计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(9):61−67 [doi: 10.15888/j.cnki.csa.006398]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于机器学习的智能出租车预测系统①

叶　锋[1], 欧阳智超[2], 陈威彪[1], 周伊琴[1], 周晓玲[1]

[1](福建师范大学 数学与信息学院, 福州 350007)
[2](厦门大学 信息科学与技术学院, 厦门 361005)

摘　要: 为了更合理地调度出租车资源, 提出基于机器学习的智能出租车预测系统. 首先, 对波尔图出租车 GPS 数据集进行分割处理, 并抽取其中的一部分作为研究对象; 接着利用回声状态网络算法预测旅行目的地; 最后利用随机森林算法在相同情况下预测出租车抵达时间. 实验表明本系统能根据当前的波尔图出租车 GPS 数据集预测出实际出租车某段旅程的目的地和旅程所需要的时间, 以达到减少出租车资源浪费的目的.

关键词: 出租车预测; 波尔图数据集; 回声状态网络算法; 随机森林算法

引用格式: 叶锋,欧阳智超,陈威彪,周伊琴,周晓玲.基于机器学习的智能出租车预测系统.计算机系统应用,2018,27(9):61–67. http://www.c-s-a.org.cn/1003-3254/6398.html

Intelligent Taxi Forecasting System Based on Machine Learning

YE Feng[1], OUYANG Zhi-Chao[2], CHEN Wei-Biao[1], ZHOU Yi-Qin[1], ZHOU Xiao-Ling[1]

[1](School of Mathematics and Informatics, Fujian Normal University, Fuzhou 350007, China)
[2](School of Information Science and Engineering, Xiamen University, Xiamen 361005, China)

Abstract: To bring more reasonable scheduling of taxi resources, this study proposes an intelligent taxi forecasting system based on machine learning. Firstly, the GPS data set of Porto taxi is preprocessed, and a part of the training sets are taken as the research object. Then the echo state network algorithm is used to predict the travel destination of the taxi under the premise of predicting the travel destination. Finally, the taxi arrival time is predicted by using random forest algorithm in the same circumstances. Experiments show that the system can predict the actual taxi destination of the part of the journey and the time required for the journey, thus achieved the purpose of reducing the waste of taxi resources based on the current Porto taxi GPS data set.

Key words: taxi forecast; Porto data set; echo state network; random forest

引言

如今, 人们可以在世界上任何地方找到和使用基于 GPS 的出租车服务系统, 如 Uber, Lyft 等. 启用 GPS 功能的的出租车系统实时收集和记录 GPS 轨迹并且将数据上传到服务器数据采集系统. 从出租车的 GPS 轨迹和实时收集的数据流中提取的集体交通动态是了解和预测未来旅程的重要信息来源. 通过对出租车目的地和抵达时间的预测, 可以动态调度出租车的资源, 使理想的行程目的地应该是非常接近下一次行程的起点.

对出租车目的地和抵达时间的预测, 目前在国内对这方面课题的研究普遍缺乏. 国外研究学者较多运用的还是 K 最邻近算法 (K-Nearest Neighbor, K-NN) 来对数据集进行分析. 但对于大型训练数据集, K-NN

① 基金项目: 福建省自然科学基金 (2017J01739); 福建师范大学教学改革研究项目 (I201602015)
Foundation item: Natural Science Foundation of Fujian Province (2017J01739); Research Project on Teaching Reform of Fujian Normal University (I201602015)
收稿时间: 2017-08-31; 修改时间: 2017-09-26, 2017-11-03; 采用时间: 2017-11-14; csa 在线出版时间: 2018-08-16

算法在计算上代价是非常大的, 对于具有 n 个训练模式和 p 维度的训练数据集, K-NN 算法的时间复杂度为 $O(np)$[1]. 而由 Jaeger 和 Hass 提出的回声状态网络 (Echo State Network, ESN) 以及相应的学习算法为递归神经网络的研究开辟了崭新的道路. 调整仅仅针对读出网络进行. 通过该算法大大降低了训练的计算量, 又避免了大多数基于梯度下降的学习算法所难回避的局限极小现象, 并同时能够取得很好的建模精度.

本文提出了一种基于机器学习的智能出租车预测系统: 先对波尔图出租车 GPS 数据集[2]进行预处理, 并对数据集进行分割, 抽取部分数据集作为研究对象; 主要借助回声状态网络算法[3,4], 随机森林算法 (Random Forest, RF)[5]等机器学习的算法, 在算法处理器上对训练集进行训练学习, 从而在测试集中预测出出租车的目的地和抵达时间.

1 系统总体结构设计与分析

提出的智能交通系统如图 1 所示, 它包括 Sklearn 开源库处理平台、Numpy、Scipy、Pandas、Matplotlib、算法处理器、波尔图 GPS 出租车数据集. 其中, 波尔图出租车 GPS 数据集是我们实验中所使用的数据集, 是实验分析的对象. Sklearn 开源库处理平台是实验中所使用的主要开源算法平台. 算法处理器用于在 Sklearn 开源库处理平台上预测波尔图出租车 GPS 数据集的目的地和抵达时间.

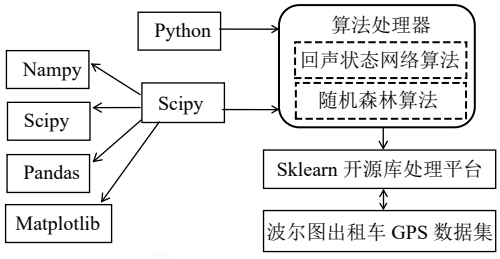

图 1 系统总体结构图

算法处理器包括回声状态网络算法和随机森林算法. 其中, 回声状态网络算法先对预处理过的波尔图出租车 GPS 训练集进行训练, 更新储备池状态, 接着导入测试集进行测试预测出该数据集出租车的目的地, 输出的数据是 WGS84 坐标. 随机森林算法将从原始训练样本集N中有放回地重复随机抽取k个样本生成新的训练样本集合, 然后根据自助样本集生成k个分类树组成随机森林, 新数据的分类结果按分类树投票多少形成的分数而定. 接着导入测试集进行测试预测出该数据集出租车的抵达时间, 输出的数据是抵达目的地所要花费的时间.

提出的系统工作流程如图 2 所示, 对波尔图 GPS 出租车数据集进行预处理抽取我们所需的特性, 并对数据集进行分割. 在 Sklearn 开源处理平台使用算法处理器预测数据集中出租车的终点和抵达时间, 将结果分别导出到两个.cvs 文件中, 分别记录出租车旅程的目的地和抵达时间.

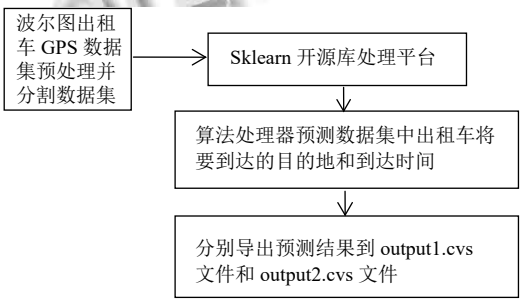

图 2 系统工作流程图

2 波尔图出租车 GPS 数据集

2.1 数据集描述

本文使用了波尔图出租车 GPS 数据集, 该数据集是公开的, 也是 Kaggle 竞赛官方使用的数据集, 故本文使用该数据集进行实验. 原始数据集中具体的属性有: TRIP_ID(旅行编号), CALL_TYPE(调度类型), ORIGIN_CALL(来电详细信息), ORIGIN_STAND(出租车站), TAXI_ID(出租车编号), TIMESTAMP(旅行开始时间戳), DAY_TYPE(日期类型), MISSING_DATA (丢失数据), PLOYLINE(旅行折线信息). PLOYLINE 是一系列 GPS 坐标, 每次坐标都是在旅程开始后的每 15 s 记录一次, 并且以上车开始, 以下车结束. 这些数据并无法直接用于出租车目的地和抵达时间的预测, 而需要进行一定的数据处理后方能使用, 以此获得旅行目的地和抵达时间, 上车坐标 (起点纬度, 起点经度) 和下车坐标 (终点纬度, 终点经度). 我们将这个数据集称为波尔图出租车 GPS 数据集.

2.2 数据集分割

1) 分割的脚本命令:

split -l 200,000 train.csv -d -a 2 train_

2) 把原来 1.80 GB 的 train 数据集分割成每个 200 MB 的数据集并对其批处理重新命名:

```
for i in *
do mv $i $i".csv"
done
```

2.3 数据集预处理

每个数据样本对应一个完成的行程. 它有 9 列的字段, 其中我们只使用:

1) TRIP_ID(String):它包含每个行程的唯一标识符.

2) POLYLINE(String):它包含映射为字符串的 GPS 坐标 (即 WGS84 格式) 列表. 字符串的开始和结尾分别用括号 (即“[”和“]”) 标识. 每对坐标也由与[LONGITUDE, LATITUDE]相同的括号确定. 该列表包含每 15 s 行程的一对坐标.

这样做的好处是能够减少冗余的字段, 加快实验中程序运行的速度, 在比较有限的时间中得出实验的结果.

3 基于回声状态网络算法预测目的地

3.1 预测流程

经过预处理的波尔图 GPS 出租车数据集在算法处理器中完成出租车目的的预测. 预测流程主要包括 3 个阶段: 预处理过程、训练数据集和测试数据集.

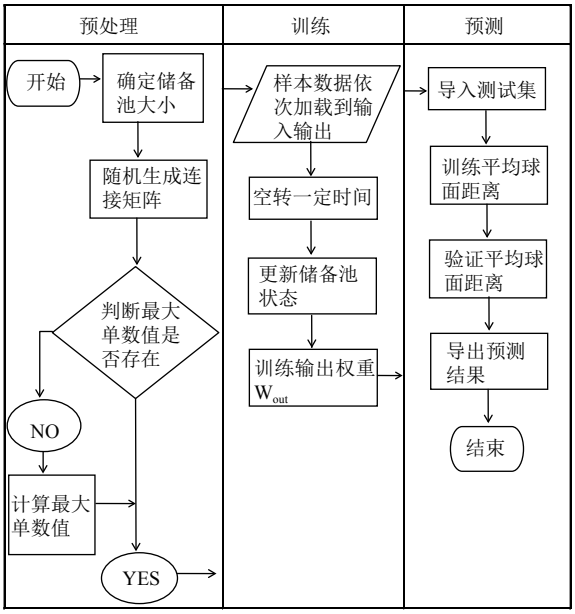

图 3 回声状态网络算法预测目的地流程图

3.2 评估指标

该系统的评估指标是平均 Haversine 距离 (MHD). 横坐标距离通常用于导航, 它根据纬度和经度来测量球体上两点之间的距离. 在两个位置之间可以计算如下:

$$\alpha=\sin^2(\frac{\phi_2-\phi_1}{2})+\cos(\phi_1)\cos(\phi_2)\sin^2(\frac{\lambda_2-\lambda_1}{2}) \quad (1)$$

$$d=2\cdot r\cdot\alpha\tan(\sqrt{\frac{a}{1-a}}) \quad (2)$$

其中ϕ是纬度, λ是经度, d是两点之间的距离, r是球体的半径, 在我们的情况下, 应以所需度量 (例如 6371 公里) 的地球半径代替.

3.3 算法描述

回声状态网络采用“储备池”代替传统神经网络[6]的隐层. “储备池”是 ESN 的核心结构, 它由大量稀疏连接的神经元组成, 并将输入信号从低维空间映射到高维空间, 唯一需要训练的参数即为输出权值矩阵. 这些特点大大简化了回声状态网络的训练算法和求解过程.

ESN 是一种特殊类型的递归神经网络, 其基本思想: 使用大规模随机连接的递归网络, 取代传统神经网络中的中间层, 从而简化网络的训练过程.

基于图 4 的结构, 我们可以看出回声状态网络是一种典型的三层递归神经网络, 由输入层、储备池、输出层构成.

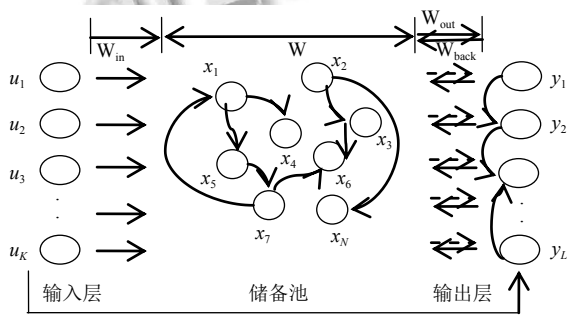

图 4 回声状态网络结构图

输入单元$u(n)$, 内部神经元状态$x(n)$以及输出单元$y(n)$在n时刻的值分别为:

$$\begin{cases} u(n)=[u_1(n),u_2(n),\cdots,u_K(n)]^{\mathrm{T}} \\ x(n)=[x_1(n),x_2(n),\cdots,x_N(n)]^{\mathrm{T}} \\ y(n)=[y_1(n),y_2(n),\cdots,y_L(n)]^{\mathrm{T}} \end{cases} \quad (3)$$

则回声状态网络状态方程为:

$$\begin{cases} x(n+1) = f(\mathrm{W}x(n) + \mathrm{W}_{\mathrm{in}}u(n) + \mathrm{W}_{\mathrm{back}}y(n)) \\ y(n+1) = f_{\mathrm{out}}(\mathrm{W}_{\mathrm{out}}[x(n+1), u(n+1), y(n)] + \mathrm{W}_{\mathrm{bias}}^{\mathrm{out}}) \end{cases} \quad (4)$$

其中W为储备池内部神经元连接权值矩阵, W_{in}为输入单元与储备池内部连接权值矩阵, $\mathrm{W}_{\mathrm{out}}$为储备池与输出单元连接权值矩阵, $\mathrm{W}_{\mathrm{back}}$为输出单元与储备池的连接权值矩阵, $f = f[f_1, f_2, \cdots, f_N]$ 表示储备池神经元激活函数, 通常情况下$f_i(i = 1,2,\cdots,N)$取做双曲正切函数. $f_{\mathrm{out}} = \left\lfloor f_{\mathrm{out}}^1, f_{out}^2, \cdots, f_{\mathrm{out}}^L \right\rfloor$表示输出函数. 特殊情况下, $f_{\mathrm{out}}^i(i = 1,2,\cdots,L)$ 取恒等函数. 在网络的训练中, 连接到储备池的连接权值矩阵W_{in}, W, $\mathrm{W}_{\mathrm{back}}$随机产生, 一经产生就固定不变. 而连接到输出单元的连接权值矩阵$\mathrm{W}_{\mathrm{out}}$ 需要经过训练得到.

3.4 模型建立

考虑到共享共同后缀的旅行导致近似或相同目的地的轨迹的内在马尔可夫性, 我们可以选择使用回声状态网络算法.

模型中有一个偏置单元和输入到读出的连接.

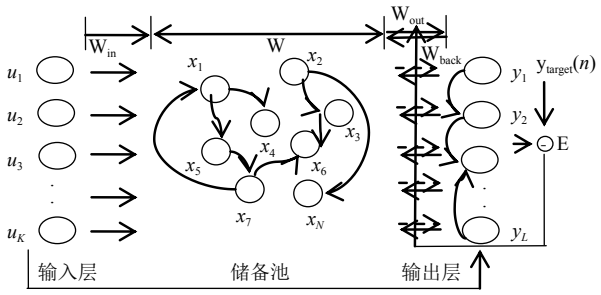

图 5 回声状态网络模型

ESN 使用具有泄漏集的离散时间连续值单元的递归神经网络 (Recurrent Neural Networks, RNN) 类型. 一般的更新方程如下[7]:

$$\tilde{x}(n) = \tanh(\mathrm{W}_{\mathrm{in}}[1; u(n)] + \mathrm{W}x(n-1)) \quad (5)$$

$$x(n) = (1-\alpha)x(n-1) + \alpha\tilde{x}(n) \quad (6)$$

其中$x(n) \in R^{Nx}$ 是储备池神经元激活的向量, $\tilde{x}$是对其的更新, 所有在时间步长n的情况下, $\tanh(\cdot)$都被应用于元素, $[\cdot;\cdot]$表示垂直向量 (或矩阵) 级联, 1 表示偏置输入, $\mathrm{W}_{\mathrm{in}} \in R^{Nx\times(1+Nu)}$和$\mathrm{W} \in R^{Nx\times Nx}$分别为输入和反复权重矩阵, $\alpha \in (0,1]$是泄露率, 除了tanh之外, 还可以使用其他S形外包装, 然而这是最常见的选择.

该模型有时被使用时没有泄露集, 这是因为$\alpha = 1$, 故$\tilde{x}(n) = x(n)$ 的特殊情况.

线性输出层的计算公式如下:

$$y(n) = \mathrm{W}_{\mathrm{out}}[1; u(n); x(n)] \quad (7)$$

其中$y(n) \in R^{Ny}$是网络输出, $\mathrm{W}_{\mathrm{out}} \in R^{Ny\times 1+Nu+Nx}$ 是输出权重矩阵. [;;]还是代表垂直向量 (或矩阵) 级联.

$\mathrm{W}_{\mathrm{out}}$训练是根据岭回归进行的, 公式如下:

$$\mathrm{W}_{\mathrm{out}} = Y_{\mathrm{target}}X^{\mathrm{T}}(XX^{\mathrm{T}} + \lambda I)^{-1} \quad (8)$$

给定收集的目标总数$Y_{\mathrm{target}} \in R^{Ny}$和$X \in R^{1+Nu+Nx}$, 用于符号简化的$X$表示$[1; U; X]$, 这考虑到偏置 (1), 输入 ($u$) 和储备池 ($x$) 的读数的连接. 它们都有助于输出, 所以它们必须被收集为x.

1) 训练阶段

对于每个波尔图出租车 GPS 训练序列:

步骤 1. 运行神经网络, 从网络状态$x(0) = 0$开始, 解除初始瞬态 (冲洗), 并用训练输入$u(1)\cdots u(N)$更新网络.

步骤 2. 对于每个时间戳n, 将储层状态$x(n)$收集到X, 目标值$y_{\mathrm{target}}(n)$ 收集到Y_{target}.

步骤 3. 冲洗是作为每个轨迹的一小部分. 即 0.2=用于冲洗的旅行点的 20%.

然后通过公式 (8) 计算出输出权重$\mathrm{W}_{\mathrm{out}}$.

当是处理大量数据而不是收集所有状态时, 矩阵$Y_{\mathrm{target}}X^{\mathrm{T}}$和$XX^{\mathrm{T}}$可以逐个计算, 一次一个模式. 因为每个模式都需要一个矩阵加法和一个外部积, 所以成本增加, 但是当计算$\mathrm{W}_{\mathrm{out}}$时, 我们已经计算了矩阵乘积. 计算公式如下:

$$Y_{\mathrm{target}}X^{\mathrm{T}} += y_{\mathrm{target}}(n) \times x(n)^{\mathrm{T}} \quad (9)$$

$$XX^{\mathrm{T}} += x(n) \times x(n)^{\mathrm{T}} \quad (10)$$

$Y_{\mathrm{target}}X^{\mathrm{T}}$ 和XX^{T} 的尺寸分别为 $(N_y \times N_x)$ 和 $(N_x \times N_x)$. 如果我们考虑到目前为止所描述的架构, 并将连接输入和偏置的x重写为$[1; U; x]$, N_x变为 $(1 + N_u + N_x)$.

2) 测试阶段

对于每个波尔图出租车 GPS 测试序列

步骤 1. 从网络状态开始, 解除初始瞬态 (冲洗), 并用公式 (8) 计算预测输出.

步骤 2. 收集每个序列的预测输出和目标值, 如果要跟踪可以按序列访问的方式收集预测输出的行为.

步骤 3. 在预测期间, 输入是相应地采用净预测模式: 生成/预测.

步骤 4. 输出波尔图出租车 GPS 测试集中每段旅程对应的目的地, 输出的数据是 WGS84 坐标.

3.5 参数选择和评估

选择通过网格搜索进行优化的参数有:

1) Nr, 储备池规模大小;

2) rho 或 sigma, 光谱半径或最大奇异值;

3) α, 泄漏率;

4) Lambda, 岭回归正则化参数;

5) Conn, 连接因子, 默认情况下为 100%.

参数的每个组合定义了一个模型, 而模型又在验证集上进行了评估. 可以通过交叉验证获得更准确的评估, 并在所有折叠上使用具有最小平均验证误差的模型.

另一个限制是随机发生器种子是固定的; 应该从不同的种子开始进行全网搜索, 从而可以生成不同的网络权重. 通过多次实验进行比较分析得到网络搜索的最佳参数情况如表 1.

表 1 网络搜索的最佳参数

参数	Nr	sigma	Lambda	α	Conn(%)
最佳值	250	0.4	0.01	1	30

4 基于随机森林算法预测抵达时间

4.1 预测流程

经过预处理的波尔图 GPS 出租车数据集在算法处理器中完成出租车目的的预测. 预测流程主要包括两个阶段: 训练数据集抽样预处理、测试数据集.

4.2 评估指标

对于行程时间, 使用均方根误差 (RMSLE) 评估预测, 定义如下:

$$RMSLE=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\ln(p_i+1)-\ln(a_i+1))^2} \quad (11)$$

这里的n是测试数据集的总观测值, p_i是观测值, a_i是旅行时间的实际值, ln是自然对数.

4.3 算法描述

随机森林中的每一棵分类树为二叉树, 其生成遵循自顶向下的递归分裂原则, 即从根节点开始依次对训练集进行划分. 在二叉树中, 根节点包含全部训练数据, 按照节点纯度最小原则, 分裂为左节点和右节点, 它们分别包含训练数据的一个子集, 按照同样的规则节点继续分裂, 直到满足分支停止规则而停止生长. 若节点n上的分类数据全部来自于同一类别, 则此节点的纯度$I(n)=0$, 纯度度量方法是 Gini 准则, 即假设$P(X_j)$是节点n上属于X_j类样本个数占训练.

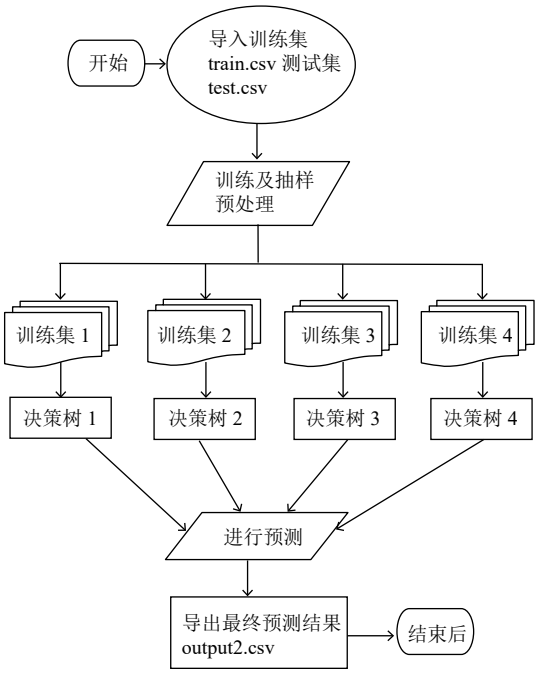

图 6 随机森林算法预测抵达时间流程图

具体算法实现过程如下:

步骤 1. 原始训练集为N, 应用自助法 (bootstrap) 有放回地随机抽取k个新的自助样本集, 并由此构建k棵分类树, 每次未被抽到的样本组成了k个袋外数据.

步骤 2. 设有m_{all}个变量, 则在每一棵树的每个节点处随机抽取m_{try}个变量 (m_{try} n m_{all}), 然后在m_{try}中选择一个最具有分类能力的变量, 变量分类的阈值通过检查每一个分类点确定.

步骤 3. 每棵树最大限度地生长, 不做任何修剪.

步骤 4. 将生成的多棵分类树组成随机森林, 用随机森林分类器对新的数据进行判别与分类, 分类结果按树分类器的投票多少而定.

4.4 抽取特征预测抵达时间

用于时间预测的一组特征与目的地预测的特征集非常相似, 差异在于将最近旅行的抵达时间视为目标变量而不是目的地. 为此提取的特征如下:

a) 旅行时间和 10 个最邻近的 Haversine 距离.

b) 内核回归特征.

除了前面描述的从部分观察到的行程中提取的所有特征之外, 我们还考虑了直接从观察到的不完全行

程中提取的以下附加的时间预测特征 (即仍在进行的行程):

a) 在迄今为止观察到的部分轨迹的最后d米和整个不完全行程上计算的平均速度, 其中$d \in \{10,20,50,100,200\}$. 这些功能在进行预测时传达最新的交通状况.

b) 到目前为止观察到的不完整旅行的最后d米的平均加速度, 其中$d \in \{10,20,50,100,200\}$.

c) 形状复杂度: (欧几里德) 行进距离与第一个和最后一个 GPS 位置之间的 Haversine 距离之间的比率. 具有更高复杂性的旅行 (例如“z - zag 之旅”(zig- zag trips) 往往是出租车司机在城市周围开车寻找乘客的行程. z-zag 的旅行时间往往较长, 所以事先确定这些行程是合理的.

d) 通过计算任何一对连续 GPS 更新之间的速度来识别 GPS 踪迹中的缺失值. 如果估计的速度超过速度限制ŷkm / h, 即使在部分观察到的行程中只有一对连续的 GPS 更新, 该行程被标记为缺少 GPS 更新的行程. 我们使用速度限制$\hat{y} \in \{100,120,140,160\}$ km/h, 缺少值的旅行往往有更长的旅程时间.

总的来说, 得出 66 个特征来预测出租车旅程的抵达时间[8].

5 检测结果分析

本系统检测算法在 Sklearn 开源库处理平台上编写, 操作系统为 Windows10, 服务器 CPU 配置为 Intel Core i5-5200U 2.2 GHz, 每台节点为 8 GB 内存. 还有一些核心包包括: Numpy, Scipy, Pandas, Matplotlib.

5.1 出租车目的地预测实验结果

出租车数据集包含 1 710 670 次旅行, 从 01/07/2013 到 24/06/2014, 其中一些是空的或缺失值, 我们在预处理时去除空轨迹, 但一些缺失值不会影响实验结果.

如图 7 是数据集中的 200 016 条训练集的终点.

图 7 部分训练集终点

对于该实验, 只使用每个行程的轨迹折线, 丢弃其他特征和空轨迹. 折线在 0-1 与最小 - 最大归一化之间进行归一化; 也可以使用 Z 分数归一化. 这样可以限制数据集的范围, 并且保证程序运行时收敛加快.

目标是轨迹的最后一点, 它被分配为每个点的目标. 因此, 网络被训练用来预测每个前缀轨迹的终点.

表 2 算法模型 MHD 值对比

算法模型	MHD
回声状态网络算法 (ESN)	2.612 19
核回归算法 (KR)[9]	2.952 36
K 最近邻算法 (KNN)[9]	2.975 27

表 2 为本算法模型的 MHD 值结果比较. 抵达目的地预测评价指标为平均 Haversine 距离 (MHD), 回声状态网络算法 (ESN) 计算结果为 2.612 19. 该值越小越好, 故 ESN 算法相对较好.

如表 3 是测试集中各段旅程目的地坐标预测的部分结果, 包括旅行 ID, 经纬度的坐标, 共有 327 条预测结果.

表 3 测试集中各旅程的目的地坐标

TRIP_ID	LATITUDE	LONGITUDE
T1	41.152 76	–8.588 52
T2	41.170 07	–8.608 96
T3	41.172 13	–8.589 54
T4	41.147 92	–8.611 06
T5	41.149 01	–8.616 17
T6	41.178 07	–8.631 67
T7	41.159 28	–8.590 06
T8	41.186 95	–8.601 18
T9	41.132 16	–8.597 57
T10	41.202 05	–8.609 02
T11	41.173 77	–8.601 04
T12	41.155 35	–8.600 55
T13	41.163 47	–8.594 59
T14	41.235 02	–8.678 71
T15	41.149 93	–8.608 34

5.2 出租车抵达时间预测实验结果

表 4 对本算法的抵达时间预测和 GBRT 和 ERT 进行了性能分析. 评价指标为 RMSLE, 可以发现随机森林算法计算结果为 0.416 74. 该值越小越好, 故 RF 算法相对较好.

表 5 是测试集中各段旅程抵达目的地所需花费时间的部分结果, 包括旅行 ID, 旅行时间, 共有 327 条预测结果. 总体表明: 提出的预测系统可以较好的完成出

租车的多项关键信息预测, 有较好的实用价值.

表 4 算法模型 *RMSLE* 值对比

算法模型	*RMSLE*
随机森林算法	0.416 74
迭代决策算法[10]	0.419 85
极端随机数算法[11]	0.416 76

表 5 测试集中各旅程抵达目的地的所需花费时间

TRIP_ID	TRAVEL_TIME(s)
T1	908.2515
T2	1020.258
T3	823.8574
T4	742.6127
T5	582.9438
T6	3180.974
T7	810.4445
T8	598.2083
T9	1162.7
T10	1552.454
T11	1841.116
T12	652.3195
T13	536.0251
T14	1493.95
T15	1199.231

6 结语

出租车公司以及近年来兴起的一批打车平台在进行车辆的动态调度时, 都需要掌握每个车辆出行终点和抵达时间的信息. 如果车辆调度员能够知道出租车完成当前出行的终点和抵达目的地所需时间, 就可以为下一个乘车需求分配距离最近且时间点最契合的车辆. 尤其是在城市的中心地带, 出租车抵达的目的地附近往往就有新的乘车需求. 因此, 对车辆目的地和抵达时间的预测具有实际的应用价值和广泛的应用市场. 本文提出了基于机器学习的智能交通预测系统, 可大致预测出租车的终点和抵达时间. 不足之处在于实验过程中, 因为电脑设备的问题, 波尔图出租车 GPS 数据集实在是太大了, 只抽取了部分的训练集来训练, 所以测试集得到的目的地和抵达时间结果有可能不够精确. 但总体来说这两种算法还是较符合我们实验的要求, 整体上性能和效果也是挺不错的.

参考文献

1 Kusner M, Tyree S, Weinberger KQ, *et al*. Stochastic neighbor compression. In: Jebara T, Xing EP, eds. Proceedings of the 31st International Conference on Machine Learning. 2014. 622–630.

2 本文使用的波尔图出租车 GPS 数据集下载官方网址: https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/rules

3 Lukosevicius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. Computer Science Review, 2009, 3(3): 127–149. [doi: 10.1016/j.cosrev.2009.03.005]

4 Gallicchio C, Micheli A. Architectural and Markovian factors of echo state networks. Neural Networks, 2011, 24(5): 440–456. [doi: 10.1016/j.neunet.2011.02.002]

5 Breiman L. Random forests. Machine Learning, 2001, 45(1): 5–32. [doi: 10.1023/A:1010933404324]

6 Fulkerson B. Pattern recognition and neural networks. Technometrics, 2009, 39(2): 233–234. [doi: 10.1080/00401706.1997.10485099]

7 Lukoševičius M. A practical guide to applying echo state networks. In: Montavon G, Orr GB, Müller KR, eds. Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 7700. Springer. Berlin, Heidelberg. 2012. 86–124. [doi: 10.1007/978-3-642-35289-8_36]

8 Tiesyte D, Jensen C. Similarity-based prediction of travel times for vehicles traveling on known routes. Annals of GIS, 2008, 14(1): 1–10. [doi: 10.1080/10824000809480633]

9 Lam HT, Bouillet E. Flexible sliding windows for kernel regression based bus arrival time prediction. Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2015. Lecture Notes in Computer Science. Springer. Cham. 2015. 68–84. [doi: 10.1007/978-3-319-23461-8_5]

10 Friedman JH. Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 2001, 29(5): 1189–1232. [doi: 10.1214/aos/1013203451]

11 Geurts P, Ernst D, Wehenkel L. Extremely randomized trees. Machine Learning, 2006, 63(1): 3–42. [doi: 10.1007/s10994-006-6226-1]