20 日**

指 导 教 师：　　**（签字）**

Data Augmentation Technology for BMS State-of-Charge Estimation of New Energy Vehicles

By

Ye Ni

Supervised by

Professor Zhenyu Chen

A Thesis
Submitted to the Software Institute
and the Graduate School
of Nanjing University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Engineering

Software Institute

May 2022

学 位 论 文 原 创 性 声 明

任何收存和保管本论文的单位和个人，未经作者本人授权，不得将本论文转借他人并复印、抄录、拍照或以任何方式传播，否则，引起有碍作者著作权益的问题，将可能承担法律责任。

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不句含其他个人或集体已经发表或撰写的作品成果。本文所引用的重要文献，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：_____

日期：　　年　　月　　日

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：面向新能源汽车 BMS 荷电状态估计的数据扩增技术

工程硕士（软件工程领域） 专业 2020 级硕士生姓名： 倪烨

指导教师（姓名、职称）： 陈振宇 教授

摘 要

全球变暖和能源危机是当今社会面临的两个重要问题，节能减排是最根本的解决办法。现如今各国开始大力发展新能源汽车来减少对化石燃料的依赖并减少温室气体的排放。新能源汽车以电池阵列作为动力源，通过电池管理系统（BMS）实现电池阵列的监测和控制。荷电状态（SOC）估计是 BMS 的核心功能，通过 SOC 估计和均衡策略，BMS 可以延长电池的寿命、防止过充和过放等安全事故的发生。然而 SOC 估计的方法尚不成熟，精确估计 SOC 面临很多挑战。得益于深度学习技术的发展和车载算力的提升，越来越多的厂商采用深度学习解决方案并取得了不错的成绩。

目前主流的模型质量保障工作还在使用黑盒测试方法，导致模型准确性和泛化程度的测试工作完全由测试数据决定，模型的不可解释性又进一步加剧了这种现象，因此，多样的测试数据集对模型质量保障工作至关重要。实车采集的测试数据不平衡性非常明显，数据中放电样本数量要远远大于充电过程，并且工况序列具有一定的周期性。由于新能源电池的特殊性，数据采集周期和成本都非常高，几乎不可能用人工采集的方式覆盖所有的场景。没有数据扩增技术的支持，很难确保新能源电池测试数据集的多样性。

现如今，数据扩增技术的研究主要集中在图像识别领域，基于时间序列的数据扩增方法并不完善。本文利用传统算子和深度学习方法提出了一套面向新能源电池的数据扩增技术，分别从数据空间和特征空间实现新能源电池的数据扩增。首次提出基于迭代预测的智能扩增方法，将编解码模型运用到新能源电池数据扩增领域。同时设计工况提取算子，利用权重控制扩增样本中充放电样本比例。实验结果表明，Teacher Forcing 机制可以将迭代预测的累积误差控制在 1% 以内，确保了扩增样本序列的可靠性。同时，工况序列提取算子中的权重参数很好地控制了扩增样本中充放电样本的比例。

关键词：电池管理系统，荷电状态估计，深度学习，时间序列数据扩增

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Data Augmentation Technology for BMS State-of-Charge Estimation of New Energy Vehicles

SPECIALIZATION: Software Engineering

POSTGRADUATE: Ye Ni

MENTOR: Professor **Zhenyu Chen**

Abstract

Global warming and energy issues are two significant crises that trouble society. Energy-saving and emission reduction is the most fundamental solution. Nowadays, many countries have begun to vigorously develop new energy vehicles (NEVs) to reduce their dependence on fossil fuels and decrease the emission of greenhouse gas. NEVs use the battery array controlled and monitored by a battery management system (BMS) whose core function is the State-of-charge (SOC) estimation as the power source. According to the SOC estimation and equilibrium strategies, BMS can extend the battery life and prevent the safety accidents such as overcharge and over-discharge. However, the method of SOC estimation is still immature, and there are many challenges to estimate SOC accurately. Fortunately, benefiting from developments of deep learning technology and improvements in in-vehicle computing power, more and more manufacturers have introduced deep learning approaches to estimate SOC value and achieved good results.

The model quality assurance work still uses black-box testing methods, resulting in the diversity of the test data directly determining the accuracy and generalization ability of the model. This phenomenon is further aggravated by the unexplainable of the model; therefore, a diverse testing dataset is critical to the model quality. It is clear the test data collected from NEVs is imbalanced as the samples in the discharge process are much larger than those in the charging process. In addition, the analysis found that the sampled sequences have periodically working conditions. It is barely possible to cover all scenarios with manual collection because of the high cost and cycle of data collection caused by the property of new energy batteries. Therefore, it is hard to

ensure the diversity of new energy test dataset without the help of data augmentation technology.

At present, the research of data augmentation is mainly focused on the image classification field, and data augmentation methods based on time series are not perfect. This thesis proposes data augmentation methods that introduce traditional operators and deep learning methods to achieve data augmentation through data space and feature space, respectively. We first proposed the model augmentation method based on iterative prediction and applied the deep learning technology to new energy battery data augmentation. We also design an algorithm to extract the working condition and control the proportion of charging and discharging samples by weight parameters. The experiment results show that the Teacher Forcing technology successfully restricts the cumulative deviation of the augmentation model method based on iterative prediction within 1%, ensuring the reliability of the amplified sample sequence. Additionally, the weight parameter in the working condition extraction operator can control the ratio of the charging and discharging samples in the augmentation dataset.

Keywords: BMS, SOC estimation, Deep learning, Time series data augmentation

目录

表 目 录

图 目 录

第一章　绪论

1.1　研究背景与意义

随着时代发展，全球变暖和能源危机已经成为全球公共问题。节能减排无论是在政府政策还是在应用技术上都占据了重要的位置，是解决当前困境的关键 [1, 2]。新能源汽车的诞生，符合人们对节能减排的迫切需要，是能源变革的一个伟大尝试，具有广阔的前景。根据工信部公开的信息[1]，2021 年，新能源汽车产销分别完成 354.5 万辆和 352.1 万辆，同比均增长 1.6 倍，市场占有率达到 13.4%。从中可以看出，每卖出六辆汽车中就有一辆是新能源汽车。

电池管理系统（Battery Manage System，BMS）是新能源汽车的核心组成部分，负责智能化监测、管理、以及维护新能源电池状态等工作。BMS 在防止电池过充/过放、延长电池使用寿命等方面发挥至关重要的作用，而荷电状态（state-of-charging，SOC）估计是 BMS 实现这些功能的基础。SOC 表征当前电池阵列的剩余能量，一般以百分数表示，数值从 0% 到 100%。一辆新能源汽车，其动力源由几百块甚至几千块锂电池组成的阵列构成，低压（4V）单体锂电池经过串并联的方式构成高压向外供电 [3]。每一块锂电池都有独立的物理特性，随着使用时间的增长，单体锂电池会呈现不同程度的老化 [4]，导致电池阵列整体一致性下降，而电池非一致性增加又加速了电池老化的进程。BMS 基于 SOC 估计采用不同的均衡策略，可以维护电池阵列的一致性从而延长电池使用寿命 [5, 6]。同时，准确的估计 SOC 状态可以帮助 BMS 有效防止过充、过放等安全事故的发生 [7, 8]，保障人们的生命财产安全。

现阶段进行 SOC 估计的方法有很多，包括：安时积分法 [9, 10]、开路电压法 [11–13]、滤波法 [14–16]、以及深度学习法 [17–21]。尽管现在所有的 BMS 都带 SOC 估计功能，但 SOC 估计技术远远没有成熟。由于数目众多的新能源电池单元、特征采集引入的随机噪声、等效建模误差等原因，想要精确估计 SOC 非常困难。不仅如此，受限于车载算力和对系统实时性的要求，BMS 无法部署高复杂度的 SOC 估计算子。随着 AI 加速芯片的出现和 5G 通信的发展，使得车载部署基于深度学习模型的 SOC 估计方法成为可能。深度学习法具有非常多的优点，其不需要对新能源电池内部复杂化学特性 [21] 进行建模，并且可以直接利用海量的实车数据。因此，越来越多的新能源厂商选择深度学习解决方案。

[1] https://www.miit.gov.cn/gzcy/zbft/art/2022/art_c1be7ff778bd42c28ffde666d93328ad.html

数据驱动的深度学习 SOC 估计方法，由于模型内部参数共享和复杂的特征组合，很难解释模型到底学习到了什么，但在新能源汽车领域，出于安全性考虑以及伦理和法律的需要，算法的可解释性又是十分必要的，模型的不可解释性给质量保障工作带来了很大挑战。现阶段主要通过黑盒测试的方式来确保深度学习模型在新能源汽车 BMS 应用的可靠性和安全性。然而，当前新能源汽车主要用于上下班通勤，造成实车采集的数据具有一定的周期性。并且，充电阶段的数据记录远远少于放电阶段，导致实车采集的数据虽然样本量很大，但是场景多样性方面并不完善，造成很大的安全隐患。由于新能源电池的特殊性，有目的构造场景进行数据搜集往往周期很长并且代价很高，比如老化数据收集，一次完整的数据采就需要报废一块新能源电池。因此，为了增强新能源电池测试数据集的多样性，需要数据扩增技术的支持。目前数据扩增技术的研究主要集中在计算机视觉和自然语言处理等领域，面向时间序列数据仍缺乏有效的数据扩增方法 [22]。针对新能源汽车电池数据扩增方法的研究还处于空白阶段，且现存的时间序列扩增技术大多并不适合新能源电池领域，新能源汽车 BMS 急需一套工况可控的数据扩增方法，来确保 SOC 估计模型的可靠性。

为此，本项目提出了一套面向新能源电池 SOC 估计的扩增方法，该方法结合传统序列扩增算子和深度学习技术，分别从数据空间和特征空间提升了原始数据集的覆盖情况。传统序列扩增方法围绕新能源电池的使用场景设计扩增算子，从扰动、数据丢失、以及噪声抑制三方面进行数据扩增。而深度学习扩增方法利用时间序列编解码模型，在给定目标工况条件下进行迭代预测生成扩增样本，从特征空间扩展原始数据集。为了进一步平衡深度学习扩增方法中充电过程和放电过程的样本数目，本项目设计了工况提取算子，该算子基于能量变化进行工况提取，利用充放电权重参数控制工况序列中充放电样本的比例。本项目提出的新能源电池扩增方法，不仅可以从数据空间和特征空间提升原始样本的覆盖情况，更可以平衡原始样本中充放电的比例，极大提高了原始数据集的多样性，帮助测试团队更好的进行质量保障工作，在缩短项目周期的同时节省了数据采集的成本。

1.2 国内外研究现状

1.2.1 SOC 估计技术研究现状

安时积分法也叫库伦积分法，是估算 SOC 最直接有效的方法之一。使用此方法需要系统给定 SOC 初始状态，然后统计后续时刻充放电电流计算系统能量变化从而更新 SOC 状态。由于错误的初始状态估计会造成序列估计的群聚失误，Saji 等人在 [9] 文章中提出利用模糊算子来缩小安时积分法初始状态设定的误差，从而提升了整体估计的精度。在 [10] 中，为了解决噪声累积造成的精度损

失问题，He 等人提出了改进型安时积分法 ICC。该方法首先判断电池是否处于稳定状态，如果不是则利用安时积分法估测 SOC 值，否则利用开路电压法估测 SOC，并过滤安时积分法中累积的噪声误差。

开路电压法估计 SOC，利用开路电压 OCV 和 SOC 的映射关系来估计当前电池的 SOC 状态。然而测量开路电压的条件过于苛刻，几乎无法在真实世界进行。研究者一般对目标电池建立等效电路，通过端电压来估算开路电压值，最后查表估算当前 SOC 状态。在 [11] 文章中，作者结合开路电压法和扩展卡尔曼滤波算子，让所有 SOC 估算结果均满足 ±5% 的规范要求。Susanna 等人在 [12] 文章中利用戴维南电池模型和简单电池模型估计 OCV 值，并利用递归最小二乘 RLS 进行模型参数调整，实验结果表明两种模型估计 OCV 的误差可控制在 1% 以内。同样结合 KF 算子和开路电压法估计 SOC，在 [13] 文章中，作者提出利用 LS-SVM 分类器从不同条件下的 OCV-SOC 映射曲线中找出最准确的一条，从而提高了 OCV-SOC 映射关系的准确度。

滤波法使用卡尔曼滤波算子估算 SOC，该方法同样需要对电池建立等效模型，与开路电压法不同的是模型的参数是时变的。在 [14] 文章中，作者建立了一个二阶 RC 等效电路模型，通过带遗忘因子的递归最小二乘法确定电池模型的参数，再利用扩展卡尔曼滤波算子 UKF 进行 SOC 估算。传统的 UKF 需要开发人员先验确定系统噪声特性，而 Ma[15] 等人将扩展卡尔曼滤波算法 UKF 和模糊算子相结合，利用模糊算子自适应的调整系统中的观测噪声特性，从而提升了算法的准确性和鲁棒性。在 [16] 文章中，作者利用 EKF 结合开路电压法和安时积分法，基于硬件在环 HIL 的方式对 SOC 估计进行了仿真实验，进一步提升了 SOC 估计的精度。

Chemali 等人在 [17] 中提出使用 LSTM-RNN 网络在不使用任何电池模型、滤波算子、以及推理系统的情况下进行 SOC 估计。Yang 等人在 [18] 中使用堆叠的 LSTM 网络进行 SOC 估计，发现 LSTM 模型方法的非线性性能要优于 UKF 滤波算子。在 [19] 文章中，作者将 LSTM 模型和向量自回归滑动平均算子（VARMA）结合，利用 VARMA 算子估算新能源电池的线性成分，用 LSTM 模型预测其中的非线性成分，即 LSTM 模型的学习标签是真实电流序列和 VARMA 估计电流序列之间的残差，实验结果表明该方法比单独使用 LSTM 模型或者 VARMA 算子的效果要好。在 [20] 文章中，Tian 等人首次提出 LSTM-ACKF 方法，利用安时积分法和 LSTM-base 模型分别计算预测 SOC 以及观测 SOC，然后利用 ACKF 算子计算出最终 SOC 估计值，实验结果显示在不同的温度下 LSTM-ACKF 方法都要优于单独使用 LSTM 以及 LSTM-CKF 的方法。Kim[21] 等人利用 LSTM 构建了一个三层网络，并基于美国公路燃油经济性试验 HWFET 标准实时估计 SOC

值，实验数据表明同时使用最近 5 个时间步的电压、电流数据可以将估计误差控制在 1% 以内。

从上面可以看出，传统方法估算 SOC 值都具有一定局限性。安时积分法要准确跟踪能量变化，不但要求 BMS 保持较高的采样率，还要能够解决误差随着时间不断累积的问题。开路电压法中，由于 SOC 对 OCV 的变化非常敏感，细微的 OCV 误差也会让 SOC 估计产生较大的偏差。虽然实验条件下有较好表现，但是实车运行的环境要比实验室设置的情况复杂的多，导致该方法实用性不高。滤波法的算法复杂度较高并且依赖很多假设，这些假设在实际环境中成立的条件过于苛刻。比如，对于高度非线性系统，EKF 估计会发散从而变得不可靠；而无迹卡尔曼滤波虽然可以作用于非线性场景但其稳定性方面需要加强 [23]。由于深度学习法不需要针对新能源电池进行建模并且可以直接使用实车采集的数据等优点，越来越多的厂商选择其进行 SOC 估计，也让 SOC 估计的质量保障工作变得更加重要。

1.2.2 时序扩增算法研究现状

时间序列数据扩增主要从数据空间和特征空间进行数据生成。

数据空间扩增主要利用数据变换的形式实现。liu 等人在 [22] 文中，使用噪声添加（AddNoise），排列（Permutation），翘曲（Warping）以及缩放（Scaling）四种方法对数据进行时序扩增，以此来增强 FCN 和 ResNet 模型的训练效果。在 [24] 文章中，Fields 等人在数据集中添加噪声以此提高模型在数据漂移情况下的预测精度，利用 Random Walk 算子在特征参数中添加随机偏移，实验表明 RNN、LSTM 等模型对偏移的鲁棒性比较好，而 CNN 受特征偏移的影响较大。同样是引入噪声的扩增方式，在 [25] 中 Goubeaud 等人提出噪声窗口（WNW）的概念，随机将时间序列中的一段连续的片段替换成高斯白噪声来生成新的训练数据，并在 KNN 模型上取得了不错的效果。在 [26] 中，Gao 等人提出一种面向异常数据检测的数据扩增技术，利用傅里叶变换（FFT）将时间序列映射到频域，在频域对序列的幅值和相位特征添加可控正太分布的噪声进行扰动变换，实验结果表明，采用了此方法训练的模型具有最好的检测精度。

特征空间扩增主要利用已知数据的分布规律产生新的时间序列，并基于已有数据分布对扩增数据进行标签。[27] 文中，作者等人提出了一种基于 DTW 的权重平均方法 DBA 来进行数据扩增，通过权重调整进行数据合成，实验结果显示该扩增方法在实验数据较少或者使用完整数据集的情况下都可提升 NN-base 分类模型的准确性。Yang 等人在 [28] 文章中利用 DTW 技术替代欧氏距离将传统合成少数过采样技术（SMOTE）运用到了时间序列应用中，提出了一种面向

ResNet 和 LSTM 模型的数据扩增方法，通过在特征空间内随机生成特征序列，根据已知 KNN 的中心点对生成数据进行分类标签。基于深度学习编解码模型，Chowdhury 等人在 [29] 文中进行了单一特征维度的时序扩增，实验结果表明使用编解码模型可以比 TimeGAN、ADASYN、和 SMOTE 方法生成与原始时间序列更加相似的扩增数据，同时用此扩增数据训练和测试 AI 模型的效果也更好。

从上可以发现，无论是分类应用还是预测应用，时间序列扩增技术都可以很好的提升模型的准确性。由于新能源电池内部机理的特殊性，针对单一维度进行序列扩增的方法可用性很低。面向新能源电池领域数据扩增方法的研究尚处于起步阶段。

1.3 本文主要工作

本文基于传统算法和深度学习模型，对新能源电池数据进行扩增。在现有数据集的基础上，生成新的数据用于新能源领域模型的质量保障工作。为了确保扩增模型在不同工况条件下生成数据的准确性，设计了模型验证方法并给出了验证指标。并且设计工况提取标准算子，通过充放电权重调整工况序列中充放电样本的比例。主要工作分为以下几个部分：

（1）基于特征工程和关联分析法对实车采样的特征数据进行特征筛选，然后利用特征融合压缩特征维度。由于 BMS 系统采集的数据包含很多空值和异常值，本文利用箱形图算法确定特征序列的双门限范围，然后过滤异常值和空值记录样本，最终获得可用于数据扩增和模型训练的预处理数据集合。

（2）利用传统算法针对扰动场景、数据丢失场景以及噪声抑制场景设计不同扩增算子，在不改变原始数据标签的情况下生成数据样本，拓宽了原始数据的表达能力。

（3）设计并训练扩增模型、确定模型验证指标。实现了在给定工况序列条件下生成响应电压值。然后根据生成样本特征的特性，基于传统差分、帧内保持等策略生成对应特征序列，并对生成的样本进行数据标注。由于生成的新能源电池数据无法由人直接判断正确与否，本项目设计了模型验证过程，利用均方误差和多帧最大均方误差作为验证指标。

（4）设计工况序列提取算子，提出以帧为单位，帧内新能源电池电荷变化的强弱为指标进行工况序列提取。为了解决 BMS 采样周期较长，特征序列细节丢失导致无法使用安时积分法对生成数据进行标注的问题。本文提出在工况提取阶段，利用差分算子提取每个时间步的电荷变化情况，以此为依据计算生成样本的 SOC 序列特征值。

（5）设计并实现了面向新能源汽车 BMS 荷电状态估计的扩增系统，包括数据仓库管理、传统算法扩增、智能扩增、以及模型验证等功能。

1.4 本文主要结构

本文总共分为八个章节，各个章节的主要内容如下：

第一章是引言部分，描述了新能源汽车荷电状态估计的背景和意义，当前SOC 估计方法的国内外研究现状及缺陷。接着介绍了深度学习技术进行 SOC 估计的优点和质量保证问题。最后描述了当前序列扩增算法的研究现状，以及现存时序扩增方法在新能源电池领域的可用性情况。

第二章是相关概念介绍与技术综述部分，描述了本项目需要使用的一些技术内容。首先介绍了主流 BMS 进行荷电状态 SOC 估计的方法，并给出了相关公式。然后描述了几种关联分析法以及数据清洗的方法，用于对采集的数据进行数据挖掘并压缩数据维度，以此来提高数据质量。其次描述了如何利用傅里叶变换算子从频域的角度分析数据。接着详细介绍了几种传统时序扩增方法及其优点和适用的场景。最后对构建扩增模型需要用的技术进行了阐述，包括模型网络，损失函数，优化算子以及防止过拟合的方法等内容。

第三章是数据预处理部分，主要描述如何对采集的原始数据进行特征分析和矩阵运算，挑选与荷电状态估计强相关的特征集合。首先分析数据特征的内容，利用特征工程和关联分析法挑选出与目标荷电状态估计相关的特征。然后对挑选出的特征进行特征融合，进一步压缩输入特征的维度。接着描述了如何利用箱形图算子计算每个特征的双门限范围，过滤其中的异常序列。最后对序列样本中的空值进行处理，得到模型训练和扩增的基础数据集。

第四章是传统扩增算法设计部分，从数据变换的角度描述如何针对新能源电池场景设计扩增算子，场景包括：扰动场景、数据丢失场景、以及噪声抑制场景。其中，扰动场景分别从时域和频域角度引入噪声；数据丢失场景设计了对称丢失和非对称丢失两种策略；噪声抑制场景基于窗口平均和指数平均两种方式生成扩增样本。

第五章是基于智能模型扩增部分，描述了如何使用深度学习技术训练扩增模型以及使用模型生成扩增数据集。首先描述扩增模型的结构并给出关键构建代码。然后描述了如何对数据进行归一化处理，以及将数据拆分成可直接用于模型训练的特征序列和对应的标签序列。接着给出模型训练的一些相关参数，包括损失函数、评估指标等。由于模型在部署阶段和训练阶段采用了不同的网络结构，因此在本章中又介绍了模型验证的方法以及验证指标内容。模型训练和验证结束后，紧接着描述了工况提取算子以及相关参数配置。最后根据生成数据样本的特征类型，详细阐述了如何使用扩增模型和提取的工况生成扩增样本。

第六章是系统需求分析以及详细设计部分，主要描述了系统的需求和结构

设计。本章首先描述了如何基于需求工程获取客户的功能性需求以及非功能性需求。然后，编写用例描述并给出 4+1 视图，分别从用例视图、逻辑视图、开发视图、进程视图、以及物理视图对系统进行了描述。接着，基于 ER 图从实体关系角度描述了数据库表的属性，并详细介绍了各属性的实际含义。最后在详细设计章节给出了各功能模块的执行流程图和核心代码，并展示了系统的界面。

第七章是系统测试和实验分析部分，首先介绍了系统功能性测试的方法和结果。然后分析了各扩增方法的结果，描述了基于传统时间序列扩增算子在数据空间进行样本生成的特性，以及扩增模型在部署阶段进行迭代预测的准确性情况。

第八章是总结与展望部分，首先总结了项目的背景以及意义，介绍了本课题解决的核心问题。然后描述了本文的主要工作以及核心创新点。在展望部分对系统的不足进行了刻画并给出提升建议以及方向。

第二章　相关技术概述

2.1　新能源电池剩余电荷 SOC

新能源电池荷电状态（SOC）表示电池当前可用容量，对其状态估计是新能源汽车电池管理系统的核心功能之一。目前进行荷电状态估计的主流方法有：(1) 开路电压法；(2) 安培积分法；(3) 卡尔曼滤波法；以及 (4) 深度学习模型法 [30–32]。

开路电压法基于 RC 等效电路利用开路电压和 SOC 之间的映射关系进行荷电状态的估计。此方法利用传感器采集的端电压信息，通过构建的 RC 等效电路计算当前时刻的开路电压值，最后通过查表的方式确定当前时刻荷电状态。开路电压法估计荷电状态如图2.1所示。

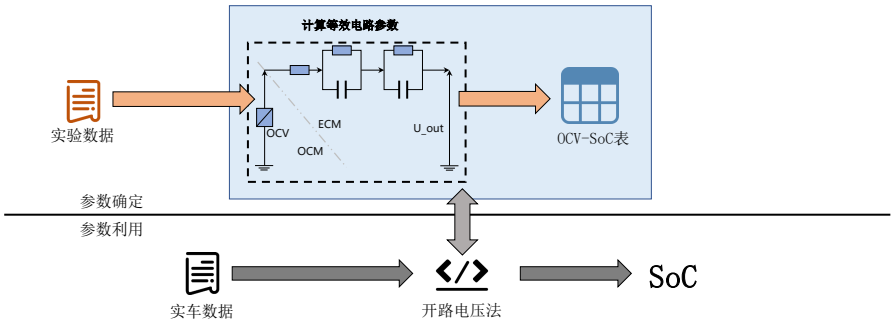

图 2.1: 开路电压法

安培积分法则是从电流的角度通过对工作过程中进入/放出电流求积分实现荷电状态的估计。其公式如下所示：

$$SOC = \overline{SOC} + \frac{\int idt}{C} \tag{2.1}$$

其中，$\overline{SOC}$ 表示开始时刻荷电状态，C 表示当前电池的最大容量。

卡尔曼滤波法将新能源电池看成一个通用的动力系统，该动力系统的输入用向量 **x** 表示，包括电流、温度、剩余容量等变量；系统输出用向量 **y** 表示，一般为可观测的变量，比如电池的工作端电压；用向量 **s** 表示其内部状态。公式分

为状态方程和观测方程两部分，具体如下：

$$\begin{aligned} \mathbf{s_{t+1}} &= A_t s_t + B_t x_t + w_t, \qquad \text{状态方程} \\ \mathbf{y_{t+1}} &= C_t s_t + v_t, \qquad \text{观测方程} \end{aligned} \tag{2.2}$$

其中 A，B，C 为卡尔曼滤波参数，在实际执行过程中自适应调整；w 和 v 为噪声统计特征，一般由开发人员先验给定。

深度学习模型法利用误差反向传播 BP 算子计算预测过程中的误差，模型一般基于 LSTM 网络，采用均方误差（MSE）函数作为损失函数，建立电池特征到 SOC 的映射。

2.2 常见序列分析和数据清洗算法

2.2.1 关联分析法

当前用于关联分析的主要方法有：

（1）灰色关联分析法，其比较适合样本少而属性较多的情况，并且需要事先定义最优或者最坏的参照样本，其公式如下：

$$\zeta_i(k) = \frac{\min\limits_i \min\limits_k |x_0(k) - x_i(k)| + \rho \cdot \max\limits_i \max\limits_k |x_0(k) - x_i(k)|}{|x_0(k) - x_i(k)| + \rho \cdot \max\limits_i \max\limits_k |x_0(k) - x_i(k)|} \tag{2.3}$$

（2）皮尔逊关联系数法，适用目标特征字段与被测特征字段之间是线性相关的情况，无法描述非线性相关的情况并且受到离群点的影响比较重，其公式如下：

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}} \tag{2.4}$$

（3）肯德尔相关系数法，又称秩相关系数，此类方法一般用于特征字段值对象是有序的分类变量情况下，使用“成对”来描述相关性的强弱关系。其有三种表现形式，对于具有重复特征值的公式如下，其中 c 和 d 表示分歧对数和一致性对数：

$$Tau - b = \frac{c - d}{\sqrt{(c + d + t_x)(c + d + ty)}} \tag{2.5}$$

（4）斯皮尔曼相关系数法，此方法受到的约束比较少并且受到离群点的影响也较低，因此适用范围较广，其相关公式如下：

$$\rho = 1 - \frac{6 \times \sum_{r=1}^{n} d_i^2}{n(n^2 - 1)} \tag{2.6}$$

式中 n 为样本的个数；d_i 为样本中的特征的秩次，即目标特征字段和对比特征字段的值经过排序后序列值的差。

2.2.2 数据清洗

数据清洗过程主要围绕空值和异常值进行，按照特征数据类型的不同，对数据集空值的处理主要有以下几种方式：（1）对于离散型数据特征，采用阶梯插值或最近邻插值算法；（2）对于连续型数据特征，采用前后均值或高阶曲线平滑插值算法；（3）直接舍去；异常值的处理首先需要确定异常值的门限，对于序列特征一般采用箱形图算子计算合理的双门限范围，其原理如图2.2所示。箱形图基于高斯分布，通过已知采样数据序列计算其有效分布范围：以均值为基准，上下偏移一个四分位距离作为边界，划定合理的范围，此范围可以覆盖 99.2% 的情况。

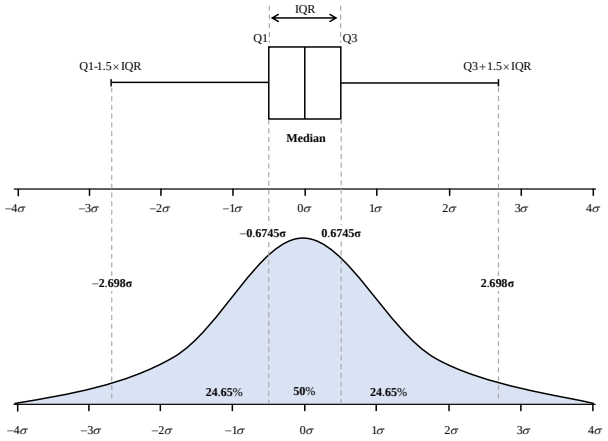

图 2.2: 箱形图基本原理

箱形图公式如下所示：

$$
\begin{aligned}
Median &= \begin{cases} \mathbf{D}\left[\dfrac{N+1}{2}\right], & \dfrac{N}{2} \neq 0. \\ \mathbf{D}\left[\dfrac{N}{2}\right], & \dfrac{N}{2} = 0. \end{cases} \\
Q_1 &= \mathbf{D}\left[\frac{N+1}{4}\right], \\
Q_3 &= \mathbf{D}\left[\frac{3*(N+1)}{4}\right], \\
IQR &= Q_3 - Q_1, \\
Low &= Q_1 - 1.5 * IRQ, \\
Hight &= Q_3 + 1.5 * IQR,
\end{aligned}
\tag{2.7}
$$

上式中 **D** 表示目标特征字段样本序列集合且已经排序（从小到大）；N 表示特征数据值的样本个数。

2.3 频域分析傅里叶变换算法

一般通过傅里叶变换算子将时间序列特征从时域映射到频域 [33]。傅里叶变换根据时域信号的连续与离散可以分为连续傅里叶变换（CFT）和离散时间傅里叶变换（DTFT）。连续傅里叶变换将时域连续的信号看成周期无限长的周期信号，解决了傅里叶级数无法处理非周期信号的问题。离散时间傅里叶变换在连续傅里叶变换的基础上，在时域对连续信号进行采样将连续信号变成离散信号，突破了傅里叶级数只能作用于连续信号的限制。为了降低算法的复杂度，1965年由 J.W. 库利和 T.W. 图基提出快速傅里叶变换（FFT），本质是利用离散傅里叶变换的对称性将计算复杂度大大降低。不同版本傅里叶变换的特性如表2.1所示。

表 2.1: 五种傅里叶变换的特性

变换	时域特性	频域特性
傅里叶变换 FT	连续、非周期	非周期、连续
傅里叶级数 FS	连续、周期	非周期、连续
离散时间傅里叶变换 DTFT	离散、非周期	周期、连续
离散傅里叶级数 DFS	离散、周期	周期、离散
离散傅里叶变换 DFT	离散、非周期	非周期、离散

快速傅里叶变换（FFT）算子仅仅是离散傅里叶变换（DFT）算子的快速计算版。对于一个长度为 N 的离散信号 $x[n],\quad n=0,K,N-1$，其离散傅里叶变换为：

$$X[k]=\sum_{n=0}^{N-1}x[n]W_N^{nk},\quad k=0,K,N-1 \tag{2.8}$$

$$x[n]=\frac{1}{N}\sum_{k=0}^{N-1}X[k]W_N^{-nk},\quad n=0,K,N-1 \tag{2.9}$$

其中：$W_N=e^{-j\frac{2\pi}{N}}$；$X[k]$ 为离散傅里叶变换 DFT，$x[n]$ 为离散傅里叶变换的反变换 IDFT。向量 $w^{(k)}=\begin{bmatrix}1 & W_N^{-k} & W_N^{-2k} & K & W_N^{-(N-1)k}\end{bmatrix}$，是构成 N 维复数空间 C^N 中的一组正交基，即 DFT 的基函数。

2.4 传统时序扩增算法

传统时序扩增算子主要包括：（1）翻转算子（flipping），该算子基于目标对象时序过程的对称性进行数据扩征，比如一个序列 $\mathbf{x}=(x_1,x_2,\ldots,x_n)$，其翻

转扩增对象为 $\hat{\mathbf{x}} = -\mathbf{x} = (x_n, x_{n-1}, \ldots, x_1)$，故翻转算子不适合无法逆向执行的领域；（2）裁剪算子（crop），将数据和标签同时裁剪成多个更短的序列构成扩增数据集，此类方法比较适合分类问题，比如心电图时序数据，将一个异常心电图序列随机裁剪成多段包含异常心率的时间序列样本，从而生成多个异常心率测试数据集；（3）排列算子（permutation），将一个完整的时序数据序列切片成 N 个离散的片段，然后对这 N 个片段在时间轴上进行随机组合，生成新的时序数据。排列扩增算子和裁剪扩增算子类似，主要用于基于时间序列的异常检测场景；（4）缩放算子（scaling），该算子首先对序列进行分帧，再通过缩放算子对帧内数据的幅值进行缩放；（5）扰动算子（jittering），扰动算子通过在传感器端添加额外的随机噪声来提高训练数据对不同类型传感器噪声的鲁棒性，比如加性噪声和乘性噪声；（6）窗口翘曲算子（warping），对数据采样的时间间隔进行操作来扰乱序列产生的时间位置，采样间隔调整方法包括下采样和上采样两种方式；（7）窗口平滑算子（window smoothing），该算子通过滑动平均技术过滤传感器采集过程中引入的随机噪声，从而消除时间序列中的趋势项生成扩增样本。

2.5 深度学习

2.5.1 LSTM 模型

LSTM[34] 是一个深度学习网络，也叫 LSTM 层。该层利用一个记忆体记录时间序列中的历史信息，因此 LSTM 层常常用于与时间序列相关的任务中 [35–37], 并且其表现要优于传统的算子, 比如 ARIMA[38]。LSTM 的层结构如图2.3所示，一个 LSTM 层由三部分组成，分别是：遗忘门，输入门，以及输出门。当时间序列数据不断输入时，LSTM 层会迭代更新记忆体状态，该记忆体负责提取和保存 LSTM 层从输入的时间序列中获得的信息，如图2.3中的 $\mathbf{C_t}$ 所示。LSTM 层输入门的输入包括两个部分，一个是当前时刻的序列特征，用 $\mathbf{X_t}$ 来表示，另一个是上一个时间步 LSTM 层的输出结果，用 $\mathbf{H_{t-1}}$ 来表示。LSTM 层将这两部分输入连接起来，构成了一个新的输入张量作为当前时刻的最终输入，用 $[\mathbf{H_{t-1}}, \mathbf{X_t}]$ 来表示。最后，将连接的最终输入送入上述的三个门，即遗忘门、输入门、以及输出门，计算最终的输出 $\mathbf{H_t}$ 并更新 LSTM 层记忆体的状态 $\mathbf{C_t}$。

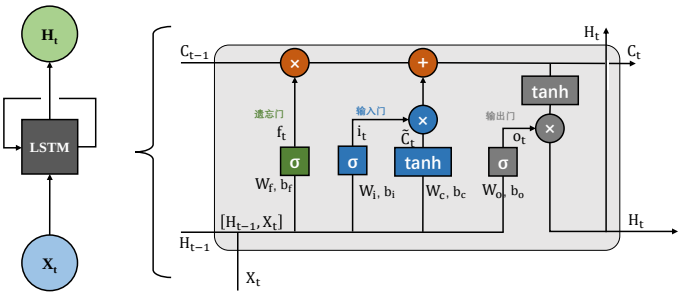

图 2.3: LSTM 模型层结构

LSTM 层的具体公式如下：

$$\mathbf{f_t} = \sigma\left(\mathbf{W_f} \cdot [\mathbf{H_{t-1}}, \mathbf{X_t}] + \mathbf{b_f}\right) \tag{2.10}$$

$$\mathbf{i_t} = \sigma\left(\mathbf{W_i} \cdot [\mathbf{H_{t-1}}, \mathbf{X_t}] + \mathbf{b_i}\right) \tag{2.11}$$

$$\widetilde{\mathbf{C}}_t = \tanh\left(\mathbf{W_c} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_c}\right) \tag{2.12}$$

$$\mathbf{C_t} = \mathbf{f_t} * \mathbf{C_{t-1}} + \mathbf{i_t} * \widetilde{\mathbf{C}}_\mathbf{t} \tag{2.13}$$

$$\mathbf{o_t} = \sigma\left(\mathbf{W_o} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_o}\right) \tag{2.14}$$

$$\mathbf{h_t} = \mathbf{o_t} * \tanh\left(\mathbf{C_t}\right) \tag{2.15}$$

$$\sigma\left(x\right) = \frac{1}{1 + e^{-x}} \tag{2.16}$$

$$\tanh\left(x\right) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{2.17}$$

上式中，$\mathbf{W_f}, \mathbf{W_i}, \mathbf{W_c}, \mathbf{W_o}$ 和 $\mathbf{b_f}, \mathbf{b_i}, \mathbf{b_c}, \mathbf{b_o}$ 是 LSTM 层的可训练参数；$\mathbf{f_t}$ 表示遗忘门，决定有多少记忆体中的信息量需要在本次推理过程中被遗忘；$\mathbf{i_t}$ 表示输入门，主要由两部分组成：第一部分决定哪些信息需要被更新，第二部分通过一个 tanh 函数获取候选记忆体 $\widetilde{\mathbf{C}}_\mathbf{t}$ 的状态；输出门 $\mathbf{o_t}$ 结合更新后的记忆体状态和输入信息来计算输出值。

2.5.2 归一化

归一化主要负责去除训练数据集中的量纲，直接将带有量纲的数据送入模型会降低模型的收敛速度，这主要由以下两个原因造成：（1）不同特征字段之间的量纲差距可能很大；（2）特征字段的数据类型不同，导致其表达能力也不

相同。使用归一化方法在提高模型训练收敛速度的同时也可以在一定程度上提高模型的精度。常用的归一化方法有：（1）最大最小标准化；（2）z-score 标准化；（3）log 对数归一化。

最大最小标准化公式如下：

$$\hat{x} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{2.18}$$

该归一化方法将特征值对称的映射到 $[0, 1]$ 之间，适用于特征值相对集中的场景。缺点是在数据量较少情况下，统计出来的最大最小值并不稳定，导致在实际使用中出现特征值超过 1 的情况发生。

z-score 标准化公式如下：

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{2.19}$$

上式中，μ，σ 分别表示特征值的均值和方差。该方法可以将特征值映射成均值为 0，方差为 1 的标准正态分布，对于原始特征不是正态分布的特征，其归一化效果往往不会太好。

对数归一化公式如下：

$$\hat{x} = \frac{\log(x)}{\log(max)} \tag{2.20}$$

该方法主要适用于特征值分布比较离散的场景，利用对数函数可以将特征值中较大的部分变小，较小的部分提升，让特征数据向中间靠拢。

2.5.3 训练参数

选择损失函数主要依赖深度学习解决的问题类型。对于回归问题一般以均方误差（MSE）损失函数为主，即求模型预测的结果和标签结果误差的平方和，以此作为依据判断模型的预测精度；对于分类问题主要采用信息熵或者交叉熵作为判断模型精度的依据，熵即预测概率分布和标签概率分布之间的距离。

优化算子大多是基于随机梯度下降方法的各种改进型或者衍生算子。传统的随机梯度下降算子 SGD 由于其在训练过程中采用固定学习率，效果往往没有其他采用非固定学习率的优化算子好。Adam[39] 算子是现在使用最为广泛的优化算子，其结合了 AdaGrad 和 RMSProp 算子的优势，能够解决稀疏梯度和噪声问题。

确定评估函数同样依赖深度学习模型所解决的问题域类型，常用的误差计

算函数有：MAE、MSE 以及 R2_score，其公式如下：

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2.21}$$

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.22}$$

$$\mathrm{R2} = 1 - \frac{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{2.23}$$

上述公式中，y_i 表示标签值，$\hat{y}_i$ 表示预测值，$\bar{y}$ 表示标签 y_i 的平均值。R2_score 以一个量化的数值来表示模型回归预测的准确率，一般来说这个标量值介于 0 和 1 之间，越接近 1 表示预测的结果越精确，越接近 0 表示预测结果越接近均值。

2.5.4 正则化

正则化主要解决模型的过拟合问题，模型过拟合主要表现在模型的预测结果在训练集上表现良好，但是在测试集上表现较差，即训练的模型缺少泛化能力。目前主要通过：（1）减少模型内部参数量；（2）对模型参数添加惩罚项；（3）使用 dropout 机制；（4）获取更多的训练数据集；来减少模型的过拟合问题。

模型惩罚项也叫做参数正则化（Regularization）[40, 41]，主要有 L1 正则化，L2 正则化以及同时使用二者。其公式如下：

$$J = J_0 + \lambda \cdot \|w\|_1 \quad \text{L1 正则化} \tag{2.24}$$

$$J = J_0 + \lambda \cdot \|w\|_2^2 \quad \text{L2 正则化} \tag{2.25}$$

上述公式中，J_0 表示损失函数值，w 代表模型中所有可训练的参数。

降低过拟合提高模型泛化能力还可以通过 Dropout 机制实现，其本质在于每一次训练推理过程中会随机丢弃其中的某些连接路线 [42, 43]。

2.6 软件设计

2.6.1 前端相关技术

Vue 是一套用于构建用户界面的渐进式框架，诞生于 2014 年。由于其易上手、低学习成本的优点迅速占领了市场，被广泛运用于 web 端、移动端以及跨平台应用等方面。Vue 结合了 Angular 和 React 的优势，支持前后端分离设计。

ElementUI 是一套以 Vue 为基础框架实现的组件库，界面设计更加符合国人对桌面端应用的美感。开发者可以直接使用 ElementUI 提供的主题和组件进行

界面开发，是目前基于 Vue 最常用的组件库。

2.6.2 后端相关技术

Django 是 python 众多 web 框架中最具代表性的一个，采用 MVT（Model, View, Template）软件设计模式，MVT 分别表示模型、视图和模板。Python+Django 架构是快速开发、设计、以及部署网站的最佳组合，开发者只需要很少的代码即可完成一个网站的大部分功能。由于 Django 解决了大部分 web 开发的问题，开发者可以将关注点集中在 app 业务中。并且 Django 同样支持前后端分离的开发模式。

TensorFlow 是谷歌提供的一个深度学习软件平台，通过 TensorFlow 开发者可以端到端的学习和训练深度学习模型。利用 TensorFlow 开发者可以在云端、本地、或者浏览器上部署自己训练的模型，支持多种语言。

2.6.3 中间件

Nginx 是俄罗斯人 Igor Sysoev 编写的轻量级 Web 服务器，它不仅是一个高性能的 HTTP 和反向代理服务器，还可以实现负载均衡任务。

Docker 是一个开源的应用容器引擎，基于 Go 语言并遵从 Apache2.0 协议开源，可以让开发者打包他们的应用以及依赖包到一个轻量级、可移植的容器中，然后发布到任何流行的 Linux 机器上。

2.7 本章小结

本章介绍了课题设计过程中需要用到的相关技术。首先描述了当前常用的 SOC 估计方法，阐述了各种方法的基本原理和实现方法。然后列举了序列分析方法和数据清洗方式，对于序列分析方法不仅描述了主要的应用场景还给出了具体实现算子，接着介绍了频域分析方法和传统扩增算子，阐述了算法的实现方式和基本原理，同样给出了适用场景范围。紧接着详细介绍了深度学习相关的知识，包括 LSTM 模型结构，数据归一化方法，模型训练参数等内容，同时介绍了如何防止模型过拟合。最后描述了系统设计过程中所使用的一些软件技术，从前端、后端、和中间件三个层次进行介绍。

第三章　数据预处理

3.1　数据特征

由于客户的数据来自实车采集且数据采集的目的是为了更好地监控实车的车体状态，因此本项目获取的新能源数据不仅仅局限于 BMS 所关注的部分。新能源汽车数据采集的特征字段及其数据类型如下：

表 3.1: 新能源数据集的特征字段

特征字段名	数据类型	特征字段名	数据类型
车辆 Vin	字符串	**直流母线电压**	浮点数
时间	字符串	**驱动电机温度**	整数
是否补发	布尔类型	**电机控制器 IGBT 温度**	整数
GPS 海拔	整数	**当前允许最大回馈功率**	浮点数
GPS 车速	整数	**当前允许最大放电功率**	整数
GPS 里程	整数	**动力电池剩余电量 SOC**	浮点数
GPS 方向	整数	**动力电池内部总电压 V1**	浮点
ON 挡唤醒信号	布尔类型	**动力电池正极绝缘电阻**	整数
快充唤醒信号	布尔类型	**动力电池负极绝缘电阻**	整数
慢充唤醒信号	布尔类型	**单体电芯最低温度**	整数
远程唤醒信号	布尔类型	**单体电芯最高温度**	整数
整车 State 状态	整数	**动力电池可用容量**	浮点数
加速踏板开度	浮点数	**动力电池可用能量**	浮点数
挡位信号	整数	**动力电池充/放电电流**	浮点数
制动信号	整数	**累计充电总安时**	整数
驱动电机状态命令	整数	**EAS 当前状态**	布尔类型
驱动电机目标转矩指令	浮点数	**PTC 当前工作状态**	布尔类型
车速信号	浮点数	**ECC 当前状态**	布尔类型
驱动电机当前转速	整数	**制动系统故障显示**	布尔类型
驱动电机当前转矩	浮点数	**DCDC 当前状态**	布尔类型
驱动电机当前状态	整数	**交流充电剩余充电时间**	整数
远驱动电机当前工作模式	布尔类型	**续驶里程**	浮点数
直流母线电流	浮点数	**整车模式**	整数
驱动电机相电流	浮点数	**高压下电请求（充放电专用）**	整数
车载充电机输出电流指令	整数	**电池均衡激活**	布尔类型
动力电池保温状态	整数	**电池包 1 最高温度**	浮点数
动力电池加热状态	整数	**电池包 1 最低温度**	浮点数
充电状态	整数	**电池包 1 电压**	浮点数
终端唤醒结束标志位	布尔类型	**电池包 1 单体温度检测点个数**	整数

(Continued)

表 3.1. *(Continued)*

特征字段名	数据类型	特征字段名	数据类型
远程充电结束指令	整数	电池包 1 单体电压个数	整数
远程空调结束指令	整数	PCU 次软件版本号	字符串
ODO 总里程	浮点数	PCU 主软件版本号	字符串
小计能耗	浮点数	VBP 次软件版本号	字符串
瞬时能耗 1	浮点数	VBP 主软件版本号	字符串
整车故障处理等级	整数	EAS 次软件版本号	字符串
驱动电机系统故障显示	布尔类型	EAS 主软件版本号	字符串
整车最高报警等级	整数	ICM 次软件版本号	字符串
MCU 次软件版本号	字符串	ICM 主软件版本号	字符串
MCU 主软件版本号	字符串	动力电池包编号 1	整数
碰撞信号状态	整数	动力电池包编号 2	整数
运行模式	整数	动力电池包编号 3	整数
直流充电剩余充电时间	整数	动力电池包编号 4	整数
直流充电端电流指令	整数	动力电池包个数	整数
BMS 次软件版本号	字符串	动力电池包编号长度	整数
电池包总成零部件号	字符串	动力电池子系统序号	字符串
BMS 主软件版本号	浮点数	动力电池单体总数	整数
驱动电机总数	整数	动力电池温度点总数	整数
驱动电机序号	字符串	动力电池故障总数	整数
驱动电机故障总数	整数	VIN 编码 1	整数
驱动电机编号 1	整数	VIN 编码 2	整数
驱动电机编号 2	整数	VIN 编码 3	整数
驱动电机编号 3	整数	动力电池系统故障显示	布尔类型
驱动电机编号 4	整数	电量低提醒	布尔类型
慢充 CC 电压	浮点数	READY 灯	字符串
快充 CC2 电压	浮点数	OBC 次软件版本号	字符串
单体电芯最低电压	浮点数	OBC 主软件版本号	浮点数
单体电芯最高电压	浮点数	电机控制器状态管理输出扭矩	浮点数
车载充电机当前状态	整数	驱动有效	布尔类型
车辆运行工况	整数	电机控制器状态管理状态	整数
DCDC 次软件版本号	整数	1-98 号电池单体电压	浮点数
DCDC 主软件版本号	整数	1-24 号温度检测点温度	浮点数
最高温度编号	整数	最低温度编号	整数
最高温度包号	整数	最低单体电压编号	整数
最低温度包号	整数	最高单体电压包号	整数
最低单体电压包号	整数	最高单体电压编号	整数

根据数据特征的物理含义和数据类型，将表3.1中的特征字段进行分类。从特征物理含义来区分，新能源汽车数据记录围绕几个重要的模块展开，包括：(1) 电池管理系统 BMS 相关的特征字段，比如温度，累积充电量；(2) 新能源汽车

电机相关的特征字段，比如电机电流，电机功率；（3）新能源汽车本身状态等一系列特征，比如当前累积行驶里程数，新能源汽车当前行驶状态，车速信号等；（4）其他与新能源汽车相关的特征字段，比如车辆 ID，数据采集的时间信息等；从数据计算机表征类型来区分，包括：（1）连续型的特征，比如：BMS 相关的电压，电流等与新能源电池直接相关的物理特性；（2）离散型的特征，比如：BMS 的型号，新能源电池是否处于充电状态等；（3）整数类型，比如：状态编码，信号状态等；（4）字符串类型，比如：版本号信息，时间信息等；（5）布尔类型，比如充放电状态等；对采集的新能源数据集进行预处理，首先利用特征工程技术，基于数据集特征字段的物理含义对目标数据特征进行筛选，过滤其中与目标特征 SOC 字段无关的特征。其次基于特征字段的数据类型，对筛选后的特征字段进行数据转化使其成为能够被模型直接利用的数据变量。

3.2 数据筛选

首先基于物理特征含义过滤与目标特征 SOC 无关的特征字段。传统方法对新能源电池系统进行建模，通常使用安时积分法或者开路电压法估算 SOC 特征值。

基于安时积分法的 SOC 估算方法将 SOC 看作是新能源电池的一个内部状态，用于表征新能源电池还能放电多少。BMS 不间断地侦测记录新能源电池的工作电流，计算电流流入和流出的情况实现 SOC 估计。由此可以得出新能源电池 SOC 特征字段与新能源电池的电流、充电过程，以及放电过程相关。基于开路电压法的 SOC 估计法将剩余电量看作是新能源电池内部开路电压的一种映射，当电池的开路电压变高时新能源电池的剩余电量也就变大，而当电池的开路电压降低时新能源电池的剩余电量也就随之变小。由于开路电压并不是可以直接测量的特征变量，实际系统中传感器采集的电压特征值为新能源电池的端电压并非开路电压，需要利用 RC 等效电路将传感器获取的端电压转换为开路电压。而 RC 等效电路的参数估计与新能源电池的健康度、工作温度相关，其中电池健康度可以由一些结果特征字段进行侧面的表达，比如，电池的最大电池容量，以及累积充电量特征字段。由此可以得出新能源电池 SOC 特征字段与新能源电池的端电压，温度，最大电池容量和累积充电量等特征字段相关。基于新能源电池数据特征字段的物理含义与传统新能源电池建模方法可以人工筛选出以下字段，如表3.2所示。

通过融合 “动力电池剩余电量 SOC” 和 “动力电池可用容量” 两个特征字段可以获得当前电池最大容量特征值。对于电压相关特征，由于数据采集的汽车只有一个电池包，故特征字段中 “动力电池内部总电压 V1” 的值与 “电池包 1 电压” 的值相等，本项目只采用前者作为目标特征字段，并且该字段作为” 1-98

表 3.2: 筛选后的新能源数据特征字段

电压	电流	温度	健康度
直流母线电压	直流母线电流	驱动电机温度	**累计充电总安时**
1-98 号电池单体电压	驱动电机相电流	**单体电芯最低温度**	**动力电池可用容量**
电池包 1 电压	**动力电池充/放电电流**	**单体电芯最高温度**	**动力电池剩余电量 SOC**
快充 CC2 电压	**快充唤醒信号**	电池包 1 最高温度	
慢充 CC 电压	**慢充唤醒信号**	电池包 1 最低温度	
动力电池内部总电压 V1	**时间**	1-24 号温度检测点温度	
单体电芯最低电压			
单体电芯最高电压			

号电池单体电压“的串联值，已经足够表征新能源的端电压特性，故舍弃”1-98 号电池单体电压“特征字段。同样的道理，舍弃“直流母线电压”。对于”快充 CC2 电压“和“慢充 CC 电压”由于其在整个数据集中的值恒定不变，故也没有必要加入目标特征字段。对于电流相关特征，舍弃“直流母线电流”和“驱动电机相电流”，因为二者是新能源汽车电机部件的工作电流，与新能源电池的输出电流特征字段“动力电池充/放电电流”高度耦合。对于温度相关特征，“驱动电机温度”作为新能源汽车电机的特征字段，在新能源汽车设计过程中，会将电机和新能源电池的冷却液连接起来，以增强二者之间的热传导，加强整个系统的热稳定性，故新能源电池的温度与电机的温度高度耦合，舍去此特征字段。与电压特征字段类似，由于只有一个电池包，“电池包 1 最高/最低温度”特征字段与“单体电芯最高/最低温度”特征字段为同一个特征，故舍弃前者。温度作为一个变化不是非常敏感的特征量，在已知最高和最低温度区间之后，无需知道每一个单体温度传感器的详细值，故舍去“1-24 号温度检测点温度”。精炼以后的特征如表3.3所示。

表 3.3: 精炼后的新能源数据特征字段

电压	电流	温度	健康度
动力电池内部总电压 V1	动力电池充/放电电流	单体电芯最低温度	累计充电总安时
单体电芯最低电压	快充唤醒信号	单体电芯最高温度	动力电池可用容量
单体电芯最高电压	慢充唤醒信号		动力电池剩余电量 SOC

基于新能源电池数据物理特征含义进行特征筛选和特征精炼之后，利用相关系数法对精炼的特征字段进行检验，确定精炼特征集合与目标特征 SOC 之间的相关性情况。在众多相关系数法中，由于受到离群点的影响较小、应用约束少等优点，斯皮尔曼相关系数法非常适合本项目，故最终采用斯皮尔曼相关系数

法进行特征字段相关性的检验工作；需要注意的是，在计算之前需要过滤数据集中的空值记录。经过计算斯皮尔曼系数矩阵如图3.1所示。

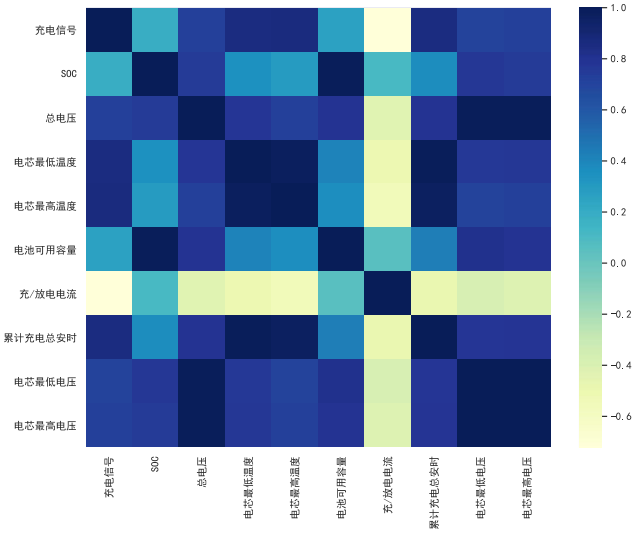

图 3.1: 斯皮尔曼相关系数矩阵

可以发现，在“SOC”特征字段这一列，特征字段几乎都具有良好的相关性。“温度”字段由于温度本身并不是一个变化十分敏感的特征，故其相关性表现并不是很明显。“充/放电电流”特征字段由于其充电过程为负值放电过程为正值，故与其它特征的相关系数为负数呈现逆向相关。

3.3 数据清洗

利用新能源电池特征字段的物理含义进行特征选择之后，需要基于特征字段的表征类型对数据进行清洗去除其中的异常值以及空值样本从而提高数据集的质量。如图3.2所示，现有的新能源数据集的样本中存在大量的空值区域和异常值。

空值主要由两个原因造成：（1）数据丢失，主要发生在数据采集的过程中，即新能源汽车处于工作状态，但是采集的数据在时间上并不连续；（2）数据采集机制，数据采集过程中，只有当新能源汽车处于工作状态时采集系统才会收集整车的数据信息，而当新能源汽车处于停车状态采集系统则不会记录当前车辆的信息。与新能源电池特征采集策略不同，采集系统的”时间“信息是不断更新的，导致在采集到的数据记录中存在大量不规则的空值，这些空值与新能源汽车的工作状态有关；异常值则由采集过程中传感器的异常所导致。

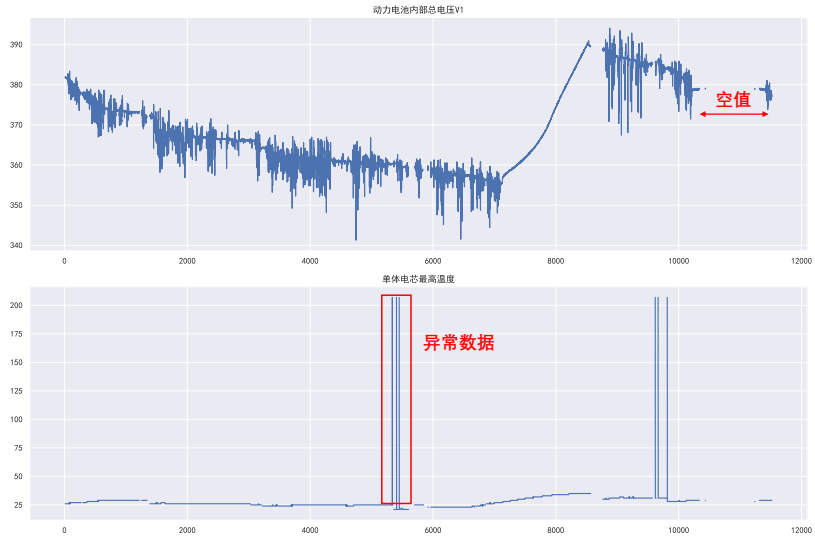

图 3.2: 异常值与空值样本曲线

3.3.1 空值处理

本项目采用直接舍去法进行数据空值处理，即只要筛选的特征字段中有任何一个特征字段为空值那么就舍弃整个数据样本。插值形式的空值处理方式比较适合空值比例较少的情况，由于数据采集策略的影响新能源数据集中存在大量的空值样本，无论使用上述哪种方式进行空值的插入都会引入干扰。空值区域越长插入的数据越多引入的干扰因素就越大，尤其是成片区域连续为空值的情况，而此类情况在本项目的数据集中出现的情况较为频繁，故最终舍弃插值的空值处理方式。

需要注意的是采用直接舍弃空值数据样本的方式也会带来其他问题。新能源电池当前的状态与过去历史工作过程息息相关，直接舍去空值样本会让新能源电池的历史信息变得不完整，这会对后面模型训练和数据扩增造成一定的影响。为了解决直接舍弃空值导致的历史信息破碎问题，需要引入额外的处理算子对清洗后的新能源数据集进行切片，将一份数据集分成多个独立的工作过程，多个工作过程之间彼此独立互不影响。具体算子将在后续章节详细介绍。

3.3.2 异常值处理

异常值的处理分为三个步骤完成，包括：划定特征字段合理双门限范围，筛选特征字段记录中的异常值，以及处理异常值。本项目采用箱形图算子进行特征字段合理边界的划定工作。利用箱形图算子计算出特征字段的合理双门限，之后基于合理双门限对数据集进行筛选，大于高门限和小于低门限的特征值被划为异常值。需要注意的是，新能源特征字段边界的统计工作，不能仅仅加载一个

数据文件就直接利用箱形图算法计算出对应特征的边界，需要以整个数据集为源进行参照，否则会导致双门限过于严格将很多正常数据划分为异常样本。合理的特征值边界要基于整个数据集进行，越多的数据统计出来的边界越可靠。筛选出所有的异常值后，首先将此类异常值置为空值，然后再走一遍空值处理过程对异常值进行清洗。这里必须先对新能源电池的数据集进行空值处理，然后才能利用箱形图算子筛选新能源电池目标特征字段的异常值，导致数据需要经历两次空值处理过程。理论上来说，如果先对新能源电池目标特征字段进行异常值的筛选，然后将筛选出的异常值置为空值，接着再进行空值处理，这样的流程只需要进行一次空值处理过程，流程更优但是却很难实现，因为箱形图算法对于空值虽然不是那么敏感，但是当一个目标特征字段存在很多空值时箱形图算子就会变得不稳定。因此从项目的可靠程度来说，本方案是先做的空值处理剔除特征字段中空值的部分，再利用箱形图算子进行异常值的筛选，最后进行空值处理。

3.4 本章小结

本章首先介绍了客户提供数据集的所有特征字段，包括新能源电池管理系统相关的部分、新能源汽车电机相关的部分以及其他部分特征字段。为了更好的利用数据同时压缩特征维度，本项目对原始数据集进行特征筛选和特征精炼工作。特征筛选首先过滤与新能源电池荷电状态估计无关的特征，再利用特征精炼进一步压缩特征字段个数，最后基于斯皮尔曼相关系数法对精炼后的数据特征进行了相关性分析。本章接着介绍了本项目采用的空值处理、异常值检测和处理方法。首先对空值进行处理，对包含空值的样本直接进行丢弃，再利用箱形图算子计算数据特征的双门限范围，过滤样本中的异常数据样本。本章节将用户提供的原始数据集进行预处理，提取成质量更高、信息密度更大的数据集，供后续扩增和模型训练任务使用。

第四章　基于传统算法新能源电池数据扩增

本章节利用扰动（jittering）、窗口翘曲（warping）和窗口平滑（window smoothing）三种传统时序扩征策略，结合新能源电池目标使用场景设计扩增算子，从而在数据空间实现时序扩增，场景包括：扰动场景、数据丢失场景、以及噪声抑制场景。扩增源数据是数据预处理章节对实车采样数据集进行处理后的预处理数据集。

4.1　面向扰动场景的传统算法数据扩增

扰动场景数据扩增基于高斯白噪声设计算子，从频域和时域两个角度引入噪声干扰。新能源电池扰动场景是指新能源汽车在使用过程中，BMS 由于传感器不断老化或者电路电流噪声影响，使得采集到的新能源电池各项物理特征时序数据中存在噪声数据。这类噪声数据一般以高斯白噪声为主，即呈现出正态分布的特性。系统在时域引入干扰，噪声会覆盖频域的所有频段；而频域高斯白噪声以频率作为基础对选定的频域增加噪声，没有选中的频段以原始数据输出，故频域噪声扰动的扩增粒度比时域噪声扰动的扩增粒度更加细致。

在频域引入噪声需要先对新能源电池数据样本进行傅里叶变换得到新能源电池系统的频域响应，在频域引入噪声后通过傅里叶逆变换算子将频域数据恢复到时域。这个过程对新能源电池管理系统的采样率提出了要求，需要满足奈奎斯特采样定理，即新能源汽车搭载的 BMS 采样率要大于电池电流周期的 2 倍。分析发现 BMS 采样率为 10 秒/次故其采样频率为 0.1Hz，而新能源汽车供电系统的工作电流为直流，其频率为 0Hz。因此本系统满足奈奎斯特采样定理不会产生频谱混叠的情况，对时序信号进行傅里叶变换后能够基于傅里叶逆变换恢复原始信号。

4.1.1　时域、频域扰动算子设计

在时域添加扰动时，产生的随机高斯白噪声序列长度与新能源电池时域特征序列长度一致，并且噪声是直接添加在新能源电池采样序列上。而在频域添加扰动则稍微复杂一些，需要先对新能源电池序列样本进行快速傅里叶变换，将其特征序列从时域变换至频域，然后以序列长度的中间点，即长度的一半作为区分点将频域划分为低频区域和高频区域：初始点至中间点之间作为低频区域，中间点至终点之间作为高频区域。需要注意的是，在进行快速傅里叶变换之前，需要对原始的新能源电池特征序列进行分帧操作，帧长一般为 256 的整数倍，然

后对每一帧特征序列进行上述操作，不足一帧的特征序列要么对结尾进行补 0 构成完整一帧，要么直接舍弃不足一帧的数据。由于低频区域表现内容，高频区域表现细节，对低频添加噪声的影响要高于在高频部分添加噪声。本项目利用高斯分布随机产生新能源电池特征序列长度一半数量的噪声序列，将其添加至频域特征序列的低频区域同时维持高频区域数据不变。扰动添加完成以后再利用快速傅里叶变换的逆变换算子将新能源电池频域特征序列变换为时序特征序列。扩增过程如算法1所示。

Algorithm 1: 面向扰动场景的传统算法数据扩增

Input: The target input feature sequences of the new energy battery, INP.

Output: The time domain jittering augmentation of the input sequences, TOUT; And the frequence domain jittering augmentation of the input sequences, FOUT;

```
/* time domain augmentation */
noise ← numpy.random.normal(size=len(INP));
TOUT ← INP + noise;
/* frequency domain augmentation */
frame_len = 1024;
N ← len(INP) // 2;
noise ← numpy.random.normal(size=N);
frames ← split INP into frames with frame_len = 1024;
foreach frame_data in frames do
    fft_data ← apply fft algorithm to frame_data;
    fft_data[0:N] += noise;
    fft_r ← apply ifft algorithm to fft_data;
    update the frame_data with fft_r;
FOUT ← frames;
```

4.2 面向数据丢失场景的传统算法数据扩增

本项目面向数据丢失场景，基于窗口翘曲策略设计了两种数据丢失算子。如图3.2所示，在数据预处理章节中，对采集的新能源电池数据分析发现：新能源电池特征序列样本的空值分为两种情况。一种空值为新能源汽车停止工作期间，由于新能源汽车 BMS 的采样策略导致的空值。汽车在停车状态不会采集新能源电池的特征，但是新能源电池数据文件的写入工作并没有停止，BMS 仍然会

不间断的以 10 秒为周期向文件中插入数据只不过插入的数据样本字段除了车辆编号、时间以及一些与新能源电池特征无关的字段外其余字段皆为空值。另一种空值则更为隐晦很难发现，尤其是数据样本非常大的情况下，数据在采样过程中并不连续存在数据丢失的情况。汽车在行驶或者充电过程中新能源电池管理系统应当以 10 秒为周期均匀的进行数据采样，但是实际分析中却发现并非如此，数据会随机性的发生丢失现象，即两条相邻的数据样本之间的时间戳大于 10 秒且为 10 秒的整数倍。针对这两种数据丢失情景，本项目利用窗口翘曲策略的压缩（下采样）算子设计了两种扩增方法：对称性和非对称性数据扩增方法。

4.2.1 对称、非对称数据丢失算子设计

扩增算子的对称性和非对称性主要描述数据丢失策略是均匀的丢失还是非均匀的丢失。

对称性数据扩增先对新能源电池的目标特征序列进行分帧，对于尾部不能构成完整一帧的部分直接舍去。帧长按照需要丢失数据的比例自适应调整，比如如果需要丢失 10% 的数据，按照目前特征样本的个数计算出需要丢失的样本数目，由于每一帧样本只会丢失一个数据，因此可以知道具体需要分为多少帧，知道目标帧数和特征字段的样本总数即可求出一帧的长度。需要注意的是，在对称性数据丢失扩增算法中每一帧的丢失样本在本帧内的位置是一致的，即如果第一帧丢失的是第一个样本，那么后续帧中丢失的也必须是第一个位置的样本。

非对称数据扩增与对称数据扩增最大的不同在于帧内丢失数据采用非等距的方式进行，与对称数据扩增一致的是在一帧内部同样只会丢失一个数据样本。非对称数据扩增算子同样先基于丢失比例信息对新能源电池的目标特征字段序列自适应进行分帧，分帧方式与对称数据扩增过程一致。接着基于数据筛选策略在每帧内部筛选出需要丢失的目标样本。本项目基于关键点策略进行帧内丢失样本的筛选工作，将帧内目标特征字段变化最大的点定义为关键点，即 $\Delta = |next - prev|$ 最大的点。以新能源电池的电压特征序列为例，如果在相同的采样间隔内，某个采样间隔内的电压变化 ΔV 比其他采样间隔内要大，那么可以确定的是，此采样间隔内新能源电池经历了比其他采样间隔更剧烈的过程，此过程可以是充电过程也可以是放电过程。这样的点比其他点在描述新能源电池的历史过程时具有更好的代表性，非对称数据丢失扩征算子就是针对这类重要点进行丢失的扩增算子，通过特征序列样本的 Δ 增量筛选数据丢失的目标样本。

对称、非对称数据丢失扩增过程如算法2所示。没有使用其扩展（上采样）策略主要是因为新能源电池的物理特性和其高分子材料的特殊性。新能源电池的

Algorithm 2: 面向数据丢失场景的传统算法数据扩增

Input: The target input feature sequences of the new energy battery, INP; dropout rate, DP.

Output: The symmetric augmentation of the input sequences, SOUT; And the asymmetric augmentation of the input sequences, AOUT;

```
dropouts = DP * len(INP);
frame_len = len(INP) // dropouts;
frames ← split INP into frames with frame length = frame_len;
foreach frame_data in frames do
    /* symmetric augmentation */
    SOUT.append(data) for data ← dropout the first point in frame_data;
    /* asymmetric augmentation */
    delta = abs(frame_data[1:] - frame_data[:-1]);
    indices = numpy.argmax(delta);
    AOUT.append(data) for data ← dropout the indices point in frame_data;
```

物理特征字段包括：电压，电流和温度，如果要采用窗口翘曲的扩展（上采样）算法则需要提前构建一个电流到电压的映射关系。温度由于变化不明显可以在插值过程中使用前后的均值，而电流的序列值是驾驶员行为的表现无法像温度那样采用前后序列均值的方式进行插值，就算能够这样做，其对应的电压值序列也不是前后均值能够表现的。窗口翘曲的压缩（下采样）方法则不同，对新能源电池的目标特征序列进行下采样并不会插入额外的历史过程，其中的数据丢失可以理解成在相同的时间内新能源电池的工况更加的剧烈，比如 20 秒为采样周期的新能源电池管理系统所采样的电池特征序列特征变化情况一般来说要比以 10 秒为采样周期的新能源电池管理系统所采样的电池特征序列特征更加的剧烈。因此，对于新能源电池数据丢失场景数据扩增只使用了窗口翘曲传统算法的压缩（下采样）算子没有使用扩展（上采样）算子。

4.3 面向噪声抑制场景的传统算法数据扩增

面向噪声抑制场景的数据扩增围绕新能源电池物理特性在采样过程中呈现随机非稳态特性进行实现。理论上来说，新能源电池的电压变化是一个平稳的过程，但是实际新能源电池管理系统采样得到的电压特征序列是一个跳变非常剧烈且离散的过程，如图4.1所示。本项目针对随机瞬时噪声基于滑动平均策略设计了两种算子，分别为窗口平均算子和指数平均算子。

在图4.1中，右上角是放大的充电过程而右下角是放大的放电过程。可以发

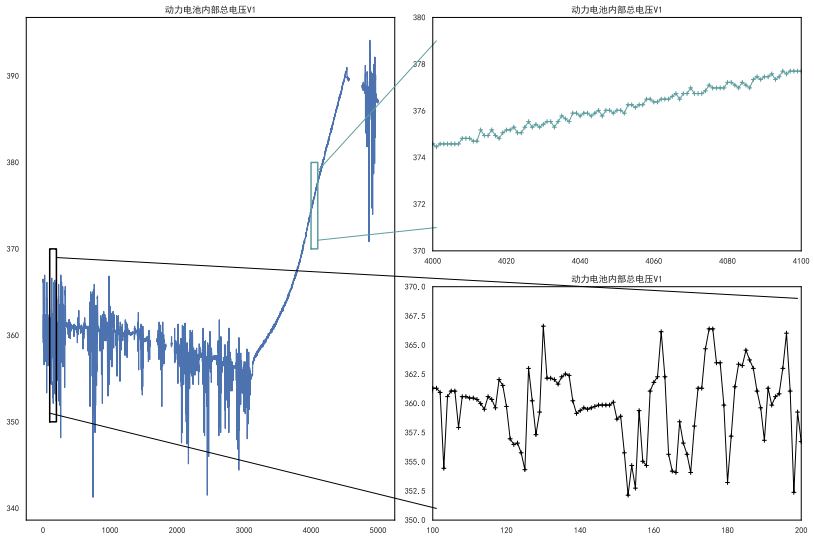

图 4.1: 新能源电池电压特征序列

现充电过程的电压序列比较平滑，但是放电过程的电压序列跳变比较严重，这是新能源电池的内部特性导致的。当新能源汽车以不同的速度行驶时，电池输出不同强度的工作电流，这是通过控制电池的外部负载实现的。新能源汽车突然加速或者减速的情况下，新能源电池的外部负载会发生瞬时变化，导致新能源电池的端电压发生瞬间的跳变。此时新能源电池管理系统采集的端电压特征值并不能正确的反应出当前新能源电池的端电压状态。正确测量新能源电池的端电压需要满足一定的条件，一般要求电池在运作一段时间后需要静置一段时间，让新能源电池内部的分子特性恢复稳定状态，然后测量才能获得比较准确的特征值。然而，新能源汽车在行驶过程中是无法达到上述条件的，为了尽可能地获取相对较准确的端电压值，排除外部负载瞬间变化导致电池端电压瞬时变化，本项目利用平滑算子抑制瞬时噪声从而生成扩增样本，平滑算子包括窗口平均和指数平均两部分。

4.3.1 窗口平均、指数平均算子设计

窗口平均算子需要一个超参数，即窗口大小，该算子对每一个时刻引入的特征值会根据当前窗口的历史信息计算均值，用计算出来的均值作为当前时刻的特征真实值。窗口平滑算子的公式如下：

$$\hat{V}_t = \frac{1}{N} \sum_{i=t}^{t-N+1} V_i \tag{4.1}$$

其中，V_t 在时刻 t 的新能源电池的电压值，N 表示窗口的大小。可以发现，窗口平滑的本质就是以当前的采样值为窗口的最后一个点向前求加权平均。指数平

滑算子的本质和窗口平滑类似，但是其并没有窗口长度这个超参数，取而代之的是一个衰减参数。窗口平滑只关心同一个窗口内部的数据情况，而指数平滑则关心更加长远的历史数据。指数平滑算子的公式如下：

$$\hat{V}_t = \beta \cdot V_{t-1} + (1-\beta) \cdot V_t \tag{4.2}$$

$$\hat{V}_{biased} = \frac{\hat{V}_t}{1-\beta^t} \tag{4.3}$$

上述公式中，β 是超参数衰减指数一般来说由用户指定，参数介于 0.9 和 0.999 之间，设置为 0.9 时可以近似为对过去 10 个时刻的历史求加权平均，设置为 0.999 时则为过去 100 个点求加权平均；$\hat{V}_{biased}$ 是指数平滑后在时刻 t 的最终特征电压值。公式4.3的目的是为了更新最初的几个电压值，如果没有这一步电压序列的初始值将从非常接近 0 的值开始。

面向噪声抑制场景的传统算法数据扩增通过窗口平滑和指数平滑两种平滑算子实现，其中前者仅仅关注同一个窗口内部的历史数据变化情况，窗口长度用户可配；而后者关心的历史数据周期更长，其平滑效果比窗口平滑更加优异。两种扩增方法的实现过程如算法3所示。

Algorithm 3: 面向噪声抑制场景的传统算法数据扩增

Input: The target input feature sequences of the new energy battery, INP; window size of the WMA algorithm, WinLen; decay rate of the EMA algorithm, DECAY.

Output: The WMA augmentation of the input sequences, WMAout; And the EMA augmentation of the input sequences, EMAout;

```
/* WMA augmentation */
/* using convolve algorithm to accurate the execution */
win ← a sequences of 1 with a length of WinLen;
WMAout ← convolve win with INP;
/* EMA augmentation */
temp = 0;
t = 0;
foreach data in INP do
    t = t + 1;
    temp = temp * DECAY + data * (1 - DECAY);
    data = temp / (1 - np.power(DECAY, t));
    EMAout.append(data);
```

4.4 本章小结

本章详细介绍了如何利用传统序列扩增策略实现面向扰动、数据丢失以及噪声抑制三种场景的数据扩增方法，并且每种扩增策略都提供了两种方式，从不同角度对源样本进行扩增。每一个小节首先介绍了传统序列扩增策略的基本原理以及和目标场景之间的相关性，然后给出了扩增策略的核心算法公式和详细的算法执行流程。

第五章　基于智能模型新能源电池数据扩增

本章主要介绍如何构建新能源电池智能扩增模型，并利用该模型在给定工况条件下实现针对 SOC 的数据扩增。

5.1　基于 Seq2Seq 构建扩增模型

5.1.1　模型结构

本项目构建的智能扩增模型是一个 Seq2Seq 模型，其接收两个时序输入，分别为头部信息序列和工况信息序列。头部信息序列，表示当前扩增目标的历史过程，模型可以从中自适应的提取表征当前目标电池健康度信息的 feature map；工况信息序列，表示新能源电池的目标扩征过程，核心组成部分是一段时间的电流工作序列。新能源智能扩增模型基于头部信息序列提取的电池健康度编码和给定的目标工况预测对应的电压响应序列。整个过程可以用以下公式表示，用 $\mathcal{F}$ 表示模型的非线性映射：

$$(\mathbf{X}_{head}, \mathbf{Y}_{states}) \longrightarrow \mathcal{F} \longrightarrow \mathbf{V} \tag{5.1}$$

其中，$\mathbf{X}_{head}$ 表示头部信息序列；$\mathbf{Y}_{states}$ 代表工况信息序列；$\mathbf{V}$ 表示智能扩增模型基于头部信息序列和工况信息序列预测的电池响应电压序列。

基于 NIN（Network In Network）原理，本项目在 LSTM 层的基础上封装了两个独立的层，分别为 Observer 层和 Generator 层。Observer 层用于对新能源电池健康度信息进行提取和编码，该层将一定时间步的头部信息转化为一个 feature map。Generator 层作为数据扩增的核心有两个输入张量，一个是 Observer 层输出的 feature map，用于初始化 Generator 内部的 LSTM 层；另一个输入张量是新能源数据扩增的目标工况序列，初始化后的 Generator 层将基于目标工况序列预测对应响应电压序列。

智能扩增模型结构如图5.1所示，可以看出智能扩增模型的输入层有两个输入张量，分别是 shape 为 (None, None, 5) 的 observer_input，以及 shape 为 (None, None, 6) 的 generator_input。observer_input 的特征维度为 5，而 generator_input 的特征维度是 6，区别在于前者比后者少一个“充电标志位” **Flag** 特征字段。observer_input 不需要“充电标志位” **Flag** 特征字段是因为 observer 层主要目的是基于输入历史信息对新能源电池的健康度进行编码，而“电池充电标志位” **Flag** 与新能源电池的健康度并没有直接或者间接的逻辑关系，仅仅是表征电池工作状态的一

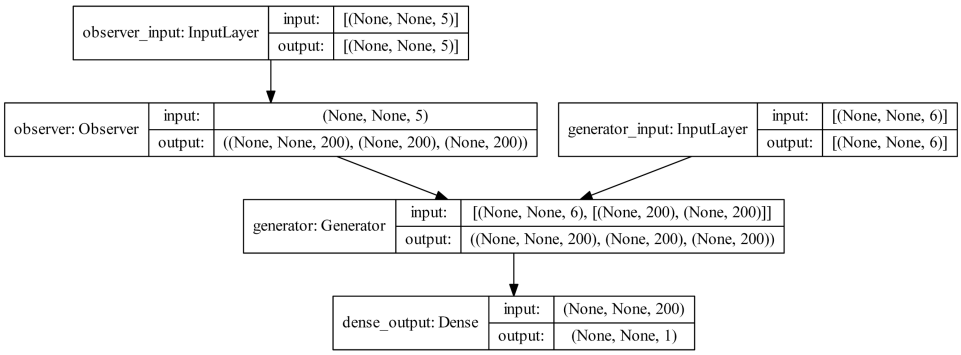

图 5.1: 扩增模型结构

个信息量。

5.1.2 构建训练数据集

构建训练数据集主要包括：数据融合、过程切分、以及 batch 提取三个步骤。数据融合，基于表3.3提取的关键特征字段进行数据融合，从而降低数据特征的维度。首先，利用“动力电池剩余电量 SOC”特征字段和“动力电池可用容量”特征字段求出当前电池的最大容量，其公式如下：

$$\mathbf{cap_{max}} = \frac{\mathbf{cap} * 100}{\mathbf{SOC}} \tag{5.2}$$

其中 **cap** 和 **SOC** 分别代表时间序列中的“动力电池可用容量”和“动力电池剩余电量 SOC”特征字段；$\mathbf{cap_{max}}$ 即新能源电池“当前最大容量”特征。然后，基于“单体电芯最低温度”和“单体电芯最高温度”特征字段计算新能源电池当前温度，一般以均值的方式表征。接着，对“快充唤醒信号”和“慢充唤醒信号”特征字段进行求或运算，结果作为表征新能源电池的“充电唤醒标志”特征字段。最后，对于“单体电芯最高电压”、“单体电芯最低电压”、以及“动力电池内部总电压 V1”三个电压相关的特征字段，本项目使用“动力电池内部总电压 V1”作为表征新能源电池电压的特征字段，另外两个特征字段保留存在并没有直接使用。数据融合之后，在预处理数据集特征的基础上确定了 6 个特征字段，缩小了数据集特征的维度。数据融合后的特征字段分别为：累积充电总安时 **Chg**、当前最大容量 **Cap**、温度 **T**、电压 **V**、电流 **C**、以及充电标志 **Flag**。

过程切分，负责将一份数据文件中的数据样本按照时间信息先拆分成多个

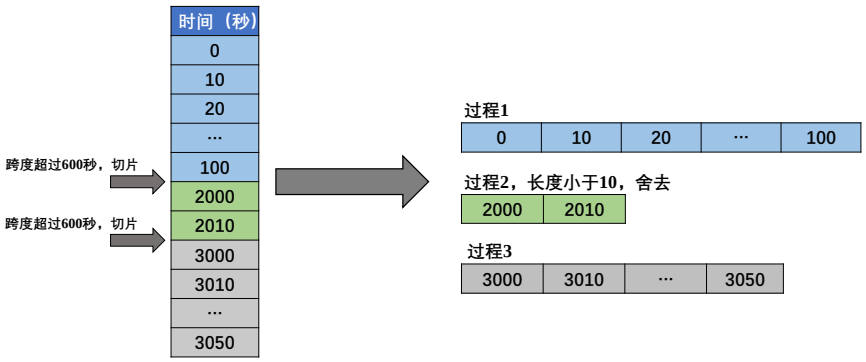

图 5.2: 按照时间切分工作过程

独立的工作过程，如图5.2所示。一个连续的工作过程需要同时满足两个条件：（1）同一个工作过程内的数据样本，其在时间维度上连续，即两条数据记录的时间差不能超过一定范围；（2）工作过程需要持续一段时间，太短的工作过程由于其信息量比较少并没有观察的价值。基于上述两点要求，本项目在过程切分中设置时间限度为 600 秒且一个工作过程内的样本条目要超过 10。该设置表示当相邻两行记录的时间差大于 600 秒时认为上下两行处于不同的工作过程，需要对数据进行拆分，如果拆分后的样本个数少于 10 则不能看作一个完整的工作过程，直接丢弃。

batch 提取，负责将数据集提取成统一的格式，能够用于模型的训练和验证工作。模型在训练过程中一次读入数据的大小称为一个 batch，采用 LSTM 层的智能扩增模型其输入张量 shape 为（批次，时间步，特征维度），数据 batch 提取过程如图5.3所示，图中按照时间步为 4 进行 batch 提取。一个 batch 又可以分为输入特征序列和对应结果标签序列，输入特征序列和对应标签序列具有相同的时间步长度，标签序列是输入序列向前一个时间步进行采样的结果。提取到一个 batch 后前进一个时间步，即从时间步 $t = 1$ 移至时间步 $t = 2$。然后，利用相同的方法提取第二个训练 batch，如此循环往复直至 batch 数据集中的标签序列达到工作过程数据的末尾。

由于训练智能扩增模型需要两个独立的输入序列，分别是 observer_input 和 generator_input，还需要进一步拆分图5.3中提取的 batch 数据，具体过程如图5.4所示。首先，将提取的 batch 数据的输入序列切分成两部分：头部信息 head_info 和状态信息 state_info，其中前者作为 observer_input 的输入，后者是 generator_input

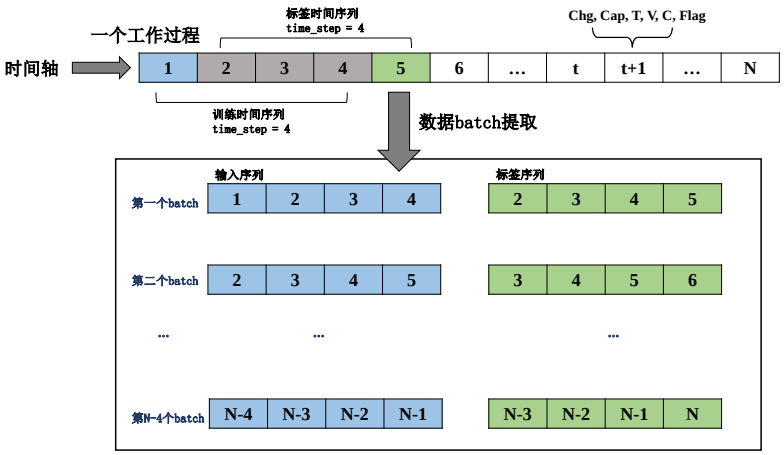

图 5.3: Batch 提取过程

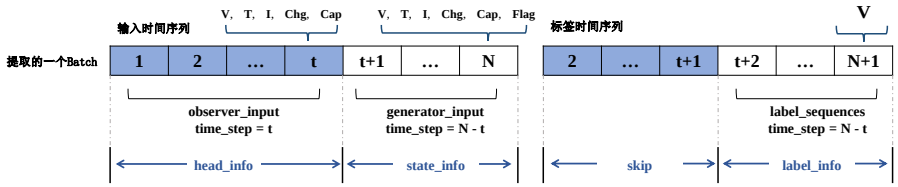

图 5.4: 训练过程输入特征序列

的输入。然后，按照 head_info 和 state_info 的时间步长度拆分 batch 中的标签序列，同样将其分成两部分：第一部分用来提取新能源电池的健康度信息，由于此部分并不会产生对应的电压响应，因此这部分的标签数据直接丢弃；第二部分才是模型训练过程中真正的标签数据，每个时间步只有一个特征，即电压特征字段，时间步与输入序列 state_info 的时间步长度一致。头部信息结构可以用 $(M, \text{HeadLen}, \text{HeadFeatures})$ 表示，其中 HeadLen 为智能扩增模型训练数据集的头部信息长度；HeadFeatures 是每个头部信息中所包含的特征字段个数，本项目其值为 5，包括新能源电池内部总电压特征字段、新能源电池温度特征字段（单体电芯最高温度和单体电芯最低温度的平均值）、新能源电池累计充电量特征字段、以及新能源电池当前容量特征字段，可参考5.4；M 表示一次送入多少个头部信息序列进行增模型训练工作。

5.1.3 数据归一化

本项目采用最大最小值归一化方法对“电压”**V**、“温度”**T** 和“当前最大容量”**Cap** 征字段进行归一化。对于“累积充电总安时”**Chg** 和“电流”**C** 特征字段则稍微复杂一些，因为二者都有很高的动态区间并且分布很不稳定。“累积充电总安时”特征字段虽然理论存在最大值但是无法统计；而“电流”特征字段数据分布太广，如果直接采用最大最小等间距归一化方法会丧失很多细节。为了解决上述两个问题，针对这些广动态范围或者分布不稳定的特征字段，本项目采用非对称的归一化方法。对于“累积充电总安时”特征字段，首先取对数提升较小数值拉低较大数值，使得特征字段中那些较小区域和较大区域的值向中间靠拢。然后再对取完对数后的值直接除以一个系数实现归一化操作。需要注意的是针对“累积充电总安时”特征字段，并没有将其严格的归一化到 0 和 1 之间。由于“电流”特征字段隐含充放电过程信息，电流为正表示放电而电流为负表示充电，为了尽可能的保留此隐藏信息，本项目将“电流”特征字段归一化到-1 和 1 之间。归一化过程和“累积充电总安时”特征相似，首先采用取对数的方式缩小较大值放大较小值，再根据电流的正负性利用不同的参数进行归一化，其公式如下：

$$C_i = \operatorname{sgn}(C_i) \cdot \frac{\log(|C_i|)}{\log k} \tag{5.3}$$

上式中，如果 C_i 大于 0，k 为 450 否则 k 等于 400，通过 sgn 函数保留了电流的正负号。

表 5.1: 特征字段归一化参数

	最小值	最大值	缩放系数
V	310	420	NA
C	-400	450	NA
T	-20	60	NA
Cap	120	150	NA
Chg	NA	NA	7
Flag	NA	NA	NA

本项目归一化过程中使用的参数如表5.1所示，其中 **V**、**C**、以及 **Cap** 的归一化参数从客户提供的新能源数据集分析获得；**T** 参数基于新能源汽车可能的工作环境情况决定；**Chg** 则由开发人员经验给出。

5.1.4 模型训练

模型的训练过程需要定义损失函数，选择优化算子，以及确定评估标准。本项目选择均方误差 MSE 函数作为模型训练的损失函数；优化算子采用目前使用最为广泛的 Adam 自适应优化算子，其优势在于可以不用手动配置学习率，在模型的训练过程会基于训练进度动态的调整学习率；本项目使用平均绝对误差 MAE 和决定系数 R2_core 作为模型验证集的评估函数。

扩增过程如图5.6所示，智能扩增模型会基于前一个时刻 $t-1$ 的工况信息预测当前时刻 t 的电压值，而预测的电压值会作为下一个时刻 t 工况序列的一部分用于预测 $t+1$ 时刻的电压值，如此循环往复直到工况序列达到最大长度。本项目将上述预测流程称为迭代预测。由于当前的预测结果依赖前一个时刻的预测状态，导致误差会随着迭代预测长度的增加不断累积，越后面的结果预测误差越大。为了解决模型迭代预测过程中误差不断累积的问题，本项目在模型的训练和部署过程采用不同的数据流结构。在模型的训练过程引入 Teacher Forcing 机制，Teacher Forcing 机制假设每次预测结果都是正确的，在模型训练过程直接使用真实值作为标签来加快模型的收敛速度，模型训练过程如图5.5所示，训练过程将迭代预测的循环联系断开，而部署阶段则采用实际的预测结果作为依据进行迭代预测。部署阶段扩增过程如图5.6所示，智能扩增模型每次的预测电压值都会作为下一个时刻模型输入的一部分。

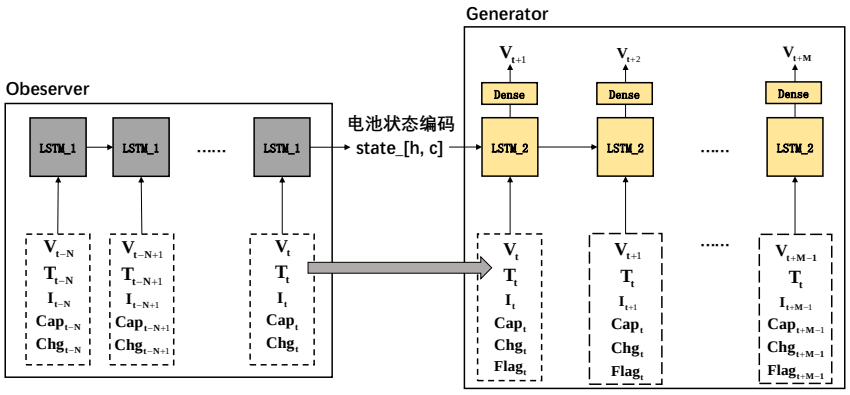

图 5.5: 采用 Teacher Forcing 机制的训练过程

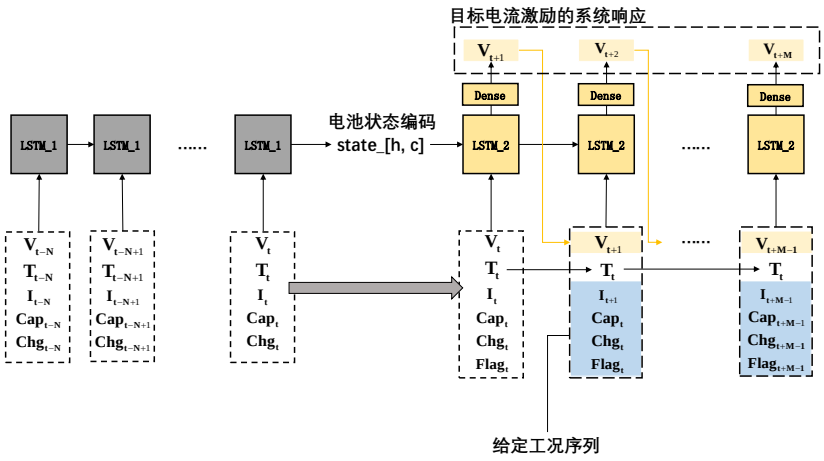

图 5.6: 模型预测过程

5.2 模型验证

新能源电池扩增样本和图像扩增不同，开发人员很难直观判断扩增样本是否正确，模型验证旨在利用可量化的指标对扩增模型预测精度进行检验。智能扩增模型验证过程如图5.7所示，将验证数据样本数据以帧为单位送入扩增模型，经过迭代预测后智能扩增模型输出给定工况下的电压序列值，比对扩增电压序列和真实电压序列的差异完成模型的验证工作同时画出对比曲线。从中可以发现 observer 层和 generator 层工作模式的区别：前者在一个预测周期内仅仅运行一次，而后者需要运行多次，运行的次数等于预测帧的长度。

训练完成后使用 TensorFlow 提供的 save_weights 接口保存参数，此接口会将模型的所有参数保存至一个 h5 格式的文件中，但是并不会保存模型的结构、损失函数的状态、以及当前优化目标函数的状态等信息。为了实现迭代预测方式，模型验证的第一步就是初始化扩增模型对象。首先，使用相同的代码在内存中创建一个模型对象，然后调用 load_weights 接口基于保存的 h5 格式文件重新加载模型参数。接着，使用 get_layer 接口从创建的模型对象中分离出 observer 层和 generator 层，并用 Model 接口创建两个新的模型对象。然后，对测试文件执行与模型训练数据处理相同的流程得到 head_info 和 state_info 信息，将 head_info 送入 observer 对象输出新能源电池的健康度编码，再将 state_info 中第一个时间步的特征信息和新能源电池健康度编码特征矩阵打包成一个 list 对象送入 generator 对象输出第一个预测电压值，以及当前的 LSTM 层内部状态信息。紧接着，利用输出的预测电压修改 state_info 中第二个时间步的电压特征字段的值，并更新第二个时间步中温度特征字段使其等同于上一个时间步的温度以此忽略温度

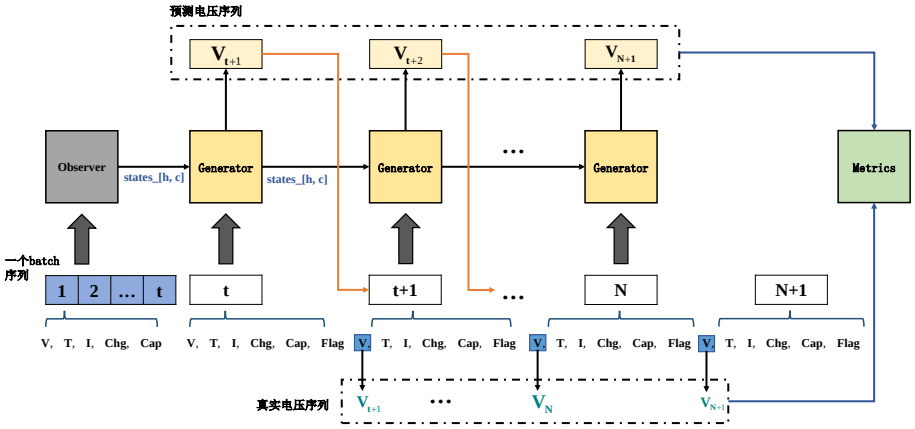

图 5.7: 模型检验过程

带来的影响，将更新后的 state_info 第二个时间步特征张量送入 generator 对象输出第二个时间步对应的预测电压值，如此循环往复得到完整一帧的预测电压值。最后，用预测帧的预测电压特征序列和真实电压特征序列进行对比，基于验证指标计算量化的模型评估结果。

5.2.1 检验指标

本项目采用均方根误差（RMSE）算子计算图5.7中预测电压序列和真实电压序列之间的偏差，完成对扩增模型迭代预测效果的评估工作。验证的输入为一个或多个 csv 文件，每个 csv 文件长度并不一致，可能存在一帧、多帧、甚至不足一帧的情况。由于模型验证以帧为单位进行电压序列的预测，对于包含多帧内容的 csv 文件，一个完整的验证过程扩增模型需要重复运行多次，每一次运行都会产生一个当前帧的 RMSE 值和最大方根误差（MRSE）值。

Algorithm 4: 验证算法执行过程

Input: The path of the h5 file that save the weights of the trained augment model, H5_PATH; The input file to validation the trained augment model, INP_FILE;

Output: The metrics list of the trained model;

```
model ← load_model(units, head_dim, info_dim);
model.load_weights("H5_PATH");
obs_m ← get_observer(model);
gen_m ← get_generator(model);
data ← preprocess(INP_FILE);
metrics = {};
foreach frame in data.to_frames() do
    head, info = split_batch(frame);
    obs_out, obs_state_h, obs_state_c = obs_m.predict(head);
    init_state = [obs_state_h, obs_state_c];
    inpseq = head.last_step();
    currents = info.get_current();
    out = [];
    for i to frame.length() do
        gen_out, gen_state_h, gen_state_c = gen_m.predict([inpseq] +
          init_state);
        init_state = [gen_state_h, gen_state_c];
        V = gen_out[0, 0, 0];
        update inpseq through V and currents;
        out.append(V);
    calculate mse, max mse each frame;
    metrics[frame] = [mse, max mse];
metrics["average mse"] ← calculate the mean value of all mse;
metrics["average max mse"] ← calculate the mean value of all max mse;
```

完整的模型检验过程如算法4所示，平均 RMSE 量化指标主要是从整体角度去度量模型预测的准确性，平均值可以忽略由于噪声引起的波动但是无法描述区域最大误差的情况，MRSE 指标来描述模型预测在一帧中最差的情况。

5.3 新能源电池目标工况下的数据扩增

本章介绍如何利用前面章节训练的智能扩增模型，基于给定目标工况实现面向新能源的数据扩增工作。智能模型数据扩增输入包括：（1）目标扩增样本的历史工作信息，旨在从电池的历史工作信息中提取电池的当前工作状态和健康度情况；（2）目标工况序列，确定电池的目标工作过程。扩增模型会根据两个输入序列生成对应的电压响应，用公式抽象描述如下：

$$\textbf{输入} \xrightarrow[\text{电池历史工作序列}]{\text{目标工况序列}} \mathcal{F} \longrightarrow \textbf{目标样本} \tag{5.4}$$

其中，$\mathcal{F}$ 表示扩增方法，包括智能扩增模型和传统变换算子。

表 5.2: 扩增样本特征字段

序号	字段名称
1	时间
2	快充唤醒信号
3	慢充唤醒信号
4	动力电池剩余电量 SOC
5	动力电池内部总电压 V1
6	单体电芯最低温度
7	电体电芯最高温度
8	动力电池可用容量
9	动力电池充/放电电流
10	累计充电总安时

由前面的分析可知，新能源电池可直接采集特征包括：电压、电流、温度，而新能源汽车 BMS 统计的特征有累积充电量，当前可用容量，时间，充放电标志等特征。因此一个完整的新能源电池扩增样本包括的特征字段如表5.2所示，其中“动力电池内部总电压 V1”特征字段为智能扩增模型基于给定目标工况和历史工作过程迭代预测生成，其他特征字段则根据新能源电池目标工况其中所提取的信息基于传统变换算子计算获得。

5.3.1 工况提取

工况提取旨在从新能源电池的数据样本中根据设定好的策略提取扩增目标工况序列。新能源电池工况序列提取同样以帧为单位进行，为了让电池的电压

特征变化更加明显，提取的一帧序列其能量变化越剧烈越好。进行工况提取，首先需要对输入数据样本进行分帧，然后基于分帧后的新能源电池充放电电流特征序列计算充放电深度，利用该深度表征本帧新能源电池的工作强度，新能源电池充放电深度计算公式如下：

$$P(i) = \sum_{j} \left(F_j * w_c * \left| c_j \right| + \left(1 - F_j \right) * (1 - w_c) * \left| c_j \right| \right) \tag{5.5}$$

其中 i 和 j 分表表示帧号和帧内点索引，F_j 表示当前时刻数据是充电状态还是放电状态，本项目定义充电状态 $F_j = 1$，放电状态 $F_j = 0$；w_c 表示充电状态下电流序列的权重，用于调整提取工况序列中充电序列和放电序列的比列；c_j 表示时刻 j 时的电流值，本系统充电状态为正值而放电状态为负值。利用上述公式对新能源电池预处理样本数据进行工况筛选，通过调整 w_c 权重参数控制提取目标工况中充电过程和放电过程的比例，尽量保证充电过程和放电过程的样本数目相等。

由于本项目新能源电池管理系统特征采集的周期为 10 秒，导致新能源电池充电过程和放电过程的很多细节都被丢弃，无法利用安时积分法对扩增后的数据进行剩余电量 SOC 标注。为了解决这个问题，本项目在工况提取阶段除了提取新能源电池充放电电流特征字段外，还提取了累计充电总安时、当前可用容量、快充唤醒标志信号、以及慢充唤醒标志信号特征字段。这些特征用于计算扩增样本中的“累计充电总安时”、“动力电池可用容量”、“动力电池剩余电量 SOC”、“快充唤醒信号”、以及“慢充唤醒信号”特征值。因此，工况序列是一个三维张量其结构为 $(N, \text{FrameLen}, \text{FrameFeatures})$，其中 FrameLen 表示一帧的长度；FrameFeatures 表示每帧数据的特征维度，本项目其值为 5；N 表示每次提取工况序列的个数。

5.3.2 数据扩增

从表5.2可知，扩增样本包含 10 个特征字段，下面将介绍如何在给定工况条件下生成这 10 个特征字段。本项目将扩增样本特征按照扩增算子的不同划分为 5 类，如表5.3所示，分别采用扩增模型、传统差分、帧内保持和用户定义等方法实现目标特征的扩增工作。

对于“动力电池内部总电压 V1”特征字段，直接利用智能扩增模型基于工况序列进行迭代预测生成。电压特征字段是新能源电池在当前工况激励下的响应，表征了新能源电池的内部状态情况，是扩增样本中的核心特征字段。

对于“动力电池可用容量”和“累计充电总安时”特征字段，首先利用差分方程 $\Delta = x(t) - x(t-1)$ 计算工况序列中每个时刻“动力电池可用容量”和“累计

表 5.3: 扩增样本特征字段分类

扩增模型	传统差分	帧内保持	用户定义	其他
动力电池内部总电压 V1	动力电池可用容量 累计充电总安时 动力电池剩余电量 SOC	单体电芯最低温度 单体电芯最高温度	快充唤醒信号 慢充唤醒信号 动力电池充/放电电流	时间

充电总安时”的变化 δ 值。然后基于头部信息最后一个时刻的特征向量，在对应特征字段上增加计算得到的每个时刻的变化 δ，从而获得对应时刻的“动力电池可用容量”和“累计充电总安时”特征字段值。对于“动力电池剩余电量 SOC”特征字段，在提取扩增样本头部信息过程中额外提取“动力电池剩余电量 SOC”特征值作为扩增基础，利用累积充电总安时法对其进行补齐，其公式如下：

$$SOC_{aug} = \min\left(SOC_{base} + \frac{\delta + cap_{base}}{cap_{max}}, 100\right) \tag{5.6}$$

上式中 SOC_{base} 和 cap_{base} 分别为头部信息中最后一条记录的“动力电池剩余电量 SOC”字段值和“动力电池可用容量”特征字段值；cap_{max} 可以利用“动力电池可用容量”和“动力电池剩余电量 SOC”特征字段值的比值求平均获得；δ 为之前差分方程 Δ 求得的每个时刻的变化；由于“动力电池剩余电量 SOC”特征字段的取值范围为 $[0, 100]$，故在公式的最后利用 min 算子和最大值 100 比较取其中的较小值，确保扩增后的特征字段值符合实际的规则。

对于“时间”特征字段的补齐则稍微灵活一点，本项目定义提取的头部信息第一个特征向量为时刻 $0s$，根据新能源电池管理系统 BMS 的采样周期逐步累加，即每产生一个扩增样本，其时间特征字段递增 $10s$。

对于“单体电芯最低温度”和“单体电芯最高温度”特征字段，虽然新能源电池的温度特征同样是对历史工况激励的响应，理论上来说其和新能源电池的工况序列和健康度信息是强相关的。但是分析发现，在一帧内新能源电池温度特征字段幅度的变化几乎可以忽略不计，故本项目采用帧内保持法将扩增样本头部信息最后一个记录的“单体电芯最低温度”和“单体电芯最高温度”特征值复制到扩增样本中去补齐。

需要注意的是，目标工况序列是从包含多个 csv 文件的新能源电池预处理数据集中提取出来的，而头部信息则是从扩增样本中获取，扩增样本一般为单个经过预处理后的 csv 文件。本项目支持用户自定义提取的工况序列个数和头部信息个数。假设用户定义需要提取的工况序列个数为 N，头部信息个数为 M，那么总的扩增样本个数为 $M * N$，即每一个工况序列会和所有的头部信息进行组合。

5.4 本章小结

本章介绍了如何利用智能扩增模型完成新能源电池的数据扩增工作，包括智能模型的训练、验证、以及扩增样本的特征补齐等内容。首先，描述了如何构建智能扩增模型，从模型结构、训练数据集提取、特征归一化方式、以及模型训练参数制定等方面对其进行了详细介绍。然后，提出了一种面向智能扩增模型的验证方法并确定验证指标，用于对部署的智能扩增模型进行验证，确保扩增出来的数据符合新能源电池的真实特性。最后，介绍了扩增样本中的特征字段以及各特征字段扩增所使用的扩增方法。

第六章　系统设计与实现

6.1　系统整体概述

本项目针对新能源领域电池特性和场景变化结合传统序列扩增算子和智能扩增模型实现面向新能源电池的数据扩增。传统序列扩增基于目标场景设计对应算子从数据空间对原始样本进行扩增。智能扩增模型则利用新能源电池特征之间的联系实现从已知激励到未知响应的推理，从而在特征空间对原始样本进行扩增。为了确保扩增模型生成数据的准确性，本项目进一步设计了模型验证模块来对部署的模型进行验证。

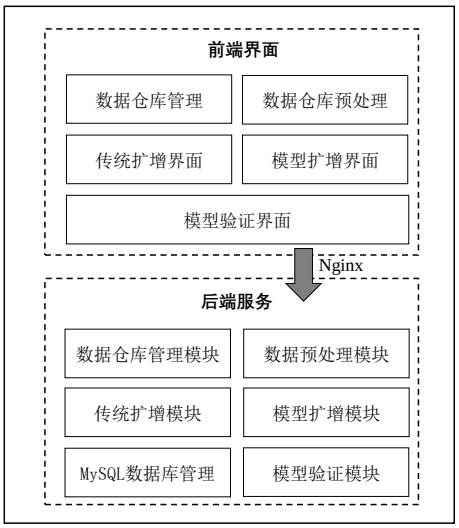

图 6.1: 系统整体模块图

系统整体模块如图6.1所示，从功能上来看，本系统主要包括数据仓库管理模块、数据预处理模块、传统扩增模块、智能扩增模块、模型验证模块、以及MySQL 数据库管理系统。数据仓库管理模块主要负责数据仓库的管理工作，包括数据集的上传、删除、以及信息显示工作。数据预处理模块主要负责对前端选中的原始数据集进行预处理，主要包括特征筛选、异常值清洗，以及空值处理等工作。传统扩增和智能扩增模块分别基于前端选中的数据集执行扩增工作，是本系统的核心模块。模型验证模块主要负责对部署的扩增模型进行验证，与图像扩增不同新能源电池的扩增样本为一个时间序列数据集，用户无法通过感官判断扩增数据集正确与否。模型验证模块验证部署的智能扩增模型的准确性

并对验证指标进行可视化展示方便用户理解。MySQL 数据库管理系统负责记录后端任务结果，保存数据仓库信息和状态等工作。

6.2 需求分析

系统设计首先需要采集用户需求，从而明确系统边界确定系统的任务和需要验收的指标内容。本章节从功能性需求、非功能性需求、以及系统用例三个角度进行需求分析，最后利用获取的需求指导系统设计。

6.2.1 功能性需求分析

功能性需求是本系统的核心需求，定义了系统主要的功能模块，是开发人员必须要实现的业务逻辑，对功能性需求的提取也是需求分析的核心目的。按照功能性需求的业务目标，可以将其分为数据管理需求、任务管理需求、以及扩增数据质量保障需求三个部分。其中数据管理需求围绕数据集的添加、删除、下载、数据清洗、以及状态显示等业务获取；任务管理需求是本系统的核心需求，围绕智能扩增、传统扩增、以及模型验证三个业务获取；扩增数据质量保障需求则围绕如何对智能扩增的数据集进行可靠性验证获取。

本项目最终获取的功能性需求如表6.1所示，数据管理需求主要包括数据集添加、数据集搜索、数据集删除、数据集状态显示、数据预处理、以及下载扩增结果六个功能。用户可以通过 shell scp 命令将目标数据集上传至服务器然后手动添加完成注册；或者先将目标数据集打包成 zip 格式，再利用前端界面的上传按钮将数据集上传至服务器完成注册。任务管理需求主要包括设置传统算法扩增策略、发起传统/智能扩增、执行模型验证、选择任务粒度、以及显示任务进度六个功能。数据扩增分为传统扩增和智能扩增两种方式，无论哪种扩增方式都需要支持显示任务进度的功能，对于智能扩增还需要支持选择任务粒度的功能。扩增数据质量保障需求包括模型验证指标可视化以及模型验证曲线可视化两个功能。质量保障需求确保智能扩增的数据符合新能源电池的机理特性。

表 6.1: 系统功能性需求列表

需求编号	需求名称	需求描述
R1	数据集添加	用户可以在数据仓库管理界面添加向系统添加数据集，对于已经存在于服务器上的数据集手动填写信息完成注册，也可通过上传压缩包的形式完成数据集的注册
R2	数据集搜索	用户可以在数据仓库管理界面查看已经添加到系统的数据集信息，并且可以基于信息完成数据集的搜索工作
R3	数据集删除	用户在数据仓库管理界面可以选择已经添加的数据集进行删除操作
R4	数据集状态显示	用户可以在数据仓库预处理界面查看当前仓库的状态，完成预处理的数据集显示“已预处理”，否则显示开始预处理按钮
R5	数据预处理	用户可以在数据仓库预处理界面对未预处理过的数据集发起预处理任务，对原始数据集进行特征筛选、异常值处理和空值清洗等过程
R6	设置传统算法扩增策略	用户可以在传统扩增界面选择序列扩增策略，然后基于选择的策略显示可配置参数信息
R7	传统扩增	用户可在传统扩增界面选择目标数据集并发起传统扩增任务
R8	智能扩增	用户可在智能扩增界面选择目标数据集并发起智能扩增任务
R9	模型验证	用户可在模型验证界面选择目标数据集并发起模型验证任务
R10	选择任务粒度	对于智能扩增和模型验证功能，用户可以根据需要选择按照目录进行还是按照具体的文件进行
R11	显示任务进度	当用户发起任务后，页面动态显示当前任务的进度信息，数值从 0 增长到 100，0 表示刚开始，100 表示完成
R12	模型验证指标可视化	系统在执行完模型验证任务后以图形的方式显示模型验证的指标信息
R13	模型验证曲线可视化	系统在执行完模型验证任务后以图形的方式对比显示预测曲线和真实曲线
R14	下载扩增结果	系统在执行完扩增任务后，用户可以点击下载按钮下载扩增数据集

6.2.2 非功能性需求分析

除了核心的功能性需求外，本系统在使用过程中还需要满足一些非功能性需求。非功能需求是软件产品为了满足客户业务需求而必须具有的除功能性需求外还应该具有的特性。因此非功能性需求并不针对具体的系统行为，而是从系统性能、安全性、可维护性、可扩展性、可靠性、易用性六个方面对系统进行约束。在系统能否持续高效稳定的向用户提供服务和提高用户体验等方面，非功能性需求起到至关重要的作用，是设计软件系统不可或缺的组成部分。本节将针对上述六个方面，对本系统的非功能需求进行设计与描述。

性能需求，主要描述系统任务的执行效率。在正常网络下，系统的任务发起、信息查询、以及数据上传/下载等操作的响应时间应尽可能的短。为了达到

该目标，在任务执行方面要求系统采用多线程技术提高系统响应效率；在流量分发方面要求系统可以利用负载均衡中间件技术对客户的服务请求进行分流操作；在数据处理方面要求系统使用矩阵运算避免使用循环语句从而提高数据的处理速度。安全性，主要考虑用户信息的隐蔽以及保护工作。系统应该保护用户上传的数据集，确保不同用户之间无法相互访问。可维护性，主要描述系统在部署阶段发生问题时，开发人员能够快速确定问题发生的原因，相关代码位置并及时解决。要求开发人员在项目设计阶段充分分析系统可能的运行环境，输入数据的规范，以及采用技术栈的限制等情况，然后根据分析结果找出可能存在的异常流程并在代码中注入异常日志。系统在运行阶段定期的将运行日志写入磁盘备存，当发生异常行为时开发人员可以根据系统日志快速定位问题的位置。可扩展性，要求系统设计应当模块化采用高内聚低耦合设计方式，以满足后续系统的持续扩展。本系统在设计阶段对业务功能性需求进行拆分组合，基于自顶向下的设计方式将互不相交的功能封装成独立的模块。当一个功能点被多个功能模块使用时，将功能点封装成一个核心类以组合的方式和功能模块交互。这样可以极大降低开发人员对代码熟悉的时间成本，如果后续有额外的功能性需求可以快速迭代并集成进当前系统之中。可靠性，与可维护性相关，不同的是可维护性关注发生错误时对系统环境的记录；而可靠性则要求系统在发生错误情况下继续执行的能力。要求系统设计应当采用兼容性设计，对于可能发生的异常流程设计应对策略，确保部署阶段系统执行中发生类似异常时可以继续执行下去。易用性，主要是针对前端设计提出的需求，描述如何降低客户使用系统的学习成本。要求前端界面设计要尽量简洁，对于功能类似的任务提供相似的界面布局来降低用户的熟悉成本；同时，对于需要进行参数配置的功能界面，对所有的参数提供默认值让用户可以在不需要进行复杂配置的情况下使用。

6.2.3 系统用例图

本系统的用例图如图6.2所示，描述了本扩增系统的核心用例。用户可以进行：（1）数据仓库管理，包括数据集的添加，数据集的搜索、以及数据集删除三个功能性需求；（2）数据预处理任务，包括查看数据集状态信息，检查当前数据集是否经过预处理；（3）传统扩增，包括设置传统算法扩增策略，显示任务进度，选择任务粒度，以及下载扩增结果四个功能性需求。扩增任务支持选择序列扩增策略，在任务执行过程中可以动态查看任务执行进度，对于已经完成的扩增任务进行下载操作；（4）智能扩增，包括显示任务进度，选择任务粒度，以及下载扩增结果三个功能。智能扩增任务可以选择是基于文件粒度扩增还是基于目录粒度扩增，在任务执行过程中可以动态查看任务执行的进度，对于已

经完成的扩增任务发起扩增结果下载操作；（5）模型验证，包括显示任务进度，选择任务粒度，模型验证指标可视化，模型验证曲线可视化四种功能。验证任务支持配置验证粒度是基于文件验证还是基于目录验证，并且在执行过程中同样可以动态查看任务进度，对于验证结果利用可视化技术进行展现。

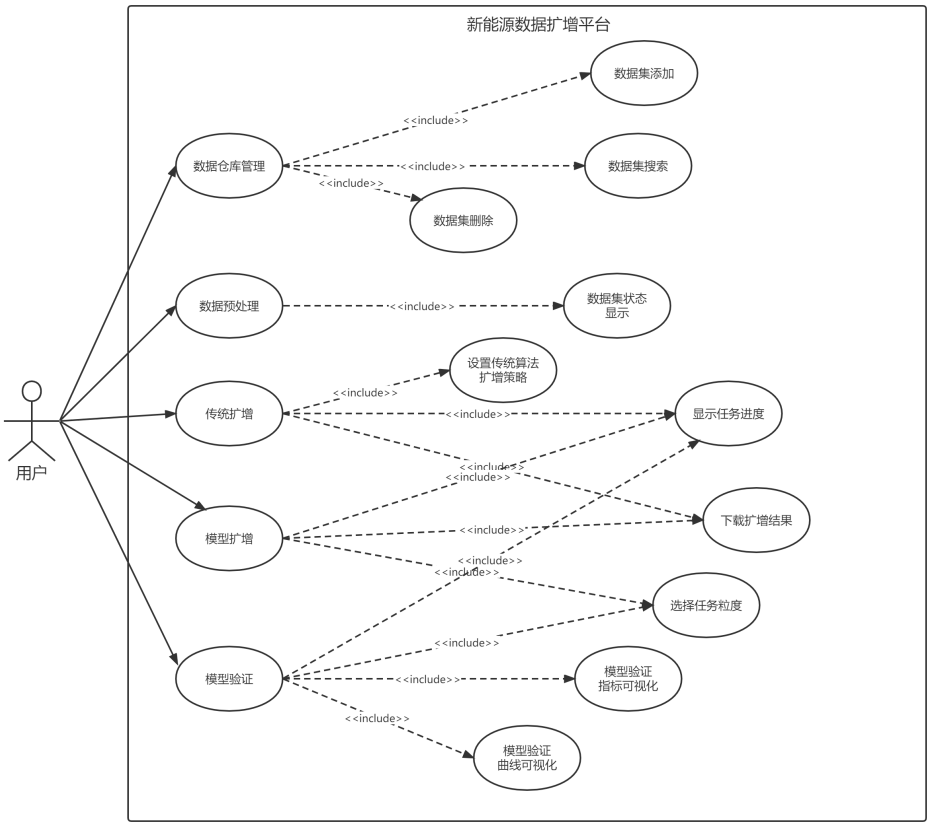

图 6.2: 系统用例图

6.2.4 系统用例描述

本节进一步详细描述系统用例图6.2中的用例，涉及的用例与功能性需求的对应关系如表6.2所示。由于论文篇幅的限制以及突出核心用例的需要，本文并没有给出 UC1 数据仓库管理和 UC2 数据预处理的用例描述。

UC3 传统扩增用例描述如表6.3所示。当用户完成注册和登录流程后，可以从左侧菜单栏点击传统扩增选项进入传统扩增界面，该界面显示了当前用户下上传或者注册所有数据集信息。用户首先选择需要进行传统扩增的数据集，在点击扩增配置按钮后，系统弹出扩增策略选择界面。用户根据需要选择一个或者多个扩增策略，系统支持时域老化扰动、频域老化扰动、对称性丢失、非对称性丢失等扩增策略。确定扩增策略后，系统根据用户选择的策略内容显示各策略需要给定的参数配置且所有参数配置均提供默认值，用户可以根据需要手动

表 6.2: 系统用例列表

用例编号	用例名称	功能需求编号
UC1	数据仓库管理	R1、R2、R3
UC2	数据预处理	R4、R5
UC3	传统扩增	R6、R7、R11、R14
UC4	智能扩增	R8、R10、R11、R14
UC5	模型验证	R9、R10、R11、R12、R13

调整，也可直接使用系统默认参数进行数据扩增。配置完成后，用户点击开始扩增按钮发起传统扩增任务，也可以点击取消按钮放弃本次过程。扩增任务开始后，系统动态获取当前扩增任务的执行进度情况，并在前端界面展示。任务执行结束后系统进入结果界面，用户可以在该界面中查看对应的结果信息以及下载本次扩增数据。

表 6.3: 传统算法扩增用例描述

ID	UC3
用例描述	基于时序扩增策略进行新能源电池数据扩增
参与者	用户
触发条件	用户在左边栏点击传统扩增选项进入该界面
前置条件	用户登录且该用户下有存在的数据集
后置条件	系统根据配置生成扩增数据供用户下载
优先级	高
正常流程	1. 用户点击传统扩增菜单选项进行传统扩增界面； 2. 选择扩增数据集并点击扩增配置按钮； 3. 系统弹出页面显示支持的扩增策略，用户根据需要勾选扩增策略然后点击下一步； 4. 系统根据用户选择的扩增方法显示可配置的扩增参数，用户在页面进行参数配置后，点击开始扩增按钮发起扩增任务； 5. 系统显示扩增进度； 6. 扩增完成后显示扩增结果； 7. 用户在结果界面点击查看 svg 按钮，系统显示扩增前后特征对比结果图片；
扩展流程	3a. 用户点击上一步重新选择扩增方法； 3b. 用户点击取消按钮放弃本次扩增； 7a. 用户点击下载最近一次扩增结果按钮获取扩增后数据集；
特殊需求	无

UC4 智能扩增用例描述如表6.4所示。用户从左侧菜单栏点击智能扩增菜单选项进入智能扩增界面，该界面显示当前用户下所有注册或上传的数据集信息。选定扩增目标数据集后，用户可以点击扩增按钮或者选择文件按钮发起扩增任

务，其中前者是基于目录进行扩增而后者是基于具体的文件进行扩增。基于文件扩增只会对选中的文件进行扩增操作，而基于目录扩增则会对选中的数据仓库中所有的文件进行扩增。如果点击的是扩增按钮，系统弹出扩增配置页面，该页面包含所有与扩增相关的配置信息且均提供了默认参数，用户可以根据需要进行修改。配置修改完成后点击开始扩增按钮发起对目录的扩增任务。如果点击的是选择文件按钮，系统会进入基于文件界面，在该页面中系统会显示当前选中的扩增目录下所有的 csv 格式的文件。用户根据需要选择具体的文件并点击扩增按钮发起基于文件的扩增任务，其中参数配置流程和基于目录扩增任务中的流程一致。扩增任务结束后用户点击 “下载最近一次扩增结果” 按钮进行扩增数据的下载。

表 6.4: 智能扩增用例描述

ID	**UC4**
用例描述	基于扩增模型进行新能源电池数据扩增
参与者	用户
触发条件	用户在左边栏点击智能扩增菜单选项进入智能扩增界面
前置条件	用户登录且该用户下有存在的数据集，并且数据集已经经过预处理
后置条件	系统根据配置生成扩增数据，可供用户下载
优先级	高
正常流程	1. 用户择扩增数据集点击扩增按钮； 2. 系统显示扩增配置界面； 3. 用户根据需要调整扩增参数，点击开始按钮发起扩增任务； 4. 系统显示扩增进度； 5. 系统显示扩增结果信息； 6. 用户点击下载最近一次扩增按钮获取扩增结果；
扩展流程	1a. 扩增数据没有经过预处理，提示用户先进行预处理； 1b. 用户点击选择文件按钮手动选择需要扩增的数据文件； 1. 系统根据选择的数据集跳转至基于文件界面，显示该数据集下所有文件列表； 2. 用户选择文件或者基于搜索框查询根据文件名查找目标文件； 3. 用户点击扩增按钮系统显示扩增参数配置界面； 4. 用户点击开始按钮启动扩增任务； 3a. 用户点击取消放弃本次扩增任务；
特殊需求	无

UC5 模型验证用例描述如表6.5所示。用户从左侧菜单栏点击模型验证菜单选项进入模型验证界面，该界面显示当前用户下所有注册或者上传的数据集信息，用户可以手动选择或者基于搜索框筛选验证数据集。选定好验证数据集后，用户可以直接点击验证按钮发起基于该验证数据集的模型验证任务；或者点击选择文件按钮，系统跳转至基于文件 tab 页面，该页面显示了当前选中的验证数

据集下所有的 csv 文件。用户可以手动或者利用筛选框选择选择具体文件，然后点击验证按钮发起基于当前选择文件的验证任务。通过点击验证按钮和选择文件按钮分别发起基于目录和基于文件的模型验证任务。二者的区别是，基于文件验证只会利用选中的文件对模型进行验证，而基于目录的验证则会利用目录下的所有 csv 文件对模型进行验证。当验证任务开始执行后，系统会动态显示验证任务的执行情况，以 0 至 100 表示，其中 0 表示刚刚发起，100 表示任务完成。验证任务结束后，系统进入结果展示界面。该界面显示了模型验证的指标信息，同时用户可以点击“查看 svg”按钮查看模型预测值和真实值的对比曲线。

表 6.5: 模型验证用例描述

ID	UC5
用例描述	对系统部署的扩增模型进行验证
参与者	用户
触发条件	用户在左边栏点击模型验证菜单选项进入模型验证界面
前置条件	1. 用户登录且该用户下有存在的并且数据集已经经过预处理； 2. 系统具有部署的扩增模型；
后置条件	系统生成扩增指标和对比曲线图片
优先级	高
正常流程	1. 用户根据需要选择验证目标数据集并点击选择文件按钮； 2. 系统跳转至基于文件界面并显示该数据集下的所有 csv 文件信息； 3. 用户选择文件或者基于搜索框查询根据文件名查找目标文件； 4. 用户点击验证按钮启动验证任务； 5. 系统显示验证进度； 6. 系统显示验证结果信息；
扩展流程	1a. 用户选择目标数据集后点击验证按钮使用当前数据集中所有文件进行模型验证； 6a. 用户点击查看图片按钮，系统显示验证结果的对比曲线图；
特殊需求	无

6.3 系统概要设计

6.3.1 系统架构设计

本系统整体架构如图6.3所示，自底向上分别为数据存储层，应用服务层和前端交互层。系统基于 BS 即 Browser/Server 架构，具有分布性强、维护方便、开发简单并且共享性强的优点，任何电脑只需要安装浏览器即可向本系统发起服务申请。MTV 即 Model-Template-View 的开发模式确保了前后端分离的设计方式，让各模块组件之间去耦合。前端交互层整体采用 Vue 和 ElementUI 框架完成静态页面布局以及页面交互设计，利用 axios 向服务器发送 GET/POST 请求，后端收到请求后创建任务线程处理任务，最后基于 JSONResponse、HTTPResponse、

以及 FileResponse 接口向前端返回结果。web 应用服务层基于 Django 框架实现并利用 Nginx 技术完成负载均衡和反向代理任务，模块包括数据仓库管理、数据预处理、传统扩增、智能扩增、以及模型验证模块，所有模块都利用 Django 框架实现，得益于其丰富的组件和 URLConf 映射机制，开发人员只需要关注对应模块下视图文件中的业务逻辑，将复杂的业务拆分成多个服务接口，确保了项目的高速迭代。数据存储层采用 MySQL 服务基于 HTTP 协议与 web 服务层通信并为其提供数据持久化服务。Django 自带的 ORM 即 Object Relational Mapping 服务会根据对接的数据库引擎将数据库操作翻译成对应的 sql 语句，让开发人员无需关心底层使用的是哪种数据库引擎，降低了项目的学习成本。为了提高系统的可移植性、隔离性、和安全性，本系统将上述三个层分别打包进三个 docker 容器中，并创建两个桥接网络用于容器之间的通信工作。两个桥接网络分别负责前端交互层和应用服务层、以及应用服务层和数据存储层之间的通信。

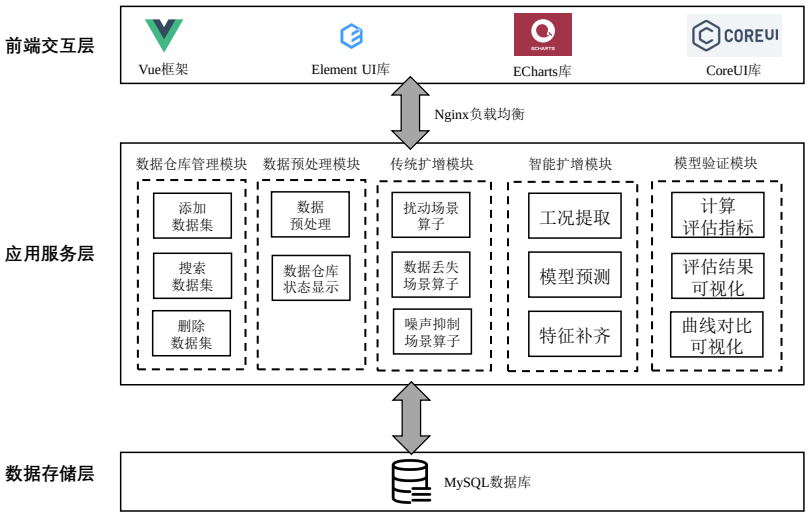

图 6.3: 系统架构设计图

前端交互层展现了用户通过浏览器可以访问的内容，通过用户管理机制，一个用户仅能访问该用户下注册或者上传的数据集文件，确保了数据信息的安全性。在前端界面用户可以进行数据集的添加、删除、搜索、以及预处理操作；同时可以发起传统算法扩增和智能扩增任务，查看扩增结果并下载扩增数据集；在模型验证界面可以对部署的模型进行验证，查看模型验证的量化指标信息以及对比曲线结果。

应用服务层根据系统需求业务分为五个功能模块，分别负责数据集的管理、数据预处理、数据扩增（传统/智能）、以及模型验证几个方面。为了提高系统响应速度，后端采用多线程技术对每个服务申请都创建任务线程去执行具体的业

务流程，确保主线程可以及时返回，提升了前端响应的速度。为了让前端可以监督后端任务线程的执行情况，后端提供了接口让前端可以获取任务线程的执行进度信息。

数据存储层主要负责完成系统的持久化设计，主要负责保存系统用户信息、执行过程中用户注册或者上传的数据集信息、扩增任务以及模型验证任务的结果等信息。

6.3.2 系统 4+1 视图

系统 4+1 视图分别为：(1) 逻辑视图、开发视图、进程视图、物理视图；(2) 以及场景视图。其中场景视图（也称为用例视图）是所有视图的核心，即 4+1 视图是围绕场景视图对整个系统的架构进行描述，其内容与系统用例图，即图6.2一致。

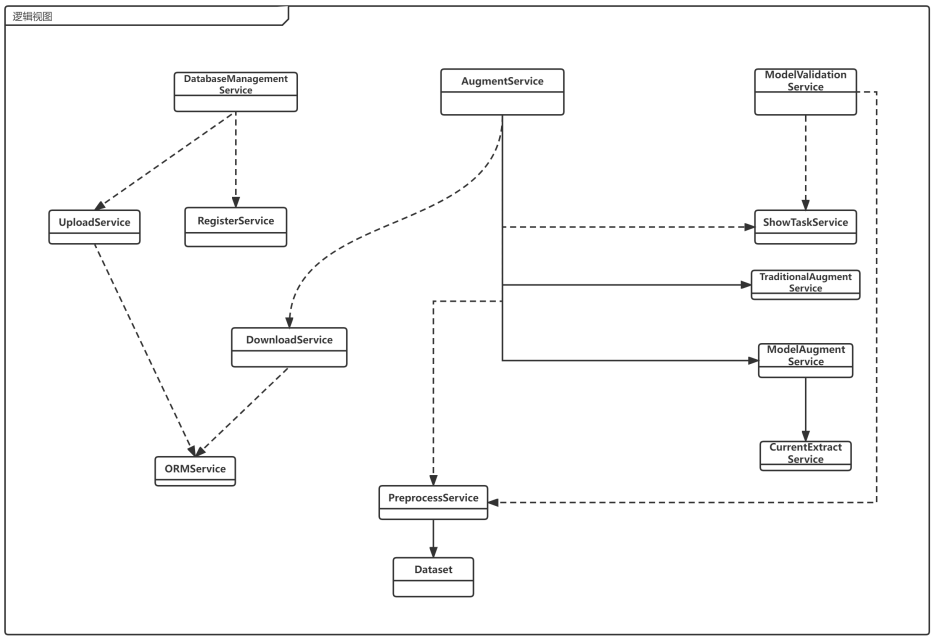

图 6.4: 系统逻辑视图

逻辑视图侧重于功能性需求，从系统最终用户的角度出发描述系统对用户提供的功能，其结构如图6.4所示。其中扩增服务 AugmentService 是核心功能，包括传统扩增服务 TraditionalAugmentService 和智能扩增服务 ModelAugmentService。而智能扩增服务 ModelAugmentService 又包括工况提取服务 CurrentExtractService，该服务负责从新能源数据集中提取目标工况序列供 ModelAugmentService 使用。模型验证服务 ModelValidationService 负责对部署的模型进行验证，从

而对系统部署的智能扩增模型扩增数据的准确性提供量化描述。扩增服务 AugmentService 和模型验证服务都包含 ShowTaskService 服务，该服务负责显示任务的进度情况。数据的上传和下载服务由 Django 的 ORMService 部件提供支持。DatabaseManagementService 服务负责对服务器端的数据集进行管理，包括数据集注册、上传、删除、显示以及查询工作。PreprocessService 是数据预处理服务，主要负责基于原始数据集进行关键特征提取、异常值以及空值处理、和数据集状态显示等工作。Dataset 是一个数据集的抽象，包含一些对数据集的基本操作，比如文件读写、数据集下 csv 文件获取等。

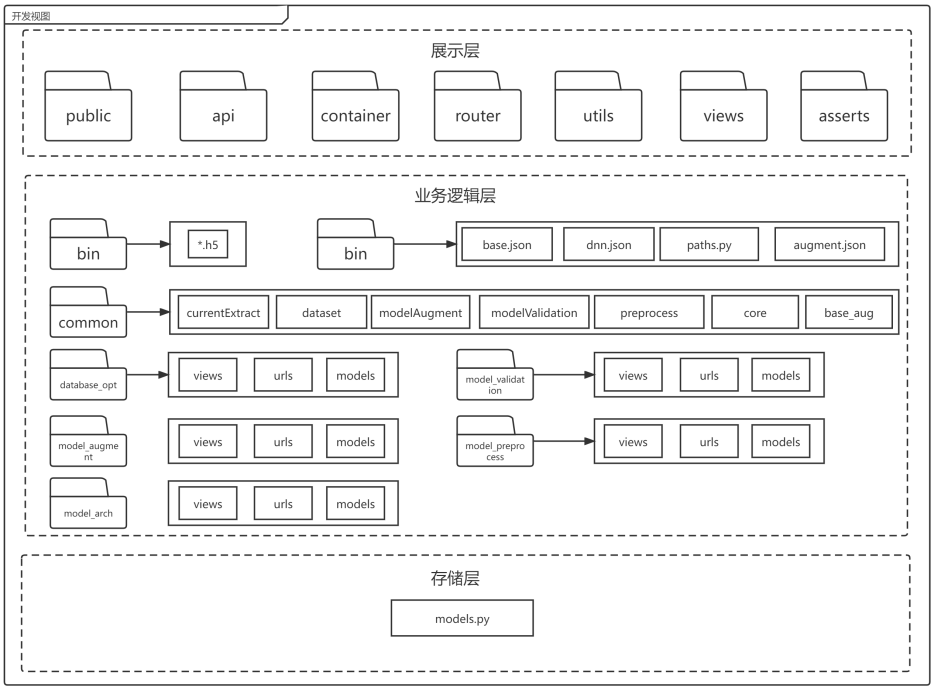

图 6.5: 系统开发视图

开发视图如图6.5所示，其侧重于系统的代码架构并从开发者的角度对系统进行描述，包括前端展示层、业务逻辑层、以及存储层三部分。本系统基于 Django 采用前后端分离设计，前端文件统一放在 appfront 目录下，主要包含 public、api、containers、router、utils、views、asserts 七个目录。public 文件夹中一般放置公有资源，比如系统图标。在 api 目录下设置前后端通讯的接口路径（URL）以及通讯协议（POST、GET）。containers 文件中保存了很多.vue 格式的文件，这些文件定义了页面的整体框架布局，每一个.vue 文件都代表了一个独立的页面。router 文件夹包含 index.js 文件，用于配置路由规则，源文件部件统一放在 views 目录下，utils 文件中提供了一些前端使用的方法，asserts 目录用于存储静态的资源，

比如图标信息。

业务逻辑层包含系统后端的实现代码，主要目录包括：（1）bin，用于存放部署的智能扩增模型，一般以.h5 的格式存在；（2）common，用于放置业务的核心方法和类，包括数据预处理模块代码，智能扩增代码、模型验证代码、仓库管理代码等；（3）configure，存放系统的配置模板；（4）static，用于存放静态文件，比如用于前端展示的模型验证的结果；（5）model_arch，存放扩增模型的结构，由于.h5 文件格式并不会存储模型的结构信息，因此在模型参数加载之前需要预先定义模型的结构，否侧无法使用模型参数；其他还包括由 Django 创建的应用目录（6）database_opt、model_augment、model_preprocess、model_validation 等，里面包含服务的业务逻辑、路由信息表、以及应用的数据库表结构信息。

存储层负责管理系统的中间信息以及结果，利用 MySQL 数据引擎进行数据存储。存储层基于 Django 的 ORM 架构，在各 app 文件夹下创建 models.py 文件定义 MySQL 表结构。

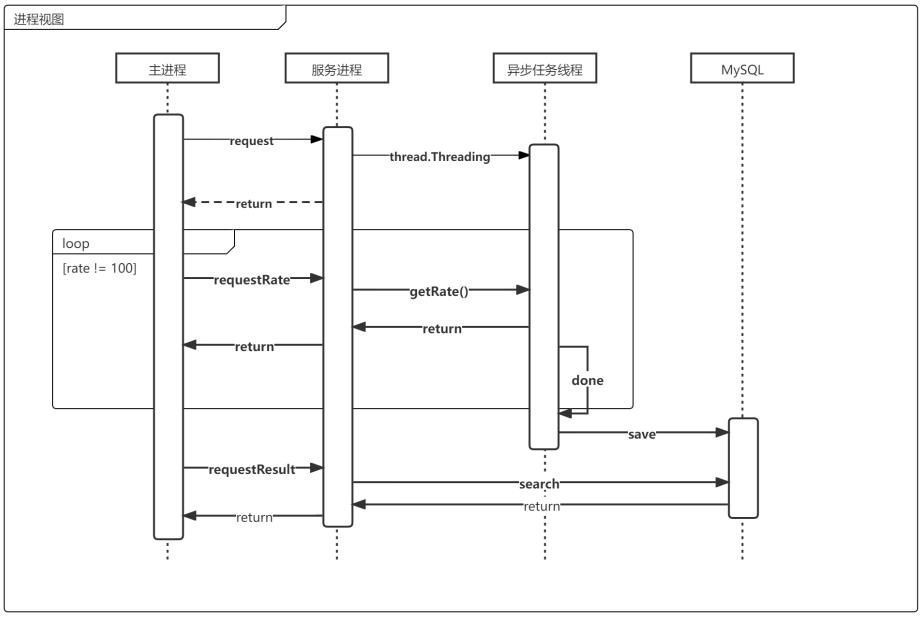

图 6.6: 系统进程视图

系统进程视图如图6.6所示，该视图从模块交互的角度对系统进行描述，主要包括系统运行过程中进程的并发、同步、以及任务调度等内容。当用户发起一个服务申请时，前端界面根据约定好的 api 接口将服务请求发送给后端对应的应用。后端接收到请求后，获取相应的参数信息并创建任务对象 task，然后通过 threading.Thread 类创建任务线程执行对应的服务代码。在任务线程执行过程中，

前端可以访问对应接口获取任务的进度，进度为 0 表示任务刚刚启动，进度为 100 表示任务完成。任务完成后，任务线程会将执行结果写入 MySQL 数据表中待前端访问。

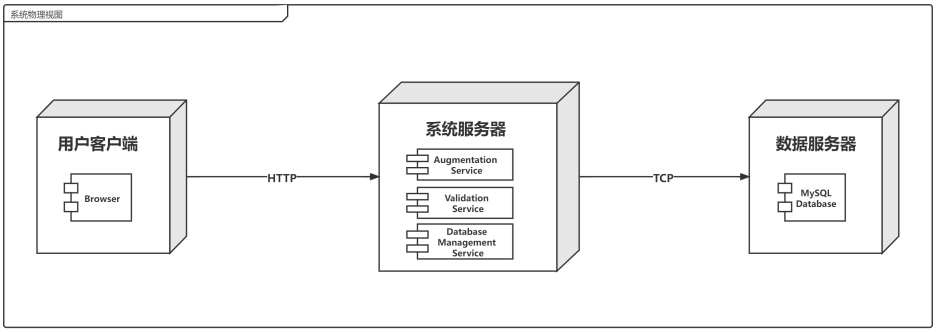

图 6.7: 系统物理视图

系统物理视图如图6.7所示，该视图主要从系统的部署、资源配置角度对系统进行描述。其中包括系统的计算资源分配、网络拓扑等信息。本系统利用 docker 技术将前端、后端、以及 Mysql 服务器分别打包进三个不同的容器并建立两个桥接网络用于容器之间的通信工作。用户首先在前端页面通过 http 向服务器发起服务请求 request，请求 request 经由 Nginx 发送至后端。后端根据请求的路由以及 api 接口情况创建任务对象并建立任务线程。任务线程调用任务对象的 run 接口执行具体业务代码，执行结束后将结果送入 MySQL 数据库表中。当后端系统收到前端接收结果的请求后，后端将最终结果打包进一个 json 结构并利用 Django 的 JsonResponse 接口返回最终结果。

6.4 数据库设计

面向新能源电池扩增系统采用 MySQL 数据引擎存储系统运行过程中的数据和任务执行结果。在对数据表进行设计之前需要定义系统中的实体对象，然后根据业务流程分析确定其中不同对象之间的交互关系以及联系。本节利用 E-R 图模型详细描述了系统中的客观实体对象、实体对象的属性，以及各实体之间的联系。

6.4.1 E-R 图

E-R 图模型是实体-关系图（Entity Relationship Diagram）的缩写，是用于描述实体、属性、以及联系之间关系的方法。实体、属性、联系是 E-R 图的三要素，其中实体包括现实世界中客观存在的物体，比如电池，也可以是抽象的概念，比如数据集合；属性是一个实体应该具有的客观特性，一个实体可以有多个属性；

联系用于描述不同实体之间的关系，可以是一对一、一对多、以及多对多联系。E-R 图将实体映射成数据模型，辅助开发人员分析和设计具体的数据库表单。

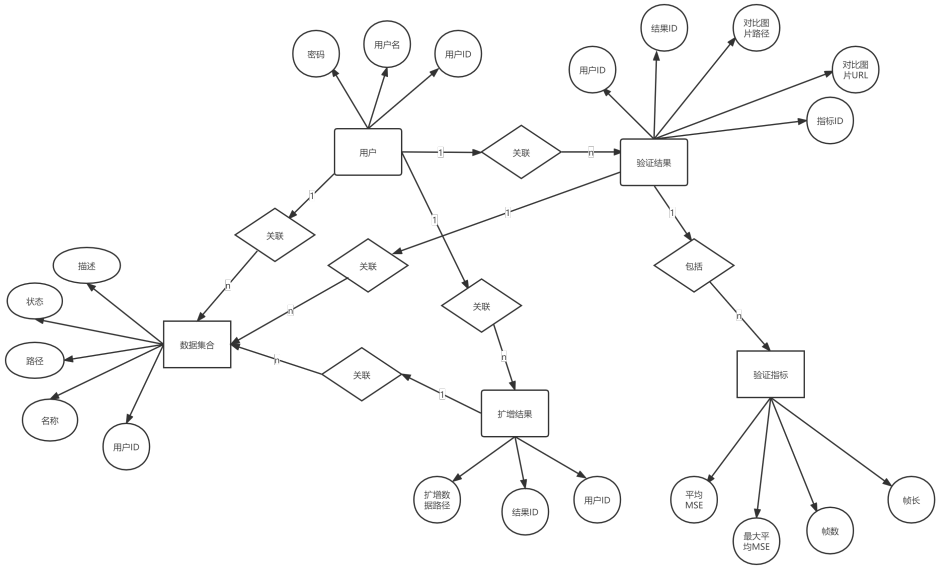

图 6.8: 系统 E-R 图

本系统的 E-R 图如图6.8所示，数据表设计围绕数据库管理、用户管理、扩增结果、以及模型验证结果进行。E-R 图中与数据库管理相关的实体包括用户和数据集，其中数据集是一个数据仓库的抽象，一个用户可以同时注册/上传多个数据集，用户表和数据集表二者是一对多的关系。同用户管理相关的实体对象是用户，可以通过超级管理员授予也可以在注册界面自行注册，无论哪种方式提交的用户信息都会存入用户表中。扩增结果相关的实体包括用户和数据集合两个表，其中扩增结果和数据结合、用户和扩增结果都是 1:n 的关系。与模型验证结果相关的实体是用户、数据集合、以及验证指标三个表，其中用户和验证结果、验证结果和数据集合、验证结果和验证指标表之间都是 1:n 的关系。

6.4.2 数据库表设计

本节基于 E-R 模型和系统功能需求，结合6.8图中的实体对象在 MySQL 数据库中设计各模块表单。涉及的表单有扩增结果信息表、验证结果信息表、验证指标信息表、数据集合信息表、以及用户信息表:

（1）扩增结果信息表

扩增结果信息表由结果标识 id 字段、拥有此结果的用户 userid 字段、以及扩增文件路径 path 字段三部分组成。结果标识 id 是该表中记录的唯一标识，每条记录都对应一条标识 id。userid 字段表示本结果记录属于哪个用户，用户只能

访问属于自己的结果信息记录。扩增文件路径 path 字段记录了扩增任务执行完毕后，扩增数据集在服务器中的地址信息。扩增结果信息表结构如表6.6所示。

表 6.6: 扩增结果信息表

字段	类型	主键	允许为空	说明
id	int	是	否	结果标识
userid	int	否	否	拥有此结果的用户
path	varchar	否	是	扩增文件路径

（2）验证结果信息表

验证结果信息表负责存储模型验证过程中的结果信息，由唯一标识 id 字段，拥有当前验证结果的用户标识 userid 字段，与验证指标信息表记录对应的 metric_id 字段，标识预测对比曲线图片地址信息的 path 字段，以及前端可直接访问图片的 url 信息字段构成。验证结果信息表详细字段内容如表6.7所示。

表 6.7: 验证结果信息表

字段	类型	主键	允许为空	说明
id	int	是	否	结果标识
userid	int	否	否	拥有此结果的用户
metric_id	int	否	是	指标标识
path	varchar	否	是	结果图片路径
url	varchar	否	是	结果图片路由

（3）验证指标信息表

验证指标信息表用于存储模型验证结果中的指标信息，包括唯一标识 id 字段，全局平均均方误差 mse 字段，均方误差最大值 mse 字段，验证帧长 frame_len 字段，以及当前验证数据集总帧数 frames 字段。验证指标信息表具体字段结构如表6.8所示。

表 6.8: 验证指标信息表

字段	类型	主键	允许为空	说明
id	int	是	否	指标信息标识
mse	double	否	是	全局平均 mse 值
max_mse	double	否	是	全局 mse 值的最大值
frames	int	否	是	验证总帧数
frame_len	double	否	是	每帧帧长

（4）数据集合信息表

数据集合信息表存储了新能源数据集合相关的结构化信息，用于系统对数据集合的管理工作，由六个字段组成。数据集合标识 id 字段是表中记录的唯一标，用户 userid 字段表征拥有当前数据集合的用户，数据集合路径 path 字段记录了当前数据集合在服务器上的地址信息，数据集合是否经过预处理 state 字段表示当前数据集合是否经过预处理。数据集合名称 name 和数据集合的描述 description 字段分别记录当前数据库的别名以及描述信息，方便用户区分和记忆。数据集合信息表的结构化信息如表6.9所示。

表 6.9: 数据集合信息表

字段	类型	主键	允许为空	说明
id	int	是	否	数据集合标识
userid	int	否	否	拥有此数据集合的用户
name	varchar	否	是	数据集合名称
description	varchar	否	是	数据集合的描述
path	varchar	否	否	数据集合路径
state	boolen	否	否	数据集合是否经过预处理

（5）用户信息表

用户表单如表所示6.10，该表用于存储所有与用户相关的结构化信息，包括用户名 username，用户密码 password，以及用户唯一标识 userid 字段。

表 6.10: 用户信息表

字段	类型	主键	允许为空	说明
userid	int	是	否	用户标识
username	varchar	否	否	用户名
password	varchar	否	否	用户密码

6.5 系统详细设计

本节围绕系统的主要功能模块详细介绍了数据集管理模、数据预处理模块、传统扩增模块、智能扩增模块、以及模型验证模块的实现方式。由于算法设计已经在前面章节中描述，故后续章节中将不再重复相关内容，而是将描述重点放在系统的流程设计以及模块类实现上。

6.5.1 数据集管理模块设计与实现

数据集管理模块主要负责管理用户的数据集信息，包括数据集添加、删除、以及查询等操作。添加数据集支持用户手动登记和 zip 包上传两种方式，用户需要提供对应数据集的名称和描述信息，如果是手动登记的方式还需要提供数据集在系统上的路径信息，数据集添加成功后，系统会以列表的形式展现当前用户下的所有数据集信息。在数据集列表界面用户不仅可以进行数据集的删除操作，还可以基于搜索框筛选数据集查看其对应状态。

数据集管理模块流程图如图6.9所示，当用户进入数据仓库管理界面时，前端通过 GET 方法向后端发送获取当前用户数据集申请。后端收到路由命令后，解析用户 userid 并通过 userid 从 MySQL 数据库中查询该用户所有已经注册的数据集信息。接着，将获取的数据集信息打包成 json 格式，通过 JsonResponse 接口发送回前端显示。数据集登记有两种方式构成，一种是数据集已经通过 shell scp 命令传送至服务器，另一种是通过用户打包上传。前者的路径由用户在上传时定义，而后者则由系统决定。前端收集数据集添加所需的所有信息后，利用 GET 方法将打包信息发送至服务端，服务端解析信息并根据添加方式执行不同的处理分支。如果是手动登记方式，如果数据集路径存在则直接在 MySQL 数据库中添加相应的记录；如果是 zip 打包上传，则首先在系统静态文件路径/static/static 下创建对应的目录，然后将用户提交的 zip 文件解压缩到该目录，最后更新 MySQL 数据库表信息。数据集的删除过程比较简单，前端利用 POST 方法发送需要删除的数据集信息，后端接收到后首先删除 MySQL 数据库中的数据集记录，然后再删除服务器上的数据集文件。

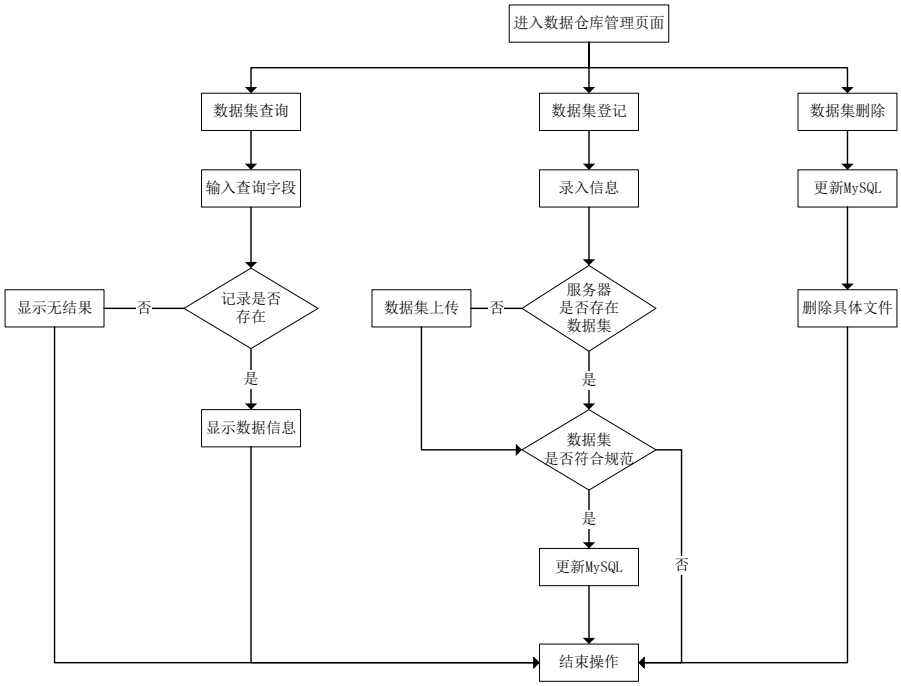

图 6.9: 数据库管理模块流程图

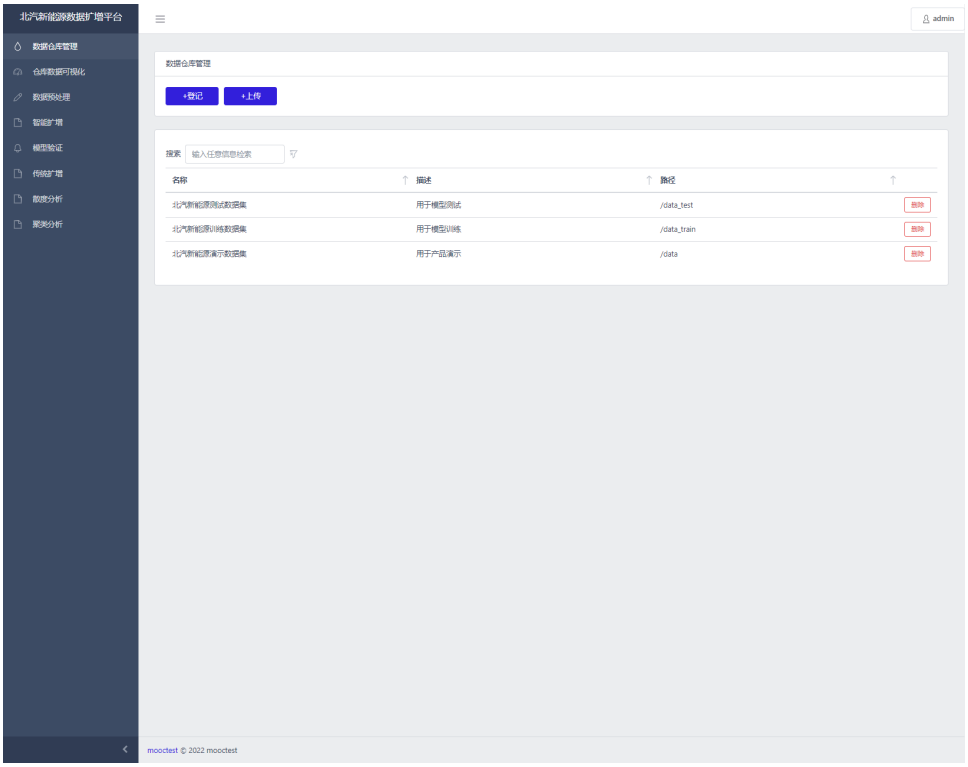

图 6.10: 数据库管理模块界面

数据库管理模块界面如图6.10所示，用户可以从该界面对数据仓库中的数据集进行登记、上传、以及删除操作，也可以查看已经注册或者上传的数据集信息。

6.5.2 数据预处理模块设计与实现

数据预处理模块主要负责对电池管理系统采集的原始数据进行筛选，过滤其中无关特征并降低特征的维度。然后，对过滤后的特征数据进行清洗，去除其中的异常值和空值样本，从而提高数据集的质量。经过预处理后的数据集才能被用于后续扩增任务和模型验证工作。

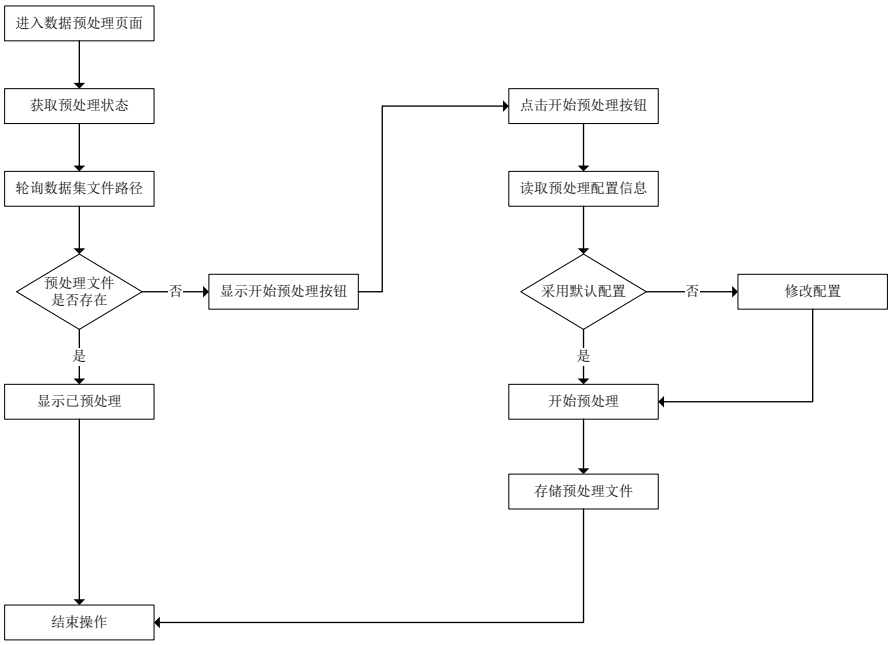

图 6.11: 数据预处理模块流程图

数据预处理模块流程图如图6.11所示，用户进入数据预处理界面时，前端会向后端发送 GET 服务请求获取当前用户下可使用的数据集信息并用列表显示，数据集列表同时显示当前数据集的预处理状态。后端获取到前端查询 request 申请后，首先从 MySQL 数据库表中查询该用户 userid 下所有的数据集记录，然后基于每个数据集记录中的路径字段扫描该路径下是否有预处理以后的文件信息，最后后端将该用户的数据集列表以及对应的预处理状态信息打包成 JSON 格式，通过 JsonResponse 接口将结果送回前端。前端收到结果后，检查状态信息，如果该数据集进过预处理则显示“已预处理”否则显示开始预处理按钮。用户点击开始预处理按钮后，前端将用户配置好的信息基于 POST 方法发送至后端发起预处理申请。后端接收到请求 request 后，创建工作线程执行预处理任务，预

处理后的数据集直接放置在源数据集路径下创建的 processed 目录中。预处理核心代码如图6.12所示，分别是空值删除 removeNaN、双门限计算 caloutlierLimits、以及异常值清洗 cleanOutliers。

```
def removeNaN(data):
    return len(np.isnan(data).any(axis=1)), data[~np.isnan(data).any(axis=1), :]
def calOutlierLimits(data):
    percentile = np.percentile(data, (25, 50, 75), axis=0, interpolation="midpoint")
    Q1 = percentile[0]
    Q3 = percentile[2]
    IQR = Q3 - Q1
    ulim = Q3 + 1.5 * IQR
    llim = Q1 - 1.5 * IQR
    limit = list(zip(np.round(llim, 2), np.round(ulim, 2)))
    return limit
def cleanOutliers(data, lims_dict):
    for k, v in lims_dict.items():
        temp = data[:, k]
        ulim = v[1]
        llim = v[0]
        outlier_idx_h = np.argwhere(temp > ulim).reshape(-1)
        outlier_idx_l = np.argwhere(temp < llim).reshape(-1)
        outlier_idx = np.hstack((outlier_idx_l, outlier_idx_h))
        data = np.delete(data, outlier_idx, axis=0)
    return data
```

图 6.12: 预处理核心代码

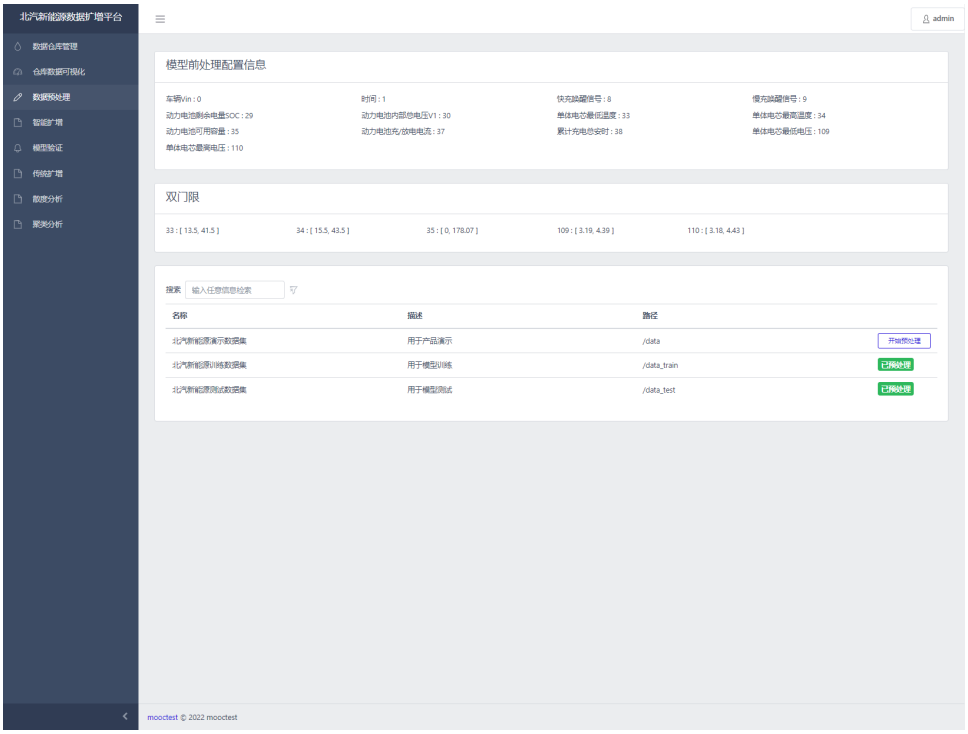

图 6.13: 数据预处理模块界面

数据预处理模块界面如图6.13所示，界面上半部分显示了特征筛选的结果以及特征字段在原始数据集中的列号；界面中部是经过特征分析后计算出来的双门限范围；界面最下面是当前用户注册的数据集列表信息。在数据集列表中，如果当前数据集经过预处理则显示“已预处理”信息；否则显示“开始预处理”按钮，用户可以点击此按钮对当前的数据集进行预处理工作。

6.5.3 传统扩增模块设计与实现

传统扩增模块主要负责在数据空间对新能源电池数据进行扩增工作，基于新能源电池可能的工作场景设计对应的扩增策略，本系统支持面向扰动、数据丢失、以及噪声抑制三种场景进行数据扩增，扩增算子包括时域噪声扰动、频域噪声扰动、对称性数据丢失、非对称性数据丢失、窗口平滑、以及指数平滑等。

传统扩增模块流程图如图6.14所示，用户进入传统扩增页面，前端利用 GET 方法发送查询申请，后端收到服务申请 request 后，根据服务申请的用户 userid 从 MySQL 数据库中获取当前用户的所有数据集信息，最后打包成 JsonResponse 发送回前端。用户根据前端显示的数据集列表信息选择需要扩增的数据集，并提供多项扩增策略供用户选择。用户选择具体的扩增策略后，前端显示相关配置信息，用户根据需要进行调整最后发起扩增任务。后端基于 POST 方法获取

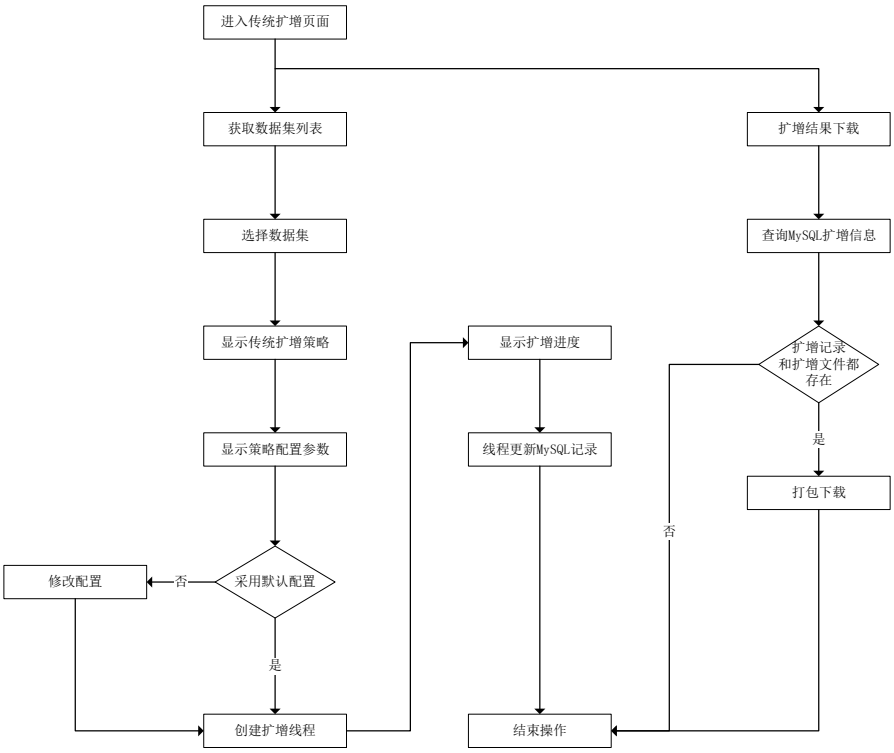

图 6.14: 传统扩增模块流程图

所有信息后打包成 json 格式，创建任务线程并将配置 json 信息发送给任务线程，任务线程根据配置信息执行传统扩增业务代码，任务线程完成后将任务结果写入 MySQL 数据库中。前端申请传统扩增服务后，定期查询任务线程的执行情况，确认任务线程执行完毕后利用 GET 方法发送获取扩增结果信息。

传统扩增模块类图如图6.15所示，其中 AugBase 类是扩增任务的基类，实现了常用的文件读写、数据集遍历等方法，并且保存了用户的扩增配置 JSON 信息。三种场景扩增策略分别对应 Jt、MovAveraging 以及 Warping 类。

基于扰动扩增的核心代码如图6.16所示，其中 time_aug 是时域扰动扩增算子，freq_aug 是频域扰动扩增算子。

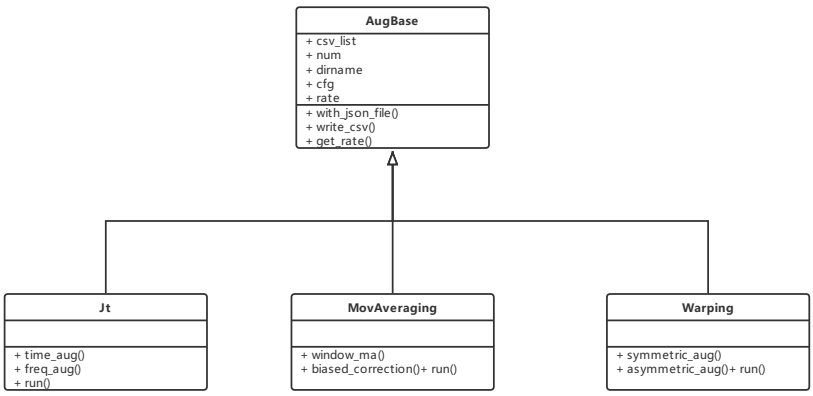

图 6.15: 传统扩增模块类图

```
noise = np.random.normal(mean, std, size=(data.shape[0], 1))
def time_aug(self, data):
    data.iloc[:, tgt_col] = data.iloc[:, tgt_col] + noise
def freq_aug(self, data):
    fft_d = fft(np.array(d))
    fft_d[0:N//2] = fft_d[0:N//2] + noise
    return np.round(np.abs(ifft(fft_d)), 2)
```

图 6.16: 扰动策略扩增核心代码

基于数据丢失扩增的核心代码如图6.17所示，其中 symmetric_aug 是对称性扩增算子，而 asymmetric_aug 是非对性扩增算子。

```
def symmetric_aug(self, data):
    drops = len(data) * propotion // 100  # propotion is a parameter got from web
    indices = [x for x in range(step - 1, N, max(N // drops, 1))]
    return np.delete(data, indices, axis=0)
def asymmetric_aug(self, data):
    delta = np.abs(nxt - pre)
    base = np.arange(0, max_len, step)
    indices = np.argmax(delta, axis=1) + 1 + base
    return np.delete(data, indices, axis=0)
```

图 6.17: 数据丢失策略扩增核心代码

基于噪声抑制扩增的核心代码如图6.18所示，其中 window_ma 是窗口平滑策略，biased_correction 是指数平滑策略。

```
def window_ma(data, win_size):
    return np.convolve(data, np.ones(win_size), 'valid') / win_size

@jit(nopython=True)
def biased_correction(data, decay):
    for i in range(len(data)):
        temp = temp * decay + (1 - decay) * data[i]
        data[i] = temp / (1 - np.power(decay, i+1))
```

图 6.18: 噪声抑制策略扩增核心代码

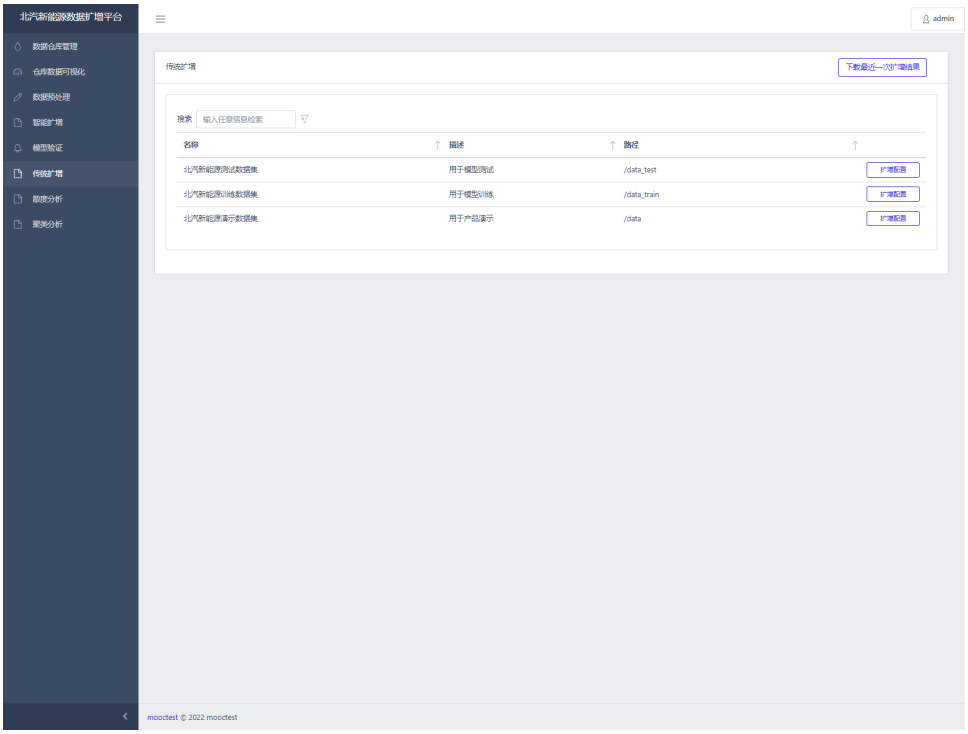

图 6.19: 传统扩增模块界面

传统扩增模块界面如图6.19所示，用户可以从基于目录扩增界面查看所有的数据集信息，发起传统扩增任务。

6.5.4 智能扩增模块设计与实现

智能扩增模块主要负责从特征空间对新能源电池的数据进行扩增。本方法利用 Seq2Seq 结构构建智能扩增模型，然后在给定目标工况和初始条件下生成电池的电压响应序列，再利用变换算子对扩增数据进行标注，最终完成面向新能源电池的智能扩增任务。

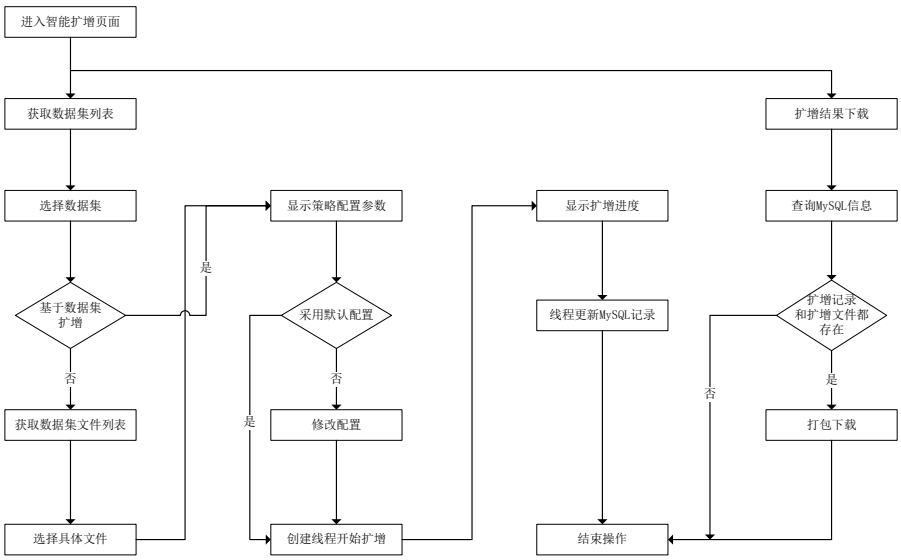

图 6.20: 智能扩增模块流程图

智能扩增模块流程图如图6.20所示，用户进入智能扩增页面，前端通过 GET 方法发送查询请求，后端收到 request 后基于用户 userid 从 MySQL 数据库中查询该用户拥有的所有数据集信息。智能扩增提供两种粒度的扩增策略，分别为基于目录扩增和基于文件扩增。用户选择目标数据集后，点击扩增按钮默认会采用基于目录扩增策略。前端显示用户需要配置的信息（所有参数均提供值），用户可以根据需求动态调整。如果用户点击选择文件按钮，系统会自动跳转到基于文件 tab 下，前端通过 GET 发送获取当前选择目录下所有的 csv 文件服务请求，后端收到 request 后，解析目录路径并从该路径下读取所有 csv 文件打包成 json 格式，最后利用 JsonResponse 接口做出响应。用户从收到的文件列表中选择一个具体 csv 文件作为扩增样本，点击扩增按钮发起扩增任务。后端收到执行扩增任务 request 后，解析用户配置信息并创建扩增对象。接着系统创建任务线程并将配置信息和扩增对象传递给任务线程。任务线程启动后，基于配置信息准备扩增环境，最后调用扩增对象的 run 方法执行数据扩增代码。任务线程执行完毕后，会将扩增数据的信息写入 MySQL 数据库。前端在等待过程中会启动

一个定时器，周期性的检查任务线程的状态和进度信息。

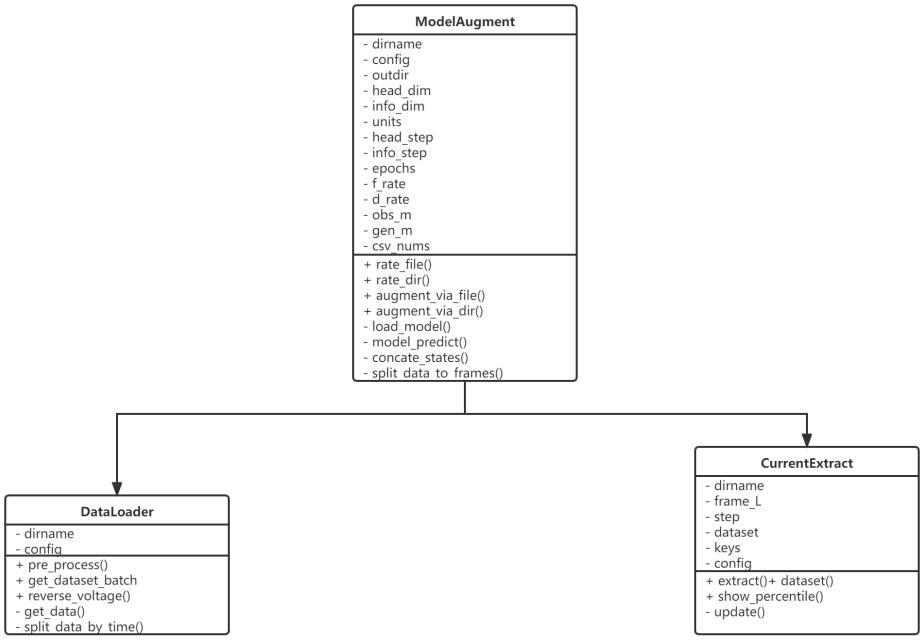

图 6.21: 智能扩增模块类图

智能扩增模块类图如图6.21所示，其中 ModelAugment 是核心功能类，提供了 augment_via_file 和 augment_via_dir 方法，分别对应基于文件扩增和基于目录扩增。ModelAugment 类在实例化过程中需要创建 DataLoader 和 CurrentExtract 两个功能类，分别实现数据加载功能和工况提取功能。DataLoader 对象负责基于预处理数据集构建能够用于模型训练和模型迭代预测的 batch 集合、数据归一化以及反归一化等任务。CurrentExtract 对象则主要负责工况提取的任务，工况提取核心代码如图6.22所示。

```
def extract(self, data, pick_num, coeff = 0.795):
    cap = data[:, :, 0]
    chg_state = np.logical_or(data[:, :, 4], data[:, :, 3]).astype(np.int8)
    delta = np.absolute(cap[:, 1:] - cap[:, :-1])
    w_state = np.where(chg_state[:, :-1] != 0, coeff, 1 - coeff)
    sum_delta = np.sum(delta * w_state, axis=1)
    pick_indices = np.argsort(sum_delta)[-min(len(sum_delta), pick_num):]
    self.update(data[pick_indices], sum_delta[pick_indices], pick_num)
```

图 6.22: 智能扩增提取工况核心代码

模型预测的核心代码如图6.24所示，model_predict 中每个时间步输入都会产生一个预测电压输出，系统用此预测值作为下一个时刻输入特征的真实电压再送入模型，如此往复实现了迭代预测。

```
def model_predict(self, data):
    for frame in range(data.shape[0]):
        init_data = data[frame, 0:self.head_step, :].copy()
        current = data[frame, self.head_step:self.head_step + self.info_step, -2]
        obs_out, init_states = self.obs_m.predict(init_data[:, :, :-1])
        gen_in = np.array(init_data[0, self.head_step - 1, :]).reshape(1, 1,
            self.info_dim).copy()
        for i in range(self.info_step):
            gen_out, init_states = self.gen_m.predict([gen_in] + init_states)
            V = gen_out[0, 0, 0]
            gen_in[0, 0, self.post_cols["V"]] = V
            gen_in[0, 0, self.post_cols["C"]] = current[i]
```

图 6.23: 模型预测核心代码

样本生成的核心代码如图6.24所示，生成样本主要经历两个处理过程，即模型预测 model_predict 和特征补齐 concate_states。concate_states 函数给出在模型输出预测电压后补齐其余特征字段的处理过程。

```
def concate_states(self, head, aug_states):
    last_step_info = head[:, -1, :]
    temp = np.repeat(last_step_info, L, axis=0).reshape(-1, L, head.shape[-1])
    aug_head = np.concatenate((head, temp), axis=1)
    for state in aug_states:
        aug_data[:, -L:, self.cols["SOC"]] = np.round(100 * (aug_data[:, -L:,
            self.cols["capacity"]] + \
            state[:, self.ext_cols["capacity"]]) / cap_max, 2)
        aug_data[:, -L:, self.cols["current"]] = state[:, self.ext_cols["current"]]
        out = np.concatenate((out, aug_data), axis = 0).astype(np.float32)
    return out
```

图 6.24: 样本生成核心代码

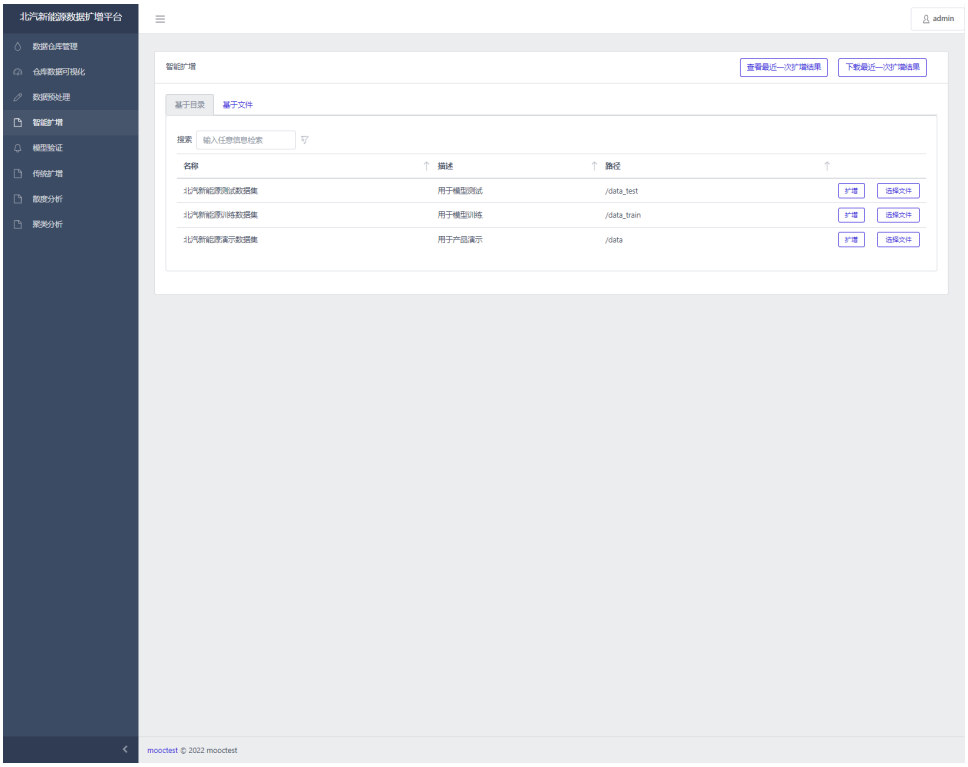

图 6.25: 智能扩增模块界面

智能扩增模块界面如图6.25所示，用户可以从基于目录扩增界面查看所有的数据集信息，并且选择基于目录扩增和基于文件扩增两种方式。

6.5.5 模型验证模块设计与实现

模型验证模块主要负责对部署的模型进行可靠性验证，确保其生成的数据符合新能源电池内在特性。本模块基于用户选择的模型验证的数据集，提供基于文件验证和基于目录验证两种方式实现模型验证任务。当验证目标确定后，系统会读入用户选择的文件并对比模型迭代预测结果和真实结果的误差，计算量化指标供前端展示。

模型验证模块流程如图6.26所示，用户进入模型验证界面，前端发送 GET 请求获取当前用户下注册的数据集列表。和智能扩增一样，模型验证同样分为两个粒度，即基于目录验证和基于文件验证两种方式。基于文件仅对选中的文件对模型进行验证、基于目录则使用目录下所有 csv 文件进行验证。基于文件点击选择文件按钮，基于目录则点击验证按钮。模型验证不需要参数配置，用户选择粒度策略后，前端就向后端发送 POST 服务请求发起模型验证任务。后端收到验证 request 请求后，根据选中的目录或者文件，创建任务线程和验证类对象，并将二者信息传递给任务线程。任务线程基于粒度策略调用验证对象的

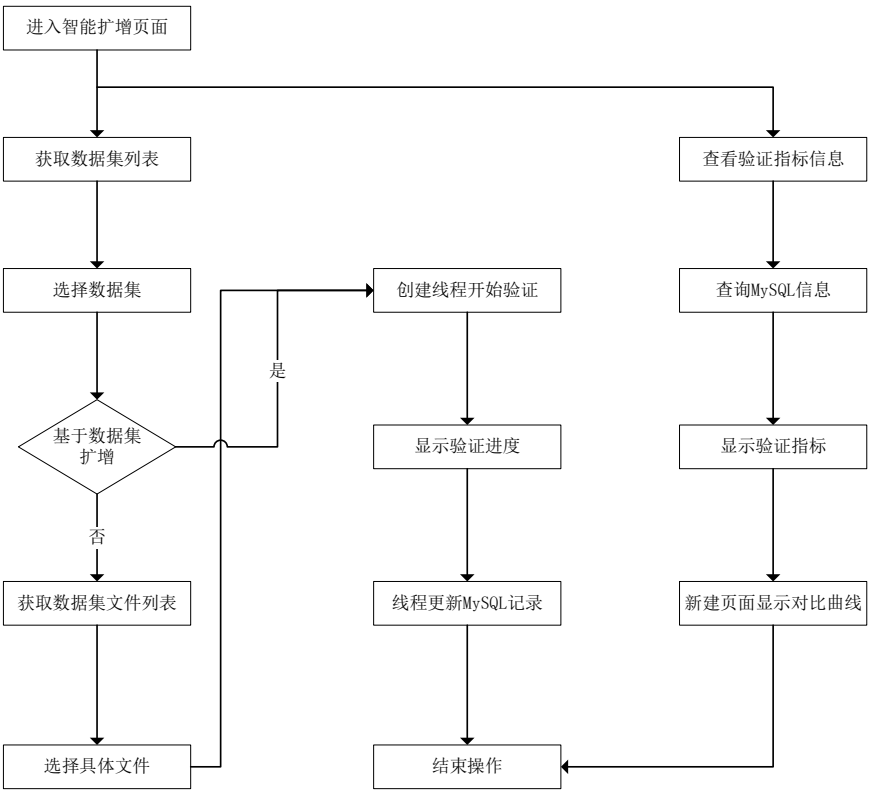

图 6.26: 模型验证模块流程图

validation_file 或者 validation_dir 方法执行模型验证代码。任务线程执行完毕后将对比曲线保存在静态资源目录/系统路径/static/media 路径下，并将路径信息以及对应的 url 字段写入 MySQL 数据库中，最后设置任务进度 100%。前端发起验证任务后，周期性检查任务线程的执行情况，当任务线程进度达到 100% 后，前端发送获取结果 POST 请求，后端从 MySQL 数据库中根据 userid 进行结果信息查询，并将结果打包成 json 格式基于 JsonResponse 接口发送给前端。

模型验证模块的类图和智能扩增的类图结构相似，二者都使用了 DataLoader 类，但是模型验证模块没有利用 CurrentExtract 类，因为验证模块中的工况是输入 csv 文件给定的，并不需要系统随机提取。验证流程核心代码如图6.27所示，validation_file 给出了基于文件验证的处理流程。

```
def validation_file(self, fname, frame_n):
    # nomalization data from fname
    norm_data, raw_vol = dl.pre_process(data)
    max_len = frame_n * self.info_step + self.head_step
    # predict
    voltages = self.model_predict(norm_data)
    reverse_voltage = dl.reverse_voltage(voltages)
    # calculate metrics
    metric = self.cal_metrics(voltages, reverse_voltage)
```

图 6.27: 模型验证核心代码

模型验证模块界面如图6.28所示，该界面以列表形式显示用户已经注册的数据集信息。每个数据集条目的末尾显示两个按钮，即“验证”和“选择文件”按钮，分别用于发起基于目录验证和基于文件验证两种粒度的模型验证任务。

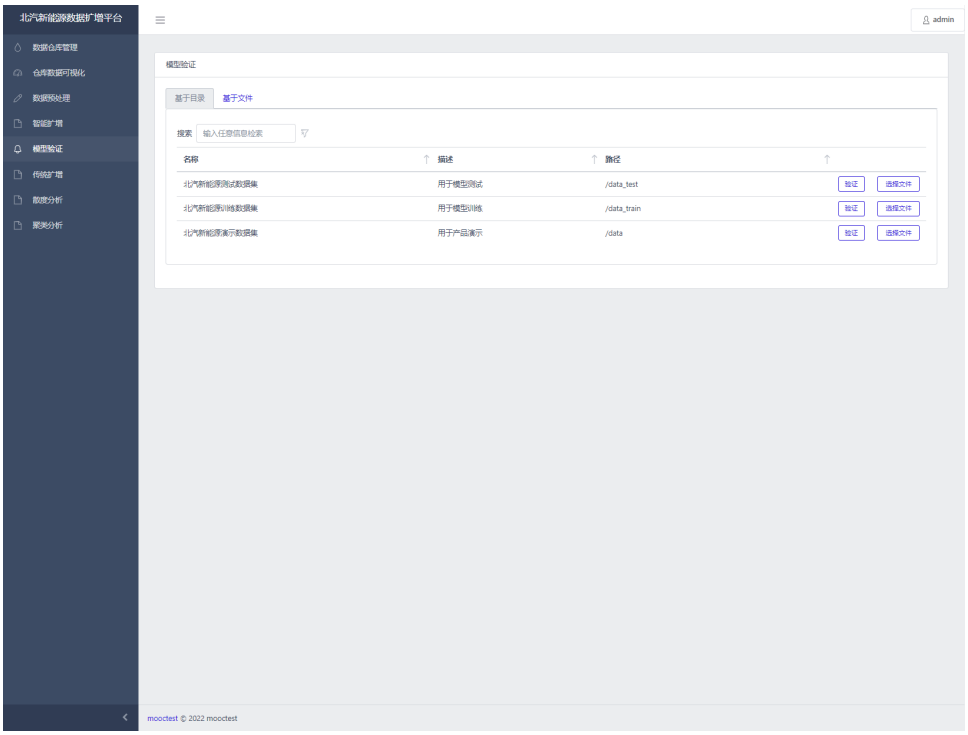

图 6.28: 模型验证界面

6.6 本章小结

本章主要介绍了系统相关的设计以及实现工作，包括项目的整体结构、需求分析和用例描述、持久化设计、以及系统各模块详细设计几个部分。需求分析

部分析了系统的功能性需求以及非功能性需求，并对核心功能模块建立了用例描述。系统概要设计部分介绍了系统的 4+1 视图，从不同视角对系统进行了描述。系统详细设计部分详细介绍了系统的实现方式，系统模块包括数据集管理模块、数据预处理模块、传统/智能扩增模块、以及模型验证模块。每个模块都给出了具体的执行流程图和核心实现代码，并进一步展示了界面布局情况。

第七章　系统测试与实验分析

7.1　系统测试

7.1.1　测试目标及测试环境

系统测试基于部署环境对系统的功能性需求进行测试，确保系统功能的可用性和稳定性。本节面向不同的功能模块设计对应的测试用例，针对数据仓库管理、数据预处理、传统扩增、智能扩增、以及模型验证模块具体功能给出具体的测试用例描述，通过比对预期目标和系统运行结果或者行为是否一致来判断系统功能是否达标。为了进一步确保系统质量，本项目还委托国家软件产品质量检验检测中心（江苏）进行可靠性测试，出具结果显示本系统所有功能均可稳定运行。

表 7.1: 测试环境参数列表

参数	参数详情
云端服务器	学院通用计算器
操作系统	Ubuntu 16.04 LTS
CPU	8
内存	16G
数据库	MySQL 5.7
容器	Docker 20.10.12
中间件	Nginx 1.21.5

测试阶段用于部署的服务器参数如表7.1所示，服务器采用 Ubuntu 操作系统，硬件配置为 8 核 16G 内存，Docker 版本号为 1.21.5。本系统利用 docker compose 方法将前端和 Nginx、后端、以及 MySQL 数据库引擎分别打包成三个独立的容器运行在服务器上，Nginx 和 MySQL 版本号分别为 1.21.5 和 5.7。

7.1.2　功能测试

功能测试主要负责对系统的功能模块进行验证，基于黑盒的方式判断测试条件下系统的输出结果或者行为与期望结果是否一致，从而对系统功能逻辑的正确性和完备性进行验证。同时，为了对系统功能的鲁棒性和健壮性进行验证，部分测试用例的规格将超出系统设计的规范。本节后续内容给出并详细介绍了围绕表6.2五个核心模块的功能测试用例。

UC1 数据仓库管理测试列表如表7.2所示，主要用于测试数据集管理功能是否与设计一致，包括数据集的注册和上传、删除、以及数据查询等。数据注册将已经存在于服务器中的数据集添加进系统进行管理，测试本项功能首先利用 scp 命令将目标数据集上传至服务器，然后点击注册按钮输入数据集名称、描述、以及路径信息，查看数据集列表是否添加成功。数据上传支持用户通过系统界面进行数据集的添加，点击上传按钮选择需要上传的 zip 数据集压缩文件，输入文件名和目录描述信息点击确认开始上传，文件上传后检查数据集列表是否自动添加相关记录信息。如果选择上传的数据集文件不是以 zip 结尾的文件，检查系统是否提示仅可上传 zip 文件包信息。测试数据集删除功能，在数据集列表中选择需要删除的对象并点击删除按钮，查看该数据集是否成功从列表中删除。测试查找功能，在查询框中输入需要检索的数据集名称检查数据集列表是否动态显示查询结果。通过比对系统行为和测试用例期望行为，实验结果显示设计的测试用例全部通过，表明数据仓库管理模块功能符合要求。

表 7.2: 数据仓库管理测试用例

编号	测试项	操作/输入	预期结果	测试结果
TC1-1-1	注册数据集	点击注册按钮并输入数据集信息	数据仓库列表显示被注册数据集	通过
TC1-1-2	删除数据集	选择需要删除的数据集并点击删除按钮	数据集从数据仓库列表中移除	通过
TC1-1-3	查找数据集	在查找框中输入数据集名称	数据仓库列表显示查询结果	通过
TC1-2-1	数据集上传	点击上传按钮选择 zip 文件包	数据集被添加进数据仓库列表	通过
TC1-2-2	数据集上传非 zip 文件	点击上传按钮选择非 zip 文件包	系统提示仅能上传 zip 文件包	通过

UC2 数据预处理测试用例入表7.3所示，测试项包括预处理状态显示，默认参数配置、以及任务进度显示三方面内容。测试默认参数配置功能，进入预处理界面检查是否显示默认的目标特征字段和双门限信息。对于预处理状态显示功能，首先检查所有新添加的数据集是否都显示预处理按钮，然后点击预处理按钮进行数据集预处理，最后确认该状态是否变为已预处理状态。对于进度显示功能，在点击预处理按钮并开始预处理后，检查系统是否实时显示预处理进度信息。所有测试用例全部通过说明预处理功能实现符合系统规范要求。

UC3 传统扩增测试列表如表7.4所示，主要测试选择扩增策略、配置策略参数、显示扩增任务进度、以及下载扩展数据集等功能。进入传统扩增页面选择要

表 7.3: 数据预处理测试用例

编号	测试项	操作/输入	预期结果	测试结果
TC2-1-1	数据预处理状态显示	进入预处理界面	预处理后的数据集显示预处理,没有预处理的数据集显示预处理按钮	通过
TC2-1-2	显示配置参数	点击预处理按钮	界面显示预处理配置参数	通过
TC2-1-3	显示预处理进度	点击开始按钮	预处理任务开始后界面动态显示任务进度	通过

扩增的文件点击扩增配置按钮，在弹出页面选择要扩增的策略点击下一步进入参数配置界面，检查参数配置是否与选中扩增策略一致，最后点击开始按钮检查系统是否进入进度显示界面。扩增任务结束后，在扩增界面或者扩增结果界面点击下载按钮，检查是否可以下载扩增数据集。

表 7.4: 传统算法扩增测试用例

编号	测试项	操作/输入	预期结果	测试结果
TC3-1-1	选择扩增策略	点击扩增按钮选择不通的扩增策略	生成对应扩增策略的数据样本	通过
TC3-1-2	提供默认参数	在参数配置界面点击下一步	使用默认配置进行系统扩增	通过
TC3-1-3	显示扩增进度	选择策略并开始扩增	扩增任务开始后界面动态显示扩增任务进度	通过
TC3-1-4	数据集下载	点击下载按钮	下载扩增数据集	通过

UC4 智能扩增测试列表如表7.5所示，首先从扩增参数配置、扩增任务进度显示、扩增结果下载等方面设计测试用例，再从扩增粒度着手设计测试用例确保系统功能可靠完整。测试基于文件智能扩增，首先进入智能扩增界面，选择目标数据集点击选择文件按钮，检查是否进入文件选择界面。然后选择需要测试的文件点击扩增按钮，检查是否有界面弹出，显示需要配置的参数且均提供了默认值。点击开始扩增按钮发起扩增任务后，检查是否进入进度显示页面。扩增任务完成后进入结果显示界面，点击下载按钮检查下载扩增数据集功能是否正常。基于目录的扩增测试，首先在模型扩增界面选择目标数据集，直接点击扩增按钮进入参数配置界面，检查参数配置是否提供默认值。然后发起扩增任务，此后流程与基于文件的扩增测试流程一致。测试用例全部通过表明智能扩增模块功能符合要求。

表 7.5: 智能模型扩增测试用例

编号	测试项	操作/输入	预期结果	测试结果
TC4-1-1	提供默认配置	进入智能扩增页面选择数据集点击扩增	系统显示默认配置	通过
TC4-1-2	智能模型扩增	点击下一步开始扩增	生成扩增数据样本	通过
TC4-1-3	显示扩增进度	点击下一步开始扩增	扩增任务开始后界面动态显示扩增任务进度	通过
TC4-1-4	数据集下载	点击下载按钮	下载扩增数据集	通过
TC4-2-1	选择文件扩增	用户选择数据集点击文件按钮	界面进入文件 tab 下并显示该目录下所有的文件, 用户可以选择具体文件进行扩增任务	通过
TC4-2-2	选择目录扩增	用户选择数据集点击扩增按钮	系统会对该目录下所有的文件执行相同的扩增任务	通过

UC5 模型验证测试列表如表7.6所示, 从模型验证的指标、任务进度显示、以及基于文件验证和基于目录验证等功能设计测试项目。基于文件验证测试，首先进入模型验证界面，选择目标数据集点击选择文件按钮，检查是否进入文件选择界面。选择需要进行验证的文件, 点击验证按钮发起验证任务, 检查任务进度显示是否正常。任务结束后检查系统是否跳转至结果界面。测试用例全部通过指示本项目模型验证模块符合功能需求。

表 7.6: 模型验证测试用例

编号	测试项	操作/输入	预期结果	测试结果
TC5-1-1	执行模型验证	进入模型验证界面选择验证数据集点击验证按钮	模型开始验证任务	通过
TC5-1-2	显示验证进度	进入模型验证界面选择验证数据集点击验证按钮	验证任务开始后界面动态显示验证任务进度	通过
TC5-1-3	显示验证指标	验证任务结束后	界面显示评估指标	通过
TC5-1-4	显示预测曲线	点击查看 SVG 图片按钮	在新界面显示预测曲线	通过
TC5-2-1	选择文件验证	用户选择数据集点击文件验证按钮	界面进入文件 tab 下并显示该目录下所有的文件, 用户可以选择具体文件进行扩增任务	通过
TC5-2-2	选择目录验证	用户选择数据集点击验证按钮	系统会对该目录下所有的文件执行相同的验证任务	通过

前面从系统功能角度设计了测试用例，测试结果表明系统设计符合需求用例描述，所有测试项目均通过确保系统功能的正确性。本章后续内容将基于实验结果，对比描述在数据空间和特征空间生成扩增样本与原始样本的区别。

7.2 实验分析

实验利用北汽新能源提供的实车采集数据、从数据空间和特征空间进行扩增对比实验。本节首先描述了传统扩增算法的结果，传统扩增算法基于序列变换算子模拟实车运行中可能的场景，从数据空间进行数据扩增。然后描述了智能扩增算法的结果，利用深度学习智能扩增模型从特征空间实现给定工况下的数据扩增。

7.2.1 传统扩增结果分析

（1）实验数据

本实验基于北汽新能源提供的数据集，对其中单车 56 小时的实车样本在数据空间进行扩增实验。

（2）扰动场景实验

扰动场景实验主要从传感器采集噪声的角度对数据进行扩增，从时域噪声扰动和频域噪声扰动两种维度完成。其结果如图7.1所示，其中蓝色曲线为 BMS 采集的原始电压数据，黄色曲线为添加噪声后的扩增数据；上图为频域引入噪声对比曲线，下图为时域引入噪声对比曲线。

图 7.1: 扰动算子数据扩增

从中可以看出时域高斯白噪声对原始数据的影响更加明显，而频域引入高斯白噪在时域的表现则不那么明显。因此可以判断，相同噪声条件下（相同的

均值和方差）时域噪声扰动扩增在数据空间的覆盖程度上要优于频域噪声扰动。在实际运用中可以调整频域噪声的方差值，从而引入更加强烈的噪声信号来加强频域扩增样本的空间覆盖能力。

（2）数据丢失场景实验

数据丢失场景实验主要从时域角度增加数据的空间覆盖情况。本实验通过预先配置的丢失比例和选择的策略（对称性、非对称性）对原始数据进行丢失操作。图7.1中显示了设置丢失参数为 20% 的对比结果，其中蓝色曲线为 BMS 采集的原始电压数据，黄色曲线为对称丢失策略数据扩增曲线，绿色为非对称丢失策略数据扩增曲线。由于原始样本样本序列过长全部显示会丢失很多细节，故图中仅仅显示了其中 1000 个数据样本的电压字段，即 2.7 小时的数据情况。

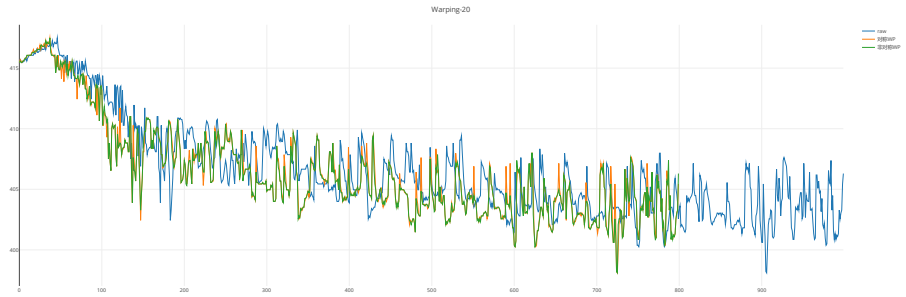

图 7.2: 窗口翘曲算子数据扩增

从中可以发现在相同的丢失比例下，黄色曲线的动态范围要稍微高于绿色曲线的动态范围，这与设计理论一致。黄色曲线基于 Warping 的下采样算子，固定丢失每一帧中固定位置的数据；而绿色曲线则根据每帧中数据的方差选择变化最大的数值进行丢弃。实际使用中，可以配置不同的丢失比例，利用对称性丢失策略和非对称性丢失策略来尽可能多的覆盖样本数据空间。

（3）噪声抑制场景实验

噪声抑制场景围绕实现数据的空间覆盖。图7.4和图7.3分别是指数平滑和窗口平滑的扩增对比曲线。图中蓝色曲线为原始样本数据，黄色曲线为平滑后的扩增样本曲线。在图7.3中，分别展示了平滑窗口为 5、10、15、20、25、以及 30 情况下的平滑对比情况；在图7.4中，通过调整衰减参数值，以 0.03 为步长从

0.90 增长到 0.99，展示了衰减参数对平滑的影响。

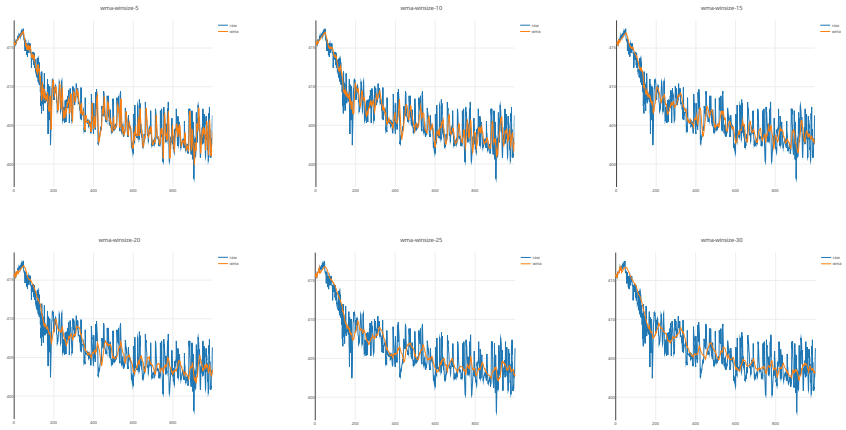

图 7.3: 窗口平滑算子数据扩增

从图7.3中可以发现，窗口平滑随着窗口长度的增加其扩增曲线趋于平缓。窗口大小从 5 增长到 15 时，对比曲线的变化比较明显。而当窗口大小从 25 开始增长到 30 时，窗口平滑的能力则没有明显的提升。在实际扩增过程中，将窗口大小设置为 5-20 可以达到最好的效果，避免过长的时间窗口引入无关的历史过程。

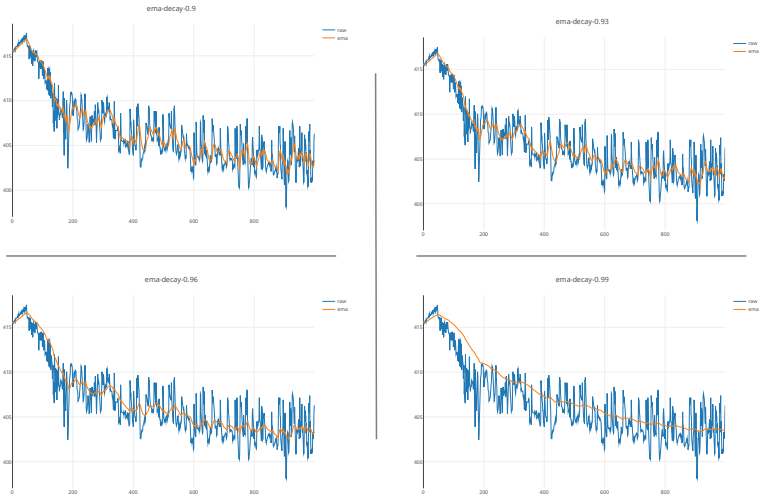

图 7.4: 指数平滑算子数据扩增

从图7.4中可以发现，随着衰减因子的增大扩增样本的曲线更加平滑。当衰

减因子增加到 0.99 时，扩增曲线几乎成为了一条平滑的曲线，原始数据中的跳变已经完全消失。对比指数平滑和窗口平滑可以发现，二者都能对数据进行滤波，让时间序列变得更加平滑；在平滑能力上来看，指数平滑对噪声的抑制要优于窗口平滑。

7.2.2 智能模型扩增结果分析

（1）实验数据和模型训练

本实验使用北汽新能源提供的数据训练扩增模型，包含 3520 小时实车记录，去除其中的空值以及异常值后剩余 3232 小时，总共 2GB 大小。将这些数据按 4:1 分配，即 80% 用于模型训练，20% 用于模型训练过程中的验证。模型训练和模型验证交叉进行，即每经过一个完整的训练过程，就进行一次模型验证。模型训练和验证的曲线如图7.5所示，图中黑色曲线为模型训练过程中的指标信息，绿色曲线为模型验证的指标信息。实验以 MSE 作为损失函数，即7.5图中的第一张子图；平均绝对误差（MAE）和决定系数（R2 score）作为验证指标，分别为图7.5中的第二、三张子图。

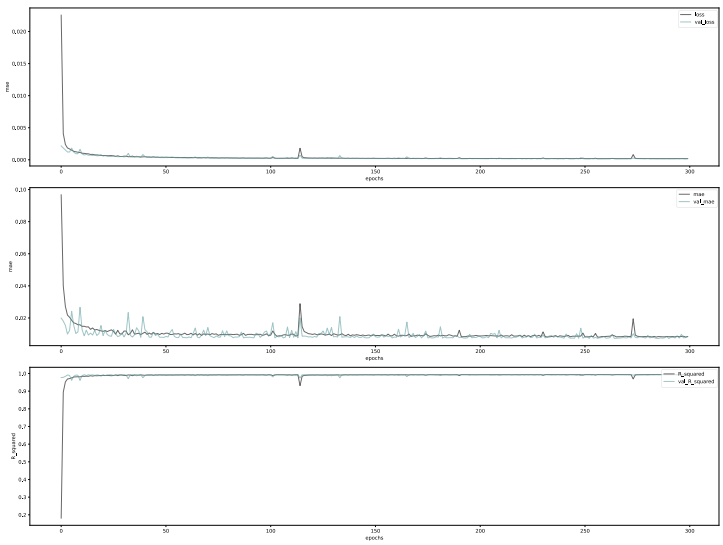

图 7.5: 模型训练和验证指标

从图7.5可以发现采用 Teacher Forcing 机制，模型的收敛速度还是非常快的，50 个训练周期基本稳定。且无论是训练指标还是验证指标都非常优异，平均绝对误差小于 0.02 且决定系数接近于 1。根据帧长的不同本实验训练了 6 个模型，帧长分别为 50，60，70，80，90，以及 100。

（2）模型扩增可靠性实验

由于模型采用迭代预测的部署方式，其预测行为和训练过程有稍微区别，故

模型训练过程中的验证指标并不能完全代表部署阶段模型的表现。对训练的六个模型进行部署，基于真实数据进行验证实验确定扩增模型迭代预测的可靠性。本实验选取连续 24 小时的样本送入扩增模型对比其迭代预测结果和真实电压结果之间的误差，其预测误差曲线如图7.7所示。图中不同颜色的曲线代表帧长不同的模型进行验证的结果，x 轴为原始数据的帧号，y 轴为对应的电压误差值。不同帧长导致相同时长的数据被分成不同的帧数，比如一帧长度为 50 个样本时，24 小时记录可以分为 180 帧。对每帧预测计算其均方根误差（RMSE），同时统计每帧最大根方误差（MRSE）情况，均方误差用来表现宏观预测表现，而最大根方误差用来说明预测细节。图7.7中上面的子图表示每帧的 RMSE 值，而下面的子图表示每帧中最大的根方误差。

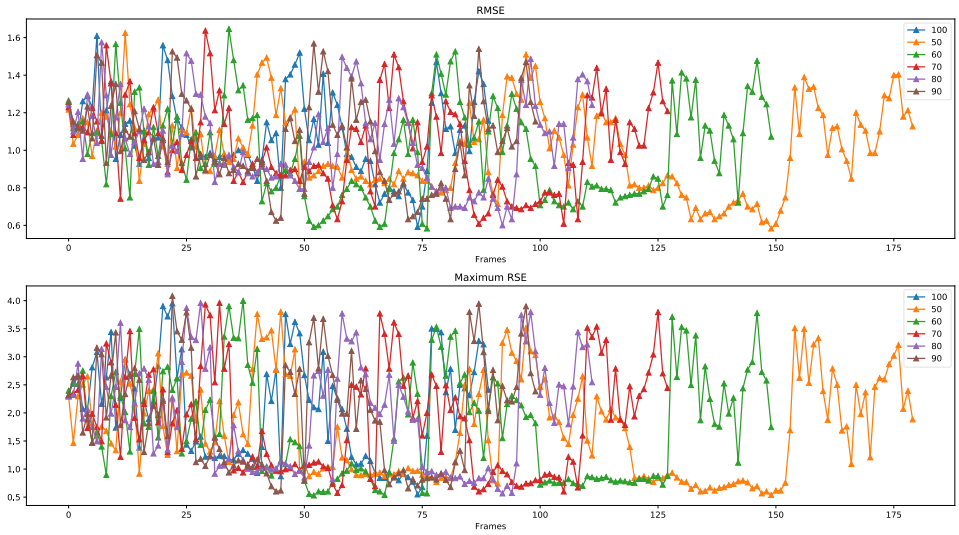

图 7.6: 部署阶段不同帧长模型验证误差曲线

从中可以发现模型预测的 RMSE 和 MRSE 别在 1.5V 和 4V 以内，具体细节如表7.7所示，实验中每验证一帧都会统计该帧下的 RMSE 和 MRSE 值。表中第一列和第二列分别为验证模型的帧长以及在该帧长下验证数据集的帧数；第三列“均方根误差”记录了所有帧数下的 RMSE 的平均值；剩余几列是基于 MRSE 的统计结果，“最大方根误差”记录了验证过程中所有帧 MRSE 值的最大值，“最大方根误差均值”和“最大方根误差标准差”统计了所有验证帧 MRSE 的均值和标准差情况。

从表7.7可以判断随着模型帧长的上升，模型迭代预测的 RMSE 值和 MRSE 呈现上升趋势，其中帧长为 50 的模型表现最佳，帧长为 100 的模型误差最大，和前期判断一致。虽然帧长越长误差越大，但是本项目很好的将均方根误差、最大方根误差控制在了 1.5V（0.36%）和 4V（1%）以内，证明迭代预测结果的可

表 7.7: 不通帧长下模型验证误差

帧长	帧数	均方根误差	最大方根误差	最大方根误差均值	最大方根误差标准差
50	180	1.33	3.80	1.73	1.0
60	150	1.38	3.99	1.79	1.0
70	128	1.38	3.95	1.86	1.0
80	112	1.41	3.96	1.94	1.0
90	100	1.45	**4.08**	1.99	1.0
100	90	**1.47**	3.95	2.05	1.0

靠性。模型扩增以帧为单位进行，帧长越短其生成的样本信息也就越少，相对的帧长越长生成的数据样本信息量就越高。在实际应用中，为了权衡误差和信息量，推荐终采用帧长为 80 所训练的模型进行部署。

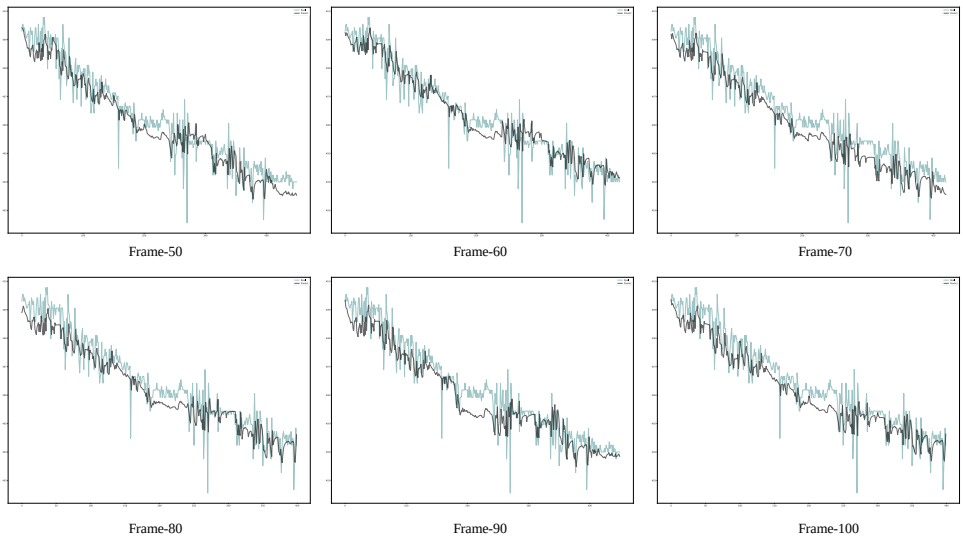

图 7.7: 部署阶段不同帧长模型预测结果

图7.6展示了部署阶段使用不同帧长训练的模型的实际验证结果曲线。从对比曲线可以看出，帧长从 50 到 100 训练的扩增模型都能较好的进行迭代预测，模型预测行为和量化指标一致。

（3）工况提取充放电比列实验

前面的实验结果论证了本项目基于迭代预测扩增模型进行数据生成的可靠性。为了控制数据生成过程中充放电样本的比例，需要调整工况提取算子的权重参数。本实验以训练样本为目标工况数据源，设置提取 50 帧工况序列，在帧长为 80 的条件下，累积提取 4000 条数据样本。权重参数从 0.2 到 0.35 结果如图7.8所示，其中黑色柱状图表示充电过程样本数目，蓝色柱状图表示放电过程

样本数目。

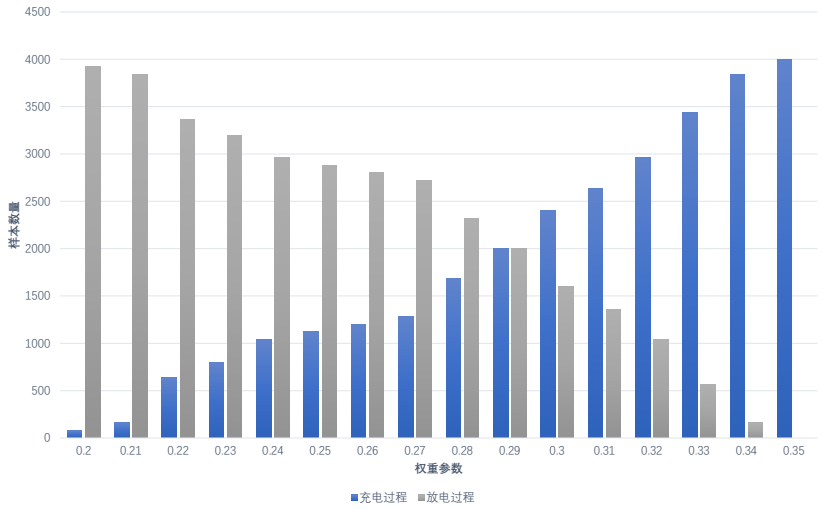

图 7.8: 权重与工况序列样本数目关系

从表中可以发现，随着权重参数的增长，提取的 20 帧目标工况中充电样本数目同步增加而放电过程同步减少，当权重为 0.29 时，充电过程和放电过程数目想等。需要注意的是，此实验得出的参数条件并不具有普适性。在实际运用中，不同数据集由于历史过程的差异导致其电流工作强度变化和本实验数据集并不一致，因此需要根据自身使用的数据集情况动态调整权重参数，使得扩增工况序列中充电过程和放电过程的样本数目保持一致。

7.3 案例演示

本节将基于系统的核心功能即智能扩增业务进行项目演示，从如何添加数据集，到对数据集进行预处理，最后执行智能扩增工作，本节详细介绍了一次智能扩增的完整流程。演示过程使用的数据集由北汽新能源汽车公司提供。

首先演示如何上传数据集，用户进入数据仓库管理界面并点击上传按钮后，系统弹框如图7.9所示，用户在该界面中输入数据集名称和描述后，点击上传按钮选择需要上传的 zip 压缩文件。上传完成后，系统会将用户上传的 zip 文件解压缩到静态文件存储目录并删除原始 zip 文件，最后将该条数据集信息添加至 MySQL 数据库。

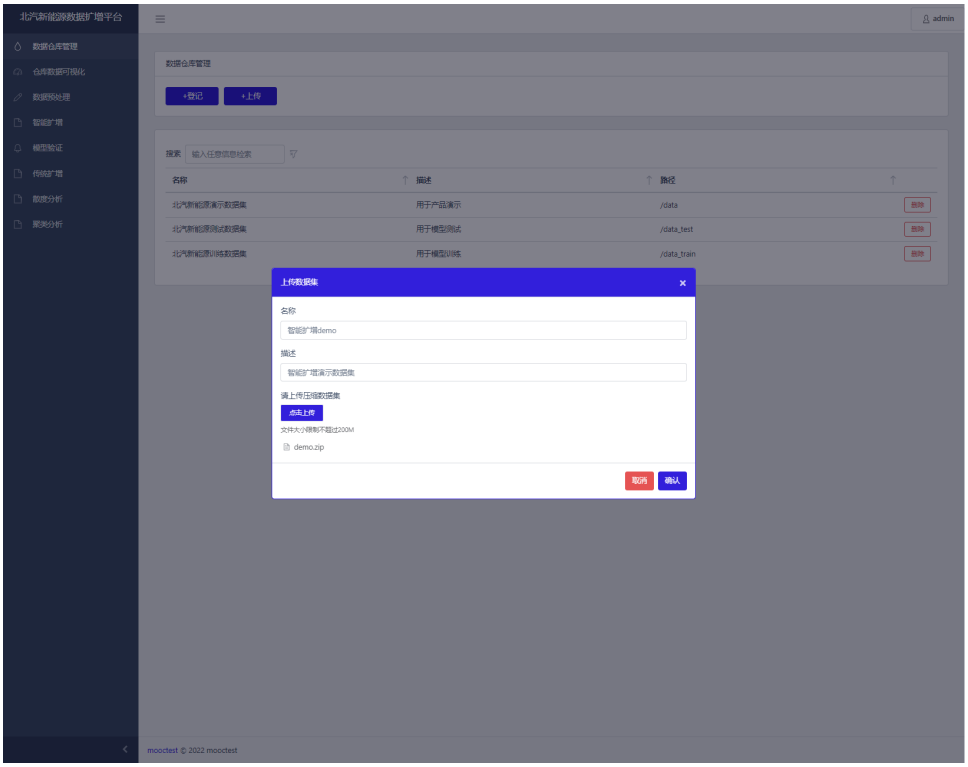

图 7.9: 添加数据集

用户可在数据仓库管理界面的数据集列表中查看之前上传的数据集信息，除了之前用户提供的数据集名称和描述外，还显示了当前数据集在服务器上的解压缩路径信息，如图7.10所示。

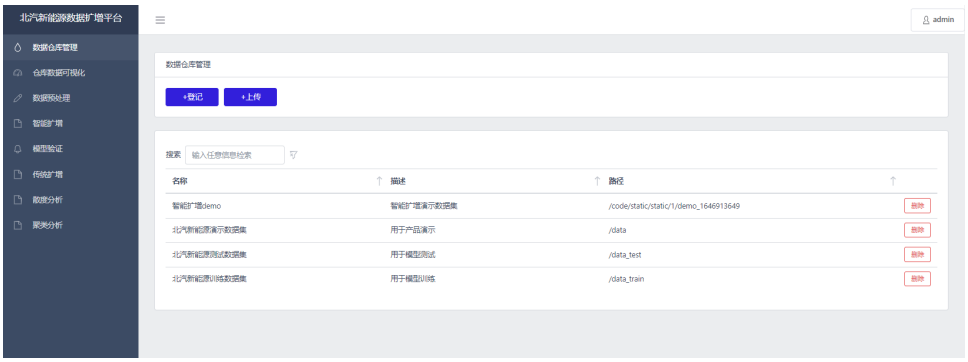

图 7.10: 用户数据集列表

数据预处理界面如图7.11所示，该界面显示了当前用户下管理的数据集状态，对于已经预处理的数据集显示“已预处理”标签，其余数据集显示“开始预处理”按钮。对于刚刚上传的数据集可以看见其状态处于未预处理阶段，如

图7.11所示。点击“开始预处理”按钮后，系统会显示特征筛选后在原始数据集中的列号信息，如图7.12所示，对于数据排布不一致的原始数据集，用户可以修改特征列号进行匹配。点击“开始”按钮发起预处理任务，任务结束后当前数据集的状态从未处理变为“已预处理”。

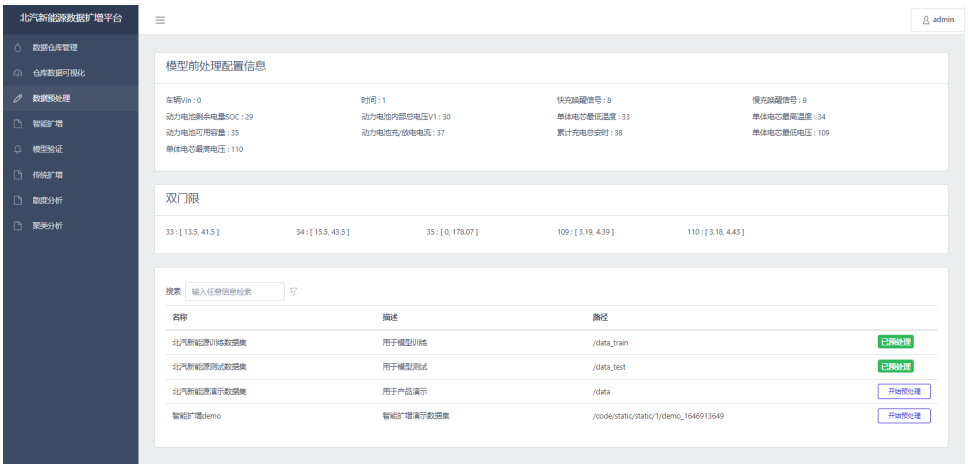

图 7.11: 数据预处理界面

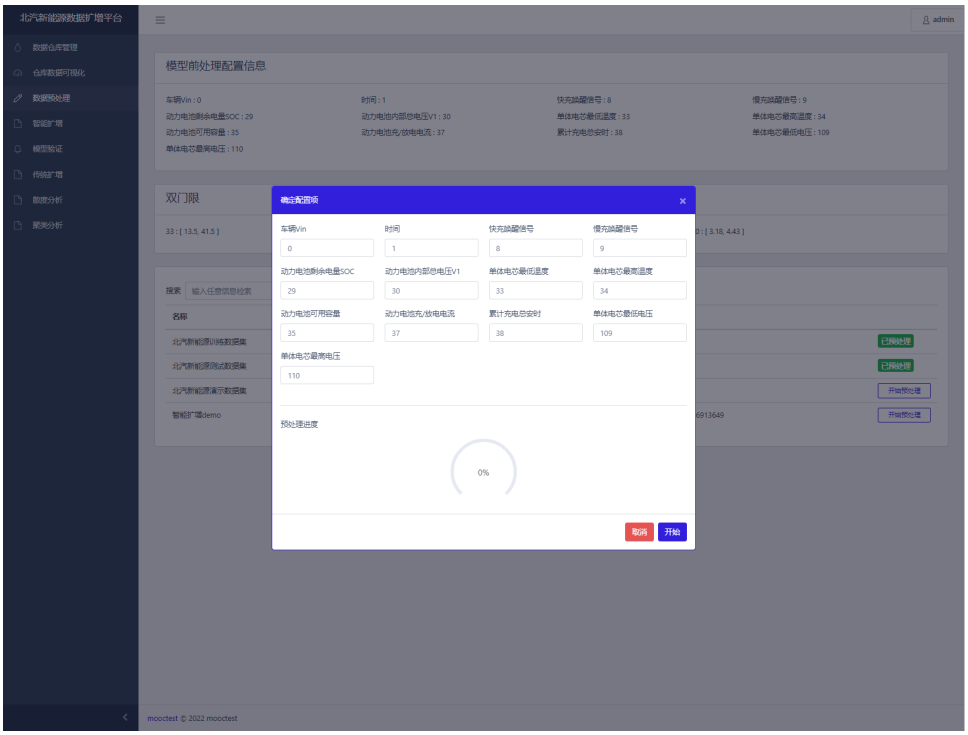

图 7.12: 预处理配置

数据扩增界面如图7.13所示，在该界面中用户可以选择目标数据集发起数据

扩增任务。选择刚刚上传的数据集点击“扩增”按钮发起扩增请求，系统会弹出页面显示智能扩增的相关配置信息。

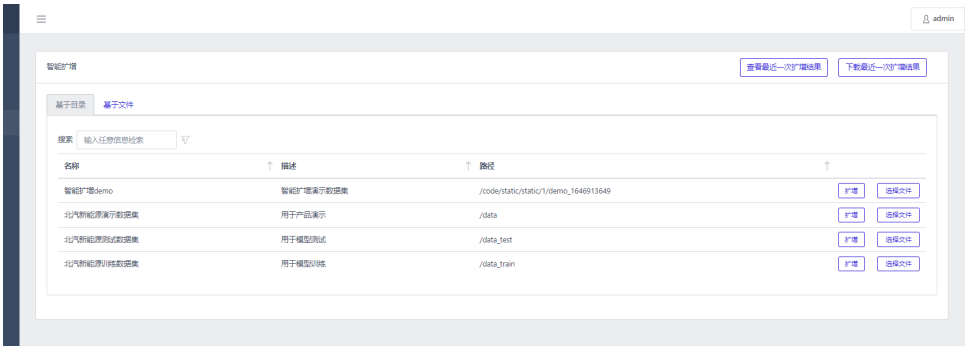

图 7.13: 智能扩增界面

扩增参数如图7.14所示：“头信息提取间隔”表示从扩增文件中提取头部序列 head_info 间隔多少数据提取一次；“头部信息提取数目”表示一个扩增文件中提取出多少个头部序列；“充电过程权重”表示提取工况时充电样本的权重；“工况提取间隔”表示从“工况目录”文件中间隔多少样本提取一帧工况数据；“工况提取帧数”表示从“工况目录”中提取的工况帧数个数。

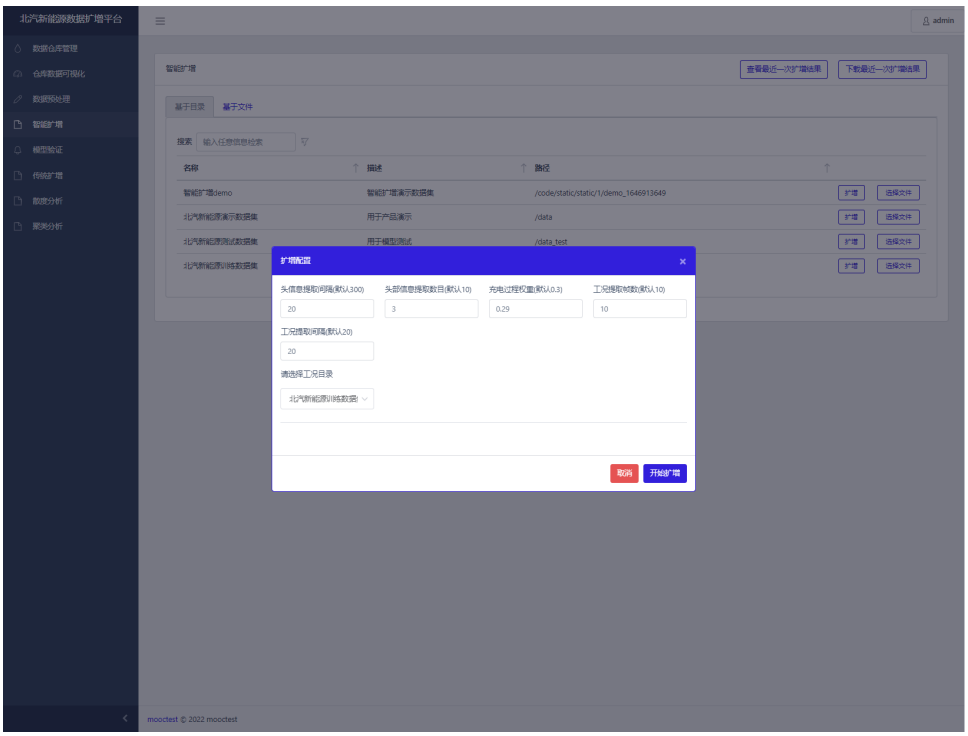

图 7.14: 智能扩增参数配置

扩增参数配置完成后，点击“开始扩增”按钮即可发起扩增任务。系统会自动跳转至进度界面，如图7.15所示，该界面会动态显示当前扩增任务的进度，从0 增长至 100，进度 0 表示刚刚开始，100 表示扩增完成。

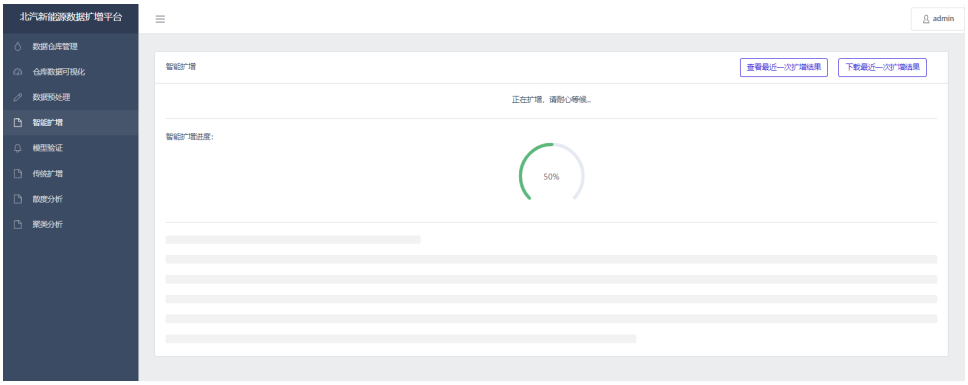

图 7.15: 智能扩增进度界面

任务进度达到 100 后，系统会跳转至结果显示界面，如图7.16所示。该界面会显示当前扩增的样本名称、扩增样本中充电数据的比例、扩增的总帧数、以及扩增数据的大小等信息。用户可以点击结果界面或者智能扩增界面中的“下载最近扩增结果”按钮下载最近一次扩增的压缩文件。

图 7.16: 智能扩增结果界面

7.4 本章小结

本章中首先描述了如何对系统进行功能性测试，确保系统的正确性和完备性。针对系统用例和功能性需求，本项目对系统中每个模块都设计了测试用例，基于黑盒测试的方式判断当前系统功能是否符合设计要求。然后，通过扩增对比实验，介绍了传统扩增和智能扩增的特点，并对智能扩增的准确性进行了验证，说明了智能扩增方法的有效性。接着，通过权重参数实验证实了工况提取算子可以有效控制扩增样本中的充放电比列。最后，对系统的核心功能即智能扩增进行了演示。

第八章　总结与展望

8.1　总结

随着深度学习技术的发展和车载算力的提升，越来越多的新能源汽车厂商选择深度技术作为预测新能源电池 SOC 的方法。由于模型的不可解释性，开发者无法确定模型从训练数据集中到底学习了什么内容，且无法从逻辑推导的角度对参数进行验证。现如今，深度学习模型的质量保障工作完全依赖测试数据集进行。受限于新能源电池的特殊性，在给定电池状态和目标工况下条件下进行数据采集往往需要很高的成本和时间开销。因此，为了提高测试数据集质量和场景多样性，就需要数据扩增技术的支撑。然而，针对新能源电池数据扩增技术的研究尚处于起步阶段。

为了解决上述难题，保障深度学习技术在新能源汽车领域安全落地。本项目基于传统算法和深度学习技术从数据空间和特征空间对新能源电池数据样本进行扩增，提高测试数据集的多样性和均衡性。本项目：（1）基于新能源测试数据集样本特性设计时间序列变换算子，模拟不同场景下的样本表现；（2）首次提出将基于迭代预测的时间序列深度学习模型运用到新能源电池数据扩增任务；（3）设计工况提取算子用于控制扩增样本的工作过程和充放电样本比列；（4）设计模型验证过程确保扩增数据在目标工况下准确可靠；（5）解决了 BMS 采样率过低导致无法使用安时积分法对扩增样本标注的问题；（6）设计并实现了云端数据扩增系统供用户使用。

本文首先介绍课题的背景和研究意义，阐述了当前 SOC 估计的主流方法以及缺陷，说明采用深度学习解决方案是大势所趋，进一步揭示了数据扩增在新能源领域的重要性和严重不足。然后描述如何基于特征工程对北汽新能源提供的实车采集数据进行特征选择和数据清洗，将精炼后的数据集作为源进行扩增工作。其次描述了传统扩增方法，该方法基于时间序列变换算子面向三个应用场景设计了六种扩增策略，围绕扰动、丢失和噪声抑制从数据空间对原始数据集进行了补充。接下来描述了智能扩增模型的相关内容，包括网络结构、配置参数、以及训练过程。同时说明了如何从现有数据集中提取目标工况序列并对智能扩增的结果进行 SOC 标注。紧接着描述了本系统的需求分析、概要设计、以及详细设计的内容，给出系统架构图、4+1 视图、以及持久化设计的 E-R 图。为了保障系统稳定运行，本文最后给出了功能性测试结果，并对扩增实验结果进行了分析。

本扩增系统通过国家软件产品质量检验检测中心检验，符合 GB/T 25000.51-2016《系统与软件工程系统与软件质量要求和评价（SQuaRE）第 51 部分：就绪可用软件产品（RUSP）的质量要求和测试细则》国家标准。现已在北汽新能源内部试运行，用于其 BMS 荷电状态估计的质量保障工作。

8.2 未来工作展望

本文设计并实现了面向新能源汽车 BMS 荷电状态估计的数据扩增系统，该系统在以下几点上仍有上升的空间：

第一，基于迭代预测的扩增模型以帧为单位进行序列扩增，帧数越长扩增样本所包含的数据信息越多。可以研究新的模型结构或者训练策略，使得模型的帧长进一步延伸。

第二，本项目设计的工况序列提取算子依据一帧的能量变化进行工况筛选，此方法每次都会输出源目录中能量变化最大的几帧内容，缺乏多指标和随机性设计。多指标即利用其他特征作为筛选指标实现目标工况的提取工作；随机性设计指在相同的参数下，每次从源目录提取的目标工况序列都会产生变化。

第三，虽然智能扩增能够在特征空间基于工况序列进行新能源电池的数据扩增，但是目前没有自适应目标数据集充放电比例的功能，导致相同权重参数在不同源数据上提取的工况序列充放电样本并不一致。当前通过人工调整的方式保证扩增样本充放电比例一致，后续可以基于输入扩增样本的充放电样本情况，自适应选择权重参数，从而平衡扩增样本中的充放电比例。

参考文献

[1] BILGIN B, MAGNE P, MALYSZ P, et al. Making the Case for Electrified Transportation[J]. IEEE Transactions on Transportation Electrification, 2015, 1(1): 4–17.

[2] SUN Y, SUN M, TANG X. Influence Analysis of Renewable Energy on Crude Oil Future Market[C] // 2019 IEEE 3rd International Conference on Green Energy and Applications (ICGEA). 2019: 167–171.

[3] CASTANO S, SERRANO-JIMENEZ D, SANZ J. BMS influence on Li-ion packs characterization and modeling[C] // 2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC). 2016: 1–6.

[4] ANDREA D. [M]. [S.l.]: Battery Management Systems for Large Lithium-Ion Battery Packs, 2010.

[5] MENG J, RICCO M, LUO G, et al. An Overview and Comparison of Online Implementable SOC Estimation Methods for Lithium-Ion Battery[J]. IEEE Transactions on Industry Applications, 2018, 54(2): 1583–1591.

[6] CHAOUI H, IBE-EKEOCHA C C. State of Charge and State of Health Estimation for Lithium Batteries Using Recurrent Neural Networks[J]. IEEE Transactions on Vehicular Technology, 2017, 66(10): 8773–8783.

[7] LU L, HAN X, LI J, et al. A review on the key issues for lithium-ion battery management in electric vehicles[J]. Journal of Power Sources, 2013, 226: 272–288.

[8] PLETT G. [M]. [S.l.]: Battery Management Systems, Volume II: Equivalent-Circuit Methods, 2015.

[9] SAJI D, BABU P S, ILANGO K. SoC Estimation of Lithium Ion Battery Using Combined Coulomb Counting and Fuzzy Logic Method[C] // 2019 4th International Conference on Recent Trends on Electronics, Information, Communication Technology (RTEICT). 2019: 948–952.

[10] HE L, GUO D. An Improved Coulomb Counting Approach Based on Numerical Iteration for SOC Estimation With Real-Time Error Correction Ability[J]. IEEE Access, 2019, 7 : 74274 – 74282.

[11] BAE J-H, ZHIGUO B. The development of technology on reduces the SOC error rate using Hybrid Kalman Filter(HKF) and the demonstration its performance using BMS platform[C] // 2016 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific). 2016 : 661 – 665.

[12] SUSANNA S, DEWANGGA B R, WAHYUNGORO O, et al. Comparison of Simple Battery Model and Thevenin Battery Model for SOC Estimation Based on OCV Method[C] // 2019 International Conference on Information and Communications Technology (ICOIACT). 2019 : 738 – 743.

[13] JU L, GENG G, JIANG Q, et al. An Adaptive OCV-SOC Curve Selection Classifier for Battery State-of-Charge Estimation[C] // 2021 3rd International Conference on Smart Power Internet Energy Systems (SPIES). 2021 : 457 – 463.

[14] WANG Y, LI L, DING Q, et al. Lithium-ion battery SOC estimation based on an improved adaptive extended Kalman filter[C] // 2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA). 2021 : 417 – 421.

[15] MA C, WU Q, HOU Y, et al. SOC Estimation of Lithium Battery based on Fuzzy Kalman Filter Algorithm[C] // 2020 35th Youth Academic Annual Conference of Chinese Association of Automation (YAC). 2020 : 324 – 329.

[16] JUNRUI W, TENG G, XINJU W, et al. SOC Estimation of Extended Kalman Filter Based on Hardware-in-the-Loop Simulation Platform[C] // 2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA). 2021 : 1610 – 1613.

[17] CHEMALI E, KOLLMEYER P J, PREINDL M, et al. Long Short-Term Memory Networks for Accurate State-of-Charge Estimation of Li-ion Batteries[J]. IEEE Transactions on Industrial Electronics, 2018, 65(8) : 6730 – 6739.

[18] YANG F, SONG X, XU F, et al. State-of-Charge Estimation of Lithium-Ion Batteries via Long Short-Term Memory Network[J]. IEEE Access, 2019, 7 : 53792 – 53799.

[19] CALIWAG A C, LIM W. Hybrid VARMA and LSTM Method for Lithium-ion Battery State-of-Charge and Output Voltage Forecasting in Electric Motorcycle Applications[J]. IEEE Access, 2019, 7 : 59680 – 59689.

[20] TIAN Y, LAI R, LI X, et al. A combined method for state-of-charge estimation for lithium-ion batteries using a long short-term memory network and an adaptive cubature Kalman filter[J]. Applied Energy, 2020, 265 : 114789.

[21] KIM S J, LEE J H, WANG D H, et al. LSTM-Based Real-Time SOC Estimation of Lithium-Ion Batteries Using a Vehicle Driving Simulator[C] // 2021 21st International Conference on Control, Automation and Systems (ICCAS). 2021 : 618 – 622.

[22] LIU B, ZHANG Z, CUI R. Efficient Time Series Augmentation Methods[C] // 2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI). 2020 : 1004 – 1009.

[23] LI Y, LIU J, YANG T. Dynamic Harmonic State Estimation of Power System Based on Sage-Husa Square-Root Unscented Kalman Filter[C] // 2019 IEEE 3rd International Electrical and Energy Conference (CIEEC). 2019 : 478 – 483.

[24] FIELDS T, HSIEH G, CHENOU J. Mitigating Drift in Time Series Data with Noise Augmentation[C] // 2019 International Conference on Computational Science and Computational Intelligence (CSCI). 2019 : 227 – 230.

[25] GOUBEAUD M, JOUßEN P, GMYREK N, et al. White Noise Windows: Data Augmentation for Time Series[C] // 2021 7th International Conference on Optimization and Applications (ICOA). 2021 : 1 – 5.

[26] GAO J, SONG X, WEN Q, et al. RobustTAD: Robust Time Series Anomaly Detection via Decomposition and Convolutional Neural Networks[J]. CoRR, 2020, abs/2002.09545.

[27] FORESTIER G, PETITJEAN F, DAU H A, et al. Generating Synthetic Time Series to Augment Sparse Datasets[C] // 2017 IEEE International Conference on Data Mining (ICDM). 2017 : 865 – 870.

[28] YANG X, ZHANG Z, CUI X, et al. A Time Series Data Augmentation Method Based on Dynamic Time Warping[C] // 2021 International Conference on Computer Communication and Artificial Intelligence (CCAI). 2021 : 116 – 120.

[29] CHOWDHURY S S, BOUBRAHIMI S F, HAMDI S M. Time Series Data Augmentation using Time-Warped Auto-Encoders[C] // 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA). 2021 : 467 – 470.

[30] 华周发; 李静;. 电动汽车动力电池 soc 估算方法综述 [J]. 电源技术, 2013, 37 : 1686 – 1689.

[31] ZHOU Y, LI X. Overview of lithium-ion battery SOC estimation[C] // 2015 IEEE International Conference on Information and Automation. 2015 : 2454 – 2459.

[32] MENG J, RICCO M, LUO G, et al. An overview of online implementable SOC estimation methods for Lithium-ion batteries[C] // 2017 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM) 2017 Intl Aegean Conference on Electrical Machines and Power Electronics (ACEMP). 2017 : 573 – 580.

[33] 杨丽娟; 张白桦; 叶旭桢. 快速傅里叶变换 fft 及其应用 [J]. 光电工程, 2004 : 1 – 3+7.

[34] GREFF K, SRIVASTAVA R K, KOUTNíK J, et al. LSTM: A Search Space Odyssey[J]. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(10) : 2222 – 2232.

[35] WANG Y, ZHU S, LI C. Research on Multistep Time Series Prediction Based on LSTM[C] // 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE). 2019 : 1155 – 1159.

[36] QU H, LI J, ZHANG Y. Long Short-term Memory Network Prediction Model Based on Fuzzy Time Series[C] // 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS). 2020 : 417 – 421.

[37] GHANBARI R, BORNA K. Multivariate Time-Series Prediction Using LSTM Neural Networks[C] // 2021 26th International Computer Conference, Computer Society of Iran (CSICC). 2021 : 1 – 5.

[38] SIAMI-NAMINI S, TAVAKOLI N, SIAMI NAMIN A. A Comparison of ARIMA and LSTM in Forecasting Time Series[C] // 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). 2018 : 1394 – 1401.

[39] KINGMA D P, BA J. Adam: A Method for Stochastic Optimization[C] // BENGIO Y, LECUN Y. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. 2015.

[40] ZHAO H, TSAI Y H, SALAKHUTDINOV R, et al. Learning Neural Networks with Adaptive Regularization[C] // WALLACH H M, LAROCHELLE H, BEYGELZIMER A, et al. Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. 2019 : 11389 – 11400.

[41] KAMALOV F, LEUNG H H. Deep learning regularization in imbalanced data[C] // 2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI). 2020 : 1 – 5.

[42] JINDAL I, NOKLEBY M, CHEN X. Learning Deep Networks from Noisy Labels with Dropout Regularization[C] // 2016 IEEE 16th International Conference on Data Mining (ICDM). 2016 : 967 – 972.

[43] KHAN S H, HAYAT M, PORIKLI F. Regularization of deep neural networks with spectral dropout[J]. Neural Networks, 2019, 110 : 82 – 90.

简历与科研成果

基本情况 倪烨，男，汉族，1992 年 3 月出生，江苏扬州人

教育背景

2020.9 – 2022.6　南京大学软件学院　　硕士

2010.9 – 2014.6　南京工程学院通信工程学院　　本科

读研期间的成果（包括发表的论文及参与的专利）

1. **Y. Ni**, Z. Xia, F. Zhao, C. Fang and Z. Chen, ”An Online Multistep-Forward Voltage-Prediction Approach Based on an LSTM-TD Model and KF Algorithm,” in Computer, vol. 54, no. 8, pp. 56-65, Aug. 2021, doi: 10.1109/MC.2021.3070314.
2. 王晓冰、**倪烨**、房春荣，“一种基于智能交通模型可靠性的变异测试方法”，申请号：020107125658，已受理。
3. 陈振宇、郭安、何天行、**倪烨**，“一种基于语义的自动驾驶测试图像场景构建方法”，申请号：2020107127649，已受理。
4. 陈振宇、邓靖琦、**倪烨**，“一种面向自动驾驶的多场景自动化点云扩增方法”，申请号：2020107127865，已受理。
5. 陈振宇、徐彬桐、**倪烨**，“一种面向自动驾驶图像数据的测试用例生成方法”，申请号：2020107127884，已受理。
6. 陈振宇、曹可凡、**倪烨**，“一种面向智能交通软硬件数据准确性的评估方法”，申请号：2020107127901，已受理。
7. 王晓冰、吉品、王兴亚、**倪烨**，“一种基于测试需求的语音识别系统众包测试用例生成方法”，申请号：2020107146476，已受理。
8. 王晓冰、徐彬桐、**倪烨**，“一种面向自动驾驶系统的蜕变测试方法”，申请号：202010714664x，已受理。

致　　谢

岁月如梭，韶光易逝，研究生生涯走到了尽头，心中充满了不舍和感激。回望在南京大学的两年时光，有辛勤的付出、激励的讨论，更有成功的收获。这里，我向所有关心和帮助过我的同学和老师表示最诚挚的感谢。

感谢陈振宇老师、房春荣老师的悉心栽培，无论是在论文撰写、专利申请还是项目设计，二位老师都给予了很多指导，帮助我克服一道道难关并最终取得满意的成果。作为我心中的榜样，陈老师乐观的生活方式、严谨的治学态度是我终身学习的目标。作为学术启蒙老师，在学术思维的构建上，房老师让我受益匪浅，充分体会到学术研究的不易与枯燥。同时，还要感谢南京大学软件学院智能软件工程实验室 (Intelligent Software Engineering, iSE) 的所有老师和同学，你们锲而不舍追求目标的精神激励我不断奋斗，奋勇前行。最后，我想感谢我的家人和室友，你们的陪伴和支持让我以乐观积极的心态度过了愉快的两年学校生活。

虽然这一段旅途即将到达终点，再过不久我们就要分离，但是回忆将永远留在我的心间，最后祝愿大家都有美好的前程。