

分类号
学校代码 10487

学号 D201880***
密级

华中科技大学

博士学位论文

(学术型 ☒ 专业型 ☐)

基于随机映射的时间序列深度学习 预测建模技术

学位申请人：张心泽

学科专业：管理科学与工程

指导教师：鲍玉昆 教授

答辩日期：2023年4月26日

答辩委员会

	姓名	职称	单位
主席	X X	教授	武汉大学 信息管理学院
委员	X X	教授	武汉大学 信息管理学院
	XXX	教授	华中科技大学 管理学院
	XXX	教授	华中科技大学 管理学院
	X X	教授	华中科技大学 管理学院

**A Dissertation Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy in Management**

**Stochastic Deep Learning Based Time Series
Forecasting Techniques**

Ph.D. Candidate : Zhang Xinze

Major : Management Science and Engineering

Supervisor : Prof. Bao Yukun

Huazhong University of Science and Technology

Wuhan 430074, P. R. China

April, 2023

独创性声明

本人声明所呈交的学位论文是我个人在导师的指导下进行的研究工作及取得的研究成果。尽我所知，除文中已标明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期： 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本论文属于 ☐ 保 密 ☐，在 ____ 年解密后适用本授权书。
☐ 不保密 ☐。

（请在以上方框内打“√”）

学位论文作者签名：

指导教师签名：

日期： 年 月 日

日期： 年 月 日

摘要

时间序列预测建模技术是能源、金融和公共卫生等众多应用领域的重要技术。针对时间序列预测建模问题，深度学习预测建模技术因其优异的预测性能成为了近年的研究热点。然而，以卷积神经网络和循环神经网络为代表的深度神经网络预测模型受限于梯度下降权重训练方法，需要在反复训练权重参数的基础上才可具备优异的预测性能，因此带来昂贵的计算消耗；加之深度学习预测模型复杂的结构参数设置，使得深度学习预测建模技术面临建模效率低与模型选择难等问题。随机映射方法作为神经网络模型的一种非迭代训练方法，通过固定模型随机初始化的输入层与隐藏层权重，利用简单直接的闭式求解算法计算模型输出层权重，以此完成权重参数训练过程，因而可有效提升神经网络模型的构造效率。

本学位论文基于随机映射方法对时间序列深度学习预测建模技术展开研究，针对随机深度神经网络预测模型在卷积结构和循环结构上的模型选择特有个性问题，提出随机卷积隐藏结构的构造优化方法和随机循环输出结构的构造优化方法；在此基础上探索随机深度神经网络特定代表结构的共性优化问题，提出随机深度神经网络预测模型一般结构下的二重特征选择方法和混合结构下的生长优化方法，进一步提升模型预测性能；在技术方法研究的同时，结合流感阳率预测、原油价格预测、电力负荷预测与电力价格预测等重要应用场景进行应用研究。本文的主要研究工作和创新性成果如下：

首先，针对卷积结构的随机深度神经网络预测模型构造与优化问题，提出基于误差反馈随机建模和贪心优化的新颖构造与优化方法，解决了既有方法的参数选择不足及输出权重病态矩阵难题。通过向隐藏层中递归添加随机卷积核，利用最小二乘法局部更新输出权重，构造出随机卷积隐藏结构，并证明了该构造方法的收敛性；借助贪心搜索优化卷积核参数，使其在单卷积层内自适应选择不同宽度的卷积核，具备了不同尺度局部特征的学习能力。在合成数据和多项现实数据上的实验表明，该模型具有极高的模型构造与选择效率，表现出与梯度下降训练的深度学习预测模型媲美甚至更优的预测性能。

其次，针对循环结构的随机深度神经网络预测模型构造与优化问题，提出了基于状态遮掩和粒子群优化的新颖构造与优化方法，解决了既有方法忽视的输出结构优化问题。基于对循环神经网络输出结构的随机映射实现，分析得出不同输出结构的异同优劣；通过向循环隐藏特征表示施加状态掩码，提出状态遮掩的随机循环输出结构，借助粒子群优化自适应选择循环隐藏特征表示和目标的映射取舍，提升了输出结构的学习能力。在人工合成、电力负荷和室外温度数据集上的实验表明，与既有随机循环神经网络及其输出结构优化方法相比，基于所提方法构造的模型具备明显的多步预测优势。

再次，针对一般结构的随机深度神经网络预测模型特征选择问题，提出了基于二重特征结构选择和树状结构帕森斯估计的新颖特征选择方法，消除了既有选择方法的一维输入特征结构局限。结合深度神经网络的时步多维度读取能力，利用滑动窗口构造时步多维度的二维输入特征结构；通过向每一输入时步添加时步掩码，建立基于时步维度与时步掩码组合的二重特征结构表示，借助树状结构帕森斯估计自适应优化特征结构参数，降低了抽象特征的学习难度。在人工合成、电力负荷和电力价格数据集上的实验表明，与既有卷积与循环结构随机深度神经网络及其特征选择方法相比，基于所提方法优化的模型具有更好的预测准确性与稳定性。

最后，针对混合结构的随机深度神经网络预测模型构造与优化问题，提出了基于误差反馈生长和三阶段优化的新颖混合结构构造与优化方法，消除了既有方法的单一结构局限。通过递归添加不同结构的随机子网络，利用岭回归局部更新输出权重，构造出鲁棒的混合隐藏结构，并证明了该构造方法的收敛性；通过预优化、子训练和调正则的三阶段优化方法对迭代添加的子网络进行完整优化，使其自适应选择出多种结构的子网络及参数，融合了不同深度结构的学习能力。在人工合成、空气污染和电力负荷数据集上的实验表明，与多种既有随机深度神经网络模型相比，基于所提方法构造的模型具有更好的预测性能。

关键词：时间序列预测；深度学习；随机映射；卷积神经网络；回声状态网络

Abstract

Time series forecasting is of great importance for a learning system in dynamic environments, playing a vital role in many real-world applications, such as energy, traffic, finance, and industry. Recent studies have shown that deep learning technique has shown intriguing prediction performance, leading to extensive research on the applications of the DNN models for time series forecasting. However, represented by the convolutional neural network and recurrent neural network models, the deep neural network-based forecasting models have complex architecture-related parameters and rely on gradient-based algorithms to train the weight-related parameters, making it extremely time-consuming and challenging to well establish a forecasting model. As an alternative method of training the neural network model, the stochastic mechanism fixes the weights of the input layer and the hidden layer after random initialization, and uses a simple and direct closed-form solution algorithm to calculate the weight of the output layer of the model, which can effectively improve the construction efficiency of the neural network model.

Therefore, this study investigates time series deep learning prediction modeling technology based on the stochastic mechanism, pays attention to the structure selection problems under the specific structures, and proposes the hidden structure construction method of the stochastic convolutional neural network and the output structure selection method of the stochastic recurrent neural network. On this basis, the optimization problems under the universal structure are explored, and the feature selection method as well as the parameter optimization method of the stochastic deep neural network prediction model are proposed to further improve the prediction performance of the models. At the same time, the application research is carried out in combination with important application scenarios, such as influenza prediction, crude oil price prediction, electricity load prediction, electricity price prediction, and so on. The main contributions of this dissertation are summarized as follows:

Focusing on the model construction problem of stochastic convolutional neural network–

based forecasting model, a novel error-feedback stochastic modeling strategy and greedy-based selection algorithm are proposed to craft the random convolutional neural network for time series forecasting. The proposed method suggests that random filters and neurons of the error-feedback fully connected layer are incrementally added to steadily compensate for the prediction error during the construction process, and then a greedy-based filter selection is introduced to enable the model to extract the different sizes of temporal features. Comprehensive experiments on the simulated dataset and several real-world datasets show that the proposed method exhibits stronger predictive power and lower computing overhead compared to trained state-of-the-art deep neural network models.

Focusing on the model construction problem of stochastic recurrent neural network-based forecasting model, a novel state mask strategy with particle swarm optimization is proposed to construct the random recurrent output structure for time series forecasting. Based on the investigation of the stochastic implementation of different recurrent output structures of training-based recurrent neural networks, the proposed method adds a mask to each step of the recurrent hidden features, and then a particle swarm optimization based mask selection is introduced to evolve the mapping relationships from recurrent hidden features to their targets, which improves the learning ability of the stochastic recurrent output architecture. Compared with the stochastic recurrent neural networks with the existing output architecture selection method, comprehensive experiments on the simulated dataset, electricity load dataset, and outside temperature dataset demonstrate the superiority of the proposed method.

Focusing on the input feature selection problem of stochastic deep neural network-based forecasting model, a novel dual feature-structured selection method with tree-structured parzen estimator is proposed. Based on the ability of deep neural architecture that can model multiple dimensions in each input time step, a multiple-step-dimension two-dimensional feature structure is established with a moving window schema. The proposed method adds a mask to each input step to represent the two-dimensional feature structure with the combination of step dimension and step mask, and then tree-structured parzen estimator is introduced to evolve the feature structure, which improves the learning ability of the stochastic deep

neural networks. Compared with the stochastic deep neural networks with the existing feature selection method, comprehensive experiments on the simulated dataset, electricity load dataset, and electricity price dataset demonstrate the superiority of the proposed method.

Focusing on the model construction problem of stochastic deep neural network-based forecasting model, a novel error-feedback triple-phase optimization strategy is proposed to grow stochastic deep neural network-based predictor with mixed deep neural architectures. The proposed method incrementally adds diverse deep subnetworks to the network, where the output weights of the subnetworks are calculated via ridge regression to improve the robustness of the constructed model, and the parameters of the subnetworks are evolved with pre-tuning, sub-tuning, and reg-tuning optimization, making the network take advantage of different deep neural architectures. Compared with the existing stochastic deep neural networks, comprehensive experiments on the simulated dataset, air pollution datasets and electricity load datasets demonstrate the superiority of the proposed method.

Keywords: Time series forecasting; deep learning; stochastic mechanism; convolutional neural network; echo state network

目 录

摘 要	I
Abstract	III
1 基于卷积结构的 SDNN 预测模型构造与优化方法	
1.1 引言	(1)
1.2 ESM-CNN 预测模型	(2)
1.3 实验设计与流程	(9)
1.4 实验结果与分析	(14)
1.5 小结	(21)
2 时间序列预测建模框架	
2.1 引言	(23)
2.2 框架需求分析	(24)
2.3 框架结构设计	(26)
2.4 框架逻辑流程	(38)
2.5 小结	(40)
致 谢	(42)
参考文献	(43)
附录 1 攻读博士学位期间取得的研究成果	(48)
附录 2 已发表论文与博士学位论文的关系	(49)
附录 3 攻读博士学位期间参与的科研项目	(50)

1 基于卷积结构的 SDNN 预测模型构造与优化方法

1.1 引言

近年来,作为深度神经网络(DNN)的代表结构之一,基于卷积结构的卷积神经网络(CNN)因其优异的特征提取能力与学习能力已被广泛地应用于时间序列预测建模问题中^[1-3]。然而,如何基于时间序列数据自适应地选择模型结构与参数是应用 CNN 预测建模技术中的关键挑战。对于模型选择问题,已有众多研究者展开了相关研究。Elsken et al.^[4]对神经网络模型的既有模型选择方法进行了研究综述,总结出定义神经网络参数搜索空间、针对参数训练模型和基于训练模型结构对所搜索参数加以优化的模型选择范式。但是,这种传统的模型选择范式需要消耗大量的计算资源与训练时间,使得卷积神经网络时间序列预测建模技术的现实应用面临参数构造难、计算开销大等阻碍^[5]。

相较于有训练权重参数的 CNN 模型,采用随机映射方法构造的 CNN 模型在图像风格迁移、语音合成、函数拟合等问题中展现出充分潜力^[6-8]。这种基于随机映射的 CNN 模型具有计算开销低、收敛速度快、学习性能好等特点,但也存在网络结构设计难、随机权重输出结果不稳定等问题。针对时间序列预测问题, Yu et al.^[8]通过将 CNN 模型中的卷积权重随机初始化并固定,利用最小二乘法闭式求解输出权重,建立了随机卷积神经网络预测模型,并结合合成时间序列数据和煤气供应数据,验证了其方法的可行性。但此预测建模方法存在若干问题。首先,其卷积结构的选择依赖于人工经验,使得模型无法根据不同预测情景自适应选择合适参数;其次,输出权重全局更新的求解方式,随着卷积核数量的增加,将使卷积层的隐藏特征维度倍数增加,导致求解输出权重的最小二乘法出现病态矩阵问题,削弱了模型的稳定性,甚至使模型出现欠拟合的问题。因此,如何建立基于卷积结构的随机深度神经网络(SDNN)预测模型的新颖构造及自适应优化方法,解决已有卷积结构 SDNN 预测建模技术中参数选择时的人工依赖、输出权重求解时的病态矩阵和预测过程的不稳定性,以有效利用随机映射方法的效率优势与卷积结构的学习能力,是本章的研究重点。

为此, 本章在已有研究的基础上, 针对基于卷积结构的 SDNN 预测模型构造与自适应优化问题, 创新地提出了一种基于误差反馈随机建模 (Error-feedback stochastic modeling, ESM) 和贪心搜索选择的 CNN 构造与优化方法 (ESM-CNN)。在 ESM-CNN 这一卷积结构的 SDNN 预测模型构造与优化方法中, ESM 策略通过递归生成随机映射卷积核的方式提高了模型构造效率, 并借助误差反馈策略, 单独求解各卷积核输出权重, 保证了所构造模型的理论收敛性, 以此高效构造出具有理论收敛性的预测模型; 同时, 通过贪心算法解决模型构造中随机映射卷积核的选择问题, 使得 ESM-CNN 能够自适应的在单卷积层内具备不同卷积宽度的卷积核, 以此增强了 ESM-CNN 学习不同尺度时间序列特征的能力, 进一步地提高了模型预测性能; 最后, 采用合成数据和多个真实数据集充分验证了 ESM-CNN 的有效性以及卷积核选择方法的必要性, 与传统随机多层感知机 (SMLP) 模型和梯度下降训练 DNN 模型相比, ESM-CNN 具有优秀的预测性能与建模效率, 是解决时间序列预测问题的有效方法。

本章的主要内容包括: 1.2 节简要介绍了传统 CNN 预测模型的构造过程, 并对本章节所提出的 ESM-CNN 模型构造与优化方法进行了详细阐述; 1.3 节从数据集选取、对比模型选取、评价指标选取、实验流程和实验环境等方面介绍了本章的实验方案; 1.4 节比较、分析和总结了各模型在所选数据集上的实验结果; 最后, 1.5 节对本章予以小结。

1.2 ESM-CNN 预测模型

1.2.1 CNN 预测模型

卷积神经网络 (CNN) 是将输入连接卷积核做局部非线性变换, 并池化展开特征图所构成的神经网络^[9]。卷积是指输入的滑动局部非线性变换。通过对时间序列输入特征的卷积操作, 卷积核 (Convolutional filter) 能够基于完整输入学习出有用的非线性特征, 并建立起所学习特征与对应卷积核宽度 (Filter size) 输入的关系, 由此组成了卷积层 (Convolutional layer) 的输出特征图 (Feature map)。这些特征图往往被送入池化层 (Pooling layer) 进行下采样 (Subsample) 以降低卷积层的输出带宽 (Output band), 同时增强神经网络对异常特征图值的鲁棒性^[10-12]。池化操作后的特

征图会继续送入神经网络的下一层结构中，以学习更抽象的表示或生成输出结果。

这里，本节对一种典型结构 CNN 预测模型进行具体介绍，该模型由生成特征图的单卷积层、进行下采样的平均池化层（Pooling layer）和生成预测结果的全连接层（Fully connected layer）组成。对于由 N 个样本组成的时间序列数据 $D = \{(X_i, Y_i) \in (\mathbb{R}^T \times \mathbb{R}^H)\}_{i=1}^N$ ，其中 X_i 表示历史 T 步内的时间序列观测值 $X_i = [x_1, \dots, x_T]^T$ ， Y_i 表示预测时长 H 步内的时间序列观测值 $Y_i = [x_{T+1}, \dots, x_{T+H}]^T$ 。一个具有 C 个卷积核的 CNN 预测模型可表示为如公式 1.1，公式 1.2 和公式 1.3 所示过程：

$$m_j^t = \sigma \left(\sum_{k_m=1}^{K_m} w_j^{k_m} x_{t+k_m-1} + b_j \right), \quad t = 1, \dots, T - K_m + 1, \quad (1.1)$$

$$p_j^i = \frac{\sum_{k_p=1}^{K_p} m_j^{i+k_p-1}}{K_p}, \quad i = 1, \dots, T - K_p - K_m + 2, \quad (1.2)$$

$$f_C = \sum_{j=1}^C \left(\sum_{i=1}^{T-K+2} \beta_j^i p_j^i \right) + \beta_0. \quad (1.3)$$

公式 1.3 中， $K = K_p + K_m$ ， K_p 和 K_m 分别为池化操作与卷积操作的宽度大小。公式 1.2 中， m_j 为特征图向量 $[m_j^1, \dots, m_j^{T-K_m+1}]^T$ ，基于第 j 个卷积在输入时间序列做如公式 1.1 所示的卷积操作后得到。 $\sigma(\cdot)$ 表示卷积核所使用的激活函数。在本章节中， $\sigma(\cdot)$ 为 sigmoid 激活函数。 $w_j = [w_j^1, \dots, w_j^{K_m}]$ 和 b_j 分别为第 j 个卷积核的权重与偏置项。 p_j 表示对 m_j 做如公式 1.2 所示平均池化操作后的池化特征图向量 $[p_j^1, \dots, p_j^{T-K+2}]^T$ 。 $[\beta_1^1, \beta_1^2, \dots, \beta_C^{T-K+2}]$ 和 β_0 为第 j 个卷积核与输出层间的全连接层权重与偏置项， $\beta_j^i = [\beta_j^{i,1}, \dots, \beta_j^{i,H}]^T$ 。

在传统的 CNN 预测模型构造方法中，通过以减小预测误差 $\|e\|$ 为目标，梯度下降训练包含所有权重项与偏置项在内的权重参数，CNN 能够有效学习出历史输入时间序列与预测目标时间序列间的函数关系，生成良好的预测结果。但这种构造方法需要预先定义包含卷积核宽度、卷积核数量、梯度下降学习速率等众多超参数（Hyper-parameters），经过反复迭代训练评价后，才能选择出合适的超参与权重参数组合。此过程往往会消耗大量的计算资源和时间，存在显著的改进空间。

为解决现有 CNN 预测建模技术模型选择困难、计算开销高等问题，本章提出了一种误差反馈随机建模（ESM）构造策略，结合基于贪心算法的卷积核选择算法，

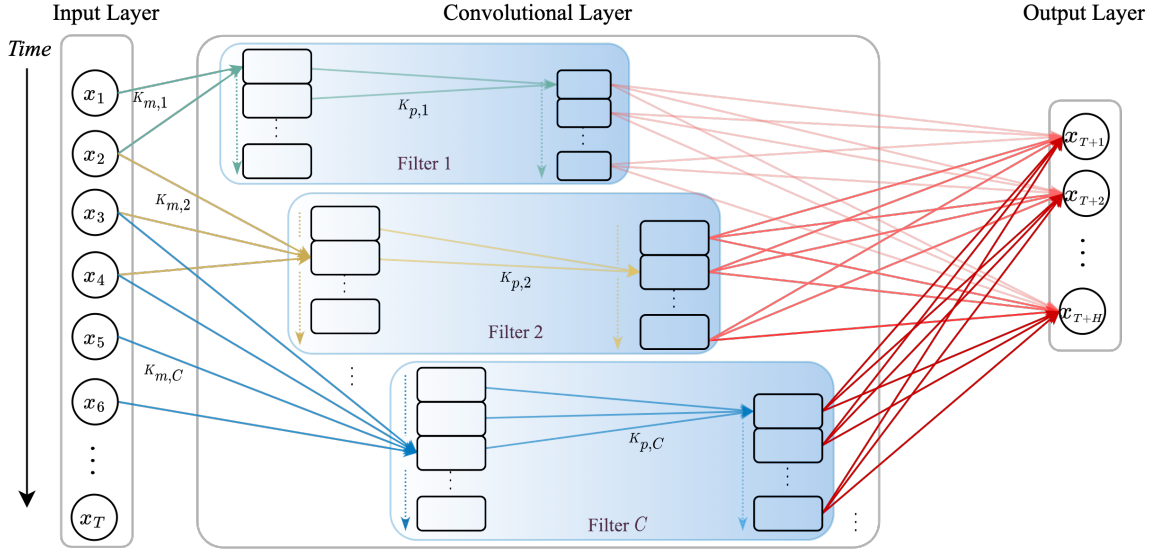


图 1.1 ESM-CNN 预测模型神经网络结构

建立 ESM-CNN 预测模型。该模型神经网络结构如图 1.1 所示，其中绿色线、黄色线和蓝色线分别表示不同结构随机卷积核的卷积与池化操作，深浅不同的红色线表示迭代局部更新的输出权重，同一深度的红色线表示链接同一卷积核的输出权重。

1.2.2 误差反馈随机建模策略

为高效构造具备可靠预测性能的卷积结构 SDNN 预测模型，利用随机映射方法的优异建模效率，同时解决随机映射算法对预测性能稳定性的干扰，本节提出了一种 ESM 策略，建立 CNN 构型下的 SDNN 预测模型。该方法通过在单卷积层内递归增长添加固定随机初始化权重卷积核的方式构造 CNN 隐藏结构，同时利用误差反馈闭式计算新增卷积核所对应的输出权重参数，以此完成整个预测模型的构造过程，同时为模型的预测函数拟合能力提供理论收敛保证。

具体地，对于一个已具有 C 个卷积核的单卷积层 ESM-CNN，其模型可被公式表述为：

$$f_C = \sum_{j=1}^C \left(\sum_{i=1}^{T-K+2} \beta_j^i p_j^i + \beta_j^0 \right), \quad (1.4)$$

并进一步简写为： $f_C = \sum_{j=1}^C \sum_{i=0}^{T-K+2} \beta_j^i p_j^i$ 。

公式 1.4 中， $\beta_j = [\beta_j^0, \beta_j^1, \dots, \beta_j^{T-K+2}]$ 表示全连接输出层中连接第 j 个卷积核池

化特征图向量与输出单元的权重参数, $p_j^0 = I$ 。ESM-CNN 的预测误差如公式 1.5 所示:

$$e_C = Y - f_C = [e_C^1, \dots, e_C^H]. \quad (1.5)$$

若预测模型的均方误差 (Mean square error, MSE) 未降低至容忍水平 ϱ , ESM 策略将继续新增一个随机映射卷积核 m_{C+1} 生成对应的池化特征图向量 p_{C+1} , 并将 p_{C+1} 全连接至输出层中以加宽模型隐藏结构。其中, 新增卷积核所对应的全连接输出层权重将基于当前模型的预测误差 e_C 反馈, 通过如公式 1.6 最小二乘法加以求解:

$$[\beta_{C+1}^0, \dots, \beta_{C+1}^{T-K+2}] = \underset{\beta}{\operatorname{argmin}} \|e_C - \sum_{i=0}^{T-K+2} \beta_{C+1}^i p_{C+1}^i\|. \quad (1.6)$$

通过此策略, 所构造的 ESM-CNN 预测模型能够具备随着卷积核数量增加而单调下降并收敛的预测误差, 从而完成对预测目标函数的逼近。该性质通过如下代数推导予以证明:

证明: 定义 ESM-CNN 的预测误差中间项为: $\tilde{e}_{C+1}^0, \dots, \tilde{e}_{C+1}^{T-K+2}$, 新增卷积核所对应的全连接输出层权重中间项为: $\tilde{\beta}_{C+1}^0, \dots, \tilde{\beta}_{C+1}^{T-K+2}$, 中间项之间的计算关系为:

$$\tilde{e}_{C+1}^{i+1} = \tilde{e}_{C+1}^i - \tilde{\beta}_{C+1}^{i+1} p_{C+1}^{i+1}, \quad i = 0, \dots, T-K+1,$$

其中,

$$\begin{aligned} \tilde{\beta}_{C+1}^{i+1} &= [\tilde{\beta}_{C+1}^{i+1,1}, \dots, \tilde{\beta}_{C+1}^{i+1,h}, \dots, \tilde{\beta}_{C+1}^{i+1,H}], \\ \tilde{\beta}_{C+1}^{i+1,h} &= \langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle / \|p_{C+1}^{i+1}\|^2, \quad h = 1, \dots, H, \end{aligned}$$

同时,

$$\begin{aligned} \tilde{e}_{C+1}^0 &= e_C - \tilde{\beta}_{C+1}^0 p_{C+1}^0, \\ \tilde{\beta}_{C+1}^0 &= [\tilde{\beta}_{C+1}^{0,1}, \dots, \tilde{\beta}_{C+1}^{0,h}, \dots, \tilde{\beta}_{C+1}^{0,H}], \\ \tilde{\beta}_{C+1}^{0,h} &= \langle e_C^h, p_{C+1}^0 \rangle / \|p_{C+1}^0\|^2, \quad h = 1, \dots, H, \\ \tilde{e}_{C+1}^{T-K+2} &= e_C - \sum_{i=0}^{T-K+2} \tilde{\beta}_{C+1}^i p_{C+1}^i. \end{aligned}$$

因新增卷积核所对应的全连接输出层权重基于最小二乘法予以求解:

$$[\beta_{C+1}^0, \dots, \beta_{C+1}^{T-K+2}] = \underset{\beta}{\operatorname{argmin}} \|e_C - \sum_{i=0}^{T-K+2} \beta_{C+1}^i p_{C+1}^i\|,$$

则 $\|e_{C+1}\|^2$ 与 $\|\tilde{e}_{C+1}^{T-K+2}\|^2$ 存有不等式关系:

$$\|e_{C+1}\|^2 = \min_{\beta} \|e_C - \sum_{i=0}^{T-K+2} \beta_{C+1}^i p_{C+1}^i\|^2 \leq \|e_C - \sum_{i=0}^{T-K+2} \tilde{\beta}_{C+1}^i p_{C+1}^i\|^2 = \|\tilde{e}_{C+1}^{T-K+2}\|^2.$$

同时, 预测误差中间项之前存在着单调下降性质:

$$\begin{aligned} & \|\tilde{e}_{C+1}^{i+1}\|^2 - \|\tilde{e}_{C+1}^i\|^2 \\ &= \sum_{h=1}^H \left(\langle \tilde{e}_{C+1}^{i,h} - \tilde{\beta}_{C+1}^{i+1,h} p_{C+1}^{i+1}, \tilde{e}_{C+1}^{i,h} - \tilde{\beta}_{C+1}^{i+1,h} p_{C+1}^{i+1} \rangle - \langle \tilde{e}_{C+1}^{i,h}, \tilde{e}_{C+1}^{i,h} \rangle \right) \\ &= \sum_{h=1}^H \left(\langle \tilde{\beta}_{C+1}^{i+1} p_{C+1}^{i+1}, \tilde{\beta}_{C+1}^{i+1} p_{C+1}^{i+1} \rangle - 2 \langle \tilde{e}_{C+1}^{i,h}, \tilde{\beta}_{C+1}^{i+1} p_{C+1}^{i+1} \rangle \right) \\ &= \sum_{h=1}^H \left(\left\langle \frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle}{\|p_{C+1}^{i+1}\|^2} p_{C+1}^{i+1}, \frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle}{\|p_{C+1}^{i+1}\|^2} p_{C+1}^{i+1} \right\rangle - 2 \langle \tilde{e}_{C+1}^{i,h}, \frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle}{\|p_{C+1}^{i+1}\|^2} p_{C+1}^{i+1} \rangle \right) \\ &= \sum_{h=1}^H \left(\frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle^2}{\|p_{C+1}^{i+1}\|^4} \langle p_{C+1}^{i+1}, p_{C+1}^{i+1} \rangle - 2 \frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle}{\|p_{C+1}^{i+1}\|^2} \langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle \right) \\ &= \sum_{h=1}^H \left(\frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle^2}{\|p_{C+1}^{i+1}\|^4} \|p_{C+1}^{i+1}\|^2 - 2 \frac{\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle^2}{\|p_{C+1}^{i+1}\|^2} \right) \\ &= \sum_{h=1}^H \left(-\langle \tilde{e}_{C+1}^{i,h}, p_{C+1}^{i+1} \rangle^2 / \|p_{C+1}^{i+1}\|^2 \right) \\ &\leq 0. \end{aligned}$$

该性质在 $\|\tilde{e}_{C+1}^0\|^2$ 与 $\|e_C\|^2$ 间仍然保持:

$$\begin{aligned} & \|\tilde{e}_{C+1}^0\|^2 - \|e_C\|^2 \\ &= \sum_{h=1}^H \left(\langle e_C^h - \tilde{\beta}_{C+1}^{0,h} p_{C+1}^0, e_C^h - \tilde{\beta}_{C+1}^{0,h} p_{C+1}^0 \rangle - \langle e_C^h, e_C^h \rangle \right) \\ &= \sum_{h=1}^H \left(\langle \tilde{\beta}_{C+1}^{0,h} p_{C+1}^0, \tilde{\beta}_{C+1}^{0,h} p_{C+1}^0 \rangle - 2 \langle e_C^h, \tilde{\beta}_{C+1}^{0,h} p_{C+1}^0 \rangle \right) \\ &= \sum_{h=1}^H \left(\left\langle \frac{\langle e_C^h, p_{C+1}^0 \rangle}{\|p_{C+1}^0\|^2} p_{C+1}^0, \frac{\langle e_C^h, p_{C+1}^0 \rangle}{\|p_{C+1}^0\|^2} p_{C+1}^0 \right\rangle - 2 \langle e_C^h, \frac{\langle e_C^h, p_{C+1}^0 \rangle}{\|p_{C+1}^0\|^2} p_{C+1}^0 \rangle \right) \\ &= \sum_{h=1}^H \left(\frac{\langle e_C^h, p_{C+1}^0 \rangle^2}{\|p_{C+1}^0\|^4} \langle p_{C+1}^0, p_{C+1}^0 \rangle - 2 \frac{\langle e_C^h, p_{C+1}^0 \rangle}{\|p_{C+1}^0\|^2} \langle e_C^h, p_{C+1}^0 \rangle \right) \end{aligned}$$

$$\begin{aligned}
 &= \sum_{h=1}^H \left(\frac{\langle e_C^h, p_{C+1}^0 \rangle^2}{\|p_{C+1}^0\|^4} \|p_{C+1}^0\|^2 - 2 \frac{\langle e_C^h, p_{C+1}^0 \rangle^2}{\|p_{C+1}^0\|^2} \right) \\
 &= \sum_{h=1}^H \left(-\langle e_C^h, p_{C+1}^0 \rangle^2 / \|p_{C+1}^0\|^2 \right) \\
 &\leq 0.
 \end{aligned}$$

因此，ESM-CNN 的预测误差收敛性可证：

$$\|e_{C+1}\|^2 \leq \|\tilde{e}_{C+1}^{T-K+2}\|^2 \leq \|\tilde{e}_{C+1}^0\|^2 \leq \|e_C\|^2.$$

与全局更新输出层权重的构造方式不同，本章所提出的 ESM 构造策略通过误差反馈迭代局部更新输出层权重的方式完成输出层权重的计算。具体地，ESM 构造策略具备以下三种优势：1) 与全局更新输出层权重相比，迭代局部更新输出层权重的方式具有更小的计算开销，这种低开销优势会随隐藏层结构的增大同时扩大；2) 随着卷积层中递归新增卷积核的不断加入，链接至输出层的池化特征图向量维度会同时倍数增大，与预测目标维度相比过大的特征维度会导致闭式求解算法的病态问题 (Ill-posed problem)^[13]，进而破坏预测性能的稳定性和鲁棒性；3) 基于误差反馈递归更新输出权重构造的预测模型能在神经网络隐藏构造的过程中不断弥补上步预测误差，同时，这种历史输出权重保持固定的方式在一定程度上承担了正则化的作用，从而增强了所构造模型的性能稳定性。

1.2.3 卷积参数贪心选择方法

针对 ESM-CNN 预测模型构造过程中的卷积核选择问题，本章节提出了一种基于贪心算法的卷积核选择方法，使得 ESM-CNN 具有在单卷积层中同时具备不同宽度的卷积核结构，以此增强模型对于不同尺度时间序列特征的学习建模能力。

对于如公式 1.4 所示具有 C 个卷积核的单卷积层 ESM-CNN，在新增卷积核结构时，基于均匀分布 (Uniform distribution) $[-\lambda, \lambda]$ 、卷积与池化宽度 $\{(K_{m,s}, K_{p,s})\}_{s=1}^S$ 生成一批对应的池化特征图向量 $\{p_{C+1,s}\}_{s=1}^S$ 表示对应的备选卷积核结构。为评价这些备选卷积核结构，利用将备选卷积核加入当前网络中所能下降的预测误差，本节提出了一种卷积核评分 $\Delta_{C+1,s}$ ，其计算方式如公式 1.7 所示：

$$\Delta_{C+1,s} = \|e_{C+1,s}\|^2 - \|e_C\|^2 \quad (1.7)$$

$$= \|e_C - \sum_{i=0}^{T-K'+2} \beta_{C+1,s}^i p_{C+1,s}^i\|^2 - \|e_C\|^2,$$

公式 1.7 中, $K' = K_{m,s} + K_{p,s}$, $p_{C+1,s}^i$ 和 $\beta_{C+1,s}^i$ 分别由公式 1.2 和公式 1.6 计算得出。基于所提出的卷积核评分 $\Delta_{C+1,s}$, 将实现最好预测效果提升的备选卷积核 p_{C+1}^* 选出, 作为新增卷积核正式加入当前神经网络结构, 该过程如公式 1.8 所示:

$$p_{C+1}^* = \underset{p_{C+1,s}}{\operatorname{argmax}} \{\Delta_{C+1,s}, s = 1, \dots, S\}. \quad (1.8)$$

在此之后, 新增卷积核所对应的全连接输出层权重通过 1.6 求解得出。结合本章节所提出的 ESM 构造策略与卷积核选择方法, 高效构造出一种具备多尺度时间序列特征学习能力、预测函数拟合收敛性和稳定预测性能的 ESM-CNN 预测模型。

1.2.4 ESM-CNN 构造过程

如图 1.2 所示, ESM-CNN 预测模型的构造过程主要包含: 预处理、初始化、评价和更新四个环节。具体地, 所述模型及构造过程为:

(1) 对于时间序列数据, 构造训练数据集 $D = \{(X_i, Y_i) \in (\mathbb{R}^T \times \mathbb{R}^H)\}_{i=1}^N$, 并将数据集 D 归一化至标准正态分布。其中, N 为数据集 D 中的总样本数, 数据集中样本 (X_i, Y_i) 由包含历史 T 时刻内的时间序列数据 $[x_1, \dots, x_T]^T$ 和未来 H 步时刻内的时间序列预测目标 $[x_{T+1}, \dots, x_{T+H}]^T$ 组成。

(2) 设定 ESM-CNN 预测模型的初始化参数, 包括容忍水平 ϱ , 最大特征图通道数 C_{\max} , 初始预测误差 $e_0 = [x_{T+1}, \dots, x_{T+H}]^T$, 卷积核宽度 K_m 选择范围 $M = \{K_{m,s}\}_{s=1}^S$ 与池化操作宽度挑选范围 $P = \{K_{p,s}\}_{s=1}^S$, 以及卷积核权重的随机初始化分布 $[-\lambda, \lambda]$ 。

(3) 在终止条件为特征图通道数 $C \leq C_{\max}$ 或当前模型预测结果的 MSE 小于 ϱ 的构造过程中, 基于卷积核宽度选择范围 M 与池化操作宽度选择范围 P , 依次从参数组合 $(K_{m,s}, K_{p,s})$ 和均匀分布 $[-\lambda, \lambda]$ 中生成 S 个备选卷积核, 以此构成卷积核备选集, 并根据得到的备选集, 计算各备选卷积核下的池化特征图向量 $\{p_{C+1,s}\}_{s=1}^S$ 。

(4) 基于公式 1.7 计算各备选卷积核的评分, 从中贪心选出最优的卷积核结构及对应卷积核权重, 将其加入当前神经网络结构中, 更新卷积层卷积核数量和模型预测误差。迭代此过程, 直至满足终止条件。

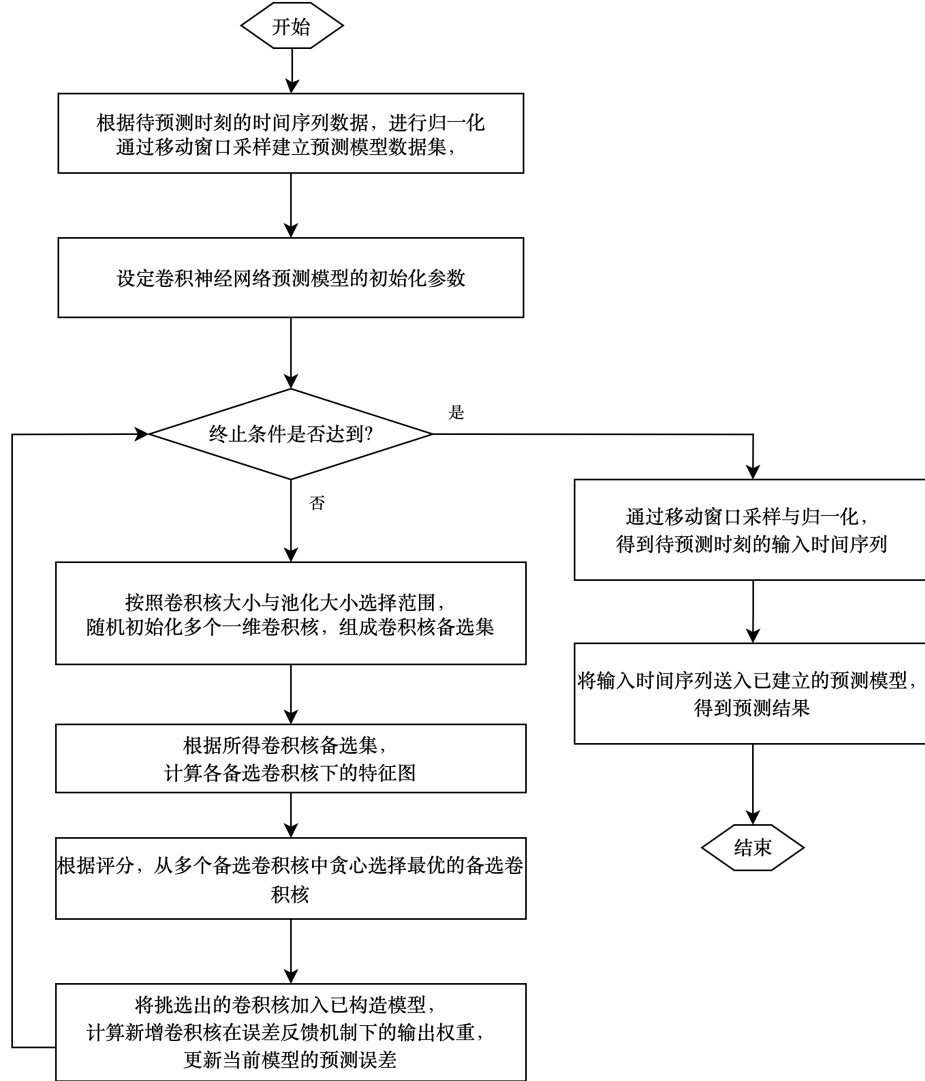


图 1.2 ESM-CNN 预测模型构造过程

1.3 实验设计与流程

1.3.1 数据集描述

为充分评估所提 ESM-CNN 预测模型的预测性能和验证其收敛能力，本章节在经典人工合成时间序列数据集和多项真实时间序列数据集上进行实验验证研究。实验数据集名称及简写如表 1.1 所示。表 1.2 展示了本章节所用数据集的统计信息。其中，第一列、第五列和第六列说明了所用数据集的名称，起始截止日期和大小。第二列展示所用数据集的平稳性（Stationarity）状态，“✓”表示该数据集为平稳的，“✗”

表 1.1 实验数据集名称及英文缩写对照

数据集中文名称	数据集英文名称	英文缩写
一阶自回归	First order autoregression	AR1
比特币两小时美元价格	Two-hourly bitcoin to U.S. dollar price	BTC
流感阳性样本率	Influenza-like illness rate	ILI
欧洲布伦特原油周价格	Weekly Europe Brent crude oil price	BRENT-weekly
欧洲布伦特原油周价格	Daily Europe Brent crude oil price	BRENT-daily
美国 WTI 原油周间价格	Weekly WTI crude oil price	WTI-weekly
标普 500 每日收盘指数	Daily close value of S&P 500 index	S&P 500
纳斯达克每日收盘指数	Daily close value of NASDAQ Composite	NASDAQ
道琼斯每日收盘指数	Daily close value of Dow Jones Average	DJI
纽约证交所每日收盘指数	Daily close value of NYSE Composite	NYSE

表 1.2 数据集信息

数据集名称	平稳性	趋势性	季节性	起始截止日期	数据集大小
AR1	✗	0.97	0.09	-	500
BTC	✗	0.99	0.66	05/25/2020 ~ 03/20/2021	2181
ILI	✓	0.51	0.61	01/15/2010 ~ 04/15/2020	535
BRENT-weekly	✗	0.97	0.07	05/15/1987 ~ 04/30/2021	1773
BRENT-daily	✗	0.97	0.06	05/20/1987 ~ 05/03/2021	8620
WTI-weekly	✗	0.96	0.08	01/03/1986 ~ 04/30/2021	1844
WTI-daily	✗	0.96	0.07	01/02/1986 ~ 05/03/2021	8904
S&P 500	✗	0.99	0.40	12/31/2009 ~ 11/15/2017	1984
NASDAQ	✗	0.99	0.27	12/31/2009 ~ 11/15/2017	1984
DJI	✗	0.99	0.40	12/31/2009 ~ 11/15/2017	1984
NYSE	✗	0.98	0.45	12/31/2009 ~ 11/15/2017	1984

表示该数据集为非平稳的，该状态通过扩张的迪基-福勒检验（Augmented Dickey Fuller, ADF）^①判断得出。第三列与第四列则分别表示了数据集的趋势性（Trend）与季节性（Seasonality）强度分数^②，分数越高表明该数据具备越强的趋势性或季节性。各数据集的说明如下：

^①<https://www.statsmodels.org/stable/generated/statsmodels.tsa.stattools.adfuller.html>

^②<https://otexts.com/fpp2/seasonal-strength.html>

针对合成数据上的预测实验研究, AR1 数据集作为评价时间序列预测模型的常用基准数据集之一, 参照相关研究^[14,15]的参数设定通过公式 1.9 予以合成:

$$x_t = \alpha + x_{t-1} + \varepsilon_t, \quad (1.9)$$

其中, $\alpha = 0.01$, $\varepsilon_t \sim i.i.d. U(-0.25, 0.25)$, 共合成出 500 个观测点。

针对比特币价格预测应用研究, 本章节所用 BTC 数据集通过 Tradeviw 平台^③采集 2020 年 5 月 25 日至 2021 年 3 月 20 日间的两小时级美元收盘价得到, 共有 2181 个观测点。

针对流感阳性样本率预测应用研究, 本章节所用 ILI 数据集通过中国国家流感中心平台^④采集 2010 年 1 月 15 日至 2020 年 4 月 15 日间的中国南方地区每周流感阳性样本率得到, 共有 535 个观测点。

针对原油价格预测应用研究, 本章节分别通过采集欧洲 BRENT 市场^⑤和美国 WTI 市场^⑥公开至 2021 年 5 月 3 日间的每周美元收盘价格及每日美元收盘价格, 得到 BRENT-weekly (1773 个观测点)、BRENT-daily (8620 个观测点)、WTI-weekly (1844 个观测点) 和 WTI-daily (8904 个观测点) 数据。

针对股票指数预测应用研究, 本章节通过 UCI 机器学习数据集^⑦得到 2009 年 12 月 31 日至 2017 年 11 月 15 日间的 S&P 500、NASDAQ、DJI 和 NYSE 每日收盘指数, 各股票指数数据均有 1984 个观测点。

在本章节进行的实验中, 所选数据集对于训练集、验证集和测试集切分遵循机器学习预测建模研究中常用的 0.64、0.16 和 0.2 的比例。为进一步分析和比较所提预测模型在提前单步 (One-step-ahead) 预测与提前多步 (Multi-step-ahead) 预测上的性能表现, 本章节在各数据集上均进行了不同提前时步的预测建模实验。具体地, AR1 数据集上的预测时长 H 分别为 1、3、6, 输入步长 T 为 15; BTC 数据集上的预测时长 H 分别为 1、3 (六小时)、6 (半日), 输入步长 T 为 24 (两日); ILI、BRENT-weekly 和 WTI-weekly 数据集上的预测时长 H 分别为 1、4 (一个月)、8 (两

^③<https://www.tradingview.com/symbols/BTCUSD>

^④<https://ivdc.chinacdc.cn/cnic/en/Surveillance/WeeklyReport/>

^⑤<https://www.eia.gov/dnav/pet/hist/RBRTed.htm>

^⑥<https://www.eia.gov/dnav/pet/hist/rwtcW.htm>

^⑦<https://archive.ics.uci.edu/ml/machine-learning-databases/00554/>

个月), 输入步长 T 为 26 (半年); BRENT-daily、WTI-daily、S&P 500、NASDAQ、DJI 和 NYSE 数据集上的预测时长 H 分别为 1、5 (一周)、10 (两周), 输入步长 T 为 30 (六周)。

1.3.2 对比模型选择

为比较 ESM-CNN 预测模型与传统统计预测模型的性能差异, 将输入时间序列最后时刻观测值作为预测值的 Naive 预测方法、自回归移动平均 (Autoregressive Integrated Moving Average, ARIMA) 方法^[16] 和 Holt's Winters 季节指数平滑 (Holt's Winters Seasonal Exponential Smoothing, HWSES) 方法^[17] 被选为对照组模型以验证所提模型的性能优势。

为比较 ESM-CNN 预测模型与经典随机映射预测模型的性能差异, RVFL^[18]、ILEM^[19] 和 SCN^[20] 这三种典型随机多层感知机 (SMLP) 被选为对照组模型。RVFL、ILEM 和 SCN 的随机映射参数 (如随机初始化隐藏层权重的分布参数、容忍水平等) 与所提 ESM-CNN 模型参数保持一致, 同时 RVFL、ILEM 和 SCN 中的隐藏层神经元数量与 ESM-CNN 中卷积核数量保持一致以更好地控制实验变量。

此外, 为比较 ESM-CNN 预测模型与梯度下降训练方法下 DNN 预测模型的性能差异, CNN、循环神经网络 (Recurrent neural network, RNN) 以及结合 CNN 与 RNN 优势的卷积循环神经网络 (Convolutional recurrent neural network, CRNN) 这三种代表性神经网络被加入对照组模型中。具体地, 本章节提供了一种网格搜索 (Grid-search) 确定隐藏层卷积核数量的单卷积层 CNN (GS-CNN) 作为 CNN 的实现模型。针对 RNN 结构, 本章节复现了 Salinas 等^[21] 提出的一种基于自回归长短时记忆 (Long short term memory, LSTM) 网络结构的 RNN 预测模型 DeepAR, 该模型在多项现实预测任务中表现出优异性能。此外, 本章节复现了 Livieris 等^[22] 基于 CNN 与 RNN 结构提出的卷积长短时记忆 (Convolutional long short term memory, CLSTM) 网络, 作为 CRNN 结构的有力实现。GS-CNN, DeepAR, 和 CLSTM 预测模型的超参数 (如梯度下降优化算法、学习速率和训练迭代次数等) 基于通行的试错 (Trial and error) 策略予以确定。

进一步地, 为验证所提 ESM-CNN 预测模型中误差反馈构造策略与卷积核结构

选择方法的必要性，本章节针对这两个要素进行了消融实验研究。通过移除卷积核选择方法保留误差反馈随机建模策略，建立一种消融的误差反馈随机卷积神经网络（Error-feedback stochastic convolutional neural network, ES-CNN）。通过完全移除误差反馈策略与卷积核选择方法，构造出 Yu et al.^[8] 所提出的单纯随机卷积神经网络（Stochastic convolutional neural network, Stoc-CNN）。在本章节所设置的所有预测任务中，ESM-CNN 的最大卷积核数量 C_{\max} 设为 100 以保证模型收敛。基于实验经验，卷积核权重的随机初始化分布 $[-\lambda, \lambda]$ 设为 $[-0.5, 0.5]$ ，备选卷积核的宽度选择范围设为 $\{K_{m,s}\}_{s=1}^S = \{T/3, T/4, T/5, T/6\}$ ，为简化卷积核结构搜索问题，参考相关研究^[11]，池化操作宽度固定为 $K_p = 3$ 。同一宽度的备选卷积核数量为 30。为保证消融实验的公平性，Stoc-CNN、ES-CNN 和 ESM-CNN 的随机设置参数与卷积核数量均保持一致。

1.3.3 评价指标设置

为多角度评价所提模型与进行对照实验研究，本章节选取了包含百分比误差与绝对值误差在内的三种常用预测准确度评价指标^[23-25]，分别为：平均绝对百分比误差（Mean absolute percentage error, MAPE），对称平均绝对百分比误差（Symmetric mean absolute percentage error, SMAPE）和根方平均平方误差（Root mean square error, RMSE）。其计算过程分别如公式 1.10、公式 1.11 和公式 1.12 所示：

$$MAPE = \frac{1}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right|. \quad (1.10)$$

$$SMAPE = \frac{1}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i + \hat{y}_i} \right|. \quad (1.11)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}. \quad (1.12)$$

其中， y_i 和 \hat{y}_i 分别表示真实值与预测值。

1.3.4 实验步骤

在本章节的实验验证研究中，所选数据集均首先基于 Z-score 方法[®]归一化至标准正太分布，而后按照 0.64，0.16 和 0.2 的比例切分为训练集、验证集和测试集。所

[®]<https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-scaler>

提 ESM-CNN 及所有对照组和消融模型均通过训练集和验证集进行训练与交叉验证。在生成预测结果计算评价指标时, 预测生成结果均先被还原至原有数值范围, 再进行预测准确度计算。

最终, 在所有预测任务中, 本章节均对每一预测模型建模 20 次, 对 20 次预测建模结果结合 MAPE、SMAPE 和 RMSE 指标计算预测准确度平均值, 以综合表现预测模型预测准确性与稳定性。本章节所有实验均基于 CUDA 10.1 版本的 GPU 加速 Pytorch 框架^[26]、Ubuntu 20.04 系统环境、Intel 8700K CPU 和 Nvidia GTX 1070 GPU 环境进行, 以保证计算环境的一致性。本章节的实验设置与算法代码已开源在 Github 平台[®], 以方便研究者复现和开展相关工作。

1.4 实验结果与分析

1.4.1 预测准确度对比与分析

表 1.3 – 1.5 分别展示了 ESM-CNN 及对照组模型在 AR1、BTC 和 ILI 数据集上的 MAPE、SMAPE 和 RMSE 平均结果及标准差, 以评价不同模型的准确度与稳定性。在表 1.3 – 1.5 中, 括号内为模型结果的标准差, 越低的标准差表示模型具有越强的预测稳定性。其中, 统计模型的 0 标准差以 “-” 表示。每一预测任务中的最优结果通过粗体加星 “*” 的形式进行展示, 次优结果通过粗体加以表示。另外, 表 1.6 – 1.8 分别展示了 ESM-CNN 及对照组模型在原油价格与股票指数数据集上的 MAPE、SMAPE 和 RMSE 结果, 以补充说明所选模型的对比效果。

基于表 1.3 – 1.8 所示的对照组模型结果, 可归纳出以下结论:

(1) 作为一种新颖的卷积结构 SDNN 模型, ESM-CNN 在本章节选取的人工合成时间序列数据集与真实时间序列数据集上都保持了优秀的预测准确度, 展现出 ESM-CNN 预测模型的良好应用性。

(2) 与 Naive、ARIMA 和 HWSES 所代表的统计模型相比, ESM-CNN 在本章节设置的所有预测任务和所有评价指标上都表现出更优的预测准确度, 展现出 ESM-CNN 预测模型的良好预测性能。

(3) 与 GS-CNN、DeepAR 和 CLSTM 所代表的先进梯度下降训练 DNN 预测模型

[®]<https://github.com/XinzeZhang/TimeSeriesForecasting-torch>

表 1.3 ESM-CNN 及对照组模型在 AR1、BTC 和 ILI 数据集上的 MAPE 结果

数据集	H	统计模型			梯度下降模型			随机映射模型			消融模型		所提模型
		Naive	ARIMA	HWSES	GS-CNN	DeepAR	CLSTM	RVFL	IELM	SCN	Stoc-CNN	ES-CNN	ESM-CNN
AR1	1	2.54e-01	5.95e-01	5.94e-01	1.06e-01	1.08e-01	8.88e-02	7.61e-01	9.84e-02	6.50e-02	6.10e+01	4.53e-02	3.49e-02*
		(-)	(-)	(-)	(7.38e-03)	(2.26e-02)	(1.42e-02)	(4.74e-01)	(1.47e-02)	(2.41e-02)	(2.52e+01)	(5.81e-03)	(5.01e-04)
	3	2.62e-01	6.00e-01	5.98e-01	1.23e-01	1.64e-01	1.20e-01	1.17e+00	1.07e-01	9.53e-02	6.34e+01	6.41e-02	5.28e-02*
		(-)	(-)	(-)	(7.15e-03)	(3.46e-02)	(1.34e-02)	(5.85e-01)	(1.22e-02)	(1.97e-02)	(2.13e+01)	(5.21e-03)	(1.77e-03)
	6	2.68e-01	6.08e-01	6.05e-01	1.48e-01	2.17e-01	1.24e-01	1.23e+00	1.26e-01	1.16e-01	1.54e+02	8.92e-02	7.70e-02*
		(-)	(-)	(-)	(6.41e-03)	(3.72e-02)	(1.84e-02)	(7.78e-01)	(1.74e-02)	(2.63e-02)	(8.02e+01)	(5.23e-03)	(2.32e-03)
BTC	1	2.55e-01	3.17e-01	3.17e-01	6.27e-02	2.26e-01	4.09e-01	1.20e+00	1.85e-01	1.11e-01	2.14e+02	4.62e-02	2.70e-02*
		(-)	(-)	(-)	(6.37e-03)	(3.09e-02)	(2.34e-02)	(7.24e-01)	(2.98e-02)	(2.45e-02)	(1.90e+02)	(1.82e-02)	(4.98e-04)
	3	2.54e-01	3.18e-01	3.18e-01	6.43e-02	1.98e-01	4.41e-01	1.13e+00	1.84e-01	1.29e-01	2.86e+02	4.73e-02	2.96e-02*
		(-)	(-)	(-)	(6.63e-03)	(3.80e-02)	(2.44e-02)	(4.82e-01)	(2.94e-02)	(2.62e-02)	(2.33e+02)	(1.76e-02)	(3.67e-04)
	6	2.53e-01	3.21e-01	3.20e-01	6.49e-02	2.35e-01	4.61e-01	1.38e+00	1.84e-01	1.40e-01	3.32e+02	4.84e-02	3.26e-02*
		(-)	(-)	(-)	(6.28e-03)	(4.20e-02)	(9.60e-03)	(6.57e-01)	(3.06e-02)	(2.54e-02)	(3.00e+02)	(1.68e-02)	(2.65e-04)
ILI	1	1.94e-01	5.70e-01	5.94e-01	1.28e-01	1.14e-01	9.50e-02	3.03e-01	1.38e-01	1.15e-01	2.48e+00	1.00e-01	8.93e-02*
		(-)	(-)	(-)	(2.30e-03)	(8.92e-03)	(2.64e-03)	(5.17e-02)	(8.14e-03)	(9.50e-03)	(3.72e-01)	(9.59e-03)	(3.10e-03)
	4	2.51e-01	5.39e-01	5.90e-01	1.52e-01	1.59e-01	1.26e-01	3.53e-01	1.58e-01	1.53e-01	2.33e+00	1.33e-01	1.25e-01*
		(-)	(-)	(-)	(1.27e-03)	(7.46e-03)	(1.89e-03)	(3.88e-02)	(5.43e-03)	(9.04e-03)	(2.25e-01)	(4.64e-03)	(3.95e-03)
	8	3.07e-01	5.34e-01	5.98e-01	1.70e-01	1.82e-01	1.45e-01*	3.67e-01	1.74e-01	1.86e-01	2.38e+00	1.62e-01	1.53e-01
		(-)	(-)	(-)	(7.76e-04)	(8.70e-03)	(3.17e-03)	(4.19e-02)	(4.81e-03)	(1.11e-02)	(3.15e-01)	(4.07e-03)	(3.33e-03)

表 1.4 ESM-CNN 及对照组模型在 AR1、BTC 和 ILI 数据集上的 SMAPE 结果

数据集	H	统计模型			梯度下降模型			随机映射模型			消融模型		所提模型
		Naive	ARIMA	HWSES	GS-CNN	DeepAR	CLSTM	RVFL	IELM	SCN	Stoc-CNN	ES-CNN	ESM-CNN
AR1	1	1.33e-01	4.31e-01	4.30e-01	5.74e-02	5.81e-02	4.74e-02	3.02e+00	5.30e-02	3.40e-02	1.21e+00	2.33e-02	1.76e-02*
		(-)	(-)	(-)	(4.33e-03)	(1.28e-02)	(8.42e-03)	(6.74e+00)	(8.80e-03)	(1.35e-02)	(5.13e-01)	(3.16e-03)	(3.11e-04)
	3	1.37e-01	4.36e-01	4.34e-01	6.73e-02	9.17e-02	6.61e-02	8.61e+00	5.76e-02	5.11e-02	1.50e+00	3.36e-02	2.72e-02*
		(-)	(-)	(-)	(4.23e-03)	(2.10e-02)	(8.15e-03)	(1.07e+01)	(7.44e-03)	(1.15e-02)	(1.36e+00)	(2.89e-03)	(1.04e-03)
	6	1.46e-01	4.43e-01	4.40e-01	8.28e-02	1.25e-01	6.82e-02	6.36e+00	6.97e-02	6.28e-02	1.52e+00	4.78e-02	4.05e-02*
		(-)	(-)	(-)	(3.84e-03)	(2.41e-02)	(1.17e-02)	(6.59e+00)	(1.06e-02)	(1.62e-02)	(1.73e+00)	(3.03e-03)	(1.53e-03)
BTC	1	1.35e-01	1.90e-01	1.90e-01	3.26e-02	1.29e-01	2.60e-01	5.04e+00	1.04e-01	5.93e-02	1.12e+00	2.37e-02	1.35e-02*
		(-)	(-)	(-)	(3.49e-03)	(2.02e-02)	(1.87e-02)	(7.64e+00)	(1.87e-02)	(1.44e-02)	(2.20e-01)	(9.73e-03)	(2.90e-04)
	3	1.35e-01	1.91e-01	1.90e-01	3.35e-02	1.12e-01	2.87e-01	2.85e+00	1.04e-01	7.05e-02	1.09e+00	2.43e-02	1.48e-02*
		(-)	(-)	(-)	(3.65e-03)	(2.38e-02)	(1.97e-02)	(4.01e+00)	(1.85e-02)	(1.54e-02)	(1.26e-01)	(9.39e-03)	(2.10e-04)
	6	1.36e-01	1.93e-01	1.92e-01	3.38e-02	1.36e-01	3.03e-01	2.69e+00	1.04e-01	7.68e-02	1.05e+00	2.48e-02	1.63e-02*
		(-)	(-)	(-)	(3.47e-03)	(2.76e-02)	(8.17e-03)	(4.68e+00)	(1.92e-02)	(1.52e-02)	(6.03e-02)	(8.99e-03)	(1.44e-04)
ILI	1	9.82e-02	4.16e-01	4.47e-01	6.91e-02	6.02e-02	4.86e-02	1.28e-01	7.49e-02	6.06e-02	9.65e-01	5.20e-02	4.60e-02*
		(-)	(-)	(-)	(1.48e-03)	(4.37e-03)	(1.52e-03)	(1.73e-02)	(4.74e-03)	(5.13e-03)	(5.61e-01)	(5.09e-03)	(1.42e-03)
	4	1.22e-01	3.86e-01	4.44e-01	8.40e-02	8.43e-02	6.71e-02	1.64e-01	8.83e-02	8.24e-02	1.39e+00	7.13e-02	6.65e-02*
		(-)	(-)	(-)	(7.92e-04)	(3.34e-03)	(1.09e-03)	(1.67e-02)	(3.64e-03)	(5.04e-03)	(4.77e-01)	(2.42e-03)	(1.97e-03)
	8	1.55e-01	3.80e-01	4.54e-01	9.79e-02	1.00e-01	8.11e-02*	2.55e-01	1.01e-01	1.04e-01	1.60e+00	9.07e-02	8.52e-02
		(-)	(-)	(-)	(5.25e-04)	(3.68e-03)	(1.99e-03)	(1.63e-01)	(3.15e-03)	(6.38e-03)	(5.97e-01)	(2.25e-03)	(1.75e-03)

相比, ESM-CNN 在本章节设置的绝大部分预测任务和评价指标上取得了更优的预测准确度; 在梯度下降训练模型取得最优结果的预测任务中, 如 ILI H8、BRENT-weekly H1、BRENT-daily H1、WTI-weekly H1 和 WTI-daily H5, ESM-CNN 预测模型也取得

表 1.5 ESM-CNN 及对照组模型在 AR1、BTC 和 ILI 数据集上的 RMSE 结果

数据集	H	统计模型			梯度下降模型			随机映射模型			消融模型		所提模型
		Naive	ARIMA	HWSES	GS-CNN	DeepAR	CLSTM	RVFL	IELM	SCN	Stoc-CNN	ES-CNN	ESM-CNN
AR1	1	1.16e+00 (-)	2.24e+00 (-)	2.23e+00 (-)	5.53e-01 (3.60e-02)	5.08e-01 (9.60e-02)	4.89e-01 (8.93e-02)	3.21e+00 (1.85e+00)	5.54e-01 (7.81e-02)	3.34e-01 (1.13e-01)	4.36e+02 (2.01e+02)	2.28e-01 (3.43e-02)	1.55e-01* (3.66e-03)
	3	1.19e+00 (-)	2.26e+00 (-)	2.25e+00 (-)	6.30e-01 (3.26e-02)	7.66e-01 (1.30e-01)	6.67e-01 (6.71e-02)	5.15e+00 (2.26e+00)	5.80e-01 (6.50e-02)	4.98e-01 (8.72e-02)	4.43e+02 (1.95e+02)	3.22e-01 (2.43e-02)	2.57e-01* (1.04e-02)
	6	1.25e+00 (-)	2.29e+00 (-)	2.28e+00 (-)	7.40e-01 (2.55e-02)	9.96e-01 (1.34e-01)	6.79e-01 (9.10e-02)	5.59e+00 (3.27e+00)	6.71e-01 (7.89e-02)	6.00e-01 (1.27e-01)	1.07e+03 (5.37e+02)	4.61e-01 (2.71e-02)	3.94e-01* (1.62e-02)
	1	1.47e+04 (-)	1.49e+04 (-)	1.49e+04 (-)	3.87e+03 (3.62e+02)	1.15e+04 (1.52e+03)	2.08e+04 (1.15e+03)	6.74e+04 (3.63e+04)	1.05e+04 (1.60e+03)	6.52e+03 (1.41e+03)	1.33e+07 (1.11e+07)	2.75e+03 (1.07e+03)	1.59e+03* (2.39e+01)
	3	1.47e+04 (-)	1.49e+04 (-)	1.49e+04 (-)	3.97e+03 (3.76e+02)	1.02e+04 (1.91e+03)	2.24e+04 (1.15e+03)	7.11e+04 (2.70e+04)	1.04e+04 (1.58e+03)	7.55e+03 (1.40e+03)	1.92e+07 (1.43e+07)	2.82e+03 (1.03e+03)	1.74e+03* (2.01e+01)
	6	1.46e+04 (-)	1.50e+04 (-)	1.50e+04 (-)	4.00e+03 (3.54e+02)	1.22e+04 (2.12e+03)	2.33e+04 (4.55e+02)	8.82e+04 (3.93e+04)	1.04e+04 (1.64e+03)	8.09e+03 (1.33e+03)	2.30e+07 (1.96e+07)	2.89e+03 (9.79e+02)	1.92e+03* (1.51e+01)
BTC	1	9.44e-01 (-)	2.23e+00 (-)	2.28e+00 (-)	8.45e-01 (1.95e-02)	7.82e-01 (1.87e-02)	7.10e-01 (2.52e-02)	1.74e+00 (3.17e-01)	8.88e-01 (4.41e-02)	7.14e-01 (3.11e-02)	1.50e+01 (2.50e+00)	6.81e-01 (5.37e-02)	6.06e-01* (8.14e-03)
	4	1.28e+00 (-)	2.25e+00 (-)	2.35e+00 (-)	1.00e+00 (9.40e-03)	9.95e-01 (1.52e-02)	9.07e-01 (1.71e-02)	1.96e+00 (2.49e-01)	1.04e+00 (3.56e-02)	9.09e-01 (2.52e-02)	1.39e+01 (1.54e+00)	8.65e-01 (2.35e-02)	8.14e-01* (8.75e-03)
	8	1.63e+00 (-)	2.36e+00 (-)	2.47e+00 (-)	1.16e+00 (5.03e-03)	1.16e+00 (2.11e-02)	1.04e+00 (1.76e-02)	2.00e+00 (2.32e-01)	1.18e+00 (2.07e-02)	1.11e+00 (2.54e-02)	1.43e+01 (2.03e+00)	1.07e+00 (1.07e-02)	1.03e+00* (6.63e-03)
	1	9.44e-01 (-)	2.23e+00 (-)	2.28e+00 (-)	8.45e-01 (1.95e-02)	7.82e-01 (1.87e-02)	7.10e-01 (2.52e-02)	1.74e+00 (3.17e-01)	8.88e-01 (4.41e-02)	7.14e-01 (3.11e-02)	1.50e+01 (2.50e+00)	6.81e-01 (5.37e-02)	6.06e-01* (8.14e-03)
	4	1.28e+00 (-)	2.25e+00 (-)	2.35e+00 (-)	1.00e+00 (9.40e-03)	9.95e-01 (1.52e-02)	9.07e-01 (1.71e-02)	1.96e+00 (2.49e-01)	1.04e+00 (3.56e-02)	9.09e-01 (2.52e-02)	1.39e+01 (1.54e+00)	8.65e-01 (2.35e-02)	8.14e-01* (8.75e-03)
	8	1.63e+00 (-)	2.36e+00 (-)	2.47e+00 (-)	1.16e+00 (5.03e-03)	1.16e+00 (2.11e-02)	1.04e+00 (1.76e-02)	2.00e+00 (2.32e-01)	1.18e+00 (2.07e-02)	1.11e+00 (2.54e-02)	1.43e+01 (2.03e+00)	1.07e+00 (1.07e-02)	1.03e+00* (6.63e-03)
ILI	1	9.44e-01 (-)	2.23e+00 (-)	2.28e+00 (-)	8.45e-01 (1.95e-02)	7.82e-01 (1.87e-02)	7.10e-01 (2.52e-02)	1.74e+00 (3.17e-01)	8.88e-01 (4.41e-02)	7.14e-01 (3.11e-02)	1.50e+01 (2.50e+00)	6.81e-01 (5.37e-02)	6.06e-01* (8.14e-03)
	4	1.28e+00 (-)	2.25e+00 (-)	2.35e+00 (-)	1.00e+00 (9.40e-03)	9.95e-01 (1.52e-02)	9.07e-01 (1.71e-02)	1.96e+00 (2.49e-01)	1.04e+00 (3.56e-02)	9.09e-01 (2.52e-02)	1.39e+01 (1.54e+00)	8.65e-01 (2.35e-02)	8.14e-01* (8.75e-03)
	8	1.63e+00 (-)	2.36e+00 (-)	2.47e+00 (-)	1.16e+00 (5.03e-03)	1.16e+00 (2.11e-02)	1.04e+00 (1.76e-02)	2.00e+00 (2.32e-01)	1.18e+00 (2.07e-02)	1.11e+00 (2.54e-02)	1.43e+01 (2.03e+00)	1.07e+00 (1.07e-02)	1.03e+00* (6.63e-03)
	1	9.44e-01 (-)	2.23e+00 (-)	2.28e+00 (-)	8.45e-01 (1.95e-02)	7.82e-01 (1.87e-02)	7.10e-01 (2.52e-02)	1.74e+00 (3.17e-01)	8.88e-01 (4.41e-02)	7.14e-01 (3.11e-02)	1.50e+01 (2.50e+00)	6.81e-01 (5.37e-02)	6.06e-01* (8.14e-03)
	4	1.28e+00 (-)	2.25e+00 (-)	2.35e+00 (-)	1.00e+00 (9.40e-03)	9.95e-01 (1.52e-02)	9.07e-01 (1.71e-02)	1.96e+00 (2.49e-01)	1.04e+00 (3.56e-02)	9.09e-01 (2.52e-02)	1.39e+01 (1.54e+00)	8.65e-01 (2.35e-02)	8.14e-01* (8.75e-03)
	8	1.63e+00 (-)	2.36e+00 (-)	2.47e+00 (-)	1.16e+00 (5.03e-03)	1.16e+00 (2.11e-02)	1.04e+00 (1.76e-02)	2.00e+00 (2.32e-01)	1.18e+00 (2.07e-02)	1.11e+00 (2.54e-02)	1.43e+01 (2.03e+00)	1.07e+00 (1.07e-02)	1.03e+00* (6.63e-03)

了差距很小的次优预测结果，展示出 ESM-CNN 预测模型与梯度下降训练深度学习预测模型匹敌的预测能力。

(4) 与 RVFL、IELM 和 SCN 所代表的先进 SMLP 预测模型相比，ESM-CNN 同样在本章节设置的所有预测任务和所有评价指标上都表现出更优的预测准确度，展示出 ESM-CNN 预测模型在随机映射预测模型中的预测优势。

基于表 1.3 – 1.8 所示的消融模型结果，可归纳出以下结论：

(1) 仅单纯引入随机映射方法的 Stoc-CNN 预测模型在本章节选取的所有预测任务和所有评价指标上都表现出显著弱于 ES-CNN 与 ESM-CNN 的预测性能，并在多项预测任务中表现出最差的预测准确度，验证了病态问题在全局更新输出权重方式下所导致的预测性能问题。

(2) 引入误差反馈随机映射构造策略的 ES-CNN 预测模型与 Stoc-CNN 相比，取得了显著的准确度提升，并在大部分预测任务和评价指标中表现出次优的水平，展现出基于误差反馈随机映射策略构造 CNN 预测模型的有效性与必要性。

(3) 与 ES-CNN 和 Stoc-CNN 预测模型相比，ESM-CNN 在本章节设置的所有预测任务和所有评价指标上都表现出更优的预测准确度，证明了所提贪心选择方法在 ESM-CNN 预测模型构造过程中的有效性与必要性。

表 1.6 ESM-CNN 及对照组模型在原油价格与股票指数数据集上的 MAPE 结果

数据集	H	统计模型			梯度下降模型			随机映射模型			消融模型		所提模型
		Naive	ARIMA	HWSES	GS-CNN	DeepAR	CLSTM	RVFL	IELM	SCN	Stoc-CNN	ES-CNN	ESM-CNN
BRENT-weekly	1	5.02e-01	1.09e-01	1.17e-01	1.60e-01	4.37e-02	8.25e-02	5.88e-02	1.57e-01	5.74e-02	1.93e+00	4.57e-02	3.97e-02*
	4	4.97e-01	2.13e-01	2.37e-01	1.78e-01	9.32e-02	1.15e-01	1.19e-01	1.73e-01	8.67e-02	2.87e+00	7.78e-02	7.49e-02*
	8	5.10e-01	3.26e-01	3.69e-01	1.82e-01	1.77e-01	1.85e-01	1.81e-01	2.12e-01	1.43e-01	3.70e+00	1.14e-01	1.11e-01*
BRENT-daily	1	4.80e-01	8.47e-02	8.92e-02	7.84e-02	1.93e-02*	3.51e-02	2.14e-02	1.36e-01	2.01e-02	5.06e-02	2.02e-02	1.99e-02
	5	4.82e-01	2.26e-01	2.45e-01	9.37e-02	3.48e-02	4.44e-02	3.92e-02	1.42e-01	3.50e-02	8.78e-02	3.42e-02	3.37e-02*
	10	4.88e-01	3.99e-01	4.34e-01	9.50e-02	5.18e-02	5.50e-02	5.51e-02	1.49e-01	5.10e-02	1.29e-01	5.37e-02	4.63e-02*
WTI-weekly	1	4.71e-01	9.99e-02	1.10e-01	2.26e-01	5.49e-02*	1.01e-01	7.10e-02	2.34e-01	6.33e-02	2.10e+00	6.27e-02	5.77e-02
	4	4.78e-01	2.06e-01	2.40e-01	2.47e-01	1.11e-01	1.45e-01	1.34e-01	2.53e-01	1.06e-01	3.11e+00	9.46e-02	8.64e-02*
	8	5.11e-01	3.18e-01	3.71e-01	2.46e-01	1.72e-01	2.16e-01	2.06e-01	2.81e-01	1.35e-01	3.70e+00	1.32e-01	1.24e-01*
WTI-daily	1	4.44e-01	7.82e-02	8.33e-02	9.30e-02	2.11e-02	3.62e-02	2.15e-02	1.45e-01	2.14e-02	7.24e-02	2.08e-02	2.06e-02*
	5	4.49e-01	2.05e-01	2.21e-01	9.90e-02	3.50e-02*	4.65e-02	4.14e-02	1.51e-01	3.67e-02	1.22e-01	3.62e-02	3.57e-02
	10	4.60e-01	3.57e-01	3.86e-01	9.96e-02	5.21e-02	5.76e-02	5.93e-02	1.58e-01	5.20e-02	1.67e-01	5.99e-02	4.99e-02*
S&P 500	1	1.50e-01	5.17e-03	8.26e-03	1.92e-02	1.28e-02	4.88e-02	8.82e-03	3.95e-02	1.18e-02	1.73e+00	5.33e-03	4.06e-03*
	5	1.46e-01	7.15e-03	1.10e-02	2.03e-02	1.87e-02	7.20e-02	1.91e-02	5.20e-02	2.20e-02	1.09e+01	7.03e-03	6.17e-03*
	10	1.48e-01	8.91e-03	1.32e-02	1.94e-02	7.97e-02	5.95e-02	2.40e-02	4.65e-02	3.46e-02	1.43e+01	8.71e-03	8.03e-03*
NASDAQ	1	1.77e-01	7.16e-03	1.04e-02	2.84e-02	7.19e-03	6.86e-02	3.65e-02	5.78e-02	1.62e-02	3.85e+01	6.68e-03	5.50e-03*
	5	1.73e-01	1.03e-02	1.40e-02	3.09e-02	2.15e-02	9.06e-02	3.50e-02	6.12e-02	2.36e-02	6.13e+01	9.12e-03	8.84e-03*
	10	1.75e-01	1.29e-02	1.73e-02	3.05e-02	1.53e-01	8.47e-02	3.52e-02	6.00e-02	4.35e-02	6.43e+01	1.15e-02	1.13e-02*
DJI	1	1.35e-01	5.24e-03	8.14e-03	2.36e-02	1.07e-02	4.28e-02	1.55e-02	4.73e-02	7.57e-03	6.20e+00	5.06e-03	4.09e-03*
	5	1.31e-01	7.53e-03	1.10e-02	2.54e-02	6.72e-02	6.54e-02	1.84e-02	4.72e-02	1.93e-02	4.73e+00	7.10e-03	6.71e-03*
	10	1.33e-01	9.55e-03	1.34e-02	2.56e-02	3.74e-02	6.64e-02	3.60e-02	4.70e-02	3.89e-02	7.16e+00	9.14e-03	9.06e-03*
NYSE	1	1.12e-01	5.61e-03	8.91e-03	1.31e-02	9.30e-03	1.53e-02	6.55e-03	2.63e-02	9.61e-03	6.68e-01	4.58e-03	4.52e-03*
	5	1.09e-01	7.68e-03	1.19e-02	1.44e-02	1.93e-02	2.30e-02	1.82e-02	2.65e-02	1.31e-02	9.32e-01	6.77e-03	6.69e-03*
	10	1.10e-01	9.24e-03	1.42e-02	1.53e-02	9.84e-02	2.89e-02	2.60e-02	2.69e-02	1.81e-02	1.45e+00	8.44e-03	8.38e-03*

1.4.2 收敛效果对比与分析

本章节对比了基于递归增长随机映射构造策略模型（即 IELM、SCN、ES-CNN 和 ESM-CNN 模型）的收敛性，以更好评价所提模型的收敛效果。图 1.3 绘制出了 ILI 数据集上随递归增长随机映射模型随隐藏层神经元或卷积核数量变化所达到的训练集与验证集 RMSE 曲线。基于该收敛性结果，可归纳出以下结论：

（1）同一预测模型在不同预测时长任务上的收敛情况是一致的。

（2）尽管 SCN 模型在不同预测任务的训练集 RMSE 上都能做出最快收敛，但通过验证集 RMSE 曲线可知，SCN 模型在进行真实预测建模中出现了明显的过拟合现象，而采用误差反馈策略的 IELM 模型并没有出现过拟合，但表现出最慢的收敛速度。这一对比结果证明了误差反馈策略解决过拟合问题的有效性，同时也指出了误

表 1.7 ESM-CNN 及对照组模型在原油价格与股票指数数据集上的 SMAPE 结果

数据集	H	统计模型			梯度下降模型			随机映射模型			消融模型		所提模型
		Naive	ARIMA	HWSES	GS-CNN	DeepAR	CLSTM	RVFL	IELM	SCN	Stoc-CNN	ES-CNN	ESM-CNN
BRENT-weekly	1	3.25e-01	1.00e-01	9.90e-02	6.70e-02	2.07e-02	3.89e-02	2.82e-02	6.64e-02	2.84e-02	9.65e-01	2.26e-02	1.97e-02*
	4	9.68e-01	6.22e-01	3.17e-01	7.37e-02	4.17e-02	5.37e-02	5.42e-02	7.15e-02	4.19e-02	1.14e+00	3.79e-02	3.67e-02*
	8	3.68e-01	7.03e-01	4.40e-01	7.74e-02	7.57e-02	7.91e-02	8.30e-02	8.42e-02	6.06e-02	1.40e+00	5.35e-02	5.25e-02*
BRENT-daily	1	6.55e-01	1.63e-01	1.69e-01	3.67e-02	9.55e-03*	1.73e-02	1.05e-02	6.05e-02	9.92e-03	3.61e-02	1.00e-02	9.91e-03
	5	4.22e-01	5.21e-01	4.84e-01	4.31e-02	1.72e-02	2.18e-02	1.87e-02	6.26e-02	1.71e-02	6.66e-02	1.70e-02	1.67e-02*
	10	4.17e-01	7.59e-01	6.83e-01	4.38e-02	2.50e-02	2.68e-02	2.58e-02	6.51e-02	2.44e-02	1.39e-01	2.60e-02	2.28e-02*
WTI-weekly	1	3.69e-01	7.82e-02	9.42e-02	7.89e-02	2.06e-02*	4.17e-02	3.08e-02	8.89e-02	2.48e-02	6.86e-01	2.52e-02	2.44e-02
	4	9.06e+01	2.10e-01	2.81e-01	8.50e-02	4.00e-02	5.86e-02	6.84e-02	9.40e-02	4.13e-02	8.43e-01	3.97e-02	3.90e-02*
	8	3.34e-01	3.80e-01	5.26e-01	8.43e-02	5.92e-02	7.93e-02	2.88e-01	1.00e-01	5.51e-02	2.75e+00	5.53e-02	5.17e-02*
WTI-daily	1	8.29e-01	1.34e-01	1.75e-01	4.26e-02	1.05e-02	1.78e-02	1.06e-02	6.45e-02	1.06e-02	1.17e-01	1.04e-02	1.02e-02*
	5	3.63e-01	6.75e-01	4.02e-01	4.50e-02	1.73e-02*	2.27e-02	1.95e-02	6.67e-02	1.79e-02	1.10e-01	1.76e-02	1.75e-02
	10	4.20e-01	7.36e-01	8.09e-01	4.53e-02	2.51e-02	2.78e-02	2.73e-02	6.94e-02	2.47e-02	1.46e-01	2.83e-02	2.41e-02*
S&P 500	1	7.60e-02	2.59e-03	4.13e-03	9.68e-03	6.47e-03	2.53e-02	4.41e-03	2.03e-02	5.97e-03	5.23e+00	2.66e-03	2.03e-03*
	5	7.46e-02	3.58e-03	5.50e-03	1.02e-02	9.48e-03	3.79e-02	9.68e-03	2.69e-02	1.12e-02	4.37e+00	3.50e-03	3.08e-03*
	10	7.57e-02	4.47e-03	6.61e-03	9.81e-03	4.24e-02	3.11e-02	1.22e-02	2.40e-02	1.77e-02	7.42e+00	4.34e-03	4.02e-03*
NASDAQ	1	9.04e-02	3.58e-03	5.19e-03	1.44e-02	3.61e-03	3.61e-02	1.76e-02	3.00e-02	8.22e-03	1.34e+00	3.33e-03	2.75e-03*
	5	8.91e-02	5.15e-03	7.01e-03	1.57e-02	1.09e-02	4.85e-02	1.69e-02	3.19e-02	1.20e-02	1.88e+00	4.54e-03	4.42e-03*
	10	9.08e-02	6.50e-03	8.69e-03	1.55e-02	8.73e-02	4.53e-02	1.71e-02	3.12e-02	2.24e-02	1.77e+00	5.73e-03	5.66e-03*
DJI	1	6.83e-02	2.62e-03	4.08e-03	1.20e-02	5.36e-03	2.21e-02	7.66e-03	2.45e-02	3.80e-03	4.52e-01	2.52e-03	2.05e-03*
	5	6.71e-02	3.78e-03	5.51e-03	1.29e-02	3.58e-02	3.44e-02	9.04e-03	2.44e-02	9.78e-03	7.47e-01	3.54e-03	3.36e-03*
	10	6.81e-02	4.80e-03	6.74e-03	1.30e-02	1.92e-02	3.50e-02	1.73e-02	2.43e-02	2.01e-02	1.60e+00	4.57e-03	4.53e-03*
NYSE	1	5.63e-02	2.81e-03	4.46e-03	6.60e-03	4.67e-03	7.73e-03	3.28e-03	1.33e-02	4.83e-03	1.12e+00	2.29e-03	2.26e-03*
	5	5.52e-02	3.84e-03	5.98e-03	7.27e-03	9.77e-03	1.17e-02	9.21e-03	1.35e-02	6.59e-03	4.99e+00	3.38e-03	3.35e-03*
	10	5.61e-02	4.64e-03	7.12e-03	7.73e-03	5.49e-02	1.48e-02	1.33e-02	1.37e-02	9.13e-03	3.56e+00	4.22e-03	4.19e-03*

差反馈策略下 MLP 结构的性能局限性。

(3) 基于卷积结构的 ESM-CNN 和 ES-CNN 在所有的验证集 RMSE 中都表现出比 IELM 和 SCN 更快且更低的收敛效果, 证明了基于误差反馈随机映射构造 CNN 预测模型的收敛性与鲁棒性。

(4) 与 ES-CNN 相比, ESM-CNN 具备更低的训练集 RMSE 曲线和测试集 RMSE 曲线, 从收敛性角度再次验证了所提卷积核选择方法的有效性与必要性。

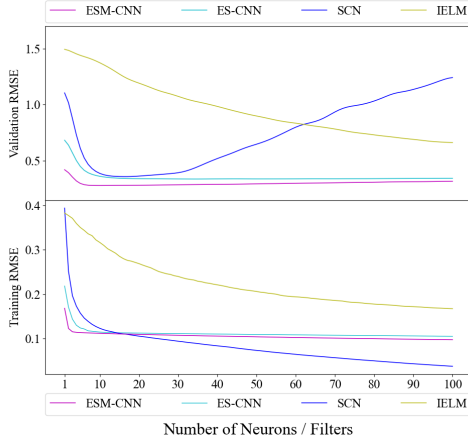
(5) 在 ESM-CNN 与 ES-CNN 的收敛过程中, 预测性能的最大增益明显来自于构造过程的前 10 步, 表现出 ESM-CNN 具备通过牺牲微小性能换来巨大效率提升的平衡策略, 从而使得 ESM-CNN 具有更加广泛和高效的现实应用性。

进一步地, 对于本章实验中图 1.3 所显示的 SCN 与 Stoc-CNN 过拟合现象以及

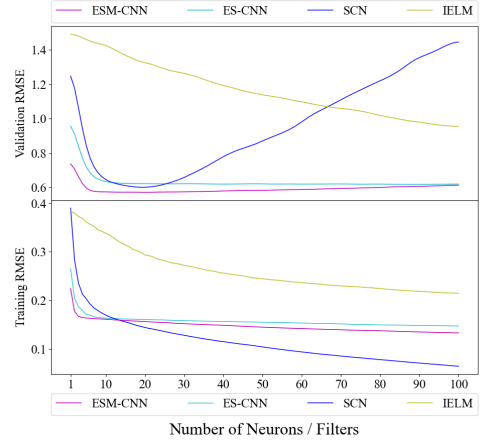
表 1.8 ESM-CNN 及对照组模型在原油价格与股票指数数据集上的 RMSE 结果

数据集	H	统计模型			梯度下降模型			随机映射模型			消融模型		所提模型
		Naive	ARIMA	HWSES	GS-CNN	DeepAR	CLSTM	RVFL	IELM	SCN	Stoc-CNN	ES-CNN	ESM-CNN
BRENT-weekly	1	3.24e+01	7.23e+00	7.49e+00	1.01e+01	2.90e+00	5.21e+00	3.86e+00	9.91e+00	3.73e+00	2.67e+02	2.89e+00	2.62e+00*
	4	3.21e+01	1.70e+01	1.78e+01	1.12e+01	6.06e+00	7.75e+00	9.06e+00	1.08e+01	5.81e+00	3.83e+02	5.20e+00	5.14e+00*
	8	3.31e+01	2.96e+01	3.13e+01	1.12e+01	1.15e+01	1.13e+01	1.41e+01	1.31e+01	9.31e+00	4.86e+02	7.61e+00	7.52e+00*
BRENT-daily	1	3.14e+01	5.89e+00	5.95e+00	5.22e+00	1.32e+00*	2.29e+00	1.44e+00	8.70e+00	1.35e+00	4.19e+00	1.35e+00	1.34e+00
	5	3.17e+01	1.88e+01	1.88e+01	6.19e+00	2.34e+00	2.93e+00	2.70e+00	9.05e+00	2.35e+00	7.92e+00	2.32e+00	2.29e+00*
	10	3.21e+01	3.53e+01	3.51e+01	6.26e+00	3.82e+00	3.73e+00	3.75e+00	9.48e+00	3.41e+00	1.19e+01	3.68e+00	3.16e+00*
WTI-weekly	1	2.87e+01	7.00e+00	7.31e+00	1.22e+01	2.64e+00*	5.35e+00	4.04e+00	1.24e+01	3.08e+00	2.36e+02	3.05e+00	2.81e+00
	4	2.91e+01	1.60e+01	1.68e+01	1.34e+01	5.92e+00	8.39e+00	9.19e+00	1.33e+01	5.50e+00	3.65e+02	5.21e+00	5.01e+00*
	8	2.95e+01	2.79e+01	2.94e+01	1.34e+01	8.93e+00	1.43e+01	1.42e+01	1.45e+01	7.51e+00	4.25e+02	7.65e+00	7.21e+00*
WTI-daily	1	2.83e+01	5.37e+00	5.43e+00	5.97e+00	1.37e+00	2.24e+00	1.39e+00	8.91e+00	1.41e+00	6.97e+00	1.37e+00	1.35e+00*
	5	2.85e+01	1.71e+01	1.69e+01	6.36e+00	2.24e+00*	2.92e+00	2.79e+00	9.26e+00	2.41e+00	1.25e+01	2.33e+00	2.32e+00
	10	2.89e+01	3.23e+01	3.15e+01	6.43e+00	3.54e+00	3.85e+00	4.08e+00	9.70e+00	3.39e+00	1.74e+01	3.84e+00	3.23e+00*
S&P 500	1	4.24e+02	1.67e+01	2.49e+01	5.48e+01	3.46e+01	1.47e+02	2.61e+01	1.15e+02	3.44e+01	5.89e+03	1.56e+01	1.28e+01*
	5	4.20e+02	2.22e+01	3.32e+01	5.77e+01	5.29e+01	2.09e+02	5.58e+01	1.45e+02	6.34e+01	4.22e+04	2.12e+01	1.93e+01*
	10	4.23e+02	2.75e+01	4.00e+01	5.49e+01	2.39e+02	1.71e+02	6.83e+01	1.34e+02	1.01e+02	5.56e+04	2.58e+01	2.43e+01*
NASDAQ	1	1.23e+03	5.42e+01	7.54e+01	2.07e+02	4.99e+01	5.17e+02	3.20e+02	4.30e+02	1.29e+02	3.70e+05	4.95e+01	4.24e+01*
	5	1.22e+03	7.41e+01	1.02e+02	2.23e+02	1.57e+02	6.80e+02	3.44e+02	4.51e+02	1.79e+02	5.97e+05	6.75e+01	6.54e+01*
	10	1.23e+03	9.38e+01	1.26e+02	2.18e+02	1.19e+03	6.56e+02	3.19e+02	4.41e+02	3.16e+02	6.22e+05	8.36e+01	8.29e+01*
DJI	1	3.32e+03	1.44e+02	2.09e+02	6.28e+02	2.70e+02	1.19e+03	4.56e+02	1.26e+03	2.06e+02	2.64e+05	1.30e+02	1.10e+02*
	5	3.29e+03	2.03e+02	2.87e+02	6.72e+02	1.98e+03	1.76e+03	6.17e+02	1.25e+03	5.07e+02	2.02e+05	1.84e+02	1.80e+02*
	10	3.32e+03	2.59e+02	3.58e+02	6.70e+02	1.01e+03	1.80e+03	1.27e+03	1.23e+03	1.07e+03	2.85e+05	2.36e+02	2.33e+02*
NYSE	1	1.55e+03	8.98e+01	1.33e+02	1.84e+02	1.27e+02	2.31e+02	9.40e+01	3.64e+02	1.42e+02	1.66e+04	6.97e+01	6.86e+01*
	5	1.54e+03	1.17e+02	1.76e+02	2.02e+02	2.90e+02	3.45e+02	2.82e+02	3.68e+02	1.93e+02	2.28e+04	1.05e+02	1.03e+02*
	10	1.55e+03	1.38e+02	2.10e+02	2.14e+02	1.68e+03	4.31e+02	4.05e+02	3.72e+02	2.56e+02	3.93e+04	1.28e+02	1.26e+02*

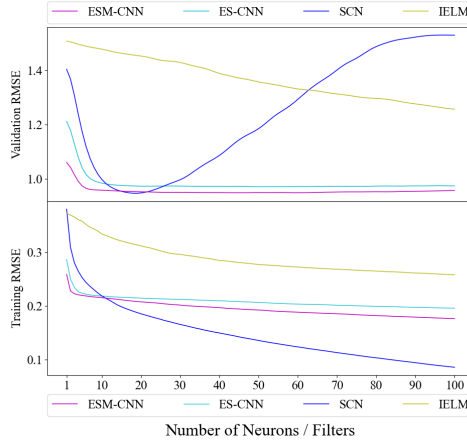
ESM-CNN 与 ES-CNN 模型的鲁棒性特点，其解释如下：由于 SCN 与 Stoc-CNN 均采用最小二乘法全局更新输出权重的方式，使其难以克服现实复杂时间序列中的噪声数据，因此出现了过拟合的现象；与全局更新输出层权重的构造方式相反，ESM-CNN 与其消融模型 ES-CNN 采用基于构造过程预测误差反馈局部更新输出权重的方式，随着卷积核的不断增加，一方面，既有预测误差被不断弥补，使得噪声数据的影响被不断削弱，另一方面历史加入的卷积核权重及其链接的输出权重被固定住，这些固定的权重在一定程度上充当了正则化的作用，使得新加入卷积核的输出权重 L_2 范数越来越小，并在整体上小于 SCN 与 Stoc-CNN，从而使模型具有更高的稳定性与鲁棒性。



(a) ILI, $H = 1$



(b) ILI, $H = 4$



(c) ILI, $H = 8$

图 1.3 ILI 数据集上递归增长随机映射神经网络的 RMSE 曲线

1.4.3 建模效率对比与分析

此外，本章节对比了所选模型中神经网络预测模型的单次平均建模时间，以更好评价所提模型的实际建模效率。表 1.9 展示了所选神经网络预测模型在 AR1、BTC 和 ILI 数据集上以秒为单位的单次平均建模时间。其中，因 RVFL 与 Stoc-CNN 的构造过程仅由随机初始化隐藏层权重与闭式求解输出权重两部分组成，在当前实验环境下，模型构造时间可被省略，因此用“-”进行标记。

通过对比卷积结构的训练 DNN 模型（即 GS-CNN）与循环结构的训练 DNN 模型（即 DeepAR 和 CLSTM），GS-CNN 模型具有更高的建模效率，其用时约是纯循

表 1.9 神经网络模型在 AR1、BTC 和 ILI 数据集上的单次平均建模时间 (s)

建模方法	模型	AR1			BTC			ILI		
		1	3	6	1	3	6	1	4	8
梯度下降	GS-CNN	74.65	76.95	78.55	93.70	96.45	99.40	67.05	70.25	73.00
	DeepAR	225.25	238.95	257.70	223.75	251.10	268.65	256.80	279.10	275.30
	CLSTM	150.45	154.80	160.25	143.00	141.50	142.25	167.40	170.80	175.30
随机映射	RVFL	-	-	-	-	-	-	-	-	-
	IELM	2.65	2.85	2.80	3.40	3.30	3.20	2.55	2.55	2.40
	SCN	22.85	57.40	107.75	21.10	58.00	111.40	16.10	49.60	90.95
消融方法	Stoc-CNN	-	-	-	-	-	-	-	-	-
	ES-CNN	6.50	6.40	6.50	6.35	6.50	6.45	6.45	6.50	6.50
所提方法	ESM-CNN	9.65	9.65	9.70	7.10	7.25	7.25	7.15	7.15	7.20

环结构模型 DeepAR 的三分之一，约是堆叠卷积与循环结构模型 CLSTM 的二分之一，验证了卷积结构的效率优势。

进一步地，结合前文的预测结果可知，ESM-CNN 与 ES-CNN 仅需相对极少的建模时间即可达到与梯度下降训练深度学习预测模型媲美甚至更好的预测性能，同时参考图 1.3 所示的收敛性结果可知，该建模时间仍有相当压缩空间以进一步提高建模效率，证明了误差反馈随机映射方法构造 CNN 模型的建模高效性。同时，ESM-CNN 与 ES-CNN 相比，增加的卷积核选择过程在未明显增加建模时间的情况下显著提高了预测性能，证明了 ESM-CNN 的建模优势。

1.5 小结

面对预测建模技术研究中卷积结构 SDNN 预测模型的构造与自适应优化问题，针对已有方法的诸多不足，本章提出了一种新颖的 ESM-CNN 预测模型，基于误差反馈随机映射建模策略和卷积参数贪心选择方法，构造出一种兼具优异建模效率与预测性能的卷积结构 SDNN 预测模型，并证明了其收敛性。

本章节以经典人工时间序列数据、比特币价格、流感阳性样本率、国际原油价格、股票指数等多项真实时间序列数据为预测对象，进行多个预测时长下的预测任务，基于不同预测准确度评价指标，通过预测准确度、收敛性和建模效率等多个视角的对比分析验证了所提 ESM-CNN 模型的性能与效率，结果表明该模型是一种兼

具优秀预测性能与建模效率的卷积结构 SDNN 预测模型。有望基于该模型为比特币价格预测、流感阳性样本率预测、国际原油价格预测和股票指数预测等管理应用提供相比传统统计方法和传统随机映射方法更为精准的预测结果，同时可利用其远优于训练深度学习预测方法的建模效率优势，为时变或在线环境下的复杂预测管理应用问题提供可能的解决方案。

2 时间序列预测建模框架

2.1 引言

时间序列预测建模技术是能源、交通、金融和工业生产等众多重要现实管理应用领域的重要支撑技术。在不同场景中，时间序列数据表现出动态机制不确定、模态表现不相同、数据规模不一致的高度复杂特征。因此，时间序列预测问题是一项颇具挑战和相当复杂的问题。如何针对不同场景构造准确、稳定和鲁棒的时间预测模型是众多研究者与工作者所关注的重要课题。

针对时间序列预测建模问题，各种预测建模技术推陈出新。其中，以深度神经网络为代表的深度学习预测建模技术因其优异的预测性能成为了近年的研究热点，涌现出大量基于深度神经网络模型的技术应用研究^[25,27-34]。但在复杂多变的不同应用场景中，这些深度学习预测建模方法往往无法给予稳定的性能保证和一致的性能优势。原因在于，作为一种基于复杂神经网络结构构造的学习预测模型，深度学习预测模型往往具有远超传统机器学习方法或统计方法的参数设置，同时，传统深度学习预测模型的梯度下降迭代训练权重参数方法进一步降低了模型构造的效率，使得深度学习预测模型的构造过程面临建模效率低、模型选择难的问题。

作为深度学习预测建模技术应用与发展的关键，深度学习预测模型的高效构造与选择问题受到学术界与工业的广泛关注。为便捷构造深度神经网络模型，Google、Meta（Facebook）和 Microsoft 分别推出了 TensorFlow^[35]、Pytorch^[26] 和 CNTK^[36] 等神经网络模型构造框架，通过封装常用深度神经网络基础结构（如卷积核结构、循环层结构和线性结构），集成多种先进梯度下降训练算法和矩阵运算与处理算法，为深度学习模型的深入研究与广泛应用提供了框架。为解决深度学习模型复杂的超参数搜索与优化问题，Bennet 等^[37] 提出了一种开放优化框架 Nevergrad，通过内置贝叶斯优化、粒子群优化和差分进化等优化算法为深度学习模型参数搜索与优化提供了便利平台。Akiba 等^[38] 基于贝叶斯类优化方法提供了一套运行定义（Define-by-run）模型的优化框架 OPTUNA，通过更个性化的搜索空间定义接口，支持搜索范围可根据采样结果自适应变动的复杂搜索设置。Liaw 等^[39] 通过集成 Nevergrad 和 OPTUNA

等优化框架，提供了一套内置多种优化算法的分布式优化平台 Ray，进一步提高了模型参数优化的效率。

然而，这些建模与优化框架技术主要关注于广义建模技术中的特定环节，欠缺针对时间序列预测领域的建模考量。在面对复杂多变的时间序列管理应用预测需求中，仅依托于建模技术中的某一环节构造预测建模对于提升模型准确性、提高模型构造与选择效率、适应运营管理水平是不够的。因此，本章节基于前述各章在随机映射深度学习预测建模技术模型构造、隐藏结构优化、多输出结构优化和多输入特征选择等研究的基础上，设计出一套时间序列预测建模框架。通过整合既有的优秀神经网络建模框架与参数优化框架，针对时间序列预测问题，设计符合时间序列数据的数据定义与处理机制，完备包含时间序列预测建模的数据初始化、数据预处理、模型构造、模型优化和模型评价等构造与评价流程，集成多类别、多结构的现有对比预测方法，基于高度模块化和接口标准化的规范，使得用户能够较好地节约开发精力、复现已有工作、探索全新思路和解决预测问题。

2.2 框架需求分析

为应对不同场景下动态机制不确定、模态特征不相同、数据规模不一致的时间序列数据预测问题，基于数据驱动，构造自适应优化输入特征结构、模型隐藏结构与输出结构的时间序列预测模型，并呈现模型训练与预测效果的强弱程度，时间序列预测建模框架应当至少满足以下几点需求：

2.2.1 数据管理功能

深度学习预测建模技术的基础是大量的时间序列数据。时间序列数据高效与准确的处理以及统一的特征表示是构造普适结构深度学习预测模型的基础。然而，基于数据来源的场景与获取方式的不同，时间序列预测建模框架所面临的时间序列数据往往表现出不同的数据存储格式（如 json、txt、csv 和 xlsx 等文本格式）、时间粒度（如分钟、小时、日、周和月等）、数据缺失与异常情况（如 WTI 原油价格在 2020 年 4 月 20 日收盘时价格报-37.63）等数据表示。针对不同的时间序列预测建模技术，模型受入的时间序列数据结构也并不尽同，如本文第 5 章节所述，支持向量

机（SVM）与多层感知机（MLP）的多输入时间序列数据结构为包含样本数与特征维度的二维结构，而卷积神经网络（CNN）和循环神经网络（RNN）的多输入结构为包含样本数、时步维度和输入步长的三维结构。

因此，时间序列预测建模框架需要基于异构时间序列数据，提供差异化的数据读取功能，并针对不同类别的时间序列预测建模方法，提供差异的数据特征表示，而针对相同类别的预测建模方法，则需要提供统一的数据表示。同时，在此基础上需提供一致的数据预处理和后处理操作，以方便不同方法间的比较、展示与分析。

2.2.2 模型构造功能

构造时间序列预测模型是预测建模框架的核心功能。然而，受时间序列输入数据特征、模型自身参数和预测时长等因素的影响，预测模型在不同的时间序列预测任务上的表现可能有所差别。如本文第 4 章所述，节 4.3 中的实验结果表明，即使是同一预测任务上相同编码结构的回声状态网络（ESN）预测模型，受不同多输出结构的影响，多步预测性能亦表现出了明显差异。再如本文第 3 章所述，节 3.3 中的消融实验结果表明，通过对随机映射子网络的权重参数施加训练，可进一步提升模型预测性能。此外，为分析所构造预测模型的预测性能，丰富多样的对比预测模型是必要的，从而客观评价所构造模型的优劣。

因此，时间序列预测建模框架需要单元化、模块化和标准化的模型结构设计，以满足模型高度自定义的构造需求。同时，框架应包含多类别、多结构的优秀对比方法，为判断预测模型性能提供依据，从而适应不同预测任务的需要。

2.2.3 模型优化功能

解决时间序列预测模型的模型选择问题，优化所构造的预测模型是提升模型预测准确度从而增强模型应用性的关键。然而，不同预测模型建模方法的参数搜索优化空间各不相同，不同参数优化方法适应解决的优化问题也不相同。例如，作为群体智能优化算法的代表之一，粒子群优化（PSO）算法被应用于解决本文第 4 章节 ESN 预测模型的状态掩码优化问题，但不适用于解决第 5 章节所述的多输入特征结构优化问题。甚至，同一预测建模方法在不同预测任务上的参数搜索空间与优化算

法设置亦不相同。

因此，时间序列预测建模框架需要在标准化设计的模型接口基础上，提供多种优秀的优化算法，根据不同的时间序列预测任务情形，智能、高效和便捷地解决模型结构参数优化、多输出结构优化和多输入特征选择等模型选择问题，从而提高模型预测精度，进一步建立与任务情形相匹配的预测模型。

2.3 框架结构设计

根据时间序列预测任务的需要，融合既有模型构造与选择框架的优势^[26,39]，本章节设计并开源提供了一套名为“OpenForecasting”的预测建模框架。该框架基于 Python 这一动态语言，充分利用 Python 类的继承与多态特征，通过对 Pytorch 与 Ray 的封装，提供数据处理高度个性化、模型高度模组化、接口高度标准化的交互功能，基于完备的预测建模流程逻辑，有效地实现复杂时间序列数据的预测建模任务。这种框架结构有利于用户通过定义符合接口标准的模组，扩展现有方法或功能，满足更复杂的预测建模需求。

具体地，“OpenForecasting”预测框架主要通过 TaskWrapper 实现预测建模流程控制，利用 TaskLoader 实现数据载入与预处理，基于 TaskTuner 实现预测模型优化，辅以统一接口的模型库和优化方法库，完成预测建模。

2.3.1 预测流程控制

在本框架设计的 TaskWrapper 类中，基于接收用户指定的预测任务数据名称、预测时长、预测模型等预测任务信息，通过嵌套与调用数据加载、模型选择、模型预测和模型评价功能，自适应地完成时间序列预测模型的构造、优化与评价。

图 2.1 展示了“OpenForecasting”预测建模框架的流程控制主文件：task.py。通过接收用户传入的数据集名称、模型名称、预测时长、重复执行次数和预测准确度指标等任务信息，Task 类在初始化时完成如图 2.2 所示的数据预处理与加载、模型参数初始化和实验日志初始化过程。以本文第 4 章节中 ESN 模型在 MG 数据集上的预测建模实验为例，用户可采用如图 2.4 所示方式，通过自定义 task 主文件显式指定任务参数，构造基于 MG 数据集、预测时长为 17、重复实验次数为 20、预测评价

```
1 from task.parser import
   get_parser
2 from task.TaskWrapper import Task
3
4 if __name__ == "__main__":
5     args = get_parser()
6
7     task = Task(args)
8     task.tuning()
9     task.conduct()
10    task.evaluation(args.metrics)
```

图 2.1 预测建模框架的主文件：task.py

```
1 from task.TaskLoader import Opt
2 import importlib
3 import ...
4
5 class Task(Opt):
6     def __init__(self, args):
7
8         self.data_config(args)
9         self.model_config(args)
10        self.exp_config(args)
11
12    ...
```

图 2.2 TaskWrapper 中的 Task 类

```
1 from task.parser import get_parser
2 from task.TaskWrapper import Task
3 if __name__ == "__main__":
4
5     args = get_parser()
6
7     args.cuda = True
8     args.datafolder = 'paper.sm'
9     args.dataset = 'mg'
10    args.H = 17
11    args.model = 'esn'
12    args.rep_times = 20
13    args.metrics = ['rmse', 'nrmse', 'mape']
14
15    task = Task(args)
16    task.tuning()
17    task.conduct()
18    task.evaluation(args.metrics)
```

图 2.3 自定义主文件 task.py 传入任务参数的示例

```
1 python task.py -datafolder paper.sm -dataset mg -H 17 -model esn -rep_times
   20 -metrics rmse nrmse mape
```

图 2.4 由命令行传入任务参数的示例

指标为 RMSE、NRMSE 和 MAPE 的 ESN 预测模型，并利用“task.tuning”函数完成 ESN 模型的参数优化，在优化结果的基础上更新模型预设参数，通过“task.conduct”函数进行预测建模，最终基于“task.evaluation”函数完成模型在各评价指标上的结果统计，获取对应指标的平均值和标准差等统计信息。用户亦可采用如图 2.4 所示的命令行传入参数方式，完成相同预测建模任务。此外，通过执行多个传入不同参数的 task.py 程序，用户可通过并行的方式针对同一预测建模任务构造不同的预测模型，


```
1 class TaskDataset(Opt):
2     def __init__(self, args):
3         super().__init__()
4         self.args = args
5         self.info = Opt()
6         self.info.normal = args.normal
7         self.info.H = args.H
8         self.info_config()
9         self.sub_config()
10        self.info.num_series = len(self.seriesPack)
11
12    def info_config(self, ):
13        pass
14
15    def sub_config(self,):
16        pass
17
18    ...
```

图 2.5 TaskLoader 中的 TaskDataset 类

亦可基于相同的预测模型完成不同的预测建模任务，显著提高了用户使用的便利性。

2.3.2 数据预处理与加载

为应对不同数据存储结构的时间序列数据，适应不同输入步长设置、不同预测时长设置的时间序列预测任务需要，本框架通过设计 TaskLoader 中的 TaskDataset 父类，为不同时间序列数据提供加载接口。在转换得到统一数据格式的时间序列数据后，对数据进行归一化和数据集切分处理，完成预测建模任务的数据预处理与加载过程。

图 2.5 展示了“OpenForecasting”预测建模框架的 TaskDataset 父类，通过提供“info_config”与“sub_config”接口，使得用户可针对不同数据与任务情形，创建继承 TaskDataset 的 Data 子类，通过重构“info_config”与“sub_config”函数，灵活处理源数据以返回符合框架标准的加工数据。以本文第 3 章节中在 GEF 数据集上的预测建模实验为例，用户可采用如图 2.6 所示方式，创建继承 TaskDataset 的 gef_data 子类，在重构的“info_config”函数中指定了预测任务的归一化方法、输入步长、子时间序列数据名称等基础数据信息。继而，在重构的“sub_config”函数中，针对以“xlsx”文件格式存储的电力负荷数据，利用 pandas 数据分析工具^①，构造出输入时步

^①<https://pandas.pydata.org>.

```
1 from task.TaskLoader import TaskDataset
2 import pandas as pd
3
4 class gef_data(TaskDataset):
5     def __init__(self, args):
6         super().__init__(args)
7
8
9     def info_config(self):
10        self.info.normal = False
11        self.info.data_path = 'data/src/gef/2017_smd_hourly.xlsx'
12        self.info.series_name = ['ME', 'NH', 'VT', 'CT', 'RI', 'SEMA', 'WCMA', 'NEMA',
13                                '']
14        self.info.steps = 24*7
15
16    def sub_config(self,):
17        self.seriesPack = []
18
19        for i, name in enumerate(self.info.series_name):
20            df = pd.read_excel(self.info.data_path, sheet_name=name, index_col
21                              =None, header=0)
22            raw_ts = df['RT_Demand'].values
23            sub = self.pack_dataset(raw_ts)
24            sub.index = i
25            sub.name = name
26            sub.merge(self.info)
27
28            self.seriesPack.append(sub)
```

图 2.6 GEF 数据集数据预处理示例

为 168、提取出“ME”、“NH”、“VT”、“CT”、“RI”、“SEMA”、“WCMA”和“NEMA”共 8 个不同地区的电力负荷数据，基于 TaskDataset 父类提供的“pack_dataset”函数完成对子时间序列数据的预处理。

2.3.3 模型初始化与构造

预测模型的构造是预测框架中最为核心也是最为复杂的部分。为应对不同模型的不同参数设置与结构设置需求，高效兼容各类模型以完成参数初始化、模型构造与模型优化等流程，本框架通过基于结构模组化、输出标准化和参数接口化的设计思想，耦合模型参数接口和结构构造接口，统一预测模型输入输出标准，使得用户可针对不同需求灵活定义模型的相关参数或功能，实现模型的高效扩展，从而满足预测建模任务的需要。

```
1 import torch
2 import torch.nn as nn
3
4 class ESNCCell(nn.Module):
5     def __init__(self, init = 'vanilla', hidden_size = 500, input_dim = 1,
6         nonlinearity = 'tanh', leaky_r = 1, weight_scale = 0.9, iw_bound = (-
7         0.1, 0.1), hw_bound=(-1,1), device='cpu'):
8         super(ESNCCell, self).__init__()
9         self.input_dim = input_dim
10        self.Hidden_Size = hidden_size
11        ...
12
13        self.ih = nn.Linear(self.input_dim, self.Hidden_Size, bias=False).to(
14            self.device)
15        self.hh = nn.Linear(self.Hidden_Size, self.Hidden_Size, bias=False).
16            to(self.device)
17
18        self.weight_init(self,)
19        ...
20
21    def forward(self, current_input, last_state):
22        current_state = self.ih(current_input) + self.hh(last_state)
23        current_state = self.act_f(current_state)
24        _last_state = (1- self.leaky_r) * last_state + self.leaky_r *
25            current_state
26
27        return _last_state
28
29 ...
```

图 2.7 回声状态网络单元模组示例

2.3.3.1 结构模组化

基于 Pytorch 框架自动差分、硬件加速和扩展性强等优势^[26]，本框架对 Pytorch 框架提供的神经网络单元结构进行了封装，建立起常用的深度神经网络结构模组，进一步增强了用户构造深度神经网络预测模型的可读性、便利性与扩展性。

例如，图 2.7 展示了基于 Pytorch 框架实现的 ESN 单元模组。而后，调用 ESN 单元模组循环读取时间序列输入特征，实现出如图 2.8 所示的 ESN 编码过程模组。类似的，以本文第 5 章中所构造的回声状态卷积结构和卷积回声状态结构为例，通过如图 2.9 所示的简洁模组调用方式，即可建立起耦合卷积结构和回声状态结构的神经网络混合结构。通过本框架对深度神经网络结构的模组化，用户得以快捷构造出所需模型。


```
1 import torch
2 import torch.nn as nn
3
4 class esnLayer(nn.Module):
5     def __init__(self, init = 'vanilla', hidden_size = 500, input_dim = 1,
6         nonlinearity = 'tanh', leaky_r = 1, weight_scale = 0.9, iw_bound = (-
7         0.1, 0.1), hw_bound=(-1,1), device='cpu'):
8         super(esnLayer, self).__init__()
9         self.esnCell = ESNCell(init=init,hidden_size=hidden_size,input_dim=
10         input_dim,nonlinearity=nonlinearity,leaky_r=leaky_r,weight_scale=
11         weight_scale,iw_bound=iw_bound,hw_bound=hw_bound,device=device)
12
13     def forward(self, x, _last_state = None):
14         with torch.no_grad():
15             samples, time_steps = x.shape[0], x.shape[2]
16             layer_hidden_state = torch.empty(samples, self.esnCell.Hidden_Size
17             , time_steps).to(self.esnCell.device)
18
19             last_state = torch.zeros(samples, self.esnCell.Hidden_Size).to(
20             self.esnCell.device) if _last_state is None else _last_state.
21             detach().clone().to(self.esnCell.device)
22
23             for t in range(time_steps):
24                 last_state = self.esnCell(x[:, :, t], last_state)
25                 layer_hidden_state[:, :, t] = last_state
26
27         return layer_hidden_state, last_state
28
29 ...
```

图 2.8 回声状态网络编码过程模组示例

2.3.3.2 输出标准化

尽管不同预测建模技术所构造出的模型结构可能有较大差别，但在相同的预测任务下，这些模型的预测输出结果应当具有相同的数据结构形式。基于此，本框架通过统一预测模型的拟合过程接口，在不同预测模型方法间建立出一种标准化的输入输出规范，从而为框架的模型优化与评价功能奠定基础。

具体地，本框架通过 Python 类的形式构造具有标准输出形式的不同预测模型。以图 2.10 所展示的 ESN 模型为例，具体说明本框架的模型输出标准化思想。对于本框架所实现的预测模型，在类初始化“__init__”阶段，均会被赋予一个记录模型拟合信息的属性“fit_info”。而后，通过“xfit”函数，模型将读取数据预处理与加载环节切分出的训练集与验证集数据（对应图 2.10 中第 15 与 16 行），用于训练模型相关参数和验证模型效果（对应图 2.10 中第 17 至 24 行），并将对应的训练集误差与验证集误差保存在“fit_info”中（对应图 2.10 中第 26 与 27 行），供流程控制程序进行下

```
1 class cesLayer(nn.Module):
2     def __init__(self, cnn_hyper, esn_hyper):
3         super(cesLayer, self).__init__()
4         self.cnnLayer = cnnLayer(**cnn_hyper.dict)
5         self.esnLayer = esnLayer(**esn_hyper.dict)
6
7     def forward(self, x)
8         fm, _ = self.cnnLayer(x)
9         layer_hidden_state, last_state = self.esnLayer(fm)
10        return layer_hidden_state, last_state
11
12    ...
```

(a) 卷积回声状态结构代码示例

```
1 class esnLayer(nn.Module):
2     def __init__(self, cnn_hyper, esn_hyper):
3         super(esnLayer, self).__init__()
4         self.cnnLayer = cnnLayer(**cnn_hyper.dict)
5         self.esnLayer = esnLayer(**esn_hyper.dict)
6
7     def forward(self, x)
8         layer_hidden_state, _ = self.esnLayer(x)
9         fm, fm_flatten = self.cnnLayer(layer_hidden_state)
10        return fm, fm_flatten
11
12    ...
```

(b) 回声状态卷积结构代码示例

图 2.9 基于结构模组化设计的神经网络混合结构代码示例

一步的优化或评价工作。

2.3.3.3 参数接口化

如图 2.11 所示，本框架通过设计模型参数设置的 `Model_base` 父类，为框架内的每个模型提供 `hyper`、`tuner` 和 `tuning` 三种属性以及“`base_init`”、“`setting_init`”和“`setting_modify`”三种函数接口，将模型的结构超参数设置、优化器参数设置与参数优化结果进行单独管理。

图 2.12 展示了基于 `Model_base` 父类的 `esn_base` 子类，通过重构 `Model_base` 中的“`base_init`”函数，将该子类与对应路径下存储的 ESN 模型结构类进行匹配；通过调用与重构“`setting_init`”函数，为 ESN 模型提供了隐藏层数量为 400、激活函数类别为“`sigmoid`”、输入与隐藏层权重随机初始化分布为 $[-0.1, 0.1]$ 和 $[-1, 1]$ 、隐藏状

```
1 import torch.nn as nn
2 from task.metric import rmse
3 import ...
4
5 class EchoStateNetwork(nn.Module):
6     def __init__(self, opts=None, logger=None):
7         super(EchoStateNetwork, self).__init__()
8         self.opts = opts
9         self.logger = logger
10
11         self.fit_info = Opt()
12         ...
13
14     def xfit(self, train_data, val_data):
15         train_loader = self.data_loader(train_data)
16         val_loader = self.data_loader(val_data)
17         h_states, x, y = self.batch_transform(train_loader)
18
19         self.update_readout(h_states, x, y)
20
21         pred = self.readout(self.io_check(h_states, x)[:,-1])
22
23         _, y, pred = self.loader_pred(train_loader)
24         _, val_y, vpred = self.loader_pred(val_loader)
25
26         self.fit_info.trmse = rmse(y, pred)
27         self.fit_info.vrmse = rmse(val_y, vpred)
28
29         return self.fit_info
30
31     ...
```

图 2.10 ESN.py 文件中的 EchoStateNetwork 类示例

```
1 class Model_base(Opt):
2     def __init__(self):
3         super().__init__()
4
5         self.hyper = Opt()
6         self.tuner = Opt()
7         self.tuning = Opt()
8
9         self.base_init()
10        self.setting_init()
11        self.setting_modify()
12
13    ...
```

图 2.11 Model_base 类示例

态读取步长为 2 的基本参数。通过这种方法，用户可以通过在子类中修改“base_init”函数，使得多个不同模型共享一套参数设置，从而更好观测相同参数设置不同模型结构的预测模型性能差异。用户亦可通过共享父类中的“base_init”和“setting_init”

```
1 class esn_base(Model_base):
2     def base_init(self,):
3         self.import_path='models/stochastic/esn/ESN.py'
4         self.class_name = 'EchoStateNetwork'
5
6     def setting_init(self,):
7         self.hyper.input_dim = 1
8         self.hyper.hidden_size = 400
9         self.hyper.nonlinearity = 'sigmoid'
10        self.hyper.iw_bound = (-0.1, 0.1)
11        self.hyper.hw_bound = (-1, 1)
12        self.hyper.readout_steps = 2
13        ...
14
15    ...
```

图 2.12 继承 Model_base 类的 esn_base 子类

```
1 class fcd_esn(esn_base):
2
3     def setting_modify(self,):
4         self.hyper.readout_steps
5             = 1
6         ...
```

(a) 继承 esn_base 类的 fcd_esn 子类示例

```
1 class stk_esn(esn_base):
2
3     def setting_modify(self,):
4         self.hyper.readout_steps
5             = 168 / 2
6         ...
```

(b) 继承 esn_base 类的 stk_esn 子类示例

图 2.13 基于 esn_base 类构造的不同多输出结构模型参数示例

```
1 from ray import tune
2 from task.ModelSetting import esn_base
3
4 class fcd_esn(esn_base):
5     def __init__(self):
6         super().__init__()
7         self.tuner.algo = 'tpe'
8         self.tuner.iters = 200
9         self.tuner.resource = {
10             'cpu': 5, 'gpu': 0.5
11         }
12         self.tuning.iw_bound = tune.loguniform(1e-5, 1e-1)
13         self.tuning.weight_scaling = tune.uniform(0.2, 0.99)
14         self.tuning.nonlinearity = tune.choice(['sigmoid', 'tanh'])
15         self.tuning.hidden_size = tune.qrandint(500, 1500, 50)
16
17    ...
```

图 2.14 优化器资源与搜索参数设置示例

```
1 import ray,importlib
2 from ray import tune
3 from ray.air import session
4 from ray.tune.search.optuna import OptunaSearch
5 import ...
6
7 class TaskTuner(Opt):
8     def __init__(self, opts, subPack, logger):
9         super().__init__()
10        self.merge(opts)
11        self.points_to_evaluate = []
12        if 'points_to_evaluate' in self.tuner.dict:
13            self.points_to_evaluate = self.tuner.points_to_evaluate
14
15        self.train_data = subPack.train_data
16        self.valid_data = subPack.valid_data
17        self.batch_size = subPack.batch_size
18        self.metric = 'vrmse'
19        self.logger = logger
20        self.algo_init()
21
22    def algo_init(self,):
23        tpe_search = OptunaSearch(metric=self.metric,mode='min',
24            points_to_evaluate=self.points_to_evaluate)
25        self.algo_func = tpe_search()
26
27    def fitness(self, config):
28        _hyper = Opt(init=self.hyper)
29        _hyper.update(config)
30
31        model = importlib.import_module(self.import_path)
32        model = getattr(model, self.class_name)
33        model = model(_hyper, self.logger)
34        fit_info = model.xfit(self.train_data, self.valid_data)
35        trmse, vrmse = fit_info.trmse, fit_info.vrmse
36
37        session.report({'trmse': trmse,'vrmse': vrmse,})
38
39    def conduct(self,):
40        ray.init()
41        ray_tuner = tune.Tuner(
42            tune.with_resources(tune.with_parameters(self.fitness),resources=
43                self.tuner.resource),
44            param_space=self.tuning.dict,
45            tune_config=tune.TuneConfig(search_alg=self.algo_func,num_samples
46                =self.tuner.iters),
47        )
48
49        results = ray_tuner.fit()
50        best_result = results.get_best_result(self.metric, 'min')
51        return best_result.config
```

图 2.15 TaskTuner 类示例

函数，自定义“setting_modify”函数，实现相同模型结构不同模型参数的预测模型，从而满足预测建模实验分析的需要。以本文第 4 章节中构造基于 GEF 数据集的全连接解码器（FCD）和堆栈结构（STK）ESN 预测模型的实验为例，如图 2.13 所示，通

过重构“setting_modify”函数调整隐藏状态读取步长，利用 Python 语言中的多态形式，便捷构造出了 FCD-ESN 和 STK-ESN 预测模型。

此外，用户通过设置或修改 tuner 和 tuning 相关参数，可灵活调整优化器的计算资源分配与搜索空间，从而优化利用当前计算资源条件，满足差异化的建模任务要求。具体地，图 2.14 展示了一种优化器资源与搜索空间参数的设置示例。结合本文第 4 章中基于 AMD 5950X CPU（16 核 32 线程）与三卡 Nvidia RTX A4000 GPU 的计算条件，tuner 的参数表明：优化算法为 TPE 算法，TPE 算法最多采样次数为 200，优化过程以并行计算的方式加速进行，单个模型程序占用 5 个 CPU 线程与半张 GPU 计算卡资源，模型并发数为 6；tuning 的参数指代：ESN 模型输入权重分布参数的搜索边界为 [0.00001, 0.1]，隐藏层权重缩放参数搜索边界为 [0.2, 0.99]，激活函数的选择范围为 {sigmoid, tanh}，隐藏层神经元数量的选择范围为步进 50，边界为 500 和 1500 的数列 {500, 550, 600, ..., 1500}。

2.3.4 模型选择与优化

利用 Ray 框架所提供的丰富优化算法与并行计算机制，在前述设计思想的基础上，本框架对 Ray 框架提供的优化接口结合预测建模任务背景进行了封装，建立起适用于解决时间序列预测建模技术模型选择问题的优化平台，从而为不同预测模型提供模型选择与优化功能，进一步满足用户对于模型预测性能的要求。具体地，本章节通过如图 2.15 所示的优化器 TaskTuner 示例，详细说明本框架的模型选择与优化过程。

在 TaskTuner 类的“__init__”阶段，通过接收参数“opts”，数据集包“subPack”和实验日志“logger”完成优化器的初始化。其中，“opts”即为如图 2.14 所示构造的模型参数类，通过图 2.15 中第 12 行所示的“merge”操作，将模型的 hyper、tuner 与 tuning 等参数信息传递给 TaskTuner 类，从而确定模型初始参数、优化器参数和参数搜索空间等设置。“subPack”为如图 2.5 所示经预处理的数据文件，从而传入模型训练与验证所需的训练集与验证集数据。此外，如图 2.5 中第 22 行至 26 行，TaskTuner 类通过调用“algo_init”函数，完成以优化算法的初始化。显然，本框架用户可以通过重构“algo_init”函数，灵活修改或扩展优化算法种类，从而满足不同优化场景的

表 2.1 “OpenForecasting” 框架提供的预测模型

模型名称	参考文献
Naive	Box et al. Time Series Analysis: Forecasting and Control ^[40] .
ARIMA	Hyndman et al. Automatic Time Series Forecasting: The Forecast Package for R ^[41] .
HW	Chatfield, Chris. The Holt-Winters Forecasting Procedure ^[17] .
MSVR	Bao et al. Multi-Step-Ahead Time Series Prediction Using Multiple-Output Support Vector Regression ^[42] .
MLP	Bao et al. PSO-MISMO Modeling Strategy for MultiStep-Ahead Time Series Prediction ^[23] .
CLPSO-ML	Yang et al. Comprehensive Learning Particle Swarm Optimization Enabled Modeling Framework for Multi-Step-Ahead Influenza Prediction ^[43] .
Elman RNN	Hewamalage et al. Recurrent Neural Networks for Time Series Forecasting: Current Status and Future Directions ^[44] .
GRU	Cho et al. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches ^[45] .
LSTM	Hochreiter et al. Long Short-Term Memory ^[46] .
DeepAR	Salinas et al. DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks ^[21] .
CNN	Koprinska et al. Convolutional Neural Networks for Energy Time Series Forecasting ^[12] .
ConvRNN	Livieris et al. A CNN–LSTM Model for Gold Price Time Series Forecasting ^[22] .
RVFL	Igelnik et al. Stochastic Choice of Basis Functions in Adaptive Function Approximation and the Functional-Link net ^[18] .
IELM	Huang et al. Extreme Learning Machine: Theory and Applications ^[47] .
SCN	Wang et al. Stochastic Configuration Networks: Fundamentals and Algorithms ^[20] .
ESN	Jaeger, Herbert. The “Echo State” Approach to Analysing and Training Recurrent Neural Networks with an Erratum Note ^[48] .
DESN	Gallicchio et al. Echo State Property of Deep Reservoir Computing Networks ^[49] .
GESN	Qiao et al. Growing Echo-State Network with Multiple Subreservoirs ^[50] .
PSO-GESN	Li et al. PSO-Based Growing Echo State Network ^[51] .
Stoc-CNN	Yu et al. Impact of Random Weights on Nonlinear System Identification Using Convolutional Neural Networks ^[8] .
ESM-CNN	Zhang et al. Error-Feedback Stochastic Modeling Strategy for Time Series Forecasting with Convolutional Neural Networks ^[52] .

表 2.2 “UniV-Forecasting” 框架支持的优化算法

算法名称	参考文献
RS	Bergstra et al. Random Search for Hyper-Parameter Optimization ^[53] .
BO	Snoek et al. Practical Bayesian Optimization of Machine Learning Algorithms ^[54] .
BOHB	Falkner et al. BOHB: Robust and Efficient Hyperparameter Optimization at Scale ^[55] .
CFO	Wu et al. Frugal Optimization for Cost-Related Hyperparameters ^[56] .
BS	Wang et al. Economic Hyperparameter Optimization with Blended Search Strategy ^[57] .
PSO	Kennedy et al. Particle Swarm Optimization ^[58] .
TPE	Bergstra et al. Algorithms for Hyper-Parameter Optimization ^[59] .

需要。

TaskTuner 类的优化过程执行阶段，如图 2.5 中第 41 行至 50 行，利用前述输出标准化与参数接口化的设计思想与规范，通过对模型初始化与构造的封装，确定优化控制器 ray_tuner 的适应度 “fitness” 函数，通过模型 tuner 参数中的相关设置，完成 ray_tuner 的初始化。在如图 2.5 中第 48 行所示的优化过程中，基于模型 tuning 属性所定义参数空间 “param_space”（对应图 2.5 中第 44 行），ray_tuner 会对被优化的参数进行采样或更新得到新的参数解，这些解以 “fitness” 函数的 “config” 形式传递给模型的既有参数集合 “_hyper”，并对 “_hyper” 进行更新，而后基于更新后的参数集合构造预测模型，通过模型 “xfit” 函数所界定的标准化输入输出形式，得到模型在当前参数下的训练集误差与验证集误差。这些误差被汇报给 ray_tuner 以进行下一轮参数采样或更新过程，在完成最大迭代次数后，返回模型的最优参数。

基于本节所述的框架结构设计方式，如表 2.1 和表 2.2 所示，本框架提供了多种预测建模技术与方法，使得用户可在预测建模实验中对分析所提模型与既有模型间的性能差异，找寻不同预测模型结构或智能优化方法在所选预测任务上的优劣，从而启发出更多预测建模方法，进一步提高解决时间序列预测建模问题的能力与水平。

2.4 框架逻辑流程

图 2.16 展示了本框架的逻辑流程图。该流程图中的步骤说明如下：

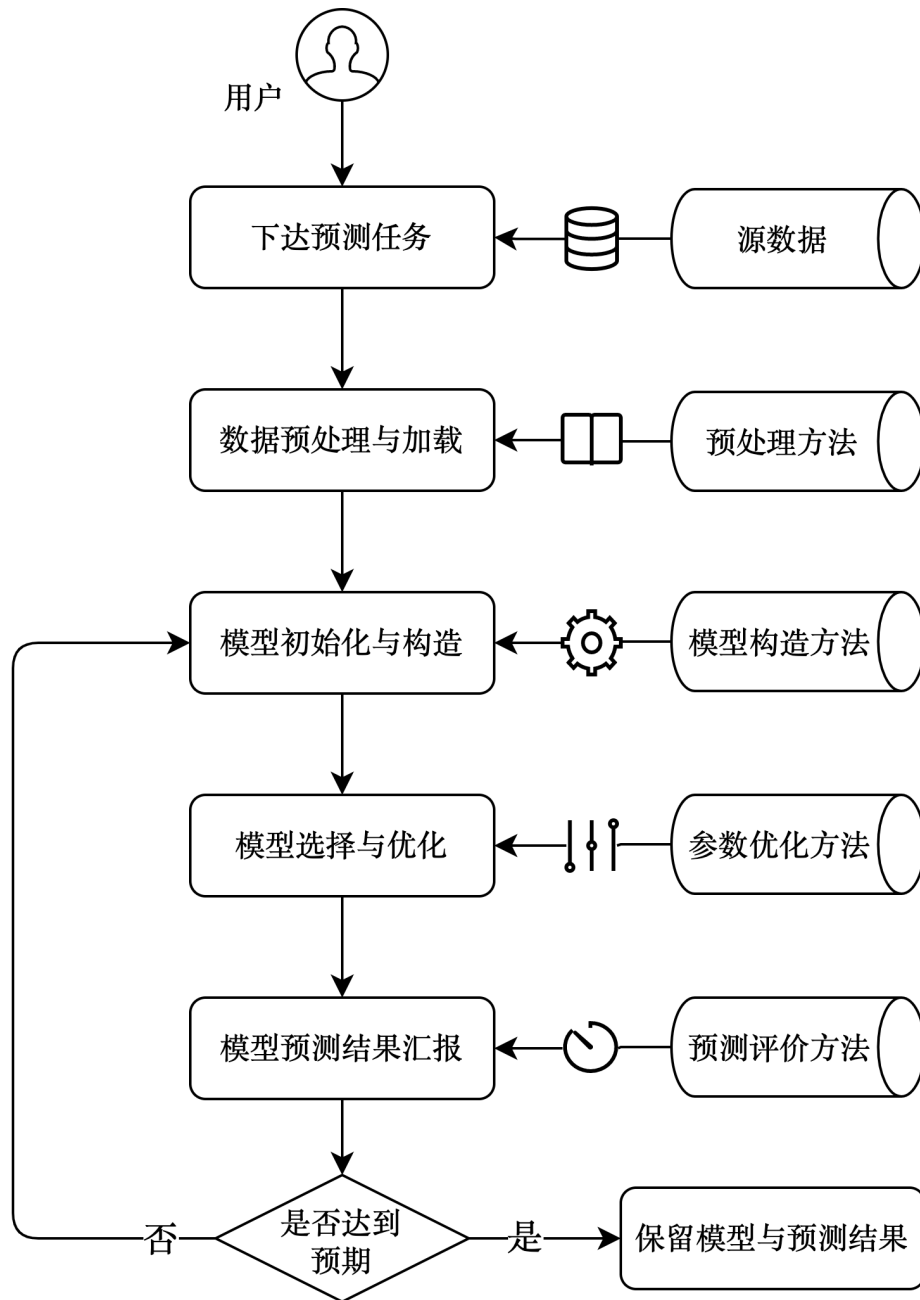


图 2.16 框架逻辑流程图

(1) 用户下达预测任务。基于不同的时间序列预测任务情形，用户收集到所需的时间序列源数据。这些源数据可以是以“txt”、“xlsx”、“csv”和“json”等格式保存的本地文件，也可以是存储在本地或云端数据库系统中的数据库文件。

(2) 数据预处理与加载。基于所收集到的源数据，定义继承 TaskDataset 父类的数据文件 Data 子类，通过重构“info_config”函数完成时间序列预测任务的基础数

据信息设定, 利用“sub_config”函数抽取出预测任务所需的时间序列数据, 并结合与“info_config”中的信息设定相匹配的预处理方法, 完成时间序列数据的异常值处理、归一化、数据集切分等预处理工作, 打包预处理数据与对应信息设定并予以加载。

(3) 模型初始化与构造。基于使用本框架目标的不同, 用户可从内置预测模型方法中选择出合适的模型或自行构造新的模型。针对所选择或设计的模型构造方法, 定义继承 Model_base 父类的模型参数 Model 子类, 通过修改“base_init”函数指定模型构造方法的保存路径与模型类名, 通过修改“setting_init”或“setting_modify”函数指定模型参数的初始化设置、模型搜索空间设置和优化器参数设置等参数信息。而后, 框架将基于已加载的数据信息和已指定的模型参数信息, 自动完成模型的初始化与构造工作。

(4) 模型选择与优化。基于已加载的训练集与验证集数据以及已指定的模型参数搜索空间和优化器参数等设置, 框架将自动分配计算资源, 调用优化器高效进行模型选择过程, 完成模型参数的优化。该优化过程不仅包含模型隐藏层神经元数量、激活函数种类和隐藏层层数等模型结构参数, 通过本文第 5 章节所提出的二重特征结构及其编码表示方法, 多输入特征选择与优化亦被囊括在内。同时, 用户可重构 TaskTuner 类的“algo_init”函数, 灵活选取不同的优化算法, 进一步提高模型预测性能。

(5) 模型预测结果汇报。基于已加载的测试集数据和已优化的预测模型, 框架将基于所设置的重复试验次数与指定的预测准确度指标, 自动完成模型测试, 并整理模型在当前设置下的预测结果, 详细汇报出各预测准确度指标的平均值、标准差、时步误差曲线等信息, 同时提供 Naive 方法的测试结果以做基准参考。若未达预期, 则可通过修改模型结构设计、参数搜索空间和优化算法选择等设置, 继续调整模型直到符合预期, 最终保留模型与预测结果。

2.5 小结

面对复杂多变的时间序列预测需求, 为提升时间序列预测建模技术的集成水平与运行效率, 本章节在已有优秀模型构造框架与参数优化框架的基础上, 设计并开

源了一套兼具高集成度与强扩展性的时间序列预测建模框架“OpenForecasting”。通过对时间序列预测建模框架的用户需求分析，归纳出包含数据管理功能、模型构造功能与模型优化功能在内的核心需求功能，完成了时间序列预测建模框架的系统结构与逻辑流程设计。基于此，本框架内置了不同类别的多种优秀预测模型方法，通过良好的接口设计规范，使得用户能够灵活处理不同结构的时间序列源数据，设计模型结构，调整模型参数和选择优化算法，从而更好解决时间序列预测问题。

致 谢

在博士研究生经历中，有过许多挑战和成长。回想一路走来，有过研究时的迷惘，有过发表时的兴奋，也有过展望时的徘徊，感慨万千。在此，向所有的支持者表示感谢。

.....

参考文献

- [1] O. B. Sezer, A. M. Ozbayoglu. Algorithmic Financial Trading with Deep Convolutional Neural Networks: Time Series to Image Conversion Approach. *Applied Soft Computing*, 2018, 70: 525–538
- [2] S. Cavalli, M. Amoretti. CNN-Based Multivariate Data Analysis for Bitcoin Trend Prediction. *Applied Soft Computing*, 2021, 101: 107065
- [3] H. J. Sadaei, P. C. d. L. e Silva, F. G. Guimarães, et al. Short-Term Load Forecasting by Using a Combined Method of Convolutional Neural Networks and Fuzzy Time Series. *Energy*, 2019, 175: 365–377
- [4] T. Elsken, J. H. Metzen, F. Hutter. Neural Architecture Search: A Survey. *Journal of Machine Learning Research*, 2019, 20: 1–21
- [5] A. Zela, A. Klein, S. Falkner, et al. Towards Automated Deep Learning: Efficient Joint Neural Architecture and Hyperparameter Search. in: *International Conference on Machine Learning Workshop on AutoML*. 2018
- [6] K. He, Y. Wang, J. Hopcroft. A Powerful Generative Model Using Random Weights for the Deep Image Representation. in: *Advances in Neural Information Processing Systems*. 2016: 631–639
- [7] J. M. Antognini, M. Hoffman, R. J. Weiss. Audio Texture Synthesis with Random Neural Networks: Improving Diversity and Quality. in: *IEEE International Conference on Acoustics, Speech and Signal Processing*. 2019: 3587–3591
- [8] W. Yu, M. Pacheco. Impact of Random Weights on Nonlinear System Identification Using Convolutional Neural Networks. *Information Sciences*, 2019, 477: 1–14
- [9] Y. LeCun, Y. Bengio, et al. Convolutional Networks for Images, Speech, and Time Series. *The Handbook of Brain Theory and Neural Networks*, 1995, 3361(10): 1995
- [10] J. Yang, M. N. Nguyen, P. P. San, et al. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. in: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. 2015: 3995–4001
- [11] B. Zhao, H. Lu, S. Chen, et al. Convolutional Neural Networks for Time Series Classification. *Journal of Systems Engineering and Electronics*, 2017, 28(1): 162–169
- [12] I. Koprinska, D. Wu, Z. Wang. Convolutional Neural Networks for Energy Time Series Forecasting. in: *International Joint Conference on Neural Networks*. 2018: 1–8

- [13] C. R. Vogel. Computational Methods for Inverse Problems. 2002
- [14] M. Qi, G. P. Zhang. Trend Time-Series Modeling and Forecasting with Neural Networks. IEEE Transactions on Neural Networks, 2008, 19(5): 808–816
- [15] S. F. Crone, S. Häger. Feature Selection of Autoregressive Neural Network Inputs for Trend Time Series Forecasting. in: International Joint Conference on Neural Networks. 2016: 1515–1522
- [16] J. J. Flores, M. Graff, H. Rodriguez. Evolutive Design of ARMA and ANN Models for Time Series Forecasting. Renewable Energy, 2012, 44: 225–230
- [17] C. Chatfield. The Holt-Winters Forecasting Procedure. Journal of the Royal Statistical Society: Series C (Applied Statistics), 1978, 27(3): 264–279
- [18] B. Igel'nik, Y.-H. Pao. Stochastic Choice of Basis Functions in Adaptive Function Approximation and the Functional-Link Net. IEEE Transactions on Neural Networks, 1995, 6(6): 1320–1329
- [19] G.-B. Huang, L. Chen, C. K. Siew, et al. Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes. IEEE Transactions on Neural Networks, 2006, 17(4): 879–892
- [20] D. Wang, M. Li. Stochastic Configuration Networks: Fundamentals and Algorithms. IEEE Transactions on Cybernetics, 2017, 47(10): 3466–3479
- [21] D. Salinas, V. Flunkert, J. Gasthaus, et al. DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. International Journal of Forecasting, 2020, 36(3): 1181–1191
- [22] I. E. Livieris, E. Pintelas, P. Pintelas. A CNN-LSTM Model for Gold Price Time-Series Forecasting. Neural Computing and Applications, 2020, 32: 17351–17360
- [23] Y. Bao, T. Xiong, Z. Hu. PSO-MISMO Modeling Strategy for MultiStep-Ahead Time Series Prediction. IEEE Transactions on Cybernetics, 2014, 44(5): 655–668
- [24] T. Guo, Z. Xu, X. Yao, et al. Robust Online Time Series Prediction with Recurrent Neural Networks. in: IEEE International Conference on Data Science and Advanced Analytics. 2016: 816–825
- [25] Y. Zhao, J. Li, L. Yu. A Deep Learning Ensemble Approach for Crude Oil Price Forecasting. Energy Economics, 2017, 66: 9–16
- [26] A. Paszke, S. Gross, F. Massa, et al. Pytorch: An Imperative Style, High-Performance Deep Learning Library. in: Advances in Neural Information Processing Systems. 2019: 8026–8037

- [27] Y.-X. Wu, Q.-B. Wu, J.-Q. Zhu. Improved EEMD-Based Crude Oil Price Forecasting Using LSTM Networks. *Physica A: Statistical Mechanics and its Applications*, 2019, 516: 114–124
- [28] O. B. Sezer, M. U. Gudelek, A. M. Ozbayoglu. Financial Time Series Forecasting with Deep Learning : A Systematic Literature Review: 2005–2019. *Applied Soft Computing*, 2020, 90: 106181
- [29] M. Cai, M. Pipattanasomporn, S. Rahman. Day-Ahead Building-Level Load Forecasts Using Deep Learning vs. Traditional Time-Series Techniques. *Applied Energy*, 2019, 236: 1078–1088
- [30] K. B. Lindberg, P. Seljom, H. Madsen, et al. Long-Term Electricity Load Forecasting: Current and Future Trends. *Utilities Policy*, 2019, 58: 102–119
- [31] H. Shi, M. Xu, R. Li. Deep Learning for Household Load Forecasting—A Novel Pooling Deep RNN. *IEEE Transactions on Smart Grid*, 2018, 9(5): 5271–5280
- [32] N. Laptev, J. Yosinski, L. E. Li, et al. Time-Series Extreme Event Forecasting with Neural Networks at Uber. in: *International Conference on Machine Learning*. 2017: 1–5
- [33] J. Wang, Q. Gu, J. Wu, et al. Traffic Speed Prediction and Congestion Source Exploration: A Deep Learning Method. in: *IEEE 16th International Conference on Data Mining*. 2016: 499–508
- [34] J. Xiao, Z. Xiao, D. Wang, et al. Short-Term Traffic Volume Prediction by Ensemble Learning in Concept Drifting Environments. *Knowledge-Based Systems*, 2019, 164: 213–225
- [35] M. Abadi, P. Barham, J. Chen, et al. Tensorflow: A System for Large-Scale Machine Learning. in: *12th USENIX Symposium on Operating Systems Design and Implementation*. 2016: 265–283
- [36] F. Seide, A. Agarwal. CNTK: Microsoft’s Open-Source Deep-Learning Toolkit. in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016: 2135–2135
- [37] P. Bennet, C. Doerr, A. Moreau, et al. Nevergrad: Black-Box Optimization Platform. *ACM SIGEVOlution*, 2021, 14(1): 8–15
- [38] T. Akiba, S. Sano, T. Yanase, et al. Optuna: A next-Generation Hyperparameter Optimization Framework. in: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019: 2623–2631

- [39] R. Liaw, E. Liang, R. Nishihara, et al. Tune: A Research Platform for Distributed Model Selection and Training. in: International Conference on Machine Learning. 2018
- [40] G. E. Box, G. M. Jenkins, G. C. Reinsel. Time Series Analysis: Forecasting and Control. 2015
- [41] R. J. Hyndman, Y. Khandakar. Automatic Time Series Forecasting: The Forecast Package for R. Journal of Statistical Software, 2008, 27: 1–22
- [42] Y. Bao, T. Xiong, Z. Hu. Multi-Step-Ahead Time Series Prediction Using Multiple-Output Support Vector Regression. Neurocomputing, 2014, 129: 482–493
- [43] S. Yang, Y. Bao. Comprehensive Learning Particle Swarm Optimization Enabled Modeling Framework for Multi-Step-Ahead Influenza Prediction. Applied Soft Computing, 2021, 113: 107994
- [44] H. Hewamalage, C. Bergmeir, K. Bandara. Recurrent Neural Networks for Time Series Forecasting: Current Status and Future Directions. International Journal of Forecasting, 2021, 37(1): 388–427
- [45] K. Cho, B. van Merriënboer, D. Bahdanau, et al. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. Proceedings of SSST EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, 2014, : 103–111
- [46] S. Hochreiter, J. Schmidhuber. Long Short-Term Memory. Neural Computation, 1997, 9(8): 1735–1780
- [47] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew. Extreme Learning Machine: Theory and Applications. Neurocomputing, 2006, 70(1): 489–501
- [48] H. Jaeger. The “Echo State” Approach to Analysing and Training Recurrent Neural Networks-with an Erratum Note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report, 2001, 148(34): 13
- [49] C. Gallicchio, A. Micheli. Echo State Property of Deep Reservoir Computing Networks. Cognitive Computation, 2017, 9(3): 337–350
- [50] J. Qiao, F. Li, H. Han, et al. Growing Echo-State Network with Multiple Subreservoirs. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(2): 391–404
- [51] Y. Li, F. Li. PSO-Based Growing Echo State Network. Applied Soft Computing, 2019, 85: 105774

- [52] X. Zhang, K. He, Y. Bao. Error-Feedback Stochastic Modeling Strategy for Time Series Forecasting with Convolutional Neural Networks. *Neurocomputing*, 2021, 459: 234–248
- [53] J. Bergstra, Y. Bengio. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 2012, 13(2)
- [54] J. Snoek, H. Larochelle, R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. in: *Advances in Neural Information Processing Systems*. 2012: 2960–2968
- [55] S. Falkner, A. Klein, F. Hutter. BOHB: Robust and Efficient Hyperparameter Optimization at Scale. in: *International Conference on Machine Learning*. 2018: 1437–1446
- [56] Q. Wu, C. Wang, S. Huang. Frugal Optimization for Cost-Related Hyperparameters. in: *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*. 2021: 10347–10354
- [57] C. Wang, Q. Wu, S. Huang, et al. Economic Hyperparameter Optimization with Blended Search Strategy. in: *International Conference on Learning Representations*. 2020
- [58] J. Kennedy, R. Eberhart. Particle Swarm Optimization. in: *International Conference on Neural Networks*. 1995: 1942–1948
- [59] J. Bergstra, R. Bardenet, Y. Bengio, et al. Algorithms for Hyper-Parameter Optimization. in: *Advances in Neural Information Processing Systems*. 2011: 2546–2554

附录 1 攻读博士学位期间取得的研究成果

已发表论文:

- [1] **Xinze Zhang**, Kun He, and Yukun Bao. Error-feedback Stochastic Modeling Strategy for Time Series Forecasting with Convolutional Neural Networks. *Neurocomputing*, 2021, 459:234-248.(SCI 源刊, IF 5.719, 署名单位: 华中科技大学)
- [2] Jianhua Yang, **Xinze Zhang**, and Yukun Bao. Short-term Load Forecasting of Central China based on DPSO-LSTM. In *Proceedings of IEEE 4th International Electrical and Energy Conference, IEEC 2021*. (EI 会议, 署名单位: 华中科技大学)
- [3] **Xinze Zhang**, Junzhe Zhang, Zhenhua Chen, and Kun He. Crafting Adversarial Examples for Neural Machine Translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, ACL 2021*. (CCF A 类国际会议, 署名单位: 华中科技大学)

工作论文:

- [1] **Xinze Zhang**, Kun He, Yukun Bao, and Qi Sima. Error-feedback Triple-phase Optimization to Configurable Convolutional Echo State Network for Time Series Forecasting.
- [2] **Xinze Zhang**, Qi Sima, Kun He, Yukun Bao, and Shuhan Chen. Enhancing Echo State Network with Particle Swarm Bayesian Optimization Enabled Echo State Selection for Time Series Forecasting.
- [3] Qi Sima, **Xinze Zhang**, and Yukun Bao. Reinforced Decoder: Towards Training Sequence-to-Sequence Model for Time Series Forecasting.

附录 2 已发表论文与博士学位论文的关系

序号	成果名称	成果形式	成果主要内容	与学位论文 对应关系
1	Error-feedback Stochastic Modeling Strategy for Time Series Forecasting with Convolutional Neural Networks	SCI 期刊 (已发表)	提出了误差反馈随机建模的迭代构造策略, 结合贪心搜索自适应确定卷积结构, 构造出一种新颖、高效且准确的随机卷积神经网络预测模型	该成果为论文第 3 章主要内容
2	Short-term Load Forecasting of Central China based on DPSO-LSTM	EI 会议 (已发表)	提出了时步多维度的时序输入特征二维结构, 结合粒子群优化算法选择时步维度与模型参数, 提升了 LSTM 模型的短期电力负荷预测效果	基于对该成果的改进, 构成论文第 5 章主要内容

附录3 攻读博士学位期间参与的科研项目

1. 国家自然科学基金面上项目：大数据环境下基于计算智能的预测建模技术及其在电力负荷预测中的应用，批准年限：2019/01 - 2022/12。
2. 国家自然科学基金面上项目：自然语言处理深度模型的对抗攻防关键算法研究，批准年限：2021/01 - 2024/12。
3. 湖北省国际科技合作项目：深度学习模型对抗攻防基础理论与算法研究，批准年限：2021/09 - 2023/09。
4. 国家电网公司科技项目：华中区域共享型电力交易与服务平台关键技术研究，批准年限：2020/06 - 2021/11。
5. 国家电网公司科技项目：渝鄂直流运行条件下华中消纳西南水电交易电量库模型建立与效益分析，批准年限：2019/05 - 2020/11。