# Giới thiệu

Mạng neural nhân tạo là một mô hình toán học đã được nghiên cứu từ lâu và được ứng dụng nhiều vào các bài toán mô phỏng, nhận dạng, dự đoán. Gần đây mạng neural nhân tạo được quan tâm và ứng dụng ngày càng nhiều vào các bài toán dự báo dữ liệu chuỗi thời gian. Mục đích của đề tài này là tìm hiểu về nguyên tắc hoạt động, hai giải thuật huấn luyện mạng neural: lan truyền ngược (backpropagation) và RPROP, cách áp dụng mạng neural nhân tạo vào việc dự báo dữ liệu chuỗi thời gian, hiện thực một chương trình dự báo dữ liệu chuỗi thời gian sử dụng mạng neural nhân tạo với hai giải thuật học là lan truyền ngược và RPROP , chạy thử nghiệm chương trình trên một số bộ dữ liệu mẫu để đánh giá độ chính xác dự báo và tính hữu hiệu của các giải thuật.

Bài báo cáo chia làm 4 chương.

**Chương 1:** giới thiệu về cấu trúc, nguyên tắc hoạt động của mạng neural nhân tạo cùng với hai giải thuật huấn luyên: lan truyền ngược và RPROP.

**Chương 2:** giới thiệu về cách ứng dụng mạng neural nhân tạo vào công tác dự báo dữ liệu chuỗi thời gian.

**Chương 3:** trình bày việc hiện thực chương trình dự báo dữ liệu chuỗi thời gian bằng mạng neural nhân tạo.

**Chương 4:** trình bày việc chạy thử nghiệm chương trình trên một số bộ dữ liệu và đánh giá tính hiệu quả của hai giải thuật lan truyền ngược và RPROP.

# Mạng neural nhân tạo: cấu trúc, nguyên tắc hoạt động và các giải thuật huấn luyện.

## Cấu trúc của mạng neural nhân tạo.

Mạng neural nhân tạo là một mạng gồm một tập các node được kết nối với nhau bằng các cạnh có trọng số, mỗi node là một đơn vị tính toán thường gọi là perceptron.

Một perceptron (Hình 1) thực hiện một công việc rất đơn giản: nó nhận tín hiệu vào từ các perceptron phía trước hay một nguồn bên ngoài và sử dụng chúng để tính tín hiệu ra. Mỗi perceptron có thể có nhiều tín hiệu đầu vào nhưng chỉ có một tín hiệu đầu ra duy nhất. Đôi khi các perceptron còn có một giá trị gọi là độ lệch (bias) được gộp vào các tính hiệu đầu vào để tính tín hiệu ra.



Hình 1: Perceptron

Trong một mạng nơron có ba kiểu perceptron:

1) Các perceptron đầu vào, nhận tín hiệu từ bên ngoài;

2) Các perceptron đầu ra, gửi dữ liệu ra bên ngoài;

3) Các perceptron ẩn, tín hiệu vào của nó được truyền từ các perceptron khác và tín hiệu ra được truyền đến các perceptron khác trong mạng.

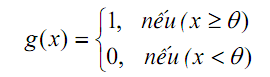
Khi nhận được các tín hiệu đầu vào, một perceptron sẽ nhân mỗi tín hiệu với trọng số tương ứng rồi lấy tổng các giá trị vừa nhận được. Kết quả sẽ được đưa vào một hàm số gọi là hàm kích hoạt mà đôi khi còn gọi là hàm chuyển để tính ra tín hiệu đầu ra.

Có 4 loại hàm kích hoạt thường dung:

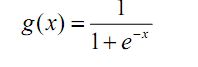
1. Hàm đồng nhất (Identity function):



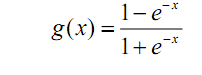
1. Hàm ngưỡng:



1. Hàm sigmoid:



1. Hàm sigmoid lưỡng cực



Các perceptron liên kết với nhau qua các cạnh có trong số tạo thành mạng neural nhân tạo. Tùy theo số lượng các perceptron và cách thức liên kết của chúng mà tạo thành các mạng neural khác nhau có khả năng khác nhau. Có hai loại mạng neural nhân tạo cơ bản là mạng truyền thẳng và mạng hồi quy.

1. Mạng truyền thẳng: Một perceptron ở lớp đứng trước sẽ kết nối với tất cả các perceptron ở lớp đứng sau. Tín hiệu chỉ được truyền theo một hướng từ lớp đầu vào qua các lớp ẩn (nếu có) và đến lớp đầu ra. Nghĩa là tín hiệu ra của một perceptron không được phép truyền cho các perceptron trong cùng lớp hay ở lớp trước. Đây là loại mạng rất phổ biến và được dung nhiều trong việc dự báo dữ liệu chuỗi thời gian. Bài báo cáo này chỉ tập trung vào mô hình mạng này.



Hình : mạng neural truyền thẳng

1. Mạng hồi quy: Khác với mạng truyền thẳng, mạng hồi quy có chứa các liên kết ngược từ một perceptron đến perceptron ở lớp trước nó.



Hình : mạng neural hồi quy

## Nguyên tắc hoạt động của mạng neural nhân tạo.

Trong mạng neural truyền thẳng nhiều lớp, các tín hiệu sẽ được truyền từ các perceptron đầu vào qua các perceptron ở lớp ẩn (nếu có) và đến các perceptron ở lớp đầu ra. Các perceptron đầu vào là các perceptron đặc biệt, chúng không thực hiện một tính toán nào cả mà chỉ truyền tín hiệu từ môi trường ngoài sang các perceptron ở lớp sau. Một perceptron i không phải là perceptron ở lớp đầu vào sẽ tính giá trị đầu ra của nó như sau. Đầu tiên nó sẽ tính tổ hợp tuyến tính các giá trị đầu vào của mình



Ở đây pred(j) là các perceptron đứng trước i và kết nối với i, sj là đầu ra của cá perceptron này và cũng là đầu vào của I, wij trọng số của cạnh nối từ perceptron j đến perceptron i,  là độ lệch (bias) của perceptron i. Đôi khi giá trị độ lệch được biểu diễn bằng một trọng số của một cạnh kết nối từ một perceptron giả có giá trị đầu ra luôn là 1 đến perceptron i. Sau đó giá trị neti sẽ được truyền vào hàm kết hợp để tính giá trị đầu ra của si của perceptron i. Nếu perceptron i là perceptron ở tầng suất của mạng thì si chính là tín hiệu đầu ra của mạng neural, nếu không thì si sẽ được truyền đến làm giá trị đầu vào cho các perceptron ở lớp kế sau. Ví dụ nếu hàm kích hoạt là hàm sigmoid thì si được tính như sau .

Mạng neural nhân tạo có thể được xem như một hàm số từ một tập các giá trị đầu vào đến một tập các giá trị đầu ra có các tham số là số lớp, số perceptron trên mỗi lớp và các tập hợp các trọng số wij. Bài toán trên mạng neural là làm sao xác định được các thông số trên để mạng neural thực hiện được yêu cầu mong muốn. Điều này được thực hiện qua một quá trình gọi là quá trình huấn luyện mạng.

## Huấn luyện mạng neural.

Chức năng của một mạng nơron được quyết định bởi các nhân tố như: hình dạng mạng (số lớp, số đơn vị trên mỗi lớp, cách mà các lớp được liên kết với nhau) và các trọng số của các liên kết bên trong mạng. Hình dạng của mạng thường là cố định, và các trọng số được quyết định bởi một thuật toán huấn luyện (training algorithm). Tiến trình điều chỉnh các trọng số để mạng “nhận biết” được quan hệ giữa đầu vào và đích mong muốn được gọi là học (learning) hay huấn luyện (training). Rất nhiều thuật toán huấn luyện đã được phát minh để tìm ra tập trọng số tối ưu làm giải pháp cho các bài toán. Các thuật toán đó có thể chia làm hai nhóm chính: Học có giám sát (Supervised learning) và Học không có giám sát (Unsupervised Learning).

Học có giám sát: mạng được huấn luyện bằng cách cung cấp cho nó các cặp mẫu đầu vào và các đầu ra mong muốn (target values). Các cặp này có sẵn trong quá trình thu nhập dữ liệu. Sự khác biệt giữa các đầu ra theo tính toán trên mạng so với các đầu ra mong muốn được thuật toán sử dụng để thích ứng các trọng số trong mạng. Điều này thường được đưa ra như một bài toán xấp xỉ hàm số - cho dữ liệu huấn luyện bao gồm các cặp mẫu đầu vào x, và một đích tương ứng t, mục đích là tìm ra hàm f(x) thoả mãn tất cả các mẫu học đầu vào. Đây là mô hình học rất phổ biến trong việc áp dụng mạng neural vào bài toán dự báo dữ liệu chuỗi thời gian. Hai giả thuật được đề cập trong bài báo cáo này, lan truyền ngược và RPROP là hai giải thuật học thuộc mô hình này.



Hình : Mô hình học có giám sát

Học không có giám sát: với cách học không có giám sát, không có phản hồi từ môi trường để chỉ ra rằng đầu ra của mạng là đúng. Mạng sẽ phải khám phá các đặc trưng, các điều chỉnh, các mối tương quan, hay các lớp trong dữ liệu vào một cách tự động. Trong thực tế, đối với phần lớn các biến thể của học không có giám sát, các đích trùng với đầu vào. Nói một cách khác, học không có giám sát luôn thực hiện một công việc tương tự như một mạng tự liên hợp, cô đọng thông tin từ dữ liệu vào.

Để huấn luyện một mạng và xét xem nó thực hiện tốt đến đâu, ta cần xây dựng một hàm mục tiêu (hay hàm đánh giá) để cung cấp cách thức đánh giá khả năng hệ thống một cách không nhập nhằng. Việc chọn hàm mục tiêu là rất quan trọng bởi vì hàm này thể hiện các mục tiêu thiết kế và quyết định thuật toán huấn luyện nào có thể được áp dụng. Để phát triển một hàm mục tiêu đo được chính xác cái chúng ta muốn không phải là việc dễ dàng. Trong các bài toán dự báo dữ liệu chuỗi thời gian sử dụng các giải thuật học có giám sát thì hàm tổng bình phương lỗi là hàm đánh giá rất hay dùng.



Ở đây, P là tập hợp các mẫu dự liệu, mỗi mẫu p thuộc P là một cặp các giá trị gồm một vector các giá trị đầu vào  và một vector các giá trị đầu ra  mong muốn, là giá trị đầu ra tính toán theo mạng neural, n là số perceptron ở tầng xuất.

Việc đạt được mục tiêu học tương đương với việc tìm được giá trị của các trọng số wij sao cho E đạt được giá trị tối ưu. Các trọng số liên kết được thay đổi theo hướng tìm kiếm d(t), sẽ dần đưa các trọng số về giá trị làm cho E được cực tiểu.



Ở đây  là hệ số học (learning rate), có vai trò điều tiết mức độ thay đổi của trọng số trong các bước cập nhập. Nó thông thường có giá trị nhỏ (ví dụ 0.1) và thỉnh thoảng được điều chỉnh giảm khi số lần cập nhập các trọng số tăng lên. Giá trị của d(t) được xác định giựa vào độ dốc (gradient) của E(t), cụ thể là .

### Giải thuật lan truyền ngược.

Giải thuật lan truyền ngược tìm tập các trọng số thích hợp cho một mạng neural truyền thẳng nhiều lớp. Nó áp dụng phương pháp giảm độ dốc (gradient descent) để tối thiểu hóa bình phương sai số giữa kết quả xuất của mạng với kết quả xuất mong muốn. Ý tưởng chính của giải thuật là giá trị lỗi sẽ được lan truyền ngược từ tầng xuất về tầng nhập để tính (đạo hàm riêng phần của hàm mục tiêu E theo từng trọng số wij).

Ta tính đạo hàm riêng phần này như sau:



với 

Ở đây wij là trọng số của cạnh nối perceptron j đến perceptron i, si là kết quả xuất của perceptron i, f() là hàm kích hoạt của các perceptron, 

 được tính theo hai trường hợp sau:

1. Nếu perceptron i là perceptron ở tầng xuất thì:

, với ti là giá trị xuất mong muốn của perceptron i

1. Nếu perceptron i là perceptron ở tầng ẩn thì:



Ở đây succ(i) là các perceptron ở lớp ngay sau perceptron i. Các công thức này cho phép ta xây dựng một thủ tục tính đạo hàm riêng của hàm mục tiêu E theo các trọng số wij như sau: Bắt đầu tính toán từ perceptron ở tầng xuất, sau đó sử dụng kết quả vừa tính được vào việc tính toán ở các perceptron ở tầng trước. Nói các khác thông tin về độ dốc (gradient) được lan truyền từ tầng xuất đến tầng nhập. Do đó giả thuật này được gọi là giải thuật lan truyền ngược.

Mỗi khi thông tin về đạo hàm riêng phần đã biết, bước tiếp theo trong giải thuật lan truyền ngược là cập nhập các trọng số wij.



Ở đây  là hệ số học (learning rate) có vai trò điều tiết mức độ thay đổi của trọng số trong các bước cập nhập. Kỷ thuật tối ưu này gọi là giảm độ dốc (gradient descent).

Mặc dù giải thuật lan truyền ngược tương đối đơn giản nhưng trong thực tế việc lựa chọn một hệ số học phù hợp là không hề đơn giản. Hệ số học quá nhỏ sẽ dẫn đến thời gian hội tụ của giải thuật quá lâu, ngược lại hệ số học quá lớn sẽ dẫn đến hiện tượng giao động (oscillation), ngăn không cho giá trị hàm mục tiêu hội tụ về một diểm nhất định. Hơn nữa, mặc dù điểm tối ưu cục bô có thể được chứng minh là luôn có thể đạt được ở một vài trường hợp cụ thể nhưng không có gì đảm bảo giải thuật sẽ tìm được tối ưu toàn cục của hàm mục tiêu E. Một vấn đề khác nữa là kích cỡ của đạo hàm cũng ảnh hướng đến sự cập nhập các trọng số. Nếu đạo hàm riêng phần quá nhỏ thì  nhỏ, nếu đạo hàm riêng phần lớn thì  lớn. Độ lớn của đạo hàm riêng phần thay đổi không thể biết trước được theo hình dạng của hàm lỗi E trong mỗi lần lặp. Do đó quá trình học không ổn định.

Để cho quá trình học ổn định người ta thêm vào một hệ số quán tính (momentum term)



Hệ số quán tính có tác dụng điều chỉnh mức độ ảnh hưởng của giá trị ở bước lặp trước lên giá trị .

Cơ bản có hai phương pháp cập nhập các trọng số phân loại theo thời điểm cập nhập: học theo mẫu (learning by pattern) và hoc theo epoch (learning by epoch). Một epoch là một lần học duyệt qua tất cả các mẫu trong tập dữ liệu mẫu dùng để học.

Trong phương pháp học theo mẫu đôi khi còn dược gọi là học trực tuyến (online learning), cứ mỗi lần một mẫu trong tập dữ liệu được duyệt qua thì các trọng số sẽ được cập nhập. Phương pháp này cố gắng tối thiểu hàm lỗi tổng thể (overall error) bằng cách tối ưu hàm lỗi cho từng mẫu trong tập dữ liệu học. Phương pháp này làm việc tốt cho các tập dữ liệu mẫu có kích cỡ lớn và chứa đựng nhiều thông tin dư thừa.

Phương pháp học theo epoch (learning by epoch) thực hiện lấy tổng tất cả thông tin về độ dốc (gradient) cho toàn bộ tập mẫu (pattern set) sau đó mới cập nhập các trọng số, nghĩa là nó thực hiện việc cập nhập trọng số sau khi đã duyệt qua hết các mẫu trong tập dữ liệu. Phương pháp này còn có tên gọi khác là học theo bó (batch learning).

Sau đây là mã giả cho giải thuật lan truyền ngược theo phương pháp học trực tuyến (online learning):

1. *Initialize all weights to small random number*
2. *Loop until terminated condition satisfied*
   1. *For each training pattern, do*
      1. *Input the training pattern to the network and compute the network outputs.*
      2. *For each output perceptron k*
         1. *Compute *
      3. *For each hidden perceptron h, from the last hidden to the first hidden*
         1. *Compute *
      4. *For each wij in the network* 
         1. *Compute *
         2. *Compute *
      5. *End for*
   2. *End for*
3. *End loop*

Giải thuật lan truyền ngược cần hai thông số nhập vào đó là hệ số học và hệ số quán tính. Đối với mỗi bài toán khác nhau các thông số này cần có các giá trị khác nhau để đạt được sự hiệu quả trong quá trình học. Việc xác định các thông số này một cách đúng đắn không phải là một việc dễ dàng cần nhiều công sức và kinh nghiệm.

### Giải thuật RPROP.

Giải thuật lan truyền ngược gặp một vấn đề ở chỗ giá trị cập nhập trọng số () không những phụ thuộc vào dấu của đạo hàm riêng phần mà còn bị ảnh hưởng bởi độ lớn của nó, điều này làm cho quá trình học không được ổn định. Việc thêm vào hệ số quán tính không giải quyết trọn vẹn vấn đề bởi vì ta không biết giá trị tối ưu cho hệ số này là bao nhiêu. Để giải quyết vấn đề trên người ta đưa ra một giải thuật mới gọi là RPROP.

RPROP là viết tắt của từ ‘resilient propagation’, nghĩa là lan truyền đàn hồi. RPROP thực hiện cập nhập các trọng số wij dựa vào thông tin về dấu của các đạo hàm riêng phần điều này giúp nó tránh được sự ảnh hưởng của độ lớn của các đạo hàm riêng phần này. Để thực hiện điều này các trọng số sẽ có một giá trị cập nhập riêng chỉ phụ thuộc vào dấu của . Giá trị này được cập nhập trong quá trình học theo quy luật sau:



Ở đây  là các hệ số cố định của quá trình học dùng để hiệu chỉnh các giá trị cập nhập cho từng trọng số tùy theo hình dạng của hàm lỗi.

Mỗi lần đạo hàm riêng phần theo trọng số wij của hàm lỗi E đổi dấu, nghĩa là giá trị cập nhập vừa thực hiện là quá lớn và giải thuật đã nhảy vượt qua điểm tối ưu cục bộ thì giá trị cập nhập sẽ giảm đi theo một thừa số . Ngược lại nếu đạo hàm riêng phần vẫn giữ nguyên dấu thì giá trị cập nhập sẽ được tăng lên để tăng tốc độ hội tụ. Cứ mỗi lần giá trị cập nhập được biết thì các trọng số được điều chỉnh theo luật sau: nếu đạo hàm riêng phần dương thì trọng số được giảm đi một lượng bằng với giá trị cập nhập (), nếu đạo hàm riêng phần âm thì giá trị cập nhập được cộng thêm vào trọng số.





Tuy nhiên có một trường hợp đặc biệt đó là khi đạo hàm riêng phần đổi dấu, nghĩa là bước cập nhập trước quá lớn làm cho điểm tối ưu bị nhảy vượt qua. Giá trị trọng số phải được trả về giá trị trước khi thay đổi, giá trị cập nhập  sẽ được giảm xuống và ở bước kế sau ta sẽ không cập nhập giá trị này. Thực tế ta có thể làm việc này bằng cách gán  và 

Giải thuật RPROP thực hiện việc thích nghi các giá trị cập nhập của các trọng số tùy theo độ dốc (gradient) của hàm lỗi E, mà thông tin về tổng độ dốc của hàm lỗi cho toàn bộ tập dữ liệu mẫu đáng tin hơn thông tin về độ dốc chỉ cho một mẫu trong tập mẫu nên giải thuật RPROP thực hiện theo mô hình học theo bó (học theo epoch). Các thông tin về đạo hàm riêng phần sẽ được cộng dồn qua từng mẫu trong tập huấn luyện và các trọng số sẽ được cập nhập sau khi đã duyệt qua hết các mẫu.

Giải thuật RPROP ban đầu cũng thực hiện các bước giống như giải thuật lan truyền ngược, các thông tin về đạo hàm riêng phần của hàm lỗi theo các trọng số sẽ được lan truyền ngược từ các lớp sau đến các lớp trước. Khi các thông tin về các đạo hàm riêng phần này có đủ thì giải thuật sẽ thực hiện việc cập nhập các trọng số theo các quy tắc nêu ở trên.

Mã giả cho phần cập nhập trọng số của giải thuật RPROP như sau:

*For all weights and biases {*

*if () then {*

* = minimum ()*

* = - sign ()\**

* = + *

*} else if () then {*

* = maximum ()*

* = - *

**

*}*

*else if () then {*

* = - sign ()\**

* = + *

*}*

*}*

Ở đây hàm số minimum và maximum lần lược là hai hàm trả về giá trị nhỏ nhất và lớn nhất của hai số. Hàm số sign(x) trả về 1 nếu x dương, trả về -1 nếu x âm và trả về 0 trong các trường hợp còn lại.

Ban đầu các giá trị cập nhập  sẽ được khởi tạo một giá trị dương ban đầu . Lựa chọn tốt cho  là 0.1 nhưng theo các nghiên cứu thì việc lựa chọn tham số này không ảnh hưởng nhiều đến tốc độ hội tụ của giải thuật. Các thông số về  và để tránh vấn đề tràn số của các biến thực. Giá trị  được thiết lập thường là 1.0e-6, còn giá trị  là 50.0. Thông thường độ hội tụ của giải thuật không bị ảnh hưởng bởi các thông số này nhưng đôi khi thông số  được chọn là một giá trị nhỏ (ví dụ 1.0) để ngăn giải thuật không rơi quá nhanh vào một cực tiểu cục bộ. Hai thông số  và được cố định ở hai giá trị lần lược là 1.2 và 0.5, để việc lựa chọn các tham số cho giải thuật được đơn giản. Trong thực tế, hai thông số cần được lưa chọn cho giải thuật RPROP là  và .