2022 年中国海洋大学数学建模竞赛

参赛队员信息

序号	学号	姓名	学院	专业	年级	联系电话	电子邮箱
	20110011006	高旭乐	管理学院	工商管理	2020级	13513532867	1635753793@qq.com
	20020006153	赵亮	信息科学与工程学部	微电子科学与工程	2020级	13061474706	thebestzhaoliang@gmail.com
	19090021001	陈佩璇	工程学院	工业设计	2019级	15079868306	chenpeixuan0824@163.com

基于聚类和长短期记忆网络的电力负荷预测

摘要

科学的预测是正确决策的依据和保证，电力系统负荷预测是电力系统的规划、营销、市场交易、调度等部门工作的重要依据。如今，在“双碳”政策下，电力将从过去的二次能源转变为各行业事实上的“基础能源”，维持电网电能供需的实时平衡尤为重要。精准的电力负荷预测是电网调度优化的必要条件，因此，本文通过分析电力负荷、气象、社会、经济等历史数据，探索电力负荷历史数据的变化规律，以此为基础建立了 K 均值聚类和 LSTM 模型来解决电力系统中短期负荷预测问题。

针对问题一，对附件 1、附件 3 的数据进行预处理，探索电力负荷变化的周期性与规律性，分析各个因素对用电负荷的影响。对历史数据样本进行特征提取，并将其划分为训练集与测试集，构建 K 均值聚类预测模型，利用 Python 对模型进行求解，得出该地区电网未来十天间隔 15 分钟负荷预测结果以及未来三个月日负荷最大值和最小值预测结果，并利用 RMSE、MAE、MAPE、MSE 等指标对预测精度作出评价。

针对问题二，计算附件 2 中各行业用电负荷数据的一阶差分，以 3%和 97%分位数为标准，挖掘各行业用电负荷突变时间、量级，对照附件 3 对突变日类型进行分析，得出各行业用电突变的可能原因。另外，对各行业用电负荷历史数据样本进行特征提取，划分训练集与测试集，利用 LSTM 预测模型对该地区各行业未来三个月日负荷最大值与最小值进行预测，并对预测精度做出分析。

最后，通过对各行业目前实际状况的分析，本文指出“双碳”政策对未来各行业用电负荷可能产生的影响，并针对工业、商业以及电力行业未来的发展提出可行性建议。

关键词：中短期负荷预测；LSTM；K 均值聚类；双碳

1. 问题重述

1.1 问题背景

准确的电力负荷预测结果有助于发电厂合理地调度发电量，安排发电机组的起停，提高发电设备利用率，降低发电成本,为各类用户提供经济、可靠和高质量的电能，满足用户对负荷需求量与负荷特性的要求。随着新型电力系统的不断发展和市场化消费程度的不断提高，传统时间序列分析方法难以学习短期电力负荷数据的非线性特征。因此，不断改进电力负荷预测技术，探索精度更高的预测模型，是当前电力体制改革大背景下一个具有巨大现实意义的的课题。

1.2 问题要求

本题提供了三个附件，其中附件一提供了某地区电网间隔 15 分钟的负荷数据，附件二给出了各个行业日负荷数据，附件三列出了该地区主要的气象数据，包括天气状况，气温状况以及相关风力风向的信息。

电力系统负荷易受气象情况、社会事件和时代背景等不确定因素的影响，制定科学的预测方案尤为重要。

因此，本题要求建立数学模型，解决以下问题;

1. 考虑气象条件、日类型等因素，预测该地区电网未来 10 天每隔 15 分钟的电力负荷情况及未来三个月的日负荷的最大值、最小值情况，分析预测精度。

2. 对附件二中的数据进行处理分析，探索该地区各行业用电突变时的时间、量级和原因。

3. 预测该地区各行业未来 3 个月日负荷的最大值、最小值，分析预测精度。

4. 考虑各行业的自身情况，研究当前国家“双碳”目标的时代背景对其未来用电的可能影响并提出建议。

2. 问题分析

2.1 问题一

问题一要求处理分析附件中提供的某地区电网间隔 15 分钟的负荷数据，建立数学模型，预测该地区电网未来 10 天间隔 15 分钟的负荷情况以及未来三个月日负荷的最大值与最小值情况。首先可利用 Excel 软件处理数据，进行缺失值填补及异常值修复。得到较为合理的数据后，分析数据特点，结合实际我们建立 K 均值聚类模型对所需数据进行预测。

模型：K 均值聚类

通过对附件中数据的分析，可知电力系统的负荷情况易受气象条件、日类型等因素影响，且相似条件下的电力负荷情况类似。因此我们通过 Excel 软件处理数据，将每天的最高温度、最低温度、白天与夜晚的风向、日类型等数据处理为若干个特征值。基于这些特征值，建立 K 均值聚类模型，将数据集分为若干个簇，使各个簇之内的数据特征最为相似，而各个簇之间的数据特征相似度差别尽可能大。然后通过收集未来数据的各项特征，将其分配到对应的簇中，以此实现对未来的电力负荷情况的预测。

2.2 问题二

第一问，对附件中不同行业的用电负荷情况分析可知，各行业的用电负荷存在突变情况，要求分析说明突变的时间、量级及可能原因。首先，利用

Matlab 软件对各行业用电负荷的数据进行处理，可求得数据突变的时间与量级。其次，分析可得，各个行业的用电负荷情况主要受日类型、气象条件等因素的影响，因此我们将考虑上述主要影响因素，对突变的可能原因进行阐述。

第二问，要求建立数学模型，对大工业、非普工业、普通工业和商业的用电负荷进行预测。为了更好地计算出各个影响因素与用电情况之间的相关性，排除相关性较弱的影响因素，我们尝试构建 LSTM 神经网络对该地区各行业未来三个月的日负荷情况分别进行预测，并通过 MAPE、MAE、MSE 等评价指标对预测精度做出分析。

模型：LSTM 神经网络

作为智能预测方法之一，LSTM 神经网络具有较强的深度学习能力，因此可应用于电力负荷预测之中。首先，可对电力负荷特性进行分析并进行特征提取。通过对影响电力负荷内外部因素进行相关度分析，可筛选出与该地电力负荷变化相关性较强的影响因素，并提取其作为预测的输入特征组。然后以该输入特征组作为输入，待预测负荷作为输出，建立 LSTM 神经网络预测模型，从而获取对未来的电力负荷情况的预测结果。

第三问，要求研究国家“双碳”目标对各行业未来用电负荷的可能影响并提出建议。因此，我们将通过搜集资料，了解国家“双碳”目标的内容及目的，从各行业的实际情况出发，分析“双碳”对其未来用电的可能影响并从多方面提出合理且有效的建议。

3. 数据预处理

电力负荷数据的好坏直接影响着电力负荷预测的准确性，尽管通过智能电表采集数据相较于人工记录更加规范有效，但是仍然不能避免数据出现缺失或异常的情况，如电表故障或拉闸限电等。因此，需要对原始数据做预处理，如数据的缺失值填补、异常值鉴别与修正以及大数量级的数据归一化等。

3.1 缺失值查找

附件一与附件二中的数据都是某地区电网某段连续时间的负荷数据，首先可初步计算出该连续时间段内应有的数据数量，判断数据有无缺失。若有，则利用 Python 生成该时间段内完整的时间序列，与表中提取出的序列作比较，找出所有缺失数据。

3.2 缺失值填补

根据日类型相同原则对电力负荷数据缺失值进行填补，工作日的负荷数据由当日相邻时间和相邻工作日的数据修复；休息日缺失值由当日相邻时间和相邻休息日的数据修复，公式如下：

$$Y(d,t) = \omega_1 Y(d,t-1) + \omega_2 Y(d,t+1) + \omega_3 Y(d-1,t) + \omega_4 Y(d+1,t) \quad (3.1)$$

（取$\omega_1 = \omega_2 = \omega_3 = \omega_4 = 0.25$）

3.3 数据异常点识别与修正

由于气象条件突变、社会突发性事件、拉闸限电等原因会导致系统负荷数据发生异常突变。同时，电表在采集数据时也可能收到外界因素干扰，使得某些时间点负荷数据与真实值相差较大，因此对异常值进行如下处理：

（1）水平处理法[1]：某一时间的负荷数据和前后相邻的数据相差不会太大，若当前时刻负荷与前一时刻负荷存在以下关系：

$$|Y(d,t) - Y(d,t-1)| > \varepsilon(t) \quad (3.2)$$

首先取$Y(d,t)$相邻的平均数得到$Y_1(d,t)$:

$$Y_1(d,t)=\frac{Y(d,t-1)+Y(d,t+1)}{2} \tag{3.3}$$

接着取$Y(d,t)$相邻 5 个数的平均值得到

$$Y_2(d,t)=\frac{Y(d,t-2)+Y(d,t-1)+Y_1(d,t)+Y(d,t+1)+Y(d,t+2)}{5} \tag{3.4}$$

然后取$Y(d,t)$相邻 3 个数的平均值得到$Y_3(d,t)$:

$$Y_3(d,t)=\frac{Y(d,t-1)+Y_2(d,t)+Y(d,t+1)}{3} \tag{3.5}$$

最后经加权平均即可得到水平处理后的电力负荷值Y(d, t)：

$$Y(d,t)=\omega_1 Y_1(d,t)+\omega_2 Y_2(d,t)+\omega_3 Y_3(d,t) \tag{3.6}$$

取ω_1=0.3，ω_2=0.5，ω_3=0.2

（2）垂直处理法[1]：电力负荷特性的变化具有周期性，相邻工作日或休息日的同一时间点的电力负荷值应相差不大，若存在以下关系：

$$|Y(d,t)-M(t)|>r(t) \tag{3.7}$$

则处理后的负荷数据：

$$Y(d,t)=\begin{cases}M(t)+r(t), Y(d,t)>M(t)\\ M(t)-r(t), Y(d,t)<M(t)\end{cases} \tag{3.8}$$

4. 模型假设与符号说明

4.1 模型假设

●设样本容量足够大，2018 年 1 月 1 日至 2021 年 8 月 31 日的电力符合数据均真实有效，能够反映具体情况；

●文中所引用的文献和结论均正确且可靠；

●不考虑自然灾害，极度特殊天气、战争等突发事件等因素的影响；

●对缺失值、异常值的判断处理正确，神经网络训练期间，“坏数据”带来的训练误差，不会使网络不能收敛到理想效果；

●对未来几天的特征输入较为符合实际情况。

4.2 符号说明

符号	含义
x_i	第i个样本的特征向量
d_{ij}	第i个样本和第j个样本特征向量的欧式距离
S_{ij}	第i个样本和第j个样本的相似度
C	聚类中心构成的集合
G	簇
x_t	LSTM 的输入
f_t	遗忘门
i_t	输入门
o_t	输出门
h_t	隐藏状态

C_t	单位状态
W, b	训练中的矩阵，网络单元学习元值

5. 模型的建立与求解

5.1 模型建立前的准备

电力系统负荷预测是根据电力负荷、经济、社会、气象等的历史数据，探索电力负荷历史数据变化规律对未来负荷的影响，寻求电力负荷与各种相关因素之间的内在联系，从而对未来的电力负荷进行科学的预测。因此，在模型建立前，我们先对电力负荷历史数据变化规律进行探索。

通过 MATLAB 作图，发现区域一天内用电负荷大致呈如下趋势。

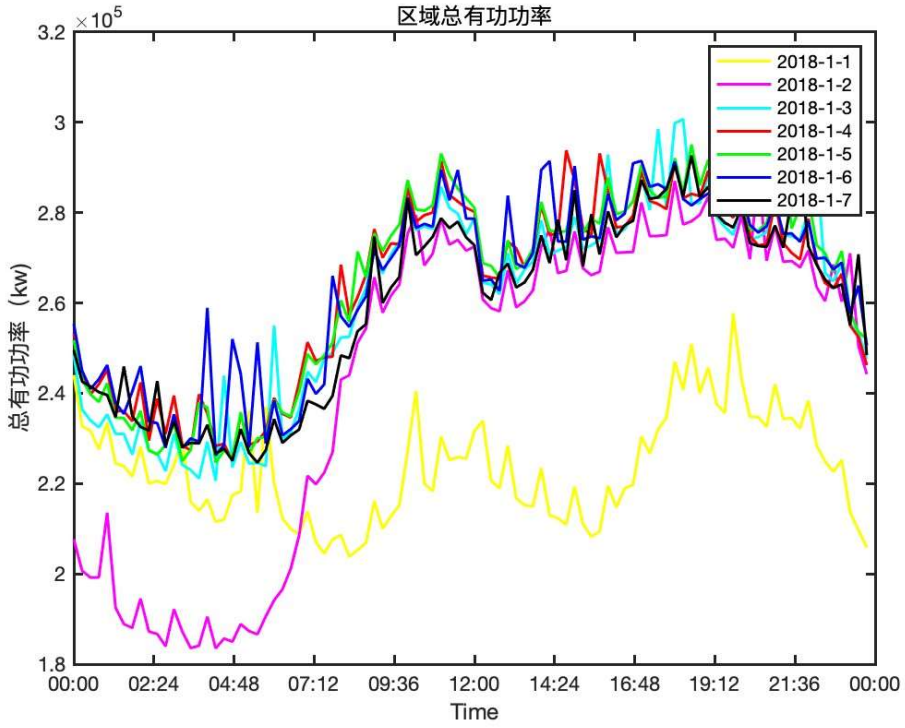


图 1 2018-1-1 至 2018-1-7 区域每 15 分钟总有功功率曲线

此外，用电负荷具有一定的周期性，如下图:

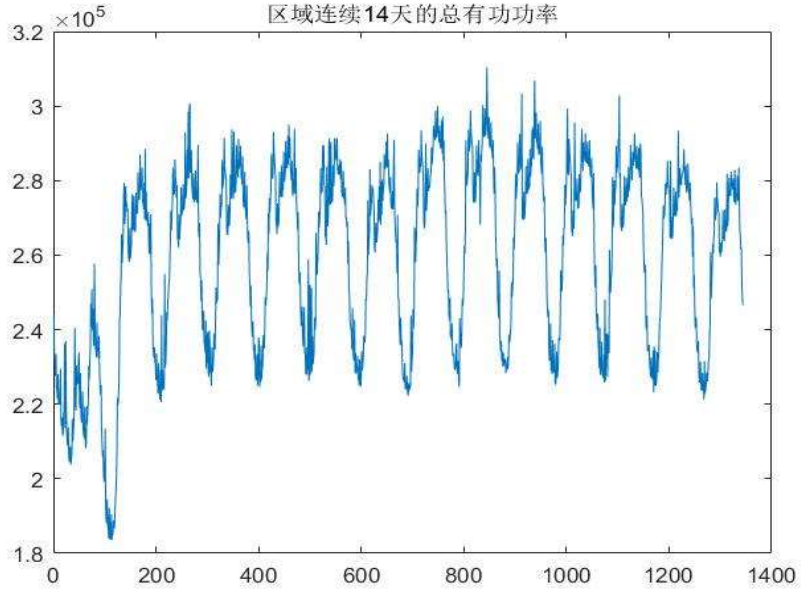


图 2 区域内连续 14 天的总有功功率曲线

显然，用电量有明显的波动特征，且其与温度有明显的相关性，当温度升高时，用电量也处于升高的阶段；当温度降低时，用电量也处于降低的阶段。两者变化的峰和谷也基本重合。

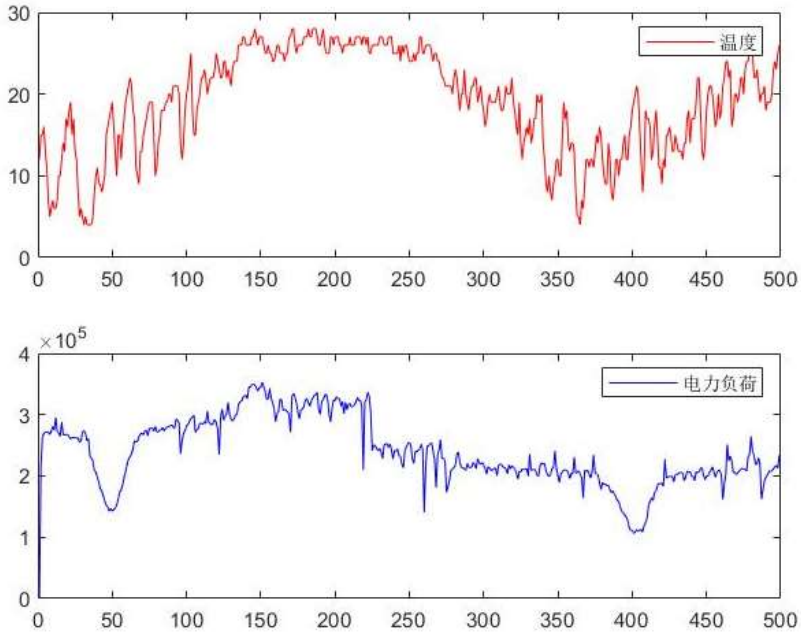


图 3 2018-2021 年当地温度与工业每日用电总负荷的曲线

最后，我们引入预测精度评价指标，为之后的模型评价提供依据。

假设预测值: $\hat{y}=\{\hat{y}_1,\hat{y}_2,\cdots\hat{y}_n\}$,真实值： $y=\{y_1,y_2,\cdots y_n\}$,则有

（1）均方误差

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{5.1}$$

（2）均方根误差

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{5.2}$$

（3）平均绝对百分比误差

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \tag{5.3}$$

（4）平均绝对误差

$$MAE = \frac{100\%\sum_{i=1}^{n}|\hat{y}_i - y_i|}{n} \tag{5.4}$$

5.2 K 均值聚类预测模型

在预测活动中，可以根据预测对象与类似已知事物的发展状况进行类比，更可以与其历史发展规律进行类比，从而推知对象的未来发展规律。在电力负荷预测中也可使用该种技术。例如，各年春节期间的日负荷曲线往往表现出彼此相同、但与其他日负荷曲线完全不同的形态，因此，节假日曲线形状的预测，可以参照往年的情况得出预测结果。

聚类的目的是把具有相同特征的对象分成一类，而把具有差异的对象尽可能分开，从而实现无监督式的学习。

5.2.1 聚类问题定义

设有n个样品的p元观测数据组成一个数据矩阵

$$X = \begin{bmatrix} x_{11}, x_{12}, \cdots x_{1p} \\ x_{21}, x_{22}, \quad \cdots x_{2p} \\ \vdots \\ x_{n1}, x_{n2}, \cdots x_{np} \end{bmatrix} \tag{5.5}$$

其中，每一列表示一个样品，每一列表示一个属性，x_{ij}表示第i个样品的第j项属性的观测值。

设第i个样品的p元观测数据

$$x_i = \left(x_{i1}, x_{i2}, \cdots x_{ip}\right)^T (i = 1,2 \cdots n) \tag{5.6}$$

每个样品可认为是p元空间的一个点，即一个p维向量，两个向量之间的距离记为$d(x_i, x_j)$, 使用欧氏距离，即

$$d\left(x_i, x_j\right) = \left|\sum_{k=1}^{p}\left(x_{ik} - x_{jk}\right)^2\right|^{\frac{1}{2}} \tag{5.7}$$

以此距离作为衡量两个样本相似度的指标，若两个样本的欧式距离小，则表明

它们之间关系密切，属同一类型，否则，属于不同类型。

因此，定义两个样本之间的相似度指标S_{ij}

$$S_{ij} = \frac{1}{d(x_i, x_j)} \tag{5.8}$$

相似度越大，表明两个样本类型越接近。

5.2.2 K 均值聚类算法[2]

根据实际类型确定分类数k，在每一类中选择有代表性的样品，称为聚点，一般按最小最大的原则选取。若将n个样品分成k类，需选择所有样品中彼此相隔最远的两个样品x_{i_1}, x_{i_2}为聚点，使得

$$\mathrm{d}(x_{i_1}, x_{i_2}) = \mathrm{d}_{\mathrm{i_1 i_2}} = \max\{\mathrm{d_{ij}}\} \tag{5.9}$$

接着选择第三个聚类点x_{i_3}, 使得x_{i_3}与前两个聚点的相隔距离最小者等于所有其余的与x_{i_1}, x_{i_2}的较小相隔距离中最大的，即

$$\min\{\mathrm{d}(\mathrm{x_{i_3}}, \mathrm{x_{i_r}}), \mathrm{r} = 1,2\} = \max\{\min\{\mathrm{d}(\mathrm{x_j}, \mathrm{x_{i_r}}), \mathrm{r} = 1,2\}, \mathrm{j} \neq \mathrm{i_1}, \mathrm{i_2}\} \tag{5.10}$$

最后按照相同的方法选取x_{i_k},重复前面的步骤，直至确定k个聚点$x_{i_1}, x_{i_2}, \cdots x_{i_k}$。K 均值聚类的步骤如下：

（1）设第k个初始聚点的集合是$L^{(0)} = \left\{x_1^{(0)}, x_2^{(0)}, \cdots x_k^{(0)}\right\}$,

记$G_i^{(0)} = \left\{x: d\left(x, x_i^{(0)}\right) \leq d(x, x_j^{(0)}), j = 1,2, \cdots k, j \neq i\right\} (i = 1,2, \cdots k)$,

于是，将样品分成不相交的k类，得到一个初始分类

$$G^{(0)} = \left\{G_1^{(0)}, G_2^{(0)}, \cdots G_k^{(0)}\right\} \tag{5.11}$$

（2）从初始类$G^{(0)}$开始计算新的聚类集合$L^{(1)}$,计算

$$x_i^{(1)} = \frac{1}{n_i} \sum_{x_i \in G_i^{(0)}} x_i \ \ (i = 1,2, \cdots k) \tag{5.12}$$

其中n_i是类$G^{(0)}$中的样本数，得到一个新的集合

$$L^{(1)} = \left\{x_1^{(1)}, x_2^{(1)}, \cdots x_k^{(1)}\right\} \tag{5.13}$$

从$L^{(1)}$开始再进行分类，记

$$G_i^{(1)} = \left\{x: d\left(x, x_i^{(1)}\right) \leq d\left(x, x_j^{(1)}\right), j = 1,2, \cdots k, j \neq k\right\} \ (i = 1,2, \cdots k) \tag{5.14}$$

得到一个新的分类

$$G^{(1)} = \left\{G_1^{(1)}, G_2^{(1)}, \cdots G_k^{(1)}\right\} \tag{5.15}$$

（3）重复以上步骤m次，可得

$$G^{(m)} = \left\{G_1^{(m)}, G_2^{(m)}, \cdots G_k^{(m)}\right\} \tag{5.16}$$

其中，$x_i^{(m)}$是类$G_i^{(m-1)}$的重心，$x_i^{(m)}$不一定是原始样本，当m逐渐增加时，分类也趋于稳定。即若对应某个m, 有$G^{(m+1)} = \left\{G_1^{(m+1)}, G_2^{(m+1)} \cdots G_k^{(m+1)}\right\}$与$G^{(m)} = \left\{G_1^{(m)}, G_2^{(m)}, \cdots G_k^{(m)}\right\}$相同，则计算结束。

5.2.3 基于聚类分析的相似日选择

1. 特征因素选取

利用聚类分析挑选相似日时，需要先确定样本的属性，即确定能够明显引起负荷变化的影响因子，通过对影响因素进行相关性分析，我们确定的主要影响因素有：年份、月份、星期类型（周一至周日）、日最高温度、日最低温度、白天天气状况、夜晚天气状况及相应风力风向，是否是节假日等。

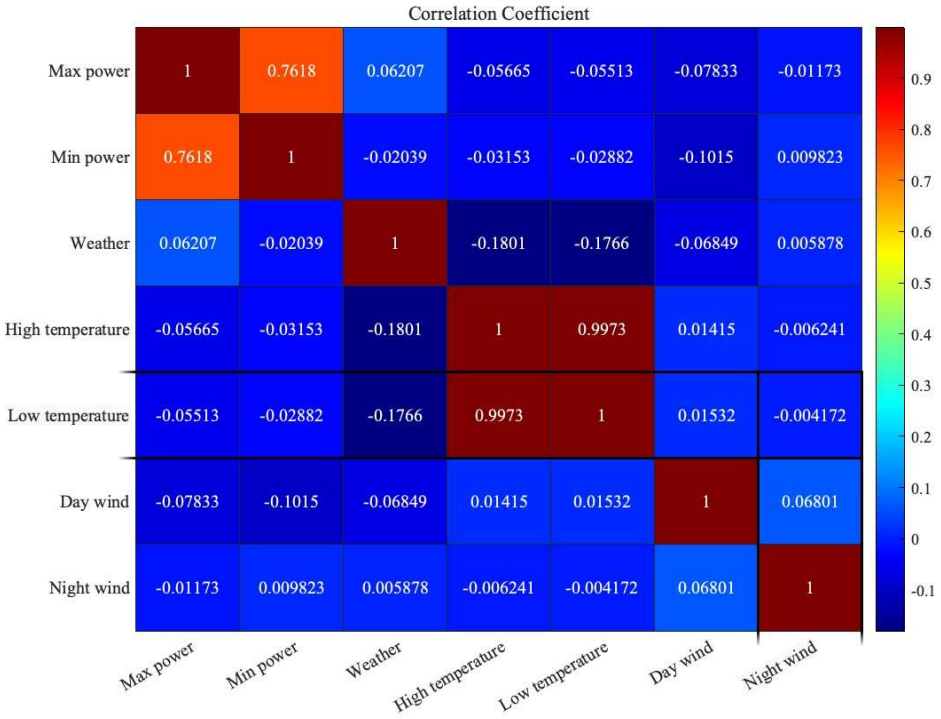


图 4 电力负荷、天气、温度和风向的关系热度图

2. 映射数据库设计

需要把各个物理量做适当映射，把不同量纲的值映射到数据库中，使各个量之间具有数值上的可比较性，进而对相似度和差异度经行定量的计算。

对题中数据建立映射数据库如下：

表 1 对各类气象状况的赋值

天气	赋值
晴	1
晴间多云	1.5
多云、局部多云	2
阴	3
雾	4
雷阵雨、阵雨	4.5
小雨	5
小到中雨、小雨-中雨	5.5
中雨	6
中到大雨、中雨-大雨	6.5
大雨	7
大到暴雨	7.5
暴雨	8

表 2 对各级风力的赋值

风力	赋值
<3 级	1
1～2 级、2 级	2
3 级、3～4 级	3
4 级、4～5 级	4
8～9 级	8

5.2.4 模型求解

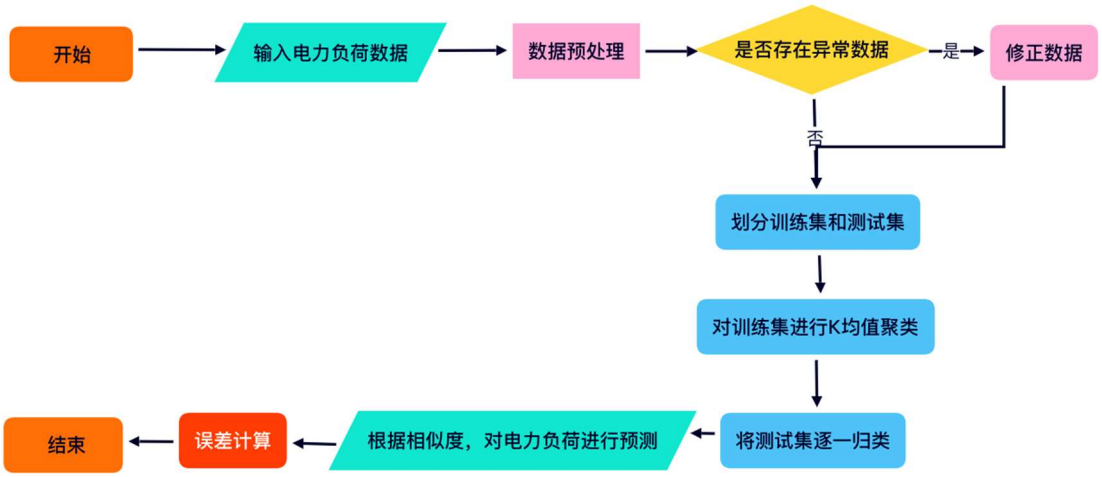


图 5 聚类模型的流程图

具体步骤如下：

（1）将数据集（附件一）中的电力负荷量记为y_i，对应特征记为x_i,对电力负荷异常值进行修复处理。

（2）将数据集划分为训练集和测试集（选取 90%作为训练集，10%作为预测集），设训练集数据个数为l_1，测试集数据个数l_2。

（3）利用欧氏距离对训练集数据进行聚类，将其分为 365 类，得到 365 个聚类中心，记为

$$C = \{\widehat{x_1}, \widehat{x_2}, \dots, \widehat{x_{365}}\} \tag{5.17}$$

（4）对测试集中每一个数据$x_i^*(i = 1,2, \cdots l_2)$，将其归类，标准如下：

$$d(x_i^*, \widehat{x_k}) = \min\{d(x_i^*, \widehat{x_j}), j = 1,2, \dots,365\} \tag{5.18}$$

则将x_i^*归类于以$\widehat{x_k}$为聚类中心的簇，记为G_k，设

$$G_k = \{\widetilde{x_1}, \widetilde{x_2}, \dots, \widetilde{x_n}\} \tag{5.19}$$

（5）计算x_i^*的预测值y_i^*，遍历G_k中数据$\widetilde{x_j}$，计算欧式距离$d(x_i^*, \widetilde{x_j}), (j = 1,2, \dots, n)$，则相似度

$$S(x_i^*, \widetilde{x_j}) = \frac{1}{d(x_i^*, \widetilde{x_j})} \tag{5.20}$$

$$y_i^* = \frac{\sum_{j=1}^{m} S(x_i^*, \widetilde{x_j})\widetilde{y_j}}{\sum_{j=1}^{m} S(x_i^*, \widetilde{x_j})} \tag{5.21}$$

（6）利用训练集的真实值y_i与预测值$y_i^*(i = 1,2, \dots, l_2)$，计算$RMSE, MAPE, MAE$对模型进行精度分析。

5.2.5 预测结果

根据 K 均值聚类预测模型，利用处理过后的附件 1 所给区域 15 分钟负荷数据（详见附件一），取 90%的数据为训练集，10%为测试集，利用 Python 编程（详见附录三），得出测试集的预测结果如下：

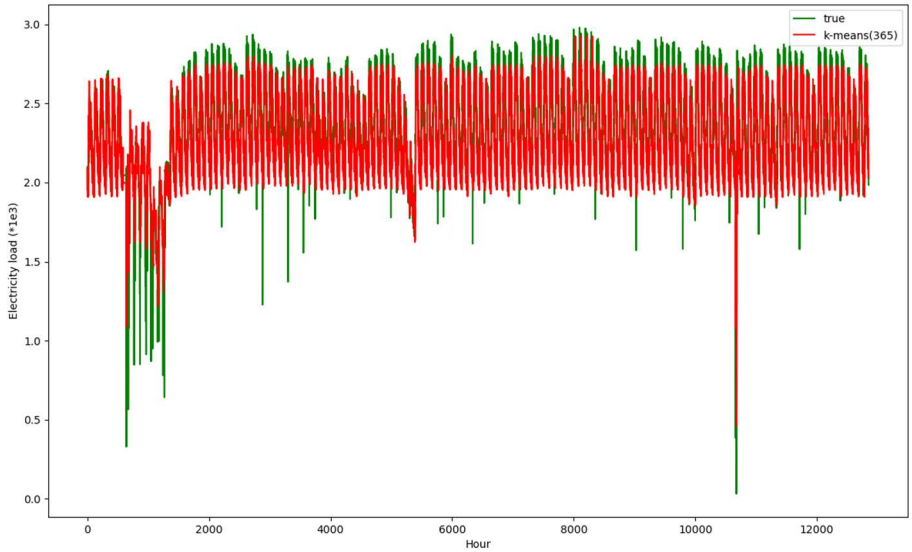


图 6 聚类模型的预测结果

输入该区域未来 10 天的天气状况、日类型特征（采用邻近年份同一天的天气状况），用同样的方法预测出该区域未来 10 天间隔 15 分钟负荷量，部分结果如下（详见附件三）：

表 3 该区域未来 10 天间隔 15 分钟电力负荷部分预测结果

时间 / 日期	0:00	0:15	…	23:30	23:45
2021/9/1	222208.2	198648.5	…	248048.4	211316.9
2021/9/2	203375.7	198116.9	…	229598.6	211906.2
2021/9/3	202923.0	204411.8	…	229699.1	227043.3
2021/9/4	203512.3	185579.3	…	230040.4	208593.5
2021/9/5	220319.0	185126.6	…	239674.2	208694.0
2021/9/6	201869.2	185715.9	…	221501.7	209035.3
2021/9/7	201969.7	212868.5	…	221534.1	226378.2
2021/9/8	202310.9	194418.7	…	221002.5	208205.7
2021/9/9	216782.1	194519.2	…	230606.2	208238.0
2021/9/10	198609.6	194860.5	…	211773.7	207706.4

对于未来三个月负荷预测，仍采用上述方法，输入未来三个月样本特征，得到每天最大负荷和最小负荷的预测值以及对应时间，部分结果如下（详见附件三）：

表 4 该区域未来三个月每天的最大功率、最小功率及对应的时间

日期	最小功率	时间	最大功率	时间
2021/9/1	201554.3735	5:30	259474.2666	11:00
2021/9/2	201729.1901	5:15	259496.9097	11:00
2021/9/3	201770.7209	5:30	259384.0896	11:00
2021/9/4	201534.6457	5:30	259494.6932	11:00
2021/9/5	201705.6719	6:15	259404.4343	11:00
2021/9/6	201792.4343	5:30	259472.4707	11:00
…	…	…	…	…
2021/11/29	201769.9140	4:45	259382.3470	17:00
2021/11/30	201533.8686	5:30	259495.4555	11:00

5.2.6 预测精度分析

聚类预测模型通过对训练集数据样本进行聚类，实现对测试集样本数据的预测，为检验预测的准确性，我们通过预测值、真实值及前文提到的误差指标，得到误差分析结果如下表所示，由此可见该模型是较为简单、实用、泛化能力好的模型。

表 5 聚类模型的各评价指标值

MAPE	*MAE*	*RMSE*	*NRMSE*
0.0362	63.9931	99.5940	0.0338

5.3 LSTM 模型

5.3.1 LSTM 基本原理[3]

长短期记忆网络（LSTM，Long Short-Term Memory）是由 Sepp Hochreiter

和 Jurgen Schmidhuber 在 20 世纪末提出的一种时间循环神经网络，是为了解决一般的 RNN（循环神经网络）存在的长期依赖问题而专门设计出来的，使其在含有时序相关问题的解决上，“记忆”能力优于 RNN 神经网络。LSTM 的网络结构如下图：

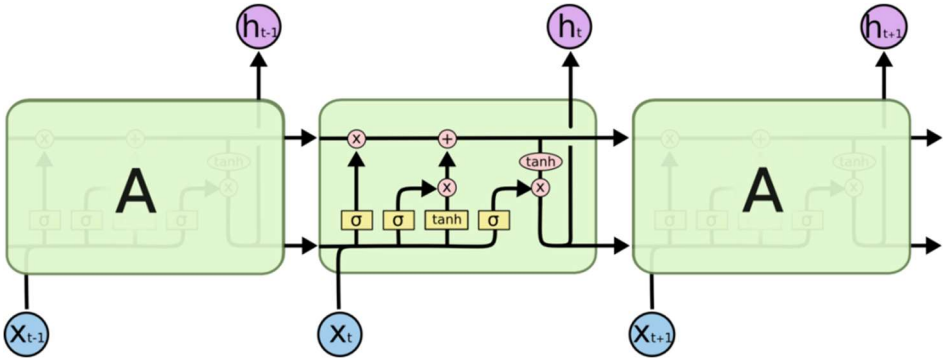


图 7 LSTM 网络结构图

相较于 RNN 神经网络，LSTM 神经网络的核心之处为：在细胞状态中增加遗忘门、记忆门、输出门，使其能够对流入细胞状态的信息进行选择“记忆”，从而避免当输入信息较多时，出现梯度消失致神经网络学习停止的现象。下以t时刻细胞工作流程为例进行分析：

（1）遗忘门决定需要舍弃的信息

遗忘门能够决定细胞状态并舍弃部分信息。这个阶段主要是对上一个节点传进来的输入进行选择性忘记，即为“忘记不重要的，记住重要的”。

遗忘门的工作原理可以用下面的公式表示:

$$f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \tag{5.22}$$

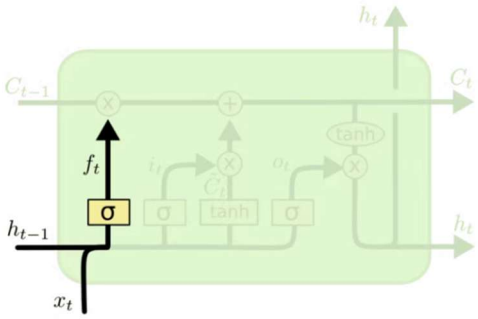


图 8 LSTM 网络的遗忘门结构

（2）记忆门决定更新信息和细胞状态

记忆门能够决定细胞状态并增加部分信息。这个阶段将输入进行有选择性地“记忆”。将重要的部分着重记录下来，不重要的部分，则少记一些。记忆门的工作原理可以用下面的公式表示：

$$\tilde{C_t} = \tanh(W_t[h_{t-1}, x_t] + b_c) \tag{5.23}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5.24}$$

同时，上一时刻细胞状态C_{t-1}，通过遗忘门丢弃需要舍弃的信息，通过记忆门添加需要更新的信息，最终得到现在时刻的细胞状态C_t。细胞状态更新的工作原理可以用下面的公式表示：

$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t} \tag{5.25}$$

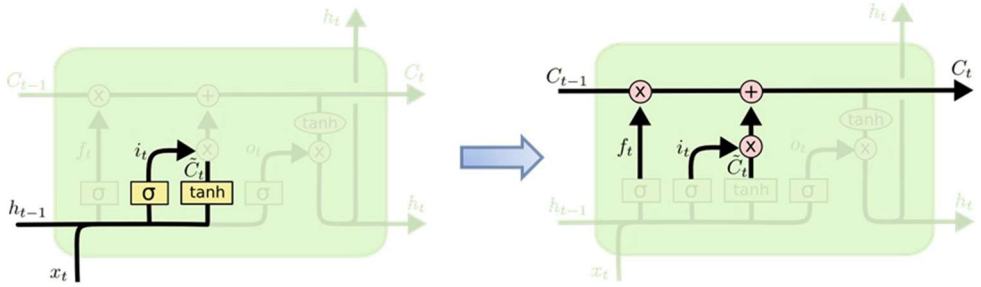


图 9 图左为 LSTM 网络的记忆门，图右为 LSTM 网络的状态更新门

（3）输出门决定输出信息

输出门能够决定细胞状态并输出部分信息。这个阶段将决定哪些内容将会被当成当前状态的输出，还对上一阶段得到的结果进行了放缩（通过一个激活函数$tanh$进行调节）。

输出门的工作原理可以用下面的公式表示：

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5.26}$$

$$h_t = o_t * \tanh(C_t) \tag{5.27}$$

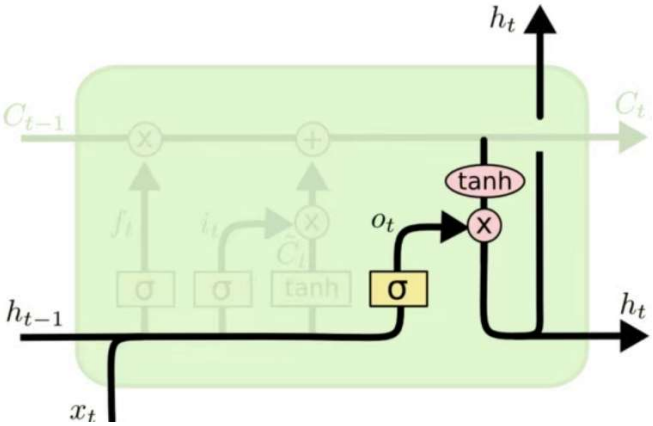


图 10 LSTM 网络的输出门

5.3.2 模型建立

LSTM 神经网络模型包括：输入层、隐含层（又称做 LSTM 层）和输出层。

（1）输入层和输出层维数确定

由于需确定模型的输入维度，因此我们对电力负荷的特性进行了分析，为预测输入候选特征的选择提供依据。通过对影响电力负荷的因素进行相关性分析，筛选出与该地电力负荷变化较强的影响因素，分别为白天及夜晚的风力风向和天气状况，每天的最高温度与最低温度，日类型特征。与上述 K 均值聚类模型类似，为了将不易处理的文本数据转化成易处理的数值数据，对各类天气和风力进行赋值（详见表 1 和表 2），将处理后的上述影响因素的数据作为预测的输入特征组，共计 35 项（详见附件三）。因此，我们确定模型的输入维度为 35，输入层的神经元个数为 28，输出维度为 1，模型输出层的神经元个数为 1。

（2）优化算法的确定

信息在 LSTM 神经网络模型中进行反向传播时，沿着需要优化参数的负梯度方向持续寻找更优的点，直至函数收敛。因此，可以将信息反向传播过程视

为目标函数的优化问题，通过计算得到目标函数的全局最优解。通过自适应参数优化算法，可以自动学习调整全局学习率的大小。我们选择使用 Adam 算法，因为 Adam 算法凭借计算时所需的内存较小，计算效率高的特点，在输入数据数量较多以及维数较大的情况下，仍能取得较好的优化效果，Adam 算法应用于 LSTM 神经网络中，避免了学习率选择不当的问题，提高模型的收敛速度和精度。

（3）激活函数的确定

设定隐含层中的激活函数为 sigmoid 函数：

$$S(x) = \frac{1}{1 + e^{-x}} \tag{5.28}$$

输出的激活函数为$tanh$函数

$$tanhx = \frac{sinhx}{coshx} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{5.29}$$

5.3.3 模型求解

本模型所用的四个行业的数据分别约 34000 项，按照 9：1 的比例，将数据划分为训练集和测试集。其中，训练集中数据用于电力负荷预测模型的学习与训练，测试集中数据用于电力负荷预测模型的测试与验证。经过多次实验，确定模型结构的参数设置如下表所示：

表 6 LSTM 模型各参数设定值

参数项	参数值
Hidden_Unit	100
Batch_Size	100
Epoch	200
Learning_Rate	0.01
Dropout	0.2

四个行业的测试集与训练集拟合图像如下：

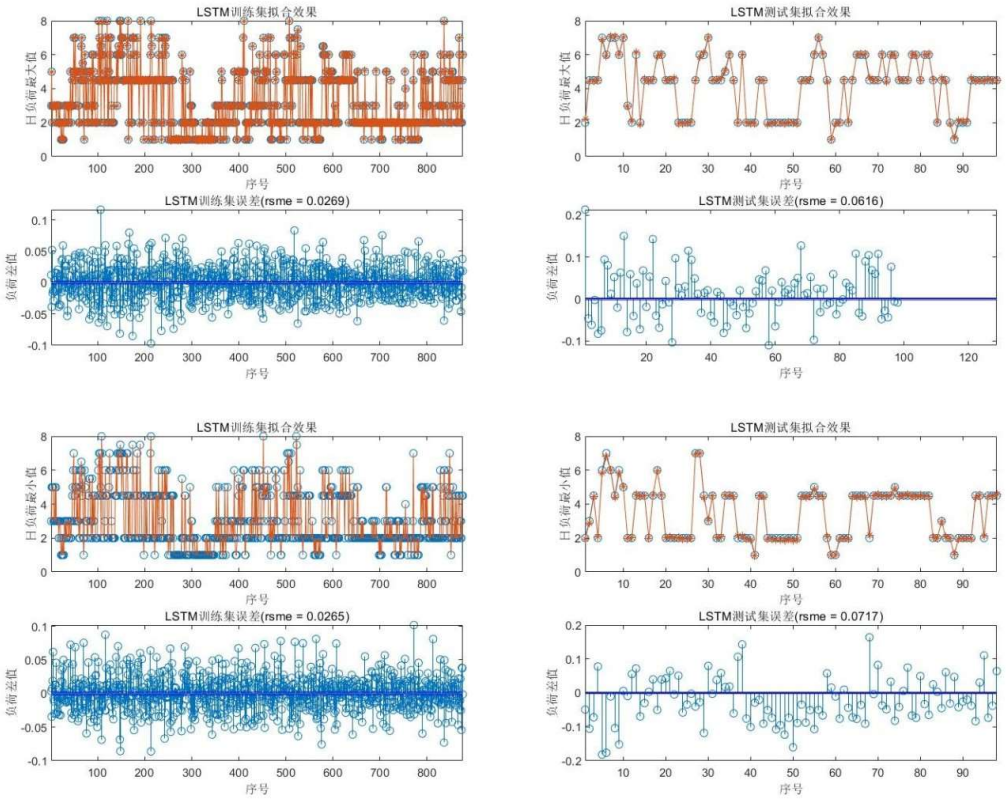


图 11 大工业训练结果

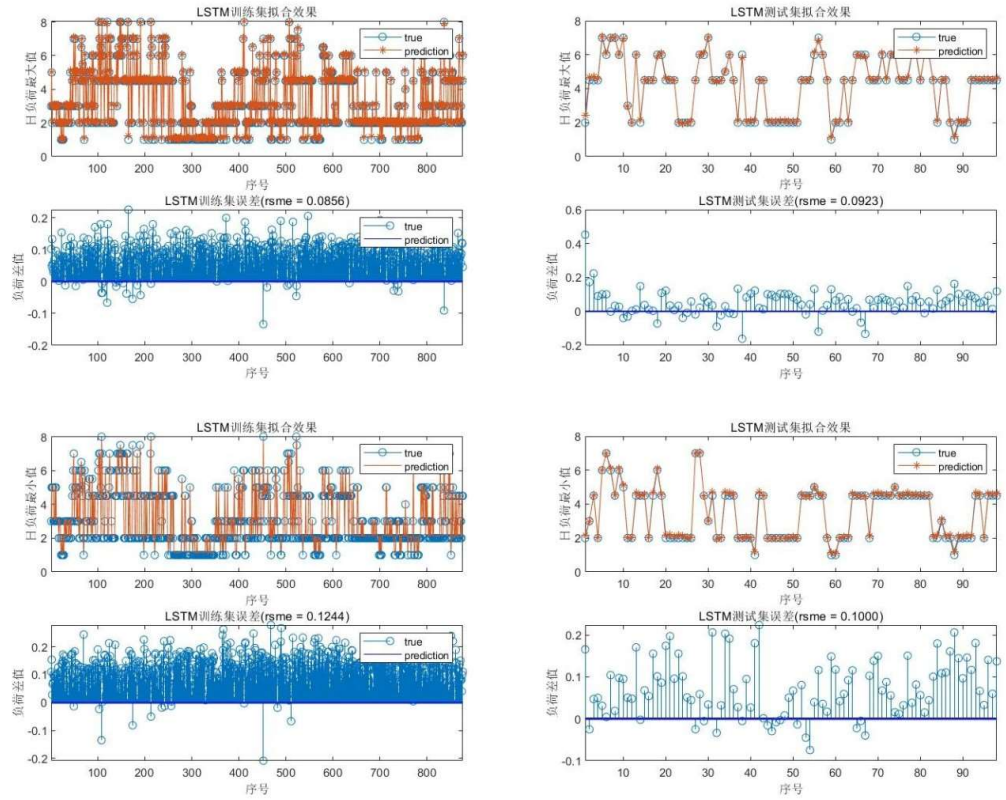


图 12 非普工业训练结果

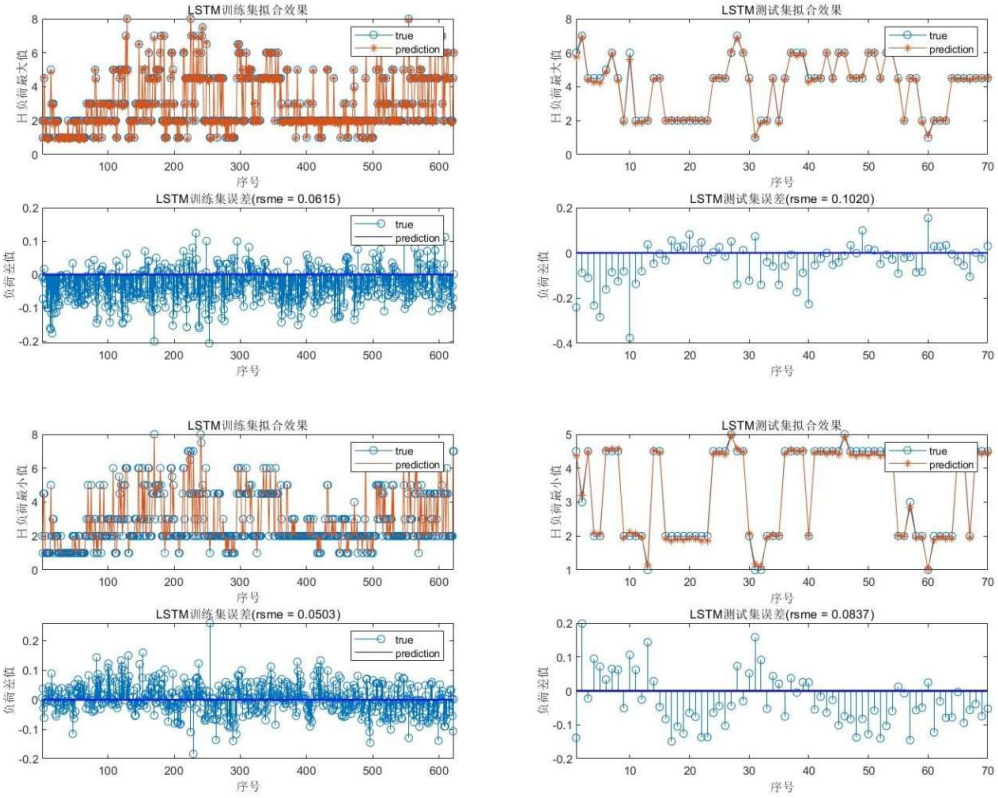


图 13 普通工业训练结果

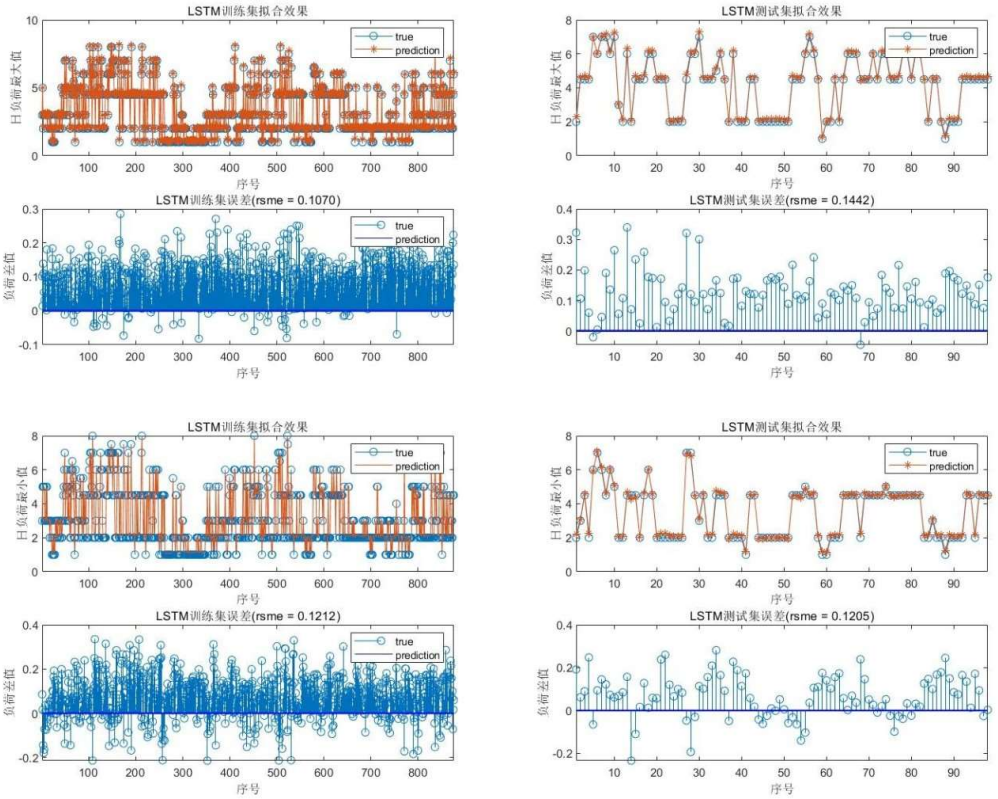


图 14 商业训练结果

5.3.4 预测结果

根据 LSTM 模型，利用 Matlab 编程训练出相应的神经网络结构，可得到各行业未来三个月日负荷最大值和最小值的预测结果。部分结果如下（详见附件四）：

表 7 大工业及非普工业未来三个月日负荷的最大值与最小值部分预测结果

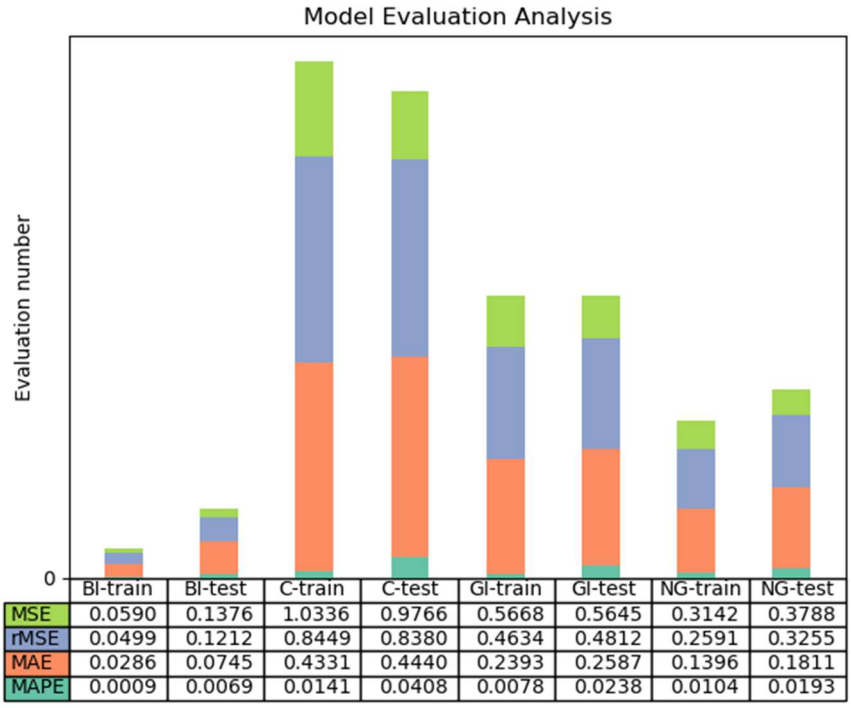
行业 / 时间	大工业		非普工业	
	最大功率	最小功率	最大功率	最小功率
2021/9/1	102365.2	91897.9	3254.379	1397.422
2021/9/2	104182.5	90663.9	3341.821	1394.524
2021/9/3	102649.4	87543.8	3172.482	1394.077
2021/9/4	100676.3	84022.5	2459.723	1393.726
2021/9/5	95612.0	81936.0	2535.478	1393.837
…	…	…	…	…
2021/11/29	114458.9	86836.9	3221.645	1395.069
2021/11/30	113141.1	90992.9	3186.740	1395.069

表 8 普通工业及商业未来三个月日负荷的最大值与最小值部分预测结果

行业 / 时间	普通工业		商业	
	最大功率	最小功率	最大功率	最小功率
2021/9/1	10057.42	3801.124	100909.6	20959.72
2021/9/2	9859.496	3741.026	98084.65	20931.97
2021/9/3	9919.303	3593.144	96793.17	20460.16
2021/9/4	6699.687	3336.392	90203.67	19470.4
2021/9/5	6551.074	3375.708	87430.23	19407.3
…	…	…	…	…
2021/11/29	7959.409	3500.322	80798.04	16713.49
2021/11/30	7787.703	3742.172	79496.69	16441.62

5.3.5 预测精度分析

LSTM 模型分别对训练集和测试集进行了预测，为检验模型预测效果，我们需要通过预测值和真实值计算模型的训练误差和测试误差，防止模型出现过拟合和欠拟合现象。利用前文提到的误差评价指标，得到误差结果如下图所示（其中 BI,C,GI,NG 分别表示大工业、商业、普通工业、非普工业，train, test 分别表示训练集和测试集），由此可见 LSTM 模型是预测精度高、泛化能力好的模型。



	BI-train	BI-test	C-train	C-test	GI-train	GI-test	NG-train	NG-test
MSE	0.0590	0.1376	1.0336	0.9766	0.5668	0.5645	0.3142	0.3788
rMSE	0.0499	0.1212	0.8449	0.8380	0.4634	0.4812	0.2591	0.3255
MAE	0.0286	0.0745	0.4331	0.4440	0.2393	0.2587	0.1396	0.1811
MAPE	0.0009	0.0069	0.0141	0.0408	0.0078	0.0238	0.0104	0.0193

图 15 各行业训练集和测试集预测误差评价指标

6. 数据挖掘与分析

6.1 突变时间、量级及原因分析

对各行业用电负荷数据进行一阶差分，即

$$df_k(d) = y_k(d) - y_k(d-1), d = 2, \cdots n-1 \tag{6.1}$$

其中，n为所给数据的总天数，df即为该行业的每日用电负荷变化值，对该行业用电负荷的一阶差分求平均值df_k^*, 标准差σ_k, 若$|df_k(d) - df_k^*| > 3\sigma_k$则认为此日该行业用电量发生突变，根据$df_k(d)$确定突变的量级。

通过挖掘数据，分析出各行业突变时间存在及量级存在以下规律：

对于大型工业、普通工业与非普工业，在元旦、春节、清明节、中秋节、国庆节等大型节假日用电负荷会大幅度减少，其中大型工业的突变量级为 20000kw 左右，普通工业的突变量级为 5000kw 左右，非普工业的突变量级为 1000kw 左右。同时，在周六日此三种行业用电负荷也存在较小范围的减少，相应的，周一以及假期结束的第一天用电负荷会大幅度的增加。

对于商业用电负荷，突变时间恰好与上述三种相反，在周末即大型节假日用电负荷会大幅度增加，而在工作日用电负荷相对较低，突变量级在 50000kw 左右。

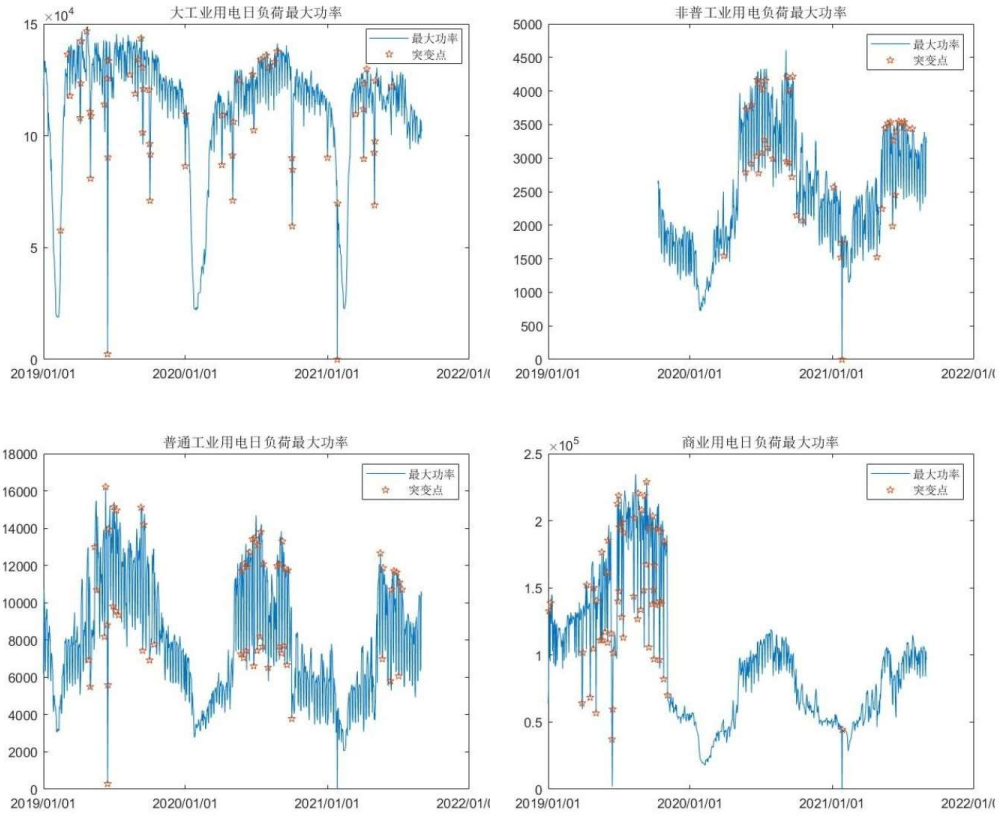


图 16 各行业电力负荷突变的时间点

下选取部分数据展现各行业用电负荷突变的时间、量级及原因（详见附件五）。

表 9 各行业用电负荷突变的部分情况说明

行业	突变时间	突变量级	原因
大工业	2019/4/5	-24020	清明节放假
	2019/4/8	18770	复工
	2019/6/17	43150	周一复工
	2020/1/1	-17340	元旦放假
非普工业	2020/5/25	1080.6264	周一复工
	2020/10/1	-1376.0184	国庆放假
	2021/6/5	-1068.8196	周六放假
	2021/6/15	1131.7836	端午复工
普通工业	2019/5/13	5049.036	周一复工
	2019/9/13	-7189.47	中秋节放假
	2020/6/28	5638.59	端午复工
	2021/5/22	-4955.99	周六放假
商业	2019/3/31	42156.74	周日消费高峰
	2019/6/3	-51864	周一消费低谷
	2019/10/1	69741.19	国庆节消费增长
	2019/10/5	-63916.6	国庆消费高峰结束

整体来看，自 2019 年 12 月末疫情爆发以来，各行业用电量都有所下降。疫情防控力度加大，对零售、餐饮、旅游运输、文娱等行业造成一定程度的冲击，因此，商业用电量自 2019 年末以来下降幅度最为明显。随着疫情形势好转，各行业逐渐复工复产，用电负荷又有所上升。但因疫情的反复，各行业用电量仍较 2019 年初有所降低。

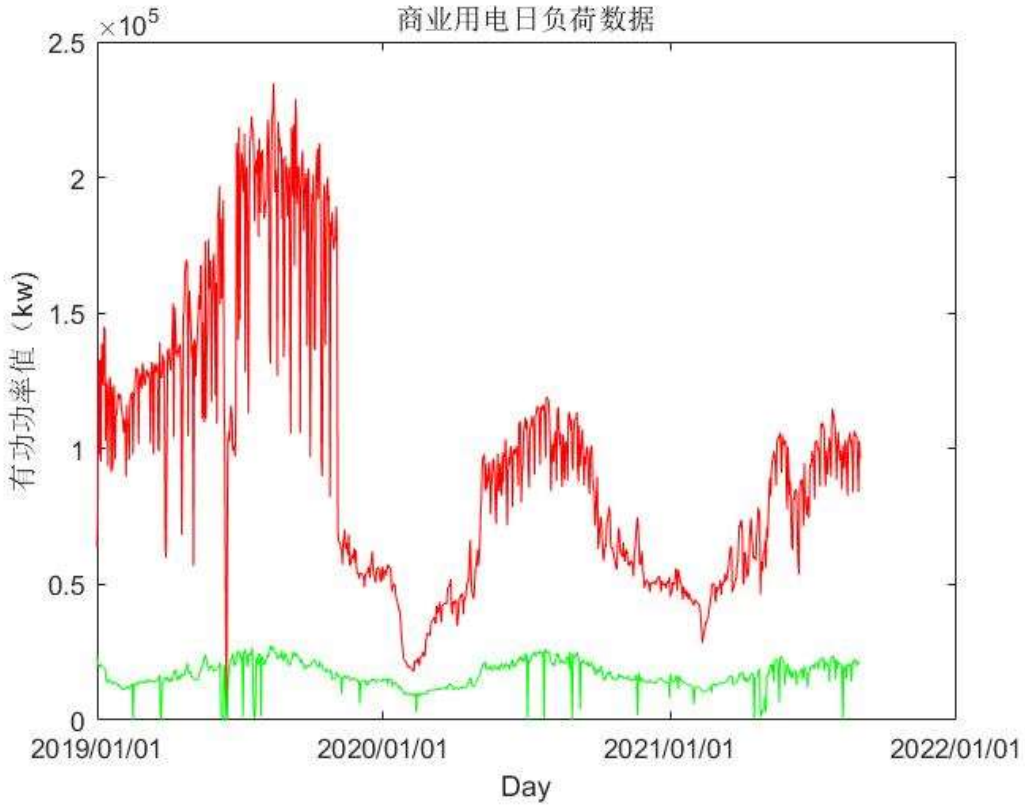


图 17　2019-1-1 至 2021-8-31 商业用电最大值和最小值曲线

6.2“双碳”政策下对各行业未来发展的建议

以二氧化碳为代表的温室气体大量排放对全球生态系统和人类社会的可持续发展均产生了严重威胁。作为目前世界上最大的二氧化碳排放经济体，中国展现出发展中大国的责任担当[4]，于 2020 年 9 月提出“二氧化碳排放力争于 2030 年前达到峰值，努力争取 2060 年前实现碳中和”的目标(简称“双碳”目标）。

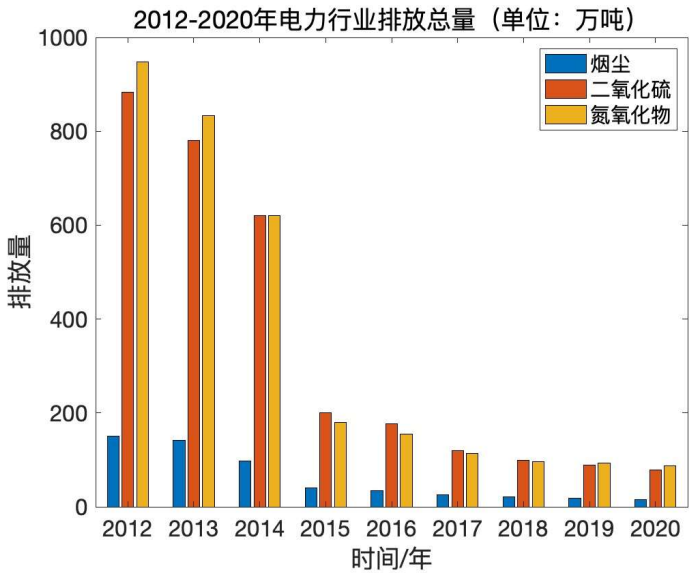


图 18　2012-2020 年电力行业排放总量条形图

降耗减碳不是我国发展历程的新课题，我国过去在发展规划、节能减排、环境治理等领域采取了一系列重大举措并取得了丰硕成效。2060 年碳中和愿景无疑将该课题推到了前所未有的高度，新时代减碳工作对我国相关规划、产业、行业都提出了新要求。

因此，我们根据各行业现状，对各行业未来发展提出如下建议。

6.2.1 对工业的建议

第一，规划绿色布局和绿色体系，构建合理的工业生产布局体系，减少无效的碳排放。

第二，继续提升能源利用效率，目前钢铁等行业通过高效节能，如数字化、智能化节能新手段，把能效利用率提升到了 60%以上。

第三，优化能源结构，融入相关产业和城市的循环经济发展，比如大工业中的钢铁和建材、钢铁和化工都可以做到有机结合。

第四，突破核心技术，如氢能源的利用，包括氢能源的运输和存储在内。此外，双碳目标下各工业都需考虑脱碳转电，因此，电力的供应尤为重要。发展负碳吸收技术，主要是 CCUS 技术[5]（Carbon Capture&Storage），即二氧化碳的捕集、储存和利用技术。

第五，做好制度建设，各个公司、企业应将减排目标和路线图化为各部门的减排目标和路线图，并将减排目标纳入部门负责人考核体系。设立公司“碳税”，在公司内部交易中，通过建立模拟市场的方式将碳税成本计入模拟利润计算，让各部门主动承担起减少碳排放的责任。

6.2.2 对商业的建议

构建基于能源互联网的新型商业模式，可以从实际出发构建“物联网+”“互联网+”“能源+”三类商业模式，打造大数据体系、互联网体系、物联网体系三大模块，推进基于能源互联网平台商业系统的构建实施。

同时，各大商业银行应在帮助企业进行节能减排的产线改造、支持绿色低碳项目、参与碳排放交易等方面发挥重要作用，做好气候环境风险管理，提升绿色金融业务[6]占比，树立负责商业银行的形象。

6.2.3 对电力行业的建议

交通、建筑、工业等行业纷纷将电气化作为实现双碳目标的重要措施，对电力行业的发展提出新的挑战。电力将从过去的二次能源转变为其他行业事实上的基础能源，“双碳”目标下电力行业急需转型发展。

一方面，电力行业要保障电力安全可靠供应，以满足国民经济发展目标以及人民生活用电需求,在保障电力燃料供应的同时密切跟踪经济走势、电力需求、天气变化合理安排电网运行方式，加强电网运行方式和电力电量平衡协调。推动建立跨省跨区备用辅助服务市场，强化跨省跨区交易组织保障，充分应用跨区跨省输电通道能力。对各种情形下电网供电能力进行风险评估，并根据结果制 定合理的解决措施。

另一方面，电力行业需加快清洁低碳供应结构转型进程，实现碳减排目标。目前，煤电仍是国家能源发电的主力军，也是二氧化碳污染的主要来源，面对能源变革，煤电面临淘汰风险。因此，新能源的开发更为重要，电力行业应大力发展风电、水电、太阳能发电、光电、储能技术等，促进传统发电向新能源发电转型，最终实现对化石能源的彻底取代。

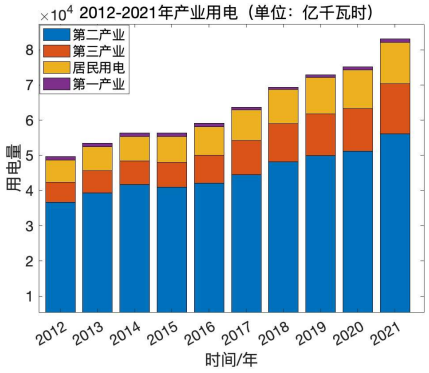


图 19　2012-2021 年不同产业用电量

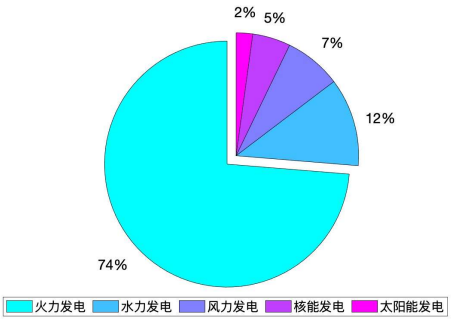


图 20　2020 年电力来源占比

总之，“双碳”目标重在落实，必须以习近平生态文明思想为指引，保持生态文明建设的战略定力，统筹考虑能源安全、经济增长、社会民生、成本投入等诸多因素，实现碳达峰、碳中和的目标愿景。

7. 模型的评价与推广

7.1 模型评价

7.1.1 优点

（1）充分考虑外部因素的影响有助于提高负荷预测精度。

（2）K 均值聚类算法原理简单，易于操作，且执行效率高。

（3）LSTM 神经网络模型在序列建模问题上有一定优势，具有长时记忆功能，解决了长序列训练过程中存在的梯度消失和梯度爆炸的问题。

7.1.2 缺点

（1）K 均值聚类算法受初值及离群点的影响较大，易造成结果不稳定。

（2）LSTM 本身的模型结构相对复杂，训练比较耗时。

（3）设计外部因素影响提高了输入信号的维数，从而加重了预测模型的学习负担，影响学习效率。

7.2 模型改进

模型中利用了临近日的天气情况来代替所需预测日期的天气情况，不够准

确，可通过登陆气象网站收集更为准确合理的气象数据，提高预测精度。

由于 LSTM 训练比较耗时，因此可以利用遗传算法或其他优化算法对 LSTM 神经网络进行优化，优化的参数主要有：LSTM 的层数、隐藏层神经元的个数、全连接层的层数及神经元个数，以此提高 LSTM 预测模型的运行效率。

电力系统负荷预测是在一定假设条件下进行的，其中包含了许多不确定性因素，采用单一的预测方法很难取得精确的结果，因此，后续可考虑将多种预测方法进行组合优化，得到更为精确的预测模型。

7.3 模型推广

本文中使用的模型对未来的电力负荷情况做了较为准确的预测，综合考虑了各项气象指标、日类型等因素,因而对于实际中的电力负荷预测有一定的借鉴意义。

文中的聚类预测模型有较好的负荷预测精度，且在处理周期性和随机非线性负荷预测问题时效果出色。因此可将其应用于其他有相似变化规律的预测问题中，如温度预测、用水量预测等。LSTM 具有长时记忆功能，可以用于解决股票价格、空气污染物含量预测等问题。

8. 参考文献

[1] 余阳. 基于深度学习的电力系统短期负荷预测研究[D]. 南昌大学, 2021.

[2] 廖宗明. 基于 K 均值聚类与支持向量机的电力系统短期负荷预测[D]. 华南理工大学, 2012.

[3] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8): 1735-1780, 1997.

[4] 虞卫东. “双碳”对企业和行业发展的影响与应对[J]. 现代企业, 2022(01):78-79.

[5] 胡其会, 李玉星, 张建, 俞欣然, 王辉, 王武昌, 殷布泽, 龚雾昱. “双碳”战略下中国 CCUS 技术现状及发展建议[J/OL]. 油气储运:1-14[2022-05-09].

[6] 陈毓佳. 绿色金融支持“双碳”目标实现的作用机制研究[J]. 老字号品牌营销, 2022(09):60-62.

9. 附录

下面附上本文涉及的部分核心代码，分别为缺失值查找及填补、异常值修复、K 均值聚类、LSTM 的程序代码。

1. 缺失值查找及填补 Python 代码

```
#encoding=utf-8
import openpyxl
import pandas as pd
import datetime
index = pd.date_range(start='1/1/2018',periods=128544,freq='15min')
def load_Data():
    df0 = pd.read_csv("D:\数学建模校赛\全部数据\附件 1-区域 15 分钟负荷数据.csv")
    df0['数据时间']=pd.to_datetime(df0['数据时间'])
    return df0
#缺失值处理，插值替换
df1 = load_Data()   #加载数据
df1_date = df1['数据时间'].tolist()  #数据日期转为列表
df1_power = df1[ '总有功功率（kw）'].tolist()   #数据值转为列表
index_date=index.tolist()
index_power=[0 for x in range(0,128544)]
flag=0
for j in range(0, len(df1_date)):
    date0 = index_date[j+flag]
    date_i = df1_date[j]    #顺序选取数据中日期列表里对应各日期
    while date_i != date0:#如数据中日期列表与期望日期序列不相等，即存在缺失值执行
while 程序
        flag=flag+1
        date0 += datetime.timedelta(minutes=15) #日期加一
            index_power[j+flag]=df1_power[j]

for i in range(0,len(index_power)):
    if index_power[i]==0:
        index_power[i]=(index_power[i-1]+index_power[i-2]+index_power[i-3])
wk=openpyxl.load_workbook("D:/数学建模校赛/附件 1 完整.xlsx")
sheet=wk["Sheet"]
col=['数据时间','总有功功率（kw）']
for i in range(0,2):
    sheet.cell(row=1,column=i+1).value=col[i]

for i in range(0,len(index_date)):
    sheet.cell(row=i+2,column=1).value=index_date[i]
    sheet.cell(row=i+2,column=2).value=index_power[i]
```

```
wk.save("D:/数学建模校赛/附件 1 完整.xlsx")
```

2. 异常值识别及处理 Matlab 代码

```
Y=xlsread('E:\区域 15 分钟负荷数据.xlsx',1,'B2:B128545');
%水平修正 相邻点的变化值不应过大
N1=Y(1:end-1);
N2=Y(2:end);
N=N2-N1;
%boxplot(N);
m1=mean(N);
s1=std(N,1);
D1=find(abs(N-m1)>3*s1);    %找出异常值的下标
d=D1+1;     %异常数据在原始数据中的下标
%对异常数据进行修正(三次修正 w1 w2 w3 自己设置)
w1=0.3;w2=0.5;w3=0.2;
for i=d
    Y1=(Y(i-1)+Y(i+1))/2;
    Y2=(Y(i-2)+Y(i-1)+Y1+Y(i+1)+Y(i+2))/5;
    Y3=(Y(i-1)+Y2+Y(i+1))/3;
    Y(i)=w1*Y1+w2*Y2+w3*Y3;
end
%共 1339 天，每天 96 个点    每列代表一天的数据
Data=reshape(Y,96,1339);
%垂直修正   每天同一时间的值相差不应过大
m2=zeros(96,1);
s2=zeros(96,1);
for i=1:96
    m2=mean(Data(i,:));
    s2=std(Data(i,:));
    D2=find(abs(Data(i,:)-m2)>3*s2);    %找出异常值下标
    for j=D2      %进行修正
        if Data(i,j)>m2
            Data(i,j)=m2+3*s2;
        else
            Data(i,j)=m1-3*s2;
        end
    end
end
%Data2=reshape(Data,1399*96,1);
Data=Data';     %1399*96 维矩阵
```

3. 聚类模型 Python 代码

```
import math
import matplotlib.pyplot as plt
```

```
import numpy as np
import pandas as pd
from scipy.cluster.vq import kmeans
from scipy.spatial import distance
from scipy.spatial.distance import euclidean
from tools import statistics

def kMeansClustering(x,k):
    conv = np.asarray(x)
    centroids = kmeans(conv, k, iter=10)[0]
    labels = []
    for y in range(len(x)):
        minDist = float('inf')
        minLabel = -1
        for z in range(len(centroids)):
            e = euclidean(conv[y], centroids[z]) # 欧式距离
            if (e < minDist):
                minDist = e
                minLabel = z
        labels.append(minLabel)
   return (centroids, labels)
# Performs a weighted clustering on the examples in xTest
# Returns a 1-d vector of predictions
def predictClustering(clusters,clusterSets,xTest,metric):
    clustLabels = []
    simFunction = getDistLambda(metric)
    for x in range(len(xTest)):
        clustDex = -1
        clustDist = float('inf')
        for y in range(len(clusters)):
            dist = simFunction(clusters[y],xTest[x])
            if (dist < clustDist):
                clustDist = dist
                clustDex = y
        clustLabels.append(clustDex)
    predict = np.zeros(len(xTest))
    for x in range(len(xTest)):
        predict[x] =
weightedClusterClass(xTest[x],clusterSets[clustLabels[x]],simFunction)
    return predict
# Performs a weighted cluster classification
def weightedClusterClass(xVector,examples,simFunction):
    pred = 0.0
    normalizer = 0.0
```

```
    ctr = 0
    for x in examples:
        similarity = 1.0/simFunction(xVector,x[0])
        pred += similarity*x[1]
        normalizer += similarity
        ctr += 1
    return (pred/normalizer)

def getDistLambda(metric):
    if (metric == "manhattan"):
        return lambda x,y : distance.cityblock(x,y)
    elif (metric == "cosine"):
        return lambda x,y : distance.cosine(x,y)
    else:
        return lambda x,y : distance.euclidean(x,y)

# define a function to convert a vector of time series into a 2D matrix 定义
将时间序列向量转换为二维矩阵的函数
def convertSeriesToMatrix(vectorSeries, sequence_length):
    matrix=[]
    for i in range(len(vectorSeries)-sequence_length+1):
        matrix.append(vectorSeries[i:i+sequence_length])
    return matrix

# load raw data 加载原始数据
df_raw = pd.read_csv('../data/ENTSO-E/our_load.csv', header=0,
usecols=[0,1])
df_raw_array = df_raw.values
list_load = [df_raw_array[i, 1] / 1000 for i in range(0, len(df_raw))]
print ("Data shape of list_load: ", np.shape(list_load))
# 异常值处理
k = 0
for j in range(0, len(list_load)):
    if(abs(list_load[j] - list_load[j - 1])>2 and abs(list_load[j] -
list_load[j + 1])>2):
        k = k + 1
        list_load[j] = (list_load[j - 1] + list_load[j + 1]) / 2 +
list_load[j - 24] - list_load[j - 24 - 1] / 2
    sum = 0
    num = 0
    for t in range(1,8):
        if(j - 96*t >= 0):
            num = num + 1
            sum = sum + list_load[j - 96 * t]
```

```
            if(j + 96*t < len(list_load)):
                num = num + 1
                sum = sum + list_load[j + 96 * t]
        sum = sum / num
        if(abs(list_load[j] - sum)>3):
            k = k + 1
            if(list_load[j] > sum): list_load[j] = sum + 3
            else: list_load[j] = sum - 3
# shift all data by mean 去均值
list_load = np.array(list_load)
shifted_value = list_load.mean()
list_load -= shifted_value
# the length of the sequnce for predicting the future value
sequence_length = 96
# convert the vector to a 2D matrix
matrix_load = convertSeriesToMatrix(list_load, sequence_length)
matrix_load = np.array(matrix_load)
print("Data shape: ", matrix_load.shape)
# split dataset: 90% for training and 10% for testing 切分数据集
train_row = int(round(0.9 * matrix_load.shape[0]))
print('train:', train_row, 'test:', int(round(0.1 * matrix_load.shape[0])))
train_set = matrix_load[:train_row, :]
np.random.seed(1234)
np.random.shuffle(train_set)
X_train = train_set[:, :-1]
y_train = train_set[:, -1]
print(X_train[0], y_train[0])
X_test = matrix_load[train_row:, :-1]
y_test = matrix_load[train_row:, -1]
time_test = [df_raw_array[i,0] for i in range(train_row+23, len(df_raw))]
ckmeans_365,lkmeans_365 = kMeansClustering(X_train, 365)
c = [ckmeans_365]
l = [lkmeans_365]
algNames = ["true", "k-means(365)"]
preds = []
preds.append(y_test)
for t in range(len(c)):
       centroids = c[t]
        labels = l[t]
        clusterSets = []
       timeLabels = []
    for x in range(len(centroids)):
        clusterSets.append([])
    for x in range(len(labels)):
```

```
                clusterSets[labels[x]].append((X_train[x], y_train[x]))
        n = 1000
    predicted_values = predictClustering(centroids, clusterSets, X_test,
"euclidean")
    mape = statistics.mape((y_test + shifted_value) * n, (predicted_values +
shifted_value) * n)
    print('MAPE is ', mape)
    mae = statistics.mae((y_test + shifted_value) * n, (predicted_values +
shifted_value) * n)
    print('MAE is ', mae)
    mse = statistics.meanSquareError((y_test + shifted_value) * n,
(predicted_values + shifted_value) *n)
    print('MSE is ', mse)
    rmse = math.sqrt(mse)
    print('RMSE is ', rmse)
    nrmse = statistics.normRmse((y_test + shifted_value) * n,
(predicted_values + shifted_value) * n)
    print('NRMSE is ', nrmse)
    preds.append(predicted_values)
# show
fig = plt.figure()
colors = ["g", "r", "b", "c", "m", "y", "k", "w"]
legendVars = []
for j in range(len(preds)):
    print(j)
    x, = plt.plot(preds[j]+shifted_value, color=colors[j])
    legendVars.append(x)
plt.xlabel('Time')
plt.ylabel('Electricity load (*1e3)')
plt.legend(legendVars, algNames)
plt.show()
fig.savefig('../result/clustering_result.jpg', bbox_inches='tight')
data1 = np.array(preds+shifted_value)
data1 = data1.T
data1 = np.reshape(data1, (int(round(0.1 * matrix_load.shape[0])), 2))
pred = pd.DataFrame(data1)
pred.to_csv('../data/prediction.csv')
```

4. LSTM 模型 Matlab 代码

```
filename = '/Commerce_load.csv';
file = readtable(filename, 'VariableNamingRule', 'preserve');
data = table2array(file(:, 3:end));
% 输入、输出数据提取
```

```
output = data(:, 3:4);
input = data(:, 3:end);
train_number = floor(0.9 * size(data, 1));
test_number = size(data, 1) - train_number;
xTrain = input(1:train_number, :);
yTrain = output(1:train_number, :);
xTest = input(train_number+1 : end, :);
yTest = output(train_number+1 : end, :);
% 归一化
[XTrain, PSx] = mapminmax(xTrain');
[YTrain, PSy] = mapminmax(yTrain');
XTest = mapminmax('apply', xTest', PSx);
%%%%%%%%%%%%%%%%%%%%%%%%% 构建 LSTM 网络
% 输入特征的维度
numFeatures = size(input, 2);
% LSTM 网路包含的隐藏单元数目
numHiddenUnits = 100;
% 输出响应维度
numResponses = 2;
% 重点注意 regressionLayer
layers = [sequenceInputLayer(numFeatures)
lstmLayer(numHiddenUnits)
fullyConnectedLayer(numResponses)
regressionLayer];
miniBatchSize = 100;
options = trainingOptions('adam', ...
'ExecutionEnvironment', 'cpu', ...
'MaxEpochs', 200, ...
'MiniBatchSize', miniBatchSize, ...
'GradientThreshold', 1, ...
'InitialLearnRate', 0.01, ...
'LearnRateSchedule', 'piecewise', ...
'LearnRateDropPeriod', 250, ...
'LearnRateDropFactor', 0.2, ...
'Verbose', false, ...
'Plots', 'training-progress');
% 训练
net = trainNetwork(XTrain, YTrain, layers, options);
```

```
yTrain_pre = predict(net, XTrain, 'MiniBatchSize', miniBatchSize,
'SequenceLength', 'longest');
% 反归一化
yTrain_pre = mapminmax('reverse', yTrain_pre, PSy);
yTrain_pre = yTrain_pre';
% 差值
err_train1 = yTrain_pre(:,1) - yTrain(:,1);
err_train2 = yTrain_pre(:,2) - yTrain(:,2);
% 均方误差
rmse_train1 = rms(err_train1);
rmse_trarin2 = rms(err_train2);
line_acc = 0.002 * ones(1, train_number);

mape = mean(abs((yTrain - yTrain_pre)./yTrain))
mae = mean(abs(yTrain - yTrain_pre))
mse = sqrt(sum((yTrain- yTrain_pre).^2)) ./ train_number
figure
yTest_pre = predict(net, XTest, 'MiniBatchSize', miniBatchSize,
'SequenceLength', 'longest');
% 反归一化
yTest_pre = mapminmax('reverse', yTest_pre, PSy);
yTest_pre = yTest_pre';
% 差值
err_test1 = yTest_pre(:,1) - yTest(:,1);
err_test2 = yTest_pre(:,2) - yTest(:,2);
% 均方误差
rmse_test1 = rms(err_test1);
rmse_test2 = rms(err_test2);
mape2 = mean(abs((yTest - yTest_pre)./yTest))
mae2 = mean(abs(yTest - yTest_pre))
mse2 = sqrt(sum((yTest- yTest_pre).^2)) ./ test_number
```