TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**NHẬP MÔN TRÍ TUỆ NHÂN TẠO**

**Đề Tài 20**

*Người thực hiện*: **LÊ HUYỀN ĐỨC – 51900789**

**NGUYỄN TRUNG KIÊN – 51900811**

Lớp **: 19050402**

Khoá  **: 23**

*Người hướng dẫn*: **TS BÙI THANH HÙNG**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**NHẬP MÔN TRÍ TUỆ NHÂN TẠO**

**Đề Tài 20**

*Người thực hiện*: **LÊ HUYỀN ĐỨC – 51900789**

**NGUYỄN TRUNG KIÊN – 51900811**

Lớp **: 19050402**

Khoá  **: 23**

*Người hướng dẫn*: **TS BÙI THANH HÙNG**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022**

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn sâu sắc và chân thành nhất tới thầy Bùi Thanh Hùng. Trong suốt cả thời gian học tập và nghiên cứu trong bộ môn Nhập môn Trí Tuệ Nhân Tạo, thầy đã luôn rất tâm huyết và sẵn sàng giải đáp toàn bộ thắc mắc cho chúng em. Tuy kiến thức môn học rất khô khan, phức tạp và khó hiểu, nhưng thông qua thầy, thầy đã cho các bài tập để chúng em có thể bám sát và vận dụng các kiến thức đã học cũng như tự tìm hiểu để có thể hiểu sâu hơn các kiến thức được truyền đạt sau mỗi buổi học.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của chúng tôi và được sự hướng dẫn của TS. Bùi Thanh Hùng. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 16 tháng 11 năm 2022*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Duc*

*Lê Huyền Đức*

*Kien*

*Nguyễn Trung Kiên*

PHẦN ĐÁNH GIÁ CỦA GIẢNG VIÊN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Bài toán đầu tiên là sử dụng phương pháp reinforcement learning để viết chương trình cho game tic tac toe có số hàng, cột, số điểm chiến thắng và lượt đi trước do người chơi chọn. Phương pháp làm là huấn luyện cho máy chơi game tic tac toe bằng phương pháp reinforcement learning để sinh ra policy rồi máy sẽ sử dụng policy ấy để đấu với người chơi. Kết quả đạt được là tạo ra thành công 1 chương trình chơi game tic tac toe đúng yêu cầu đề bài. Nhóm em có nhận xét là phương pháp reinforcement learning phù hợp với việc huấn luyện cho máy chơi game tic tac toe. Khuyết điểm là nó phụ thuộc vào số lần huấn luyện nên để tăng độ chính xác cần tốn nhiều thời gian.

Bài toán thứ 2 là bài toán nhận dạng chữ số dựa trên hình ảnh viết tay dùng dữ liệu MNIST (Dữ liệu này đã được chuyển thành pixel) bằng nhiều phương pháp học máy, học sâu. Nhóm em sử dụng 2 phương pháp là KNN và CNN để giải quyết bài toán này. Kết quả là cả 2 phương pháp đều đưa ra kết quả chính xác rất cao nhưng xét về tính phù hợp cho bài toán xử lý ảnh thì CNN vẫn hiệu quả hơn.

MỤC LỤC

[LỜI CẢM ƠN 1](#_Toc120730282)

[PHẦN ĐÁNH GIÁ CỦA GIẢNG VIÊN 3](#_Toc120730283)

[TÓM TẮT 4](#_Toc120730284)

[MỤC LỤC 1](#_Toc120730285)

[DANH MỤC CÁC HÌNH VẼ 3](#_Toc120730286)

[CHƯƠNG 1 5](#_Toc120730287)

[Sử dụng giải thuật Reinforcement Learning cho Game Tic Tac Toe người chơi với máy 5](#_Toc120730288)

[1.1 Mô tả cấu trúc dữ liệu 5](#_Toc120730289)

[1.2 Sơ đồ giải thuật 5](#_Toc120730290)

[1.2.1 Sơ đồ tổng quát 6](#_Toc120730291)

[1.2.2 Sơ đồ huấn luyện máy 6](#_Toc120730292)

[1.2.3 Sơ đồ trận đấu giữa người và máy 8](#_Toc120730293)

[1.3 Hiện thực 8](#_Toc120730294)

[1.4 Kết quả và thảo luận 24](#_Toc120730295)

[1.4.1 Trình bày kết quả đạt được 24](#_Toc120730296)

[1.4.2 Thảo luận về kết quả và giải thuật đã hiện thực 24](#_Toc120730297)

[CHƯƠNG 2 26](#_Toc120730298)

[Tìm hiểu cơ bản về thị giác máy tính với dữ liệu MNIST nổi tiếng 26](#_Toc120730299)

[2.1 Giới thiệu về bài toán 26](#_Toc120730300)

[2.2 Phân tích yêu cầu của bài toán 26](#_Toc120730301)

[2.2.1 Yêu cầu của bài toán 26](#_Toc120730302)

[2.2.2 Các phương pháp giải quyết bài toán 27](#_Toc120730303)

[2.2.2.1 Phương pháp K-nearest neighbors: 27](#_Toc120730304)

[2.2.2.2 Phương pháp SVM: 27](#_Toc120730305)

[2.2.2.3 Phương pháp CNN (Convolutional Neural Network): 29](#_Toc120730306)

[2.2.3 Phương pháp đề xuất giải quyết bài toán 31](#_Toc120730307)

[2.3 Phương pháp giải quyết bài toán 32](#_Toc120730308)

[2.3.1 Mô hình tổng quát 32](#_Toc120730309)

[2.3.1.1 Mô hình KNN 32](#_Toc120730310)

[2.3.1.2 Mô hình CNN 32](#_Toc120730311)

[2.3.2 Đặc trưng của mô hình đề xuất 33](#_Toc120730312)

[2.3.2.1 Xử lý dữ liệu chung cho 2 mô hình 33](#_Toc120730313)

[2.3.2.2 Mô hình KNN 33](#_Toc120730314)

[2.3.2.3 Mô hình CNN 33](#_Toc120730315)

[2.4 Thực nghiệm 34](#_Toc120730316)

[2.4.1 Dữ liệu 35](#_Toc120730317)

[2.4.2 Xử lý dữ liệu 35](#_Toc120730318)

[2.4.3 Công nghệ sử dụng 35](#_Toc120730319)

[2.4.4 Cách đánh giá 35](#_Toc120730320)

[2.5 Kết quả đạt được 35](#_Toc120730321)

[2.6 Kết luận 36](#_Toc120730322)

[2.6.1 Trình bày tóm tắt các kết quả đạt được của bài toán trên: 36](#_Toc120730323)

[2.6.2 Hạn chế: 37](#_Toc120730324)

[2.6.3 Hướng phát triển trong tương lai: 37](#_Toc120730325)

[TÀI LIỆU THAM KHẢO 39](#_Toc120730326)

[TỰ ĐÁNH GIÁ 40](#_Toc120730327)

DANH MỤC CÁC HÌNH VẼ

[Hình 1.1 : Khai báo thư viện. 8](#_Toc120730261)

[Hình 1.2: Hàm khởi tạo init và getHash lưu lại trạng thái bàn cờ dưới dạng mảng 1 chiều. 9](#_Toc120730262)

[Hình 1.3: Hàm winner để xét và kiểm tra các điều kiện thắng trong một ván chơi (1). 10](#_Toc120730263)

[Hình 1.4: Hàm winner để xét và kiểm tra các điều kiện thắng trong một ván chơi (2). 10](#_Toc120730264)

[Hình 1.5 : Hàm winner để xét và kiểm tra các điều kiện thắng trong một ván chơi (3). 11](#_Toc120730265)

[Hình 1.6: Hàm xét các vị trí có thể đi, hàm cập nhật trạng thái bàn cờ, hàm xét lại reward để hiển thị kết quả của một ván đấu đã kết thúc, hàm reset lại bàn cờ. 12](#_Toc120730266)

[Hình 1.7: Hàm play để huấn luyện mô hình, cho hai máy tự đánh với nhau. 13](#_Toc120730267)

[Hình 1.8: Hàm play1 để đi vào trường hợp người đánh với máy (Người đi đầu). 14](#_Toc120730268)

[Hình 1.9: Hàm play2 để đi vào trường hợp người đánh với máy (Người đi sau). 15](#_Toc120730269)

[Hình 1.10: Hàm showBoard để vẽ bàn cờ và hiển thị bàn cờ cùng các trạng thái đã thực hiện trên nó. 16](#_Toc120730270)

[Hình 1.11: Hàm để khởi tạo các đối tượng của class Player, hàm để lưu lại trạng thái bàn cờ dưới dạng mảng 1 chiều và hàm chọn hành động đặt cờ cho máy. 17](#_Toc120730271)

[Hình 1.12: Hàm để lưu lại trạng thái bàn cờ trong quá trình huấn luyện, hàm để cập nhật các trọng số cho các trường hợp khi kết thúc ván cờ, hàm reset lại ván cờ, hàm lưu giữ các trạng thái đã được huấn luyện và hàm để lấy ra các trạng thái huấn luyện để đánh với người chơi. 18](#_Toc120730272)

[Hình 1.13: Hàm khởi tạo đối tượng người chơi, hàm chọn vị trí đánh dựa vào vị trí hàng và cột. 19](#_Toc120730273)

[Hình 1.14: Hàm Tic\_Tac\_Toe, thực hiện khởi tạo bàn cờ, khởi tạo đối tượng chơi và kiểm tra các vị trí đánh hợp lý, đồng thời save và load policy phù hợp (1). 20](#_Toc120730274)

[Hình 1.15: Hàm Tic\_Tac\_Toe, thực hiện khởi tạo bàn cờ, khởi tạo đối tượng chơi và kiểm tra các vị trí đánh hợp lý, đồng thời save và load policy phù hợp (2). 21](#_Toc120730275)

[Hình 1.16: Quá trình người đánh với máy (Máy đi trước) (1). 22](#_Toc120730276)

[Hình 1.17: Quá trình người đánh với máy (Máy đi trước) (2). 23](#_Toc120730277)

[Hình 1.18: Quá trình người đánh với máy (Máy đi trước) (3). 24](#_Toc120730278)

[Hình 2.1: Tập dữ liệu MNIST. 31](#_Toc120730279)

[Hình 2.2: Biểu đồ thể hiện accuracy của 2 mô hình KNN và CNN. 36](#_Toc120730280)

[Hình 2.3: Kết quả gán nhãn cho các dữ liệu của file test.csv. 37](#_Toc120730281)

CHƯƠNG 1

Sử dụng giải thuật Reinforcement Learning cho Game Tic Tac Toe người chơi với máy

1.1 Mô tả cấu trúc dữ liệu

Cấu trúc dữ liệu được dùng cho bài toán là ma trận 2 chiều có n hàng, m cột, từ đây ta sẽ xây dựng nên một bàn cờ với các giá trị khởi tạo là 0.

1.2 Sơ đồ giải thuật

1.2.1 Sơ đồ tổng quát

Đấu xong?

Yes

Có policy ?

Đấu người và máy

Huấn luyện mô hình

Nhập n,m,x,p

Yes

No

No

1.2.2 Sơ đồ huấn luyện máy

Người tiếp theo đánh tương tự

Cập nhật giá trị trạng thái bằng công thức V(S\_t) = V(S\_t) + alpha [V(S{t+1}) - V(S\_t)]

Reset trạng thái của 2 người chơi và bàn cờ

Cập nhật trạng thái bàn cờ và trạng thái mới của người vừa đánh sau lượt đánh

Kết thúc ván?

Yes

No

Chọn số lần huấn luyện (Số lần chơi)

Người chơi chọn vị trí đánh dựa trên giá trị cao nhất của trạng thái bàn cờ (Random vị trí nếu chưa có trạng thái nào có giá trị)

Nhập n,m,x,p

Tạo 2 người chơi và bàn cờ

Còn lượt ?

Yes

No

1.2.3 Sơ đồ trận đấu giữa người và máy

Người nhập số dòng, số cột muốn đi

Máy đánh theo policy đã huấn luyện

Kết thúc ?

Kết thúc ?

Máy đánh theo policy đã huấn luyện

Người nhập số dòng, số cột muốn đi

Ai đi trước?

Yes

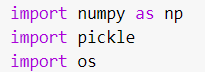
Yes

No

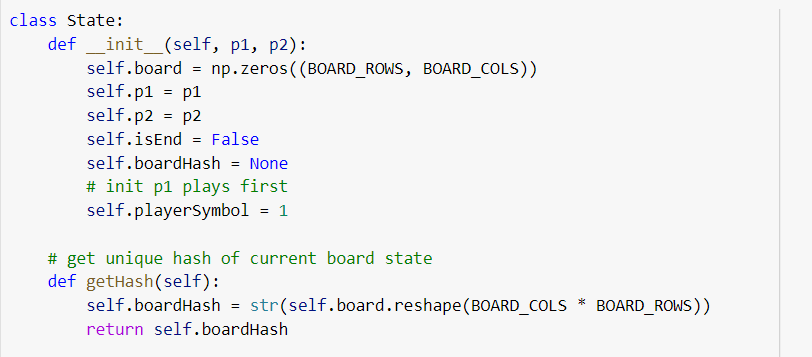
No

1.3 Hiện thực

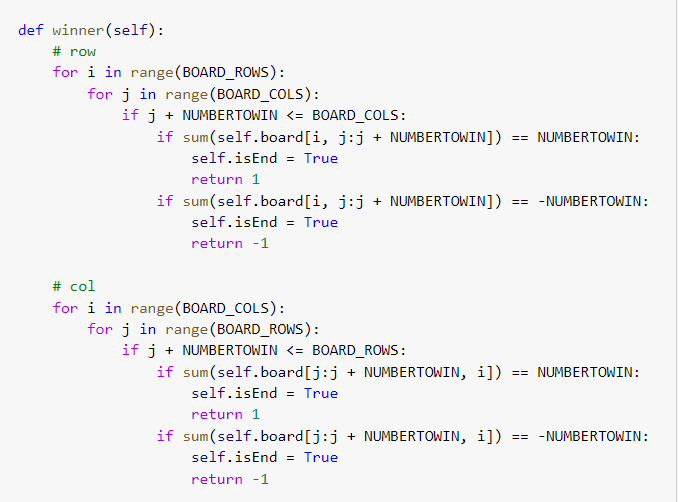
Trình bày code hiện thực của bài toán



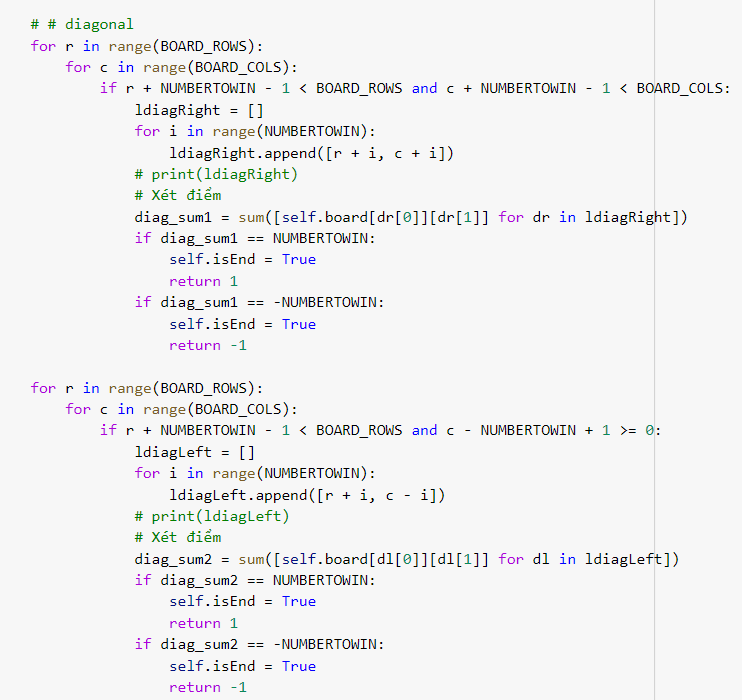
Hình 1.1 : Khai báo thư viện.



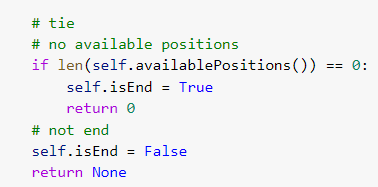
Hình 1.2: Hàm khởi tạo init và getHash lưu lại trạng thái bàn cờ dưới dạng mảng 1 chiều.



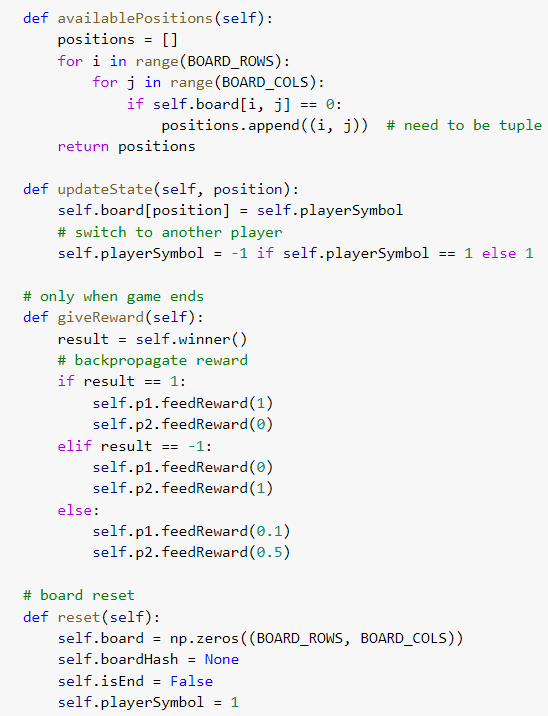
Hình 1.3: Hàm winner để xét và kiểm tra các điều kiện thắng trong một ván chơi (1).



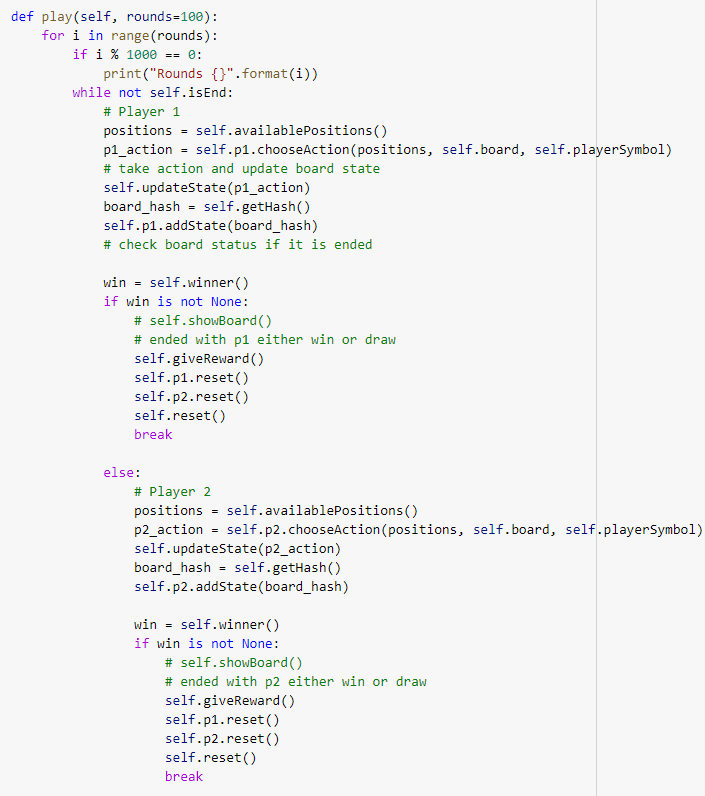
Hình 1.4: Hàm winner để xét và kiểm tra các điều kiện thắng trong một ván chơi (2).



Hình 1.5 : Hàm winner để xét và kiểm tra các điều kiện thắng trong một ván chơi (3).



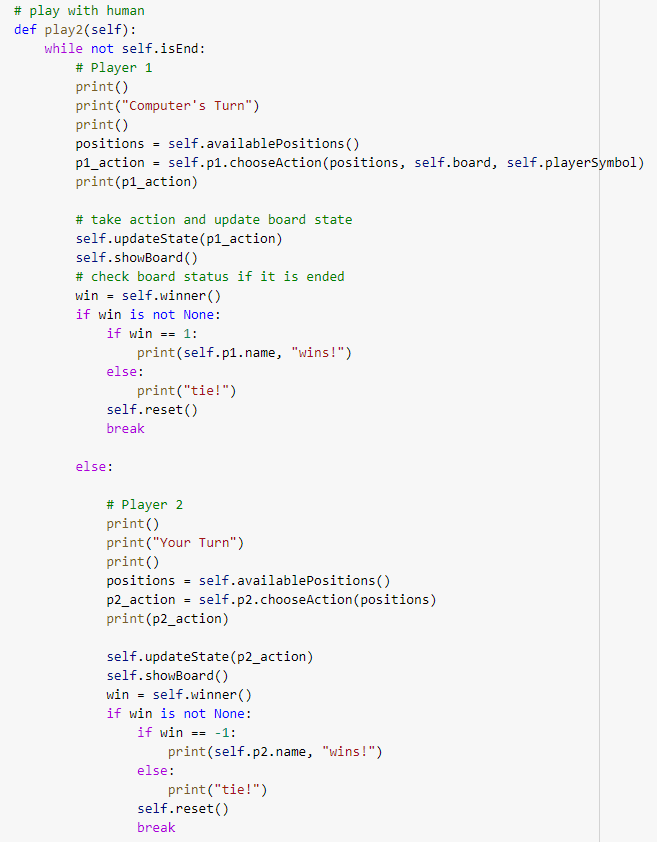
Hình 1.6: Hàm xét các vị trí có thể đi, hàm cập nhật trạng thái bàn cờ, hàm xét lại reward để hiển thị kết quả của một ván đấu đã kết thúc, hàm reset lại bàn cờ.



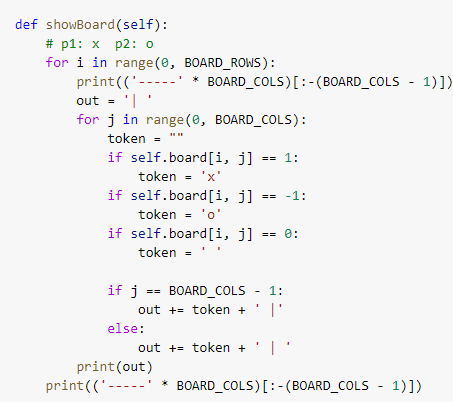
Hình 1.7: Hàm play để huấn luyện mô hình, cho hai máy tự đánh với nhau.



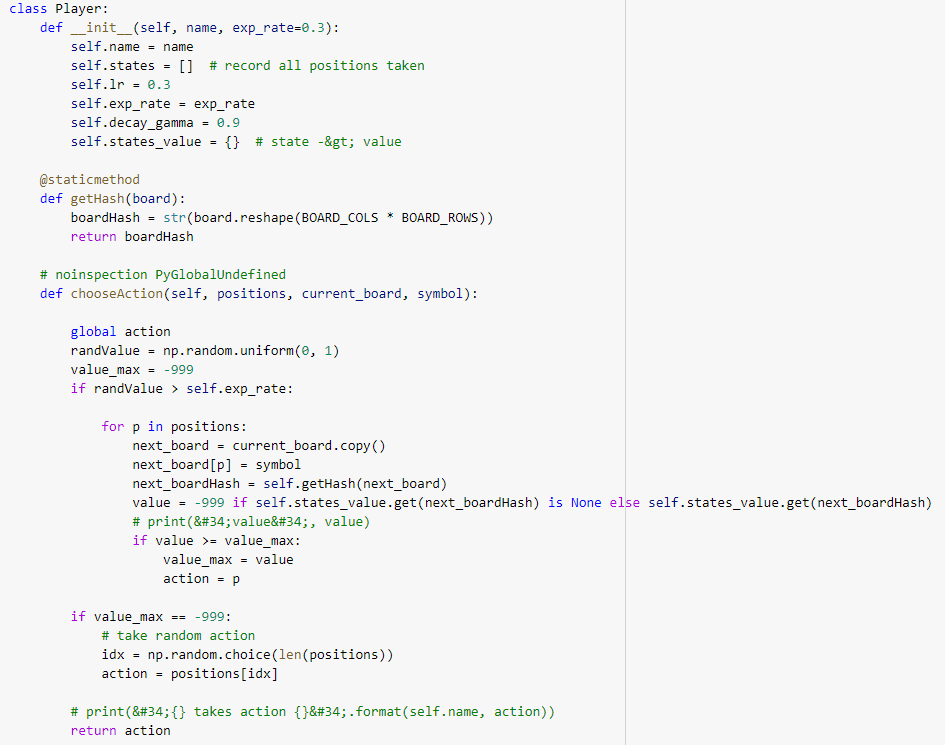
Hình 1.8: Hàm play1 để đi vào trường hợp người đánh với máy (Người đi đầu).



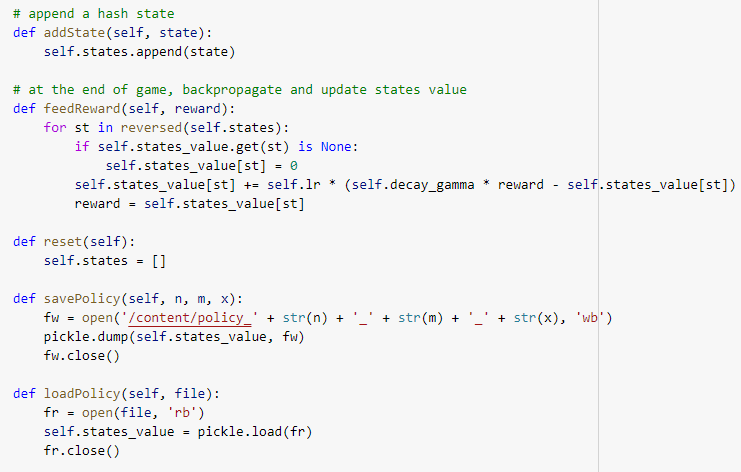
Hình 1.9: Hàm play2 để đi vào trường hợp người đánh với máy (Người đi sau).



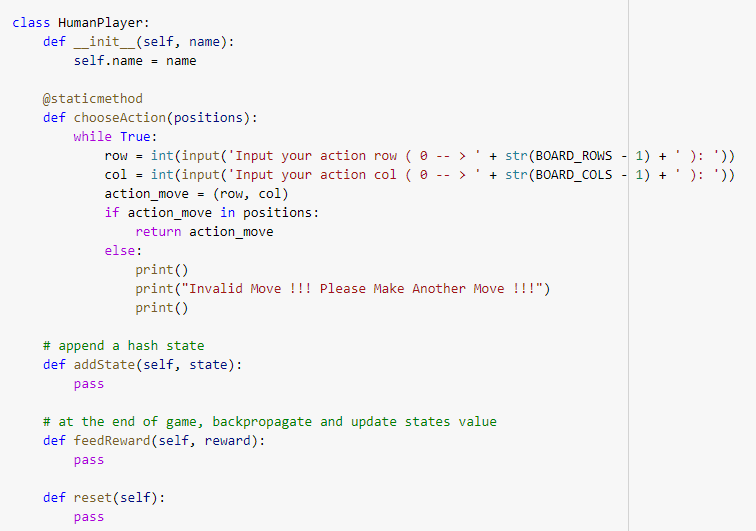
Hình 1.10: Hàm showBoard để vẽ bàn cờ và hiển thị bàn cờ cùng các trạng thái đã thực hiện trên nó.



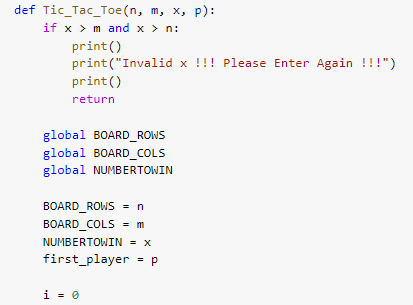
Hình 1.11: Hàm để khởi tạo các đối tượng của class Player, hàm để lưu lại trạng thái bàn cờ dưới dạng mảng 1 chiều và hàm chọn hành động đặt cờ cho máy.



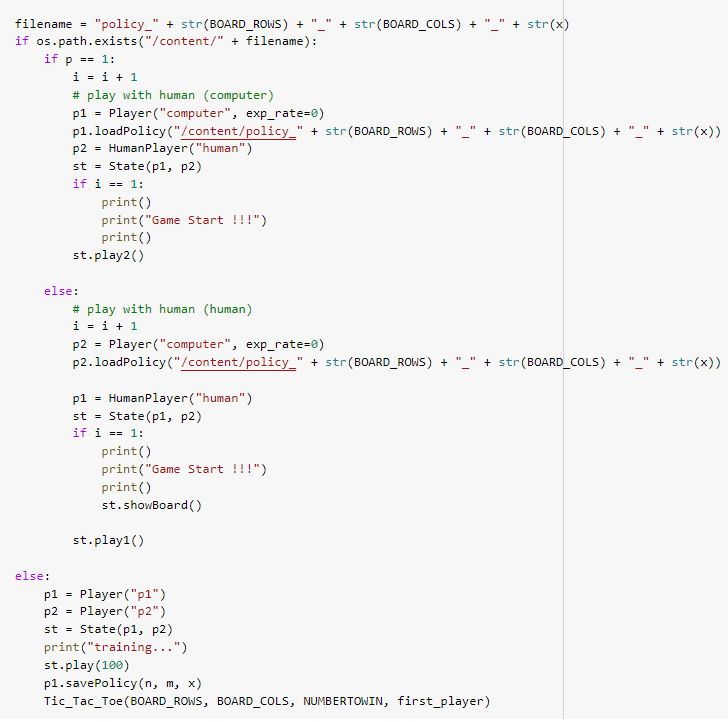
Hình 1.12: Hàm để lưu lại trạng thái bàn cờ trong quá trình huấn luyện, hàm để cập nhật các trọng số cho các trường hợp khi kết thúc ván cờ, hàm reset lại ván cờ, hàm lưu giữ các trạng thái đã được huấn luyện và hàm để lấy ra các trạng thái huấn luyện để đánh với người chơi.



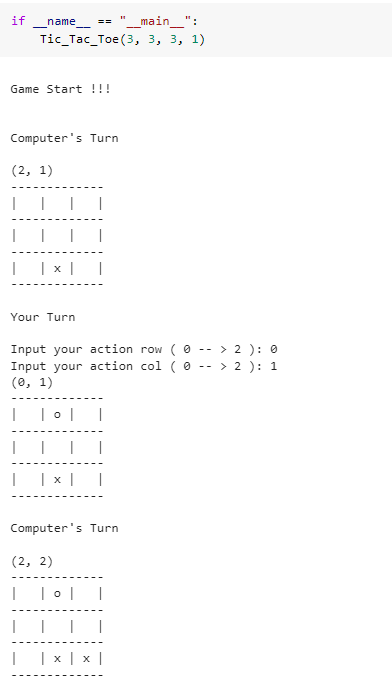
Hình 1.13: Hàm khởi tạo đối tượng người chơi, hàm chọn vị trí đánh dựa vào vị trí hàng và cột.



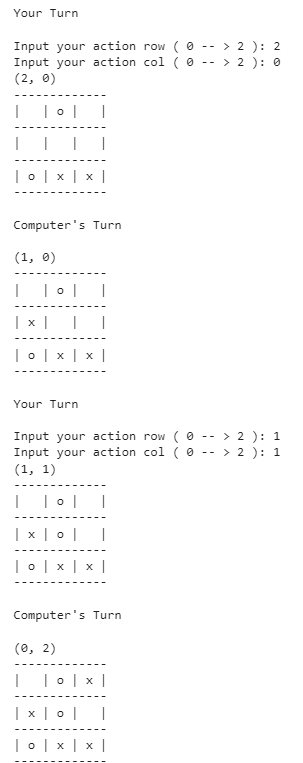
Hình 1.14: Hàm Tic\_Tac\_Toe, thực hiện khởi tạo bàn cờ, khởi tạo đối tượng chơi và kiểm tra các vị trí đánh hợp lý, đồng thời save và load policy phù hợp (1).



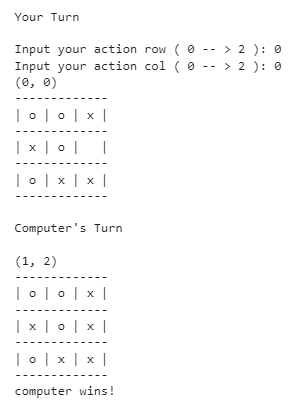
Hình 1.15: Hàm Tic\_Tac\_Toe, thực hiện khởi tạo bàn cờ, khởi tạo đối tượng chơi và kiểm tra các vị trí đánh hợp lý, đồng thời save và load policy phù hợp (2).



Hình 1.16: Quá trình người đánh với máy (Máy đi trước) (1).



Hình 1.17: Quá trình người đánh với máy (Máy đi trước) (2).



Hình 1.18: Quá trình người đánh với máy (Máy đi trước) (3).

1.4 Kết quả và thảo luận

1.4.1 Trình bày kết quả đạt được

Kết quả đạt được là tạo thành công 1 chương trình chơi tic tac toe giữa người với máy. Kết quả tốt hay xấu phụ thuộc vào số lần mà mô hình được huấn luyện. Nếu số lần huấn luyện cao thì sẽ cho ra kết quả tốt hơn nhưng thời gian mà mô hình tiêu tốn lại nhiều hơn.

1.4.2 Thảo luận về kết quả và giải thuật đã hiện thực

Kết quả này tương đối ổn, tuy nhiên do giới hạn về thời gian nên số lần huấn luyện được thực hiện hơi ít dẫn tới mô hình chưa thật sự tối ưu.

Giải thuật này nhìn chung là phù hợp cho bài toán lập trình trò chơi tic tac toe vì bài toán này chỉ cần khả năng vét cạn các trường hợp cũng như việc tự học trong một môi trường vì trong mỗi trận đấu mỗi người sẽ có các cách đánh khác nhau, tuy nhiên việc vét cạn sẽ trở nên khó khăn khi thực hiện trên một bàn cờ lớn, có nhiều nước đi phức tạp. Đồng thời, mỗi lần nhập các tham số đầu vào mới mà chưa có policy thì ta phải tốn thời gian huấn luyện để sinh ra policy mới.

CHƯƠNG 2

Tìm hiểu cơ bản về thị giác máy tính với dữ liệu MNIST nổi tiếng

2.1 Giới thiệu về bài toán

Đây là bài toán nhận dạng chữ số mà các chữ số thuộc trong bộ dữ liệu MNIST. Như vậy ở đây sẽ phải hiểu được rằng MNIST là gì ? Thì MNIST hay còn gọi lại MNIST database, viết một cách đầy đủ là Modified National Institute of Standards and Technology database, nó là một hệ thống cơ sở dữ liệu lớn chứa các chữ số viết tay và thường được áp dụng vào huấn luyện các hệ thống xử lý hình ảnh, ngoài ra thì chính bộ dữ liệu này cũng là một nguồn tài nguyên vô cùng phổ biến trong việc dùng để huấn luyện và kiểm thử trong lĩnh vực học máy.

Do bài toán này được thực hiện và lấy từ Kaggle cho nên mục đích chính của bài báo cáo bài toán này là nói về việc cách mà từng mô hình khác nhau thực hiện xác định một cách chính xác các chữ số từ bộ dữ liệu lớn chứa hàng chục nghìn hình ảnh chữ viết tay, tùy chọn mô hình phù hợp với các kernel để dễ dàng xử lý từng pixel của ảnh, lựa chọn nó là regression hoặc đi theo dạng neural network chẳng hạn như các mô hình neural network, hoặc các mô hình phân loại nhãn như SVM và K-nearest neighbors, lựa chọn xây dựng các tầng và các tham số thích hợp để từ đó tìm ra được đâu là thuật toán, đâu là mô hình tối ưu nhất cho bài toán này.

2.2 Phân tích yêu cầu của bài toán

2.2.1 Yêu cầu của bài toán

Xác định chính xác các chữ số từ bộ dữ liệu gồm hàng chục nghìn hình ảnh viết tay bằng nhiều thuật toán khác nhau và so sánh các thuật toán ấy.

Sử dụng mô hình để gắn nhãn cho tập test.csv

2.2.2 Các phương pháp giải quyết bài toán

2.2.2.1 Phương pháp K-nearest neighbors:

Mô hình K-nearest neighbors là một trong những mô hình có thuật toán đơn giản nhất và nó là loại học có giám sát. Mô hình này thường được dùng trong khai phá dữ liệu và học máy. Về bản chất thì K-nearest neighbors còn được gọi là “lazy learning” vì nó hoàn toàn không học được gì từ dữ liệu mới mà thực hiện tính toán mới khi cần dự đoán một nhãn của dữ liệu mới.

Như vậy, mô hình này trên thực tế là tính toán khoảng cách giữa các điểm trong tập dữ liệu với một dữ liệu đầu vào mới, chẳng hạn như ta đưa vào tham số k = 3, nghĩa là khi dữ liệu cần dự đoán nhãn được đưa vào thì thuật toán sẽ tính toán và lựa ra 3 điểm có khoảng cách gần với dữ liệu được dự đoán, nếu 2 trong số điểm đó chứa chung một kiểu nhãn, điểm còn lại là một nhãn khác thì dữ liệu được dự đoán sẽ lấy nhãn theo số đông, nghĩa là lấy nhãn theo 2 điểm còn lại.

Bài báo dùng phương pháp K-nearest neighbors được lấy để dựa theo và áp dụng vào bài báo cáo là bài báo tên: “K-Nearest Neighbors on MNIST dataset” của tác giả WALID MOUROU được viết vào năm 2019 (K-Nearest Neighbors on MNIST dataset, WALID MOUROU, 2019) [3].

Trong bài báo cáo này, mô hình K-nearest neighbors được sử dụng từ thư viện sklearn với k được dùng là 3. Thuật toán huấn luyện mô hình cho ra xác suất accuracy là 97%.

Tuy nhiên thì hạn chế của mô hình và thuật toán này chính là việc phải lựa chọn tham số k cho thật phù hợp, bởi nếu k không khớp với dữ liệu mô hình thì sẽ dễ dẫn tới kết quả thiếu chính xác, ngoài ra thì việc tính toán các dữ liệu cũng mất rất nhiều thời gian do lúc nào cũng phải thực hiện tính toán lại khoảng cách của các dữ liệu mới. [1]

2.2.2.2 Phương pháp SVM:

SVM hay tên đầy đủ là Support Vector Machine, nó là một thuật toán giám sát và có thể sử dụng cho cả hai bài toán là phân loại hoặc đệ quy. Nhưng phần lớn thì được dùng cho bài toán phân loại.

Nói một cách đơn giản dễ hiểu thì thuật toán này sẽ chia ra một đường thẳng (biên giới) để phân loại ra các phần dữ liệu với các nhãn chung khác nhau tương ứng. Và đường biên giới này được gọi là Hyperplane.

Việc vẽ ra được một Hyper-plane sẽ được dựa trên tiêu chí chủ yếu sau đâu: Trước tiên là trong các Hyperplane được vẽ ra thì thuật toán sẽ chọn ra Hyperplane phân chia được các lớp tốt nhất. Thứ hai là dựa trên “Margin” để mà lựa chọn, Margin ở đây chính là khoảng cách siêu phẳng đến 2 hoặc n điểm dữ liệu gần nhất tương ứng cho các lớp. Như vậy thì SVM sẽ luôn ưu tiên cực đại hóa Margin này để giảm thiểu việc phân lớp sai.

Ngoài ra SVM còn có một khả năng gọi là kernel trick, phương pháp này sẽ biến đổi các dữ liệu phức tạp, từ đó tìm ra cách phân tách dữ liệu dựa trên nhãn mà chúng ta đã xác định trước.

Bài báo được sử dụng để áp dụng và thực hiện trong bài báo cáo này là bài báo có tiêu đề: “MNIST Digit recognition using SVM” của tác giả: “NISHAN PATEL” được viết vào nằm 2018 (MNIST Digit recognition using SVM, NISHAN PATEL, 2018). [4]

Trong bài báo cáo này, mô hình svm được sử dụng từ thư viện sklearn. Thuật toán huấn luyện mô hình cho ra xác suất accuracy là 97%.

Tuy nhiên thì hạn chế của mô hình và thuật toán này chính là việc phải lựa chọn số chiều sao cho phù hợp để mô hình đưa ra kết quả chính xác, ngoài ra SVM hoạt động theo kiểu vẽ ra biên giới để chia ra các lớp, như vậy bản chất của việc chia ra xác suất từng phần hay một đối tượng thuộc vào một lớp là chưa rõ hoặc không có, tuy là vậy nhưng nhờ có Margin nên chúng ta có thể đánh giá hiệu quả mô hình thông qua kỹ thuật này. [2]

2.2.2.3 Phương pháp CNN (Convolutional Neural Network):

CNN(Mạng nơ-ron tích chập) là một trong những mô hình học sâu tiên tiến, được sử dụng nhiều trong các bài toán nhận dạng các đối tượng trong ảnh[5].

Khái niệm về Convolutional: Là việc nhân ma trận đầu vào với ma trận Filter để tạo ra ma trận Convolved Feature nhằm trích xuất đặc trưng ảnh.

* Filter hay còn gọi là kernel là một ma trận có kích thước thường nhỏ hơn ma trận đầu vào. Các tham số trong ma trận ấy sẽ được máy tính tự học.
* Ví dụ: Ta có ma trận đầu vào là và kernel là thì ma trận Convolved Feature là [5].

Cấu trúc mạng CNN:

* CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo[5].
* Lớp tiếp theo trong mạng là kết quả convolution từ lớp trước đó có được các kết nối cục bộ. Kết luận, mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó.
* Do có quá nhiều filter nên cần sử dụng vài lớp như pooling hoặc subsampling để chọn ra các thông tin hữu ích hơn.
* Trong quá trình huấn luyện mạng, CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà bạn thực hiện[5].
* Với cùng một đối tượng, khi ta quan sát ở các góc độ khác nhau thì sẽ cho ra kết quả khác nhau. Lớp Pooling có tính bất biến khi xem xét 1 đối tượng với phép dịch chuyển, phép quay và phép co giãn. Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua convolution từ các filter[5]. Do đó mô hình CNN có độ chính xác rất cao.

Bài báo sử dụng phương pháp CNN:

* Tiêu đề bài báo: Study and Develop a ConvolutionalNeural Network for MNIST HandwrittenDigit Classiﬁcation
* Tác giả: Disha Jayswal, Brijeshkumar Y. Panchal, Bansari Patel, Nidhi Acharya,Rikin Nayak, and Parth Goe
* Năm: 2022
* Phương pháp giải quyết: vào tháng 6 năm 2021, nhóm tác giả sử dụng các phương pháp là Decision tree, logistic regression, k-nearest neighbors (KNN),CNN cho bài toán so sánh, phân tích các thuật toán được sử dụng trong xác định chữ số viết tay
* Dữ liệu thực nghiệm: Bộ dữ liệu MNIST, được xuất bản bởi Y. Nó bao gồm 60.000 ô vuông có thang độ xám 28x28 pixel thể hiện các số viết tay từ 0 đến 9. Bộ dữ liệu này chứa 60.000 hình ảnh đào tạo và 10.000 hình ảnh thử nghiệm. Đối với bộ thử nghiệm, hơn 250 người viết khác nhau đã được chọn cho các mẫu dữ liệu viết tay. Hình sau thể hiện tập dữ liệu MNIST



Hình 2.1: Tập dữ liệu MNIST.

* Kết quả đạt được: vào tháng 6 năm 2021, độ chính xác (Accuracy) của từng thuật toán là:
  + Decision tree: 86.6%
  + Logistic regression: 92.6%
  + k-nearest neighbors (KNN): 96.89%
  + CNN: 99%
* Kết luận: CNN hoạt động tốt hơn nhiều thuật toán khác vì nó cung cấp hiệu quả tính toán cực lớn đồng thời phát hiện các đặc điểm quan trọng mà không cần sự can thiệp của con người. CNN là một kiến trúc đặc biệt để phát hiện các tính năng phức tạp trong dữ liệu mang lại sự thuận tiện để thực hiện nhận dạng. Hơn nữa, nó cũng có một trong những tính năng độc đáo là kết hợp các lớp tích chập và tổng hợp khác nhau giúp cải thiện độ chính xác. Các mô hình CNN giờ đây có thể chạy trên bất kỳ thiết bị nào, khiến chúng trở nên hấp dẫn trên toàn cầu bằng cách kết hợp hai lớp tích chập và tổng hợp từng lớp [6].

2.2.3 Phương pháp đề xuất giải quyết bài toán

Hai phương pháp được lựa chọn để giải quyết bài toán này là KNN và CNN. Lý do chọn KNN vì đây là mô hình học có giám sát mà dữ liệu là kiểu số nguyên và có nhãn nên phù hợp với mô hình này trong việc quan sát và đánh giá. Còn đối với mô hình học sâu CNN, mô hình này mang các chức năng đặc thù trong việc xử lý ảnh vì nó có khả năng tính toán tự động qua các tầng dựa trên các tham số bộ lọc mà mô hình cung cấp để đưa ra nhận dạng hình ảnh có độ chính xác cao.

2.3 Phương pháp giải quyết bài toán

2.3.1 Mô hình tổng quát

2.3.1.1 Mô hình KNN

Kiểm tra dữ liệu null

Chuẩn hóa dữ liệu

Chia tập train/test

Đưa dữ liệu cho mô hình học

Xây dựng mô hình KNN

Đánh giá mô hình

Đọc dữ liệu

Xem thông tin dữ liệu

2.3.1.2 Mô hình CNN

Đưa dữ liệu cho mô hình học

Đánh giá mô hình

Chia tập train/test

Xây dựng mô hình CNN

Kiểm tra dữ liệu null

Chuẩn hóa dữ liệu

Đọc dữ liệu

Xem thông tin dữ liệu

Điều chỉnh chiều dữ liệu thành ma trận 28x28

Tối ưu mô hình

Lưu mô hình

Dùng mô hình gắn nhãn cho file test.csv

2.3.2 Đặc trưng của mô hình đề xuất

2.3.2.1 Xử lý dữ liệu chung cho 2 mô hình

Đọc dữ liệu bằng thư viện pandas của python

Xem thông tin dữ liệu bằng hàm info()

Kiểm tra dữ liệu null bằng hàm isnull()

Chuẩn hóa dữ liệu bằng cách chia các giá trị cột pixel cho 255

2.3.2.2 Mô hình KNN

Chia tập train/test theo tỉ lệ 8:2 bằng hàm train\_test\_split() với random\_state = 1

Xây dựng mô hình KNN với số lượng neighbor là 3

Huấn luyện mô hình với dữ liệu rồi đánh giá kết quả bằng accuracy.

2.3.2.3 Mô hình CNN

Dùng hàm reshape() để điều chỉnh lại dữ liệu thành ma trận 28x28

Chia tập train/test theo tỉ lệ 8:2 bằng hàm train\_test\_split() với random\_state = 1

Xây dựng mô hình CNN với các lớp sau:

* Lớp tích chập Conv2D thứ nhất có 16 bộ lọc, ma trận kernel là 5x5, đầu vào kích thước là (28,28,1), hàm kích hoạt là relu(hàm này sẽ trả kết quả bằng 0 nếu kết quả là số âm và giữ nguyên kết quả nếu nó lớn hơn hoặc bằng 0)
* Lớp tích chập Conv2D thứ hai có 32 bộ lọc, ma trận kernel là 5x5, hàm kích hoạt là relu
* Lớp tích hợp cực đại Max Pooling nén kết quả lớp trước lại thành ma trận 2x2 để đặc trưng nổi bật hơn
* Hàm Dropout(0.3) giúp bỏ bớt 30% lượng bộ lọc dư thừa
* Lớp tích chập Conv2D thứ ba có 64 bộ lọc, ma trận kernel là 3x3, hàm kích hoạt là relu
* Lớp tích hợp cực đại Max Pooling nén kết quả lớp trước lại thành ma trận 2x2 để đặc trưng nổi bật hơn
* Hàm Dropout(0.3) giúp bỏ bớt 30% lượng bộ lọc dư thừa
* Dùng Flatten để làm phẳng kết quả của lớp trước
* Tạo 1 tầng Dense có 128 neuron để tính toán giá trị, hàm kích hoạt là relu
* Hàm Dropout(0.3) giúp bỏ bớt 30% lượng neuron dư thừa
* Tầng Dense cuối có 10 neuron để tính toán giá trị xác suất của 10 nhãn(từ 0 đến 9), hàm kích hoạt là softmax(hàm này sẽ lấy giá trị xác suất cao nhất của một trong 10 nhãn làm kết quả đầu ra)

Tối ưu mô hình bằng hàm RMSprop và hàm loss là categorical\_crossentropy

Tránh overfitting bằng cách tạo thêm dữ liệu bằng ImageDataGenerator

Huấn luyện mô hình với dữ liệu rồi đánh giá kết quả bằng accuracy.

Cuối cùng là dùng thư viện pickle để tại mô hình về và load lên để gán nhãn cho dữ liệu trong file test.csv

2.4 Thực nghiệm

2.4.1 Dữ liệu

Dữ liệu được lấy trong phần data của cuộc thi “Digit Recognizer” trên trang Kaggle. Nguồn: <https://www.kaggle.com/competitions/digit-recognizer/data>

Dữ liệu gồm hai file là train.csv và test.csv

* File train.csv dùng để huấn luyện và đánh giá mô hình. Nó gồm 785 cột với cột đầu tiên là label(giá trị từ 0 đến 9). Còn các cột còn lại là giá trị pixel của ảnh (từ 0 đến 255)
* File test.csv gồm 784 cột là giá trị pixel của ảnh (từ 0 đến 255). Ta sẽ dùng mô hình huấn luyện được để gắn nhãn cho các dữ liệu trong file này.

2.4.2 Xử lý dữ liệu

Giá trị của các pixel cần được chuẩn hóa sao cho nằm trong đoạn [0,1] cho việc tính toán nhẹ nhàng hơn và giảm nhiễu bằng cách chia chúng cho 255

Dữ liệu cần được chỉnh lại thành ma trận 28x28 để trích xuất đặc trưng được chính xác. Sử dụng hàm reshape để làm điều này.

2.4.3 Công nghệ sử dụng

Ngôn ngữ lập trình sử dụng là Python, và dùng Google Colab để viết chương trình.

Các thư viện được sử dụng : numpy, pandas, keras, matplotlib, sklearn, pickle

2.4.4 Cách đánh giá

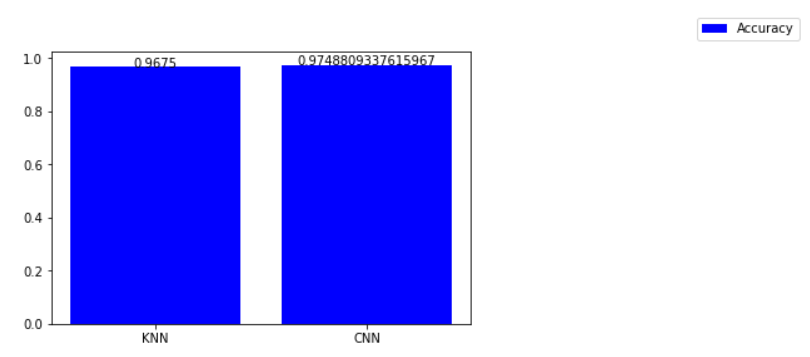
Độ đo accuracy được tính bằng công thức tỉ lệ những điểm dự đoán đúng chia cho tổng số điểm được dự đoán

2.5 Kết quả đạt được

Tham số **c**ụ thể hóa các thực nghiệm:

* KNN:
  + Số neighbors (k) = 3
* CNN:
  + Số epoch: 1
  + Số tầng để trích xuất đặc trưng: 7
  + Số tầng để tính toán trong mạng neuron network: 3

Accuracy của KNN và CNN lần lượt là 0.9675 và 0.9749.



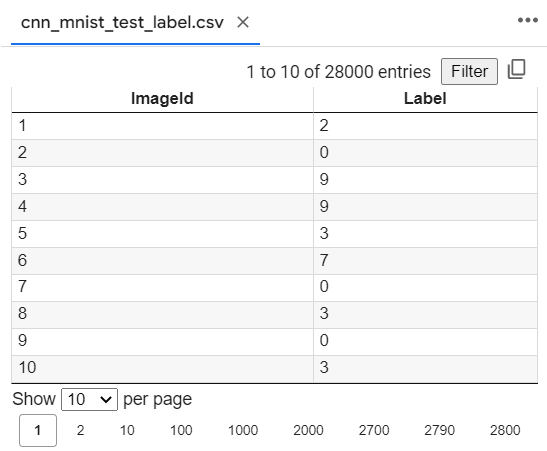
Hình 2.2: Biểu đồ thể hiện accuracy của 2 mô hình KNN và CNN.

Sự khác nhau giữa các kết quả của KNN và CNN là do mỗi mô hình có cách tính toán khác nhau. Trong khi KNN thì phân lớp theo nhóm k dựa trên việc tính trung bình thì CNN lại trích xuất đặc trưng qua các tầng, mỗi tầng lại sẽ có các tham số riêng do người dùng tự chọn sao cho phù hợp.

2.6 Kết luận

2.6.1 Trình bày tóm tắt các kết quả đạt được của bài toán trên:

Hình dưới cho ta thấy được kết quả dự đoán các dữ liệu từ tập test của mô hình:



Hình 2.3: Kết quả gán nhãn cho các dữ liệu của file test.csv.

2.6.2 Hạn chế:

KNN: không trích xuất được đặc trưng ảnh mà chỉ dựa vào giá trị pixel. Việc chọn số lượng neighbor là ngẫu nhiên.

CNN: Mô hình xây dựng có nhiều tham số phức tạp, việc chọn lựa các tham số cũng như các tầng cho mô hình cũng phải thật sự hợp lý và phù hợp thì đầu ra dữ liệu mới thật sự đúng và chính xác như mong muốn. Tuy nhiên thì việc xây dựng mô hình với các cấu trúc phức tạp sẽ tốn rất nhiều thời gian trong việc huấn luyện cũng như phải tiền xử lý dữ liệu sao cho phù hợp với đầu vào của mô hình.

2.6.3 Hướng phát triển trong tương lai:

Để cải thiện độ chính xác của mô hình, nhóm em cần phải tìm hiểu thêm về các tham số khác trong mô hình CNN và cách kết hợp với nhiều mô hình học sâu khác thì bài toán sẽ có hiệu xuất cao hơn.

TÀI LIỆU THAM KHẢO

1. Nguyen Thi Hop (2019), KNN (K-Nearest Neighbors) #1, <https://viblo.asia/p/knn-k-nearest-neighbors-1-djeZ14ejKWz>

2. Huynh Chi Trung (2020), Giới thiệu về Support Vector Machine (SVM), <https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB>

3. WALID MOUROU (2019), K-Nearest Neighbors on MNIST dataset, <https://www.kaggle.com/code/walidmourou/k-nearest-neighbors-on-mnist-dataset/notebook>

4. NISHAN PATEL (2018), MNIST Digit recognition using SVM, <https://www.kaggle.com/code/nishan192/mnist-digit-recognition-using-svm>

5. Top Dev Blog, “Thuật toán CNN là gì? Cấu trúc mạng Convolutional Neural Network”, <https://topdev.vn/blog/thuat-toan-cnn-convolutional-neural-network/#:~:text=5%20b%C3%AAn%20tr%C3%A1i.-,C%E1%BA%A5u%20tr%C3%BAc%20m%E1%BA%A1ng%20CNN,cho%20c%C3%A1c%20l%E1%BB%9Bp%20ti%E1%BA%BFp%20theo>.

6. Disha Jayswal, Brijeshkumar Y. Panchal, Bansari Patel, Nidhi Acharya,Rikin Nayak, and Parth Goe (2022),”Study and Develop a Convolutional Neural Network for MNIST Handwritten Digit Classification”, <https://www.researchgate.net/publication/361728427_Study_and_Develop_a_Convolutional_Neural_Network_for_MNIST_Handwritten_Digit_Classification>

7. Phạm Duy Tùng (2020), “Reinforcement Learning và tictactoe”

<https://www.phamduytung.com/blog/2020-12-26---tic-tac-toe/>

8. Bùi Thanh Hùng(2022), “Project 20”, khoa Công nghệ thông tin trường Đại học Tôn Đức Thắng

9. Yassine Ghouzam(2017), “Introduction to CNN Keras - 0.997 (top 6%)”

<https://www.kaggle.com/code/yassineghouzam/introduction-to-cnn-keras-0-997-top->6

**TỰ ĐÁNH GIÁ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Câu | Nội dung | Điểm chuẩn | Tự chấm | Ghi chú |
| 1  (4.0) | **1.1 Mô tả cấu trúc dữ liệu** | 0.5 | 0.5 |  |
| **1.2 Sơ đồ giải thuật** | 1 | 1 |  |
| **1.3 Hiện thực** | 2 | 2 |  |
| **1.4 Kết quả và thảo luận** | 0.5 | 0.25 | Không có cách đánh giá bài toán |
| 2  (4.5) | **2.1 Giới thiệu về bài toán** | 0.25 | 0.25 |  |
| **2.2 Phân tích yêu cầu của bài toán** | 1 | 1 |  |
| **2.3 Phương pháp giải quyết bài toán** | 1.25 | 1.25 |  |
| **2.4 Thực nghiệm** | 1 | 0.5 | KNN không có overfitting, chỉ có 1 cách đánh giá |
| **2.5 Kết quả đạt được** | 0.75 | 0.5 | Chỉ có 1 cách đánh giá |
| **2.6 Kết luận** | 0.25 | 0.25 |  |
| 3 | **Điểm nhóm** | 0.5đ | 0.5 |  |
| 4 | **Báo cáo (**chú ý các chú ý 2,3,4,6 ở trang trước, nếu sai sẽ bị trừ điểm nặng**)** | 1đ | 1 |  |
| **Tổng điểm** | | | 9 |  |