Python 在短临气象预报检验中的应用

何佳[1] 惠建忠[1*] 何险峰[2] 王曙东[1] 高金兵[1]

(1 中国气象局公共气象服务中心，北京 100081；2 华风气象传媒集团有限责任公司，北京 100081)

摘要 基于机器学习方法的短临多气象要素预报系统(Weather Elements Nowcasting based on machine learning，简称 WEN)具有高发布频次、高时间分辨率、基于候和时辰的复杂预报模型等特点。应用多维标签数组、机器学习工具、并行计算框架等 Python 库，以快速计算为目标，建立以预报模型覆盖时间范围为统计检验时间边界的检验子系统，客观给出预报性能，为模型调优效果评估、产品业务化运行提供依据。

关键词 并行计算；检验评估；标签化多维数组；Python

中图分类号：P409 **DOI**：10.19517/j.1671-6345.20200399 **文献标识码**：A

引言

气象服务的核心竞争力是不断提升的预报质量。通过检验评估给出客观结果，一方面便于管理人员进行考核，另一方面便于研发人员对比不同方法带来的预报性能改变，促进产品不断改进完善。

国家气象中心建设开发的基于 Web 的国家级天气预报检验分析系统[1]，实现了涵盖国家级省级智能网格预报、全国城镇天气预报等数十项检验业务产品的检验评估。用户可以通过浏览器查看相应产品的检验反馈信息，为各省以及国家级预报业务考核提供了信息支撑。万夫敬[2]等分析了 ECMWF 模式气温预报在青岛地区的预报性能，根据模式误差特点，给出主观订正参考值，提高了气温预报正确率。刘丽敏[3]等通过对中央台精细化温度预报在伊春市的检验和误差分析，掌握其预报性能，在实际工作中有针对性加以订正，为本地释用提供依据。赵平伟[4]等开展了 GPM IMAGE、ERA5 降水数据在云南的适用性评估和对比。蔡晓杰[5]等利用上海沿岸海域 19 个站点风场资料使用平均误差、预报准确率等指标对风速、风向、大风过程进行检验。吴瑞姣[6]等运用两分类 TS 评分、邻域法 FSS 评分方法对安徽省 WRF 模式短时强降水开展检验。郑琳琳[7]等采用均方误差、TS 评分等方法对改进的降水临近外推预报技术方法进行效果检验，得出新方法更具优势的结论。刘凑华[8]等研究了基于目标的降水检验方法并对其改进后进行业务应用。可见检验在业务中发挥着重要作用。目前检验针对的产品在时次、时效上主要是天或小时量级，少有针对分钟级这样高频次发布，高时间分辨率产品的检验。产品在时间维度精细化的提升意味着检验计算时间复杂度的增加。

Python 是近几年越来越受到开发人员关注和使用的编程语言。其简洁、易于理解、功能丰富的特点无论是专业编程人员还是普通用户都能够快速上手。在气象领域，推出了很多易用工具包。如标签化多维数组和数据集 xarray[9]——在吸收用于实现科学数据自我描述的通用数据模型(Common Data Model)基础上，建立类似 pandas 的工具，使分析多维数组变得简单、高效、不易出错。国家气象中心预报技术研发室使用 Python 研发了全流程检验程序库[10]，在数据层提供了数据读写、转换等基础函数；在检验算法层提供有无预报、多分类、概率预报等检验算法；在检验产品制作层提供了数值型、表格型、图片型、误差序列分析等函数，成为国内第一款专门用于气象预报检验的 python 程序库。本文在检验

国家重点研发计划课题“气象预警精准快速发布业务化中试/示范平台技术研发”(2018YFC1507805)资助

作者简介：何佳，女，1983 年生，高级工程师，研究领域为气象信息技术，Email：hejia_1002@163.com

收稿日期：2020 年 9 月 23 日；定稿日期：2021 年 7 月 5 日

* 通信作者，Email：1404713480@qq.com

结果可视化阶段使用了该库提供的功能。

基于机器学习方法的短临多要素气象预报系统(Weather Elements Nowcasting based on machine learning，简称 WEN)，每 10 min 滚动更新发布未来 2 h 多气象要素，覆盖中国区域的短时临近预报，具有高更新频次、高时空分辨率特点。系统在分布式计算环境下，建立 72 候(1 年)×12 时辰(1 天每 2 h 为 1 个时辰)多预报时段复杂预报模型。如何快速获得检验结果是本项工作面临的挑战。

如图 1 所示，WEN 检验子系统主要分为数据准备、检验计算和检验分析 3 个阶段。第 1 阶段进行数据读取、合并，目标是设计通用的数据模型。第 2 阶段进行检验计算，包括连续型数值预报检验、晴雨检验和降水分级检验 3 类检验方法，全部要素共计 10 个检验指标。第 3 阶段分要素、区域对 WEN 模型进行性能评估。

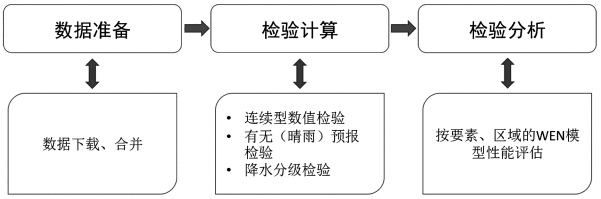

图 1　WEN 检验业务流程

1　数据模型

1.1　检验数据

多维数组(又称 n 维，ND，有时称为张量)是计算科学的一个重要部分，在很多领域都有应用，包括物理、天文学、地球科学、生物信息学、工程、金融和深度学习。在 Python 中，NumPy 为处理原始 ND 数组提供了基本的数据结构和 API。然而，真实的数据集通常不仅仅是原始的数字，它们有标签来编码关于数组值如何映射到空间、时间等位置的信息。Xarray 在原始的与 Numpy 类似的多维数组上引入了维度、坐标和属性形式的标签，使得开发更直观、简洁、更少出错。其核心数据结构 Dataset 被设计为 NetCDF 文件格式的数据模型的内存表示。

WEN 系统采用网络通用数据格式(Network Common Data Form，NetCDF)，使用 TDS 工具(THREDDS Data Server，TDS)管理文件并提供数据访问服务。预报和观测数据(以下统称检验数据)均为每 10 min 发布一次。每个时次发布的站点预报数据包括预报时刻、站点、气象要素 3 个维度，观测数据包括站点和气象要素 2 个维度。

为避免每次在线获取耗时缺点，同时提高处理和共享能力，采用地球科学科研领域元数据标准 CF(Climate and Forecast Metadata Conventions)[11]，将检验数据按天合并后保存到本地。数据模型构建分析如下：①检验用的站点数据属于离散采样几何类型(Discrete Sampling Geometries)；②一天多个时次更新发布，构成了时间序列(Time Series)。基于此，检验数据采用正交多维数组的时间序列数据表示方法，站点预报数据模型如图 2 所示。预测数据模型去掉“预报时效”维度即为观测数据模型。在计算机内存中对应 Xarray 的 Dataset。在预测和观测数据匹配时，通过表示位置的“station”和表示时间的“time”两个“标签”即可实现检验数据配对，体现了 xarray 直观、简洁的优点。

```
dimensions:
  station = 105567;          // 站点（用于空间定位）
  time = UNLIMITED;          // 预报时间
  dtime = 12;                // 预报时效

variables:
  float pressure(station,time,dtime);
    pressure:standard_name = “surface air pressure”;
    pressure:coordinates =  “lat lon alt station_id”
    pressure:units = “hPa”;
  float humidity(station,time,dtime);
    humidity:standard_name = “relative humidity”;
    humidity:coordinates = “lat lon alt station_id”;
    humidity:units = 1
  ……                              // 其它预报要素
  double time(time);
    time:standard_name = “forecast reference time”;
    time:units = “minutes since 1970-01-01 00:00:00”;
    time:calendar = “gregorian”;
  float lon(station);
    lon:standard_name = “longitude”;
    lon:long_name = “station longitude”;
    lon:units = “degrees_east”;
  float lat(station);
    lat:standard_name = “latitude”;
    lat:long_name = “station latitude”;
    lat:units = “degrees_north”;
  float alt(station);
    alt:standard_name = “height”;
    alt:long_name = “vertical distance above the surface”;
    alt:units = “m”;
    alt:positive = “up”;
    alt:axis = “Z”;
  int station_id(station);
    station_id:long_name = “station id”;
    station_id:units = 1;
    station_id:cf_role = “timeseries_id”;
attributes:
    :title = “station forecast by WEN”
    :featureType = “timeSeries”;
```

图 2　WEN 检验数据模型(按天合并的站点预报)

1.2 结果数据

检验方法不同对应指标、要素不同。如晴雨检验和降水分级检验仅针对降水要素，包括 TS 评分、空报率等指标。连续型预报数值型检验包括平均误差、均方根误差等指标。WEN 规定按候输出检验结果。基于此，检验结果按检验方法分类生成，在时间维度包括时效、时辰等。

1.3 站点数据

传统检验方法基于点对点的对比，即在一个时间点上参与检验的样本数为该区域的站点数。WEN 系统包括 105567 个站点。考虑观测数据质量，选取包括国家级气象站（基准站、基本站、一般站）和质控后的部分区域自动站，共计 10656 个站点。在此基础上将全国分成 8 个区域，分区范围见图 3，各分区站点数见表 1。

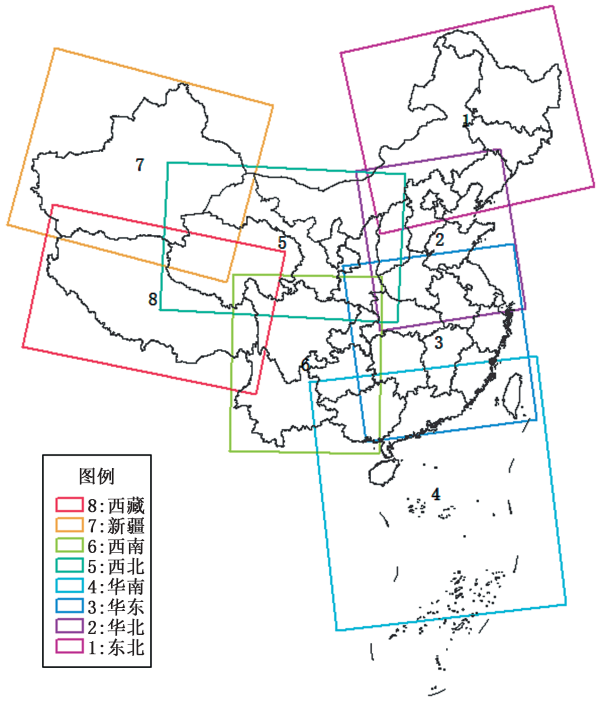

图 3 检验分区示意

表 1 分区站点数

序号	区域	站数	序号	区域	站数
0	中国	10656	5	西北	2047
1	东北	2347	6	西南	2413
2	华北	31556	7	新疆	912
3	华东	3927	8	西藏	400
4	华南	1555			

2 检验计算

WEN 采用连续型预报数值检验、有无预报检验（晴雨预报检验）和多分类检验（降雨分级检验）等 3 类检验算法。后两类属于分类问题，前者是常见的二分类，后者是类别数大于 2 的分类。在分类算法的实现中应用混淆矩阵获得分类样本统计。针对耗时任务给出了并行解决方案。

2.1 检验方法

连续型预报检验采用平均误差、平均绝对误差、均方根误差、偏差 4 种统计方法，检验变量包括气压、温度、相对湿度、风速等近地面要素。降水采用技巧评分（TS）、相当技巧评分（ETS）、空报率、漏报率、预报偏差、准确率 6 种检验方法。上述方法已在业务中广泛应用，在此不做详细介绍。计算公式主要参考由陈昊明等[12]的高分辨率区域数值预报业务检验评估指标及算法。降水分级检验主要参考关于下发中短期天气预报质量检验办法（试行）的通知（气发〔2005〕109 号）中中短期天气预报质量检验办法（试行）解释说明（表 2）和短时气象服务降雨量等级划分[13]（表 3）。

表 2 降水分级检验评定简表

实况	预报						
	小雨	中雨	大雨	暴雨	大暴雨	特大暴雨	无降水
小雨	NA_1	NB_2	NB_3	NB_4	NB_5	NB_6	NC_1
中雨	NC_2	NA_2	NB_3	NB_4	NB_5	NB_6	NC_2
大雨	NC_3	NC_3	NA_3	NB_4	NB_5	NB_6	NC_3
暴雨	NC_4	NC_4	NC_4	NA_4	NB_5	NB_6	NC_4
大暴雨	NC_5	NC_5	NC_5	NC_5	NA_5	NB_6	NC_5
特大暴雨	NC_6	NC_6	NC_6	NC_6	NC_6	NA_6	NC_6
无降水	NB_1	NB_2	NB_3	NB_4	NB_5	NB_6	—

注：NA_k 为预报正确站（次）数，NB_k 为空报站（次）数，NC_k 为漏报站（次）数。k 为 1～6，分别代表各级降水。

表 3 短时气象服务降雨量划分 mm

等级	时段		
	10 min	30 min	1 h
短时小雨	＜0.5	＜1.0	＜2.0
短时中雨	0.5～0.9	1.0～1.9	2.0～3.9
短时大雨	1.0～1.9	2.0～3.9	4.0～7.9
短时暴雨	2.0～4.9	4.0～9.9	8.0～19.9
短时大暴雨	5.0～15.0	10.0～30.0	20.0～50.0
短时特大暴雨	＞15.0	＞30.0	＞50.0

2.2　数据缺失处理

为保证检验结果的正确性及连续性,缺失数据的处理是必不可少的关键步骤。数据缺失有两种表现,一是数据文件缺失,二是数据文件中存在空数据。第 1 种情况表明某时次没有可用于检验的数据,可将检验结果置为一个约定值,例如－999。第 2 种情况则需要将空数据去除,从而保证统计样本的有效性。需要注意的是空数据并非一一对应,可应用公共掩码矩阵对检验数据进行剔除。技术实现的关键代码见图 4。

```
import numpy as np

# ############################
# 对NaN值进行处理：NaN值不参与检验
# ############################

f_mask = np.isnan(f_data)      //预测数据空值布尔型矩阵
o_mask = np.isnan(o_data)      //观测数据空值布尔型矩阵
mask = f_mask + o_mask         //检验数据空值布尔型矩阵
f_data = f_data[~mask]         //参与检验计算的预测数据
o_data = o_data[~mask]         //参与检验计算的观测数据
```

图 4　缺失值处理关键代码

2.3　混淆矩阵

混淆矩阵是机器学习中总结分类模型预测结果的情形分析表,以矩阵形式将数据集中的记录按照真实的类别与分类模型预测的类别进行判断汇总。其中矩阵的行表示真实值,矩阵的列表示预测值。基于 Python 语言的机器学习工具 scikit-learn 的 metrics 模块的混淆矩阵函数 confusion_matrix 实现了混淆矩阵的计算。通过传递真实类别和预测类别样本,返回一个 n 阶矩阵 $\boldsymbol{C}$(n 表示类别数,如二分类 $n=2$),$C_{i,j}$表示真实类别为 i 预测类别为 j 的样本数量。以二分类晴雨检验为例,$C_{0,0}$表示真反例即命中否的样本数,对应表 2 中 ND;$C_{1,0}$表示假反例即漏报的样本数,对应表 2 中 NC;$C_{0,1}$表示假正例即空报的样本数,对应表 2 中 NB;$C_{1,1}$表示真正例即命中的样本数,对应表 2 中 NA。二分类混淆矩阵与晴雨判定对应关系见表 4。通过将气象领域分类检验的判定表与机器学习领域分类模型评估分析表即混淆矩阵建立对应关系,应用混淆矩阵函数实现分类检验计算公式中各参量,再根据公式最终获得检验计算结果。晴雨检验计算实现的关键代码见图 5。

表 4　混淆矩阵晴雨检验对应关系

		预测	
		无雨	有雨
观测	无雨	$C_{0,0}$(命中否)	$C_{0,1}$(空报)
	有雨	$C_{1,0}$(漏报)	$C_{1,1}$(命中)

```
import numpy as np
import sklearn.metrics
import confusion_matrix

# ############################
# 晴雨检验实现代码
# ############################

# ############################
# 二值化处理
# ############################
f_data[f_data < threshold] = 0          // 预测数据无雨（小于阈值），赋值0
f_data[f_data >= threshold] = 1         // 预测数据有雨（大于等于阈值）赋值1
o_data[o_data < threshold] = 0          // 观测数据无雨（小于阈值），赋值0
o_data[o_data >= threshold] = 1         // 观测数据有雨（大于等于阈值）赋值1

# ############################
# 混淆矩阵
# ############################
cm = confusion_matrix(o_data, f_data)   // 通过传递真值（观测数据）和预测数据，计算晴雨检验分析表
nd, nb, nc, na = cm.ravel()             // 将2阶矩阵扁平化为一维数据，得到命中否、空报、漏报、命中4个参量

# ############################
# 检验计算
# ############################
ts = na / (na + nb + nc)                // 计算TS评分
far = nb / (na + nb)                    // 计算空报率
……
```

图 5　晴雨检验计算实现代码

2.4　并行计算

2.4.1　并行计算概念、方式、环境

从计算复杂性的角度来看,一个算法的复杂性可表示为空间复杂性和时间复杂性两个方面。并行计算(Parallel Computing)的基本思想是用多个处理器来协同求解同一个问题,即将被求解的问题分解成若干个部分,各部分均由一个独立的处理器或处理机来并行计算。并行计算的目标就是尽量减少时间复杂性,通常这是通过增加空间复杂性来实现的。并行计算可分为时间上的并行和空间上并行。时间上的并行就是指流水线技术。空间上的并行则是指多个处理器并发的执行计算。并行环境多为 MPP(Massively Parallel Processing,大规模并行处理)并行机。不同于普通 PC 机,通常并行机有特定的系统结构,支持指定的并行环境(常见的有 MPI(Message Passing Interface)),硬件价格昂贵,同时并行计算的实现也是一项复杂而艰巨的工作。

曹晓钟等[14]针对大量神经网络训练、预报的需求,在 SP2 并行机上实现了基于主从计算模式,采用粗粒度任务划分的并行方案。赵军等[15]在多核高性能计算机上使用 MPI/OpenMP 混合并行计算编程,以增量方式和较小的并行化代价实现了对 AREM 模式并行化改进,取得了较好的并行计算效

果。二者均是在特定并行机环境下实现的并行化。周钦强等[16]采用 Java 语言基于 Eclispes 平台开发实现了基于 TCP 多连接通信实时并发数据处理系统，满足对周期性大批量浪涌数据进行实时快速处理的要求。系统运行于单处理机上。其主要利用线程池化技术和并发管理机制，实现过程不仅需要对不同线程进行详细的程序流程分析，还要处理复杂的线程安全问题，开发成本相对较大。韩帅[17]等利用多节点分块并行与模式并行结合的计算方案，解决了系统文件大小受限、计算资源和计算时效等问题。是一种时间和空间二维并行的方案。

2.4.2 DASK 并行计算框架

DASK 是一款开源免费、与现有科学计算项目 Numpy、Pandas、Scikit-Learn 高度协同的基于 python 语言的并行计算框架[18]。它以一种更方便简洁的方式处理大数据量，能够在普通单机或集群中进行分布式并行计算。DASK 由两个部分组成，动态任务调度和“大数据”集合。动态任务调度包括单机调度器和分布式调度器，分布式调度器可以在单机上运行，提供了比单机调度器更丰富的功能，如诊断面板。在单机中通过 scheduler 参数设置线程或者进程来并行处理，也称为伪分布式。“大数据”集合提供的 3 种基本数据结构分别是 arrays、dataframes、bags。Arrays 是对 Numpy 中的 ndarray 的部分接口进行了改进，从而方便处理大数据量。Dataframes 是基于 Pandas DataFrame 改进的一个可以并行处理大数据量的数据结构。以上两者通过对大数据量进行分块处理，提供了大于内存数据的解决方案。Bags 主要用于半结构化的数据集，比如日志或者博客等。DASK 提出的延迟计算是在一个普通的 Python 函数上面通过 dask.delayed 函数进行封装，得到一个 delayed 对象。该对象只是构建一个任务图(Task Graph)，并没有进行实际的计算，只有调用 cumpute 的时候才开始进行计算。延迟计算非常适合当面对的问题是可并行的但是不适合高度抽象如 Dask Array 或 Dask Dataframe 的场景。下面给出一段在单机上使用分布式调度器、延迟计算、多进程实现并进行化的代码示例，见图 6。

2.4.3 WEN 检验并行化实现

并行计算首先是一种解决计算问题的思想。分析 WEN 检验，以数据准备环节为例，处理过程分为两个步骤：①读取数据，这个操作对每个文件是相同的；②所有文件读取结束后进行合并操作。读取一个文件是一个独立动作，互相之间无依耐关系，可同时执行。利用 dask.delayed 将读取文件进行并行化改造。以处理 156 个观测文件(1 d+2 h)为例，程序分别在两台具备不同核数的普通 PC 机上运行。考虑网络带宽时间维度不均匀性，为使测试结果更客观，选取 2020 年 9 月 1—5 日，每个日期在不同时间分 3 次运行，分别记录每次的运行时间，再计算平均用时。并行前后程序执行时间见表 5。以平均数衡量，56 核处理机效率提升 11 倍，96 核处理机效率提升 9 倍。进行并行化改造增加的代码不超过 10 行。

```
import dask
import dask.delayed
import dask.distributed
import Client

client = Client()
client.config.set(scheduler='processes')      // 使用“进程”并行化

x = dask.delayed(inc)(1)          // 使用“延迟计算”封装函数inc（参数1）
y = dask.delayed(inc)(2)          // 使用“延迟计算”封装函数inc（参数2）
z = dask.delayed(add)(x, y)       // 使用“延迟计算”封装函数add
z.compute()                       // 根据delayed对象构建的任务图开始计算
z.vizualize()                     // 任务图可视化

client.close()
```

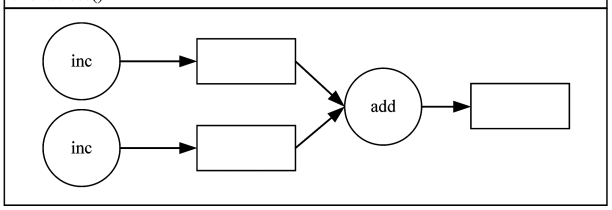

图 6 一种 DASK 并行化示例代码

表 5 观测数据采集合并运行时间测试结果 s

日期(月-日)	56 核						96 核					
	串行			并行			串行			并行		
	1	2	3	1	2	3	1	2	3	1	2	3
09-01	461	474	457	49	36	42	465	444	449	48	52	42
09-02	467	446	444	47	32	36	425	436	436	60	74	40
09-03	484	472	461	54	34	38	450	437	427	57	39	38
09-04	472	456	481	48	34	42	441	441	493	76	40	40
09-05	477	458	491	53	33	52	438	452	500	41	44	39
平均	467			42			449			48		

运用相同思路，对检验计算环节代码进行并行化改造。以计算 1 天数据每个时次每个时效数值型预报检验为例(仅 9 月 1 日数据缺失 36 个时次，其它日期完整)，时次和时效两个维度同时并行，即并

行任务有 144×12＝1728 个,测试结果见表 6。并行后,56 核环境效率提升 20 倍,96 核环境提升 51 倍。改造后的关键代码见图 7。

表 6　连续型数值型预报检验(每时次每时效)运行时间　　s

日期(月-日)	56 核		96 核	
	串行	并行	串行	并行
09-01	2042	99	5579	131
09-02	2861	129	7445	168
09-03	2481	129	7496	171
09-04	2437	129	7502	165
09-05	2473	128	7457	159
平均	2459	123	7096	139

从测试结果来看,效率提升并非与 CPU 核数成线性递增。首先,该测试结果采用单机多进程并行方案,且并行任务数大于核数,即并行任务与 CPU 并非一次分配到位,而是进行了多次分配。其次,当进程间传输数据较大时,会产生性能损失。从表 5 和表 6 并行方案测试结果来看,96 核环境耗时均高于 56 核环境。但总体上计算效率并行方案都远高于串行方案。

对于串行方案,表 5 测试结果显示,56 核环境略高于 96 核,但相差不大。主要原因是表 5 对应的业务属于 IO 密集型,主要受实时网络流量影响,受 CPU 型号差异带来的性能影响较小。表 6 测试结果,96 核环境耗时是 56 核环境的 3 倍左右。表 6

```
import numpy as np
import xarray as xr
import dask.delayed
From dask.distributed import Client

# ###计算每个时次每个时效
def checkByTimeAndDtime(f_data, o_data):
    dif = f_data - o_data                                    // 计算误差
    me = dif.mean(dim='station')                             // 计算平均误差
    mae = xr.ufuncs.fabs(dif).mean(dim='station')            // 计算平均绝对误差
    ……                                                       // 其它指标计算
    …….                                                      // 计算结果存储
    return result                                            // 返回计算结果

# ###并行计算全部时次、时效
def checkParallel(f_data, o_data):
    delayedObjs = []                                         // 初始化一个空列表对象，用于存储delayed对象
    time_list = f_data.time.values.tolist()                  // 时次列表
    dtime_list = f_data.datime.values.tolist()               // 时效列表
    for time in time_list:
        f = f_data.sel(time=time)                            // 一个时次预报数据
        for dtime in time_list:
            f = f.sel(dtime=dtime)                           // 一个时效预报数据
            o_time = time + timedelta(minutes=10*dtime       // 计算观测数据时间
            o = o_data.sel(time=o_time)                      // 与预报匹配的观测数据
            delayedObjs.add(dask.delayed(checkByTimeAndDtime)(f, o))   // 构建任务图
    result = dask.compute(*delayedObjs)                      // 并行计算

# ###主函数(程序执行入口)
main():
    if len(sys.argv) != 2 :
        print("Usage: Script YYYYMMDD")
        exit(-1)

    client = Client()
    dask.config.set(scheduler='processes')
    checkParallel()                                          // 调用并行计算函数
    client.close()

if __name__ == "__main__":
    main()
```

图 7　WEN 数值型检验方法并行化实现关键代码

对应的业务属于计算密集型，主要受 CPU 性能影响。56 核环境 CPU 型号高于 96 核环境。两者的型号分别是 56 Intel(R) Xeon(R) Gold 5120 CPU @ 2.20 GHz 和 96 Intel(R) Xeon(R) CPU E7-8850 v2 @ 2.30 GHz。

3 检验分析

图 8 是 2021 年 3 月 1～6 候模型的全国区域气温要素检验结果(篇幅有限仅以气温要素均方根误差指标为例进行说明)。图中横轴表示一个候的预报模型，每个刻度代表一个模型(即每天 12 个时辰之一，世界时)。从图中可以看出，①3 月 WEN 模型气温均方根误差最大 4.77 ℃(第 1 候第 11 时辰)，最小 1.13 ℃(第 4 候第 5 时辰)；②每候总的趋势表现为：后 5 个时辰较前 7 个时辰大，即夜间(北京时晚 10:00 至次日早 08:00)误差较白天大。对比了 21 年 3—5 月的检验结果，发现不同月份表现差的模型均集中在夜间。

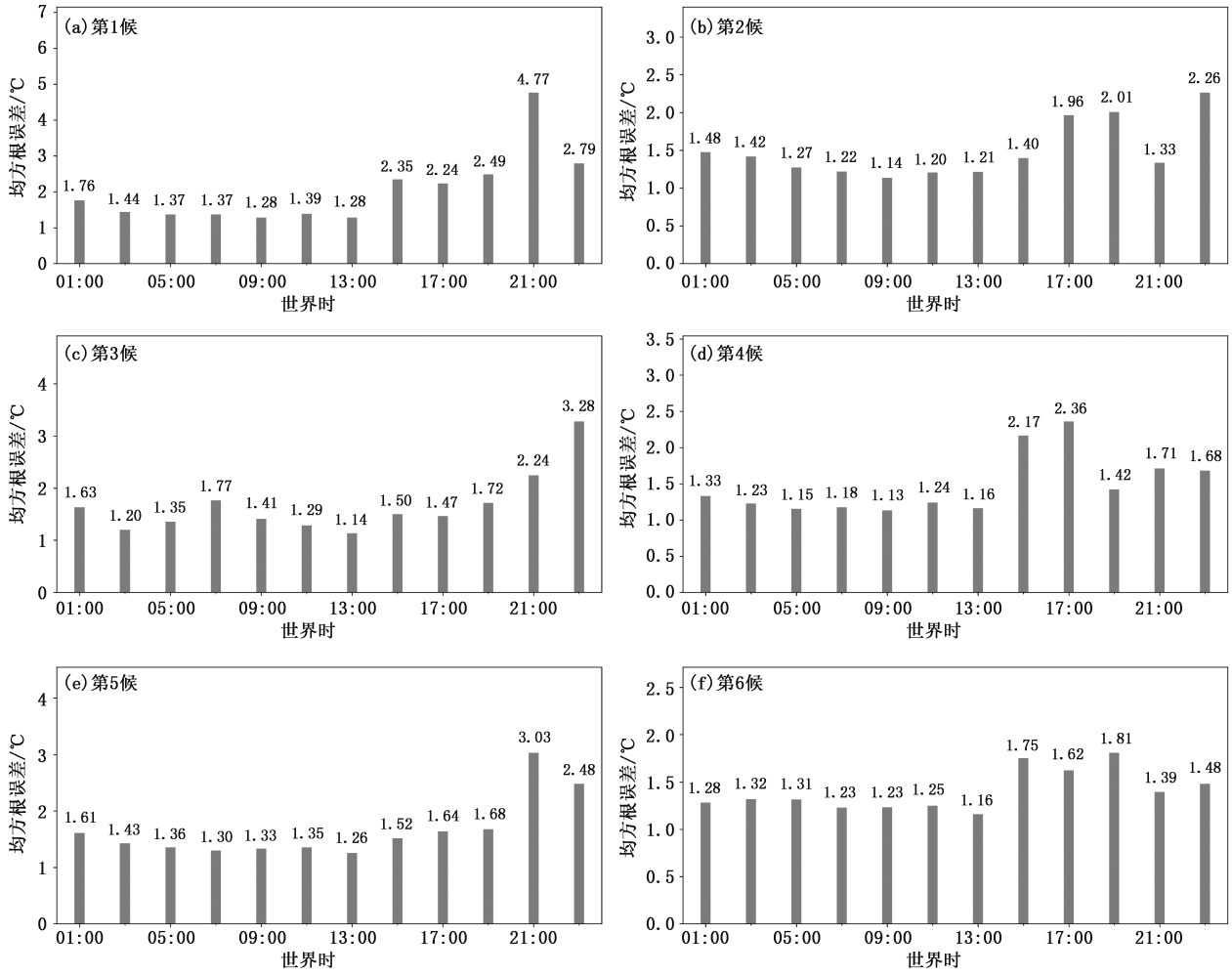

图 8 2021 年 3 月第 1～6 候 WEN 模型全国气温均方根误差

4 结论

本文在分析业务需求的基础上，设定快速计算为目标，重点围绕气象领域检验分类问题和应用并行计算改善时效性两个方面，设计并实现了针对高频次、高时间分辨率短临气象预报产品的检验系统。今后将围绕检验报告自动生成、容器化部署等开展工作，推动公共气象服务中心产品检验业务常态运行，为实现气象服务高质量发展贡献力量。

参考文献

[1] 韦青，李伟，彭颂，等. 国家级天气预报检验分析系统建设与应

用[J]. 应用气象学报,2019,30(2):245-256.
[2] 万夫敬,赵传湖,马艳,等. ECMWF 模式气温预报在青岛地区的检验与评估[J]. 气象科技,2018,46(1):112-120.
[3] 刘丽敏,石磊,赵波,等. 中央台精细化温度预报在伊春市的检验和误差分析[J]. 黑龙江气象,2018,35(2):24-25.
[4] 赵平伟,李斌,王佳妮,等. GPM IMAGE 和 ERA5 降水数据精度在云南复杂地形区域的评估检验[J]. 气象科技,2021,49(1):114-123.
[5] 蔡晓杰,戴建华,朱智慧,等. 上海沿岸海域风场质量控制与预报检验[J]. 气象科技,2019,47(2):214-221.
[6] 吴瑞姣,邱学兴,周昆,等. 安徽省 WRF 模式短时强降水的预报检验[J]. 气象科技,2020,48(2):254-262.
[7] 郑琳琳,邱学兴. 一种改进的降水临近外推预报技术方法研究及效果检验[J]. 气象科技,2020,48(1):97-106.
[8] 刘凑华,牛若芸. 基于目标的降水检验方法及应用[J]. 气象,2013,39(6):681-690.
[9] Xarray developers. Xarray: N-D labeled arrays and datasets in Python[EB/OL]. (2014-2020)[2020-09-21]. http://xarray.pydata.org/en/stable/
[10] 刘凑华,代刊,韦青,等. meteva—让预报检验不再重复造轮子:全流程检验程序库[EB/OL]. (2020-06-23)[2020-09-21]. https://www.showdoc.cc/meteva.
[11] Brian E, Jonathan G, Bob D, et al. NetCDF Climate and Forecast (CF) Metadata Conventions[EB/OL]. (2004-02-20)[2020-12-15]. http://cfconventions.org/Data/cf-conventions/cf-conventions-1.8/cf-conventions.pdf.
[12] 陈昊明,周佰铨,李成伟,等. 高分辨率区域数值预报业务检验评估指标及算法[EB/OL]. [2020-09-20]. http://10.1.64.154/areaHighResolution
[13] 王曙东,张国平,惠建忠,等. T/CMSA 0013-2019 短时气象服务降雨量等级[S]. 北京:中国气象服务协会,2019.
[14] 曹晓钟,刘还珠. 神经网络在气象应用中的并行实现[J]. 计算机工程与应用,2003,(36):207-209.
[15] 赵军,吴建平,宋君强,等. 多核环境下 AREM 模式混合并行计算研究[J]. 计算机工程与应用,2011,47(21):61-63.
[16] 周钦强,谭鉴容,伍光胜,等. 基于 TCP 多连接通信实时并发数据处理技术研究[J]. 计算机工程与应用,2007,43(18):246-248.
[17] 韩帅,师春香,姜志伟. CMA 高分辨率路面数据同化系统(HRCLDAS-V1.0)研发及进展[J]. 气象科技进展,2018,8(1):102-108.
[18] Anaconda, Inc. and contributors. Dask[EB/OL]. [2020-09-21]. https://docs.dask.org/en/latest/.

Application of Python in Test of Nowcasting

HE Jia[1]　HUI Jianzhong[1]　HE Xianfeng[2]　WANG Shudong[1]　GAO Jinbing[1]
(1 Public Meteorological Service Center of CMA, Beijing 100081; 2 Huafeng Meteorological Media Group, Beijing 100081)

Abstract: Weather Elements Nowcasting based on machine learning (WEN) has the characteristics of high release frequency, high time resolution, and complex forecast model based on climate and time. Using Python libraries such as multidimensional tag array, machine learning tools, parallel computing framework aiming at "fast computing", a testing subsystem is established. It takes the time range covered by the "prediction model" as the statistical test time boundary. It objectively gives the prediction performance, which provides a basis for evaluating model optimization effect and the operation of products.

Keywords: parallel computing; testing and evaluation; N-D labelled arrays; Python