MỤC LỤC

MỞ ĐẦU

Sự phát triển mạnh mẽ của Công nghệ nói chung và Công nghệ thông tin nói riêng đã tạo nên nhiều hệ thống thông tin phục vụ việc tự động hoá mọi hoạt động kinh doanh cũng như quản lý trong xã hội. Điều này đã tạo ra những dòng dữ liệu khổng lồ trở thành hiện tượng "bùng nổ thông tin". Nhiều hệ quản trị cơ sở dữ liệu mạnh với các công cụ phong phú và thuận tiện đã giúp con người khai thác có hiệu quả các nguồn tài nguyên dữ liệu lớn nói trên. Bên cạnh chức năng khai thác cơ sở dữ liệu có tính tác nghiệp, sự thành công trong kinh doanh không chỉ thể hiện ở năng suất của các hệ thống thông tin mà người ta còn mong muốn cơ sở dữ liệu đó đem lại tri thức từ dữ liệu hơn là chính bản thân dữ liệu. Phát hiện tri thức trong cơ sở dữ liệu (Knowledge Discovery in Databases - KDD) là một quá trình hợp nhất các dữ liệu từ nhiều hệ thống dữ liệu khác nhau tạo thành các kho dữ liệu, phân tích thông tin để có được nhiều tri thức tiềm ẩn có giá trị. Trong đó, khai phá dữ liệu (Data Mining) là quá trình chính trong phát hiện tri thức. Sử dụng các kỹ thuật và các khái niệm của các lĩnh vực đã được nghiên cứu từ trước như học máy, nhận dạng, thống kê, hồi quy, xếp loại, phân nhóm, đồ thị, mạng nơron, mạng Bayes,... được sử dụng để khai phá dữ liệu nhằm phát hiện ra các mẫu mới, tương quan mới, các xu hướng có ý nghĩa.

Luận văn với đề tài "***Học mạng nơron theo mô hình SOM và ứng dụng trong bài toán quản lý khách hàng vay vốn Ngân hàng"*** khảo sát lĩnh vực khai phá dữ liệu dùng mạng nơron. Luận văn tập trung vào phương pháp học mạng nơron có giám sát và không có giám sát, dùng thuật toán SOM để giải quyết bài toán phân cụm theo mô hình mạng nơron.

Phương pháp nghiên cứu chính của luận văn là tìm hiểu các bài báo khoa học được xuất bản trong một vài năm gần đây về khai phá dữ liệu dùng mạng nơron và áp dụng công cụ SOM ToolBox để giải quyết bài toán phân tích dữ liệu khách hàng vay vốn trong Ngân hàng.

Nội dung của bản luận văn gồm có phần mở đầu, ba chương và phần kết luận. Chương 1 giới thiệu về mạng nơron và các thành phần chính trong mạng nơron (mục 1.1), dùng mạng nơron trong khai phá dữ liệu nói chung và dữ liệu tài chính nói riêng (mục 1.2) và các phương pháp học sử dụng mạng nơron gồm học có giám sát (mục 1.3.1) với thuật toán BBP (Boosting-Based Perceptron) và học không có giám sát (mục 1.3.2).

Chương 2 trình bày chi tiết việc áp dụng mạng nơron trong khai phá dữ liệu mà đặc biệt là phân cụm dữ liệu (mục 2.1 và 2.2), có liên quan đến hai thuật toán học không có giám sát đó là thuật toán học ganh đua (mục 2.2.1) và thuật toán SOM (2.2.2). Trên cơ sở đó luận văn giới thiệu một số ứng dụng điển hình của SOM trong lĩnh vực tài chính (mục 2.3).

Chương 3, áp dụng SOM để giải quyết bài toán phân tích thông tin khách hàng vay vốn Ngân hàng, gồm việc tìm hiểu quy trình lập hồ sơ khách hàng vay vốn (mục 3.1), tìm hiểu bộ công cụ SOM Toolbox (mục 3.2 và 3.3) để xây dựng chương trình cho bài toán nói trên. Và cuối cùng là một số kết quả chạy chương trình và nhận xét.

Luận văn này được thực hiện dưới sự hướng dẫn khoa học của TS. Hà Quang Thụy. Tôi xin chân thành cảm ơn sâu sắc tới Thầy đã chỉ dẫn tận tình giúp tôi có thể hoàn thành bản luận văn này. Tôi xin chân thành cảm ơn các thầy giáo và các bạn trong bộ môn Các Hệ thống Thông tin đã có những góp ý hữu ích trong quá trình thực hiện bản luận văn. Tôi cũng vô cùng cảm ơn sự giúp đỡ và động viên khích lệ của người thân trong gia đình tôi, bạn bè và các đồng nghiệp trong Ngân hàng VPBank trong suốt quá trình thực hiện luận văn.

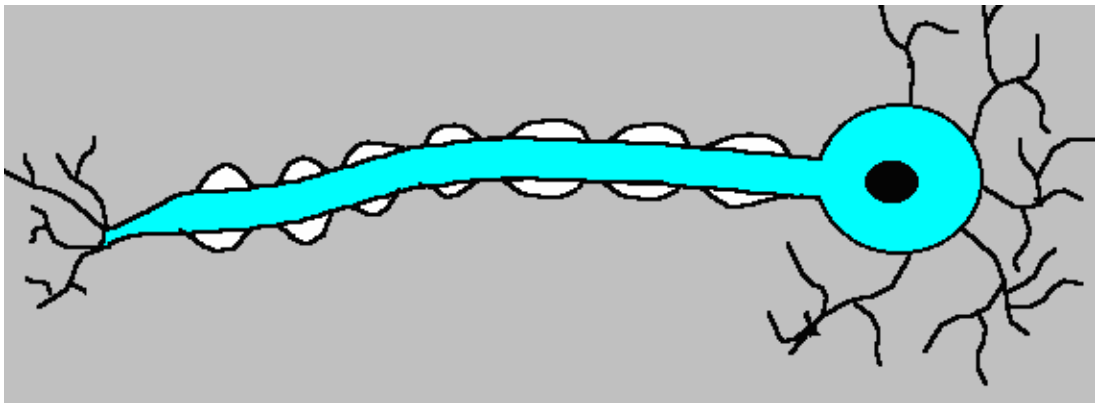
Hà nội, tháng 03 năm 2004

Đỗ Cẩm Vân

CHƯƠNG 1. MẠNG NƠRON VÀ ỨNG DỤNG TRONG HỌC MÁY

1.1 Mạng nơron

Bộ não con người chứa khoảng 10^{11} các phần tử (được gọi là nơron) liên kết chặt chẽ với nhau. Đối với mỗi nơron, có khoảng 10^4 liên kết với các nơron khác. Một nơron được cấu tạo bởi các thành phần như tế bào hình cây, tế bào thân và sợi trục thần kinh (axon). Tế bào hình cây có nhiệm vụ mang các tín hiệu điện tới tế bào thân, tế bào thân sẽ thực hiện gộp (sum) và phân ngưỡng các tín hiệu đến. Sợi trục thần kinh làm nhiệm vụ đưa tín hiệu từ tế bào thân tới tế bào hình cây của các nơron liên kết.



Hình 1. Nơron sinh học

Điểm tiếp xúc giữa một sợi trục thần kinh của nơron này với một tế bào hình cây của một nơron khác được gọi là khớp thần kinh (synapse). Sự sắp xếp các nơron và mức độ mạnh yếu của các khớp thần kinh do các quá trình hoá học phức tạp quyết định, sẽ thiết lập chức năng của mạng nơron.

Khi con người sinh ra, một bộ phận các nơron đã có sẵn trong não, còn các bộ phận khác được phát triển thông qua quá trình học, và trong quá trình đó xảy ra việc thiết lập các liên kết mới và loại bỏ đi các liên kết cũ giữa các nơron.

Cấu trúc mạng nơron luôn luôn phát triển và thay đổi. Các thay đổi có khuynh hướng chủ yếu là làm tăng hay giảm độ mạnh các mối liên kết thông qua các khớp thần kinh.

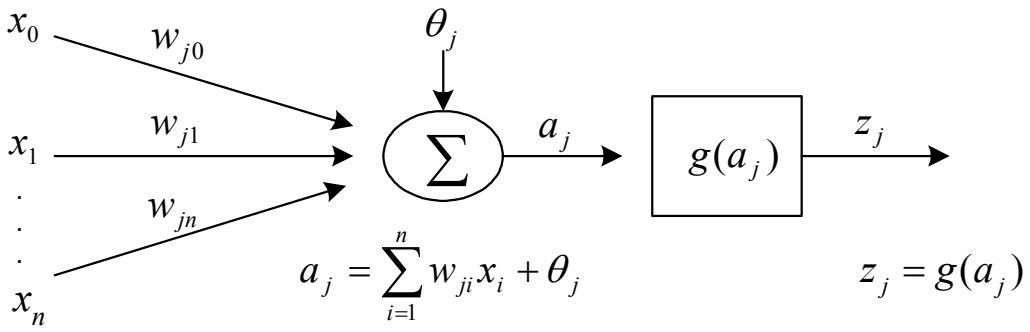
Một trong những phương pháp điển hình giải quyết bài toán học máy là thiết lập các mạng nơron nhân tạo. Mạng nơron nhân tạo chưa tiếp cận được sự phức tạp của bộ não. Tuy nhiên, do mô phỏng hoạt động học trong não mà về cơ bản có hai sự tương quan giữa mạng nơron nhân tạo và nơron sinh học. Thứ nhất, cấu trúc tạo thành chúng đều là các thiết bị tính toán đơn giản (với mạng nơron sinh học đó là các tế bào thần còn với mạng nhân tạo thì đơn giản hơn nhiều) được liên kết chặt chẽ với nhau. Thứ hai, các liên kết giữa các nơron quyết định chức năng hoạt động của mạng.

Mạng nơron, được xem như hoặc là mô hình liên kết (connectionist model), hoặc là mô hình phân bố song song (parallel-distributed model) và có các thành phần phân biệt sau đây:

1) Tập các đơn vị xử lý;
2) Trạng thái kích hoạt hay đầu ra của đơn vị xử lý;
3) Liên kết giữa các đơn vị, mỗi liên kết được xác định bởi một trọng số w_{ji} cho ta biết hiệu ứng mà tín hiệu của đơn vị j có trên đơn vị i;
4) Luật lan truyền quyết định cách tính tín hiệu ra của đơn vị từ đầu vào của nó;
5) Hàm kích hoạt, xác định mức độ kích hoạt khác dựa trên mức độ kích hoạt hiện tại;
6) Đơn vị điều chỉnh (độ lệch - bias) của mỗi đơn vị;
7) Phương pháp thu thập thông tin (luật học – learning rule);
8) Môi trường hệ thống có thể hoạt động.

1.1.1 Đơn vị xử lý

Một đơn vị xử lý, cũng được gọi là một nơron hay một nút (node), thực hiện công việc rất đơn giản: nhận tín hiệu vào từ các đơn vị khác hay một nguồn bên ngoài và sử dụng chúng để tính tín hiệu ra sẽ được lan truyền sang các đơn vị khác.



Hình 2. Đơn vị xử lý

trong đó:

x_i *:* các đầu vào của đơn vị thứ *j*,

w_{ji} : hệ số nối tới đơn vị thứ *j*,

θ_j : độ lệch đối với đơn vị thứ *j*,

a_j : tổng thứ *j* của đầu vào mạng (*net input*), tương ứng với đơn vị thứ *j*,

z_j : đầu ra của đơn vị thứ *j*,

g(x) : hàm kích hoạt.

Trong một mạng nơron có 3 kiểu đơn vị:

1) Các đơn vị đầu vào (input unit), nhận tín hiệu từ bên ngoài;
2) Các đơn vị đầu ra (output unit), gửi tín hiệu ra bên ngoài;
3) Các đơn vị ẩn (hidden unit), đầu vào (input) và đầu ra (output) của chúng đều nằm trong mạng.

Như được thể hiện trong hình 2, mỗi đơn vị *j* có thể có một hoặc nhiều đầu vào: x_0, x_1, x_2, ..., x_n, nhưng chỉ có một đầu ra z_j. Mỗi đầu vào của một đơn vị có thể là dữ liệu từ bên ngoài mạng, hoặc đầu ra của một đơn vị khác, hoặc đầu ra của chính đơn vị đó.

1.1.2 Hàm xử lý

1.1.2.1 Hàm kết hợp

Mỗi đơn vị trong mạng nơron kết hợp các tín hiệu đưa vào nó thông qua các liên kết với các đơn vị khác, sinh ra một giá trị gọi là *net input*. Hàm thực hiện nhiệm vụ này gọi là hàm kết hợp, được định nghĩa bởi một luật lan truyền cụ thể. Trong phần lớn các mạng nơron, giả sử rằng mỗi đơn vị cung cấp một đầu vào cho đơn vị mà nó có liên kết. Tổng đầu vào đơn vị *j* đơn giản chỉ là tổng theo trọng số của các đầu ra riêng lẻ từ các đơn vị kết nối tới nó cộng thêm ngưỡng hay độ lệch θ_j:

$$a_j = \sum_{i=1}^{n} w_{ij} x_i + \theta_j$$

Trường hợp $w_{ji} > 0$, nơron được coi là ở trong trạng thái kích thích. Ngược lại khi $w_{ji} < 0$, nơron được coi là ở trạng thái kiềm chế. Chúng ta gọi đơn vị với luật lan truyền như trên là đơn vị tổng (sigma unit).

Trong một vài trường hợp người ta cũng có thể sử dụng các luật lan truyền phức tạp hơn. Một trong số đó là luật tổng – tích (sigma-pi rule), có dạng sau:

$$a = \sum_{i=1}^{n} w_{ji} \prod_{k=1}^{m} x_{ik} + \theta_j$$

Rất nhiều hàm kết hợp sử dụng "độ lệch" để tính *net input* tới đơn vị. Đối với một đơn vị đầu ra tuyến tính, thông thường, độ lệch θ_j được chọn là hằng số và trong bài toán xấp xỉ đa thức $\theta_j = 1$.

1.1.2.2 Hàm kích hoạt

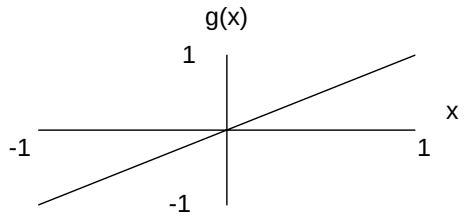
Phần lớn các đơn vị trong mạng nơron chuyển *net input* bằng cách sử dụng một hàm vô hướng gọi là hàm kích hoạt, nếu kết quả của hàm này là một giá trị gọi là

mức độ kích hoạt của đơn vị. Ngoại trừ khả năng đơn vị đó là một lớp ra, giá trị kích hoạt được đưa vào một hay nhiều đơn vị khác. Các hàm kích hoạt thường bị ép vào một khoảng giá trị xác định, do đó thường được gọi là các hàm bẹp (squashing). Các hàm kích hoạt hay được sử dụng là:

- Hàm đồng nhất (Linear function, Identity function)

$$g(x) = x$$

Nếu coi đầu vào là một đơn vị thì sẽ sử dụng hàm này. Đôi khi một hằng số được nhân với *net input* để tạo ra một hàm đồng nhất.



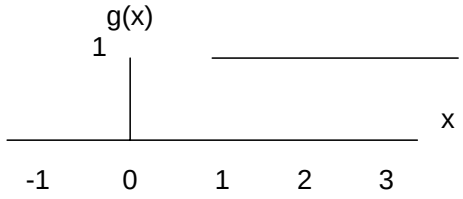
Hình 3. Hàm đồng nhất

- Hàm bước nhị phân (Binary step function, Hard limit function)

Hàm này cũng được biết đến với tên "hàm ngưỡng" (threshold function). Đầu ra của hàm này được giới hạn vào một trong hai giá trị.

$$g(x) = \begin{cases} 1, & if\,(x \geq \theta) \\ 0, & if\,(x, \theta) \end{cases}$$

Dạng hàm này được sử dụng trong các mạng chỉ có một lớp. Trong hình vẽ sau θ được chọn bằng 1.

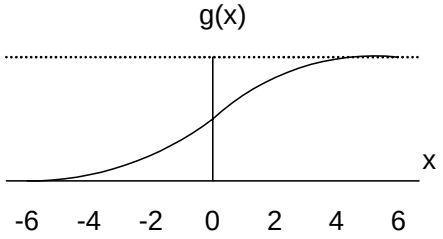


Hình 4. Hàm bước nhị phân

- Hàm sigmoid (Sigmoid function)

$$g(x) = \frac{1}{1+e^{-x}}$$

Hàm này đặc biệt thuận lợi khi sử dụng cho các mạng huấn luyện, bởi nó dễ lấy đạo hàm, do đó có thể giảm đáng kể tính toán trong quá trình huấn luyện. Hàm này được ứng dụng cho các chương trình ứng dụng mà các đầu ra mong muốn rơi vào khoảng [0,1].

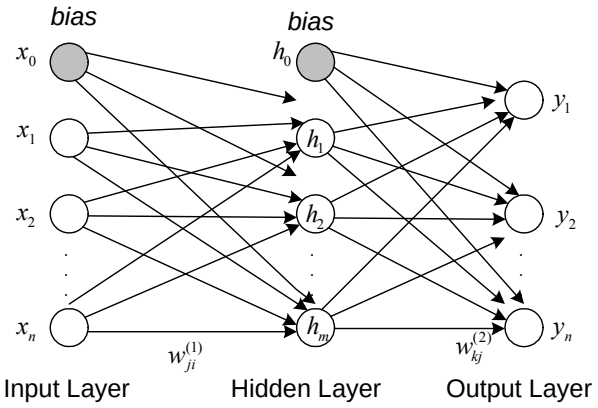


Hình 5. Hàm Sigmoid

1.1.3 Hình trạng mạng

Hình trạng của mạng được định nghĩa bởi: số lớp (layer), số đơn vị trên mỗi lớp, và sự liên kết giữa các lớp như thế nào. Các mạng về tổng thể được chia thành hai loại dựa trên cách thức liên kết các đơn vị.

1.1.3.1 Mạng truyền thẳng

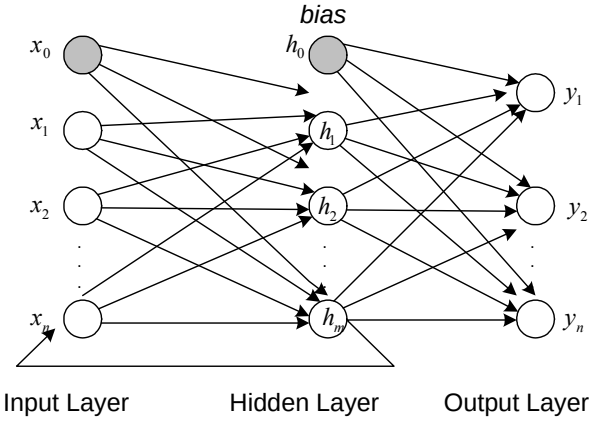


Hình 6. Mạng nơron truyền thẳng nhiều lớp

Dòng dữ liệu giữa đơn vị đầu vào và đầu ra chỉ truyền thẳng theo một hướng. Việc xử lý dữ liệu có thể mở rộng ra thành nhiều lớp, nhưng không có các liên kết phản hồi. Điều đó có nghĩa là không tồn tại các liên kết mở rộng từ các đơn vị đầu ra tới các đơn vị đầu vào trong cùng một lớp hay các lớp trước đó.

1.1.3.2 Mạng hồi quy

Trong mạng hồi quy, tồn tại các liên kết ngược. Khác với mạng truyền thẳng, thuộc tính động của mạng hồi quy có được từ các liên kết ngược như vậy có ý nghĩ rất quan trọng. Trong một số trường hợp, các giá trị kích hoạt của các đơn vị trải qua quá trình nới lỏng (tăng giảm số đơn vị và thay đổi các liên kết) cho đến khi mạng đạt đến trạng thái ổn định và các giá trị kích hoạt không thay đổi nữa. Trong các ứng dụng khác mà cách chạy tạo thành đầu ra của mạng thì những sự thay đổi các giá trị kích hoạt là đáng quan tâm.



Hình 7. Mạng nơron hồi quy

1.2 Mạng nơron trong khai phá dữ liệu

1.2.1 Khai phá dữ liệu

Mục đích quan trọng của công việc khai phá dữ liệu là để hiểu được ý nghĩa về nội dung sâu sắc bên trong các bộ dữ liệu lớn. Thông thường, các giải pháp phổ biến đạt được mục đích này đều liên quan đến phương pháp học máy để xây dựng một cách

quy nạp các mô hình dữ liệu trong tương lai. Mạng nơron được áp dụng trong hàng loạt các ứng dụng khai phá dữ liệu trong tài chính ngân hàng, dự đoán tỷ giá quy đổi, lập lịch cho tàu con thoi, ... Các thuật toán học mạng nơron đã được ứng dụng thành công trong một số lĩnh vực liên quan đến học có giám sát và không giám sát. Hướng phát triển mới học mạng nơron là cải tiến quá trình học cho dễ hiểu hơn và thời gian học nhanh hơn, mà đây là vấn đề thường xuyên được đề đến cập đầu tiên trong khai phá dữ liệu [12].

Học quy nạp là một trong những phương pháp phổ biến trong khai phá dữ liệu bởi vì nó xây dựng được các mô hình diễn tả việc thu thập dữ liệu cho phép hiểu thấu đáo bên trong dữ liệu đó. Tuỳ theo công việc cụ thể mà có thể sử dụng phương pháp học có giám sát hoặc học không giám sát các mô hình. Trong cả hai trường hợp học có giám sát và không giám sát, các thuật toán học là khác nhau thông qua cách thể hiện các mô hình khác nhau. Các phương pháp học mạng nơron thể hiện các giải pháp học dùng tham số thực trong một mạng gồm các đơn vị xử lý đơn giản. Các kết quả nghiên cứu chứng tỏ rằng mạng nơron là công cụ khá hiệu quả trong khai phá dữ liệu, đặc biệt đối với khuynh hướng học theo quy nạp.

Chúng ta lướt qua nội dung sơ bộ về thuật toán có khuynh hướng quy nạp trong khai phá dữ liệu, mà cụ thể là thuật toán học theo quy nạp. Cho một tập cố định các ví dụ huấn luyện, thuật toán học có khuynh hướng quy nạp quyết định các thông số của một mô hình bằng cách tính toán lặp đi lặp lại theo dạng của mô hình đó. Có hai xu hướng xác định hướng ưu tiên của thuật toán. Không gian giả thuyết giới hạn đề cập đến ràng buộc thuật toán học thay cho giả thuyết mà nó có thể tạo ra. Ví dụ, không gian giả thuyết của một bộ cảm ứng được giới hạn bởi các hàm tuyến tính đặc biệt. Hướng ưu tiên của thuật toán đề cập đến việc sắp xếp ưu tiên thay cho các mô hình kết hợp trong không gian giả thuyết. Ví dụ, phần lớn các thuật toán học ban đầu cố gắng đáp ứng một giả thuyết đơn giản để đưa ra một tập huấn luyện sau đó khảo sát dần các giả thuyết phức tạp cho đến khi thuật toán tìm được hướng có thể chấp nhận được.

Mạng nơron là phương pháp học khá phổ biến không chỉ vì lớp các giả thuyết do chúng có thể đại diện, mà đơn giản là vì chúng đem lại giả thuyết khái quát hơn so với các thuật toán cạnh tranh khác. Một số công trình nghiên cứu đã xác định rằng có một số lĩnh vực mà trong đó mạng nơron cung cấp dự đoán chính xác.

Giả thuyết được thể hiện trong mạng nơron huấn luyện bao gồm:

(1) Hình trạng của mạng;
(2) Hàm chuyển đổi dùng cho các đơn vị ẩn và đơn vị đầu ra;
(3) Các tham số giá trị thực liên quan đến kết nối mạng (trọng số kết nối).

Các giả thuyết là rất đa dạng. Đầu tiên, các mạng tiêu biểu có hàng trăm hàng nghìn các tham số giá trị thực, các tham số mã hoá có liên quan đến đầu vào x và giá trị đích y. Mặc dù, mã hoá các tham số của loại này không khó, song sự chênh lệch số lượng các tham số trong mạng có thể làm cho việc hiểu chúng trở nên khó khăn hơn. Thứ hai, trong mạng đa lớp, các tham số có thể có mối quan hệ không tuyến tính, không đơn điệu giữa đầu vào và đầu ra. Vì vậy thường làm cho nó không thể xác định rõ sự ảnh hưởng của các đặc điểm đưa ra trong các giá trị mong muốn.

Quá trình học của phần lớn các phương pháp học mạng nơron đều liên quan đến việc dùng một số phương pháp tối ưu cơ bản gradient để điều chỉnh các tham số mạng. Giống như các phương pháp tối ưu, học mạng nơron thực hiện lặp đi lặp lại hai bước cơ bản: tính toán gradient của hàm lỗi và điều chỉnh các tham số mạng theo hướng tiến bộ bởi gradient. Việc học có thể là rất chậm chạp và tuỳ thuộc các phương pháp khác nhau bởi vì thủ tục tối ưu thường bao gói một số lượng lớn các bước nhỏ và chi phí tính toán gradient cho mỗi bước có thể là rất lớn.

Hướng mong muốn của phương pháp học mạng nơron là tìm ra các thuật toán học tuyến tính, có nghĩa là chúng được cập nhập các giả thuyết sau mỗi ví dụ. Vì các tham số được cập nhập đều đặn, các thuật toán học mạng nơron tuyến tính thường nhanh hơn thuật toán xử lý theo khối. Đây là một đặc điểm có lợi cho tập dữ liệu

lớn. Một giải pháp được gọi là tốt nếu như mô hình có thể được phát hiện chỉ trong một lần duyệt qua một tập dữ liệu lớn. Lý do này, chứng tỏ thời gian huấn luyện của các phương pháp học mạng nơron là chấp nhận cho việc khai phá dữ liệu.

1.2.2 Khai phá dữ liệu tài chính

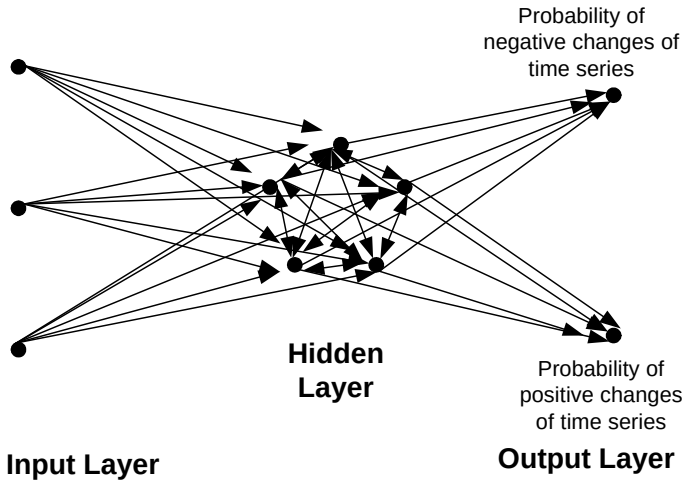
Theo đánh giá của Rao vào năm 1993 [4]: "Các kết quả đáng chú ý trong mạng nơron trong suốt mấy năm qua thu được từ việc tổng quát hoá bằng hệ học các ví dụ (trường hợp) cơ bản. Kết quả cũng cho thấy là các mạng có khả năng hình thành một độ xấp xỉ đóng tuỳ ý cho bất kỳ ánh xạ không tuyến tính liên tục".

Trong thực tế, mạng nơron được dùng khá phổ biến trong lĩnh vực tài chính. Những công bố từ nhiều bài báo khoa học xung quanh các ví dụ dùng mạng nơron đơn giản, hồi quy, và tiền xử lý dữ liệu cho thấy sử dụng mạng nơron là có lợi hơn nhiều so với các phương pháp khác. Các tác giả [4] chỉ ra rằng: (1) dùng mạng nơron đơn giản rất thích hợp đối với các hệ thống tài chính thương mại; (2) các hệ thống mạng nơron mờ lại thích hợp cho việc xây dựng mô hình tài chính và dự báo; (3) dùng mạng nơron hồi quy trong tài chính để dự đoán lỗi trong kinh doanh... Tiền xử lý cũng được dùng phổ biến trong tổng quát hoá cũng như trong các ứng dụng mạng nơron trong tài chính. Một hướng chung của tiền xử lý là dùng hàm sigmoid và các cách biến đổi khác nhau làm thay đổi các giá trị lớn hơn 1. Mục đích của công việc đó là nhằm tăng tốc độ huấn luyện mạng. Ví dụ, đối với bài toán dự báo giá cổ phiếu, dùng mạng nơron gặp ba thiếu sót: (1) khả năng giải thích chưa thật tốt; (2) khó phù hợp với thói quen dùng các quan hệ logic; (3) khó khăn khi chấp nhận dữ liệu bị thiếu hụt. Tuy nhiên, mạng nơron vẫn khẳng định những lợi điểm của nó như tốc độ đáp ứng nhanh, chấp nhận sự phức tạp, tương đối độc lập với đặc tính chuyên môn của lĩnh vực ứng dụng, tính linh hoạt và cô đọng.

Các mạng nơron hồi quy đã được dùng trong một số ứng dụng tài chính khá điển hình [4]. Đặc biệt, mạng nơron hồi quy đã được phát triển để dự đoán tỷ giá hoán đổi ngoại tệ hàng ngày với sự kết hợp với các kỹ thuật khác. Dùng mạng nơron hồi

quy vì hai lý do. Một là, mô hình cho phép xác định các quan hệ tạm thời cùng với chuỗi thời gian bằng cách duy trì một khoảng trạng thái. Hai là, các luật giải thích dễ hiểu có thể được rút ra từ mạng hồi quy đã được huấn luyện. Cụ thể, người ta dùng mạng nơron gồm:

- Ba nơron đầu vào. Nơron đầu tiên được dùng để thể hiện đặc trưng của chuỗi dữ liệu theo thời gian x*(t)*, x*(t-1*), x*(t-2)*, ..., x*(t-k)* với k là các khoảng thời gian. Các đầu vào sau được dùng cho hai nơron đầu vào, tăng cường trong quá trình huấn luyện.
- Một lớp ẩn với năm liên kết các nơron đầy đủ.
- Hai nơron ra. Nơron đầu tiên được huấn luyện để dự đoán khả năng của thay đổi khẳng định (positive change), và nơron thứ hai được huấn luyện để dự đoán khả năng của phủ định (negative change).



Hình 8. Một ví dụ dùng mạng nơron hồi quy trong dự báo tài chính

Sự mô tả cô đọng, coi như một chỉ số, được dùng để giữ cho mạng nơron nhỏ hơn. Năm 1997 Kohonen sử dụng kỹ thuật SOM để lấy ra chỉ số. Đây là một quá trình học không giám sát, học sự phân bố của một tập các mẫu không có bất kỳ sự phân lớp thông tin nào. Chi tiết thuật toán SOM và cách phân lớp thông tin cũng như ứng dụng của SOM vào một bài toán cụ thể sẽ là chủ đề chính của bản luận văn này và sẽ được đề cập chi tiết hơn trong chương 2.

Các bước trích luật từ mạng nơron hồi quy là:

Bước 1: Phân cụm các giá trị kích hoạt tình trạng của các nơron hồi quy.

Bước 2: Xác định các tình trạng cho các cụm.

Bước 3: Chèn các biến đổi giữa các cụm trong các biểu tượng đầu vào thích hợp.

Kết quả của thuật toán trên là một tập các luật dự đoán được gán bằng các biểu tượng có nghĩa được lấy từ một chuỗi thời gian. Hiểu cách hoạt động của mạng nơron có thể rút ra được các luật. Dưới đây là bảng kết quả của thuật toán.

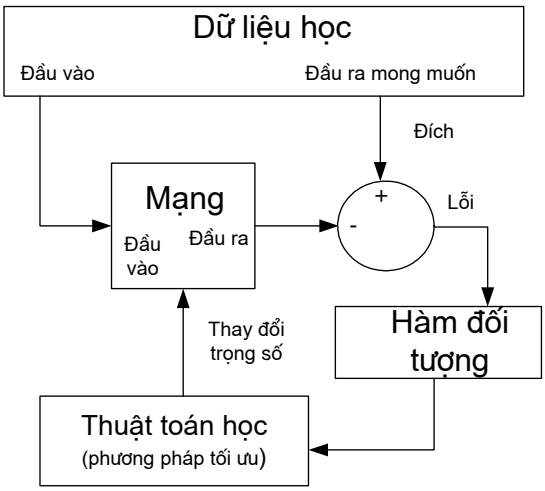
Tập các luật	**Các luật dự báo được rút ra**
1	Luật 1. Nếu thay đổi lần cuối trong chuỗi là phủ định, thì thay đổi tiếp theo sẽ là khẳng định. Luật 2. Nếu thay đổi lần cuối trong chuỗi là khẳng định, thì thay đổi tiếp theo sẽ là phủ định
2	Luật 1. Nếu thay đổi lần cuối trong chuỗi là phủ định, thì thay đổi tiếp theo sẽ là khẳng định. Luật 2. Nếu thay đổi lần cuối trong chuỗi là khẳng định, thì thay đổi tiếp theo sẽ là khẳng định
3	Luật 1. Nếu thay đổi lần cuối trong chuỗi là khẳng định, thì thay đổi tiếp theo sẽ là khẳng định. Luật 2. Nếu thay đổi lần cuối trong chuỗi là phủ định và các lần thay đổi trước không phải là khẳng định, thì thay đổi tiếp theo sẽ là khẳng định

1.3 Các phương pháp học sử dụng mạng nơron

Chức năng của mạng nơron được quyết định bởi các nhân tố như: hình trạng mạng (số lớp, số đơn vị trên mỗi tầng, và cách mà các lớp liên kết với nhau) và các trọng số của các liên kết nội tại trong mạng. Hình trạng của mạng thường là cố định còn các trọng số được quyết định bởi một thuật toán huấn luyện. Tiến trình điều chỉnh các trọng số để mạng "nhận biết" được quan hệ giữa đầu vào với đích (đầu ra) mong muốn được gọi là học hay huấn luyện. Thuật toán học được chia làm hai

nhóm chính: Học có giám sát (supervised learning) và học không có giám sát (unsupervised learning).

1.3.1 Học có giám sát



Hình 9. Mô hình 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. Các cặp mẫu được cung cấp bởi "thầy", hay bởi hệ thống trên đó mạng hoạt động. Mục đích là xây dựng mạng để đối với đầu vào trong tập huấn luyện thì kết quả đầu ra của mạng cho đúng đầu ra mong muốn mà để làm được điều đó phải điều chỉnh dần mạng do tồn tại sự khác biệt giữa đầu ra thực tế và đầu ra mong muốn (đã được biết trước) .Sự khác biệt này được thuật toán học sử dụng để điều chỉnh các trọng số trong mạng.Việc điều chỉnh các trọng số như vậy thường được mô tả như một bài toán xấp xỉ số - cho dữ liệu huấn luyện bao gồm các cặp (mẫu đầu vào x, và một đích tương ứng t), mục đích là tìm hàm f*(x)* thoả mãn tất cả các mẫu học đầu vào.

Thuật toán BBP (Boosting-Based Perceptron)

Thuật toán BBP (Jackson & Carven, 1996) [12] là thuật toán học có giám sát được phát triển trên cơ sở thuật toán AdoBoost (Freund & Schapire, 1995) [11], là

phương pháp học giả thuyết nổi (hypothesis – boosting). Thuật toán học một tập các giả thuyết và sau đó kết hợp chúng vào một giả thuyết tổng thể. Thuật toán giả thuyết nổi là thuật toán kết hợp cho ra các giả thuyết bằng thuật toán học yếu (weak learning) trong một giả thuyết mạnh. Giả thuyết yếu là giả thuyết mà dự đoán chỉ tốt hơn không đáng kể so với phỏng đoán ngẫu nhiên, ngược lại giả thuyết mạnh là giả thuyết mà khi dự đoán cho kết quả chính xác cao.

Thuật toán BBP được dùng nhiều cho các ứng dụng khai phá dữ liệu vì nó có những đóng góp đáng kể trong các mạng học. Phương pháp học này không giống như các phương pháp mạng nơron truyền thống là vì nó không liên quan đến việc huấn luyện bằng một phương pháp tối ưu dựa trên gradient (gradient-based). Tuy nhiên do các giả thuyết học là các bộ cảm ứng vì vậy chúng ta xem nó là một phương pháp mạng nơron.

Ý tưởng chính của phương pháp là thêm các đơn vị đầu vào mới cho một giả thuyết học, dùng phân bố xác suất trên toàn bộ tập huấn luyện để chọn lọc ra một đầu vào thích hợp. Vì thuật toán thêm các đầu vào có trọng số cho các giả thuyết nên độ phức tạp của các giả thuyết có thể kiểm soát được dễ dàng.

Các đầu vào được kết hợp chặt chẽ trong một giả thyết tương ứng với các hàm Boolean có ánh xạ đến {-1,+1}. Mặt khác, các đầu vào là các đơn vị nhị phân có một kích hoạt hoặc –1 hoặc +1. Các đầu vào có thể tương ứng với các giá trị Boolean hoặc chúng có thể tương đương với các giá trị thử nghiệm định danh hay số (ví dụ, *màu = đỏ,* $x_1>0.8$) hoặc các kết hợp logic các giá trị (ví dụ, *[màu = đỏ]* $\wedge$*[hình = tròn]*). Hơn nữa, thuật toán cũng có thể kết hợp một đầu vào tương ứng hàm *true*. Trọng số gắn với một đầu vào tương xứng với *ngưỡng* của bộ cảm ứng.

Trong mỗi lần lặp, đầu vào được lựa chọn từ một tập các khả năng có thể và thêm vào các giả thuyết. Thuật toán BBP đo độ tương quan của mỗi đầu vào với hàm mục tiêu bằng cách học, và sau đó tìm đầu vào có sự tương quan lớn nhất. Sự tương

quan giữa khả năng chọn lựa và hàm mục tiêu được thay đổi qua mỗi lần lặp do được điều chỉnh bằng cách thay đổi một phân bố qua tập huấn luyện.

Ban đầu, thuật toán BBP giả thiết có phân bố đồng đều trên tập huấn luyện. Khi lựa chọn đầu vào đầu tiên, BBP ấn định mức độ quan trọng ngang nhau cho mọi trường hợp trong tập huấn luyện. Mỗi khi một đầu vào được thêm vào, phân bố được điều chỉnh theo hướng là trọng số lớn hơn được đưa tới các ví dụ mà đầu vào không dự đoán chính xác. Điều đó có nghĩa là, thuật toán hướng người học tập trung chú ý vào các ví dụ mà giả thuyết hiện tại không giải thích đúng.

Thuật toán dừng việc thêm trọng số đầu vào cho các giả thuyết sau khi đã thực hiện lặp một số lần đã được xác định trước, gặp tình huống không còn lỗi đối với tập huấn luyện. Vì chỉ có một đầu vào được thêm vào mạng trong mỗi lần lặp, kích thước của bộ cảm ứng cuối cùng có thể kiểm soát theo bởi số lần lặp. Giả thuyết trả về của BBP là một bộ cảm ứng có trọng số kết hợp với mỗi đầu vào là một hàm lỗi của đầu vào. Bộ cảm ứng dùng hàm dấu để xác định lớp trả về:

$$sign(x) = \begin{cases} 1 \text{ if } x > 0 \\ -1 \text{ if } x <= 0 \end{cases}$$

Thuật toán BBP có hai hạn chế [12]:

- Một là, nó được thiết kế cho các nhiệm vụ học phân lớp nhị phân. Thuật toán có thể được áp dụng cho vấn đề học đa lớp bằng cách mỗi lớp học một bộ cảm ứng.
- Hai là, nó giả sử đầu vào là các hàm boolean, cho nên các lĩnh vực áp dụng có giá trị thực cần phải xử lý bằng cách rời rạc hóa các giá trị như đã nói ở trên.

Thuật toán

Input: Tập *S* gồm *m* ví dụ, tập đầu vào *C* có ánh xạ tới {-1,+1}, số các tương tác *T*

Output: Hàm h(x)

Nội dung thuật toán:

for all $x \in S$

/* Phân bố ban đầu là như nhau */

$D_1(x) := 1/m$

for $t:=1$ **to** T **do**

/*Thêm giả thuyết */

$h_t := \text{argmax}_{ci \in C} | E_{Dt} [f(x).c_i(x)] |$

/* Xác định lỗi */

$\varepsilon_t := 0$

for all $x \in S$

if $h_t(x) \neq f(x)$ **then** $\varepsilon_t := \varepsilon_t + D_t(x)$

/* Cập nhập lại phân bố */

$\beta_t := \varepsilon_t / (1-\varepsilon_t)$

for all $x \in S$

if $h_t(x) = f(x)$ **then**

$D_{t+1}(x) := \beta_t D_t(x)$

else

$D_{t+1}(x) := D_t(x)$

/* Cập nhập lại */

$Z_t = \sum_x D_{t+1}(x)$

for all $x \in S$

$D_{t+1}(x) := D_{t+1}(x)/Z_t$

Return: h(x) = $sign\left(\sum_{i=1}^{T} -\ln(\beta_i) h_i(x) \right)$

1.3.2 Học không giám sát

Học mạng nơron không giám sát là cách học không có phản hồi từ môi trường để chỉ ra rằng đầu ra của mạng là đúng như thế nào. 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 giám sát, các đích trùng với đầu vào. Nói một cách khác, học không giám sát thực hiện một công việc tương tự như một mạng tự nhiên liên hợp, cô đọng thông tin từ dữ liệu vào. Một số thuật toán học không giám sát được trình bày chi tiết trong chương 2.

1.4 Kết luận chương 1

Chương này luận văn trình bày những nội dung chính yếu về cấu trúc mạng nơron gồm các đơn vị xử lý; trạng thái kích hoạt; các liên kết, luật lan truyền; hàm kích hoạt; độ lệch; luật học và môi trường hệ thống có thể hoạt động được. Về tổng thể, hình trạng mạng nơron được chia làm hai loại là mạng nơron truyền thẳng và mạng nơron hồi quy. Các thuật toán học mạng nơron đã làm cho quá trình học cho dễ hiểu hơn và chi phí thời gian học ít hơn, đây là vấn đề thời sự trong khai phá dữ liệu.

Thuật toán học mạng nơron được chia làm hai nhóm chính đó là học có giám sát và học không có giám sát. Trong đó thuật toán BBP là thuật toán đặc trưng cho học có giám sát mạng nơron đơn lớp.

CHƯƠNG 2. THUẬT TOÁN SOM VỚI BÀI TOÁN PHÂN CỤM

Như đã trình bày trong chương 1, học không giám sát là một trong hai nhóm học chính của mạng nơron. Học không giám sát là cách học không có phản hồi từ môi trường. Chương này sẽ giới thiệu một thuật toán học không giám sát phổ biến nhất đó là học ganh đua và sau đó cũng sẽ giới thiệu một thuật toán sử dụng thuật toán ganh đua và qua một quá trình tự tổ chức (self - organizing) sắp xếp đầu ra cho bài toán phân cụm.

2.1 Các phương pháp phân cụm

Mục đích của phân cụm là làm giảm kích thước dữ liệu bằng cách phân loại hoặc nhóm các thành phần dữ liệu giống nhau. Tồn tại một số kỹ thuật phân cụm điển hình [9]:

- *Phân cụm theo phân cấp* được thực hiện theo hai phương pháp. Phương pháp đầu tiên là hợp nhất các cụm dữ liệu nhỏ hơn thành các cụm lớn hơn theo một vài tiêu chuẩn (từ dưới lên). Phương pháp thứ hai đó là làm ngược lại, chia các cụm lớn hơn thành các cụm nhỏ (từ trên xuống). Kết quả của cả hai phương pháp là một cây phân cụm (được gọi là dendrogram) để chỉ ra các cụm có liên quan.
- *Phân cụm bộ phận* phân tích dữ liệu vào một tập các cụm rời rạc. Thuật toán phân cụm tối thiểu một hàm chuẩn. Độ chuẩn này thường liên quan đến viêc tối thiểu một vài độ đo giống nhau trong tập ví dụ với mỗi cụm, trong khi đó việc tối đa các cụm là không giống nhau. Đã tồn tại một vài phương pháp phân cụm bộ phận, mà điển hình nhất là dùng thuật toán K thành phần chính.
- *Phân cụm dựa trên mật độ* (density-base) là các phương pháp phân cụm dựa vào liên kết và các hàm mật độ.
- *Phân cụm dựa trên lưới* (grid-base) sử dụng cấu trúc nhân đa mức loang dần các cụm.

- *Phân cụm dựa trên mô hình* (model-base) được tiến hành bằng cách dựng lên một mô hình giả định cho mỗi cụm và ý tưởng là chọn mô hình tốt nhất trong số các mô hình của các cụm.
- Các phương pháp khác như là *tiếp cận mạng nơron* và *học ganh đua*.

Các kỹ thuật phân cụm đã và đang được áp dụng trong nhiều vấn đề nghiên cứu. Ví dụ như, trong lĩnh vực y tế: phân loại bệnh, cách chữa bệnh, hoặc triệu chứng bệnh; trong lĩnh vực tài chính đặc biệt là nghiên cứu thị trường, lựa chọn quỹ đầu tư, ước định rủi ro tín dụng, ...; trong xử lý ảnh, nhận dạng mẫu, ...; trong web như phân lớp tài liệu, phân cụm dữ liệu Weblog để phát hiện ra các nhóm có mẫu truy cập giống nhau,...

2.2 Dùng mạng nơron trong phân cụm

2.2.1 Học ganh đua

Học không giám sát liên quan đến việc dùng các phương pháp quy nạp để phát hiện tính quy chuẩn được thể hiện trong tập dữ liệu. Mặc dù có rất nhiều thuật toán mạng nơron cho học không giám sát, trong đó có thuật toán học ganh đua (competitive learning, Rumelhart & Zipser, 1985) [12]. Học ganh đua có thể coi là thuật toán học mạng nơron không giám sát thích hợp nhất trong khai phá dữ liệu, và nó cũng minh họa cho sự phù hợp của các phương pháp học mạng nơron một lớp.

Nhiệm vụ học xác định bởi học ganh đua là sự phân chia một ví dụ huấn luyện cho trước vào trong một tập các cụm dữ liệu. Các cụm dữ liệu sẽ thể hiện các quy tắc biểu diễn trong tập dữ liệu như các minh hoạ giống nhau được ánh xạ vào trong các lớp giống nhau.

Biến thể của học ganh đua mà chúng ta xét ở đây đôi khi được gọi là học ganh đua đơn điệu, liên quan đến việc học trong mạng nơron một lớp. Các đơn vị đầu vào trong mạng có các giá trị liên quan đến lĩnh vực đang xét, và k đơn vị đầu ra thể hiện k lớp ví dụ đầu vào được phân cụm.

Giá trị đầu vào cho mỗi đầu ra trong phương pháp này là một tổ hợp tuyến tính của các đầu vào:

$$net_j = \sum_i w_{ji} x_i$$

Trong đó, x_i là đầu vào thứ i, và w_{ji} là trọng số liên kết đầu vào thứ i với đầu ra thứ j. Tên thuật toán xuất phát từ việc quyết định số các lớp ẩn. Đơn vị đầu ra có giá trị đầu vào lớn nhất được coi là chiến thắng, và kích hoạt đó được coi bằng 1, còn các kích hoạt khác của đầu ra được cho bằng 0.

$$a_j = \begin{cases} 1 & if \sum_i w_{ji} x_i > \sum_i w_{hi} x_i \forall h \neq j \\ 0 & else \end{cases}$$

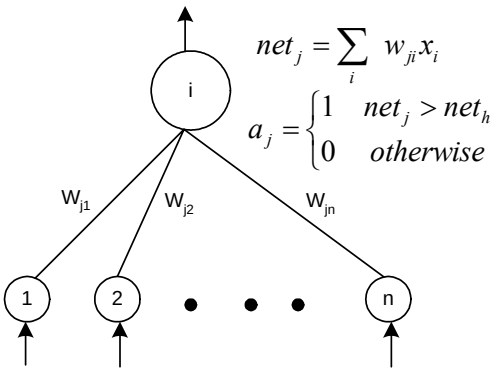
Quá trình huấn luyện cho học ganh đua liên quan đến hàm chi phí:

$$C = \frac{1}{2} \sum_j \sum_i a_j (x_i - w_{ji})^2$$

với a_j là kích hoạt của đầu ra thứ j, x_i là đầu vào thứ i, và w_{ji} là trọng số từ đầu vào thứ i với đầu ra thứ j. Luật cập nhập các trọng số là:

$$\Delta w_{ji} = -\alpha \partial C \partial w_{ji} = \alpha a_j (x_i - w_{ji})$$

với α là hệ số tỷ lệ học.

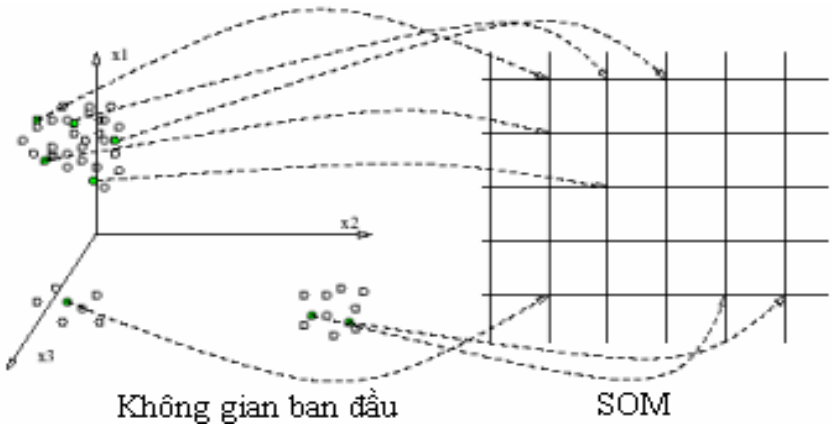


Hình 10. Đơn vị xử lý ganh đua

Ý tưởng chính của học ganh đua là đối với mỗi đầu ra là lấy ra "độ tin cậy" cho tập con các ví dụ huấn luyện. Chỉ một đầu ra là chiến thắng trong số ví dụ đưa ra, và vectơ trọng số cho đơn vị chiến thắng được di chuyển về phía vectơ đầu vào. Giống như quá trình huấn luyện, vectơ trọng số của mỗi đầu ra di chuyển về phía trung tâm của các ví dụ. Huấn luyện xong, mỗi đầu ra đại diện cho một nhóm các ví dụ, và vectơ trọng số cho các đơn vị phù hợp với trọng tâm của các nhóm.

Học ganh đua có liên quan mật thiết với phương pháp thống kê nổi tiếng như là phương pháp phân cụm K thành phần chính. Khác nhau cơ bản giữa hai phương pháp là học ganh đua là phương pháp trực tuyến, nghĩa là trong suốt quá trình học nó cập nhập trọng số mạng sau mỗi ví dụ được đưa ra, thay vì sau tất cả các ví dụ được đưa ra như được làm trong phương pháp phân cụm K thành phần chính. Học ganh đua phù hợp với các tập dữ liệu lớn, vì các thuật toán trực tuyến thường có giải pháp nhanh hơn trong mọi trường hợp.

2.2.2 Thuật toán SOM



Hình 11. Không gian ban đầu và SOM

Thuật toán SOM (Self–Organizing Map) được giáo sư Teuvo Kohonen phát triển [10,11,13,15] vào những năm 80, là một công cụ rất thích hợp trong khai phá dữ liệu [9]. SOM thực hiện một ánh xạ làm giảm kích thước của tập huấn luyện. Ánh

xạ sinh ra hàm phân bố xác suất của dữ liệu và linh hoạt với dữ liệu còn thiếu. Nó được giải thích dễ dàng, đơn giản và quan trọng nhất là dễ hình dung. Mô phỏng dữ liệu đa chiều là một lĩnh vực áp dụng chính của SOM.

SOM là một kỹ thuật mạng nơron truyền thẳng sử dụng thuật toán học không giám sát (học ganh đua) và qua quá trình ”tự tổ chức”, sắp xếp đầu ra cho một thể hiện hình học của dữ liệu ban đầu [10,11].

Thuật toán

Xét một tập dữ liệu là các vectơ trong không gian *n* chiều:

$$x = \left[x_1, x_2, ..., x_n\right]^T \in \Re^n$$

Thông thường SOM gồm M nơron nằm trong một lưới (thường có kích thước 2 chiều). Một nơron thứ *i* là một vectơ mẫu có kích thước p:

$$m_i = \left[m_{i1}, ..., m_{ip}\right]^T \in \Re^p$$

Các nơron trong lưới có liên kết đến các nơron lân cận bằng một quan hệ láng giềng. Các láng giềng liền kề là các nơron lân cận tuỳ theo bán kính lân cận của nơron thứ *i*.

$N_i(d) = \left\{j, d_{i,j} \le d\right\}$ với *d* là bán kính lân cận

Các nơron lân cận tuỳ thuộc vào bán kính, được sắp xếp trong lưới theo hình chữ nhật hoặc hình lục giác. Số các lân cận xác định trọng tâm của ma trận kết quả, có ảnh hưởng đến độ chính xác và khả năng sinh ma trận của SOM.

Hình 12. Các lân cận

Trong thuật toán SOM, các quan hệ hình học và số các nơron là cố định ngay từ đầu. Số lượng nơron thường được chọn đủ lớn nếu có thể, bằng cách điều khiển kích thước lân cận cho phù hợp. Nếu kích thước lân cận được lựa chọn là phù hợp thì ma trận không bị mất mát thông tin nhiều ngay cả khi số các nơron vượt quá số các vectơ đầu vào. Tuy nhiên, nếu kích thước của ma trận tăng, ví dụ đến mười nghìn nơron thì quá trình huấn luyện trở nên nặng nề vì việc tính toán sẽ không còn hợp lý cho phần lớn các ứng dụng.

Trước khi huấn luyện các giá trị ban đầu được đưa ra là các vectơ trọng số. SOM là không phụ thuộc nhiều đối với dữ liệu ban đầu (dữ liệu có thể bị thiếu), nhưng thuật toán SOM vẫn hội tụ nhanh. Dùng một trong ba thủ tục khởi tạo điển hình sau :

- Khởi tạo ngẫu nhiên, vectơ trọng số ban đầu được gán giá trị là các giá trị ngẫu nhiên đủ nhỏ.
- Khởi toạ ví dụ, vectơ trọng số ban đầu được gắn với các ví dụ ngẫu nhiên rút ra từ tập dữ liệu.
- Khởi tạo tuyến tính, vectơ trọng số ban đầu được gắn trong một không gian con tuyến tính bởi hai vectơ của tập dữ liệu ban đầu.

Trong mỗi bước huấn luyện, chọn ngẫu nhiên một vectơ ví dụ x trong tập dữ liệu ban đầu. Tính toán khoảng cách giữa x đến tất cả các vectơ mẫu, trong đó c là đơn vị có mẫu gần x nhất gọi là BMU (Best Matching Unit), được xác định như sau:

$$\|x - m_c\| = \min_i \{\|x - m_i\|\}$$

với ||.|| là độ đo khoảng cách.

Sau khi tìm được BMU, vectơ trọng số của SOM được cập nhập lại. Vectơ trọng số của BMU và các lân cận hình thái của nó di chuyển dần đến vectơ trong không gian đầu vào. Thủ tục cập nhập này trải dài theo BMU và các hình trạng lân cận của nó về phía vectơ ví dụ.

SOM cập nhập luật cho vectơ trọng số của đơn vị thứ *i* là:

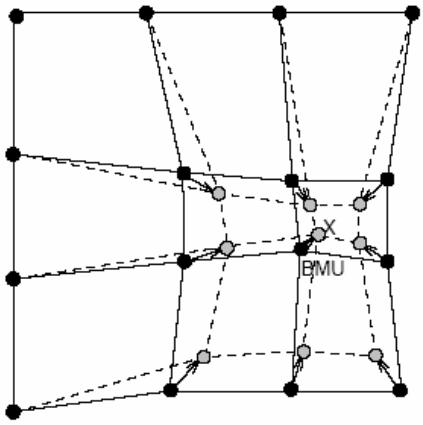
$$m_i(t+1) = m_i(t) + \alpha(t)h_{ci}(t)\left[x - m_i(t)\right]$$

với *t:* là thời gian,

x: vectơ đầu vào ngẫu nhiên rút ra từ tập dữ liệu đầu vào tại thời điểm t,

α(t): hệ số tỷ lệ học,

$h_{ci}(t)$*:* nhân (kernel) lân cận quanh *c* tại thời điểm *t,* là hàm lân cận Gauxơ.



Hình 13 BMU

Nhân lân cận xác định vùng ảnh hưởng mà ví dụ đầu vào có trong SOM. Nhân được thể hiện gồm hai phần: hàm lân cận *h(t,d)* và hàm tỷ lệ học *α(t)*:

$$h_{ci}(t) = h\left(\left\|r_c - r_i\right\|, t\right)\alpha(t)$$

r_c, r_i là các vị trí nơron *i* và *c*.

Hàm lân cận đơn giản nhất đó là hàm nổi bọt: nó gồm toàn bộ lân cận của đơn vị chiến thắng và bằng không nếu ngược lại (hình 14). Ngoài ra, còn có hàm lân cận Gauxơ:

$$h_{ci}(t) = e^{\frac{-\left\|r_c - r_i\right\|^2}{2\sigma^2(t)}}$$

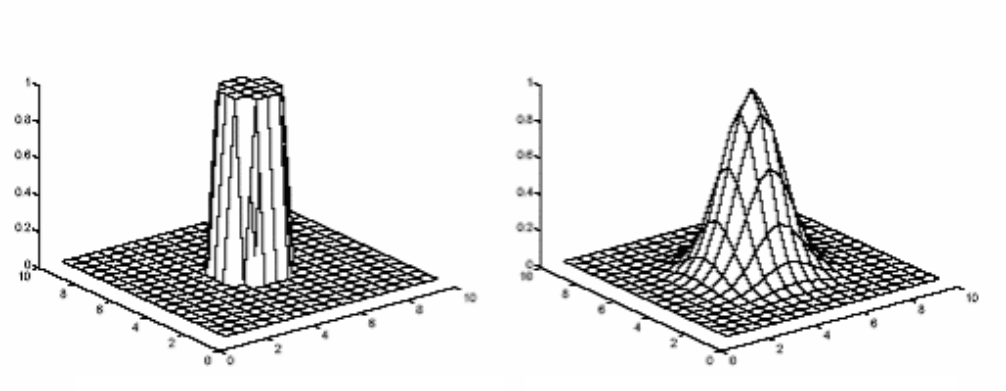
với $\sigma(t)$: là bán kính lân cận.

Hàm lân cận Gauxơ cho ra kết quả tốt hơn, nhưng việc tính toán lại nặng nề hơn. Thường thì ban đầu bán kính lân cận lớn và giảm dần xuống 1 trong suốt quá trình huấn luyện.

Tỷ lệ học $\alpha(t)$ là một hàm giảm dần theo thời gian. Hai mẫu dùng phổ biến là hàm tuyến tính và hàm nghịch đảo theo thời gian:

$$\alpha(t) = \frac{A}{t + B}$$

với A và B là các hằng số.



(a) Lân cận Bubble (b) Lân cận Gauxơ

Hình 14. Hai hàm lân cận cơ bản

Việc huấn luyện thường được tiến hành trong hai giai đoạn. Giai đoạn đầu, có liên quan đến việc sử dụng giá trị ban đầu α đủ lớn và các bán kính lân cận. Trong giai đoạn sau giá trị α và bán kính lân cận đủ nhỏ ngay từ khi bắt đầu. Thủ tục này phù hợp với việc điều chỉnh xấp xỉ ban đầu của SOM trong cùng một không gian giống như dữ liệu đầu vào và sau đó điều chỉnh tốt trên ma trận.

Có nhiều biến thể của SOM. Một chủ đề khác của SOM là dùng tỷ lệ học mạng nơron và các kích thước lân cận. Ngoài ra có thể sử dụng cấu trúc ma trận một cách

thích hợp hoặc ngay cả cấu trúc đang phát triển. Mục đích của các biến đổi này là thiết lập SOM theo hình trạng tốt hơn trong khuôn khổ của tập dữ liệu hoặc thực hiện kết quả lượng tử hoá (quantization) tốt hơn.

2.2.3 Sử dụng SOM trong khai phá dữ liệu

Thuật toán SOM với những ưu điểm của nó, đã trở thành công cụ có ích trong khai phá dữ liệu. Đó là, tạo ra hàm phân bố xác suất cho tập dữ liệu ban đầu, dễ giải thích và quan trọng nhất là trực quan hoá tốt [8,10,11]. Tuỳ theo vấn đề cần giải quyết, các chuyên gia khai phá dữ liệu có thể chọn các phương pháp khác nhau để phân tích dữ liệu đưa ra. Thế nhưng với phương pháp SOM có thể làm nhiều công việc cùng một lúc và cho kết quả tương đương với việc kết hợp nhiều phương pháp khác với nhau. Như đã trình bày, SOM rất hiệu quả trong việc phân cụm và rút gọn kích thước dữ liệu. Nếu tích hợp SOM với các phương pháp khác có thể sinh luật.

Trực quan hoá rất có ý nghĩa trong khai phá dữ liệu, là yếu tố quan trọng trong báo cáo kết quả hoặc "tạo" tri thức [10]. Các minh hoạ trực quan dùng để hiểu thấu đáo tập dữ liệu và tóm tắt cấu trúc tập dữ liệu. Có thể khẳng định điểm mạnh của SOM là phương pháp trực quan hoá . Các kỹ thuật trực quan hoá dùng SOM gồm:

- Trực quan hoá ma trận gồm trực quan hoá các thành phần (component planes) của vectơ và sự tương quan giữa chúng; trực quan hoá ma trận hợp nhất khoảng cách U (unified distance matrix – U Matrix) để biểu diễn cấu trúc cụm của dữ liệu; ánh xạ Sammon [11] thể hiện hình ảnh của ma trận trong không gian đầu vào; các biểu đồ dữ liệu và phương pháp chiếu tập dữ liệu cho mục đích trực quan.
- Trực quan hoá đối tượng thực chất là áp dụng SOM để chọn lọc đặc tính nổi trội của các thành phần dữ liệu, bằng cách đánh màu tự động cho mỗi đơn vị của ma trận hoặc ấn định màu bằng tay. Mỗi điểm của đối tượng được đánh dấu bằng màu phù hợp với màu BMU của điểm đó.

Độ đo ma trận (Map measures) là độ đo chất lượng của SOM thường được ước lượng dựa trên độ phân giải của nó và cách bảo toàn tốt hình thái của tập dữ liệu trên ma trận. Các độ đo chất lượng khác của ma trận có thể dựa vào sự phân cụm chính xác của ma trận đó, nhưng lại đòi hỏi các ví dụ đầu vào phải được gán nhãn.

Ngoài độ đo trên, chất lượng của SOM có liên quan đến kích thước thật của tập dữ liệu ban đầu. Nếu kích thước ma trận SOM lớn hơn kích thước dữ liệu đầu vào, thì ma trận không thể thể hiện theo phân bố của tập dữ liệu ban đầu. Như vậy sẽ mâu thuẫn với mục đích bảo toàn trạng thái và độ phân giải của ma trận. Một ma trận với độ phân giải không phù hợp có thể phá vỡ hình thái của nó.

Thường độ phân giải là một độ đo trung bình lỗi lượng tử trên toàn bộ tập dữ thử nghiệm:

$$\varepsilon_q = \frac{1}{N}\sum_{i=1}^{N}\|x_i - m_c\|$$

Phân cụm: các thuật toán phân cụm dữ liệu như là K thành phần chính hoặc ISODATA [9], thường tối thiểu khoảng cách trong cụm và cực đại khoảng cách giữa các cụm. Độ đo khoảng cách có thể căn cứ vào liên kết đơn hoặc liên kết đầy đủ. Liên kết đơn là độ đo khoảng cách từ một cụm X đến cụm Y nào đó bằng cách cực tiểu khoảng cách giữa thành phần các cụm q_X ($q_X \in X$) và q_Y ($q_Y \in X$), liên kết đầy đủ là độ đo khoảng cách bằng cách cực đại, thường được xác định như sau:

$$d_s(X,Y) = \min\{d(q_X,q_Y) \mid q_X \in X, q_Y \in Y\}$$

$$d_c(X,Y) = \max\{d(q_X,q_Y) \mid q_X \in X, q_Y \in Y\}$$

Hạn chế trong liên kết đơn đó là các cụm dễ trở thành chuỗi dài do đó không điển hình cho dữ liệu. Mặt khác, với liên kết đầy đủ đôi khi vượt quá giới hạn cho phép. Ý tưởng kết hợp giữa liên kết đơn và liên kết đầy đủ hoàn toàn có thể thực hiện được. Bằng cách gắn độ đo cho các điểm trong cụm với trọng số phù hợp. Như vậy,

độ đo vừa gắn được giá trị cho tất cả các điểm giống như khoảng cách vừa giữ được hình thái của cụm dữ liệu. Phương pháp SOM hoàn toàn có thể được dùng như một phép đo.

2.2.4 SOM với bài toán phân cụm

SOM là phương pháp phân cụm theo cách tiếp cận mạng nơron và thuật toán học ganh đua. Vectơ trọng số của ma trận SOM chính là trọng tâm cụm, việc phân cụm có thể cho kết quả tốt hơn bằng cách kết hợp các đơn vị trong ma trận để tạo thành các cụm lớn hơn. Một điểm thuận lợi của phương pháp này là vùng Voronoi của các đơn vị ma trận là lồi, bằng cách kết hợp của một số đơn vị trong ma trận với nhau tạo nên các cụm không lồi. Việc sử dụng các độ đo khoảng cách khác nhau và các chuẩn kết liên kết khác nhau có thể tạo thành các cụm lớn hơn.

Ma trận khoảng cách: chiến lược chung trong phân cụm các đơn vị của SOM là tìm ma trận khoảng cách giữa các vectơ tham chiếu và sử dụng giá trị lớn trong ma trận như là chỉ số của đường biên cụm [11]. Trong không gian ba chiều, các cụm sẽ được thể hiện như "các thung lũng". Vấn đề là làm sao để quyết định các đơn vị trong ma trận thuộc về một cụm nào đó cho trước.

Để giải quyết được vấn đề này, người ta thường sử dụng thuật toán tích tụ (agglomerative algorithm), gồm các bước:

1.Quy cho mỗi đơn vị trong ma trận một cụm riêng.

2.Tính toán khoảng cách giữa tất cả các cụm.

3.Ghép hai cụm gần nhất.

4. Nếu số cụm tồn tại bằng số cụm do người dùng định nghĩa trước thì dừng, nếu không lặp lại từ bước 2 .

SOM là thuật toán phân cụm vì mỗi đơn vị trong ma trận ngay từ đầu là một cụm con gồm các ví dụ trong tập Voronoi của nó. SOM có thể được hiểu như cụm mờ: mỗi ví dụ là bộ phận của mọi cụm với thành phần giá trị tỷ lệ với hàm lân cận tại điểm BMU của nó. Sự giải thích này có thể phù hợp nếu số lượng các ví dụ cho mỗi cụm ban đầu là nhỏ hoặc phương pháp mờ được dùng như một bước xử lý sau dựa vào kết quả đầu ra của SOM.

Mặc dù, không giống hầu hết các phương pháp lấy mẫu cơ bản, trạng thái tối ưu đối với SOM là bằng không, khi số các mẫu bằng số các cụm. Để thay đổi trạng thái tối ưu thì số các đơn vị trong SOM phải lớn hơn số các cụm đưa ra. Hàm lân cận thể hiện các đơn vị lân cận trong ma trận, vì vậy các đơn vị này phải có thuộc tính giống nhau hơn so với các đơn vị trong các cụm khác. Sự di chuyển từ một cụm này sang cụm khác trong ma trận diễn ra từ từ trên một số đơn vị trong ma trận. Điều này có nghĩa là nếu số cụm mong muốn là đủ nhỏ thì ma trận SOM cũng phải được phân cụm.

Dùng SOM như một bước trung gian để phân cụm, đó là cách tiếp cận gồm hai mức: đầu tiên phân cụm tập dữ liệu, và sau đó phân cụm SOM. Với mỗi vectơ dữ liệu của tập dữ liệu ban đầu thuộc cùng một cụm có mẫu gần nó nhất. Một ưu điểm của cách tiếp cận này là giảm thời gian tính toán, điều này dễ dàng phân biệt được với các thuật toán phân cụm khác mà điển hình là cây phân cấp thậm chí với một lượng nhỏ các ví dụ ban đầu cũng trở nên nặng nề. Chính vì vậy cách tiếp cận này là hoàn toàn phù hợp cho việc phân cụm một tập các mẫu hơn là làm trực tiếp trên tập dữ liệu.

Có thể sử dụng các phương pháp phân cụm bộ phận hay phân cụm theo phân cấp để phân cụm SOM. Các mẫu có thể được phân cụm trực tiếp hoặc phân cụm theo một số đặc tính xác định trước của SOM. Trong phân cụm bộ phận các đơn vị nội suy có thể bị bỏ qua khi phân tích [3]. Trong phân cụm tích tụ quan hệ lân cận SOM có thể được dùng để ràng buộc khả năng hợp nhất trong cấu trúc dạng cây dendrogram.

Nếu điều này được dùng kết hợp với các ràng buộc lân cận, các đơn vị nội suy để thể hiện đường biên trong ma trận mà vẫn tuân theo cấu trúc dendrogram.

Ngoài ra, có thể dùng trực tiếp ma trận khoảng cách làm cơ sở phân cụm. Vì ma trận khoảng cách cho biết khoảng cách trung bình của mỗi vectơ mẫu đến các lân cận của nó và dự đoán được phân bố xác suất cục bộ, việc tối thiểu cục bộ của ma trận được dùng làm trọng tâm hay điểm nhân của cụm. Sự phân chia có thể được thực hiện ngay sau đó bằng cách xác định đơn vị trong ma trận gần tâm nhất hoặc dùng cách loang theo tối thiểu cục bộ.

SOM cũng được áp dụng trong phân cụm tập dữ liệu không chuẩn hoá. Dùng quy tắc của học ganh đua [5], vectơ trọng số có thể điều chỉnh theo hàm phân bố xác suất của các vectơ đầu vào. Sự tương đồng giữa vectơ đầu vào x và vectơ trọng số w được tính toán bằng khoảng cách Ơclit. Trong suốt quá trình huấn luyện một vectơ trọng số w_j tuỳ ý được cập nhập tại thời điểm t là:

$$\Delta w_j(t) = \alpha(t) h_{cj}(t) \left[x(t) - w_j(t) \right]$$

Với $\alpha(t)$ là tỷ lệ học giảm dần trong quá trình huấn luyện, và $h_{ci}(t)$ là hàm lân cận giữa vectơ trọng số chiến thắng w_c, và vectơ trọng số w_j , $h_{ci}(t)$ cũng giảm dần trong quá trình huấn luyện. Mối quan hệ lân cận được xác định bằng cấu trúc hình học và mối quan hệ này cố định trong suốt quá trình học. Kết thúc quá trình học, điều chỉnh lại bán kính lân cận đủ nhỏ để cập nhập lại cho các vectơ trọng số chiến thắng w_c và các lân cận gần chúng nhất. Đối với cấu trúc một chiều nó có thể được biểu diễn bằng luật huấn luyện. Công thức trên là một sấp xỉ của hàm đơn điệu của phân bố xác suất trên các vectơ đầu vào. Trong cấu trúc hai chiều thì kết quả trả về là một sự tương quan giữa độ xấp xỉ và bình phương lỗi tối thiểu của vectơ lượng tử.

Trong trường hợp tồn tại vùng thoả mãn và tồn tại phân bố các tâm cụm, việc ước lượng quan hệ chiến thắng của các nơron là để mô phỏng trực quan các cụm. Hình 15 thể hiện năm cụm bằng cách mã hoá mức xám cho histogram chiến thắng. Dữ

liệu hỗn hợp Gauxơ được sinh ra bằng việc cố định năm tâm cụm và năm ma trận khác nhau. Kích thước của tập dữ liệu sinh ra và tập dữ liệu thực nghiệm là bằng nhau, và dự đoán tổng thể các ma trận được xấp xỉ bằng nhau. Các đơn vị được gán màu đen trong hình 15 là các nơron chết, các nơron này dễ dàng phân biệt các cụm với nhau.

Hình 15. Vectơ chiến thắng liên tục đối với SOM có 30x40 nơron cho dữ liệu hỗn hợp Gauxơ

Để bảo toàn hình thái lân cận trong ma trận, vectơ trọng số trong không gian đầu vào cũng được đặt gần nhau trong không gian đầu ra. Ánh xạ từ không gian đầu vào tới không gian đầu ra hầu như liên tục, nhưng ngược lại thì không đúng. Vì vậy, hai vectơ trọng số về mặt hình học là gần nhau nhưng không phải cùng thể hiện trên một cụm. Nếu khoảng cách của chúng là nhỏ, thì chúng có thể là một cụm, nếu ngược lại chúng xuất hiện ở các cụm khác nhau. Trực quan hoá khoảng cách lân cận giữa các vectơ trọng số được đưa ra trong ma trận hợp nhất khoảng cách.Với mọi vectơ trọng số w_{xy}, với x và y là các chỉ số hình thái, khoảng cách Ơclit dx và dy giữa hai lân cận và khoảng cách dxy tới lân cận tiếp theo được tính như sau:

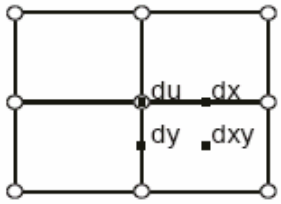
$$dx(x,y) = \left\| w_{x,y} - w_{x+1,y} \right\|$$

$$dy(x,y) = \left\| w_{x,y} - w_{x,y+1} \right\|$$

$$dxy(x,y) = \frac{1}{2}\left(\frac{\left\| w_{x,y} - w_{x+1,y+1} \right\|}{\sqrt{2}} + \frac{\left\| w_{x,y+1} - w_{x=1,y} \right\|}{\sqrt{2}} \right)$$

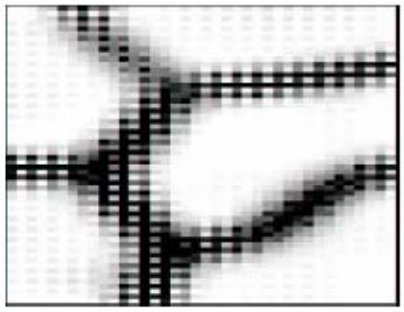
Khoảng cách *du* được tính bằng giá trị trung bình của tám khoảng cách biên xung quanh. Với bốn khoảng cách cho mỗi nơron *dx*, *dy*, *dxy* và *du*, như vậy dễ dàng xác định ma trận hợp nhất và ma trận này có kích thước là $(2n_x\text{-}1)(2n_y\text{-}1)$.

$$U = \begin{bmatrix} du(1,1) & dx(1,1) & du(2,1) & \dots & du(n_x,1) \\ dy(1,1) & dxy(1,1) & dy(2,1) & \dots & dy(n_x,1) \\ du(1,2) & dx(1,2) & du(2,2) & \dots & du(n_x,2) \\ dy(1,2) & dxy(1,2) & dy(2,2) & \dots & dy(n_x,2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ du(1,n_y) & dx(1,n_y) & du(2,n_y) & \dots & du(n_x,n_y) \end{bmatrix}$$



Hình 16. Định nghĩa một U-Matrix

Trong hình 17 các thành phần của U-matrix được mã hoá theo mức xám. Chỗ sáng là các giá trị thấp và chỗ tối cho giá trị cao. Như vậy, các cụm trên ma trận là các vùng có khoảng cách nhỏ giữa các trọng số và giữa các cụm với nhau lại có khoảng cách lớn.



Hình 17. U-Matrix của SOM trong hình 15

2.2.5 Các phương pháp phân cụm khác

a. Cây phân cấp [9]

Mục đích là kết nối liên tiếp các đối tượng với nhau vào trong các cụm lớn, dùng một số độ đo như khoảng cách hay thuộc tính giống nhau. Xét một biểu đồ cây có thứ tự và nằm ngang, bắt đầu từ đối tượng bên trái của biểu đồ, tưởng tượng rằng

trong mỗi bước chúng ta "nới lỏng" dần các tiêu chuẩn. Hay diễn đạt bằng cách khác là giảm dần ngưỡng khi đưa ra quyết định có hai hay nhiều đối tượng là các thành phần của cùng một nhóm.

Bằng cách này chúng ta có thể kết nối ngày càng nhiều các đối tượng lại với nhau và một tập hợp ngày càng lớn các cụm khác nhau. Cuối cùng, tất cả các đối tượng được nối lại với nhau. Trong các biểu đồ, trục hoành xác định khoảng cách liên kết. Vì vậy mỗi nút trên đồ thị chúng có thể thể hiện khoảng cách tiêu chuẩn mà các thành phần tương ứng được liên kết với nhau trong một cụm đơn. Khi cấu trúc dữ liệu rỗng các thành phần của trong các cụm của đối tượng mà giống nhau thì cấu trúc sẽ được thể hiện trong cây phân cấp như các nhánh riêng biệt

b. K thành phần chính (Hartigan, 1975) [9]

Đây là phương pháp phân cụm rất khó, giả sử rằng luôn có các giả thuyết liên quan đến một số nhóm trong các ví dụ. Điều mong muốn là có thể sắp xếp một cách chính xác các cụm rời rạc nhau. Các nghiên cứu cho thấy rằng chỉ có thể thực hiện được bởi thuật toán K thành phần chính. Tóm lại phương pháp K thành phần chính sẽ đưa ra chính xác *k* cụm tách biệt lớn nhất có thể.

Cho một cơ sở dữ liệu của *n* đối tượng và *k* là số các cụm cho trước, thuật toán tổ chức phân chia các đối tượng vào k phần ($k \leq n$). Các cụm được thiết lập theo một tiêu chuẩn phân chia khách quan, thường được gọi là hàm tương đồng (*similarity function*), dùng khoảng cách để xác định các đối tượng trong một cụm là "giống nhau" và "khác nhau" về tính chất dữ liệu.

Thuật toán K thành phần chính được thực hiện theo bốn bước sau:

- Xác định thành phần các đối tượng vào trong *k* tập con khác rỗng.
- Tính các điểm nhân của cụm trong các thành phần hiện tại.

- Chia đối tượng vào cụm khi đối tượng đó có khoảng cách gần điểm nhân nhất.
- Lặp lại bước 2, và dừng khi không còn sự phân chia mới.

Thuật toán:

Input: số các cụm *k* và một dữ liệu gồn *n* đối tượng.

Output: Một tập gồm *k* cụm và tối thiểu tiêu chuẩn bình phương lỗi.

Phương pháp:

(1) Chọn tuỳ ý *k* đối tượng và coi là các nhân cụm ban đầu;
(2) Lặp
(3) Xác định lại mỗi đối tượng vào cụm sao cho đối tượng đó là giống nhau nhất, dựa vào giá trị trung bình của các đối tượng trong cụm;
(4) Cập nhập lại các nhân cụm, bằng cách tính giá trị trung bình của các đối tượng cho mỗi cụm;
(5) Cho đến khi không còn thay đổi nào.

c. Cực đại kỳ vọng (Expectation Maximization)[9]

Đây là phương pháp gần giống như K thành phần chính, kỹ thuật này tìm cụm trong số các đối tượng quan sát hoặc các biến thể và ấn định các đối tượng đó vào các cụm. Một ví dụ ứng dụng nhiều nhất cho phân tích này là nghiên cứu thị trường để biết thái độ của người tiêu dùng có liên quan đến đối tượng nghiên cứu. Mục đích của nghiên cứu này là để tìm ra "các mảng thị trường". Trong khi thuật toán K thành phần chính đưa ra một số cố định *k* các cụm, thì cực đại kỳ vọng mở rộng cách tiếp cận này để phân cụm bằng hai cách sau:

- Thay thế việc xác định các trường hợp hoặc các quan sát đến các cụm để cực đại hoá sự khác nhau cho các biến thể tiếp theo, cực đại kỳ vọng tính

toán các khả năng của các thành phần trong cụm dựa trên phân bố xác suất. Mục tiêu của thuật toán phân cụm sau này là cực đại toàn bộ xác suất hoặc các khả năng có thể xảy ra của dữ liệu, cuối cùng mới đưa ra các cụm.

- Không giống như phân cụm K thành phần chính, thuật toán tính cực đại kỳ vọng có thể được áp dụng cho cả các biến thay đổi liên tục và các biến cố định (trong khi K thành phần chính có thể cũng được điều chỉnh để phù hợp với các biến cố định).

2.3 Một vài ứng dụng của SOM

Thuật toán SOM đã được sử dụng trong nhiều lĩnh vực khác nhau với trên 5000 ứng dụng [13], SOM đã khẳng định được các ưu điểm sau:

- SOM rất có hiệu quả trong quá trình phân tích đòi hỏi trí thông minh để đưa ra quyết định nhanh chóng trên thị trường. Nó giúp cho người phân tích hiểu vấn đề hơn trên một tập dữ liệu tương đối lớn.
- Có khả năng biểu diễn dữ liệu đa chiều dùng trong trình bày và làm báo cáo. Và đây cũng là một vấn đề chính đã được đề cập đến nhiều trong luận văn này.
- Xác định các cụm dữ liệu (ví dụ các nhóm khách hàng) giúp cho việc tối ưu phân bố nguồn lực (quảng cáo, tìm kiếm sản phẩm, ...).
- Có thể dùng để phát hiện sự gian lận trong thẻ tín dụng, và các lỗi dữ liệu.

Luận văn đề cập đến các vấn đề về tài chính và ngân hàng do đó chúng ta sẽ chưa đề cập đến các ứng dụng của SOM trong các ngành khác.Trong phần này giới thiệu hai ứng dụng của SOM trong lĩnh vực tài chính, đến chương sau sẽ trình bày các cách thức xây dựng một ứng dụng cụ thể của SOM trong phân cụm với một bài toán cụ thể trên dữ liệu của một Ngân hàng ở Việt Nam.

2.3.1 Lựa chọn quỹ đầu tư

Khi chọn lựa các quỹ cho mục đích đầu tư, nhà đầu tư thường phải xem xét đến nhiều chỉ tiêu: kết quả báo cáo tài chính trong những năm gần đây; các rủi ro; năng lực tài chính của quỹ; tỷ lệ doanh thu; chi phí; thời gian bổ nhiệm của người quản lý. Phần lớn trong thực tế các chương trình đã có thường làm việc trên hai hoặc ba chỉ tiêu; hay các chương trình có minh hoạ hình vẽ cũng bị giới hạn cách thể hiện trong không gian.

Với SOM nó có thể kết hợp tốt với bất kỳ kích thước nào của tập dữ liệu và đưa ra cách thể hiện thu gọn của dữ liệu trong ma trận hai chiều cùng với việc lấy ra các tính chất tuỳ ý hoặc đánh trọng số cho các cột, xây dựng chỉ số hợp nhất hoặc cho mục đích tổng thể.

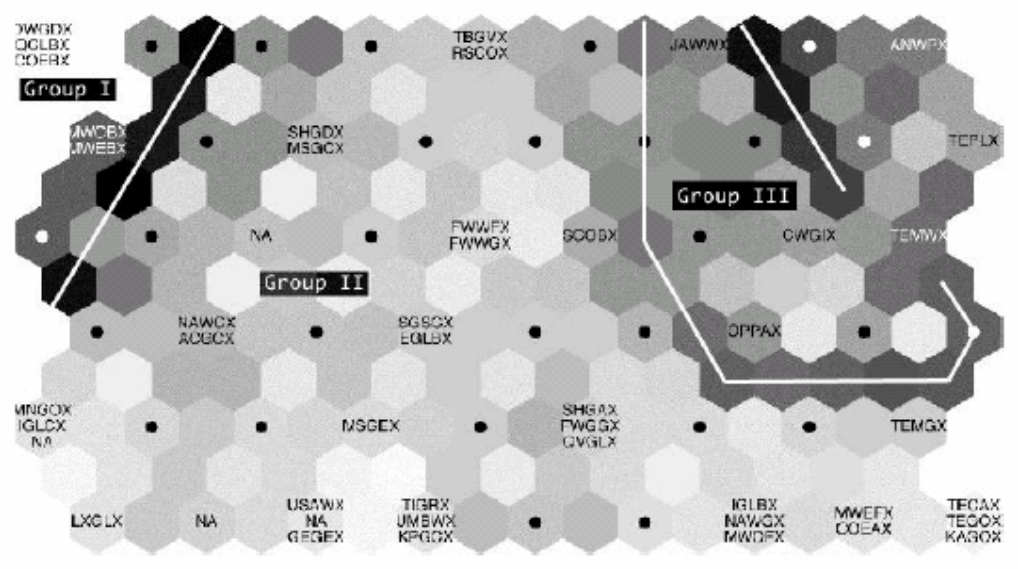
Để minh hoạ cho vấn đề này, chúng ta sử dụng cơ sở dữ liệu của Morningstart^{TM} [7] tìm kiếm và phân tích thông tin trong một tập hợp các quỹ. Trong ví dụ này chúng ta tập hợp các quỹ có đầu tư vào thị trường chứng khoán thế giới. Các tiêu chuẩn dùng để lựa chọn là (1) nhiệm kỳ quản lý của nhà lãnh đạo là >= 3 năm;(2) Số cổ đông >= B^+ (B là một số >=3); (3) Tỷ lệ Morningstart >=4; (4) Tỷ lệ chi phí >=1%. Dựa vào đây chúng ta đưa ra khoảng 50 quỹ có đầu tư chứng khoán. Căn cứ vào cơ sở dữ liệu của các quỹ chọn ra các biến chính. Tập dữ liệu đầu vào được chọn lọc sao cho giá trị của mỗi cột là bình đẳng.

Một ma trận gồm 50 quỹ được thể hiện trong hình vẽ. SOM thể hiện sự khác nhau giữa các quỹ với tỷ lệ 4 hoặc 5. SOM thu được dựa vào sự mô tả dữ liệu cho biết sự khác nhau giữa các quỹ được phân cụm theo cùng một tên loại. Thông tin tốt hơn, trong sự khác biệt chính các quỹ với nhau đã giúp cho việc lựa chọn tốt thành phần các danh mục vốn đầu tư có ảnh hưởng tốt hơn mong muốn của nhà đầu tư.

Tóm lại, từ 50 quỹ của thị trường chứng khoán thế giới, từ kết quả của SOM chúng ta có 3 nhóm chính. Từ kết quả này sẽ hỗ trợ cho việc ra quyết định nên chọn nhà quản lý nào

• Nhóm 1: là tập hợp các quỹ có người quản lý có ít hơn 3 năm nhiệm kỳ, danh mục vốn đầu tư của họ có doanh thu cao hơn và tỷ lệ phí tổn cũng cao.

• Nhóm 2: chiếm phần đông, gồm các quỹ có người quản lý có nhiều hơn số năm nhiệm kỳ, doanh thu của vốn đầu tư ít hơn và tỷ lệ phí tổn thấp hơn.



Hình 18. Mô phỏng SOM cho 50 quỹ có đầu tư chứng khoán

Nhóm	Số lg nhóm	Nhà quản lý	Giá trị tài sản	Tỷ lệ D/Thu	Front Load	Defer Load	Tỷ lệ phí tổn
1	5	2.8	658.2	80.8	0	4.6	2.3
2	36	3.3	272.4	70.7	2.2	0.1	1.7
3	6	7.2	6638.3	52.7	4.8	0	1

• Nhóm 3: là các nhóm có người quản lý có số năm nhiệm kỳ cao hơn (gấp 2 lần nhóm 1), doanh thu vốn đầu tư ít hơn nhóm 2 và tỷ lệ phí tổn cũng ít nhất

2.3.2 Đánh giá rủi ro tín dụng giữa các nước

Một ví dụ khác liên quan đến việc phân tích các cơ hội đầu tư cho thị trường mới phát triển. Trong ví dụ này tập trung vào các rủi ro liên quan trong đầu tư vào trong

các thị trường chứng khoán. SOM được dùng để phân tích các rủi ro và nhóm các nước có độ rủi ro gần giống nhau.

Việc phân tích dựa vào một bài báo của Greg Ip phát hành trong bài báo của Phố Wall (WSJ) năm 1997 [7]. Trong bài "mục đích đầu tư: trò chơi rủi ro" Greg Ip đã sắp xếp 52 quốc gia trên thế giới dựa vào hiệu quả kinh tế ; chính trị, kinh tế và rủi ro của thị trường; khả năng thanh toán của các thị trường chứng khoán; sự điều chỉnh và hiệu quả trên thị trường của các quốc gia. Các quốc gia được phân chia thành năm nhóm: (1) các nước giống Mỹ nhất; (2) các nước phát triển khác; (3) các thị trường mới và phát triển; (4) các thị trường mới hoàn toàn; (5) các thị trường ranh giới.

Trong đó US được coi là một điểm chuẩn để phân lớp các quốc gia; các quốc gia được chia thành năm nhóm; chỉ tiêu được dùng để phân chia không được cung cấp rõ ràng; các quốc gia thuộc nhóm số năm có rất nhiều dữ liệu bị thiếu.

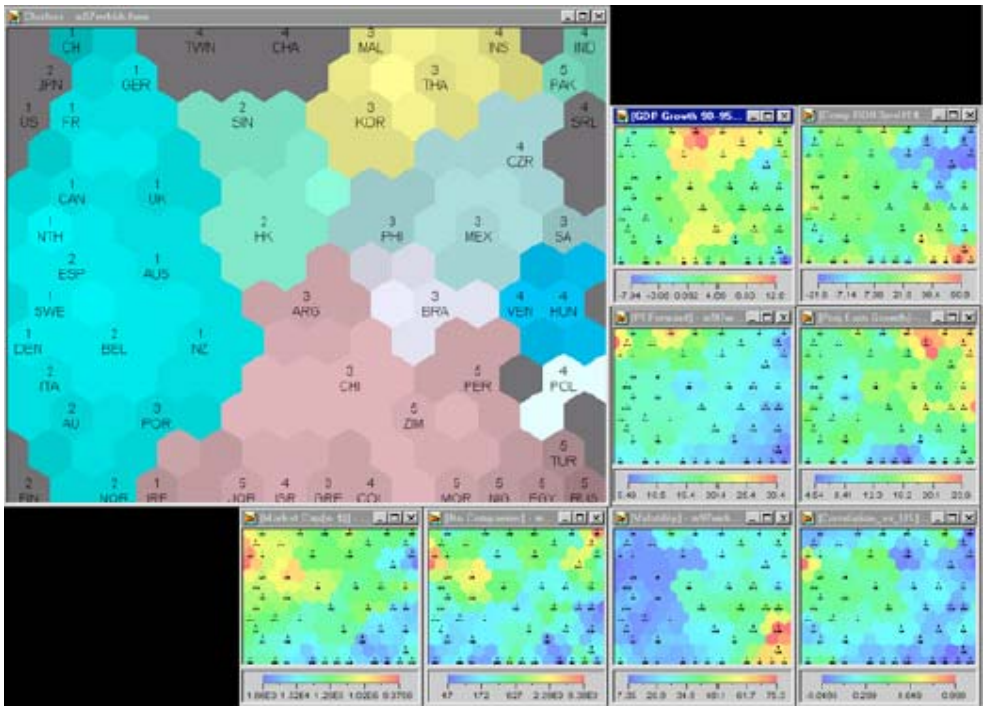
Cùng một dữ liệu và cùng một công việc phân tích trong một cách tương tác và cách trực quan để tạo ra một SOM chúng ta nhận được kết quả hoàn toàn khác.

Trong hình 20 các cửa sổ thành phần của tỷ lệ giá hiện tại và tỷ lệ giá forward, lợi tức, chiếm dụng vốn thị trường, số các công ty và tính chất không ổn định được thể hiện. Đối với mỗi cửa sổ thành phần, màu ở mỗi nút thể hiện khoảng giá trị của mỗi thành phần, giá trị thấp hơn được đánh màu xanh và giá trị cao được đánh màu đỏ; các giá trị giữa màu xanh sáng đến màu xanh lá cây, đến màu vàng, màu cam. So sánh các giá trị thành phần trong số các vùng có thể thể hiện được sự phụ thuộc không tuyến tính và vì vậy nhận diện trực quan ý nghĩa của các cụm.

Một ma trận với các rằng buộc cho biết số lượng các cụm như sau US, Ấn độ và Nhật ở các cụm khác nhau, US và Nhật có thị trường với ảnh hưởng lớn, Ấn độ có số lượng lớn các công ty liệt kê trong thanh toán hối phiếu; Thổ Nhĩ Kỳ và Phần Lan xác định một nhóm và các quốc gia còn lại chưa được phân hoá. Rõ ràng giả sử

giới hạn của SOM về năm cụm trong trường hợp này là không cung cấp được một lượng lớn các thông tin mới.

Hình 19. Mô phỏng các quốc gia có rủi ro tín dụng theo dữ liệu của WSJ



Hình 20. Mô phỏng các quốc gia có rủi ro tín dụng theo dữ liệu của WSJ

Nếu các ràng buộc giả sử được thay đổi chúng ta thu được các nhóm quốc gia hoàn toàn khác dựa trên các rủi ro của quốc gia:

Cụm1: Úc, New Zealand, Canada và phần lớn các nước Châu Âu.
Cụm2.: Phần lớn các nước Mỹ La tinh, và Đông Âu.
Cụm3: Mexico, Philippines, Bắc Phi và Cộng hoà Séc.
Cụm4: Nam Triều Tiên, Malaysia, Thái Lan và Indonesia.
Cụm5: Singapore và Hồng Kông.
Cụm6: Hungary và Venezuela.
Cụm7: Brazil.
Cụm8: Phần lan.
Cụm9: Ấn độ và Pakistan.

2.4 Kết luận chương 2

Phương pháp học mạng nơron không giám sát đặc trưng là thuật toán học ganh đua là phương pháp thích hợp trong khai phá dữ liệu. Trong chương này tập trung chính vào thuật toán SOM với bài toán phân cụm. SOM là một kỹ thuật mạng nơron truyền thẳng sử dụng thuật toán học không giám sát (học ganh đua) và qua một quá trình "tự tổ chức", sắp xếp đầu ra cho trong thể hiện hình học của dữ liệu đầu vào. Sử dụng SOM trong khai phá dữ liệu như một bước trung gian để giải quyết bài toán phân cụm dữ liệu. Mà trước tiên là dùng SOM để phân cụm tập dữ liệu đầu vào, sau đó SOM thu được lại được phân cụm bằng phương pháp phân cụm theo phân cấp hoặc phân cụm bộ phận.

So sánh SOM với một số phương pháp phân cụm đã được dùng như phân cụm theo cây phân cấp, K thành phần chính, cực đại kỳ vọng, ... thấy rằng phương pháp SOM có nhiều ưu điểm như:

- Dữ liệu đầu vào có thế lớn, không hạn chế kích thước của dữ liệu.

- Mô phỏng trực quan dữ liệu chính xác từ đó hiểu được cấu trúc của dữ liệu.
- Tiết kiệm được thời gian vì khi làm việc trên các mẫu thì nhanh hơn so với dữ liệu trực tiếp.

Trong chương này cũng đã đề cập đến hai ứng dụng điển hình của SOM trong khai phá dữ liệu tài chính là bài toán lựa chọn quỹ cho đầu tư tài chính và đánh giá rủi ro tín dụng của các quốc gia trên thị trường chứng khoán thế giới.

CHƯƠNG 3. ỨNG DỤNG MÔ HÌNH SOM TRONG BÀI TOÁN NGÂN HÀNG

3.1 Phát biểu bài toán

Có rất nhiều phương pháp cho việc khám phá tri thức và khai phá dữ liệu trong tài chính và kinh tế có sử dụng mạng nơron không giám sát. Đặc biệt, phương pháp sử dụng SOM có thể trực quan hoá tốt hơn đối với dữ liệu có kích thước lớn; tạo ra biểu diễn các mối quan hệ phức tạp; cải thiện cụm và rút gọn dữ liệu; tạo điều kiện thuận lợi cho việc khám phá tri thức qua việc xác định các cấu trúc và mẫu mới trong dữ liệu. Nhiều ứng dụng của SOM đã được sử dụng làm công nghệ và các lĩnh vực khoa học khác. Các ứng dụng của SOM trong tài chính, kinh tế và thị trường hầu hết còn mới lạ. Trong phần ứng dụng của SOM ở trên đã nêu ra hai ví dụ điển hình trong tài chính đã được áp dụng để phân lựa chọn các quỹ đầu tư cho các dự án và đánh giá rủi ro tín dụng của các nước trong lĩnh vực chứng khoán.

Căn cứ vào quy trình thực tế của phòng tín dụng tại các Ngân hàng, tôi muốn đề xuất áp dụng SOM trong việc đánh giá khách hàng là các doanh nghiệp có nhu cầu vay vốn. Bằng các thể hiện trực quan của SOM điều này có thể giúp cho cán bộ tín dụng cũng như ban lãnh đạo có những quyết định khi duyệt đơn vay của khách hàng.

Quy trình cho vay tín dụng tại Ngân hàng được thực hiện theo các bước sau:

- Khách hàng đến vay vốn tại ngân hàng phải cung cấp đầy đủ thông tin gồm: chỉ tiêu thanh khoản (khả năng thanh toán), chỉ tiêu hoạt động (vòng quay hàng tồn kho, kỳ thu tiền bình quân, doanh thu trên tổng tài sản), chỉ tiêu cân đòn nợ (nợ phải trả), chỉ tiêu thu nhập, kinh nghiệm trong ngành của ban giám đốc,...Ngoài ra, khách hàng sẽ phải trình bày phương án kinh doanh (sử dụng tiền vay) và trên cơ sở đó cán bộ tín dụng dự kiến tỷ lệ khả thi của phương án.

- Cán bộ tín dụng nhập dữ liệu vào chương trình quản lý khách hàng của ngân hàng và thực hiện phân loại khách hàng.
- Nhân viên tín dụng thay mặt khách hàng bảo vệ kế hoạch vay vốn của khách hàng trước hội đồng tín dụng.
- Các thành viên hội đồng tín dụng có/không chấp thuận cho khách hàng vay vốn căn cứ vào hồ sơ của khách hàng đã được nhập vào chương trình quản lý khách hàng.

Áp dụng SOM trong việc phân tích thông tin khách hàng vay vốn trên một khối lượng lớn các dữ liệu về khách hàng có nhu cầu (có thể chưa phải là khách hàng chính thức hoặc cũng có thể là khách hàng đã từng vay vốn) vừa có ích cho nhận định của nhân viên tín dụng làm cơ sở để bảo vệ khách hàng trước hội đồng tín dụng. Ngoài ra, nó còn trợ giúp cho các thành viên trong hội đồng đưa ra quyết định đồng ý hay không đồng ý cho khách hàng vay vốn.

Sử dụng công cụ SOM ToolBox để thể hiện trực quan các thông tin khách hàng. Dữ liệu được lấy từ chương trình quản lý chung của Ngân hàng.

3.2 Giới thiệu công cụ SOM Toolbox

Công cụ SOM Toolbox, một sản phẩm của nhóm SOM Toolbox thuộc trường Đại học Kỹ thuật Helsinki, là một thư viện gồm các hàm viết bằng Matlab. Đây là một bộ công cụ dễ sử dụng để xây dựng SOM cho các mục đích nghiên cứu. Đặc biệt, trong lĩnh vực khai phá dữ liệu, các nhà nghiên cứu đã coi đây là một công cụ có đặc thù riêng, và chính vì vậy SOM Toolbox định hướng trực tiếp đến các hàm trực quan.

Công cụ có thể được dùng để xử lý dữ liệu, khởi tạo và huấn luyện SOM trên một loạt các loại trạng thái hình học, SOM thể hiện trực quan bằng nhiều cách khác nhau, và phân tích các thuộc tính của SOM và dữ liệu ban đầu, ví dụ như là đặc tính của SOM, các cụm trên ma trận và sự liên quan giữa các thuộc tính. Trong khai phá

dữ liệu, công cụ Toolbox và SOM nói chung là một cặp phù hợp nhất cho việc hiểu dữ liệu một cách tổng quát, mặc dù nó cũng có thể được dùng cho xây dựng hình mẫu.

3.3 Cấu trúc chương trình

Công cụ SOM Toolbox gồm các hàm được viết bằng Matlab. Sử dụng công cụ này đề xây dựng hệ thống phân tích thông tin khách hàng, theo các bước sau:

- Đọc dữ liệu;
- Xây dựng cấu trúc dữ liệu;
- Xử lý dữ liệu trước khi đưa vào huấn luyện;
- Khởi tạo mẫu và huấn luyện theo thuật toán SOM;
- Mô phỏng kết quả;
- Phân tích kết quả.

3.3.1 Xây dựng tập dữ liệu

Đầu tiên, dữ liệu phải được đưa vào trong Matlab. Dữ liệu được lấy từ chương trình quản lý của Ngân hàng lưu trong hệ quản trị cơ sở dữ liệu SQL. Dữ liệu chính là một bảng thông tin khách hàng đã được lọc, chỉ lấy các thuộc tính được xét như sau:

Bảng 1: Thông tin khách hàng (dữ liệu của 30 khách hàng)

Khả năng thanh toán	**Vòng quay hàng tồn kho**	**Kỳ thu tiền bình quân**	**Doanh thu trên tổng tài sản**	**Nợ phải trả/tổng tài sản**	**Thu nhập trước thuế/doanh thu**	**Kinh nghiệm của ban GĐ**	**Tỷ lệ khả thi của PA kinh doanh**
4.86	10	40	0.46	19.3	21.3	2.2	0.56
2.9	7	32	4	25	8	10.5	0.85
2.3	6.5	37	3.5	35	7.5	5.8	0.78
1.7	6	43	3	45	7	2.7	0.35

1.4	5.5	50	2.5	55	6.5	1.8	0.33
1.3	5.2	60	1.5	61	4.8	0.9	0.60
1.17	7	30.7	4.03	85.3	2.3	5	0.75
1.14	4.21	28.9	2.1	61	2.8	11	0.81
4.7	18	11	4	19	8.3	12	0.55
1.25	1.83	43	0.76	72	0.03	6	0.44
1.7	6	43	3	45	7	2.8	0.38
2.3	6.5	37	3.5	35	7.5	6	0.45
2.9	7	32	4	25	8	13	0.97
1.4	5.5	50	2.5	55	6.5	1	0.23
1	4	60	2	65	5	1	0.13
0	2.3	58	1.02	0	15	2.6	0.8
6.4	0	102	0.75	15	22	2.5	0.72
0.85	3	60	1.2	75	3.5	0.5	0.25
2.5	4.3	30	4.2	40	6.5	11	0.98
1	3.4	55	1.5	55	4	0.9	0.36
1.8	4	40	3.5	45	6	8	0.57
1.3	3.7	50	2.5	50	5	4.5	0.64
1	2	55	3.5	55	8	3.6	0.46
1.2	3	50	4.2	50	9	7.5	0.67
2.3	3.5	40	5	45	10	10.2	0.69
0.9	1	60	2.5	60	7	1	0.31
0.75	0.8	71	2.3	66	6.8	0.9	0.3
1.4	14.2	8	1.8	28	0.7	12	0.8
2.5	4.2	10	1.5	40	1.1	10.5	0.74
7.6	1.3	61	0.2	7	17	1.5	0.65

Mỗi dòng dữ liệu là một ví dụ hay một vectơ, các giá trị trong dòng đó là các thành phần của vectơ hay các biến thể của tập dữ liệu. Các biến thể có thể là các thuộc

tính của dữ liệu hoặc là một tập hợp các giá trị tại cùng một thời điểm phân tích. Một vài giá trị có thể bị thiếu.

Công cụ SOM Toolbox gồm các cấu trúc dữ liệu sau:

- Data struct: gồm tất cả các thông tin liên quan đến tập dữ liệu.

Tên trường	Kiểu	Kích thước	Ý nghĩa
.type	string		Định danh kiểu cấu trúc (som_data)
.name	string		Định danh tập dữ liệu
.data	matrix	[m x n]	Bảng dữ liệu ban đầu
.lables	matrix	[m x k] (k<n)	Các lable
.comp_name	matrix	[n x 1]	Tên thuộc tính/thành phần
.comp_norm	matrix	[n x1]	Cấu trúc chuẩn hoá cho mỗi thuộc tính
.lable_name	matrix	[k x 1] (k<n)	Tên các lable

- Map struct: gồm các thông tin đầy đủ về SOM.

Tên trường	Kiểu	Kích thước	Ý nghĩa
.type	string		Định danh kiểu cấu trúc (som_map)
.name	string		Định danh của ma trận
.codebook	matrix	[munits x n]	Ma trận tín hiệu. Mỗi dòng tương ứng với vectơ trọng số của một map.
.topol	topology struct		Cấu trúc hình học của map: kích thước, kiểu lưới và hình dạng.
.lables	matrix	[munits x *]	Các nhãn trên ma trận
.neigh	string		Tên hàm lân cận (gaussian, cutgaussian, bubble, ep)
.mask	vector	[n x 1]	Mặt nạ tìm kiếm BMU
.trainhist	struct	[* x 1]	Cấu trúc mảng của các cấu trúc huấn

			luyện
.comp_name	matrix	[n x 1]	Tên thuộc tính/thành phần
.comp_norm	matrix	[n x 1]	Cấu trúc chuẩn hoá cho mỗi thuộc tính

- Topology struct: gồm các thông tin về cấu trúc hình học của ma trận.

Tên trường	Kiểu	Kích thước	Ý nghĩa
.type	string		Định danh kiểu cấu trúc (som_topol)
.msize	vector	[* x 1] (*>2)	Kích thước của ma trận
.lattice	string		Kiểu lưới, mặc định là lục giác
.shape	string		Hình dạng tổng quát của ma trận

- Normalization struct: thông tin chuẩn hoá.

Tên trường	Kiểu	Kích thước	Ý nghĩa
.type	string		Định danh kiểu cấu trúc (som_norm)
.method	string		Phương pháp chuẩn hoá (var, range, log, logistic, histD, histC)
.params	varies		Tuỳ theo phương pháp khác nhau có tham số khác nhau
.status	string		Trạng thái chuẩn hoá

- Traing struct: gồm các thông tin khi khởi tạo và huấn luyện.

Tên trường	Kiểu	Kích thước	Ý nghĩa
.type	string		Định danh kiểu cấu trúc (som_train)
.algorithm	string		Thuật toán huấn luyện/khởi tạo

.data_name	string		Tên dữ liệu huấn luyện
.mask	vector	[n x 1]	Mặt nạ tìm kiếm BMU
.neigh	string		Tên hàm lân cận (gaussian, cutgaussian, bubble, ep)
.radius_ini	scalar		Bán kính lân cận ban đầu
.radius_fin	scalar		Bán kính lân cận cuối cùng
.alpha_ini	scalar		Tỷ lệ học ban đầu tại thời điểm bắt đầu huấn luyện
.alpha_type	string		Kiểu hàm xác định tỷ lệ học
.trainlen	scalar		Độ dài huấn luyện
.time	string		Ngày và giờ thực hiện huấn luyện

- Grid struct: gồm các thông tin để trực quan hoá SOM.

Tên trường	Kiểu	Kích thước	Ý nghĩa
.type	string		Định danh kiểu cấu trúc (som_grid)
.lattice	string		Kiểu lưới
.shape	string		Hình dạng tổng quát của ma trận
.msize	vector	[1 x 2]	Kích thước của ma trận
.coord	matrix	[munits x 2] hoặc [munits x 3]	Toạ độ các đơn vị trong ma trận. Có kích thước là 2 hoặc 3
.line	string		Kiểu đường thẳng dùng cho các đường liên kết
.linecolor	string		Màu đường thẳng
.linewidth	scalar		Độ nét của đường thẳng
.marker	string		Kiểu dấu cho các đơn vị trong ma trận
.markersize	scalar		Kích thước kiểu dẫu
.markercolor	string		Màu kiểu dấu

.surf	empty vector RGB	Mặc định là rỗng. Nếu có giá trị thì là nét vẽ thêm vào, nếu là RGB thì nó là chỉ số mầu để trang trí
.label	string	Nhãn cho mỗi ô
.labelcolor	string	Màu cho nhãn
.labelsize	scalar	Kích thước chữ trên nhãn

3.3.2 Xử lý dữ liệu trước huấn luyện

Một cách tổng quát khi tiền xử lý dữ liệu có thể chỉ là sự chuyển đổi đơn giản hoặc thực hiện chuẩn hoá trên số liệu, sàng lọc để loại bỏ các giá trị vô lý, tính toán các giá trị mới thay thế chúng. Cân bằng các giá trị trong bộ công cụ này là đặc biệt quan trọng, vì thuật toán SOM dùng độ đo Ơclit để tính toán khoảng cách giữa các vectơ. Nếu chỉ có một giá trị nằm trong khoảng [0,...,1000] và các giá trị khác nằm trong khoảng [0,...,1] thì sẽ ảnh hưởng đến tổ chức của ma trận vì tác động của nó đến độ đo khoảng cách. Nói chung, chuẩn hoá dữ liệu mục đích là làm cho các giá trị là ngang bằng nhau. Cách thức mặc định để thực hiện vấn đề này đó là cân bằng tuyến tính tất cả các giá trị sao cho mỗi độ chênh lệch khác biệt bằng một. Điều này có thể thực hiện đơn giản bằng hàm sD = som_normalize(sD,'var') hoặc D = som_normalize(D,'var').

Một điều thuận lợi cho việc dùng các cấu trúc dữ liệu để thay thế cho các ma trận dữ liệu đó là cấu trúc dữ liệu thể hiện được thông tin chuẩn hoá trong trường *.com_norm*. Dùng hàm som_denormalize(sD) có thể khôi phục lại giá trị ban đầu.

3.3.3 Khởi tạo SOM và huấn luyện

Có hai cách để khởi tạo SOM đó là khởi tạo một cách ngẫu nhiên và khởi tạo tuyến tính, sử dụng hai thuật toán huấn luyện là thuật toán huấn luyện tuần tự và huấn luyện theo khối.

a. Thuật toán huấn luyện tuần tự

SOM được huấn luyện lặp đi lặp lại. Trong mỗi bước huấn luyện, chọn ngẫu nhiên một vector ví dụ x lấy từ tập dữ liệu đầu vào và tính khoảng cách giữa x với tất cả các vectơ trọng số của SOM theo một vài độ đo. Nơron có vector trọng số gần với vector đầu vào x nhất được gọi là BMU, xác định bởi c:

$$\|x - m_c\| = \min_i \{\|x - m_i\|\}$$

||.|| là độ đo khoảng cách Ơclit. Ở đây việc tính toán khoảng cách là đơn giản hơn là vì

- Các giá trị thiếu: các giá trị này được thay thế bằng giá trị NaN trong vectơ hoặc ma trận dữ liệu. Các thành phần thiếu được xử lý một cách đơn giản bằng cách loại trừ (ví dụ, giả sử rằng khoảng cách $||x\text{-}m_i||$ là bằng 0). Vì các giá trị giống nhau bị bỏ qua trong mỗi lần tính khoảng cách, điều này hoàn toàn là hợp lý.
- Mặt nạ (mask): Mỗi biến có một phần kết hợp phụ, được định nghiã trong trường *.mask* của ma trận và cấu trúc huấn luyện. Trường này được dùng chủ yếu trong mẫu nhị phân để loại trừ các giá trị nào đó từ tiến trình tìm BMU (1 giữ lại, 0 loại bỏ). Tuy nhiên, mặt nạ có thể lấy bất kỳ giá trị nào, nên nó có thể được dùng cho các giá trị đi kèm theo mức độ quan trọng của chúng.

Với mỗi lần thay đổi, độ đo khoảng cách là:

$$\|x - m\|^2 = \sum_{k \in K} w_k (x_k - m_k)^2$$

Với k là tập các giá trị (không có giá trị thiếu) của vector ví dụ x, x_k và m_k là các thành phần thứ k của ví dụ và vectơ trọng số và w_k là giá trị mặt nạ thứ k.

Sau khi tìm BMU, các vectơ trọng số của SOM được cập nhập sao cho BMU được di chuyển gần đến vector đầu vào hơn trong không gian đầu vào. Các lân cận của BMU được xem là như nhau. Sự mô phỏng này thể hiện độ loang của BMU và hình thái các lân cận của chúng về phía vectơ ví dụ. SOM cập nhập quy tắc cho vectơ trọng số của đơn vị i là:

$$m_i(t+1) = m_i(t) + \alpha(t)h_{ci}(t)\left[x(t) - m_i(t)\right]$$

t: thời điểm,

$x(t)$: một vectơ đầu vào lấy ngẫu nhiên từ tập dữ liệu đầu vào tại thời điểm t,

$h_{ci}(t)$: lân cận kernel quanh đơn vị chiến thắng c,

$\alpha(t)$: tỷ lệ học tại thời điểm t.

Lân cận Kernel là một hàm không tăng của thời gian và khoảng cách của đơn vị i từ đơn vị chiến thắng c. Nó xác định vùng ảnh hưởng của ví dụ đầu vào có trong SOM.

Việc huấn luyện thường diễn ra hai giai đoạn. Giai đoạn đầu, có liên quan đến tỷ lệ học ban đầu α_0 và bán kính ban đầu σ_0. Giai đoạn sau, giảm tỷ lệ học và bán kính vừa đủ nhỏ so với ban đầu. Đây là thủ tục phù hợp để điều chỉnh xấp xỉ SOM tới không gian tương đồng với dữ liệu đầu vào và sau đó điều chỉnh ma trận cho đúng.

b. Thuật toán huấn luyện khối

Thuật toán huấn luyện khối cũng là thuật toán lặp, nhưng thay vì chỉ dùng một vectơ tại một thời điểm, mà toàn bộ tập dữ liệu được thể hiện trên ma trận trước khi có bất kỳ điều chỉnh nào. Trong mỗi bước huấn luyện, tập dữ liệu được phân chia theo vùng Voronoi của các vectơ trọng số. Sau đó, các vectơ trọng số được tính toán như sau:

$$m_i(t+1) = \frac{\sum_{j=1}^{n} h_{ic}(t) x_j}{\sum_{j=1}^{n} h_{ic}(t)}$$

với $c = argmin_k\{||x_j - m_k||\}$ là chỉ số BMU của dữ liệu ví dụ x_j. Vectơ trọng số là một giá trị trọng số trung bình của các ví dụ, với trọng số của mỗi ví dụ là giá trị hàm lân cận $h_{ic}(t)$ tại BMU của nó. Giống như thuật toán huấn luyện tuần tự, các giá trị thiếu được bỏ qua trong khi tính toán giá trị trọng số trung bình.

Chú ý rằng trong thuật toán xử lý khối của K thành phần chính, các vectơ trọng số đơn giản chỉ là giá trị trung bình của tập dữ liệu Voronoi.

Có khả năng, có thể tính toán trước tổng các vectơ trọng số trong mỗi Voronoi:

$$s_i(t) = \sum_{j=1}^{nv_i} x_j$$

với nv_i là số các ví dụ trong tập Voronoi của đơn vị i. Sau đó, các giá trị mới của vectơ trọng số có thể được tính toán như sau:

$$m_i(t+1) = \frac{\sum_{j=1}^{m} h_{ij}(t) s_j(t)}{\sum_{j=1}^{m} nv_j h_{ij}(t)}$$

với m là số các đơn vị trong ma trận.

Hàm som_make lựa chọn kích thước ma trận và các tham số tự động, mặc dù nó có một số các tham biến. Nếu muốn giám sát chặt chẽ bằng các tham số huấn luyện, thì có thể sử dụng sự khởi tạo phù hợp và các hàm huấn luyện trực tiếp dùng các hàm som_lininit, som_randinint, som_seqtrain và som_batchtrain. Ngoài ra, các hàm

som_topol_struct, som_train_struct có hể được dùng để lấy các giá trị mặc định cho hình dạng ma trận, và các tham số huấn luyện tương ứng.

3.3.4 Mô phỏng (trực quan hoá)

SOM có thể được dùng như một nền tảng thích hợp cho việc thể hiện các đặc điểm khác nhau của SOM (hay của dữ liệu). Trong công cụ SOM Toolbox, có một số hàm mô phỏng SOM, được chia làm 3 loại theo trực quan ban đầu:

a. Mô phỏng ô (cell) dựa vào cách trình bầy ma trận lưới trong không gian đầu ra.

Mô phỏng ô thể hiện SOM trong không gian đầu ra: một lưới hình chữ nhật của các ô thuộc tính thể hiện các giá trị liên quan. Chú ý rằng, mô phỏng chỉ làm việc với các ma trận 1-2 chiều và các hình ‘cell’ và ‘toroid’ và mặc định là ‘sheet’.

Công cụ cơ bản là hàm som_show: som_show(sM); mặc định thể hiện ban đầu là ma trận hợp nhất khoảng cách được tính toán dựa trên tất cả các giá trị và sau đó thể hiện các mặt phẳng thành phần

- Ma trận hợp nhất khoảng cách mô phỏng khoảng cách giữa các đơn vị trong ma trận lân cận và hỗ trợ thể hiện cấu trúc cụm của ma trận: các giá trị lớn của ma trận hợp nhất khoảng cách cho biết ranh giới các cụm, các vùng giống nhau có giá trị thấp xác định cụm.
- Mỗi mặt phẳng thành phần thể hiện các giá trị của mỗi đơn vị trong ma trận.

Các giá trị thể hiện dùng chỉ số bảng màu. Với các màu khác nhau, SOM Toolbox sử dụng câu lệnh colormap, jet, hot, gray. Ngoài ra, các kiểu khác của mặt phẳng có thể là:

- Một lưới rỗng chỉ thể hiện một phần (edges) của các đơn vị. Điều này có thể được dùng như một cơ sở cho việc gắn nhãn hoặc các mô phỏng khác với màu nền có thể làm nhạt hơn.
- Trong plane màu của mỗi đơn vị đều là cố định màu. Điều này có thể được dùng để thể hiện cho ví dụ phân cụm hoặc thông tin nhận dạng khác cho việc liên kết các trực quan khác nhau. Có các công cụ đặc biệt như som_colorcode và som_clustercolor là các công cụ về màu sắc.

Trong hàm som_show có nhiều tham biến đầu vào mà có thể được dùng để điều khiển các loại plane để thể hiện và sắp xếp chúng. Các giá trị cân bằng có thể được chuẩn hoá lại thành dữ liệu ban đầu (nếu có thể) và có nhiều tham số thay đổi cách nhìn của sự mô phỏng nói chung, giống như sự định hướng của bảng màu.

Một hàm liên quan trong som_show_add thiết lập các thông tin thêm vào một con số được tạo ra bởi som_show như là: nhãn, biểu đồ (hit histogram), quỹ đạo (trajeactories).

- Gắn nhãn, được thực hiện bởi hàm som_autolabel, được dùng cho các loại đơn vị (hoặc một vài đơn vị), bằng cách ghi tên của chúng.
- Biểu đồ được đánh dấu thể hiện phân bố của các đơn vị phù hợp nhất cho một tập dữ liệu đưa ra. Nhiều biểu đồ có thể được vẽ và chúng được nhận dạng bởi các màu khác nhau và/hoặc các dấu khác nhau. Như vậy có thể so sánh các tập dữ liệu bằng phân bố 'hits' của chúng trên một ma trận. Các biểu đồ có thể được tính toán dùng hàm som_hits.
- Quỹ đạo thể hiện các đơn vị phù hợp nhất đối với một tập dữ liệu thể hiện là chuỗi thời gian (time series) (hoặc bất kỳ chuỗi được sắp). Nó có thể là một đường kết nối liên tục các đơn vị phù hợp nhất hoặc một "vệt" quỹ đạo giữa đơn vị phù hợp nhất hiện tại (dữ liệu ví dụ đầu tiên) có dấu lớn nhất và đơn vị phù hợp nhất cuối cùng (dữ liệu ví dụ cuối cùng) có dấu nhỏ nhất. Hàm som_trajectory được dùng để tác động quỹ đạo để phân tích và thậm

trí cho phần điều khiển ma trận và chuỗi thời gian trong suốt quá trình nghiên cứu quỹ đạo.

Som-show dùng thủ tục som_cplane làm cơ sở. Thủ tục này có thể được dùng để xây dựng tuỳ biến các kiểu mô phỏng ô. Các tham số tuỳ chọn gồm:

- Màu của các đơn vị,
- Kích thước cân bằng các đơn vị,
- Vị trí các đơn vị,
- Hình mẫu của đơn vị (đa giác tuỳ ý),
- Mẫu của các đơn vị (bằng cách cân bằng vị trí của các đỉnh).

b. Mô phỏng hình ảnh thể hiện một hình ảnh đơn giản trong mỗi đơn vị của ma trận.

Mô phỏng hình ảnh phần lớn là vẽ codebook của SOM, là một tập các hình ảnh thông thường. Ý tưởng là mỗi đơn vị của codebook được thể hiện bằng biểu đồ hình tròn, và các biểu đồ được bố trí cùng một cách như là các đơn vị trong các mô phỏng ô.

- Biểu đồ hình tròn (som_pieplane) là ý tưởng thể hiện các giá trị tỷ lệ. Màu sắc và kích thước các phần chia có thể được thay đổi bằng cách dùng các tham số khác nhau.
- Biểu đồ khối (som_barplane) phù hợp với việc thể hiện các giá trị các loại khác nhau. Màu sắc của mỗi khối và khoảng trống có thể được xác định trước.
- Hình dấu (som_plotplane) thể hiện các vectơ codebook như các hình học đơn giản. Màu sắc của nét vẽ có thể được xác định đối với mỗi đường riêng biệt.

c. Mô phỏng lưới thể hiện ma trận như một lưới hay đồ thị phân tán (scatter plot)

Hàm som_grid có thể được dùng để vẽ lại kiểu lưới. Hàm này xuất phát từ ý tưởng mô phỏng lại tập dữ liệu chỉ đơn giản gồm một tập các đối tượng, với mỗi một ví trí, màu sắc và hình ảnh. Hơn nữa, các liên kết giữa các đối tượng, ví dụ quan hệ lân cận, có thể được thể hiện dùng các đường thẳng. Với som_grid người sử dụng có thể ấn định tuỳ ý các giá trị cho mỗi thuộc tính của chúng. Ví dụ các toạ độ *x, y, z*, kích thước đối tượng và màu sắc có thể mỗi trạng thái cho một biến, vì thế có thể mô phỏng đồng thời năm biến. Các lựa chọn khác nhau là:

- Vị trí của đối tượng có thể có kích thước là 2-3.
- Màu sắc của các đối tượng có thể lựa chọn tuỳ ý từ vectơ RGB, sử dụng chỉ số màu đặc thù.
- Hình ảnh của đối tượng có thể là bất kỳ dấu của matlab (‘.’,’+’).
- Hơn nữa để các đối tượng kết hợp với các nhãn là có thể được thể hiện.
- Bề mặt giữa các đơn vị trong ma trận có thể được vẽ thêm vào lưới.

3.3.5 Phân tích kết quả

Để phân tích định lượng của SOM thì chỉ có ở một vài công cụ. Tuy nhiên, dùng các hàm như som_neighborhood, som_bmus, và som_unit_dists, thì cũng dễ dàng thực hiện một số phân tích. Nhiều nghiên cứu đang được thực hiện trong lĩnh vực này, và nhiều hàm mới cho việc phân tích sẽ được đưa thêm vào công cụ SOM Toolbox trong tương lai, ví dụ các công cụ phân cụm và phân tích các thuộc tính của cụm. Ngoài ra, sử dụng hàm som_quality(sMap,D) để xác định độ đo chất lượng của ma trận SOM trong dữ liệu ban đầu. Hàm trả về hai kết quả, một là khoảng cách trung bình của mỗi vectơ dữ liệu với BMU của chúng (lỗi lượng tử hoá), và hai là tỷ lệ của tất cả các vectơ dữ liệu đối với BMU thứ nhất và thứ hai không liền kề (lỗi hình thái).

3.4 Một số nhận xét

3.4.1 Độ phức tạp tính toán

Mỗi một giai đoạn của thuật toán huấn luyện tuần tự có thể được thực thi như sau:

```
for (j=0; j<n; i++) {
        bmu=-1; min=1000000;
        for (i=0; i<m; i++){
                dist=0;
                for (k=0; k<d; k++) { diff=X[j][k] –M[i][k]; dist+=diff*diff;}
                if (dist<min) {min=dist; bmu=i;}
        }
        for (i=0;i<m; i++) {
                h = alpha*exp(U(bmu,i)/r);
                for (k=0; k<d; k++) M[i][k]-=h*(M[i][k] – X[j][k]);
        }
}
```

Với X[j][k] là thành phần thứ k của ví dụ thứ j, M[i][k] là thành phần thứ k của đơn vị thứ i và U là một bảng khoảng cách ma trận lưới bình phương giữa các đơn vị trong ma trận được tính toán trước. Giả sử dùng hàm lân cận Gauxơ và bán kính r tương đương với $–2r(t)^2$. Do đó, mỗi giai đoạn cho thuật toán huấn luyện theo khối sẽ là:

```
for (i=0; i<m; i++){ vn[i] = 0; for (k=0; k<d; k++)  S[i][k] = 0;} /* khởi tạo */
for (j=0; j<n; i++) {
        bmu=-1; min=1000000;
        for (i=0; i<m; i++){
                dist=0;
                for (k=0; k<d; k++) { diff=X[j][k] –M[i][k]; dist+=diff*diff;}
                if (dist<min) {min=dist; bmu=i; vn[bmu]++;}
```

```
        }
        for (k=0; k<d; k++) S[bmu][k] += X[j][k];
}
for (i=0; i<m; i++) for (k=0; k<d; k++) M[i][k] = 0;
for (i1=0; i1<m; i1++) {
        htot = 0;
        for (i2=0; i2<m; i2++) {
                h = exp(U[i1][i2]/r);
                for (k=0; k<d; k++) m[I1][K] += H*S[i2][k];
                htot += h*vn[i2];
        }
        for (k=0; k<d; k++) M[i1][k] /=htot;
}
```

Có 6nmd + 2nm các toán tử (cộng, trừ, nhân, chia hoặc luỹ thừa) trong thuật toán huấn luyện tuần tự và 3nm + (2d +5)m^2 + (n+m)d các toán tử trong thuật toán huấn luyện khối. Vì vậy, độ phức tạp tính toán cho mỗi lần huấn luyện của thuật toán tuần tự là *O(nmd)* và nếu n>=m,độ phức tạp tính toán cho huấn luyện khối chỉ bằng một nửa của thuật toán tuần tự.

Nếu sử dụng các tham số mặc định đối với các hàm trong ToolBox thì cũng có thể tính toán được độ phức tạp trong toàn bộ quá trình huấn luyện. Số các đơn vị của m ma trận là tỷ lệ với căn bậc hai của n và số lượng các lần huấn luyện tỷ lệ với m/n. Vậy độ phức tạp tính toán cho toàn bộ quá trình tạo SOM là *O(nd)* nên có thể áp dụng cho các tập dữ liệu lớn, mặc dù kích thước các ma trận lớn đòi hỏi tốn nhiều thời gian hơn. Tất nhiên, trong một vài trường hợp số lượng các đơn vị trong ma trận cần được lựa chọn là khác nhau, ví dụ m=0.1n thì trong một vài trường hợp độ phức tạp lại là $O(n^2d)$.

Tuy nhiên, trên thực tế cũng có một vài sự khác biệt đáng kể của SOM. Về cơ bản có những nghiên cứu chỉ trong một số lượng nhỏ các đơn vị trong ma trận làm tăng tốc độ tìm kiếm phần tử chiến thắng nên độ phức tạp chỉ là *O(md)* đến *O(log(m)d)*.

Sau đây là một số kết quả so sánh giữa thuật toán huấn luyện tuần tự và thuật toán huấn luyện theo khối. Bảng 2 thể hiện các chỉ số ban đầu

Tham số	Giá trị
Kích thước dữ liệu	10,30,50,100
Độ dài dữ liệu	300,1000,3000,10000,30000
Số các đơn vị trong ma trận	30,100,300,1000
Hàm huấn luyện	som_batchtrain som_seqtrain
Hàm lân cận	‘gaussian'
Kỳ huấn luyện	10 kỳ

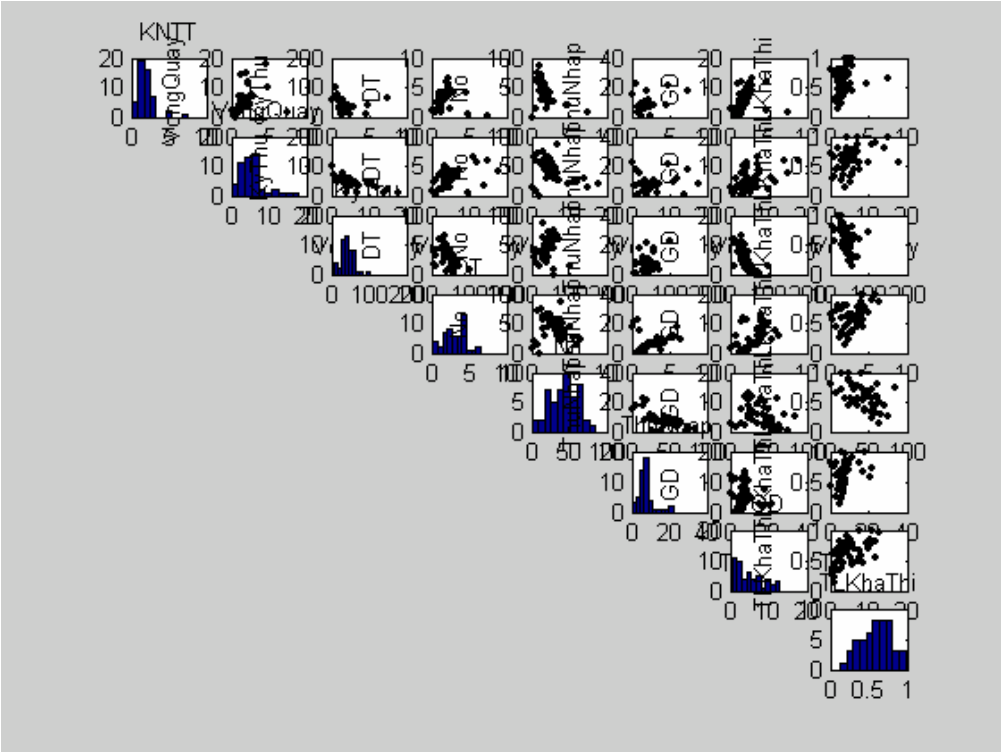
Bảng 3: Kết quả thời gian tính toán (10 kỳ huấn luyện)

Dữ liệu	Số đơn vị trong ma trận	Thuật toán som_batchtrain	Thuật toán som_seqtrain
[300 x 30]	100	0.4 s	4 s
[1000 x 30]	100	1.0 s	13 s
[3000 x 30]	100	2.6 s	40 s
[10000 x 30]	100	8.6 s	2.3 min
[3000 x 10]	300	5.4 s	43 s
[3000 x 30]	300	7.7 s	1.3 min
[3000 x 50]	300	9.8 s	1.8 min
[3000 x 100]	300	16 s	3.8 min
[3000 x 30]	30	14 s	4.4 min
[3000 x 30]	100	26 s	6.7 min
[3000 x 30]	300	1.1 min	13 min
[3000 x 30]	1000	4.5 min	34 min

3.4.2 Kết quả chạy chương trình

Một số kết quả chạy chương trình trên số liệu có cấu trúc như bảng 1. Bộ dữ liệu được lấy ngẫu nhiên từ chương trình quản lý thông tin khách hàng (gồm 150 khách hàng).

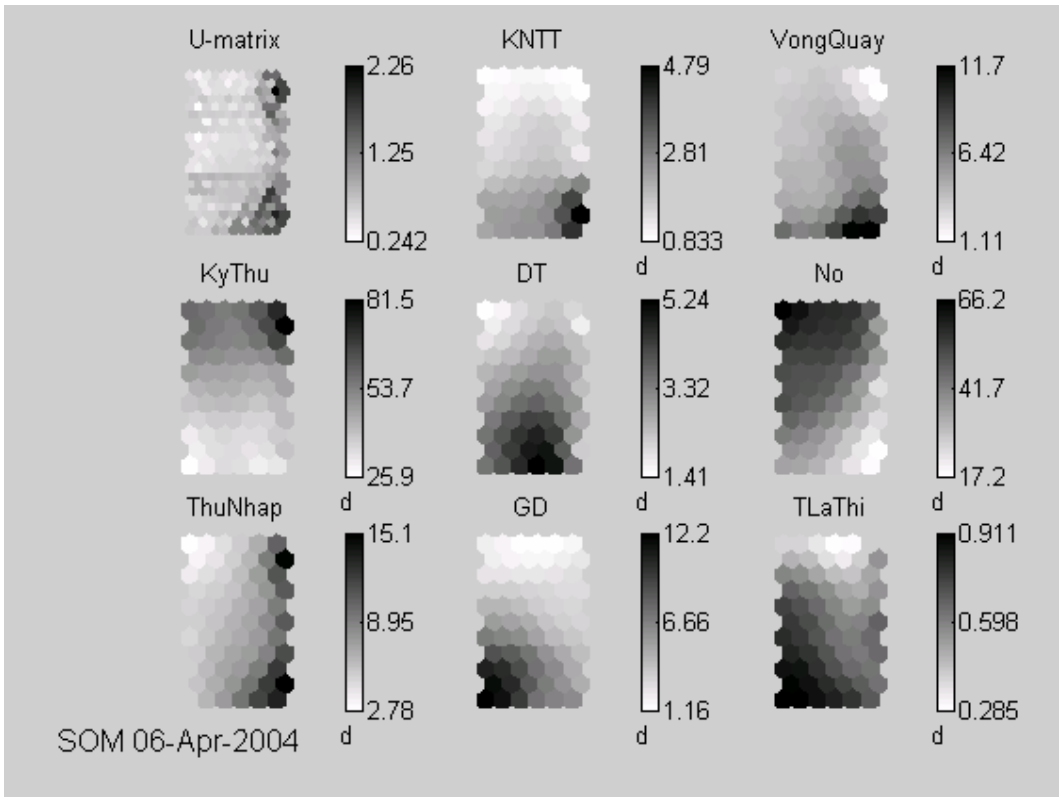
```
%    BUOC 1: DOC DU LIEU TU FILE
%    ======================
try,
  sD = som_read_data('custbank4.data');

data read ok
end
pause % An phim bat ky de tiep tuc...
```



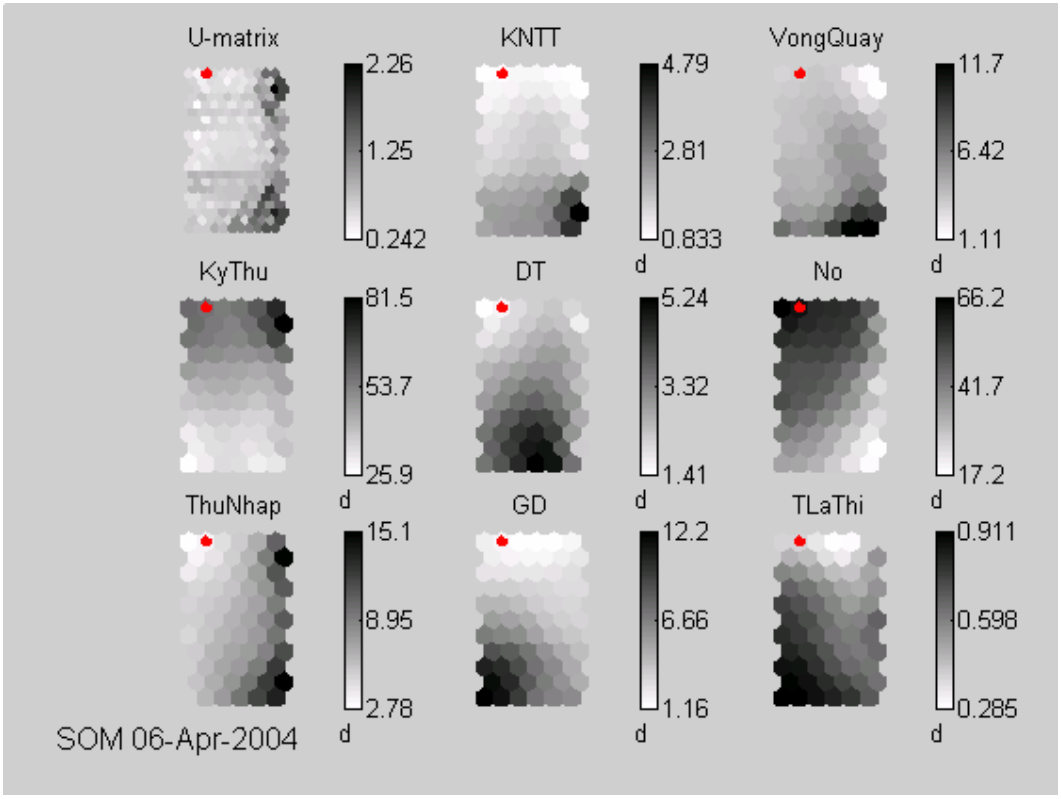
```
%    BUOC 2: XU LY DU LIEU
%    ========================
sD = som_normalize(sD,'var');

x = sD.data(1,:)

x =

    0.7042   -0.1638   -0.9779    0.8998   -0.3327   -0.0307    2.6831
1.6677


orig_x = som_denormalize(x,sD)

orig_x =

    2.5000    4.3000   30.0000    4.2000   40.0000    6.5000   15.0000
1.0000


pause % An phim bat ky de huan luyen...
```

```
%    BUOC 3: HUAN LUYEN DU LIEU
%    ====================
sM = som_make(sD);
Determining map size...
kich thuoc cua dlen: 150
 kich thuoc cua munits: 62
 kich thuoc cua munits: 62
 kich thuoc cua sTopol.msize: 8
 kich thuoc cua sTopol.msize: 8
  map size [11, 6]
Initialization...
kich thuoc cua munits: 100
 kich thuoc cua sTopol.msize: 10
 kich thuoc cua sTopol.msize: 10
 Training using batch algorithm...
Rough training phase...
kich thuoc cua munits: 66
 kich thuoc cua dlen: 150
 kich thuoc cua mpd: 4.400000e-001
 kich thuoc cua traninlen: 5

Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Finetuning phase...
kich thuoc cua munits: 66
 kich thuoc cua dlen: 150
 kich thuoc cua mpd: 4.400000e-001
 kich thuoc cua traninlen: 18

Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Training:   0/   0 s
Final quantization error: 1.071
Final topographic error:  0.033

pause % An phim bat ky de tiep tuc...
```

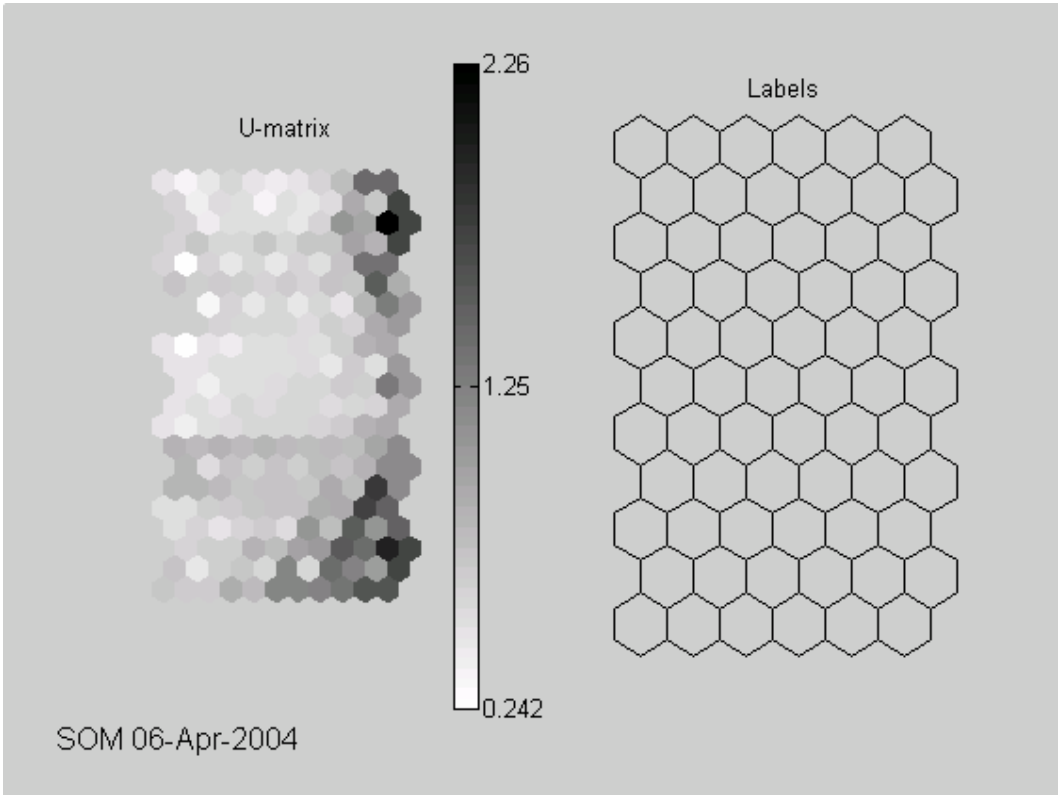
```
%    BUOC 4: TRUC QUAN HOA SELF-ORGANIZING MAP: SOM_SHOW
%    ===================================================
colormap(1-gray)
som_show(sMap,'norm','d')

pause % An phim bat ky de tiep tuc...
```



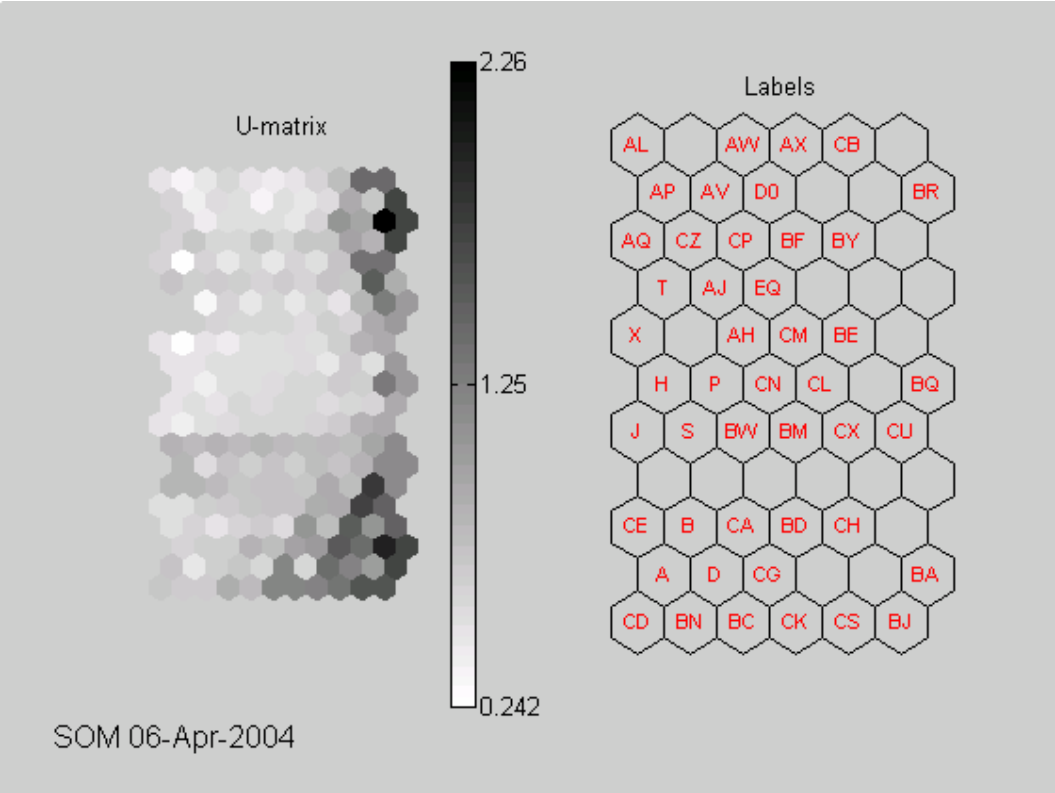
```
%    BUOC 4: TRUC QUAN HOA SELF-ORGANIZING MAP: SOM_SHOW
%    ===================================================
h=zeros(sMap.topol.msize); h(1,2) = 1;
som_show_add('hit',h(:),'markercolor','r','markersize',0.5,'subplot','all')

pause % An phim bat ky de tiep tuc...
```



```
%    BUOC 4: TRUC QUAN HOA SELF-ORGANIZING MAP: SOM_SHOW
%    ===================================================
som_show(sMap,'umat','all','empty','Labels')

pause % An phim bat ky de tiep tuc...
```



```
%    BUOC 4: TRUC QUAN HOA SELF-ORGANIZING MAP: SOM_SHOW
%    ===================================================
som_show_add('label',sMap,'Textsize',8,'TextColor','r','Subplot',2)

pause % An phim bat ky de tiep tuc...
```



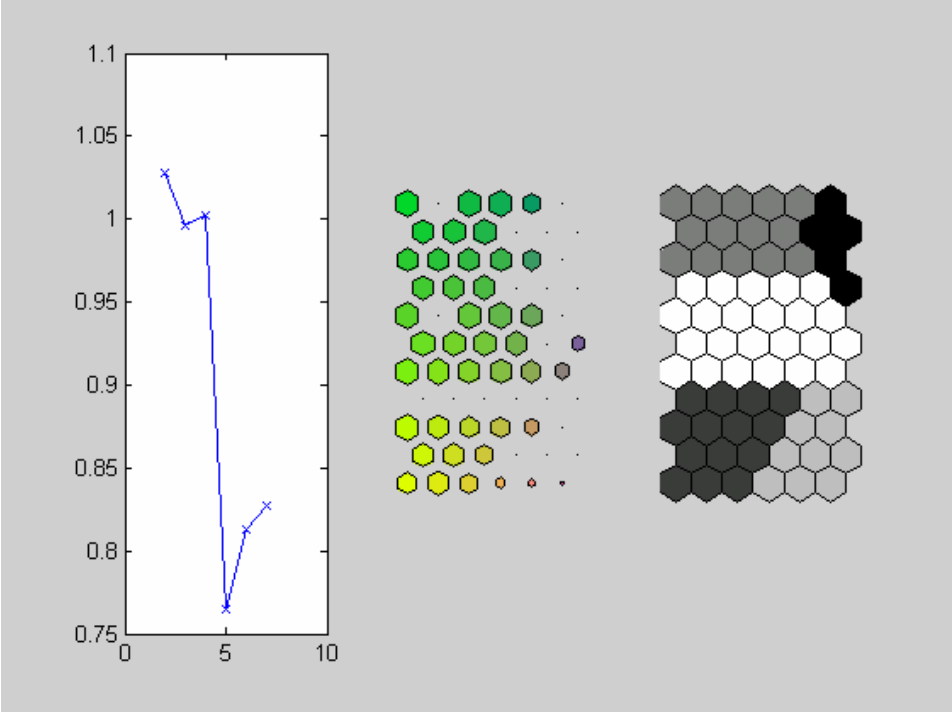
Kết quả trên cho thấy thông tin khách hàng sử dụng công cụ SOM ToolBox có 03 cụm:

Cụm 1: có khách hàng BR

Cụm 2: gồm các khách hàng A, D, FA, CE, B, CA, BD, CH, BA, CD, BN, BC, CK, CS, BJ.

Cụm 3: gồm các khách hàng AL, AW, AX, CB, AP, AV, DO, AQ, CZ, CP, BF, BY, T, AJ, EQ, X, AH, CM, BE, H, P, CN, CL, BQ, J, S, BW, BM, CX, CU.

```
%    STEP5: CLUSTERING OF THE MAP
%    ====================
sM = som_autolabel(sM,sD,'vote');
kich thuoc cua bmu: 150
kich thuoc cua Labels: 66

subplot(1,3,1)
[c,p,err,ind] = kmeans_clusters(sM, 7); %Chia SOM thành 07 cum
n_max: 7
c_max: 5
 plot(1:length(ind),ind,'x-')
[dummy,i] = min(ind)
dummy =    0.7652
i =  5 %So cum co duoc tu thuat toán

subplot(1,3,2)
[Pd,V,me,l] = pcaproj(sD,2); Pm = pcaproj(sM,V,me);
Code = som_colorcode(Pm);
hits = som_hits(sM,sD);
U = som_umat(sM);
Dm = U(1:2:size(U,1),1:2:size(U,2));
Dm = 1-Dm(:)/max(Dm(:)); Dm(find(hits==0)) = 0;
som_cplane(sM,Code,Dm);

subplot(1,3,3)
som_cplane(sM,cl)

pause % Strike any key to continue...
```



3.4.3 So sánh với các công cụ khác

Cho đến nay phần lớn các ứng dụng của SOM được xây dựng bằng các phần mềm bởi các nhà nghiên cứu. Công cụ SOM ToolBox và SOM_PAK là các công cụ có sẵn và không cần bản quyền. Trong phạm vi của luận văn đã sử dụng công cụ SOM ToolBox để áp dụng cho bài toán phân loại khách hàng tín dụng của Ngân hàng. Công cụ SOM_PAK, một công cụ của có giá trị của trường Đại học Kỹ thuật Helsinki. SOM_PAK đặc biệt phù hợp với các nghiên cứu khoa học chạy trên máy UNIX, không dùng cho các hệ điều hành của Microsoft (MS DOS, WINDOWS).

Ngoài ra, còn có các công cụ phần mềm thương mại cho SOM có trên thị trường [8]. Về cơ bản các phần mềm này cũng được xây dựng là như nhau. Tuy nhiên các phần mềm thương mại được thiết kê phù hợp với các hệ điều hành chuẩn và có thêm bước xử lý trước và xử lý sau dữ liệu. Sau đây là một danh sách các phần mềm thương mại hiện có:

1. SAS Neural Network Application
2. Professional II+ from NeuralWorks
3. MATLAB Neural Network Toolbox
4. NeuroShell2/NeuroWindows
5. NeuroSolutions v3.0
6. NeuroLab, A Neural Network Library
7. havFmNet++
8. Neural Connection
9. Trajan 2.1 Neural Network Simulator
10. Viscovery®

Một công cụ mới nhất hiện nay là Viscovery®, một sản phẩm của Eudaptics Software GmbH, là công cụ có giao diện thân thiện, linh hoạt và là công cụ mạnh cho việc tạo SOM. Viscovery® cung cấp một số đặc điểm quan trọng cần thiết

trong các ứng dụng tài chính, kinh tế và marketing mà ở các công cụ không có bản quyền không có được.

Dưới đây là một số so sánh giữa các công cụ SOM với nhau [8]

Các chỉ tiêu	Viscovery®	SOM_PAK	SOM Toolbox	NeNet
Hệ điều hành	Windows 95 Windows NT 4.0	UNIX Ms DOS	MatLab Version 5.0 trở lên	Windows
Tiền xử lý	Có 4 chọn lựa	không có	Có	Có
Đặc điểm SOM				
• Thuật toán	Thuật toán chuẩn	Thuật toán chuẩn	Thuật toán chuẩn	Thuật toán chuẩn
• Kích thước ma trận	Không giới hạn.	Không giới hạn.	Không giới hạn.	Không giới hạn.
• Khởi tạo ma trận	Lục giác.	Chữ nhật, lục giác.	Chữ nhật, lục giác.	Chữ nhật, lục giác.
• Huấn luyện	Mặt phẳng chính	Tuyến tính, ngẫu nhiên	Tuyến tính, ngẫu nhiên	Tuyến tính, ngẫu nhiên
• Gán nhãn	Định nghĩa trước	Bất kỳ giai đoạn nào	Bất kỳ giai đoạn nào	Bất kỳ giai đoạn nào
• Xử lý thành phần thiếu	Tự động, bằng tay, kéo thả	Tự động, bằng tay	Tự động, bằng tay	Tự động, bằng tay
• Tốc độ	Có thể xử lý, Nhanh	Có thể xử lý, Nhanh	Có thể xử lý, Vừa phải	Có thể xử lý, Nhanh
• Giới hạn đầu	Không	Không	Không	Tối đa [100x100]

vào				
• Trực quan hoá	U-matrix, component planes, trajectories, Iso-contours	U-matrix, component planes, trajectories	U-matrix, component planes, trajectories, hit historgrams	U-matrix, component planes, trajectories, hit historgrams
Xử lý sau	Có	Có	Không	Không
Giao diện	Thân thiện. Giao diện OLE: MS Excel, Text file, SQL & DB2	Câu lệnh C. Giao diện OLE: Text file	GUI (Matlab) Giao diện OLE: Text file	GUI (Windows 95) Giao diện OLE: Text file

3.5 Kết luận chương 3

Áp dụng phương pháp SOM vào bài toán cụ thể trong Ngân hàng, bài toán phân tích thông tin khách hàng là các Doanh nghiệp có nhu cầu vay vốn. Nội dung chính trong chương này là:

- Tìm hiểu về quy trình tác nghiệp tại phòng Tín dụng của Ngân hàng đã giải quyết bài toán.
- Tìm hiểu bộ công cụ SOM ToolBox, từ đó xây dựng chương trình giải quyết bài toán.
- Một số kết quả thu được khi chạy chương trình.
- Đánh giá, so sánh bộ công cụ SOM Toolbox với các công cụ khác trên thị trường.

KẾT LUẬN

Mạng nơron là một phương pháp rất thích hợp trong khai phá dữ liệu với mô hình học máy, đặc biệt là học không giám sát. Với trên 5000 ứng dụng trên nhiều lĩnh vực, thuật toán học mạng nơron theo SOM rất hữu dụng trong các bài toán tài chính kinh tế. Nhiều công trình nghiên cứu đã khẳng định thuật toán SOM là phù hợp với các ứng dụng có khối lượng dữ liệu lớn như dữ liệu trong Ngân hàng.

1. Luận văn đã thực hiện được kết quả sau:

- Trình bày một cách tổng quát về mô hình mạng nơron và ứng dụng mạng nơron trong khai phá dữ liệu. Trình bày một cách hệ thống các giải pháp học mạng nơron không giám sát và có giám sát.
- Nghiên cứu, phân tích việc sử dụng thuật toán SOM giải quyết bài toán phân cụm theo mô hình mạng nơron.
- Nghiên cứu cấu trúc hoạt động của bộ công cụ SOM Toolbox và phương pháp sử dụng công cụ để giải quyết bài toán phân cụm dữ liệu.
- Xây dựng bài toán phân tích thông tin khách hàng tại Ngân hàng và sử dụng công cụ SOM Toolbox để giải quyết bài toán được đề xuất. Các kết quả thử nghiệm là phù hợp với các phân tích của các nhà chuyên môn trong lĩnh vực Ngân hàng.

2. Trong quá trình nghiên cứu để hoàn thành luận văn, thông qua việc tổng hợp và phân tích một hoạt động cốt yếu của Ngân hàng là phân tích thông tin khách hàng vay vốn, tôi nhận thấy việc phát triển nội dung luận văn là rất cần thiết để sử dụng mạng nơron trong khai phá dữ liệu Ngân hàng. Để mở rộng kết quả nội dung của luận văn này, hướng nghiên cứu và phát triển tiếp theo là *tìm hiểu các phương pháp sinh luật từ mạng nơron* (phần này đã được đề cập trong chương 1) và *ứng dụng hỗ trợ quyết định trong đầu tư tài chính*.

TÀI LIỆU THAM KHẢO

TÀI LIỆU TIẾNG VIỆT

[1]. Nguyễn Đình Thúc (2000), *Trí tuệ nhân tạo Mạng nơron phương pháp & ứng dụng*, Nhà xuất bản Giáo Dục.

[2]. Trần Đức Minh (2002), *Mạng nơron truyền thẳng và thuật toán lan truyền ngược*, Luận văn Thạc sĩ cao học, Khoa Công nghệ, Trường Đại học Quốc gia Hà Nội.

TÀI LIỆU TIẾNG ANH

[3]. Bart De Ketelaere, Demitrios Moshou, Peter Coucke, Josse De Baerdemaeker (1997), *A herachical Self-Organizing Map for classification problems*.

[4]. Boris Kovalerchuk & Evgenii Vityaev (2001), *Data mining in finance advances in Relational and Hybrid Methods*, Kluwer Academic Publishers.

[5] David Sommer & Martin Golz (2001), *Clustering of EEG-Segments Using Heirarchical Agglomerative Methods and Self-Organizing Maps*, University of Applied Sciences Germany, Department of Computer Science.

[6].Ed Guido Deboeck & Teuvo Khohonen (1998), *Visual Intelligence in Finance using Self-organizing Maps,* Chapter 7: Self-organizing Maps for Initial Data Analysis: let Financial Data Speak for Themselves, Speinger Verlag.

[7]. Guido Deboeck, Ph.D (1999), *Self-Organizing Maps facilitate knowleadge discovery in finance.*

[8]. Guido Deboeck, Ph.D (2000), *Public domain versus commercial tools for creating Self-Organizing Maps*.

[9]. J. Han and M. Kamber (2001), *Data Mining - Concepts and Techniques,* Chapter 8: Cluster Analysis. Morgan Kaufmann.

[10]. Juha Vesanto (1997), *Data Mining techniques based on the Self-Organizing Map*, Thesis for the degree of Master in Engineering, Helsinki University of Technology.

[11]. Juha Vesanto (2000), *Using SOM in Data Mining*, Licentiate's thesis, Helsinki University of Technology.

[12]. Mark W.Craven & Jude W.Shavlik (2000), *Using Neural Networks for Data Mining*, Submitted to the Future Generation Computer Systems specical issues on Data Mining.

[13] Merja Oja, Samuel Kaski, and Teuvo Kohonen (2003), *Bibliography of Self-Organizing Map (SOM)* Papers: 1998-2001 Addendum, Neural Computing Surveys, 3: 1-156.

[14]. Mark W.Craven (1996), *Extracting comprehensible models from trained neural networks*, Chapter 7: The Boosting – Based Perceptron learning algorithm, Doctor of philosophy (Computer Sciences).

[15].Tom Gemano (1999), *Self Organizing Maps*.

[16]. Usama M.Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth & Ramasamy Uthrusamy (1996), *Advanes in Knowledge Discovery and Data mining*,AAAI Press/The MIT Press.