**Trình bày về mạng Hopfield**

(Người thực hiện: Đỗ Văn Thiện)

Mở đầu

Tài liệu giới thiệu về mô hình mạng Hopfiled

**1. Mạng Neuron HopField:**

Mạng nơ-ron Hopfield được phát minh bởi Tiến sĩ John J. Hopfield vào năm 1982. Nó bao gồm một lớp duy nhất chứa một hoặc nhiều nơ-ron tái phát được kết nối đầy đủ. Mạng Hopfield thường được sử dụng cho các nhiệm vụ tự động liên kết và tối ưu hóa.

Mạng Hopfield đóng vai trò là hệ thống [bộ nhớ](https://en.wikipedia.org/wiki/Content-addressable_memory) có [thể định địa chỉ](https://en.wikipedia.org/wiki/Content-addressable_memory) theo [nội dung ("liên kết")](https://en.wikipedia.org/wiki/Content-addressable_memory) với [các nút](https://en.wikipedia.org/wiki/Artificial_neuron) ngưỡng [nhị phân](https://en.wikipedia.org/wiki/Binary_numeral_system). Mạng Hopfield cũng cung cấp một mô hình để hiểu trí nhớ của con người.

[Mô hình Ising](https://en.wikipedia.org/wiki/Ising_model) của một mạng nơron như một mô hình bộ nhớ được [William A. Little](https://en.wikipedia.org/w/index.php?title=William_A._Little&action=edit&redlink=1) đề xuất lần đầu tiên vào năm 1974, được Hopfield thừa nhận trong bài báo năm 1982 của ông.

**2. Cấu trúc:**

a. Cấu trúc:

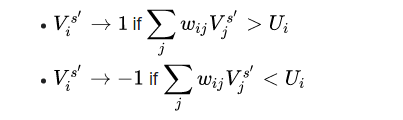
Các đơn vị trong lưới Hopfield là các đơn vị ngưỡng nhị phân, tức là các đơn vị chỉ nhận hai giá trị khác nhau cho các trạng thái của chúng và giá trị được xác định bằng cách liệu đầu vào của đơn vị có vượt quá ngưỡng của nó hay không. Các mạng Hopfield rời rạc mô tả mối quan hệ giữa các nơron nhị phân (kích hoạt hoặc không kích hoạt){\ displaystyle 1,2, ... i, j, ... N}. Tại một thời điểm nhất định, trạng thái của mạng nơron được mô tả bằng một vectơ{\ displaystyle V}, ghi lại những tế bào thần kinh nào đang kích hoạt trong một từ nhị phân gồm N bit.

Các tương tác wij {\ displaystyle w\_ {ij}}giữa các nơron có các đơn vị thường nhận giá trị 1 hoặc -1, tuy nhiên, các tài liệu khác có thể sử dụng các đơn vị nhận giá trị 0 và 1. Những tương tác này được "học" thông qua luật kết hợp của Hebb, như vậy, đối với một trạng thái nhất định v8{\ displaystyle V ^ {s}} : wij = vi8vj8{\ displaystyle w\_ {ij} = V\_ {i} ^ {s} V\_ {j} ^ {s}}

nhưng wij = 0

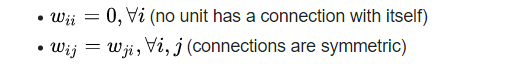
\* Quy tắc Hebbian learning{\ displaystyle w\_ {ii} = 0}: wij = (2vi8-1) (2vj8-1)  khi các đơn vị giả định các giá trị trong {0, 1}

Sau khi mạng được đào tạo, {\ displaystyle w\_ {ij}}wij không còn tiến hóa. Nếu một trạng thái mới của tế bào thần kinh v8’{\ displaystyle V ^ {s '}} được đưa vào mạng nơ-ron, mạng hoạt động trên các nơ-ron sao cho



Bằng cách này, mạng Hopfield có khả năng "ghi nhớ" các trạng thái được lưu trữ trong ma trận tương tác, bởi vì nếu một trạng thái mới v8’ {\ displaystyle V ^ {s '}} chịu sự tác động của ma trận tương tác, mỗi nơron sẽ thay đổi cho đến khi khớp với trạng thái ban đầu v8

Các kết nối trong mạng Hopfield thường có các hạn chế sau:

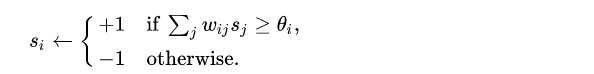


b. Ràng buộc :

Các trọng số là đối xứng đảm bảo rằng hàm năng lượng giảm đơn điệu trong khi tuân theo các quy tắc kích hoạt. Một mạng có trọng số không đối xứng có thể biểu hiện một số hành vi tuần hoàn hoặc hỗn loạn; tuy nhiên, Hopfield nhận thấy rằng hành vi này được giới hạn trong các phần tương đối nhỏ của không gian pha và không làm suy giảm khả năng của mạng hoạt động như một hệ thống bộ nhớ liên kết có địa chỉ theo nội dung.

Hopfield cũng lập mô hình mạng nơ-ron cho các giá trị liên tục, trong đó sản lượng điện của mỗi nơ-ron không phải là hệ nhị phân mà là một số giá trị từ 0 đến 1. Ông phát hiện ra rằng loại mạng này cũng có thể lưu trữ và tái tạo các trạng thái đã ghi nhớ.

Cập nhật một đơn vị (nút trong biểu đồ mô phỏng nơron nhân tạo) trong mạng Hopfield được thực hiện theo quy tắc sau:

* wij là cường độ của trọng lượng kết nối từ đơn vị j đến đơn vị i (trọng lượng của kết nối).
* {\ displaystyle s\_ {i}}Sj là trạng thái của đơn vị i.
* {\ displaystyle \ theta \_ {i}}θi: là ngưỡng của đơn vị i.

c. Cập nhật:

- Các cập nhật trong mạng Hopfield có thể được thực hiện theo hai cách khác nhau:

* **Không đồng bộ** : Chỉ một đơn vị được cập nhật tại một thời điểm. Đơn vị này có thể được chọn ngẫu nhiên, hoặc một thứ tự xác định trước có thể được áp đặt ngay từ đầu.
* **Đồng bộ** : Tất cả các đơn vị được cập nhật cùng một lúc. Điều này yêu cầu một đồng hồ trung tâm cho hệ thống để duy trì sự đồng bộ hóa. Phương pháp này được một số người coi là kém thực tế hơn, dựa trên sự vắng mặt của đồng hồ toàn cầu quan sát được ảnh hưởng đến các hệ thống sinh học hoặc vật lý tương tự được quan tâm.

### - Tế bào thần kinh "hút hoặc đẩy nhau" trong không gian trạng thái

Trọng lượng giữa hai đơn vị có tác động mạnh mẽ đến giá trị của các tế bào thần kinh. Cân nhắc trọng lượng kết nối wij giữa hai nơron i và j. Nếu wij > 0, quy tắc là:

+ khi nào Sj = 1,  đóng góp của j trong tổng trọng số là dương nên Sj được j kéo về phía Sj = 1{\ displaystyle s\_ {j} = 1}

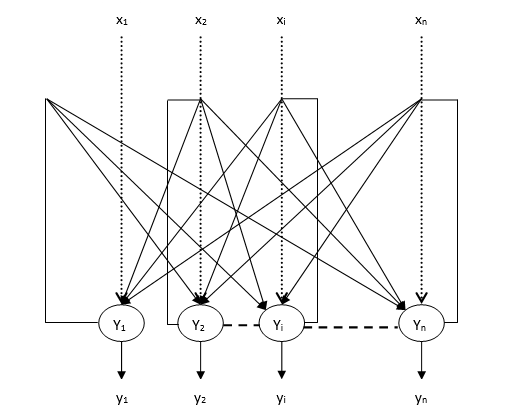
+ khi nào Sj = -1,  đóng góp của j trong tổng trọng số là âm nên sau đó Sj được j kéo về phía Sj = -1{\ displaystyle s\_ {j} = 1}

Do đó, các giá trị của nơron i và j sẽ hội tụ nếu trọng số giữa chúng là dương. Tương tự, chúng sẽ phân kỳ nếu trọng lượng âm.

**3. Mạng Hopfield rời rạc**

Một mạng Hopfield hoạt động theo kiểu đường rời rạc hay nói cách khác, có thể nói các mẫu đầu vào và đầu ra là vector rời rạc, có thể là nhị phân  0 , 1 hoặc lưỡng cực + 1 , - 1trong bản chất. Mạng có trọng số đối xứng không có tự kết nối, tức là **w ij = w ji** và **w ii = 0** .

* Mô hình này bao gồm các tế bào thần kinh với một đầu ra đảo ngược và một đầu ra không đảo ngược.
* Đầu ra của mỗi tế bào thần kinh phải là đầu vào của các tế bào thần kinh khác nhưng không phải là đầu vào của bản thân.
* Trọng lượng / cường độ kết nối được biểu thị bằng **w ij** .
* Các kết nối có thể kích thích cũng như ức chế. Sẽ rất kích thích nếu đầu ra của nơron giống với đầu vào, ngược lại là ức chế.
* Các trọng lượng phải đối xứng, tức là **w ij = w ji**



Đầu ra từ **Y 1** đến **Y 2** , **Y i** và **Y n** có trọng số lần lượt là **w 12** , **w 1i** và **w 1n** . Tương tự, các cung khác có trọng lượng trên chúng.

a. Thuật toán traning:

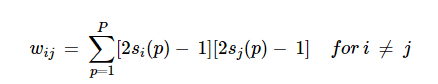
Trong quá trình huấn luyện mạng Hopfield rời rạc, các trọng số sẽ được cập nhật. Chúng ta có thể có các vectơ đầu vào nhị phân cũng như các vectơ đầu vào lưỡng cực. Do đó, trong cả hai trường hợp, cập nhật trọng lượng có thể được thực hiện với mối quan hệ sau

**\* Trường hợp 1** - Mẫu đầu vào nhị phân

Đối với một tập hợp các mẫu nhị phân **s** p**, p = 1 đến P**

Đây, **s** p**= s 1** p**, s 2** p**, ..., si** p**, ..., sn** p

Ma trận trọng lượng được đưa ra bởi

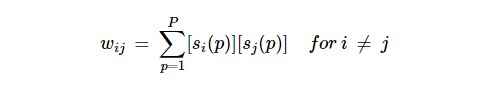


**\* Trường hợp 2** - Mẫu đầu vào lưỡng cực

Đối với một tập hợp các mẫu nhị phân **s** p**, p = 1 đến P**

Đây, **s** p**= s 1** p**, s 2** p**, ..., si** p**, ..., sn** p

Ma trận trọng lượng được đưa ra bởi



b. Thuật toán Testing:

**Bước 1** - Khởi tạo các trọng số thu được từ thuật toán huấn luyện bằng cách sử dụng nguyên tắc Hebbian.

**Bước 2** - Thực hiện các bước 3-9, nếu các kích hoạt của mạng không được hợp nhất.

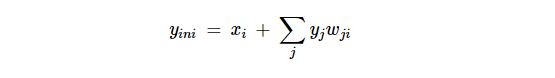
**Bước 3** - Đối với mỗi vectơ đầu vào **X** , thực hiện các bước 4-8.

**Bước 4** - Thực hiện kích hoạt ban đầu của mạng bằng vectơ đầu vào bên ngoài **X** như sau:

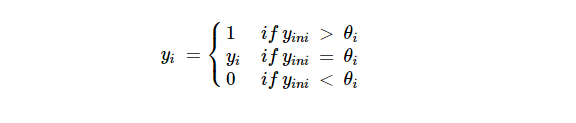
Yi = xi for i = 1 đến n

**Bước 5** - Đối với mỗi đơn vị **Yi** , thực hiện các bước 6-9.

**Bước 6** - Tính toán đầu vào thực của mạng như sau:



**Bước 7** - Áp dụng kích hoạt như sau trên đầu vào ròng để tính toán đầu ra –



θi là ngưỡng.

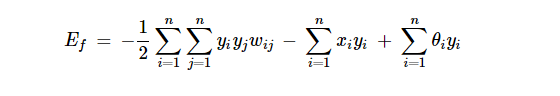
**Bước 8** - Phát đầu ra **yi** đến tất cả các đơn vị khác.

**Bước 9** - Kiểm tra mạng để kết hợp.

### c. Đánh giá chức năng năng lượng

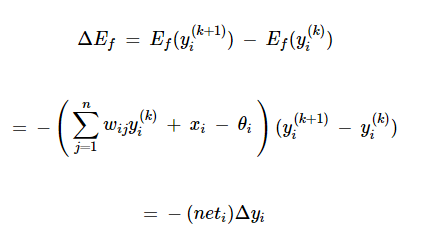
Một chức năng năng lượng được định nghĩa là một chức năng được liên kết và chức năng không tăng của trạng thái của hệ thống.

Hàm năng lượng **Ef,**  còn được gọi là **hàm Lyapunov** xác định độ ổn định của mạng Hopfield rời rạc, và được đặc trưng như sau:



**Điều kiện:** - Trong một mạng ổn định, bất cứ khi nào trạng thái của nút thay đổi, cơ năng trên sẽ giảm.

Giả sử khi nút **tôi** đã thay đổi trạng thái từ yi(k) đến yi(k+1) Khi đó năng lượng thay đổi ΔEf được đưa ra bởi quan hệ sau:



với:

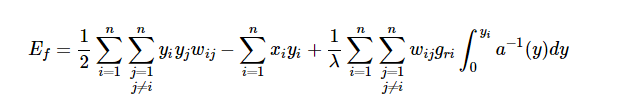


Sự thay đổi năng lượng phụ thuộc vào thực tế là chỉ một đơn vị có thể cập nhật kích hoạt của nó tại một thời điểm.

## 4. Mạng Hopfield liên tục

So với mạng Hopfield rời rạc, mạng liên tục có thời gian là một biến liên tục. Nó cũng được sử dụng trong các vấn đề liên kết và tối ưu hóa như vấn đề nhân viên bán hàng đi du lịch.

**Mô hình** - Mô hình hoặc kiến ​​trúc có thể được xây dựng bằng cách thêm các thành phần điện như bộ khuếch đại có thể ánh xạ điện áp đầu vào với điện áp đầu ra qua một chức năng kích hoạt sigmoid.



Ở đây **λ** là tham số khuếch đại và độ dẫn đầu vào **g**ri

**5. Tối ưu hóa trong mạng Hopfield:**

Tối ưu hóa là một hành động làm cho một cái gì đó như thiết kế, tình huống, tài nguyên và hệ thống trở nên hiệu quả nhất có thể. Sử dụng sự tương đồng giữa hàm chi phí và hàm năng lượng, chúng ta có thể sử dụng các nơ-ron có tính liên kết cao để giải quyết các vấn đề tối ưu hóa. Một loại mạng nơ-ron như vậy là mạng Hopfield, bao gồm một lớp duy nhất chứa một hoặc nhiều nơ-ron lặp lại được kết nối đầy đủ. Điều này có thể được sử dụng để tối ưu hóa.

Những điểm cần nhớ khi sử dụng mạng Hopfield để tối ưu hóa -

* Chức năng năng lượng phải là tối thiểu của mạng.
* Nó sẽ tìm ra giải pháp thỏa đáng hơn là chọn một trong số các mẫu được lưu trữ.
* Chất lượng của giải pháp được tìm thấy bởi mạng Hopfield phụ thuộc đáng kể vào trạng thái ban đầu của mạng.

## \*Bài toán: Vấn đề về nhân viên bán hàng đi du lịch

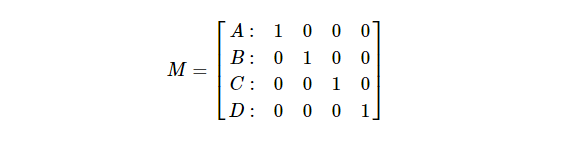
Tìm tuyến đường ngắn nhất mà nhân viên bán hàng đi là một trong những bài toán tính toán, có thể được tối ưu hóa bằng cách sử dụng mạng nơ-ron Hopfield.

### a. Khái niệm cơ bản về TSP

Vấn đề về nhân viên bán hàng đi du lịch TSP là một bài toán tối ưu hóa cổ điển, trong đó một người bán hàng phải đi đến **n** thành phố, các thành phố này được kết nối với nhau, giữ cho chi phí cũng như quãng đường đi được là tối thiểu. Ví dụ: nhân viên bán hàng phải đi một tập hợp 4 thành phố A, B, C, D và mục tiêu là tìm ra chuyến đi vòng quanh ngắn nhất, ABC – D, để giảm thiểu chi phí, cũng bao gồm chi phí đi từ thành phố D cuối cùng đến thành phố A đầu tiên.

### b. Biểu diễn ma trận

Trên thực tế, mỗi chuyến tham quan TSP của n thành phố có thể được biểu diễn dưới dạng ma trận **n × n** mà hàng **thứ i** mô tả vị trí của thành phố **thứ i** . Ma trận này, **M** , cho 4 thành phố A, B, C, D có thể được biểu diễn như sau:



## - Giải pháp của Hopfield Network

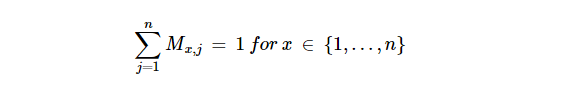
Trong khi xem xét giải pháp của mạng TSP bởi Hopfield này, mọi nút trong mạng tương ứng với một phần tử trong ma trận.

### **+** Tính toán hàm năng lượng

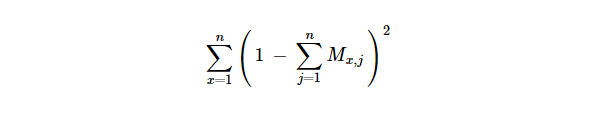
Để trở thành giải pháp tối ưu hóa, hàm năng lượng phải ở mức tối thiểu. Trên cơ sở các ràng buộc sau, chúng ta có thể tính hàm năng lượng như sau

### + Ràng buộc-I

Ràng buộc đầu tiên, trên cơ sở đó chúng ta sẽ tính toán hàm năng lượng, là một phần tử phải bằng 1 trong mỗi hàng của ma trận **M** và các phần tử khác trong mỗi hàng phải bằng **0** vì mỗi thành phố chỉ có thể xảy ra ở một vị trí trong Tham quan TSP. Ràng buộc này có thể được viết về mặt toán học như sau:

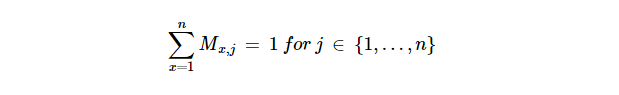


Bây giờ hàm năng lượng được tối thiểu hóa, dựa trên ràng buộc ở trên, sẽ chứa một số hạng tỷ lệ với –

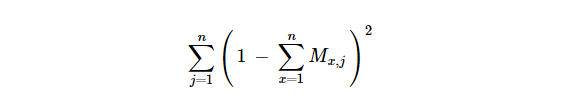


### + Ràng buộc-II

Như chúng ta đã biết, trong TSP, một thành phố có thể xuất hiện ở bất kỳ vị trí nào trong chuyến tham quan do đó trong mỗi cột của ma trận **M** , một phần tử phải bằng 1 và các phần tử khác phải bằng 0. Ràng buộc này có thể được viết về mặt toán học như sau:



Bây giờ hàm năng lượng được tối thiểu hóa, dựa trên ràng buộc ở trên, sẽ chứa một số hạng tỷ lệ với



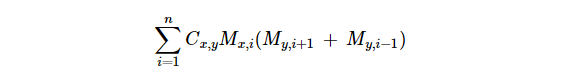
### **+** Tính toán hàm chi phí

Giả sử một ma trận vuông của ( **n × n** ) ký hiệu là **C** biểu thị ma trận chi phí TSP cho **n** thành phố nơi **n> 0** . Sau đây là một số tham số trong khi tính toán hàm chi phí:

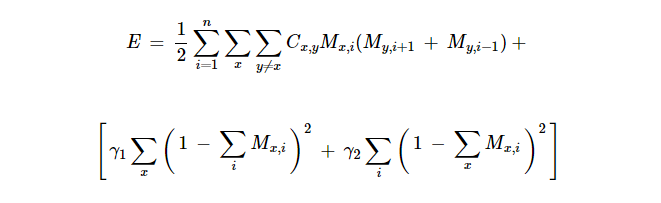
* **C x, y** - Phần tử của ma trận chi phí biểu thị chi phí đi từ thành phố **x** đến **y** .
* Sự liền kề của các phần tử của A và B có thể được thể hiện qua



Trong Ma trận giá trị đầu ra của mỗi nút có thể là 0 hoặc 1, do đó đối với mọi cặp thành phố A, B, chúng ta có thể thêm các số hạng sau vào hàm năng lượng:



Trên cơ sở của hàm chi phí và giá trị ràng buộc ở trên, hàm năng lượng cuối cùng **E** có thể được đưa ra như sau:



## 6. Ứng dụng:

## Tại sao lại sử dụng mạng thần kinh nhân tạo?

Trước khi nghiên cứu các lĩnh vực mà ANN đã được sử dụng rộng rãi, chúng ta cần hiểu tại sao ANN lại là lựa chọn ứng dụng ưu tiên.

Chúng ta cần hiểu câu trả lời cho câu hỏi trên bằng một ví dụ về một con người. Khi còn nhỏ, chúng ta thường học những thứ với sự giúp đỡ của những người lớn tuổi, bao gồm cả cha mẹ hoặc giáo viên . Rồi sau này bằng cách tự học hoặc luyện tập chúng ta tiếp tục học hỏi trong suốt cuộc đời. Các nhà khoa học và nhà nghiên cứu cũng đang làm cho cỗ máy trở nên thông minh giống như con người và ANN cũng đóng một vai trò rất quan trọng do những lý do sau:

* Với sự trợ giúp của mạng nơ-ron, chúng ta có thể tìm ra lời giải cho những vấn đề như vậy mà phương pháp thuật toán nào là tốn kém hoặc không tồn tại.
* Mạng nơ-ron có thể học theo ví dụ, do đó chúng ta không cần phải lập trình nó ở nhiều mức độ.
* Mạng nơron có độ chính xác và tốc độ nhanh hơn đáng kể so với tốc độ thông thường.

## Lĩnh vực ứng dụng

Tiếp theo là một số lĩnh vực mà ANN đang được sử dụng. Nó gợi ý rằng ANN có một cách tiếp cận liên ngành trong quá trình phát triển và ứng dụng của nó.

### **\* Nhận dạng giọng nói**

Lời nói chiếm một vai trò nổi bật trong tương tác giữa con người với con người. Do đó, mọi người mong đợi giao diện giọng nói với máy tính là điều tự nhiên. Trong thời đại hiện nay, để giao tiếp với máy móc, con người vẫn cần những ngôn ngữ phức tạp, khó học và sử dụng. Để giảm bớt rào cản giao tiếp này, một giải pháp đơn giản có thể là giao tiếp bằng ngôn ngữ nói mà máy móc có thể hiểu được.

Tuy nhiên, lĩnh vực này đã đạt được nhiều tiến bộ vượt bậc, những hệ thống kiểu như vậy vẫn đang phải đối mặt với vấn đề hạn chế về từ vựng hoặc ngữ pháp cùng với vấn đề đào tạo lại hệ thống cho những người nói khác nhau trong những điều kiện khác nhau. ANN đang đóng một vai trò quan trọng trong lĩnh vực này. Các ANN sau đây đã được sử dụng để nhận dạng giọng nói -

* Mạng nhiều lớp
* Mạng nhiều lớp với kết nối lặp lại
* Bản đồ tính năng tự tổ chức của Kohonen

Mạng hữu ích nhất cho việc này là bản đồ tính năng “Kohonen Self-Organizing

”, có đầu vào là các đoạn ngắn của dạng sóng giọng nói. Nó sẽ ánh xạ cùng một loại âm vị với mảng đầu ra, được gọi là kỹ thuật trích xuất đặc trưng. Sau khi trích xuất các tính năng, với sự trợ giúp của một số mô hình âm thanh như xử lý back-end, nó sẽ nhận ra lời nói.

### **\* Nhận dạng ký tự**

Đây là một vấn đề thú vị thuộc lĩnh vực chung của Nhận dạng mẫu. Nhiều mạng nơ-ron đã được phát triển để tự động nhận dạng các ký tự viết tay, cả chữ cái hoặc chữ số. Sau đây là một số ANN đã được sử dụng để nhận dạng ký tự -

* Mạng nơron đa lớp chẳng hạn như mạng nơron lan truyền ngược.
* Neocognitron

Mặc dù mạng nơ-ron lan truyền ngược có một số lớp ẩn, nhưng mô hình kết nối từ lớp này sang lớp tiếp theo được bản địa hóa. Tương tự, neocognitron cũng có một số lớp ẩn và quá trình đào tạo của nó được thực hiện từng lớp cho các loại ứng dụng như vậy.

### **\* Ứng dụng Xác minh Chữ ký**

Chữ ký là một trong những cách hữu ích nhất để ủy quyền và xác thực một người trong các giao dịch hợp pháp. Kỹ thuật xác minh chữ ký là một kỹ thuật không dựa trên thị lực.

Đối với ứng dụng này, cách tiếp cận đầu tiên là trích xuất đặc trưng hay đúng hơn là tập hợp đặc trưng hình học đại diện cho chữ ký. Với các bộ tính năng này, chúng ta phải huấn luyện các mạng nơ-ron bằng cách sử dụng một thuật toán mạng nơ-ron hiệu quả. Mạng nơ-ron được đào tạo này sẽ phân loại chữ ký là thật hay giả mạo trong giai đoạn xác minh.

### **\* Nhận dạng khuôn mặt người**

Nó là một trong những phương pháp sinh trắc học để xác định khuôn mặt đã cho. Đây là một nhiệm vụ điển hình vì đặc điểm của hình ảnh "không có khuôn mặt". Tuy nhiên, nếu một mạng nơron được đào tạo tốt, thì nó có thể được chia thành hai lớp: hình ảnh có khuôn mặt và hình ảnh không có khuôn mặt.

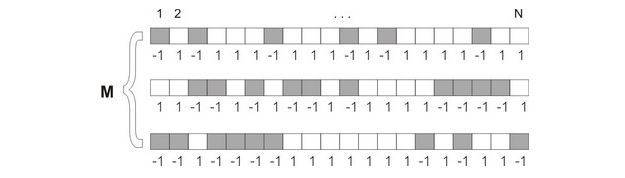
Đầu tiên, tất cả các hình ảnh đầu vào phải được xử lý trước. Sau đó, kích thước của hình ảnh đó phải được giảm bớt. Và, cuối cùng nó phải được phân loại bằng cách sử dụng thuật toán đào tạo mạng nơ-ron. Các mạng nơ-ron sau được sử dụng cho mục đích đào tạo với hình ảnh được xử lý trước -

* Được đào tạo mạng nơ-ron truyền về phía trước nhiều lớp được kết nối đầy đủ với sự trợ giúp của thuật toán lan truyền ngược.
* Để giảm kích thước, Phân tích thành phần chính PCA Được sử dụng.

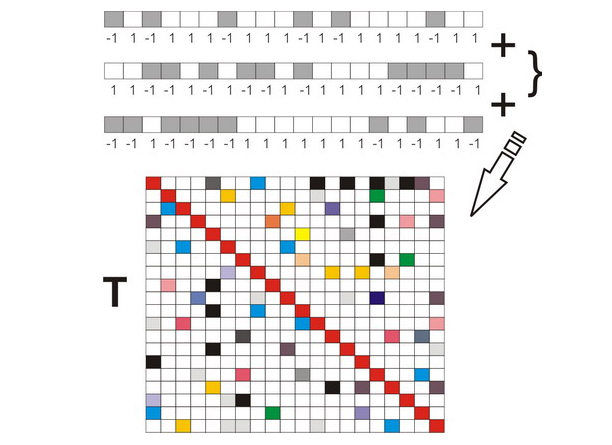
(Chú thích: Bài làm dựa trên tài liệu <https://www.tutorialspoint.com>)

**7. Chương trình**

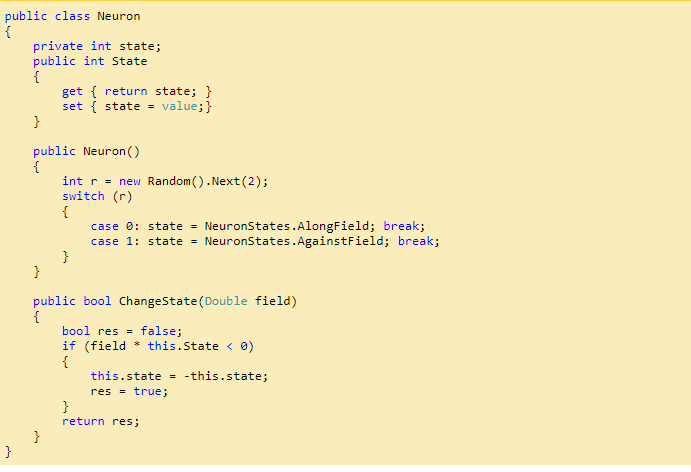
Chúng ta hãy xem xét HNN trên quan điểm thực tế. Giả sử bạn có M , N vectơ nhị phân chiều (hình 3) và bạn muốn lưu trữ chúng trong mạng nơ-ron.



Ta thêm chúng vào ma trận liên kết, sử dụng phép tính tổng đơn giản:

****

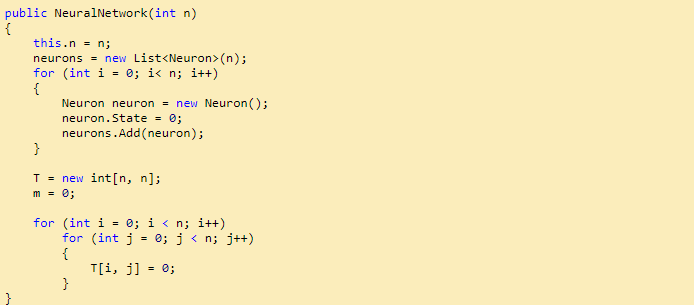
Nó bao gồm hai lớp chính: Neuron và  NeuralNetwork. Neuron là một lớp cơ sở, chứa thuộc tínhState và phương thức ChangeState(). Statelà một số Int32, nhưng thực tế nó chỉ nhận hai giá trị: +1 hoặc -1 (Các giá trị này cũng có thể truy cập được từ lớp tĩnh NeuronStates. Trong đó NeorunStates.AlongFieldbằng 1 và NeorunStates.AgainstFieldbằng -1. ChangeState()nhận giá trị của trường tác động lên tế bào thần kinh và đưa ra quyết định thay đổi trạng thái của chính nó hay không. ChangeState()trả lại true nếu State đã được thay đổi.



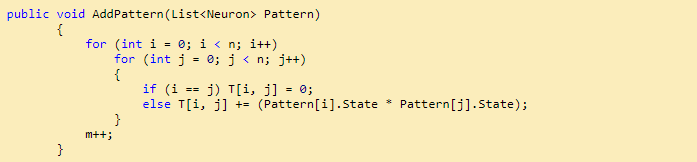
NeuralNetwork lớp chứa danh sách đã nhập của các nơ-ron, các phương thức để thêm mẫu và chạy động lực học



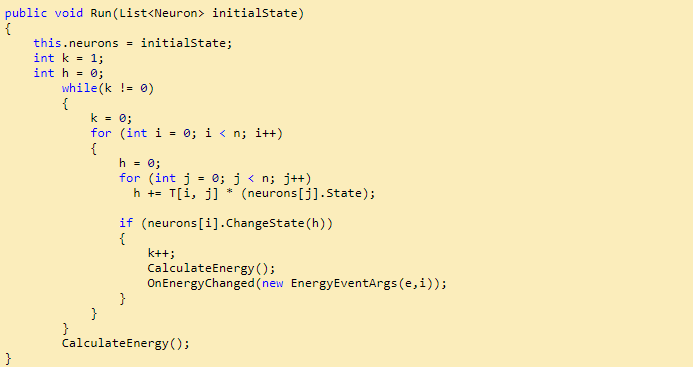
Phương thức khởi tạo lớp khởi tạo tất cả các trường, tạo danh sách và mảng và điền vào ma trận liên kết bằng các số không:



Các phương thức AddPattern()và AddRandomPattern() cho biết thêm nào đó (hoặc một cách ngẫu nhiên tạo ra) mẫu vào ma trận kết nối:



Phương thức Run() chạy các động lực học



Mỗi khi bất kỳ vòng quay nào thay đổi trạng thái của nó, năng lượng của hệ thống thay đổi và NN tạo ra sự kiện EnergyChanged. Sự kiện này cho phép người đăng ký theo dõi trạng thái NN kịp thời.

8. Nhận xét:

- Mô tả về Running:

Theo mô tả trong phần Running, một nút phải được chọn, sau đó hành vi được xác định. Đây là một mô tả khá không đầy đủ. Hành vi của nút được xác định như thế nào sau khi nó được chọn? Nó đang cập nhật hoặc thay thế giá trị s\_i bằng nó là kích hoạt a\_i (theo định nghĩa được đưa ra trước đó), nhưng sẽ tốt hơn nếu điều này được nêu rõ ràng

- Trọng số đối xứng:

Điều kiện của các trọng số đối xứng đảm bảo rằng việc tuân theo quy tắc cập nhật làm cho năng lượng trở thành một hàm giảm đơn điệu, đảm bảo sự hội tụ đến cực tiểu cục bộ, tuy nhiên, các trọng số không đối xứng dường như không ngụ ý việc sử dụng mạng như một hệ thống bộ nhớ có địa chỉ nội dung.

- Các kết nối không cần đối xứng:

Hiệu suất của các mạng với "trường hợp đặc biệt" của các trọng số đối xứng, mạng cũng hoạt động tốt với các trọng số không đối xứng. Cụ thể, "Luồng trong không gian được tạo ra bởi thuật toán mô hình này có các thuộc tính cần thiết cho bộ nhớ địa chỉ nội dung cho dù Tij có đối xứng hay không" -> các trọng số ít nhiều phải đối xứng để hội tụ.

- Kết nối giữa Hopfield và mô hình Ising:

Mô hình Ising là một mô hình của thuyết [sắt từ](https://en.wikipedia.org/wiki/Ferromagnetism" \o ") . Nguyên tử có tính lưỡng cực (nghĩa là dương hoặc âm) và chúng có các liên kết và tương tác cục bộ của các nguyên tử có thể dẫn đến một số chuyển đổi trạng thái ở cấp độ toàn cục. Chúng là nền tảng lý thuyết của Hopfield Networks và Hopfield đặc biệt đề cập đến chúng và thay đổi các nguyên tử thành , tức là chúng có một ngưỡng.

- Lưới Hopfield có thể có các đơn vị nhận giá trị 1 hoặc -1 hoặc các đơn vị nhận giá trị 1 hoặc 0 và tiếp tục đưa ra các quy tắc cập nhật trong hai trường hợp. Điều này có vẻ như gây chú ý nhiều đến một vấn đề nhỏ là mở rộng quy mô. Khi chọn quy ước và sau đó đề cập rằng quy ước khác cũng có thể được sử dụng. Điều quan trọng là các đơn vị cũng có thể liên tục. Các đơn vị lưỡng cực chỉ là một trường hợp cụ thể được nghiên cứu trong mạng Hopfield.

- Năng lượng:

Về chức năng năng lượng. Đôi khi ngưỡng (θi) là một hàm phức tạp hơn và chúng ta không thể dễ dàng kết hợp nó vào hàm Năng lượng

- Hạn chế:

Chúng ra chưa biết có thể huấn luyện các lưới Hopfield để biến các đầu vào cụ thể trở thành cực tiểu cục bộ hay không. Nó cũng không cho biết liệu bạn có thể có hai mạng lưới Hopfield với cùng một cực tiểu cục bộ nhưng được đề cập khác nhau hay không.

- Đơn vị ngưỡng θi:

+ Có lẽ phần đơn vị treshold Binary có thể được sáp nhập với [Perceptron](https://en.wikipedia.org/wiki/Perceptron" \o "Perceptron)

+ Một perceptron và một mạng hopfield khác nhau bởi hình dạng của mạng của chúng: perceptron là chuyển tiếp trong khi các lưới hopfield là lặp lại. Vì vậy, nó có thể sẽ là sai lầm khi liên kết hai thành phần với nhau.

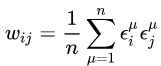
+ Ngoài ra, mạng Hopfield có thể sử dụng bất kỳ loại phi tuyến tính nào, không chỉ là một ngưỡng. Vì vậy, nói một mạng Hopfield là một kết nối tuần hoàn đối xứng của các đơn vị perceptron là không đủ chung.

- Khởi tạo mạng:

Một mạng Hopfield trước hết được huấn luyện với các mẫu cố định các trọng số. Để khởi tạo, người dùng cần đặt giá trị của các đơn vị cho mẫu đầu vào.

- Chuẩn hóa:

Hiện tại quy tắc cho việc học là:



với n là số mẫu

Nó chỉ cần thiết để tránh trọng lượng 'rất lớn' khi lưu trữ 'rất nhiều' nội dung.

- Dung lượng:

Sử dụng ~ 0,5N ^ 2 trọng số (float) để nhớ ~ 0,138N ^ 2 bit dường như khá ít.