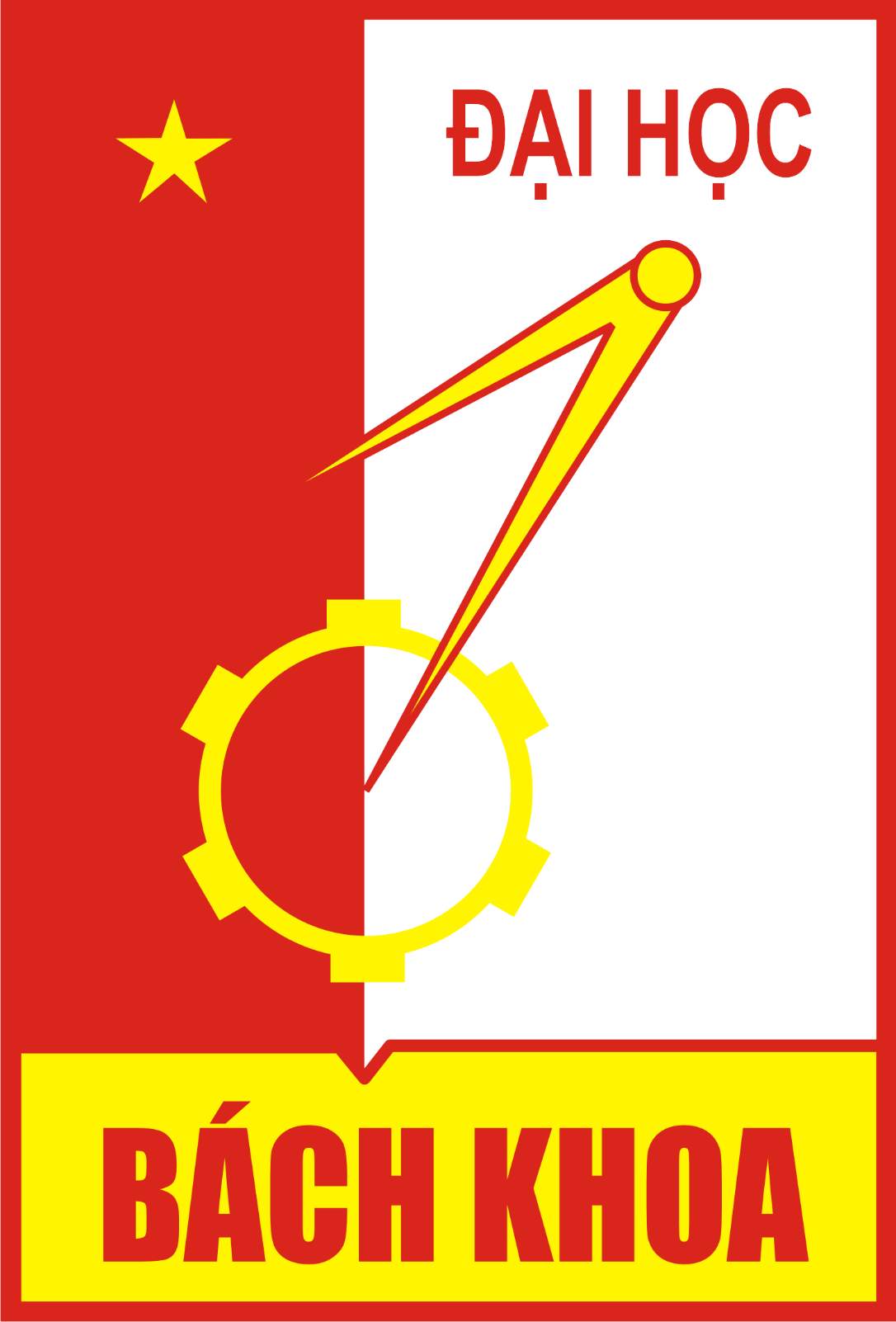
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN ĐIỆN TỬ - TRUYỀN THÔNG**



**ĐỒ ÁN THIẾT KẾ 3**

*Đề tài:* ỨNG DỤNG CÁC GIẢI THUẬT KHÁC NHAU

VÀO BÀI TOÁN ĐỊNH VỊ TRONG PHÒNG SỬ DỤNG SÓNG WIFI

Sinh viên thực hiện: Vũ Đức Tùng

Mã số sinh viên: 20134459

Lớp: KT ĐT-TT 02 K58

Giảng viên hướng dẫn: TS. Phạm Doãn Tĩnh

Hà Nội, 8-2017

***Lời mở đầu***

Ngày nay, vạn vật kết nối đang là một chủ để nóng được nhiều quốc gia, doanh nghiệp và người tiêu dùng quan tâm. Cùng với trí tuệ nhân tạo, vạn vật kết nối đã và đang có tầm ảnh hưởng lớn tới cuộc sống của con người trong đó định vị trong phòng có một ứng dụng rất tốt cho vạn vật kết nối. Bằng việc sử dụng các thuật toán học máy, việc định vị trong phòng sử dụng sóng wifi là sự kết hợp giữa vạn vật kết nối và các phương pháp học máy cũng đang là một đề tài nghiên cứu rất được quan tâm. Nhờ ý tưởng của TS. Phạm Doãn Tĩnh, em đã thực hiện môn học Đồ Án Thiết Kế 3 cùng với đề tài định vị trong phòng sử dụng sóng wifi và đã có một số kết quả nhất định.

Trong thời gian hoàn thành môn học em đã nhận được nhiều sự giúp đỡ từ thầy cô và các anh chị khóa trên.

Em xin gửi lời cảm ơn chân thành đến TS. Phạm Doãn Tĩnh đã nhiệt tình hướng dẫn và đưa cho em các gợi ý về ý tưởng và phương pháp làm việc.

Em cũng chân thành cảm ơn các thầy cô trong viện Điện Tử Truyền Thông đại học Bách Khoa đã cho em các kiến thức về các môn học từ đại cương đến chuyên ngành tạo điều kiện cho em hoàn thành môn học này.

Cuối cùng, em xin chân thành cảm ơn gia đình và bạn bè đã luôn tạo điều kiện quan tâm, giúp đỡ và động viện em trong suốt quá trình học tập và hoàn thành môn học này.

*Sinh viên thực hiện*

**Vũ Đức Tùng**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

………………………………………………………………………………

**Mục Lục**

[**Chương I: Giới thiệu** 5](#_Toc490293284)

[**1.1** **Đặt vấn đề và lý do chọn đề tài** 5](#_Toc490293285)

[**1.2** **Mục tiêu của đề tài** 5](#_Toc490293286)

[**1.3** **Phạm vi thực hiện** 6](#_Toc490293287)

[**Chương II: Định vị sử dụng sóng WIFI** 6](#_Toc490293288)

[**2.1 Khảo sát tín hiệu** 6](#_Toc490293289)

[**2.2 Lưu dữ liệu đo được** 7](#_Toc490293290)

[**2.3 Ý tưởng sử dụng sóng WIFI để xác định vị trí** 8](#_Toc490293291)

[**Chương III: Ứng dụng các thuật toán vào bài toán định vị trong phòng bằng sóng WIFI** 8](#_Toc490293292)

[**3.1 Xử lý dữ liệu** 8](#_Toc490293293)

[**3.2 Thuật toán K-Nearest Neighbors (KNN)** 11](#_Toc490293294)

[**3.3 Neural Network** 13](#_Toc490293295)

[**3.4 Support Vector Machines (SVMs)** 16](#_Toc490293296)

[**3.5 Kết luận** 18](#_Toc490293297)

# **Chương I: Giới thiệu**

* 1. **Đặt vấn đề và lý do chọn đề tài**

Trong vạn vật kết nối, việc sử dụng các vật kết nối mạng là điều thiết yếu. Việc biết được vị trí của các vật thể, chúng ta có thể điều khiển chúng dễ dàng hơn, hoặc giúp ích cho việc trao đổi thông tin giữa các vật thể đó.

Chúng ta có thể dùng tín hiệu GPS để xác định vị trí của các vật thể. Nhưng do độ chính xác của GPS không ổn định và phụ thuộc rất nhiều vào các điều kiện ngoại cảnh như thời tiết, địa hình... Hơn nữa, các thiết bị vạn vật kết nối thường là các thiết bị đơn giản, không có GPS, vậy nên việc sử dụng GPS để định vị các vật thể trong phòng là không hợp lý.

Ngoài việc sử dụng GPS, các phương pháp định vị trong nhà sử dụng các loại công nghệ không dây và cảm ứng khác cũng được nghiên cứu và xây dựng. Như sử dụng hồng ngoại, sóng vô tuyến, hệ thống cảm biến, siêu âm cũng có thể định vị cho độ chính xác cao. Tuy nhiên, những hệ thống này đòi hỏi thiết bị cơ sở hạ tầng và hệ thống các thiết bị cảm ứng giá thành cao nên gặp khó khăn trong việc triển khai với quy mô lớn.

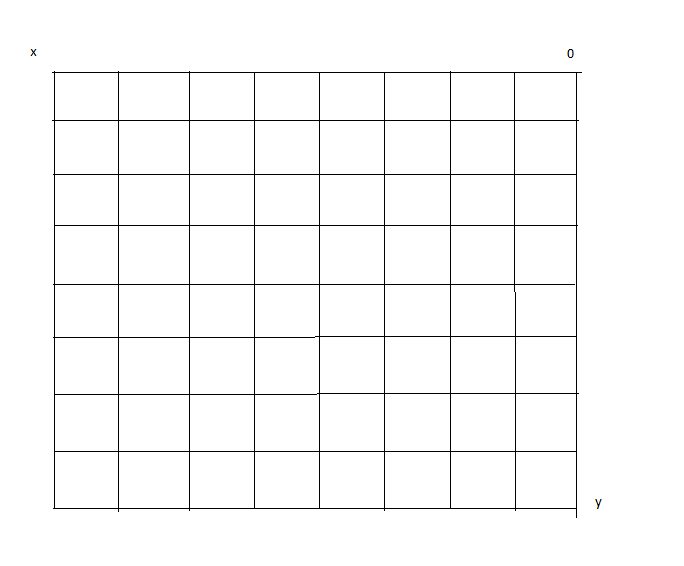
Ngày nay, các thiết bị vạn vật kết nối thường nhỏ, riêng biệt và có kết nối không dây, điển hình là hỗ trợ kết nối WIFI. Cùng với việc WIFI ngày càng nhiều và phổ biến, việc sử dụng sóng WIFI để định vị các vật thể kết nối mạng trở nên dễ dàng hơn và mở ra nhiều hướng nghiên cứu mới. Tận dụng các điểm truy cập WIFI ở nhà hay công cộng, ta có thể biết được vị trí chính xác của các vật thể kết nối trong mạng internet.

* 1. **Mục tiêu của đề tài**
* Sử dụng các thuật toán khác nhau để ứng dụng vào bài toán định vị vật thể kết nối mạng WIFI
* So sánh để tìm ra giải thuật tốt nhất về cả thời gian và độ chính xác
  1. **Phạm vi thực hiện**
* Định vị vật thể trong phòng kín không gian 2 chiều
* Đánh giá hiệu quả của các thuật toán khác nhau

# **Chương II: Định vị sử dụng sóng WIFI**

## **2.1 Khảo sát tín hiệu**

Giả sử ta chia căn phòng thành một mạng lưới 9x9 vị trí như hình 1



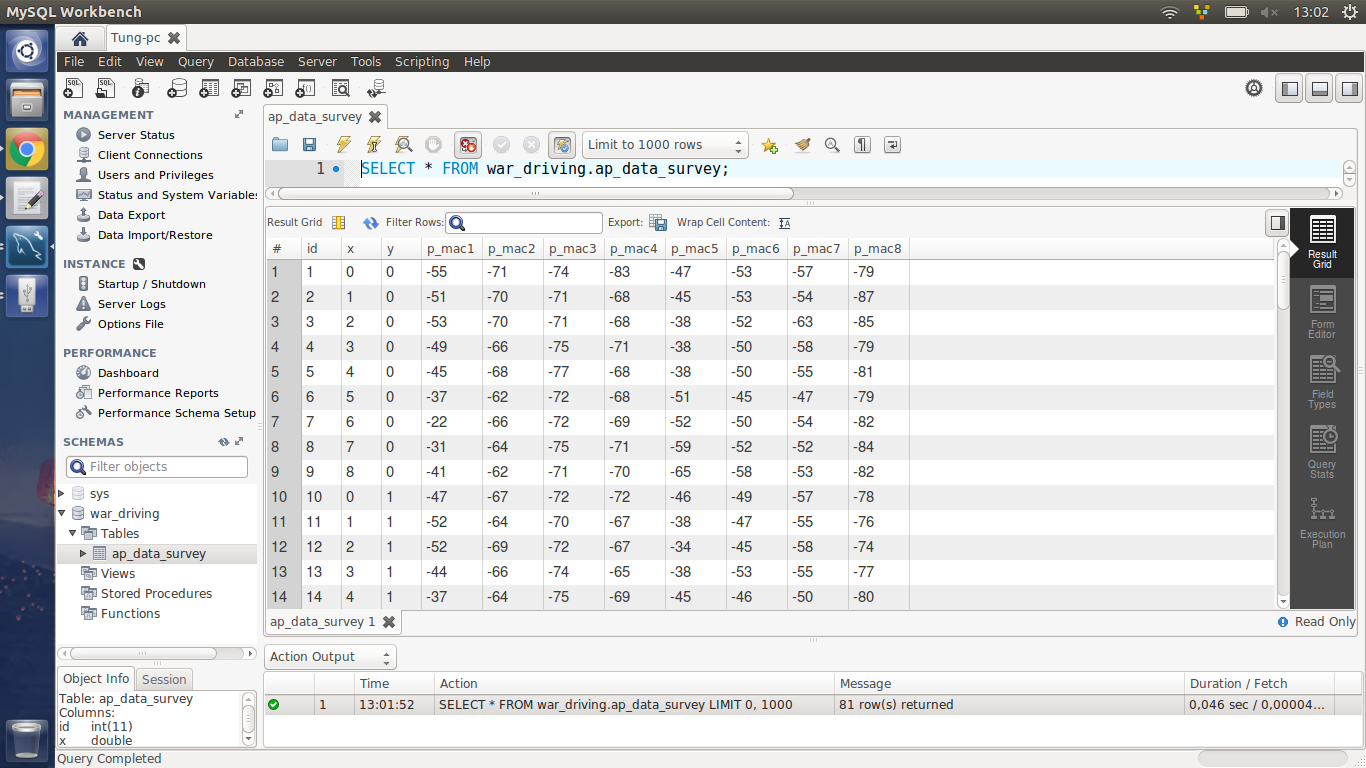
Hình 1. *Bản đồ chia tọa độ cho phòng thí nghiệm*

Tại mỗi điểm đo cường độ của tất cả các sóng WIFI có thể bắt được tại điểm đó. Ta sẽ thu được 81 điểm được khảo sát. Trong kết quả khảo sát trên phòng thí nghiệm, số điểm Access Point đo được tại tất cả các điểm là 8 Access Point. Việc xác định vị trí của các vật thể kết nối mạng trong căn phòng này sẽ dựa vào kết quả đã đo được trên đây.

Mỗi điểm khảo sát sẽ có các thuộc tính như sau: tọa độ (x,y) và cường độ các sóng WIFI (p1, p2, p3, p4, p5, p6, p7, p8). Cường độ sóng WIFI được đo bằng điện thoại di động và lấy giá trị bằng cách đo nhiều lần tại một vị trí và lấy giá trị trung bình của các lần đo đó, nhằm mục đích tăng độ chính xác của tín hiệu WIFI tại mỗi điểm.

## **2.2 Lưu dữ liệu đo được**

Dữ liệu đo được được lưu vào MySQL như hình bên dưới



Hình 2: *Dữ liệu lưu trong MySQL*

Bảng kết quả đo có 81 điểm bao gồm các thuộc tính id, tọa độ (x,y) và cường độ của 8 sóng WIFI tương ứng với 8 Access Point.

## **2.3 Ý tưởng sử dụng sóng WIFI để xác định vị trí**

Do việc phụ thuộc giữa tọa độ và cường độ sóng WIFI nên ta có thể tìm mối quan hệ giữa hai thuộc tính này, từ mối quan hệ đó, khi biết cường độ sóng WIFI tại một điểm bất kỳ ta có thể suy ra tọa độ của điểm đó nhờ việc sử dụng các thuật toán học máy.

Ý tưởng của việc sử dụng các thuật toán học máy vào bài toán này: do các thuật toán học máy dựa vào kết quả hiện tại của bài toán để dự đoán các khả năng trong tương lai, nên khi biết cường độ sóng và tọa độ của các vị trí hiện tại, ta có thể dự đoán vị trí của các điểm chưa biết dựa trên cường độ tín hiệu của các điểm đó.

# **Chương III: Ứng dụng các thuật toán vào bài toán định vị trong phòng bằng sóng WIFI**

## **3.1 Xử lý dữ liệu**

Do đây là quá trình khảo sát tính hiệu quả của các thuật toán, nên bộ dữ liệu 81 điểm sẽ được chia làm 2 phần, 66 điểm ngẫu nhiên sẽ được chọn làm các điểm training và 15 điểm còn lại được dùng làm các điểm test (điểm đánh giá kết quả chính xác của thuật toán).

Việc xử lý dữ liệu trước khi đưa vào training là việc cần thiết do dữ liệu ban đầu được lưu trong MySQL là dữ liệu thô.

Ngôn ngữ lập trình được sử dụng là Python, toàn bộ mã nguồn được sử dụng trong bài toán này được chia làm 3 tệp khác nhau. Tệp thứ nhất là obj.py có mã nguồn như sau:

class DBC:

import mysql.connector as con

un = 'root'

pss = 'Javafirst'

hst = 'localhost'

db = 'war\_driving'

connection = con.connect(user=un, password=pss, host=hst, database=db)

def getConnection(self):

return self.connection

class Point:

def \_\_init\_\_(self, \_id):

self.\_id = \_id

def setXY(self, x, y):

self.x = x

self.y = y

def setStrength(self, p1, p2, p3, p4, p5, p6, p7, p8):

self.p1 = p1

self.p2 = p2

self.p3 = p3

self.p4 = p4

self.p5 = p5

self.p6 = p6

self.p7 = p7

self.p8 = p8

if \_\_name\_\_=="\_\_main\_\_":

dbc = DBC()

connection = dbc.getConnection()

print connection

Tệp obj.py chứa 2 lớp là DBC và Point. Lớp DBC phụ trách việc lấy kết nối đến cơ sở dữ liệu. Lớp Point là khuôn mẫu cho mỗi điểm trên bản đồ ảo bao gồm các thuộc tính id, tọa độ (x,y) và 8 cường độ WIFI.

Tệp thứ 2 là helper.py có mã nguồn như sau:

#get data from database and split to 2 fields: listTest, listTrain

def getData():

from obj import DBC, Point

#init variable

listPoint = []

index\_test = [5, 8, 13, 20, 25, 30, 35, 40, 46, 53, 58, 64, 69, 72, 80] #points are used for testing

index\_train = [] #no. of trainer

listTest = [] #save test point

listTrain = [] #save train point

#connect to database and collect data from db

dbc = DBC()

connection = dbc.getConnection()

cursor = connection.cursor()

cursor.execute("select \* from ap\_data\_survey")

#convert data from database to data training

index = 0

for \_id1, x, y, p1, p2, p3, p4, p5, p6, p7, p8 in cursor:

p = Point(index)

p.setXY(x,y)

p.setStrength(p1,p2,p3,p4,p5,p6,p7,p8)

if index in index\_test:

listTest.append(p)

else:

listTrain.append(p)

index = index + 1

connection.close()

cursor.close()

return listTest, listTrain

#convert raw data to standard data

def getStandardData():

listTest, listTrain = getData()

train\_in = []

train\_out\_x = []

train\_out\_y = []

test\_in = []

test\_out\_x = []

test\_out\_y = []

train\_in, train\_out\_x, train\_out\_y = createInOut(listTrain)

test\_in, test\_out\_x, test\_out\_y = createInOut(listTest)

return train\_in, train\_out\_x, train\_out\_y, test\_in, test\_out\_x, test\_out\_y

def createInOut(listPoint):

test\_in1 = []

test\_out\_x = []

test\_out\_y = []

for i in listPoint:

test\_in = []

test\_in.append(i.p1)

test\_in.append(i.p2)

test\_in.append(i.p3)

test\_in.append(i.p4)

test\_in.append(i.p5)

test\_in.append(i.p6)

test\_in.append(i.p7)

test\_in.append(i.p8)

test\_in1.append(test\_in)

test\_out\_x.append(i.x)

test\_out\_y.append(i.y)

return test\_in1, test\_out\_x, test\_out\_y

if \_\_name\_\_ == "\_\_main\_\_":

l1, l2 = getData()

print len(l1), len(l2)

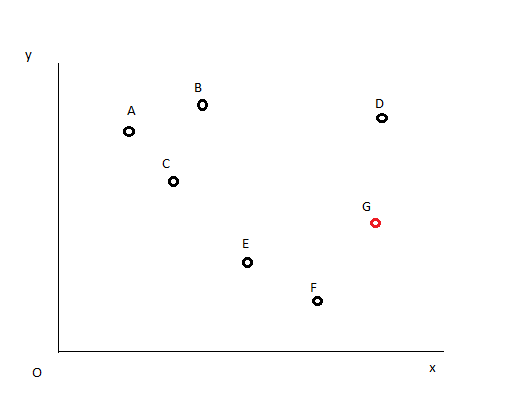
Trong tệp helper.py có 2 hàm. Hàm thứ nhất là getData() sẽ lấy dữ liệu trong cơ sở dữ liệu rồi chia làm 2 list, listTrain chứa các điểm dùng cho training, listTest chứa các điểm để đánh giá kết quả, sau đó hàm này sẽ trả về listTrain và listTest. Do hai list trên chứa các đối tượng Point, và việc khảo sát các thuật toán sử dụng thư viện sklearn trong Python nên để đưa dữ liệu vào cho việc training, ta phải chuẩn hóa theo dữ liệu qui định của sklearn, đó là trách nhiệm của hàm getStandardData().

Hàm getStandardData() sẽ chuyển đổi listTrain và listTest thành các mảng 2 chiều trong numpy. Trong bài toán này, em có tách training thành 2 phần riêng là tìm tọa độ x và tìm tọa độ y, do việc chuẩn hóa dữ liệu trong sklearn chỉ cho phép đầu ra có một tham số. Dữ liệu đầu vào cũng sẽ tách riêng thành 2 phần, cường độ phụ thuộc vào x tương ứng dữ liệu ra là x, cường độ phụ thuộc vào y tương ứng dữ liệu ra là y. Do đó hàm getStandardData() sẽ trả về train\_in, train\_out\_x, train\_out\_y và test\_in, test\_out\_x, test\_out\_y.

## **3.2 Thuật toán K-Nearest Neighbors (KNN)**

3.2.1 Hoạt động của thuật toán

Giả sử ta có hệ tọa độ Oxy và các điểm A, B, C, D, E, F. Các điểm này có 2 thuộc tính là tọa độ (x,y) và cường độ WIFI (p1, p2, p3, p4, p5, p6, p7, p8) đã biết. Điểm G đã biết cường độ WIFI (gp1, gp2, gp3, ..., gp8) nhưng chưa biết (x,y) như hình 3.



Hình 3. *Mô tả thuật toán KNN*

Thuật toán KNN dựa trên ý tưởng tìm K điểm gần nhất, K giả sử cho bằng 3, do đó ta sẽ tìm 3 điểm gần G nhất. Do các điểm càng gần nhau thì sự khác nhau giữa tín hiệu sóng WIFI cảng nhỏ, nên 3 điểm gần G nhất chính là 3 điểm có sự khác biệt giữa cường độ sóng WIFI của chúng với G là nhỏ nhất. Sự khác biệt giữa thuộc tính các cường độ tín hiệu WIFI giữa 2 điểm được tính bằng công thức

D2 =

Trong đó D là sự khác biệt giữa thuộc tính các cường độ tín hiệu WIFI. là cường độ các WIFI tại điểm thứ nhất. là cường độ các WIFI tại điểm thứ 2.

Sau khi tìm được K điểm gần nhất với G, thì giá trị trung bình các giá trị tọa độ của K điểm đó là tọa độ của G.

3.2.2 Triển khai thuật toán

Trong tệp main.py, thuật toán KNN được sử dụng thông qua sklearn như sau:

import math

import helper

train\_in, train\_out\_x, train\_out\_y, test\_in, test\_out\_x, test\_out\_y = helper.getStandardData()

from sklearn.neighbors import KNeighborsClassifier

neigh1 = KNeighborsClassifier(n\_neighbors = 7)

list\_x = neigh1.fit(train\_in, train\_out\_x).predict(test\_in)

neigh2 = KNeighborsClassifier(n\_neighbors = 7)

list\_y = neigh1.fit(train\_in, train\_out\_y).predict(test\_in)

#calculate accuracy

s = 0

for i in range(len(test\_out\_x)):

acc = math.sqrt((test\_out\_x[i]-list\_x[i])\*\*2+(test\_out\_y[i]-list\_y[i])\*\*2)

s = s + acc

accuracy = s/len(test\_out\_x)

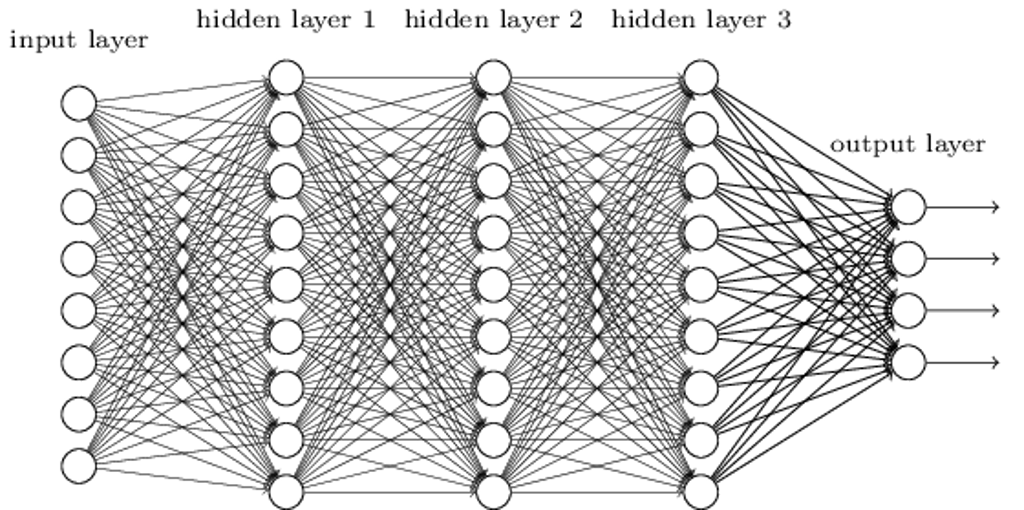
print "accuracy = ", accuracy

Sử dụng số lượng Neighbor là 7 thì độ chính xác được in ra màn hình là 3.06 m.

## **3.3 Neural Network**

3.3.1 Hoạt động của thuật toán

Trong một mạng neural như hình 4, ta có lớp input là lớp neural đầu vào chính là các thuộc tính ban đầu của dữ liệu. Sau quá trình biến đổi qua các lớp ẩn (hidden layer) sẽ cho ra dữ liệu đầu ra (output layer) là kết quả của dữ liệu đầu vào đã được phân loại.



Hình 4. *Mạng neural*

Giả sử ta có lớp input gồm có các thuộc tính p1, p2,...., p8 và lớp hidden layer 1 gồm các thuộc tính , tương tự với các lớp ẩn khác. Hoạt động của neural network là tìm mối quan hệ giữa lớp trước và lớp sau. Giả sử

...

Từ đó để tìm được hidden layer 1 ta dựa trên ma trận

(\*)

Tương tự với các hidden layer khác.

Ma trận (\*) có thể được tìm theo phương pháp Gradient Descent.

Việc training các ma trận (\*) là cốt lõi hoạt động của neural network, dựa trên dữ liệu đầu vào, các ma trận (\*) dần được hình thành và hoàn thiện hơn để cho ra các kết quả đầu ra theo phân loại. Khi đã training được các ma trận (\*), để phân loại dữ liệu mới, ta chỉ việc lấy các thuộc tính đầu vào của dữ liệu mới và nhân lần lượt với các ma trận (\*) là thu được kết quả phân loại đầu ra.

3.3.2 Triển khai thuật toán

import math

import helper

train\_in, train\_out\_x, train\_out\_y, test\_in, test\_out\_x, test\_out\_y = helper.getStandardData()

size1 = 5

size2 = 18

clf\_y = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden\_layer\_sizes=(size1,size2), random\_state=1)

clf\_y.fit(train\_in, train\_out\_y)

list\_y = clf\_y.predict(test\_in)

#print "prediction: ", clf\_y.predict(test\_in)

#print "reality: ", test\_out\_y

clf\_x = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden\_layer\_sizes=(size1,size2), random\_state=1)

clf\_x.fit(train\_in, train\_out\_x)

list\_x = clf\_x.predict(test\_in)

#calculate accuracy

s = 0

for i in range(len(test\_out\_x)):

acc = math.sqrt((test\_out\_x[i]-list\_x[i])\*\*2+(test\_out\_y[i]-list\_y[i])\*\*2)

s = s + acc

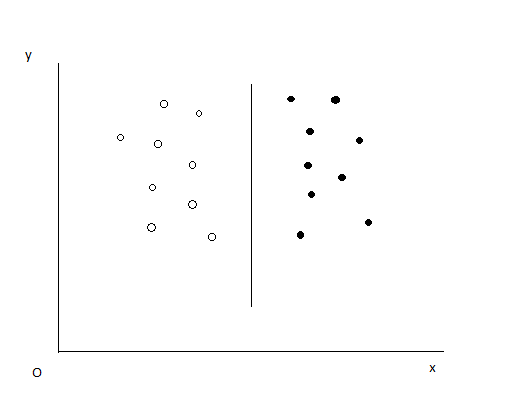
accuracy = s/len(test\_out\_x)

print "accuracy = ", accuracy

Độ chính xác là 2.09 m.

## **3.4 Support Vector Machines**

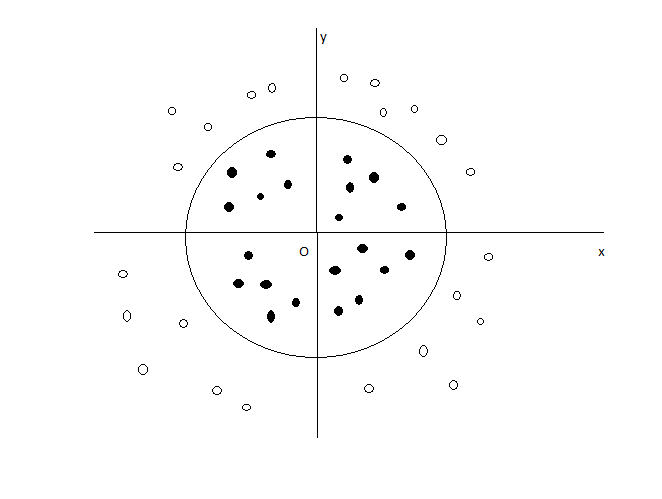
3.4.1 Hoạt động của thuật toán



Hình 5. *Phân loại tuyến tính*

Giả sử ta có dữ liệu cần được phân loại như hình 5, khi đó đường phân loại là đường thẳng và ta chỉ cần tìm mối quan hệ tuyến tính giữa x và y. Với ví dụ trên, đầu vào sẽ là x, y và đầu ra sẽ là phương trình

Nhưng giả sử dữ liệu được cho như hình 6 dưới đây, mối quan hệ tuyến tính giữa x và y không thể mô tả được đường phân loại.



Hình 6. *Phân loại phi tuyến tính*

Như vậy mối quan hệ giữa x và y là phi tuyến. Để tìm quan hệ đó, phương pháp Support Vector Machine làm bằng cách thêm các thuộc tính bổ sung ví dụ

Khi đó trong ví dụ trên, đầu vào sẽ là x, y, z. Đầu ra sẽ có dạng

3.4.2 Triển khai thuật toán

import math

import helper

train\_in, train\_out\_x, train\_out\_y, test\_in, test\_out\_x, test\_out\_y = helper.getStandardData()

#SVR

from sklearn.svm import SVR

svr1 = SVR(kernel = 'poly', C=1e4, gamma = 0.00001)

list\_x = svr1.fit(train\_in, train\_out\_x).predict(test\_in)

svr2 = SVR(kernel = 'poly', C=1e4, gamma = 0.00001)

list\_y = svr2.fit(train\_in, train\_out\_y).predict(test\_in)

#acc = 2.03m rbf c=1e4 gamma=0.00001

#acc = 1.97m when kernel = poly

#calculate accuracy

s = 0

for i in range(len(test\_out\_x)):

acc = math.sqrt((test\_out\_x[i]-list\_x[i])\*\*2+(test\_out\_y[i]-list\_y[i])\*\*2)

s = s + acc

accuracy = s/len(test\_out\_x)

print "accuracy = ", accuracy

Độ chính xác là 2.03 m khi các tham số truyền vào là kernel=‘rbf’, c=1e4 và gamma=0.00001

Độ chính xác là 1.97 m khi các tham số truyền vào là kernel=’poly’, c=1e4 và gamma=0.00001

## **3.5 Kết luận**

Trong các phương pháp phân loại đã được sử dụng trong bài toán định vị trong phòng sử dụng sóng WIFI thì phương pháp sử dụng Support Vector Machine cho độ chính xác cao nhất là 1.97 m.