Rain Forecasting Using Machine Learning

Fabing Lin

April 2023

1 Abstract

Rainfall forecasting is important for many catchment management applications, in particular for flood warning systems. The variability of rainfall in space and time, however, renders quantitative forecasting of rainfall extremely difficult. The depth of rainfall and its distribution in the temporal and spatial dimensions depends on many variables, such as pressure, temperature, and wind speed and direction. Due to the complexity of the atmospheric processes by which rainfall is generated and the lack of available data on the necessary temporal and spatial scales, it is not feasible generally to forecast rainfall using a physically based process model. Recent developments in artificial intelligence and, in particular, those techniques aimed at pattern recognition, however, provide an alternative approach for developing of a rainfall forecasting model. Artificial neural networks (ANNs), which perform a nonlinear mapping between inputs and outputs, are one such technique. Presented in this paper are the results of a study investigating the application of ANNs to forecast the spatial distribution of rainfall for an urban catchment. Three alternative types of ANNs, namely multilayer feedforward neural networks, partial recurrent neural networks, and time delay neural networks, were identified, developed and, as presented in this paper, found to provide reasonable predictions of the rainfall depth one time-step in advance. The data requirements for and the accuracy obtainable from these three alternative types of ANNs are discussed.(待修改)

目录

2 Introduction

气象预测直接关乎人类社会的生产生活，一直以来是人类重点研究的科学领域之一。传统的天气预报采用非常复杂的数值气象模型，近年来越来越多的机器学习，深度学习方法被应用到这一领域，他们将气象预测抽象成一个时空预测问题，尝试通过多维时间序列、图神经网络（GNN）等方法来解决。本研究针对 ERA5 公用数据集进行探究。收集整理温度、风的成分、涡量特征使用机器学习模型对数据进行建模，进行未来的短时或长期分类预测，以这个结果对数值气象模型进行修正。在本研究中我们使用了机器基准模型进行基本的分类预测效果的记录。分类是一个有监督的学习过程，目标数据库中有哪些类别是已知的，分类过程需要做的就是把每一条记录归到对应的类别之中。由于必须事先知道各个类别的信息，并且所有待分类的数据条目都默认有对应的类别。在本研究之中，我们旨在对序列站点进行建模，从而达到更高的模型精度来预知未来的降雨情况。

2.1 晴雨预报

Predicting the environment like rain and temperature gives us the advantage of understanding the climate and decision-making based on it. Environmental conditions like temperature, wind, and humidity affect human livelihood. In recent times, forecasting rainfall is the most wanted research area because of the complexity of data processing and application in monitoring the weather. Traffic and road accidents are often reported during heavy rainfall and storm. Autonomous vehicles, like driving cars, highly depend on environmental conditions. Heavy rain, wind, traffic, and congestion occur in severe climate conditions. The early prediction of the environment is essential to predict and warn of climate disasters.

Preserving life and property, promoting public health and safety, and sustaining economic growth and quality of life are only some of the societal benefits that can be supported by accurate weather forecasts, which is why this field of study is so important. To maximize productivity, farmers might benefit from having access to up-to-date weather reports. Predicting the weather's course is essential for preventing the spread of pests and illnesses among crops. Pests that consume crops might be affected by the weather. The previous studies are based on the Numerical Weather Prediction (NWP) methods (Schultz and Martin, 2021, Markovics et al., 2022). It is associated

with many challenges, such as a lack of knowledge and understanding of physical mechanisms. Collecting necessary information from a huge amount of data requires a lot of power consumption and superpower computers for processing. The accuracy of the numerical weather prediction methods is based on the usage of the physical model. It is mostly based on nonlinear equations called primitive equations (Markovics et al., 2022). The minor changes in the atmosphere will have more impact on the model result when the model receives input with incomplete information about the atmosphere. The error, which is the difference between the current time and the predicted time, increases. This error automatically reduces the model' s accuracy. Another enormous problem with the existing model is the deluge of information (Zheng et al., 2014). The data are collected from the census from the autonomous observing platform, and the model generates a huge amount of results day by day. The datasets used in the numerical weather prediction also comprise much uncertainty. This nonlinear equation requires supercomputers for computations, which are more expensive.

In recent decades, weather forecasting techniques based on deep learning and neural networks have developed significantly. Deep learning technology has many applications in computer vision; time series problems can effectively predict spatial and temporal information (Utku and Kaya, 2022, Mladenovic et al., 2022). The sample and typical geospatial data are meteorological data. The deep learning-based weather prediction method is a solid alternative to the conventional weather prediction method. Deep learning model such as Artificial Neural Networks (ANN), which is the same as the human brain and relates the complex relationship of the input and output of the task, is a more powerful technology and data modeling by implementing the artificial system which carried out the task (Weyn et al., 2021). Nonlinear techniques like weather forecasting can be easily solved by using deep learning techniques.

The deep neural network (DNN) is used in another variant of the artificial neural network (Chantry Matthew, 2021). It is composed of multi-layer architecture, which reconstructs the data from the original data collection. This architecture learns the features from the training instead of manual hand-crafting features. This feature learning automation results in high accuracy and recognition of a dataset. Continuous data like time series data problem solving are based on the Deep Neural network (DNN) for the identification of correlation between the distinct features (Ghosal et al., 2021, Chapman and William, 2022, Brecht and Bihlo, 2022). The behavior of systems, such as weather forecasting, is based on spatial and time conditions. The traditional machine learning approaches will not be suitable for this spatial and temporal context; hence the deep learning approaches can extract the features automatically and better understand and predict the features and results. The accuracy of deep learning systems can be increased with the help of correlation between the features and their representation. Deep learning methods are suitable for the time series data, which is multidimensional. Deep learning techniques were used to forecast the weather for analyzing the time series data.

2.2 暴雨预报

Rainstorms and floods contribute about 68% of all natural disasters in China. Indeed, China is one of the nations most prominently influenced by rainstorms and floods (Tao et al., 1979). For instance, an estimated 79 deaths and economic losses of 11.6 billion RMB were incurred as a result of the so-called "Beijing 721 extraordinary rainstorm" (Wu et al., 2017). Research has shown that the intensity and frequency of extreme strong precipitation have increased owing to the rise in global average temperature, which potentially enhances the risk of flooding too (Westra et al., 2013, 2014).

Before the 1990s, the method for forecasting rainstorms was mainly based on weather charts, hydrodynamic theory, and the forecaster's experience. After the 1990s, numerical models gradually came to prominence along with the rapid development of computers. In recent years, artificial intelligence technology has developed at an astonishing speed, and is rapidly emerging within the field of meteorology.

Wheeling (2020) evaluated the manifestation of generative adversarial networks within the fields of weather prediction and climate forecasting, and found that they produce better results in climate models. Shen et al. (2020) used a long short-term memory neural network to realize the forecasting of summertime seasonal-scale precipitation. Google uses a neural weather model for precipitation forecasting (MetNet) to predict the probability of precipitation in the continental United States (Sønderby et al., 2020).

在本研究之中，假设一个城市 C 某天 n 的降水总量 (Total Precipitation) 为 T_p，那么则有:

$$\begin{cases} C \text{ 城市第 } n \text{ 天将没有降雨;} & T_p \leq 0.1\text{mm} \\ C \text{ 城市第 } n \text{ 天将会有降雨;} & T_p > 0.1\text{mm} \\ C \text{ 城市第 } n \text{ 天将有大暴雨.} & T_p > 32\text{mm} \end{cases} \tag{1}$$

3 Data source and presentation

3.1 ERA5 再分析数据介绍

ERA5 是 ECMWF 对全球气候的第五代大气再分析。再分析将模型数据与来自世界各地的观测数据结合起来，形成一个全球完整的、一致的数据集。ERA5 取代了其前身 ERA-Interim 再分析。

ERA5 DAILY 提供每天 7 个 ERA5 气候再分析参数的汇总值：2 米空气温度、2 米露点温度、总降水量、平均海平面气压、表面气压、10 米 u 风分量和 10 米 v 风分量。(参数较多，其他参数可自行查看）此外，根据每小时的 2 米空气温度数据，计算出 2 米处的每日最低和最高空气温度。每日总降水值以每日总和给出。所有其他参数都以日平均数提供。

ERA5 的数据跨度为 1979 年 1 月 1 日至 2021 年 1 月 31 日。以每 $25km \times 25km$ 为采值范围，也就是这个范围内就有一个单位观测值。

3.2 数据预处理

数据以 *.nc* 为后缀的文献存储方式储存，使用 *Python* 的指定包可以直接读取，其中的数据均已每小时为记录间隔，我们需要从原始数据中提取我们需要的特征进行时序建模以及相应分析。在数据处理之中，这些数据可以主要分为两类。

- 不包含字段 *Level* 的数据，此数据的处理仅为四个维度，分别为
  (时间戳 (以小时为单位), 数据, 经度, 纬度)。
- 含有字段 *Level* 的数据，此类数据的纬度将会多一个纬度，在目前处理的数据中 *Level* 字段的大小范围均为 1-19，本研究所取的是中间变量，也就是 *Level* 为 10 的数据，其取值纬度为 (时间戳, 数据,10(*Level*), 经度, 纬度)。

在本研究中，我们需要进行分类的目标变量是每日总降水量 T_p，数据中降雨量的体现是以小时为单位的，假设地区 n 的某个小时 t 的降水量为 p, 那么每日降水量的定义如公式 2所示:

$$T_p = \sum_{i=0}^{t} p \tag{2}$$

而对于原始数据中的协变量，我们将使用数据的均值来进行清洗，例如我们假设某地区 n 的某个小时 t 的温度为 t, 那么协变量的取值 T 则如公式 3所示。

$$T = \frac{\sum_{i=0}^{t} t}{24} \tag{3}$$

相类似的，除了每日的目标变量降雨量使用每日总和数据，其他的协变量均使用均值数据清洗。

由于本次研究的任务仅是预测福建省内时候的节点的降雨预测，所以我们将使用经纬度将福建省内的数据切割出来，本研究选取的切割经度范围为 115.5-120.47，纬度范围为 23.33-28.19。根据此范围我们能够切割出一个 19×20 的网格点，其中每一个网格点包含了一个 1979 年 1 月 1 日至 2021 年 1 月 31 日的数据矩阵 $R_{\text{Time}\times\text{FeatureNum}}$。每一个格点为一个 excel 文件，最后可以得到一共 380 个格点的时间序列数据。

3.3 特征介绍

在 */mnt/usbshare2* 的目录下，一共包含了 ERA5 数据的 20 个特征。其具体特征如表格 1所示。表格中红色的变量为预测分类的目标变量。蓝色的是已经使用的协变量，**黑色的是由于算力问题还未清洗出的变量**。后续研究将会继续添加。

3.4 数据异常剔除

ERA5 降雨数据中存在着一些异常，例如在 2018 年 8 月 1 日这一天数据缺失，正常 *.nc* 的数据格式读取出来降雨数据的纬度为 (时间戳, 数据, 纬度, 经度), 而数据缺失的读取后的降雨纬度是 (时

表 1: ERA5 数据的特征汇总

Feature	***Variable Name***	***Introduction***
Total Precipitation	T_p	每日 24 小时的总降水量
Convective Precipitation	C_p	对流性降水，是来自对流云中的降水
10m v Component of Wind	W_{cv}	10 米处风的组成成分 v
10m u Component of Wind	W_{cu}	10 米处风的组成成分 u
2m Temperature	T_{2m}	空间 2m 处的温度
Temperature	T	每日温度
Relative Humidity	H_r	相对湿度 水蒸气的分压与水饱和蒸气压的百分比。
Specific Humidity	H_s	比湿 水汽的质量与该团空气总质量的比值。
Vorticity	V	涡量 用以描述流体的旋转情况。
Divergence	D	该参数是速度的水平散度。
Geopotential	G	地势 是指地表在水平和垂直面的起伏与规模
Large scale precipitaiton	P_s	大规模降水
Mean top net long wave radiation flux	F_r	平均顶层净长波辐射通量
Potential vorticity	P_v	位涡度
Surface latent heat flux	F_h	表面潜热通量
Surface net solar radiation	R_s	地表净太阳辐射
Surface net solar radiation clear sky	R_{sk}	晴空地表净太阳辐射
Surface pressure	P	表面压力

间戳,0,19(level), 纬度, 经度)，数据的纬度是不存在数据的。在后续的处理中我将这类数据插值为 0。除此之外，由于降雨量不可能为负值，所以我们将存在负值的数据替换成 0。即：

$$T_p < 0 \Rightarrow T_p = 0 \tag{4}$$

4 Method

4.1 时间序列分类问题定义

Time series classification (TSC) is a form of machine learning where the features of the input vector are real valued and ordered. This scenario adds a layer of complexity to the problem, as important characteristics of the data can be missed by traditional algorithms. Over recent years, a

new set of TSC algorithms have been developed which have made significant improvement over the previous state of the art (Bagnall et al. 2017).

The main focus has been on univariate TSC, i.e. the problem where each case has a single series and a class label. In reality, it is more common to encounter multivariate TSC (MTSC) problems where the time series for a single case has multiple dimensions. Human activity recognition, diagnosis based on electrocardiogram (ECG), electroencephalogram (EEG) and Magnetoencephalography (MEG), and systems monitoring problems are all inherently multivariate. Despite this, much less consideration has been given to MTSC than the univariate case. The UCR archive has provided a valuable resource for univariate TSC, and its existence may explain the growth of algorithm development for this task. Until recently, there were few resources for MTSC. An archive of 30 MTSC problems released in Bagnall et al. (2018) has made comparison of algorithms easier and will we hope spur further research in this field. We compare recently proposed bespoke MTSC algorithms to simple adaptations of univariate approaches on the 26 equal length problems in the UEA MTSC archive. We find that dynamic time warping (DTW) is still hard to beat in MTSC, but that four algorithms are significantly more accurate than this benchmark on this archive. It is dangerous to infer too much from results achieved on 26 problems collected in an arbitrary way across a wide range of problem domains. Nevertheless, some advice for practitioners for a starting point in an analysis is always helpful. We conclude that one recently published algorithm, ROCKET (Dempster et al. 2020), is our recommended choice due to high overall accuracy and remarkably fast training time.

4.2 模型的输入与输出

4.2.1 单模型的输入与输出

模型的输入特征如表格 4.2.1所示，假设某个城市其中一天 t 的数据的第一个特征为 $X^1(t)$, 那么其输入模型的数据矩阵则为公式 5(矩阵 A): 其中模型的输入为前七天的特征值 (表格 4.2.1)，模型的输出则为第八天的站点一天总降水量，如公式 6所示，即为矩阵 $B.T$(矩阵 B 的转置)。数据集采集为迭代采集，采集方法如图 4.2.1所示。

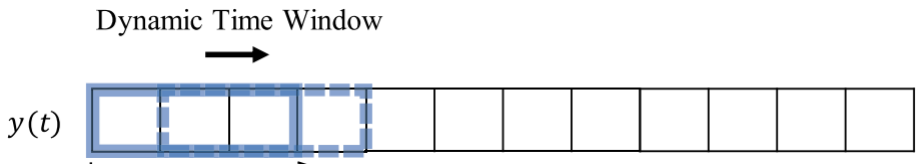

图 1: 数据迭代采集

$$A = \begin{bmatrix} x_1^1(t) & X_1^2(t) & \cdots & X_1^7(t+7) \\ X_2^1(t+1) & X_2^2(t+1) & \cdots & X_2^7(t+8) \\ \vdots & \vdots & \ddots & \vdots \\ X_{n+7}^1(t+n) & X_{n+7}^2(t+n) & \cdots & X_{n+7}^7(t+n+7) \end{bmatrix} \tag{5}$$

$$B = \begin{bmatrix} y_{t+1} & y_{t+2} & \cdots & y_{t+n} \end{bmatrix} \tag{6}$$

模型的输出 (label) 我们则作为一个二分类问题，那么针对于晴雨预报与特大暴雨预报就可以分为两个二分类问题，即:

$$\begin{cases} y = 1 & T_p > 0.1mm | T_p > 32mm \\ y = 0 & T_p \leq 0.1mm | T_p \leq 32mm \end{cases} \tag{7}$$

对于机器学习来说，模型的输入不一定是越多越好，所以选取合适的特征能够适当提升模型的效果。本次实验中我们还在采集了三种模型的输入以比较模型的效果:

1. 仅采用过去七天的总降水量数据;
2. 采用过去七天的总降水量、风的组成成分 (uv)，对流性降水，空间两米的温度 (表格 4.2.1的蓝色与红色的字段);
3. 表格 4.2.1的所有特征 (包括绿色字段)

对于这些数据我们将分别记录模型效果并进行比较，事实上，在时间序列预测领域，我们一般使用 $Spearman$、$Pearson$、$Kendall$、MIC 系数等系数量化特征序列与目标序列之间的相关性。时间序列分类领域的使用目前较少，本来对于实验中是有这方面的安排，但是由于数据并未完全清洗得到，所以暂时搁置。对于此方面的实验后续会进行补充。

4.2.2 分解模型拼接模型的输入与输出

本研究尝试了在单降雨序列中尝试对序列进行分解的数据增强操作。其中每七天的序列被分解成三条子序列 $(k = 3)$，即模型的输入变成矩阵 8, 模型的输出不变:

$$\begin{bmatrix} x_1^1(t) & X_1^2(t) & \cdots & IMF\{X_1^1(t)\} & \cdots & IMF_1\{X^7(t+7)\} \\ X_2^1(t+1) & X_2^2(t+1) & \cdots & IMF\{X_2^1(t)\} & \cdots & IMF_2\{X^7(t+8)\} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ X_{n+7}^1(t+n) & X_{n+7}^2(t+n) & \cdots & IMF_{n+7}\{X^1(t+n)\} & \cdots & IMF_{n+7}\{X^7t+n+7)\} \end{bmatrix} \tag{8}$$

表 2: 模型的输入与输出

Feature	*Variable Name*	*Introduction*
Total Precipitation	T_p	每日 24 小时的总降水量
Convective Precipitation	C_p	对流性降水，是来自对流云中的降水
10m v Component of Wind	W_{cv}	10 米处风的组成成分 v
10m u Component of Wind	W_{cu}	10 米处风的组成成分 u
2m Temperature	T_{2m}	空间 2m 处的温度
Temperature	T	每日温度
Relative Humidity	H_r	相对湿度 水蒸气的分压与水饱和蒸气压的百分比。
Specific Humidity	H_s	比湿 水汽的质量与该团空气总质量的比值。

4.3 机器学习

4.4 线性分类模型

4.4.1 Logistic Regression

多重线性回归模型要求因变量是连续型的正态分布变量，且自变量与因变量呈线性关系。当因变量是分类变量，且自变量与因变量不呈线性关系时，就不能确足多重线性回归模型的适用条件。此时，处理该类资料常用 Logistic 回归模型。Logistic 回归分析属于非线性回归，它是研究因变量为二项分类或多项分类结果与某些影响因素之间关系的一种多重回归分析方法。

在疾病的病因学研究中，经常需要分析疾病的发生与各危险因素之间的定量关系。比如，研究食管癌的发生与吸烟、饮酒、不良饮食习惯等危险因素的关系。如果采用多重线性回归分析，由于因变量 y 为二分类变量（通常取值 0 或 1），不满足正态分布和方差齐等应用条件，若强行使用线性回归分析，其预测值可能会大于 1 或小于，而无法解释。在流行病学研究中，虽然可以用 Mantel-Haenszel 分层分析方法分析多个因素的混杂作用，但这种经典方法有其局限性，随着混杂因素的增加，分层越来越细，致使每层内的数据越来越少，使相对危险度的估计产生困难。Logistic 回归模型较好地解决了上述问题，已经成为医学研究，特别是流行病学病因研究中最常用的分析方法之一。

logistic 回归是一种广义线性回归（Generalized linear model），因此与多重线性回归分析有很多相同之处。它们的模型形式基本上相同，都具有 $wx+b$，其中 w 和 b 是待求参数，其区别在于他们的因变量不同，多重线性回归直接将 $wx+b$ 作为因变量，即 $y=wx+b$，而 logistic 回归则通过函数 L 将 $wx+b$ 对应一个隐状态 p，$p=L(wx+b)$, 然后根据 p 与 $1-p$ 的大小决定因变量的值。如果 L 是 logistic 函数，就是 logistic 回归，如果 L 是多项式函数就是多项式回归。logistic 回归的因变量可以是二分类的，也可以是多分类的，但是二分类的更为常用，也更加容易解释，多类可以使用 softmax 方法进行处理。实际中最为常用的就是二分类的 logistic 回归。

Logistic 损失函数是用于衡量预测输出值与实际值有多接近的损失函数，常用于 Logistic 回归

中。它的公式为:

$$L(\hat{y}, y) = -ylog(\hat{y}) - (1-y)log(1-\hat{y})$$

该损失函数是一个凸函数，可以用梯度下降法求得使损失函数最小的参数。Logistic 损失函数也被称为对数似然损失函数或交叉熵损失函数。除了 Logistic 损失函数，还有 0-1 损失函数和对数损失函数等。在 Logistic 回归中，为了训练模型的参数 w 和 b，需要定义一个 Cost Function，指的是 sigmoid 函数，sigmoid 函数为:

$$sigmoid = \frac{1}{1+e^{-x}}$$

。

4.4.2 Support Vector Classificar

在机器学习中，支持向量机（英语：Support Vector Machine，常简称为 SVM，又名支持向量网络）是在分类与回归分析中分析数据的监督式学习模型与相关的学习算法。给定一组训练实例，每个训练实例被标记为属于两个类别中的一个或另一个，SVM 训练算法创建一个将新的实例分配给两个类别之一的模型，使其成为非概率二元线性分类器。SVM 模型是将实例表示为空间中的点，这样映射就使得单独类别的实例被尽可能宽的明显的间隔分开。然后，将新的实例映射到同一空间，并基于它们落在间隔的哪一侧来预测所属类别，如图 2。除了进行线性分类之外，SVM 还可以使用所

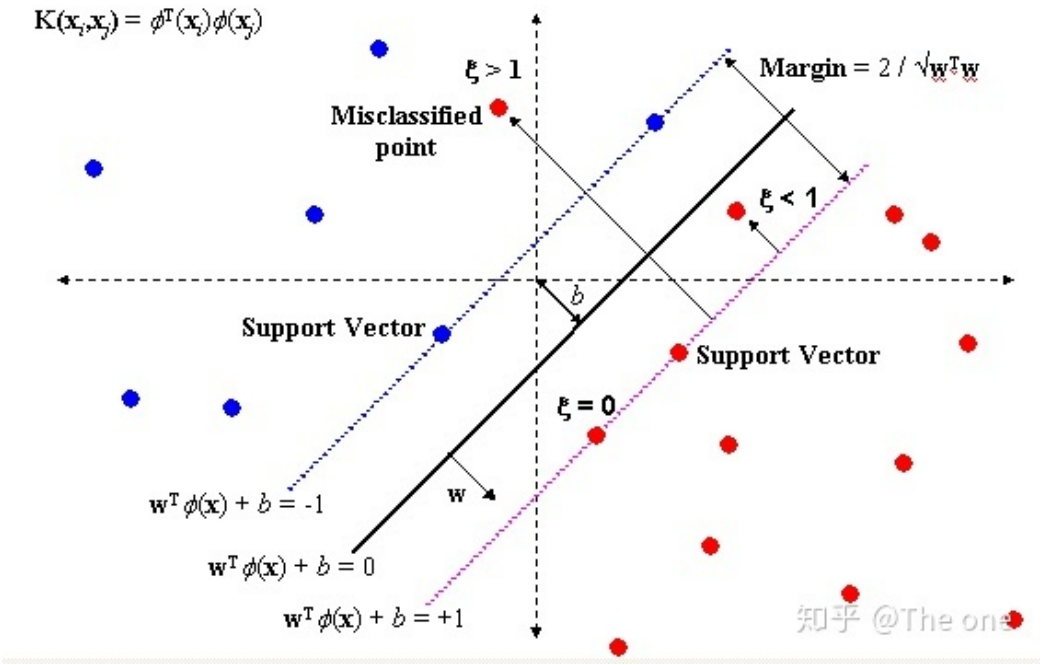

图 2: SVC 中的分类

谓的核技巧有效地进行非线性分类，将其输入隐式映射到高维特征空间中。

当数据未被标记时，不能进行监督式学习，需要用非监督式学习，它会尝试找出数据到簇的自然聚类，并将新数据映射到这些已形成的簇。将支持向量机改进的聚类算法被称为支持向量聚类，当数据未被标记或者仅一些数据被标记时，支持向量聚类经常在工业应用中用作分类步骤的预处理。

更正式地来说，支持向量机在高维或无限维空间中构造超平面或超平面集合，其可以用于分类、回归或其他任务。直观来说，分类边界距离最近的训练资料点越远越好，因为这样可以缩小分类器的泛化误差。尽管原始问题可能是在有限维空间中陈述的，但用于区分的集合在该空间中往往线性不可分。为此，有人提出将原有限维空间映射到维数高得多的空间中，在该空间中进行分离可能会更容易。为了保持计算负荷合理，人们选择适合该问题的核函数 $k(x, y)$ 来定义 SVM 方案使用的映射，以确保用原始空间中的变量可以很容易计算点积。[3] 高维空间中的超平面定义为与该空间中的某向量的点积是常数的点的集合。定义超平面的向量可以选择在数据集中出现的特征向量 x_i 的图像的参数 α_i 的线性组合。通过选择超平面，被映射到超平面上的特征空间中的点集 x 由以下关系定义：$\sum_i \alpha_i k(x_i, x) = \text{constant}$. 注意，如果随着 y 逐渐远离 x $k(x, y)$ 变小，则求和中的每一项都是在衡量测试点 x 与对应的数据基点 x_i 的接近程度。这样，上述内核的总和可以用于衡量每个测试点相对于待分离的集合中的数据点的相对接近度。

4.4.3 Broad Learning System

Inspired by Random Vector Function Link Neural Network (RVFLNN), a brand new neural network, Broad Learning System (BLS), is proposed by Chen et al. in 2017 . Compared with traditional Machine Learning and Deep Learning algorithms, BLS develops a simple structure instead of the setting of complex layers; thus, it has a fast training process which advance the whole effectiveness of the model. Figure 3 indicates the structure of the BLS, which demonstrate the procedure from different parts. Assume that a training set $X \in \mathbb{R}^{N \times M}$ provides the features which input to n mapped feature groups, each group respectively generates k nodes. The mapped feature Z_i is derived from the feature mapping function:

$$Z_i = \phi\left(XW_{e_i} + \beta_{e_i}\right), \quad i = 1, 2, \ldots, n \tag{9}$$

where W_{e_i} and β_{e_i} are randomly initialized. $\phi(\cdot)$ is the activation function of feature mapping. All of the feature nodes can be combined as $Z^n = [Z_1, Z_2, \ldots, Z_n]$. Then, the j th group of enhancement node can be denoted as:

$$H_j = \zeta_j\left(Z^n W_{h_j} + \beta_{h_j}\right), \quad j = 1, 2, \ldots, m \tag{10}$$

where ζ is the activation function of enhancement nodes. Similarly, all of the enhancement nodes can be combined as $H^m = [H_1, H_2, \ldots, H_m]$. Then, the output Y of the BLS can be represented as:

$$\begin{aligned} Y &= [Z_1, Z_2, \ldots, Z_n \mid H_1, \ldots, H_m] W^m \\ &= [Z^n \mid H^m] W^m \end{aligned} \tag{11}$$

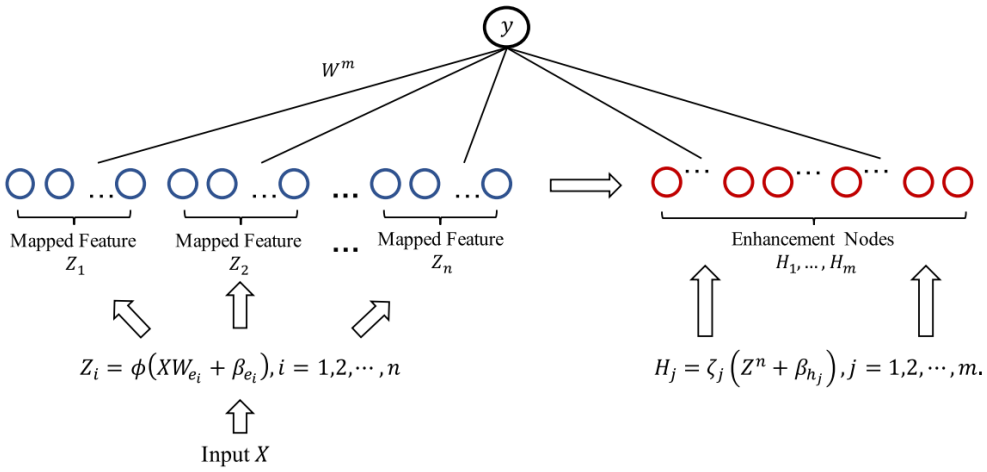

图 3: BLS 宽度学习

4.5 树形模型

4.5.1 Decision Tree

机器学习中，决策树是一个预测模型；他代表的是对象属性与对象值之间的一种映射关系。树中每个节点表示某个对象，而每个分叉路径则代表某个可能的属性值，而每个叶节点则对应从根节点到该叶节点所经历的路径所表示的对象的值。决策树仅有单一输出，若欲有复数输出，可以建立独立的决策树以处理不同输出。数据挖掘中决策树是一种经常要用到的技术，可以用于分析数据，同样也可以用来作预测。从数据产生决策树的机器学习技术叫做决策树学习，通俗说就是决策树。

决策树学习也是数据挖掘中一个普通的方法。在这里，每个决策树都表述了一种树型结构，它由它的分支来对该类型的对象依靠属性进行分类。每个决策树可以依靠对源数据库的分割进行数据测试。这个过程可以递归式的对树进行修剪。当不能再进行分割或一个单独的类可以被应用于某一分支时，递归过程就完成了。决策树同时也可以依靠计算条件概率来构造。

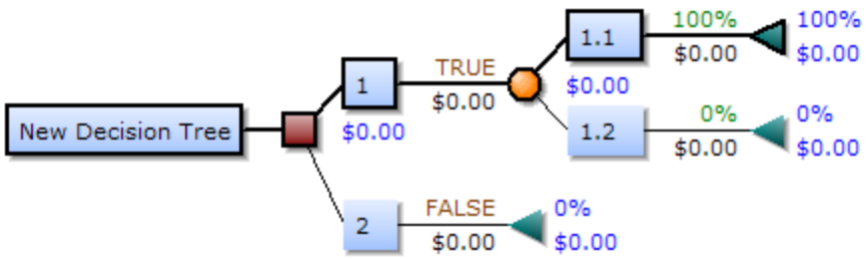

图 4: 决策树的构造

$$I_E(i) = -\sum_{j=1}^{m} f(i,j) \log_2 f(i,j) \tag{12}$$

C4.5 和 C5.0 生成树算法使用熵。这一度量是基于资讯科学理论中熵的概念。

4.5.2 Random Forest

在机器学习中，随机森林是一个包含多个决策树的分类器，并且其输出的类别是由个别树输出的类别的众数而定。

这个术语是 1995 年由贝尔实验室的何天琴所提出的随机决策森林（random decision forests）而来的。然后 Leo Breiman 和 Adele Cutler 发展出推论出随机森林的算法。而"Random Forests" 是他们的商标。这个方法则是结合 Breimans 的"Bootstrap aggregating" 想法和 Ho 的"random subspace method" 以建造决策树的集合。

随机森林训练算法把 Bagging 的一般技术应用到树学习中。给定训练集 $X = x_1, x_2, \cdots, x_n$ 和目标 $Y = y_1, y_2, \cdots, y_n$，Bagging 方法重复（$B$ 次）从训练集中有放回地采样，然后在这些样本上训练树模型:

For $b = 1, \ldots, B$:

1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.

2. Train a classification or regression tree f_b on X_b, Y_b.

在训练结束之后，对末知样本 x 的预测可以通过对 x 上所有单个回归树的预测求平均来实现:

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b(x') \tag{13}$$

或者在分类任务中选择多数投票的类别。这种 bagging 方法在不增加偏置的情况下降低了方差，从而带来了更好的性能。这意味着，即使单个树模型的预测对训练集的噪声非常敏感，但对于多个树模型，只要这些树并不相关，这种情况就不会出现。简单地在同一个数据集上训练多个树模型会产生强相关的树模型（甚至是完全相同的树模型）。Bootstrap 抽样是一种通过产生不同训练集从而降低树模型之间关联性的方法。此外，x' 上所有单个回归树的预测的标准差可以作为预测的不确定性的估计:

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B} \left(f_b(x') - \hat{f}\right)^2}{B-1}}. \tag{14}$$

样本或者树的数量 B 是一个自由参数。通常使用几百到几千棵树，这取决于训练集的大小和性质。使用交叉验证，或者透过观察 out-of-bag 误差（那些不包含 x_i 的抽样集合在样本 x_i 的平均预测误差），可以找到最优的 B 值。当一些树训练到一定程度之后，训练集和测试集的误差开始趋于平稳。

4.5.3 GBDT

梯度提升决策树（Gradient Boosting Decision Tree，GBDT）是一种基于 Boosting 集成思想的加法模型，训练时采用前向分布算法进行贪婪的学习，每次迭代都学习一棵 CART 树来拟合之前 $t-1$ 棵树的预测结果与训练样本真实值的残差。

梯度提升决策树算法（GBDT）如下：

输入: 训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$ $x_i\in X\subseteq R^n, y_i\in Y\subseteq R$; 损失函数 $L(y,f(x))$ ； 输出: 回归树 $\hat{f}(x)$。

1. 始化弱学习器

$$f_0(x)=\arg\min_c\sum_{i=1}^{N}L\left(y_i,c\right) \tag{15}$$

2. 假设取损失函数为平方损失。因为平方损失函数是一个凸函数，直接对 c 求导:

$$\sum_{i=1}^{N}\frac{\partial L\left(y_i,c\right)}{\partial c}=\sum_{i=1}^{N}\frac{\partial\left(\frac{1}{2}\left(y_i-c\right)^2\right)}{\partial c}=\sum_{i=1}^{N}\left(c-y_i\right) \tag{16}$$

3. 令导数等于 0 ，得:

$$c=\sum_{i=1}^{N}y_i/N \tag{17}$$

4. 所以初始化时，c 取值为所有训练样本标签值的均值。此时得到初始学习器:

$$f_0(x)=c \tag{18}$$

迭代训练 $m=1,2,\cdots,M$ 棵树

5. 对每个样本 $i=1,2,\cdots,N$ ，计算负梯度，即残差:

$$r_{mi}=-\left[\frac{\partial L\left(y_i,f\left(x_i\right)\right)}{\partial f\left(x_i\right)}\right]_{fx)=f_{m-1}(x)} \tag{19}$$

6. 将上步得到的残差 r_{mi} 作为样本新的真实值，并将数据 (x_i,r_{im}) $i=1,2,\cdots,N$ 作为下棵F的训练数据，得到一颗新回归树，其对应的叶子节点区域 R_{mj} $j=1,2,\cdots,J$ 。其中 J 为回归树的叶子结点的个数。

7. 对 $j=1,2,\cdots,J$ 个叶子结点，计算最佳拟合值:

$$c_{mj}=\arg\min_c\sum_{x_i\in R_{mj}}L\left(y_i,f_{m-1}\left(x_i\right)+c\right)\quad>c_{mj} \tag{20}$$

8. 更新强学习器:

$$f_m(x)=f_{m-1}(x)+\textstyle\sum_{j=1}^{J}c_{mj}I\left(x\in R_{mj}\right) \tag{21}$$

9. 得到最终学习器 GBDT：

$$\hat{f}(x) = f_M(x) = f_0(x) + \sum_{m=1}^{M}\sum_{j=1}^{J} c_{mj} I\left(x \in R_{mj}\right) \tag{22}$$

4.5.4 XGBoost

XGBoost（eXtreme Gradient Boosting）极致梯度提升，是一种基于 GBDT 的算法或者说工程实现。

XGBoost 的基本思想和 GBDT 相同，但是做了一些优化，比如二阶导数使损失函数更精准；正则项避免树过拟合；Block 存储可以并行计算等。

XGBoost 具有高效、灵活和轻便的特点，在数据挖掘、推荐系统等领域得到广泛的应用。

4.5.5 Etr

Extremely Randomized Trees Classifier(极度随机树) 是一种集成学习技术，它将森林中收集的多个去相关决策树的结果聚集起来输出分类结果。极度随机树的每棵决策树都是由原始训练样本构建的。在每个测试节点上，每棵树都有一个随机样本，样本中有 k 个特征，每个决策树都必须从这些特征集中选择最佳特征，然后根据一些数学指标 (一般是基尼指数) 来拆分数据。这种随机的特征样本导致多个不相关的决策树的产生。

在构建森林的过程中，对于每个特征，计算用于分割特征决策的数学指标（如使用基尼指数）的归一化总缩减量，这个值称为基尼要素的重要性。基尼重要性按降序排列后，可根据需要选择前 k 个特征。

4.5.6 Lgb

常用的机器学习算法，例如神经网络等算法，都可以以 mini-batch 的方式训练，训练数据的大小不会受到内存限制。而 GBDT 在每一次迭代的时候，都需要遍历整个训练数据多次。如果把整个训练数据装进内存则会限制训练数据的大小；如果不装进内存，反复地读写训练数据又会消耗非常大的时间。尤其面对工业级海量的数据，普通的 GBDT 算法是不能满足其需求的。

LightGBM 提出的主要原因就是为了解决 GBDT 在海量数据遇到的问题，让 GBDT 可以更好更快地用于工业实践。

4.6 深度学习

本章节中所展现的方法均是调用 Python 中 HyperTS 库自动分类完成，对于模型的结构均是由 API 自动调整得到。

4.6.1 RNN

RNN 用于处理序列数据。在传统的神经网络模型中，是从输入层到隐含层再到输出层，层与层之间是全连接的，每层之间的节点是无连接的。但是这种普通的神经网络对于很多问题却无能无力。例如，你要预测句子的下一个单词是什么，一般需要用到前面的单词，因为一个句子中前后单词并不是独立的。RNN 之所以称为循环神经网路，即一个序列当前的输出与前面的输出也有关。具体的表现形式为网络会对前面的信息进行记忆并应用于当前输出的计算中，即隐藏层之间的节点不再无连接而是有连接的，并且隐藏层的输入不仅包括输入层的输出还包括上一时刻隐藏层的输出。理论上，RNN 能够对任何长度的序列数据进行处理。但是在实践中，为了降低复杂性往往假设当前的状态只与前面的几个状态相关。RNN 的基本结果如图 5所示。

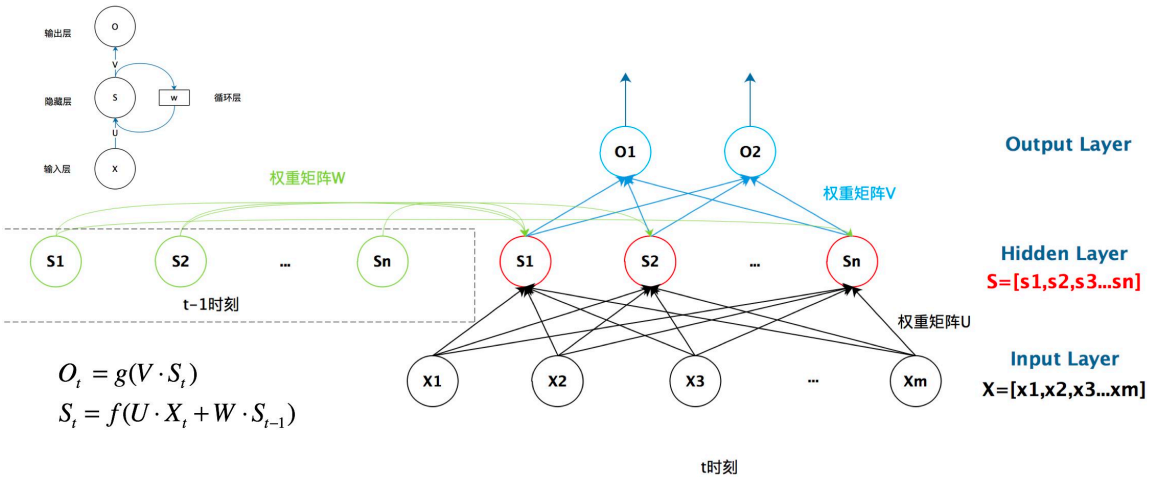

图 5: RNN 的模型结构图

4.6.2 LSTM

长短期记忆（Long short-term memory, LSTM）是一种特殊的 RNN，主要是为了解决长序列训练过程中的梯度消失和梯度爆炸问题。简单来说，就是相比普通的 RNN，LSTM 能够在更长的序列中有更好的表现。

传统的循环神经网络 RNN 可以通过记忆体实现短期记忆进行连续数据的预测，但是当连续数据的序列变长时，会使展开时间步过长，在反向传播更新参数时，梯度要按照时间步连续相乘，很容易导致梯度消失，所以 LSTM 就出现了。

相比 RNN 只有一个传递状态 h^t ，LSTM 有两个传递状态，一个 c^t (cell state，细胞态，长期记忆)，和一个 h^t (hidden state，隐态，记忆体，短期记忆) 。(Tips: RNN 中的 h^t 对于 LSTM 中的 c^t) 。其中对于传递下去的 c^t 改变得很慢，通常输出的 c^t 是上一个状态传过来的 c^{t-1} 加上一些数值，而 h^t 则在不同节点下往往会有很大的区别。下面具体对 LSTM 的内部结构来进行剖析：首先使用 LSTM 的当前输入 x^t 和上一个状态传递下来的 h^{t-1} 拼接训练得到四个状态。其中，z^f z^i z^o

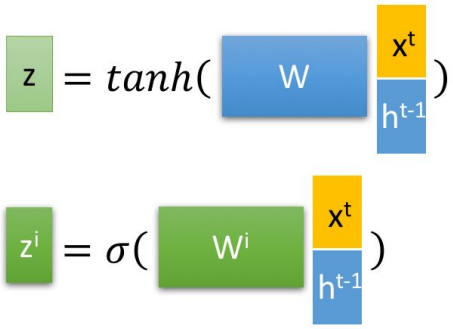

图 6: LSTM 模型状态 1,2

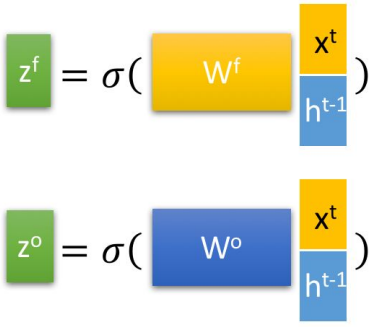

图 7: LSTM 模型状态 3,4

是由拼接向量乘以权重矩阵之后，再通过一个 sigmoid 激活函数转换成 0 到 1 之间的数值，来作为一种门控状态，分别表示: z^i 表示输入门，z^f 表示遗忘门，z^o 表示输出门。而 z 则是将结果通过一个 tanh 激活函数将转换成-1 到 1 之间的值（这里使用 tanh 是因为这里是将其做为输入数据，而不是门控信号），z 表示候选态，由上一时刻的短期记忆 h^{t-1} 和当前时刻的输入特征 x^t 决定。

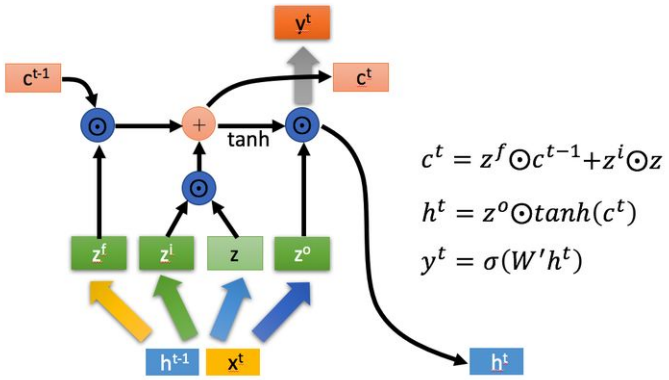

图 8: LSTM 的流程图

4.6.3 GRU

门控循环单元（gated recurrent unit，GRU）是 LSTM 网络的一种效果很好的变体，它较 LSTM 网络的结构更加简单，而且效果也很好，因此也是当前非常流行的一种网络。GRU 既然是 LSTM 的变体，因此也是可以解决 RNN 网络中的长依赖问题。

在 LSTM 中，引入了三个门函数：输入门、遗忘门和输出门。而在 GRU 中，只有两个门：更新门和重置门。模型的结构如图 9所示：

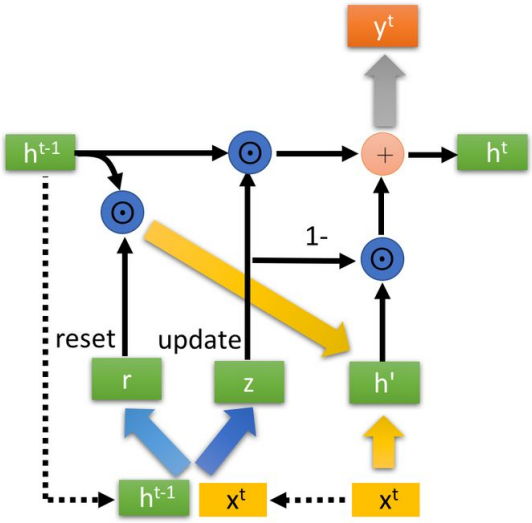

图 9: GRU 中的遗忘门

4.6.4 LSTNet

多变量时间序列预测常常面临一个重大的研究挑战，即如何捕捉和利用多变量之间的动态相关性。具体来说，现实世界的应用程序通常需要短期和长期重复模式的混合。一个成功的时间序列预测模型应该捕捉两种重复的模式，以便准确预测。

传统的方法，如自回归方法中的大量工作在这方面存在不足，因为它们大多数都没有区分这两种模式，也没有明确、动态地对它们的相互作用进行建模。解决现有方法在时间序列预测中的局限性是本文的主要重点，为此，LSTNet 框架利用了深度学习研究的最新发展。

利用了卷积层和循环层的优势，前者可以发现多维输入变量之间的局部依赖模式，后者可以捕获复杂的长期依赖。利用输入时间序列信号的周期性特性，设计了一种新颖的递归结构，即递归跳跃，用于捕获非常长期的依赖模式，并使优化更容易。LSTNet 在非线性神经网络部分的基础上加入了传统的自回归线性模型，使得非线性深度学习模型对于尺度违规变化的时间序列具有更强的鲁棒性。在对真实季节时间序列数据集的实验中，该模型的性能优于传统的线性模型和 GRU 递归神经网络。

- **Convolutional Component**

  LSTnet 的第一层结构是没有池化层的卷积神经网络，该结构可以提取在时间维度中的短期特征，并且提取变量之间的局部依赖性。卷积层包含多个过滤器，过滤器宽度为 w，高度为 n(高

度设置为与变量的个数相同)。第 k 个过滤器扫描输入矩阵 X 并产生:

$$h_k = RELU(W_k X + B_k) \tag{23}$$

- **Recurrent Component**

  卷积层的输出同时进入一个带有 *skipconnection* 的 GRU。并使用 $RELU$ 函数作为隐藏的更新激活函数。公式如下:

$$\begin{aligned} r_t &= \sigma\left(x_t W_{xr} + h_{t-1} W_{hr} + b_r\right) \\ u_t &= \sigma\left(x_t W_{xu} + h_{t-1} W_{hu} + b_u\right) \\ c_t &= RELU\left(x_t W_{xc} + r_t \odot \left(h_{t-1} W_{hc}\right) + b_c\right) \\ h_t &= (1 - u_t) \odot h_{t-1} + u_t \odot c_t \end{aligned} \tag{24}$$

- **Recurrent-skip Component** 带有 GRU 和 LSTM 单元的递归层经过精心设计，以记住历史信息，从而了解相对长期的依赖关系。然而由于梯度消失问题，GRU 和 LSTM 在实践中通常不能捕捉到非常长期的相关性。我们提出通过一种新的递归跳过组件来缓解这个问题，该组件利用了真实数据中的周期性模式。例如，每天的用电量和交通情况都包含明显的模式。如果我们想要预测今天 t 时的电量消耗，季节预测模型的一个经典技巧是利用同期的记录（比如周期是一年，就是去年同期），而不是最近的记录。

  主要是因为一个周期 (24 小时) 的长度往往比较长，以及随后的优化问题，这种类型的依赖关系很难被现成的循环单元捕获。

  受此技巧的启发，LSTnet 开发了一个具有时间跳跃连接的递归结构，以扩展信息流的时间跨度，从而简化优化过程。具体地，在当前的隐藏单元和相邻周期的同一相位的隐藏单元之间添加跳转链接。更新过程可以使用公式表示为:

$$\begin{aligned} r_t &= \sigma\left(x_t W_{xr} + h_{t-p} W_{hr} + b_r\right) \\ u_t &= \sigma\left(x_t W_{xu} + h_{t-p} W_{hu} + b_u\right) \\ c_t &= \mathrm{RELU}\left(x_t W_{xc} + r_t \odot \left(h_{t-p} W_{hc}\right) + b_c\right) \\ h_t &= (1 - u_t) \odot h_{t-p} + u_t \odot c_t \end{aligned} \tag{25}$$

- **时间注意力层**

  Recurrent-skip Component 需要一个预先定义好的超参数 p(表示周期)，这对于非季节性的时间序列预测是不利的，或者其周期长度是随时间变化的，或不同序列的周期本身也不同。为了缓解这一问题，我们考虑了另一种方法，即注意力机制，它学习输入矩阵每个窗口位置的隐藏表示的加权组合。具体的，在当前时间戳 t 时刻的注意力权重 $a(t)$(该值维度为 q*1) 可以按如下公式计算:

$$\boldsymbol{\alpha}_t = \mathrm{AttnScore}\left(H_t^R, h_{t-1}^R\right) \tag{26}$$

这里就是 many-to-one 的 attention 机制，return sequences，然后最后一个时间步和其它所有的时间步直接做 attention 计算就可以了，得到的 context vector 和最后一个时间步的 hidden state 做 concat 然后接一个 linear 层就可以了。

$$h_t^D = W\left[c_t; h_{t-1}^R\right] + b \tag{27}$$

- **自回归组件**

  LSTNet 的最终预测分解为一个线性部分（主要关注局部尺度问题）和一个包含重复模式的非线性部分。在 LSTNet 体系结构中，我们采用经典的自回归 (AR) 模型作为线性分量。将 AR 组件的预测结果表示为 $hL(t)$ ，该值维度为 n ；将 AR 模型的系数表示为 $W(ar)$ ，该值维度为 $q(ar)$ $b(ar)$ 维度为 $1 * 1$ ，其中 $q\hat{}(ar)$ 是输入矩阵上的输入窗口的大小。注意，在我们的模型中，所有维都共享相同的一组线性参数。AR 模型可以按如下公式表示:

$$h_{t,i}^L = \sum_{k=0}^{q^{ar}-1} W_k^{ar} \boldsymbol{y}_{t-k,i} + b^{ar} \tag{28}$$

  将神经网络部分的输出与 AR 分量进行求和，得到 LSTNet 的最终预测结果:

$$\hat{Y}_t = h_t^D + h_t^L \tag{29}$$

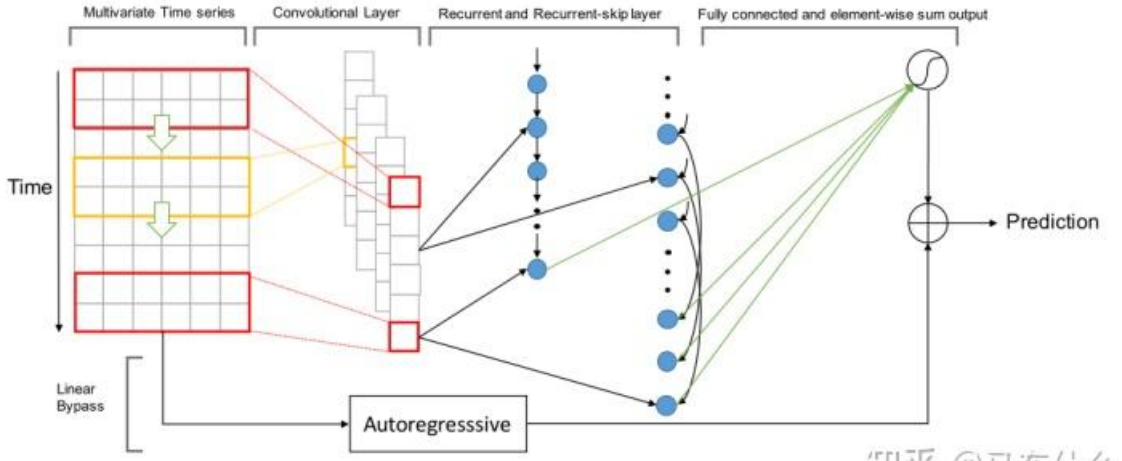

Figure 2: An overview of the Long- and Short-term Time-series network (LSTNet)

图 10: LSTNet 的结构图

4.7 Variational mode decomposition

Konstantin et al. proposed an adaptive, completely non-recursive VMD based on Empirical Mode Decomposition (EMD). This technique has the advantage of determining the number of modal

decompositions. VMD can decompose a complex signal into a series of sub-signals (i.e.modes) with a specific bandwidth in the frequency domain, thus exploring the characteristics of complex non-linear signals. In this process, the algorithm can adaptively match the optimal center frequency and the limited bandwidth of each mode, effectively separate the intrinsic mode function (IMF) and divide the signal into frequency domains, then get the effective decomposition component of the signal. The essence of VMD is to construct and solve variational proble.

1. Assuming that a multicomponent signal consis of K modes components $w_k(t)$ with limited bandwidth. The decomposition process using VMD can be transformed into the following optimization problems. The goal of optimization is to find K modes so that the total bandwidth of all modes is the smallest. The constraint is that the sum of all modes equals the input signals. The concrete construction steps are as follows:

$$\begin{aligned} \min_{\{v_k\},\{w_k\}} &\left\{ \sum_k \left\| \partial_{\mathrm{t}} \left[\left(\delta(t) + \frac{j}{\pi t} \right) * v_k(t) \right] \mathrm{e}^{-jw_k t} \right\|_2^2 \right\}. \\ &\text{s.t.} \qquad \sum_k v_k = s \end{aligned} \tag{30}$$

   where, $\{v_k\} = \{v_1, \cdots, v_K\}$ represents the decomposed IMF components, $\{\omega_k\} = \{\omega_1, \cdots, \omega_K\}$ represents the center frequency of each component, s represents the original input signal.

2. The lagrange multiplier τ(t) and quadratic penalty factor α are introduced to transform the above-mentioned constrained variational problem into an unconstrained optimization problem, namely:

$$\begin{aligned} \mathrm{L}\left(\{v_k\}, \{\omega_k\}, \tau\right) = \alpha \sum_k &\left\| \partial_{\mathrm{t}} \left[\left(\delta(t) + \frac{j}{\pi t} \right) * v_k(t) \right] \mathrm{e}^{-jw_k t} \right\|_2^2 \\ + &\left\| s(t) - \sum_k v_k(t) \right\|_2^2 + \left\langle \tau(\mathrm{t}), s(\mathrm{t}) - \sum_k v_k(t) \right\rangle \end{aligned} \tag{31}$$

3. The alternate direction multiplier method (ADMM) is used to continuously update each component and its center frequency to solve the saddle point of the unconstrained model, which is the best solution to the original problem [18]. According to the ADMM optimization method, the iteration formula of all components is:

$$\begin{aligned} \hat{v}_k^{\mathrm{n}+1}(\omega) &= \frac{\hat{\mathrm{s}}(\omega) - \sum_{i \neq k} \hat{v}_i^{\mathrm{n}+1}(\omega) + \hat{\tau}(\omega)/2}{1 + 2\alpha \left(\omega - \omega_k\right)^2} \\ \omega_k^{n+1} &= \frac{\int_0^\infty \omega \left| \hat{v}_k^{\mathrm{n}+1}(\omega) \right|^2 d\omega}{\int_0^\infty \left| \hat{v_k}^{\mathrm{n}+1}(\omega) \right|^2 d\omega} \\ \hat{\lambda}^{n+1}(\omega) &= \hat{\lambda}^n(\omega) + \tau \left(\hat{\mathrm{s}}(\omega) - \sum_k \hat{v}_k^{\mathrm{n}+1}(\omega) \right) \end{aligned} \tag{32}$$

where $\hat{v}_k^{n+1}(\omega), \hat{s}(\omega), \hat{\lambda}(\omega)$ denote fourier transform of $v_k^{n+1}(t), s(t), \lambda(t)$ respectively.

Repeat the above steps until the iteration stop condition $\sum_k \left\|\hat{v}_k^{n+1} - \hat{v}_k^n\right\|_2^2 / \left\|\hat{v}_k^n\right\|_2^2 < \varepsilon$ is satisfied, where n represent the number of iterations, τ denotes the update coefficient.

4.7.1 分解集成模型

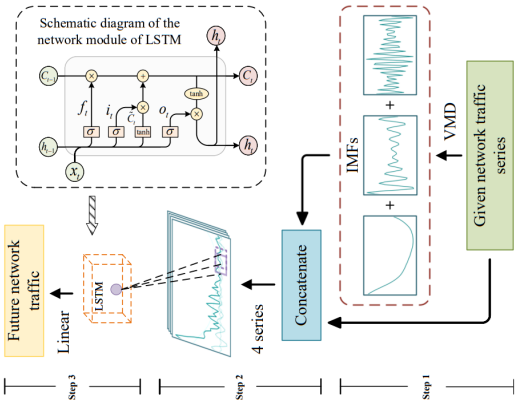

图 11: VMD-LSTM

在小节 4.7中，我们介绍了 VMD 变态模态分解的思想以及算法，那么我们将要把 VMD 算法作为一个数据增强的方法对原数据的序列进行增强 (其中 VMD 的超参数 k 选择为 3，其余为默认参数) 后合并原序列输入模型算法中 (例如 LSTM 与 VMD 的耦合算法如图 11)，由于算力有限，所以在这个模块中，暂时只完成了对于单序列模型的增强并得出结果。

5 Result

5.1 评估指标介绍

对于 $0/20/30/50mm$ 中每个阈值，判断实况和预报是否满足阈值 (即 0 或 1)，进行 TS 检验。

假设 NA_k 为预报正确次数 $(T=1;P=1)$，NB_k 为空报次数 $(T=0;P=1)$，NC_k 为漏报次数 $(T=1;P=0)$，则:

- 准确率 Acc：所有与报个点中预报正确的格点所占百分比;
- 命中率:

$$POD_k = \frac{NA_k}{NA_k + NC_k} \times 100\% \tag{33}$$

- 空报率:

$$FAR_k = \frac{NB_k}{NA_k + NB_k} \times 100\% \tag{34}$$

- 漏报率:

$$PO_k = \frac{NC_k}{NA_k + NC_k} \times 100\% \tag{35}$$

- **TS 评分:**

$$TS_k = \frac{NA_k}{NA_k + NB_k + NC_k} \times 100\%. \tag{36}$$

在本实验中我们主要以 TS_k 为主要评分模型效果的手段，Acc 作为辅助手段，主要判断模型效果是否超过了基准模型。

其中模型的训练集时间跨度为 1970.01.01-2018.01.31，测试集时间跨度为 2018.01.31-2021.01.31。

5.2 数据不平衡问题

所谓的不平衡指的是不同类别的样本量差异非常大，或者少数样本代表了业务的关键数据（少量样本更重要），需要对少量样本的模式有很好的学习。样本类别分布不平衡主要出现在分类相关的建模问题上。样本类别分布不平衡从数据规模上可以分为大数据分布不平衡和小数据分布不平衡两种。

- 大数据分布不均衡。这种情况下整体数据规模大，只是其中的少样本类的占比较少。但是从每个特征的分布来看，小样本也覆盖了大部分或全部的特征。例如拥有 1000 万条记录的数据集中，其中占比 50 万条的少数分类样本便于属于这种情况。
- 小数据分布不均衡。这种情况下整体数据规模小，并且占据少量样本比例的分类数量也少，这会导致特征分布的严重不平衡。例如拥有 1000 条数据样本的数据集中，其中占有 10 条样本的分类，其特征无论如何拟合也无法实现完整特征值的覆盖，此时属于严重的数据样本分布不均衡。

样本分布不均衡将导致样本量少的分类所包含的特征过少，并很难从中提取规律；即使得到分类模型，也容易产生过度依赖于有限的数据样本而导致过拟合的问题，当模型应用到新的数据上时，模型的准确性和鲁棒性将很差。

本研究中的暴雨预报分类就属于其中的小数据分布不均衡问题，降雨量 $T_p > 32mm$ 仅占总数的 $\frac{1}{50}$ 左右，机器学习算法极难解决这种问题。在这种情况下假设模型在非线性系统中什么都没有学到，模型达到的精确度 Acc 也达到了 98%。这种情况我们称其为基准模型，在本研究中所提出或者运用的模型需要均超过基准模型才有改进和运用的可能性。需要说明的是在后续的结果中，模型的效果均已超出了基准模型。

5.3 模型效果

5.3.1 通用模型效果

此实验为每个格点进行统一建模，选取的参数均是一样的。我们做了三个实验，分别使用了单序列特征、五个特征与八个特征 (表格 4.2.1所提到的)，接下来我们将分别比较这些模型的效果。模型效果具体如表格 5.3.1所示。

- 使用单个特征进行通用模型的建模。其模型的输入是七天降雨量的序列。输出为第八天的总降水量。可以看到所有模型中最好的模型是 GBDT 模型。TS 评分为 0.719 分。
- 使用五个特征进行通用模型的建模。模型的输入是前七天的五种特征，输出为第八天的总降水量。可以看到最好的模型是 GBDT 模型。TS 评分为 0.731 分，也是目前最好通用模型的评分。
- 使用八个特征进行通用模型的建模。模型的输入是前七天的八种特征，输出为第八天的总降水量，最好的仍然是 GBDT 模型。TS 评分为 0.729 分。

在基模型之中，我们选择了效果最好的 GBDT 进行模型修改与改进，使用 VMD 算法增强特征与拼接，组成 VMD-GBDT 算法，但算法效果不尽如意，TS 评分仅为 0.718 分。

对于大暴雨预测，单序列模型效果较差，而五个特征与七个特征的模型 TS 评分相差不多，模型效果如表格 5.3.1。最好的模型的特征输入是五个，模型为决策树。**TS 评分为 0.08 分**。

表 3: 降雨预报的 TS 评分

Machine Learning	***Number of features***	
	Five characteristics	**Eight characteristics**
Logistic	0.02	0.02
SVC	0	0
BLS	0	0
DT	**0.08**	**0.07**
RF	0.04	0.06
GBDT	0.04	0.04
XGBoost	0.03	0.03
Etr	0	0
Lgb	0.04	0.04

5.3.2 站点调参效果

对于每个站点的调参的方法为使用上述通用模型中最好的 GBDT 模型进行每个站点不同参数的优化。从而使得每个望各站点能够拥有相互独立的参数。输入为五个特征/七个特征，输出仍然是模型第八天的数值。**晴雨预报使用的是 GBDT 模型目前还在迭代寻优，目前最优化是五个特征的 TS 评分 0.77 分，降雨预报使用五个特征的 DT(决策树模型)。其最佳 TS 评分为 0.141 分**。结果体现在表格 5.3.2中。

表 4: 晴雨预报的 TS 评分

Machine Learning	***Number of features***			***Deep Learning/VMD-ML***	
	One	**Five**	**Eight**		**One**
Logistic	0.671	0.703	0.694	RNN	0.651
SVC	0.672	0.693	0.699	LSTM	**0.668**
BLS	0.667	0.672	0.669	GRU	0.664
DT	0.624	0.672	0.682	LSTNet	0.653
RF	0.702	0.714	0.711		
GBDT	**0.719**	**0.731**	**0.729**	VMD-GBDT	0.718
XGBoost	0.709	0.722	0.714		
Etr	0.710	0.717	0.712		
Lgb	0.714	0.722	0.724		

表 5: 调参后的 TS 评分

	Machine Learning	***The target variable***	
		Rainy forecast	**Rainstorm forecast**
Five	GBDT	0.77($Iterative$)	$None$
	DT	$None$	0.141
Eight	GBDT	0.75($Iterative$)	$None$
	DT	$None$	0.129

6 一些问题及思考

1. 目前仍然存在的问题就是算力不足的情况，因为提供的是小时级别的数据，里面可以包含非常多可用信息，包括使用 Prophet 模型等都可以提取出每小时组成的序列中的特征。但是清洗和训练的成本非常大。

2. 超参数优化的问题，对于每个节点的超参数优化速度十分缓慢，导致每跑一次的实验需要的时间非常久，尤其是晴雨预报使用的是网格优化寻找超参数。后续时候用 $Kmeans$ 或者 DTW 做个分类（聚类）后进行统一调优有待尝试。