DỰ BÁO MỨC ĐỘ CHỈ SỐ CHẤT LƯỢNG KHÔNG KHÍ TẠI BA THÀNH PHỐ CỦA VIỆT NAM SỬ DỤNG PHƯƠNG PHÁP HỌC MÁY VÀ HỌC SÂU

1st Cao Hoài Sang
Khoa Hệ Thống Thông Tin
TP.HCM, Việt Nam
21522541@gm.uit.edu.vn

2nd Thi Thành Công
Khoa Hệ Thống Thông Tin
TP.HCM, Việt Nam
21521897@gm.uit.edu.vn

***Tóm tắt nội dung*—Mục tiêu chính của nghiên cứu này là dự đoán chất lượng không khí tai ba thành phố được chỉ định (Hà Nội, Việt Trì và Đà Nẵng) của Việt Nam bằng cách kết hợp các thuật toán học máy và học sâu. Các mô hình bao gồm Gauss Newton Method Non-Linear, Residual Convolutional Neural Networks (ResCNN), NBEATS, Dynamic Linear Model (DLM), Simple Exponential Smoothing (SES), Linear Regression (LR), Autoregressive Integrated Moving Average (ARIMA), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Long Short Term Memory (LSTM). Hiệu quả của tất cả các mô hình được đề cập trên được đo lường bằng Mean Absolute Percentage Error (MAPE), Root Mean Squared Error, nhằm đạt được độ chính xác tối đa trong dự báo chuỗi thời gian chất lượng không khí chính xác.**

***Index Terms*—Nonlinear regression, Gauss-Newton, generalized least squares, iteratively reweighted least squares**

Lời cảm ơn

Nhóm xin gửi lời cảm ơn đến thầy Nguyễn Đình Thuân cùng giảng viên hướng dẫn Nguyễn Minh Nhực cho sự hướng dẫn tận tình, kiến thức bổ trợ và phản hồi giúp nhóm đạt được kết quả mong muốn. Đặc biệt cảm ơn các thành viên của nhóm nghiên cứu vì những nỗ lực siêng năng và đóng góp sáng tạo trong suốt dự án này.

I. Giới Thiệu

Với sự gia tăng dân số nhanh chóng cùng với sự công nghiệp hóa ngày càng mạnh mẽ ở các thành phố lớn Việt Nam, vấn đề chất lượng không khí trờ thành mối quan tâm lớn do có liên quan tới đời sống hàng ngày. Sự suy giảm chất lượng không chỉ gây ra những rủi ro lớn đối với sức khỏe của cư dân mà còn đe dọa sự cân bằng sinh thái của khu vực. Trong bối cảnh này, việc dự đoán chính xác và kịp thời các mức độ chất lượng không khí là rất cần thiết cho các chiến lược giảm thiểu hiệu quả. Nghiên cứu của nhóm sử dụng các mô hình học máy và học sâu để dự đoán chất lượng không khí, bằng cách tận dụng sức mạnh của các thuật toán phức tạp như Gauss Newton Method Non-Linear, Resilient Convolutional Neural Networks (ResCNN), NBEATS, Dynamic Linear Model (DLM), Simple Exponential Smoothing (SES), Linear Regression (LR), Autoregressive Integrated Moving Average (ARIMA), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), và Long Short Term Memory (LSTM).

II. Nghiên cứu liên quan

A. Gauss-Newton nonlinear method

Vào năm 2023 , Gauss-Newton được Zhifa Ke, Junyu Zhang và Zaiwen Wen ứng dụng để tối ưu hóa một biến thể của sai phân bình phương trung bình Bellman (MSBE). Cụ thể, Gauss-Newton được sử dụng trong mỗi vòng lặp của phương pháp học Tăng Cường Sai Phân Gauss-Newton (Gauss-Newton Temporal Difference - GNTD) để thực hiện các bước cập nhật tham số của mô hình xấp xỉ hàm phi tuyến. [1]. Hạn chế của phương pháp này là đòi hỏi tính toán gradient và ma trận Jacobi tại mỗi bước lặp, làm tăng tải tính toán và cần một lượng dữ liệu lớn để đạt độ chính xác mong muốn

B. Residual Convolutional Neural Network

Phân loại ECG bằng bộ Ensemble của Residual CNNs với Cơ chế Chú ý.

Trong phân loại ECG, sử dụng ResNet kết hợp với cơ chế chú ý đa đầu đã chứng minh hiệu quả. Giải pháp của Đội ISIBrno-AIMT trong Cuộc thi PhysioNet 2021 đã thể hiện sự vượt trội bằng việc phân loại ECG thành 26 nhóm khác nhau. Phương pháp này tích hợp các hàm mất mát và tối ưu hóa tiến hóa, mang lại đóng góp quan trọng cho lĩnh vực này. [4]

C. Dynamic linear models

Vào năm 2014, D. Osthus, P. C. Caragea, D. Higdon, S. K. Morley, G.D. Reeves, và B. P. Weaver đã sử dụng các mô hình tuyến tính động để dự báo electron trong vành đai bức xạ bằng cách so sánh độ chính xác dự báo trước 1 ngày của một mô hình tuyến tính động đơn giản với mô hình dự báo electron tương đối hiện tại (REFM) [5]. Hạn chế của nghiên cứu của họ là việc mô hình mang lại hiệu quả thấp đối với dữ liệu có tính biến thiên cao.

D. NBEATS

Năm 2021, một nhóm nghiên cứu tại Đại học Khoa học và Công nghệ Hà Nội và Đại học FPT đã áp dụng mô hình NBEATS để dự báo nhu cầu điện ngắn hạn ở Việt Nam [6]. Sau khi quan sát thấy mô hình hoạt động tốt với dữ liệu chuỗi thời gian trong các lĩnh vực khác nhau như công nghiệp, tài chính. Tuy nhiên, những phương pháp này có nhược điểm là thời gian chạy tương đối chậm, làm cho việc triển khai chúng trong các ứng dụng thực tế gặp khó khăn.

E. Recurrent Neural Network

Trong nghiên cứu về hiểu biết ô nhiễm không khí, Brian S. Freemana, Graham Taylora, Bahram Gharabaghia, và Jesse Thé từ Trường Kỹ thuật, Đại học Guelph, Guelph, Ontario, Canada dự báo chuỗi thời gian chất lượng không khí bằng cách sử dụng mô hình học sâu mạng nơ-ron hồi quy (RNN) với bộ nhớ dài hạn (LSTM) [7]. Nghiên cứu của nhóm đã chỉ ra được hạn chế lớn nhất của RNN chính là vấn đề Vanishing Gradient.

F. SES

Nhóm tác giả từ University of Karachi, Pakistan đã sử dụng mô hình ARIMA và SES để dự đoán lượng khí thải CO2 từ một số nước châu Á như Japan, Bangladesh, China, Pakistan, India, Sri Lanka, Iran, Singapore, và Nepal trong khoảng từ năm 1971 đến 2014. [8]

G. GRU

Nhóm tác giả Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag và Yan Liu đã cải tiến GRU dành cho đa biến ở bài toán thiếu dữ liệu cho Time Series [13]. Tuy nhiên, mô hình GRU-D gặp hạn chế trong việc đòi hỏi tài nguyên tính toán lớn, khó khăn trong việc điều chỉnh siêu tham số và khả năng tổng quát hóa chưa tốt khi dữ liệu bị thiếu không tuân theo các mô hình dự đoán được xác định sẵn

H. Linear Regression

Năm 2022, tại Đại học Manonmaniam Sundaranar, A. Loganathan, Sumithra Palraj, và Deneshkumar V đã công bố một nghiên cứu về việc sử dụng hồi quy tuyến tính để dự báo chỉ số Chất lượng không khí (AQI). Nghiên cứu tập trung vào phân tích dữ liệu AQI thu thập từ trạm giám sát ở Chennai, Ấn Độ, và xây dựng một mô hình hồi quy tuyến tính đa biến. Đánh giá về tính hợp lệ của mô hình được thực hiện thông qua phân tích dư thừa. [12]

I. LSTM

Ricardo Navares và José L. Aznarte đã sử dụng mô hình LSTM để dự đoán chất lượng không khí tại thành phố Marid. Nhóm tác giả dựa vào các chỉ số không khí gây hại như CO, NO2, O3, PM10, SO2 được thu thập tại các địa điểm khác nhau của Madrid. Sau đó sử dụng các cấu hình khác nhau của mô hình LSTM để dự đoán chất lượng không khí và so sánh cấu hình LSTM nào sẽ là tốt nhất. [9]

J. ARIMA

ARIMA đã được áp dụng để giải quyết các vấn đề thực tế trong thị trường chứng khoán bằng cách dự báo giá cổ phiếu của bốn công ty hàng đầu trong chỉ số Nifty Midcap-50 bằng MATLAB kèm theo các chỉ số hiệu suất [10] .Kết hợp mô hình hồi quy mờ và mô hình ARIMA, mô hình ARIMA mờ (FARIMA) đã được phát triển để dự đoán tỷ giá đồng Đài tệ sang Đô la Mỹ [11].

III. Tài nguyên

A. Bộ dữ liệu

Dự báo chất lượng không khí là rất quan trọng để giảm thiểu các tác động tiêu cực của ô nhiễm lên sức khỏe con người và môi trường. Phân tích chuỗi thời gian nổi lên như một công cụ mạnh mẽ trong lĩnh vực này, cho phép dự đoán các mức độ chất lượng không khí trong tương lai dựa trên dữ liệu lịch sử.

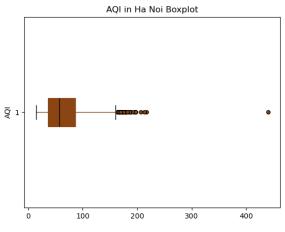
Do các vấn đề phổ biến liên quan đến bụi mịn ở khu vực phía Bắc, nhóm đã chọn một bộ dữ liệu chi tiết về chất lượng không khí của ba thành phố lớn ở Việt Nam: Đà Nẵng, Hà Nội và Việt Trì.

Bộ dữ liệu chủ yếu kéo dài từ năm 2019 đến năm hiện tại, 2024, bao gồm 6 cột tương ứng với các thành phần khác nhau trong không khí và chất lượng không khí được đánh giá thông qua Chỉ số Chất lượng Không khí (AQI): nồng độ PM2.5, nồng độ PM10, nồng độ O3, nồng độ NO2, nồng độ SO2 và nồng độ CO.

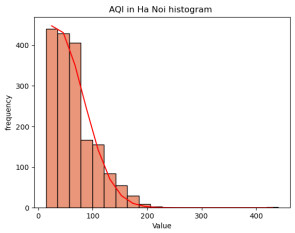
B. Thống kế mô tả

Bảng I
HaNoi, DaNang, VietTri's Descriptive Statistics

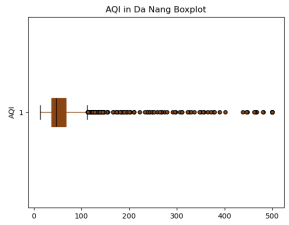
	HaNoi	DaNang	VietTri
Count	2840	3136	2348
Mean	75.351	78.721	60.160
Std	42.285	73.749	41.381
Min	11	13	15
25%	43.0	43.0	31.0
50%	66.0	59.0	47.0
75%	101.0	88.0	81.0
Max	498	500	828



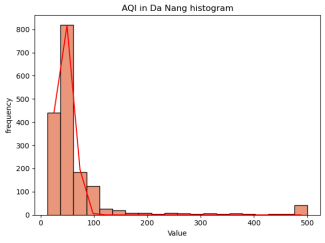
Hình 1. HaNoi AQI's boxplot



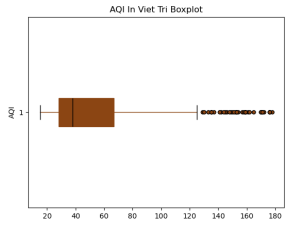
Hình 2. HaNoi AQI's histogram



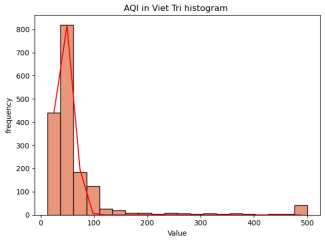
Hình 3. DaNang AQI's boxplot



Hình 4. DaNang AQI's histogram



Hình 5. VietTri AQI's boxplot



Hình 6. VietTri AQI's histogram

IV. Phương pháp luận

A. 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 các biến số. Kỹ thuật thống kê này được sử dụng để tìm một phương trình dự đoán tốt nhất cho biến y như một hàm tuyến tính của các biến x.

Một mô hình hồi quy tuyến tính đa biến có dạng:

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

Trong đó:

- Y là biến phụ thuộc (biến mục tiêu).
- $X_1, X_2, \ldots, X_k$ là các biến độc lập (biến giải thích).
- β_0 là hệ số chặn.
- $\beta_1, \ldots, \beta_k$ là các hệ số hồi quy cho các biến độc lập.
- ε là thuật ngữ lỗi.

B. Gauss newton method nonlinear

1) Phương pháp Newton

Với hàm $y = y_0 e^{-kt}$, chúng ta tìm giá trị nhỏ nhất của SSE. Chúng ta tìm giá trị k bằng phương pháp Newton.

$$SSE = \sum_i^n (y_i - y_0 e^{-kt_i})^2$$

$$k_{\text{new}} = k_{\text{old}} - \frac{f'(k_{\text{old}})}{f''(k_{\text{old}})}$$

Hoặc chúng ta có thể giải thích bằng ma trận Hessian như sau:

$$\begin{pmatrix} k_{\text{new}} \\ y_{0_{\text{new}}} \end{pmatrix} = \begin{pmatrix} k_{\text{old}} \\ y_{0_{\text{old}}} \end{pmatrix} - H^{-1} G$$

Trong đó: • $H = \begin{bmatrix} \frac{\partial^2 f}{\partial k^2} & \frac{\partial^2 f}{\partial k \partial y_0} \\ \frac{\partial^2 f}{\partial y_0 \partial k} & \frac{\partial^2 f}{\partial y_0^2} \end{bmatrix}$ • $G = \begin{bmatrix} \frac{\partial f}{\partial k} \\ \frac{\partial f}{\partial y_0} \end{bmatrix}$

Vấn đề với phương pháp Newton trong hồi quy phi tuyến là việc tính toán ma trận Hessian và nghịch đảo của nó gặp khó khăn. Để giải quyết vấn đề này, phương pháp Gauss-Newton thay thế bằng cách xấp xỉ ma trận Hessian.

Chúng ta có thể viết lại SSE như sau:

$$SSE = \sum_i^n r_i^2 = r^T r$$

Trong đó:

• r là một vector chứa các sai số dư

2) Phương pháp Gauss-Newton

Chúng ta lấy đạo hàm của SSE theo các tham số trong mô hình bằng quy tắc chuỗi, chúng ta có được phương trình sau:

$$\frac{\partial SSE}{\partial \beta_j} = 2 \sum_i^n r_i \frac{\partial r_i}{\partial \beta_j}$$

Sau đó, bỏ số hai vì nó sẽ không ảnh hưởng đến việc ước tính các tham số. Tương ứng với ma trận Jacobian.

$$J_r = \begin{bmatrix} \frac{\partial r_1}{\partial \beta_1} & \frac{\partial r_1}{\partial \beta_2} \\ \frac{\partial r_2}{\partial \beta_1} & \frac{\partial r_2}{\partial \beta_2} \\ \vdots & \vdots \\ \frac{\partial r_n}{\partial \beta_1} & \frac{\partial r_n}{\partial \beta_2} \end{bmatrix}$$

Với phương trình sau cho tổng các sai số bình phương (SSE):

$$SSE = \sum_{i=1}^{n} (y_i - y_0 e^{-kt_i})^2$$

Chúng ta có:

$$\frac{\partial^2 SSE}{\partial \beta_j \partial \beta_k} = \sum_i^n \left(\frac{\partial r_i}{\partial \beta_j} \frac{\partial r_i}{\partial \beta_k} + r_i \frac{\partial^2 r_i}{\partial \beta_j \partial \beta_k} \right)$$

Sự khác biệt chính giữa phương pháp Newton và Gauss-Newton là phương pháp Gauss-Newton bỏ qua $r_i \frac{\partial^2 r_i}{\partial \beta_j \partial \beta_k}$. Do đó, đạo hàm bậc hai được xấp xỉ bằng hàm sau:

$$\frac{\partial^2 SSE}{\partial \beta_j \partial \beta_k} \approx \sum_i^n \left(\frac{\partial r_i}{\partial \beta_j} \frac{\partial r_i}{\partial \beta_k} \right) = J_r^T J_r$$

Sử dụng quy tắc cập nhật sau trong phương pháp Newton. Đối với Gauss-Newton, đơn giản chỉ cần cắm vào xấp xỉ cho ma trận Hessian và gradient.

$$\begin{pmatrix} k_{\text{new}} \\ y_{0_{\text{new}}} \end{pmatrix} = \begin{pmatrix} k_{\text{old}} \\ y_{0_{\text{old}}} \end{pmatrix} - (J_r^T J_r)^{-1} J_r^T r$$

Với β là một vector cột với các tham số được ước tính. Đối với ví dụ đơn giản chỉ ước tính hai tham số, phương trình trông như sau:

$$\beta_{\text{new}} = \beta_{\text{old}} - (J_r^T J_r)^{-1} J_r^T r(\beta_{\text{old}})$$

$$\begin{pmatrix} k_{\text{new}} \\ y_{0_{\text{new}}} \end{pmatrix} = \begin{pmatrix} k_{\text{old}} \\ y_{0_{\text{old}}} \end{pmatrix} - (J_r^T J_r)^{-1} J_r^T r \begin{pmatrix} k_{\text{old}} \\ y_{0_{\text{old}}} \end{pmatrix}$$

C. Simple Exponential Smoothing (SES)

Simple exponential smoothing (SES) hay san bằng hàm mũ đơn giản là một phương pháp dự đoán dữ liệu chuỗi thời gian. SES dựa vào tổng trung bình trọng số của dữ liệu thực tế gần nhất và dữ liệu được dự đoán trước đó để dự đoán dữ liệu tiếp theo trong tương lai. SES phù hợp với dữ liệu chuỗi thời gian không có xu hướng và không có tính mùa vụ.. SES có thể viết ở hai dạng: dạng trung bình trọng số và dạng đối tượng.

Dạng trung bình trọng số:

$$\hat{y}_{t+1|t} = \alpha y_t + (1-\alpha)\hat{y}_{t|t-1}$$

Trong đó:

• $\hat{y}_{t+1|t}$: giá trị dự đoán ở thời điểm t+1 dựa vào giá trị thực tế ở thời điểm t.

• α: hệ số làm trơn $(0 \leq \alpha \leq 1)$.

• $\hat{y}_{t|t-1}$: giá trị dự đoán ở thời điểm t dựa vào giá trị thực tế ở thời điểm t-1.

Dạng đối tượng:

• Phương trình dự đoán: $\hat{y}_{t+h|t} = l_t$, h=1,2,3,...

• Phương trình làm trơn: $l_t = \alpha y_t + (1-\alpha) l_{t-1}$

Trong đó:

• $\hat{y}_{t+h|t}$: giá trị dự đoán tại thời điểm t+h dựa trên giá trị thực tế ở thời điểm t.

• α: hệ số làm trơn $(0 \leq \alpha \leq 1)$.

• l_t: cấp độ (giá trị đã được làm trơn) của chuỗi tại thời điểm t.

D. Autoregressive Integrated Moving Average (ARIMA)

ARIMA là viết tắt của "Autoregressive Integrated Moving Average". Mô hình ARIMA thường được sử dụng để dự báo dữ liệu chuỗi thời gian đơn biến. Mô hình ARIMA có thể xử lý một chuỗi thời gian nếu chuỗi đó là dừng và không có dữ liệu bị thiếu. Phương pháp này được sử dụng trong nhiều nghiên cứu để dự báo. ARIMA là sự kết hợp của 3 thành phần, Auto-Regressive – AR, Integrated – I và Moving Average – MA tương ứng với các tham số p, d và q đại diện cho ba thành phần chính của mô hình, trong đó:

• $p-AR(p)$: Tham số p đại diện cho số lượng các quá trình tự hồi quy trong thành phần tự hồi quy (AutoRegressive) của mô hình ARIMA. Nó chỉ ra số lượng ngày quá khứ của chuỗi dữ liệu mà được sử dụng để dự đoán giá trị hiện tại. Mỗi giá trị quá khứ được sử dụng là một hệ số trong mô hình tự hồi quy. Giá trị của p phụ thuộc vào sự phụ thuộc tạm thời trong chuỗi dữ liệu và có thể được xác định bằng cách sử dụng các phương pháp như đồ thị tự tương quan (ACF - AutoCorrelation Function) hoặc hàm tương quan một lệnh (PACF - Partial AutoCorrelation Function). Phương trình tự hồi quy AR được tổng quát như sau:

$$Y_t = c + \varphi_1 Y_{t-1} + \varphi_2 Y_{t-2} + \ldots + \varphi_p Y_{t-p} + \epsilon_t$$

Trong đó:

– Y_t đại diện cho giá trị dữ liệu tại thời điểm t
– c là hằng số chặn
– φ là hệ số AutoRegressive(AR)
– ϵ_t là sai số ngẫu nhiên
– p là số bậc

• $q-MA(q)$: Tham số q đại diện cho số lượng các thành phần trung bình động (Moving Average) trong mô hình ARIMA. Nó chỉ ra số lượng giá trị trung bình động được sử dụng để dự đoán giá trị hiện tại.

$$Y_t = c + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_p \epsilon_{t-p} + \epsilon_t$$

Trong đó:

– Y_t đại diện cho giá trị dữ liệu tại thời điểm t
– c là hằng số chặn
– θ là hệ số Moving Average(MA)
– ϵ_t là hệ số tương quan
– p là số bậc

• $d-I(d)$: Tham số d đại diện cho số lần lấy đạo hàm-sai phân (differencing) trên chuỗi dữ liệu ban đầu để loại bỏ xu hướng (trend) và/hoặc thành phần mùa vụ (seasonality) hay chuyển đổi dữ liệu thành chuỗi dừng. chuỗi dừng là chuỗi có trung bình, phương sai và tự tương quan không đổi theo thời gian. Một chuỗi thời gian được coi là chuỗi dừng nếu nó có trung bình không đổi, phương sai không đổi và tự tương quan không đổi. Chuỗi dừng là Công thức tính sai phân tại thời điểm t như sau

$$\Delta y_t = y_t - y_{t-1}$$

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

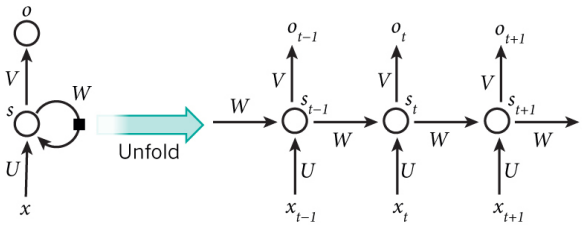
$$\Delta y_t = c + \varphi_1 \Delta y_{t-1} + \ldots + \varphi_p \Delta y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} \quad (1)$$

Trong đó:

– Y_t đại diện cho giá trị dữ liệu tại thời điểm t
– c là hằng số chặn
– θ là hệ số Moving Average(MA)
– ϵ_t là hệ số tương quan
– φ là hệ số AutoRegressive(AR)

E. Recurrent Neural Networks (RNN)

Ý tưởng chính của RNN (Recurrent Neural Network) là sử dụng chuỗi các thông tin. Trong các mạng nơ-ron truyền thống tất cả các đầu vào và cả đầu ra là độc lập với nhau. Tức là chúng không liên kết thành chuỗi với nhau. Nhưng các mô hình này không phù hợp trong rất nhiều bài toán. RNN được gọi là hồi quy (Recurrent) bởi thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng nhớ các thông tin được tính toán trước đó.



Hình 7. Một mạng lưới thần kinh tái diễn và sự diễn ra theo thời gian của quá trình tính toán liên quan đến tính toán chuyển tiếp.

Mô hình trên mô tả phép triển khai nội dung của một RNN. Triển khai ở đây có thể hiểu đơn giản là ta vẽ ra một mạng nơ-ron chuỗi tuần tự. Ví dụ ta có một câu gồm 5 chữ, thì mạng nơ-ron được triển khai sẽ gồm 5 tầng nơ-ron tương ứng với mỗi chữ một tầng. Lúc đó việc tính toán bên trong RNN được thực hiện như sau:

- x_t là đầu vào tại bước t. Ví dụ, x_t là một vec-tơ one-hot tương ứng với từ thứ 2 của câu.
- s_t là trạng thái ẩn tại bước t. Nó chính là bộ nhớ của mạng. s_t được tính toán dựa trên cả các trạng thái ẩn phía trước và đầu vào tại bước đó: $s_t = f(Ux_t + Ws_{t-1})$. Hàm f thường là một hàm phi tuyến tính như tang hyperbolic (tanh) hay ReLu.
- o_t là đầu ra tại bước t. Ví dụ, ta muốn dự đoán từ tiếp theo có thể xuất hiện trong câu thì o_t chính là một vec-tơ xác xuất các từ trong danh sách từ vựng của ta: $o_t = \text{softmax}(Vs_t)$

F. Residual Convolutional Neural Network (ResCNN)

1) Tổng quan

Mô hình ResCNN là một mô hình dự đoán chuỗi thời gian kết hợp giữa LSTM và CNN 1D sử dụng kết nối bỏ qua.

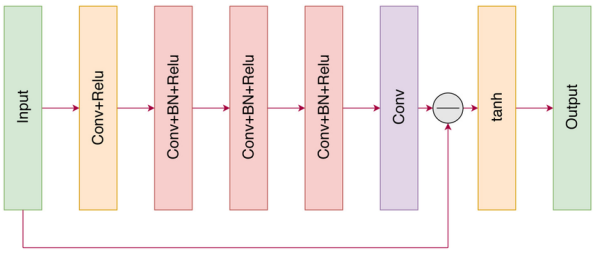
2) ResCNN với cách tiếp cận dựa trên độ dốc

a) Giới thiệu mô hình

Chúng tôi đề xuất một mô hình ResCNN với kết nối dư trên các lớp CNN 1D, giúp tránh mất thông tin quan trọng khi áp dụng nhiều lớp tích chập liên tiếp.

b) Thiết kế mô hình

Kiến trúc mô hình ResCNN được thể hiện trong ảnh sau



Hình 8. Mô hình ResCNN được đề xuất

c) Các thành phần chính

- **Input**: Dữ liệu chuỗi thời gian đầu vào.
- **LSTM**: Xử lý và ghi nhớ thông tin tuần tự, nắm bắt mối quan hệ dài hạn.
- **Conv + ReLU**:
  - **Conv**: Trích xuất đặc trưng không gian.
  - **ReLU**: Học đặc trưng phi tuyến tính.
- **Conv + BN + ReLU**:
  - **Conv**: Tiếp tục trích xuất đặc trưng.
  - **BN**: Chuẩn hóa theo batch, ổn định và tăng tốc huấn luyện.
  - **ReLU**: Học đặc trưng phi tuyến tính.
- **Conv**: Lớp tích chập cuối cùng không có hàm kích hoạt.
- **Skip Connection**: Bảo toàn thông tin từ đầu vào ban đầu.
- **Tanh**: Đưa đầu ra về khoảng giá trị [-1, 1].
- **Output**: Kết quả cuối cùng sau tất cả các lớp.

Tóm lại, mô hình này sử dụng các lớp tích chập để trích xuất đặc trưng phức tạp, ReLU và chuẩn hóa theo batch tăng tính phi tuyến và ổn định, kết nối tắt và tanh cải thiện hiệu quả học tập và chuẩn hóa đầu ra.

3) 1D-CNN

CNN 1D được sử dụng trong xử lý ngôn ngữ tự nhiên và phân tích chuỗi thời gian, trích xuất đặc trưng bằng cách di chuyển kernel dọc theo trục thời gian. Công thức tổng quát:

$$\text{out}(N_i, C_{out_j}, L_{out_l}) = \text{bias}(C_{out_j}) + \sum_{k=0}^{C_{in}-1} \text{weight}(C_{out_j}, k) * \text{input}(N_i, k, L_{l-k}) \quad (2)$$

4) Lớp Chuẩn hóa Theo Batch

Chuẩn hóa đầu ra của lớp tích chập để cải thiện huấn luyện:

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

5) Lớp kích hoạt cuối cùng

Hàm kích hoạt `tanh` biến đổi đầu ra thành khoảng $(-1, 1)$:

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

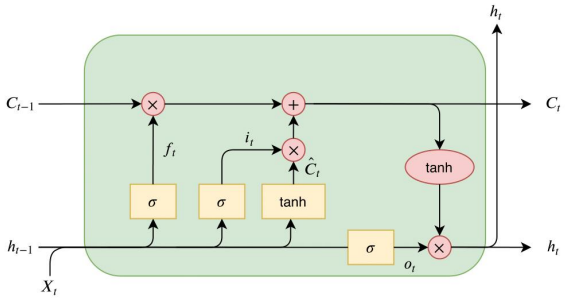
6) Kết nối còn lại

Kết nối còn lại thêm đầu vào ban đầu vào đầu ra của lớp tích chập cuối cùng:

$$\textbf{out}_{res}(x) = x + \text{Conv}(x)$$

G. Long Short Term Memory (LSTM)

Mạng bộ nhớ dài-ngắn (Long Short Term Memory), thường được gọi là LSTM - là một dạng cải tiến của RNN, nó có khả năng học được các phụ thuộc xa .LSTM gồm trạng thái ẩn h_t (hidden state) và c_t (cell state). LSTM gồm 3 thành phần chính:



Hình 9. Một ô LSTM ở trạng thái t.

Cổng quên(Forget gate):Quản lý việc thông tin đến từ trạng thái trước sẽ được giữ lại hoặc loại bỏ.

$$f_t = \sigma(W_{fx}.x_t + W_{fh}.h_{t-1} + b_f)$$

Cổng vào (Input gate): Quản lý việc học thông tin mới từ đầu vào.

$$i_t = \sigma(W_{ix}.x_t + W_{ih}.h_{t-1} + b_i)$$

$$\widetilde{C}_t = tanh\left(W_{cx}.x_t + W_{ch}.h_{t-1} + b_c\right)$$

$$C_t = f_t \circ c_{t-1} + i_t \circ \widetilde{C}_t$$

Cổng ra (Output gate): Quản lý việc đưa thông tin mới cập nhật vào trạng thái kế tiếp.

$$o_t = \sigma(W_{ox}.x_t + W_{oh}.h_{t-1} + b_o)$$

$$h_t = o_t \circ tanh(C_t)$$

Trong đó:

- x_t : Vector đầu vào các đặc trưng tại thời điểm t.
- h_{t-1} : Trạng thái ẩn đầu ra của tại thời điểm t-1.
- $W_{fx}, W_{fh}, W_{ix}, W_{ih}, W_{cx}, W_{hx}$, : Các ma trận trọng số.
- b_f, b_i, b_c : Các hệ số bias.
- Tanh: Hàm tanh có giá trị tử -1 đến 1.
- i_t : Giá trị cổng đầu vào tại thời điểm t.
- σ : Hàm Sigmoid có giá trị tử 0 đến 1.
- f_t : Giá trị của cổng quên tại thời điểm t.
- $\widetilde{C}_t$: Ô nhớ mới.
- C_t : Ô nhớ tổng thể.
- $\circ$: Phép nhân Hadamard.
- $f_t \circ c_{t-1}$: Quyết định lượng thông tin nào từ trạng thái ô nhớ trước đó sẽ được giữ lại cho ô tiếp theo.
- $i_t \circ \widetilde{C}_t$: Quyết định lượng thông tin nào từ $\widetilde{C}_t$ sẽ được thêm vào C_t.
- o_t : Giá trị đầu ra tại thời điểm t.

H. Gated Recurrent Unit (GRU)

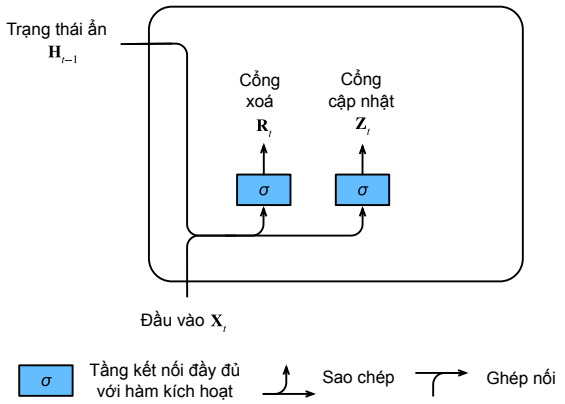
1) GRU là gì?

GRU là một biến thể của LSTM. Không có ô nhớ riêng biệt, thay vào đó, nó sử dụng một "trạng thái ẩn tiềm năng" và hai cổng (reset gate và update gate).

- **Cổng thiết lập lại (reset gate)**: Quyết định bao nhiêu trạng thái ẩn trước đó cần quên.
- **Cổng cập nhật (update gate)**: Quyết định bao nhiêu của vector kích hoạt ứng viên cần tích hợp vào trạng thái ẩn mới.

2) Cổng thiết lập lại và cổng cập nhật

Đầu tiên ta giới thiệu thiết lập lại và cổng cập nhật. Ta thiết kế chúng thành các vector có các phần tử trong khoảng (0,1) để có thể biểu diễn các tổ hợp lồi. Chẳng hạn, một biến xóa cho phép kiểm soát bao nhiêu phần của trạng thái trước đây được giữ lại. Tương tự, một biến cập nhật cho phép kiểm soát bao nhiêu phần của trạng thái mới sẽ giống trạng thái cũ.



Hình 10. Cổng thiết lập lại và cổng cập nhật trong GRU

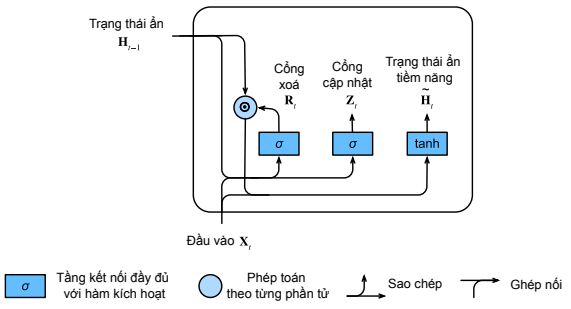
Tại bước thời gian t, với đầu vào là $\mathbf{X}_t \in \mathbb{R}^{n\times d}$ (số lượng mẫu: n, số lượng đầu vào: d) và trạng thái ẩn ở bước thời gian gần nhất là $\mathbf{H}_{t-1} \in \mathbb{R}^{n\times h}$ (số lượng trạng thái ẩn: h), cổng thiết lập lại $\mathbf{R}_t \in \mathbb{R}^{n\times h}$ và cổng cập nhật $\mathbf{Z}_t \in \mathbb{R}^{n\times h}$ được tính như sau:

$$\mathbf{R}_t = \sigma(\mathbf{X}_t\mathbf{W}_{xr} + \mathbf{H}_{t-1}\mathbf{W}_{hr} + \mathbf{b}_r)$$
$$\mathbf{Z}_t = \sigma(\mathbf{X}_t\mathbf{W}_{xz} + \mathbf{H}_{t-1}\mathbf{W}_{hz} + \mathbf{b}_z).$$

Ở đây, $\mathbf{W}_{xr}, \mathbf{W}_{xz} \in \mathbb{R}^{d\times h}$ và $\mathbf{W}_{hr}, \mathbf{W}_{hz} \in \mathbb{R}^{h\times h}$ là các ma trận trọng số và $\mathbf{b}_r, \mathbf{b}_z \in \mathbb{R}^{1\times h}$ là các hệ số điều chỉnh. Ta sẽ sử dụng hàm sigmoid để biến đổi các giá trị đầu vào nằm trong khoảng $(0, 1)$.

3) Tính toán trạng thái ẩn mới

Hình 11 minh họa luồng tính toán sau khi áp dụng cổng thiết lập lại. Ký hiệu $\odot$ biểu thị phép nhân theo từng phần tử giữa các tensor.



Hình 11. Tính toán trạng thái tiềm năng trong một GRU. Phép nhân được thực hiện theo phần tử

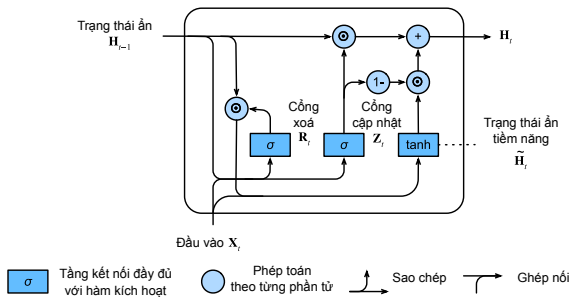
Ta có công thức để tính trạng thái ẩn tiềm năng:

$$\tilde{\mathbf{H}}_t = \tanh(\mathbf{X}_t\mathbf{W}_{xh} + (\mathbf{R}_t \odot \mathbf{H}_{t-1})\mathbf{W}_{hh} + \mathbf{b}_h).$$

4) Tính toán trạng thái ẩn cuối cùng

Xác định mức độ giống nhau giữa trạng thái mới $\mathbf{H}_t$ và trạng thái cũ $\mathbf{H}_{t-1}$ cũng như mức độ trạng thái tiềm năng $\widetilde{\mathbf{H}}_t$ được sử dụng. Cổng cập nhật $\mathbf{Z}_t$ được sử dụng cho mục đích này bằng cách áp dụng tổ hợp lồi giữa trạng thái cũ và trạng thái tiềm năng. Ta có phương trình cập nhật cuối cho GRU.

$$\mathbf{H}_t = \mathbf{Z}_t \circ \mathbf{H}_{t-1} + (1 - \mathbf{Z}_t) \circ \tilde{\mathbf{H}}_t.$$



Hình 12. Tính toán trạng thái ẩn cuối cùng trong GRU.

I. Dynamic Linear Model (DLM)

1) Tổng quan:

Dynamic Linear Model là một trường hợp đặt biệt của mô hình trạng thái không gian - mô hình biểu hiện trạng thái của một hệ thống. Model mô tả hành động của một hệ thống động bằng cách sử dụng một bộ biến, được gọi là biến trạng thái để miêu tả trạng thái hiện tại của hệ thống thay đổi theo thời gian.

Đây là một mô hình hồi quy tuyến tính cơ bản:

$$y_t = \alpha + \beta x_i + e_i \;\text{ với }\; e_i \sim N(0, \sigma^2)$$

Từ đây ta có thể viết lại dưới dạng ma trận:

$$y_t = \begin{bmatrix}1 & x_i\end{bmatrix} \begin{bmatrix}\alpha \\ \beta\end{bmatrix} + e_i \;\Rightarrow\; y_t = \mathrm{X}_i^T\theta + e_i$$

Với $\mathrm{X}_i^T = \begin{bmatrix}1 & x_i\end{bmatrix}$ và $\theta = \begin{bmatrix}\alpha & \beta\end{bmatrix}^T$.

Và vì trong một mô hình tuyến tính động, tham số hồi quy thay đổi liên tục nên ta viết lại:

$$y_t = \mathrm{X}_i^T\theta + e_i \;\text{ (tĩnh) }\Rightarrow\; y_t = \mathrm{X}_i^T\theta_t + e_i \;\text{ (động)}$$

Từ đó chúng ta có thể nhận thấy chỉ số t thể hiện thông tin về thời gian thứ tự của các nút dữ liệu trong y. => Mối quan hệ giữa y và X là riêng biệt với mỗi t khác nhau.

Tuy nhiên, quan sát kỹ hơn, ta có một vấn đề lớn cho việc ước lượng tham số của $y_t = \mathrm{X}_i^T\theta_t + e_i$. Nhận thấy chỉ có 1 điểm dữ liệu cho mỗi bước thời gian (tức là y_t là vô hướng) => mô hình liên hệ dữ liệu được quan sát với các biến trạng thái ẩn, nhưng không nắm bắt được động lực thời gian của hệ thống. Do đó, mô hình sẽ ước tính một tập hợp tham số mô tả mối quan hệ giữa dữ liệu được quan sát và các biến trạng thái tại một thời điểm cố định.

Để có thể giải quyết vấn đề này, ta có thể thêm một phương trình để giới hạn lại tham số hồi quy phụ thuộc vào khoảng thời gian từ t tới t + 1:

$$\theta_t = G_t\theta_{t-1} + w_t \;\text{ với }\; w_t \sim N(0, W_t)$$

Với G_t là ma trận chuyển trạng thái tại thời điểm t và w_t là nhiễu đại diện cho vector nhiễu trạng thái tại thời điểm t.

Kết hợp cả hai lại ta có được một mô hình tuyến tính động cơ bản (Dynamic Linear Model) trong dạng không gian trạng thái bao gồm hai mô hình:

• Mô hình quan sât liên hệ giữa các biến số và dữ liệu.

$$y_t = \mathrm{X}_i^T\theta_t + e_i$$

• Mô hình trạng thái xác định tham số "tiến hóa"theo thời gian.

$$\theta_t = G_t\theta_{t-1} + w_t$$

2) Ma trận trạng thái G:

Ma trận G, hay ma trận chuyển trạng thái, trong mô hình tuyến tính động được thiết kế để phản ánh tham số thay đổi như thế nào theo thời gian. Cấu trúc của ma trận G tùy thuộc vào thành phần có trong mô hình, như xu hướng, tính mùa màng, etc...

a) Thành phần xu hướng và ảnh hưởng mùa:

Mới một mô hình có xu hướng tuyến tính và tính mùa vụ với giai đoạn S:

• Mức độ (μ): Đại diện cho mức độ cơ sở của chuỗi thời gian.

• Đô dốc (β_t): Đại diện cho tốc độ thay đổi (xu hướng).

• Tính thời vụ (γ): Đại diện cho tính thời vụ.

$$\mu_t = \mu_{t-1} + \beta_{t-1} + \omega_{\mu,t} \;\text{ với }\; \omega_{\mu,t} \sim N(0, W_\mu)$$

$$\beta_t = \beta_{t-1} + \omega_{\beta,t} \;\text{ với }\; \omega_{\beta,t} \sim N(0, W_\beta)$$

$$\gamma_t = \gamma_{t-S} + u_t \;\text{ với }\; u_t \sim N(0, W_\gamma)$$

b) Xây dựng ma trận G:

Để có thể có được ma trận chuyển trạng thái G, trước tiên cần có vector trạng thái θ_t tại thời điểm t, ví dụ ta có thể xây dựng vector trạng thái với S = 4 như sau:

$$\theta_t = \begin{bmatrix}\mu_t \\ \beta_t \\ \gamma_t \\ \gamma_{t-1} \\ \gamma_{t-2} \\ \gamma_{t-3}\end{bmatrix}$$

Sau đó ma trận chuyển trạng thái sẽ 'mô tả' từng thành phần trong θ_t thay đổi theo thời gian. Cụ thể, thành phần xu hướng (mức độ và độ dốc):

• Mức độ μ_t tại thời điểm t phụ thuộc vào mức độ trước μ_{t-1} và độ dốc trước β_{t-1}.

• Độ dốc β_t tại thời điểm t chỉ phụ thuộc vào mức độ trước β_{t-1}.

Thành phần mùa: Tính thời vụ γ_t phụ thuộc vào tính mùa vụ tại thời điểm t-S từ S giai đoạn trước. Trong ví dụ chọn S = 4 $(\gamma_t, \gamma_{t-1}, \gamma_{t-2}, \gamma_{t-3})$ dại diện cho 4 quý của một năm, là mô hình sẽ thay đổi trạng thái mùa thông qua ma trận vòng.

Với những quan hệ ràng buộc trên, ma trận G có thể được xây dựng như sau:

$$G = \begin{bmatrix}1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0\end{bmatrix}$$

- Hàng đầu tiên: Mức độ mới μ_t là mức độ trước μ_{t-1} cộng với độ dốc trước β_{t-1} phản ánh xu hướng tuyến tính.
- Hàng thứ hai: Độ dốc mới β_t là giống với độ dốc trước β_{t-1}, chỉ ra sự không thay đổi của độ dốc.

- Hàng thứ ba tới hàng thứ sáu: Những hàng này thay đổi trạng thái mùa. Mỗi trạng thái mùa và phiên bản thay đổi của trạng thái mùa trước: γ_t đặt thành γ_{t-1} -> γ_{t-1} đặt thành γ_{t-2} -> γ_{t-2} đặt thành γ_{t-3} -> γ_{t-3} quay trở lại đầu chu kỳ (sang năm mới) nên đặt thành γ_t.

J. NBEATS

1) Tổng quan

Mô hình NBEATS, viết tắt của "The Neural Basis Expansion Analysis for Time Series forecasting"được phát triển bởi nhóm nghiên cứu từ Element AI và được công bố trong bài báo năm 2019 bởi Boris Oreshkin và cộng sự.

2) Backcast và Forecast

Trong NBEATS, mỗi khối (block) tạo ra hai đầu ra: backcast và forecast. Backcast tái tạo lại dữ liệu đầu vào để mô hình có thể học từ phần dư sau khi trừ đi phần tái tạo này, giúp khối tiếp theo xử lý thông tin còn lại. Forecast là đầu ra dự báo các giá trị tương lai, góp phần vào kết quả cuối cùng.

3) Dữ liệu đầu vào

NBEATS làm việc với chuỗi thời gian đơn biến. Đầu vào là một chuỗi thời gian có độ dài t:

$$x = [y_{T-t+1}, y_{T-t+2}, \ldots, y_T]$$

Mục tiêu là dự báo giá trị cho H bước tiếp theo:

$$y = [y_{T+1}, y_{T+2}, \ldots, y_{T+H}]$$

Khoảng thời gian xem lại (lookback period) thường có độ dài $t = nH$ với n từ 2 đến 7.

4) Kiến trúc tổng quát

Mô hình NBEATS gồm nhiều block xếp chồng lên nhau. Đầu vào của block đầu tiên là dữ liệu x. Mỗi block tạo ra hai đầu ra: backcast làm đầu vào cho block kế tiếp, và forecast dùng để tổng hợp dự đoán cuối cùng. Đầu vào của block thứ i (với $i > 1$) là:

$$x_i = x_{i-1} - \hat{x}_{i-1}$$

Dự đoán cuối cùng là tổng hợp các đầu ra forecast của tất cả các block:

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

5) Kiến trúc của block

Mỗi block gồm các lớp kết nối đầy đủ (fully connected, FC) với hàm kích hoạt ReLU:

$$FC(x_i) = ReLU(Wx_i + b)$$

Trong đó, W và b là các tham số được học. Mỗi block có M lớp FC chung, sau đó có hai lớp FC để chia đầu ra thành hai nhánh với các hệ số θ_f và θ_b. Các hệ số này được ánh xạ thành đầu ra backcast và forecast qua các hàm g_f và g_b.

6) Các loại block

- **Generic Block**: Sử dụng các ánh xạ tuyến tính.
- **Trend Block**: Phân tích xu hướng dữ liệu bằng các hàm đa thức bậc thấp:

$$\hat{y}_i = \sum_{j=0}^{p} \theta_{f,j} t^j$$

- **Seasonality Block**: Nhận diện tính mùa vụ qua các hàm tuần hoàn, sử dụng chuỗi Fourier:

$$\hat{y}_i = \sum_{j=0}^{H/2-1} \theta_{f,j} \cos(2\pi jt) + \theta_{f,j+H/2} \sin(2\pi jt)$$

V. Kết quả

A. Phương pháp đánh giá và kết quả mô hình

Mean Percentage Absolute Error (MAPE): là một phép đo đánh giá mức độ sai lệch tương đối giữa các giá trị dự đoán và giá trị thực tế trong dự đoán

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

Mean Absolute Error (MAE): được tính bằng trung bình của sai số tuyệt đối giữa giá trị thực tế và giá trị dự đoán trong tập dữ liệu.

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

Root Mean Square Error (RMSE): cho biết mức độ chênh lệch trung bình giữa giá trị dự đoán của mô hình và giá trị thực tế.

$$RMSE = \sqrt{\frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{N}}$$

Bảng II
BẢNG ĐÁNH GIÁ ĐỘ ĐO TRÊN DATASET VIỆT TRÌ

Model	Train:Test	RMSE	MAPE (%)	MAE
		Value	Value	Value
GaussNewtonNonlinear	7:3	56.59	143.99	49.95
	8:2	**25.19**	**39.11**	**19.29**
	9:1	28.01	76.27	25.23
ResCNN	7:3	24.544	43.435	16.32
	8:2	22.256	37.291	17.102
	9:1	**15.148**	**22.727**	**10.148**
GRU	7:3	56.16	2161.69	52.97
	8:2	56.05	1614.02	52.32
	9:1	**52.88**	**1439.80**	**51.55**
ARIMA	7:3	36.37	**31,39**	21,76
	8:2	**27,71**	42,95	**21,41**
	9:1	34,08	104,89	32,95
NBEATS	7:3	20.64	**26.02**	**14.78**
	8:2	19.76	43.13	15.99
	9:1	**19.36**	37.76	15.18
Linear Regression	7:3	47.146	70.275	36.808
	8:2	43.092	64.671	35.84
	9.1	**23.933**	**30.738**	**16.738**
DLM	7:3	50.22	72.03	40.87
	8:2	31.83	63.62	26.06
	9:1	**24.98**	**41.56**	**19.80**
RNN	7:3	22.20	29.52	13.57
	8:2	**17.66**	**22.83**	**11.66**
	9:1	18.66	29.27	13.90
SES	7:3	29.72	36.02	18.90
	8:2	29.26	66.71	66.71
	9:1	**18.00**	**31.29**	**13.89**
LSTM	7:3	22.59	27.09	12.78
	8:2	15.86	**21.36**	**10.36**
	9:1	**14.05**	22.64	10.53

Bảng III
BẢNG ĐÁNH GIÁ ĐỘ ĐO TRÊN DATASET HÀ NỘI

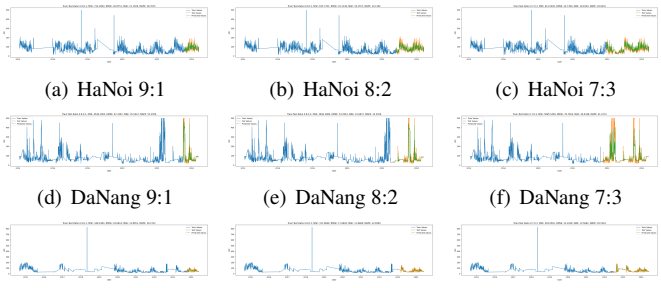
Model	Train:Test	RMSE	MAPE (%)	MAE
		Value	Value	Value
GaussNewtonNonlinear	7:3	99.75	106.15	83.29
	8:2	50.95	40.74	38.15
	9:1	**51.96**	**76.15**	**46.45**
ResCNN	7:3	28.739	31.412	**20.543**
	8:2	32.038	30.35	24.295
	9:1	**28.71**	**26.15**	21.505
ARIMA	7:3	49.32	**43.87**	36.28
	8:2	54.44	77.00	48.37
	9:1	**31.09**	49.10	**26.47**
DLM	7:3	61.12	114.34	52.26
	8:2	**42.51**	**38.64**	**32.04**
	9:1	76.20	69.34	61.13
GRU	7:3	81.26	746.69	76.35
	8:2	88.87	**627.15**	86.02
	9:1	**84.89**	642.77	**82.38**
NBEATS	7:3	30.38	26.82	22.84
	8:2	**27.64**	**25.62**	**21.20**
	9:1	31.69	33.08	26.12
RNN	7:3	24.92	29.35	10.09
	8:2	**25.73**	**24.47**	**19.43**
	9:1	27.63	29.48	22.03
SES	7:3	57.67	49.73	44.02
	8:2	68.84	61.06	57.16
	9:1	**32.02**	**26.83**	**22.77**
Linear Regression	7:3	64.77	59.609	51.019
	8:2	65.904	58.715	54.566
	9:1	**40.55**	**33.03**	**30.33**
LSTM	7:3	26.56	31.35	19.76
	8:2	27.85	**27.55**	21.43
	9:1	**25.24**	26.79	**19.22**

Bảng IV
BẢNG ĐÁNH GIÁ ĐỘ ĐO TRÊN DATASET ĐÀ NẴNG

Model	Train:Test	RMSE	MAPE (%)	MAE
		Value	Value	Value
GaussNewtonNonlinear	7:3	145.73	**104.77**	**88.44**
	8:2	**133.19**	190.12	105.14
	9:1	136.55	149.32	93.19
ResCNN	7:3	**82,208**	**62,294**	**41,028**
	8:2	92,096	74,695	51,225
	9:1	93,917	68,218	47,323
GRU	7:3	117.04	**1094.12**	85.75
	8:2	**102.37**	1340.91	**77.82**
	9:1	127.76	1388.00	96.50
ARIMA	7:3	154.76	313.9	142.54
	8:2	**146.74**	**50.15**	**66.27**
	9:1	194.28	71.58	105.03
NBEATS	7:3	79.46	**68.11**	**43.31**
	8:2	101.91	155.01	74.18
	9:1	**65.87**	101.96	44.18
LinearRegression	7:3	**120.89**	**42,574**	**54,318**
	8:2	131,599	47,929	60,224
	9:1	140,505	60,559	66,297
DLM	7:3	**122.86**	**40.93**	**55.68**
	8:2	204.85	300.90	160.82
	9:1	154.70	69.37	78.50
RNN	7:3	76.78	41.23	36.03
	8:2	**53.3**	**32.39**	**22.66**
	9:1	67.44	52.2	35.54
SES	7:3	**121.30**	**38.11**	**53.38**
	8:2	124.75	81.16	64.94
	9:1	140.65	58.48	65.63
LSTM	7:3	**87.20**	**51.02**	**39.57**
	8:2	100.42	83.03	48.04
	9:1	114.13	79.01	60.05

B. Kết quả chạy 2 mô hình RNN và LSTM với các chỉ số tốt nhất cho 3 bộ dữ liệu HaNoi, DaNang và VietTri

2 hình ảnh dưới đây lần lượt thể hiện kết quả chạy mô hình RNN và mô hình LSTM cho 3 bộ dữ liệu sử dụng.



(a) HaNoi 9:1 (b) HaNoi 8:2 (c) HaNoi 7:3

(d) DaNang 9:1 (e) DaNang 8:2 (f) DaNang 7:3

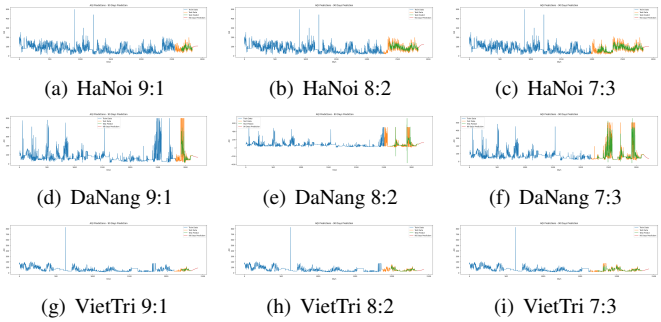
(g) VietTri 9:1 (h) VietTri 8:2 (i) VietTri 7:3

Hình 13. Kết quả chạy mô hình RNN trên 3 dataset HaNoi, DaNang và VietTri

C. Dự đoán giá trị AQI trong 30, 60 và 90 ngày kế tiếp của 3 bộ dữ liệu HaNoi, DaNang và VietTri

Để thực hiện dự đoán, chúng tôi đã sử dụng hai mô hình học máy tiên tiến, được lựa chọn dựa trên hiệu suất cao trong các thử nghiệm ban đầu. Cả hai mô hình này đã được huấn luyện và kiểm tra trên ba bộ dữ liệu khác nhau tương ứng với ba thành phố để đảm bảo tính tổng quát và độ chính xác của dự đoán.

2 mô hình được nói đến ở đây lần lượt là LSTM và RNN có kết quả dự đoán 30, 60 và 90 ngày trên 3 bộ dữ liệu, với tỉ lệ



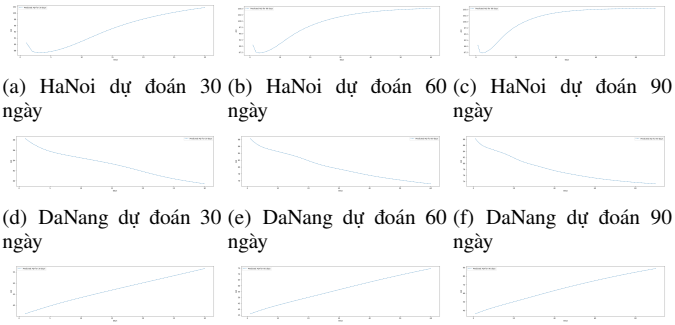
(a) HaNoi 9:1 (b) HaNoi 8:2 (c) HaNoi 7:3

(d) DaNang 9:1 (e) DaNang 8:2 (f) DaNang 7:3

(g) VietTri 9:1 (h) VietTri 8:2 (i) VietTri 7:3

Hình 14. Kết quả chạy mô hình LSTM trên 3 bộ dữ liệu HaNoi, DaNang và VietTri

chia tập huấn luyện và tập kiểm tra là 9:1 được thể hiện trong 2 ảnh sau:

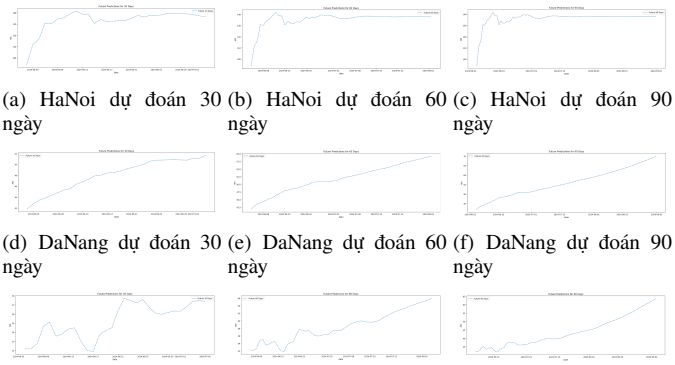


(a) HaNoi dự đoán 30 ngày (b) HaNoi dự đoán 60 ngày (c) HaNoi dự đoán 90 ngày

(d) DaNang dự đoán 30 ngày (e) DaNang dự đoán 60 ngày (f) DaNang dự đoán 90 ngày

(g) VietTri dự đoán 30 ngày (h) VietTri dự đoán 60 ngày (i) VietTri dự đoán 90 ngày

Hình 15. Kết quả chạy mô hình LSTM trên 3 bộ dữ liệu HaNoi, DaNang và VietTri với tỉ lệ tập huấn luyện và kiểm tra 9:1



(a) HaNoi dự đoán 30 ngày (b) HaNoi dự đoán 60 ngày (c) HaNoi dự đoán 90 ngày

(d) DaNang dự đoán 30 ngày (e) DaNang dự đoán 60 ngày (f) DaNang dự đoán 90 ngày

(g) VietTri dự đoán 30 ngày (h) VietTri dự đoán 60 ngày (i) VietTri dự đoán 90 ngày

Hình 16. Kết quả chạy mô hình RNN trên 3 bộ dữ liệu HaNoi, DaNang và VietTri với tỉ lệ tập huấn luyện và kiểm tra 9:1

VI. Tổng kết

A. Tổng quan

Phân tích cho thấy các mô hình mạng nơ-ron tiên tiến như ResCNN, NBEATS, RNN, GRU và LSTM cho kết quả vượt trội hơn các mô hình thống kê truyền thống như Gauss Newton Non-Linear, LR và ARIMA về độ chính xác và độ tin cậy. Các mô hình học sâu cho hiệu suất vượt trội nhờ khả năng nắm bắt các phức tạp và các mối quan hệ phi tuyến tính vốn có trong dữ liệu chất lượng không khí. Trong số này, mô hình LSTM và RNN cung cấp các dự báo nhất quán và đáng tin cậy nhất trên cả ba thành phố.

B. Định hướng phát triển tương lai

• Thứ nhất, việc kết hợp, chọn lọc các nguồn dữ liệu bổ sung như dữ liệu khí tượng, lưu lượng giao thông và hoạt động công nghiệp có thể cải thiện độ chính xác của mô hình.

• Thứ hai, tìm hiểu các mô hình kết hợp các điểm mạnh khác nhau để mang lại hiệu suất dự báo tốt hơn là sử dụng riêng lẽ để cho từng dự báo, đặc biệt rõ ràng ở việc sử dụng RNN + LSTM.

Tài liệu

[1] Ke, Z., Zhang, J., & Wen, Z. (2023). Gauss-Newton Temporal Difference Learning with Nonlinear Function Approximation.

[2] H. Choi, C. Jung, T. Kang, H. J. Kim, and I. -Y. Kwak, "Explainable time-series prediction using a residual network and gradient-based methods,"in IEEE Access, vol. 10, pp. 108469-108482, 2022.

[3] Doan Vo Duy Thanh, Nguyen Van Cuong, Vo Tan Phat, Le Khac Hong Phuc, Nguyen Duy Du, Nguyen Van Quang, Pham Minh Duc, Pham Hong Vinh, and Nguyen Canh Thuong. *Gated Recurrent Unit (GRU)*. Accessed May 26, 2024. https://d2l.aivivn.com/chapter_recurrent-modern/gru_vn.html.

[4] Nejedly P, Ivora A, Viscor I, Koscova Z, Smisek R, Jurak P, Plesinger F. Classification of ECG using ensemble of residual CNNs with or without attention mechanism. Physiol Meas. 2022 Apr 28;43(4). doi: 10.1088/1361-6579/ac647c. PMID: 35381586.

[5] D. Osthus, P. C. Caragea, D. Higdon, S. K. Morley, G.D.Reeves and B. P. Weaver, "Dynamic linear models for forecasting of radiation beltelectrons and limitations on physical interpretationof predictive models,"vol. 12, no. 6, pp. 323-446, 2014.

[6] Nguyen Anh Tuan, Le Anh Ngoc . "APPLYING N-BEATS MODEL FOR SHORT-TERM LOAD FORECASTING IN VIETNAM" (2022). URL: https://khcncongthuong.vn/tin-tuc/t18804/ung-dung-mo-hinh-n-beats-cho-du-bao-phu-tai-dien-ngan-han-o-viet-nam.html

[7] B. S, Freemana, G. Taylora, B. Gharabaghia and J. Thé, "Forecasting air quality time series using deep learning,"Journal of the air & waste management association, vol. 68, no. 8, pp. 866-886, 2017-2018.

[8] Fatima, Samreen & Saad, Sayed & Zia, Syeda & Hussain, Ehtesham & Fraz, Tayyab & Shafi, Mehwish & Khan,. (2019). Forecasting Carbon Dioxide Emission of Asian Countries Using ARIMA and Simple Exponential Smoothing Models. International Journal of Economic and Environment Geology. 10. 10.46660/ojs.v10i1.219.

[9] Navares, Ricardo & Aznarte, José. (2019). Predicting air quality with deep learning LSTM: Towards comprehensive models. Ecological Informatics. 55. 101019. 10.1016/j.ecoinf.2019.101019.

[10] Devi, B. Uma, D. Sundar, and P. Alli. "An Effective Time Series Analysis for Stock Trend Prediction Using ARIMA Model for Nifty Midcap-50."

[11] Tseng, Fang-Mei, et al. "Fuzzy ARIMA model for forecasting the foreign exchange market."Fuzzy sets and systems 118.1 (2001): 9-19

[12] Loganathan, A. & Palraj, Sumithra & V, Deneshkumar. (2022). Estimation of Air Quality Index Using Multiple Linear Regression. Applied Ecology and Environmental Sciences. 10. 717-722. 10.12691/aees-10-12-3.

[13] Che, Zhengping, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. "Recurrent Neural Networks for Multivariate Time Series with Missing Values."arXiv preprint arXiv:1606.01865 (2016). URL: https://doi.org/10.48550/arXiv.1606.01865.