

DỰ BÁO CHẤT LƯỢNG KHÔNG KHÍ Dựa VÀO CÁC MÔ HÌNH THỐNG KÊ, HỌC MÁY VÀ HỌC SÂU

ĐOÀN NGỌC TUẤN¹, DOÃN CÔNG TRÍ², TRẦN LÊ TÚ³, TRẦN QUỐC HƯNG⁴, NGUYỄN PHƯỚC HUY⁵

Tóm tắt nội dung—Trong bối cảnh biến đổi khí hậu ngày càng trở nên đáng lo ngại, việc dự đoán và đánh giá khí hậu trở thành một yếu tố quan trọng để giúp quản lý nguồn lực và phát triển bền vững. Đặc biệt, ở Việt Nam, một quốc gia nằm trong khu vực có đặc điểm khí hậu phức tạp, việc dự đoán khí hậu có thể đóng vai trò quan trọng trong việc xây dựng các chiến lược quản lý môi trường và phát triển kinh tế. Trong nghiên cứu này, chúng tôi tiếp cận vấn đề này bằng cách sử dụng dữ liệu chuỗi thời gian về khí hậu của ba thành phố lớn tại Việt Nam: Hà Nội, Đà Nẵng và Việt Trì. Chúng tôi áp dụng một loạt các mô hình dự đoán như VAR (Vector Autoregression), Linear Regression áp dụng CalendarFourier và DeterministicProcess, SES (Simple Exponential Smoothing), DLinear (Dynamic Linear Model), và NBEATS (Neural Basis Expansion Analysis Time Series) để so sánh và đánh giá hiệu suất dự đoán của mỗi mô hình trên dữ liệu thực tế.

Index Terms—Keywords - Time series , statistical method, forecasting AQI index, Linear Regression, VAR, DLinear, SES, NBEATS, Linear regression CalendarFourier and DeterministicProcess, ARIMA, RNN, GRU, LSTM

I. INTRODUCTION

Sự gia tăng của ô nhiễm không khí không chỉ ảnh hưởng đến sức khỏe con người mà còn gây ra nhiều vấn đề khác liên quan đến môi trường và kinh tế. Đặc biệt là ở Việt Nam, một quốc gia nằm trong khu vực có đặc điểm khí hậu phức tạp. Trong bối cảnh này, việc dự đoán và đánh giá chất lượng không khí trở thành một mối quan tâm hàng đầu. Và việc lựa chọn một mô hình dự đoán phù hợp và chính xác không phải là việc đơn giản. Mục tiêu của nghiên cứu này là áp dụng các phương pháp thống kê, mô hình học máy và học sâu để dự đoán chất lượng không khí. Chúng tôi đã lựa chọn và áp dụng một loạt các mô hình dự đoán như VAR (Vector Autoregression), Linear Regression áp dụng CalendarFourier và DeterministicProcess, SES (Simple Exponential Smoothing), DLinear (Dynamic Linear Model), và NBEATS (Neural Basis Expansion Analysis Time Series) để so sánh và đánh giá hiệu suất dự đoán của mỗi mô hình trên dữ liệu thực tế. Câu hỏi đặt ra cho nghiên cứu là “Các mô hình đã lựa chọn có đạt được hiệu suất dự đoán như mong đợi hay không?”. Nghiên cứu sẽ tập trung vào việc áp dụng các phương pháp dự đoán chất lượng không khí trên dữ liệu thực tế của ba thành phố lớn ở Việt Nam: Hà Nội, Đà Nẵng và Việt Trì. Phạm vi của nghiên cứu sẽ giới hạn trong việc

so sánh và đánh giá hiệu suất của các mô hình đã lựa chọn. Kết quả của nghiên cứu sẽ cung cấp thông tin quan trọng cho quyết định quản lý chất lượng không khí và phát triển bền vững tại các thành phố lớn. Ngoài ra, nó cũng có thể đề xuất các phương pháp mới và hiệu quả hơn trong việc dự đoán chất lượng không khí, mang lại lợi ích lớn cho việc quản lý môi trường và kinh tế.

II. RELATED WORKED

Trong những năm gần đây, dự báo chất lượng không khí đã trở thành một lĩnh vực nghiên cứu quan trọng do lo ngại ngày càng gia tăng về ô nhiễm môi trường. Các nhà nghiên cứu đã khám phá nhiều phương pháp thống kê, thuật toán học máy và kỹ thuật học sâu khác nhau để dự báo dữ liệu về thời tiết: Nur Izzah Jamil và đồng nghiệp (2019) đã tiến hành một phân tích toàn diện về chỉ số chất lượng không khí (Air Pollutant Index - API) tại Petaling Jaya, Malaysia[1]. Bằng cách sử dụng thuật toán Single Exponential Smoothing (SES), Double Exponential Smoothing (DES) và một số thuật toán khác để đánh giá và phân tích dữ liệu, họ đã đề xuất một phương pháp hiệu quả để đo lường và dự đoán mức độ ô nhiễm không khí trong khu vực. [1]

Jialin và đồng nghiệp (2023) đã đề xuất một phương pháp kết hợp xử lý ngoại lai (Outlier Correction), phân tách tín hiệu được cải tiến (Optimized Decomposition) và mô hình dự báo DLinear để dự báo tốc độ gió nhiều bước (multi-step-ahead wind speed forecast) tiên tiến, có độ chính xác cao. [2]

Taesung và đồng nghiệp đề xuất một phương pháp mới sử dụng mạng nơ-ron nhân tạo học sâu (deep neural networks) N-BEASTS[3] để xử lý các giá trị thiếu trong dữ liệu chất lượng không khí theo chuỗi thời gian. [3]

Linear Regression là một trong những phương pháp phổ biến và lâu đời được sử dụng cho bài toán dự đoán chất lượng không khí. A. Loganathan*, P. Sumithra, V. Deneshkumar đã đề xuất phương pháp ước tính mô hình hồi quy tuyến tính bội dựa trên thông tin liên quan đến chỉ số chất lượng không khí được ghi lại tại một trạm quan trắc ở Chennai, Ấn Độ. [4]

Mô hình Vector AutoRegression (VAR) có thể được áp dụng để dự đoán chất lượng không khí (AQI) với một số lợi ích quan trọng. Khaerun Nisa SH, Irfan Irfani, Utriweni Mukhaiyar đã đưa ra nghiên cứu dự đoán mức độ ô nhiễm không khí ở Jakarta bằng phương pháp phân tích Vector Autoregression

(VAR) trên dữ liệu chuỗi thời gian về mức độ ô nhiễm không khí (AQI) và nồng độ hạt vật chất (PM2.5) [5]

Linear Regression kết hợp với CalendarFourier và DeterministicProcess giúp cải thiện khả năng dự đoán chất lượng không khí (AQI) bằng cách xử lý thông tin về thời gian và tích hợp các yếu tố không ngẫu nhiên. Trong nghiên cứu này, các mô hình Hồi quy tuyến tính đa biến dự đoán Chỉ số chất lượng không khí ngắn hạn của Thị trấn Amravati nằm ở bang Maharashtra [6]

Bhalgat và cộng sự. (2019) đã trình bày một mô hình tích hợp để dự đoán mức độ ô nhiễm không khí bằng cách sử dụng mạng lưới thần kinh nhân tạo và kriging trong nghiên cứu của họ. Mô hình này sử dụng giao thức hồi quy tuyến tính và perceptron đa lớp (ANN) để dự đoán ngày hôm sau. Mô hình AR và ARIMA dự đoán thành công giá trị SO₂, nhưng cần nhiều nghiên cứu hơn để dự đoán PM2.5 và tính toán AQI [7]

Ong và cộng sự. (2016) đã kết nối RNN để dự đoán PM2.5 bằng thông tin cảm biến tự nhiên, từ đó đưa ra kết quả chính xác. Nghiên cứu của Kurt và Oktay (2010) về việc dự kiến ô nhiễm không khí với hệ thống thần kinh cho thấy ưu thế và khả năng đạt được của chiến lược. Liang và cộng sự. (2015) đã xuất bản một tập dữ liệu chứa ước tính PM2.5 vừa được ước tính ở Bắc Kinh. Sau đó, Liang và cộng sự. (2016) đã phân phối một bộ dữ liệu lớn hơn để phân tích yếu tố độc hại ở 5 khu vực thành thị của Trung Quốc. Tất cả các tập dữ liệu được sử dụng ở trên đều là thông tin theo điểm, không cho phép chúng tôi trình bày theo cách thể hiện không gian [8]

Bài toán dự đoán nồng độ chất ô nhiễm không khí là bài toán dự đoán chuỗi thời gian điển hình với nhiều biến đầu vào. Với mạng thần kinh tái phát GRU (Gated Recurrent Unit) có thể tìm hiểu thông tin phụ thuộc dài hạn và chúng ta có thể sử dụng nó trong lĩnh vực dự đoán chuỗi thời gian. Trong khi đó, nghiên cứu của Chung J đã xác nhận rằng GRU hoạt động tốt hơn trong việc hội tụ về thời gian CPU, cập nhật tham số và khai quát hóa so với LSTM (Mạng bộ nhớ ngắn dài). Công việc của chúng tôi chủ yếu là sử dụng GRU để thiết lập mô hình dự đoán nồng độ trung bình mỗi giờ PM2,5 cho Bắc Kinh, bao gồm mô hình quanh năm và bốn mô hình tương ứng với bốn mùa xuân, hạ, thu và đông, để cải thiện độ chính xác của Dự đoán nồng độ PM2.5. [9]

RNN có xu hướng tạo ra độ dốc khi xử lý chuỗi thời gian dài nên độ chính xác của nó thường kém. Để giải quyết vấn đề này, LSTM lần đầu tiên được giới thiệu bởi Hochreiter, S. et al. và tái xuất hiện như một kiến trúc thành công. Mạng nơ-ron LSTM là một biến thể của cấu trúc RNN. Ý tưởng chính của nó là giới thiệu một cơ chế chọn cổng thích ứng. Nó xác định mức độ mà đơn vị LSTM vẫn ở trạng thái trước đó. Nó cũng có thể ghi nhớ các tính năng được trích xuất của dữ liệu đầu vào hiện tại. Mặc dù có nhiều biến thể của LSTM được đề xuất nhưng kiến trúc tiêu chuẩn của LSTM được áp dụng trong bài báo này để dự đoán chất lượng không khí. [10]

III. MATERIALS

A. DATASET

Mục đích của nghiên cứu là dự báo chất lượng không khí tại 3 thành phố: Hà Nội, Việt Trì và Đà Nẵng. Vì vậy, một

bộ dữ liệu đã được thu thập về các phép đo liên quan đến chất lượng không khí ở ba thành phố này. Ba bộ dữ liệu đã được thu thập trên trang web AQICN, trong khoảng thời gian từ ngày 1 tháng 1 năm 2019 đến ngày 10 tháng 3 năm 2024 và bao gồm các thuộc tính được mô tả sau:

Bảng I
MÔ TẢ THUỘC TÍNH

Attribute	Description
Date	Ngày đo chất lượng không khí (dd/mm/yyyy)
PM2.5	Nồng độ bụi mịn PM2.5 (microgram/m ³)
PM10	Nồng độ bụi mịn PM10 (microgram/m ³)
O ₃	Nồng độ khí Ozon (microgram/m ³)
NO ₂	Nồng độ khí Nitrogen dioxide (microgram/m ³)
SO ₂	Nồng độ khí Sulfur dioxide (microgram/m ³)
CO	Nồng độ khí Carbon monoxide (mg/m ³)

B. DESCRIPTIVE STATISTICS

Bảng II
THỐNG KÊ MÔ TẢ AQI HÀ NỘI, ĐÀ NẴNG, VIỆT TRÌ

	AQIVietTri	AQIDaNang	AQIHanoi
Count	1695	1695	1695
Mean	75.55	77.51	110.49
Std	44.84	46.45	51.8
Min	10	10	15
25%	44	42.06	64
50%	58	71.19	108.33
75%	98	86	152.14
Max	228	362	267

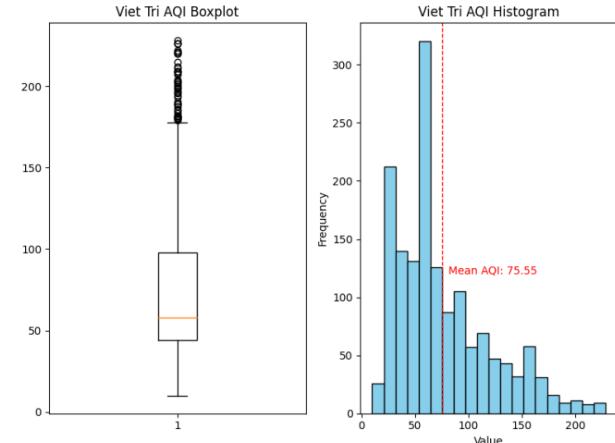


Fig1. Box Plot và Histogram Chart của AQI Việt Trì

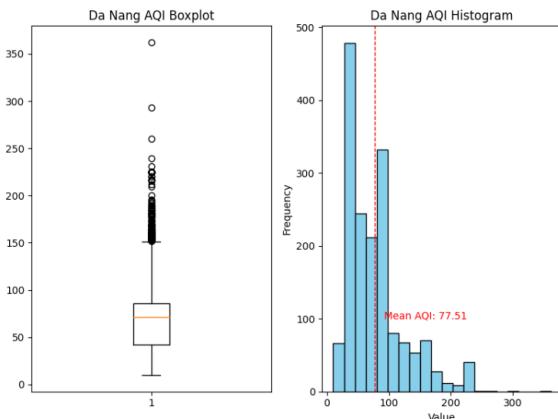


Fig2. Box Plot và Histogram Chart của AQI Đà Nẵng

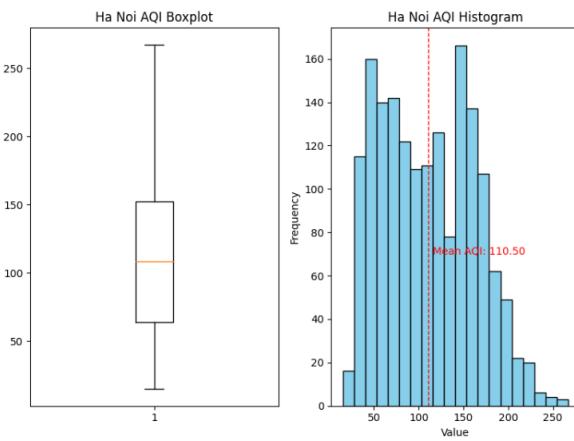


Fig3. Box Plot và Histogram Chart của AQI Hà Nội

C. TOOL

Trong nghiên cứu này, chúng tôi sử dụng ngôn ngữ lập trình Python cùng với các thư viện và công cụ phổ biến trong lĩnh vực thống kê, máy học và học sâu, với mục đích thực hiện phân tích thống kê và huấn luyện mô hình cũng như đánh giá hiệu suất của các mô hình dự đoán chất lượng không khí của chúng tôi một cách hiệu quả và linh hoạt.

D. DATA SPLIT RATIO

Phân chia dữ liệu là một phần thiết yếu của việc xây dựng mô hình. Phân chia dữ liệu kém có thể dẫn đến hiệu suất mô hình không chính xác và biến động cao. Tập huấn luyện bao gồm các điểm dữ liệu được sử dụng trực tiếp để xây dựng mô hình. Tập kiểm tra bao gồm dữ liệu được sử dụng để đánh giá hiệu suất của mô hình. Để đảm bảo khả năng tổng quát, dữ liệu kiểm tra không được sử dụng trong quá trình xây dựng mô hình. Một điều kiện cần thiết cho một mô hình hiệu quả là kết quả đánh giá trên cả tập huấn luyện và tập kiểm tra đều cao.

Dựa trên kết quả kiểm tra, tỷ lệ dữ liệu tối ưu cho mô hình được chọn là 80:20, với 80% dữ liệu được sử dụng cho việc huấn luyện và 20% dữ liệu được sử dụng cho việc kiểm tra

E. MODEL EVALUATION

Root Mean Squared Error (RMSE) đo lường sự khác biệt trung bình giữa giá trị dự đoán của mô hình thống kê và giá trị thực tế. Về mặt toán học, đó là độ lệch chuẩn của phần dư. Phần dư biểu thị khoảng cách giữa đường hồi quy và các điểm dữ liệu. RMSE được tính theo công thức sau:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2}$$

Mean absolute error (MAE) là thước đo mức trung bình của các sai số trong một tập hợp các dự đoán mà không tính đến hướng của chúng. Nó được đo bằng chênh lệch tuyệt đối trung bình giữa giá trị dự đoán và giá trị thực tế và được sử dụng để đánh giá tính hiệu quả của mô hình hồi quy. MAE được tính theo công thức sau:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

Mean absolute percentage error (MAPE) tính toán chênh lệch phần trăm trung bình giữa giá trị dự báo và giá trị thực tế. Điều này giúp đánh giá tính chính xác của một mô hình dự báo hoặc quy trình dự báo cụ thể. MAPE được tính theo công thức sau:

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100$$

Where:

- n là số lượng quan sát trong tập dữ liệu.
- y_i là giá trị thực.
- \hat{y}_i là giá trị dự đoán

IV. METHODOLOGY

A. VAR - VECTOR AUTOREGRESSION

Tự hồi quy vectơ (VAR) là một mô hình thống kê được sử dụng trong kinh tế lượng, có thể được sử dụng khi hai hoặc nhiều chuỗi thời gian ảnh hưởng lẫn nhau. Nghĩa là, mỗi quan hệ giữa chuỗi thời gian liên quan là hai chiều. Nó ước tính từng phương trình của từng biến chuỗi theo độ trễ của biến (p) và tất cả các biến còn lại. n dự đoán tốt nhất biến y là hàm tuyến tính của các biến x .

Một phương trình mô hình AR(p) điển hình có dạng sau:

$$Y_t = \alpha_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_p Y_{t-p} + \varepsilon_t$$

Trong đó:

- α_0 là hệ số chặn
- β_1, \dots, β_p là các hệ số hồi quy cho các biến độc lập.
- ε là hệ số lỗi.

Đối với mô hình VAR với 2 chuỗi thời gian (Y_1, Y_2), ta có hệ phương trình như sau:

$$Y_{1,t} = \alpha_0 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \cdots + \varepsilon_{1,t}$$

$$Y_{2,t} = \alpha_0 + \beta_{21,1}Y_{1,t-1} + \beta_{22,1}Y_{2,t-1} + \dots + \varepsilon_{2,t}$$

Trong đó:

- $Y_{1,t}, Y_{2,t}$ lần lượt là độ trễ đầu tiên của chuỗi thời gian Y1 và Y2.
- $\beta_{ij,k}$ Hệ số đo lường tác động của biến trễ $Y_{i,t-k}$ on $Y_{i,t}$.
- $\varepsilon_{1,t}, \varepsilon_{2,t}$ là những hệ số lỗi.

B. LINEAR REGRESSION APPLY CALENDARFOURIER AND DETERMINISTICPROCESS

1) Linear Regression

Mô hình hồi quy tuyến tính là một phương pháp trong thống kê để dự đoán giá trị của biến phụ thuộc dựa trên 2 hoặc nhiều biến độc lập. Nó giả định mối quan hệ tuyến tính giữa các biến và sử dụng phương pháp hồi quy để ước lượng các hệ số.

2) CalendarFourier

Lớp CalendarFourier trong statsmodels là một công cụ mạnh mẽ để mô hình các thành phần xác định dựa trên thời gian trong lịch. Đặc biệt, nó hữu ích khi làm việc với dữ liệu chuỗi thời gian có các mẫu theo mùa, được sử dụng để tạo ra các thành phần xác định dựa trên thời gian trong chuỗi thời gian bằng cách sử dụng chuỗi Fourier. Thường được sử dụng để mô hình hóa các yếu tố theo chu kỳ trong chuỗi thời gian, ví dụ như yếu tố hàng năm hoặc hàng tháng.

CalendarFourier được sử dụng để mô hình hóa tính thời vụ bằng chuỗi Fourier. Nó có thể hữu ích để chụp các mẫu theo mùa phức tạp

$$\text{Fourier terms} = [\sin(t), \cos(t), \sin(2t), \cos(2t), \dots, \sin(kt), \cos(kt)]$$

3) DeterministicProcess

DeterministicProcess là một lớp chung có thể được sử dụng để chỉ định thành phần xác định trong mô hình chuỗi thời gian. Nó có thể bao gồm các thuật ngữ như biến giả không đổi, xu hướng và theo mùa

Hỗ trợ tạo ra các thành phần xác định như hằng số, xu hướng thời gian, hoặc các yếu tố theo chu kỳ như mùa vụ hoặc chuỗi Fourier.

Thường được sử dụng để thêm các thành phần xác định vào mô hình hồi quy tuyến tính.

Trong phân tích chuỗi thời gian và hồi quy tuyến tính, chúng ta sử dụng CalendarFourier và DeterministicProcess để mô hình hóa các yếu tố xác định dựa trên thời gian.

4) Linear Regression áp dụng calendarFourier and DeterministicProcess

Xử lý các thành phần chu kỳ: CalendarFourier được sử dụng để mô hình các thành phần chu kỳ, như các mẫu mùa vụ, trong dữ liệu chuỗi thời gian bằng cách sử dụng chuỗi Fourier. Điều này cho phép mô hình nắm bắt được các biến động mùa vụ trong dữ liệu

Tăng tính linh hoạt của mô hình: Sử dụng các thành phần xác định như Constant và Trend từ DeterministicProcess có thể tăng tính linh hoạt của mô hình để nắm bắt cả các biến động dài hạn và ngắn hạn trong dữ liệu chuỗi thời gian.

C. ARIMA

ARIMA là viết tắt của "Autoregressive Integrated Moving Average", là một phương pháp dự báo thống kê được sử dụng rộng rãi trong phân tích chuỗi thời gian. Mô hình này tích hợp các thành phần Autoregressive (AR), Moving Average (MA), và Differencing (I) để bắt các mối quan hệ giữa giá trị hiện tại và quá khứ của một chuỗi thời gian.

Autoregressive (AR):

AR (Tự hồi quy) - là quá trình tìm mối quan hệ giữa dữ liệu hiện tại và p dữ liệu quá khứ trước đó (Gọi là lag):

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + \epsilon_t$$

Trong đó:

- y_t là giá trị hiện tại.
- c là thuật ngữ hằng số.
- p là số lượng đơn đặt hàng.
- ϕ là hệ số tự hồi quy.
- ϵ_t là thuật ngữ lỗi.

Moving Average (MA): MA (Trung bình di động) - là quá trình tìm mối quan hệ giữa dữ liệu hiện tại và q phần lỗi quá khứ trước đó.

$$y_t = c + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \dots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Differencing (I): So sánh sự khác nhau giữa d quan sát (Hiệu giữa giá trị hiện tại và d giá trị trước đó):

Nếu $d = 0$: $\Delta Y_t = Y_t$

Nếu $d = 1$: $\Delta Y_t = Y_t - Y_{t-1}$

Nếu $d = 2$: $\Delta Y_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2}) = Y_t + 2Y_{t-1} + Y_{t-2}$

Sau khi kết hợp chúng, chúng ta sẽ có ARIMA (p, d, q) được biểu diễn như sau:

$$\Delta Y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} \\ \dots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Mô hình ARIMA nắm bắt mối quan hệ giữa giá trị hiện tại và quá khứ của một chuỗi thời gian, tích hợp các thành phần tự hồi quy, trung bình di động và đạo hàm. Chúng được sử dụng rộng rãi để dự báo giá trị tương lai của các chuỗi thời gian khác nhau, bao gồm dữ liệu kinh tế, giá cổ phiếu và mô hình thời tiết.

D. LINEAR REGRESSION

Phân tích hồi quy là một công cụ để xây dựng các mô hình toán học và thống kê mô tả mối quan hệ giữa một biến phụ thuộc và một hoặc nhiều biến độc lập hoặc biến giải thích, tất cả đều là số. Kỹ thuật thống kê này được sử dụng để tìm phương trình dự đoán tốt nhất biến y dưới dạng hàm tuyến tính của biến x. Mô hình hồi quy tuyến tính bối có dạng:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k + \varepsilon$$

Trong đó:

- Y là biến phụ thuộc (Biến mục tiêu)
- X_1, X_2, \dots, X_k là các biến độc lập
- β_0 là hệ số chặn.
- β_1, \dots, β_k là các hệ số hồi quy cho các biến độc lập.
- ε là hệ số lỗi.

E. SES - SIMPLE EXPONENTIAL SMOOTHING

1) Exponential Smoothing

Exponential Smoothing là một kỹ thuật thống kê cơ bản có thể được sử dụng để làm mịn chuỗi thời gian. Các mô hình chuỗi thời gian thường có nhiều biến đổi dài hạn nhưng cũng có nhiều biến đổi ngắn hạn (ồn ào). Làm mịn cho phép đường cong của chuỗi thời gian mượt mà hơn để sự biến đổi trong thời gian dài trở nên rõ ràng hơn và các mẫu ngắn hạn (ồn ào) được loại bỏ. Phiên bản trơn tru này của chuỗi thời gian sau đó có thể được sử dụng để phân tích.

2) Simple Moving Average

Simple moving average là kỹ thuật làm mịn đơn giản nhất. Nó bao gồm việc thay thế giá trị hiện tại bằng giá trị trung bình của giá trị hiện tại và một vài giá trị trong quá khứ. Số lượng các giá trị trong quá khứ được sử dụng là một tham số. Càng sử dụng nhiều giá trị, đường cong sẽ càng mượt mà. Đồng thời, sẽ ngày càng mất đi nhiều biến thể.

3) Simple Exponential Smoothing

Simple exponential smoothing tương tự simple moving average. Nhưng thay vì lấy giá trị trung bình, nó lấy giá trị trung bình có trọng số của các giá trị trong quá khứ. Giá trị lùi xa hơn sẽ có trọng số thấp hơn và giá trị gần đây hơn sẽ có trọng số nhiều hơn.

Simple exponential smoothing có công thức như sau:

$$F_t = F_{t-1} + a \cdot (A_{t-1} - F_{t-1})$$

Với a là smoothing factor, và $0 \leq a \leq 1$. F_t (giá trị dự đoán ở thời gian t) là trung bình có trọng số của A_{t-1} (giá trị thực tế tại thời gian $t-1$) và F_{t-1} (giá trị dự đoán tại thời gian $t-1$).

Không có một cách cụ thể để tính toán a . Nhưng cũng có nhiều cách để tối ưu hóa độ chính xác của a . Ví dụ, có thể sử dụng phương pháp bình phương tối thiểu để tính giá trị của a , cụ thể là tìm a để tổng các $(F_t - A_{t+1})^2$ là thấp nhất.

F. DECOMPOSED LINEAR (D-LINEAR)

D-Lineare sử dụng một phương pháp phân rã để chia dữ liệu thô thành phần xu hướng và mùa vụ. Sau đó, hai mạng tuyến tính một tầng được áp dụng cho mỗi thành phần và các đầu ra được cộng lại để có được dự đoán cuối cùng.

1) Cấu trúc của DLinear

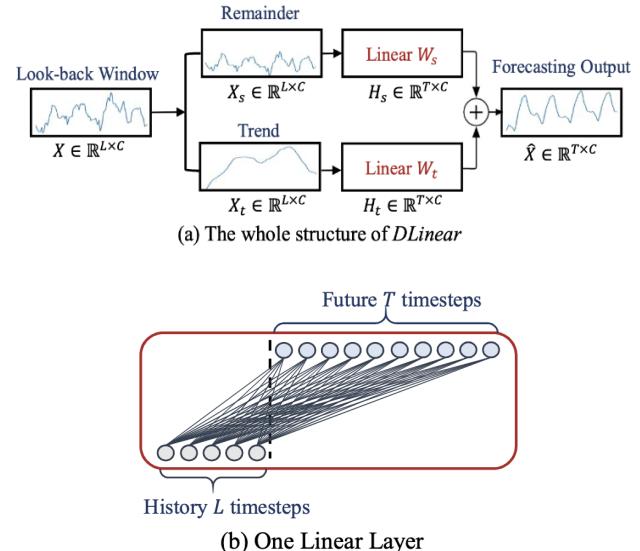


Fig. Hình minh họa của mô hình Decomposition Linear

Cấu trúc tổng quan được thể hiện ở hình (a). Toàn bộ quá trình: $\hat{H} = H_s + H_t$, trong đó $H_s = W_s X_s \in \mathbb{R}^{T \times C}$ và $H_t = W_t X_t \in \mathbb{R}^{T \times C}$ là các thành phần xu hướng và phần dư đã được phân rã. $W_s \in \mathbb{R}^{T \times C}$ và $W_t \in \mathbb{R}^{T \times C}$ là hai lớp tuyến tính, như được minh họa trong Hình (b).

2) Kiến trúc Phân rã

Với độ dài- L của chuỗi đầu vào $X \in \mathbb{R}^{T \times C}$:

$$\begin{aligned} X_t &= \text{AvgPool}(\text{Padding}(X)) \\ X_s &= X - X_t, \end{aligned}$$

trong đó $X_s, X_t \in \mathbb{R}^{T \times C}$ đại diện cho phần mùa và phần xu hướng chu kỳ được trích xuất tương ứng.

G. NEURAL BASIS EXPANSION ANALYSIS

1) Dữ liệu đầu vào

Xét bài toán dự báo đơn chuỗi thời gian:

Từ chuỗi thời gian có độ dài t là $x = [y_{T-t+1}, y_{T-t+2}, \dots, y_T] \in \mathbb{R}^T$ ta cần dự báo giá trị của H bước tiếp theo là $x = [y_{T+1}, y_{T+2}, \dots, y_{T+H}] \in \mathbb{R}^H$. Ta kí hiệu \hat{y}_i là dự đoán của mô hình cho vectơ y . Kích thước của dữ liệu đầu vào là $t = nH$ (n thường nằm trong khoảng từ [2, 7]) được gọi là khoảng thời gian xem lại (lookback period). Mô hình N-BEATS sẽ học và tìm hiểu chuỗi thời gian từ khoảng thời gian xem lại để dự đoán giá trị của H điểm tiếp theo.

2) Kiến trúc tổng quát của mô hình

Kiến trúc tổng quát của mô hình N-BEATS được mô tả như trong hình bên dưới. Mô hình bao gồm các block được xếp chồng lên nhau. Đầu vào của block đầu tiên là đầu vào tổng thể của mô hình hay còn gọi là khoảng thời gian xem lại x . Mỗi block có hai đầu ra. Một đầu ra (backcast) sẽ được làm

đầu vào cho block tiếp theo. Đầu ra còn lại (forecast) sẽ được tổng hợp để đưa ra kết quả cuối cùng.

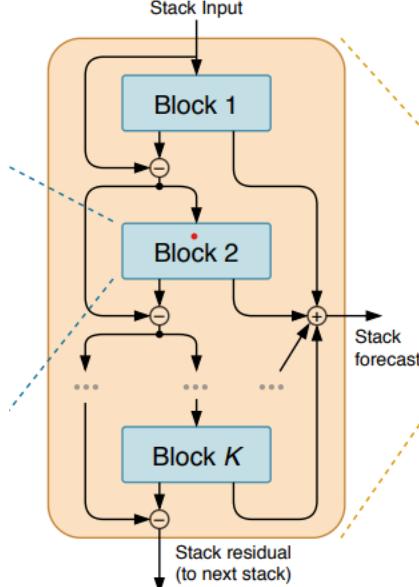


Fig4. Kiến trúc tổng quát của mô hình N-BEATS

Chi tiết hơn, với block thứ i trong mô hình:

- Nếu $i = 1 = 1$, tức là block đầu tiên, đầu vào chính là cửa sổ dữ liệu x
 - Nếu $i > 1$, đầu vào của block thứ i là: $x^i = x^{i-1} - \hat{x}^{i-1}$
- Block thứ i gồm hai đầu ra:
- Đầu ra backcast \hat{x}^i được làm đầu vào cho block tiếp theo.
 - Đầu ra focecast \hat{y}^i dùng để tổng hợp cho kết quả dự đoán cuối cùng:

$$\hat{y} = \sum_{i=1}^R \hat{y}^i$$

Từ đây, ta thấy được đầu vào của block sẽ không chứa mà block trước đó đã dự đoán được. Điều này giúp các block sau tập trung vào những phần thông tin chưa được dự đoán.

3) Kiến trúc của block

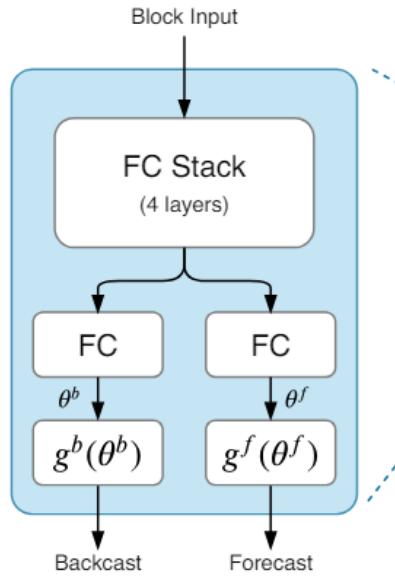


Fig5. Kiến trúc của block

Hình trên mô tả kiến trúc trong block thứ i của mô hình. Kiến trúc của block đơn giản chỉ bao gồm các lớp kết nối đầy đủ **FC** (fully connected). Các lớp kết nối đầy đủ đơn giản là một lớp ánh xạ tuyến tính với hàm kích hoạt **RELU**. Ví dụ:

$$FC(x^i) = RELU(Wx^i + b)$$

trong đó W và b lần lượt là các ma trận sẽ được học trong quá trình huấn luyện. Hàm **RELU** là hàm $f(x) = \max(0, x)$. Block bao gồm M lớp **FC** chung (M thường là 4). Sau đó, sẽ có hai lớp **FC** là $M+1$ và $M+2$ để chia đầu ra thành 2 nhánh. Đầu ra của hai lớp trên lần lượt là hai vectơ hệ số θ^f và θ^b . Cuối cùng, ta sẽ ánh xạ hai vectơ trên thành các đầu ra backcast và forecast thông qua các hàm g^b và g^f . Tùy vào dạng hàm g khác nhau mà chúng ta sẽ có các loại block với vai trò khác nhau. Cụ thể, mô hình N-BEATS giới thiệu ba loại block: trend block, seasonality block, generic block.

Generic block là kiến trúc không mang tính diển giải được trong chuỗi thời gian. Generic block đơn giản đặt các ánh xạ g^b và g^f là các ánh xạ tuyến tính đối với đầu ra của lớp trước.

Trend block là loại block để nhận biết và phân tích được tính xu hướng của dữ liệu. Một đặc điểm của xu hướng thường phần lớn là các hàm đơn điệu hoặc thay đổi chậm. Để tìm hiểu được tính xu hướng của dữ liệu, các hàm g^b và g^f sẽ là các hàm đa thức với bậc p nhỏ theo thời gian. Khi đó, các hệ số θ^f và θ^b đóng vai trò là các hệ số của đa thức. Ví dụ đầu ra forecast của trend block thứ i là:

$$\hat{y}^i = \sum_{j=1}^R \theta_j^f t^j$$

Trong đó, θ_j^f là phần tử thứ j trong vectơ hệ số θ^f và t là phần tử trong vectơ lướt thời gian:

$$t = \frac{[0, 1, 2, \dots, H-2, H-1]^T}{H}$$

Cuối cùng, loại seasonality block dùng để nhận biết tính mùa của dữ liệu. Đặc điểm chính của tính mùa là dữ liệu thường

có tính chu kỳ. Do đó, để mô phỏng tính mùa, các hàm g^b và g^f sẽ là các loại hàm tuần hoàn, ví dụ $y_t = y_{t+\delta}$ trong đó δ là chu kỳ mùa. Sử dụng chuỗi Fourier để xây dựng các hàm tuần hoàn trên, ta có đầu ra forecast của seasonality block thứ i là:

$$y^i = \sum_{j=0}^{[\frac{H}{2}-1]} \theta_j^f \cos(2\pi i t) + \theta_{j+[\frac{H}{2}]}^f \sin(2\pi i t)$$

Sau khi giới thiệu ba loại block trên, mô hình N-BEATS được xây dựng dựa trên số lượng và thứ tự của mỗi loại block.

H. Gated Recurrent Units (GRU)

Gated Recurrent Units được giới thiệu bởi Kyunghyun Cho và các đồng nghiệp vào năm 2014 như một giải pháp cho vấn đề vanishing gradient. GRU sử dụng cơ chế cổng để kiểm soát luồng thông tin. Những cổng này xác định thông tin nào nên được chuyển đến đầu ra và thông tin nào nên tiếp tục được giữ lại trong trạng thái nội bộ của mạng, cho phép mô hình nắm bắt tốt hơn các phụ thuộc cho các chuỗi có độ dài đa dạng.

Kiến trúc GRU

GRU có hai cổng:

Cổng Cập Nhật: Cổng cập nhật giúp mô hình xác định bao nhiêu thông tin từ quá khứ (từ các bước thời gian trước) cần được truyền đến tương lai. Điều này rất quan trọng để mô hình có thể nắm bắt các phụ thuộc dài hạn và quyết định cái gì cần giữ trong bộ nhớ.

Cổng Đặt Lại: Cổng đặt lại quyết định bao nhiêu thông tin từ quá khứ cần được quên đi. Nó cho phép mô hình quyết định mức độ quan trọng của mỗi bước vào đối với trạng thái hiện tại và hữu ích cho việc đưa ra dự đoán.

Các cổng này là các vector chứa giá trị từ 0 đến 1. Các giá trị này được tính bằng cách sử dụng hàm kích hoạt sigmoid. Giá trị gần với 0 có nghĩa là cổng đóng và không có thông tin nào được truyền qua, trong khi giá trị gần với 1 có nghĩa là cổng mở và tất cả thông tin được truyền qua.

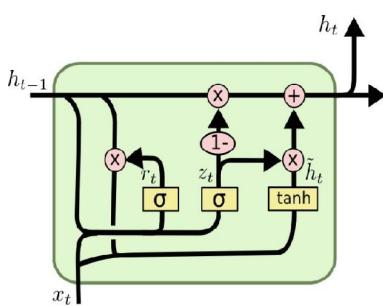


Fig. Sơ đồ luồng kiến trúc của GRU.

Các phương trình được sử dụng để tính toán cổng đặt lại, cổng cập nhật và trạng thái ẩn của một GRU như sau:

$$\text{Reset gate: } r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

$$\text{Update gate: } z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$\text{Candidate hidden state: } \tilde{h}_t = \tanh(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h)$$

$$\text{Hidden state: } h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

Trong :

- h_t là vector lớp ẩn.
- x_t là vector đầu vào.
- b_z, b_r, b_h là những bias vector.
- W_z, W_r, W_h là các ma trận tham số.
- σ, \tanh là những hàm kích hoạt.

I. LONG SHORT-TERM MEMORY

Mô hình đề xuất dựa trên mô hình mạng thần kinh sâu LSTM, đây là một dạng đặc biệt của RNN (Recurrent Neural Network - Mạng thần kinh hồi quy). LSTM được giới thiệu bởi Hochreiter và Schmidhuber (1997) nhằm giải quyết các bài toán về phụ thuộc xa (long-term dependency)

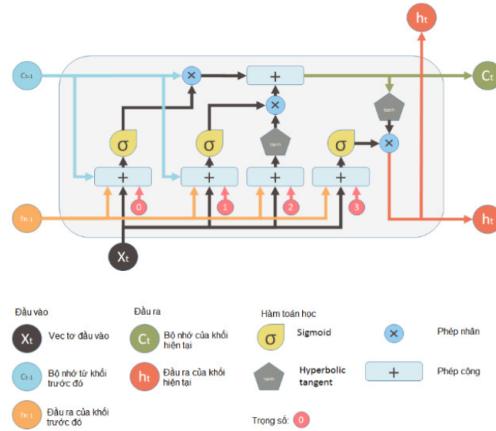


Fig6. Cấu trúc của mô hình LSTM.

Theo Olah (2015), mọi mạng hồi quy đều có dạng là một chuỗi các mô đun lặp đi lặp lại của một mạng thần kinh, mỗi mô đun này thường có cấu trúc đơn giản được gọi là một tầng “tanh”. LSTM cũng có kiến trúc dạng chuỗi như vậy và thay vì chỉ có 1 tầng mạng thần kinh như RNN chuẩn thì chúng có tới 4 tầng và tương tác với nhau một cách đặc biệt. Cấu trúc của mô hình mạng thần kinh LSTM được thể hiện ở Hình 1. Cốt lõi của LSTM bao gồm trạng thái tế bào (cell state) và cổng (gate). Trạng thái tế bào giống như băng chuyền, chạy xuyên suốt qua tất cả các nút mạng giúp thông tin được truyền đạt dễ dàng, còn cổng là nơi sàng lọc thông tin đi qua nó, chúng được kết hợp bởi một tầng mạng sigmoid. Một LSTM gồm có 3 cổng để duy trì hoạt động trạng thái của tế bào. Bước đầu tiên của mô hình LSTM được gọi là tầng cổng quên (forget gate layer). Bước này sẽ quyết định xem thông tin nào cần bỏ đi từ trạng thái tế bào. Đầu vào cho bước này là h_{t-1} (giá trị đầu ra tại thời điểm $t-1$) và x_t (dữ liệu đầu vào); đầu ra f_t là một số trong khoảng từ 0 đến 1 cho mỗi số trong trạng thái tế bào C_{t-1} .

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Trong đó: σ là hàm sigmoid, W_f và b_f lần lượt là trọng số và tham số của tầng cổng quên.

Các bước tiếp theo sẽ quyết định thông tin lưu vào trạng thái tế bào và cập nhật giá trị cho trạng thái. Bao gồm một tầng sigmoid hay còn được gọi là cổng vào (input gate layer, i_t) và một véc tơ giá trị được tạo từ tầng tanh.

$$\begin{aligned} i_t &= \sigma(W_t \cdot [h_{t-1}, x_t] + b_t) \\ \hat{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \hat{C}_t \end{aligned}$$

Trong đó: C_{t-1} và C_t là trạng thái tế bào lần lượt ở thời điểm $t-1$ and t ; W_C và b_C lần lượt là trọng số và tham số của trạng thái tế bào.

Ở bước cuối cùng, giá trị đầu ra (h_t) sẽ được quyết định bởi trạng thái của tế bào muốn xuất ra (output gate, o_t).

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh(C_t) \end{aligned}$$

J. RNN - Recurrent Neural Network

1) Recurrent Neural Networks

Mạng hồi truy hồi (Recurrent Neural Network) là một họ của các mạng neural dùng để chuyên xử lý dữ liệu dạng chuỗi.

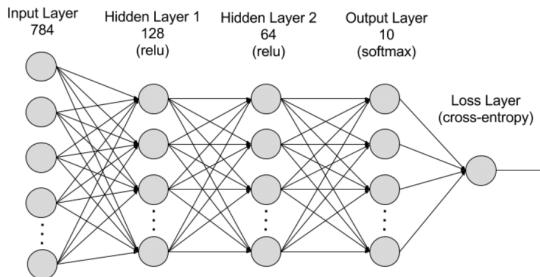


Fig. RNN Diagram

RNN được tạo thành từ các nơ-ron: các nút xử lý dữ liệu phối hợp với nhau để thực hiện các tác vụ phức tạp. Các nơ-ron được tổ chức thành các lớp đầu vào, đầu ra và các lớp ẩn. Lớp đầu vào nhận thông tin để xử lý và lớp đầu ra cung cấp kết quả. Việc xử lý, phân tích và dự đoán dữ liệu diễn ra ở lớp ẩn.

RNN hoạt động bằng cách truyền dữ liệu mà nó nhận được đến các lớp ẩn từng bước một. Tuy nhiên, chúng cũng có quy trình làm việc tự lặp hoặc lặp lại: lớp ẩn có thể ghi nhớ và sử dụng các thông tin input trước đó cho việc dự đoán trong bộ nhớ ngắn hạn. Nó sử dụng input hiện tại và bộ nhớ được lưu trữ để dự đoán trình tự tiếp theo.

2) Mô hình RNN đơn giản

Gọi t là thời điểm xét, $t-1$ là thời điểm quá khứ sau một đơn vị so với t :

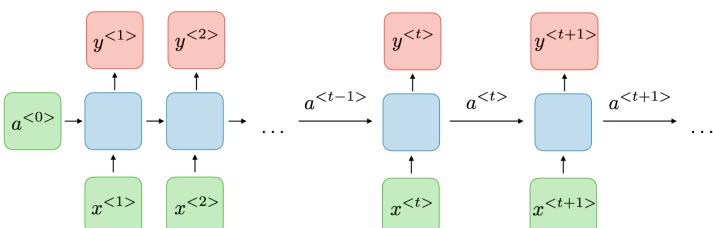


Fig. Mô hình RNN đơn giản

$a^{<t>}$ là trạng thái thời điểm t .

$y^{<t>}$ là giá trị output thời điểm t .

$x^{<t>}$ là giá trị input thời điểm t .

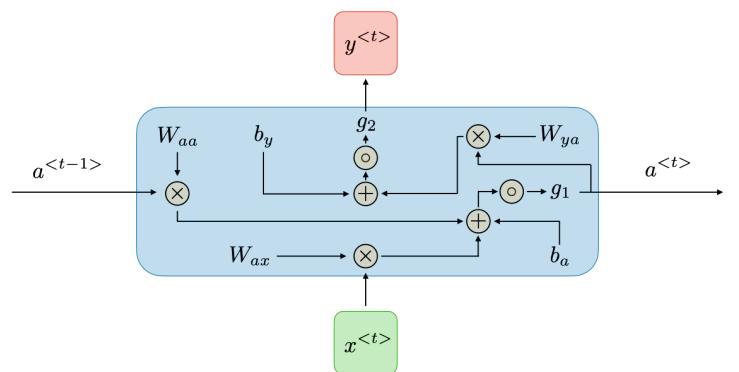


Fig. Mô hình RNN đơn giản

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$

Và

$$\tilde{y}_t = y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$$

Với W_{ax} , W_{aa} , W_{ya} là các trọng số, thường là random, g_1 , g_2 là các hàm kích hoạt, b_a , b_y là các bias.

$$\text{Loss function} = \frac{1}{n} \left(\sum_{t=1}^N (y_t - \tilde{y}_t)^2 \right).$$

Thuật toán RNN trong chuỗi thời gian:

Algorithm 1: Thuật toán Backpropagation trong mạng nơ-ron hồi quy (RNN)

Đầu vào: Dữ liệu đầu vào chuỗi thời gian x_1, x_2, \dots, x_T , nhãn thực tế tương ứng y_1, y_2, \dots, y_T , số lượng mẫu trong chuỗi T , tỷ lệ học η

Đầu ra : Cập nhật các tham số W_{ay} , b_y , W_{aa} , W_{xa} , b_h

Khởi tạo tham số: Khởi tạo W_{ay} , b_y , W_{aa} , W_{xa} , b_h ;

for $t \leftarrow 1$ to T **do**

Lần truyền tiền (Forward pass):
 $a_t = \sigma(W_{aa}a_{t-1} + W_{xa}x_t + b_h)$;
 $y_t = \text{softmax}(W_{ay}a_t + b_y)$;

 Tính toán hàm mất mát:;

$$L = -\sum_{t=1}^T \sum_i y_{t,i} \log(\hat{y}_{t,i})$$

Lần truyền ngược (Backward pass):

for $t \leftarrow T$ to 1 **do**

 Tính gradient của loss theo đầu ra y_t ;
 $dL_dy = \text{gradient_cross_entropy_loss}(y[t], y_true[t])$;
 Tính gradient của loss theo các tham số:;
 $dW_{ay} += dL_dy \times a_t$;
 $db_y += dL_dy$;
 $dh = (dL_dy \times W_{ay}) + dh_next$;
 $dh_raw = dh \times \text{activation_derivative}(h[t])$;
 $dW_{aa} += dh_raw \times a_{t-1}$;
 $dW_{xa} += dh_raw \times x_t$;
 $db_h += dh_raw$;
 $dh_next = dh_raw \times W_{aa}$;

 Cập nhật tham số:;

$$W_{ay} -= \eta \times dW_{ay}$$

$$b_y -= \eta \times db_y$$

$$W_{aa} -= \eta \times dW_{aa}$$

$$W_{xa} -= \eta \times dW_{xa}$$

$$b_h -= \eta \times db_h$$

 Lặp lại quá trình trên cho một số lần hoặc cho đến khi hàm mất mát giảm dù;

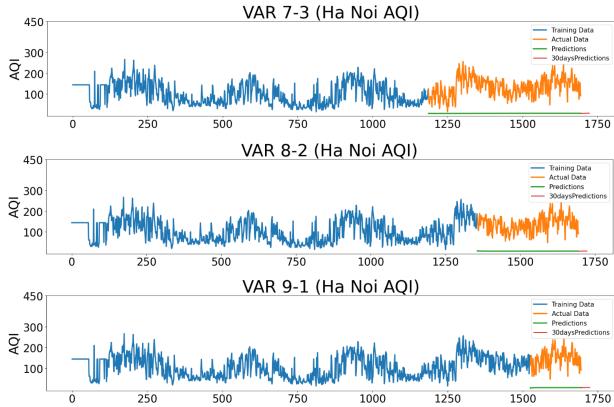
Fig. Thuật toán RNN trong chuỗi thời gian

V. EXPERIMENT

A. MODEL SETTING

1) VAR

a) Ha Noi AQI

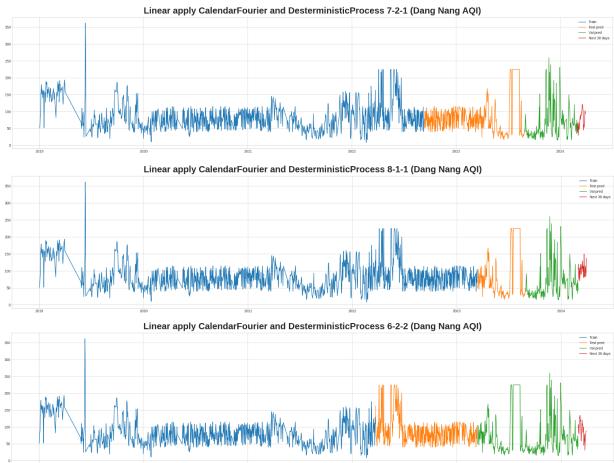


b) Da Nang AQI

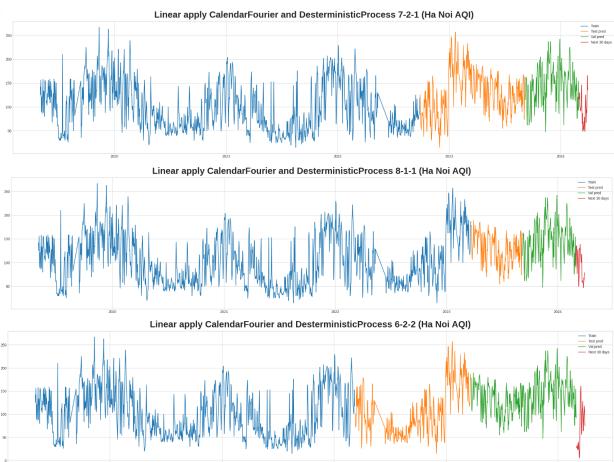
c) Viet Tri AQI

2) LINEAR REGRESSION APPLY CALENDARFOURIER. DETERMINISTICPROCESS

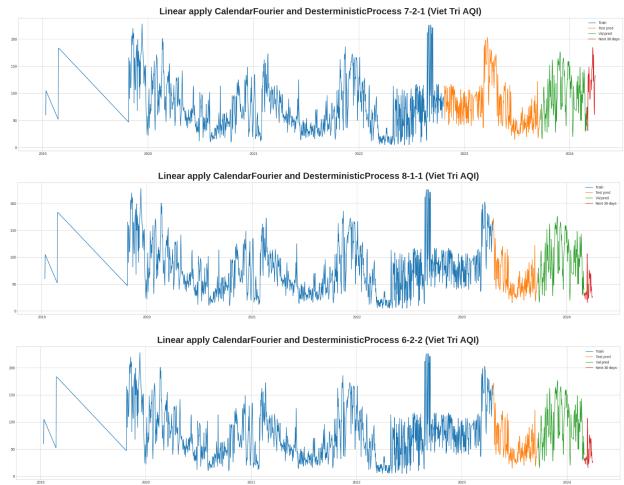
a) Da Nang AQI



b) Ha Noi AQI

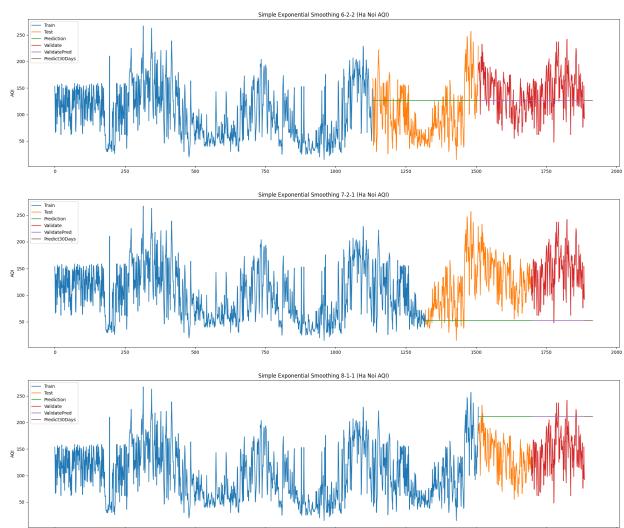


c) Viet Tri AQI

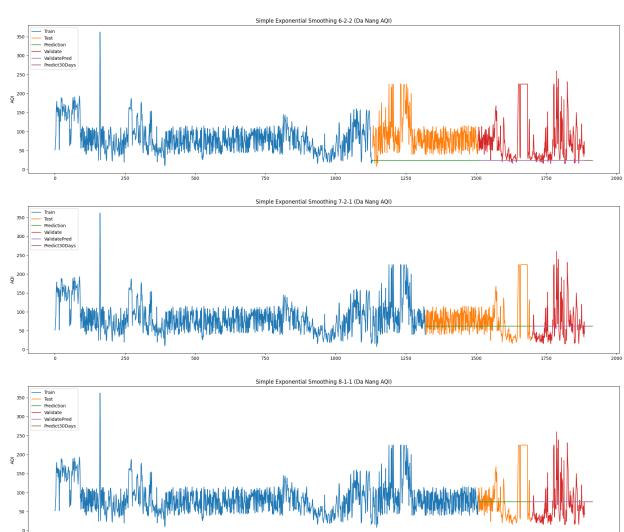


3) SES

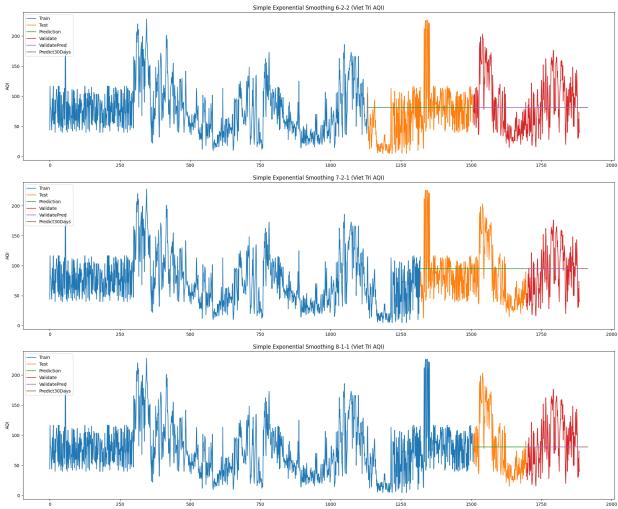
a) Ha Noi AQI



b) Da Nang AQI



c) Viet Tri AQI

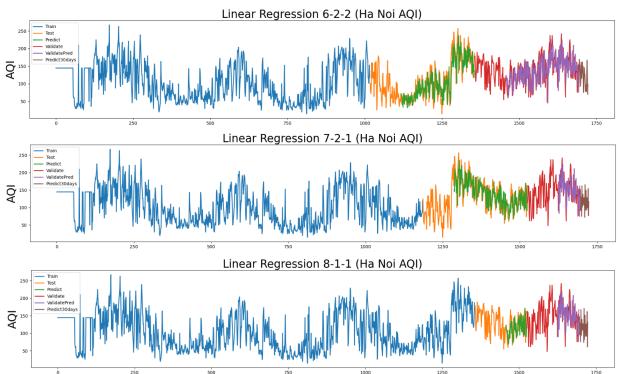


4) DLINAR

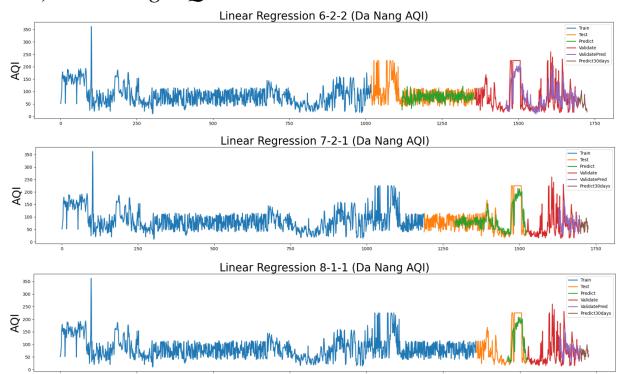
5) NBEATS

6) LINEAR REGRESSION

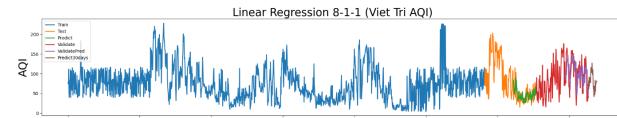
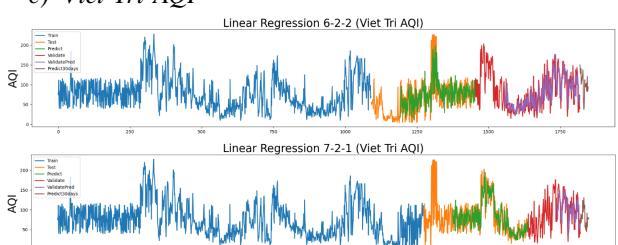
a) Ha Noi AQI



b) Da Nang AQI

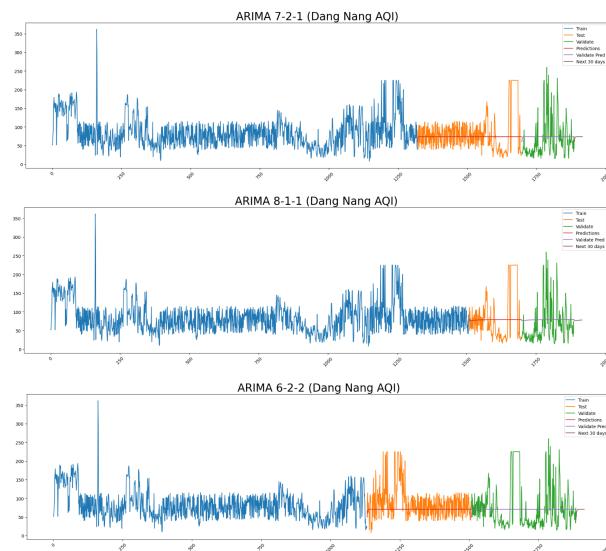


c) Viet Tri AQI

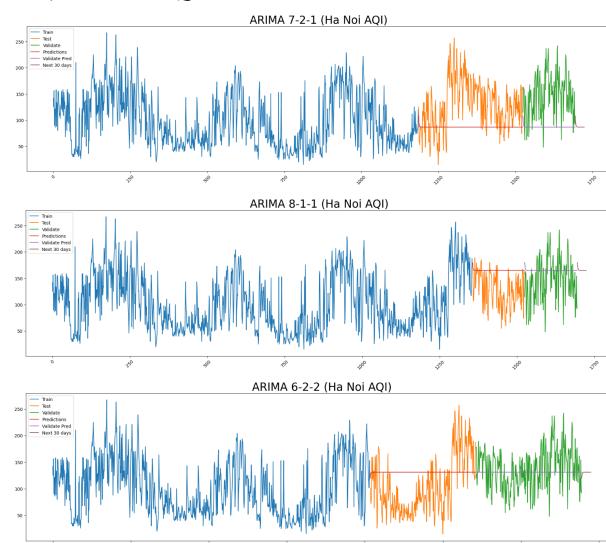


7) ARIMA

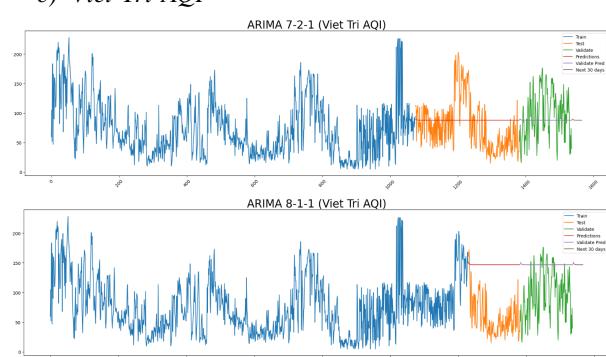
a) Da Nang AQI

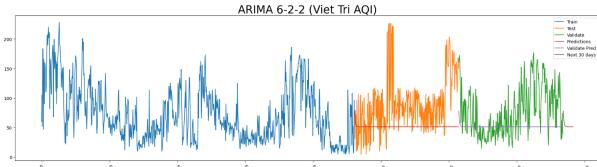


b) Ha Noi AQI



c) Viet Tri AQI





8) *RNN*

9) *GRU*

10) *LSTM*

B. EVALUATION MODELS AND DISCUSSION

VI. CONCLUSION

ACKNOWLEDGMENT

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first . . .”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

Tài liệu

- [1] Jamil, Nur & Amit, Norani & Yusof, Noreha. (2020). Model Evaluation on Air Pollutant Index (API) in Petaling Jaya, Malaysia. 29. 1959-1966
- [2] Liu, Jialin. “Multi-Step-Ahead Wind Speed Forecast Method Based on Outlier Correction, Optimized Decomposition, and DLinear Model.” MDPI, 17 6 2023, .
- [3] Kim, Taesung. “Missing Value Imputation of Time-Series Air-Quality Data via Deep Neural Networks.” MDPI,
- [4] A. Loganathan, P. Sumithra, and V. Deneshkumar, “Estimation of Air Quality Index Using Multiple Linear Regression.” Applied Ecology and Environmental Sciences, vol. 10, no. 12 (2022): 717-723. doi: 10.12691/aees-10-12-3.
- [5] Khaerun Nisa SH, Irfan Irfani, Utriweni Mukhaiyar, “Predicting Air Pollution Levels in Jakarta Using Vector Autoregressive Analysis”, Proceedings of the 5th International Conference on Statistics, Mathematics, Teaching, and Research 2023 (ICSMTR 2023)
- [6] Aarthi, P. Gayathri, N. R. Gomathi, et.al., (2020), "Air Quality Prediction Through Regression Model", International Journal of Scientific & Technology Research Vol 9, pp 923-928.
- [7] H. N. Mahendra , S. Mallikarjunaswamy, D. Mahesh Kumar, Shilpi Kumari, Shubhali Kashyap, Sapna Fulwani and Aishee Chatterjee* (2022) Assessment and Prediction of Air Quality Level Using ARIMA Model: A Case Study of Surat City, Gujarat State, India
- [8] Athira Va , Geetha Pb , Vinayakumar Rab, Soman K P (2018, DeepAir-Net: Applying Recurrent Networks for Air Quality Prediction
- [9] Xinxing Zhou et al 2019 J. Phys, Air Pollutant Concentration Prediction Based on GRU Method, <https://iopscience.iop.org/article/10.1088/1742-6596/1168/3/032058>
- [10] Zehua Du and Xin Lin 2020, Air Quality Prediction Based on Neural Network Model of Long Short-term Memory