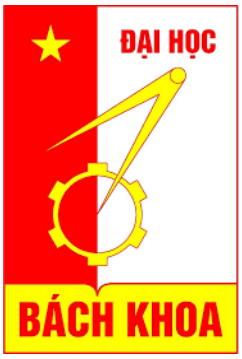
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
_____ *** _____



BÁO CÁO PROJECT1

GAME RẮN SĂN MỒI ỨNG DỤNG MẠNG NEURAL

Giáo viên hướng dẫn: **TS. NGUYỄN TUẤN DŨNG**
VIỆN CNTT&TT ĐẠI HỌC BÁCH KHOA HÀ NỘI

Sinh viên thực hiện: ĐÀO MINH KHÁNH -20183562 -KHMT04 -K63

HÀ NỘI 2020

Mục lục

Chương 1 Giới thiệu đề tài

1.1 Giới thiệu đề tài

Trò chơi kinh điển xuất hiện trên hầu hết các điện thoại 'cục gạch' của Nokia. Game không kén chọn người dùng, được nhiều lứa tuổi yêu thích. Với 4 phím di chuyển, người chơi chỉ cần khéo léo điều khiển rắn ăn các con mồi xuất hiện trên màn hình để nó dài ra mà không đụng vào tường hay cơ thể của nó.Tính đến năm 2005 đã có đến 350 triệu điện thoại cài đặt game Snake và con số này hiện vẫn không ngừng tăng lên.

Năm 1984, Nokia ra mắt chiếc điện thoại đầu tiên có tên là Mobira Talkman. Đây cũng là chiếc điện thoại đầu tiên trên thế giới có thể di chuyển. Sau hơn 10 năm phát triển, năm 1997, Nokia bắt đầu đưa lên những chiếc điện thoại của mình game 'rắn săn mồi' (snake). Tựa game đơn giản nhưng dễ gây nghiện nhanh chóng trở thành một hiện tượng trên toàn thế giới. Dù là game thứ 3 xuất hiện trên điện thoại di động nhưng Snake lại được hầu hết mọi người mặc định là game di động đầu tiên bởi sự nổi tiếng và phổ biến của nó.

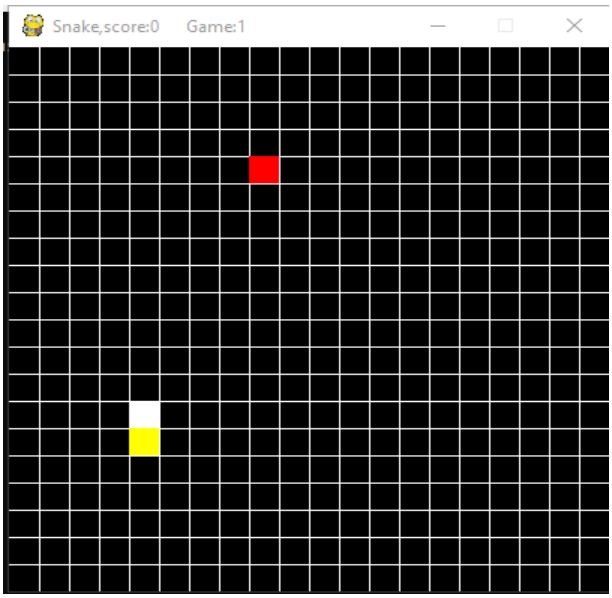
Chương 2 Xây dựng trò chơi

2.1 Nêu ý tưởng thực hiện và xây dựng trò chơi

Ở mỗi trạng thái, snake sẽ quan tâm đến môi trường xung quanh để có thể đưa ra quyết định là sẽ đi theo hướng nào là tốt cho minh và có thể ăn được quả táo. Ở tại một thời điểm thì snake chỉ có thể đi theo 3 hướng và snake chỉ quan tâm đến một cài thông số như:

- Chặn trái
- Chặn phải
- Chặn trên
- Góc giữa hướng đang đi của snake và hướng từ đầu snake đến quả táo
- 1 hướng đi cho tương lai để nó vừa không chết nhưng có thể đến ăn quả táo một cách nhanh nhất

Vậy nên chúng ta sẽ tận dụng điều này để có thể buid một model dựa trên 5 thông số trên.
Tiếp theo chúng ta sẽ cần xây dựng giao diện trò chơi rắn săn mồi để có thể khởi tạo dữ liệu cho mạng neural. Và giao diện của trò chơi được miêu tả như hình dưới đây(Hình 1).



2.2 Khởi tạo dữ điệu train cho mạng neural

Đối với mối điểm dữ liệu thì sẽ có 6 tham số bao gồm:

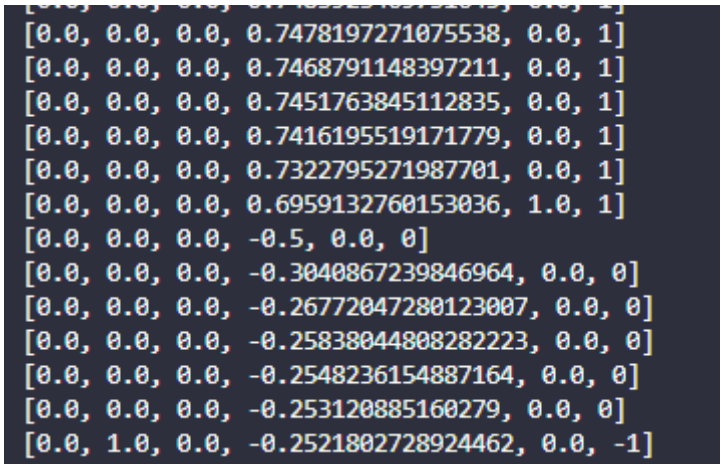
- Chặn trái: 0 hoặc 1

- Chặn trước: 0 hoặc 1
- Chặn phải: 0 hoặc 1
- Góc giữa hướng đang đi của snake và hướng với quả táo: kết quả sẽ đc normalize về khoảng -1 đến 1 để thuận tiện cho việc tính toán khi feedforward và backpropagation
- Hướng đi dự kiến: tại mỗi trạng thái của con rắn sẽ có 3 hướng đi tương ứng với:
  - -1: sang trái
  - 0: đi thẳng
  - 1: sang phải
- Nhãn: sẽ có 3 giá trị tương ứng với
  - -1: con rắn sẽ chết nếu va chạm với tường hoặc va chạm với chính nó
  - 0: con rắn sống nhưng đang đi ra xa so với quả táo
  - 1: con rắn số và có hướng đi đúng

Và sau khi đã định nghĩa được các tham số cần tìm thì đến bước khởi tạo dữ liệu. Sẽ có 2 các để khởi tạo dữ liệu là:

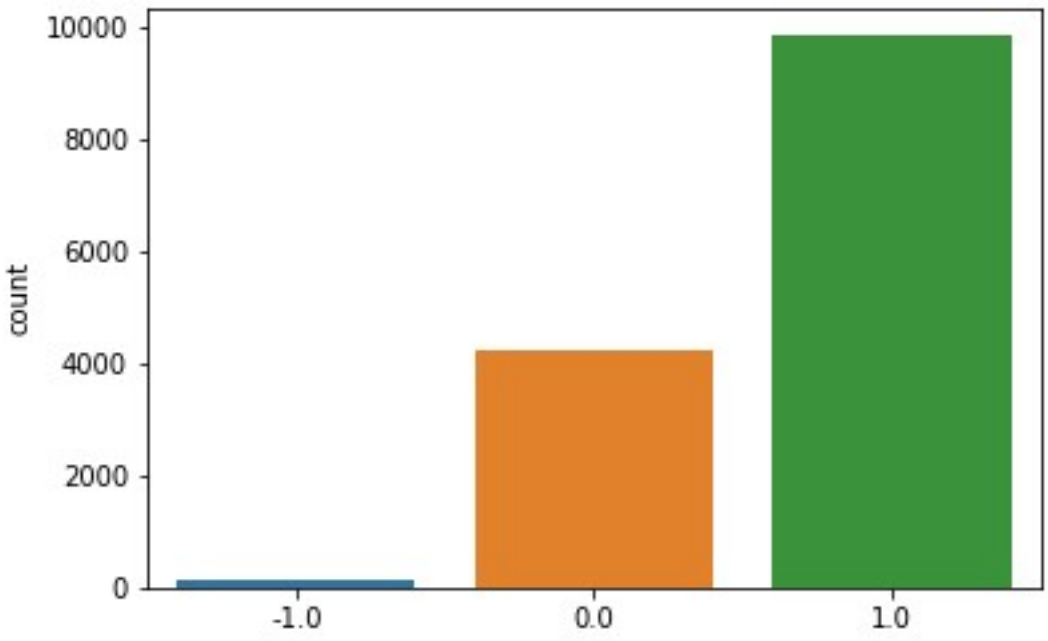
- Cho snake chạy random theo hướng đi bất kỳ
- Tự chơi

Dưới đây là một ví dụ về một phần các kết quả khi khởi tạo dữ liệu cho project này.

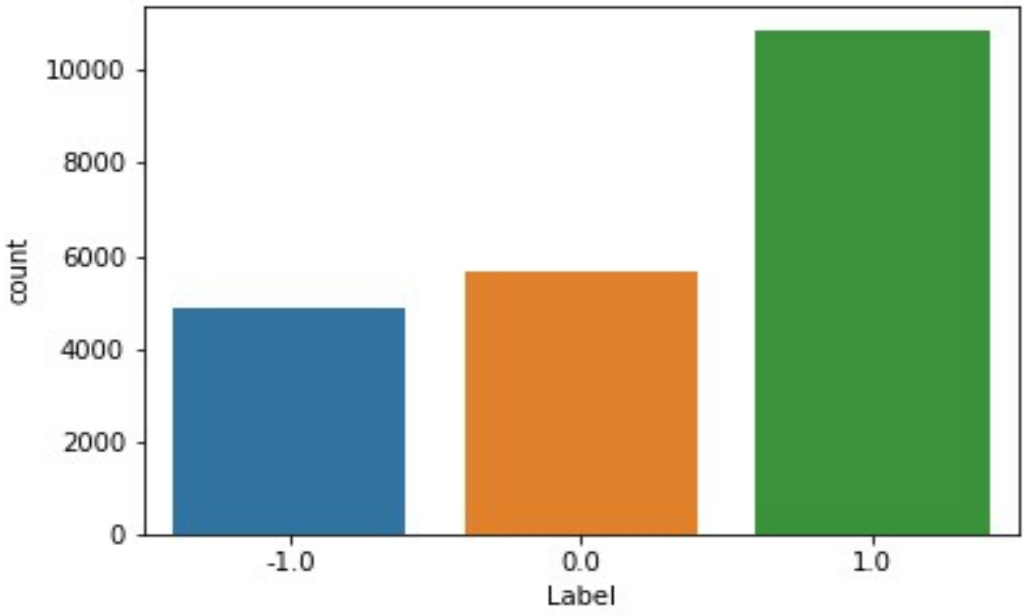


Và sau khi chơi được 120 ván game thì em có được một biểu đồ thống kê về nhãn cho các điểm dữ liệu (Hình 3) sau khi chơi. Tuy nhiên thì nhìn vào biểu đồ thì có thể thấy được có sự không cần bằng về 3 nhãn. Cụ thể:

- Có 120 giá trị -1
- Có 4251 giá trị 0
- Có 9856 nhãn 1



Do đó em đã cho con rắn chạy theo hướng random để có thể lấy thêm được nhiều nhãn -1 thêm và thu được bộ data có sự cân bằng giữa các nhãn hơn bộ data trước.



2.3 Xậy dựng model cho bài toán

Trong quá trình xây dựng cho model thì với tập dữ liệu mà em có thì em đã chia thành 3 tập con để có thể đánh giá được hiểu quả của model đang xây dựng:

- Tập train
- Tập test
- Tập validation

Xây dựng model:

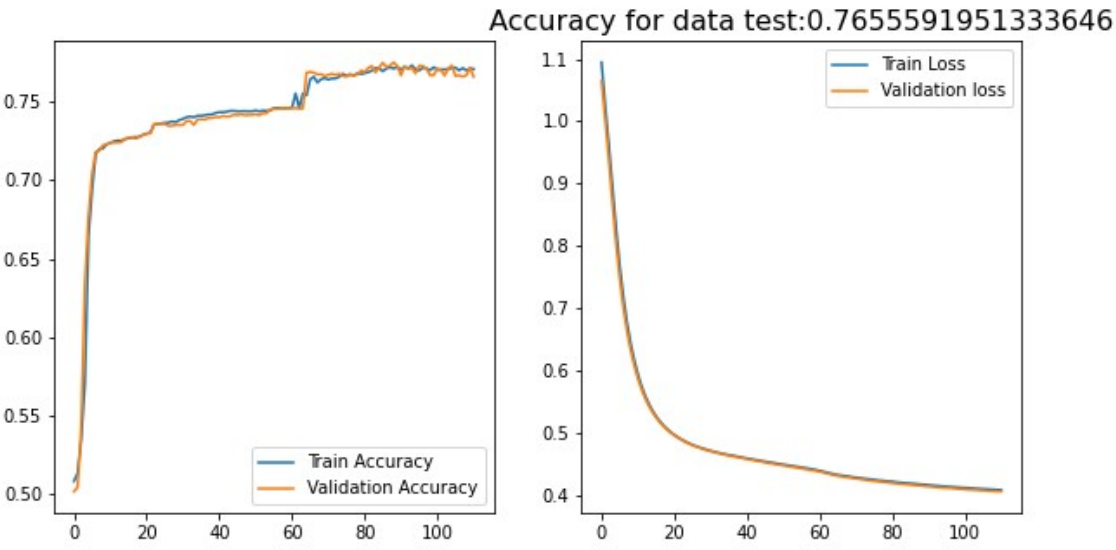
- **1 Hiden Layer**

  Đầu tiên thì em thử xem model hoạt động với 1 hiden layer có 9 unit.

```
model=Sequential()
model.add(Dense(9,input_dim=5,activation='relu'))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

```

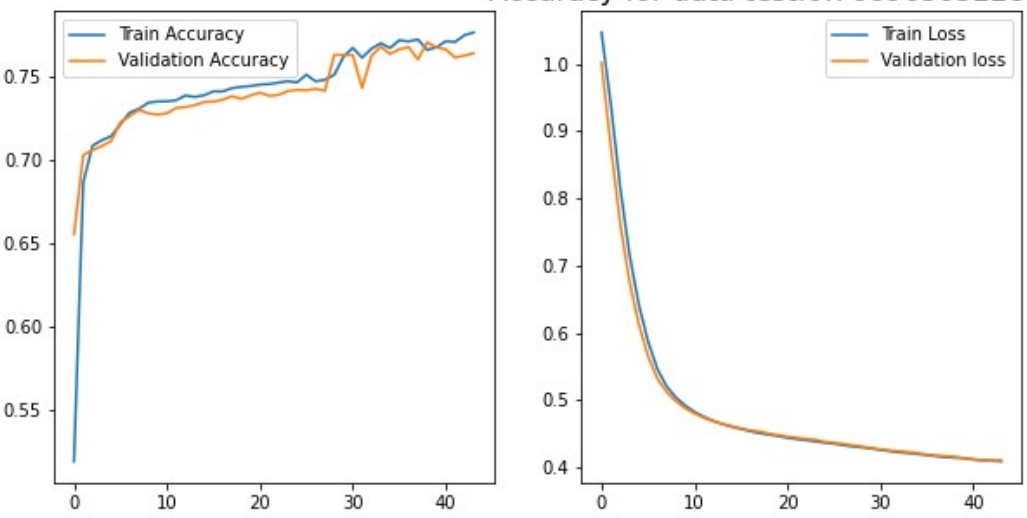
Và kết quả accuracy không quá tệ thu được từ một model đơn giản



- **2 Hiden layer**

```
model=Sequential()
model.add(Dense(9,input_dim=5,activation='relu'))
model.add(Dense(units=15, activation='relu'))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

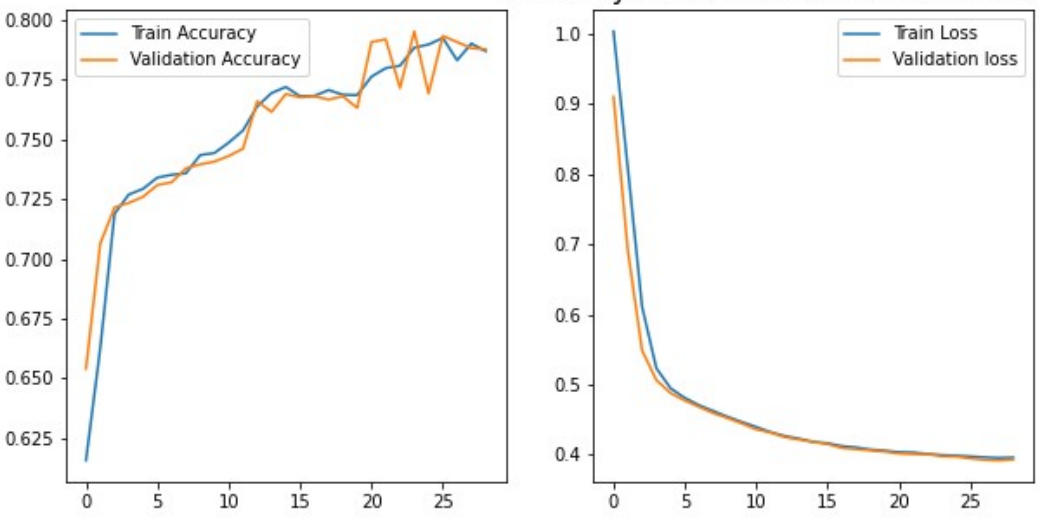
Accuracy for data test:0.7669630322882546



- **3 Hiden layer**

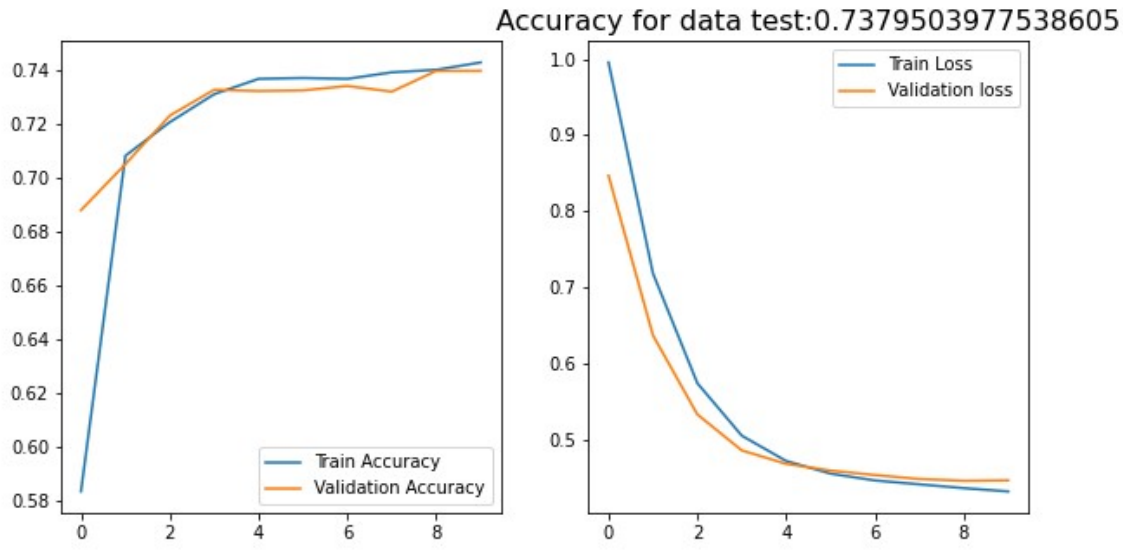
```
model=Sequential()
model.add(Dense(9,input_dim=5,activation='relu'))
model.add(Dense(units=15, activation='relu'))
model.add(Dense(units=25, activation='relu'))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

Accuracy for data test:0.7901263453439401



- **4 Hiden layer**

```
model=Sequential()
model.add(Dense(9,input_dim=5,activation='relu'))
model.add(Dense(units=15, activation='relu'))
model.add(Dense(units=30, activation='relu'))
model.add(Dense(units=10, activation='relu'))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```



Qua hình vẽ trên ta có thể thấy với model có 4 hiden layer có kết quả không tốt hơn so với model có 3 hiden layer ở trên

➔Sau nhiều lần thử thêm các hiden layer thì em nhận thấy đối với model có 3 hiden layer ở trên là có accuracy tốt nhất

2.4 Lựa chọn hướng đi cho snake

Sau khi đã có được model, thì tại mỗi thời điểm chúng ta sẽ có 3 hướng đi: đi sang trái, đi sang phải, đi thẳng

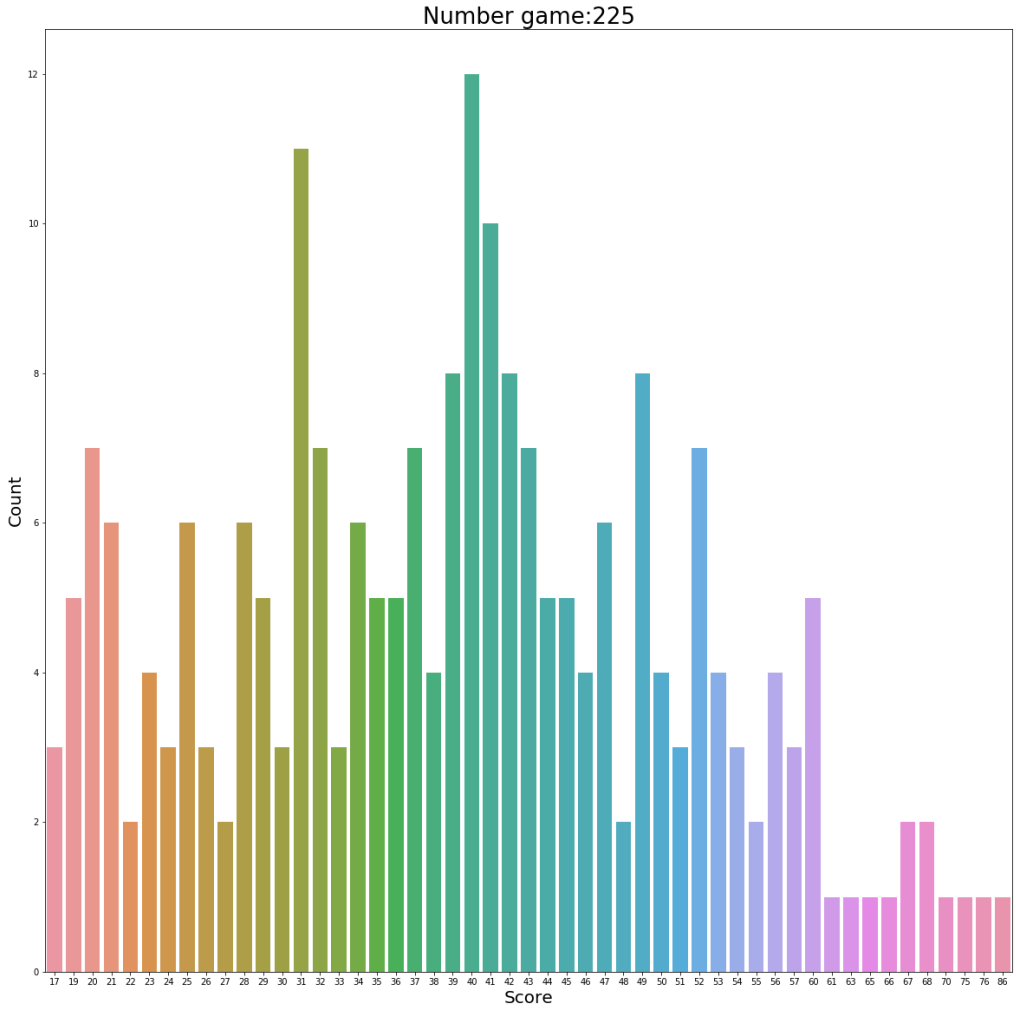
Do đó model sẽ phải dự đoạn cho 2 điểm dữ liệu và giả sử kết quả thu được lần lượt là:

Hướng đi	Nếu đi sẽ chết	Nếu đi thì sống nhưng đi ra xa	Nếu đi thì sống và đi đúng hướng
-1	A1	A2	A3
0	B1	B2	B3
1	C1	C2	C3

B1: Khởi tạo tập H={A3,B3,C3 },Y={A1,B2,C1}
B2: Chọn i trong H thỏa mãn H[i]=max(H)
B3: Kiểm tra nếu Y[i] >0.8 thì xóa H[i] khỏi tập H và Y[i] ra khỏi tập I vì xác xuất con rắn sẽ chết khá là cao nếu đi theo hướng này nên chúng ta muốn chọn một hướng đi có thể đi ra xa quả táo hơn nhưng có xác xuất chết là nhỏ hơn, chuyển sang bước 2, ngược lại chuyển sang bước 4
B4: Di chuyển theo hướng đi đã được chọn ở trên

Sau khi test với model và đưa ra hướng đi dựa trên ý tưởng trên thì em cho chạy với gần 250 game thì có được một biểu đồ thống kế số điểm

mà con rắn đạt được như hình dưới đây.



Chương 3 Kết luận

- Qua quá trình xây dựng thì em rút ra được một số nhận xét sau:
  - **Ưu điểm**
    - Việc khởi tạo dữ liệu khá là đơn giản tuy nhiên lại tốn khá nhiều thời gian
    - Model gồm 3 hiden layer nhưng hoạt động khá là tốt và đơn giản, phù hợp cho người mới bắt đầu tiếp cận với deeplearning
  - **Nhược điểm**
    - Model chưa thể dự đoán trước một cách hiệu quả, vì có xu hướng đi theo hướng gần với quả táo nên có thể nó sẽ không đoán trước được là đang đi ngõ cụt

- Phương pháp cải thiện: Dùng phương pháp học tăng cường, tuy nhiên là phương pháp này khá phực tạp và kiến thức của em còn hạn hẹp nên chưa thể xây dựng dựa trên phương pháp này

- Qua môn học này thì em tỉm hiểu được một số kiến thức cơ bản mở đầu về deeplearning và tự xây dựng được kiến trúc nhỏ, hi vọng thầy cô và các bạn có thể góp ý thêm để em có thể củng cố thêm kiến thức và học hỏi thêm được nhiều thứ hơn. Em xin chân thành cảm ơn.