昵称：rubbninja
园龄：1年11个月
粉丝：18
关注：3
+加关注

< 2017年4月 >

日	一	二	三	四	五	六
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

搜索
找找看
谷歌搜索

常用链接
我的随笔
我的评论
我的参与
最新评论
我的标签

我的标签
机器学习(13)
学习笔记(12)
室内定位(7)
室内定位系列(7)
卡尔曼滤波(2)
C++(2)
ipython(1)
ipython notebook(1)
Java(1)
python(1)
更多

室内定位系列（三）——位置指纹法的实现（KNN）

位置指纹法中最常用的算法是k最近邻（kNN）：选取与当前RSS最邻近的k个指纹的位置估计当前位置，简单直观有效。本文介绍kNN用于定位的基本原理与具体实现（matlab、python）。

基本原理

位置指纹法可以看作是分类或回归问题（特征是RSS向量，标签是位置），监督式机器学习方法可以从数据中训练出一个从特征到标签的映射关系模型。kNN是一种很简单的监督式机器学习算法，可以用来做分类或回归。

对于在线RSS向量s，分别计算它与指纹库中各个RSS向量$\{s_1, s_2, \ldots, s_M\}$的距离（比如欧氏距离），选取最近的$k$个位置指纹（一个指纹是一个RSS向量与一个位置的对应）。

- 对于knn回归，标签是坐标x和坐标y，可以进行数值计算，使用这k个指纹的位置坐标取平均，得到作为定位结果。
- 对于knn分类，将定位区域划分为1m×1m的网格，每个网格是看作一个类别，用网格标号代替，对k个网格标号计数投票，选择票数做多的网格作为定位结果。

kNN是一种lazy式的学习方法，在上面的过程中不需要使用训练数据进行"学习"，在定位的时候直接在训练数据中搜索就可以。一些工具包中的kNN算法的训练过程中会建立一个kd树（一种数据结构），有利于在线预测时的搜索。

具体实现

Github地址，包括matlab版本和python版本
数据来源说明：http://www.cnblogs.com/rubbninja/p/6118430.html

导入数据

```
# 导入数据
import numpy as np
import scipy.io as scio
offline_data = scio.loadmat('offline_data_random.mat')
online_data = scio.loadmat('online_data.mat')
offline_location, offline_rss = offline_data['offline_location'],
offline_data['offline_rss']
trace, rss = online_data['trace'][0:1000, :], online_data['rss'][0:1000, :]
del offline_data
del online_data
```

```
# 定位准确度
def accuracy(predictions, labels):
    return np.mean(np.sqrt(np.sum((predictions - labels)**2, 1)))
```

knn回归

```
# knn回归
from sklearn import neighbors
knn_reg = neighbors.KNeighborsRegressor(40, weights='uniform', metric='euclidean')
predictions = knn_reg.fit(offline_rss, offline_location).predict(rss)
acc = accuracy(predictions, trace)
print "accuracy: ", acc/100, "m"
```

```
accuracy:  2.24421479398 m
```

knn分类

```
# knn分类，需要把坐标转换成网格标号，预测后将网格标号转换为坐标
labels = np.round(offline_location[:, 0]/100.0) * 100 + np.round(offline_location[:,
1]/100.0)
from sklearn import neighbors
```

随笔档案

2017年1月 (1)

2016年12月 (4)

2016年11月 (2)

2016年2月 (2)

2016年1月 (2)

2015年11月 (3)

2015年10月 (4)

2015年7月 (3)

2015年5月 (2)

C++基础

最新评论

1. Re:室内定位系列（五）——目标跟踪（卡尔曼滤波）

博主可以加我微信吗？ 我的名字叫吴志国，微信号：18813157365

--gguo_2017

2. Re:室内定位系列（五）——目标跟踪（卡尔曼滤波）

@gguo_2017不错哦！...

--rubbninja

3. Re:室内定位系列（五）——目标跟踪（卡尔曼滤波）

您好！我是北邮研二在读学生，研究方向室内定位，目前有个国内室内定位交流群，主要成员都是在读的研究生和博士生，还有部分的国外研究生博士生。如果博主有兴趣，欢迎加入！希望和博主多交流学习。室内定位交流群号.....

--gguo_2017

4. Re:室内定位系列（一）——WiFi位置指纹（译）

@king-blues没有。很多分类器都可以用来代替最基本的knn，但感觉没这个必要。后续会尝试各种分类器来做这个，不过都用现成的包。...

--rubbninja

```
knn_cls = neighbors.KNeighborsClassifier(n_neighbors=40, weights='uniform',
metric='euclidean')
predict_labels = knn_cls.fit(offline_rss, labels).predict(rss)
x = np.floor(predict_labels/100.0)
y = predict_labels - x * 100
predictions = np.column_stack((x, y)) * 100
acc = accuracy(predictions, trace)
print "accuracy: ", acc/100, 'm'
```

```
accuracy:  2.73213398632 m
```

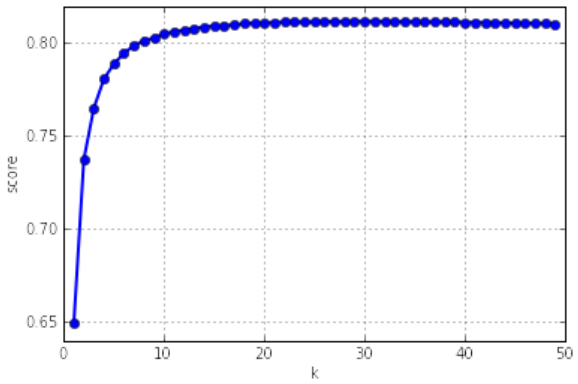
定位算法分析

加入数据预处理和交叉验证

```
# 预处理，标准化数据(其实RSS数据还算正常，不预处理应该也无所谓，特征选择什么的也都不需要)
from sklearn.preprocessing import StandardScaler
standard_scaler = StandardScaler().fit(offline_rss)
X_train = standard_scaler.transform(offline_rss)
Y_train = offline_location
X_test = standard_scaler.transform(rss)
Y_test = trace
```

```
# 交叉验证，在knn里用来选择最优的超参数k
from sklearn.model_selection import GridSearchCV
from sklearn import neighbors
parameters = {'n_neighbors':range(1, 50)}
knn_reg = neighbors.KNeighborsRegressor(weights='uniform', metric='euclidean')
clf = GridSearchCV(knn_reg, parameters)
clf.fit(offline_rss, offline_location)
scores = clf.cv_results_['mean_test_score']
k = np.argmax(scores) #选择score最大的k
```

```
# 绘制超参数k与score的关系曲线
import matplotlib.pyplot as plt
%matplotlib inline
plt.plot(range(1, scores.shape[0] + 1), scores, '-o', linewidth=2.0)
plt.xlabel("k")
plt.ylabel("score")
plt.grid(True)
plt.show()
```




```
# 使用最优的k做knn回归
knn_reg = neighbors.KNeighborsRegressor(n_neighbors=k, weights='uniform',
metric='euclidean')
predictions = knn_reg.fit(offline_rss, offline_location).predict(rss)
acc = accuracy(predictions, trace)
print "accuracy: ", acc/100, "m"
```

```
accuracy:  2.22455511073 m
```

```
# 训练数据量与accuracy
k = 29
data_num = range(100, 30000, 300)
acc = []
for i in data_num:
    knn_reg = neighbors.KNeighborsRegressor(n_neighbors=k, weights='uniform',
metric='euclidean')
    predictions = knn_reg.fit(offline_rss[:i, :], offline_location[:i, :]).predict(rss)
    acc.append(accuracy(predictions, trace) / 100)
```

5. Re:室内定位系列（一）——WiFi位置指纹（译）

你好，请问目前是否用到PSO训练ANN的方式来作指纹定位算法。

--king-blues

阅读排行榜

1. 室内定位系列（一）——WiFi位置指纹（译）(1789)

2. 统计信号处理-简单看看克拉美罗界(1016)

3. Tensorflow使用环境配置(896)

4. 室内定位系列（〇）——从人耳听觉定位原理到室内定位技术(689)

5. 室内定位系列（五）——目标跟踪（卡尔曼滤波）(634)

评论排行榜

1. 统计信号处理-简单看看克拉美罗界(3)

2. 室内定位系列（五）——目标跟踪（卡尔曼滤波）(3)

3. 室内定位系列（一）——WiFi位置指纹（译）(2)

推荐排行榜

1. 统计信号处理-简单看看克拉美罗界(2)

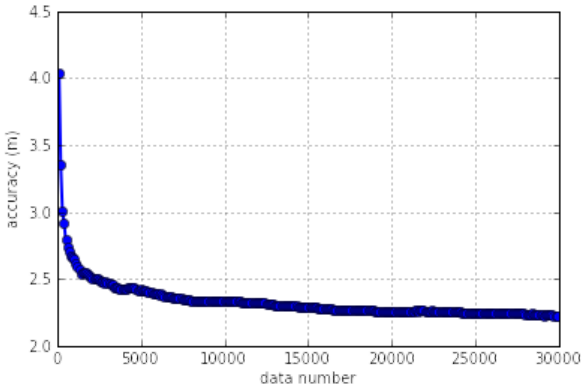
2. 室内定位系列（一）——WiFi位置指纹（译）(2)

3. 使用Java练习算法常用的基本操作(1)

4. 室内定位系列（〇）——从人耳听觉定位原理到室内定位技术(1)

5. 小技能——markdown(1)

```
# 绘制训练数据量与accuracy的曲线
import matplotlib.pyplot as plt
%matplotlib inline
plt.plot(data_num, acc, '-o', linewidth=2.0)
plt.xlabel("data number")
plt.ylabel("accuracy (m)")
plt.grid(True)
plt.show()
```



作者：rubbninja
出处：http://www.cnblogs.com/rubbninja/
关于作者：目前主要研究领域为机器学习与无线定位技术，欢迎讨论与指正！
版权声明：本文版权归作者和博客园共有，转载请注明出处。

标签：室内定位，机器学习，室内定位系列

好文要顶 关注我 收藏该文

rubbninja
关注 - 3
粉丝 - 18
+加关注

1 0

« 上一篇：室内定位系列（一）——WiFi位置指纹（译）
» 下一篇：室内定位系列（四）——位置指纹法的实现（测试各种机器学习分类器）

posted @ 2016-12-05 16:45 rubbninja 阅读(556) 评论(0) 编辑 收藏

刷新评论 刷新页面 返回顶部

发表评论

昵称：1bit

评论内容：

提交评论 退出 订阅评论

[Ctrl+Enter快捷键提交]