BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NÔNG LÂM THÀNH PHỐ HỒ CHÍ MINH

KHÓA LUẬN TỐT NGHIỆP

NGHIÊN CỨU ÁP DỤNG MẠNG NEURON NHÂN TẠO PHỤC VỤ BÀI TOÁN NHẬN DẠNG TRONG GIS

Họ và tên sinh viên: NGUYỄN NGỌC MINH TIẾN

Ngành: Hệ thống Thông tin Địa lý

Niên khóa: 2012 - 2016

Tp. Hồ Chí Minh, tháng 06 / 2016

NGHIÊN CỨU ÁP DỤNG MẠNG NEURON NHÂN TẠO PHỤC VỤ BÀI TOÁN NHẬN DẠNG TRONG GIS

Tác giả

NGUYỄN NGỌC MINH TIẾN

Khóa luận được đệ trình để đáp ứng yêu cầu
cấp bằng Kỹ sư ngành
Hệ thống Thông tin Địa lý

Giáo viên hướng dẫn

Th.S Khưu Minh Cảnh

Tp.Hồ Chí Minh, tháng 06 / 2016

LỜI CẢM ƠN

Trước hết, tôi xin chân thành cảm ơn thầy Th.S Khưu Minh Cảnh, cán bộ công tác tại Sở Khoa học và Công nghệ thành phố Hồ Chí Minh, người đã hướng dẫn tôi hoàn thành đề tài tốt nghiệp này. Cảm ơn thầy đã tận tình chỉ bảo, hỗ trợ và động viên tôi trong suốt thời gian qua. Đồng thời tôi cũng xin gửi lời cảm ơn đến thầy ThS. NCS. Cao Duy Trường đã góp ý, chia sẻ thêm kinh nghiệm về bài luận.

Tôi cũng xin chân trọng cảm ơn Ban lãnh đạo Sở Khoa học và Công nghệ thành phố Hồ Chí Minh đã tạo điều kiện cho tôi được thực tập, làm việc tại quý cơ quan. Đặc biệt, tôi xin gửi lời cảm ơn đến phòng Trung tâm Ứng dụng Hệ thống Thông tin Địa lý TP.HCM (HCMGIS) đã tận tình trao đổi các kiến thức, kinh nghiệm quý báu cũng như chia sẻ tài liệu, dữ liệu.

Tôi xin gửi lời tri ân sâu sắc đến thầy PGS.TS Nguyễn Kim Lợi, thầy Th.S Lê Văn Phận, cô Th.S Nguyễn Thị Huyền, thầy Ks Nguyễn Duy Liêm, thầy Ks Lê Hoàng Tú, các anh chị phòng Trung tâm nghiên cứu khí hậu RICC, quý thầy cô trường đại học Nông Lâm thành phố Hồ Chí Minh cùng với tập thể lớp DH12GI. Cảm ơn quý thầy cô, quý anh chị và các bạn về những kiến thức, kinh nghiệm và sự giúp đỡ chân tình đã dành cho tôi trong suốt bốn năm học tập tại trường.

Cuối cùng, con xin nói lời biết ơn sâu sắc nhất đến với cha mẹ, những người đã chăm sóc, nuôi dạy con thành người và luôn động viên tinh thần, hỗ trợ mọi thứ cho con để con yên tâm học tập.

Nguyễn Ngọc Minh Tiến
Chuyên ngành Hệ thống Thông tin Địa lý
Khoa Môi trường & Tài nguyên
Trường đại học Nông Lâm Tp. Hồ Chí Minh
Tp. Hồ Chí Minh, Tháng 06 / 2016

TÓM TẮT

Khóa luận tốt nghiệp "Nghiên cứu áp dụng mạng neuron nhân tạo phục vụ bài toán nhận dạng trong GIS" đã được thực hiện trong khoảng thời gian từ ngày 01/03/2016 đến ngày 07/06/2016. Phương pháp tiếp cận của đề tài là kết hợp công nghệ GIS với mạng Neuron nhân tạo (ANN) tập trung về mạng lan truyền ngược (BP), một mảng của trí thông minh nhân tạo (AI). Theo đó GIS với khả năng hỗ trợ mạnh mẽ trong việc quản lý và tương tác tốt đối với cả hai loại dữ liệu thuộc tính và dữ liệu không gian cùng với sự thay đổi về thời gian trong khi đó mạng Neuron có tốc độ xử lý rất nhanh, có khả năng học hỏi, cho phép học những gì mà ta yêu cầu và lợi thế lớn nhất của ANN là khả năng được sử dụng như một cơ chế xấp xỉ hàm tùy ý mà 'học' được từ các dữ liệu quan sát.

Việc kết hợp thế mạnh của GIS và mạng Neuron nói riêng cũng như trí thông minh nhân tạo nói chung mang đến một giải pháp mới để giải quyết các vấn đề lớn, mang nhiều đặc điểm khác nhau với tính cấp bách điển hình là các vấn đề liên quan đến tai nạn giao thông.

Luận văn đã đề cập đến các nội dung sau:

- Tìm hiểu, xây dựng dữ liệu tai nạn giao thông tại thành phố Hồ Chí Minh.
- Tìm hiểu và nắm được quy trình xây dựng mạng neuron để phân tích khai phá dữ liệu không gian (data mining).
- Thực hiện thử nghiệm phân tích mạng thần kinh nhân tạo để nhận dạng bộ dữ liệu tai nạn giao thông đã xây dựng.
- Tìm hiểu lập trình về ngôn ngữ Python.
- Tìm hiểu, sử dụng công cụ MATLAB.

Kết quả đạt được của luận văn gồm:

- Xây dựng cơ sở dữ liệu không gian về các vụ tai nạn giao thông tại thành phố Hồ Chí Minh.
- Xây dựng được bản đồ các vụ tai nạn giao thông tại TPHCM.

- Tiếp cận được phương pháp phân tích mạng neuron nhân tạo.
- Nắm bắt được cấu hình mạng của neuron dựa trên dữ liệu tai nạn giao thông tại TPHCM.

MỤC LỤC

DANH MỤC VIẾT TẮT

AI	Artificial Intelligence (Trí thông minh nhân tạo)
ANN	Artificial Neural Network (Mạng Nơ-ron nhân tạo)
BP	Back Propagation (Mạng lan truyền ngược)
CSDL	Cơ sở dữ liệu
DD	Decimal Degree (Phép chiếu tọa độ theo dạng độ thập phân)
ESRI	Environmental Systems Research Institute (Viện nghiên cứu hệ thống môi trường)
GIS	Geographic Information System (Hệ thống Thông tin Địa lý)
MATLAB	Matrix Laboratory (Phần mềm tính toán Neural)
OSM	OpenStreetMap (Bản đồ đường sá mở)
SQL	Structured Query Language (Ngôn ngữ truy vấn mang tính cấu trúc)
TNGT	Tai nạn giao thông
TPHCM	Thành phố Hồ Chí Minh
UTM	Universal Trasverse Mercator (Phép chiếu tọa độ theo dạng mét)
WHO	World Health Organization (Tổ chức y tế thế giới)

DANH MỤC BẢNG BIỂU

DANH MỤC HÌNH ẢNH

CHƯƠNG 1
ĐẶT VẤN ĐỀ

1.1 Tính cấp thiết của đề tài

Sự phát triển của giao thông đường bộ là một biểu hiện của sự tiến bộ của nhân loại, nhưng một trong những mặt trái của nó là tình trạng mất an toàn và tai nạn giao thông (TNGT). Hiện nay tình trạng giao thông ngày càng gia tăng do số lương phương tiện cá nhân tăng theo nhu cầu của con người. Theo báo cáo của Tổ chức Y tế Thế giới (WHO) công bố năm 2013, mỗi năm, trên thế giới có khoảng 1,25 triệu người chết vì tai nạn giao thông, trung bình mỗi ngày khoảng 3.400 người chết vì TNGT trên đường bộ.

Tại Việt Nam, tình hình tai nạn giao thông cũng vô cùng nghiêm trọng. Phó thủ tướng Nguyễn Xuân Phúc khi nói về tình trạng tai nạn giao thông phải thốt lên rằng, nhiều nước đang xảy ra chiến tranh trên thế giới cũng không có nhiều người chết như ở nước ta. Số liệu từ Ủy ban An toàn giao thông quốc gia cho thấy trong năm 2014, cả nước xảy ra 25.322 vụ tai nạn, làm chết 8.996 người, bị thương 24.417 người. Riêng tại TPHCM đã xảy ra 2.587 vụ TNGT từ ít nghiêm trọng trở lên, làm chết 2.435 người và làm 1.186 người bị thương. Trung bình mỗi năm trên địa bàn thành phố xảy ra 952 vụ, làm 811 người chết và làm 395 người bị thương.

Trong quá trình nghiên cứu đặc điểm phân bố TNGT trên phạm vi toàn cầu dựa trên tiêu chí mức thu nhập, các nhà khoa học đã thấy rằng các nước có mức thu nhập trung bình có tỉ lệ người chết do TNGT cao nhất (20,1 người chết/100.000 dân), thấp nhất là các nước có mức thu nhập cao (8,7 người chết/100.000 dân), trong khi tỉ lệ người chết do TNGT trung bình trên thế giới hiện là 18 người chết/100.000 dân. Trên thực tế các nước có mức thu nhập trung bình có số lượng phương tiện giao thông chiếm 52% tổng số toàn cầu, còn các nước có mức thu nhập cao chiếm 47%, như vậy số lượng gần ngang nhau. Điều đó cho thấy, số người chết do TNGT không tỉ lệ với số lượng phương tiện mà do các nguyên nhân quan trọng khác như: trình độ dân trí, hiểu biết luật giao thông, ý

thức tham gia giao thông và kết cấu hạ tầng giao thông, độ tuổi, nghề nghiệp, loại phương tiện.

Chính vì vậy luận văn với đề tài *"Nghiên cứu áp dụng mạng neuron nhân tạo phục vụ bài toán nhận dạng trong GIS"* đã đi sâu nghiên cứu và áp dụng mạng neural nhân tạo cùng với công nghệ GIS nhằm xây dựng hệ thống giúp các nhà quản lý nhận dạng được các vụ TNGT tại thành phố Hồ Chí Minh (TPHCM) một cách nhanh chóng, hiệu quả mà chi phí thấp để từ đó các nhà quản lý có thể nắm bắt và đề ra phương hướng cũng như kế hoạch giảm thiểu, ứng phó với tình hình TNGT hiện nay. Đồng thời ứng dụng của đề tài có thể giúp người dân nhận thấy mối liên hệ giữa các nguyên nhân xảy ra TNGT.

1.2 Mục tiêu của đề tài

- **Mục tiêu chung**

  Sử dụng mạng neuron nhân tạo để nhận dạng tai nạn giao thông tại TPHCM.

- **Mục tiêu cụ thể**

  - Phân tích đặc điểm không gian các vụ TNGT.
  - Đánh giá sai số sau khi chạy mạng neural.
  - Tìm cấu hình mạng phù hợp với bộ dữ liệu TNGT tại TPHCM.

1.3 Kết quả mong đợi

- Xây dựng được dữ liệu không gian và thuộc tính các vụ TNGT tại TPHCM.
- Đánh giá được khả năng nhận diện dữ liệu các vụ TNGT tại TPCHM bằng mạng neuron nhân tạo.

1.4 Đối tượng và phạm vi nghiên cứu

Thời gian thực hiện đề tài là 3 tháng từ ngày 01/03/2016 đến ngày 07/06/2016

Đối tượng nghiên cứu của đề tài là các vụ TNGT xảy ra tại TPHCM.

Phạm vi nghiên cứu của đề tài được giới hạn tại TPHCM.

1.5 Ý nghĩa khoa học và thực tiễn

1.5.1 Ý nghĩa khoa học

Kết quả của đề tài cung cấp cơ sở để đánh giá việc áp dụng trí tuệ nhân tạo và khả năng khai phá thông tin vào dữ liệu TNGT tại TPHCM nói riêng cũng như dữ liệu TNGT

cả nước nói chung trong việc nâng cao khả năng đánh giá các vụ TNGT. Đồng thời xem xét khả năng phân tích nhận diện các dạng dữ liệu khác có tính tương đồng như: Nhận dạng dữ liệu các vụ trộm cắp, nhận dạng dữ liệu tình trạng kẹt xe, nhận dạng dữ liệu phòng cháy chữa cháy.

1.5.2 Ý nghĩa thực tiễn

Kết quả nghiên cứu của đề tài là tài liệu tham khảo hữu ích cho các cơ quan quản lý trong việc ứng dụng các công nghệ mới để nâng cao mức độ hiệu quả trong việc khai thác thông tin các vụ TNGT tại TPHCM.

CHƯƠNG 2

TỔNG QUAN ĐỀ TÀI

2.1 Khái quát khu vực nghiên cứu

2.1.1 Vị trí địa lý

Hình 2.1: Bản đồ ranh giới hành chính TPHCM

Thành phố Hồ Chí Minh nằm trong toạ độ địa lý khoảng 10^0 10’ – 10^0 38’ vĩ độ bắc và 106^0 22’ – 106^0 54’ kinh độ đông. Thành phố Hồ Chí Minh giáp 6 tỉnh gồm:

- Phía Bắc giáp tỉnh Bình Dương
- Tây Bắc giáp tỉnh Tây Ninh
- Đông và Đông Bắc giáp tỉnh Đồng Nai

- Đông Nam giáp tỉnh Bà Rịa -Vũng Tàu
- Tây và Tây Nam giáp tỉnh Long An và Tiền Giang.

Thành phố cách thủ đô Hà Nội gần 1.730km đường bộ, nằm ở ngã tư quốc tế giữa các con đường hàng hải từ Bắc xuống Nam, từ Đông sang Tây, là tâm điểm của khu vực Đông Nam Á. Tính theo đường chim bay thì trung tâm thành phố cách bờ biển Đông 50 km, sân bay quốc tế Tân Sơn Nhất cách trung tâm thành phố 7km, chiều dài của thành phố theo hướng Tây Bắc – Đông Nam khoảng 100 km và chiều ngang nơi rộng nhất là hơn 40 km.

Thành phố Hồ Chí Minh gồm có bốn điểm cực:

- Cực Bắc là xã Phú Mỹ Hưng, huyện Củ Chi.
- Cực Tây là xã Thái Mỹ, huyện Củ Chi.
- Cực Nam là xã Long Hòa, huyện Cần Giờ.
- Cực Đông là xã Thạnh An, huyện Cần Giờ.

2.1.2 Tình hình tai nạn giao thông tại TPHCM

Theo số liệu thống kê của Phòng Cảnh sát giao thông đường bộ, đường sắt Công an thành phố Hồ Chí Minh, trong ba năm (từ 2011 đến 2013) trên địa bàn thành phố đã xảy ra 2.587 vụ TNGT từ ít nghiêm trọng trở lên, làm chết 2.435 người và làm 1.186 người bị thương. Trung bình mỗi năm trên địa bàn thành phố xảy ra 952 vụ, làm 811 người chết và làm 395 người bị thương.

Trong 10 tháng đầu năm 2015 trên địa bàn TPHCM đã xảy ra 3.050 vụ tai nạn giao thông, làm bị thương 2.657 người và 596 người chết. nguyên nhân gây ra tai nạn giao thông trong 10 tháng đầu năm chủ yếu do lái xe cơ giới lưu thông không đúng phần đường gây ra 113 vụ, đối tượng gây ra tai nạn phần lớn là xe 2 bánh gắn máy với 480 vụ chiếm 75%.

Và tính đến quý 1 năm nay toàn thành phố xảy ra 823 vụ TNGT. Các vụ tai nạn này làm 199 người chết, tăng 15 người (tương đương mức tăng 8,15%), làm 664 người bị thương (giảm 21,33% so với cùng kỳ năm ngoái). Có 10 quận, huyện giảm được số người

chết vì TNGT; 11 quận, huyện tăng số người chết vì TNGT, trong đó đáng ngại nhất là ở địa bàn quận 5 và quận Tân Bình vì có tình hình TNGT tăng cao.

Theo kết quả phân tích cho thấy:

- Địa bàn xảy ra tai nạn: TNGT xảy ra trên tất cả các tuyến đường trong đó: Quốc lộ: chiếm 8% số vụ, 17% người chết và 13% số người bị thương; Tỉnh lộ: chiếm 3% số vụ, 5% người chết và 6% số người bị thương; Đường nội thành: chiếm 71% số vụ, 51% người chết và 59% số người bị thương; Đường ngoại thành: chiếm 10% số vụ, 19% người chết và 15% số người bị thương.
- Đối tượng gây tai nạn: Đối tượng gây TNGT rất đa dạng, trong đó Xe tải: chiếm 8,38% (tổng số vụ); Xe khách: chiếm 1,21%; Xe taxi: 0,52%; Xe buýt: chiếm 1%; Xe đầu kéo (container): chiếm 1,80%; Xe ba bánh, gắn máy: 0,42%; Xe hai bánh, gắn máy: 75,16%; Xe hai, ba bánh đạp điện: 0,94%; Bộ hành: 7,66%. Như vậy, đối tượng gây ra vụ TNGT đường bộ chủ yếu là xe hai bánh, gắn máy (75,16%), xe tải (8,38%), bộ hành (7,66%), xe du lịch (1,8%) và xe khách (1,52%).
- Thời gian xảy ra tai nạn: Kết quả phân tích cho thấy TNGT trên địa bàn thành phố xảy ra ở tất cả các giờ trong ngày nhưng nhiều nhất vào lúc 15h, 18h, 19h, 20h, 21h và 23h; 19h - 23h là thời gian TNGT nhiều nhất trong ngày, trong đó 18h - 21h mật độ TNGT xảy ra cao nhất; khung giờ xảy ra tai nạn thấp nhất là từ 3h đến 5h và từ 7h đến 9h. Thời điểm 6h sáng TNGT xảy ra nhiều trong khung giờ này.
- Độ tuổi, giới tính thương vong do TNGT: Kết quả phân tích cho thấy nam giới chết và bị thương do TNGT chiếm tỉ lệ cao, cụ thể: Chết do TNGT: nam giới chiếm 83%, nữ giới chiếm 17%; Bị thương do TNGT: nam giới chiếm 79%, nữ giới chiếm 21% .
- Độ tuổi thương vong chiếm tỉ lệ cao nhất lần lượt là: từ 25 - 30 tuổi (chết chiếm 17,12%, bị thương chiếm 15,04%), từ 31 - 40 tuổi (chết chiếm 16,23%, bị thương chiếm 42,30%), từ 19 - 24 tuổi (chết chiếm 15,06%, bị thương chiếm 15,38%), từ

41 - 50 tuổi (chết chiếm 13,36%, bị thương chiếm 10,25%), từ 51 - 60 tuổi (chết chiếm 10,29%, bị thương chiếm 6,2%) và trên 60 tuổi (chiếm 9,32%, bị thương chiếm 6,66%).

Điều này cho thấy, người bị thương vong do tai nạn giao thông chiếm tỉ lệ cao chủ yếu là thanh niên, người trong độ tuổi lao động và đủ độ tuổi cấp giấy phép lái xe mô tô và ô tô theo luật định.

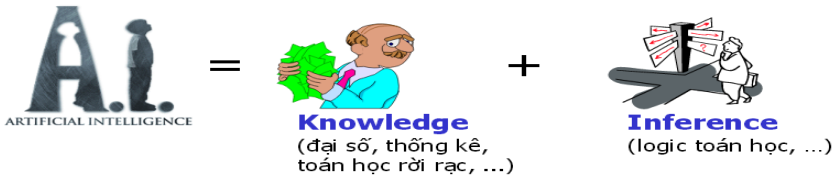
2.2 Trí tuệ nhân tạo

2.2.1 Định nghĩa về trí tuệ nhân tạo

Khái niệm trí tuệ nhân tạo (Artificial Intelligence- viết tắt là AI) được hiểu một cách đơn giản, trí tuệ nhân tạo là một lĩnh vực của khoa học và công nghệ nhằm làm cho máy có những khả năng trí tuệ của con người như: biết suy nghĩ và lập luận để giải quyết vấn đề, biết giao tiếp do hiểu ngôn ngữ và tiếng nói, biết học và tự thích nghi, …

Một vài định nghĩa khác nhau về trí tuệ nhân tạo:

- Bellman (1978) định nghĩa: Trí tuệ nhân tạo là tự động hoá các hoạt động phù hợp với suy nghĩ con người, chẳng hạn các hoạt động ra quyết định, giải bài toán..
- Rich và Knight (1991) cho rằng trí tuệ nhân tạo là lĩnh vực nghiên cứu để làm cho máy tính làm được những việc mà con người đang làm tốt hơn.
- Winston (1992) cho rằng trí tuệ nhân tạo là lĩnh vực nghiên cứu các tính toán để máy có thể nhận thức, lập luận và tác động.



Hình 2.2: Thành phần của trí tuệ nhân tạo

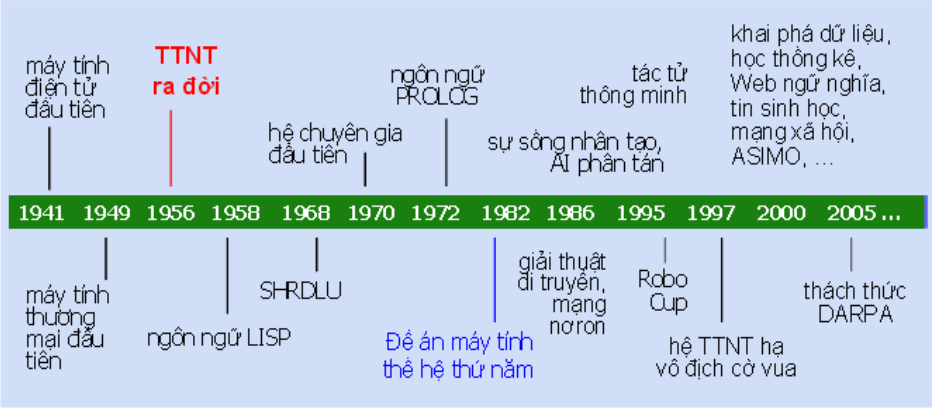
(Nguồn: Hồ Tú Bảo - Phòng thí nghiệm Phương pháp luận Sáng tạo Tri thức Viện Khoa học và Công nghệ Tiên tiến Nhật bản)

2.2.2 Lịch sử về trí tuệ nhân tạo

Vào tháng 10 năm 1950, nhà bác học người Anh Alan Turing đã xem xét vấn đề "liệu máy có khả năng suy nghĩ hay không?" (I propose to consider the question, "Can machines think?") [5]. Để trả lời câu hỏi này, ông đã đưa ra khái niệm "phép thử bắt chước" (Imitation test) mà sau này người ta gọi là "phép thử Turing" (Turing test) Phép thử được phát biểu dưới dạng một trò chơi. Theo đó, có ba đối tượng tham gia trò chơi (gồm hai người và một máy tính). Một người (người thẩm vấn) ngồi trong một phòng kín tách biệt với hai đối tượng còn lại. Người này đặt các câu hỏi và nhận các câu trả lời từ người kia (người trả lời thẩm vấn) và từ máy tính. Cuối cùng, nếu người thẩm vấn không phân biệt được câu trả lời nào là của người, câu trả lời nào là của máy tính thì lúc đó có thể nói máy tính đã có khả năng "suy nghĩ" giống như người.

Như vậy lịch sử AI có thể được tóm gọn trong 3 giai đoạn: [7]

- Giai đoạn một (1950-1965): Các công trình nghiên cứu của họ được Bộ Quốc Phòng Mỹ tài trợ và họ đầy lạc quan về tương lai của bộ môn mới này (Chương trình chơi cờ của Samuel, Chương trình lý luận logic của Newell & Simon, Chương trình chứng minh các định lý hình học của Gelernter.).
- Giai đoạn hai (1965 - 1975): Tập trung vào việc biểu diễn tri thức và phương thức giao tiếp giữa người và máy tính bằng ngôn ngữ tự nhiên. Giai đoạn ba (từ 1975): Trí tuệ nhân tạo dần trở thành một ngành công nghiệp. Các hệ thống và các chương trình trong lĩnh vực này đã được dùng trong thương mại và mang lại lợi nhuận cho người sử dụng.



Hình 2.3: Chặng đường của trí thông minh nhân tạo

(Nguồn: Hồ Tú Bảo - Phòng thí nghiệm Phương pháp luận Sáng tạo Tri thức Viện Khoa học và Công nghệ Tiên tiến Nhật bản)

2.2.3 Các lĩnh vực của AI [10]

- Lập luận, suy diễn rộng: Suy diễn logic, rút ra kết luận từ những giả thiết đã cho.
- Học máy: Nghiên cứu về khả năng học của máy tính mà không cần phải lập trình tường minh ngay từ đầu.
- Xử lý ngôn ngữ tự nhiên: Ứng dụng dựa trên ngôn ngữ của con người: Nhận dạng tiếng nói, nhận dạng chữ viết, dịch tự động, tìm kiếm thông tin.
- Robot: Chế tạo robot đối phó và dò tìm các nạn nhân trong thảm họa, xe tự lái, robot phụ vụ.

2.2.4 Các thành tựu của AI

Ngày 11/05/1997, siêu máy tính Deep Blue của IBM đã đánh bái đại kiện tướng cờ vua người Nga, Garry Kimovich Kasparov.

Ngày 31/10/2000, Honda cho ra mắt robot thông minh tên ASIMO có thể bước chéo, nhảy múa, leo cầu thang, đứng một chân tiến và lùi... đặc biệt có thể thực hiện các cử chỉ, biểu lộ giống như con người: khóc, tức giận, vui mừng, ngạc nhiên.

Từ 9 – 15/03/2016, AlphaGo của Google đã đánh bại đương kim vô địch cờ vây thế giới Lee Se-dol sau 5 ván đấu tại Seoul, Hàn Quốc.

2.3 Mạng noron nhân tạo (Artificial Neural Network)

2.3.1 Giới thiệu mạng Nơ-ron

Mạng Nơ-ron nhân tạo (ANN) là một tập hợp các phần tử xử lý đơn giản được kết nối với nhau. Mỗi phần tử xử lý này chỉ có thể thực hiện được một thao tác tính toán nhỏ, nhưng một mạng lưới các phần tử như vậy có một khả năng to lớn hơn nhiều. Mạng Nơ-ron nhân tạo được nghiên cứu trên cơ sở bộ não con người.

- **Bộ não con người**

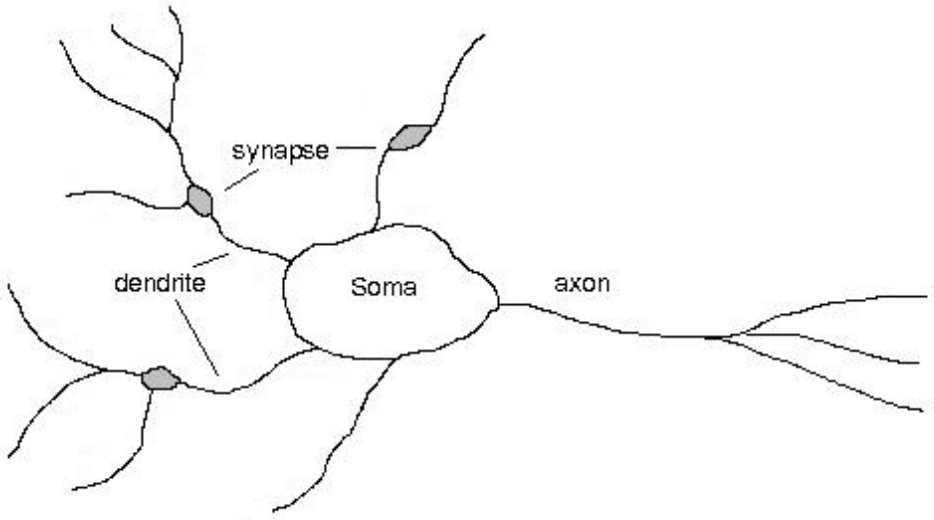
Bộ não của chúng ta bao gồm khoảng 100 tỷ đơn vị xử lý liên kết nhau để tạo thành mạng lưới xử lý thông tin mà ta gọi là các Nơ-ron thần kinh. Mỗi nơ-ron hoạt động như một bộ xử lý đơn giản. Chính sự tương tác khổng lồ giữa tất cả các tế bào này cùng với quá trình xử lý song song của chúng tạo nên khả năng tuyệt vời của bộ não

Các sợi nhánh lá các ống phân nhánh như cành cây từ thân nơ-ron, chúng tiếp nhận tín hiệu từ bên ngoài.

Thân nơ-ron chứa nhân và các cấu trúc khác, hỗ trợ quá trình xử lý hóa học và sản xuất ra chất dẫn truyền thần kinh.

Sợi trục là một ống đặc biệt dẫn truyền tín hiệu đến các nơ-ron khác, cơ quan phản ứng (vận động) hoặc vùng đệm. Các sợi trục đi chung với nhau thành từng bó gọi là dây thần kinh.

Khu vực kết thúc của sợi trục tiếp xúc các sợi nhánh của nơ-ron khác được gọi là xi-nap.



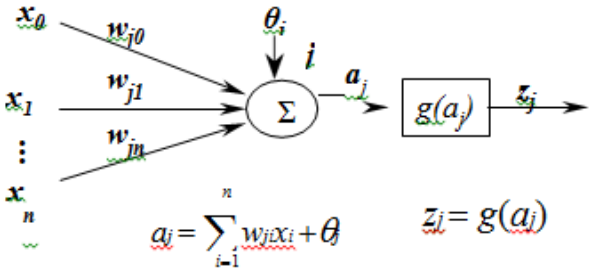
Hình 2.4: Cấu tạo của tế bào noron sinh học.

(Nguồn: Phạm Thị Hoàng Nhung – Khoa Công nghệ thông tin - Đại học Thủy Lợi)

- **Mạng nơ-ron nhân tạo**

Năm 1943, Warren McCulloch và Walter Pitts đưa ra một mô hình đơn giản các nơ-ron nhân tạo. Đây cũng chính là bước khởi đầu lịch sử của ANN. Cho tới tận ngày nay, mô hình này vẫn được xem như là nền tảng cho hầu hết các ANN. Ở đây, các nơ-ron được gọi là các Perceptron. [13]

Nơ-ron nhân tạo cơ bản



Hình 2.5: Cấu tạo của noron nhân tạo

(Nguồn: Trần Đức Minh, Luận văn tốt nghiệp cao học, Hà Nội, 12/2002)

Trong đó:

x_i : các đầu vào

w_{ji} : các trọng số tương ứng với các đầu vào

θ_j : độ lệch (bias)

a_j : đầu vào mạng (net-input)

z_j : đầu ra của nơron

g(x): hàm chuyển (hàm kích hoạt).

Trong một mạng nơron có ba kiểu đơn vị:

- Các đơn vị đầu vào (Input units), nhận tín hiệu từ bên ngoài;
- Các đơn vị đầu ra (Output units), gửi dữ liệu ra bên ngoài;
- Các đơn vị ẩn (Hidden units), tín hiệu vào (input) và ra (output) của nó nằm trong mạng.

Mỗi đơn vị *j* có thể có một hoặc nhiều đầu vào: x_0, x_1, x_2, … x_n, nhưng chỉ có một đầu ra z_j. Một đầu vào tới một đơn vị có thể là dữ liệu từ bên ngoài mạng, hoặc đầu ra của một đơn vị khác, hoặc là đầu ra của chính nó.

2.3.2 Hàm xử lý

Để chạy các dữ liệu trong mạng buộc phải có các hàm xử lý. Hàm thực hiện nhiệm vụ này gọi là hàm kết hợp. Trong phần lớn các mạng nơron, chúng ta giả sử rằng mỗi một đơn vị cung cấp một bộ cộng như là đầu vào cho đơn vị mà nó có liên kết. Tổng đầu vào đơn vị *j* đơn giản chỉ là tổng trọng số của các đầu ra riêng lẻ từ các đơn vị kết nối cộng thêm ngưỡng hay độ lệch (bias) θj :

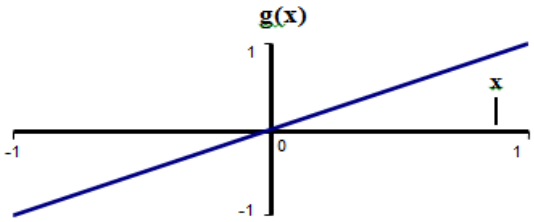
$$a_j = \sum_{i=1}^{n} w_{ij} x_i + \theta_j$$

Trường hợp $w_{ji} > 0$, nơron được coi là đang ở trong trạng thái kích thích. Tương tự, nếu như $w_{ji} < 0$, nơron ở trạng thái kiềm chế. Chúng ta gọi các đơn vị với luật lan truyền như trên là các sigma units. Phần lớn các đơn vị trong mạng nơron chuyển net input bằng cách sử dụng một hàm vô hướng gọi là hàm kích hoạt, kết quả của hàm này là một giá trị gọi là mức độ kích hoạt của đơn vị. Các hàm kích hoạt thường bị ép vào một khoảng giá trị xác định. Các hàm kích hoạt hay được sử dụng là:

- Hàm đồng nhất (Linear function, Identity function)

$$g(x) = x$$

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



Hình 2.6: Đồ thị của hàm đồng nhất

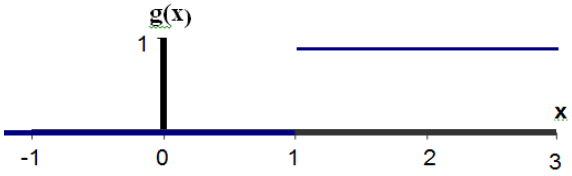
(Nguồn: Trần Đức Minh, 12/2002)

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

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

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

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



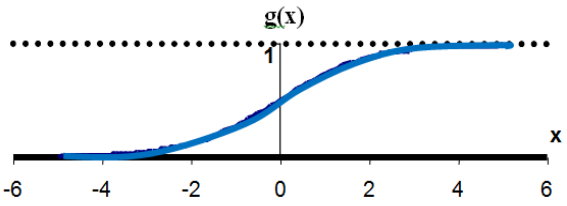
Hình 2.7: Đồ thị của hàm bước nhị phân

(Nguồn: Trần Đức Minh, 12/2002)

- Hàm sigmoid (Sigmoid function (logsig))

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

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



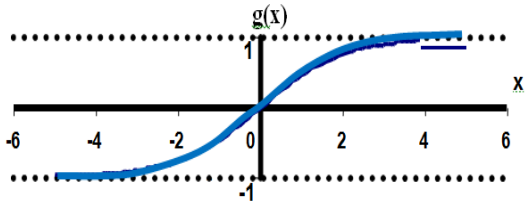
Hình 2.8: Đồ thị của hàm Sigmoid

(Nguồn: Trần Đức Minh, 12/2002)

- Hàm sigmoid lưỡng cực (Bipolar sigmoid function (tansig))

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

Hàm này có các thuộc tính tương tự hàm sigmoid. Nó làm việc tốt đối với các ứng dụng có đầu ra yêu cầu trong khoảng [-1,1].



Hình 2.9: Đồ thị của hàm sigmoid lưỡng cực

(Nguồn: Trần Đức Minh, 12/2002)

2.3.3 Chọn lớp ẩn

- **Số lớp ẩn**

Về lý thuyết không có lý do nào sử dụng các mạng có nhiều hơn hai lớp ẩn. Người ta đã xác định rằng đối với phần lớn các bài toán cụ thể chỉ cần sử dụng một lớp ẩn cho mạng là đủ để giải quyết. Các bài toán sử dụng hai lớp ẩn hiếm khi xảy ra trong thực tế. Ngoài ra, việc huấn luyện mạng thường rất chậm khi mà số lớp ẩn sử dụng càng nhiều. Lý do sau đây giải thích cho việc sử dụng càng ít các lớp ẩn càng tốt là:

- Phần lớn các thuật toán luyện mạng cho các mạng noron truyền thẳng đều dựa trên phương pháp gradient. Các lớp thêm vào sẽ thêm việc phải lan truyền các lỗi làm cho vector gradient rất không ổn định.
- Số các cực trị địa phương tăng lên rất lớn khi có nhiều lớp ẩn và xác suất khá cao là chúng ta sẽ bị tắc trong một cực trị địa phương sau rất nhiều thời gian

lặp, khi đó sẽ ta phải bắt đầu lại.

- Về tổng thể, người ta cho rằng việc đầu tiên là nên xem xét khả năng sử dụng mạng chỉ có một lớp ẩn. Nếu dùng một lớp ẩn với một số lượng lớn các đơn vị mà không có hiệu quả thì nên sử dụng thêm một lớp ẩn nữa với một số ít các đơn vị.

- **Số nơron trong lớp ẩn**

Một vấn đề quan trọng trong việc thiết kế một mạng là cần có bao nhiêu nơron trong lớp ẩn. Sử dụng quá ít nơron có thể dẫn đến việc không thể nhận dạng được các tín hiệu đầy đủ trong một tập dữ liệu phức tạp. Sử dụng quá nhiều nơron sẽ tăng thời gian luyện mạng. Số lượng tốt nhất của các đơn vị ẩn phụ thuộc vào rất nhiều yếu tố - số đầu vào, đầu ra của mạng, số trường hợp trong tập mẫu, độ nhiễu của dữ liệu đích, độ phức tạp của hàm lỗi, kiến trúc mạng và thuật toán luyện mạng.

Có rất nhiều cách để lựa chọn số đơn vị trong các lớp ẩn chẳng hạn nằm giữa khoảng kích thước lớp vào, lớp ra

$$m \in [t,z]$$

$$m = \frac{2(t+z)}{3}$$

$$m < 2t$$

$$m = \sqrt{t.z}$$

Trong đó:

m: Số nơron lớp ẩn

t: Số lớp đầu vào

z: Số lớp đầu ra

Các luật này chỉ có thể được coi như là các lựa chọn thô khi chọn lựa kích thước của các lớp. Chúng không phản ánh được thực tế, bởi lẽ chúng chỉ xem xét đến nhân tố kích thước đầu vào, đầu ra mà bỏ qua các nhân tố quan trọng khác như: số trường hợp đưa vào huấn luyện, độ nhiễu ở các đầu ra mong muốn, độ phức tạp của hàm lỗi, kiến trúc của mạng (truyền thẳng hay hồi quy), và thuật toán học.

Trong phần lớn các trường hợp, không có một cách để có thể dễ dàng xác định

được số tối ưu các đơn vị trong lớp ẩn mà không phải luyện mạng sử dụng số các đơn vị trong lớp ẩn khác nhau và dự báo lỗi tổng quát hóa của từng lựa chọn. Cách tốt nhất là sử dụng phương pháp thử-sai (trial-and-error). Trong thực tế, có thể sử dụng phương pháp Lựa chọn tiến (forward selection) hay Lựa chọn lùi (backward selection) để xác định số đơn vị trong lớp ẩn.

2.3.4 Giải thuật lan truyền ngược.

Đến thời điểm hiện tại, đã có rất nhiều dạng Mạng Nơ-ron khác nhau được đề ra. Và do bởi được nghiên cứu rộng rãi trong nhiều ngành khoa học (Khoa học máy tính, Kĩ thuật điện tử, sinh học và tâm lý học) nên mạng nơ-ron mang nhiều tên gọi khác nhau, ví như Mạng nơ-ron nhân tạo (Artificial Neural Networks ANNs), Mô hình kết nối (Connectionism or Connectionist Models), Perceptron đa lớp (Multi-player Perceptrons MLPs) và xử lý phân tán song song (Parallel Distributed Processing PDP).

Các mạng nơron truyền thẳng nhiều lớp được luyện bằng phương pháp học có thầy. Phương pháp này căn bản dựa trên việc yêu cầu mạng thực hiện chức năng của nó và sau đó trả lại kết quả, kết hợp kết quả này với các đầu ra mong muốn để điều chỉnh các tham số của mạng, nghĩa là mạng sẽ học thông qua những sai sót của nó

- **Mô tả thuật toán**

Ta sẽ sử dụng dạng tổng quát của mạng nơron truyền thẳng nhiều lớp. Khi đó, đầu ra của một lớp trở thành đầu vào của lớp kế tiếp. Phương trình được thể hiện như sau:

$$a^{m+1} = f^{m+1}(W^{m+1}a^m + b^{m+1}) \text{ với } m = 0, 1, ..., M - 1,$$

Trong đó M là số lớp trong mạng. Các nơron trong lớp thứ nhất nhận các tín hiệu từ bên ngoài:

$$a^0 = p$$

Chính là điểm bắt đầu của phương trình phía trên. Đầu ra của lớp cuối cùng được xem là đầu ra của mạng:

$$a = a^M$$

Đối với mạng nơron truyền thẳng nhiều lớp, lỗi không chỉ là một hàm của chỉ các trọng số trong lớp ẩn, do vậy việc tính các đạo hàm từng phần này là không đơn giản.

Chính vì lý do đó mà ta phải sử dụng luật xích để tính. Và kết quả của thuật toán giảm theo hướng được biểu diễn như sau:

$$w_{ji}^{m}(\mathrm{k}+1) = w_{ji}^{m}(k) - \alpha s_j^m a_i^{m-1}$$

$$b_j^m(k+1) = b_j^m(k) - \alpha s_j^m$$

Ở dạng ma trận sẽ là:

$$W^m(k+1) = W^m(k) - \alpha s^m (a^{m-1})^{\mathrm{T}}$$

$$b^m(k+1) = b^m(k) - \alpha s^m$$

Bây giờ ta cần tính toán nốt ma trận độ nhạy cảm s^m. Để thực hiện điều này cần sử dụng một áp dụng khác của luật xích. Quá trình này cho ta khái niệm về sự lan truyền ngược bởi vì nó mô tả mối quan hệ hồi qui trong đó độ nhạy cảm s^m được tính qua độ nhạy cảm s^{m+1} của lớp m+1. Để dẫn đến quan hệ đó, ta sẽ sử dụng ma trận Jacobian. Và kết quả đươc biểu diễn như sau:

$$s_j^m = -2(t_j - a_j) f^M(n_j^M)$$

Biểu diễn dưới dạng ma trận:

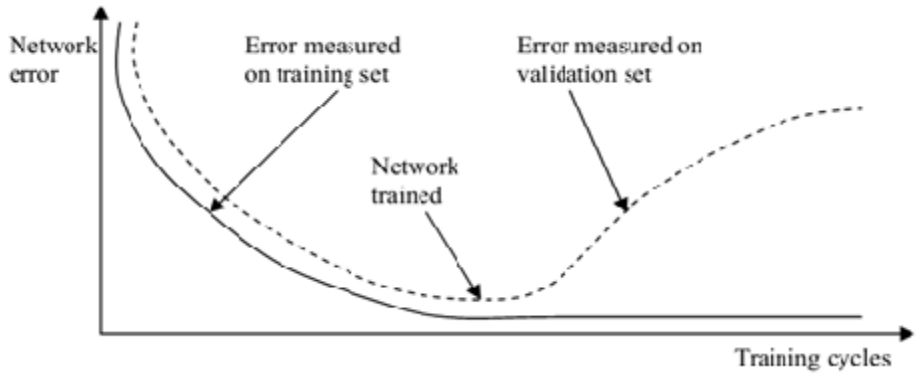
$$s^M = -2 f^M(n^M)(t-a)$$

2.3.5 Dừng quá trình huấn luyện và đánh giá sai số mạng

Quá trình huấn luyện mạng nơ-ron có thể được dừng lại khi mạng có thể nhận ra tất cả các mẫu hoặc đáp ứng được mức sai số hay yêu cầu cụ thể. Ta có thể ước lượng tổng sai số của mạng bằng cách cộng dồn các sai số của mỗi lần huấn luyện.

Nói cách khác, mạng nơ-ron tiếp tục quá trình huấn luyện tất cả các mẫu lặp đi lặp lại cho đến khi tổng sai số giảm dưới một giá trị đích định trước rồi dừng. Khi tính toán tổng sai số cho điều kiện dừng của mạng, cần chuyển các sai số về giá trị dương.

Sau khi mạng nơ-ron được huấn luyện, nó sẽ có khả năng nhận ra được không chỉ các mẫu ví dụ hoàn thiện mà còn có thể nhận dạng các mẫu bị lỗi, nhiễu. Thực tế, ta có thể chủ động thêm vào tập mẫu huấn luyện các mẫu nhiễu để cải thiện khả năng kháng lỗi của mạng. Quá trình đạo tạo sẽ hiệu quả hơn nếu các mẫu áp dụng cho việc huấn luyện được sắp xếp theo thứ tự ngẫu nhiên.



Hình 2.10: Đánh giá sai số của mạng neuron sau khi huấn luyện

(Nguồn: Trần Đức Minh, Luận văn tốt nghiệp cao học, Hà Nội, 12/2002)

Khi mạng nơ-ron được huấn luyện đầy đủ, sai số tập hợp lệ sẽ đạt cực tiểu. Khi mạng bị huấn luyện quá mức, giá trị sai số của tập hợp lệ bắt đầu tăng dần, và khi đó, mạng nơ-ron này sẽ mất dần khả năng xử lý các dữ liệu nhiễu.

2.3.6 Vấn đề của mạng lan truyền ngược

Mạng lan truyền ngược có nhiều vấn đề cần được quan tâm. Vấn đề được biết đến nhiều nhất là "Cực tiểu cục bộ", xảy ra do giải thuật luôn luôn tìm cách điều chỉnh trọng số để giảm giá trị sai số. Như đôi lúc giá trị sai số cần phải tăng ở một khu vực cục bộ để đảm bảo quá trình giảm toàn cục và giải thuật bị mắc kẹt, dừng lại khi sai số chưa phải là cực tiểu mong muốn.

Khi mạng trở nên lớn hơn, ta sẽ phải đối mặt với nhiều vấn đề hơn nhưng hầu hết đều có thể giải quyết bằng cách khởi tạo lại trọng số của mạng. Và hiện nay cũng đã có nhiều dạng khác nhau của giải thuật lan truyền ngược được phát triển để giải quyết các vấn đề này.

Ưu điểm của mạng sử dụng giải thuật lan truyền ngược là khả năng nhận dạng mẫu. Các mẫu được trình diện trực tiếp cho mạng được xác định vị trí trên lưới ô vuông và đúng kích thước. Nhược điểm trong nhận dạng mẫu của nó là khả năng xử lý các mẫu trong các quan cảnh hỗn loạn như nhận dạng khuôn mặt trong đám đông hay 1 kí tự trong một trang in. Do đó, chúng ta sẽ phải cần tiền xử lý dữ liệu để có được định dạng chuẩn trước khi áp dụng cho mạng.

2.3.7 Các nghiên cứu đã thực hiện

Hiện nay trên thế giới việc sử dụng trí thông minh nhân tạo cụ thể là mạng neuron nhân tạo và thuật toán lan truyền ngược được thực hiện khá nhiều trong các vấn đề:

Phi công tự động, giả lập đường bay, các hệ thống điều khiển lái máy bay, bộ phát hiện lỗi, các hệ thống dẫn đường tự động cho ô tô, các bộ phân tích hoạt động của xe, định vị - phát hiện vũ khí, dò mục tiêu, phát hiện đối tượng, nhận dạng nét mặt, các bộ cảm biến thế hệ mới, xử lý ảnh radar, dự đoán mã tuần tự, sơ đồ chip IC, điều khiển tiến trình, phân tích nguyên nhân hỏng chip, nhận dạng tiếng nói, mô hình phi tuyến,…

Đặc biệt mạng neuron nhân tạo được áp dụng trong vấn đề tai nạn giao thông đã được nhiều bài báo khoa học trên thế giới (Thái, Negeria,…). Điển hình như:

- Miao M. Chong, Ajith Abraham, Marcin Paprzycki đã sự dụng kỹ thuật ANN và Cây quyết định để đánh giá dữ liệu các vụ tai nạn giao thông năm 1997 tại khu vực trung tâm Florida, Mỹ. Mục tiêu của nghiên cứu nhằm cải thiện chính xác hơn các dự báo về TNGT, tiến đến phát triển một mô hình dự báo các yếu tố ảnh hưởng nghiêm trọng đến chấn thương trong các vụ TNGT. [18]
- Francisca Nonyelum Ogwueleka, Toochukwu Chibueze Ogwueleka, L. Fernandez-Sanz đã tiến hành nghiên cứu áp dụng ANN vào dự đoán TNGT đường bộ tại các nước đang phát triển điển hình là tại Nigeria bằng cách sử dụng dữ liệu từ năm 1998 đến 2010. Báo cáo đi sâu vào phân tích các lỗi trong các vụ TNGT chủ yếu liên quan đến con người. Mục đích là so sánh, lập mô hình và dự báo TNGT để từ đó cung cấp cho chính phủ Nigeria thông tin cần để đưa ra biện pháp giải quyết phù hợp. [15]
- Vào tháng 5 năm 2010, F. Rezaie Moghaddam, Sh. Afandizadeh, M. Ziyadi đã đưa ra báo cáo nghiên cứu ứng dụng mạng ANN để dự đoán các vụ TNGT nghiêm trọng. Thông qua việc thu thập 2889 thông tin vụ TNGT trong năm 2006 để thành lập mô hình mô tả, dữ đoán các yếu tố ảnh hưởng lớn đến các vụ TNGT. [14]

2.4 Phân tích hồi quy tương quan

2.4.1 Phương trình hồi quy

Phương trình hồi quy có dạng:

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \dots \beta_k X_k$$

Trong đó:

- α là hệ số tự do.
- $\beta_{1,} \beta_{2,\dots} \beta_k$ là hệ số hồi quy.
- Y là biến phụ thuộc.
- $X_1, X_2 \dots X_k$ là biến độc lập.

2.4.2 Hệ số xác định R^2

Hệ số xác định R^2 (Coeffocient of determination) (đơn vị: %) là một trong các chỉ tiêu dung để đánh giá mức độ phù hợp của các mô hình thể hiện mối liên hệ tương quan tuyến tính, hệ số xác định chính là bình phương của hệ số tương quan. Là tỷ lệ (hoặc %) của sự biến động của biến phụ thuộc Y được giải thích bởi các biến độc lập $X_{i.}$

$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST} \ (0 \leq R^2 \leq 1)$$

Trong đó:

- $SSE = \sum_{i=1}^{n} e_i^2$: Tổng bình phương sai số.
- $SSR = \sum_{i=1}^{n} (y_i - \bar{y})^2$: Tổng bình phương hồi quy.
- $SST = \sum_{i=1}^{n} [e_i^2 + (y_i - \bar{y})^2]$: Tổng bình phương tổng công.

Giá trị R^2 thường được tính bằng % và cách đánh giá mối liên hệ từ hệ số xác định như sau:

Bảng 2.1: Bảng đánh giá mức độ tương quan

Giá trị	Mức độ tương quan
$R^2 \leq 10\%$	Tương quan ở mức thấp
$10\% \leq R^2 \leq 25\%$	Tương quan ở mức trung bình
$25\% \leq R^2 \leq 50\%$	Tương quan khá chặt chẽ
$50\% \leq R^2 \leq 80\%$	Tương quan chặt chẽ

$80\% \leq R^2$	Tương quan rất chặt chẽ

2.4.3. Hệ số tương quan bội

Hệ số tương quan (Corelation Coeffcient) đo lường mức độ quan hệ tuyến tính giửa 2 biến, chính xác hơn là quan hệ tuyến tính giữa các biến không phân biệt biến này phụ thuộc vào biến kia

Hệ số tương quan bội nói lên tính chặt chẽ của mối liên hệ giữa biến phụ thuộc Y và các biến độc lập X_i.

$$R = \sqrt{R^2} \ (-1 \leq R \leq 1)$$

Các đánh giá mối liên hệ từ hệ số tương quan như sau:

Bảng 2.2: Bảng đánh giá mới liên hệ tương quan

Giá trị R	Mức độ tương quan
$R < 0,3$	Tương quan ở mức thấp
$0,3 \leq R \leq 0,5$	Tương quan ở mức trung bình
$0,5 \leq R \leq 0,7$	Tương quan khá chặt chẽ
$0,7 \leq R \leq 0,9$	Tương quan chặt chẽ
$0,9 \leq R$	Tương quan rất chặt chẽ

2.5 Ngôn ngữ Python

2.5.1 Python là gì

Python là một ngôn ngữ lập trình thông dịch (interpreted), tức là ngôn ngữ không cần phải biên dịch một lần ra file chạy mà đọc code đến đâu chạy đến đó do Guido van Rossum tạo ra năm 1990. Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động, do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl.

Theo đánh giá của Eric S. Raymond, Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu, như nhận định của chính Guido van Rossum trong một bài phỏng vấn ông.

2.5.2 Ưu, nhược điểm của Python

- **Ưu điểm**

– Được biết đến như một ngôn ngữ lập trình dễ học và dễ đọc.

– Thư viện có sẵn nhiều và hỗ trợ mạnh mẽ.

– mạnh trong việc xử lý các loại dữ liệu chuỗi, tập hợp

– Chạy trong nhiều hệ thống và nền tảng như MacOSX, Windows, Linux

– Python cung cấp giao diện cho tất cả các cơ sở dữ liệu thương mại lớn.

– Có thể thêm các module ở mức độ thấp để các thông dịch Python.

- **Nhược điểm**

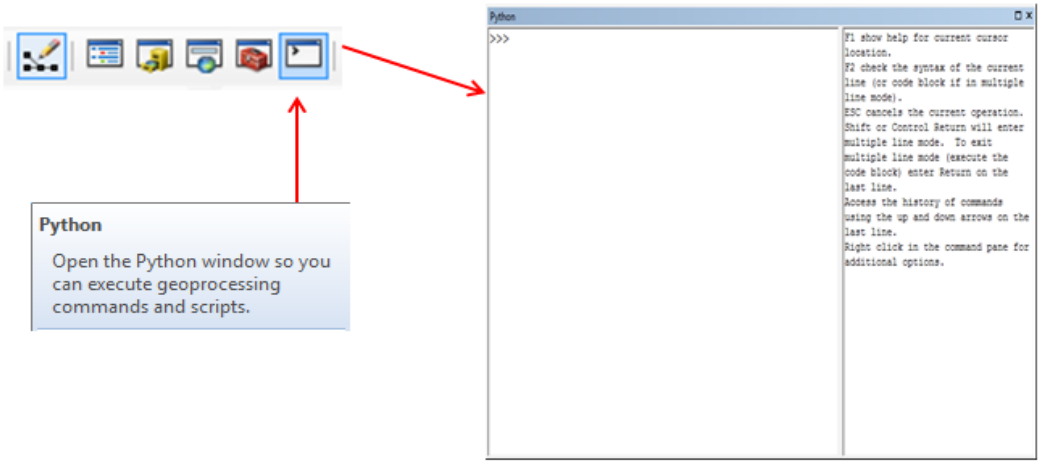
– Tuy là Python nhanh hơn so với PHP, nhưng lại không nhanh hơn so với C++, Java.

– Đối với ngôn ngữ lập trình Python thì không có vòng lặp do…while và switch….case.

– Python cũng không có các thuộc tính như: protected, private hay public.

2.5.3 Python trong GIS

Esri cũng tích hợp các thành phần mã nguồn mở tốt nhất trong ArcGIS. Kể từ ArcGIS 9, phần mềm đã tích hợp Python- một ngôn ngữ lập trình mã nguồn mở, và trong ArcGIS 10 đã giới thiệu ArcPy- một gói Python để đơn giản hóa và tự động hóa kịch bản Python.



Hình 2.11: Chương trình Python trong ArcGIS 10.3

2.6 Phần mềm MATLAB

2.6.1 Giới thiệu về MATLAB

Matlab là viết tắt từ "Matrix Laboratory", được Cleve Moler phát minh vào cuối thập niên 1970, và sau đó là chủ nhiệm khoa máy tính tại Đại học New Mexico. là phần mềm cung cấp môi trường tính toán số và lập trình, do công ty MathWorks thiết kế. MATLAB cho phép tính toán số với ma trận, vẽ đồ thị hàm số hay biểu đồ thông tin, thực hiện thuật toán, tạo các giao diện người dùng và liên kết với những chương trình máy tính viết trên nhiều ngôn ngữ lập trình khác. Với thư viện Toolbox, MATLAB cho phép mô phỏng tính toán, thực nghiệm nhiều mô hình trong thực tế và kỹ thuật.

2.6.2 Cấu trúc

MATLAB gồm 5 phần chính:

- Development Environment: là một bộ các công cụ giúp ta sử dụng các hàm và tập tin của MATLAB. Nó bao gồm: MATLAB desktop, Command Window, a command history, an editor, debugger, browsers for viewing help, the workspace, files, the search path.
- MATLAB Mathematical Function Library: tập hợp các hàm tóan học như sum, sine, số học,….
- MATLAB Language (scritp): ngôn ngữ lập trình bậc cao.
- Graphics: các công cụ giúp hiễn thị dữ liệu dưới dạng đồ thị. Ngòai ra nó còn cho phép xây dựng giao diện đồ họa.
- MATLAB Application Program Interface: bộ thư viện cho phép ta sử dụng các hức năng tính tóan của MATLAB trong chương trình C hay FORTRAN.

2.6.3 Đặc điểm của MATLAB

- Là một ngôn ngữ cấp cao cho tính toán số, hình dung và phát triển ứng dụng.
- Nó cũng cung cấp một môi trường tương tác thăm dò lặp đi lặp lại, thiết kế và giải quyết vấn đề.

- Cung cấp các thư viện lớn các chức năng toán học cho đại số tuyến tính, thống kê, phân tích Fourier, lọc, tối ưu hóa, hội nhập số và giải phương trình vi phân thường.
- Cung cấp được xây dựng trong đồ họa để hình dung dữ liệu và các công cụ cho việc tạo ra các ô tùy chỉnh.
- Giao diện lập trình MATLAB cung cấp cho các công cụ phát triển để cải thiện chất lượng mã bảo trì và tối đa hóa hiệu suất.
- Nó cung cấp các công cụ để xây dựng các ứng dụng với giao diện tùy chỉnh đồ họa.
- Cung cấp các chức năng cho việc tích hợp các thuật toán MATLAB dựa với các ứng dụng bên ngoài và các ngôn ngữ như C, Java, .NET và Microsoft Excel.

2.6.4 Khả năng ứng dụng của MATLAB

MATLAB là một bộ chương trình phần mềm lớn dành cho tính toán kỹ thuật. ta có thể dùng MATLAB để:

- Ma trận và Mảng
- Vẽ 2-D và 3-D và đồ họa
- Đại số tuyến tính
- đại số Equations
- Chức năng phi tuyến tính
- Thống kê
- Phân tích dữ liệu
- Calculus và Differential Equations
- Tính toán số học
- Hội nhập
- Transforms
- Lắp đường cong

Các ngành nghề kỹ thuật được Matlab hỗ trợ đắc lực:

- Xử lý tín hiệu và Truyền thông

- Hình ảnh và Video Processing
- Hệ thống kiểm soát
- Kiểm tra và đo lường
- Tính toán Tài chính
- Sinh Học Máy Tính

CHƯƠNG 3
DỮ LIỆU VÀ PHƯƠNG PHÁP NGHIÊN CỨU

3.1 Dữ liệu thu thập

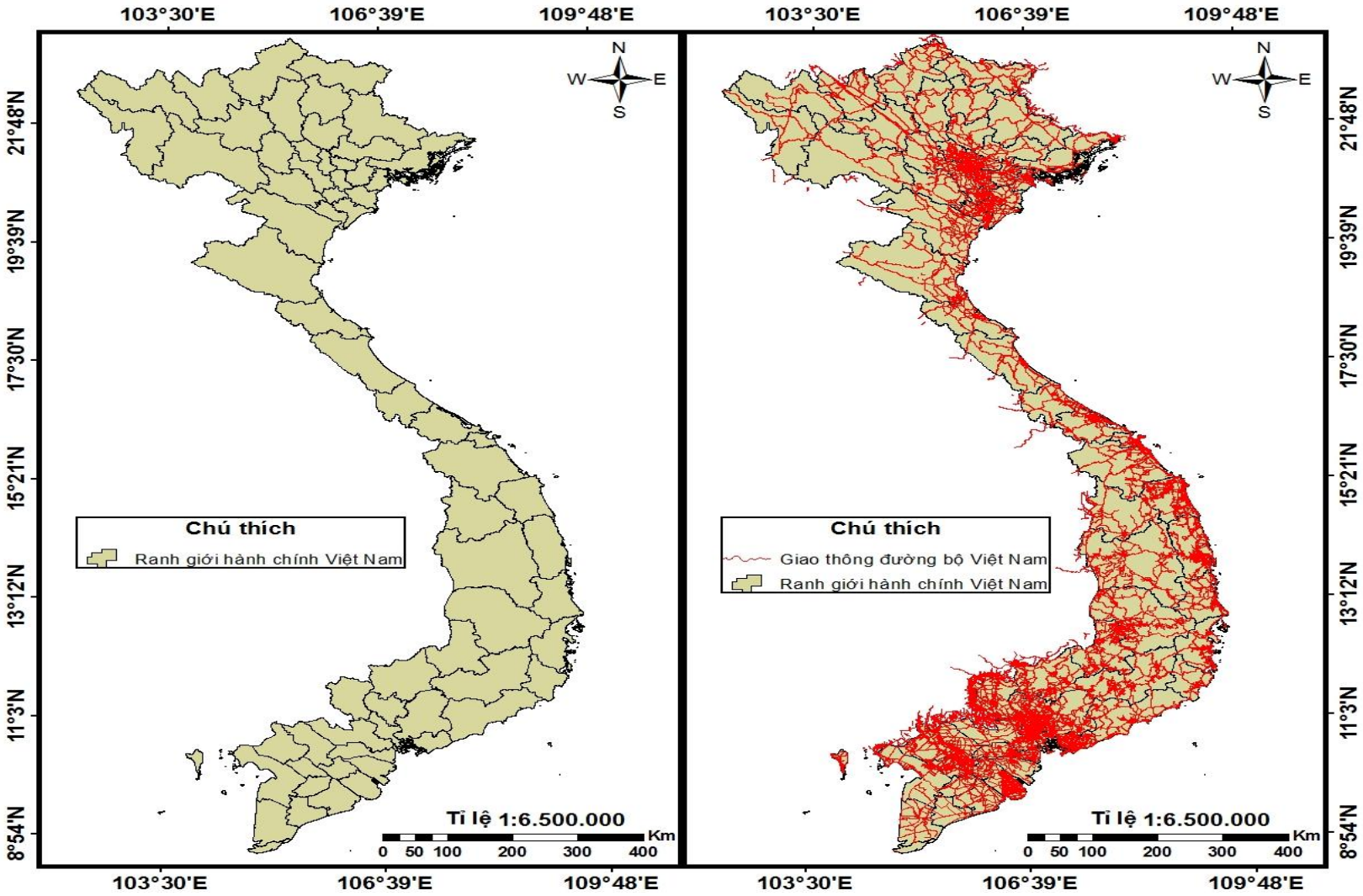
Nguồn dữ liệu bao gồm dữ liệu không gian và dữ liệu thuộc tính được thu thập từ Trung tâm Ứng dụng Hệ thống Thông tin Địa lý TP.HCM (HCMGIS) và từ một số dịch vụ bản đồ thế giới trực tuyến phi lợi nhuận như OpenStreetMap (OSM), DIVA-GIS. Thông tin chi tiết được mô tả trong bảng 3.1 và hình 3.1.

Bảng 3.1: Thông tin các lớp dữ liệu sử dụng trong bài luận.

STT	Tên dữ liệu	Mô tả chi tiết	Nguồn - Link
1	Ranh giới hành chính 63 tỉnh thành cả nước.	Loại: Shape file Hệ tọa độ: decimal degree (dd) Dữ liệu dạng vùng (polygon) thể hiện chi tiết ranh giới hành chính 63 tỉnh thành trong cả nước.	Nguồn: diva-gis Link: http://www.diva-gis.org/gdata
2	Mạng lưới giao thông cả nước.	Loại: Shape file Hệ tọa độ: decimal degree (dd) Dữ liệu dạng đường (polyline) bao gồm các tuyến đường giao thông trên cả nước.	Nguồn: OSM Link: http://download.geofabrik.de/asia/vietnam.html

3	Tọa độ các điểm tai nạn giao thông tại TPHCM	Loại: Excel Dữ liệu bao gồm các thông tin chi tiết về các vụ TNGT tại TPHCM gồm: – Số lượng người bị / gây tai nạn. – Quê quán người bị / gây tai nạn. – Tuổi người bị / gây tai nạn. – Phương tiện liên quan. – Thời gian, địa chỉ (tên đường, phường, huyện, tỉnh)xảy ra tai nạn. – Nguyên nhân xảy ra tai nạn.	Nguồn: Trung tâm Ứng dụng Hệ thống Thông tin Địa lý TPHCM

Thông tin về các điểm TNGT tại TPHCM xin xem phần phụ lục.



Hình 3.1: Shapeflie dữ liệu ranh giới hành chính và hệ thống giao thông cả nước.

3.2 Phương pháp nghiên cứu

Việc thực hiện đề tài được chia thành 4 giai đoạn chính cụ thể như sau:

- Giai đoạn 1: Thu thập, xây dựng, xử lý dữ liệu không gian và thông tin các vụ TNGT.
- Giai đoạn 2: Số hóa các tọa độ TNGT và đánh giá độ chính xác.
- Giai đoạn 3: Phân tích dữ liệu, chọn yếu tố đều vào / ra phù hợp và chuyển dữ liệu sang mã nhị phân.
- Giai đoạn 4: Phân tích mạng Neuron dựa trên dữ liệu đã mã hóa và đánh giá sai số.

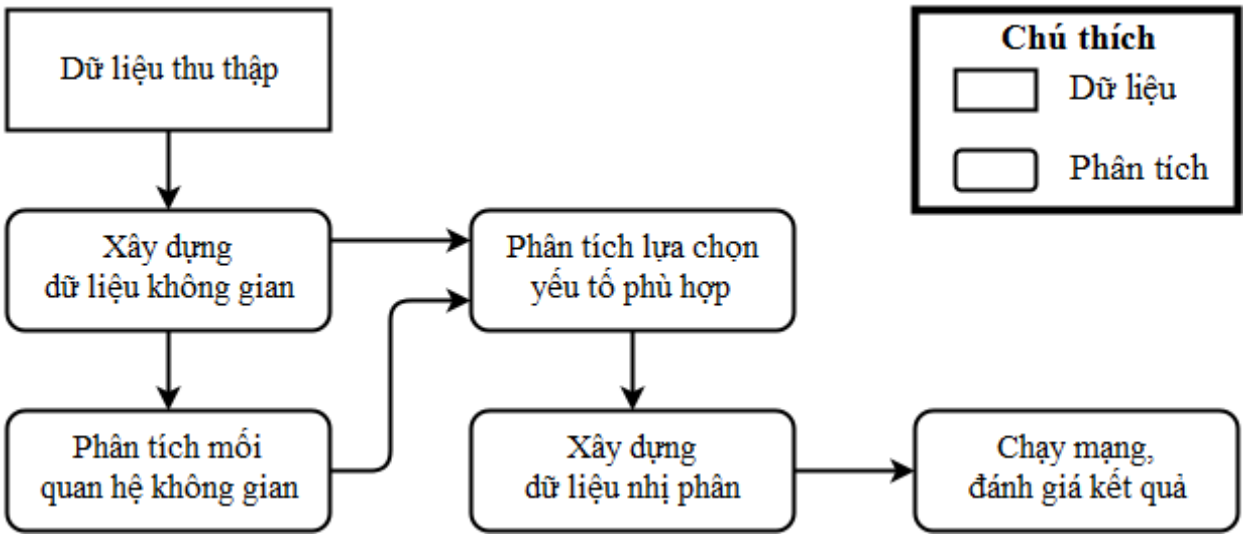
Trong giai đoạn 1 dựa trên mục tiêu đề tài là áp dụng mạng neural nhân tạo để nhận diện các vụ TNGT tại TPHCM, vì vậy dữ liệu quan trọng nhất phục vụ cho bài toán là dữ liệu mô tả chi tiết các vụ TNGT tại TPHCM. Ngoài ra, các dữ liệu không gian về ranh giới, giao thông của TPHCM cũng được thu thập nhằm đánh giá độ chính xác của dữ liệu TNGT thu thập và thể hiện trực quan hóa dữ liệu TNGT trên bản đồ. Đồng thời xây dựng dữ liệu các vụ TNGT đã thu thập được để có thể phân tích.

Giai đoạn 2 tiến hành số hóa dữ liệu và kiểm tra độ chính xác của dữ liệu trên không gian. Đồng thời tính toán lại các điểm TNGT bị sai sót, phân bố không hợp lý về mặt không gian để có thể đưa vào phân tích.

Giai đoạn 3 được thực hiện ngay sau khi dữ liệu được lựa chọn và hoàn tất. Vì dữ liệu gốc bao gồm rất nhiều thông tin do đó cần phải phân tích, lựa chọn loại thông tin cần thiết nhất cho đề tài tránh bị rối và quá tải thông tin. Ngoài ra mặc dù dữ liệu đã được hoàn tất nhưng để phân tích được mạng Neuron thì buộc dữ liệu phải nằm ở dạng số (number) cụ thể hơn trong bài phải là dạng nhị phân 0 và 1 trong khi dữ liệu sau khi xây dựng vẫn còn chứa thông tin chữ. Vì vậy cần dựa vào lớp dữ liệu giao thông, dữ liệu ranh giới hành chính các quận / huyện, thông tin từ việc thống kê TNGT trên địa bàn thành phố và kiến thức cá nhân để phân chia dữ liệu sang dạng 0 và 1 cũng như lựa chọn lớp dữ liệu vào / ra cho phù hợp.

Giai đoạn 4 sau khi đã mã hóa dữ liệu sang dạng nhị phân 0 và 1, dữ liệu sẽ được đưa vào công cụ MATLAB để phân tích mạng neuron để lấy cấu hình mạng và tiến hánh

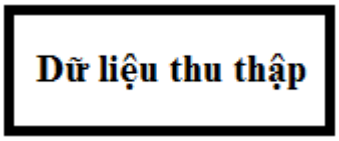
đánh giá sai số. Đồng thời chạy lại lại mạng nhiều lần để lấy kết quả cấu hình mạng phù hợp nhất có thể với bộ dữ liệu đã xây dựng trên.



Hình 3.2: Sơ đồ phương pháp nghiên cứu

CHƯƠNG 4
KẾT QUẢ, THẢO LUẬN

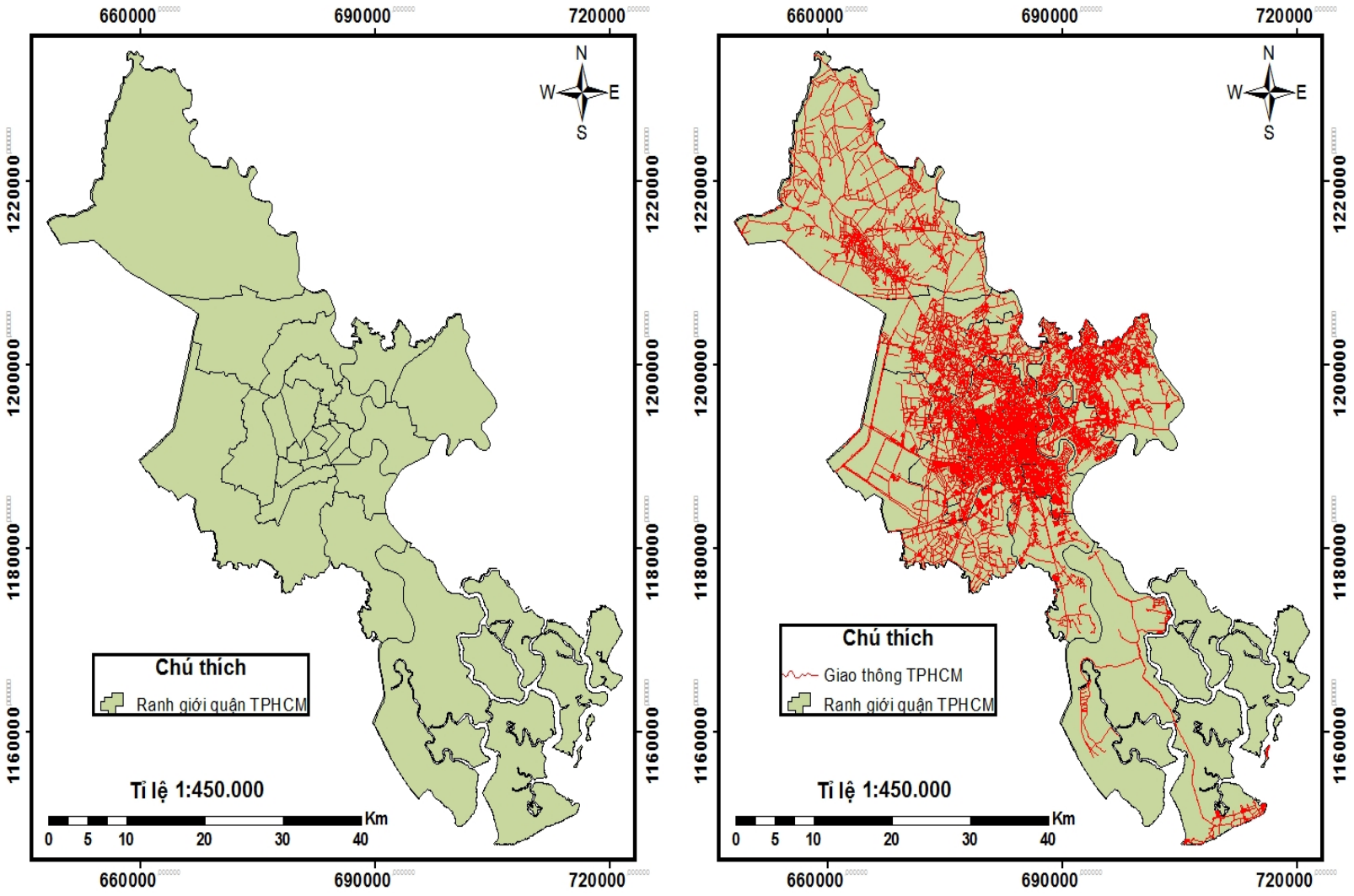
4.1 Giai đoạn 1



Hình 4.1: Giai đoạn thu thập dữ liệu

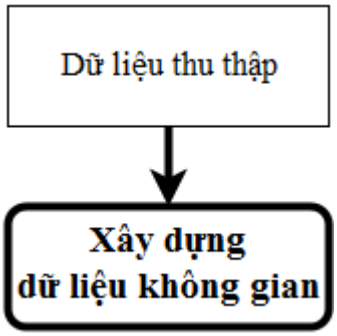
Việc thu thập, xây dựng, xử lý dữ liệu thu được những kết quả như sau:

- Dữ liệu các vụ TNGT tại TPHCM: Thu thập và xây dựng bộ dữ liệu của 339 vụ TNGT gồm các thông tin: Kinh độ, vĩ độ, thông tin của người bị / gây tai nạn (tuổi, nghề nghiệp, giới tính, quê quán, phương tiện giao thông, tình trạng), thời gian và địa điểm xảy ra tai nạn, số lượng người bị / gây tai nạn, nguyên nhân.
- Dữ liệu không gian: Vì phạm vi của bài luận là TPHCM do đó phải tách ranh giới hành chính và hệ thống giao thông của TPHCM bằng công cụ clip. Ngoài ra hệ tọa độ dữ liệu khi thu thập được nằm ở dạng decimal degree nên cần đổi về hệ tọa độ UTM. Kết quả được mô ta trong hình 4.1



Hình 4.2: Ranh giới hành chính quận (trái) và hệ thống giao thông (phải) TPHCM

4.2 Giai đoạn 2



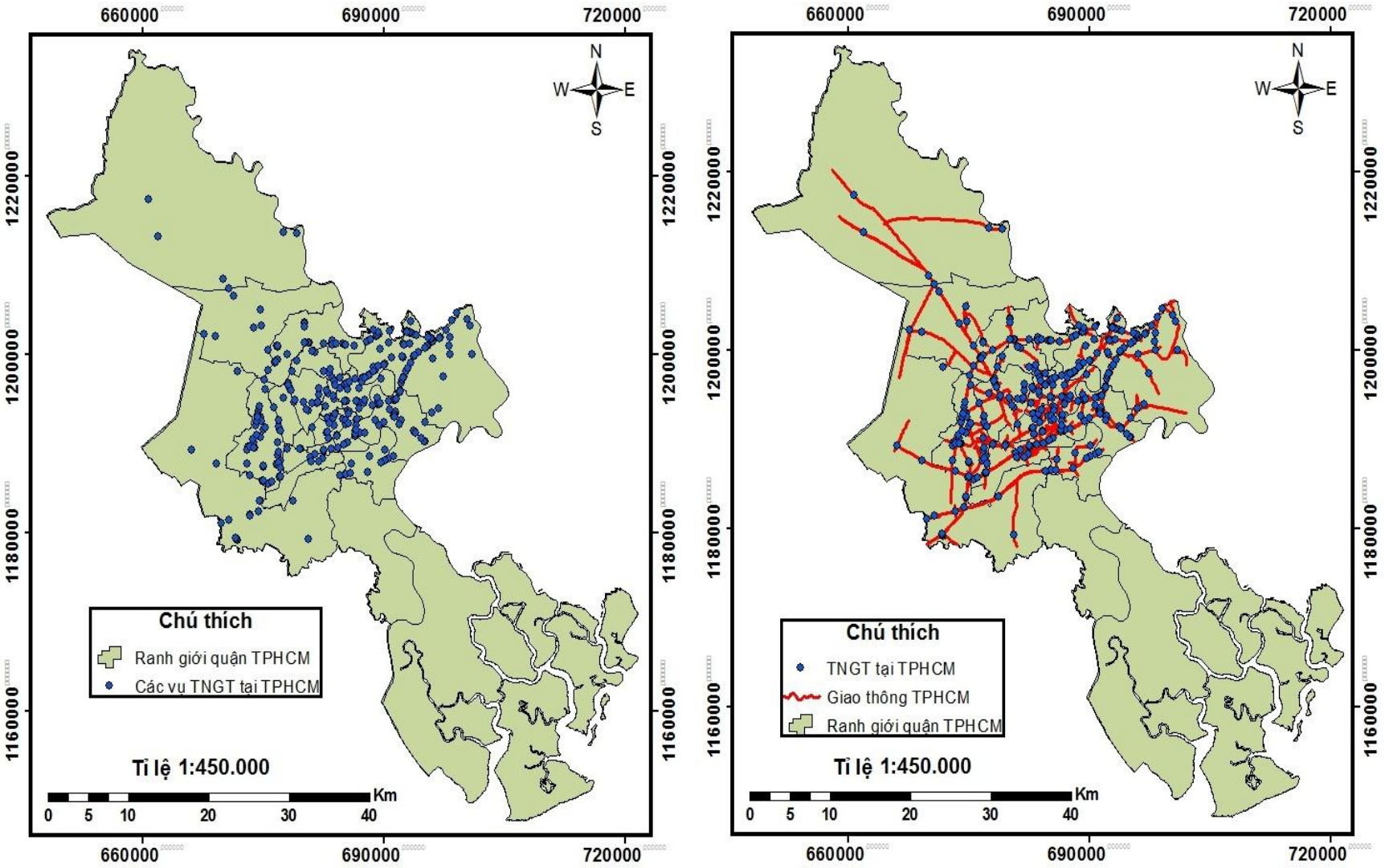
Hình 4.3: Sơ đồ xây dựng dữ liệu không gian

Sau khi đã có được tọa độ của các điểm TNGT, tiếp theo sẽ số hóa các điểm này, đồng thời dựa vào ranh giới hành chính TPHCM cũng như vị trí của các điểm này với đường giao thông nhằm xem xét trường hợp điểm tai nạn bị lệch ra khỏi ranh giới thành phố hoặc không nằm trên đường giao thông.

Tuy nhiên vấn đề nằm ở chỗ tọa độ các điểm TNGT được xác định thông qua Google Map dựa trên thông tin được cung cấp về vị trí xảy ra tai nạn (số nhà, tên đường, phường xã, quận huyện) do đó việc xác định các điểm tai nạn có nằm trên đường giao thông hay không chỉ dừng lại ở mức tương đối, chỉ xem xét với các trường hợp các điểm tai nạn nằm lệch quá xa các tuyến đường mới tiến hành chỉnh sửa. Mặc dù vậy thông tin về tọa độ với trên 95% các điểm đều nằm trên đường giao thông và chỉ chỉnh sửa không quá 15 điểm.

Ngoài ra nhằm làm cho việc quan sát, đánh giá vị trí các điểm tai nạn trên các con đường được dễ dàng hơn nên một số đường giao thông sẽ được lược bỏ bớt.

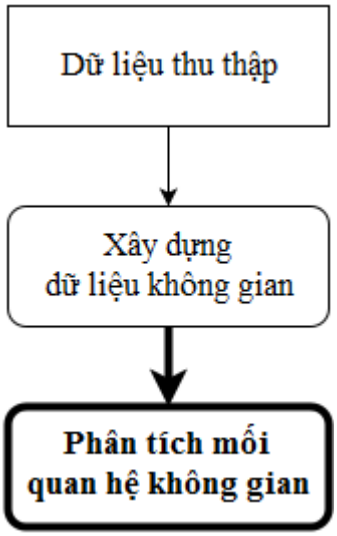
Kết quả số hóa các vụ TNGT được mô tả trong hình 4.2



Hình 4.4: Các vụ TNGT tại TPHCM sau khi được số hóa (trái)

Hệ thống giao thông sau khi đơn giản hóa (phải)

4.3 Giai đoạn 3:



Hình 4.5: Sơ đồ phân tích mối quan hệ không gian

Đây là giai đoạn rất quan trọng vì dữ liệu khi xây dựng không phải 100% các trường dữ liệu đều đầy đủ thông tin các vụ TNGT, do đó sẽ phải loại đi những trường dữ liệu này. Dữ liệu còn lại gồm các trường sau:

Bảng 4.1: Mô tả dữ liệu sau khi chọn lọc

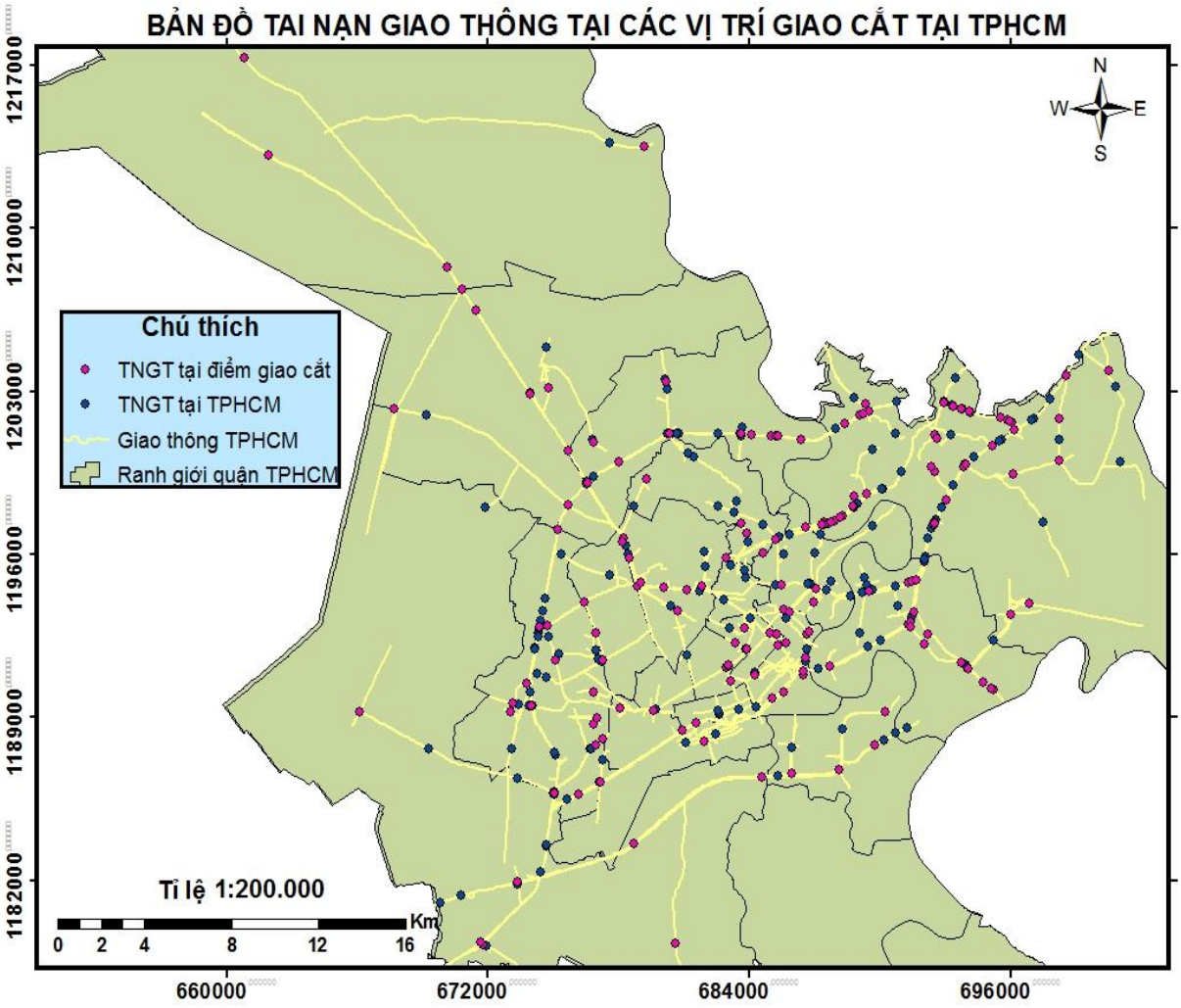
Tên trường	Định dạng dữ liệu	Mô tả
Kinh độ	Decimal Degree (dd)	Kinh độ các vụ TNGT tại TPHCM.
Vĩ độ	Decimal Degree (dd)	Vĩ độ các vụ TNGT tại TPHCM.
Vị trí	Dạng chữ	Mô tả vị trí địa điểm các vụ xảy ra TNGT gồm: Tên đường, tên quận/ huyên, tên tỉnh.
Giờ	Dạng h,m (Với h là giờ và m là phút)	Thời gian xảy ra các vụ TNGT. Ví dụ: 9h45, 14h

Số lượng người bị tai nạn	Dạng number	Tổng số người bị tai nạn trong 1 vụ tai nạn gồm chết, bị thương, không bị thương.
Phương tiện bị tai nạn	Dạng text	Mô tả, phân loại chi tiết các loại xe: xe máy, xe khách, xe buýt,…
Phương tiện gây tai nạn	Dạng text	Mô tả, phân loại chi tiết các loại xe: xe máy, xe khách, xe buýt,…
Tình trạng người bị tai nạn	Dạng text	Mô tả tình trạng của nạn nhân ngay lúc ghi nhận tại hiện trường vụ tai nạn (chết, bị thương, không).
Tình trạng người gây tai nạn	Dạng text	Mô tả tình trạng của người gây tai nạn ngay lúc ghi nhận tại hiện trường vụ tai nạn (chết, bị thương, không).

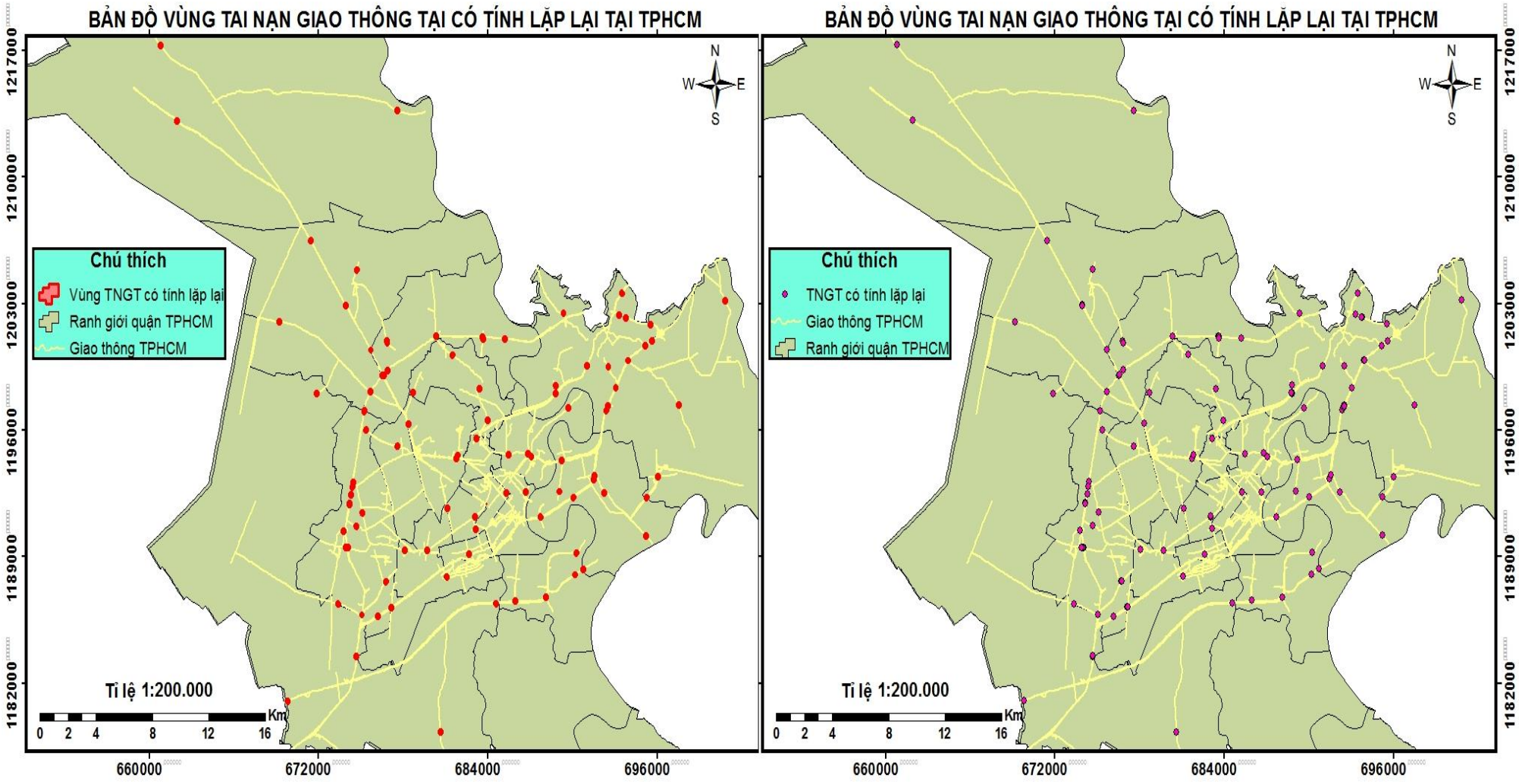
Ngoài các loại dữ liệu đã nêu trên, đề tài còn xây dựng dữ liệu dựa trên những yếu tố cấu trúc hạ tầng giao thông và tần suất xảy ra các vụ TNGT tại mỗi vị trí đã xảy ra TNGT theo thời gian. Để làm được việc này, đề tài căn cứ theo các cơ sở sau

- Cấu trúc hạ tầng giao thông: Nhiều vụ TNGT xảy ra ngoài việc do lỗi người điều khiển (thiếu quan sát, ý thức kém, chạy vượt tốc độ, say xỉn) thì còn xảy ra do cấu trúc giao thông (mặt đường gồ ghề, đường gấp khúc nhiều chỗ, các chỗ giao điểm ngã 3, ngã 4, các giao điểm vòng xoay, điểm giao giữa đường dẫn và đường cao tốc / đại lộ, tín hiệu giao thông). Tuy nhiên nhiều yếu tố do lý khách quan không thể kiểm tra ngoài thực tế như cấu trúc mặt đường hay tín hiệu giao thông do đó đề tài chỉ dựa trên vị trí các điểm TNGT và dữ liệu giao thông TPHCM để phân tích và xây dựng dữ liệu. Dữ liệu được mô tả trong hình 4.4

– Tần suất các vụ TNGT (Tính lặp lại): Nhiều vị trí TNGT xảy ra có tính lạp lại trong một khoảng thời gian nhất định, không cần thiết ngay đúng vị trí cũ mà có thể xảy ra cách vị trí đã xảy TNGT trước đó khoảng 30m, 50m. Do đó đề tài sẽ dựa trên dữ liệu không gian để xác định những vụ TNGT xảy ra lặp lại nhiều lần (trên 2 lần) tại 1 vị trí. Để làm tăng độ chính xác hơn và cũng như để tăng tốc độ phân tích các vụ TNGT có tính lặp lại, đề tài sẽ tạo vùng đệm cho các vụ TNGT với bán kính 25m, sau đó tìm kiếm những vụ tai nạn có vùng đệm bị trùng lắp nhau. Từ đó sẽ chọn ra những vụ TNGT có tính lặp lại. Kết quả được mô tả trong hình 4.5



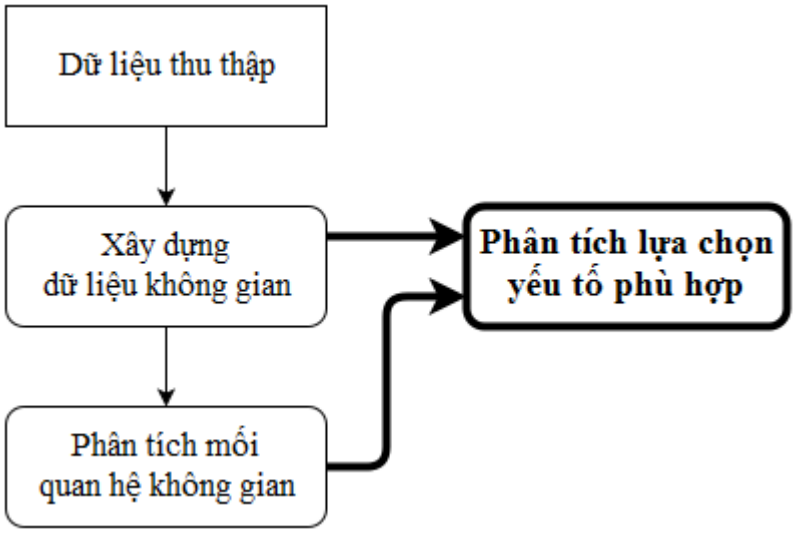
Hình 4.6: Bản đồ TNGT tại vị trí giao cắt tại TPHCM



Hình 4.7: Bản đồ TNGT có tính lặp lại tại TPHCM

(trái) Bản đồ vùng TNGT có tính lặp lại

(phải) Bản đồ TNGT có tính lặp lại tương ứng theo vùng TNGT có tính lặp lại tại TPHCM



Hình 4.8: Sơ đồ phân tích lựa chọn các yếu tố phù hợp

Như vậy sau khi xây dựng tất cả các dữ liệu cần thiết, đề tài sẽ tiến hành đánh giá hệ số tương quan để chọn ra những yếu tố có tính tương quan cao trong số 8 yếu tố (Số lượng người bị tai nạn, Thời gian, Thứ, Khu vực, Giao cắt, Tính lặp lại, Phương tiện, TÌnh trạng) nhằm lấy ra để phân tích mạng neural.

Mặc dù vậy, hệ số tương quan sau khi đánh giá ra giá trị rất thấp, không có giá trị về mặt cơ sở để chọn ra các yếu tố tốt nhất để phân tích mạng neural.

Variable	Coefficient
PHUONG_TIEN	-0.145579
LAP_LAI	-0.012571
KHU_VUC	-0.149214
THU	0.207798
THOI_GIAN	0.169106
GIAO_CAT	-0.019728
TINH_TRANG	0.021927
C	1.534982
R-squared	0.014732

Hình 4.9: Chỉ số tương quan

Do đó đề tài sẽ chuyển hướng thay vì đánh giá tương quan để chọn ra yếu tố tốt nhất để phân tích mạng neural sang tổ hợp các yếu tố và thực hiện phân tích mạng neural sau đó chọn ra tố hợp cho ra mạng neural có kết quả tốt nhất.

Việc tổ hợp sẽ cho ra rất nhiều trường hợp chọn yếu tốt đầu ra và đầu vào như sau:

– Yếu tố đầu vào có: $C_8^1 . C_7^1 . C_6^1$ = 336 cách chọn nếu chọn trước.

– Yếu tố đầu ra có: $C_8^1 . C_7^1$ = 56 cách chọn nếu chọn trước.

Như vậy đề tài sẽ phải làm rất nhiều trường hợp để có được kết quả tốt nhất. Nhưng do hạn chế về mặt thời cũng như nguồn lực không cho phép thực hiện quá nhiều lần. Nên đề tài sẽ tiến hành chọn ngẫu nhiên các yếu tố trên để tạo đầu vào và đầu ra và chỉ tiến hành phân tích mạng neural từ 2 lần tổ hợp.

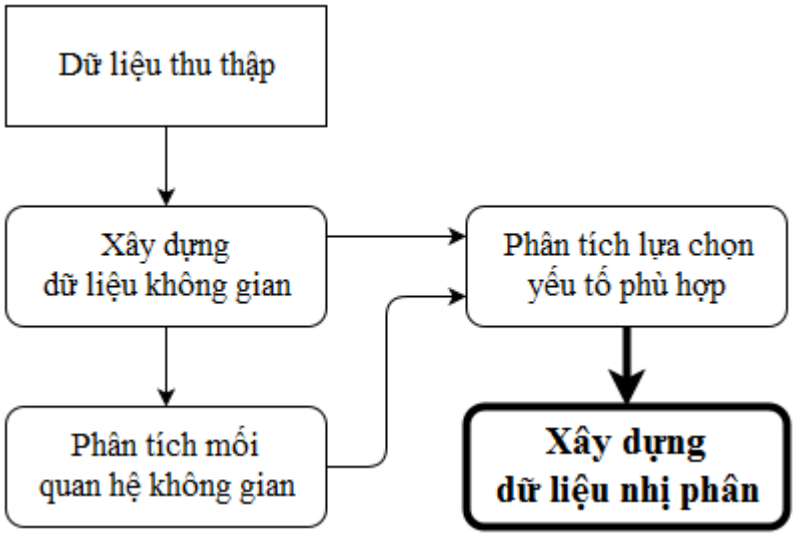
– Lần tổ hợp thứ nhất: Đề tài sẽ chọn yếu tố đầu vào gồm: Thời gian, Thứ, Khu vực. Yếu tố đầu gồm ra: Tính lặp lại và Tình trạng.

– Lần tổ hợp thứ 2: Đề tài sẽ chọn yếu tố đầu vào gồm: Số lượng người bị tai nạn, Giao cắt, Phương tiện. Yếu tố đầu ra gồm: Tính lặp lại và tình trạng.

Nhìn vào đó có thể thấy đề tài lựa chọn đầu ra theo tiêu chí mà nhiều người quan tâm nhất khi nhắc đến các vụ TNGT như: Tình trạng TNGT hiện tại như thế nào hay tại sao TNGT lại ra liên tục tại địa điểm này (điểm đen, điểm nóng). Còn đầu vào đề tài sẽ nhóm theo 2 nguyên nhân gồm:

– Nhóm 1: Nhóm theo yếu tố không gian thời gian

– Nhóm 2: Nhóm theo yếu tố con người và tính chất khu vực xảy ra tai nạn.



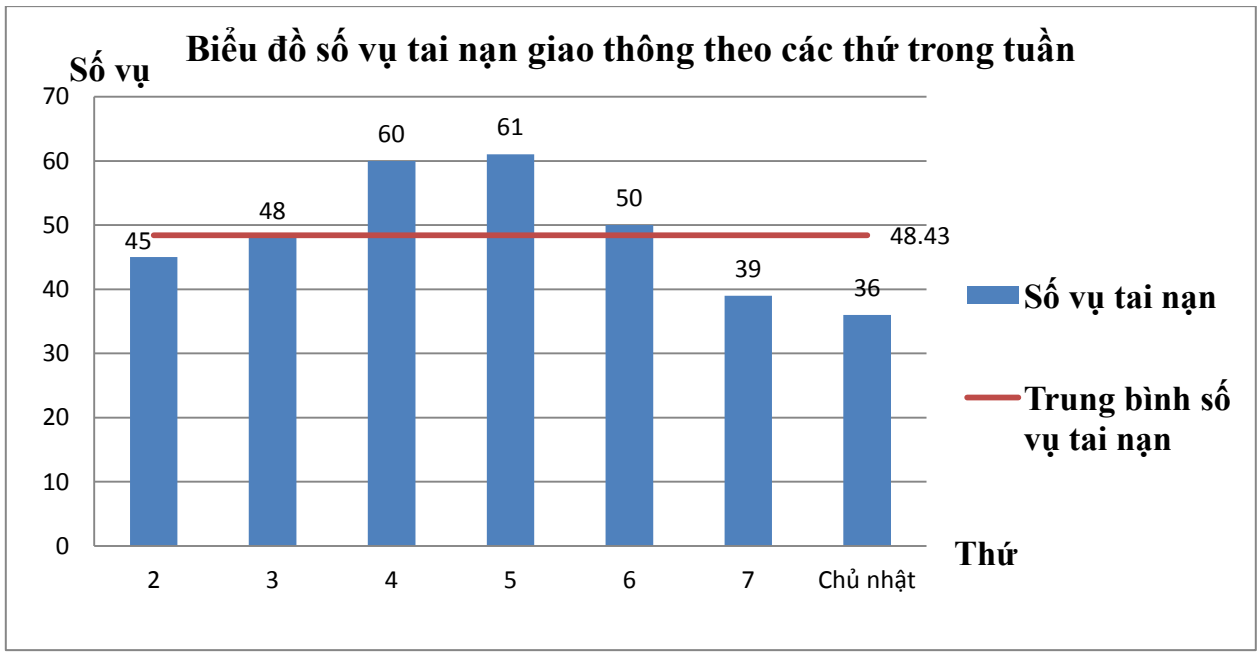
Hình 4.10: Sơ đồ xây dựng dữ liệu nhị phân

Tiếp theo sau khi đã chọn được các yếu tố đầu vào và đầu ra, đề tài sẽ bắt đầu chuyển dữ liệu sang dạng nhị phân gồm 0 và 1. Nhưng để có thể biết được thông số nào của dữ liệu sẽ là 0 dữ liệu nào là 1, đề tài sẽ dựa vào việc đánh giá thống kê sơ bộ từng yếu tố theo dữ liệu đã xây dựng được cùng với kiến thức cá nhân và các thống kê trên của các cơ quan liên quan tại TPHCM.

Trường dữ liệu thứ:

Thống kê sơ bộ dữ liệu xây dựng cho thấy, các vụ TNGT tại TPHCM xảy ra cao nhất vào thứ 4, thứ 5 và thứ 6. Qua đó 3 ngày này trong tuần có số vụ TNGT xảy ra cao hơn mức trung bình cả thành phố (48,43 vụ) lần lượt là: 60 vụ, 61 vụ, 50 vụ. Trong đó thứ 4 và thứ 5 có số vụ tai nạn xảy ra cao vượt trội còn thứ 6 thì chỉ cao hơn mức trung bình khoảng 2 vụ. Các thứ khác trong tuần đều có số vụ TNGT xảy ra ít hơn mức turng bình cả thành phố.

Biểu đồ 4.1: Biểu đồ số vụ TNGT theo các thứ trong tuần

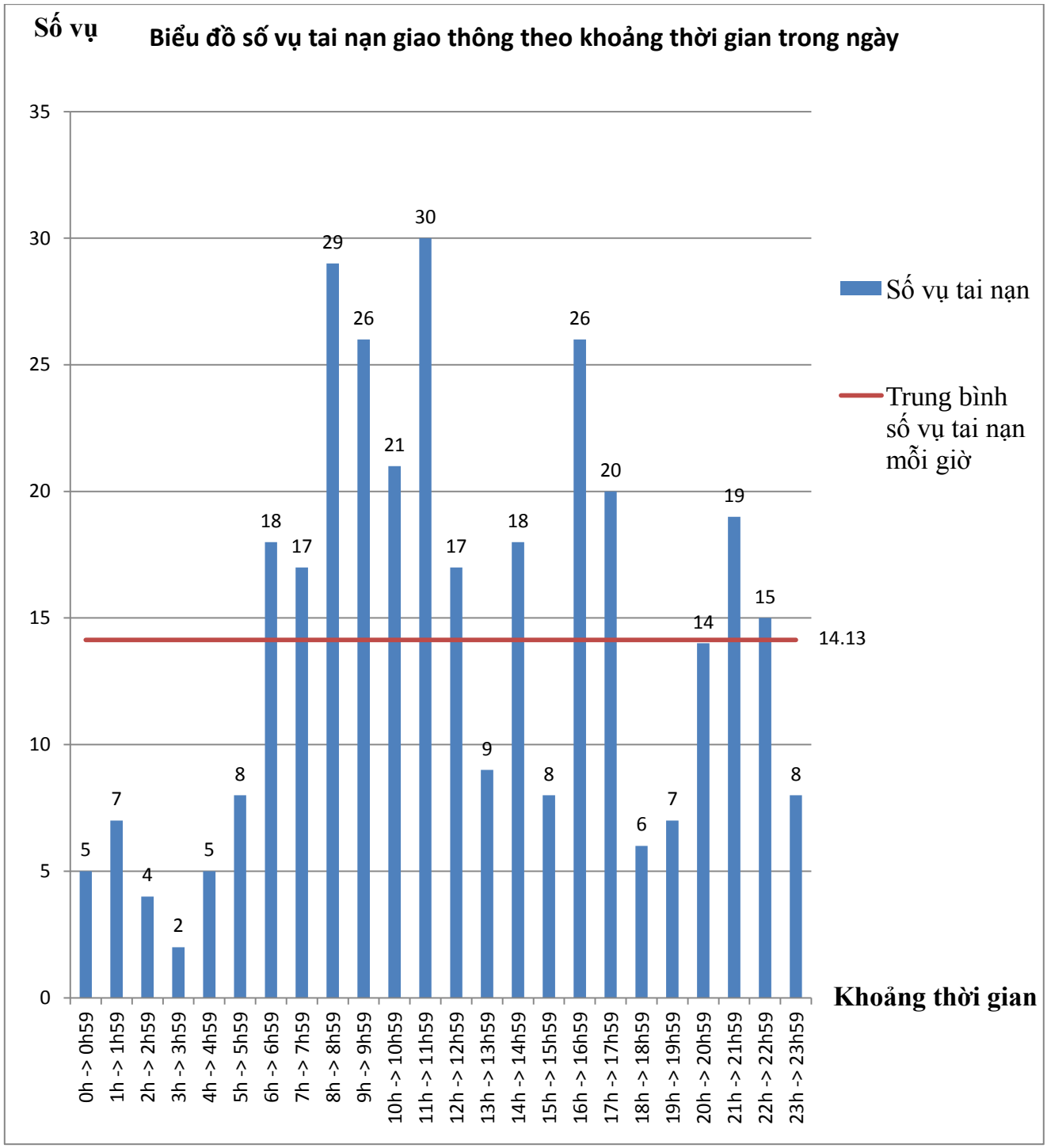


Theo dữ liệu giờ

Thống kê dữ liệu cho ra kết quả các vụ TNGT tại TPHCM xảy ra nhiều nhất vào các khoảng thời gian từ 6h sang đến 13h trưa, 14h đến 15h, 16h đến 18h và 21h đến 22h

với số vụ tai nạn xảy ra từ 15 vụ đến 30 vụ cao hơn mức trung bình 14,13 vụ của thành phố.

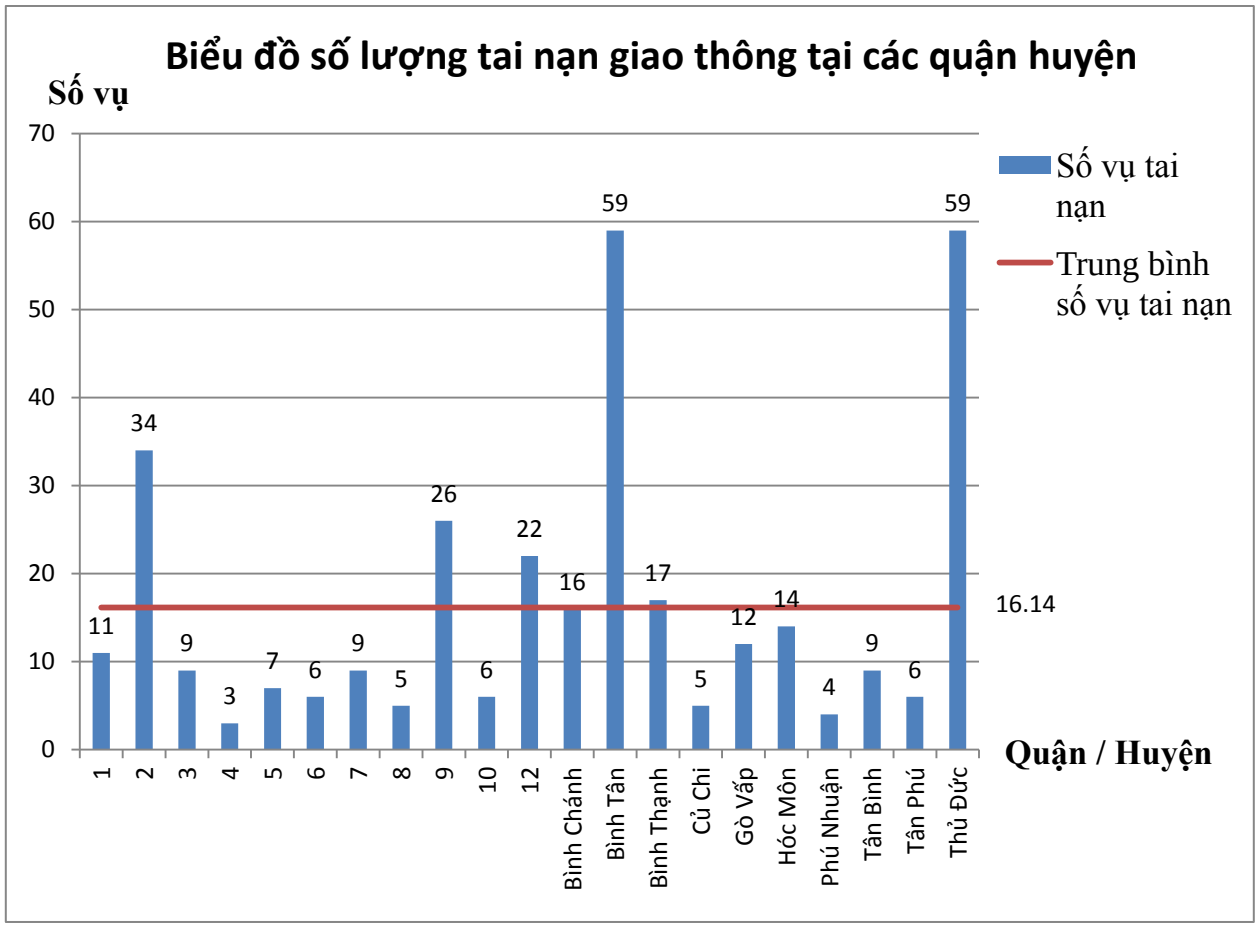
Biểu đồ 4.2: Biểu đồ số vụ TNGT theo khoảng thời gian trong ngày tại TPHCM



Theo dữ liệu khu vực

Thống kê cho thấy các khu vực gồm quận 2, 9, 12, Bình Tân, Bình Thạnh, Thử Đức có số vụ TNGT xảy ra nhiều hơn mức trung bình của thành phố 16,14 vụ lần lượt là: 34,26,22, 59, 17, 59 vụ.

Biểu đồ 4.3: Biểu đồ số lượng TNGT tại các quận huyện tại TPHCM



Theo dữ liệu phương tiện, tình trạng, số lượng

Đề tài sẽ gom nhóm 3 trường dữ liệu này như sau:

- Số lượng: Nhóm 1 có tổng số người bị tai nạn <2 người và nhóm 2 có tổng số người bị tai nạn >= 2 người.
- Phương tiện: Đề tài sẽ chia thành cùng loại nếu phương tiên của người bị tai nạn và người gây ra tai nạn giống như. Khác loại nếu phương tiện của người bị tai nạn và gây tai nạn khác nhau.
- Tình trạng: Nếu tai nạn có người tử vong (cả bên bị tai nạn và gây tai nạn) thì sẽ nhóm vào 1 nhóm và ngược lại.

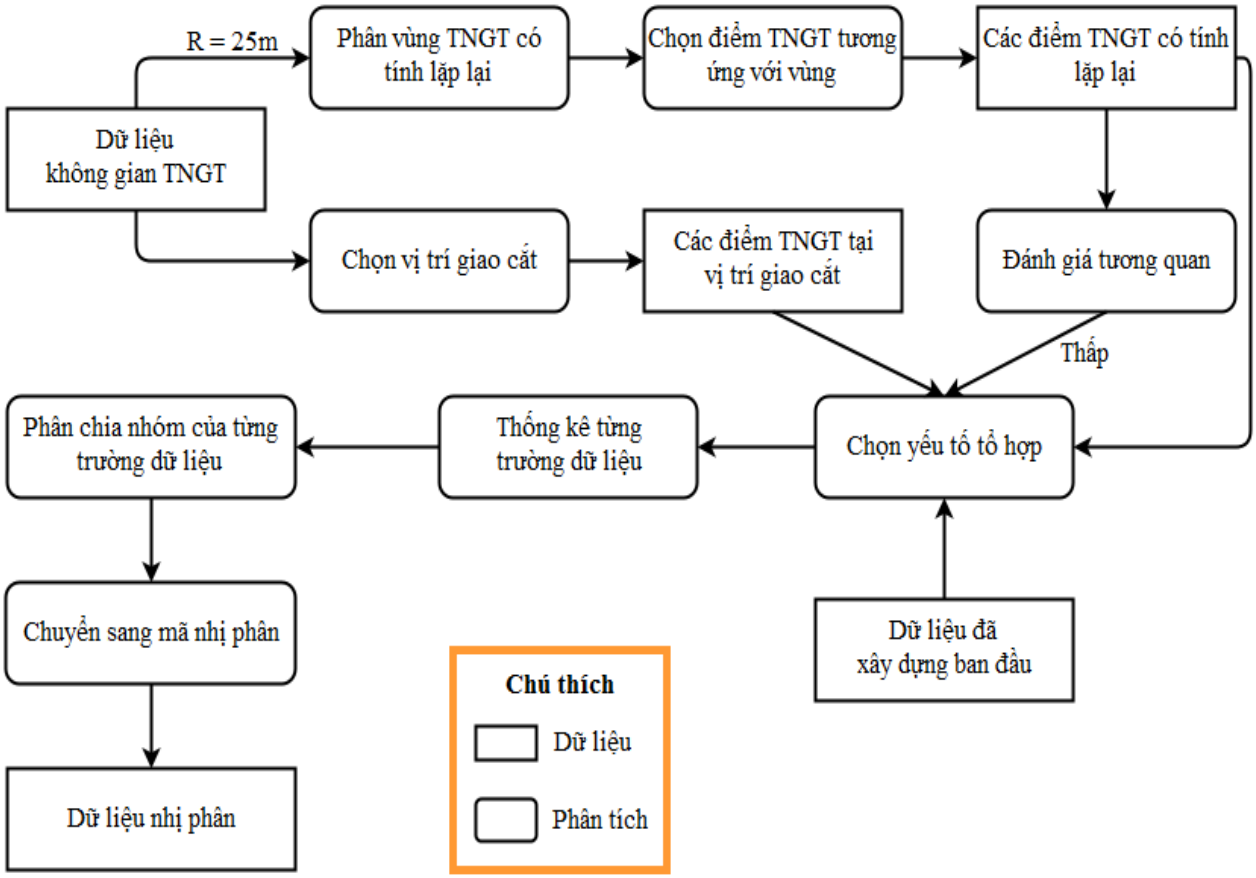
Như vậy, việc phân chia và chuyển sang mã nhị phân có thể được tóm gọn theo bảng 4.2

Bảng 4.2: Bảng tóm tắt sơ sở chuyển dữ liệu sang nhị phân

Trường dữ liệu	0	1
Số lượng người bị tai nạn	Nhóm >= 2 người	Nhóm < 2 người
Tình trạng	Nhóm không có người chết	Nhóm có người chết
Phương tiện	Nhóm khác loại phương tiện	Nhóm cùng loại phương tiện
Thứ	Nhóm giờ xảy ra số TNGT thấp	Nhóm giờ xảy ra TNGT cao
Giờ	Nhóm thứ xảy ra số TNGT thấp	Nhóm thứ xảy ra số TNGT cao
Khu vực	Nhóm khu vực xảy ra số TNGT thấp	Nhóm khu vực xảy ra số TNGT cao
Giao cắt	Nhóm điểm TNGT xảy ra khu vực không có giao cắt	Nhóm điểm TNGT xảy ra khu vực có giao cắt
Tính lặp lại	Nhóm điểm xảy ra TNGT không có tính lặp lại	Nhóm điểm xảy ra TNGT có tính lặp lại

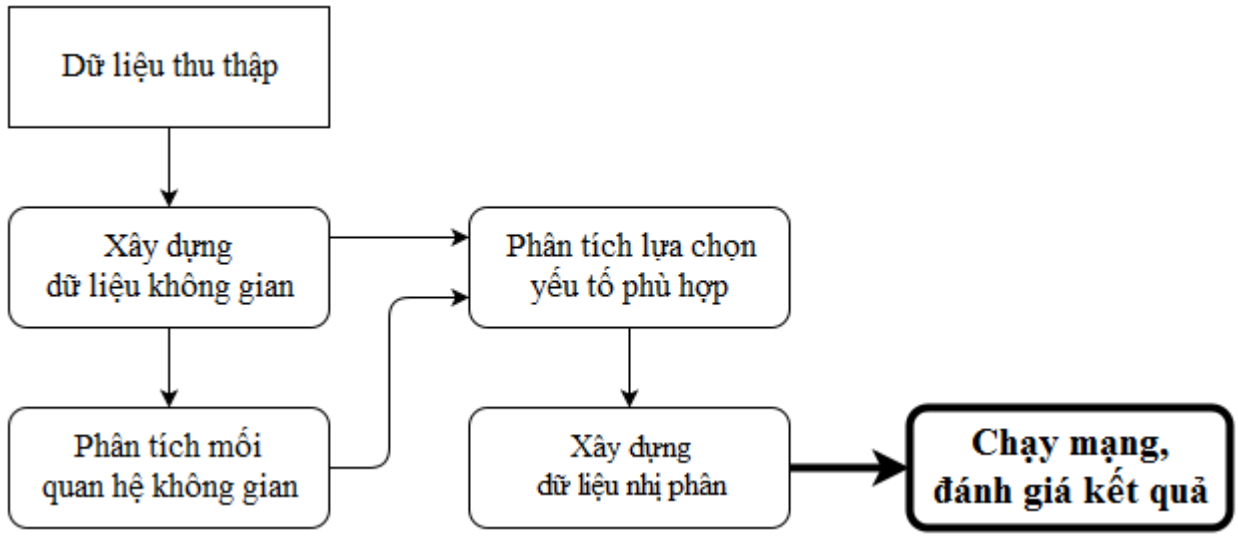
Thông tin về dữ liệu sau khi đã chuyển sang mã nhị phân xin xem phần phụ lục.

Như vậy qua trinh thực hiện trong giai đoạn 3 sẽ tóm tắt qua sơ đồ phương pháp chi tiết sau:



Hình 4.11: Sơ đồ phương pháp chi tiết thực hiện trong giai đoạn 3

4.4 Giai đoạn 4



Hình 4.12: Sơ đồ phương pháp chạy mạng, đánh giá kết quả

Sau khi đã có dữ liệu nhị phân, đề tài sẽ tiếp tục phân tích mạng neural.

Về vấn đề chọn số đơn vị trong lớp ẩn thì căn cứ vào các vấn đề đã nêu ở chương 2, việc chọn số lớp mạng tùy thuộc vào mỗi bài, mỗi yếu tố khác nhau. Do đó trong bài luận, đề tài sẽ chọn số lớp ẩn dao động từ bằng số lớp đầu ra (2 lớp) cho đến 3 lần tổng của đầu ra và vào.

Nên số lớp ẩn mà đề tài chọn sẽ nằm trong khoảng 2 đến 15 lớp.

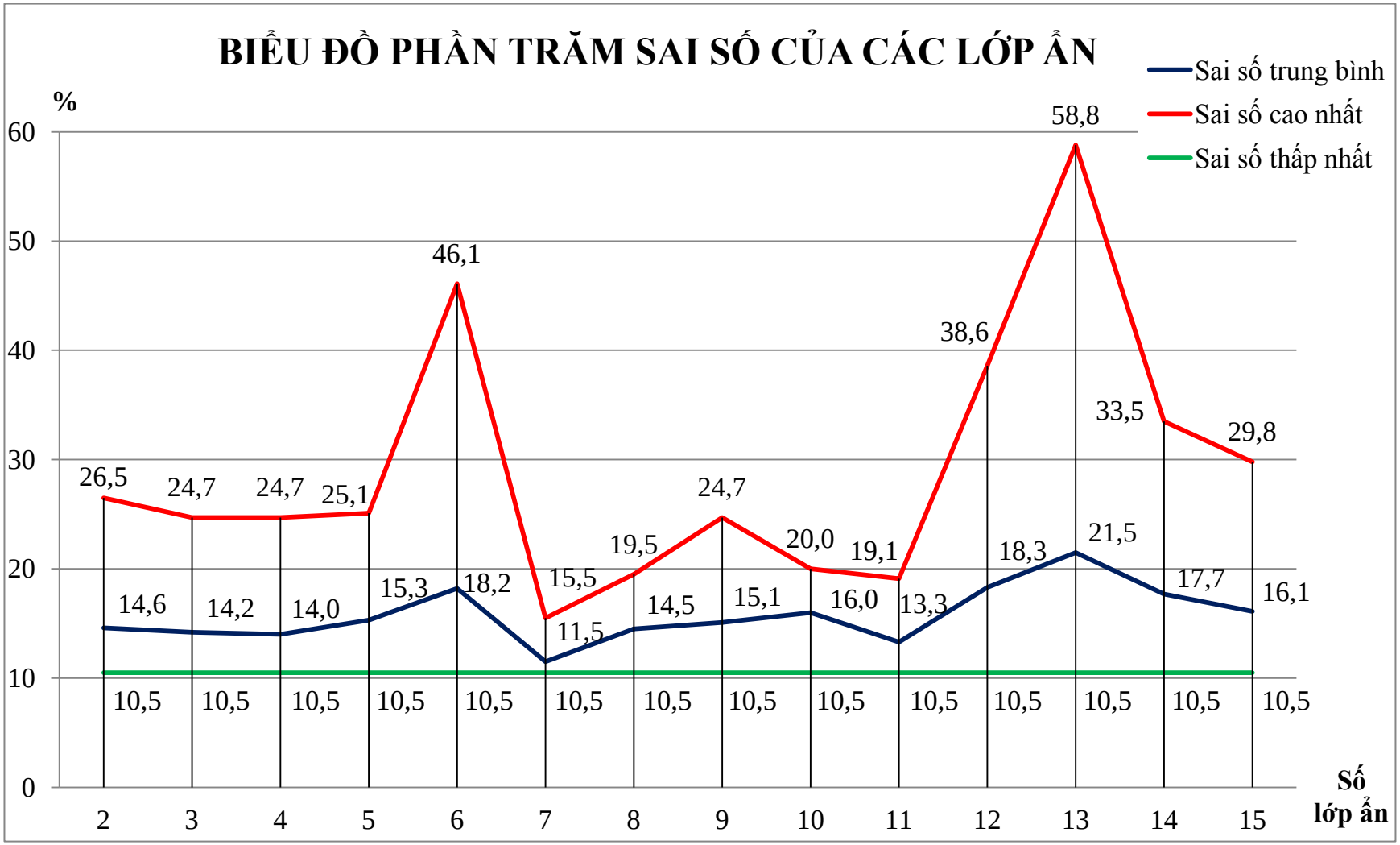
Đề tài sẽ tiến hành chạy mạng khoảng 10 lần cho mỗi lớp và lấy trung bình để cho ra kết quả tương đối chính xác hơn.

Đề tài sẽ dùng phần mêm MATLAB để phân tích mạng neural. Chi tiết quá trình thực hiện phân tích bằng công cụ mạng neuron trên Matlab được trình bày chi tiết trong phụ lục.

- **Nhóm dữ liệu thứ nhất:**
  - Đầu vào: Giờ, Thứ, Khu vực.
  - Đầu ra: Tính lặp lại, Tình trạng.

  Bảng quá trình phân tích sai số lớp ẩn xin xem phụ lục

Biểu đồ 4.4: Biểu đồ phân trăm sai số của các lớp ẩn



Bảng 4.3: Bảng biến động sai số của các lớp ẩn

Số lớp ẩn	Biến động sai số (%)
2	16,0
3	14,2
4	14,2
5	14,6
6	35,6
7	5,0
8	9,0
9	14,2
10	9,5
11	8,6
12	28,1
13	48,3
14	23,0
15	19,3

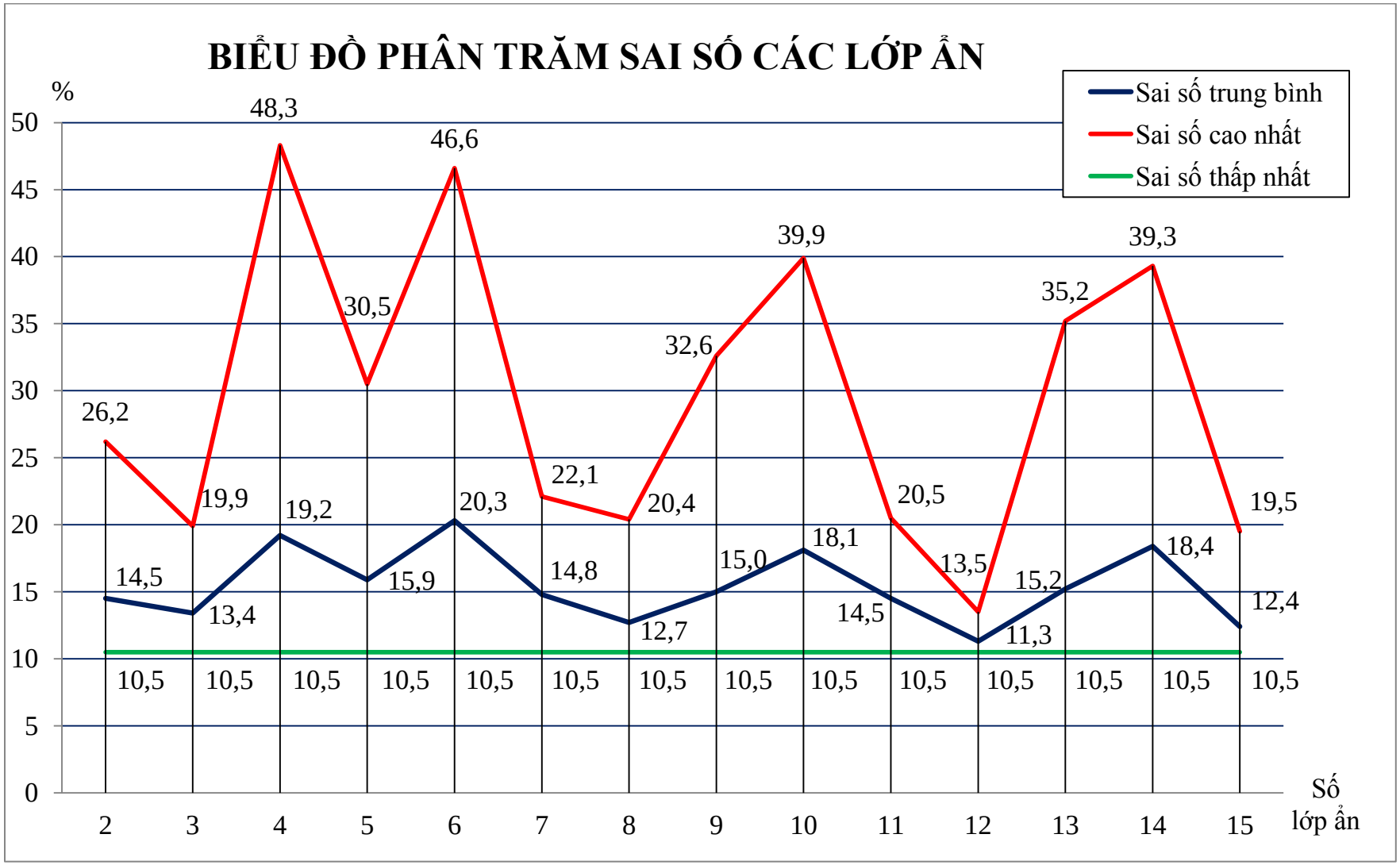
Kết quả đánh giá sai số của các lớp ẩn cho thấy quá trình thực hiện các sai số trung bình đều thấp hơn 22%. Trong số 14 cách chọn lớp ẩn nhận thấy duy nhất chỉ có cách chọn 7 lớp ẩn là có sai số trung bình thấp nhất với 11,5%. Do đó đây là cách chọn lớp phù hợp nhất trong số 14 cách chọn. Ngoài ra sai số thấp nhất cảu tất cả các lớp ẩn đều bằng nhau 10,5%. Điều này cho thấy được sai số thấp nhất dường như là cố định không bị ảnh hưởng bởi cách chọn số lớp ẩn.

Nếu xét theo góc độ biến động giữa sai số lớn nhất và sai số thấp nhất thì cách chọn 7 lớp ẩn vẫn là tốt nhất do biến động là bé nhất chỉ 5% biến thiên từ 10,5% đến 15,5%. Trong khi cách chọn 13 lớp ẩn thì độ biến động lớn nhất đến 48,3% biến thiên từ 10,5% đến 58,8%.

- **Nhóm dữ liệu thứ hai:**
  - Đầu vào: Số lượng người bị nạn, Phương tiện, Giao cắt.
  - Đầu ra: Tính lặp lại, Tình trạng.

  Bảng quá trình phân tích sai số lớp ẩn xin xem phụ lục

Biểu đồ 4.5: Biểu đồ phần trăm sai số của các lớp ẩn



Bảng 4.4: Bảng biến động sai số của các lớp ẩn

Số lớp ẩn	Biến động sai số (%)
2	15,7
3	9,4
4	37,8
5	20,0
6	36,1
7	11,6
8	9,9
9	22,1
10	29,4
11	10,0
12	3,0
13	24,7
14	28,8
15	9,0

Kết quả đánh giá sai số của các lớp ẩn cho thấy quá trình thực hiện các sai số trung bình đều thấp hơn 21%. Trong số 14 cách chọn lớp ẩn nhận thấy duy nhất chỉ có cách chọn 12 lớp ẩn là có sai số trung bình thấp nhất với 11,3%. Do đó đây là cách chọn lớp phù hợp nhất trong số 14 cách chọn. Ngoài ra sai số thấp nhất cảu tất cả các lớp ẩn đều bằng nhau 10,5%. Điều này cho thấy được sai số thấp nhất dường như là cố định không bị ảnh hưởng bởi cách chọn số lớp ẩn.

Nếu xét theo góc độ biến động giữa sai số lớn nhất và sai số thấp nhất thì cách chọn 12 lớp ẩn vẫn là tốt nhất do biến động là bé nhất chỉ 3,0% biến thiên từ 10,5% đến 13,5%. Trong khi cách chọn 4 lớp ẩn thì độ biến động lớn nhất đến 37,8% biến thiên từ 10,5% đến 48,3%.

- **Kết luận sau 4 giai đoạn:**

Trong quá trình phân tích mạng neural dựa trên 2 lựa chọn tổ hợp khác nhau, cho thấy sai số thấp nhất của các lớp ẩn đều không phụ thuộc vào cách chọn số lớp ẩn, đồng thời các kết quả sai số trung bình của từng cách chọn của mỗi tổ hợp đều cho ra kết quả như mong muốn với sai số có thể chấp nhận được.

Tuy nhiên như đã nêu ở trên, do thời gian hạn chế và điều kiện không cho phép dẫn đến chưa thể thử hết các tổ hợp. Do đó mặc dù sai số của 2 tổ hợp lựa chọn ngẫu nhiên này có thể tạm chấp nhận trong đề tài nhưng chưa hẳn đã tốt hơn so với các tổ hợp chưa thử khác.

Dựa trên phân tích, đánh giá và nhận xét sai số của 2 tổ hợp nói trên, đề tài sẽ dựa vào 2 yếu tố là sai số trung bình thấp nhất, mức độ biến động của từng tổ hợp để chọn loại tổ hợp tốt nhất. Nên sau khi xem xét, đánh giá thì đề tài sẽ chọn tổ hợp thứ 2 là loại tổ hợp tốt nhất của đề tài. Vì sai số trung bình thấp nhất của tổ hợp này thấp hơn (11,3%) so với của tố hợp đầu (11,5%). Đồng thời biến động sai số của tổ hợp này vẫn thấp hơn (3,0%) và ổn định hơn khi phần trăm biến động sai số chỉ dao động từ 3,0% đến 37,8% so với tổ hợp thứ nhất lần lượt là 5% và 5% đến 48,3%.

CHƯƠNG 5
KẾT LUẬN

5.1 Kết luận

Thông qua kết quả xây dựng dữ liệu, phân tích mạng neuron, đề tài nhận thấy sai số mặc dù có thể chấp nhận được cho tất cả các cách chọn số lớp ẩn của 2 loại tổ hợp mà đề tài đã chọn nhưng xét về mặt dữ liệu cũng như trong quá trình xây dựng vẫn còn vài điểm chưa ổn định. Điều này có thể giải thích như sau:

- Đến từ dữ liệu: Đánh giá lại dữ liệu thu thập được cho thấy mặc dù dữ liệu thu thập, phân loại đầy đủ nhưng vẫn tồn tại yếu tố khách quan đến từ người thu thập trực tiếp tại hiện trường các vụ TNGT cũng như chuyển dữ liệu trên giấy có khả năng chưa chuẩn xác hoặc kiến thức người thu thập chưa đánh giá đúng dẫn đến sai lệch dữ liệu từ ban đầu.
- Đến từ bản thân người thực hiện xây dựng, phân tích, chọn lọc dữ liệu: Việc thực hiện số hóa dữ liệu xây dựng lại dữ liệu cho phù hợp cũng như phân tích, chọn lọc các yếu tố phù hợp để thực hiện đề tài đòi hỏi cần có kiến thức tổng quát về tình hình TNGT tại TPHCM cũng như khả năng phân tích khối lượng lớn thông tin, kinh nghiệm vẫn còn hạn chế.

Việc chọn kết quả cấu hình mạng mặc dù lấy được kết quả như mong muốn tuy nhiên vẫn còn mặt hạn chế là đề tài không thử hết được nên việc lựa chọn này có thể vẫn chưa phù hợp thật sự vì khả năng còn loại tổ hợp khác sẽ cho ra kết quả tốt hơn và biến động có thể ổn định hơn rất nhiều so với loại tổ hợp mà đề tài đã chọn.

Ngoài ra, lượng thông tin dữ liệu về TNGT tại TPHCM mà bài luận đã xây dựng vẫn chưa thực sự mô tả hết về tình hình TNGT tại TPHCM vì số lượng TNGT tại thành phố thì rất nhiều theo như mô tả của các cơ quan tại TPHCM được nêu ra ở chương 1 trong khi số lượng TNGT của đề tài thì chỉ dừng lại 339 vụ tai nạn. Điều này có thể giải thích vì nhiều vụ TNGT đề tài không có khả năng thu thập, điều tra cũng như thời gian

hạn chế khiến đề tài chỉ có thể chọn những vụ TNGT trọng điểm tại TPHCM để đưa vào nghiên cứu và phân tích.

Kết quả của đề tài có thể được sử dụng cho các cơ quan để tham khảo nguyên nhân cũng như khả năng nhận dạng nhanh chóng các vụ TNGT để từ đó đưa ra các quyết định phù hợp nhất. Đồng thơi kết quả của đề tài có thể được sử dụng để tham khảo cho một đề tài khác cũng sử dụng mạng neural để nhận dang một loại dữ liệu khác.

Kết quả của đề tài sẽ tốt hơn nếu dữ liệu được cung cấp đầy đủ, thông tin dữ liệu chính xác và có thêm thời gian để tính toán ra cấu hình phù hợp nhất có thể cho loại dữ liệu TNGT tại TPHCM.

5.2 Cấu hình mạng của đề tài

Như đã nêu ở chương 5, đề tài sẽ chọn cấu hình của loại tổ hợp thứ 2 mà đề tài đã xây dựng. Đề tài xin đưa ra cấu hình mạng phù hợp nhất dựa theo dữ liệu, kết luận và các kết quả phân tích đã nêu ra ở trên như sau:

Bảng 5.1: Bảng cấu hình mạng của đề tài

Tên	Mô tả
Đầu vào (lớp huấn luyện)	Số lượng người bị nạn Phương tiện Giao cắt
Đầu ra (mục tiêu)	Tính lặp lại Tình trạng
Số lớp ẩn	12 lớp
Số vụ TNGT cần nhận diện	339 vụ
Số vụ nhận diện được	~ 300 vụ

Dữ liệu để huấn luyện	90%
Dữ liệu để nhận dạng	5%
Dữ liệu để kiểm tra	5%
Khả năng nhận dạng TNGT	88,7%
Sai số trung bình (khả năng bỏ sót)	11,3%
Sai số thấp nhất	10,5%
Sai số cao nhất	13,5%
Biến động sai số	3,0%

5.3 Khả năng mở rộng của đề tài

Vấn đề sử dụng mạng neural nhân tạo đã được đề cập và được sử dụng nhiều lĩnh vực khác nhau. Với khả năng của GIS đã nêu ở chương 1 cũng như kết quả mà đề tài đã thực hiện được thì quá trình mở rộng cả đề tài có thể sẽ thêm nhiều yếu tố mới như yếu tố về quê quán, tình hình kinh tế - xã hội, tình trạng cơ sở hạ tầng giao thông, nhu cầu cũng như ý thức của người dân từ đó khả năng nhận dạng cũng như đánh giá về TNGT tại TPHCM sẽ chính xác và gần gũi với thực tế hơn.

Đồng thời, việc áp dụng dữ liệu của các tỉnh khác hay mở rộng ra dữ liệu TNGT ra cả nước vẫn có thể áp dụng bằng phương pháp thực hiện của đề tài để.

Cuối cùng với sự phát triển của công nghệ thông tin và việc áp dụng phương pháp của đề tài có thể triển khai những ứng dụng giúp mọi người có thể nắm bắt tình hình TNGT của cả nước và tích hợp thêm viễn thám cùng thế mạnh của AI sẽ dự báo được các

loại hình TNGT có thể xảy ra tại các khu vực khác nhau tại TPHCM nói riêng cũng như cả nước nói chung.

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Bộ Giao thông Vận tải, 2015.*Tiêu chuẩn quốc gia về kiến trúc hệ thống giao thông thông minh*. Hà Nội.

[2] Bộ ngoại giao. *Một số thông tin về Việt Nam*. Cổng thông tin điện tử chính phủ nước Cộng hòa Xã hội Chủ nghĩa Việt Nam. Địa chỉ: < http://www.chinhphu.vn/portal/page/portal/chinhphu/NuocCHXHCNVietNam/ThongTinTongHop/dialy >, Ngày truy cập [16/05/2016].

[3] Công ty Almec và Công ty TNHH Nippon Koei, 03/2009. *Nghiên cứu quy hoạch tổng thể an toàn giao thông đường bộ tại nước Cộng hòa Xã hội Chủ nghĩa Việt Nam đến năm 2020, Báo cáo cuối kỳ, tập 1: Phân tích.* Cơ quan hợp tác quốc tế Nhật Bản (JICA), Ủy ban an toàn giao thông quốc gia Việt Nam (NTSC).

[4] Dư Phước Tân, 2013. *Cấu trúc đô thị tphcm và các nguyên nhân tác động gia tăng sử dụng xe gắn máy – thử tìm mối quan hệ*. Viện Nghiên cứu phát triển TPHCM. Địa chỉ: < http://cti.gov.vn/bantin/noidung.php?id=49 >, Ngày truy cập [19/4/2016].

[5] Hồ Tú Bảo, 2000. *Nhìn lại 25 năm phát triển ngành trí tuệ nhân tạo*. Phòng nhận dạng và Công nghệ tri thức Viện Công nghệ thông tin & Phòng thí nghiệm Phương pháp luận Sáng tạo Tri thức Viện Khoa học và Công nghệ tiên tiến Nhật Bản. Địa chỉ: < http://www.jaist.ac.jp/~bao/Writings/AI25years.pdf >, Ngày truy cập: [10/05/2016].

[6] Hồ Tú Bảo, 9/2008. *50 năm trí tuệ nhân tạo*. Viện Khoa học và Công nghệ Việt Nam, Viện Khoa học và Công nghệ Nhật Bản. Địa chỉ: < http://www.jaist.ac.jp/~bao/Writings/AI50years.pdf >, Ngày truy cập: [10/05/2016].

[7] Lê Thị Cẩm Bình. *Trí tuệ nhân tạo – Một phương diện của văn hóa ứng dụng.* Nghiên cứu văn hóa, Số 5, Tạp chí nghiên cứu văn hóa trường Đại hoa văn hóa Hà Nội. Địa

chỉ: <http://huc.edu.vn/vi/spct/id150/TRI-TUE-NHAN-TAO---MOT-PHUONG-DIEN-CUA-VAN-HOA-UNG-DUNG/ >, Ngày truy cập: [10/05/2016].

[8] Minh Quyết, 17/11/2014. *Tai nạn giao thông khủng khiếp hơn bom đạn chiến tranh.* Báo vtc online. Địa chỉ: < http://vtc.vn/tai-nan-giao-thong-khung-khiep-hon-bom-dan-chien-tranh.2.516308.htm >, Ngày truy cập [20/3/2016].

[9] Nguyễn Đình Thúc, Hoàng Đức Hải, *2000, Giáo trình mạng trí tuệ nhân tạo mạng Nơron: Phương pháp và ứng dụng*, NXB Giáo Dục.

[10] Nguyễn Quang Hoan, 2007. *Nhập môn trí tuệ nhân tạo.* Học nghệ bưu chính viễn thông, Hà Nội.

[11] Quốc Toản, 28/02/15. *Tai nạn giao thông ở Việt Nam: Đâu là con số thực?*. Báo giáo dục. Địa chỉ: < http://giaoduc.net.vn/Xa-hoi/Tai-nan-giao-thong-o-Viet-Nam-Dau-la-con-so-thuc-post155891.gd >, Ngày truy cập [20/3/2016].

[12] Trần Đức Minh, 2002. *Mạng nơron truyền thẳng và ứng dụng trong dự báo dữ liệu*, Khoa Công nghệ - ĐHQG HN.

Tiếng Anh

[13] Chin-Teng Lin, C.S. George Lee, 1996. *Neural fuzzy systems: a neuro- fuzzy synergism to intelligent systems*, Prentice-Hall Inc.

[14] F. Rezaie Moghaddam1, Sh. Afandizadeh2, M. Ziyadi1, 2010, *Prediction of accident severity using artificial neural networks*. International Journal of Civil Engineering, page 41 – 48. [20]

[15] Francisca Nonyelum Ogwueleka, 2014. *An Artificial Neural Network Model for Road Accident Prediction: A Case Study of a Developing Country*. Acta Polytechnica Hungarica, Vol. 11, No. 5, 2014, page 177. [19]

[16] Guido van Rossum, 27/01/2013. Programming at Python Speed. Địa chỉ: < http://www.artima.com/intv/speed.html >, Ngày truy cập: [11/05/2016].

[17] Kermit Sigmon, 1992. *MATLAB Primer*. Department of Mathematics, University of Florida.

[18] Miao M. Chong, Ajith Abraham, Marcin Paprzycki. *Traffic accident analysis using decision trees and neural networks*. Computer Science Department, Oklahoma State University, USA. [18]

[19] National Center for Statistics And Analysis. *2012 Motor Vehicle Crashes: Overview*. 1200 New Jersey Avenue SE., Washington, DC 20590.

[20] National Center for Statistics And Analysis. *Traffic Safety Facts 2003 Data*. 400 Seventh St., S.W., Washington, D.C. 20590.

[21] Turing, A.M, 1950. *Computing machinery and intelligence*. Mind, 59, 433-460. Địa chỉ: < http://www.loebner.net/Prizef/TuringArticle.html >, Ngày truy cập: [10/05/2016].

[22] Volvo Truck, 09/01/2013. *European Accident Research and Safety Report 2013*.

PHỤ LỤC

- **Phụ lục 1:**

Thông tin mẫu chi tiết về dữ liệu sau khi xây dựng, phân tích và chọn lọc:

STT	Kinh độ	Vĩ độ	Giao cắt	Tính lặp lại	Tình trạng người bị TN	Khu vực	Số người bị TN	Phương tiện bị TN	Giờ	Thứ	Phương tiện gây TN	Tình trạng người gây TN
1	106,685918	10,755617	Không	Không	chết	5	1	xe máy	9h30	3	xe máy	chết
2	106,654816	10,746549	Có	Không	bị thương	5	1	xe máy	16h	1	xe ô tô	bị thương
3	106,692389	10,860718	Có	Không	không	9	10	xe khách	9h	1	xe tải	bị thương
4	106,599356	10,879769	Có	Không	chết	Hóc Môn	1	xe máy	11h	6	xe ô tô	bị thương
5	106,59441	10,784736	Không	Không	bị thương	Bình Tân	4	xe máy	7h	7	xe tải	bị thương
6	106,601561	10,737128	Không	Không	bị thương	Bình Tân	1	xe máy	14h	1	xe tải	bị thương
7	106,708795	10,803033	Không	Có	chết	Bình Thạnh	1	xe máy	9h15	2	xe khách	bị thương
8	106,708795	10,803033	Không	Có	bị thương	Bình Tân	9	xe khách	17h	3	xe khách	bị thương
9	106,708795	10,803033	Không	Có	chết	Bình Tân	1	xe khách	17h	3	xe khách	bị thương
10	106,708795	10,803033	Không	Có	chết	Bình Tân	1	xe khách	17h	3	xe khách	bị thương
11	106,61615	10,739352	Không	Có	bị thương	6	1	xe máy	5h20	4	xe tải	bị thương
12	106,760961	10,826641	Có	Có	không	9	1	xe ô tô	17h	5	xe tải	bị thương
13	106,760961	10,826641	Có	Có	không	9	2	xe ô tô	17h	5	xe tải	bị thương
14	106,760961	10,826641	Có	Có	bị thương	9	1	xe ô tô	17h	5	xe tải	bị thương
15	106,760961	10,826641	Có	Có	bị thương	9	1	xe ô tô	17h	5	xe tải	bị thương
16	106,600921	10,722957	Có	Có	không	Bình Tân	1	xe tải	11h	6	xe tải	bị thương
17	106,595045	10,691748	Không	Không	bị thương	Bình Chánh	1	xe máy	16h	7	xe tải	bị thương

18	106,660159	10,85267	Không	Có	chết	Gò Vấp	2	xe máy	11h	1	xe tải	bị thương
19	106,660159	10,85267	Không	Có	bị thương	Gò Vấp	4	xe máy	11h	1	xe tải	bị thương
20	106,750019	10,803354	Có	Không	không	2	0	xe ô tô	14h30	2	xe khách	bị thương
21	106,747146	10,846728	Không	Có	bị thương	Thủ Đức	1	xe máy	10h	3	xe máy	bị thương
22	106,747146	10,846728	Không	Có	bị thương	Thủ Đức	1	xe máy	10h	3	xe máy	bị thương
23	106,685183	10,769091	Không	Không	không	3	1	xe ô tô	16h	6	xe ô tô	không
24	106,697778	10,814531	Không	Không	không	Bình Thạnh	1	xe ô tô	21h30	4	xe ô tô	không
25	106,662388	10,800557	Không	Có	bị thương	Phú Nhuận	1	xe máy	21h30	5	xe máy	bị thương
26	106,662388	10,800557	Không	Có	bị thương	Phú Nhuận	1	xe máy	21h30	5	xe máy	bị thương
27	106,687969	10,728023	Có	Có	không	Bình Chánh	1	xe tải	6h	5	xe tải	bị thương
28	106,794534	10,862418	Có	Không	không	9	0	xe ô tô	22h15	4	xe tải	bị thương
29	106,758173	10,783226	Có	Có	bị thương	2	1	xe máy	20h	3	xe máy	nhẹ
30	106,758173	10,783226	Có	Có	chết	2	1	xe máy	20h	3	xe máy	nặng
31	106,636193	10,802755	Có	Không	không	Tân Phú	1	xe tải	9h40	3	xe máy	chết
32	106,772941	10,870719	Có	Có	không	Thủ Đức	1	xe tải	0h30	3	xe máy	chết
33	106,745446	10,794172	Không	Không	bị thương	2	1	xe máy	15h15	1	xe tải	bị thương
34	106,593149	10,778708	Không	Có	chết	Bình Tân	1	xe máy	19h30	7	xe tải	bị thương
35	106,593149	10,778708	Không	Có	bị thương	Bình Tân	1	xe máy	19h30	7	xe tải	bị thương
36	106,619934	10,726584	Có	Có	bị thương	Bình Tân	2	xe máy	16h	7	xe tải	bị thương
37	106,806748	10,826599	Không	Có	bị thương	9	2	xe máy	16h	6	xe ô tô	bị thương
38	106,806748	10,826599	Không	Có	không	9	1	xe máy	16h	6	xe ô tô	bị thương
39	106,664716	10,810065	Không	Không	bị thương	Tân Bình	1	xe máy	10h15	6	xe ô tô	bị thương
40	106,593084	10,778345	Không	Có	bị thương	Bình Tân	1	xe máy	8h	6	xe khách	bị thương
41	106,593084	10,778345	Không	Có	bị thương	Bình Tân	1	xe máy	8h	6	xe khách	bị thương

- **Phụ lục 2:**

  Dữ liệu mẫu chuyển về dạng nhị phân

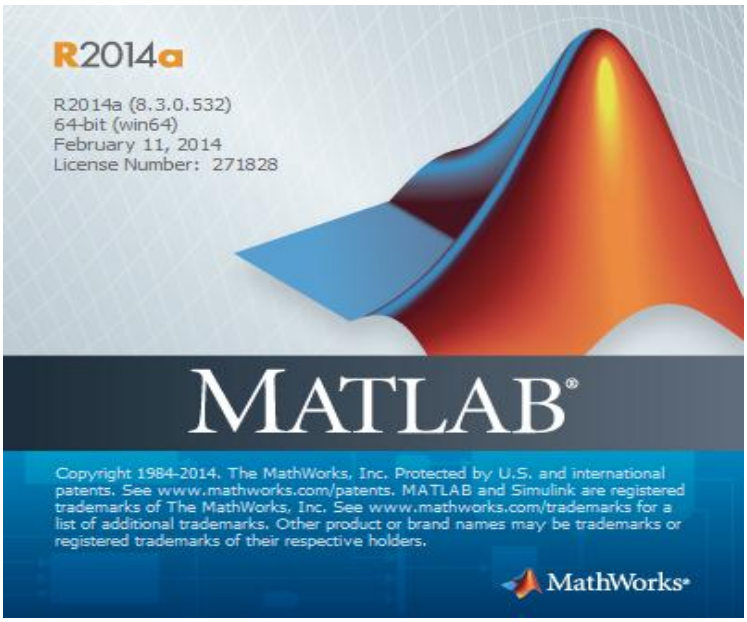
Số lượng người bị tai nạn	Giao_cat	Phương tiện
1	0	1
1	1	0
0	1	0
1	1	0
0	0	0
1	0	0
1	0	0
0	0	1
1	0	1
1	0	1
1	0	0
1	1	0
0	1	0
1	1	0
1	1	0
1	1	1
1	0	0
0	0	0
0	0	0
1	1	0
1	0	1
1	0	1
1	0	1
1	0	1
1	0	1
1	0	1
1	1	1

1	1	0
1	1	1
1	1	1
1	1	0
1	1	0
1	0	0
1	0	0
1	0	0
0	1	0
0	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	1
1	1	1
1	0	1
1	0	1
0	0	0
1	0	0
1	0	1
1	0	0
1	1	0
1	1	0
1	0	0
1	1	0
1	0	0
0	0	0
1	1	0
1	1	0

- **Phụ lục 3**

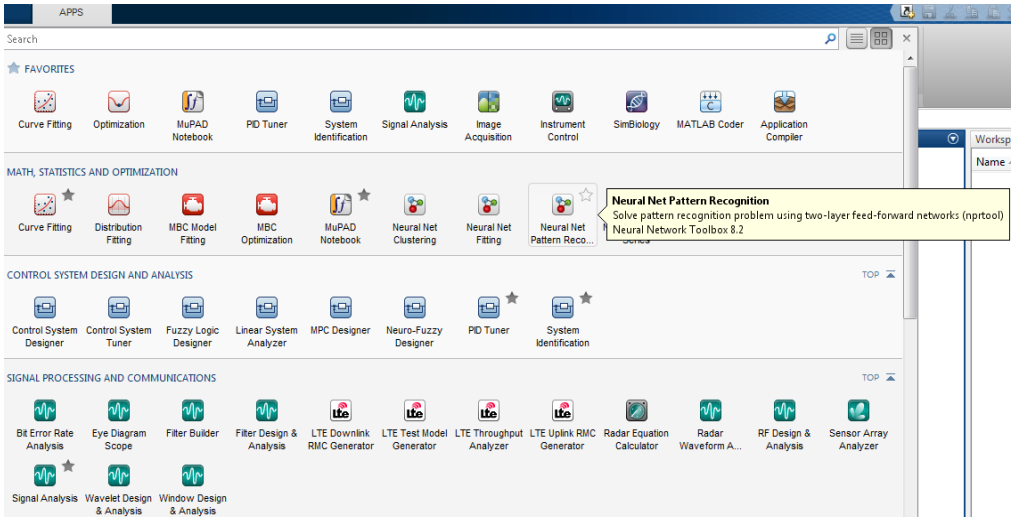
Quá trình thực hiện phân tích dữ liệu bằng công cụ neuron trên Matlab

Mở phần mềm Matlab. Ở đây đề tài sử dụng Matlab R2014a phiên bản dành cho win 64 bit.



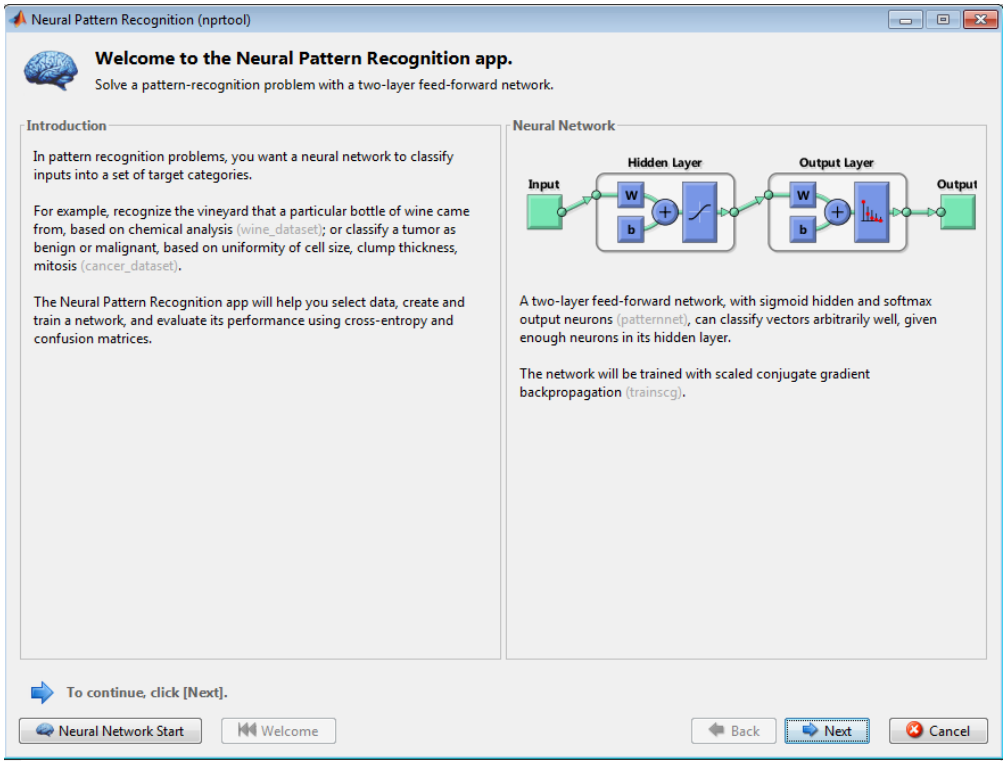
Hình 1: Phiên bản phần mềm Matlab

Sau đó khởi động chương trình NeuronNet Pattern Recognition, đây là công cụ phân tích mạng neural phù hợp với đề tài



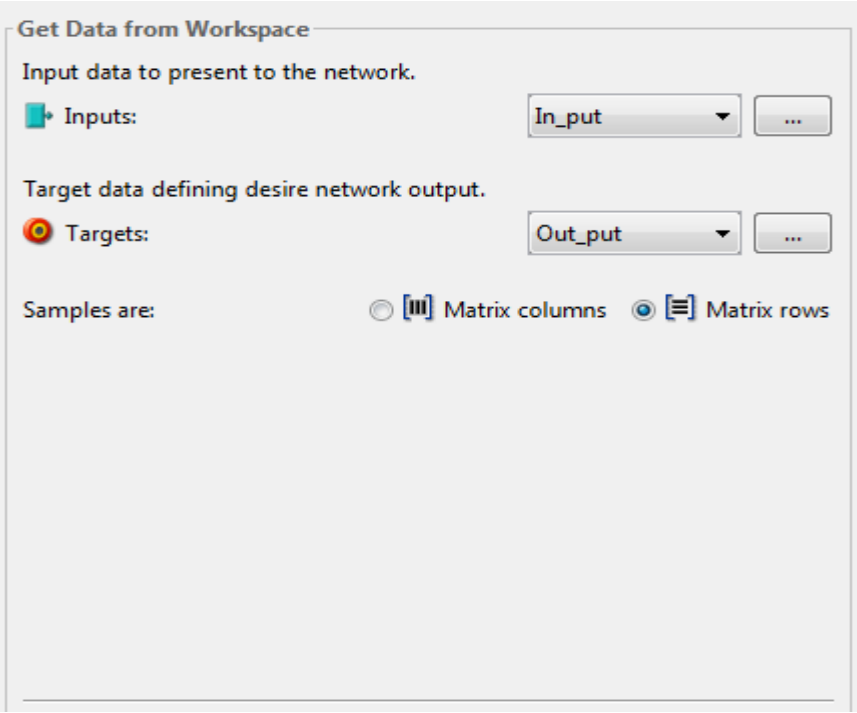
Hình 2: Công cụ phân tích mạng Neuron

Khởi động vào chương trình với giao diện như sau:



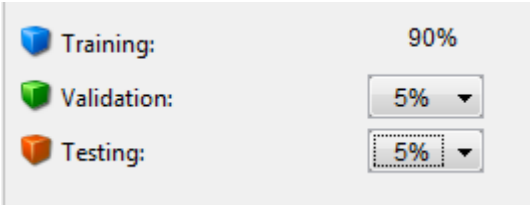
Hình 3: Giao diện chương trình NeuronNet Pattern Recognition.

Để bắt đầu quá trình huấn luyện cần import dữ liệu đầu vào và dữ liệu đầu ra vào chương trình. Ở đây dữ liệu đầu vào và đầu ra nằm ở dạng nhị phân như đã nêu trong chương 4 và được lưu trữ ở file text với tên: In.txt, OutT1.txt và OutT2.txt. Tuy nhiên do dữ liệu được sắp xếp theo dạng dòng nên trước khi chạy chương trình cần phải tích vào đúng định dạng của dữ liệu dể chương trình có thể đọc là: Matrix rows.



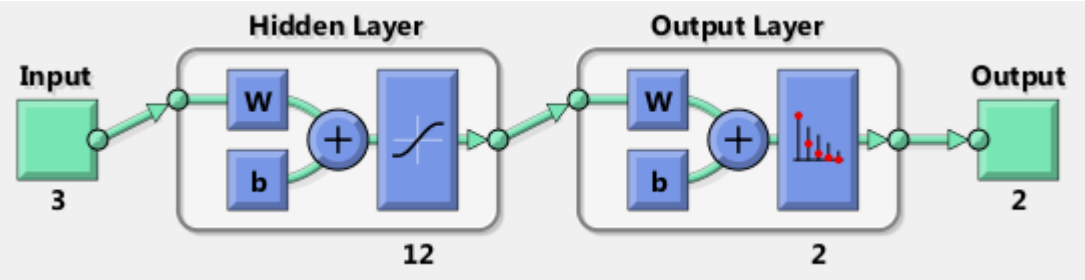
Hình 4: Quá trình thêm dữ liệu vào chương trình

Tiếp theo cần chọn ra số lượng dữ liệu để nhận dạng và số lượng dữ liệu để kiểm tra sau khi huấn luyện. Ở đây đề tài chọn lấy ra 5% dữ liệu để nhận dạng, 5% dữ liệu để kiểm tra và còn lại 90% dùng huấn luyện để nhận dạng đặc điểm TNGT.



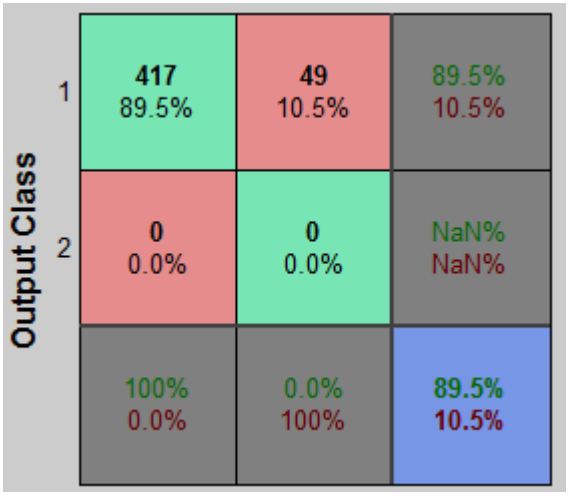
Hình 5: Phân chia dữ liệu dùng để huấn luyện

Song song đó một bước quan trọng không thể thiếu là lựa chọn số lớp ẩn (Hidden) để tiến hành phân tích mạng neuron. Ở đây đề tài sẽ lấy 15 lớp ẩn phù hợp với kết luận trên.



Hình 6: Số lượng lớp ẩn

Bắt đầu quá trình huấn luyện. Sau khi huấn luyện thành công, công cụ sẽ cho ra bảng tính toán sai số tổng.



Hình 7: Sai số huấn luyện

Như vậy là quá trình huấn luyện đã kết thúc. Tuy nhiên để chính xác hơn thì cần phải huyến luyện lại (retrain) nhiều lần để có kết quả sai số tổng quát hơn. Vấn đề này đã được đề cập ở chương 5.

- **Phụ lục 4: Bảng tính sai số lớp ẩn**

Tổ hợp 1

Số lớp	1	2	3	4	5	6	7	8	9	10
2	89,5	89,5	80,0	74,9	89,5	89,5	89,5	73,5	88,5	89,5
3	80,3	89,5	84,5	85,4	89,5	89,5	89,5	75,3	84,5	89,5
4	84,3	89,5	84,5	89,5	89,5	84,3	75,3	89,5	84,5	89,5
5	84,3	89,5	89,5	80,9	89,5	84,5	89,5	74,9	89,5	75,3
6	89,5	85,0	85,0	53,9	75,8	79,8	89,5	85,0	89,5	84,5
7	89,5	89,5	89,5	89,5	84,5	89,5	85,0	89,5	89,5	88,5
8	80,9	80,5	84,5	85,4	85,4	89,5	89,5	84,5	85,4	89,5
9	89,5	85,4	84,5	85,4	89,5	85,0	89,5	75,3	84,5	80,5
10	84,5	80,9	80,3	89,5	80,5	80,0	85,4	84,5	84,5	89,5
11	89,5	85,0	89,5	84,3	84,5	80,9	89,5	89,5	84,5	89,5
12	80,0	84,5	80,5	85,4	89,5	61,4	85,4	89,5	85,4	75,3
13	89,5	89,5	84,5	84,5	85,0	60,7	85,4	41,2	80,0	84,3
14	80,9	85,4	80,5	84,5	80,3	66,5	89,5	89,5	80,5	85,4
15	89,5	85,4	89,5	84,3	80,5	80,5	70,2	85,4	84,5	89,5

Tổ hợp 2

Số lớp	1	2	3	4	5	6	7	8	9	10
2	89,5	86,3	89,5	89,5	86,3	88,5	73,8	88,4	73,8	89,5
3	89,5	88,4	89,5	89,5	88,6	80,5	88,4	80,1	86,5	85,4
4	51,7	65,9	89,5	80,8	87,6	80,5	84,5	89,5	89,5	88,5
5	69,5	82,4	88,4	89,5	73,8	89,5	89,5	88,6	80,7	89,5
6	89,5	80,1	88,2	84,8	53,4	73,8	85,4	89,5	62,7	89,5
7	88,4	88,6	89,5	80,1	79,8	77,9	89,5	80,2	88,4	89,5
8	89,5	81,7	88,7	79,6	88,4	88,6	89,5	88,6	88,4	89,5
9	89,5	67,4	89,5	89,5	87,6	89,5	88,6	86,3	80,1	82,2

10	80,1	80,5	79,3	60,1	82,2	89,5	89,5	88,3	80,1	89,5
11	89,5	86,3	88,6	89,5	80,1	82,3	79,5	80,1	89,5	89,5
12	87,9	89,5	88,4	87,9	88,5	89,5	86,5	89,5	89,5	89,4
13	84,3	88,4	86,3	89,5	85,4	89,5	64,8	80,8	89,5	89,5
14	66,5	84,8	87,6	69,5	89,5	89,5	88,6	89,5	89,5	60,7
15	89,5	88,4	89,5	89,5	80,5	87,6	86,3	88,6	87,6	88,4

- **Phụ lục 5: Code chuyển dữ liệu sang dạng ma trận**

```
mang = []
        dong = 0
        f = open("Đường dẫn đến tập tin (*.txt)")
        dong = int(f.readline())
        matran = f.readlines()
        dong1 = matran[0].split("    ")
        sophantu_trong_mot_dong = len(dong1)
        mang = [[0 for i in range(sophantu_trong_mot_dong)] for j in range(dong)]
        for p in range(dong-1):
                array_tung_dong = matran[p].split("\t")
                for q in range(sophantu_trong_mot_dong):
                        mang[p][q] = int( array_tung_dong[q])
print mang
```

- **Phụ lục 6: Code Python demo sử dụng trong ANN và BP**

```
import math
import random
import sys

INPUT_NEURONS = 4
```

```
HIDDEN_NEURONS = 6
OUTPUT_NEURONS = 14

LEARN_RATE = 0.2 # Rho.
NOISE_FACTOR = 0.58
TRAINING_REPS = 10000
MAX_SAMPLES = 14

TRAINING_INPUTS = [[1, 1, 1, 0],
                   [1, 1, 0, 0],
                   [0, 1, 1, 0],
                   [1, 0, 1, 0],
                   [1, 0, 0, 0],
                   [0, 1, 0, 0],
                   [0, 0, 1, 0],
                   [1, 1, 1, 1],
                   [1, 1, 0, 1],
                   [0, 1, 1, 1],
                   [1, 0, 1, 1],
                   [1, 0, 0, 1],
                   [0, 1, 0, 1],
                   [0, 0, 1, 1]]

TRAINING_OUTPUTS = [[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
```

```
            [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
            [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
            [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
            [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
            [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
            [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
            [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
            [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
            [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]

class Example_4x6x16:
    def __init__(self, numInputs, numHidden, numOutput, learningRate, noise, epochs,
numSamples, inputArray, outputArray):
        self.mInputs = numInputs
        self.mHiddens = numHidden
        self.mOutputs = numOutput
        self.mLearningRate = learningRate
        self.mNoiseFactor = noise
        self.mEpochs = epochs
        self.mSamples = numSamples
        self.mInputArray = inputArray
        self.mOutputArray = outputArray

        self.wih = [] # Input to Hidden Weights
        self.who = [] # Hidden to Output Weights
        inputs = []
        hidden = []
        target = []
```

```
    actual = []
    erro = []
    errh = []
    return

def initialize_arrays(self):
    for i in range(self.mInputs + 1): # The extra element represents bias node.
        self.wih.append([0.0] * self.mHiddens)
        for j in range(self.mHiddens):
            # Assign a random weight value between -0.5 and 0.5
            self.wih[i][j] = random.random() - 0.5

    for i in range(self.mHiddens + 1): # The extra element represents bias node.
        self.who.append([0.0] * self.mOutputs)
        for j in range(self.mOutputs):
            self.who[i][j] = random.random() - 0.5

    self.inputs = [0.0] * self.mInputs
    self.hidden = [0.0] * self.mHiddens
    self.target = [0.0] * self.mOutputs
    self.actual = [0.0] * self.mOutputs
    self.erro = [0.0] * self.mOutputs
    self.errh = [0.0] * self.mHiddens

    return

def get_maximum(self, vector):
    # This function returns an array index of the maximum.
```

```
    index = 0
    maximum = vector[0]
    length = len(vector)

    for i in range(length):
        if vector[i] > maximum:
            maximum = vector[i]
            index = i

    return index

def sigmoid(self, value):
    return 1.0 / (1.0 + math.exp(-value))

def sigmoid_derivative(self, value):
    return value * (1.0 - value)

def feed_forward(self):
    total = 0.0

    # Calculate input to hidden layer.
    for j in range(self.mHiddens):
        total = 0.0
        for i in range(self.mInputs):
            total += self.inputs[i] * self.wih[i][j]

        # Add in bias.
        total += self.wih[self.mInputs][j]
```

```
            self.hidden[j] = self.sigmoid(total)

        # Calculate the hidden to output layer.
        for j in range(self.mOutputs):
            total = 0.0
            for i in range(self.mHiddens):
                total += self.hidden[i] * self.who[i][j]

            # Add in bias.
            total += self.who[self.mHiddens][j]
            self.actual[j] = self.sigmoid(total)

        return

    def back_propagate(self):
        # Calculate the output layer error (step 3 for output cell).
        for j in range(self.mOutputs):
            self.erro[j] = (self.target[j] - self.actual[j]) * self.sigmoid_derivative(self.actual[j])

        # Calculate the hidden layer error (step 3 for hidden cell).
        for i in range(self.mHiddens):
            self.errh[i] = 0.0
            for j in range(self.mOutputs):
                self.errh[i] += self.erro[j] * self.who[i][j]

            self.errh[i] *= self.sigmoid_derivative(self.hidden[i])

        # Update the weights for the output layer (step 4).
```

```
    for j in range(self.mOutputs):
        for i in range(self.mHiddens):
            self.who[i][j] += (self.mLearningRate * self.erro[j] * self.hidden[i])

        # Update the bias.
        self.who[self.mHiddens][j] += (self.mLearningRate * self.erro[j])

    # Update the weights for the hidden layer (step 4).
    for j in range(self.mHiddens):
        for i in range(self.mInputs):
            self.wih[i][j] += (self.mLearningRate * self.errh[j] * self.inputs[i])

        # Update the bias.
        self.wih[self.mInputs][j] += (self.mLearningRate * self.errh[j])

    return

def print_training_stats(self):
    sum = 0.0

    for i in range(self.mSamples):
        for j in range(self.mInputs):
            self.inputs[j] = self.mInputArray[i][j]

        for j in range(self.mOutputs):
            self.target[j] = self.mOutputArray[i][j]

        self.feed_forward()
```

```
            if self.get_maximum(self.actual) == self.get_maximum(self.target):
                sum += 1
            else:
                sys.stdout.write(str(self.inputs[0]) + "\t" + str(self.inputs[1]) + "\t" +
str(self.inputs[2]) + "\t" + str(self.inputs[3]) + "\n")
                sys.stdout.write(str(self.get_maximum(self.actual)) + "\t" +
str(self.get_maximum(self.target)) + "\n")

        sys.stdout.write("Network is " + str((float(sum) / float(MAX_SAMPLES)) * 100.0)
+ "% correct.\n")

        return

    def train_network(self):
        sample = 0

        for i in range(self.mEpochs):
            sample += 1
            if sample == self.mSamples:
                sample = 0

            for j in range(self.mInputs):
                self.inputs[j] = self.mInputArray[sample][j]

            for j in range(self.mOutputs):
                self.target[j] = self.mOutputArray[sample][j]
```

```
            self.feed_forward()

            self.back_propagate()

        return

    def test_network(self):
        for i in range(self.mSamples):
            for j in range(self.mInputs):
                self.inputs[j] = self.mInputArray[i][j]

            self.feed_forward()

            for j in range(self.mInputs):
                sys.stdout.write(str(self.inputs[j]) + "\t")

            sys.stdout.write("Output: " + str(self.get_maximum(self.actual)) + "\n")

        return

    def test_network_with_noise(self):
        # This function adds a random fractional value to all the training inputs greater than
zero.
        for i in range(self.mSamples):
            for j in range(self.mInputs):
                self.inputs[j] = self.mInputArray[i][j] + (random.random() * NOISE_FACTOR)

            self.feed_forward()
```

```
            for j in range(self.mInputs):
                sys.stdout.write("{:03.3f}".format(((self.inputs[j] * 1000.0) / 1000.0)) + "\t")
            sys.stdout.write("Output: " + str(self.get_maximum(self.actual)) + "\n")

        return

if __name__ == '__main__':
    ex = Example_4x6x16(INPUT_NEURONS, HIDDEN_NEURONS,
OUTPUT_NEURONS, LEARN_RATE, NOISE_FACTOR, TRAINING_REPS,
MAX_SAMPLES, TRAINING_INPUTS, TRAINING_OUTPUTS)
    ex.initialize_arrays()
    ex.train_network()
    ex.print_training_stats()
    sys.stdout.write("\nTest network against original input:\n")
    ex.test_network()
    sys.stdout.write("\nTest network against noisy input:\n")
    ex.test_network_with_noise()
```