**Computer Science and Application** 计算机科学与应用, 2021, 11(4), 867-875
Published Online April 2021 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2021.114089

Hans汉斯

# 基于大数据的疫情监控与预测系统

**宋　吉，王得江，兰　红**

江西理工大学信息工程学院，江西　赣州
Email: 719792323@qq.com

收稿日期：2021年3月15日；录用日期：2021年4月10日；发布日期：2021年4月21日

## 摘　要

**针对疫情的实时监控和预测困难等问题，本文采用爬虫技术爬取大量疫情数据，采用大数据分析处理技术对疫情数据源进行清洗与聚类统计，采用SEIR模型与遗传算法对疫情变化趋势进行预测，采用Vue.js与Echarts进行可视化展现，以实现疫情监控与预测。**

## 关键词

**疫情预测，数据可视化，爬虫，大数据，SEIR模型，差分进化算法**

# Epidemic Monitoring and Forecasting System Based on Big Data

**Ji Song, Dejiang Wang, Hong Lan**

School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou Jiangxi
Email: 719792323@qq.com

Received: Mar. 15th, 2021; accepted: Apr. 10th, 2021; published: Apr. 21st, 2021

## Abstract

**In view of the difficulties in real-time monitoring and prediction of the epidemic situation, this paper uses crawler technology to crawl a large number of epidemic data, then uses big data analysis and processing technology to clean and cluster the epidemic data sources, and uses SEIR model and genetic algorithm to predict the trend of the epidemic situation, Vue.js Visual display with Echarts to realize epidemic monitoring and prediction.**

## Keywords

**Epidemic Prediction, Data Visualization, Crawler, Big Data, SEIR Model,**

文章引用: 宋吉, 王得江, 兰红. 基于大数据的疫情监控与预测系统[J]. 计算机科学与应用, 2021, 11(4): 867-875.
DOI: 10.12677/csa.2021.114089

**Differential Evolution Algorithm**



## 1. 引言

当前，全球新冠肺炎疫情状况高烧不退，并且可能在未来较长的时间内持续存在，如果能实时的监控疫情分布并且预测其变化趋势，那么对疫情防控，及其策略的调整，有着重要的参考价值。因此一款能够实时、直观、全方位的监控疫情，并且能够动态预测疫情变化的系统非常有必要。

就目前来看，市场上有一些针对疫情可视化展现与预测的系统或平台，但是存在系统功能单一，或是在疫情可视化展现和预测方法上的问题或缺陷。

从疫情数据可视化方面来看，当前，权威的疫情消息，都由各地卫建委每天以文字的形式发布，信息往往缺乏全面性和直观性，容易引起视觉疲劳，无法直观的掌握疫情严重程度，以及与周边地区的疫情对比情况，这降低了用户的敏感度，不利于疫情防控的全面进行。

从疫情变化趋势预测方面来看，因为在疫情发展初期，疫情传播动力学研究建模过程中，缺乏足够的原始数据，所以较难对疫情发展态势进行准确预测。再加上证据表明不易被有效隔离的潜伏期患者具有较强的病毒传播能力，而研究人员之前建立的疫情传播动力学模型忽略了潜伏期患者导致的传播风险。

综上，尽管目前已经提出了众多的疫情发展趋势预测方法，但疫情的变化往往受国家政策、人民防控意识、地区医疗水平等难以量化因素影响，且不同地区的情况各不相同，这就导致传统的预测方法难以令人满意的效果，当前阶段仍有必要对 COVID-19 疫情传播规律进行研究，这对疫情的态势分析和防控指导等有着现实意义。

为更好的服务于疫情防控，本文围绕“疫情可视化监控”与“疫情发展趋势预测”展开。

对于疫情可视化监控，系统提供了“疫情分布大屏”，它能直观的查看世界疫情点状分布图，用户可以指定日期并查看详细到市区的累计确诊、现有确诊、死亡人数、治愈人数的疫情分布地图。不仅如此，为了让用户直观地比较不同地区的疫情数据，系统还提供了以国家或省份/市区划分的疫情统计表格，以便对不同地区的疫情数据进行横向比较。

对于疫情发展趋势预测，系统提供了“开放性疫情预测模型”，利用大数据和 DE 差分算法对全球的大规模疫情数据进行参数寻优，并把最优参数导入 SEIR (疫情预测模型)模型进行预测拟合。从而得到了各地区的感染人数预测、治愈人数预测、床位警示预测。不仅如此，为了保证模型的普适性，系统开放了模型中部分参数，用户可根据本地情况对参数调整并查看调整后的预测结果。

## 2. 总体设计

结合功能需求，系统设计了四大功能模块，分别是数据获取、数据分析、后台服务、前台页面。通过四个功能模块组成系统，以实现疫情可视化监控和预测功能，系统总体功能设计如图 1 所示。

## 3. 数据获取模块设计

### 3.1. 数据爬取

为了爬取大量的疫情数据，必须充分考虑到爬虫在爬取数据中可能遇到的各种问题，例如反爬虫、

网络连接超时、爬取耗时过长等。因此为系统设计了基于 Python 语言的 Scrapy [1]爬虫框架的爬虫代码。Scrapy 框架是基于 Python 语言的爬虫框架，Scrapy 爬虫框架与传统爬虫框架(如 Requests、Urllib)相比，有着诸如爬取效率高、异步请求、非阻塞等特点，并且可以基于配置省略大量需要手写的功能代码，例如 IP 代理、随机延时请求、失败重试等功能。

结合具体所需爬取的疫情数据，得到数据爬取流程，如图 2 所示。

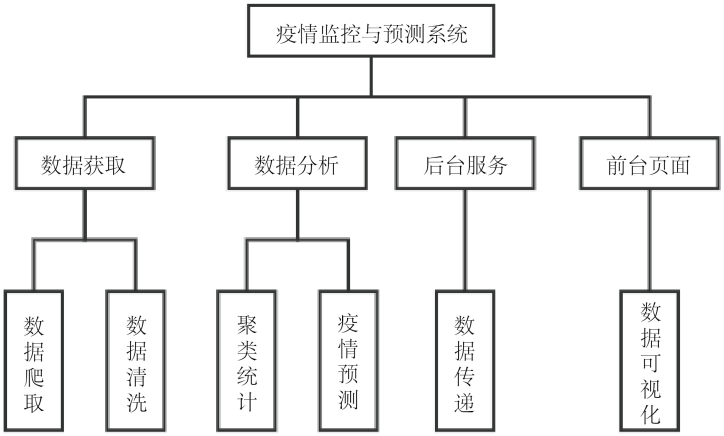


**Figure 1.** Overall system function diagram
**图 1.** 系统总体功能图

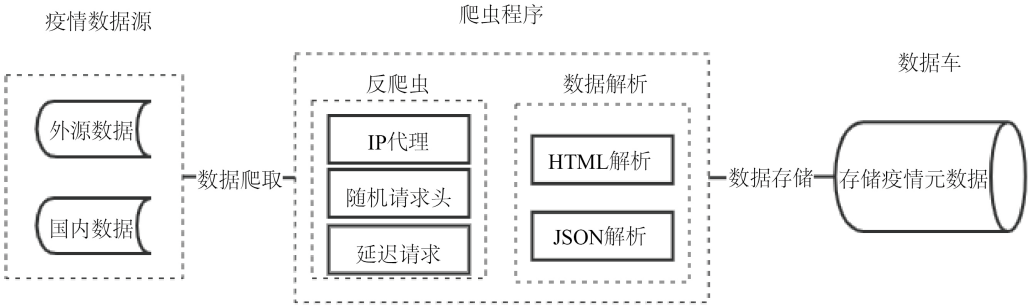


**Figure 2.** Data crawling process
**图 2.** 数据爬取流程图

## 3.2. 数据清洗

考虑到系统爬取的疫情数据源分为国内中文数据和国外英文数据，为了方便用户查看疫情数据，必须将外源英文数据进行中文翻译处理。因此为系统设计了基于 Python 语言的 translate 框架的文本翻译代码，translate 是一个用 python 编写的简单而强大的翻译工具，支持多个翻译提供商，能够轻松的将输入的文本翻译成所需语言的文本。

结合系统所需数据的格式要求，得到数据清洗算法步骤如下。

**Step1**：加载数据库中的疫情源数据；

**Step2**：创建用于匹配英文的正则表达式对 regex；

**Step3**：创建 translate 实例对象 trans；

**Step4**：迭代疫情源数据，设置迭代对象为 Itemi；

**Step5**：使用 regex 对 Itemi 进行判断，如果是英文数据则转至 Step6，否则转至 Step7；

**Step6**：使用 trans 对象对 Itemi 进行中文翻译处理；

**Step7**：保存 Itemi 对象到数据库中。

## 4. 数据分析模块设计

### 4.1. 疫情数据聚类统计

因为爬取的疫情源元数据在地区层面上较为分散，例如含有区/市、省/州，国家等层级，为了得到各层级的准确统计数据，则必须对整体数据按照不同层级进行聚类统计处理。并且考虑到疫情数据量巨大，且系统对数据处理效率上要求较高。因此系统选用了 Spark [2]作为大数据处理分析框架，Apache Spark 是专为大规模数据处理而设计的快速通用的计算引擎。Spark，拥有 Hadoop MapReduce 所具有的优点；但不同于 MapReduce [3]的是——Job 中间输出结果可以保存在内存中，从而不再需要读写 HDFS，在内存计算下，Spark 比 Hadoop [4]快许多。

### 4.2. 疫情预测

#### 4.2.1. 疫情预测模型

SEIR [5]是人工智能领域解决传染病预测等问题的重要数学模型。可通过研究传染病的传播速度，空间范围，传播途径，动态机制等问题，指导传染病的有效防治。基于传染病模型进行推广生成传染病动态模型，是开展理论定量研究的重要方法。本文引用的 SEIR 传染病模型，在人群划分和转换方面的关系如图 3 所示。

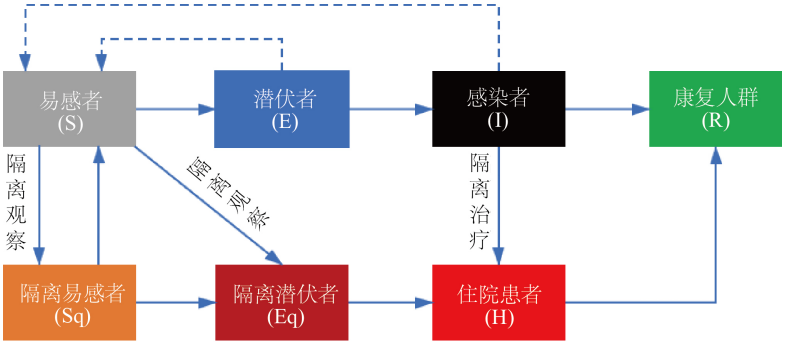


**Figure 3.** The relationship of the SEIR model
**图 3.** SEIR 模型关系图

该模型对易感者人数的控制如公式(1)所示：

$$\mathrm{d}S/\mathrm{d}t = -\left[\rho c\beta + \rho cq(1-\beta)\right]S(I+\theta E) + \lambda S_q \tag{1}$$

公式(1)中 $q$ 是隔离比，$\beta$ 是感染的概率，$c$ 是接触率，$\rho$ 是有效接触系数，通常是一个固定值为 1，$\rho c$ 是有效接触率。易感者的转化率分别为 $S$、孤立的易感者 $S_q$、孤立的潜在的 $E_q$ 和潜在的 $E$，为公式中(1)的 $cq(1-\beta)$、$\rho c$ 的 $q\beta(1-q)$ 和 $\rho c\beta(1-q)$。考虑到非孤立感染者 I 和潜伏者 $E$ 对易感者的影响，敏感者已经从隔离中释放出来，$S_q$ 重新转化为 $S$，$\theta$ 是潜在感染者与感染者传播能力的比率。假设潜伏期的患者与那些已经表现出症状的患者具有相同的传染性，即 $\theta = 1$。$\lambda$ 是隔离释放率，其值为 1/14，隔离持续时间为 14 天。

模型整体动力学方程构造如公式(2)所示：

$$\begin{cases} \mathrm{d}S/\mathrm{d}t = -\left[\rho c\beta + \rho cq(1-\beta)\right]S(I+\theta E) + \lambda S_q \\ \mathrm{d}E/\mathrm{d}t = \rho c\beta(1-q)S(I+\theta E) - \sigma E \\ \mathrm{d}I/\mathrm{d}t = \sigma E - (\delta_I + \alpha + \gamma_I)I \\ \mathrm{d}S_q/\mathrm{d}t = \rho cq(1-\beta)S(I+\theta E) - \lambda S_q \\ \mathrm{d}E_q/\mathrm{d}t = \rho c\beta qS(I+\theta E) - \delta_q E_q \\ \mathrm{d}H/\mathrm{d}t = \delta_I I + \delta_q E_q - (\alpha + \gamma_H)H \\ \mathrm{d}R/\mathrm{d}t = \gamma_I I + \gamma_H H \end{cases} \tag{2}$$

在公式(2)中，$\sigma$ 为潜伏者向感染者的转化速率，取 1/7(潜伏期为 7 天)，$\alpha$ 为病死率，$\delta_I$ 是感染者的隔离速率，$\gamma_I$ 是感染者的恢复率。$\delta_q$ 是隔离潜伏者向隔离感染者的转化速率，$\gamma_H$ 是隔离感染者的恢复速率。

而在真正使用中发现，由于 2 月 12 日起，湖北省等地区改变新增确诊病例的统计方式，将临床诊断病例计入新增病例中，使得官方公布的新增病例大幅上升。统计方式的改变导致官方公布的数据高于原文献中的理论模型预测结果。考虑前后的统计方法不同，因此对模型参数进行适当修正。通过考虑部分非二次确诊病例的计入导致平均感染系数较低，以及总病例数增大导致接触系数增大，需要引入一种动态调整参数的方式，使用动态寻优后的修正 SEIR 模型计算感染人数理论结果。

#### 4.2.2. 参数寻优算法

差分进化算法(Differential Evolution, DE)又称微分进化算法，是一种求解最优化问题的进化算法。因为进化算法对于最优化问题的要求极少，所以被视为一种后设启发式算法。虽然后设启发式算法适用于多种最优化问题，但是并不保证可以找到全局最优解。

差分进化算法被使用在多维度实数编码的最优化问题。因为此算法不使用问题的梯度信息，故可解不可微分的最优化问题。也因此，差分进化算法可用于不连续的，噪声的，随着时间改变的最优化问题。

差分进化算法类似遗传算法，包含变异，交叉操作，淘汰机制。本质上说，它是一种基于实数编码的具有保优思想的贪婪遗传算法[6]。而差分进化算法与遗传算法不同之处，在于变异的部分是随选两个解成员变量的差异，经过伸缩后加入当前解成员的变量上，因此差分进化算法无须使用几率分布产生下一代解成员[7]。

算法的原理采用对个体进行方向扰动，以达到对个体的函数值进行下降的目的，同其他进化算法一样，差分进化算法不利用函数的梯度信息，因此对函数的可导性甚至连续性没有要求，适用性很强。同时，算法与粒子群优化有相通之处，但因为差分进化算法在一定程度上考虑了多变量间的相关性，因此相较于粒子群优化在变量耦合问题上有很大的优势。由于差分进化算法在连续域优化问题的优势已获得广泛应用，并引发进化算法研究领域的热潮。

DE 差分进化算法经典的策略算子如公式(3)所示：

$$v_i = x_i^1 + F \times \left(x_i^2 - x_i^3\right) \tag{3}$$

公式(3)中 $x^1$、$x^2$、$x^3$ 是从当前种群中选择的三个个体，$i$ 表示第 $i$ 个基因位，而 $F$ 是一个量化因子，可用于控制新生个体与父代个体 $x^1$ 之间的差别。由于在 DE 算子中参考的父代个体较多，新产生的后代子个体与原有的父代个体之间的差别也较大，所以在全局搜索能力上有不俗的表现，经常被用于处理复杂优化问题，以避免算法陷入局部最优。

针对模型参数的动态优化问题，本文利用 DE 差分进化算法根据真实数据来对模型的参数进行动态优化，使模型得出的预测效果更符合实际发展情况。

具体算法步骤：

**Step1**：初始化种群控制参数：种群规模 pop_size,最大进化次数 max_Iterate;

$$\left\{X_i(0) \mid x_{i,j}^L \le x_{i,j}(0) \le x_{i,j}^U; i=1,2,\cdots,NP;\ j=1,2,\cdots,D\right\} \tag{4}$$

公式(4)中，$X_i$ 是第个 $i$ 个体，$j$ 表示第 $j$ 维。

$$x_{i,j}(0) = x_{i,j}^L + rand(0,1)\left(x_{i,j}^U - x_{i,j}^L\right) \tag{5}$$

公式(5)中 $X_{i,j}^L$ 和 $X_{i,j}^U$ 分别为第 $i$ 维的下界和上界，$rand(0,1)$表示在区间[0, 1]上的随机数。

**Step2**：随机初始化 cn 个类簇中心，并生成初始种群 Chrom，对每个类簇中心使用公式(3)计算各样本的隶属度，以及用公式(3)计算每个个体的适应度值，其中 $i=1,2,\cdots,$pop_size ;

**Step3**：设置进化迭代参数 Iterate = 0；

**Step4**：对种群 Chrom 进行选择、交叉、变异等遗传操作，对新产生的个体使用公式(4)、(5)计算 cn 个类簇中心，各样本的隶属度，并用公式(3)计算每一个体的适应度值 $f_i$。若 $f_i > F_i$，则新个体替换旧个体；否则，以概率 $P=\exp\left(\left(F_i - f_i\right)T\right)$。接受新个体，抛弃旧个体；

**Step5**：令 Iterate = Iterate+1。若 Iterate < max_Iterate，则转至 Step4；否则执行 Step6；

**Step6**：令 $T_i = J * T_{i-1}$。若 $T_i > T_{end}$，则转至 Step3；否则，执行 Step7；

**Step7**：输出最优解。

由此，得到 cn 个不同簇间疏散，同簇内密集的类簇。

## 5. 后台服务模块设计

为了方便的读取数据分析模块统计与预测的数据结果，同时给用户在前台页面良好的响应体验，后台服务应该满足与数据分析模块有良好的兼容与对接能力，同时能满足一定的高并发条件下的性能要求。

因此系统后台服务选用了 Java 语言编写的 SpringBoot [8]框架，SpringBoot 框架拥有项目快速搭建、与主流框架无配置集成、云计算天然集成等特点，这使得后台服务能够轻松地对接本系统的大数据框架，进而在快速完成开发的前提下，还能够轻松的读取大数据模块处理的结果。

本文采用的 SpringBoot 架构设计如图 4 所示。

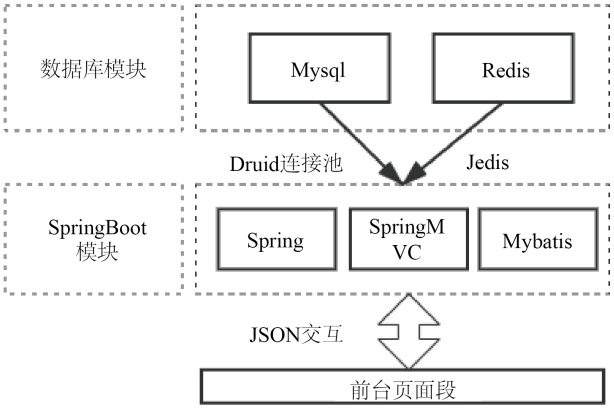


**Figure 4.** SpringBoot background framework design
**图 4.** SpringBoot 后台框架设计图

## 6. 前台服务模块设计

### 环绕式页面布局

环绕式布局它是在界面设计中最灵活的布局之一。环绕布局将内容分成多个有意义的部分，不仅充分的利用了屏幕空间，还能让用户一目了然的了解页面所包含的内容。

本系统前台页面采取的布局方式为环绕式布局，设计图如图 5 所示，其中具体布局块解释如下：

① 放置标题。

② 放置数据地图。

③ 放置疫情确诊人数。

④ 中放置全球国家(地区)目录，点击目录在区域②的地图上显示该国疫情情况，由于医院数据量大，区域有限所以设置一个滚动条可以下拉查看更多数据。

⑤ 放置当前时间信息。

⑥ 放置根据国家(地区)搜索详情疫情的搜索栏。

⑦ 放置一个日历，主要用于根据日期在区域展示全球各地区当天的疫情情况。

⑧ 放置一个折线直方图，展示某地区当月每日疫情变化情况。

⑨ ⑩ ⑪ ⑫ 分别放置四个按钮，使区域②分别展示这四个疫情指标的分布情况。

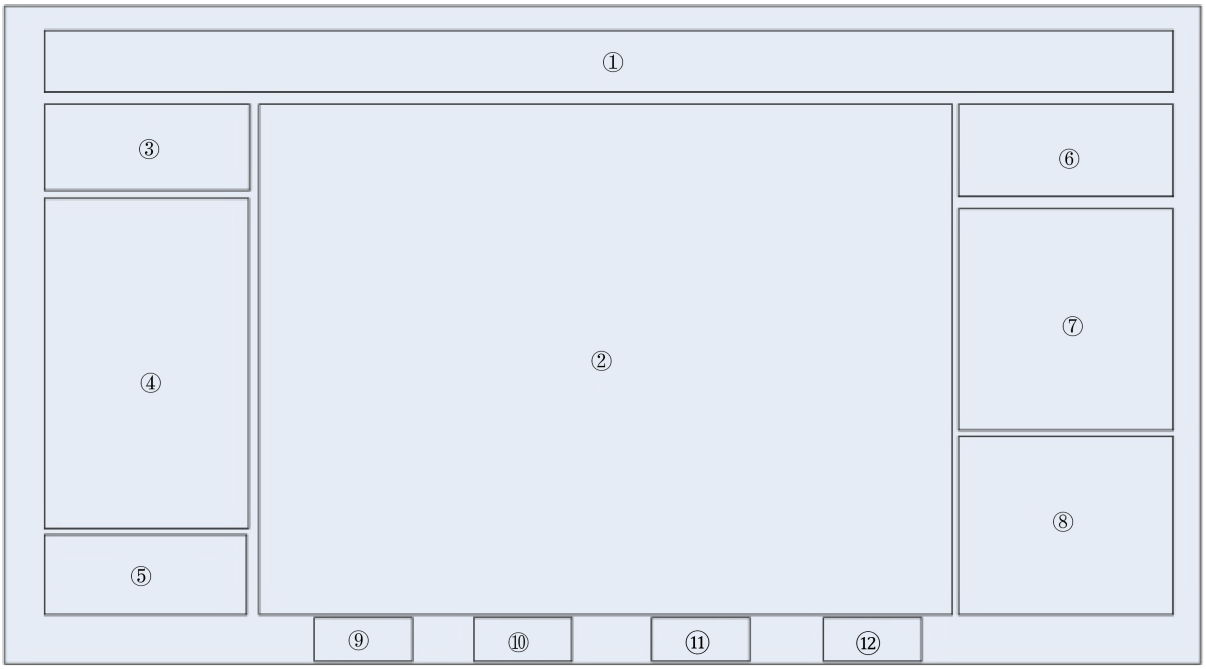


**Figure 5.** Page layout design
**图 5.** 页面布局设计图

## 7. 项目实现

根据上述总体设计和关键技术实现逻辑，完成系统全部代码的编写。并将编写的程序台部署在 Linux 服务器上运行。通过测试该系统实现了系统需求的基本功能。即用户可点开浏览器便可查看到系统页面，具体实现效果如图 6 至图 7 所示。

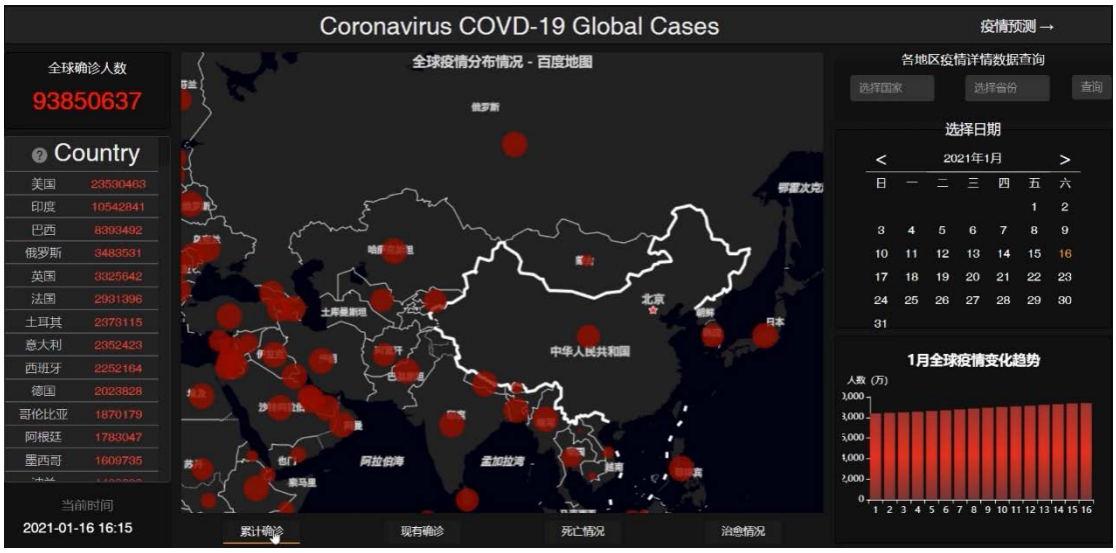


**Figure 6.** Large screen of epidemic visualization
**图 6.** 疫情可视化大屏界面图

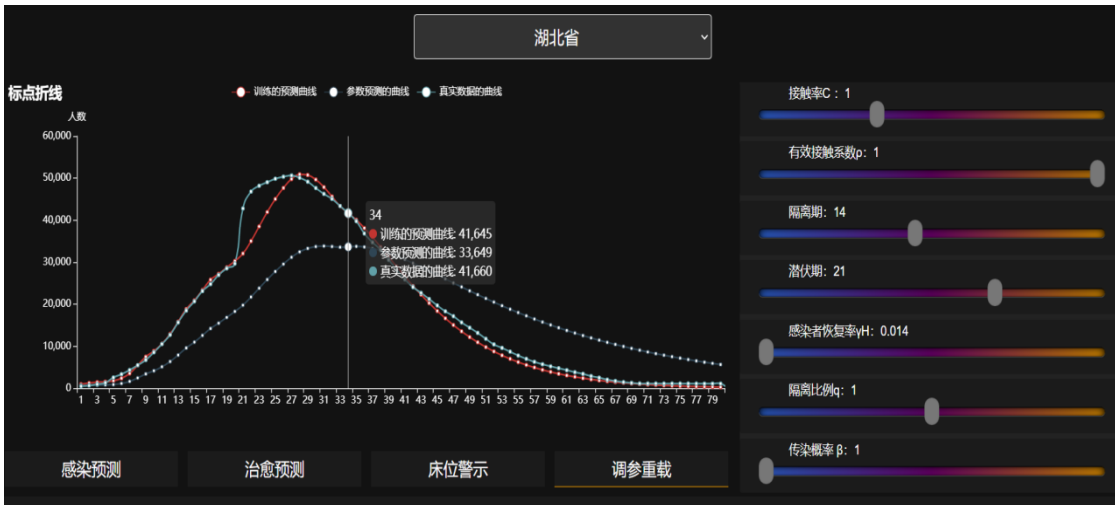


**Figure 7.** Epidemic prediction interface
**图 7.** 疫情预测界面图

## 8. 总结

系统以更好的实现疫情数据可视化监控与发展预测为开发目的，使用了 Scrapy 爬虫框架爬取了大量疫情数据源，使用了 Spark 大数据处理框架清洗并聚类统计了大量疫情数据，重点讨论了在疫情发展预测分析中 SEIR 模型的使用，在模型初始化参数上，系统使用 DE 差分进化算法并结合大量真实疫情数据，针对不同地区，拟合出其最优的模型初始化参数，从而得到了更好的预测拟合效果，为利用大数据技术实现疫情可视化监控和普适性的疫情模型预测提供实现的方法与思路。

## 参考文献

[1] Wang, J. and Guo, Y.C. (2012) Scrapy-Based Crawling and User-Behavior Characteristics Analysis on Taobao. 2012 *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Sanya, 10-12 October 2012, 44-52. https://doi.org/10.1109/CyberC.2012.17

[2] Meng, X., Bradley, J., Yavuz, B., *et al*. (2016) Mllib: Machine Learning in Apache Spark. *The Journal of Machine Learning Research*, **17**, 1235-1241.

[3] Dean, J. and Ghemawat, S. (2008) MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, **51**, 107-113. https://doi.org/10.1145/1327452.1327492

[4] White, T. (2012) Hadoop: The Definitive Guide. O'Reilly Media, Inc., Sebastopol.

[5] Li, M.Y., Graef, J.R., Wang, L., *et al*. (1999) Global Dynamics of a SEIR Model with Varying Total Population Size. *Mathematical Biosciences*, **160**, 191-213. https://doi.org/10.1016/S0025-5564(99)00030-9

[6] Diekmann, O., Heesterbeek, H. and Britton, T. (2012) Mathematical Tools for Understanding Infectious Disease Dynamics (Vol. 7). Princeton University Press, Princeton. https://doi.org/10.1515/9781400845620

[7] Ng, T.W., Turinici, G. and Danchin, A. (2003) A Double Epidemic Model for the SARS Propagation. *BMC Infectious Diseases*, **3**, Article No. 19. https://doi.org/10.1186/1471-2334-3-19

[8] 熊永平. 基于 SpringBoot 框架应用开发技术的分析与研究[J]. 电脑知识与技术, 2019(36): 76-77.