

**VIỆN HÀN LÂM KHOA HỌC VÀ CÔNG NGHỆ VIỆT NAM
VIỆN CƠ HỌC**

**BÁO CÁO KẾT QUẢ THỰC HIỆN ĐỀ TÀI CƠ SỞ
CẤP VIỆN CƠ HỌC NĂM 2017**

**ĐỀ TÀI:
ỨNG DỤNG MẠNG NƠON NHÂN TẠO
VÀO BÀI TOÁN DỰ BÁO**

Chủ nhiệm Đề tài: TS. Nguyễn Chính Kiên

HÀ NỘI – 2017

DANH SÁCH CÁN BỘ THAM GIA THỰC HIỆN ĐỀ TÀI

TT	Họ và tên	Phòng chuyên môn
1	TS Nguyễn Chính Kiên	Thủy động lực và Giảm nhẹ thiên tai trong lưu vực
2	TS Nguyễn Tiến Cường	Thủy động lực và Giảm nhẹ thiên tai trong lưu vực
3	CN Nguyễn Tuấn Anh	Thủy động lực và Giảm nhẹ thiên tai trong lưu vực

MỤC LỤC

DANH MỤC KÝ HIỆU, CHỮ VIẾT TẮT THƯỜNG SỬ DỤNG	ii
DANH MỤC CÁC BẢNG	iii
DANH MỤC CÁC HÌNH VẼ VÀ ĐỒ THỊ	iv
MỞ ĐẦU	1
I. GIỚI THIỆU VỀ MẠNG NƠON NHÂN TẠO	2
1.1 Khái niệm, mô hình và khả năng ứng dụng	2
1.2 Mạng nơon truyền thẳng nhiều lớp.....	9
II. MỘT SỐ GIẢI THUẬT TỐI ƯU TRONG SỐ MẠNG NƠON NHÂN TẠO ..	14
2.1 Giải thuật Lan truyền ngược sai số	14
2.2 Giải thuật Di truyền.....	20
2.3 Kết hợp các giải thuật.....	23
III. XÂY DỰNG PHẦN MỀM DỰ BÁO BẰNG MẠNG NƠON NHÂN TẠO	26
3.1 Ngôn ngữ và giao diện phần mềm	26
3.2 Mô đun giải thuật Di truyền	29
3.3 Mô đun giải thuật Lan truyền ngược sai số.....	30
3.4 Kết hợp các giải thuật.....	31
3.5 Một số kỹ thuật xử lý	33
3.5.1 Kỹ thuật tính toán song song	33
3.5.2 Kỹ thuật phân tích dữ liệu đầu vào Wavelet	43
IV. MỘT SỐ KẾT QUẢ TÍNH TOÁN	47
4.1 Kết quả mô phỏng và dự báo thủy lực lưu vực đồng bằng sông Hồng	47
4.2 Kết quả mô phỏng và dự báo thủy văn lưu lượng vào hồ	54
4.3 Kết quả mô phỏng và dự báo độ mặn tại vùng Tứ Giác Long Xuyên	56
KẾT LUẬN.....	59
TÀI LIỆU THAM KHẢO	61

**DANH MỤC KÝ HIỆU,
CHỮ VIẾT TẮT THƯỜNG SỬ DỤNG**

ANN	Artificial Neural Network - mạng thần kinh nhân tạo
AI	Trí tuệ nhân tạo
GA	Genetic Algorithm - Giải thuật Di truyền
BP	Backpropagation - Giải thuật Lan truyền ngược sai số
RMS	Sai số căn quân phương
NSE	Chỉ số đánh giá Nash-Sutcliffe
GPU	Đơn vị xử lý đồ họa
CUDA	Thư viện tính toán song song trên card đồ họa Nvidia
CWT	Biến đổi Wavelet liên tục
DWT	Biến đổi Wavelet rời rạc
TL	Thủy lực

DANH MỤC CÁC BẢNG

Bảng 1.1 Các hàm kích hoạt thường được sử dụng	4
Bảng 2.1 Giá trị đầu vào và ra của bài toán XOR	23
Bảng 2.2 So sánh khả năng hội tụ của mạng khi sử dụng hai phương pháp học GA và BP với sai số dừng lặp khác nhau	25
Bảng 3.1 Thông số card NVIDIA GeForce GTX 1060 6GB	39
Bảng 3.2 Thông số các bài toán thử nghiệm.	41
Bảng 3.3 Thời gian tính trung bình của các bài toán thử nghiệm.....	41
Bảng 4.1 Kết quả tính phương án 1	48
Bảng 4.2 Kết quả tính phương án 2	50
Bảng 4.3 Kết quả tính phương án 3	51
Bảng 4.4 Kết quả tính phương án 4	52
Bảng 4.5 Kết quả tính phương án 5	54
Bảng 4.6 Kết quả tính phương án dự báo lưu lượng vào hồ Hòa Bình	56
Bảng 4.7 Kịch bản và kết quả độ mặn tính toán của mô hình thủy lực	57
Bảng 4.8 Kết quả tính phương án tính mặn	58

DANH MỤC CÁC HÌNH VẼ VÀ ĐỒ THỊ

Hình 1.1 Cấu tạo của tế bào noron sinh học	2
Hình 1.2 Noron nhân tạo.....	3
Hình 1.3 Mạng noron chỉ có 1 nút và có sự phản hồi.....	6
Hình 1.4 Mạng noron truyền thẳng 1 lớp (Single-layer feedforward network)	6
Hình 1.5 Mạng noron hồi quy 1 lớp	6
Hình 1.6 Mạng MLP tổng quát	7
Hình 1.7 Hàm sigmoid $g(x) = 1/(1+e^{-x})$	11
Hình 2.1 Lan truyền tín hiệu trong quá trình học theo pp lan truyền ngược sai số	14
Hình 2.2 Sai số E được xét là hàm của trọng số W	16
Hình 2.3 Minh họa về ý nghĩa của quán tính trong thực tế	19
Hình 2.4 Mô hình mạng noron cho bài toán XOR	23
Hình 3.1 Giao diện của phần mềm dự báo bằng mạng noron.	27
Hình 3.2 Lựa chọn phương án tính	28
Hình 3.3 Cấu trúc mạng, thông số và các giá trị ban đầu của phương án đã chọn.	28
Hình 3.4 Sơ đồ thuật toán của giải thuật di truyền	29
Hình 3.5 Sơ đồ thuật toán lan truyền ngược sai số	31
Hình 3.6 Sơ đồ thuật toán kết hợp giải thuật Di truyền và Lan truyền ngược sai số.....	33
Hình 3.7 Mô hình Fork-Join	35
Hình 3.8 So sánh kiến trúc CPU và GPU.	36
Hình 3.9 Tiến trình thực hiện của 1 chương trình CUDA	37
Hình 3.10 Mô hình siêu máy tính có card mạng kết nối chuyên dụng GPUDirect RDMA.....	38
Hình 3.11 Sơ đồ khối cách thực hiện song song trên GPU	40
Hình 3.12 Biểu đồ hiệu năng card đồ họa khi đang xử lý song song trên GPU bài toán 2.....	42
Hình 3.13 Phân tích đa phân giải sử dụng biến đổi wavelet rời rạc	45
Hình 3.14 Phân tích chuỗi dữ liệu mực nước trạm Hưng Yên	46
Hình 4.1 Sơ đồ mạng sông vùng nghiên cứu.....	47
Hình 4.2 Kết quả so sánh mực nước Sơn Tây năm 2000,2002,2003 giữa thực đo và mạng ANN hiệu chỉnh.	48

Hình 4.3 Kết quả so sánh mực nước Sơn Tây năm 2004 giữa thực đo và mạng ANN kiểm định.	48
Hình 4.4 Kết quả so sánh mực nước Sơn Tây năm 2000, 2002, 2003 và nửa năm 2004 giữa thực đo và mạng ANN hiệu chỉnh.	49
Hình 4.5 Kết quả so sánh mực nước Sơn Tây nửa sau năm 2004 giữa thực đo và mạng ANN kiểm định sau khi học bổ sung.	49
Hình 4.6 Kết quả so sánh mực nước Sơn Tây và Hà Nội năm 2000, 2002, 2003 giữa thực đo và mạng ANN hiệu chỉnh.	50
Hình 4.7 Kết quả so sánh mực nước Sơn Tây và Hà Nội năm 2004 giữa thực đo và mạng ANN kiểm định.....	51
Hình 4.8 Kết quả so sánh mực nước Sơn Tây năm 2015 giữa thực đo và mạng ANN hiệu chỉnh.	52
Hình 4.9 Kết quả so sánh mực nước Sơn Tây năm 2016 giữa thực đo và mạng ANN kiểm định	52
Hình 4.10 Kết quả so sánh mực nước Sơn Tây năm 2015 giữa thực đo và mạng ANN hiệu chỉnh có bổ sung số liệu trạm Hưng Yên.	53
Hình 4.11 Kết quả so sánh mực nước Sơn Tây năm 2016 giữa thực đo và mạng ANN kiểm định có bổ sung dữ liệu trạm Hưng Yên.....	54
Hình 4.12 Kết quả so sánh lưu lượng vào hồ Hòa Bình giữa thực đo và mạng ANN tính kiểm định	55
Hình 4.13 Kết quả so sánh lưu lượng vào hồ Hòa Bình giữa thực đo và mạng ANN tính dự báo.....	56
Hình 4.14 Mô hình tính tại vùng Tứ Giác Long Xuyên	57
Hình 4.15 Kết quả so sánh độ mặn Rạch Giá giữa thực đo và mạng ANN hiệu chỉnh.....	58
Hình 4.16 Kết quả so sánh độ mặn Rạch Giá giữa thực đo và mạng ANN dự báo.....	58

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Dự báo thủy văn, thủy lực là công việc dự báo trong tương lai trước một cách có khoa học về trạng thái biến đổi các yếu tố thủy văn, tuy nhiên sự biến đổi này là một quá trình tự nhiên phức tạp, chịu tác động của rất nhiều yếu tố. Tính biến động của các yếu tố này phụ thuộc vào cả không gian và thời gian nên gây khó khăn rất lớn cho quá trình dự báo, tìm ra được mối liên quan giữa các yếu tố. Thêm vào đó, do thiếu các trạm quan trắc cần thiết và thiếu sự kết hợp giữa các ngành liên quan cho nên dữ liệu quan trắc thực tế thường là không đầy đủ, không mang tính chất đại diện.

Hiện nay, có rất nhiều phương pháp dự báo đã được đưa ra dựa trên mô hình vật lý và mô hình toán học, kết quả của các mô hình nói trên đã đạt được một số thành công đáng ghi nhận. Tuy nhiên, vấn đề tìm kiếm phương pháp đủ tốt, đáp ứng các yêu cầu thực tế giải quyết bài toán dự báo thủy văn, thủy lực vẫn là nội dung nghiên cứu thời sự hiện nay.

Một hướng tiếp cận mới để mô hình hoá các hiện tượng thủy văn, thủy lực là dựa trên các công nghệ học máy. Mô hình này dựa trên một cơ sở dữ liệu thực tế đủ lớn và áp dụng kỹ thuật tính toán hiện đại - phương pháp học máy, một phần của trí tuệ nhân tạo – tận dụng sự phát triển của công nghệ tính toán đã đang là một trong những lĩnh vực nghiên cứu phát triển mạnh mẽ.

2. Mục đích nghiên cứu

- Mạng nơron nhân tạo và các giải thuật liên quan,
- Xây dựng phần mềm dự báo bằng mạng nơron nhân tạo.

3. Phương pháp nghiên cứu

- Phân tích, thống kê và tính toán các tư liệu thu thập được,
- Mô hình hóa bài toán dự báo bằng việc phát triển chương trình tính toán ngôn ngữ Fortran.

CHƯƠNG I

GIỚI THIỆU VỀ MẠNG NƠN NHÂN TẠO

1.1 Khái niệm, mô hình và khả năng ứng dụng

1.1.1 Khái niệm [5]

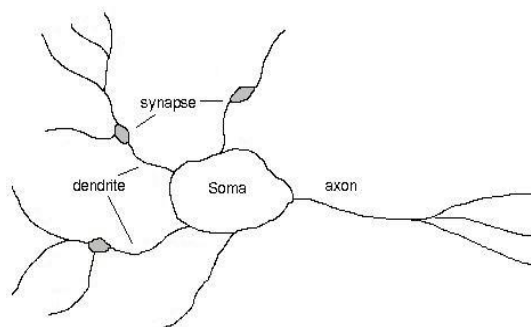
Theo các nhà nghiên cứu sinh học, hệ thống thần kinh của con người bao gồm khoảng 10^{11} tế bào thần kinh, thường gọi là các nơron. Mỗi tế bào nơron gồm ba phần:

- Thân nơron với nhân bên trong (gọi là soma), là nơi tiếp nhận hay phát ra các xung động thần kinh.

- Một hệ thống dạng cây các dây thần kinh vào (gọi là dendrite) để đưa tín hiệu tới nhân nơron. Các dây thần kinh vào tạo thành một lưới dày đặc xung quanh thân nơron, chiếm diện tích khoảng $0,25 \text{ mm}^2$

- Đầu dây thần kinh ra (gọi là sợi trục axon) phân nhánh dạng hình cây, có thể dài từ một cm đến hàng mét. Chúng nối với các dây thần kinh vào hoặc trực tiếp với nhân tế bào của các nơron khác thông qua các khớp nối (gọi là synapse). Có hai loại khớp nối, khớp nối kích thích (excitatory) sẽ cho tín hiệu qua nó để tới nơron còn khớp nối ức chế (inhibitory) có tác dụng làm cản tín hiệu tới nơron. Người ta ước tính mỗi nơron trong bộ não của con người có khoảng 10^4 khớp nối.

Chức năng cơ bản của các tế bào nơron là liên kết với nhau để tạo nên hệ thống thần kinh điều khiển hoạt động của cơ thể sống. Các tế bào nơron truyền tín hiệu cho nhau thông qua các dây thần kinh vào và ra, các tín hiệu đó có dạng xung điện và được tạo ra từ các quá trình phản ứng hoá học phức tạp. Tại nhân tế bào, khi điện thế của tín hiệu vào đạt tới một ngưỡng nào đó thì nó sẽ tạo ra một xung điện dẫn tới trục dây thần kinh ra. Xung này truyền theo trục ra tới các nhánh rẽ và tiếp tục truyền tới các nơron khác.

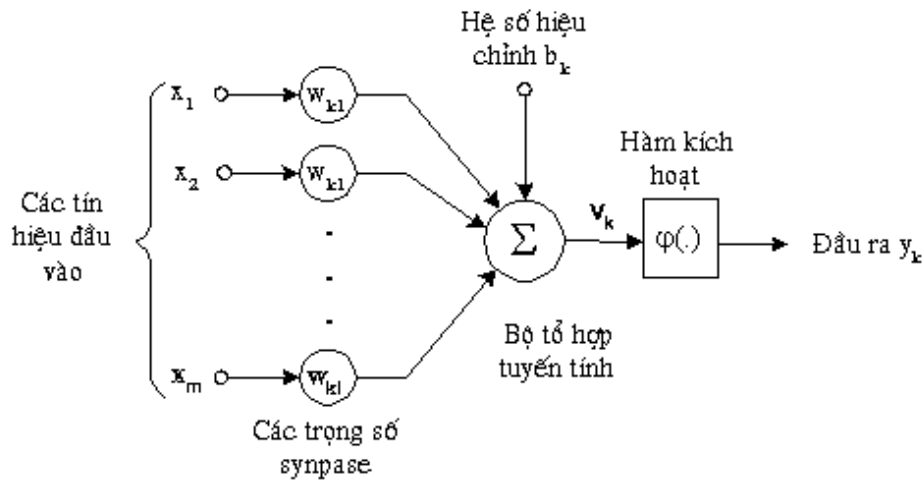


Hình 1.1 Cấu tạo của tế bào nơron sinh học

Mạng nơron nhân tạo, Artificial Neural Network (ANN) gọi tắt là mạng nơron, neural network, là một mô hình xử lý thông tin phỏng theo cách thức xử lý thông tin của các hệ nơron sinh học. Nó được tạo nên từ một số lượng lớn các phần tử (gọi là phần tử xử lý hay

noron) kết nối với nhau thông qua các liên kết (gọi là trọng số liên kết) làm việc như một thể thống nhất để giải quyết một vấn đề cụ thể nào đó.

Một noron là một đơn vị xử lý thông tin và là thành phần cơ bản của một mạng noron. Cấu trúc của một noron được mô tả trên hình dưới.



Hình 1.2 Noron nhân tạo

Các thành phần cơ bản của một noron nhân tạo bao gồm:

- Tập các đầu vào: Là các tín hiệu vào (input signals) của noron, các tín hiệu này thường được đưa vào dưới dạng một vec-tơ m chiều.


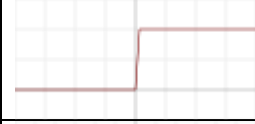

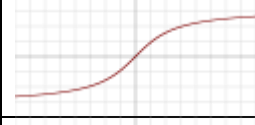

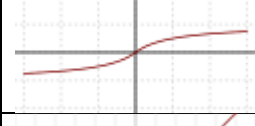
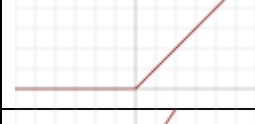
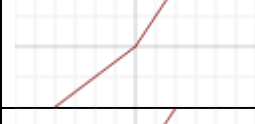
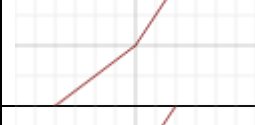

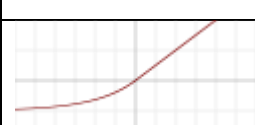
- Tập các liên kết: Mỗi liên kết được thể hiện bởi một trọng số (gọi là trọng số liên kết – Synaptic weight). Trọng số liên kết giữa tín hiệu vào thứ j với noron k thường được kí hiệu là w_{jk} . Thông thường, các trọng số này được khởi tạo một cách ngẫu nhiên ở thời điểm khởi tạo mạng và được cập nhật liên tục trong quá trình học mạng.

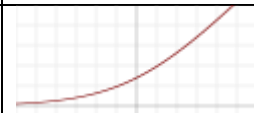
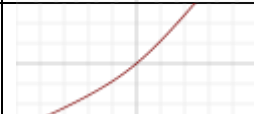
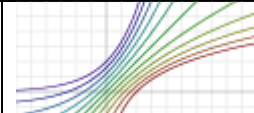
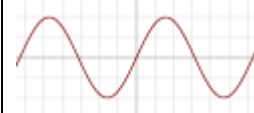
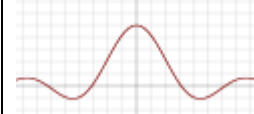
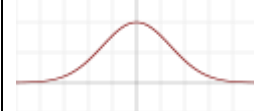
- Bộ tổng (Summing function): Thường dùng để tính tổng của tích các đầu vào với trọng số liên kết của nó.

- Ngưỡng (còn gọi là một độ lệch - bias): Ngưỡng này thường được đưa vào như một thành phần của hàm truyền.

- Hàm truyền (Transfer function) – còn gọi là Hàm kích hoạt (Activation function): Hàm này được dùng để giới hạn phạm vi đầu ra của mỗi noron. Nó nhận đầu vào là kết quả của hàm tổng và ngưỡng đã cho. Thông thường, phạm vi đầu ra của mỗi noron được giới hạn trong đoạn $[0,1]$ hoặc $[-1, 1]$. Các hàm truyền rất đa dạng được liệt kê trong bảng 1.1, có thể là các hàm tuyến tính hoặc phi tuyến. Việc lựa chọn hàm truyền nào là tùy thuộc vào từng bài toán và kinh nghiệm của người thiết kế mạng.

Bảng 1.1 Các hàm kích hoạt thường được sử dụng

ST T	Tên hàm	Dạng đồ thị	Phương trình
1	Identity		$f(x) = x$
2	Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
3	Logistic		$f(x) = \frac{1}{1 + e^{-x}}$
4	TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
5	ArcTan		$f(x) = \tan^{-1}(x)$
6	Softsign		$f(x) = \frac{x}{1 + x }$
7	Rectified linear unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
8	Leaky rectified linear unit (Leaky ReLU)		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
9	Parameteric rectified linear unit (PReLU)		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
10	Randomized leaky rectified linear unit (RReLU)		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
11	Exponential linear unit (ELU)		$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
12	Scaled exponential linear unit (SELU)		$f(\alpha, x) = \lambda \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ with $\lambda = 1.0507$ and $\alpha = 1.67326$
13	S-shaped rectified linear activation unit (SReLU)		$f(x) = \begin{cases} t_l + a_l(x - t_l) & \text{for } x \leq t_l \\ x & \text{for } t_l < x < t_r \\ t_r + a_r(x - t_r) & \text{for } x \geq t_r \end{cases}$ t_l, a_l, t_r, a_r are parameters

14	Adaptive piecewise linear (APL)		$f(x) = \max(0, x) + \sum_{s=1}^S a_s^s \max(0, -x + b_s^s)$
15	SoftPlus		$f(x) = \ln(1 + e^x)$
16	Bent identity		$f(x) = \frac{\sqrt{x^2 + 1} - 1}{2} + x$
17	SoftExponential		$f(\alpha, x) = \begin{cases} -\frac{\ln(1-\alpha(x+\alpha))}{\alpha} & \text{for } \alpha < 0 \\ x & \text{for } \alpha = 0 \\ \frac{e^{\alpha x} - 1}{\alpha} + \alpha & \text{for } \alpha > 0 \end{cases}$
18	Sinusoid		$f(x) = \sin(x)$
19	Sinc		$f(x) = \begin{cases} 1 & \text{for } x = 0 \\ \frac{\sin(x)}{x} & \text{for } x \neq 0 \end{cases}$
20	Gaussian		$f(x) = e^{-x^2}$

– Đầu ra: Là tín hiệu đầu ra của một neuron, với mỗi neuron sẽ có tối đa là một đầu ra.

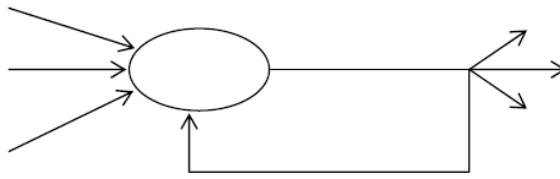
Như vậy tương tự như neuron sinh học, neuron nhân tạo cũng nhận các tín hiệu đầu vào, xử lý (nhân các tín hiệu này với trọng số liên kết, tính tổng các tích thu được rồi gửi kết quả tới hàm truyền), và cho một tín hiệu đầu ra (là kết quả của hàm truyền).

1.1.2 Mô hình mạng neuron nhân tạo [6]

Mặc dù mỗi neuron đơn lẻ có thể thực hiện những chức năng xử lý thông tin nhất định, sức mạnh của tính toán neuron chủ yếu có được nhờ sự kết hợp các neuron trong một kiến trúc thống nhất. Một mạng neuron là một mô hình tính toán được xác định qua các tham số: kiểu neuron (như là các nút nếu ta coi cả mạng neuron là một đồ thị), kiến trúc kết nối (sự tổ chức kết nối giữa các neuron) và thuật toán học (thuật toán dùng để học cho mạng).

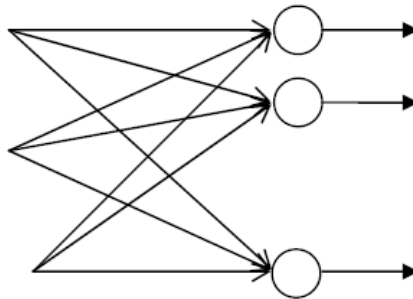
Về bản chất một mạng neuron có chức năng như là một hàm ánh xạ $F: X \rightarrow Y$, trong đó X là không gian trạng thái đầu vào (input state space) và Y là không gian trạng thái đầu ra (output state space) của mạng. Các mạng chỉ đơn giản là làm nhiệm vụ ánh xạ các vec-tơ đầu vào $x \in X$ sang các vec-tơ đầu ra $y \in Y$ thông qua “bộ lọc” (filter) các trọng số. Tức là $y = F(x) = s(W, x)$, trong đó W là ma trận trọng số liên kết. Hoạt động của mạng thường là các tính toán số thực trên các ma trận.

Trong bộ não của con người, các tế bào nơron liên kết với nhau chằng chịt và tạo nên một mạng lưới vô cùng phức tạp, tuy nhiên mạng nơron nhân tạo được chia thành các loại chính sau:



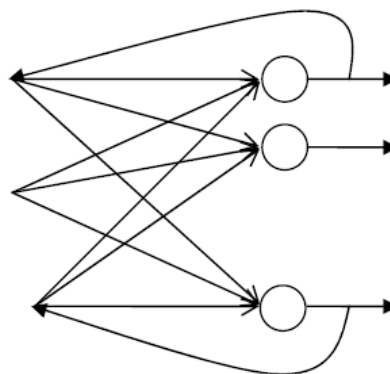
Hình 1.3 Mạng nơron chỉ có 1 nút và có sự phản hồi

Mạng nơron truyền thẳng một lớp (perceptron) là loại mạng chỉ có lớp nơron đầu vào và một lớp nơron đầu ra (thực chất lớp nơron đầu vào không có vai trò xử lý, do đó ta nói mạng chỉ có một lớp). Loại mạng này còn được gọi là mạng perceptron một lớp. Mỗi nơron đầu ra có thể nhận tín hiệu từ các đầu vào x_1, x_2, \dots, x_m để tạo ra tín hiệu đầu ra tương ứng.



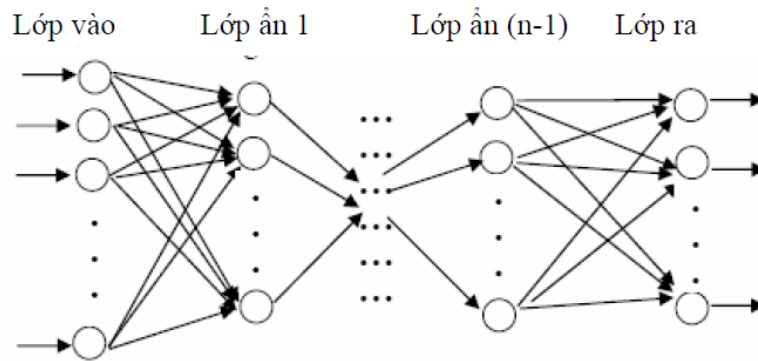
Hình 1.4 Mạng nơron truyền thẳng 1 lớp (Single-layer feedforward network)

Mạng có phản hồi (feedback network) là mạng mà đầu ra của một nơron có thể trở thành đầu vào của nơron trên cùng một lớp hoặc của lớp trước đó. Mạng feedback có chu trình khép kín gọi là mạng quy hồi (recurrent network).



Hình 1.5 Mạng nơron hồi quy 1 lớp

Mô hình mạng truyền thẳng nhiều lớp: Mô hình mạng nơron được sử dụng rộng rãi nhất là mô hình mạng nhiều tầng truyền thẳng (MLP: Multi Layer Perceptron). Một mạng MLP tổng quát là mạng có n ($n \geq 2$) lớp (thông thường lớp đầu vào không được tính đến): trong đó gồm một lớp đầu ra (lớp thứ n) và $(n-1)$ lớp ẩn.



Hình 1.6 Mạng MLP tổng quát

Một số kết quả đã được chứng minh với mạng MLP:

- Bất kì một hàm Boolean nào cũng có thể biểu diễn được bởi một mạng MLP 2 lớp trong đó các nơron sử dụng hàm truyền sigmoid.
- Tất cả các hàm liên tục đều có thể xấp xỉ bởi một mạng MLP 2 lớp sử dụng hàm truyền sigmoid cho các nơron lớp ẩn và hàm truyền tuyến tính cho các nơron lớp ra với sai số nhỏ tùy ý.
- Mọi hàm bất kỳ đều có thể xấp xỉ bởi một mạng MLP 3 lớp sử dụng hàm truyền sigmoid cho các nơron lớp ẩn và hàm truyền tuyến tính cho các nơron lớp ra.

Mô hình mạng kết hợp : Trong thực tế, người ta thường kết hợp nhiều loại mạng để giải quyết các bài toán. Có thể tham khảo mô hình ANN hệ thống chơi cờ vây AlphaGo của Google bao gồm kết hợp 3 mạng ANN. Mạng 1 SL có cấu trúc xoắn bao gồm 13 lớp ẩn với tập dữ liệu học có giám sát là hàng triệu nước cờ đã có thu thập từ các ván cờ các kỳ thủ đã chơi. Mạng 2 RL có cấu trúc giống mạng SL nhưng được tối ưu hóa bộ trọng số qua bộ huấn luyện là các ván cờ tự chơi với nhau giữa các máy. Mạng 3 Value dự đoán kết quả trò chơi dựa trên RL, giá trị của vị trí p được định nghĩa như 1 kỳ vọng phân phối các kết quả phát từ vị trí p đến cuối ván cờ. Cuối cùng, AlphaGo kết hợp mạng trên với thuật toán Monte Carlo Tree Search để đưa ra đánh giá vị trí cuối cùng cho nước đi tiếp theo.

Quá trình học của mạng nơron

Học là quá trình cập nhật trọng số sao cho giá trị hàm lỗi là nhỏ nhất. Một mạng nơron được huấn luyện sao cho với một tập các vec-tơ đầu vào X , mạng có khả năng tạo ra tập các vec-tơ đầu ra mong muốn Y của nó. Tập X được sử dụng cho huấn luyện mạng được gọi là tập huấn luyện (training set). Các phần tử x thuộc X được gọi là các mẫu huấn luyện (training example). Quá trình huấn luyện bản chất là sự thay đổi các trọng số liên kết của mạng. Trong quá trình này, các trọng số của mạng sẽ hội tụ dần tới các giá trị sao cho với mỗi vec-tơ đầu vào x từ tập huấn luyện, mạng sẽ cho ra vec-tơ đầu ra y như mong muốn.

Có ba phương pháp học phổ biến là học có giám sát (*supervised learning*), học không giám sát (*unsupervised learning*) và học tăng cường (*Reinforcement learning*).

Học có giám sát trong các mạng nơron

Học có giám sát có thể được xem như việc xấp xỉ một ánh xạ: $X \rightarrow Y$, trong đó X là tập các vấn đề và Y là tập các lời giải tương ứng cho vấn đề đó. Các mẫu (x, y) với $x = (x_1, x_2, \dots, x_n) \in X$, $y = (y_1, y_2, \dots, y_m) \in Y$ được cho trước. Học có giám sát trong các mạng nơron thường được thực hiện theo các bước sau:

- Bước 1: Xây dựng cấu trúc thích hợp cho mạng nơron, chẳng hạn có n nơron vào, m nơron đầu ra, và khởi tạo các trọng số liên kết của mạng.
- Bước 2: Đưa một vec-tơ x trong tập mẫu huấn luyện X vào mạng
- Bước 3: Tính vec-tơ đầu ra z của mạng
- Bước 4: So sánh vec-tơ đầu ra mong muốn t (là kết quả được cho trong tập huấn luyện) với vec-tơ đầu ra z do mạng tạo ra; nếu có thể thì đánh giá lỗi.
- Bước 5: Hiệu chỉnh các trọng số liên kết theo một cách nào đó sao cho ở lần tiếp theo khi đưa vec-tơ x vào mạng, vec-tơ đầu ra z sẽ giống với t hơn.
- Bước 6: Nếu cần, lặp lại các bước từ 2 đến 5 cho tới khi mạng đạt tới trạng thái hội tụ. Việc đánh giá lỗi có thể thực hiện theo nhiều cách, cách dùng nhiều nhất là sử dụng lỗi tức thời: $Err = (z - t)$, hoặc $Err = |z - t|$; lỗi trung bình bình phương (MSE: mean-square error): $Err = (z - t)^2/2$.

Có hai loại lỗi trong đánh giá một mạng nơron. Thứ nhất, gọi là lỗi rõ ràng (apparent error), đánh giá khả năng xấp xỉ các mẫu huấn luyện của một mạng đã được huấn luyện. Thứ hai, gọi là lỗi kiểm tra (test error), đánh giá khả năng tổng quát hóa của một mạng đã được huấn luyện, tức khả năng phản ứng với các vec-tơ đầu vào mới. Để đánh giá lỗi kiểm tra chúng ta phải biết đầu ra mong muốn cho các mẫu kiểm tra.

Thuật toán tổng quát ở trên cho học có giám sát trong các mạng nơron có nhiều cài đặt khác nhau, sự khác nhau chủ yếu là cách các trọng số liên kết được thay đổi trong suốt thời gian học. Trong đó tiêu biểu nhất là thuật toán lan truyền ngược sai số.

1.1.3 Khả năng ứng dụng của mạng nơron nhân tạo

Đặc trưng của mạng nơron nhân tạo là khả năng học. Nó có thể gần đúng mối quan hệ tương quan phức tạp giữa các yếu tố đầu vào và đầu ra của các quá trình cần nghiên cứu và khi đã học được thì việc kiểm tra độc lập thường cho kết quả tốt. Sau khi đã học xong, mạng nơron nhân tạo có thể tính toán kết quả đầu ra tương ứng với bộ số liệu đầu vào mới.

Về mặt cấu trúc, mạng nơron nhân tạo là một hệ thống gồm nhiều phần tử xử lý đơn giản cùng hoạt động song song. Tính năng này của ANN cho phép nó có thể được áp dụng để giải các bài toán lớn.

Về khía cạnh toán học, theo định lý Kolmogorov, một hàm liên tục bất kỳ $f(x_1, x_2, \dots, x_n)$ xác định trên khoảng I^n (với $I = [0,1]$) có thể được biểu diễn dưới dạng [4]:

$$f(x) = \sum_{j=1}^{2n+1} \chi_j \left(\sum_{i=1}^n \Psi_{ij}(x_i) \right)$$

trong đó: χ_j, Ψ_{ij} là các hàm liên tục một biến. Ψ_{ij} là hàm đơn điệu, không phụ thuộc vào hàm f . Mặt khác, mô hình mạng nơron nhân tạo cho phép liên kết có trọng số các phần tử phi tuyến (các nơron đơn lẻ) tạo nên dạng hàm tổng hợp từ các hàm thành phần. Do vậy, sau một quá trình điều chỉnh sự liên kết cho phù hợp (quá trình học), các phần tử phi tuyến đó sẽ tạo nên một hàm phi tuyến phức tạp có khả năng xấp xỉ hàm biểu diễn quá trình cần nghiên cứu. Kết quả là đầu ra của nó sẽ tương tự với kết quả đầu ra của tập dữ liệu dùng để luyện mạng. Khi đó ta nói mạng nơron nhân tạo đã học được mối quan hệ tương quan đầu vào - đầu ra của quá trình và lưu lại mối quan hệ tương quan này thông qua bộ trọng số liên kết giữa các nơron. Do đó, mạng nơron nhân tạo có thể tính toán trên bộ số liệu đầu vào mới để đưa ra kết quả đầu ra tương ứng.

Với những đặc điểm đó, mạng nơron nhân tạo đã được sử dụng để giải quyết nhiều bài toán thuộc nhiều lĩnh vực của các ngành khác nhau. Các nhóm ứng dụng mà mạng nơron nhân tạo đã được áp dụng rất có hiệu quả là:

- **Bài toán phân lớp:** Loại bài toán này đòi hỏi giải quyết vấn đề phân loại các đối tượng quan sát được thành các nhóm dựa trên các đặc điểm của các nhóm đối tượng đó. Đây là dạng bài toán cơ sở của rất nhiều bài toán trong thực tế: nhận dạng chữ viết, tiếng nói, phân loại gen, phân loại chất lượng sản phẩm, ...
- **Bài toán dự báo:** Mạng nơron nhân tạo đã được ứng dụng thành công trong việc xây dựng các mô hình dự báo sử dụng tập dữ liệu trong quá khứ để dự đoán số liệu trong tương lai. Đây là nhóm bài toán khó và rất quan trọng trong nhiều ngành khoa học.
- **Bài toán điều khiển và tối ưu hoá:** Nhờ khả năng học và xấp xỉ hàm mà mạng nơron nhân tạo đã được sử dụng trong nhiều hệ thống điều khiển tự động cũng như góp phần giải quyết những bài toán tối ưu trong thực tế.

Tóm lại, mạng nơron nhân tạo được xem như là một cách tiếp cận đầy tiềm năng để giải quyết các bài toán có tính phi tuyến, phức tạp và đặc biệt là trong tình huống mối quan hệ bản chất vật lý của quá trình cần nghiên cứu không dễ thiết lập tường minh.

1.2 Mạng nơron truyền thẳng nhiều lớp [1]

Mạng perceptron một lớp do F.Rosenblatt đề xuất năm 1960 là mạng truyền thẳng chỉ một lớp vào và một lớp ra không có lớp ẩn. Trên mỗi lớp này có thể có một hoặc nhiều nơron. Rosenblatt đã chứng minh rằng quá trình học của mạng Perceptron sẽ hội

tụ tới bộ trọng số W , biểu diễn đúng các mẫu học với điều kiện là các mẫu này biểu thị các điểm rời rạc của một hàm khả tách tuyến tính nào đó ($f: \mathbb{R}^n \rightarrow \mathbb{R}$ được gọi là khả tách tuyến tính nếu các tập $\{F^{-1}(x_k)\}$, với x_k thuộc miền trị của f , có thể tách được với nhau bởi các siêu phẳng trong không gian \mathbb{R}_n).

Năm 1969, Minsky và Papert đã chứng minh một cách chặt chẽ rằng lớp hàm thể hiện sự phụ thuộc giữa đầu vào và đầu ra có thể học bởi mạng Perceptron một lớp là lớp hàm khả tách tuyến tính. Khả tách tuyến tính là trường hợp tồn tại một mặt siêu phẳng để phân cách tất cả các đối tượng của một lớp này với một lớp khác, ví dụ một mặt phẳng sẽ phân chia không gian ba chiều thành hai vùng riêng biệt. Mở rộng ra, nếu có n đầu vào, $n > 2$ thì công thức $\sum_{j=1}^n w_{ij}x_j = \theta_i$ tạo nên một siêu phẳng có $n-1$ chiều trong không gian n chiều, nó chia không gian đó thành hai nửa. Trong nhiều bài toán thực tế đòi hỏi chia các vùng của các điểm trong một siêu không gian thành các lớp riêng biệt. Loại bài toán này gọi là bài toán phân lớp. Bài toán phân lớp có thể giải quyết bằng cách tìm các tham số thích hợp cho một siêu phẳng để nó có thể chia không gian n chiều thành các vùng riêng biệt.

Với tính chất của như đã nêu trên, mạng perceptron một lớp có thể mô tả các hàm logic như AND, OR và NOT. Tuy nhiên nó không thể hiện được hàm XOR. Như vậy chúng ta mô hình perceptron một lớp không thể giải quyết bài toán này. Vấn đề này sẽ được giải quyết bằng mô hình mạng nơron perceptron nhiều lớp (Multi Layer Perceptron - MLP)

Mạng perceptron nhiều lớp (Multilayer Perceptron – MLP) còn được gọi là mạng truyền thẳng nhiều lớp là sự mở rộng của mô hình mạng perceptron với sự bổ sung thêm những lớp ẩn và các nơron trong các lớp ẩn này có hàm truyền (hàm kích hoạt) dạng phi tuyến. Mạng MLP có một lớp ẩn là mạng nơron nhân tạo được sử dụng phổ biến nhất, nó có thể xấp xỉ các hàm liên tục được định nghĩa trên một miền có giới hạn cũng như những hàm là tập hợp hữu hạn của các điểm rời rạc.

Cấu trúc của một mạng MLP tổng quát có thể mô tả như sau:

- Đầu vào là các vec-tơ (x_1, x_2, \dots, x_p) trong không gian p chiều, đầu ra là các vec-tơ (y_1, y_2, \dots, y_q) trong không gian q chiều. Đối với các bài toán phân loại, p chính là kích thước của mẫu đầu vào, q chính là số lớp cần phân loại.
- Mỗi nơron thuộc lớp sau liên kết với tất cả các nơron thuộc lớp liền trước nó.
- Đầu ra của nơron lớp trước là đầu vào của nơron thuộc lớp liền sau nó.

Hoạt động của mạng MLP như sau: tại lớp đầu vào các nơron nhận tín hiệu vào xử lý (tính tổng trọng số, gửi tới hàm truyền) rồi cho ra kết quả (là kết quả của hàm truyền); kết quả này sẽ được truyền tới các nơron thuộc lớp ẩn thứ nhất; các nơron tại đây tiếp nhận

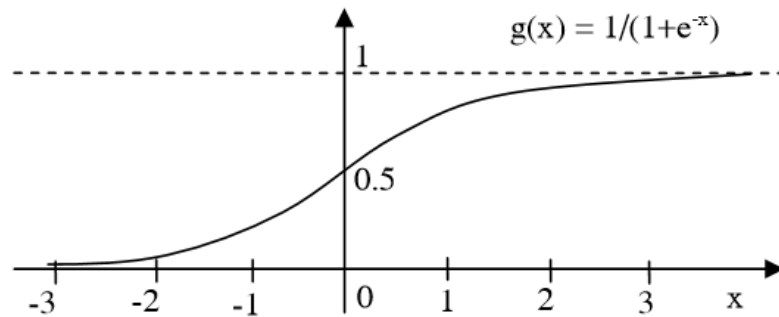
như là tín hiệu đầu vào, xử lý và gửi kết quả đến lớp ẩn thứ 2 ; ... ; quá trình tiếp tục cho đến khi các nơon thuộc lớp ra cho kết quả.

1.2.1 Một số vấn đề cần chú ý khi sử dụng mạng MLP

Mạng nơon perceptron nhiều lớp là loại mạng nơon được sử dụng trong nhiều ứng dụng thực tế. Tuy nhiên, để mạng có thể đưa ra kết quả tốt, chúng ta cần quan tâm đến một số vấn đề có ảnh hưởng khá quan trọng đến hiệu quả làm việc của nó bao gồm: vấn đề chuẩn hoá số liệu đầu vào, vấn đề học chưa đủ và học quá của mạng, vấn đề lựa chọn một cấu trúc mạng phù hợp với bài toán.

a. Vấn đề chuẩn hoá số liệu đầu vào

Mạng MLP thường sử dụng hàm truyền là hàm sigmoid có dạng như sau:



Hình 1.7 Hàm sigmoid $g(x) = 1/(1+e^{-x})$

Với dạng hàm này, giá trị ở đầu ra của mỗi nơon nằm trong phạm vi khoảng (0,1) và nó đạt các giá trị bão hoà (xấp xỉ 0 hay 1) khi $|x|$ lớn. Do đó, khi đầu vào của mạng có giá trị tuyệt đối lớn thì ta cần chuẩn hoá nó về khoảng có giá trị nhỏ, nếu không thì các nơon tại các lớp ẩn ngay ban đầu đã có thể đạt giá trị bão hoà và quá trình học của mạng không đạt kết quả mong muốn. Với dạng hàm như trên thì giá trị đầu vào của mạng thường được chuẩn hoá về khoảng thuộc đoạn $[-3, 3]$. Mặt khác, do tín hiệu đầu ra của nơon nằm trong khoảng giá trị (0,1) nên các giá trị đầu ra thực tế trong các mẫu học cũng cần chuẩn hoá về khoảng giá trị này để có thể dùng cho quá trình luyện mạng. Do vậy trong quá trình tính toán, để có các giá trị thực tế ở đầu ra của mạng chúng ta cần phải chuyển các giá trị trong khoảng (0,1) về miền các giá trị thực tế.

b. Vấn đề học chưa đủ và học quá thuộc của mạng

Vấn đề mấu chốt khi xây dựng một mạng nơon nhân tạo là làm thế nào mạng có khả năng tổng quát hoá cao để đưa ra kết quả tốt cả với những trường hợp đầu vào của mạng không nằm trong tập mẫu đã dùng để luyện mạng. Giống như các mô hình hồi quy phi tuyến khác, đối với mạng nơon nhân tạo ta cũng phải giải quyết hai vấn đề là ANN học chưa đủ (underfitting) và học quá (overfitting). Khi mạng có cấu trúc (số nút ẩn và liên kết) cũng như số lần học chưa đủ so với nhu cầu của bài toán thì sẽ dẫn tới tình trạng mạng

không đủ khả năng mô tả gần đúng mối quan hệ tương quan giữa đầu vào và đầu ra của quá trình cần dự báo và dẫn tới học chưa đủ. Trái lại, nếu mạng quá phức tạp (quá nhiều nút ẩn và quá nhiều tham số) và được học “quá khít” đối với các mẫu dùng để luyện mạng thì có thể dẫn tới tình trạng mạng học cả thành phần nhiễu lẫn trong các mẫu đó, đây là tình trạng “học quá thuộc” của mạng. Vấn đề nêu trên có thể làm cho nhiều loại mạng nơron, đặc biệt là mạng MLP có thể có những trường hợp cho kết quả dự đoán rất sai lệch với thực tế.

Một số giải pháp cho vấn đề học quá của mạng:

- Sử dụng tập số liệu có tính đại diện tốt để luyện mạng: Đây được xem là một cách khá tốt để tránh hiện tượng overfitting. Khi tập mẫu dùng để luyện mạng thể hiện được nhiều trạng thái có thể xảy ra của quá trình cần nghiên cứu thì sau khi học mạng sẽ có khả năng tổng quát hoá tương đối tốt từ tập dữ liệu đó và sẽ không chịu ảnh hưởng nhiều của hiện tượng overfitting. Ngoài ra một số biện pháp dưới đây cũng có thể góp phần quan trọng giúp khắc phục hiện tượng overfitting của mạng.

- Lựa chọn cấu trúc mô hình phù hợp: Việc lựa chọn mô hình của mạng (số lớp ẩn, số nơron trên mỗi lớp ẩn) có ảnh hưởng quan trọng đến hiện tượng học chưa đủ (underfitting) và học quá (overfitting) của mạng. Các mạng có độ phức tạp hơn tuy nó có thể học khá chính xác các mẫu được sử dụng nhưng chính điều này lại làm cho nó học quá nhiều cả thành phần nhiễu nên khả năng tổng quát hoá giảm và dẫn tới hiện tượng học quá (overfitting).

- Dừng học đúng lúc: giải pháp dừng học đúng lúc để tránh hiện tượng học quá của mạng như sau :

- + Tập mẫu được chia làm hai phần: một phần dùng để luyện mạng và phần còn lại để kiểm thử.
- + Sử dụng các giá trị khởi tạo nhỏ.
- + Sử dụng hằng số tốc độ học có giá trị thấp.
- + Tính toán sự thay đổi lỗi kiểm thử trong quá trình luyện mạng.
- + Dừng học khi thấy lỗi kiểm thử bắt đầu tăng.

c. Lựa chọn kích thước mạng

Các công trình dựa trên định lý của Kolmogorov dự kiến rằng toàn bộ các ánh xạ liên tục từ $[0,1]^p$ đến $[0,1]^n$ đều có thể được xấp xỉ bằng một mạng perceptron ba lớp có lớp vào gồm p nơron, lớp ra gồm n nơron và lớp ẩn gồm $(2p+1)$ nơron. Tuy nhiên không thể chỉ ra được chính xác số lượng nơron tối ưu trong mạng, tính chất của các nơron, tức là dạng phi tuyến cụ thể thực hiện phép xấp xỉ này. Một số công trình nghiên cứu về chủ đề này cho rằng số nơron tối ưu ở lớp ẩn thường nhỏ hơn $(2p+1)$. Ngoài ra cũng cần phải nói cơ sở dữ liệu học phải có kích thước phù hợp với kiến trúc mạng. Theo Vapnik và Chervonenkis, cơ

sở dữ liệu học phải có số mẫu thoả mãn: $N \approx 10.N_w$, ở đó N_w là số trọng số của mạng. Gọi số nơon thuộc lớp ẩn là L , số nơon ở lớp vào là p thì trọng số của các kết nối giữa lớp vào và lớp ẩn thứ nhất (kể cả ngưỡng) là: $D=(p+1).L$. Theo một số kết quả nghiên cứu, số mẫu của cơ sở dữ liệu học cần phải thoả mãn: $N \approx 4.D$. Khi số lượng mẫu của cơ sở dữ liệu học chưa đạt đến giới hạn cần thiết thì ta nên làm giảm số lượng các kết nối để tránh hiện tượng học thuộc lòng.

CHƯƠNG II

MỘT SỐ GIẢI THUẬT TỐI ƯU HÓA TRỌNG SỐ MẠNG NƠON NHÂN TẠO

2.1 Thuật toán học theo phương pháp lan truyền ngược sai số [3]

Thuật toán học theo phương pháp lan truyền ngược sai số là một trong số những kết quả nghiên cứu quan trọng nhất đối với sự phát triển của mạng nơon nhân tạo. Thuật toán này được áp dụng cho mạng truyền thẳng nhiều lớp trong đó các nơon có thể sử dụng các hàm truyền là các hàm liên tục có các dạng khác nhau.

2.1.1 Mô hình của thuật toán lan truyền ngược sai số

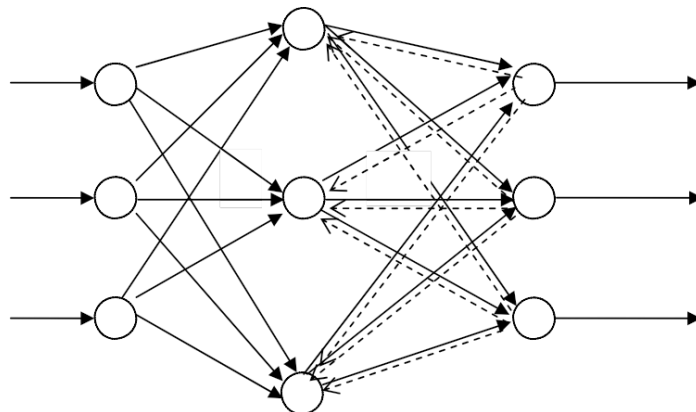
Thuật toán sử dụng một tập các mẫu gồm các cặp đầu vào - đầu ra để luyện mạng. Với mỗi cặp đầu vào - đầu ra $(x^{(k)}, d^{(k)})$ thuật toán lan truyền ngược sai số thực hiện hai giai đoạn sau:

- Giai đoạn thứ nhất, mẫu đầu vào $x^{(k)}$ được truyền từ lớp vào tới lớp ra, và ta có kết quả đầu ra tính toán được là $y^{(k)}$.

- Giai đoạn tiếp theo, tín hiệu lỗi được tính toán từ sự khác nhau giữa đầu ra quan sát được $d^{(k)}$ với đầu ra tính toán $y^{(k)}$ sẽ được lan truyền ngược lại từ lớp ra đến các lớp trước để điều chỉnh các trọng số của mạng.

Để làm ví dụ ta xét mạng truyền thẳng có một lớp ẩn dưới đây, đối với các mạng có kích thước lớn hơn thì thao tác cũng tương tự.

Mạng nơon được xét có m nơon ở lớp vào, l nơon trong lớp ẩn và n nơon ở lớp ra. Đường kẻ liền thể hiện luồng tín hiệu được truyền từ đầu vào tới đầu ra còn các đường kẻ nét đứt thể hiện luồng tín hiệu lỗi được truyền ngược trở lại từ đầu ra.



Hình 2.1 Lan truyền tín hiệu trong quá trình học theo pp lan truyền ngược sai số

Chúng ta xét một cặp đầu vào - đầu ra để luyện mạng (x, d) , để đơn giản chúng ta bỏ ký hiệu mũ k thể hiện số thứ tự của cặp mẫu này trong bộ mẫu dùng để luyện mạng. Khi đưa vào đầu vào x , nơon thứ q trong lớp ẩn sẽ nhận tín hiệu vào của mạng là :

$$net_q = \sum_{j=1}^m v_{qj} x_j$$

Nơon q ở lớp ẩn sẽ tính toán và tạo kết quả ở đầu ra của nó là :

$$z_q = g(net_q) = g\left(\sum_{j=1}^m v_{qj} x_j\right)$$

Do đó tín hiệu vào của nơon thứ i trên lớp ra sẽ là :

$$net_i = \sum_{q=1}^l w_{qi} z_q = \sum_{q=1}^l w_{qi} g\left(\sum_{j=1}^m v_{qj} x_j\right)$$

Và cuối cùng, đầu ra của nơon i trên lớp ra sẽ là :

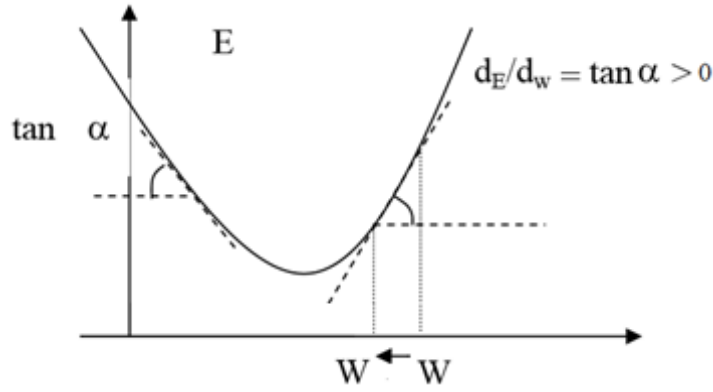
$$y_i = g(net_i) = g\left(\sum_{q=1}^l w_{qi} z_q\right) = g\left(\sum_{q=1}^l w_{qi} g\left(\sum_{j=1}^m v_{qj} x_j\right)\right)$$

Công thức trên cho biết quá trình lan truyền tín hiệu từ đầu vào qua lớp ẩn tới đầu ra. Tiếp theo chúng ta xét tín hiệu lỗi được lan truyền ngược lại từ lớp ra. Trước hết, đối với mỗi cặp giá trị vào – ra chúng ta xây dựng một hàm giá như sau :

$$E(w) = \frac{1}{2} \sum_{i=1}^n (d_i - y_i)^2 = \frac{1}{2} \sum_{i=1}^n [d_i - g(net_i)]^2 = \frac{1}{2} \sum_{i=1}^n \left[d_i - g\left(\sum_{q=1}^l w_{qi} z_q\right) \right]^2$$

Như vậy với một tập gồm p mẫu học, chúng ta lần lượt xây dựng được p hàm giá như vậy. Việc học của mạng hay nhiệm vụ của giải thuật thực chất là tìm kiếm tập trọng số W trong không gian R^M (M là số trọng số có trong mạng) để lần lượt tối thiểu hoá các hàm giá như vậy. Điều đáng chú ý là việc tối thiểu hoá được tiến hành liên tiếp nhau và theo chu kỳ đối với các hàm giá.

Để tối thiểu hoá các hàm giá như vậy, giải thuật Lan truyền ngược sai số sử dụng phương pháp giảm gradient để điều chỉnh các trọng số liên kết giữa các nơon. Bản chất của phương pháp này là khi sai số E được vẽ như hàm của tham số gây ra sai số sẽ phải có một cực tiểu tại bộ giá trị nào đó của tham số. Khi quan sát độ dốc của đường cong, chúng ta quyết định phải thay đổi tham số thế nào để có thể tiến gần đến cực tiểu cần tìm kiếm hơn. Trong hình vẽ dưới đây, giá trị của trọng số phải giảm nếu đạo hàm dE/dw là dương.



Hình 2.2 Sai số E được xét là hàm của trọng số W

Bằng biểu thức, chúng ta có thể biểu diễn phương pháp giảm gradient như sau :

$$\Delta w = w^{(\text{new})} - w^{(\text{old})} = -\eta \cdot \partial E / \partial w$$

Ở đây η là hằng số dương xác định tốc độ giảm giá trị của w , còn dấu âm chỉ chiều giảm gradient.

Áp dụng phương pháp giảm gradient đối với các trọng số liên kết giữa các nơron trong lớp ẩn tới các nơron của lớp ra ta có:

$$\Delta w_{iq} = -\eta \frac{\partial E}{\partial w_{iq}}$$

Do hàm sai số E là một hàm phức tạp và là hàm gián tiếp của trọng số w_{iq} , sử dụng nguyên tắc tính đạo hàm của hàm gián tiếp cho $\frac{\partial E}{\partial w_{iq}}$ ta có

$$\Delta w_{iq} = -\eta \left[\frac{\partial E}{\partial y_i} \right] \left[\frac{\partial y_i}{\partial \text{net}_i} \right] \left[\frac{\partial \text{net}_i}{\partial w_{iq}} \right] = -\eta [d_i - y_i] [g'(\text{net}_i)] [z_q] \triangleq \eta \delta_{oi} z_q$$

Trong đó δ_{oi} là tín hiệu sai số và chỉ số oi có nghĩa là nút thứ i trong lớp ra.

Tín hiệu sai số được tính như sau:

$$\delta_{oi} \triangleq - \left[\frac{\partial E}{\partial \text{net}_i} \right] = - \left[\frac{\partial E}{\partial y_i} \right] \left[\frac{\partial y_i}{\partial \text{net}_i} \right] = [d_i - y_i] [g'(\text{net}_i)]$$

Trong đó net_i là tín hiệu vào của nơron thứ i trên lớp ra và $g'(\text{net}_i) = \partial g(\text{net}_i) / \partial \text{net}_i$.

Để điều chỉnh trọng số của các liên kết giữa lớp vào tới lớp ẩn ta cũng sử dụng phương pháp giảm gradient và lấy đạo hàm theo các biến trung gian như đã áp dụng ở trên. Xét liên kết giữa nơron thứ j ở lớp vào và nơron thứ q trên lớp ra:

$$\begin{aligned} \Delta v_{qj} &= -\eta \frac{\partial E}{\partial v_{qj}} = -\eta \left[\frac{\partial E}{\partial \text{net}_q} \right] \left[\frac{\partial \text{net}_q}{\partial v_{qj}} \right] = -\eta \left[\frac{\partial E}{\partial z_q} \right] \left[\frac{\partial z_q}{\partial \text{net}_q} \right] \left[\frac{\partial \text{net}_q}{\partial v_{qj}} \right] \\ \Delta v_{qj} &= \eta \sum_{i=1}^n [(d_i - y_i) g'(\text{net}_i) w_{iq}] g'(\text{net}_q) x_j \end{aligned}$$

$$\Delta v_{qj} = \eta \sum_{i=1}^n [\delta_{oi} w_{iq}] g'(net_q) x_j = \mu \delta_{hq} x_j$$

Trong đó δ_{hq} là tín hiệu lỗi của nơron thứ q trong lớp ẩn và được định nghĩa như sau:

$$\delta_{hq} \triangleq - \left[\frac{\partial E}{\partial net_q} \right] = - \left[\frac{\partial E}{\partial z_q} \right] \left[\frac{\partial z_q}{\partial net_q} \right] = g'(net_q) \sum_{i=1}^n [\delta_{oi} w_{iq}]$$

Với net_q là tín hiệu vào của nơron thứ q, như vậy tín hiệu lỗi của nơron trên lớp ẩn khác với tín hiệu lỗi của nơron trên lớp ra. Vì sự khác nhau này, thủ tục điều chỉnh trọng số được gọi là luật học delta mở rộng. Nhìn lại công thức tín hiệu lỗi δ_{hq} của nơron thứ q trong lớp ẩn được xác định từ các tín hiệu lỗi δ_{oi} của các nơron trên lớp ra.

Tổng quát đối với lớp bất kỳ, luật lan truyền ngược có dạng:

$\Delta w_{ij} = \eta \delta_i x_j = \eta \delta_{\text{output}_i} x_{\text{input}_j}$ trong đó “output_i” là đầu ra của nơron i và “input_j” là đầu vào của nơron j, δ_i là tín hiệu học được định nghĩa trong công thức trên.

Thuật toán lan truyền ngược sai số được xây dựng như sau:

Xét một mạng nơron truyền thẳng có Q lớp, $q = 1, 2, \dots, Q$, và gọi net_i và y_i là tín hiệu vào và ra của nơron thứ i trong lớp q. Mạng này có m đầu vào và n đầu ra. Đặt w_{ij} là trọng số của liên kết từ nơron thứ j trong lớp q-1 tới nơron thứ i trong lớp q.

Đầu vào: Một tập các cặp mẫu học $\{(x^{(k)}, d^{(k)}) \mid k = 1, 2, \dots, p\}$

– Bước 1 (khởi tạo): Chọn một hằng số $\eta > 0$ và E_{max} (dung sai cho phép). Khởi tạo ngẫu nhiên các trọng số w_{ij} trong khoảng giá trị nhỏ. Đặt $E = 0$ và $k = 1$.

– Bước 2 (thực hiện một quá trình lặp cho việc huấn luyện mạng)

+ Sử dụng mẫu học thứ k ;

+ Tại lớp vào ($q=1$), với mọi i ta có: ${}^q y_i = {}^1 y_i = x^{(k)}_i$

– Bước 3 (lan truyền tín hiệu từ lớp vào tới lớp ra)

$${}^q y_i = g({}^q net_i) = g \left(\sum_j {}^q w_{ij} {}^{q-1} y_j \right)$$

– Bước 4 (xác định tín hiệu lỗi ${}^Q \delta_i$ tại lớp ra)

$$E = \frac{1}{2} \sum_{i=1}^n (d_i^{(k)} - {}^Q y_i)^2 + E$$

$${}^Q \delta_i = (d_i^{(k)} - {}^Q y_i) g'({}^Q net_i)$$

– Bước 5 (lan truyền ngược sai số)

Lan truyền ngược sai số để điều chỉnh các trọng số và tính toán tín hiệu lỗi ${}^{q-1} \delta_i$ cho các lớp trước :

$$\Delta^q w_{ij} = \eta \cdot {}^q \delta_i \cdot {}^{q-1} y_i$$

$${}^q w_{ij}^{new} = {}^q w_{ij}^{old} + \Delta^q w_{ij}$$

$${}^{q-1} \delta_i = g'({}^{q-1} net_i) \sum_j {}^q w_{ij} {}^q \delta_i \quad \text{với } q=Q, Q-1, \dots, 2$$

– Bước 6 (kiểm tra điều kiện lặp)

If ($k < p$) then

$k = k + 1$;

goto 2 ;

End if

– Bước 7 (kiểm tra lỗi tổng cộng hiện thời đã chấp nhận được chưa)

If ($E > E_{max}$) then

$E = 0$;

$k = 1$;

Else

{kết thúc quá trình học và đưa ra bộ trọng số cuối cùng}

End if

Mỗi lần toàn bộ tập mẫu học được lan truyền qua mạng được gọi là một epoch. Số epoch phụ thuộc vào từng trường hợp cụ thể và sự khởi tạo ban đầu. Có trường hợp thuật toán phải sau hàng chục nghìn epoch mới hội tụ tới lời giải. Nếu tham số khởi tạo không phù hợp có thể làm cho quá trình học không đạt kết quả mong muốn.

Đối với mỗi epoch ta tính sai số trung bình của mạng theo công thức sau:

$$RMS = \sqrt{\frac{\sum_{k=1}^p \sum_{i=1}^n (d_i - y_i)^2}{p \cdot n}}$$

Trong đó p là số mẫu được dùng để luyện mạng, n là số biến của véc-tơ đầu ra. Sai số RMS thường được dùng để đánh giá kết quả học của mạng nơron.

2.1.2 Một số yếu tố ảnh hưởng đến quá trình học theo phương pháp lan truyền ngược

Khởi tạo các trọng số

Các giá trị được khởi tạo ban đầu cho các trọng số trong mạng lan truyền ngược sai số ảnh hưởng rất lớn đến kết quả học cuối cùng của mạng. Các giá trị này thường được khởi tạo ngẫu nhiên trong phạm vi giá trị tương đối nhỏ. Thông thường hàm truyền sử dụng cho mạng MLP là hàm sigmoid, do vậy nếu ta chọn các giá trị trọng số khởi tạo lớn thì các hàm này có thể bão hoà ngay từ đầu và dẫn tới hệ thống có thể bị tắc ngay tại một cực tiểu địa phương hoặc tại một vùng bằng phẳng nào đó gần điểm xuất phát.

Hằng số học

Hằng số học η cũng là một yếu tố quan trọng ảnh hưởng đến hiệu quả và độ hội tụ của giải thuật lan truyền ngược sai số. Không có hằng số η phù hợp cho tất cả các bài toán khác nhau. Hằng số học này thường được chọn bằng thực nghiệm cho mỗi bài toán ứng dụng cụ thể bằng phương pháp thử sai.

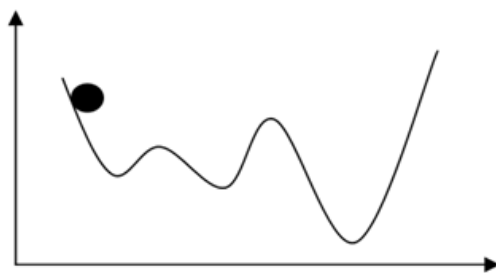
Trong nhiều ứng dụng thực tế cho thấy một hằng số học có thể phù hợp ở thời điểm bắt đầu của quá trình học nhưng lại không phù hợp với giai đoạn sau của quá trình học. Do đó, có một phương pháp hiệu quả hơn đó là sử dụng hằng số học thích nghi. Một cách xử lý đơn giản cho vấn đề này đó là kiểm tra xem các trọng số mới có làm giảm hàm giá hay không, nếu không thì có thể các trọng số đã vượt quá xa vùng cực tiểu và như vậy hằng số η cần phải giảm. Trái lại, nếu sau vài vòng lặp, hàm giá liên tục giảm thì ta có thể thử tăng hằng số η để đẩy nhanh hơn tốc độ hội tụ đến giá trị cực tiểu.

Hằng số quán tính

Tốc độ học của giải thuật Lan truyền ngược sai số có thể rất chậm nếu hằng số học nhỏ, nhưng nếu hằng số học lớn thì nó lại có thể gây ra sự dao động lớn trong quá trình tìm giá trị cực tiểu theo phương pháp giảm gradient. Để giải quyết vấn đề này người ta thường thêm thành phần quán tính vào các phương trình hiệu chỉnh trọng số như sau:

$$\Delta w(t) = -\eta \nabla E(t) + \alpha \Delta w(t-1) \quad \text{với } \alpha \text{ là hằng số quán tính, } \alpha \in [0, 1]$$

Nhờ thành phần này, quá trình học có thể vượt qua điểm cực tiểu địa phương để tìm đến điểm cực tiểu toàn cục, đồng thời thành phần quán tính cũng ngăn cản sự thay đổi đột ngột của các trọng số theo hướng khác với hướng mà lời giải đang di chuyển đến.



Hình 2.3 Minh họa về ý nghĩa của quán tính trong thực tế

Hàm giá

Trong phần nghiên cứu trên hàm giá được chọn là hàm bình phương sai số.

$$E(w) = \frac{1}{2} \sum_{i=1}^n (d_i - y_i)^2$$

Tuy nhiên, nó có thể được thay thế bằng một hàm $F(y,d)$ bất kỳ có đạo hàm và đạt cực tiểu khi hai đối số d_i và y_i bằng nhau. Thông thường hàm giá được chọn có dạng:

$$E(w) = \frac{1}{p} \sum_{i=1}^n (d_i - y_i)^p$$

Với $1 \leq p \leq \infty$, như vậy khi chọn $p = 2$ ta có hàm giá là hàm bình phương sai số như đã xét ở trên.

2.2 Giải thuật di truyền [2]

Từ trước tới nay, trong các nghiên cứu và ứng dụng tin học đã xuất hiện nhiều bài toán chưa tìm ra được phương pháp giải nhanh và hợp lý. Phần lớn đó là các bài toán tối ưu nảy sinh trong các ứng dụng. Để giải các bài toán này người ta thường phải tìm đến một giải thuật hiệu quả mà kết quả thu được chỉ là xấp xỉ tối ưu. Trong nhiều trường hợp chúng ta có thể sử dụng giải thuật xác suất, tuy không bảo đảm kết quả tối ưu nhưng cũng có thể chọn các giá trị sao cho sai số đạt được sẽ nhỏ như mong muốn.

Theo lời giải xác suất, việc giải bài toán quy về quá trình tìm kiếm trên không gian tập hợp các lời giải có thể. Tìm được lời giải tốt nhất và quá trình được hiểu là tối ưu. Với miền tìm kiếm nhỏ, một số thuật toán cổ điển được sử dụng. Tuy nhiên đối với các miền lớn, phải sử dụng các kỹ thuật trí tuệ nhân tạo đặc biệt, giải thuật Di truyền là một trong những công cụ đó. Ý tưởng của giải thuật Di truyền là mô phỏng những gì mà tự nhiên đã thực hiện. Giải thuật Di truyền hình thành dựa trên quan niệm cho rằng: quá trình tiến hóa tự nhiên là quá trình hoàn hảo nhất, hợp lý nhất và tự nó đã mang tính tối ưu. Quá trình tiến hoá thể hiện tính tối ưu ở chỗ thế hệ sau bao giờ cũng tốt hơn thế hệ trước.

Giải thuật Di truyền áp dụng quá trình tiến hóa tự nhiên để giải các bài toán tối ưu trong thực tế (từ tập các lời giải có thể ban đầu thông qua nhiều bước tiến hóa hình thành các tập hợp mới với lời giải tốt hơn và cuối cùng sẽ tìm được lời giải gần tối ưu).

Giải thuật Di truyền là một kỹ thuật của khoa học máy tính nhằm tìm kiếm giải pháp thích hợp cho các bài toán tối ưu tổ hợp (combinatorial optimization). Giải thuật Di truyền là một phân ngành của giải thuật tiến hóa vận dụng các nguyên lý của tiến hóa như di truyền, đột biến, chọn lọc tự nhiên, và trao đổi chéo.

Giải thuật Di truyền thường được ứng dụng nhằm sử dụng ngôn ngữ máy tính để mô phỏng quá trình tiến hoá của một tập hợp những đại diện trừu tượng (gọi là những nhiễm sắc thể) của các giải pháp có thể (gọi là những cá thể) cho bài toán tối ưu hóa vấn đề. Tập hợp này sẽ tiến triển theo hướng chọn lọc những giải pháp tốt hơn.

Thông thường, những giải pháp được thể hiện dưới dạng nhị phân với những chuỗi 0 và 1, nhưng lại mang nhiều thông tin mã hóa khác nhau. Quá trình tiến hóa xảy ra từ một tập hợp những cá thể hoàn toàn ngẫu nhiên ở tất cả các thế hệ. Trong từng thế hệ, tính thích nghi của tập hợp này được ước lượng, nhiều cá thể được chọn lọc định hướng từ tập hợp hiện thời (dựa vào thể trạng), được sửa đổi (bằng đột biến hoặc tổ hợp lại) để hình thành một tập hợp mới. Tập hợp này sẽ tiếp tục được chọn lọc lặp đi lặp lại trong các thế hệ kế tiếp của giải thuật.

2.2.1 Các khái niệm cơ bản

Giải thuật Di truyền dựa vào quá trình tiến hoá trong tự nhiên nên các khái niệm và thuật ngữ của nó đều có liên quan đến các thuật ngữ của di truyền học.

a. Cá thể, nhiễm sắc thể

Một cá thể trong giải thuật Di truyền, biểu diễn một giải pháp của bài toán. Tuy nhiên không giống với trong tự nhiên, một cá thể có nhiều nhiễm sắc thể (NST), có 1 thì gọi là thể đơn bội, còn nếu có nhiều thì là thể đa bội, ở đây để giới hạn trong giải thuật Di truyền ta quan niệm một cá thể có một nhiễm sắc thể. Do đó khái niệm cá thể và nhiễm sắc thể trong giải thuật Di truyền coi như là tương đương.

Một NST được tạo thành từ nhiều gen, mỗi gen có thể có các giá trị khác nhau để quy định một tính trạng nào đó. Trong giải thuật Di truyền, một gen được coi như một phần tử trong chuỗi NST.

b. Quần thể

Quần thể là một tập hợp các cá thể có cùng một số đặc điểm nào đấy. Trong giải thuật Di truyền ta quan niệm quần thể là một tập các lời giải của một bài toán.

c. Các toán tử di truyền

- **Chọn lọc:** Trong tự nhiên, quá trình chọn lọc và đấu tranh sinh tồn đã làm thay đổi các cá thể trong quần thể. Những cá thể tốt, thích nghi được với điều kiện sống thì có khả năng đấu tranh lớn hơn, do đó có thể tồn tại và sinh sản. Các cá thể không thích nghi được với điều kiện sống thì dần mất đi. Dựa vào nguyên lý của quá trình chọn lọc và đấu tranh sinh tồn trong tự nhiên, chọn lựa các cá thể trong giải thuật Di truyền chính là cách chọn các cá thể có độ thích nghi tốt để đưa vào thế hệ tiếp theo hoặc để cho lai ghép, với mục đích là sinh ra các cá thể mới tốt hơn. Có nhiều cách để lựa chọn nhưng cuối cùng đều nhằm đáp ứng mục tiêu là các cá thể tốt sẽ có khả năng được chọn cao hơn.
- **Lai ghép:** Lai ghép trong tự nhiên là sự kết hợp các tính trạng của bố mẹ để sinh ra thế hệ con. Trong giải thuật di truyền, lai ghép được coi là một sự tổ hợp lại các tính chất (thành phần) trong hai lời giải cha mẹ nào đó để sinh ra một lời giải mới mà có đặc tính mong muốn là tốt hơn thế hệ cha mẹ. Đây là một quá trình xảy ra chủ yếu trong giải thuật Di truyền.
- **Đột biến:** Đột biến là một sự biến đổi tại một (hay một số) gen của nhiễm sắc thể ban đầu để tạo ra một nhiễm sắc thể mới. Đột biến có xác suất xảy ra thấp hơn lai ghép. Đột biến có thể tạo ra một cá thể mới tốt hơn hoặc xấu hơn cá thể ban đầu. Tuy nhiên trong giải thuật Di truyền thì ta luôn muốn tạo ra những phép đột biến cho phép cải thiện lời giải qua từng thế hệ.

2.2.2 Mô hình giải thuật di truyền

Với các khái niệm được nêu ở trên, giải thuật Di truyền được mô tả như sau:

1. [**Bắt đầu**] Nhận các tham số cho thuật toán.
2. [**Khởi tạo**] Sinh ngẫu nhiên một quần thể gồm n cá thể (là n lời giải cho bài toán).
3. [**Quần thể mới**] Tạo quần thể mới bằng cách lặp lại các bước sau cho đến khi quần thể mới hoàn thành.
 - a. [**Thích nghi**] Ước lượng độ thích nghi $eval(x)$ của mỗi cá thể.
 - b. [**Kiểm tra**] Kiểm tra điều kiện kết thúc giải thuật.
 - c. [**Chọn lọc**] Chọn hai cá thể bố mẹ từ quần thể cũ theo độ thích nghi của chúng (cá thể có độ thích nghi càng cao thì càng có nhiều khả năng được chọn).
 - d. [**Lai ghép**] Với một xác suất lai ghép được chọn, lai ghép hai cá thể bố mẹ để tạo ra một cá thể mới.
 - e. [**Đột biến**] Với một xác suất đột biến được chọn, biến đổi cá thể mới.
4. [**Chọn kết quả**] Nếu điều kiện dừng được thỏa mãn thì thuật toán kết thúc và trả về lời giải tốt nhất trong quần thể hiện tại.

2.2.3 Áp dụng giải thuật Di truyền cho bài toán mạng nơron nhân tạo

Để có thể sử dụng được giải thuật Di truyền vào việc học của mạng nơron cần phải thực hiện một số bước như sau:

– Xây dựng hàm giá: Hàm giá này sẽ được sử dụng để tạo nên độ phù hợp của các cá thể và của cả quần thể trong GA. Trong nghiên cứu này sử dụng hàm sai số RMS tương tự như trong giải thuật Lan truyền ngược sai số.

– Mã hoá nhiễm sắc thể: Mỗi cá thể trong GA sẽ thay mặt cho một bộ trọng số của mạng nơron. Ở đây ta không cần phải phân biệt trọng số nào ở lớp nào mà ta chỉ cần trải tất cả các trọng số lên sơ đồ gen của nhiễm sắc thể.

– Thực hiện giải thuật Di truyền:

- + Chọn lọc: Gán 1 nhiễm sắc thể con bằng một nhiễm sắc có hàm giá nhỏ hơn của 2 nhiễm sắc thể bố mẹ ngẫu nhiên (hoặc là bố, hoặc là mẹ).
- + Lai ghép: Toán tử lai ghép này sẽ đưa một giá trị vào mỗi vị trí của nhiễm sắc thể con bằng cách lấy ngẫu nhiên một giá trị tại cùng vị trí của nhiễm sắc thể cha hoặc mẹ.
- + Đột biến: Một gen (trọng số) được lựa chọn ngẫu nhiên với một xác suất p_{mutation} để tiến hành đột biến sử dụng phương pháp BIASED: với mỗi gen

được chọn đột biến nó sẽ được cộng thêm một giá trị ngẫu nhiên quanh giá trị gốc ban đầu.

2.3 Kết hợp các giải thuật

Như chúng ta đã biết sử dụng giải thuật Lan truyền ngược sai số để tối ưu hoá trọng số của mạng nơron nhân tạo đang được sử dụng rộng rãi hiện nay. Tuy nhiên, giải thuật này hoạt động theo cơ chế giảm gradient nên nó khó có thể tìm ra được cực trị toàn cục. Trong nghiên cứu này sử dụng giải thuật Di truyền để tối ưu hoá trọng số của mạng giúp quá trình học của mạng được tốt hơn.

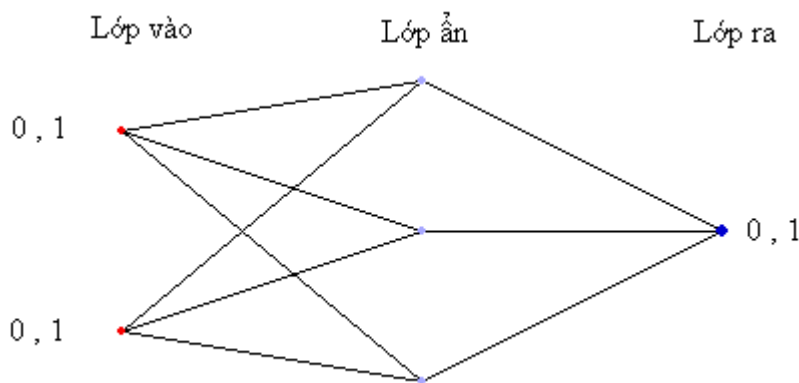
Thử nghiệm hai phương pháp với bài toán nổi tiếng XOR. Có 4 mẫu học như sau:

Bảng 2.1 Giá trị đầu vào và ra của bài toán XOR

X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Tham số chung cho cả hai phương pháp:

- Mạng nơron sử dụng là mạng có một lớp ẩn
- Số nơron trong lớp ẩn: 3
- Ngưỡng dừng lặp: 1000 vòng lặp



Hình 2.4 Mô hình mạng nơron cho bài toán XOR

Tham số của giải thuật lan truyền ngược sai số:

- Hằng số học: 0.3

Tham số của giải thuật di truyền:

- Số lượng quần thể: 100

- Xác suất lai: 0.3
- Xác suất đột biến: 0.1

a. Kết quả trong 1.000.000 lần chạy với sai số dừng 0.05:

- Giải thuật Di truyền chạy thành công được 837.532 lần, trung bình 19 vòng lặp một lần chạy.
- Giải thuật Lan truyền ngược sai số chạy thành công được 763.908 lần trung bình 214 vòng lặp.

Ta có thể thấy rằng giải thuật Di truyền có khả năng đạt được yêu cầu về hội tụ (sai số ≤ 0.05) tức tìm vùng chứa cực trị toàn cục dễ dàng hơn so với giải thuật Lan truyền ngược sai số. Hay nói cách khác giải thuật Lan truyền ngược sai số dễ rơi vào vùng chứa cực tiểu cục bộ hơn giải thuật di truyền.

b. Kết quả trong 1.000.000 lần chạy với sai số dừng 0.02:

- Giải thuật Di truyền chạy thành công được 402.466 lần trung bình 51 vòng lặp một lần chạy.
- Giải thuật Lan truyền ngược sai số chạy thành công được 567.466 lần trung bình 202 vòng lặp.

Mặc dù giải thuật Di truyền có khả năng đạt tới cực trị toàn cục cho quá trình tìm kiếm nhưng do có kết hợp những yếu tố ngẫu nhiên nên tốc độ tìm kiếm nói chung là rất chậm. Mặt khác nó không thể hoàn toàn đạt được tới cực trị toàn cục mà chỉ cho những kết quả xung quanh đó. Đối lập với GA, giải thuật Lan truyền ngược sai số lại cho phép đạt được những cực trị nếu như điểm xuất phát của quá trình tìm kiếm nằm trong vùng cực trị toàn cục.

Từ 2 kết quả trên ta có nhận xét: giải thuật Di truyền có thể đạt đến vùng chứa cực tiểu toàn cục (sai số 0.05) dễ dàng hơn so với Giải thuật Lan truyền ngược sai số. Tuy nhiên, để đạt đến chính xác vị trí cực tiểu toàn cục (sai số nhỏ dần) thì giải thuật Di truyền lại rất kém. Trong khi đó, hầu hết các trường hợp Giải thuật Lan truyền ngược sai số khi đã đưa mạng đến được vùng chứa cực tiểu toàn cục (sai số 0.05) thì Giải thuật Lan truyền ngược sai số sẽ đưa mạng đến chính xác cực tiểu toàn cục. Do đó, việc kết hợp giải thuật Di truyền và Giải thuật Lan truyền ngược sai số có nhiều cơ hội đưa mạng đến được chính xác cực tiểu toàn cục.

Ta thấy rằng có thể kết hợp giải thuật Di truyền và Giải thuật Lan truyền ngược sai số nhằm nâng cao hiệu quả của BP. Giải thuật Di truyền sẽ khoanh vùng chứa cực tiểu toàn cục của hàm lỗi, sau đó Giải thuật Lan truyền ngược sai số xuất phát từ bộ trọng số này để tiến đến cực tiểu toàn cục.

Có nhiều cách để kết hợp giải thuật Di truyền vào mạng nơron nhưng cách đơn giản và khá hiệu quả là ta thực hiện lai ghép hai giải thuật nối tiếp nhau.

Tập trọng số được mã hoá thành các nhiễm sắc thể và được tiến hoá nhờ GA. Kết thúc quá trình tiến hoá, bộ trọng số tốt nhất tương ứng với cá thể ưu việt nhất trong quần thể được lựa chọn làm những trọng số khởi tạo cho giải thuật BP. Nó chính là bộ tham số cho phép xác định điểm gần cực trị nhất của hàm giá.

Với việc lai ghép này, giải thuật Lan truyền ngược sai số lược bỏ đi một số bước sau:

- Không khởi tạo các giá trị trọng số ban đầu vì tập trọng số đã được lấy từ kết quả của giải thuật di truyền.
- Thành phần quán tính trong các phương trình hiệu chỉnh trọng số là không cần thiết vì tập trọng số xuất phát đã khá gần lời giải; tác dụng chống dao động và thay đổi đột ngột các trọng số theo hướng khác với hướng của lời giải trở nên không cần thiết.

Việc thử nghiệm giải thuật kết hợp này được tiến hành với bài toán XOR ngưỡng sai số mong muốn là 0.001. Các tham số cũng như các phần trước. Giải thuật gồm hai bước chính:

- (1) GA sẽ đưa mạng đạt đến sai số 0.05;
- (2) BP sẽ nhận bộ trọng số tốt nhất của giải thuật Di truyền đóng vai trò là trọng số khởi tạo (có sai số 0.05) để đưa mạng đến sai số mong muốn 0.001.

Ta có thể tóm tắt khả năng hội tụ của mạng với hai phương pháp học: giải thuật Di truyền và giải thuật Lan truyền ngược sai số khi ngưỡng sai số dừng lặp khác nhau trong bảng 2.2.

Bảng 2.2 So sánh khả năng hội tụ của mạng khi sử dụng hai phương pháp học GA và BP với sai số dừng lặp khác nhau

Sai số dừng lặp	Số lần hội tụ trong 1.000.000 lần thử nghiệm		
	GA	BP	Kết hợp
0.05	83,75%	75,64%	
0.02	40,25%	56,75%	
0.001	0%	18,44%	60,45%

So sánh với việc sử dụng GA và BP riêng rẽ thì giải thuật kết hợp này cho kết quả tốt hơn rất nhiều.

CHƯƠNG III

XÂY DỰNG PHẦN MỀM DỰ BÁO BẰNG MẠNG NƠRON NHÂN TẠO

3.1 Ngôn ngữ và giao diện phần mềm

a. Ngôn ngữ phát triển

Chương trình tính toán được xây dựng từ đầu không sử dụng nền tảng hay mã nguồn sẵn có nào, trên cơ sở kết hợp giải thuật lan truyền sai số ngược và giải thuật Di truyền cho mạng MLP. Nó bao gồm hai giai đoạn luyện mạng. Giai đoạn đầu tiên sử dụng thuật toán di truyền nhằm đẩy nhanh toàn bộ quá trình luyện mạng. Thuật toán di truyền thực hiện tìm kiếm toàn cục và tìm ra vùng chứa điểm tối ưu cho giai đoạn thứ hai. Trong đó, mỗi nhiễm sắc thể được sử dụng để mã hóa các trọng số của mạng nơron. Hàm giá (hàm mục tiêu) cho các thuật toán di truyền được xác định là sai số quân phương (RMS) của mạng nơron tương ứng. Trong giai đoạn thứ 2 sẽ sử dụng kỹ thuật lan truyền ngược với các bước học được thay đổi.

Phiên bản F nhằm kết hợp với các phần mềm cơ học được xây dựng bằng ngôn ngữ Fortran bằng bộ công cụ lập trình Visual Fortran. Phiên bản được tối ưu trên nền hệ điều hành 64bit nhằm tận dụng hết sức mạnh của hệ điều hành khi làm việc với dữ liệu lớn. Với phiên bản Fortran này, chúng tôi cũng đã kết hợp xử lý song song (được giới thiệu chi tiết ở phần 3.5.1) nhằm nâng cao tốc độ tính toán của chương trình.

Phiên bản C phục vụ các ứng dụng khác mang tính ứng dụng nhận dạng như phân tích xử lý ảnh, âm thanh, dịch ngôn ngữ,... được xây dựng trên ngôn ngữ C# nhằm tận dụng các thư viện tiên tiến như xử lý bản đồ GIS, phân tích ảnh OpenCV, kết nối các thiết bị ngoại vi máy tính,... để tạo thành một công cụ hoàn chỉnh đáp ứng các bài toán thực tế khác.

b. Danh sách các file của phần mềm

Danh sách các file số liệu đầu vào:

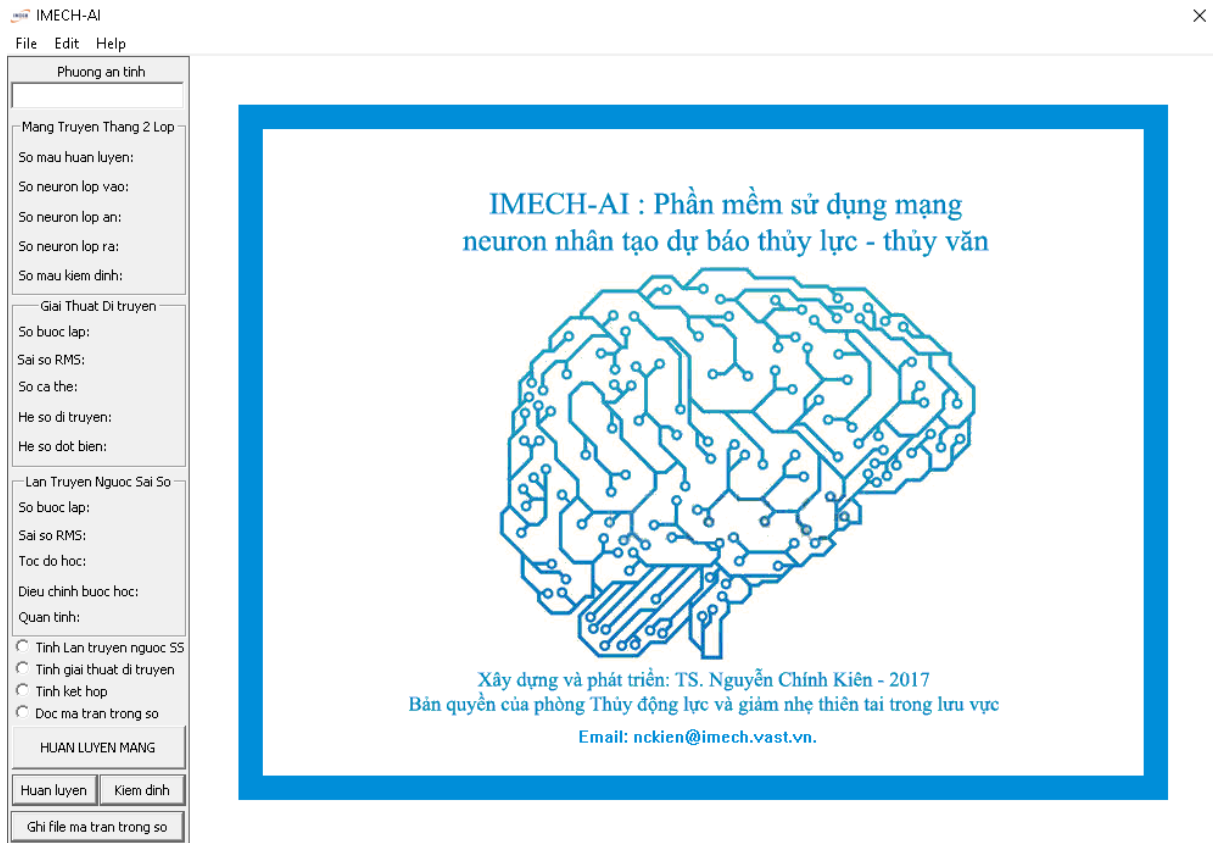
- File tham số ban đầu:
 - + Cấu trúc mạng nơron: số lớp vào, số lớp ẩn, số lớp ra;
 - + Loại hàm truyền của tương ứng với từng lớp;
 - + Tham số giải thuật di truyền: giá trị ngưỡng sai số, số lượng quần thể, tỉ lệ di truyền, lai ghép, đột biến, số bước dừng học;
 - + Tham số giải thuật lan truyền ngược: số bước dừng học, giá trị ngưỡng sai số, giá trị bước học ban đầu, quán tính học,...
- File số liệu huấn luyện: các tín hiệu đầu vào, đầu ra cho trước để huấn luyện.

- File số liệu kiểm tra, dự báo: các tín hiệu đầu vào cần dự báo.

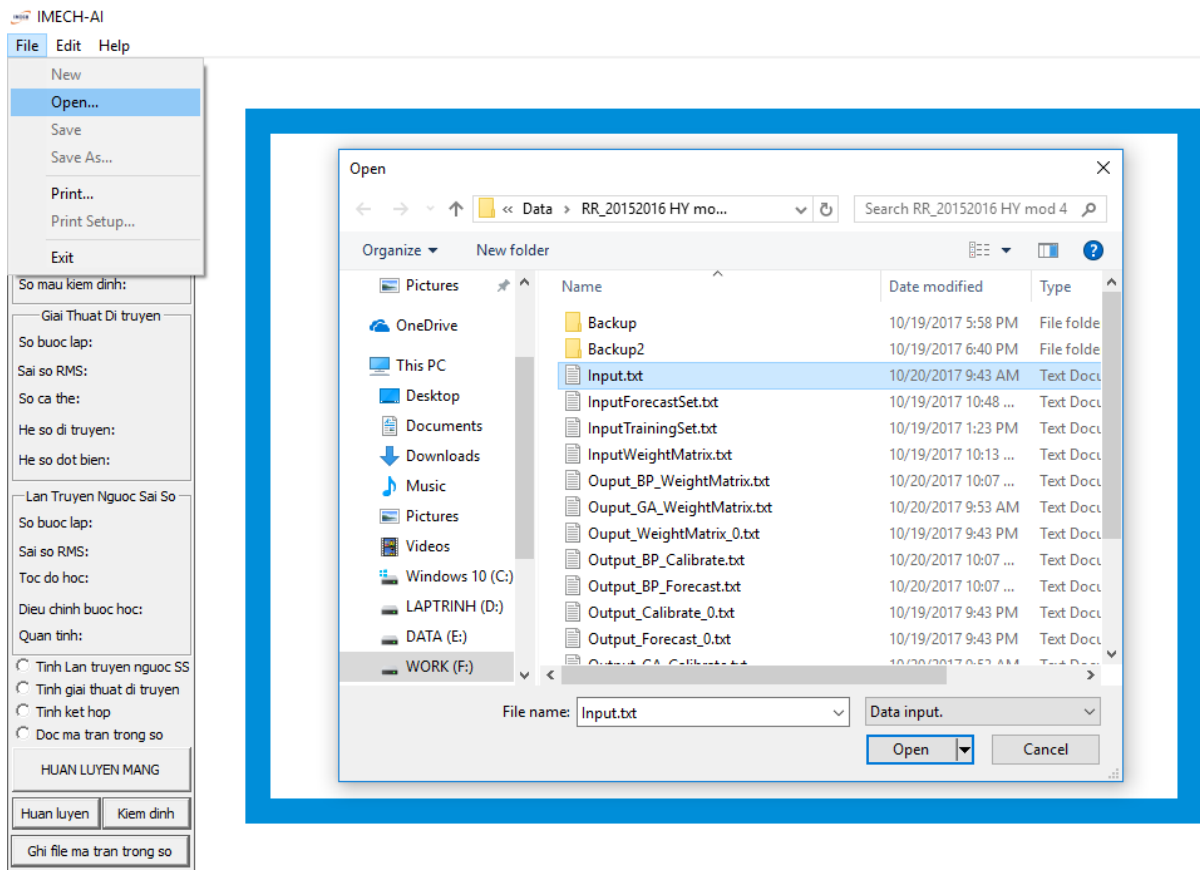
Danh sách các file được kết xuất:

- File kết quả ma trận trọng số liên kết của từng bước.
- File kết quả số liệu đầu ra của phương án tính.

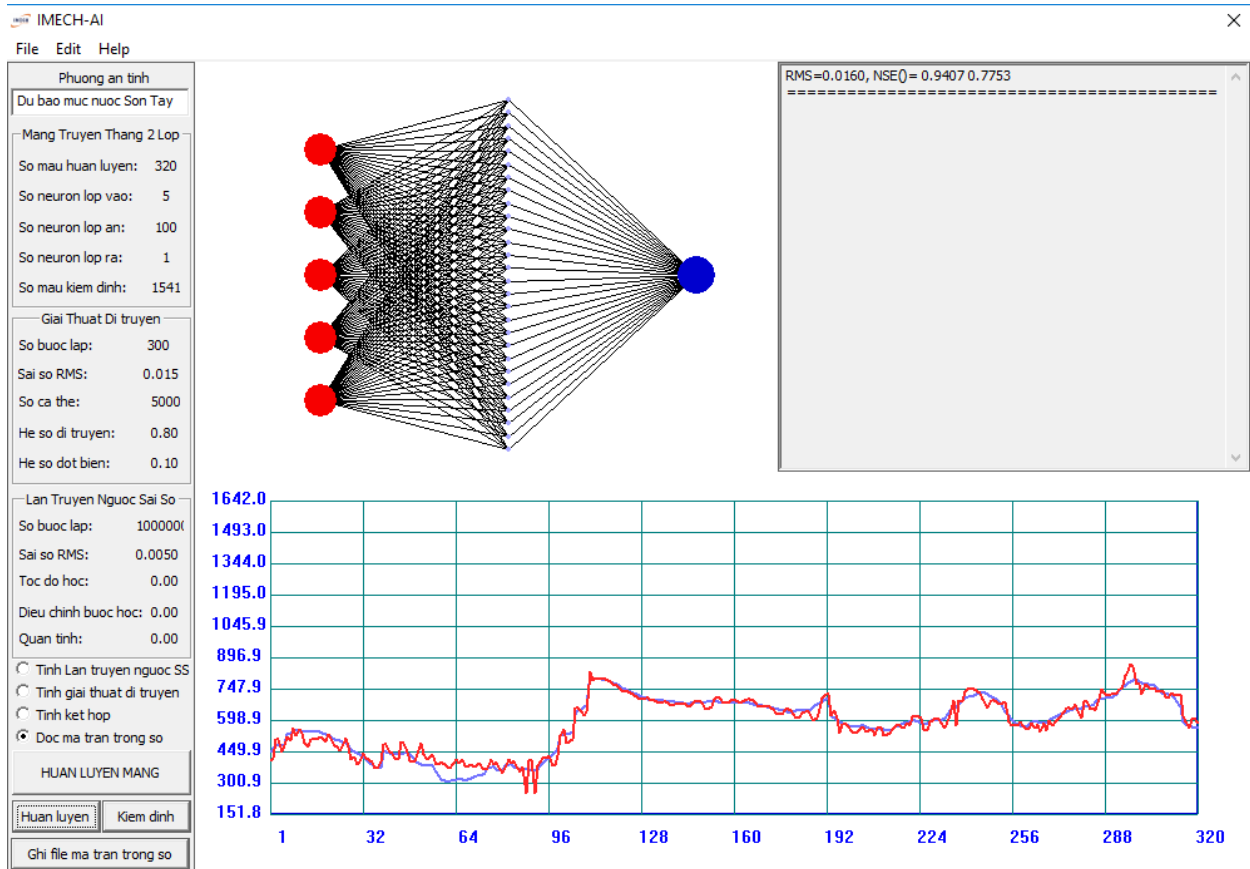
c. Giao diện của phần mềm



Hình 3.1 Giao diện của phần mềm dự báo bằng mạng neuron.



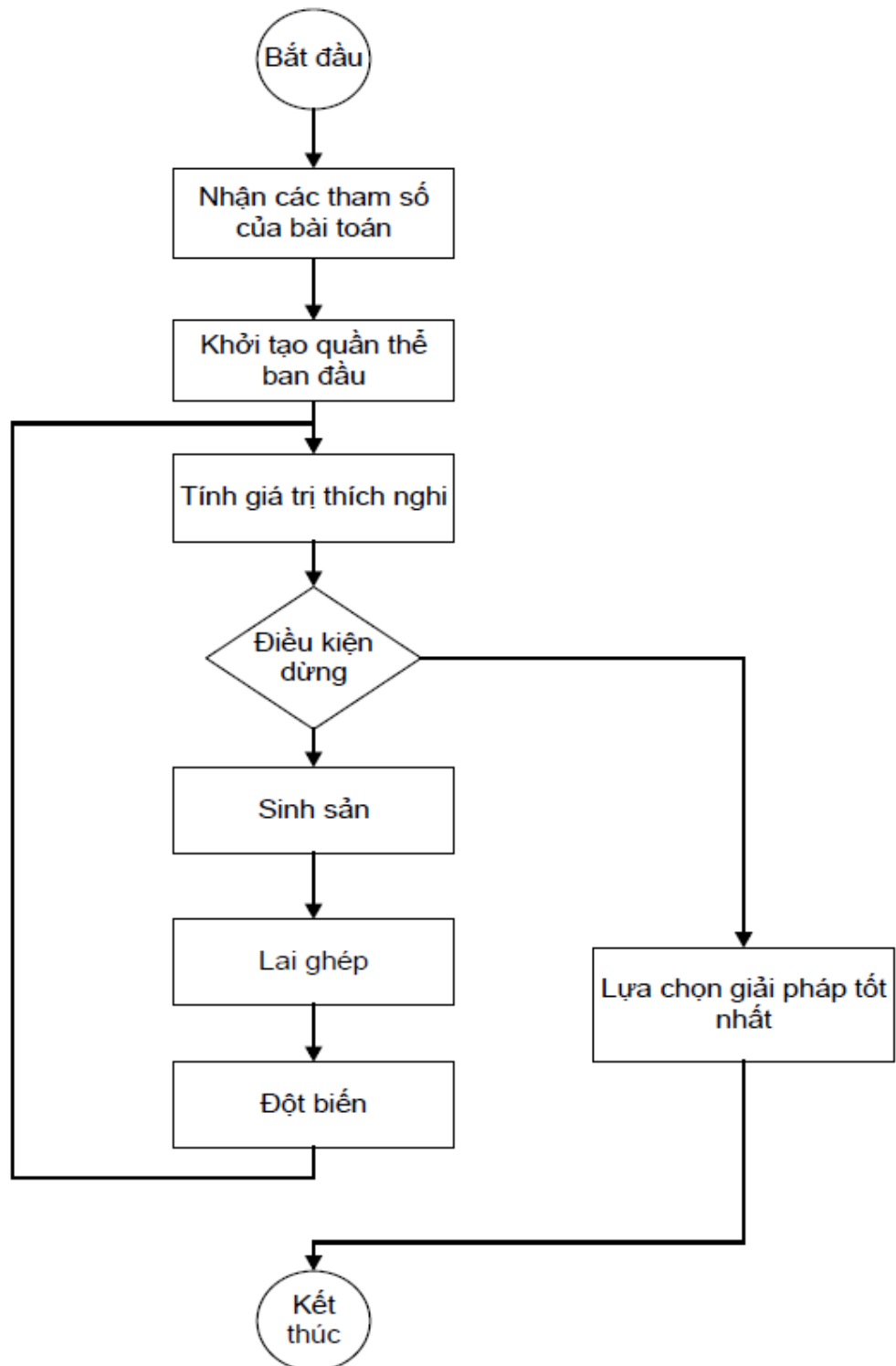
Hình 3.2 Lựa chọn phương án tính



Hình 3.3 Cấu trúc mạng, thông số và các giá trị ban đầu của phương án đã chọn.

3.2 Modul Giải thuật di truyền

Nếu phương án tính có lựa chọn giải thuật Di truyền hoặc Kết hợp, modul giải thuật Di truyền được sử dụng với sơ đồ khối như dưới đây:



Hình 3.4 Sơ đồ thuật toán của giải thuật di truyền

Chi tiết được diễn giải như sau:

Bước 1: Tạo n (n là số lượng cá thể của quần thể di truyền) vector mảng 1 chiều bao gồm m số là tổng của số phần tử ma trận trọng số lớp ẩn và lớp ra. Các giá trị của vector

này được máy tính sinh ngẫu nhiên trong khoảng định trước. Theo kinh nghiệm của chúng tôi, thường các giá trị ngẫu nhiên xuất phát trong khoảng $[-6,6]$. Mỗi vector này chính là 1 nghiệm của bài toán cần tìm.

Bước 2: Với mỗi vector nghiệm, thực hiện tính các hàm giá tương ứng của mạng nơron. Nếu giá trị nhỏ nhất của hàm giá vector nào đó thỏa mãn điều kiện dừng thì giải nén vector đó thành các ma trận trọng số của các lớp ẩn, lớp ra.

Bước 3: Thực hiện các toán tử di truyền:

+ Sinh sản (chọn lọc): Thay thế toàn bộ n vector: Giá trị một vector mới được lấy bởi giá trị vector có hàm giá nhỏ hơn giữa 2 cặp vector ngẫu nhiên.

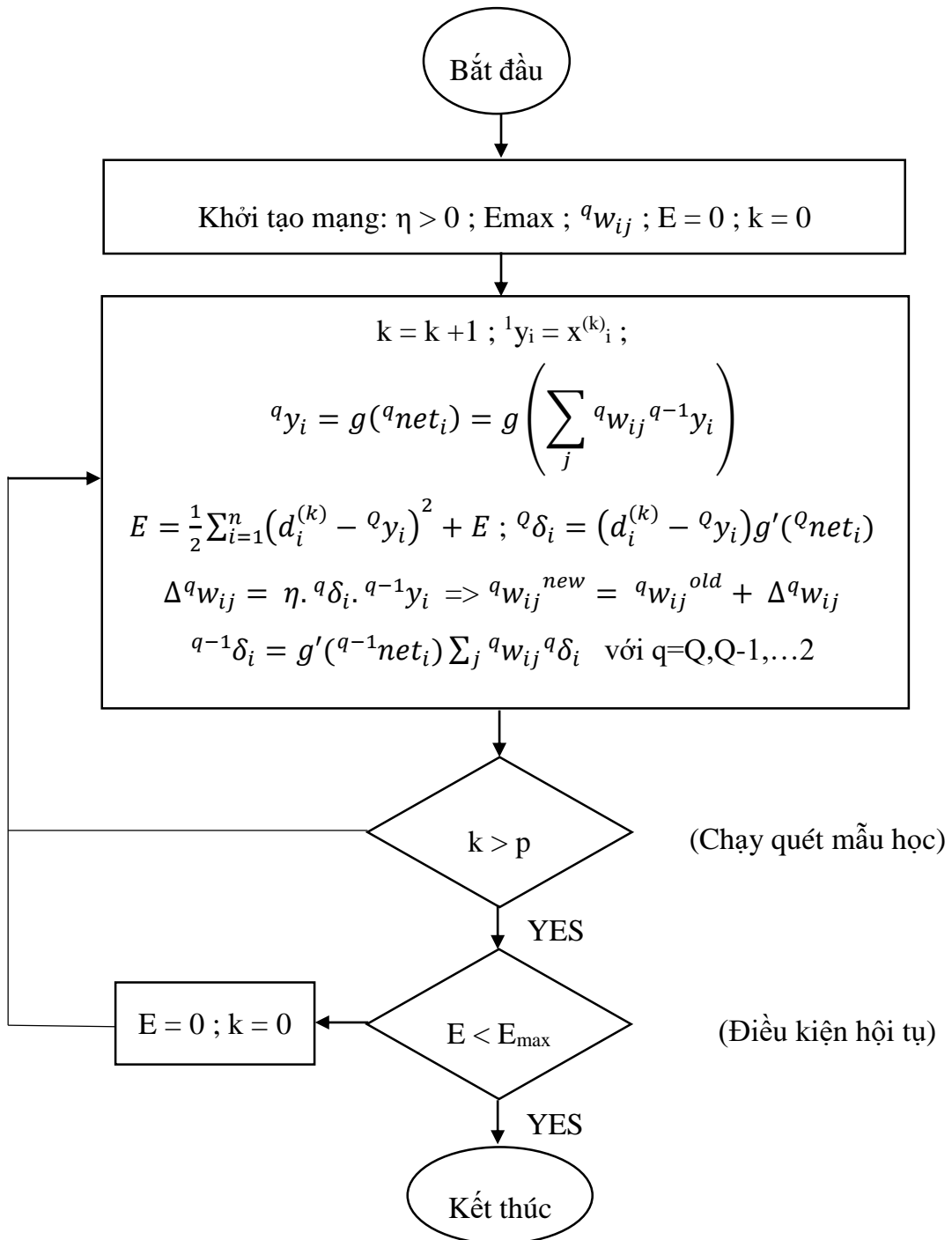
+ Lai ghép: Với tỉ lệ lai ghép cho trước (không lấy toàn bộ số lượng n), mỗi vector mới được lấy ngẫu nhiên một phần giá trị từ 2 vector cũ.

+ Đột biến: Với tỉ lệ đột biến cho trước (không lấy toàn bộ số lượng n), mỗi giá trị trong vector mới được cộng thêm ngẫu nhiên $[-1,1]$ vào giá trị của vector cũ.

Lặp lại bước 2 và 3 khi gặp điều kiện dừng.

Sau mỗi bước thực hiện các toán tử di truyền, các giá trị đánh giá (RMS, NSE) toàn mạng được cập nhật liên tục và được lưu lại các giá trị nhỏ nhất, lớn nhất tương ứng của toàn bộ mạng nơron và được hiển thị trực tiếp lên màn hình tính toán để theo dõi.

3.3 Modul Giải thuật Lan truyền ngược sai số



Hình 3.5 Sơ đồ thuật toán lan truyền ngược sai số

Chi tiết các bước thực hiện đã được trình bày ở phần trên, tuy nhiên đối với từng phương án tính toán mà giải thuật Lan truyền ngược sai số được áp dụng linh hoạt khác nhau.

3.4 Kết hợp các giải thuật

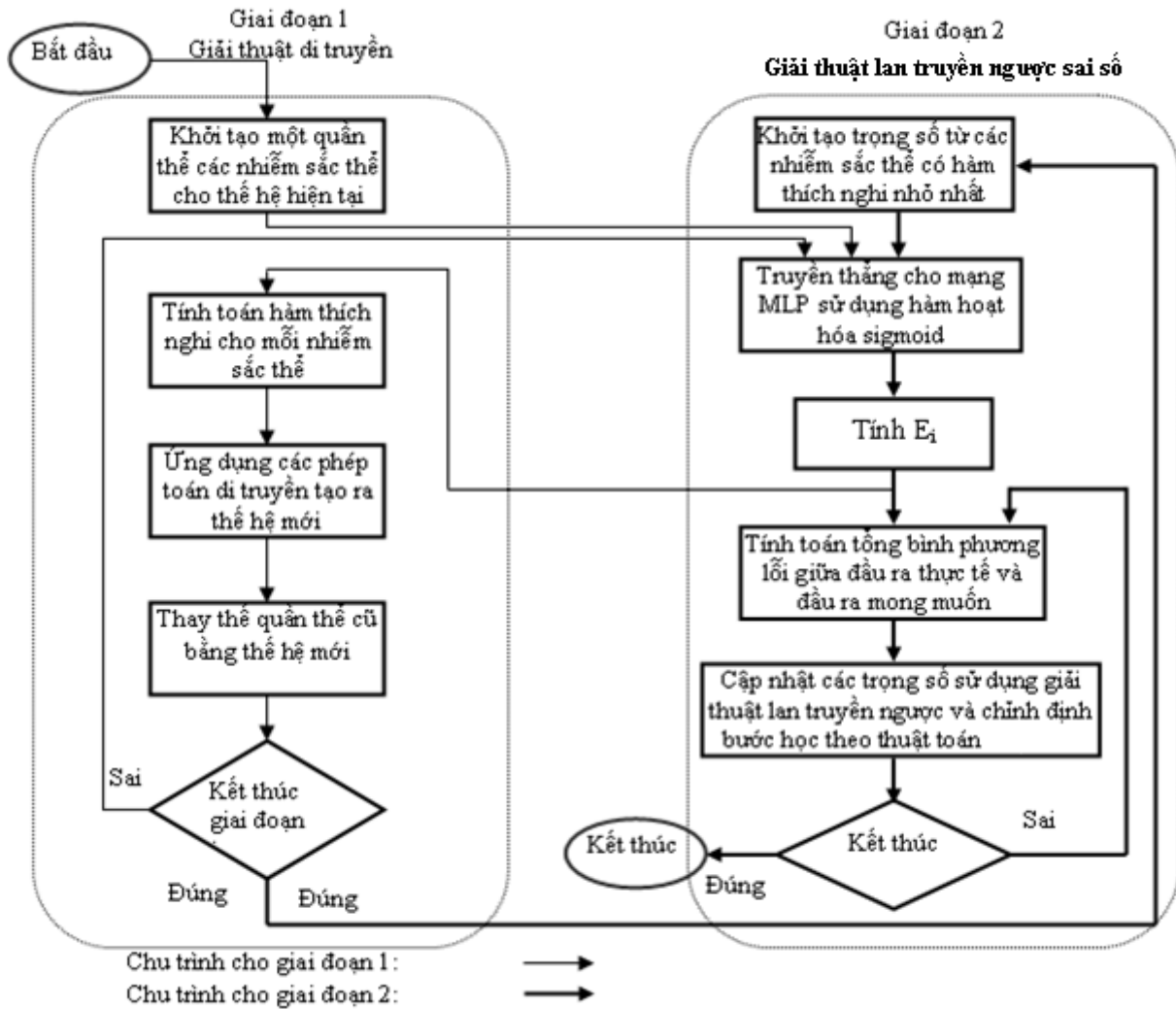
Chương trình cho người dùng có thể lựa chọn phương án tính có sử dụng giải thuật Lan truyền ngược sai số riêng rẽ hoặc kết hợp với giải thuật Di truyền hoặc đọc ma trận trọng số đã có để tính tiếp thì modul này được sử dụng theo những cách khác nhau tương ứng với các đầu vào khác nhau:

+ Phương án chỉ tính giải thuật Lan truyền ngược sai số: Một vectơ ban đầu được sinh ngẫu nhiên tương tự giải thuật Di truyền. Từ nghiệm ngẫu nhiên ban đầu này, thực hiện các bước như trong sơ đồ khối để tìm được nghiệm cuối cùng thỏa mãn điều kiện dừng.

+ Phương án đọc lại ma trận trọng số cũ: Đọc từ file vào các giá trị của phương án đã tính và gán cho vectơ nghiệm xuất phát (không cần sinh ngẫu nhiên như phương án trước). Sau đó thực hiện các bước như trong sơ đồ khối để tìm được nghiệm cuối cùng thỏa mãn điều kiện dừng.

+ Phương án tính kết hợp: Thực hiện tính giải thuật Di truyền, vectơ nghiệm cuối cùng của giải thuật Di truyền (có hàm giá nhỏ nhất) được lấy làm vectơ nghiệm xuất phát của giải thuật Lan truyền ngược sai số. Sau đó thực hiện các bước như trong sơ đồ khối để tìm được nghiệm cuối cùng thỏa mãn điều kiện dừng. Chi tiết kết hợp hai giải thuật được mô tả ở sơ đồ khối Hình 3.6.

Sau mỗi bước thực hiện huấn luyện mạng cho toàn các mẫu học, các giá trị đánh giá (RMS, NSE) toàn mạng được cập nhật liên tục và được lưu lại các giá trị nhỏ nhất, lớn nhất tương ứng của toàn bộ mạng nơron và được hiển thị trực tiếp lên màn hình tính toán để theo dõi.



Hình 3.6 Sơ đồ thuật toán kết hợp giải thuật Di truyền và Lan truyền ngược sai số

3.5 Một số kỹ thuật xử lý

Trái ngược với bước áp dụng bộ trọng số để tính toán dự báo chỉ bao gồm nhân liên tiếp vài bước các ma trận kích thước xác định nên chỉ mất thời gian tính cỡ đơn vị giây, bước huấn luyện mạng để tìm bộ trọng số bao gồm hàng chục ngàn vòng lặp (có thể lên đến hàng triệu – tùy theo điều kiện dừng của sai số) áp dụng cho hàng ngàn, chục ngàn mẫu học, do đó thời gian huấn luyện rất dài. Đối với một số bài toán thực đơn giản cũng có thể lên đến vài ngày, ví dụ như với tiền lực mạnh của Google, cỗ máy chơi cờ vây Alpha Go phải mất đến 40 ngày mới huấn luyện xong mạng từ vài triệu nước cờ có sẵn. Do đó, cần phải nâng cao khả năng tính toán nhờ qua việc song song hóa để đáp ứng nhu cầu thực tiễn.

3.5.1 Kỹ thuật tính toán song song

Tính toán song song hiệu suất cao được thực hiện bằng cách tách các nhiệm vụ lớn và phức tạp ra nhiều phần để chạy trên nhiều đơn vị xử lý. Mặc dù có nhiều phương pháp có thể sử dụng để cải thiện hiệu suất hệ thống các máy tính, nhưng phương pháp thông

dụng nhất để tổ chức và điều phối quá trình xử lý song song là viết code tự động phân tích bài toán sắp đến và cho phép các đơn vị xử lý liên lạc với nhau khi cần thiết trong khi thực hiện công việc.

Không phải tất cả các bài toán đều có thể tính toán song song. Nếu không có công việc con nào có thể thực hiện đồng thời hoặc nếu hệ đang được mô tả phụ thuộc lẫn nhau mạnh (bài toán "fine-grained"), mọi nỗ lực song song hoá nó có thể dẫn tới tăng thời gian tính toán. May mắn là rất nhiều các bài toán khoa học phức tạp có thể phân tích thành các nhiệm vụ riêng biệt để thực hiện độc lập và đồng thời bởi nhiều đơn vị xử lý. Thông thường nhất là việc tách các toạ độ không gian (hoặc thời gian) của hệ được mô hình thành các vùng không gian con (hay khoảng thời gian con) và có thể tính toán đồng thời. Những ứng dụng loại này, được gọi là "coarse grained", khá dễ dàng song song hoá và chúng ta có thể có được lợi ích lớn nhất từ xử lý song song.

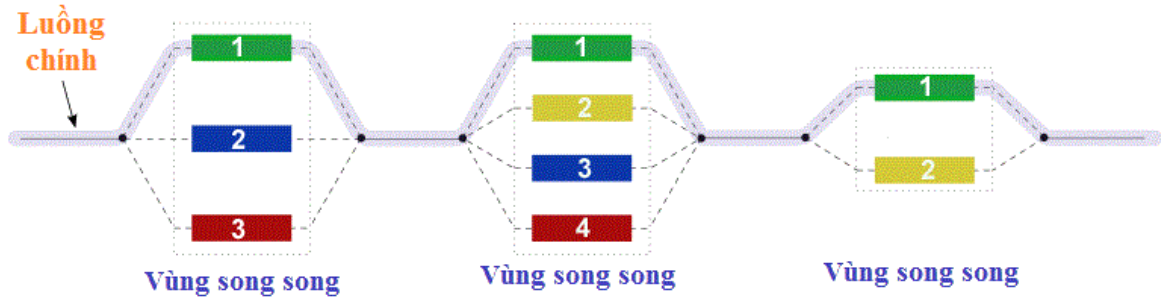
Việc song song hóa một bài toán phụ thuộc vào điều kiện về phần cứng và phần mềm để từ đó lựa chọn phương pháp tối ưu cho bài toán đó. Các phương pháp song song thường dựa trên kiến trúc bộ nhớ cơ bản - bộ nhớ chia sẻ, bộ nhớ phân tán, hay hỗn hợp. Các ngôn ngữ lập trình chia sẻ bộ nhớ giao tiếp bằng cách điều khiển các biến chia sẻ bộ nhớ (OpenMP là API phổ biến nhất sử dụng bộ nhớ chia sẻ), bộ nhớ phân tán sử dụng phương pháp truyền tin trong đó (giao diện truyền tin MPI là API sử dụng hệ thống truyền tin nổi bật nhất). Trong 10 năm trở lại đây, các hệ thống sử dụng card đồ họa (GPU) tính toán được phát triển mạnh mẽ nhờ ưu điểm của chúng so với CPU (số lõi tính toán lớn – hàng ngàn đơn vị xử lý trên 1 card), mỗi card đồ họa có 1 bộ nhớ riêng, các bộ nhớ này được giao tiếp với bộ nhớ của CPU qua các thư viện riêng (ví dụ như CUDA của dòng card đồ họa Nvidia).

a. OpenMP

OpenMP được coi như một giao diện lập trình ứng dụng API (Application Program Interface) chuẩn dành cho lập trình với bộ nhớ chia sẻ. Hệ thống bộ nhớ chia sẻ bao gồm nhiều bộ xử lý CPU, mỗi bộ xử lý truy cập tới bộ nhớ chung thông qua các siêu kết nối hoặc các đường bus. Việc sử dụng không gian địa chỉ đơn làm cho mỗi bộ xử lý đều có một cái nhìn giống nhau về bộ nhớ được sử dụng. Truyền thông trong hệ thống bộ nhớ chia sẻ thông qua cách đọc và ghi dữ liệu giữa các bộ xử lý với nhau lên bộ nhớ. Với cách này, thời gian truy cập tới các phần dữ liệu là như nhau, vì tất cả các quá trình truyền thông đều thông qua đường bus.

Ưu điểm của kiến trúc này là dễ dàng lập trình, bởi vì không yêu cầu sự truyền thông giữa các bộ xử lý với nhau, chúng chỉ đơn giản là truy cập tới bộ nhớ chung.

Có thể xem mô hình lập trình OpenMP như là một mô hình Fork-Join, Hình 3.4.



Hình 3.7 Mô hình Fork-Join

Trong mô hình Fork-Join, tất cả các chương trình OpenMP đều bắt đầu bởi một tiến trình đơn. Đó là master thread (luồng chính), luồng chính này được thực hiện tuần tự cho đến khi gặp chỉ thị khai báo vùng cần song song hóa.

Fork: sau khi gặp chỉ thị khai báo song song, master thread sẽ tạo ra một nhóm các luồng song song. Khi đó, các câu lệnh trong vùng được khai báo song song sẽ được thực hiện song song hóa trên nhóm các luồng vừa được tạo.

Join: khi các luồng đã thực hiện xong nhiệm vụ của mình, chúng sẽ tiến hành quá trình đồng bộ hóa, ngắt luồng, và chỉ để lại 1 luồng duy nhất là master thread.

b. MPI

MPI là một chuẩn chính thức về truyền thông giữa các bộ nhớ phân tán tạo ra bởi một uỷ ban, gọi là Message Passing Interface Forum (MPIF), phát hành năm 1994. Chuẩn này mô tả các đặc điểm và cú pháp của thư viện lập trình song song cần phải có. Có rất nhiều các thư viện dựa trên chuẩn MPI đã được các nhà phát triển phần mềm viết trên nhiều hệ máy tính khác nhau. Nổi bật nhất trong số chúng là MPICH và LAM/MPI. Hiện nay MPI đã nâng cấp lên chuẩn MPI phiên bản 3.0 năm 2012 nhanh hơn và có tính khả chuyển cao hơn và được sử dụng hầu hết ở các hệ thống siêu máy tính trên thế giới.

c. GPU

Kỹ thuật tính toán dùng đơn vị xử lý đồ họa đa dụng - General-Purpose computing of Graphics Processing Units (GPGPU, hay còn gọi tắt là GP²U) là kỹ thuật sử dụng đơn vị xử lý đồ họa GPU (vốn được thiết kế để tính toán đồ họa máy tính) để thực hiện những tác vụ trước đây được xử lý bởi CPU. Thông thường các chức năng của GPU được giới hạn trong việc xử lý đồ họa máy tính. Trong rất nhiều năm, GPU chỉ được sử dụng để tăng

tốc một vài phần trong đồ họa đường ống (graphics pipeline). Từ đòi hỏi của thị trường cho đồ họa 3D thời gian thực và đồ họa với độ phân giải cao, các GPU đã phát triển với kiến trúc song song hóa mức cao, xử lý đa luồng với kiến trúc manycore processor đã đem lại khả năng tính toán cùng với băng thông bộ nhớ rất cao, thậm chí còn vượt qua những CPU thông thường.

Lý do dẫn đến việc GPU có khả năng tính toán các phép tính dấu phẩy động cao hơn CPU là vì GPU được thiết kế cho các tác vụ đòi hỏi sự song song hóa ở mức cao, đó cũng chính là đòi hỏi của việc rendering đồ họa. Chính vì thế nên trong thiết kế, GPU sử dụng phần lớn transistors cho việc xử lý dữ liệu hơn là việc điều khiển luồng và đưa dữ liệu vào bộ nhớ đệm (data caching).



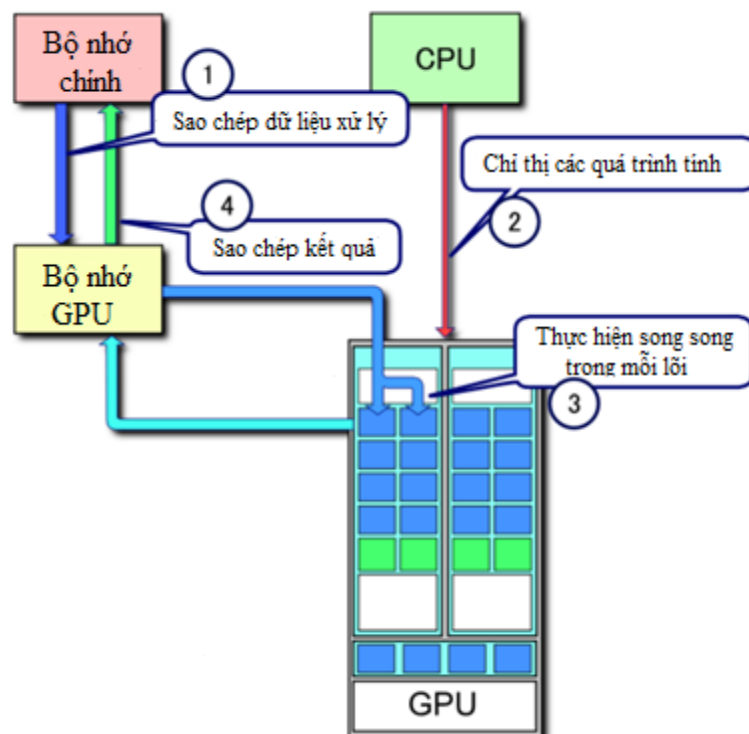
Hình 3.8 So sánh kiến trúc CPU và GPU.

GPU có các đặc tính sau:

- Nhấn mạnh xử lý song song (Emphasize parallelism): GPU là về cơ bản máy song song và việc sử dụng hiệu quả nó phụ thuộc vào mức độ xử lý song song trong khối lượng công việc. Ví dụ, NVIDIA CUDA chạy hàng ngàn luồng chạy tại một thời điểm, tối đa hóa cơ hội che giấu độ trễ bộ nhớ bằng cách sử dụng đa luồng. Nhấn mạnh xử lý song song đòi hỏi lựa chọn các thuật toán mà chia miền tính toán thành càng nhiều mảnh độc lập càng tốt. Để tối đa hóa số lượng luồng chạy đồng thời, GPU lập trình cũng nên tìm cách giảm thiểu việc sử dụng thread chia sẻ tài nguyên (như dùng các thanh ghi cục bộ và bộ nhớ dùng chung CUDA), và nên sự đồng bộ giữa các luồng là ít đi.
- Giảm thiểu sự phân kỳ SIMD (Minimize SIMD divergence): GPU cung cấp một mô hình lập trình SPMD: nhiều luồng chạy cùng một chương trình tương tự, nhưng truy cập dữ liệu khác nhau và do đó có thể có sự khác nhau trong thực thi của chúng. Tuy nhiên, trong một số trường hợp đặc biệt, GPU thực thi chế độ SIMD cho các lô các luồng. Nếu luồng trong một lô trệch ra, toàn bộ lô sẽ

thực thi cùng các đường code cho đến khi các luồng hội tụ lại. Tính toán hiệu năng cao GPU đòi hỏi cơ cấu code sao cho giảm thiểu sự phân kỳ trong lô.

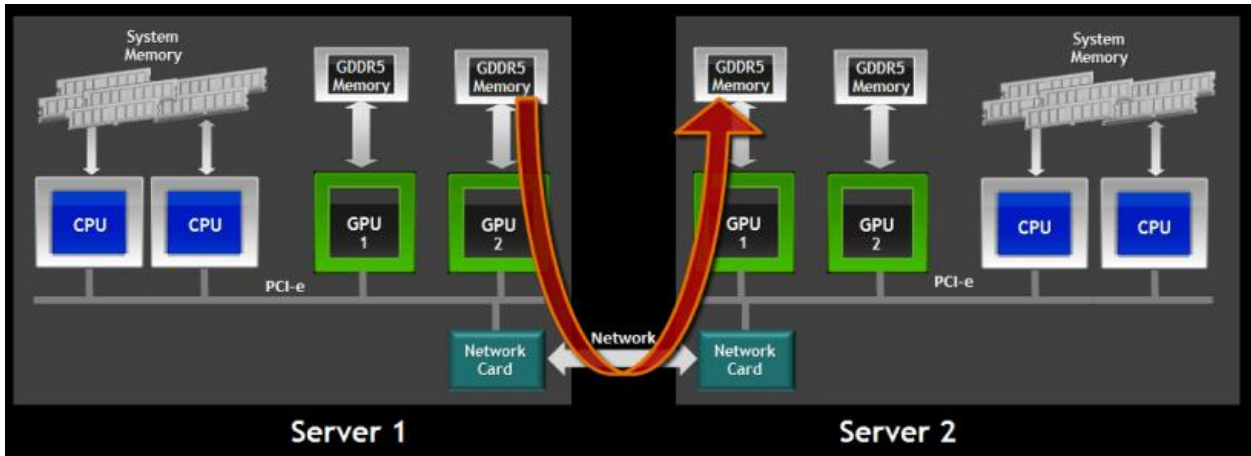
- Tăng tối đa cường độ số học (Maximize arithmetic intensity): Trong khung cảnh tính toán ngày nay, các tính toán thực tế là tương đối tốt nhưng băng thông bị hạn chế. Điều này thật sự rất đúng với GPU nơi hỗ trợ rất tốt tính toán dấu phẩy động. Để tận dụng tối đa sức mạnh đó cần cấu trúc thuật toán để tối đa hóa cường độ số học, hoặc số lượng các tính toán trên số thực hiện trong mỗi thao tác với bộ nhớ. Truy cập dữ liệu mạch lạc bằng các luồng trợ giúp riêng biệt bởi vì các thao tác này có thể kết hợp để làm giảm tổng số thao tác bộ nhớ. Sử dụng bộ nhớ dùng chung CUDA trên GPU NVIDIA cũng giúp giảm overfetch (do các luồng có thể giao tiếp) và cho phép các chiến lược "blocking" việc tính toán trên bộ nhớ của chip.
- Khai thác băng thông dòng (Exploit streaming bandwidth): Mặc dù có tầm quan trọng của cường độ số học, nó là cần lưu ý rằng GPU có băng thông rất ít (very high peak) trên bộ nhớ đi kèm, cỡ 1/10 băng thông bộ nhớ thông dụng trên nền máy PC. Đây là lý do tại sao GPU có thể thực thi tốt hơn CPU ở các tác vụ như sắp xếp, trong đó có tỷ lệ tính toán/băng thông thấp. Để đạt được hiệu năng cao trên các ứng dụng như thế đòi hỏi các mẫu truy cập bộ nhớ dòng (streaming) trong đó các luồng đọc và ghi vào các khối lớn liền mạch (tối đa hóa băng thông cho mỗi giao dịch) nằm trong các khu vực riêng biệt của bộ nhớ (tránh các rủi ro dữ liệu).



Hình 3.9 Tiến trình thực hiện của 1 chương trình CUDA

d. Mô hình kết hợp

Tận dụng ưu điểm của các mô hình trên, phần lớn các siêu máy tính hiện nay được xây dựng là sự kết hợp của các công nghệ trên. Hệ thống này bao gồm nhiều máy chủ, trên mỗi máy chủ bao gồm nhiều CPU và GPU cùng thực hiện chức năng tính toán. Chúng được liên kết với nhau qua các card mạng Ethernet có tốc độ Gbps hoặc các card chuyên dụng có tốc độ lên đến 100Gbps trực tiếp giữa các bộ nhớ của GPU.



Hình 3.10 Mô hình siêu máy tính có card mạng kết nối chuyên dụng GPUDirect RDMA

Bên cạnh đó, các hãng sản xuất hiện nay cũng đang tập trung nghiên cứu sản xuất các bộ vi xử lý có cấu trúc tối ưu phục vụ tính toán trí tuệ nhân tạo như các dòng FPGA (Field-Programmable Gate Arrays).

e. Ứng dụng kỹ thuật song song vào xây dựng phần mềm mạng nơ-ron thần kinh nhân tạo

Để tạo điều kiện thuận lợi cho phát triển mã nguồn cũng như sử dụng, chúng tôi thực hiện 2 giải pháp song song cho phần mềm của mình là kỹ thuật OpenMP và kỹ thuật dùng card đồ họa GPU của Nvidia trên nền CUDA. Sự tiện lợi của hai giải pháp này là người lập trình không cần phải có kiến thức về cài đặt, cấu hình chính xác mạng máy tính mà chỉ đóng gói kèm thư viện động hoặc cần cài các driver mặc định của nhà sản xuất phần cứng là sử dụng được. Khi được chuyển giao, người sử dụng cũng dễ dàng chạy trên các máy tính cá nhân nhanh chóng mà không cần kiến thức chuyên sâu về song song.

+ **OpenMP**: Ứng dụng được cho mọi máy tính phổ thông CPU đa lõi hiện có trên thị trường. Trong thử nghiệm này, chúng tôi sử dụng CPU Intel i7 3770 4 lõi 8 thread với tốc độ 3.9 Ghz, năng lực tính toán đầu phẩy động theo công bố của nhà sản xuất là 125 GFLOP.

Khai báo thư viện sử dụng:

```
#ifdef _OPENMP
    include 'omp_lib.h'
#endif
#ifdef _OPENMP
```

Đối với cả hai giải thuật, trong vòng lặp điều kiện dừng (RMS nhỏ hơn sai số cho trước hoặc số bước lặp lớn hơn số cho trước), thao tác huấn luyện mạng được chạy cho nhiều mẫu học. Mỗi mẫu học này được chương trình phân bổ trên các lõi CPU. Do đó, sau dòng lệnh vòng lặp điều kiện dừng và trước dòng lệnh vòng lặp huấn luyện theo các mẫu, bấy song song được đặt để chia nhỏ công việc cho các CPU, có kèm ghi chú các biến dùng chung cũng như các biến riêng cho từng lõi.

```
!$omp parallel SHARED(danh sách biến chung...) private(danh sách biến riêng...)
!$omp do
```

Kết thúc vòng lặp huấn luyện, đặt bẫy chặn để chờ các lõi hoàn tất phần tính của mình, các giá trị tính được này được tập trung cập nhật vào bộ nhớ chung và hoàn tất phần tính song song.

```
!$omp end do
!$omp barrier
!$omp end parallel
```

+ **GPU**: Trong khuôn khổ đề tài cơ sở này, chúng tôi có sử dụng kinh phí để mua thiết bị tính toán là 01 card NVIDIA GeForce GTX 1060 6GB với thông số được trình bày như bảng 3.1. Đây là một cấu hình khuyến cáo tối thiểu của nhiều chuyên gia dành cho một hệ thống trí tuệ nhân tạo phục vụ công việc nghiên cứu, ứng dụng cơ bản,... cũng như tham gia sàn đấu kháng **Kaggle** dành cho các chương trình học máy.

Bảng 3.1 Thông số card NVIDIA GeForce GTX 1060 6GB

Thông số	Chi tiết
Tên mã thiết bị	GP106 (8/2016)
Transistors	4.4 tỉ
Số lõi tính	1280
Xung nhịp tính	1708 Mhz
Bộ nhớ	6144 MB / GDDR5 / 192 bit / 192.2 GB/s
Băng thông giao tiếp	PCIe 3.0 x16 / 16Gbps
Năng lực tính toán dấu phẩy động	4.375 TFLOPS

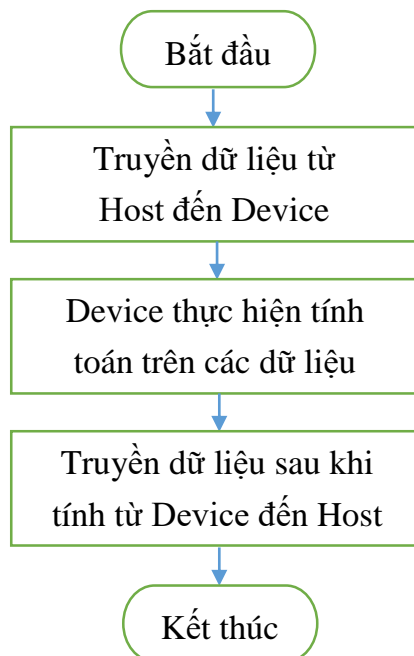
Cách thức thực hiện với GPU như sau:

Đối với module Giải thuật Di truyền, chúng tôi chọn phương án song song dựa trên số lượng quần thể, mỗi cá thể trong quần thể sẽ được 1 đơn vị tính toán xử lý (xử lý trên tất cả các mẫu học của cá thể đó). Điều này sẽ giúp thuật toán song song có lợi khi số lượng

quần thể là lớn. Tuy nhiên, chúng tôi chỉ xử lý song song với các cá thể khi tính toán các hàm mục tiêu (hàm thích nghi), còn bỏ qua phần xử lý các phép toán di truyền chỉ chiếm một khối lượng nhỏ tính toán, chủ yếu là các phép gán giá trị mảng.

Đối với giải thuật Lan truyền ngược sai số, do chỉ còn 1 cá thể tối ưu nhất, các phép lặp chỉ thực hiện theo các mẫu học, do đó, cần chia mỗi lần xử lý 1 mẫu học cho 1 đơn vị xử lý. Việc xử lý trên mỗi nút tính toán chỉ bao gồm các bước tính xuôi và lan truyền ngược, điều này giúp thuật toán song song có lợi khi số lượng mẫu học là lớn. Ưu điểm của phần song song này so với Giải thuật Di truyền là song song được phần lớn các bước tính, dữ liệu truyền sau mỗi bước tính là nhỏ.

Sơ đồ khối chung cho cả 2 giải thuật khi song song trên GPU được thể hiện như hình 3.11.



Hình 3.11 Sơ đồ khối cách thực hiện song song trên GPU

Trong đó: Host: là những tác vụ và cấu trúc phần cứng, phần mềm được xử lý từ CPU.

Driver: là những tác vụ và cấu trúc phần cứng, phần mềm được xử lý từ GPU.

Quá trình được diễn giải như sau:

- Dữ liệu cần được tính toán luôn ở trên bộ nhớ của Host vì vậy bước 1 là truyền dữ liệu cần tính toán từ bộ nhớ Host qua bộ nhớ Device.
- Sau đó Device sẽ gọi các hàm riêng của mình để tính toán dữ liệu đó.
- Sau khi tính toán xong, dữ liệu cần được trả về lại cho bộ nhớ của Host.

f. Một số kết quả thu được

Để đánh giá hiệu quả của phương pháp song song, chúng tôi lựa chọn 2 bài toán với các số liệu thực có cấu trúc mạng và số lượng mẫu học, số lượng quần thể khác nhau được mô tả trong Bảng 3.2.

Bảng 3.2 Thông số các bài toán thử nghiệm.

STT	Thông số	Bài toán 1	Bài toán 2
1	Số neuron lớp vào	8	49
2	Số neuron lớp ẩn	13	100
3	Số neuron lớp ra	1	1
4	Số lượng trong quần thể di truyền	1.000	2.000
5	Số bước lặp di truyền	100	50
6	Số bước lặp phương pháp Lan truyền ngược sai số	100.000	100.000
7	Số lượng mẫu học	480	1.000

Do tính chất ngẫu nhiên của nghiệm ma trận trọng số ban đầu, sai số dừng của 2 giải thuật được đặt thật nhỏ để đảm bảo không hội tụ trước khi đạt số vòng lặp dừng (bắt buộc mọi lần tính toán phải có cùng 1 số lượng phép tính). Như vậy rõ ràng, số phép tính của bài toán 2 là hơn hơn nhiều so với bài toán 1. Khi đó, từ kết quả 2 bài toán có thể đánh giá được cả thời gian trễ khi truyền dữ liệu qua lại giữa các đơn vị tính toán.

Mỗi bài toán được chúng tôi chạy 100 lần để tính thời gian trung bình của mỗi phương pháp, kết quả được thể hiện ở bảng 3.3 dưới đây.

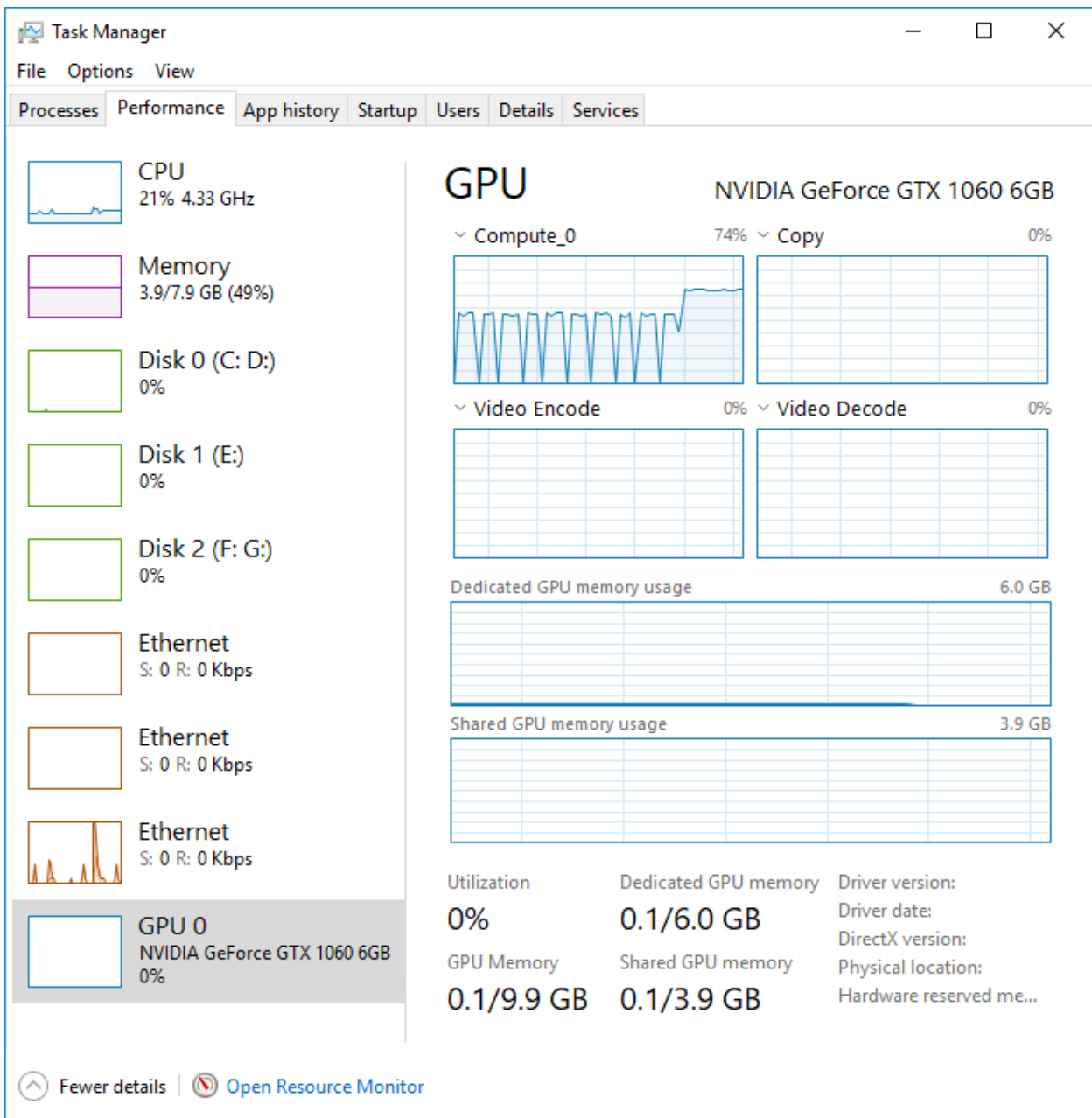
Bảng 3.3 Thời gian tính trung bình của các bài toán thử nghiệm

Phương án tính	Tính đơn nhân CPU	Tính đa nhân CPU (OpenMP)	Tính đa nhân GPU	Tỉ lệ thời gian
Bài toán 1	52s	19s	90s	1:0.36:1.73
Bài toán 2	1h:22m:53s (4973s)	0h:20m:24s (1224s)	0h:04m:34s (274s)	1:0.25:0.055 ~ (1:4:18)

Nhận thấy: Khi kích thước các ma trận là nhỏ, số lượng mẫu học nhỏ, số lượng quần thể là nhỏ, việc song song trên GPU lại không mang lại hiệu quả, thậm chí còn gây ra sự chậm trễ tổng thể tính toán do phải truyền dữ liệu nhiều; tính đa lõi CPU bằng OpenMP đạt hiệu quả tốt nhưng cũng chưa đạt hết mức độ thiết kế. Khi số lượng neuron các lớp tăng, số lượng mẫu học tăng, số lượng quần thể tăng, việc tính toán song song cho bài toán mạng neuron nhân tạo càng hiệu quả do tính song song của các bài toán này lớn. Tính song song thể hiện ở chỗ: các mẫu huấn luyện mạng, các bước huấn luyện mạng là độc lập. Do đó,

khi áp dụng tính song song cho bài toán mạng nơon nhân tạo đem lại hiệu quả cao. Đối với các bài toán khác trong thực tế, thời gian huấn luyện lên đến đơn vị hàng tháng, hoặc các tác vụ yêu cầu xử lý thời gian thực, việc song song hóa tính toán là một yêu cầu bắt buộc.

Ngoài ra, chúng tôi đánh giá kết quả qua việc phân tích biểu đồ đo hiệu năng card đồ họa của Windows bằng phần mềm thông dụng Task Manager (phiên bản Windows 10 Fall Creator phát hành vào 17/10/2017) như hình 3.12.



Hình 3.12 Biểu đồ hiệu năng card đồ họa khi đang xử lý song song trên GPU bài toán 2

Ta có một vài nhận xét:

- + Thuật toán song song áp dụng cho tập dữ liệu 2 bài toán trên chưa tận dụng hết 1280 lõi của card đồ họa nên chưa đạt hiệu quả cao nhất. Khi xử lý giải thuật

Di truyền chỉ đạt ~50%, khi xử lý giải thuật Lan truyền ngược sai số chỉ đạt 75% công suất.

- + Giải thuật Di truyền như đã mô tả ở trên chưa được song song các phép toán di truyền, khối lượng truyền dữ liệu lại lớn, nên xảy ra hiện tượng ngắt quãng thời gian xử lý trên card đồ họa mặc dù bài toán 2 là bài toán có khối lượng tính toán lớn so với bài toán 1. Do đó cần tối ưu việc song song trên giải thuật Di truyền thêm nữa để có hiệu năng tốt cho mọi bài toán từ nhỏ đến lớn.

3.5.2 Kỹ thuật phân tích dữ liệu đầu vào

Khi ta thu nhận dữ liệu đo đạc để làm dữ liệu đầu vào thì gặp phải vấn đề đáng lo ngại ảnh hưởng đến tính đúng đắn của dữ liệu là tỷ lệ nhiễu có trong dữ liệu. Nhìn chung, nhiễu được xem như là một dữ liệu và là thành phần không mong muốn trong dữ liệu thủy văn đầu vào. Vì vậy, dữ liệu đo đạc thu được cung cấp những thông tin ban đầu được gọi là dữ liệu thô. Chất lượng của dữ liệu đầu vào có thể được xác định theo:

- Tỷ lệ dữ liệu trên nhiễu cao nhất trong thông tin của dữ liệu đầu vào thì càng tốt và số lượng nhiễu nên thấp nhất.
- Sự biến dạng của dữ liệu đầu vào càng ít càng tốt.

Do đó, để có được dữ liệu đáng tin cậy người ta đã sử dụng rất nhiều phương pháp khác nhau để phân tích và xử lý dữ liệu.

a. Biến đổi Fourier

Biến đổi Fourier dựa trên cơ sở chia một dữ liệu thành tổng các hàm sin với tần số khác nhau hay nó là kỹ thuật biến đổi dữ liệu từ miền thời gian sang miền tần số. Phép biến đổi Fourier là phương pháp phổ biến nhất được sử dụng để xác định phổ tần số của dữ liệu. Nhưng hạn chế lớn của phương pháp này là khi biến đổi sang miền tần số thì thông tin thời gian bị mất đi, khi nhìn vào một biến đổi Fourier của dữ liệu không thể biết được thời gian xảy ra sự kiện. Nó chỉ thích hợp với các dữ liệu dừng, không thích hợp trong xử lý những dữ liệu không dừng như dữ liệu thủy văn, thủy lực.

b. Phép biến đổi wavelet

Một lý do chính trong việc nghiên cứu và sử dụng phép biến đổi wavelet là phép biến đổi Fourier không chứa thông tin cục bộ của dữ liệu. Vì thế phép biến đổi Fourier không được sử dụng để phân tích dữ liệu trong cùng một miền thời gian và tần số. Trong khi đó, wavelet lại có ưu điểm chính là khả năng thực hiện phân tích cục bộ, thể hiện được đặc tính của dữ liệu mà các kỹ thuật phân tích khác không có. Wavelet là dạng sóng có thời gian duy trì tới hạn với giá trị trung bình bằng không. Wavelet có thời gian giới hạn, bất thường và bất đối xứng. Phân tích wavelet chia các dữ liệu thành các tham số dịch chuyển và tham số tỷ lệ của các wavelet mẹ.

Như đã phân tích ở trên, với các đặc tính phù hợp, chúng tôi lựa chọn phép biến đổi Wavelet để chuẩn hóa các dữ liệu đầu vào phục vụ cho mô hình học máy đã được xây dựng.

Xét tín hiệu chuỗi tín hiệu liên tục theo thời gian $f(t)$, phép biến đổi *wavelet* liên tục (*continuous wavelet transform - CWT*) có thể định nghĩa như là một ánh xạ chuyển hàm một biến $f(t)$ thành hàm 2 biến $W(a, b)$ như sau:

$$W(a, b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t) \Psi^* \left(\frac{t-b}{a} \right) dt$$

Trong đó:

+ $\psi(t)$ = hàm wavelet mẹ (mother wavelet) thỏa mãn tính chất sóng (nghĩa là dao động với giá trị trung bình của hàm wavelet bằng không) và có năng lượng là đơn vị; ký hiệu ‘*’ để chỉ liên hợp phức (complex conjugate).

+ a : Nghịch đảo của tần số (nó chia tỷ lệ một hàm bằng việc nén hoặc giãn một tín hiệu).

+ b : Dịch chuyển của tín hiệu dọc theo trục thời gian.

Thực chất CWT tìm kiếm sự tương quan giữa tín hiệu liên tục và các hàm wavelet. Hiện tại có thể tìm thấy nhiều loại hàm wavelet mẹ khác nhau, chẳng hạn như Haar wavelet, Daubechies wavelet, Coiflet wavelet hoặc Biorthogonal wavelet. Tuy nhiên, việc áp dụng loại hàm wavelet mẹ nào hoàn toàn phụ thuộc vào từng bài toán cụ thể cũng như là đặc trưng của tập dữ liệu được sử dụng.

Trong thực tế chuỗi tín hiệu quan trắc luôn luôn là rời rạc, vì vậy nếu ta chọn tham số a và b trên cơ sở hàm mũ của 2 thì khi đó phép biến đổi *wavelet* sẽ trở nên đơn giản hơn. Người ta gọi nó là biến đổi *wavelet* rời rạc (*discrete wavelet transform - DWT*). Khi đó, ta có phép biến đổi *DWT* cho mỗi tín hiệu rời rạc f_i như sau:

$$W_{m,n} = 2^{-\frac{m}{2}} \sum_{i=0}^{N-1} f_i \Psi^* (2^{-m} i - n)$$

Trong đó: chỉ số i có giá trị $i = 0, 1, 2, \dots, N-1$ và $N = 2^M$; hai chỉ số m và n kiểm soát tần số và vị trí; $W_{m,n}$ = hệ số *wavelet* cho tham số $a = 2^m$ và $b = 2^m n$. Sử dụng phép biến đổi *DWT* ngược (*inverse discrete wavelet transform*) ta có thể nhận được biểu thức cho tín hiệu rời rạc f_i ban đầu như sau:

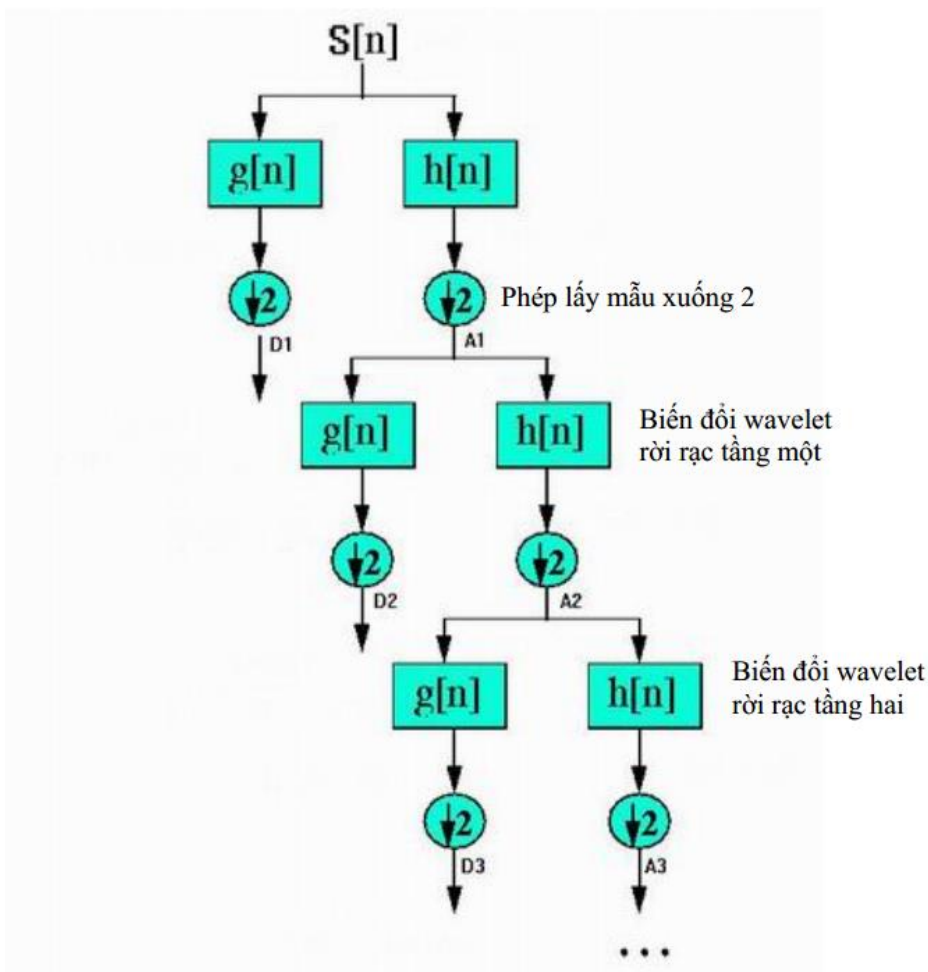
$$f_i = A_{M,i} + \sum_{m=1}^M \sum_{n=0}^{(2^{M-m}-1)} W_{m,n} 2^{-\frac{m}{2}} \Psi(2^{-m} i - n)$$

Hoặc có thể viết lại dưới dạng đơn giản hơn:

$$f_i = A_{M,i} + \sum_{m=1}^M D_{m,i}$$

Trong đó: mỗi một tín hiệu f_i được phân tích thành hai thành phần: thành phần xấp xỉ (*approximation*) $A_{M,i}$ (tương ứng với tần số thấp) và thành phần chi tiết (*detail*) $D_{m,i}$ (tương ứng với tần số cao) tại các tầng (*level*) m khác nhau ($m = 1, 2, \dots, M$). Thành phần xấp xỉ cung cấp thông tin chung và cơ bản nhất của tín hiệu, trong khi đó các thành phần chi tiết cung cấp thông tin về những dị biệt của tín hiệu. Việc phân tích tín hiệu thành hai thành phần thông qua hai bộ lọc thông thấp và thông cao như mô tả trong hình 3.10. Trong đó, bộ lọc thông cao sử dụng hàm wavelet $\psi(x)$ và bộ lọc thông thấp sử dụng hàm tỉ lệ (scaling function) $\Phi(x)$.

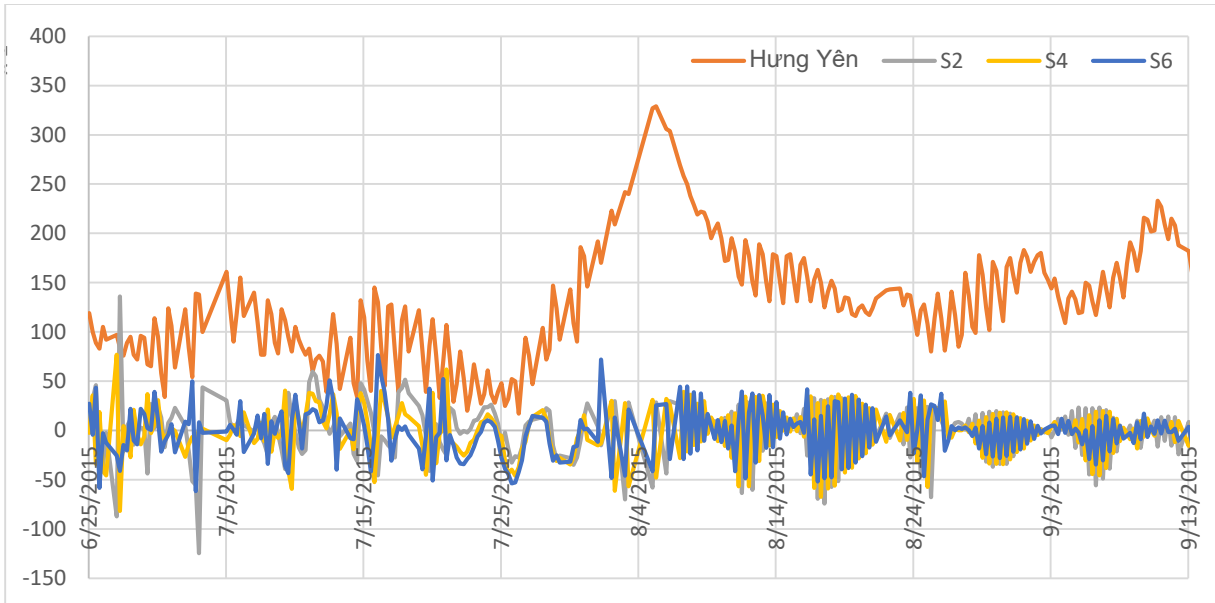
Các phép lọc được tiến hành với nhiều tầng (level) khác nhau và để khối lượng tính toán không tăng, khi qua mỗi bộ lọc, tín hiệu được lấy mẫu xuống 2. Ứng với mỗi tầng, tín hiệu có độ phân giải khác nhau. Do đó, phép biến đổi wavelet rời rạc được gọi là phân tích đa phân giải (Multiresolution analysis).



Hình 3.13 Phân tích đa phân giải sử dụng biến đổi wavelet rời rạc

Phương pháp *WT* có thể kết hợp với mô hình *ANN* cho các bài toán liên quan đến việc phân tích chuỗi số liệu rời rạc theo thời gian. Trong trường hợp đó, trước hết chuỗi số liệu sẽ được chuyển đổi thành bộ số liệu mới với các thành phần xấp xỉ và thành phần chi tiết dựa vào phép biến đổi *DWT* tại một tầng thích hợp nhất. Các thành phần này sau đó sẽ được sử dụng như là đầu vào cho mạng *ANN*.

Thử nghiệm với chuỗi số liệu mực nước đo được tại trạm thủy văn Hưng Yên mùa lũ năm 2016, sử dụng họ wavelet Daubechies phân tích 3 bậc thành các chuỗi số liệu làm tín hiệu đầu vào cho mạng *ANN* được mô tả như hình 3.14.



Hình 3.14 Phân tích chuỗi dữ liệu mực nước trạm Hưng Yên

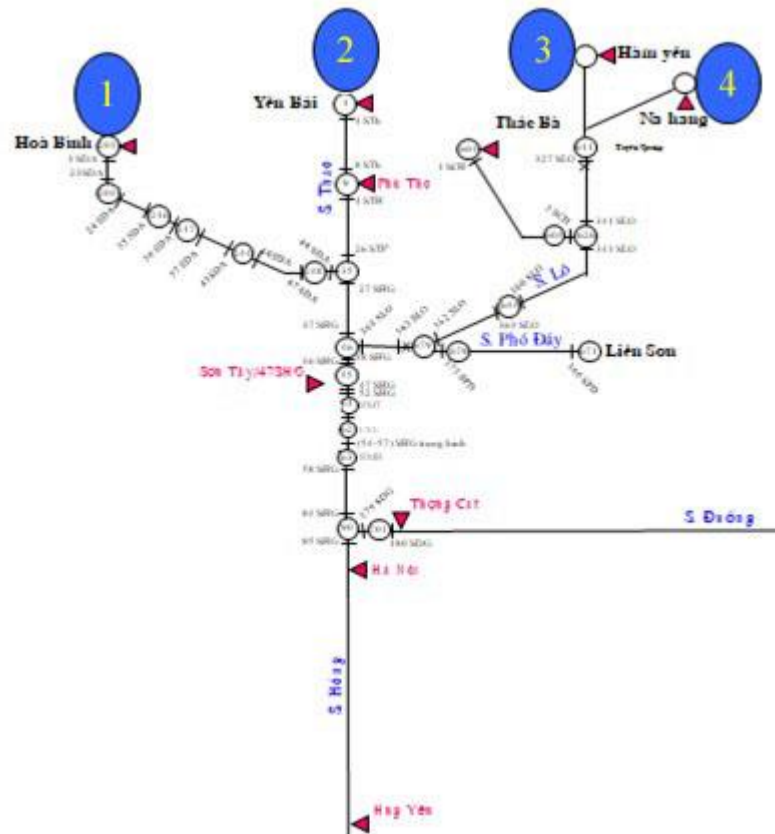
CHƯƠNG IV

MỘT SỐ KẾT QUẢ TÍNH TOÁN

4.1 Kết quả mô phỏng và dự báo thủy lực lưu vực đồng bằng sông Hồng

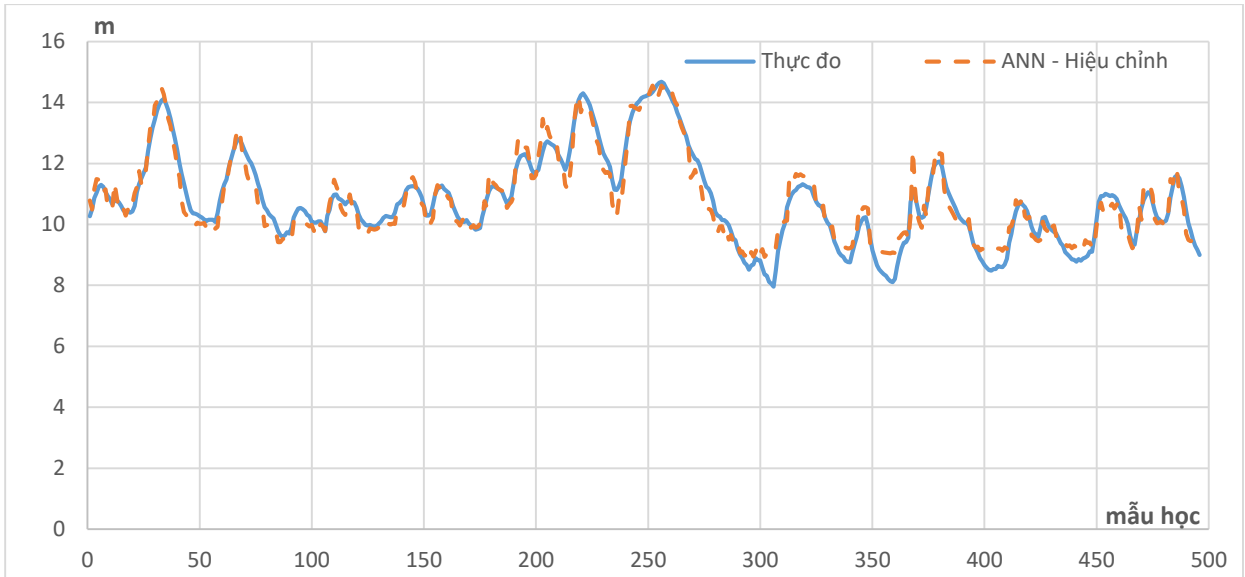
Hàng năm, từ 15 tháng 6 đến 15 tháng 9 cư dân vùng châu thổ sông Hồng - Thái Bình luôn sống trong tình trạng bị uy hiếp bởi nạn lũ lụt. Chỉ trong mấy thập kỷ vừa qua, chúng ta đã phải chứng kiến các trận lũ lớn trong các năm 1945, 1971, 1996. Nền kinh tế càng phát triển thì thiệt hại do lũ lụt gây ra càng lớn. Để chủ động phòng tránh và hạn chế các tác hại của lũ lụt, song song với việc phát triển kinh tế, chúng ta phải luôn luôn tìm kiếm các biện pháp hữu hiệu đối phó với lũ lụt. Tập thể phòng Thủy động lực học và **Giảm nhẹ Thiên tai trong lưu vực – Viện Cơ học** đã xây dựng và phát triển nhiều mô hình thủy lực nhằm mô phỏng và dự báo các yếu tố thủy văn thủy lực, từ đó phục vụ cho việc đề xuất, đánh giá và điều hành phòng chống lụt bão.

Trong đề tài này, sẽ sử dụng lại một phần số liệu được chọn lọc [7] của mô hình thủy lực đã phát triển để tính toán, thử “dự báo” lại mực nước và lưu lượng tại một số trạm trong hệ thống.

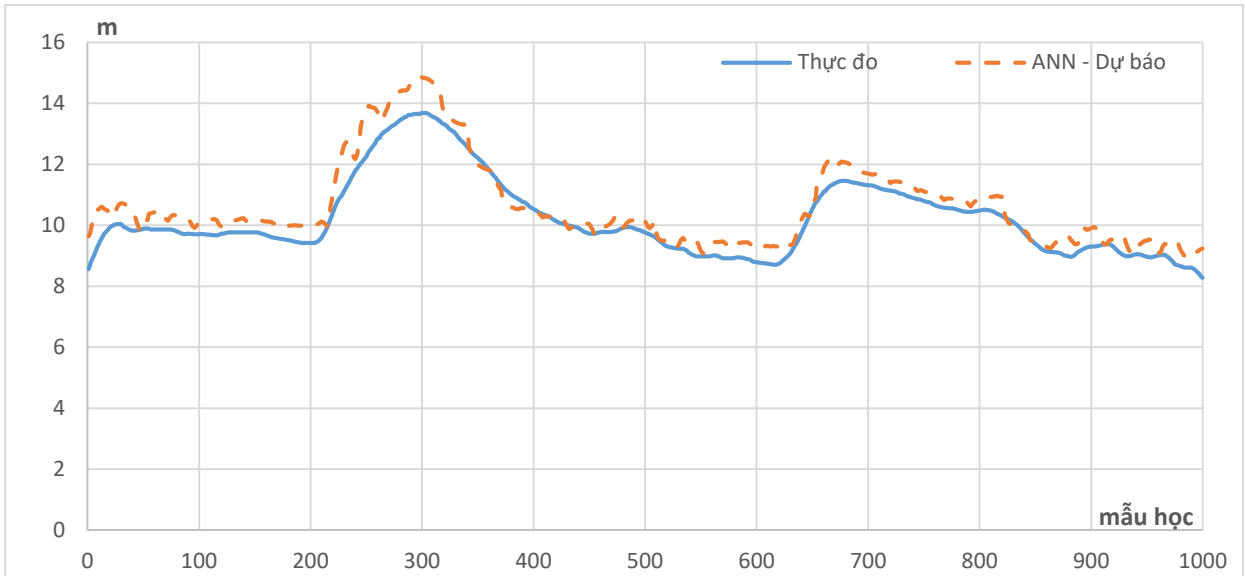


Hình 4.1 Sơ đồ mạng sông vùng nghiên cứu

Phương án 1: Lấy dữ liệu 4 nguồn tại Hòa Bình, Yên Bái, Hàm Yên, Na Hang các năm 2000, 2002, 2003 làm thành tập huấn luyện, dữ liệu năm 2004 dùng để làm tập kiểm định “dự báo lại”.



Hình 4.2 Kết quả so sánh mực nước Sơn Tây năm 2000,2002,2003 giữa thực đo và mạng ANN hiệu chỉnh.



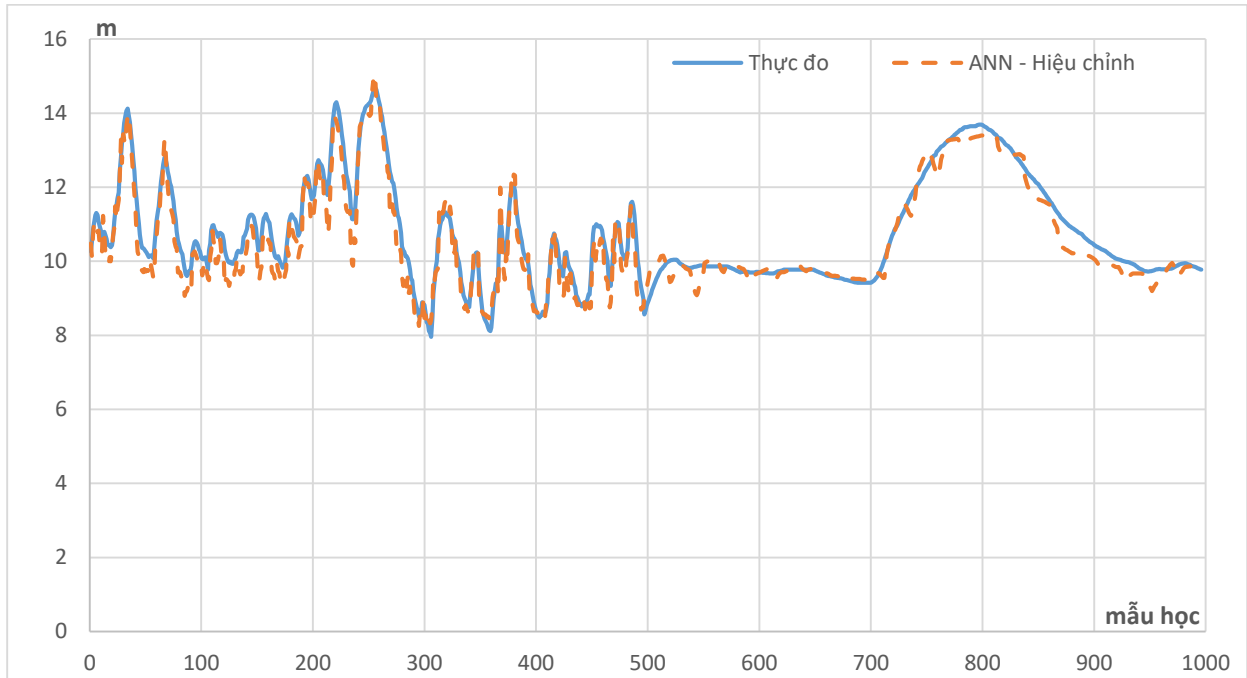
Hình 4.3 Kết quả so sánh mực nước Sơn Tây năm 2004 giữa thực đo và mạng ANN kiểm định.

Các kết quả phương án tính được trình bày trong bảng 4.1

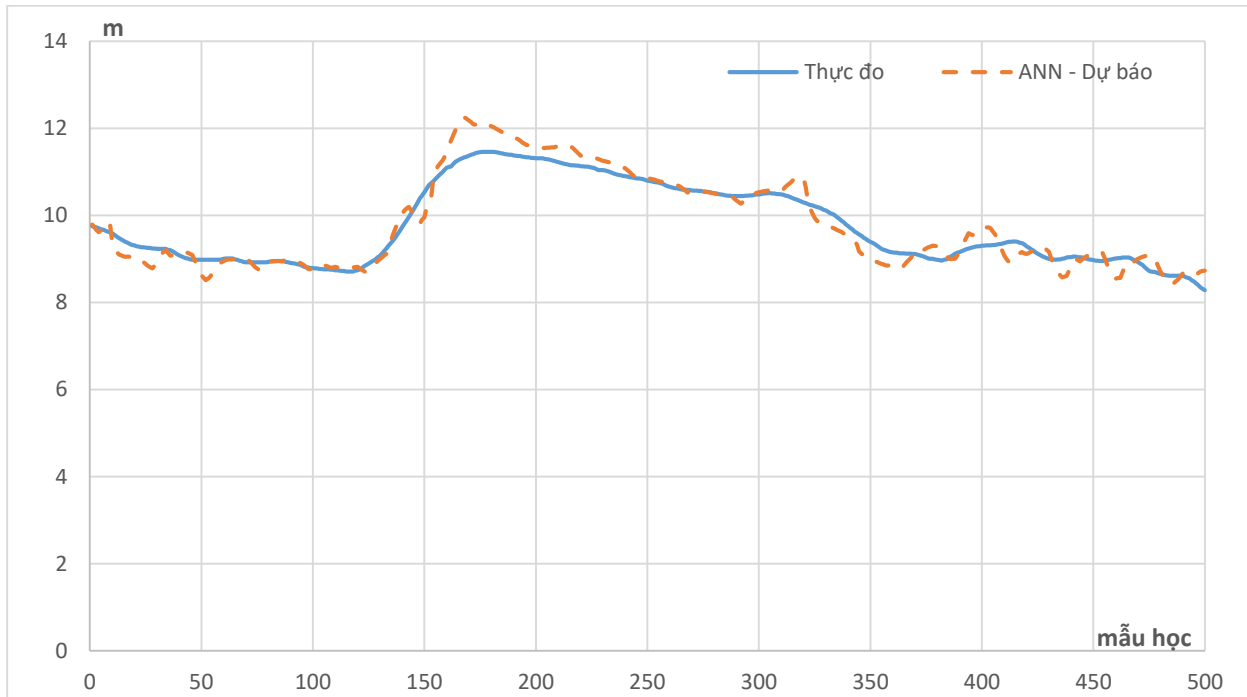
Bảng 4.1 Kết quả tính phương án 1

Giải thuật	RMS	NSE Hiệu chỉnh	NSE dự báo
GA	0.0164	0.918	0.775
GA+BP	0.0095	0.933	0.826

Phương án 2: Ngoài dữ liệu các năm 2000, 2002, 2003, lấy thêm nửa tập dữ liệu năm 2004 làm thành tập huấn luyện, dữ liệu còn lại của năm 2004 dùng để làm tập kiểm định “dự báo lại”.



Hình 4.4 Kết quả so sánh mực nước Sơn Tây năm 2000,2002,2003 và nửa năm 2004 giữa thực đo và mạng ANN hiệu chỉnh.



Hình 4.5 Kết quả so sánh mực nước Sơn Tây nửa sau năm 2004 giữa thực đo và mạng ANN kiểm định sau khi học bổ sung.

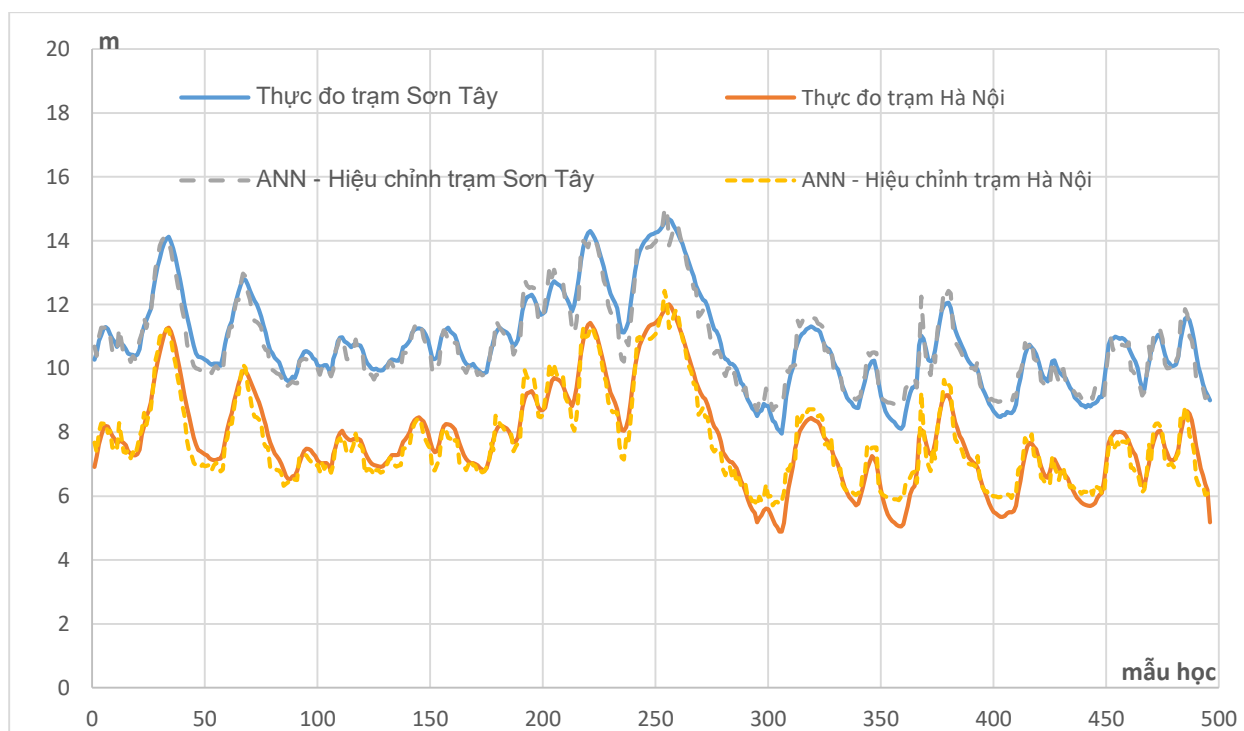
Các kết quả phương án tính được trình bày trong bảng 4.2

Bảng 4.2 Kết quả tính phương án 2

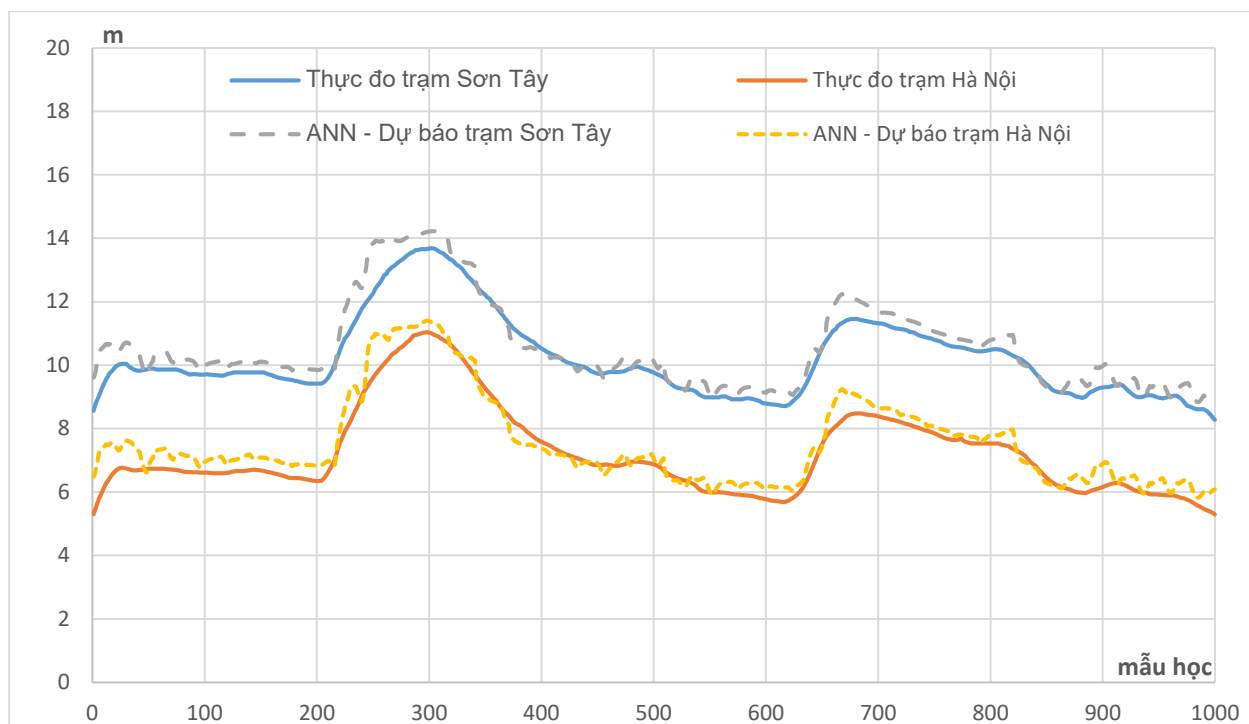
Giải thuật	RMS	NSE Hiệu chỉnh	NSE dự báo
GA	0.0173	0.905	0.795
GA+BP	0.0106	0.922	0.909

Nhận xét: Khi cập nhật thêm dữ liệu cho tập huấn luyện để “học thêm”, mạng ANN đã cho kết quả tốt hơn. Dữ liệu học thêm khá gần với dữ liệu dự báo nên phản ánh tốt hơn quan hệ giữa đầu vào và đầu ra của mạng, bổ sung các yếu tố có thể đã thay đổi (lòng dẫn, nhám, cấu trúc mạng sông,... của hệ thống sông năm 2004) mà các năm trước chưa có. Tuy nhiên vẫn có sự sai khác tại các đỉnh lũ, điều này lý giải bởi dạng lũ các năm là khác nhau, còn các yếu tố ảnh hưởng khác chưa được sử dụng làm tín hiệu đầu vào,... Do đó cần tiếp tục thêm vào mô hình các dữ liệu đầu vào khác cũng như tiếp tục cho chương trình học các dạng lũ khác nhau.

Phương án 3: Đầu vào của phương án này vẫn lấy như phương án 1, tuy nhiên kết quả đầu ra không chỉ là 1 giá trị mực nước ở trạm Sơn Tây mà còn thêm trạm Hà Nội (số neuron lớp ra là 2).



Hình 4.6 Kết quả so sánh mực nước Sơn Tây và Hà Nội năm 2000,2002,2003 giữa thực đo và mạng ANN hiệu chỉnh.



Hình 4.7 Kết quả so sánh mực nước Sơn Tây và Hà Nội năm 2004 giữa thực đo và mạng ANN kiểm định

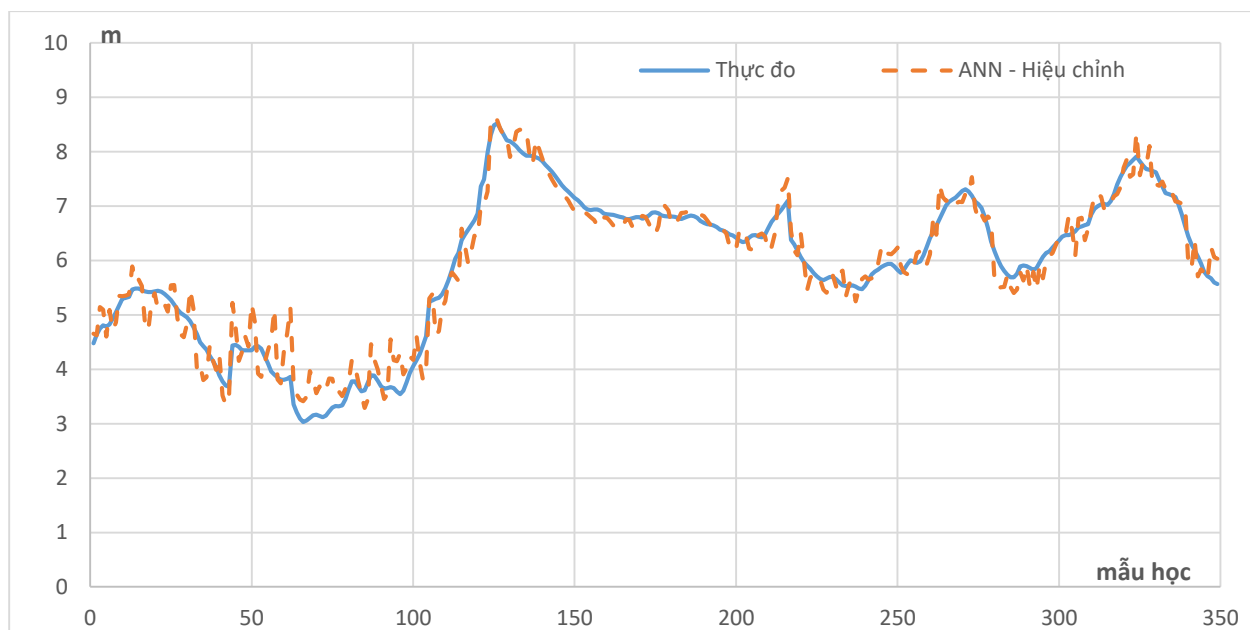
Các kết quả phương án tính 3 được trình bày trong bảng 4.3

Bảng 4.3 Kết quả tính phương án 3

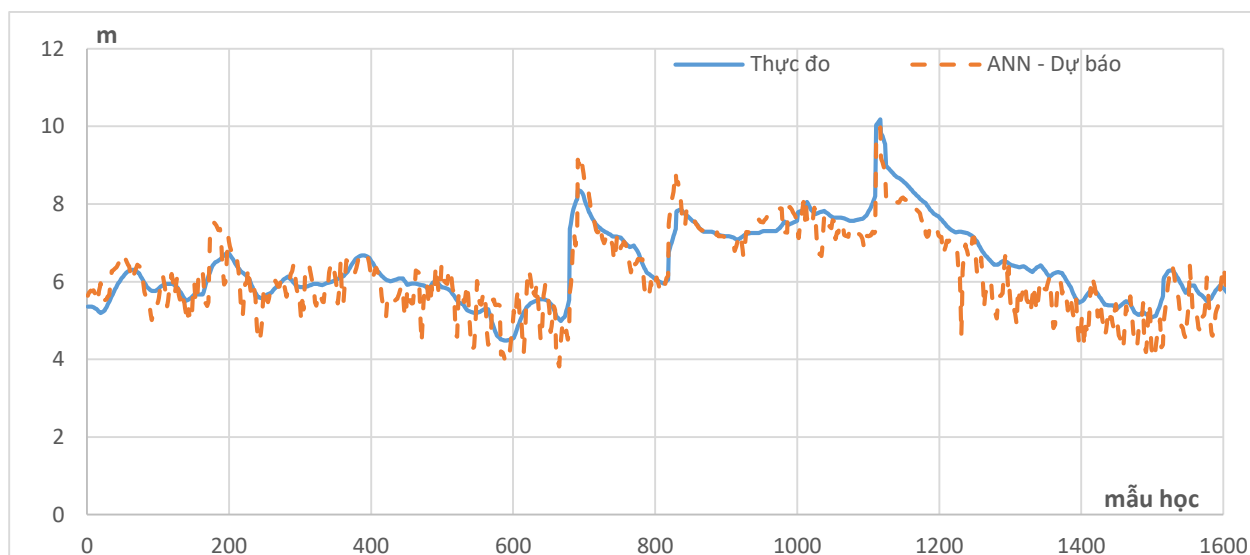
Giải thuật	RMS	NSE Hiệu chỉnh trạm Sơn Tây	NSE dự báo trạm Sơn Tây	NSE Hiệu chỉnh trạm Hà Nội	NSE dự báo trạm Hà Nội
GA	0.0245	0.855	0.691	0.855	0.790
GA+BP	0.0107	0.942	0.864	0.928	0.892

Nhận xét: Qua phương án 3 thử nghiệm cho thấy, mạng ANN hoàn toàn có thể huấn luyện để cùng cho kết quả tốt với nhiều nơon lớp ra. Nhu cầu thực tế của bài toán dự báo đặt ra là cùng một lúc phải đưa ra được nhiều điểm trong mạng cũng như bức tranh đơn giản toàn bộ hệ thống tuy không đầy đủ như các mô hình số tính thủy văn, thủy lực.

Phương án 4: Lấy dữ liệu 4 nguồn tại Hòa Bình, Yên Bái, Hàm Yên, Na Hang năm 2015 làm thành tập huấn luyện, dữ liệu năm 2016 dùng để làm tập kiểm định “dự báo lại”.



Hình 4.8 Kết quả so sánh mực nước Sơn Tây năm 2015 giữa thực đo và mạng ANN hiệu chỉnh.



Hình 4.9 Kết quả so sánh mực nước Sơn Tây năm 2016 giữa thực đo và mạng ANN kiểm định

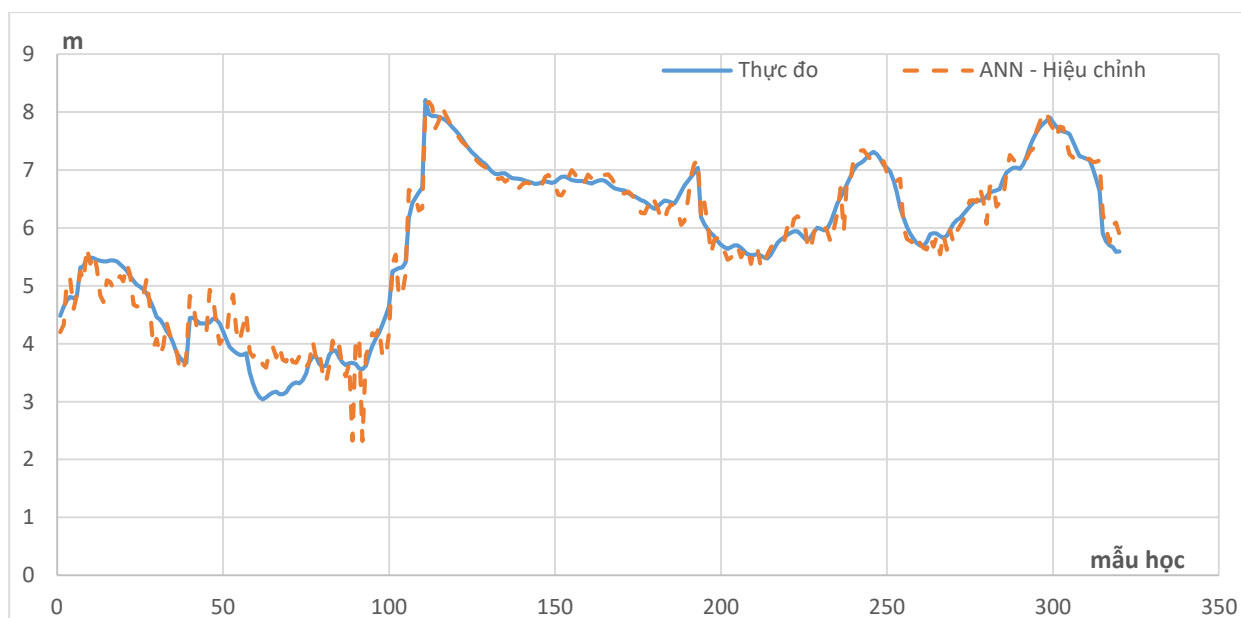
Các kết quả phương án 4 tính được trình bày trong bảng 4.4

Bảng 4.4 Kết quả tính phương án 4

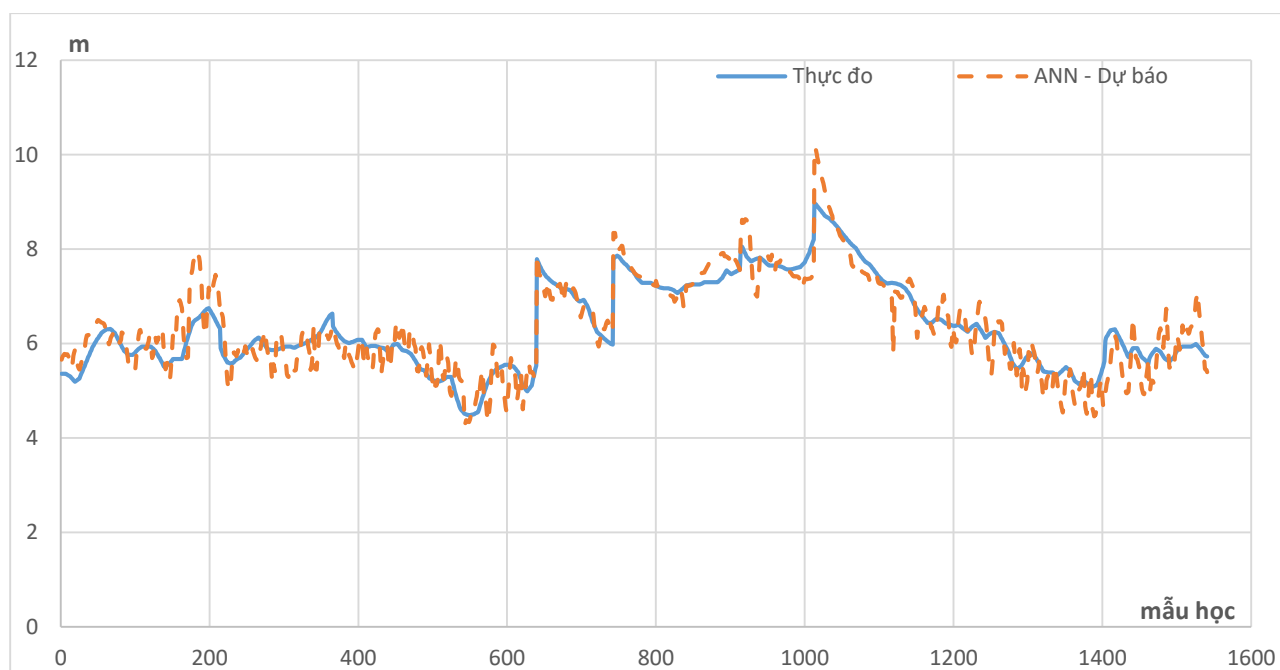
Giải thuật	RMS	NSE Hiệu chỉnh	NSE dự báo
GA	0.0266	0.903	0.535
GA+BP	0.0177	0.946	0.709

Nhận xét: Năm 2015, mực nước Sơn Tây dưới 600cm là không nhiều, do đó khi mạng ANN học bộ dữ liệu này, mảng trọng số chưa thể hiện đúng quan hệ đầu vào – đầu ra khi mực nước thấp. Sang đến năm 2016, mực nước Sơn Tây chủ yếu lại nằm dưới mức 600cm, tuy nhiên, các kết quả “dự báo lại” vẫn nằm ở mức chấp nhận được ($NSE > 0.7$ – đạt loại khá). Thêm một lần nữa chứng tỏ, dù còn thiếu các yếu tố ảnh hưởng khác cũng như chưa học được các dữ liệu phù hợp nhưng mạng ANN có thể phản ánh được khá tốt quan hệ đầu vào – đầu ra của mô hình mạng thủy lực.

Phương án 5: Các số liệu đầu vào và ra tương tự như phương án 4 nhưng bổ sung thêm tín hiệu đầu vào trạm Hưng Yên ở hạ du. Theo nguyên tắc, việc bổ sung thêm trạm hạ du này sẽ làm tăng tính chính xác của mô hình. Do đó, chúng tôi lựa chọn phương án 5 để đánh giá việc bổ sung tín hiệu này.



Hình 4.10 Kết quả so sánh mực nước Sơn Tây năm 2015 giữa thực đo và mạng ANN hiệu chỉnh có bổ sung số liệu trạm Hưng Yên.



Hình 4.11 Kết quả so sánh mực nước Sơn Tây năm 2016 giữa thực đo và mạng ANN kiểm định có bổ sung dữ liệu trạm Hưng Yên

Các kết quả phương án 5 tính được trình bày trong bảng 4.5

Bảng 4.5 Kết quả tính phương án 5

Giải thuật	RMS	NSE Hiệu chỉnh	NSE dự báo
GA	0.0238	0.927	0.684
GA+BP	0.0154	0.953	0.821

Nhận xét: Đúng như nguyên tắc của mạng học máy, khi ta bổ sung thêm dữ liệu (tín hiệu đầu vào) có ảnh hưởng, mạng đã phản hồi với kết quả tốt hơn (các chỉ số NSE của các phép thử đều tốt hơn rõ rệt). Đây cũng chỉ ra một lợi thế của mô hình học máy: khi còn thiếu dữ liệu, hoặc dữ liệu không gian không đầy đủ, mạng ANN vẫn cho ra được kết quả dự báo dù độ chính xác chưa bằng được khi có đầy đủ dữ liệu vào nhưng cũng kết quả đạt sai số chấp nhận được.

4.2 Kết quả mô phỏng và dự báo thủy văn lưu lượng vào hồ

4.2.1 Dự báo lưu lượng về hồ Hòa Bình

Bài toán dự báo lưu lượng dòng chảy đến hồ Hoà Bình phục vụ tốt mục tiêu điều tiết vận hành hồ. Nguồn nước chính cung cấp cho hồ Hoà Bình là nguồn nước từ sông Đà. Dọc theo hệ thống sông Đà có các trạm đo quan trắc khí tượng và thủy văn bắt đầu hoạt động từ năm 1902. Trạm đo thủy văn gần hồ Hoà Bình nhất là trạm đo Tạ Bú. Dự báo

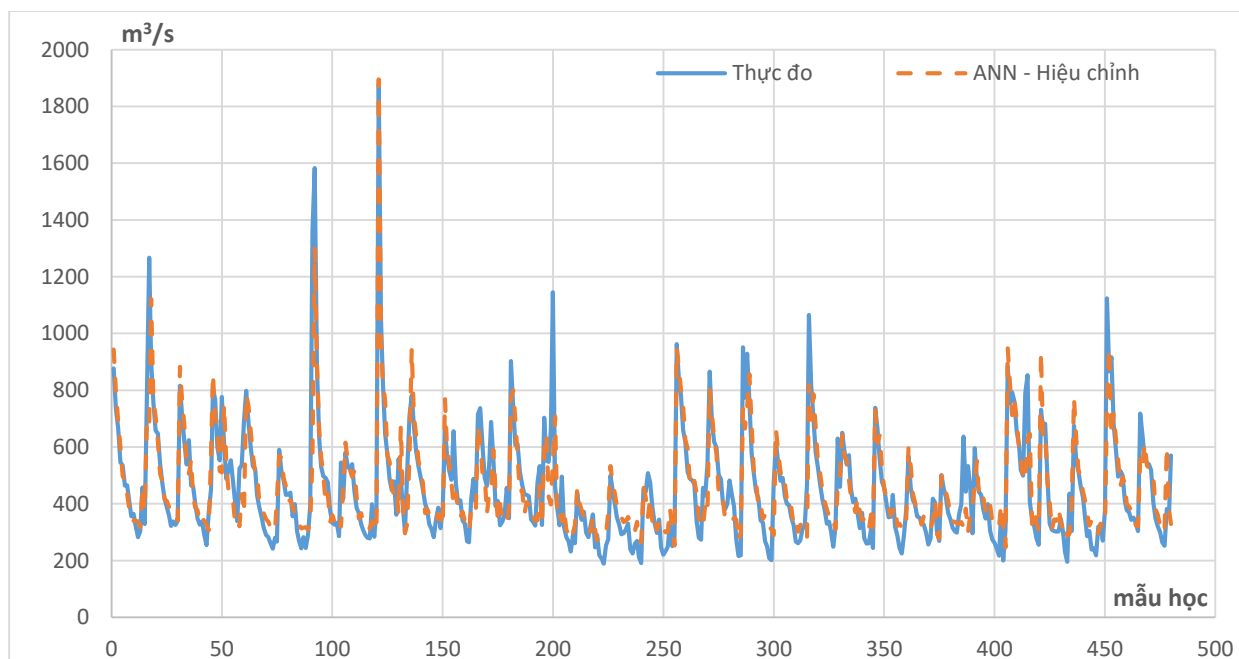
nước đến hồ Hoà Bình thực chất là dự báo lưu lượng nước tại trạm Tạ Bú. Tuy hiện tại trên lưu vực sông Đà đã có thêm 2 nhà máy điện lớn chính là Lai Châu và Sơn La, tuy nhiên điểm dự báo Tạ Bú không bị ảnh hưởng khi tính theo phương pháp này.

Chúng tôi sử dụng số liệu trong [8] bao gồm các số liệu từ năm 1964 đến năm 2002 tại trạm đo Tạ Bú trên sông Đà, trạm đo lưu lượng gần hồ Hoà Bình nhất. Số liệu này được đo trong mùa cạn từ tháng 12 năm trước đến tháng 5 năm sau. Các giá trị được lấy làm đầu vào cho mô hình như sau:

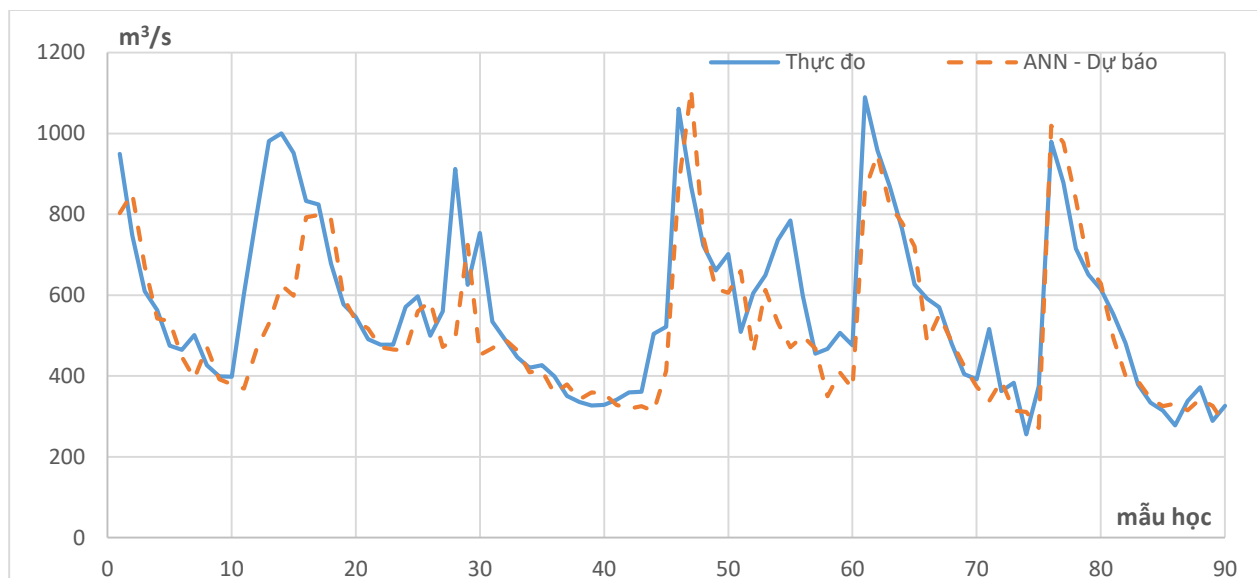
- Lưu lượng nước trung bình hiện tại: $Q(t)$
- Lưu lượng nước trước đó 10 ngày: $Q(t-10)$
- Lưu lượng nước trước đó 20 ngày: $Q(t-20)$
- Trung bình lượng mưa hiện tại: $X(t)$
- Trung bình lượng mưa trước đó 10 ngày: $X(t-10)$
- Trung bình lượng mưa trước đó 20 ngày: $X(t-20)$
- Lưu lượng nước của ngày hiện tại: $Q_{ng}(t)$
- Lượng mưa của ngày hiện tại: $X_{ng}(t)$

Bộ dữ liệu được chia làm hai phần:

- Phần dữ liệu học (training set): Từ cuối năm 1964 đến đầu năm 1995 có tổng cộng 480 mẫu học.
- Phần dữ liệu kiểm tra (test set): Từ cuối năm 1995 đến đầu năm 2002 tổng cộng có 90 mẫu kiểm tra.



Hình 4.12 Kết quả so sánh lưu lượng vào hồ Hòa Bình giữa thực đo và mạng ANN tính kiểm định



Hình 4.13 Kết quả so sánh lưu lượng vào hồ Hòa Bình giữa thực đo và mạng ANN tính dự báo

Các kết quả phương án tính được trình bày trong bảng 4.6

Bảng 4.6 Kết quả tính phương án dự báo lưu lượng vào hồ Hòa Bình

Giải thuật	RMS	NSE Hiệu chỉnh	NSE dự báo
GA	0.0285	0.726	0.521
GA+BP	0.0215	0.758	0.579

Nhận xét: Với tập dữ liệu thu được trải dài từ năm 1964 đến 2002, số lượng mẫu học không nhiều, các giá trị đầu vào cách nhau khá rời rạc, mạng ANN cũng đã tạo được quan hệ đầu vào – đầu ra một cách tương đối thể hiện ở kết quả tính (NSE hiệu chỉnh chỉ đạt 0.75, NSE dự báo chỉ đạt 0.58).

4.3 Kết quả mô phỏng và dự báo độ mặn tại vùng Tứ Giác Long Xuyên

Vùng ĐBSCL có 3 khu vực nhiễm mặn đáng chú ý, đó là: vùng mặn sông Vàm Cỏ, vùng Bán đảo Cà Mau, vùng ven biển phía Tây của Tứ Giác Long Xuyên. Tứ Giác Long Xuyên là vùng nằm dọc theo kênh Rạch Giá - Hà Tiên, bị ảnh hưởng trực tiếp của nước mặn phía biển Tây. Vùng này có các kênh tiếp nước đều xuất phát từ miền nước ngọt của sông Hậu, độ mặn ở đây được quyết định chủ yếu bởi khả năng tải nước của các kênh dẫn và lượng nước đã dùng trên dọc các tuyến kênh đó. Tuy nhiên các trạm đo mặn trong vùng rất ít và số liệu cũng không đồng bộ, đầy đủ.



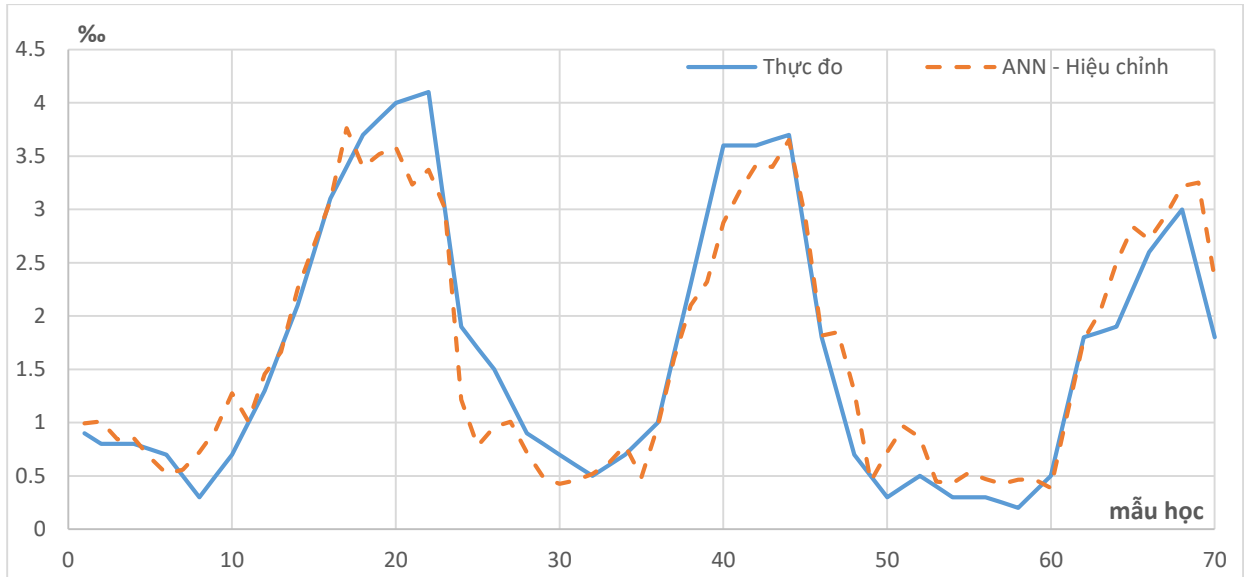
Hình 4.14 Mô hình tính tại vùng Tứ Giác Long Xuyên

Mô hình của bài toán là tính giá trị mặn tại Rạch Giá qua các giá trị tín hiệu đầu vào tại các điểm như Châu Đốc, Vàm Nao, Cần Thơ, từ dao động mực nước biển xa (tính theo các hằng số điều hòa) cũng như từ giá trị mực nước tại chính Rạch Giá.

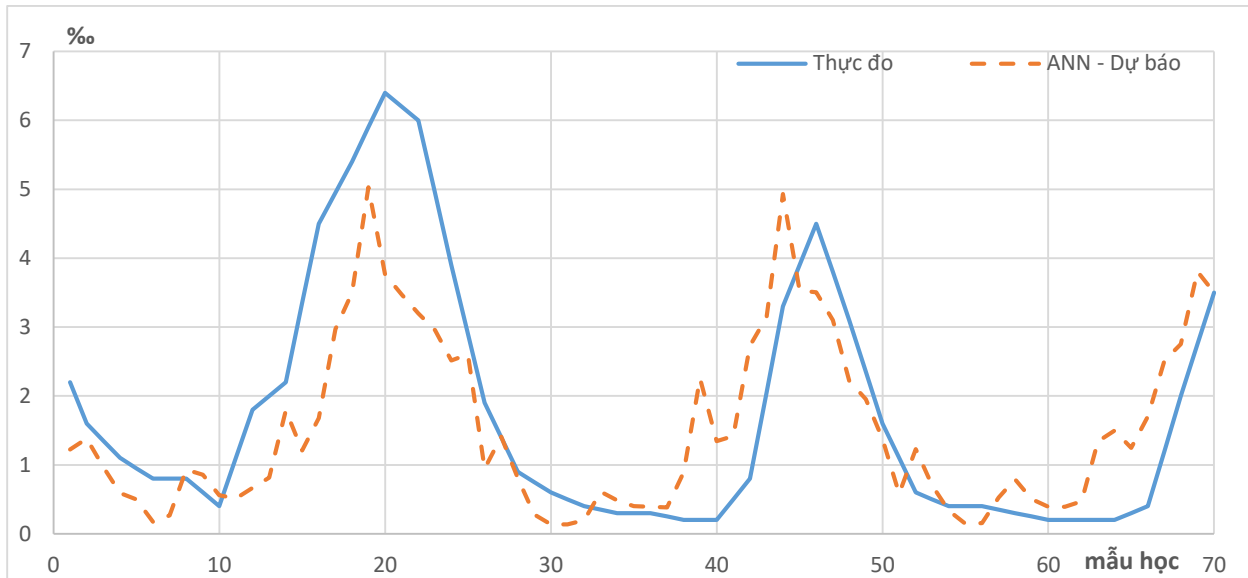
Trong đề tài này, sử dụng các kịch bản tính theo Bảng 4.7 mà tác giả đã thực hiện trong luận án tiến sỹ để xây dựng bộ trọng số mạng nơron nhân tạo, từ đó có thể dự báo các giá trị mặn khi chuỗi số liệu thực đo không có hoặc không đầy đủ nhằm làm đầu vào tính toán cho các bài toán khác.

Bảng 4.7 Kịch bản và kết quả độ mặn tính toán của mô hình thủy lực

Năm	Mùa	Triều	Trạm	Phương án tính toán	NSE
2008	Mùa khô	Triều cường	Rạch Giá	Hiệu chỉnh	0.96
2011	Mùa khô	Triều cường	Rạch Giá	Kiểm định	0.61



Hình 4.15 Kết quả so sánh độ mặn Rạch Giá giữa thực đo và mạng ANN hiệu chỉnh



Hình 4.16 Kết quả so sánh độ mặn Rạch Giá giữa thực đo và mạng ANN dự báo

Các kết quả phương án tính mặn được trình bày trong bảng 4.8.

Bảng 4.8 Kết quả tính phương án tính mặn

Giải thuật	RMS	NSE Hiệu chỉnh	NSE dự báo
GA+BP	0.0426	0.9094	0.6159

Nhận xét: với đầu vào ít, tuy nhiên các kết quả tính toán của mô hình mạng ANN đưa ra đạt giá trị tốt so với kết quả tính toán của mô hình thủy lực đòi hỏi nhiều dữ liệu như các dữ liệu mặt cắt, dữ liệu địa hình đáy biển,... Kết quả tính cũng đạt giá trị chấp nhận được (chỉ số NSE > 0.61) cho dù các yếu tố dữ liệu ảnh hưởng còn chưa được đánh giá hết (do không có dữ liệu).

KẾT LUẬN VÀ KIẾN NGHỊ

Đề tài định hướng nội dung nghiên cứu vào mạng nơron nhân tạo, thuật toán di truyền, mô hình kết hợp giữa phương pháp học máy mạng nơron với thuật toán di truyền, ứng dụng vào bài toán dự báo thủy lực, thủy văn, môi trường,...

Các đóng góp khoa học của đề tài

- Hệ thống hóa các nội dung cơ bản về mạng nơron nhân tạo và thuật toán di truyền.

- Nghiên cứu các phương pháp kết hợp giải thuật Di truyền với giải thuật Lan truyền ngược sai số nhằm đạt tới một kết quả trọn vẹn của bài toán tối ưu trọng số mạng nơron nhân tạo.

- Xây dựng chương trình dự báo theo phương pháp kết hợp giải thuật Di truyền và giải thuật Lan truyền ngược sai số kết hợp với các kỹ thuật song song OpenMP trên đa lõi CPU đạt hiệu năng tăng gấp 4 lần trên chip Intel Core I7 3770 4 lõi 8 luồng, song song CUDA trên đa lõi GPU đạt hiệu năng tăng gấp 18 lần trên card Nvidia 1060 so với giải tuần tự trên 1 lõi tính thông thường, áp dụng phép biến đổi Wavelet để loại bỏ nhiễu và tạo dữ liệu đầu vào cho mạng ANN. Ngoài ra, chương trình có khả năng tùy chọn số lớp ẩn cũng như tùy chọn loại hàm kích hoạt (20 loại khác nhau) để phục vụ các bài toán khác nhau.

- Kết quả mô phỏng và dự báo thủy lực các trạm Sơn Tây, Hà Nội trên hệ thống lưu vực sông Hồng, các kết quả thu được của việc hiệu chỉnh và kiểm định đạt tốt ($NSE > 0.75$).

- Kết quả mô phỏng và dự báo thủy văn lưu lượng đến hồ đạt được kết quả khá. Do các yếu tố đầu vào (mưa phân bố) mang tính ngẫu nhiên, chưa đại diện hết cho các yếu tố ảnh hưởng, tuy nhiên kết quả thu được từ việc mô phỏng và dự báo cũng đạt mức trung bình đến tốt ($NSE \sim 0.6$).

- Dự báo giá trị mặn tại vùng Tứ Giác Long Xuyên: Kết quả mạng ANN mô phỏng được rất tốt, tuy nhiên các giá trị dự báo còn ở mức trung bình chấp nhận được, tương đương với kết quả tính bằng mô hình thủy lực – lan truyền chất trên mạng sông và biển (1-2D).

- Các phương án mô phỏng và dự báo được chạy nhiều kịch bản khác nhau để qua đó đánh giá được ưu nhược điểm của mạng ANN đã xây dựng.

Hướng phát triển

- Thử nghiệm ứng dụng công tác dự báo lũ lụt hàng năm, thử nghiệm phát dự báo trực tuyến thời gian thực qua website tận dụng việc tính kết quả dự báo nhanh chóng của mạng ANN.
- Tối ưu mã nguồn song song GPU để sử dụng tối đa công suất của card đồ họa.
- Kết hợp với thư viện phân tích ảnh OpenCV để ứng dụng dự báo mưa từ ảnh vệ tinh.
- Ứng dụng vào các bài toán khác trong thực tiễn: nhận dạng chữ viết tay, nhận dạng chuyển động, nhận dạng đồ vật, nhận dạng biển số xe, nhận dạng và giả lập giọng nói qua việc học các mẫu âm thanh,...
- Xây dựng một hệ thống siêu máy tính mini phục vụ công việc huấn luyện mạng ANN cũng như các công việc khác của phòng chuyên môn.
- Để ứng dụng được nhiều hơn nữa các bài toán dự báo thủy lực, thủy văn ở các phân lớp khác, ví dụ như các giá trị đầu ra $f(X)$ có gián đoạn, không liên tục, cần cải tiến mô hình mạng ANN cũng như kết hợp các phương pháp học khác.

TÀI LIỆU THAM KHẢO

1. Blum, E. K. and L. K. Li. *Approximation Theory and feedforward networks*, Neural Networks, 1991, Vol. 4, pp. 511-515.
2. D.E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison Wesley, Reading, MA, (1989).
3. Hecht-Nielsen, R. 1989. *Theory of backpropagation neural network*, In Proceedings of the International Joint Conference on Neural Networks, Washington DC., (June 1989), IEEE TAB Neural Network Committee, pp. I593-P605.
4. Kolmogorov A. N. *On the representation of continuous functions of many variables by superposition of continuous function of one variable and addition*, Dokl, Akad, Nauk SSSR, 114, 953-956, Trans. Am. Math-Soc. 2(28), 55-59, (1957).
5. Lê Minh Trung. *Giáo trình mạng neuron nhân tạo*, Nhà xuất bản thống kê, (1999).
6. Martin T. Hagan, *Neural Network Design*, PWS Publishing Company, (1996).
7. Oscar R. Dolling, Eduardo A. Varas, *Artificial neural networks for stream flow prediction*, Journal of Hydraullic research, 40(5), 547-554, (2002).
8. Phạm Thị Hoàng Nhung, *Nghiên cứu ứng dụng các phương pháp học máy tiên tiến trong công tác dự báo, vận hành hồ Hòa Bình*, Luận văn Thạc sỹ, (2007).