ĐẠI HỌC QUỐC GIA TP HCM

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

LÊ HỮU VINH

ÁP DỤNG DEEP LEARNING CHO BÀI TOÁN DỰ BÁO CHUỖI THỜI GIAN

LUẬN VĂN THẠC SĨ

NGÀNH KHOA HỌC MÁY TÍNH

Mã số: 8480101

TP HỒ CHÍ MINH – NĂM 2019

ĐẠI HỌC QUỐC GIA TP HCM

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

LÊ HỮU VINH

ÁP DỤNG DEEP LEARNING CHO BÀI TOÁN DỰ BÁO CHUỖI THỜI GIAN

LUẬN VĂN THẠC SĨ

NGÀNH KHOA HỌC MÁY TÍNH

Mã số: 8480101

NGƯỜI HƯỚNG DẪN KHOA HỌC

PGS.TS NGUYỄN ĐÌNH THUÂN

TP HỒ CHÍ MINH – NĂM 2019

LỜI CẢM ƠN

Lời đầu tiên, em xin gởi lời cảm ơn chân thành đến quý thầy cô trường Đại học Công Nghệ Thông Tin đã tận tình dạy bảo cho em nhiều kiến thức bổ ích trong suốt thời gian học tập tại trường, cũng như tạo điều kiện cho em thực hiện đề tài này. Kính chúc quý thầy cô luôn dồi dào sức khoẻ và thành công trong cuộc sống.

Đặc biệt, em xin bày tỏ lòng biết ơn chân thành, sâu sắc đến thầy Nguyễn Đình Thuân. Thầy đã tận tâm, nhiệt tình hướng dẫn và chỉ bảo cho em trong suốt quá trình thực hiện đề tài. Luận văn này sẽ rất khó hoàn thành nếu không có sự truyền đạt kiến thức quý báu và sự hướng dẫn nhiệt tình của thầy.

Xin cảm ơn tất cả bạn bè đã động viên, giúp đỡ và đóng góp nhiều ý kiến quý báu, qua đó, giúp em hoàn thiện hơn đề tài này.

Em xin gửi lời cám ơn đến gia đình đã tạo mọi điều kiện thuận lợi về vật chất và tinh thần, giúp em hoàn thành luận văn một cách tốt nhất.

Và cuối cùng, em cũng không quên gửi lời cảm ơn đến tác giả của các báo cáo nghiên cứu khoa học mà em đã tham khảo và tìm hiểu cho đề tài.

Luận văn đã hoàn thành với một số kết quả nhất định, tuy nhiên vẫn không tránh khỏi thiếu sót. Kính mong sự đóng góp ý kiến từ quý thầy cô và các bạn.

Một lần nữa, em xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày 05 tháng 07 năm 2019

Học viên

Lê Hữu Vinh

LỜI CAM ĐOAN

Tôi xin cam đoan:

1. Những nội dung trong luận văn này là do tôi thực hiện dưới sự hướng dẫn của thầy PGS.TS Nguyễn Đình Thuân.
2. Mọi tham khảo trong luận văn đều được trích dẫn rõ ràng tên công trình, tên tác giả, thời gian công bố.

Mọi sao chép không hợp lệ, vi phạm quy chế đào tạo, tôi xin chịu hoàn toàn trách nhiệm.

TP. Hồ Chí Minh, ngày 05 tháng 07 năm 2019

Học viên

Lê Hữu Vinh

MỤC LỤC

DANH MỤC CÁC KÝ HIỆU, THUẬT NGỮ VÀ CHỮ VIẾT TẮT

ACF	: Auto Correlation Function
AIC	: Akaike Information Criterion
ANN	: Artificial Neural Network
AR	: Autoregressive
ARIMA	: Autoregressive Intergrated Moving Average
ARMA	: Autoregressive Move Average
BIC	: Bayesian Information Criterion
BJ	: Phương pháp Box – Jenkins
CNN	: Convolutional Neural Network
FFNN	: Feedforward Neural Network
Linear Regression	: Hồi quy tuyến tính
LSTM	: Long Short-Term Memory
MA	: Moving Average
MAPE	: Mean Absolute Percentage Error
PACF	: Partial Autocorrelation Function
Polynomial Regression	: Hồi quy đa thức
RMSE	: Root Mean Square Error
SVM	: Support Vector Machine
SVR	: Support Vector Regression
Time Series	: Chuỗi thời gian

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

MỞ ĐẦU

Ngày nay, dự báo là một cơ sở đáng tin cậy để con người ra quyết định, lựa chọn những giải pháp phù hợp trong quản lý, trong kinh tế, xã hội,… Và dự báo đang trở thành một trong những vấn đề được ưu tiên hàng đầu ở hầu hết các lĩnh vực. Trong các bài toán dự báo, người ta ngày càng quan tâm đến các dự báo liên quan đến chuỗi thời gian như dự báo giá vàng, chứng khoán, tiền điện tử, nhu cầu năng lượng,... vì những lợi ích lớn về kinh tế mà chúng mang lại.

Bài toán dự báo chuỗi thời gian (time-series forecasting) ra đời để giải quyết vấn đề trên. Dự báo chuỗi thời gian là việc sử dụng mô hình để dự đoán sự kiện theo thời gian dựa vào các sự kiện đã biết trong quá khứ, để từ đó, dự đoán các sự kiện trước khi nó xảy ra trong tương lai. Có nghĩa là dựa vào các thông tin đã biết, các nhà nghiên cứu sẽ xây dựng một số mô hình có khả năng dự báo trước các thông tin cần biết ở một thời điểm xác định t+1, t+2,…

Với tầm quan trọng của dự báo chuỗi thời gian, các nhà nghiên cứu đã đưa ra nhiều phương pháp để giải quyết bài toán này. Từ các mô hình dự báo thông kê như ARIMA đến các mô hình máy học như hồi quy, SVM, mạng nơron. Và trong những năm gần đây, các mô hình dự báo chuỗi thời gian sử dụng Deep Learning là những mô hình mang đến nhiều hứa hẹn nhất. Deep Learning có khả năng tự động học mối liên hệ giữa các giá trị trong chuỗi thời gian, tự động phát hiện yếu tố xu hướng và mùa vụ trong chuỗi dữ liệu, từ đó có thể nâng cao độ chính xác cho dự báo.

Việc áp dụng Deep Learning cho bài toán dự báo chuỗi thời gian được kỳ vọng giải quyết một số thách thức của bài toán này, đặc biệt là nâng cao độ chính xác dự báo. Luận văn sẽ trình bày các mô hình Deep Learning cũng như một số mô hình dự báo chuỗi thời gian. Sau đó, các phương pháp kết hợp Deep Learning và một số mô hình khác sẽ được giới thiệu với kỳ vọng nâng cao độ chính xác dự báo chuỗi thời gian. Các mô hình dự báo sẽ được thử nghiệm để dự báo giá đóng cửa của Bitcoin trong ngày kế tiếp để đánh giá độ chính xác dự báo của các mô hình.

CHƯƠNG 1 TỔNG QUAN

1.1 Đặt vấn đề

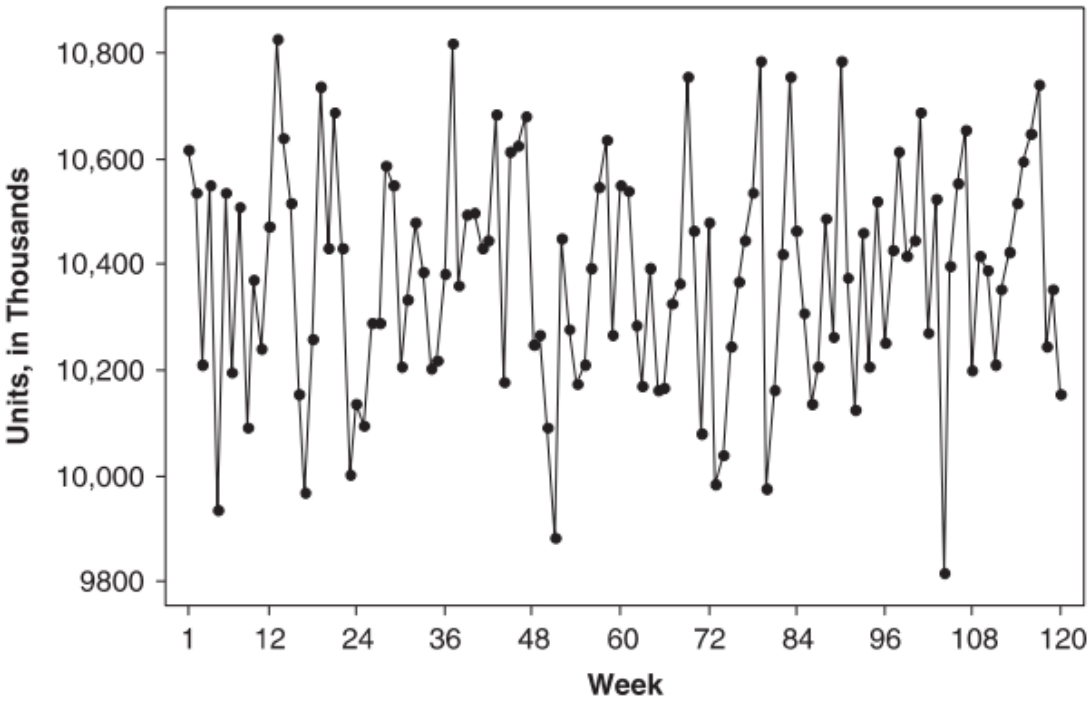
Dự báo là một vấn đề quan trọng trong hầu hết các lĩnh vực bao gồm kinh doanh, công nghiệp, kinh tế, khoa học môi trường, y học, khoa học xã hội, chính trị và tài chính,…Những nghiên cứu ứng dụng dự báo trong khắp các lĩnh vực được thực hiện ngày càng nhiều. Đầu tiên phải kể đến các dự báo trong kinh tế và tài chính như dự báo chứng khoán [7, 9, 10, 13, 16], dự báo tỷ giá hối đoái [5]. Một lĩnh vực dự báo khác rất quan tâm hiện nay là dự báo tiền điện tử (cryptocurrency) [3, 4, 6, 25, 28]. Ngoài ra, dự báo cũng được áp dụng trong rất nhiều lĩnh vực khác. Từ dự báo lưu lượng giao thông [8], dự báo tiêu thụ năng lượng [17, 20] đến dự báo chất lượng nước [14]. Những dự báo trong các lĩnh vực như môi trường, thể thao, giáo dục [1, 2] đều đã được thực hiện.

Các vấn đề dự báo thường được phân thành dự báo ngắn hạn (short term), trung hạn (medium term) và dài hạn (long term). Dự báo ngắn hạn là các dự báo trong tương lai gần như một ngày, một tuần hoặc một tháng. Dự báo trung hạn thì thời gian có thể kéo dài hơn từ 1 đến 2 năm. Các vấn đề dự báo dài hạn thì thời gian dự báo có thể kéo dài trong nhiều năm.

Có ba phương pháp dự báo chính là phương pháp phán đoán (judgmental method), phương pháp nhân quả (causal method) và phương pháp chuỗi thời gian (time series method). Phương pháp phán đoán dựa vào các ý kiến, kinh nghiệm và ước tính xác suất chủ quan của chuyên gia. Phương pháp này được sử dụng trong trường hợp thiếu dữ liệu lịch sử, không có cơ sở nào để dự báo [37]. Phương pháp nhân quả xác định mối liên hệ giữa biến giá trị cần dự báo (dependent variable) với các biến giá trị khác (independent variables). Dựa trên mối liên hệ đó, các dự báo được thực hiện. Ví dụ, khi dự báo tỷ giá ngoại tệ USD/VND, rất có thể tỷ giá này sẽ có liên quan đến sự tăng giảm của giá dầu mỏ, giá vàng. Một ví dụ khác, cổ phiếu của Facebook với sự phát hành của đồng Libra có thể có ảnh hưởng đến giá của các đồng tiền điện tử như Bitcoin, Ethereum,… Thách thức lớn nhất của phương pháp nhân quả là thu thập dữ liệu của các biến giá trị liên quan và cần có những công cụ

thật sự hiệu quả để xác định mối liên hệ giữa các biến giá trị trong dự báo. Trong khi đó, dự báo chuỗi thời gian là phương pháp dự báo được sử dụng rộng rãi trong các lĩnh vực và nhận được sự quan tâm của các nhà nghiên cứu [35]. Phương pháp này thực hiện dự báo dựa vào giá trị chuỗi thời gian được thu thập trong những khoảng thời gian với tần suất thống nhất. Sau đó, các mô hình dự báo sẽ được xây dựng nhằm tìm kiếm mối liên hệ giữa các giá trị trong chuỗi thời gian để thực hiện dự báo.

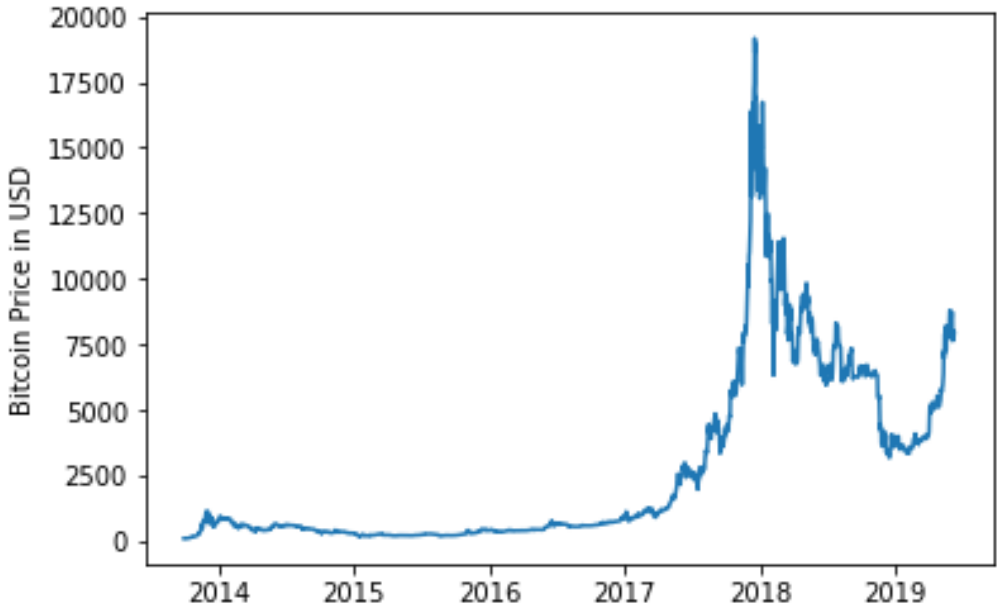
Chuỗi thời gian (time series) là một tập hợp các điểm dữ liệu (data points) liên tiếp được thu thập và sắp xếp theo thứ tự thời gian. Trong toán học, chuỗi thời gian được định nghĩa là một vector $x(t)$, t = 0, 1, 2,…với t đại diện cho các điểm thời gian. Biến $x(t)$ được xem như một biến ngẫu nhiên [36].



Hình 1.1. Chuỗi thời gian doanh số dược phẩm bán được hàng tháng

Nguồn: *Một số ví dụ về chuỗi thời gian* [35]

Một chuỗi thời gian được gọi là đơn biến nếu tại mỗi điểm dữ liệu chỉ bao gồm một biến duy nhất. Ngược lại, nếu tại mỗi điểm dữ liệu gồm nhiều hơn một biến thì chuỗi thời gian đó được gọi là đa biến. Ví dụ, chuỗi thời gian là giá đóng cửa của Bitcoin theo từng ngày là chuỗi thời gian đơn biến, còn chuỗi thời gian là giá mở cửa, giá đóng cửa, giá cao nhất, giá thấp nhất theo từng ngày là chuỗi thời gian đa biến.



Hình 1.2. Chuỗi thời gian giá đóng cửa của Bitcoin

Nguồn: *Dữ liệu giá Bitcoin thu thập trên CoinDesk*

Một chuỗi thời gian thường chịu ảnh hưởng bởi 4 thành phần là xu hướng (trend), chu kỳ (cyclical), mùa (seasonal), biến đổi ngẫu nhiên (irregular) [36]:

- Sự tăng, giảm hoặc không thay đổi của một chuỗi thời gian trong một thời gian dài được gọi là xu hướng. Tính xu hướng là chỉ ra sự vận động lâu dài của một chuỗi thời gian.
- Chu kỳ là sự thay đổi của chuỗi thời gian trong một khoảng thời gian dài trung bình và sự thay đổi này được lặp lại sau khoảng thời gian đó. Thông thường, khoảng thời gian của một chu kỳ lớn hơn 2 năm.
- Mùa là sự thay đổi của chuỗi thời gian theo một khoảng thời gian ngắn, thường là trong một năm.
- Biến đổi ngẫu nhiên được gây ra bởi những tác động không thể đoán trước, không thường xuyên và cũng không theo một quy luật nào.

Dự báo chuỗi thời gian là việc mô hình hóa dữ liệu, ước lượng các tham số của mô hình dựa trên những dữ liệu chuỗi thời gian trong quá khứ, từ đó đưa ra dự báo về các giá trị của chuỗi thời gian trong tương lai. Do đó, dữ liệu thời gian trong quá khứ ảnh hưởng rất lớn đến quá trình xây dựng mô hình và cải thiện kết quả dự báo

của mô hình. Trong dự báo chuỗi thời gian, nhiều thách thức được đặt ra và vẫn chờ các nhà khoa học tiếp tục nghiên cứu, giải quyết:

- Đầu tiên phải kể đến tính chất của dữ liệu chuỗi thời gian làm cho việc xử lý, dự báo gặp nhiều khó khăn. Dữ liệu chuỗi thời gian có 4 tính chất tiêu biểu là xu thế, chu kì, theo mùa, biến đổi ngẫu nhiên. Để dự báo trên dữ liệu chuỗi thời gian, các phương pháp dự báo phải phân tích tốt các tính chất này. Tuy nhiên, hiện nay chưa có phương pháp nào thực sự tốt cho việc phân tích này.
- Một trong những tính chất khó giải quyết nhất của dữ liệu chuỗi thời gian là tính chất biến đổi ngẫu nhiên. Có nghĩa là các dữ liệu mang tính thời gian dễ chịu tác động bởi các yếu tố, biến đổi một cách ngẫu nhiên, không quy luật. Và làm sao để xác định dữ liệu chuỗi thời gian có bị biến đổi ngẫu nhiên hay không là một thách thức lớn cho các nhà nghiên cứu.
- Các phương pháp, kỹ thuật dự báo chuỗi thời gian được đề xuất thường chỉ phù hợp một số dạng dữ liệu như tuyến tính hoặc phi tuyến [12, 16]. Những phương pháp kết hợp các mô hình riêng lẻ để nâng cao độ chính xác dự báo là hết sức cần thiết. Tuy nhiên, kết hợp như thế nào và khi nào là vấn đề cần được xem xét kỹ để đem đến kết quả dự báo tốt nhất.

Các mô hình dự báo chuỗi thời gian như ARIMA, hồi quy, SVM, mạng nơron đều đã đáp ứng được phần nào yêu cầu dự báo và hỗ trợ ra quyết định trong các lĩnh vực. Tuy nhiên, độ chính xác dự báo có thể được nâng cao hơn nữa. Và trong những năm gần đây, các mô hình dự báo chuỗi thời gian sử dụng Deep Learning là những mô hình mang đến nhiều hứa hẹn nhất. Deep Learning có khả năng tự động học mối liên hệ giữa các giá trị trong chuỗi thời gian, tự động phát hiện yếu tố xu hướng và mùa vụ trong chuỗi dữ liệu, từ đó có thể nâng cao độ chính xác cho dự báo. Deep Learning là một lĩnh vực nghiên cứu của Machine Learning. Deep Learning là một phương pháp học với nhiều cấp độ nhằm biểu diễn dữ liệu và rút trích thông tin để giúp nhận biết được dữ liệu như hình ảnh, âm thanh và văn bản. Deep Learning bản chất là một mạng nơron "rộng" và "sâu". Với "rộng" là số input đầu vào lớn, thường là một ma trận với số chiều lớn, chủ yếu là số mẫu lớn, mỗi mẫu là một hàng của ma

trận. Với "sâu" là có nhiều lớp neural ẩn (hơn hoặc bằng 2). Hay nói cách khác, Deep Learning là một neural network (1) bao gồm nhiều lớp hoặc giai đoạn của xử lý thông tin (2) phương pháp học có giám sát hoặc học không giám sát với số đặc trưng lớn.

Việc áp dụng Deep Learning cho bài toán dự báo chuỗi thời gian được kỳ vọng giải quyết một số thách thức của bài toán này, đặc biệt là nâng cao độ chính xác dự báo. Luận văn sẽ nghiên cứu các mô hình Deep Learning cũng như một số mô hình dự báo chuỗi thời gian. Sau đó, các phương pháp kết hợp Deep Learning và một số mô hình khác sẽ được giới thiệu với kỳ vọng nâng cao độ chính xác dự báo. Các mô hình dự báo sẽ được thử nghiệm để dự báo giá đóng cửa của Bitcoin trong ngày kế tiếp để đánh giá độ chính xác dự báo của các mô hình.

1.2 Mục tiêu và phạm vi của luận văn

Mục tiêu của luận văn là nghiên cứu, áp dụng Deep Learning cho bài toán dự báo chuỗi thời gian với kỳ vọng nâng cao độ chính xác dự báo. Luận văn đề ra những mục tiêu, phạm vị cụ thể như sau:

- Tìm hiểu các mô hình dự báo chuỗi thời gian đơn biến với mục tiêu thực hiện các dự báo ngắn hạn.
- Tìm hiểu các mô hình Deep Learning áp dụng cho dự báo chuỗi thời gian đơn biến với mục tiêu thực hiện các dự báo ngắn hạn.
- Nghiên cứu các phương pháp kết hợp Deep Learning và các mô hình khác để nâng cao độ chính xác dự báo.
- Thử nghiệm các mô hình dự báo trên tập dữ liệu chuỗi thời gian đơn biến để đánh giá độ chính xác dự báo của các mô hình.

1.3 Nội dung và phương pháp nghiên cứu

Để đạt được các mục tiêu đặt ra, luận văn tiến hành thực hiện các nội dung sau:

- Khảo sát, tổng quan các phương pháp, kỹ thuật trong bài toán dự báo chuỗi thời gian gồm mô hình ARIMA, mạng nơron, SVM, hồi quy.
- Khảo sát các mô hình Deep Learning đã có để giải quyết bài toán dự báo chuỗi thời gian.

- Nghiên cứu các phương pháp kết hợp Deep Learning và các mô hình khác để nâng cao độ chính xác dự báo như kết hợp ARIMA và Deep Learning, kết hợp dựa trên độ biến động, kết hợp với phương trình hồi quy.
- Thực nghiệm các mô hình dự báo trên tập dữ liệu giá Bitcoin để dự báo giá đóng cửa của Bitcoin trong ngày tiếp theo.

1.4 Bố cục của luận văn

Bố cục của luận văn gồm các chương sau:

- **Chương 1. Tổng quan**: Giới thiệu đề tài, lý do chọn đề tài, mục tiêu và phạm vi cũng như nội dung thực hiện của đề tài.
- **Chương 2. Các mô hình dự báo chuỗi thời gian**: Trình bày cơ sở lý thuyết, cách áp dụng các mô hình trong dự báo chuỗi thời gian.
- **Chương 3. Mô hình Deep Learning cho bài toán dự báo chuỗi thời gian**: Trình bày cơ sở lý thuyết, cách áp dụng các mô hình Deep Learning trong dự báo chuỗi thời gian như RNN, LSTM.
- **Chương 4. Kết hợp Deep Learning với một số mô hình để nâng cao độ chính xác dự báo**: Giới thiệu các phương pháp kết hợp Deep Learning và một số mô hình khác để nâng cao độ chính xác dự báo như phương pháp lai ARIMA và Deep Learning, kết hợp Deep Learning và các mô hình dựa trên độ biến động và kết hợp Deep Learnig và các mô hình bằng phương trình hồi quy.
- **Chương 5. Áp dụng các mô hình dự báo chuỗi thời gian cho dự báo Bitcoin**: Thực nghiệm các mô hình dự báo chuỗi thời gian trên dữ liệu giá Bitcoin để dự báo giá đóng cửa của Bitcoin trong ngày kế tiếp.
- **Chương 6. Kết luận và hướng phát triển**
- **Danh mục công bố khoa học của tác giả**
- **Tài liệu tham khảo**

CHƯƠNG 2 CÁC MÔ HÌNH DỰ BÁO CHUỖI THỜI GIAN

Dự báo chuỗi thời gian được áp dụng rộng rãi trong rất nhiều lĩnh vực. Các mô hình dự báo chuỗi thời gian đang được tập trung nghiên cứu và phát triển để đáp ứng các yêu cầu dự báo. Chương này sẽ trình bày chi tiết một số mô hình dự báo chuỗi thời gian như mô hình thống kê gồm AR, MA, ARIMA hay mô hình máy học gồm mạng nơron, các phương trình hồi quy hay SVM.

2.1 Các mô hình thống kê

2.1.1 Mô hình tự hồi quy

Mô hình tự hồi quy (Autoregressive - AR) là một mô hình được sử dụng phổ biến trong thống kê, kinh tế lượng và xử lý tín hiệu. Mô hình này dự báo giá trị của chuỗi thời gian dựa vào một hoặc nhiều giá trị trong chuỗi thời gian trước đó cộng với giá trị ngẫu nhiên, được gọi là nhiễu trắng (white noise). Mô hình tự hồi quy AR bậc p – AR(p) là một quá trình tuyến tính được xác định bởi phương trình [37]:

$$y_t = c + \sum_{i=1}^{p} \emptyset_i y_{t-i} + \varepsilon_t = c + \emptyset_1 y_{t-1} + \emptyset_2 y_{t-2} + \cdots + \emptyset_p y_{t-p} + \varepsilon_t \qquad (2.1)$$

với y_t là giá trị tại thời điểm t, c là hằng số, $\emptyset_i$ là hệ số tự tương quan tại các thời điểm t-1, t-2, ..., t-p trước đó và ε_t là một số hạng sai số ngẫu nhiên không tương quan, có giá trị trung bình bằng 0 và phương sai không đổi σ_ε^2.

2.1.2 Mô hình trung bình trượt

Mô hình trung bình trượt (Moving Average - MA) là một mô hình biểu diễn sự phụ thuộc của điểm thời gian đang xét vào các lỗi dự báo trước đó (số hạng sai số ngẫu nhiên) và giá trị trung bình của chuỗi thời gian. Một mô hình trung bình trượt MA bậc q – MA(q) được xác định bởi phương trình [37]:

$$y_t = c + \sum_{i=0}^{q} \theta_i \varepsilon_{t-i} = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_{t-q} \varepsilon_{t-q} \qquad (2.2)$$

với c là hằng số và là giá trị trung bình của chuỗi, θ_i là trọng số của các số hạng sai số ngẫu nhiên tại các thời điểm t, t-1, …, t-q, ε_t là một số hạng sai số ngẫu nhiên không tương quan, có giá trị trung bình bằng 0 và phương sai không đổi σ_ε^2.

2.1.3 Mô hình ARMA

Giả sử một chuỗi thời gian tuân theo cả quá trình tự hồi quy và trung bình trượt, chúng ta có thể kết hợp hai mô hình lại với nhau, gọi là mô hình ARMA (Autoregressive Moving Average) để biểu diễn chuỗi thời gian như sau [36]:

$$y_t = c + \sum_{i=1}^{p} \emptyset_i y_{t-i} + \varepsilon_t + \sum_{i=0}^{q} \theta_i \varepsilon_{t-i} \quad (2.3)$$

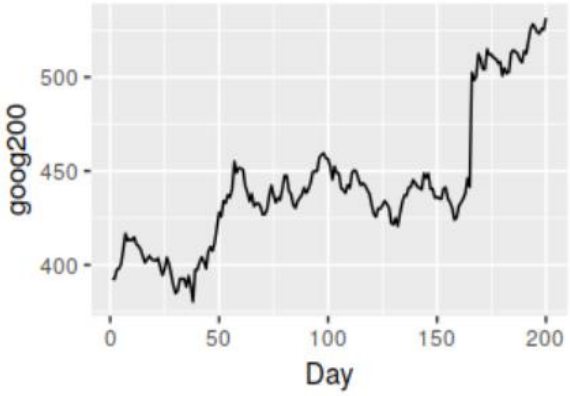
với $\emptyset_i \neq 0$, $\theta_i \neq 0$ và $\sigma_\varepsilon^2 > 0$.

2.1.4 Mô hình ARIMA

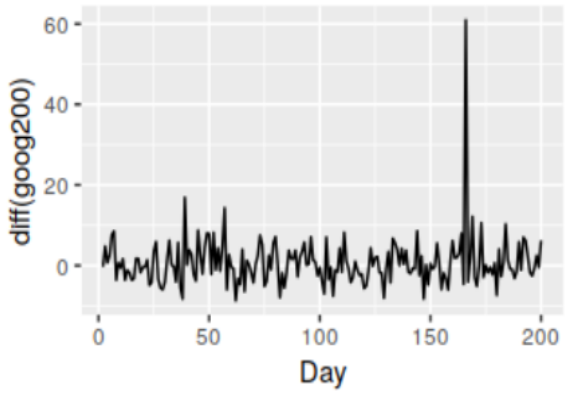
Các mô hình dự báo chuỗi thời gian như AR, MA, ARMA là những mô hình tuyến tính. Vì thế, khi áp dụng các mô hình này thì chuỗi thời gian phải có tính dừng để làm cơ sở dự báo. Một chuỗi thời gian có tính dừng (stationary time series) là một chuỗi có các giá trị không phụ thuộc vào thời điểm đang xem xét [37]. Do đó, một chuỗi thời gian có các thành phần xu hướng hoặc theo mùa không phải là một chuỗi dừng. Tóm lại, một chuỗi thời gian dừng sẽ có giá trị trung bình và hiệp phương sai không đổi. Một chuỗi thời gian $\{Y_t\}$ có tính dừng nếu thỏa mãn hai tính chất [35]:

(1) $E(Y_t) = \mu_Y(t)$

(2) $Cov(Y_t, Y_{t+k}) = \gamma_k(t)$



Hình 2.1. Chuỗi thời gian không dừng
Nguồn: *Mô hình ARIMA* [37]



Hình 2.2. Chuỗi thời gian dừng
Nguồn: *Mô hình ARIMA* [37]

Trên thực tế, không phải chuỗi thời gian nào cũng có tính dừng. Những chuỗi thời gian không dừng cần được biến đổi về dạng chuỗi thời gian dừng để áp dụng các mô hình dự báo. Cách đơn giản để biến đổi chuỗi thời gian về dạng chuỗi dừng là lấy

sai phân. Thông thường, sau một hoặc hai lần lấy sai phân thì chuỗi thời gian sẽ về dạng chuỗi thời gian dừng. Sai phân bậc 1 được tính bởi công thức:

$$y'_t = y_t - y_{t-1} \quad (2.4)$$

với y'_t là chuỗi thời gian sau khi lấy sai phân bậc 1, y_t và y_{t-1} là giá trị chuỗi thời gian tại thời điểm t và t-1.

Trong trường hợp cần lấy sai phân bậc 2 để cho chuỗi thời gian dừng thì sai phân bậc 2 được tính bởi công thức:

$$y''_t = y'_t - y'_{t-1} \quad (2.5)$$

Để xác định một chuỗi thời gian có tính dừng hay không có thể dựa vào đồ thị của chuỗi thời gian, đồ thị hàm tự tương quan mẫu hoặc kiểm định Dickey – Fuller:

- *Dựa vào đồ thị của chuỗi thời gian*: Một cách trực quan, chuỗi thời gian có tính dừng nếu như đồ thị của chuỗi thời gian cho thấy giá trị trung bình và phương sai của chuỗi thời gian không thay đổi theo thời gian.
- *Dựa vào hàm tự tương quan mẫu (Sample Auto Correlation – SAC)*: Nếu đồ thị của hàm tự tương quan mẫu của một chuỗi thời gian giảm nhanh và tắt dần về 0 thì chuỗi có tính dừng.
- *Dựa vào kiểm định Dickey – Fuller*: xác định xem chuỗi thời gian có phải là Bước ngẫu nhiên (Random walk, tức là $Y_t = Y_{t-1} + e_t$) hay không. Nếu chuỗi thời gian là một Random walk thì không có tính dừng.

Sau khi chuỗi thời gian đã dừng, mô hình ARIMA (AutoRegressive Integrated Moving Average) được sử dụng để dự báo cho chuỗi thời gian dừng sau khi lấy sai phân. Một mô hình ARIMA được biểu diễn bởi phương trình [37]:

$$y'_t = c + \emptyset_1 y'_{t-1} + \cdots + \emptyset_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \quad (2.6)$$

Mô hình ARIMA (p, d, q) gồm 3 thành phần:

- AR (Autoregressive) là mô hình tự hồi quy biểu diễn sự phụ thuộc của điểm thời gian đang xét vào một số điểm thời gian trước đó (p).
- I (Integrated) là số lần lấy sai phân để chuỗi thời gian có tính dừng (d).

- MA (Moving Average) là mô hình trung bình trượt biểu diễn sự phụ thuộc của điểm thời gian đang xét vào một vài số hạng sai số ngẫu nhiên và giá trị trung bình của các điểm thời gian trước đó (q).

2.1.5 Phương pháp Box-Jenkins

Để xây dựng và lựa chọn một mô hình ARIMA phù hợp cho dự báo chuỗi thời gian với từng tập dữ liệu, phương pháp luận Box và Jenkins được đề xuất theo hướng tiếp cận thực nghiệm [36] các mô hình để tìm ra mô hình phù hợp nhất gồm 4 bước sau:

Bước 1. Nhận dạng mô hình ARIMA (p, d, q): Xác định các giá trị thích hợp của p và q thông qua hàm tự tương quan riêng phần (PACF) và hàm tự tương quan (ACF). Giá trị d là số lần lấy sai phân để chuỗi thời gian có tính dừng.

Hàm tự tương quan (Autocorrelation Function - ACF) r_k của chuỗi thời gian dừng Y_t đo lường sự phụ thuộc tuyến tính giữa các cặp quan sát *Y(t)* và *Y(t + k)*, ứng với từng độ trễ *k = 1,2,…* Với mỗi độ trễ *k*, hàm tự tương quan tại độ trễ *k* được xác định thông qua độ lệch giữa các biến ngẫu nhiên *Y(t)* và *Y(t + k)* so với giá trị trung bình của chuỗi thời gian và được chuẩn hóa qua phương sai của chuỗi thời gian đó. Hàm tự tương quan tại các độ trễ *k* khác nhau sẽ có giá trị khác nhau. Hàm tự tương quan được định nghĩa như sau [37]:

$$r_k = \frac{\sum_{t=k+1}^{T}(y_t-\bar{y})(y_{t-k}-\bar{y})}{\sum_{t=1}^{T}(y_t-\bar{y})^2} \quad (2.7)$$

với T là số điểm thời gian trong chuỗi thời gian.

Hàm tự tương quan từng phần (Partial Autocorrelation Function - PACF) α_k tại các độ trễ *k = 1,2,..* cũng được xác định nhằm đo lường sự phụ thuộc tuyến tính giữa giá trị quan sát *Y(t)* và các giá trị trung gian trong khoảng giữa *Y(t)* và *Y(t + k)*. Hàm tự tương quan từng phần tại các độ trễ *k* khác nhau sẽ có giá trị khác nhau và được định nghĩa như sau [37]:

$$\alpha_0 = 1 \text{ và } \alpha_k = \phi_{kk}, k > 1 \quad (2.8)$$

với ϕ_{kk} là thành phần cuối cùng của vector ϕ_k

Chọn các giá trị của p tại các độ trễ mà tại đó giá trị của hàm PACF khác không về mặt thống kê. Tương tự, chọn các giá trị của q tại các độ trễ mà tại đó giá trị của hàm ACF khác không về mặt thống kê. Trong thống kê, các giá trị PACF và ACF khác không về mặt thống kê thì phải vượt qua khỏi giới hạn $\pm 1.96/\sqrt{T}$ [42].

Bước 2. Ước lượng mô hình: Dựa vào dữ liệu chuỗi thời gian để tìm các tham số của mô hình tự hồi quy và trung bình trượt. Những phương pháp được sử dụng phổ biến để ước lượng tham số như moments, bình phương tối thiểu (linear least squares), ước lượng hợp lý cực đại (maximum likelihood). Hiện nay, các phần mềm thống kê như R, Eviews, SPSS,... đã tích hợp các module giúp tự động tính toán, ước lượng các tham số này.

Bước 3. Kiểm định mô hình: Xem xét mô hình đã xây dựng có phù hợp với tập dữ liệu hay không. Một cách kiểm định đơn giản là xem xét phần dư dự báo từ mô hình có tính ngẫu nhiên thuần túy hay không. Nếu có thì mô hình có thể được chấp nhận, còn không thì phải thực hiện lại bước nhận dạng mô hình và ước lượng mô hình đến khi nào tìm được mô hình có thể chấp nhận được.

$$\varepsilon_t = X_t - \widehat{X_t} \tag{2.9}$$

với ε_t là phần dư, Y_t là giá trị thực tế, $\widehat{Y_t}$ là giá trị dự báo.

Trong trường hợp có nhiều hơn một mô hình ARIMA phù hợp với dữ liệu chuỗi thời gian. Khi đó, cần chọn một mô hình ARIMA phù hợp nhất với dữ liệu chuỗi thời gian để dự báo. Các tiêu chuẩn AIC (Akaike's Information Criterion), BIC (Bayesian Information Criterion) hỗ trợ việc kiểm định này. Mô hình ARIMA nào có các giá trị này bé nhất thì mô hình ARIMA đó được chọn để làm mô hình cho dự báo.

$$AIC = -2\log(L) + 2(p + q + k + 1) \tag{2.10}$$

với L giá trị của hàm likelihood trong thống kê, k =1 nếu c ≠ 0 và k = 0 nếu c = 0.

$$BIC = AIC + [log(T) - 2](p + q + k + 1) \tag{2.11}$$

Bước 4. Dự báo chuỗi thời gian: Sử dụng mô hình ARIMA phù hợp nhất với tập dữ liệu để dự báo tại các thời điểm t trong tương lai của chuỗi thời gian. Đối với chuỗi thời gian Y_t có tính dừng, giá trị dự báo $\widehat{Y_t}$ tại thời điểm t cũng chính là giá trị dự báo

của chuỗi thời gian tại thời điểm t. Trong trường hợp chuỗi thời gian Y_t không có tính dừng và giả sử lấy sai phân sai phân 1 lần để chuỗi có tính dừng thì $\widehat{Y_t} = Y_{t-1} + \widehat{Y'_t}$.

2.2 Các mô hình máy học

2.2.1 Mô hình mạng nơron truyền thẳng

Mạng nơron nhân tạo lấy ý tưởng từ việc mô phỏng hoạt động của não bộ con người. Mạng nơron nhân tạo có nhiều kiến trúc khác nhau như mạng nơron truyền thẳng, mạng nơron tích chập, mạng nơron hồi quy. Trong đó, mạng nơron truyền thẳng (Feedforward Neural Network – FFNN) bao gồm một lớp đầu vào (input layer), một hoặc nhiều lớp ẩn (hidden layer), lớp đầu ra (output layer). Số đặc trưng của tập dữ liệu sẽ tương ứng với số nơron trong lớp đầu vào. Tất cả các nơron này được kết nối với mỗi nơron trong lớp ẩn thông qua các đường liên kết gọi là "khớp thần kinh". Mỗi "khớp thần kinh" sẽ được gán một trọng số (weight). Các trọng số này sẽ được điều chỉnh trong quá trình học của mạng nơron nhân tạo để mô hình hóa mối liên hệ giữa lớp đầu vào và đầu ra.

Trong lớp ẩn, mỗi nơron sẽ thực hiện một hàm kích hoạt (thường là hàm sigmoid, tanh hoặc relu) có chức năng tính toán các giá trị đầu vào kết hợp với trọng số để gửi đến lớp đầu ra. Các hàm kích hoạt thường dùng như hàm sigmoid, tanh hoặc ReLU (Rectified Linear Unit):

$$\text{sigmoid(x)} = \frac{1}{1 + \exp(-x)} \tag{2.12}$$

$$\tanh(x) = \frac{sinh\,(x)}{cosh\,(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.13}$$

$$\text{ReLU(x)} = \max(0,x) \tag{2.14}$$

Lớp đầu ra sẽ nhận các giá trị từ lớp ẩn và tính giá trị đầu ra của mô hình.

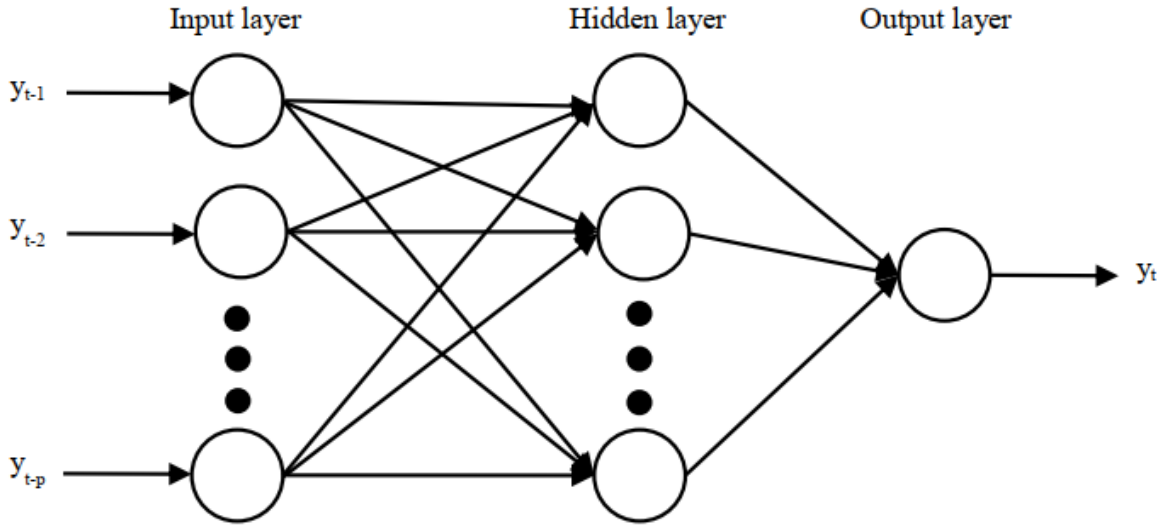
$$\hat{y}(x) = \text{output(x)} = f(w^T x + b) \tag{2.15}$$

Các giá trị thực tế trong tập huấn luyện và giá trị được tính ra bởi mô hình mạng nơron sẽ có sự sai số (error). Để mạng nơron có thể mô hình mối liên hệ giữa lớp đầu vào và đầu ra một cách tốt nhất thì mạng nơron sẽ trải qua một quá trình học hỏi gồm nhiều lần lặp đi lặp lại (epochs). Bản chất của quá trình này là mô phỏng độ lỗi dự

báo bởi một phương trình f(x) và tìm giá trị cực tiểu của phương trình này thông qua việc cập nhật các trọng số giữa các nơron trong mạng. Quá trình học hỏi này thường được thực hiện với thuật toán lan truyền ngược (Backpropagation) và thuật toán gradient descent.

Thuật toán lan truyền ngược gồm 2 giai đoạn [33]:

- **Giai đoạn 1 (forward pass)**: Ở lần học đầu tiên, các trọng số sẽ được khởi tạo và thông qua hàm kích hoạt để dự đoán kết quả output.
- **Giai đoạn 2 (backward pass)**: Các kết quả output từ mô hình sẽ có độ lỗi. Độ lỗi này sẽ được mô hình hóa bằng một hàm số *loss*(y, $\hat{y}$). Để làm cho mô hình dự đoán chính xác hơn (giảm độ lỗi) thì cần tìm giá trị cực tiểu của hàm số trên. Thuật toán gradient descent giúp thực hiện việc này thông qua cách lấy đạo hàm của hàm hợp (chain rule) rồi sau đó cập nhật lại các trọng số trong mô hình với giá trị cực tiểu tìm được để độ lỗi dần về 0.



Hình 2.3. Kiến trúc mạng nơron truyền thẳng cho dự báo chuỗi thời gian
Nguồn: *Áp dụng mạng nơron cho dự báo chuỗi thời gian* [8]

Trong dự báo chuỗi thời gian, mô hình mạng nơron FFNN sử dụng đặc trưng cho lớp đầu vào là các giá trị ở những điểm thời gian trước điểm thời gian dự báo. Mối liên hệ giữa giá trị đầu ra (y_t) và các giá trị đầu vào (y_{t-1}, y_{t-2}, …, y_{t-p}) được mô hình bởi phương trình [12]:

$$y_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j g(\beta_{0j} + \sum_{i=1}^{p} \beta_{ij}\, y_{t-i}) + \varepsilon_t \quad (2.16)$$

trong đó, α_j (j=0, 1, 2, …, q) và β_{ij} (i=0, 1, 2, …, p, j=1, 2, …, q) là các tham số của mô hình, p là số nơron lớp đầu vào và q là số nơron lớp ẩn, ε_t là sai số. Hàm kích hoạt được sử dụng trong các nơron lớp ẩn như hàm sigmoid, ReLU. Do đó, mô hình mạng nơron FFNN trong dự báo chuỗi thời gian là một mô hình phi tuyến mô tả mối quan hệ giữa các giá trị trong quá khứ (y_{t-1}, y_{t-2}, …, y_{t-p}) và giá trị tương lai (y_t) [12]:

$$y_t = f(y_{t-1}, y_{t-2}, ..., y_{t-p}, w) + \varepsilon_t \quad (2.17)$$

với w là một vector chứa tất cả các tham số của mô hình FFNN, f là một hàm số được xác định bởi cấu trúc mạng và các tham số.

Việc chọn số nơron đầu vào p và số nơron của lớp ẩn q phụ thuộc vào tập dữ liệu huấn luyện. Mỗi tập dữ liệu chuỗi thời gian khi huấn luyện với mô hình mạng nơron sẽ có p, q khác nhau. Chọn p, q để tìm được mô hình dự báo chuỗi thời gian tốt nhất phải qua thực nghiệm và so sánh giữa các mô hình.

Trong những năm gần đây, nhiều nghiên cứu đã áp dụng các mô hình mạng nơron nhân tạo trong dự báo chuỗi thời gian và đạt được một số kết quả nhất định. Bogdan Oancea [5] đã cài đặt mô hình mạng nơron với hai kiến trúc mạng là FFNN và RNN (Recurrent Neural Network) cho việc dự báo chuỗi thời gian. Tác giả chạy thực nghiệm trên tập dữ liệu tỷ giá hối đoái giữa đồng EUR/RON và USD/RON. Đầu tiên, tác giả chuẩn hóa dữ liệu bằng công thức logarit tự nhiên để nâng cao độ chính xác dự báo. Sau đó, mô hình FFNN được xây dựng với 20 nơron ở lớp đầu vào (input layer), 40 nơron ở lớp ẩn (hidden layer) và 1 nơron ở lớp đầu ra (output layer) là giá trị dự báo cho thời gian tiếp theo t+1. Tác giả chia tập dữ liệu với 80% cho huấn luyện (training) và 20% cho thử nghiệm (testing). Kế tiếp, mạng RNN được cài đặt với 20 nơron ở lớp đầu vào, 10 nơron trong lớp ẩn hồi quy và 1 nơron ở lớp đầu ra. Sau khi thực nghiệm, tác giả khẳng định mô hình RNN cho kết quả dự báo tốt hơn FFNN trên tập dữ liệu tỷ giá hối đoái.

M. Raeesi [8] sử dụng mạng nơron FFNN để dự báo dữ liệu giao thông ở thành phố Monroe, bang Louisiana, Hoa Kỳ. Nghiên cứu này đề xuất một mạng nơron sử dụng dữ liệu giao thông của ngày hôm nay, ngày hôm qua, tuần trước, hai tuần trước,

ba tuần trước và một tháng trước để làm đầu vào cho dự báo lưu lượng giao thông của ngày mai. Kết quả thực nghiệm cho thấy mô hình mạng nơron đã xây dựng có thể được sử dụng cho dự báo giao thông tại thành phố Monroe. Tuy nhiên, một vài trường hợp có kết quả dự báo với sai số lớn do những yếu tố bất thường tác động như tai nạn, thời tiết xấu,… Kumar Abhishek [7] cũng sử dụng mạng nơron FFNN với giải thuật lan truyền ngược (back-propagation) trong dự báo chứng khoán trên tập dữ liệu của tập đoàn Microsoft từ 1/1/2011 đến 31/12/2011 gồm 2 lớp đơn giản trong mạng (10 nơron lớp đầu vào, 1 nơron lớp đầu ra), độ chính xác dự báo lên đến 99%.

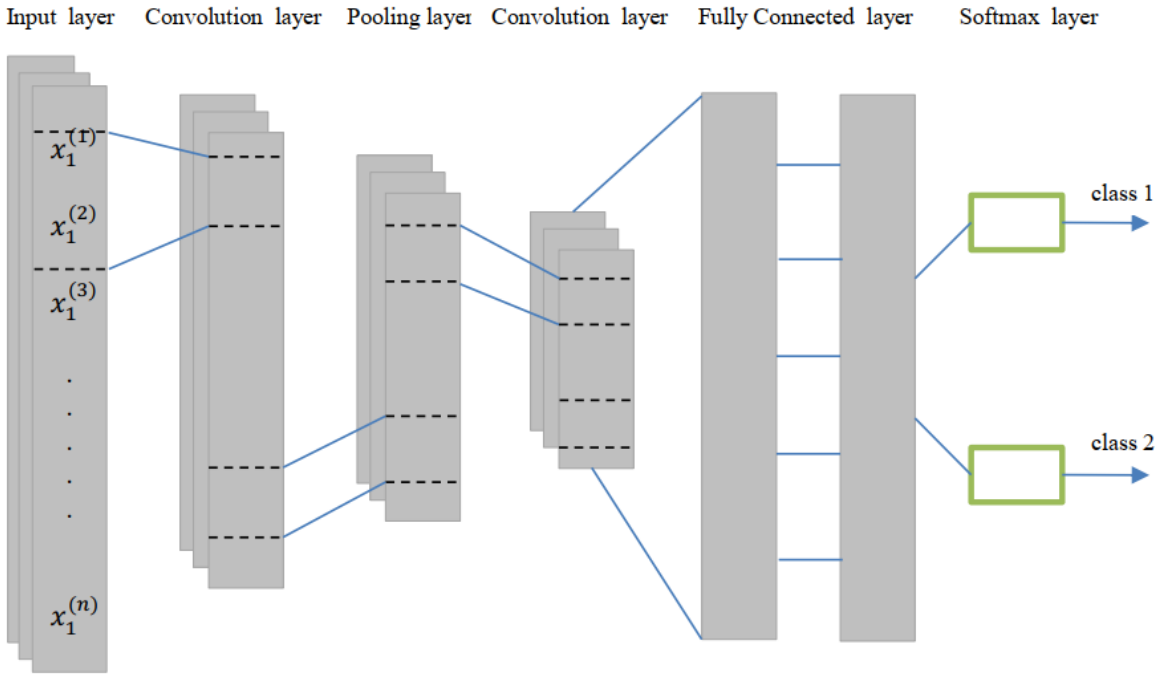
2.2.2 Mô hình mạng nơron tích chập

Mô hình mạng nơron tích chập (CNN) là một mô hình mạng nơron truyền thẳng (FFNN) [9]. Ngoài lớp đầu vào, lớp đầu ra, mạng nơron tích chập còn có các lớp đặc trưng như lớp tích chập (convolutional), lớp lấy mẫu (pooling), lớp kết nối đầy đủ (fully connected) giống như lớp ẩn trong mạng FFNN.

Lớp đầu vào của mô hình CNN là một ma trận có số chiều [rộng x cao x sâu]. Tùy vào các bài toán cụ thể, ma trận đầu vào có thể bị giảm một số chiều. Với mạng CNN, ma trận đầu vào khi qua lớp tích chập sẽ thực hiện phép tích chập (convolutional) với các bộ lọc (filters) để tạo ra một ma trận có số chiều nhỏ hơn ma trận đầu vào. Ma trận vừa được tạo ra tiếp tục thực hiện phép lấy mẫu (pooling) để giúp rút trích đặc trưng quan trọng từ lớp đầu vào. Ma trận kết quả của phép lấy mẫu sẽ được làm phẳng (flatten) để làm đầu vào cho lớp kết nối đầy đủ thực hiện như một mạng nơron truyền thẳng và đưa ra kết quả dự báo.

Mô hình CNN được đề xuất với mục tiêu ban đầu nhằm phục vụ cho công việc xử lý, nhận dạng hình ảnh. Thông thường, đầu vào của mạng CNN là một ma trận hai chiều hoặc ba chiều nhưng khi áp dụng cho dự báo chuỗi thời gian thì mạng CNN phải xử lý trên dữ liệu mảng một chiều. Tuy nhiên, với những kết quả tốt mạng lại trong xử lý ảnh, một số nghiên cứu đã thử nghiệm áp dụng mạng nơron tích chập trong dự báo chuỗi thời gian. Chẳng hạn, trong nghiên cứu của Sheng Chen [9], mô hình CNN được áp dụng cho việc dự báo sự tăng giảm của chứng khoán. Mạng CNN được huấn luyện trên tập dữ liệu chứng khoán với đầu vào gồm các yếu tố được cho

là sẽ ảnh hưởng đến sự tăng giảm của giá chứng khoán là giá mở cửa, giá đóng cửa, giá cao nhất, giá thấp nhất và khối lượng giao dịch trong ngày. Với thực nghiệm của mình, tác giả cho rằng mô hình CNN cũng đáng tin cậy trong dự báo giá chứng khoán, là một phương pháp rất đáng để thử nghiệm.



Hình 2.4. Kiến trúc mạng nơron tích chập cho dự báo chuỗi thời gian

Nguồn: *Đề xuất mô hình mạng nơron tích chập cho dự báo chuỗi thời gian* [9]

2.2.3 Mô hình SVM

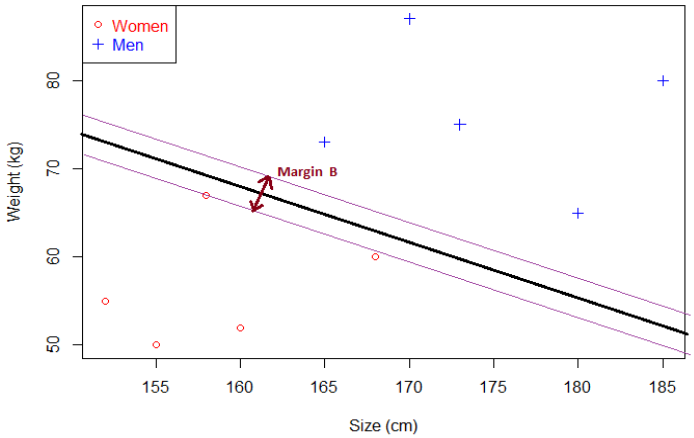
Thuật toán SVM được đề xuất để giải quyết bài toán phân lớp nhị phân bằng cách tìm ra một siêu phẳng (hyperlane) tối ưu để phân lớp dữ liệu sao cho tối đa khoảng cách (margin) giữa các lớp dữ liệu. Trong thuật toán SVM, có hai khái niệm cần làm rõ đó là siêu phẳng (hyperlane) và khoảng cách (margin).

Siêu phẳng (hyperlane) mô tả một "mặt phẳng" giúp phân chia các lớp đối tượng. Siêu phẳng trong không gian một chiều được xem như là một điểm, trong không gian hai chiều là một đường thẳng, trong không gian ba chiều là một mặt phẳng, từ không gian bốn chiều trở lên ta gọi chung là một siêu phẳng được biểu diễn:

$$w^T x = 0 \qquad (2.18)$$

với w, x là hai vector bất kỳ.

Khoảng cách (margin) là phần không gian có độ lớn bằng hai lần khoảng cách từ siêu phẳng đến điểm dữ liệu gần siêu phẳng nhất. Ví dụ siêu phẳng và khoảng cách trong một bài toán phần lớp đơn giản giữa 2 lớp (Men và Women). Việc phân lớp này dựa trên chiều cao (cm) và cân nặng (kg) của một người.



Hình 2.5. Khoảng cách phân lớp SVM

Vấn đề tìm kiếm một siêu phẳng tối ưu để phân lớp nhị phận tương đương với việc tìm kiếm một siêu phẳng với khoảng cách lớn nhất. Người ta chứng minh rằng một siêu phẳng tối ưu có margin lớn nhất thì siêu phẳng sẽ có $\vec{w}$ với độ lớn bé nhất. Vấn đề tìm siêu phẳng tối ưu được biểu diễn dưới dạng bài toán tối ưu như sau:

$$Maximize \frac{1}{\|\vec{w}\|}$$

$$subject\ to\ y_i(\vec{w}.\vec{x_i} + b) \geq 1 \qquad (2.19)$$

$(\forall i\ 1 \leq i \leq n)$ với n là số điểm dữ liệu cần phân lớp

Bài toán tối ưu trên cũng tương đương bài toán tối ưu sau:

$$Minimize \frac{\|\vec{w}\|^2}{2}$$

$$subject\ to\ y_i(\vec{w}.\vec{x_i} + b) \geq 1 \qquad (2.20)$$

$(\forall i\ 1 \leq i \leq n)$

Việc chọn hàm $y = \frac{x^2}{2}$ thay thế cho hàm $y = \frac{1}{x}$ trong bài toán tối ưu vì hàm $y = \frac{x^2}{2}$ thì việc tìm minimize của hàm này dễ dàng hơn.

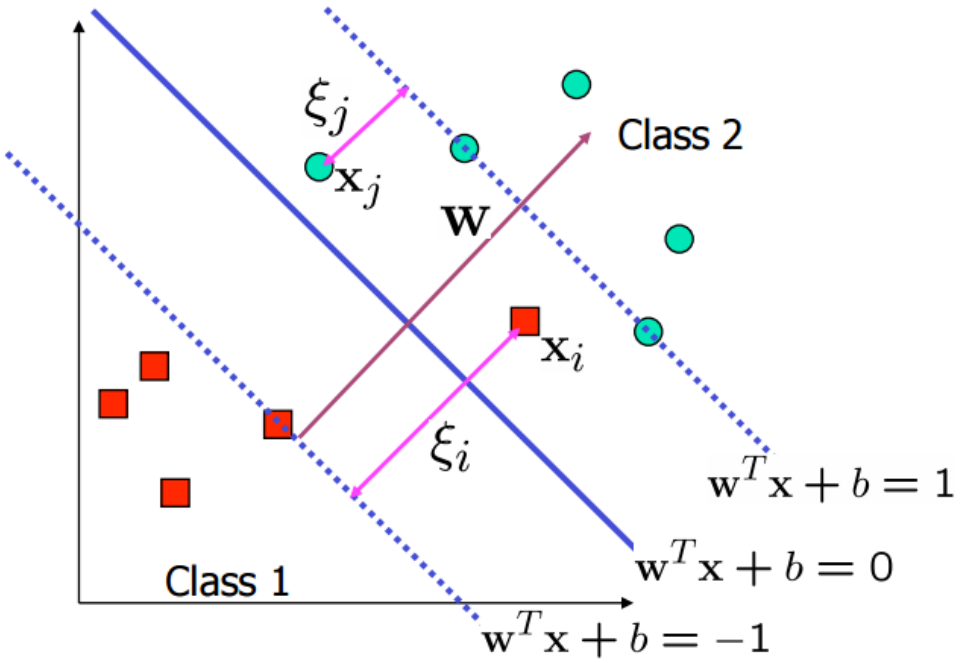
Phương pháp Lagrange multipliers giúp giải các bài toán tối ưu có ràng buộc như bài toán tối ưu margin của SVM. Áp dụng phương pháp Lagrange multipliers cho bài toán tối ưu trên, ta có bài toán tối ưu hàm Lagrange sau:

$$\textit{Maximize } \mathcal{L}(\vec{\lambda})$$

$$\textit{subject to } \lambda_i \geq 0 \text{ and } \sum_{i=1}^{n} \lambda_i y_i = 0 \qquad (2.21)$$

$$(\forall i \; 1 \leq i \leq n)$$

Phương pháp Quadratic Programming (QC) thường được sử dụng để giải bài toán trên để tìm ra w và b nhằm xác định siêu phẳng có margin lớn nhất. Tuy nhiên, trong một số trường hợp, tập dữ liệu có các giá trị nhiễu và SVM sẽ không tìm được siêu phẳng tối ưu nào để phân lớp. Bên dưới là một ví dụ giá trị nhiễu trong tập dữ liệu.



Hình 2.6. Các giá trị nhiễu trong tập dữ liệu phân lớp

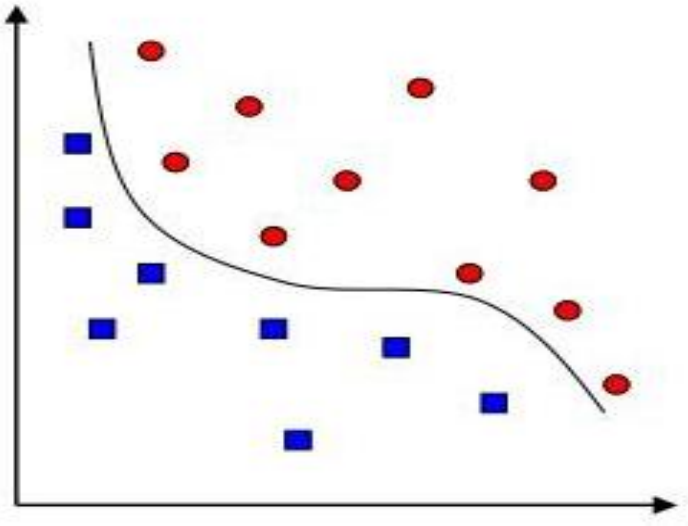
Một cải tiến của thuật toán SVM để giải quyết vấn đề trên thêm một hằng số C và margin trong trường hợp này gọi là Soft Margin:

$$\textit{Minimize } \frac{\|\vec{w}\|^2}{2} + C \sum_{i=1}^{n} \xi_i$$

$$\textit{subject to } y_i(\vec{w}.\vec{x_i} + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \qquad (2.22)$$

$$(\forall i \; 1 \leq i \leq n)$$

Một vấn đề nữa trong bài toán phân lớp bằng thuật toán SVM là các đối tượng được phân lớp có thể không được phân bố thành các cụm riêng biệt mà đan xen nhau. Do đó, những trường hợp này không thể phân lớp tuyến tính mà phải phân lớp phi tuyến bằng cách sử dụng một khái niệm gọi là Kernel $K(\overrightarrow{x_i}, \overrightarrow{x_j})$.



Hình 2.7. Các đối tượng phân bố đan xen nhau

Một số loại Kernel thường dùng như linear, polynomial hoặc Radial Basic Function (RBF). Với $K(\overrightarrow{x_i}, \overrightarrow{x_j})$, bài toán tìm một siêu phẳng của thuật toán SVM được biểu diễn lại như sau:

$$\textit{Maximize } \sum_{i=1}^{n} \lambda_i - \frac{1}{2}\sum_{i,j=1}^{n} \lambda_i \lambda_j \, y_i y_j K(\overrightarrow{x_i}, \overrightarrow{x_j})$$

$$\textit{subject to } \lambda_i \geq 0 \text{ và } \sum_{i=1}^{n} \lambda_i \, y_i = 0 \qquad (2.23)$$

$$(\forall i \; 1 \leq i \leq n)$$

Trong dự báo chuỗi thời gian, một phiên bản của SVM được sử dụng để dự báo các giá trị trong chuỗi thời gian, được gọi là Support Vector Regression (SVR). SVR được xem như là một ứng dụng của SVM trong bài toán ước lượng hồi quy. Cho chuỗi thời gian $x(t), t = 1,2 \ldots, n$. Vector $x_t = \big(x(t), x(t-\tau), \ldots, x(t-(d-1)\tau)\big)$ với τ là thời gian trễ (time delay) và d là số chiều của vector. Vector x_t được xem như một vector mô tả trạng thái của chuỗi thời gian tại thời điểm t. Nếu tìm được một hàm f để mô hình hóa cho chuỗi thời gian này thì giá trị của chuỗi thời gian tại điểm t + 1 sẽ được xác định bằng công thức $x(t+1) = f(x_t)$.

Chuyển đổi chuỗi dữ liệu thời gia thành tập dữ liệu gồm các cặp giá trị $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ với $x_i \in \mathbb{R}^m, y_i \in \mathbb{R}$ là ngẫu nhiên và độc lập. Khi đó, SVR mô tả chuỗi thời gian bằng một hàm có dạng như sau:

$$f(\vec{x}) = \vec{w}.\vec{x} + b \tag{2.24}$$

Mục tiêu của mô hình SVR là tìm kiếm một hàm f có dạng như trên sao cho độ lệch giữa giá trị của hàm f và giá trị thực tế tại các điểm thời gian không vượt một ngưỡng cho phép ε. Nói cách khác, SVR không quan tâm những độ lệch bé hơn ε và cũng không chấp nhận những độ lệch lớn hơn ε. Giá trị của $\vec{w}$ và b được xác định bằng cách *minimize* hàm rủi ro sau:

$$minimize \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n} L_\varepsilon\big(y_i, f(\vec{x_i})\big) \tag{2.25}$$

với $L_\varepsilon\big(y_i, f(\vec{x_i})\big) = \begin{cases} |y_i - f(\vec{x_i})| - \varepsilon, |y_i - f(\vec{x_i})| > \varepsilon \\ 0, |y_i - f(\vec{x_i})| \le \varepsilon \end{cases}$

Sử dụng support vectors, Kernel và Lagrange multipliers của thuật toán SVM để tìm giá trị của $\vec{w}$ và b cho hàm f(x). Hàm số này dùng để dự giá trị tại các điểm thời gian trong chuỗi thời gian.

Các nghiên cứu [10, 11] đã áp dụng SVM trong dự báo chuỗi chứng khoán. Những nghiên cứu này còn kết hợp SVM và mô hình ARIMA để nâng cao độ chính xác dự báo. Trong một nghiên cứu gần đây, Ferdiansyah [29] áp dụng SVM trong dự báo xu hướng tăng giảm của giá Bitcoin trong ngày hiện tại. Nghiên cứu cho thấy độ chính xác dự báo tốt của SVM, điều đó thể hiện SVM đã nắm bắt được xu hướng thay đổi giá của Bitcoin.

2.3 Đánh giá các mô hình dự báo chuỗi thời gian

Để đánh giá hiệu quả dự báo của các mô hình, các độ đo lỗi là công cụ hết sức cần thiết. Phần dưới trình bày 2 độ đo được sử dụng phổ biến là RMSE và MAPE.

Root Mean Squared Error (RMSE) là một độ đo cho biết sự khác biệt giữa giá trị dự báo và giá trị thực tế của chuỗi thời gian. Giá trị RMSE càng bé thì mô hình cho kết quả dự báo càng chính xác.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2} \tag{2.15}$$

với n là số điểm thời gian thử nghiệm, y_t là giá trị thực tế, $\hat{y}_t$ là giá trị dự báo từ các mô hình.

MAPE (Mean Absolute Percentage Error) là một độ đo cho biết tỉ lệ phần trăm sai lệch của kết quả dự báo so với kết quả thực tế. Giá trị MAPE càng bé thì mô hình cho kết quả dự báo càng chính xác.

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t-\hat{y}_t}{y_t}\right|*100 \qquad (2.16)$$

2.4 Kết chương

Chương này đã trình bày chi tiết một số mô hình dự báo chuỗi thời gian như các mô hình thống kê gồm AR, MA, ARMA, ARIMA và các mô hình máy học mạng nơron truyền thẳng, mạng nơron tích chập và thuật toán SVM. Các mô hình này đã có những thành công nhất định trong dự báo chuỗi thời gian nhưng cần phải cải thiện hơn nữa độ chính xác dự báo. Trong chương chương tiếp theo, mô hình Deep Learning sẽ được trình bày để áp dụng cho bài toán dự báo chuỗi thời gian. Đây là một mô hình hứa hẹn nâng cao độ chính xác dự báo.

CHƯƠNG 3 DEEP LEARNING CHO BÀI TOÁN DỰ BÁO CHUỖI THỜI GIAN

Deep Learning là một hướng nghiên cứu của máy học dựa trên ý tưởng của mạng nơron. Các mô hình Deep Learning cố gắng mô hình hóa dữ liệu ở mức cao bằng cách sử dụng nhiều lớp xử lý với cấu trúc phức tạp. Nhiều nghiên cứu đã áp dụng Deep Learning trong dự báo chuỗi thời gian [19, 20, 21, 38, 39]. Chương này trình bày các mô hình Deep Learning như RNN, LSTM cũng như thảo luận một số vấn đề huấn luyện của mạng nơron để xây dựng mô hình Deep Learning tốt nhất có thể.

3.1 Dữ liệu chuỗi thời gian cho mô hình Deep Learning

Trước khi áp dụng các mô hình Deep Learning nói riêng và mạng nơron nói chung để dự báo chuỗi thời gian, dữ liệu chuỗi thời gian cần được biến đổi để phù hợp với yêu cầu đầu vào của các mô hình này. Cho một chuỗi thời gian $\{x_t, t = 1, 2,\ldots, n\}$. Để dự báo giá trị x_t bất kỳ trong chuỗi, ta thường dựa vào các giá trị trước đó, được gọi là độ trễ (lag). Khi đó, mối liên hệ giữa giá trị output (x_t) và giá trị input (x_{t-1}, x_{t-2},…, x_{t-k}) sẽ được mô hình mạng nơron mô phỏng bằng một hàm số f có dạng:

$$x_t = f(x_{t-1}, x_{t-2},\ldots, x_{t-k}) \quad (3.1)$$

với k là độ trễ được xác định trước.

Do đó, dữ liệu chuỗi thời gian cần được biến đổi thành các cặp dữ liệu theo dạng chuỗi để làm đầu vào của mô hình nơron như sau:

$$\{(x_i, y_i)\} = \{(x_{i+k-1}, x_{i+k-2},\ldots, x_i), x_{i+k}\} \quad (3.2)$$

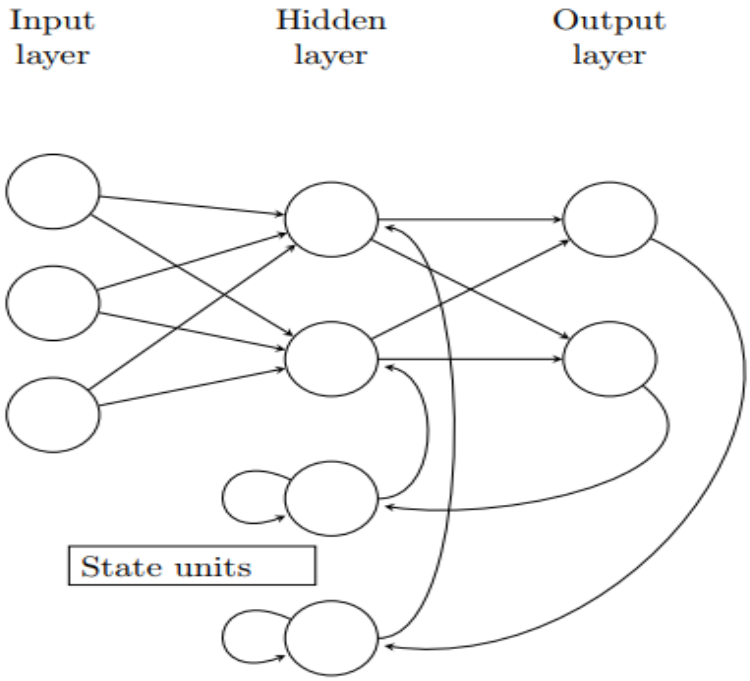
với i = 1 đến n – k là các cặp giá trị input và output sau khi biến đổi và k < n.

3.2 Mô hình mạng nơron hồi quy

Mô hình mạng nơron truyền thẳng xử lý các giá trị đầu vào một cách độc lập, không có một mối liên hệ nào giữa các giá trị này. Nhưng khi áp dụng cho chuỗi thời gian, các giá trị đầu vào trước đó sẽ rất hữu ích để giúp dự báo các giá trị tiếp theo. Do đó, nên có một mô hình mạng nơron có khả năng lưu trữ lại các thông tin trước đó của chuỗi thời gian. Mạng nơron hồi quy được sinh ra để giải quyết vấn đề này. Mạng nơron hồi quy được sử dụng nhiều trong lĩnh vực dự báo và xử lý ngôn ngữ tự nhiên.

3.2.1 Kiến trúc mạng nơron hồi quy

Mạng RNN gồm nhiều vòng lặp, mỗi vòng lặp thực hiện ở lớp ẩn (hidden layer). Khi xử lý một dữ liệu đầu vào, lớp ẩn sẽ nhận được thông tin trả về từ đầu ra được trước đó và lưu trữ vào bộ nhớ của mạng (state units). Cứ như thế, mạng RNN xử lý lặp đi lặp lại với từng mẫu trong tập dữ liệu để làm giàu bộ nhớ của mạng. Khi số mẫu dữ liệu đủ lớn, mạng có khả năng mô tả và nắm bắt quy luật của chuỗi dữ liệu.



Hình 3.1. Kiến trúc mạng nơron hồi quy

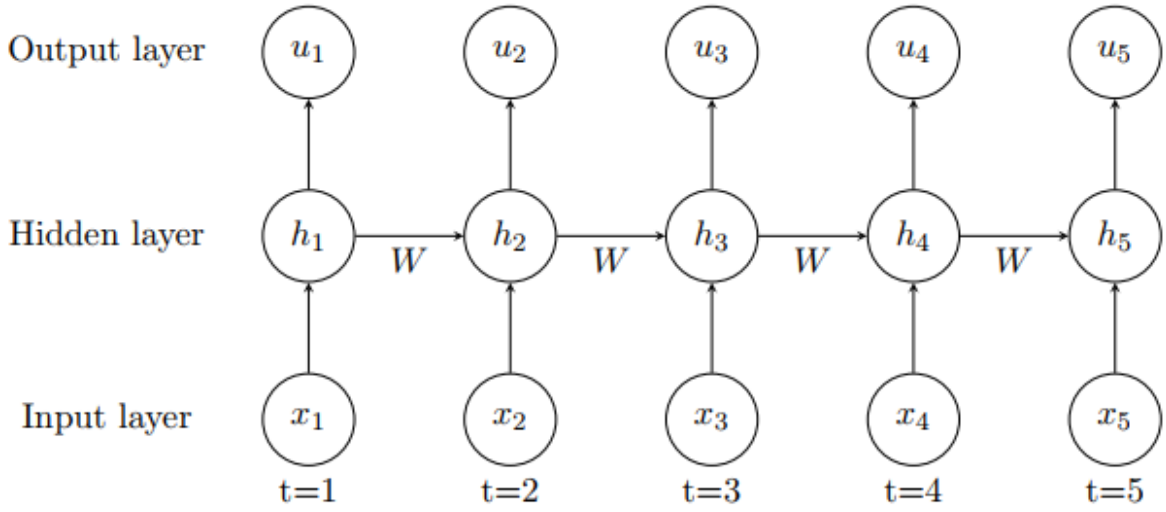
Nguồn: *Recurrent neural networks* [33]

3.2.2 Thuật toán lan truyền ngược cho mạng nơron hồi quy

Để mạng nơron hồi quy học được quy luật của chuỗi dữ liệu và làm giàu bộ nhớ của mạng, thuật toán Backpropagation through time (BPTT) được sử dụng. Ý tưởng của thuật toán BPTT dựa vào một khái niệm là *unfold* the recurrent neural network in time [33]. Tức là khi các mẫu đầu vào ở từng điểm thời gian trong chuỗi được xử lý thì các trọng số trong lớp ẩn được chia sẻ và sử dụng chung. Sau đó, các trọng số này sẽ được cập nhật như trong thuật toán Backpropagation của mạng FFNN.

Thuật toán BPTT cũng có 2 giai đoạn:

- **Giai đoạn 1 (forward pass)**: Giai đoạn này giống như ở mạng FFNN nhưng đại lượng truyền vào hàm kích hoạt ở mỗi nơron lớp ẩn ngoài giá trị đầu vào còn có thêm thông tin của điểm thời gian trước đó [33].
- **Giai đoạn 2 (backward pass)**: Kết quả dự báo tại một thời điểm trong chuỗi sẽ có sai số. Sai số này phụ thuộc vào cả thông tin được cung cấp ở điểm thời gian trước đó. Sử dụng gradient descent để tìm điểm cực tiểu của hàm lỗi và cập nhật các trọng số trong mạng RNN. Việc này lặp đi lặp lại từ điểm thời gian cuối cùng đến điểm thời gian bắt đầu trong chuỗi.



Hình 3.2. Ví dụ của Unfolded Recurrent Neural Network in time

Nguồn: *Back-propagation for recurrent neural network* [33]

3.2.3 Vấn đề phụ thuộc xa

Mạng RNN rất hữu ích trong việc nắm bắt mối liên hệ của điểm thời gian hiện tại với thông tin trước đó. Nhưng trong thực tế, RNN không thể luôn luôn thực hiện được việc này. Lý do là bởi một hiện tượng gọi là mất mát đạo hàm (vanishing gradient problem). Hiện tượng này xảy ra do khi cập nhật các trọng số thì các bước lấy đạo hàm diễn ra từ t = n,…,0. Quá trình này trải qua nhiều bước lấy đạo hàm dẫn đến đạo hàm dần về 0. Khi đó, trọng số sẽ không được cập nhật và không thể hiện được mối liên hệ giữa các điểm thời gian cách nhau quá xa (long-term dependency).

Có nhiều phương pháp để hạn chế vấn đề này như sử dụng kỹ thuật regularization hoặc sử dụng hàm kích hoạt ReLU. Một phương pháp nữa được đề xuất là thay đổi

kiến trúc của mạng RNN. Hai nhà nghiên cứu Hochreiter và Schmidhuber [18] đã đề xuất mạng LSTM để giải quyết vấn đề này.

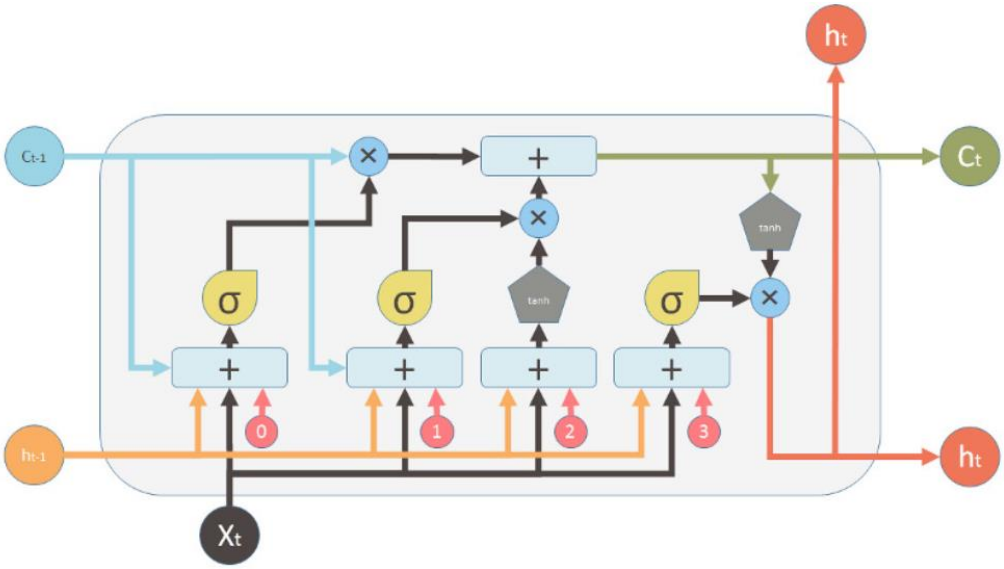
3.3 Mô hình mạng LSTM

Mô hình LSTM là một phiên bản của mạng RNN. LSTM được thiết kế để giải quyết vấn đề phụ thuộc xa. Thành phần quan trọng nhất của mạng LSTM là tế bào LSTM. Một tế bào LSTM gồm có 3 cổng chính [31]:

- **Forget gate**: $f_t = \sigma(W_f S_{t-1} + W_f S_t)$: các thông tin hiện tại rất ít có khả năng liên quan đến thông tin cách quá xa, nên cổng này quyết định khi nào nên nhớ các thông tin trước đó, khi nào thì không.
- **Input gate**: $i_t = \sigma(W_i S_{t-1} + W_i S_t)$: trị đầu vào của tế bào LSTM gồm giá trị hiện tại và giá trị trước đó.
- **Output gate**: $ot = \sigma(W_o S_{t-1} + W_o S_t)$: tính toán các giá trị đầu ra dựa vào các thông tin hiện tại và thông tin trước đó.

Mỗi tế bào LSTM còn thực hiện tính toán các trạng thái sau để lưu trữ thông tin như là bộ nhớ của mạng:

- **Intermediate Cell State**: $\tilde{C} = \tanh(W_c S_{t-1} + W_c X_t)$
- **Cell state (memory for next input)**: $c_t = (i_t * \tilde{C}_t) + (f_t * c_{t-1})$
- **Calculating new state**: $h_t = o_t * \tanh(c_t)$



Hình 3.3. Các thành phần của tế bào LSTM

Nguồn: *LSTM cell* [31]

Mỗi cổng trong tế bào LSTM đều có những trọng số riêng. Những trọng số này giúp lưu trữ các thông tin của những giá trị trước đó và tính toán giá trị đầu ra. Nhờ vậy, LSTM là một mô hình tốt để dự báo chuỗi thời gian.

3.4 Một số vấn đề huấn luyện mạng nơron

3.4.1 Tối ưu hóa mạng nơron

Gradient descent là một thuật toán giúp tìm giá trị cực tiểu của hàm lỗi (loss function) để đảm bảo việc huấn luyện hiệu quả và tìm được mô hình tốt nhất có thể. Thuật toán Gradient descent hướng đến việc tìm kiếm các điểm cực tiểu cục bộ từng bước một thông qua cập nhật điểm khởi tạo θ cho đến khi đạt kết quả chấp nhận được [33]:

$$\theta = \theta - \eta \nabla Q(\theta) \tag{3.3}$$

với η là tỉ lệ học (learning rate), $\nabla Q(\theta)$ là đạo hàm của hàm lỗi tại θ.

Gradient descent thường không được áp dụng trong huấn luyện mạng nơron vì nó dễ tìm đến local minimum không mong muốn của hàm lỗi. Stochastic Gradient Descent (SGD) là một biến thể của Gradient descent tỏ ra hiệu quả hơn trong huấn luyện. Thay vì tính đạo hàm của hàm lỗi trên toàn bộ tập huấn luyện, ta sẽ tính đạo hàm trên một số lượng nhỏ các mẫu (minibatches) trong tập huấn luyện. Số lượng các mẫu trong một minibatch gọi là *batch size*. Với SGD thì *batch size* = 1, ta được công thức cập nhật θ như sau:

$$\theta = \theta - \eta \nabla \sum_{i=1}^{n} Q(\theta, x_i, y_i) \tag{3.4}$$

với $Q(\theta, x_i, y_i)$ là hàm lỗi với chỉ 1 cặp dữ liệu (x_i, y_i), n là số mẫu trong tập dữ liệu.

SGD cũng có nhiều biến thể có nó nhằm giúp tìm ra điểm cực tiểu của hàm lỗi một cách hiệu quả và linh hoạt nhất. Một số thuật toán cải tiến như RMSProp hay Adam [33] đã được đề xuất. Ngoài ra, ta còn có khái niệm nữa là mini-batch Gradient descent. Với SGD thì *batch size* = 1, Batch Gradient descent thì *batch size* = n (số mẫu trong tập dữ liệu), mini-batch Gradient descent thì *batch size* là một số lơn hơn 1 và vẫn nhỏ hơn n rất nhiều.

3.4.2 Kỹ thuật Regularization

Một trong những thách thức lớn nhất trong việc sử dụng Deep Learning nói riêng và mạng nơron nói chung là hiện tượng underfitting và overfitting. Underfitting là hiện tượng mà mô hình huấn luyện không thể mô tả tốt tập dữ liệu, độ lỗi huấn luyện còn

lớn. Hiện tượng này có thể xảy ra khi số lần huấn luyện không đủ hoặc tình trạng thiếu dữ liệu. Overfitting là hiện tượng mà mô hình huấn luyện được mô tả quá chính xác tập dữ liệu huấn luyện nhưng không thể dự báo tốt trên tập dữ liệu kiểm tra. Hiện tượng này có thể xảy ra khi mô hình huấn luyện quá phức tạp hoặc trong tập dữ liệu có quá nhiều giá trị nhiễu và mạng nơron đã không phân biệt được các giá trị này. Có 2 phương pháp giúp hạn chế các hiện tượng này là Early stopping và Dropout.

Early stopping là một phương pháp hiệu quả để xác định số lần huấn luyện (epochs) của mạng nơron. Early stopping theo dõi độ chính xác của mô hình mạng nơron trong quá trình huấn luyện. Nếu phát hiện thấy độ lỗ không còn được cải thiện thì sẽ cho dừng việc học lại. Phương pháp này cần dữ liệu training và validation để thực hiện việc theo dõi đó. Early stopping là một trong những kỹ thuật được sử dụng phổ biến nhất trong Deep Learning, bởi vì tính đơn giản và hiệu quả của nó trong việc giảm thời gian đào tạo.

Dropout là một phương pháp phổ biến để hạn chế hiện tượng overfitting khi huấn luyện mạng nơron. Hiện tượng overfitting xảy ra chủ yếu do mô hình quá phức tạp với nhiều tham số, nhiều nơron và nhiều lớp ẩn. Dropout sẽ tạm thời bỏ bớt một số nơron một cách ngẫu nhiên trong các lớp ẩn theo một tỷ lệ được xác định trước (thường từ 0.2 đến 0.5). Từ đó, Dropout giúp điều chỉnh kiến trúc mô hình mạng nơron để phù hợp và có khả năng mô tả được tập dữ liệu một cách tốt nhất có thể.

3.5 Kết chương

Mô hình Deep Learning rất hứa hẹn trong việc nâng cao độ chính xác dự báo của chuỗi thời gian. Bởi Deep Learning có khả năng tự tự động học mối liên hệ giữa các giá trị trong chuỗi thời gian, tự động phát hiện yếu tố xu hướng và mùa vụ trong chuỗi dữ liệu. Tuy nhiên, mô hình Deep Learning gặp khó khăn trong việc lựa chọn mô hình phù hợp với tập dữ liệu. Do đó, các kỹ thuật tối ưu mạng nơron và kỹ thuật Regularization được đề xuất và phần nào giải quyết các khó khăn đó. Trong chương tiếp theo, các phương pháp kết hợp Deep Learning và một số mô hình khác sẽ được trình bày với kỳ vọng nâng cao hơn nữa độ chính xác dự báo.

CHƯƠNG 4 KẾT HỢP DEEP LEARNING VỚI MỘT SỐ MÔ HÌNH ĐỂ NÂNG CAO ĐỘ CHÍNH XÁC DỰ BÁO

Mỗi mô hình dự báo chuỗi thời gian đều có ưu điểm, khuyết điểm riêng và thông thường chỉ phù hợp với một loại dữ liệu nhất định. Các phương pháp kết hợp là rất hữu ích trong việc nâng cao độ chính xác dự báo. Trong chương này, các phương pháp kết hợp mô hình Deep Learning với các mô hình khác sẽ sẽ được trình bày với kỳ vọng sẽ nâng cao độ chính xác dự báo chuỗi thời gian.

4.1 Kết hợp Deep Learning và mô hình ARIMA

4.1.1 Ý tưởng

Mô hình ARIMA và các mô hình mạng nơron đều đã đạt được những thành công nhất định trong dự báo chuỗi thời gian. Tuy nhiên, mỗi mô hình thường chỉ phù hợp với một số tập dữ liệu nhất định. Mô hình ARIMA phù hợp với dự báo dữ liệu chuỗi thời gian dạng tuyến tính, còn mô hình mạng nơron lại phù hợp với dự báo dữ liệu chuỗi thời gian dạng phi tuyến tính. Do đó, mô hình kết hợp giữa ARIMA và mạng nơron được đề xuất với kỳ vọng có thể giúp tăng độ chính xác của dự báo trong các ứng dụng thực tế. Ý tưởng này được G. Peter Zhang giới thiệu trong nghiên cứu [12]. Mô hình kết hợp được tác giả thực nghiệm trên ba tập dữ liệu là Wolf's sunspot, Canadian lynx và tỷ giá hối đoái giữa British pound/US dollar. Kết quả thực nghiệm cho thấy các mô hình kết hợp có độ lỗi dự báo ít hơn đáng kể so với từng mô hình ARIMA và mạng nơron riêng lẻ.

Ý tưởng của mô hình này dựa trên việc xem xét dữ liệu chuỗi thời gian là sự kết hợp giữa thành phần tuyến tính và phi tuyến tính. Hai thành phần này được biểu diễn qua phương trình [12]:

$$y_t = L_t + N_t \tag{4.1}$$

với y_t là giá trị của chuỗi thời gian, L_t là thành phần tuyến tính, N_t là thành phần phi tuyến tính.

Để dự báo giá trị của chuỗi thời gian, mô hình ARIMA được sử dụng để dự báo cho thành phần tuyến tính. Những giá trị dự báo lỗi từ mô hình ARIMA sẽ được dự

báo bằng mạng nơron. Gọi e_t là giá trị còn lại sau khi sử dụng mô hình ARIMA để dự báo, e_t được xác định bởi phương trình:

$$e_t = y_t - \widehat{L_t} \quad (4.2)$$

với $\widehat{L_t}$ là giá trị dự báo cho thành phần tuyến tính tại thời điểm t.

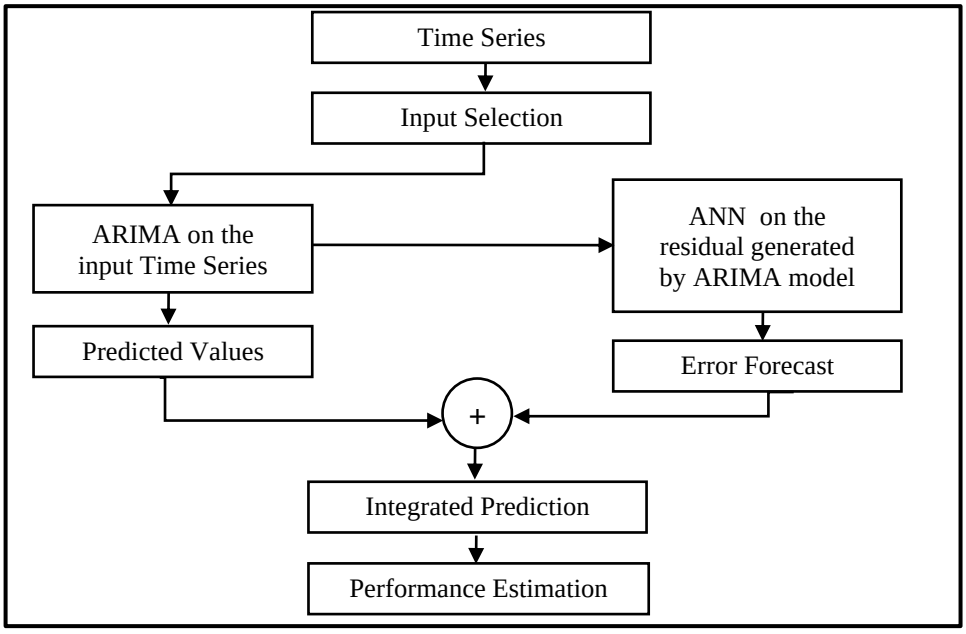
Mô hình mạng nơron được dùng để dự báo giá trị còn lại e_t sau khi dự báo bằng mô hình ARIMA sẽ được mô hình hóa bởi một hàm số:

$$e_t = f(e_{t-1}, e_{t-2}, \dots, e_{t-n}) + \varepsilon_t \quad (4.3)$$

với f là một hàm phi tuyến được xác định bằng mạng nơron, ε_t là giá trị ngẫu nhiên tại thời điểm t.

Ký hiệu $\widehat{N_t}$ là giá trị dự báo cho thành phần phi tuyến tính. Kết quả giá trị dự báo tại thời điểm t ($\widehat{y_t}$) được tính bởi phương trình:

$$\widehat{y_t} = \widehat{L_t} + \widehat{N_t} \quad (4.4)$$



Hình 4.1. Mô hình kết hợp ARIMA và mạng nơron

Nguồn: *Đề xuất phương pháp Hybrid* [16]

Phương pháp kết hợp ARIMA và mạng nơron là một phương pháp tốt để nâng cao độ chính xác dự báo chuỗi thời gian. Với ý tưởng được đề xuất bởi G. Peter Zhang [12], phương pháp này có thể được mở rộng bằng cách kết hợp ARIMA với các thuật toán máy học nói chung và Deep Learning nói riêng như LSTM, FFNN, CNN và SVR. Những kết hợp này hứa hẹn nâng cao độ chính xác dự báo thay vì chỉ kết hợp ARIMA và mạng nơron truyền thống.

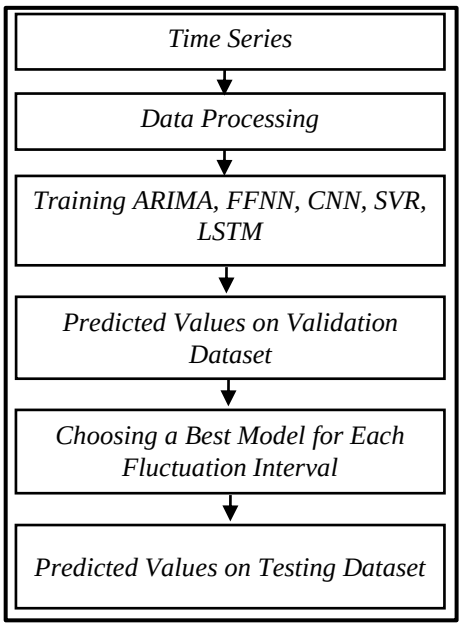
4.1.2 Các nghiên cứu liên quan

Xuất phát từ ý tưởng của G. Peter Zhang [12], nhiều nghiên cứu đã áp dụng phương pháp kết hợp này trong dự báo chuỗi thời gian ở nhiều lĩnh vực khác nhau [10, 11, 13, 14, 15, 17]. Kết quả thực nghiệm cho thấy hầu hết mô hình kết hợp đều cho độ chính xác dự báo tốt hơn từng mô hình riêng lẻ.

Ngoài ra, khi nghiên cứu dự báo xu hướng của giá chứng khoán, Nitin Merh [16] đã áp dụng mô hình kết hợp ARIMA và mạng nơron theo hai hướng tiếp cận. Hướng tiếp cận thứ nhất là xây dựng mô hình kết hợp ARIMA_ANN. Mô hình kết hợp này giống với ý tưởng của G. Peter Zhang [12] đề xuất, tức là áp dụng mô hình ARIMA để dự báo thành phần tuyến tính, sau đó sử dụng mô hình mạng nơron dự báo lỗi của mô hình (thành phần phi tuyến). Hướng tiếp cận thứ hai là xây dựng mô hình kết hợp ANN_ARIMA. Nitin Merh [16] thực hiện kết hợp ngược lại, xây dựng mô hình mạng nơron cho dự báo rồi sau đó áp dụng mô hình ARIMA để dự báo lỗi của mô hình mạng nơron. Kết quả thực nghiệm trên các tập dữ liệu chứng khoán như SENSEX, BSE IT, BSE Oil & Gas, BSE 100 and S& P CNX Nifty cho kết quả khá thú vị. Hầu hết các mô hình dự báo ANN_ARIMA đều có độ lỗi dự báo thấp hơn mô hình ARIMA_ANN.

4.2 Kết hợp Deep Learning và một số mô hình dựa trên độ biến động

4.2.1 Sơ đồ luồng xử lý kết hợp



Hình 4.2. Sơ đồ luồng xử lý kết hợp các mô hình dựa trên độ biến động

Một trong những thách thức lớn nhất trong dự báo chuỗi thời gian là tính biến động ngẫu nhiên của nó. Vẫn chưa có một mô hình thật sự hiệu quả nào để dự báo các chuỗi thời gian có biến động lớn. Trong phần này, luận văn trình bày một phương pháp kết hợp các mô hình dựa trên khoảng biến động để nâng cao dự báo cho vấn đề này. Giả sử mỗi mô hình sẽ phù hợp để dự báo trên một khoảng biến động nhất định. Ví dụ, mô hình ARIMA là mô hình tuyến tính, có khả năng dự báo tốt trên chuỗi thời gian dừng, tức là dự báo tốt trên khoảng biến động ít. Trong khi đó, các mô hình máy học và SVR có khả năng mô tả các thành phần không tuyến tính trong chuỗi thời gian, tức là dự báo tốt trên khoảng biến động lớn. Do đó, phương pháp kết hợp này rất hứa hẹn để nâng cao dự báo.

4.2.1 Các bước xây dựng mô hình kết hợp

Đầu tiên, việc tính toán độ biến động giữa 2 điểm thời gian liên tiếp sẽ được thực hiện bằng cách tính phần trăm giá trị chênh lệch của 2 điểm thời gian bởi công thức:

$$\text{fluctuation}(y_t, y_{t+1}) = \left|\frac{y_t - y_{t+1}}{y_t}\right| * 100 \quad (4.5)$$

với y_t là giá trị đang xét, y_{t+1} là giá trị kế tiếp.

Trên mỗi tập dữ liệu cụ thể, độ biến động ít nhất (min fluctuation) và cao nhất (max fluctuation) sẽ được xác định. Tần suất của độ biến động sẽ được thống kế và là cơ sở cho việc chia khoảng biến động (fluctuation interval) sau đó. Trong một số trường hợp, có một vài khoảng biến động lớn, ít xuất hiện thì ta có thể xem như trường hợp ngoại lệ và có thể điều chỉnh lại độ biến động min và max.

Kế tiếp, độ biến động sẽ được chia thành nhiều khoảng bởi các phương pháp thống kê. Các khoảng biến động có thể được xác định theo một số quy tắc sau:

- Đối với hầu hết tập dữ liệu, chia từ 6 đến 15 khoảng biến động là đủ.
- Các khoảng biến động phải bằng nhau.
- Mỗi độ biến động phải thuộc về chỉ một khoảng biến động.

Với một chuỗi dữ liệu đơn lẻ, Herbert A. Sturges [34] đề xuất một công thức xác định khoảng dữ liệu như sau:

$$c = \frac{R}{1 + 3.322 \log N} \quad (4.6)$$

với R là khoảng thống kê của chuỗi dữ liệu, N là số điểm dữ liệu trong chuỗi, c là độ lớn của mỗi khoảng dữ liệu có thể chia.

Việc chia khoảng biến động của chuỗi thời gian có thể áp dụng công thức trên để xác định số khoảng biến động với R là khoảng giữa max fluctuation và min fluctuation và N là số điểm thời gian được tính độ biến động.

Cuối cùng, các mô hình ARIMA, FFNN, CNN, LSTM, SVR sẽ được huấn luyện trên tập training và dự báo trên tập validation. Dựa vào kết quả dự báo đó, mỗi khoảng biến động sẽ được xác định một mô hình cho kết quả dự báo tốt nhất trên khoảng biến động đó và sử dụng để dự báo trên tập testing. Tóm lại, phương pháp kết hợp này có thể được thực hiện qua một vài bước sau:

Bước 1. Tính độ biến động giữa 2 điểm thời gian liên tiếp.

Bước 2. Tìm độ biến động nhỏ nhất (min fluctuation) và cao nhất (max fluctuation) trong chuỗi thời gian để làm cơ sở chia khoảng biến động.

Bước 3. Xác định số khoảng biến động được chia theo phương pháp thống kê.

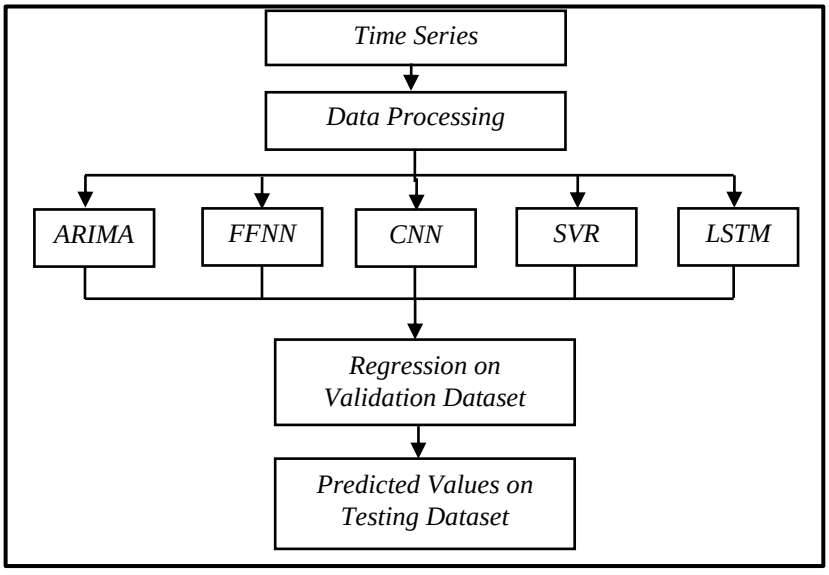
Bước 4. Huấn luyện các mô hình ARIMA, FFNN, CNN, LSTM, SVR trên tập dữ liệu training.

Step 5. Thực hiện dự báo trên tập validation với các mô hình đã huấn luyện và xác định độ biến động tại các điểm thời gian cần dự báo trong tập validation là độ biến động của 2 điểm thời gian trước đó fluctuation(y_t) = fluctuation(y_{t-2},y_{t-1}).

Bước 6. Xác định một quy tắc chọn mô hình dự báo tốt nhất trên từng khoảng biến động và dự báo trên tập testing.

4.3 Kết hợp Deep Learning và các mô hình bằng phương trình hồi quy

4.3.1 Sơ đồ luồng xử lý kết hợp



Hình 4.3. Sơ đồ luồng xử lý kết hợp các mô hình bằng phương trình hồi quy

Các biến đầu vào của mô hình kết hợp bằng phương trình hồi quy là những giá trị dự báo từ các mô hình ARIMA, FFNN, CNN, LSTM, SVR và giá trị đầu ra sẽ là giá trị thực tế trong chuỗi thời gian. Số lượng biến đầu vào phụ thuộc vào từng thực nghiệm. Chẳng hạn, nếu chỉ sử dụng kết quả dự báo từ mô hình LSTM cho phương trình hồi quy thì số biến đầu vào chỉ là 1. Chúng ta cũng có thể sử dụng kết quả dự báo của tất cả các mô hình cho phương trình hồi quy thì số lượng biến đầu vào sẽ là 5. Các mô hình sẽ được huấn luyện trên tập dữ liệu training và những giá trị dự báo trên tập validation sẽ là biến đầu vào của phương trình hồi quy. Sau đó, sử dụng phương trình hồi quy huấn luyện được để dự báo trên tập testing.

4.3.2 Các phương trình hồi quy

Hồi quy là một mô hình được sử dụng để tìm kiếm mối quan hệ giữa giá trị input và output. Mô hình hồi quy có nhiều loại như hồi quy tuyến tính (linear regression), hồi quy đa thức (polynomial regression) hoặc support vector regression,…

Hồi quy tuyến tính được sử dụng để tìm mối liên hệ giữa biến đầu ra và các biến đầu vào bằng phương trình tuyến tính. Mô hình linear regression đơn giản nhất chỉ gồm một biến đầu vào:

$$Y = C + WX \tag{4.7}$$

với Y là biến đầu ra, X là biến đầu vào, C là hằng số và W là hệ số của phương trình tuyến tính.

Multiple regression là một mở rộng của hồi quy tuyến tính đơn giản. Mô hình này sử dụng nhiều biến đầu vào để tìm mối liên hệ với giá trị đầu ra:

$$\mathrm{Y} = \mathrm{w}_0 + \mathrm{w}_1\mathrm{x}_1 + \ldots + \mathrm{w}_n\mathrm{x}_n \tag{4.8}$$

với w_1 đến w_n là các hệ số của các biến đầu vào x_1 to x_n, w_0 là một hằng số.

Phương trình hồi quy đa thức tìm mối liên hệ giữa biến đầu ra và các biến đầu vào bằng phương trình đa thức bậc n của biến đầu vào và được biểu diễn như sau:

$$\mathrm{Y} = \mathrm{w}_0 + \mathrm{w}_1 X + \mathrm{w}_2 X^2 + \cdots + \mathrm{w}_n X^n + \varepsilon \tag{4.9}$$

với w_0 đến w_n là các hệ số và ε là lỗi ngẫu nhiên.

4.4 Kết chương

Kết hợp các mô hình dự báo là một phương pháp tốt để nâng cao độ chính xác dự báo. Chương này đã trình bày ba phương pháp kết hợp các mô hình dự báo chuỗi thời

gian nói chung và kết hợp Deep Learning với các mô hình dự báo khác nói riêng. Các Ba phương pháp này gồm phương pháp kết hợp mô hình tuyến tính ARIMA với các mô hình máy học, kết hợp các mô hình dự báo dựa trên độ biến động và kết hợp các mô hình dự báo bằng phương trình hồi quy. Để kiểm chứng tính hiệu quả của các phương pháp kết hợp, các thực nghiệm dự báo giá đóng cửa của Bitcoin trong ngày kế tiếp trên chuỗi thời gian dữ liệu giá Bitcoin sẽ được thực hiện trong chương 5.

CHƯƠNG 5 ÁP DỤNG CÁC MÔ HÌNH DỰ BÁO CHUỖI THỜI GIAN CHO DỰ BÁO BITCOIN

Trong chương này, bài toán dự báo giá Bitcoin sẽ được thảo luận và nghiên cứu. Bài toán này đang được quan tâm lớn của cộng đồng các nhà nghiên cứu vì những lợi ích kinh tế mà nó mang lại. Các mô hình dự báo chuỗi thời gian sẽ được thử nghiệm để dự báo giá đóng cửa của đồng Bitcoin (USD) trong ngày tiếp theo. Kết quả dự báo của các mô hình sẽ được so sánh để xem xét mô hình nào phù hợp hơn trong việc dự báo giá Bitcoin.

5.1 Giới thiệu Bitcoin

Vào tháng 10 năm 2008, đồng tiền ảo Bitcoin được Satoshi Nakamoto lần đầu tiên giới thiệu trong báo cáo "Bitcoin: A Peer-to-Peer Electronic Cash System" [22]. Năm 2009, Nakamoto phát hành phần mềm tạo ra Bitcoin và đến nay đã có một cộng đồng rộng lớn sử dụng Bitcoin trên khắp thế giới. Ưu điểm lớn nhất của Bitcoin là được kiểm soát bởi thuật toán và giúp minh bạch hóa các giao dịch. Nakamoto tạo ra Bitcoin với mong muốn đồng tiền ảo này có thể thay thế các loại tiền tệ đang được giao dịch thông qua ngân hàng và tốn nhiều chi phí quản lý.

Trong những năm gần đây, giao dịch Bitcoin bùng nổ về lượng giao dịch cũng như số tiền thật bỏ ra để đầu tư ngày càng tăng làm đẩy mạnh sự phát triển giá trị của Bitcoin. Ước tính đến đầu 05/2019, giá trị Bitcoin trên toàn thế giới đạt đến khoảng 72.1 nghìn tỷ USD[1]. Nhiều nhà đầu tư sẵn sàng bỏ ra số tiền lớn vào Bitcoin với hy vọng nhận được những lợi nhuận lớn. Do đó, dự báo giá Bitcoin là một trong những đề tài nóng trong những năm gần đây.

Có hai phương pháp chính trong dự báo giá Bitcoin. Một là dự báo dựa vào chuỗi giá trị Bitcoin theo thời gian. Hai là xem xét giá Bitcoin với sự ảnh hưởng bởi những giá trị khác như giá chứng khoán, giá dầu mỏ, giá vàng,… Luận văn này tập trung vào phương pháp dự báo giá Bitcoin dựa trên giá trị theo chuỗi thời gian của nó. Với đặc trưng của mình, giá Bitcoin có những biến động lớn, gây khó khăn trong dự báo

[1] https://www.investopedia.com/

nhưng cũng là động lực để các tác giả tham gia nghiên cứu. Phần tiếp theo sẽ trình bày các công trình tiêu biểu trong nghiên cứu dự báo giá Bitcoin.

5.2 Các nghiên cứu trong dự báo giá Bitcoin

Như đã trình bày, dự báo giá Bitcoin có thể mang lại nguồn lợi lớn cho các nhà đầu tư. Do đó, những nghiên cứu dự báo giá Bitcoin được thực hiện ngày càng nhiều. Devavrat Shah [23] trình bày phương pháp hồi quy Bayes để dự đoán sự thay đổi giá của Bitcoin sau mỗi 10 giây. Dựa trên phương pháp này, tác giả đã đưa ra một chiến lược đơn giản để giao dịch Bitcoin. Với chiến lược đó, tác giả thực hiện 2872 các giao dịch mua bán Bitcoin, lợi nhuận đạt được trong 50 ngày khoảng 89%.

João Almeida [24] đã áp dụng mạng nơron nhân tạo để dự báo xu hướng giá Bitcoin trong ngày kế tiếp dựa vào giá và khối lượng giao dịch của Bitcoin trong những ngày trước đó. Các mô hình mạng nơron được cài đặt và thực nghiệm với thư viện Theano và công cụ MATLAB trên dữ liệu giá Bitcoin thu thập từ website Quandl. Thực nghiệm cho thấy việc thêm khối lượng giao dịch Bitcoin để làm giá trị đầu vào cho mạng nơron không phải lúc nào cũng làm tăng độ chính xác dự báo.

Isaac Madan [25] thu thập dữ liệu Bitcoin gồm 25 đặc trưng và chọn 16 đặc trưng để dự báo sự thay đổi giá hằng ngày của Bitcoin. Độ chính xác dự báo lên đến 98.7%. Song song đó, chuỗi giá Bitcoin được thu thập 10 giây một lần được huấn luyện với thuật toán random forest và mô hình tuyến tính để dự báo sự lên xuống của Bitcoin sau mỗi 10 phút. Độ chính xác trong thực nghiệm này đạt khoảng 50 – 55%.

Huisu Jang [6] đề xuất sử dụng mạng nơron Bayes để phân tích biến động của giá Bitcoin. Song song đó, tác giả cũng lựa chọn một số đặc trưng từ thông tin Block-chain có liên quan đến sự cung và cầu của Bitcoin để cải thiện kết quả dự báo. Tác giả thực nghiệm mạng nơron Bayes và một số phương pháp tuyến tính và phi tuyến tính khác trên dữ liệu giá Bitcoin. Kết quả thực nghiệm cho thấy, mạng nơron Bayes thực hiện tốt việc dự báo giá Bitcoin và mô tả được sự biến động lớn của giá Bitcoin.

Sean McNally [26] sử dụng deep learning mà cụ thể là mạng nơron hồi quy với cải tiến là LSTM để dự báo giá Bitcoin. Kết quả thực nghiệm cho thấy LSTM cho độ chính xác dự báo cao hơn hẳn các phương pháp truyền thống như ARIMA. Tác giả

cũng đã so sánh hiệu suất hoạt động của LSTM trên CPU và GPU. GPU cho hiệu suất huấn luyện tốt hơn đến 67.7% so với CPU trong thực nghiệm.

Alex Greaves [27] sử dụng biểu đồ giao dịch giá Bitcoin gồm các thông tin block-chain để làm cơ sở dự báo giá Bitcoin. Tác giả thực nghiệm dự đoán giá Bitcoin trong một giờ kế tiếp với các mô hình baseline, linear regression, svm thì mô hình linear regression cho độ lỗi MSE thấp nhất. Khi thực nghiệm dự báo giá tăng giảm của Bitcoin thì tác giả sử dụng các mô hình baseline, linear regression, svm, neural net-work. Kết quả phân lớp cho thấy neural network cho độ chính xác cao nhất.

Ferdiansyah [29] tập trung giao dịch Bitcoin bằng cách sử dụng mô hình SVM để dự đoán xu hướng thay đổi của Bitcoin trong mỗi ngày. Tác giả cho rằng cần thêm dữ liệu, ý kiến chuyên gia và một chiến lược để trading Bitcoin hiệu quả. Trong nghiên cứu tiếp theo, mạng nơron sẽ được sử dụng để cải thiện kết quả dự đoán.

Thearasak Phaladisailoed [30] tiến hành thực nghiệm các mô hình dự báo Theil-Sen Regression, Huber Regression, LSTM, GRU trên tập dữ liệu giá Bitcoin thu thập từ ngày 01/01/2012 đến 08/01/2018. Các thực nghiệm được cài đặt scikitlearn và Keras. Kết quả cho thấy LSTM và GRU cho độ lỗi dự báo thấp nhất với các độ đo MSE và R^2.

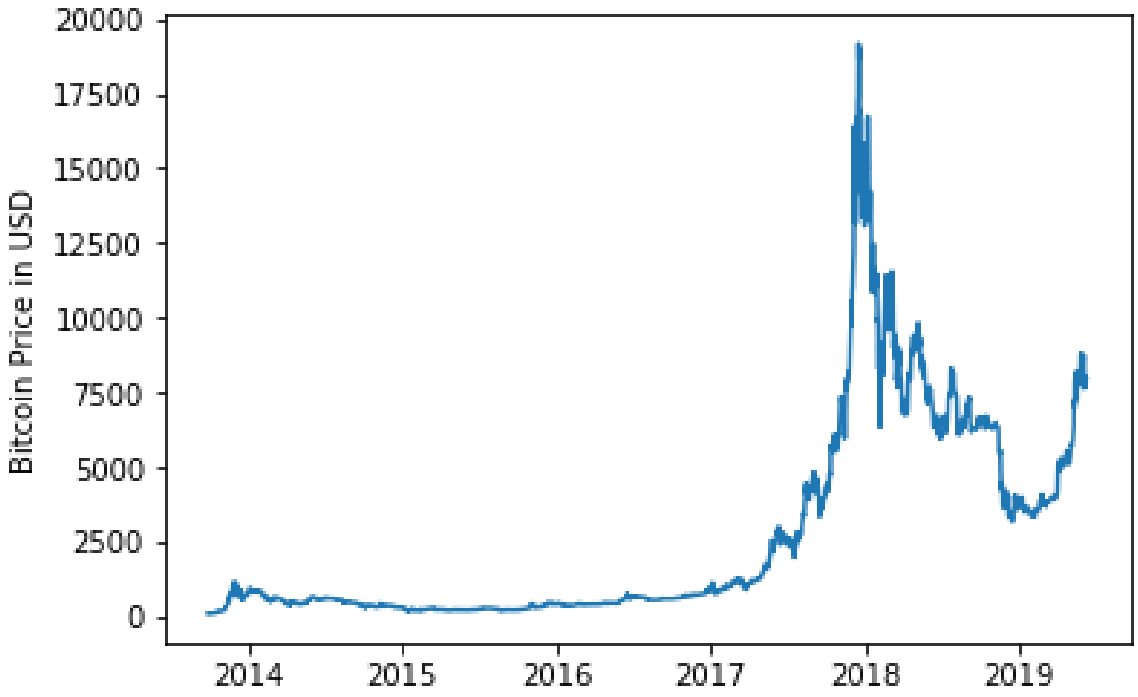
Kejsi Struga [31] cũng sử dụng LSTM cho dự báo giá Bitcoin. Thay vì chỉ sử dụng chuỗi giá trị Bitcoin, tác giả thu thập dữ liệu giá Bitcoin, các thông tin chứng khoán, tâm lý nhà đầu tư, blockchan, Coinmarketcap. Tổng số đặc trưng thu thập được là 12 với window size là 35 và số mẫu huấn luyện 1611 để làm tập huấn luyện cho mạng LSTM. Với mạng LSTM, số lỗi dự báo giảm khi tăng số lần học. Một nghiên cứu của Jang Huisu [27] cũng dự báo Bitcoin với mô hình LSTM nhưng sử dụng thêm các thông tin kinh tế vĩ mô, tỷ giá tiền tệ toàn cầu.

Dibakar Raj Pant [32] trình bày một hướng tiếp cận mới để dự báo giá Bitcoin bằng phân tích tâm lý, tình cảm của nhà đầu tư trên Twitter. Tác giả sử dụng mạng nơron hồi quy để xây dựng mô hình dự báo. Độ chính xác đạt đến 77.62%.

Hầu hết các nghiên cứu hiện nay tập trung vào dự báo xu hướng tăng giảm của giá Bitcoin bằng việc sử dụng các mô hình dự báo riêng lẻ. Phương pháp kết hợp các mô hình để nâng cao độ chính xác dự báo là việc làm hết sức cần thiết.

5.3 Tập dữ liệu giá đóng cửa của Bitcoin

Mục tiêu của thực nghiệm là dự báo giá đóng cửa của đồng Bitcoin (USD) trong ngày tiếp theo. Các mô hình dự báo sẽ được thực nghiệm trên tập dữ liệu giá đóng cửa của đồng Bitcoin, thu thập từ ngày 01/10/2013 đến ngày 08/06/2019 gồm 2070 ngày trên website CoinDesk[2]. Dựa vào biểu đồ giá đóng cửa của đồng Bitcoin, có thể thấy giá của đồng Bitcoin có những biến động lớn, đặc biệt vào năm 2017 và tiếp tục có những biến động vào năm 2018 và 2019.



Hình 5.1. Dữ liệu giá đóng cửa của Bitcoin

Tập dữ liệu thu thập có giá trị trung bình (mean) là 2642.976170. Trong đó, giá trị nhỏ nhất (min) là 108.584830, giá trị lớn nhất (max) là 19166.978740. Độ lệch chuẩn (Standard Deviation) của chuỗi giá trị là 3401.979128.

5.4 Môi trường và công cụ thực nghiệm

Các thực nghiệm được cài đặt với ngôn ngữ python 3.6 với công cụ Anaconda[3] để quản lý các thư viện. Thư viện thống kê StatsModels được sử dụng cài đặt mô hình ARIMA. Các mô hình mạng nơron được cài đặt với thư viện TensorFlow, Keras nhằm xây dựng các mô hình một cách nhanh chóng. Thư viện Pandas được dùng để

[2] https://www.coindesk.com
[3] https://www.anaconda.com

xử lý dữ liệu chuỗi thời gian, NumPy để tính toán các ma trận/vector và lưu trữ dữ liệu huấn luyện, xác thực và kiểm tra. Cuối cùng, thư viện Matplotlib dùng để vẽ các biểu đồ minh họa.

Các thử nghiệm trong luận văn được thực hiện trên các máy tính có cấu hình:

- CPU Intel Core i5-5200U 2.2GHz, Ram 12GB.
- Google Colab GPU Tesla T4, Ram 13GB.

5.5 Thực nghiệm dự báo giá Bitcoin

5.5.1 Xử lý và biến đổi dữ liệu

Các mô hình tuyến tính như ARIMA cần phải được chuẩn hóa dữ liệu để làm cơ sở dự báo. Sử dụng logarit để loại bỏ tính xu hướng của chuỗi thời gian.

$$y_t = \log x_t \tag{5.1}$$

Mô hình ARIMA chỉ hoạt động trên chuỗi thời gian có tính dừng. Thực hiện lấy sai phân bậc 1 hoặc bậc 2 để làm chuỗi có tính dừng.

$$y'_t = y_t - y_{t-1} \tag{5.2}$$

với y'_t là chuỗi thời gian sau khi lấy sai phân bậc 1, $y_t$và y_{t-1} là giá trị chuỗi thời gian tại thời điểm t và t-1.

Theo Aurélien Géron[4], để các mô hình mạng nơron học tốt, chúng ta cần thực hiện một số thao tác xử lý dữ liệu:

- Chuẩn hóa chuỗi giá trị trong khoảng [0-1].
- Tất cả các đặc trưng phải nhận các giá trị ở cùng một phạm vi.

Có nhiều phương pháp chuẩn hóa, trong đó chuẩn hóa MinMax được sử dụng phổ biến nhất:

$$y' = \frac{y - y_{min}}{y_{max} - y_{min}} \tag{5.3}$$

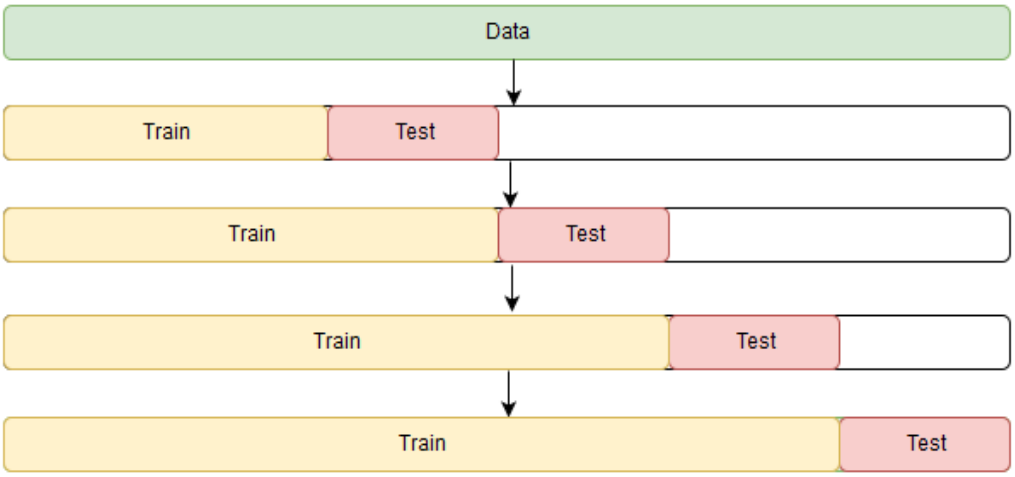
5.5.2 Chia tập dữ liệu thực nghiệm

Trong các thực nghiệm với từng mô hình riêng lẻ, tập dữ liệu được chia thành hai phần: 90% các điểm thời gian được sử dụng để huấn luyện, 10% các điểm thời gian còn lại sẽ được sử dụng cho thử nghiệm mô hình. Khi thực nghiệm với các phương pháp kết hợp mô hình tuyến tính ARIMA và các mô hình máy học, kết hợp bằng

[4] https://www.oreilly.com/library/view/hands-on-machine-learning/9781491962282/

phương trình hồi quy, tập dữ liệu được chia thành ba phần: 80% các điểm thời gian được sử dụng để huấn luyện, 10% dùng để đánh giá mô hình, 10% còn lại dùng để thử nghiệm mô hình.

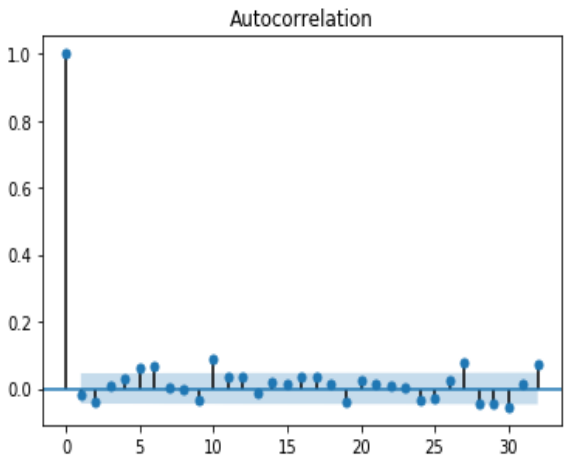
Đối với phương pháp kết hợp các mô hình dựa trên độ biến động, tập dữ liệu được chia theo phương pháp Nested Cross Validation. Tập dữ liệu được chia thành 10 phần bằng nhau. Trong lần huấn luyện đầu tiên, 50% được sử dụng huấn luyện, 10% kế tiếp cho tập đánh giá mô hình, 10% còn lại cho thử nghiệm mô hình. Cứ như thế, đến lần huấn luyện cuối cùng thì dùng 80% cho huấn luyện, 10% cho đánh giá mô hình, 10% còn lại để thử nghiệm mô hình.



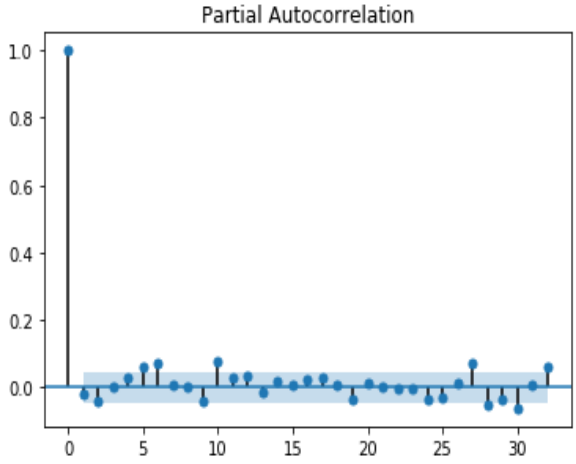
Hình 5.2. Nested Cross Validation

5.5.3 Thực nghiệm mô hình ARIMA

Dựa vào kiểm định Augmented Dickey Fuller (ADF), chuỗi dữ liệu giá đóng cửa Bitcoin không phải là một chuỗi thời gian dừng. Vì vậy, chuỗi dữ liệu cần được biến đổi thành chuỗi thời gian dừng bằng cách lấy sai phân bậc 1 (d=1), ta được chuỗi y'_t. Dùng kiểm định ADF trên chuỗi y'_t thì chuỗi dữ liệu giá đóng cửa của Bitcoin đã dừng. Các biểu đồ ACF, PACF được vẽ ra với độ trễ (lag) tối đa là 32 để xác định các giá trị q, p có thể sử dụng cho mô hình ARIMA. Các giá trị có ý nghĩa thống kê vượt khỏi giới hạn $\pm 1.96/\sqrt{2070}$ sẽ được chọn. Từ biểu đồ ACF, PACF, tham số q có thể nhận các giá trị 0, 5, 6, 10, 27, 30, 32 và tham số p có thể nhận các giá trị 0, 5, 6, 10, 27, 30, 32.



Hình 5.3. Biểu đồ ACF



Hình 5.4. Biểu đồ PACF

Nhiệm vụ kế tiếp là cần tìm ra một mô hình ARIMA phù hợp để dự báo giá Bitcoin. Các tiêu chuẩn AIC, BIC được dùng để lựa chọn mô hình phù hợp với tập dữ liệu. Mô hình ARIMA nào có các giá trị này nhỏ nhất sẽ được lựa chọn.

Bảng 5.1. Các tiêu chuẩn kiểm định của một số mô hình ARIMA

Mô hình	AIC	BIC
ARIMA (6,1,0)	-6301.521	**-6257.285**
ARIMA (0,1,5)	-6294.760	-6256.055
ARIMA (6,1,5)	**-6325.673**	-6253.791

Khi kiểm định các mô hình, mô hình ARIMA (6,1,5) được chọn làm mô hình dự báo giá đóng cửa của Bitcoin. Sử dụng mô hình ARIMA để dự báo giá Bitcoin trong ngày tiếp theo trên tập testing. Độ chính xác dự báo được thể hiện qua độ đo RMSE và MAPE. Ví dụ, kết quả dự báo trong một tuần từ 12/11/2018 đến 18/11/2018.

Bảng 5.2. Minh họa kết quả dự báo của mô hình

Ngày dự báo	Giá trị dự báo	Giá trị thực tế	RMSE	MAPE (%)
12/11/2018	6336.317748	6327.154696	9.1630	0.1448
13/11/2018	6314.072772	6282.438677	31.6341	0.5035
14/11/2018	6272.783525	5524.801093	747.9824	13.5386
15/11/2018	5560.557849	5511.152834	49.4050	0.8964
16/11/2018	5556.055314	5448.600634	107.4547	1.9721
17/11/2018	5440.482355	5497.411279	56.9289	1.0355

18/11/2018	5478.406777	5553.014895	74.6081	1.3435
		Trung bình	153.8823	2.7763

Các kết quả dự báo trong trong tập dữ liệu testing (12/11/2018 đến 08/06/2019) sẽ được tính trung bình và kết quả độ đo lỗi RMSE= 214.6536 và MAPE= 2.5626%.

5.5.4 Thực nghiệm mô hình FFNN

Để xây dựng mô hình FFNN cho dự báo giá Bitcoin, việc đầu tiên cần xác định các biến đầu vào và các biến đầu ra cho mô hình. Biến đầu vào là các giá trị dữ liệu giá đóng cửa của Bitcoin ở những ngày trước thời điểm dự báo. Số lượng biến đầu vào được xác định dựa trên thực nghiệm để tìm ra giá trị phù hợp. Biến đầu ra là một giá trị dữ liệu giá Bitcoin được dự báo từ mô hình. Việc kế tiếp là xác định số lớp ẩn và số nơron trong từng lớp ẩn. Thông qua quá trình thực nghiệm để xác định được các thông số này. Với tập dữ liệu giá Bitcoin, mô hình FFNN gồm có 3 lớp: 1 lớp đầu vào, 1 lớp ẩn, 1 lớp đầu ra được xây dựng. Số biến của lớp đầu vào (n) được thử nghiệm từ 1 đến 30 (các điểm thời gian trong một tháng trước dự báo). Số nơron lớp ẩn được thử nghiệm từ 1 đến 2n cho mỗi mô hình FFNN. Số lần huấn luyện (epochs) được lựa chọn lần lượt là 100, 200, 500. Các mô hình FFNN gồm 2 lớp ẩn cũng đã được thực nghiệm nhưng kết quả không khả quan bởi nhiều lớp ẩn sẽ dẫn đến tính toán phức tạp và xảy ra trường hợp quá khớp dữ liệu. Sau những kết quả thử nghiệm, mô hình FFNN gồm 5 nơron trong lớp input (dựa vào giá Bitcoin của 5 ngày trước để dự báo giá của ngày tiếp theo), 10 nơron trong lớp ẩn và 1 nơron trong lớp đầu ra được lựa chọn cho dự báo giá Bitcoin.

Bảng 5.3. Độ lỗi dự báo của một số mô hình FFNN

Số nơron đầu vào	Số nơron lớp ẩn	Số epochs	RMSE	MAPE (%)
5	10	100	297.8668	3.7859
5	10	200	**241.0773**	**2.9170**
6	12	100	251.0424	3.0919
7	7	200	240.3842	3.3818
7	14	100	248.7884	3.2630

5.5.5 Thực nghiệm mô hình CNN

Cũng giống như mô hình FFNN, việc xác định số lượng biến đầu vào cũng phải được xác định qua thực nghiệm. Ngoài ra, mô hình CNN còn cần xác định số lớp tích chập (convolutional), số lớp lấy mẫu (pooling), số lớp kết nối đầy đủ (fully-connected). Trong dự báo chuỗi thời gian, những lớp này thường được chọn với số lượng ít để hạn chế trường hợp quá khớp dữ liệu. Trong thực nghiệm này, mô hình CNN được xây dựng cho dự báo giá Bitcoin với 1 lớp tích chập gồm 64 bộ lọc, hàm ReLU được sử dụng để phi tuyến tính giá trị đầu ra từ lớp tích chập, 1 lớp lấy mẫu với ma trận lấy mẫu là 2x1, thêm vào đó là 1 lớp kết nối đầy đủ. Sau đó, lớp đầu ra sẽ có 1 nơron để tính giá trị dự báo cho mô hình.

Bảng 5.4. Độ lỗi dự báo của một số mô hình CNN

Số nơron đầu vào	Số nơron lớp fully connected	Số epochs	RMSE	MAPE (%)
3	6	100	408.0421	4.7511
3	6	200	298.6308	4.0791
4	8	100	384.6678	4.9115
6	6	200	359.9081	4.4630
7	14	200	**307.9527**	**4.0437**

5.5.6 Thực nghiệm mô hình SVR

Xác định số lượng biến đầu vào của mô hình SVR sẽ thông qua thực nghiệm. Các biến đầu vào sẽ được thử nghiệm từ 1 đến 7, tức là dựa vào các giá trị trong khoảng một tuần để dự báo giá Bitcoin. Các Kernel được sử dụng trong mô hình SVR là Linear, Polynomial và Radial Basic Function (RBF).

Bảng 5.5. Độ lỗi dự báo của một số mô hình SVR

Số biến đầu vào	Kernel	RMSE	MAPE (%)
1	Linear	351.6466	7.2127
2	RBF	233.1121	3.2476
4	RBF	**232.5739**	**2.8558**
5	RBF	237.6272	3.1626

6	RBF	236.1728	2.9153

5.5.7 Thực nghiệm mô hình LSTM

Cũng giống như mô hình FFNN, việc xác định số lượng biến đầu vào cũng phải được xác định qua thực nghiệm. Trong thực nghiệm này, mô hình LSTM gồm 1 lớp input, 1 lớp LSTM, 1 lớp output sẽ được xây dựng. Vấn đề còn lại là lựa chọn số đơn vị ẩn (hidden units) trong lớp LSTM cũng như số lần huấn luyện mạng. Số đơn vị ẩn trong lớp LSTM được thử nghiệm từ 1 đến 200. Số lần huấn luyện (epochs) được lựa chọn lần lượt là 100, 200, 500.

Bảng 5.6. Độ lỗi dự báo của một số mô hình LSTM

Số nơron đầu vào	Số nơron lớp ẩn trong tế bào LSTM	Số epochs	RMSE	MAPE (%)
2	200	10	226.7758	2.6654
4	6	200	228.2041	2.6367
5	4	200	218.1138	2.6000
5	100	100	**219.3169**	**2.5852**
5	90	200	225.9891	2.6274

5.5.8 Thực nghiệm kết hợp các mô hình dự báo

Thực nghiệm kết hợp ARIMA và các mô hình máy học:

Trong thực nghiệm với giá Bitcoin, mô hình ARIMA (6,1,5) tiếp tục được sử dụng cho dự báo. Các mô hình FFNN, CNN, LSTM, SVR được chạy thử nghiệm nhiều lần với các tham số gồm số biến đầu vào, số lớp ẩn, số nơron trong lớp ẩn. Sau đó, kết hợp ARIMA và các mô hình này để dự báo giá Bitcoin trên tập testing. Các kết hợp này có hiệu quả hay không phụ thuộc nhiều vào các mô hình máy học vừa xây dựng. Do đó, các mô hình này có thể được thử nghiệm nhiều lần để chọn ra mô hình kết hợp tốt nhất có thể.

Bảng 5.7. Độ lỗi dự báo của một số mô hình kết hợp

Mô hình kết hợp	RMSE	MAPE (%)
ARIMA_FFNN	213.3135	2.5849
ARIMA_CNN	213.0082	2.5563

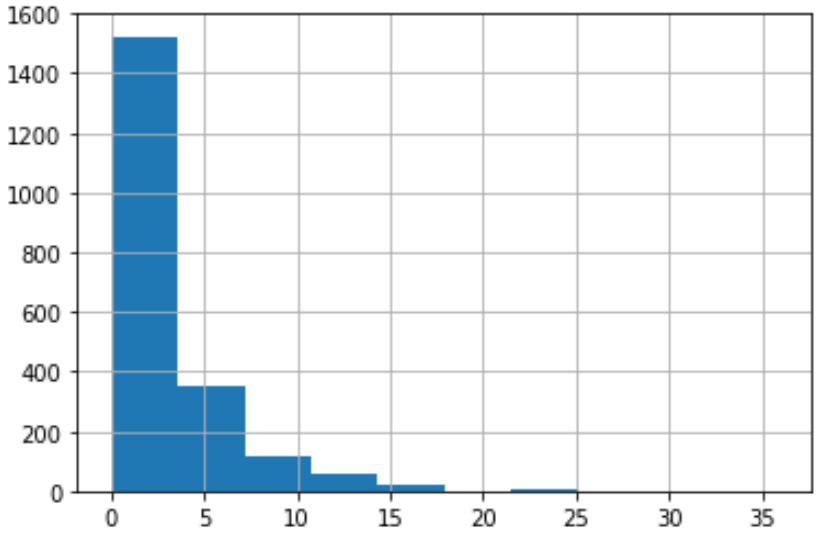
ARIMA_LSTM	**214.1901**	**2.5347**
ARIMA_SVR	213.8141	2.6144

Thực nghiệm kết hợp dựa trên độ biến động:

Trong tập dữ liệu giá đóng cửa của Bitcoin thu thập được, độ biến động nhỏ nhất (min fluctuation) là 0.000679% và cao nhất (max fluctuation) là 35.849315%. Dựa vào histogram tần suất của độ biến động, có thể thấy các độ biến động lớn hơn 25% là rất ít và có thể xem như trường hợp ngoại lệ. Do đó, khoảng biến động giữa max fluctuation và min fluctuation là R = [0-25]. Áp dụng công thức của Herbert A. Sturges [34] để xác định độ lớn của mỗi khoảng biến động có thể chia:

$$c = \frac{R}{1+3.322 \log N} = \frac{25}{1+3.322 \log 2069} = 2.08$$

Với độ lớn của mỗi khoảng biến động c = 2, ta sẽ có 12 khoảng biến động là [0-2), [2-4), [4-6), [6-8), [8-10), [10-12), [12-14), [14-16), [16-18), [18-20), [20-22), [22-24). Những độ biến động vượt quá 24 thì xem như thuộc về khoảng [22-24).



Hình 5.5. Histogram của các độ biến động trong chuỗi dữ liệu giá Bitcoin

Sử dụng Nested Cross Validation, các mô hình ARIMA, FFNN, CNN, LSTM, SVR sẽ được huấn luyện, dự báo trên tập validation và cho độ lỗi dự báo trên các khoảng biến động như sau:

Bảng 5.8. Độ lỗi dự báo của ARIMA trên từng khoảng biến động

Khoảng biến động	RMSE	MAPE (%)
[0-2]	227.2506	2.0639

[2-4)	220.4650	2.0200
[4-6)	148.9002	1.5346
[6-8)	165.3384	1.9018
[8-10)	206.2811	2.2521
[10-12)	411.8864	4.2889
[12-14)	90.6511	1.0272
[14-16)	176.1310	2.3155
[16-18)	377.5560	4.7234
[18-20)	138.1584	2.0483
[20-22)	-	-
[22-24)	90.9297	1.2699

Bảng 5.9. Độ lỗi dự báo của FFNN trên từng khoảng biến động

Khoảng biến động	RMSE	MAPE (%)
[0-2]	246.5400	2.2229
[2-4)	231.8161	2.1884
[4-6)	207.4423	1.9690
[6-8)	166.0750	1.8456
[8-10)	198.2241	2.0617
[10-12)	404.8154	4.2288
[12-14)	104.5949	1.3310
[14-16)	197.7703	2.3638
[16-18)	327.6338	4.3212
[18-20)	189.0287	2.8025
[20-22)	-	-
[22-24)	216.3643	3.1867

Bảng 5.10. Độ lỗi dự báo của CNN trên từng khoảng biến động

Khoảng biến động	RMSE	MAPE (%)
[0-2]	259.4781	2.5842
[2-4)	260.0663	2.6147
[4-6)	232.5421	2.3839

[6-8)	175.6760	1.9207
[8-10)	207.5546	2.3760
[10-12)	319.5693	3.5179
[12-14)	75.7266	1.1399
[14-16)	95.8542	1.2341
[16-18)	293.5373	3.4175
[18-20)	256.4037	3.8013
[20-22)	-	-
[22-24)	173.3843	2.5513

Bảng 5.11. Độ lỗi dự báo của SVR trên từng khoảng biến động

Khoảng biến động	RMSE	MAPE (%)
[0-2]	249.2861	2.2464
[2-4)	240.3477	2.1742
[4-6)	203.8670	1.8818
[6-8)	159.3224	1.7122
[8-10)	205.9743	2.1260
[10-12)	416.6546	4.3324
[12-14)	102.6455	1.2998
[14-16)	172.0326	2.1273
[16-18)	338.0846	4.4260
[18-20)	213.4549	3.1646
[20-22)	-	-
[22-24)	230.6129	3.4614

Bảng 5.12. Độ lỗi dự báo của LSTM trên từng khoảng biến động

Khoảng biến động	RMSE	MAPE (%)
[0-2]	243.9843	2.2542
[2-4)	250.1236	2.1934
[4-6)	166.7058	1.5746
[6-8)	160.8223	1.8843
[8-10)	206.1551	1.9222

[10-12)	458.4827	5.0125
[12-14)	109.4072	1.3506
[14-16)	171.9203	2.1836
[16-18)	376.5355	4.7956
[18-20)	149.2718	2.2130
[20-22)	-	-
[22-24)	221.4328	3.3622

Từ độ lỗi của các mô hình trên từng khoảng biến động, ta chọn mô hình tốt nhất để dự báo trên từng khoảng biến động như sau: khoảng [0, 2): ARIMA, khoảng [2,4): ARIMA, khoảng [4,6): ARIMA, khoảng [6,8): SVR, khoảng [8,10): LSTM, khoảng [10, 12): CNN, khoảng [12, 14): ARIMA, khoảng [14,16): CNN, khoảng [16,18): CNN, khoảng [18,20): ARIMA, khoảng [20,22): LSTM, khoảng [22, 24): ARIMA. Sử dụng các mô hình này để dự báo trên tập testing, độ lỗi của phương pháp kết hợp là RMSE= 214.2755, MAPE= 2.5483%.

Thực nghiệm kết hợp bằng phương trình hồi quy:

Thực nghiệm kết hợp bằng phương trình hồi quy với số lượng biến đầu vào khác nhau, có thể là 1, 2, 3, 4 hoặc 5. Kết quả cho thấy mô hình kết hợp bằng phương trình hồi quy không cho dự báo tốt như kỳ vọng. Nguyên nhân có thể là do tập huấn luyện cho phương trình hồi quy ít. Vì thế, phương trình hồi quy chưa nắm bắt được độ lỗi dự báo của các mô hình.

Bảng 5.13. Độ lỗi dự báo của mô hình kết hợp bằng phương trình hồi quy

Biến đầu vào	RMSE	MAPE (%)
ARIMA	224.2551	3.3212
ARIMA, FFNN, CNN	242.3809	3.7839
ARIMA, LSTM	**223.7082**	**3.2890**
ARIMA, SVR	225.4367	3.3954
ARIMA, FFNN, CNN, LSTM, SVR	257.9973	4.2577

5.5.9 So sánh kết quả thực nghiệm

Kết quả thực nghiệm cho thấy mô hình ARIMA, LSTM cho kết quả dự báo tốt với độ đo lỗi RMSE và MAPE. Trong khi đó, hầu hết những phương pháp kết hợp các

mô hình dự báo đều cho kết quả dự báo tốt hơn từng mô hình riêng lẻ. Điều này cho thấy sự triển vọng của các mô hình kết hợp trong dự báo giá Bitcoin nói riêng và dự báo chuỗi thời gian nói chung.

Bảng 5.14. So sánh độ lỗi dự báo của các mô hình dự báo giá Bitcoin

Mô hình dự báo	RMSE	MAPE (%)
ARIMA	214.6536	2.5626
FFNN	241.0773	2.9170
CNN	307.9527	4.0437
LSTM	219.3169	2.5852
SVR	232.5739	2.8558
ARIMA_LSTM	**214.1901**	**2.5347**
Kết hợp dựa trên độ biến động	**214.2755**	**2.5483**
Kết hợp bằng phương trình hồi quy	223.7082	3.2890

5.5.10 Đánh giá hiệu suất về thời gian của các mô hình

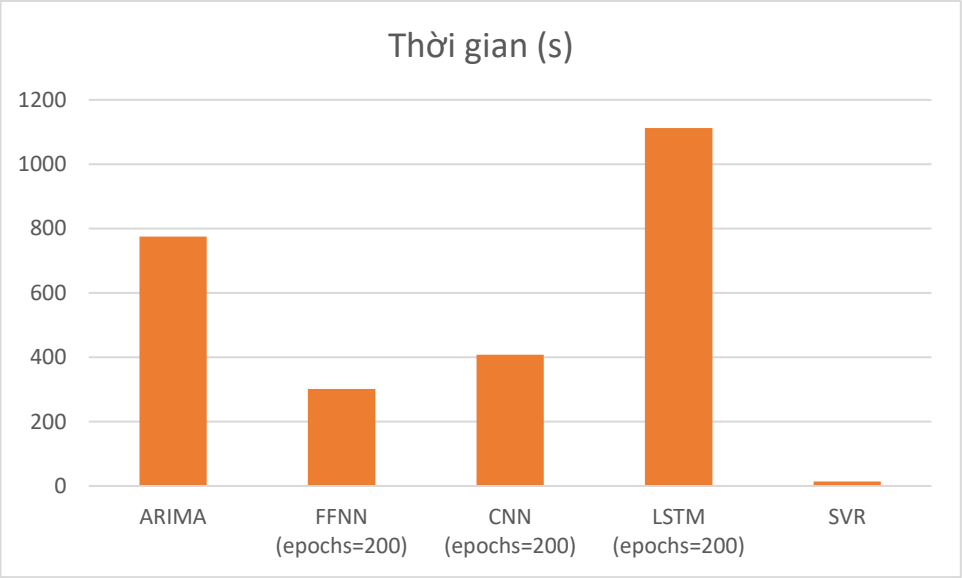
Phần này sẽ trình bày đánh giá hiệu suất về thời gian của các mô hình. Mô hình ARIMA có thời gian huấn luyện khoảng 775.0358 giây. Mô hình SVR có kết quả huấn luyện nhanh nhất khoảng 13.8750 giây. Với các mô hình mạng nơron, kết quả thời gian huấn luyện cho thấy mô hình LSTM tốn nhiều thời gian cho một lần huấn luyện và sẽ tăng rất nhiều nếu số lần học tăng lên.

Bảng 5.15. Thời gian huấn luyện của các mô hình mạng nơron

Mô hình dự báo	Epochs	Thời gian huấn luyện (giây)
FFNN	10	21.7396
	100	180.2780
	200	301.3564
CNN	10	28.4716
	100	225.0297
	200	407.7530
LSTM	10	28.2153
	100	553.5914

	200	1112.1237

Thời gian thực nghiệm của các mô hình sẽ được minh học qua biểu đồ bên dưới để thấy rõ hơn hiệu suất về thời gian của các mô hình.



Hình 5.6. Biểu đồ thời gian huấn luyện của các mô hình

5.6 Kết chương

Chương này đã trình bày các bước chi tiết trong việc thực nghiệm các mô hình ARIMA, FFNN, CNN, SVR, LSTM để dự báo giá Bitcoin. Một số phương pháp kết hợp Deep Learning và các mô hình dự báo khác cũng đã được áp dụng. Kết quả cho thấy, các phương pháp này thật sự triển vọng trong việc nâng cao độ chính xác dự báo giá Bitcoin và có thể được áp dụng để giao dịch Bitcoin trong thực tế.

CHƯƠNG 6 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Các kết quả đạt được

Dự báo nói chung và dự báo chuỗi thời gian nói riêng có nhiều giá trị thực tiễn trong cuộc sống. Đây là lĩnh vực rất được quan tâm, nghiên cứu hiện nay. Luận văn tập trung vào bài toán trên và tiến hành nghiên cứu những nội dung sau:

- Hệ thống hóa các mô hình dự báo chuỗi thời gian.
- Áp dụng mô hình Deep Learning cho dự báo chuỗi thời gian.
- Nghiên cứu, thử nghiệm một số phương pháp nâng cao độ chính xác dự báo dựa trên kết hợp các mô hình.
- Áp dụng các mô hình dự báo để dự báo giá đóng cửa của Bitcoin, một trong những lĩnh vực dự báo được quan tâm hiện nay.

Luận văn đã nghiên cứu áp dụng các mô hình ARIMA, FFNN, CNN, LSTM, SVR cho bài toán dự báo chuỗi thời gian. Trong quá trình nghiên cứu, ý tưởng kết hợp các mô hình riêng lẻ được phát triển như là một hướng tiềm năng nhằm nâng cao độ chính xác dự báo. Có ba phương pháp kết hợp được trình bày trong luận văn gồm:

- Kết hợp giữa ARIMA và các mô hình khác như mạng nơron, mô hình hồi quy và Deep Learning.
- Kết hợp dựa trên độ biến động.
- Kết hợp bằng phương trình hồi quy.

Những mô hình dự báo chuỗi thời gian đã nghiên cứu được áp dụng để dự báo giá đóng cửa của Bitcoin trong ngày tiếp theo trên tập dữ liệu gồm 2070 điểm dữ liệu giá Bitcoin được thu thập từ ngày 01/10/2013 đến ngày 08/06/2019. Các kết quả thực nghiệm cho thấy mô hình ARIMA và LSTM cho kết quả dự báo tốt hơn hẳn các mô hình khác thông qua độ đo RMSE và MAPE. Song song đó, các phương pháp kết hợp cũng cho thấy triển vọng cải thiện độ chính xác dự báo giá Bitcoin với phương pháp kết hợp giữa ARIMA và các mô hình khác, phương pháp kết hợp dựa trên độ biến động. Nhưng đối với phương pháp kết hợp bằng phương trình hồi quy lại không cho kết quả khả quan bởi dữ liệu huấn luyện cho hồi quy ít dẫn đến chưa nắm bắt được độ lỗi dự báo của các mô hình.

Các phương pháp dự báo chuỗi thời gian chỉ là một hướng nghiên cứu của vấn đề dự báo. Việc dự báo còn gặp nhiều khó khăn và sẽ cần thời gian tiếp tục nghiên cứu để cải thiện hơn nữa kết quả dự báo. Một số vấn đề có thể tiếp tục nghiên cứu phát triển sẽ được trình bày trong phần sau.

6.2 Một số hướng phát triển

Vấn đề khó khăn nhất trong dự báo chuỗi thời gian là sự biến đổi ngẫu nhiên của chuỗi thời gian. Do đó, một số phương pháp có thể giải quyết vấn đề này:

- Phát triển các phương pháp phát hiện xu hướng biến động chuỗi thời gian.
- Gom cụm chuỗi thời gian để nhận biết khoảng biến động và không biến động, từ đó nắm bắt quy luật để dự báo ngắn hạn.

Trong một số dự báo như dự báo giá Bitcoin thì giá trị của chuỗi thời gian phụ thuộc nhiều vào yếu tố tâm lý và sự quan tâm của nhà đầu tư nên cần nghiên cứu các phương pháp phân tích xu hướng tìm kiếm như Google trending hoặc phân tích tâm lý nhà đầu tư trên mạng xã hội và kết hợp với dự báo chuỗi thời gian để nâng cao độ chính xác dự báo.

Một vấn đề đặt ra nữa với các bài toán dự báo là khả năng đáp ứng trong việc dự báo thời gian thực. Các mô hình dự báo thời gian thực cần được nghiên cứu cũng như đưa ra những phương pháp đánh giá để kiểm chứng về độ chính xác dựa trên thực tế để đảm bảo độ tin cậy và có sự thuyết phục hơn về ý nghĩa thực tế.

Các phương pháp nghiên cứu dự báo nói chung và chuỗi thời gian nói riêng là tiền đề để xây dựng nhiều ứng dụng mang lại giá trị kinh tế cao trong tương lai. Một trong những ứng dụng triển vọng là xây dựng các con bot tự động giao dịch tiền điện tử (trading cryptocurrency) trên các sàn giao dịch như eToro, Binance, Huobi Pro,… Các con bot sẽ tự động quản lý ví tiền điện tử trên các sàn giao dịch, học hỏi các giao dịch lãi và lỗ để từ đó tự đưa ra quyết định mua hoặc bán tiền điện tử sao cho phù hợp với xu hướng giá hiện tại của các đồng tiền điện tử. Số lượng giao dịch càng nhiều thì bot càng thông minh và có khả năng mang lại lợi nhuận lớn cho nhà đầu tư.

DANH MỤC CÔNG BỐ KHOA HỌC CỦA TÁC GIẢ

[1] Lê Hữu Vinh, Nguyễn Đình Thuân (2019), "Dự báo giá Bitcoin bằng kết hợp mô hình ARIMA và mạng nơron", Hội nghị Quốc gia lần thứ XII về Nghiên cứu cơ bản và ứng dụng Công Nghệ thông tin (FAIR), Huế, Việt Nam.

TÀI LIỆU THAM KHẢO

Tiếng Việt

1. Hoàng Thị Tuyết (2012), *Kỹ thuật dự báo dựa theo hồi quy vectơ hỗ trợ và thử nghiệm áp dụng dự báo thành tích vận động viên*, BCKH Thạc sĩ Hệ thống thông tin, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội.
2. Dương Thu Trang (2017), *Ứng dụng mạng nơron nhân tạo dự báo số học sinh tuyển vào trung tâm GDNN-GDTX quận Đống Đa*, BCKH Thạc sĩ Công nghệ thông tin, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội.

Tiếng Anh

3. Dian Utami Sutiksno, Ansari Saleh Ahmar, Nuning Kurniasih, Eko Susanto, Audrey Leiwakabessy (2018), "Forecasting Historical Data of Bitcoin using ARIMA and α-Sutte Indicator", *J. Phys. Conf. Ser.*, vol. 1028, no. 1, p. 012194.
4. Amin Azari (2018), "Bitcoin Price Prediction: An ARIMA Approach", KTH Royal Institute of Technology.
5. Bogdan Oancea, ŞTefan Cristian Ciucu (2014), "Time series forecasting using neural networks", In Proceedings of the "Challenges of the Knowledge Society" Conference, eprint arXiv:1401.1333, pp. 1402–1408.
6. Huisu Jang, Jaewook Lee (2018), "An empirical study on modeling and prediction of Bitcoin prices with bayesian neural networks based on blockchain information", IEEE Access, vol. 6, pp. 5427–5437.
7. Kumar Abhishek, Anshul Khairwa, Tej Pratap, Surya Prkash (2012), "A stock market prediction model using Artificial Neural Network", Computing Communication & Networking Technologies (ICCCNT).
8. M. Raeesi, M. S. Mesgari, P. Mahmoudi (2014), "Traffic Time Series Forecasting by Feedforward Neural Network: a Case Study Based on Traffic Data of Monroe", The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 40(2), 219.
9. Sheng Chen, Hongxiang He (2018), "Stock Prediction Using Convolutional Neural Network", IOP Conference Series Materials Science and Engineering.

10. Ping-Feng Pai, Chih-Sheng Lin (2004), “A hybrid ARIMA and support vector machines model in stock price forecasting”, *Omega*, 33, 505–597.
11. Hongzhan NIE, Guohui LIU, Xiaoman LIU, Yong WANG (2012), “Hybrid of ARIMA and SVMs for Short-Term Load Forecasting”, International Conference on Future Energy, Environment, and Materials.
12. G. Peter Zhang (2003), “Times series forecasting using a hybrid ARIMA and neural network model”, Neurocomputing, vol. 50, pp. 159–75.
13. Akhter Mohiuddin Rather (2014), “A Hybrid Intelligent Method of Predicting Stock Returns”, Advances in Artificial Neural Systems.
14. Durdu O¨ mer Faruk (2010), “A hybrid neural network and ARIMA model for water quality time series prediction”, Engineering Applications of Artificial Intelligence, 23(4), 586-594.
15. K. Naveena, Subedar Singh, Santosha Rathod, Abhishek Singh (2017), “Hybrid ARIMA-ANN Modelling for Forecasting the Price of Robusta Coffee in India”, International Journal of Current Microbiology and Applied Sciences.
16. Nitin Merh, Vinod P. Saxena, Kamal Raj Pardasani (2010), “A comparison between hybrid approaches of ANN and ARIMA for Indian stock trend forecasting”, *Journal of Business Intelligence*, vol. 3, no. 2, pp. 23–43.
17. Xiping Wang, Ming Meng (2012), “A Hybrid Neural Network and ARIMA Model for Energy Consumption Forecasting”, *Journal of Computers*, vol.7, no.5.
18. S. Hochreiter, J. Schmidhuber (1997), “Long Short-Term Memory”, Neural Computation 9(8):1735-1780.
19. John Gamboa (2017), “Deep learning for time-series analysis”, arXiv preprint arXiv:1701.01887.
20. Enzo Busseti, Ian Osband, Scott Wong (2012), “Deep learning for time series modeling”, Stanford University, CA, Tech. Rep. CS 229.
21. Xueheng Qiu, Le Zhang, Ye Ren, P. N. Suganthan, Gehan Amaratunga (2014), “Ensemble deep learning for regression and timeseries forecasting”, Proc. IEEE Symposium on Computational Intelligence and Ensemble Learning (CIEL’14), Orlando, US.
22. Satoshi Nakamoto (2008), “Bitcoin: A Peer-to-Peer Electronic Cash System”.

23. Devavrat Shah, Kang Zhang (2014), “Bayesian regression and Bitcoin”, arXiv preprint arXiv:1410.1231.
24. João Almeida, Shravan Tata, Andreas Moser, Vikko Smit (2015), “Bitcoin prediction using ANN”.
25. Isaac Madan, Shaurya Saluja, Aojia Zhao (2015), “Automated Bitcoin trading via machine learning algorithms”.
26. Sean McNally (2016), “Predicting the price of Bitcoin using machine learning”, Ph.D. dissertion, School Comput., Nat. College Ireland, Dublin, Ireland.
27. Alex Greaves, Benjamin Au (2015), “Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin”.
28. Jang Huisu, Jaewook Lee, Hyungjin Ko, Woojin Lee (2018), “Predicting Bitcoin Prices by Using Rolling Window LSTM model”, ACM, London, UK.
29. Ferdiansyah, Edi Surya Negara, Yeni Widyanti (2019), “BITCOIN-USD Trading Using SVM to Detect The Current day’s Trend in The Market”, Journal of Information Systems and Informatics.
30. Thearasak Phaladisailoed, Thanisa Numnonda (2018), “Machine Learning Models Comparison for Bitcoin Price Prediction”, In 10th International Conference on Information Technology and Electrical Engineering (ICITEE).
31. Kejsi Struga, Olti Qirici (2018), “Bitcoin Price Prediction with Neural Networks”, University of Tirana.
32. Dibakar Raj Pant, Prasanga Neupane, Anuj Poudel, Anup Kumar Pokhrel, Bishnu Kumar Lama (2018), “Recurrent Neural Network Based Bitcoin Price Prediction by Twitter Sentiment Analysis”, IEEE 3rd International Conference on Computing, Communication and Security (ICCCS).
33. Bruno Spilak (2018), *Deep neural networks for cryptocurrencies price prediction*, Humboldt University.
34. Herbert A. Sturges (1926), “The choice of a class interval”.
35. Douglas C. Montgomery, Cheryl L. Jennings, Murat Kulahci (2015), *Introduction to time series analysis and forecasting*, Wiley Series in Probability and Statistics, John Wiley & Sons, Inc., Hoboken, New Jersey, USA.

36. R. Adhikari, R. Agrawal (2013), “An introductory study on time series modeling and forecasting”, arXiv preprint arXiv:1302.6613.
37. OTexts (July. 2019), https://otexts.com/fpp2/
38. Takashi Kuremoto, Shinsuke Kimura, Kunikazu Kobayashi, Masanao Obayashi (2014), “Time series forecasting using a deep belief network with restricted Boltzmann machines”, Neurocomputing, vol. 137, 5 August 2014, pp. 47-56.
39. Jae Young Choi, Bumshik Lee (2018), “Combining LSTM network ensemble via adaptive weighting for improved time series forecasting”, Math Probl Eng 2018, pp. 1-8.