

Dự báo chất lượng không khí tại Việt Nam bằng phương pháp chuỗi thời gian

1st Huỳnh Lê Phong

Khoa Hệ thống thông tin

Trường Đại học Công nghệ Thông tin

21520086@gm.uit.edu.vn

2nd Nguyễn Quốc Trạng

Khoa Hệ thống thông tin

Trường Đại học Công nghệ Thông tin

21521556@gm.uit.edu.vn

3rd Nguyễn Triệu Vy

Khoa Hệ thống thông tin

Trường Đại học Công nghệ Thông tin

21522812@gm.uit.edu.vn

4th Vũ Thị Phương Linh

Khoa Hệ thống thông tin

Trường Đại học Công nghệ Thông tin

20521541@gm.uit.edu.vn

5th Đỗ Đình Đăng Khoa

Khoa Hệ thống thông tin

Trường Đại học Công nghệ Thông tin

21522218@gm.uit.edu.vn

Tóm tắt nội dung—Nghiên cứu này tập trung vào phân tích chất lượng không khí ở ba thành phố lớn tại Việt Nam: Hà Nội, Hạ Long và Việt Trì. Mục tiêu của nghiên cứu là sử dụng các kỹ thuật phân tích chuỗi thời gian tiên tiến để dự báo và hiểu chỉ số PM2.5. Phương pháp phân tích bao gồm cả các thuật toán truyền thống và tiên tiến như Hồi quy Tuyển tính, Mô hình ARIMA, Mạng Nơ-ron Hồi quy (RNN), Đơn vị Hồi quy Cổng (GRU), LSTM, VAR, Rừng Ngẫu nhiên, Mô hình Mã hóa Dài Chuỗi Thời gian (TiDE), Autoformer, Mạng Nơ-ron Đa Tầng (MLP). Nhóm chúng em đã sử dụng ba tỷ lệ khác nhau giữa các tập dữ liệu train:test, cụ thể là 7:3, 8:2 và 9:1, để đánh giá hiệu suất của các mô hình dưới các điều kiện khác nhau. Sau đó so sánh hiệu suất của các mô hình khác nhau dựa trên ba chỉ số: Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), và Mean Absolute Error (MAE). Cuối cùng, hai mô hình có hiệu suất tốt nhất được sử dụng để dự báo chất lượng không khí cho 90 ngày tiếp theo, thể hiện hiệu quả của chúng trong việc dự báo chất lượng không khí. Bằng cách sử dụng các tập dữ liệu thời gian thực, nghiên cứu này nhằm dự báo, khám phá các mẫu, xu hướng và mối quan hệ giữa các chỉ số chất lượng không khí. Ngoài ra, nghiên cứu cũng nhằm đóng góp vào việc hiểu rõ hơn về việc giải quyết các vấn đề ô nhiễm không khí, thúc đẩy môi trường đô thị khỏe mạnh và bền vững hơn.

Index Terms—Chất lượng không khí, ô nhiễm không khí, phân tích chuỗi thời gian, dự báo, Hồi quy Tuyển tính, ARIMA, RNN, GRU, LSTM, VAR, Random Forest, TiDE, Autoformer, MLP, Hà Nội, Hạ Long, Việt Trì, Việt Nam.

I. GIỚI THIỆU

Trong những thập kỷ gần đây, sự chú ý toàn cầu ngày càng tập trung vào suy thoái môi trường, đặc biệt là ô nhiễm không khí, do tác động sâu rộng của nó đối với sức khỏe con người, hệ sinh thái và sự ổn định của khí hậu. Hai phần mìn (PM2.5 và PM10) cùng với các khí như ozone (O3), dioxide nitơ (NO2), dioxide lưu huỳnh (SO2), và carbon monoxide (CO) đóng vai trò quan trọng trong sự suy giảm chất lượng không khí. Trong bối cảnh này, nghiên cứu của Nhóm nhằm khám phá động lực chất lượng không khí trong môi trường đô thị, đặc biệt là ở ba thành phố lớn của Việt Nam: Hà Nội, Hạ Long và Việt Trì. Mỗi thành phố đại diện cho một ngã cảnh kinh tế-xã hội và địa lý

khác biệt, mang lại cái nhìn sâu sắc về các thách thức và cơ hội của quản lý ô nhiễm không khí. Mục tiêu tổng thể là sử dụng các kỹ thuật phân tích chuỗi thời gian tiên tiến để dự báo và phân tích các chỉ số về chất lượng không khí. Nhóm đã sử dụng một loạt các thuật toán, từ các phương pháp truyền thống như Hồi quy Tuyển tính và Mô hình Trung bình Chuyển động Tích hợp Tự động (ARIMA) đến các kiến trúc học sâu tiên tiến như Mạng Nơ-ron Hồi quy (RNN), Đơn vị Hồi quy Cổng (GRU), Bộ nhớ Ngắn hạn - Dài hạn (LSTM), Hồi quy Vector (VAR), Rừng Ngẫu nhiên, Mô hình Mã hóa Dài Chuỗi Thời gian (TiDE), Autoformer, và Mạng Nơ-ron Đa Tầng (MLP). Bằng cách tận dụng những kỹ thuật này nhằm mục đích khám phá các mẫu, xu hướng và dự báo dữ liệu chất lượng không khí trong tương lai. Các tập dữ liệu sử dụng bao gồm các số liệu đo thời gian thực về các chỉ số chất lượng không khí chính, bao gồm PM2.5, PM10, O3, NO2, SO2 và CO, được thu thập trong một khoảng thời gian dài. Những tập dữ liệu này cho phép khám phá các biến thể thời gian, sự đa dạng không gian và tương tác phức tạp giữa các chất gây ô nhiễm. Các kết quả của nhóm đóng góp vào sự hiểu biết rộng lớn hơn về việc giải quyết các thách thức đa mặt do ô nhiễm không khí gây ra, thúc đẩy môi trường đô thị khỏe mạnh và bền vững hơn. Ngoài ra, điều quan trọng là nhấn mạnh về vấn đề ô nhiễm không khí cấp bách mà Việt Nam đang phải đối mặt, đặc biệt là ở các thành phố lớn như Hà Nội, nơi chất lượng không khí đã nằm trong số tồi tệ nhất ở Đông Nam Á. Các yếu tố như mức độ cao của PM2.5 và các chất gây ô nhiễm khác đã gây ra những lo ngại sức khỏe đáng kể và tác động tiêu cực đến GDP của đất nước.

II. NGHIÊN CỨU LIÊN QUAN

Trong những năm gần đây, đã có một sự gia tăng đáng kể trong nghiên cứu nhằm dự báo chất lượng không khí bằng cách sử dụng một loạt các kỹ thuật máy học, học sâu và thống kê.

Evgenny Marinov, Dessislava Petrova-Antanova và Simeon Malinov, trong một nghiên cứu [1], tập trung vào việc cải thiện độ chính xác của việc dự báo chất lượng không khí thông qua việc áp dụng phương pháp ARIMA. Nghiên cứu của họ, dựa

trên dữ liệu từ các trạm giám sát chất lượng không khí tại Thành phố Sofia, Bulgaria, từ ngày 1 tháng 1 năm 2015 đến ngày 31 tháng 12 năm 2019, đã cho thấy hiệu quả của các mô hình ARIMA trong việc dự báo nồng độ ô nhiễm như CO, NO₂, O₃ và PM2.5.

S. H. Khaerun Nisa, Irfan Irfani và Utriweni Mukhaiyar, trong nghiên cứu của họ [2], đã khám phá việc dự báo mức độ ô nhiễm không khí tại Jakarta bằng phương pháp phân tích Vector Autoregressive (VAR). Bằng cách phân tích dữ liệu chuỗi thời gian về chỉ số chất lượng không khí (AQI) và nồng độ PM2.5 thu thập từ ngày 16 tháng 8 đến ngày 25 tháng 9 năm 2023, để huấn luyện, và từ ngày 25 tháng 9 đến ngày 1 tháng 10 năm 2023, để kiểm tra, họ đã xác định mô hình VAR(2) tối ưu cho việc dự báo ô nhiễm không khí chính xác.

Khawaja Hassan Waseem và cộng sự đã khám phá tác động của các yếu tố khí hậu đối với nồng độ PM2.5 và phát triển mô hình dự báo trong một nghiên cứu gần đây [3]. Nghiên cứu của họ, kéo dài trong 30 ngày và 72 giờ cho các dự đoán hàng ngày và hàng giờ tương ứng, đã kết hợp dữ liệu chất lượng không khí, chất gây ô nhiễm và điều kiện khí hậu từ nhiều thành phố ở Pakistan. Sử dụng các mô hình máy học và học sâu bao gồm FbProphet và LSTM, họ đã phát hiện mô hình mã hóa-giải mã LSTM vượt trội so với các mô hình khác, đạt được cải thiện đáng kể về độ chính xác dự báo.

Hai tác giả, Marwa Winis Misbah Esager và Kamil Demirberk Ünlü [4], đã áp dụng mô hình LSTM (Long Short-Term Memory) để dự báo nồng độ hàng giờ của hạt phấn mịn PM2.5 tại Tripoli, Libya. Họ đã sử dụng 100 epochs để huấn luyện mô hình, và kết quả tốt nhất đã được đạt được khi số nút được đặt là 20. Kết quả RMSE trên tập kiểm tra cho thấy mức độ lỗi thấp, khoảng 0.0146, chứng tỏ mô hình dự báo có độ chính xác khá cao.

Ngoài ra, các nghiên cứu gần đây đã sử dụng các kỹ thuật mô hình hóa tiên tiến để cải thiện dự báo chất lượng không khí. Ví dụ, Abhimanyu Das và đồng nghiệp đã giới thiệu TiDE (Time-series Dense Encoder) [5], một mô hình mới tùy chỉnh cho dự báo chuỗi thời gian dài hạn. TiDE cho thấy khả năng xử lý các biến đổi phi tuyến tính trong dữ liệu chuỗi thời gian và vượt trội hoặc tương đương với các phương pháp hiện tại trong các chỉ số dự báo dài hạn, đồng thời có tốc độ suy luận và huấn luyện nhanh hơn đáng kể.

Ngược lại, nghiên cứu trước đó của Ong và đồng nghiệp [6] đã khám phá việc sử dụng các phương pháp dựa trên RNN để dự báo mức độ chất lượng không khí, trình bày một kỹ thuật động để tiền huấn luyện mô hình tập trung vào dự báo chuỗi thời gian nhiều bước trước. Tương tự, Lim và đồng nghiệp [7] đã sử dụng RNN để dự báo các chất gây ô nhiễm không khí khác nhau nhưng không tìm thấy sự khác biệt hoặc cải thiện đáng kể so với các mô hình truyền thống.

Haixu Wu, Jiehui Xu, Jianmin Wang và Mingsheng Long đã phát triển Autoformer [8], một mô hình dự báo chuỗi thời gian dài hạn, nhằm giải quyết các thách thức trong dự báo thời tiết cực đoan và lập kế hoạch tiêu thụ năng lượng. Autoformer vượt trội so với các mô hình Transformer trước đó nhờ tích hợp các khối Phân rã và Tương quan Tự động dựa trên tính chu kỳ của chuỗi thời gian. Thử nghiệm trên sáu bài toán khác nhau cho

thấy Autoformer đạt độ chính xác hàng đầu, cải thiện 38% so với các phương pháp hiện tại.

III. TÀI NGUYÊN

A. Bộ dữ liệu

Báo cáo sử dụng 3 bộ dữ liệu lấy từ dữ liệu chất lượng không khí trên trang web aqicn.org từ 01/03/2019 - 01/06/2024 ở 3 thành phố Hà Nội, Hạ Long và Việt Trì. Bộ dữ liệu bao gồm các thuộc tính cụ thể sau:

Bảng I
MÔ TẢ THUỘC TÍNH

Thuộc tính	Mô tả
Ngày	Thời gian (YYYY-MM-DD)
Pm25	Bụi mịn có đường kính từ 2.5 đến 10 micron
Pm10	Bụi mịn có đường kính nhỏ hơn 2.5 micron
O3	Ozon
NO2	Dioxít nitơ
SO2	Dioxít lưu huỳnh
CO	Carbon monoxit

Bảng II
THỐNG KÊ MÔ TẢ CHO HÀ LONG, HÀ NỘI VÀ VIỆT TRÌ

Thống kê	Hà Long	Hà Nội	Việt Trì
Count	1920	1920	1920
Mean	40.08	63.09	42.39
Std Dev	22.95	40.26	31.66
Min	5	2	1
Q1	22	31	19
Q2 (Median)	38	54.5	35
Q3	54	88	59
Max	163	217	178
Mode	5	24	1
Variance	527.01	1620.88	1002.69
Kurtosis	0.41	0.334	1.92
Skewness	0.685	0.902	1.24
CV	0.572	0.638	0.74

Hà Nội có giá trị trung bình và mức độ biến động cao nhất trong ba thành phố. Việt Trì có độ nhọn và độ lệch cao nhất, cho thấy dữ liệu phân bố lệch và biến động lớn. Hạ Long có dữ liệu ổn định nhất với giá trị trung bình thấp nhất.

B. Công cụ

Trong quá trình nghiên cứu và phân tích dữ liệu, Nhóm đã tận dụng một loạt các công cụ phân tích thống kê trong Python để khám phá sâu hơn các mẫu dữ liệu và rút ra những kết luận có ý nghĩa. Các công cụ chính bao gồm: Darts, numpy, pandas, sklearn, matplotlib.pyplot,... Sử dụng các công cụ phân tích thống kê này đã giúp Nhóm em hiểu sâu hơn về dữ liệu và đưa ra dự báo. Chi tiết các kết quả có thể được tìm thấy trong bảng mô tả và biểu đồ đi kèm.

C. Tỷ lệ phân chia dữ liệu

Nhóm đã chia tập dữ liệu chuỗi thời gian thành hai phần với các tỷ lệ khác nhau: 70% huấn luyện và 30% kiểm tra, 80% huấn luyện và 20% kiểm tra, và 90% huấn luyện và 10% kiểm tra. Tỷ lệ phổ biến nhất là 70:30, tạo sự cân bằng giữa dữ liệu huấn luyện và kiểm tra. Tỷ lệ 80:20 thích hợp cho các mô

hình phức tạp, trong khi 90:10 phù hợp khi xử lý tập dữ liệu lớn và mô hình đơn giản, đảm bảo đủ dữ liệu huấn luyện và kiểm tra. Trong việc đánh giá hiệu suất của các mô hình, Nhóm em sử dụng ba chỉ số là Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), và Root Mean Squared Error (RMSE). Thuật toán có giá trị thấp nhất trong ba chỉ số này sẽ cho thấy mức độ chính xác tốt nhất. Dưới đây là các công thức để tính MAE, MAPE và RMSE.

IV. PHƯƠNG PHÁP

A. LINEAR REGRESSION

Hồi quy tuyến tính là một kỹ thuật quan trọng trong thống kê, cho phép chúng ta khám phá và mô hình hóa mối quan hệ giữa các biến. Giúp chúng ta hiểu rõ hơn về cách các yếu tố độc lập ảnh hưởng đến một biến phụ thuộc. Trong hồi quy tuyến tính đa biến, chúng ta có thể xem xét tác động của nhiều biến độc lập đến một biến phụ thuộc, giúp tạo ra các dự đoán hoặc giải thích phức tạp hơn về thực tế. Một mô hình hồi quy tuyến tính đa biến có dạng:

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

Trong đó:

- Y : Biến phụ thuộc.
- X_1, X_2, \dots, X_k : Các biến độc lập.
- β_0 : Hệ số chặn.
- $\beta_1, \beta_2, \dots, \beta_k$: Các hệ số hồi quy.
- ϵ : Thành phần sai số.

B. ARIMA

Phương pháp ARIMA (Autoregressive Integrated Moving Average) sử dụng một mô hình AR (Autoregressive) kết hợp với một mô hình MA (Moving Average) để thực hiện dự báo chuỗi thời gian. Các tham số chính cần xem xét bao gồm:

- Số lượng quan sát trước đó (p);
- Độ chênh lệch (d);
- Kích thước của trung bình chuyển động (q).

Mô hình AR hiển thị sự phụ thuộc của một quan sát vào một giai đoạn thời gian trước đó. Mô hình AR thu được p quan sát trước đó như sau:

$$y_t = \alpha + \sum_{i=1}^p \phi_i y_{t-i} + e_t$$

trong đó y_t là biến dự đoán cho thời điểm t từ phân phối chuẩn và y_{t-i} xác định p quan sát trước của cùng một chuỗi thời gian. ϕ_i biểu thị các hệ số hồi quy, α là một hằng số và e_t là thuật ngữ lỗi ngẫu nhiên. Thứ tự p cho mô hình AR(p) được lựa chọn dựa trên các đỉnh quan trọng của PACF (Partial Autocorrelation Function). Một chỉ báo bổ sung là sự giảm chậm chạp của ACF (Autocorrelation Function).

MA thực hiện dự báo dựa trên các trung bình chuyển động của các thuật ngữ lỗi ngẫu nhiên trước đó như sau:

$$y_t = \mu + \sum_{i=1}^q \theta_i e_{t-i}$$

trong đó θ_t đại diện cho các hệ số hồi quy, q là thứ tự của trung bình chuyển động, và μ là một hằng số. Thứ tự q cho mô hình MA(q) được lấy từ ACF, nếu nó có một đoạn cắt sắc sau lags q . PACF giảm chậm trong trường hợp này.

Mô hình ARIMA có thể được xác định như sau:

$$\phi_p(B)(1 - B)^d y_t = \theta_q(B)e_t$$

trong đó B là toán tử backshift, p là thứ tự tự hồi quy, d là thứ tự chênh lệch và q là thứ tự của trung bình chuyển động.

C. VAR

Mô hình Vector Autoregression (VAR) là một mô hình thống kê dùng trong phân tích chuỗi thời gian để nắm bắt mối quan hệ động giữa nhiều biến số chuỗi thời gian. Đây là sự mở rộng tự nhiên của mô hình autoregressive đơn biến sang chuỗi thời gian đa biến động. Mô hình VAR có tính linh hoạt cao trong dự báo, cho phép dự báo dựa trên các biến số cụ thể trong mô hình.

Giả sử $\mathbf{Y}_t = (y_{1t}, y_{2t}, \dots, y_{nt})'$ biểu thị một vector kích thước $(n \times 1)$ chứa các biến chuỗi thời gian tại thời điểm t . Mô hình Vector Autoregression bậc p cơ bản VAR(p) có dạng như sau:

$$\mathbf{Y}_t = c + \boldsymbol{\Pi}_1 \mathbf{Y}_{t-1} + \boldsymbol{\Pi}_2 \mathbf{Y}_{t-2} + \dots + \boldsymbol{\Pi}_p \mathbf{Y}_{t-p} + \epsilon_t, \quad t = 1, \dots, T$$

Trong đó:

- \mathbf{Y}_t : vector của các biến nội sinh tại thời điểm t , kích thước $(n \times 1)$.
- $\boldsymbol{\Pi}_i$: các ma trận hệ số kích thước $(n \times n)$.
- $\mathbf{Y}_{t-1}, \mathbf{Y}_{t-2}, \dots, \mathbf{Y}_{t-p}$: các giá trị trễ của biến nội sinh.
- p : độ trễ biểu thị các giá trị trước đó của các biến nội sinh.
- ϵ_t : vector kích thước $(n \times 1)$ của các thành phần sai số, được giả định là nhiễu trắng.

Ví dụ:

- Giả sử có 2 biến nội sinh là pm25 và pm10 thì $\mathbf{Y}_t = (pm25_t, pm10_t)'$.

$$\mathbf{Y}_t = \begin{pmatrix} pm25_t \\ pm10_t \end{pmatrix}$$

- Ma trận chứa các hệ số cho các biến trễ của các biến nội sinh:

$$\boldsymbol{\Pi}_1 = \begin{pmatrix} 0.5 & 0.1 \\ 0.2 & 0.3 \end{pmatrix}$$

- Với độ trễ $p = 1$, phương trình VAR(1) có dạng:

$$\begin{pmatrix} pm25_t \\ pm10_t \end{pmatrix} = c + \begin{pmatrix} 0.5 & 0.1 \\ 0.2 & 0.3 \end{pmatrix} \begin{pmatrix} pm25_{t-1} \\ pm10_{t-1} \end{pmatrix} + \epsilon_t$$

D. Random Forest

Định nghĩa

Random Forest là phương pháp xây dựng một tập hợp cây quyết định cho các bài toán phân lớp và hồi quy. Cây quyết định thường có độ lệch thấp và phương sai cao, và Random Forest cải thiện hiệu suất bằng cách lấy trung bình các cây.

Giả sử $(X_t, Y_t)_{t \in \mathbb{Z}}$ là một dãy các biến ngẫu nhiên, trong đó:

$$Y_t = f(X_t) + \epsilon_t$$

với $\mathbb{E}[\epsilon_t | X_t] = 0$. Mục tiêu là ước lượng hàm hồi quy

$$f(x) = \mathbb{E}[Y_t | X_t = x]$$

Block Bootstrap cho Chuỗi Thời Gian:

- Moving Block Bootstrap:** Chia dữ liệu thành các khối chồng lấn kích thước cố định, lấy mẫu lại để tạo chuỗi mới.
- Circular Block Bootstrap:** Bao bọc chuỗi thời gian theo vòng tròn, đảm bảo điểm dữ liệu đều đại diện.

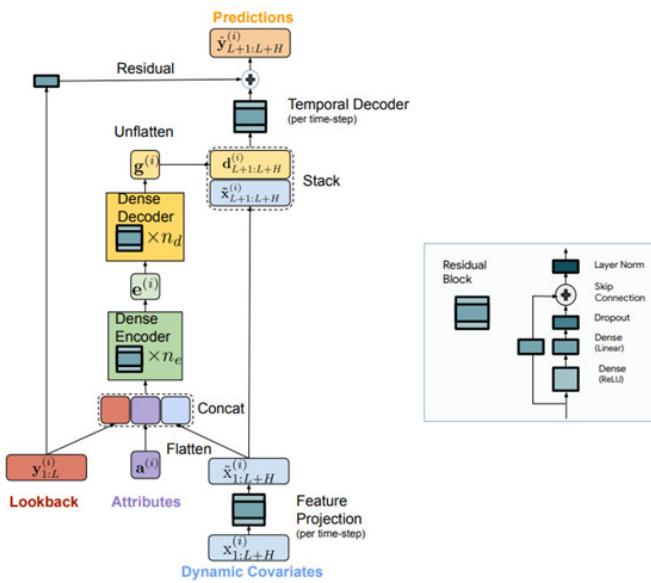
Thuật Toán:

- Đầu Vào:** Dữ liệu huấn luyện $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$, các tham số $M, \alpha_n, m_{try}, \tau_n, l_n$.
- Xây Dựng Cây cho $j = 1$ đến M :**
 - Lấy α_n quan sát bằng block bootstrap với tham số l_n .
 - Chia nút chọn chia cắt tốt nhất từ tập con ngẫu nhiên của m_{try} đặc trưng.
- Đầu Ra:** Dự đoán cho một quan sát mới x là trung bình của dự đoán từ M cây:

$$\hat{f}(x) = \frac{1}{M} \sum_{j=1}^M \hat{f}_j(x)$$

E. Time-series Dense Encoder

TiDE (Time-series Dense Encoder) là một mô hình dự báo chuỗi thời gian. Mô hình mã hóa chuỗi thời gian quá khứ cùng với hiệp phương sai bằng cách sử dụng “dense MLP” (Multi-Layer Perceptron). Sau đó, mô hình giải mã (decode) chuỗi thời gian được mã hóa (encode) cùng với các hiệp phương sai trong tương lai.



Hình 1. Tổng quan về kiến trúc TiDE

Kiến trúc tổng quan được trình bày ở Hình 4. Đầu vào (input) của mô hình là dữ liệu quá khứ và phương sai của một chuỗi thời gian tại một thời điểm $(y_{1:L}^{(i)}, x_{1:L}^{(i)}, a^{(i)})$ và ánh xạ tới dự đoán của chuỗi thời gian $\hat{y}_{L+1:L+H}^{(i)}$.

Residual Block: Là một thành phần quan trọng của kiến trúc TiDE vì nó cho phép mô hình nắm bắt các tính chất phi tuyến tính vốn có trong dữ liệu chuỗi thời gian, đồng thời duy trì các mối quan hệ tuyến tính nhằm giúp cải thiện hiệu suất dự báo dài hạn.

Giả sử đầu vào của Residual Block là x , công thức của nó có thể được mô tả như sau:

- Layer Normalization:

$$h_0 = \text{LayerNorm}(x)$$

- Lớp Dense đầu tiên với ReLU Activation:

$$h_1 = \text{ReLU}(\mathbf{W}_1 h_0 + \mathbf{b}_1)$$

- Dropout (tùy chọn):

$$h_1 = \text{Dropout}(h_1, p)$$

Trong đó, p là xác suất giữ lại của Dropout.

- Lớp Dense thứ hai:

$$h_2 = \mathbf{W}_2 h_1 + \mathbf{b}_2$$

- ReLU Activation sau Kết nối tắt (Skip Connection):

$$\tilde{x} = \text{ReLU}(x + h_2)$$

Trong đó:

- \mathbf{W}_1 và \mathbf{W}_2 là các ma trận trọng số của các lớp Dense MLP.
- \mathbf{b}_1 và \mathbf{b}_2 là các vector bù.
- $\text{LayerNorm}(\cdot)$ là hàm chuẩn hóa lớp.
- $\text{Dropout}(\cdot, p)$ là hàm Dropout với xác suất p .
- $\text{ReLU}(\cdot)$ là hàm kích hoạt ReLU.
- \tilde{x} là đầu ra của Residual Block.

Mã hóa (Encoding):

- Feature Projection:** Sử dụng residual block để ánh xạ $x_t^{(i)}$ tại mỗi time-step, hoạt động này được mô tả như sau:

$$\tilde{x}_t^{(i)} = \text{ResidualBlock}(x_t^{(i)})$$

- Dense Encoder:** Xếp chồng và làm phẳng tất cả các biến động trong quá khứ và tương lai, nối chúng với các thuộc tính tĩnh và quá khứ của chuỗi thời gian. Ánh xạ vào một embedding bằng cách sử dụng một bộ mã hóa chứa nhiều khối dữ:

$$e^{(i)} = \text{Encoder} \left(y_{1:L}^{(i)}, \tilde{x}_{1:L}^{(i)}, \tilde{x}_{1:L+H}^{(i)}, a^{(i)} \right)$$

Giải mã (Decoding):

- Dense Decoder (Bộ giải mã dense):** Đơn vị giải mã đầu tiên xếp chồng nhiều khối dữ giống bộ mã hóa với cùng kích thước lớp ẩn, nhận đầu vào là mã hóa $e^{(i)}$, ánh xạ thành vector $g^{(i)}$ kích thước $H \times p$ (decoderOutputDim), và chuyển đổi thành ma trận $D^{(i)} \in \mathbb{R}^{p \times H}$:

$$g^{(i)} = \text{Decoder}(e^{(i)}) \in \mathbb{R}^{p \times H}$$

$$D^{(i)} = \text{Reshape}(g^{(i)}) \in \mathbb{R}^{p \times H}$$

- **Temporal Decoder (Bộ giải mã thời gian):** Cuối cùng, bộ giải mã thời gian, một khối dư với đầu ra kích thước 1, ánh xạ vector $d_t^{(i)}$ tại thời điểm t kết hợp với các phương sai đã được chiếu $\hat{x}_{L+t}^{(i)}$:

$$\hat{y}_{L+t}^{(i)} = \text{TemporalDecoder}(d_t^{(i)}, \hat{x}_{L+t}^{(i)}) \quad \forall t \in [H]$$

F. RNN

Mạng nơ-ron hồi tiếp (RNN) là một loại mạng có khả năng xử lý dữ liệu tuần tự. Điểm đặc biệt của RNN là các nơ-ron trong mạng có thể kết nối với nhau theo chu kỳ. Nhờ vậy, RNN có thể học được mối quan hệ giữa các giá trị trong một chuỗi thời gian.

Công thức tính trạng thái hiện tại:

$$h_t = f(h_{t-1}, x_t)$$

trong đó:

- h_t : trạng thái hiện tại
- h_{t-1} : trạng thái trước đó
- x_t : trạng thái đầu vào

Công thức áp dụng hàm kích hoạt (tanh) và Công thức tính đầu ra:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

trong đó:

- W_{hh} : trọng số tại nơ-ron hồi quy
- W_{xh} : trọng số tại nơ-ron đầu vào
- y_t : đầu ra
- W_{hy} : trọng số tại lớp đầu ra

G. GRU

Trong lĩnh vực học máy và đặc biệt là xử lý ngôn ngữ tự nhiên (NLP), mạng nơ-ron hồi tiếp (Recurrent Neural Networks - RNNs) đóng vai trò quan trọng. Tuy nhiên, RNN truyền thống gặp phải vấn đề về vanishing gradient, làm giảm hiệu quả học của mạng khi xử lý các chuỗi dữ liệu dài. Để giải quyết vấn đề này, các kiến trúc RNN cải tiến như Long Short-Term Memory (LSTM) và Gated Recurrent Unit (GRU) đã được đề xuất. Bài viết này tập trung vào Gated Recurrent Unit (GRU), một biến thể đơn giản hơn của LSTM nhưng vẫn rất hiệu quả trong việc duy trì thông tin dài hạn.

Điểm nổi bật của GRU là sự kết hợp giữa các cơ chế cập nhật và quên trong một đơn vị duy nhất, giúp giảm thiểu số lượng tham số cần thiết và tăng tốc độ tính toán mà vẫn duy trì hiệu quả cao.

Một GRU bao gồm hai cổng chính: cổng cập nhật (update gate) và cổng đặt lại (reset gate). Cả hai cổng này cùng hoạt động để điều chỉnh thông tin nào cần được cập nhật và thông tin nào cần được quên.

Cho x_t là đầu vào tại thời điểm t , h_t là trạng thái ẩn tại thời điểm t , và h_{t-1} là trạng thái ẩn từ bước trước đó, các công thức của GRU được định nghĩa như sau:

1) Cổng Cập Nhật

Cổng cập nhật z_t quyết định phần nào của trạng thái ẩn trước đó cần được giữ lại và phần nào cần được cập nhật. Khi giá trị của z_t gần bằng 1, trạng thái ẩn hiện tại (h_t) sẽ gần giống với trạng thái ẩn trước đó (h_{t-1}), nghĩa là thông tin mới từ \tilde{h}_t sẽ bị bỏ qua. Khi giá trị của z_t gần bằng 0, trạng thái ẩn hiện tại (h_t) sẽ gần giống với trạng thái tạm thời (\tilde{h}_t), nghĩa là thông tin mới từ đầu vào hiện tại được cập nhật hoàn toàn vào trạng thái ẩn.

$$z_t = \sigma(W_z x_t + U_z h_{t-1})$$

Trong đó: z_t là giá trị của cổng cập nhật tại thời điểm t , σ là hàm kích hoạt sigmoid, W_z là ma trận trọng số dành cho cổng cập nhật, h_{t-1} là trạng thái ẩn tại thời điểm $t-1$, x_t là đầu vào tại thời điểm t .

2) Cổng Đặt Lại

Cổng đặt lại r_t xác định lượng trạng thái ẩn trước đó cần được quên. Nó lấy đầu vào là trạng thái ẩn trước đó và đầu vào hiện tại, đồng thời tạo ra một vectơ số từ 0 đến 1 để kiểm soát mức độ mà trạng thái ẩn trước đó được “đặt lại” ở bước thời gian hiện tại.

$$r_t = \sigma(W_r x_t + U_r h_{t-1})$$

Trong đó: r_t là giá trị của cổng đặt lại tại thời điểm t , σ là hàm sigmoid, W_r là ma trận trọng số liên quan đến cổng đặt lại, h_{t-1} là trạng thái ẩn tại thời điểm $t-1$, x_t là đầu vào tại thời điểm t .

3) Trạng Thái Ẩn Mới

Trạng thái ẩn mới \tilde{h}_t được tính toán bằng cách sử dụng cổng đặt lại để điều chỉnh thông tin từ trạng thái ẩn trước đó. Nó được tính toán bằng cách sử dụng hàm kích hoạt tanh để nén đầu ra của nó trong khoảng từ -1 đến 1.

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}))$$

Trong đó: \tilde{h}_t là trạng thái ẩn tạm thời tại thời điểm t , W_h là ma trận trọng số, r_t là giá trị của cổng đặt lại tại thời điểm t , h_{t-1} là trạng thái ẩn tại thời điểm $t-1$, x_t là đầu vào tại thời điểm t , tanh là hàm tanh, được sử dụng để giới hạn giá trị đầu ra trong khoảng [-1, 1].

4) Cập Nhật Trạng Thái Ẩn Cuối Cùng

Trạng thái ẩn cuối cùng h_t tại thời điểm t được tính bằng cách kết hợp trạng thái ẩn trước đó và trạng thái ẩn mới theo trọng số của cổng cập nhật:

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

Trong đó: h_t là trạng thái ẩn tại thời điểm t , \tilde{h}_t là trạng thái ẩn tạm thời tại thời điểm t , z_t là giá trị của cổng cập nhật tại thời điểm t , r_t là giá trị của cổng đặt lại tại thời điểm t , h_{t-1} là trạng thái ẩn tại thời điểm $t-1$, x_t là đầu vào tại thời điểm t .

H. LSTM

LSTM là một loại mạng nơ-ron tái hiện (RNN) được thiết kế để xử lý và dự báo chuỗi thời gian. Nó khắc phục các vấn đề về Vanishing Gradient (Đạo hàm bị triệt tiêu) và Exploding Gradient (Bùng nổ đạo hàm) trong các RNN truyền thống, cho phép mô hình lưu trữ thông tin trong thời gian dài. LSTM gồm các đơn vị nhớ (memory cell) với ba cổng chính: cổng quên (forget gate), cổng đầu vào (input gate), và cổng đầu ra (output gate). Các cổng này điều khiển dòng thông tin qua đơn vị nhớ.

a) Cổng Quên (Forget Gate)

Cổng quên quyết định lượng thông tin từ trạng thái trước đó cần được giữ lại hoặc loại bỏ:

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

trong đó, f_t là giá trị của cổng quên tại thời điểm t , W_f là trọng số của cổng quên, h_{t-1} là đầu ra từ bước thời gian trước, x_t là đầu vào tại thời điểm t , và b_f là giá trị bias của cổng quên.

b) Cổng Đầu Vào (Input Gate)

Cổng đầu vào xác định lượng thông tin mới cần được lưu trữ trong trạng thái nhớ:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

trong đó, i_t là giá trị của cổng đầu vào, W_i là trọng số của cổng đầu vào, \tilde{C}_t là giá trị của thông tin mới, W_C là trọng số của thông tin mới, và b_i, b_C lần lượt là các giá trị bias của cổng đầu vào và thông tin mới.

c) Cập Nhật Trạng Thái Nhớ

Trạng thái nhớ được cập nhật bằng cách kết hợp trạng thái cũ và thông tin mới:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

trong đó, C_t là trạng thái nhớ tại thời điểm t và C_{t-1} là trạng thái nhớ từ bước thời gian trước.

d) Cổng Đầu Ra (Output Gate)

Cổng đầu ra xác định đầu ra của đơn vị LSTM dựa trên trạng thái nhớ hiện tại:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

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

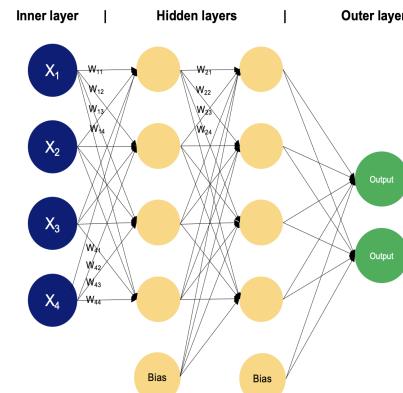
trong đó, o_t là giá trị của cổng đầu ra, W_o là trọng số của cổng đầu ra, b_o là giá trị bias của cổng đầu ra, và h_t là đầu ra của đơn vị LSTM tại thời điểm t .

Các cổng trong LSTM kiểm soát dòng thông tin qua các đơn vị nhớ, giúp mô hình duy trì và cập nhật trạng thái nhớ một cách hiệu quả và chính xác. Điều này giải quyết các vấn đề đạo hàm bị triệt tiêu và bùng nổ đạo hàm, cải thiện hiệu quả dự đoán cho các bài toán chuỗi thời gian.

I. Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) là một loại mạng nơ-ron nhân tạo, là một phần của học sâu (deep learning). MLP bao gồm ít nhất ba lớp: một lớp đầu vào, một hoặc nhiều lớp ẩn, và một lớp đầu ra. Mỗi lớp chứa nhiều nơ-ron (neurons), và mỗi nơ-ron trong một lớp kết nối với tất cả các nơ-ron trong lớp kế tiếp.

MLP có thể được sử dụng cho nhiều nhiệm vụ khác nhau như phân loại, hồi quy và dự báo chuỗi thời gian. Mỗi nơ-ron trong MLP sử dụng một hàm kích hoạt phi tuyến tính để tạo ra đầu ra của nó, giúp mạng có khả năng học các mối quan hệ phi tuyến giữa đầu vào và đầu ra.



Hình 2. Ví dụ về MLP có hai lớp ẩn

1) Mô tả thuật toán

Mạng Perceptron Đa Lớp là một loại mạng nơ-ron nhân tạo (Artificial Neural Network - ANN) bao gồm nhiều lớp perceptron được sắp xếp theo cấu trúc phân tầng. Cấu trúc của MLP thường bao gồm ba loại lớp chính:

- Lớp đầu vào (Input Layer):** Đây là lớp nhận các giá trị đầu vào và truyền chúng vào các lớp tiếp theo. Số lượng nơ-ron trong lớp này tương ứng với số lượng tính năng của dữ liệu đầu vào.
- Lớp ẩn (Hidden Layer):** MLP có thể có một hoặc nhiều lớp ẩn. Các lớp này chịu trách nhiệm học các đặc trưng phức tạp của dữ liệu. Mỗi nơ-ron trong lớp ẩn nhận đầu vào từ tất cả các nơ-ron của lớp trước đó và áp dụng một hàm kích hoạt (activation function) để xác định giá trị đầu ra.
- Lớp đầu ra (Output Layer):** Lớp này cung cấp kết quả cuối cùng của mạng. Số lượng nơ-ron trong lớp đầu ra phụ thuộc vào số lượng lớp mục tiêu (target classes) trong bài toán phân loại hoặc số lượng biến mục tiêu trong bài toán hồi quy.

Công thức cơ bản cho một nơ-ron trong lớp ẩn hoặc lớp đầu ra là:

$$y = f \left(\sum_{i=1}^n w_i \cdot x_i + b \right)$$

Trong đó:

- y là đầu ra của nơ-ron.

- f là hàm kích hoạt (chẳng hạn như hàm sigmoid, tanh, hoặc ReLU).
- x_i là các đầu vào.
- w_i là các trọng số liên kết với các đầu vào.
- b là hệ số điều chỉnh (bias).

2) Thuật Toán Gradient Descent

Gradient descent là một phương pháp tối ưu hóa phổ biến được sử dụng để tìm cực tiểu của hàm mất mát (loss function). Trong ngữ cảnh của MLP, hàm mất mát đo lường sự khác biệt giữa dự đoán của mạng và giá trị thực tế. Công thức cập nhật trọng số trong gradient descent như sau:

$$\theta_i := \theta_i - \eta \frac{\partial J(\theta)}{\partial \theta_i}$$

trong đó, θ_i là trọng số cần cập nhật, η là tốc độ học (learning rate), và $J(\theta)$ là hàm mất mát.

3) Thuật toán Backpropagation

Backpropagation là một phương pháp hiệu quả để tính gradient của hàm mất mát đối với các trọng số của MLP. Quá trình backpropagation bao gồm hai bước chính: lan truyền tiến (forward propagation) và lan truyền ngược (backward propagation).

Trong bước lan truyền tiến, chúng ta tính toán đầu ra của mạng cho một đầu vào cụ thể. Trong bước lan truyền ngược, chúng ta tính toán gradient của hàm mất mát đối với từng trọng số bằng cách áp dụng quy tắc dây chuyền (chain rule).

4) Feedforward

Feedforward là quá trình tính toán đầu ra của mạng từ đầu vào bằng cách đi qua các lớp nơ-ron. Quá trình này bao gồm việc tính toán đầu ra của mỗi nơ-ron trong lớp ẩn và lớp đầu ra.

Giả sử $a^{(l)}$ là đầu ra của lớp l , quá trình feedforward được mô tả như sau:

$$\begin{aligned} z^{(l+1)} &= W^{(l)} a^{(l)} + b^{(l)} \\ a^{(l+1)} &= \sigma(z^{(l+1)}) \end{aligned}$$

trong đó, $W^{(l)}$ và $b^{(l)}$ lần lượt là trọng số và bias của lớp l , $z^{(l+1)}$ là tổng trọng số, và σ là hàm kích hoạt.

J. Autoformer

Autoformer là một mô hình được thiết kế để dự báo chuỗi thời gian dài hạn, bao gồm ba thành phần chính: Khối phân rã chuỗi (Series Decomposition Block), Cơ chế tự tương quan (Auto-Correlation Mechanism), và Bộ mã hóa/giải mã (Encoder/Decoder).

1) Khối Phân Rã Chuỗi

Khối phân rã chuỗi được sử dụng để tách chuỗi thời gian thành các thành phần xu hướng và mùa vụ. Phương pháp này sử dụng trung bình động để làm mượt các dao động định kỳ và làm nổi bật xu hướng dài hạn. Công thức cơ bản như sau:

$$X_t = \text{AvgPool}(\text{Padding}(X))$$

$$X_s = X - X_t$$

Trong đó, X_s và X_t lần lượt là các phần mùa vụ và xu hướng của chuỗi.

2) Cơ Chế Tự Tương Quan

Cơ chế tự tương quan được thiết kế để phát hiện các phụ thuộc dựa trên chu kỳ và tập hợp các chuỗi con tương tự từ các chu kỳ ngầm. Cơ chế này thay thế cho self-attention trong Transformer và có độ phức tạp $O(L \log L)$, trong đó L là chiều dài chuỗi.

a) Công Thức Tự Tương Quan

Cơ chế tự tương quan bao gồm hai bước chính: tính toán tự tương quan và tổng hợp các giá trị tương quan.

$$\text{ACF}(X, \tau) = \frac{1}{L} \sum_{t=1}^{L-\tau} X_t \cdot X_{t+\tau}$$

Trong đó, $\text{ACF}(X, \tau)$ là hàm tự tương quan, X_t là giá trị tại thời điểm t , và τ là độ trễ.

Sau khi tính toán hàm tự tương quan, các giá trị được tổng hợp lại để tìm các chuỗi con tương tự, dựa trên công thức:

$$Y_{t+\tau} = \sum_{k=1}^K \alpha_k X_{t+\tau_k}$$

Trong đó, α_k là trọng số tự tương quan, và K là số lượng chuỗi con được chọn.

3) Bộ Mã Hóa và Bộ Giải Mã

Bộ mã hóa và bộ giải mã của Autoformer bao gồm các khối phân rã chuỗi và cơ chế tự tương quan. Bộ mã hóa xử lý dữ liệu đầu vào từ chuỗi thời gian quá khứ, trong khi bộ giải mã tính chính các thành phần mùa vụ và xu hướng để đưa ra dự báo.

a) Bộ mã hóa (Encoder)

Bộ mã hóa bao gồm các khối phân rã và cơ chế tự tương quan để xử lý và trích xuất thông tin từ chuỗi đầu vào. Định nghĩa các hàm được sử dụng như sau:

- **Decompose(X):** Hàm này phân rã chuỗi thời gian X thành các thành phần xu hướng (X_t) và mùa vụ (X_s).
- **AutoCorrelation(H):** Hàm này tính toán tự tương quan của chuỗi H để tìm các chuỗi con tương tự và các phụ thuộc chu kỳ trong dữ liệu.

$$\begin{aligned} H^{(l)} &= \text{Decompose}(X^{(l)}) \\ Z^{(l)} &= \text{AutoCorrelation}(H^{(l)}) \end{aligned}$$

Trong đó, $H^{(l)}$ là đầu ra của lớp phân rã thứ l , và $Z^{(l)}$ là đầu ra của lớp tự tương quan thứ l .

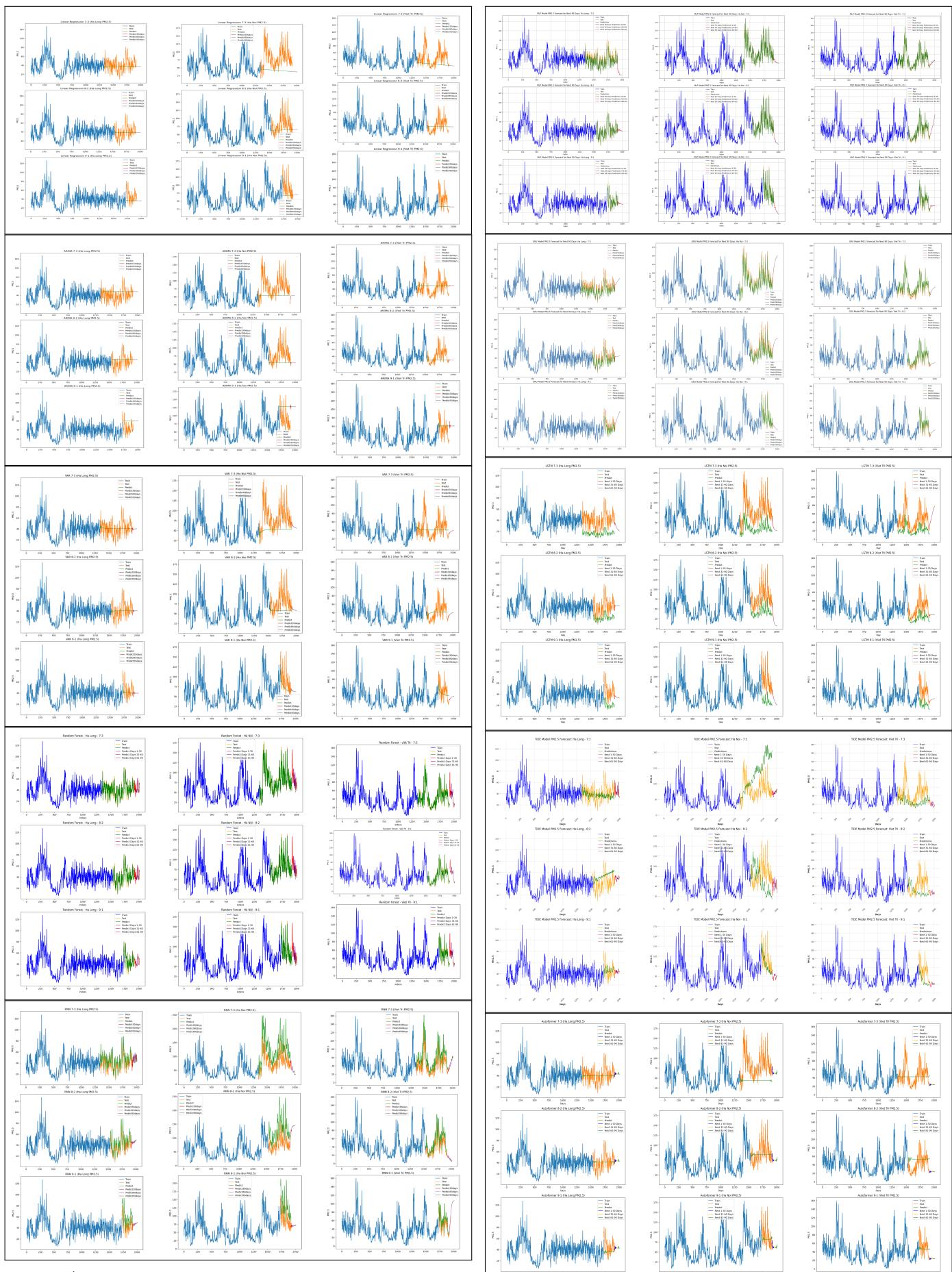
b) Bộ giải mã (Decoder)

Bộ giải mã nhận các thành phần xu hướng và mùa vụ từ bộ mã hóa và tiếp tục tinh chỉnh, sử dụng các khối phân rã và cơ chế tự tương quan để dự báo chuỗi thời gian trong tương lai. Định nghĩa các hàm cũng tương tự như trong bộ mã hóa:

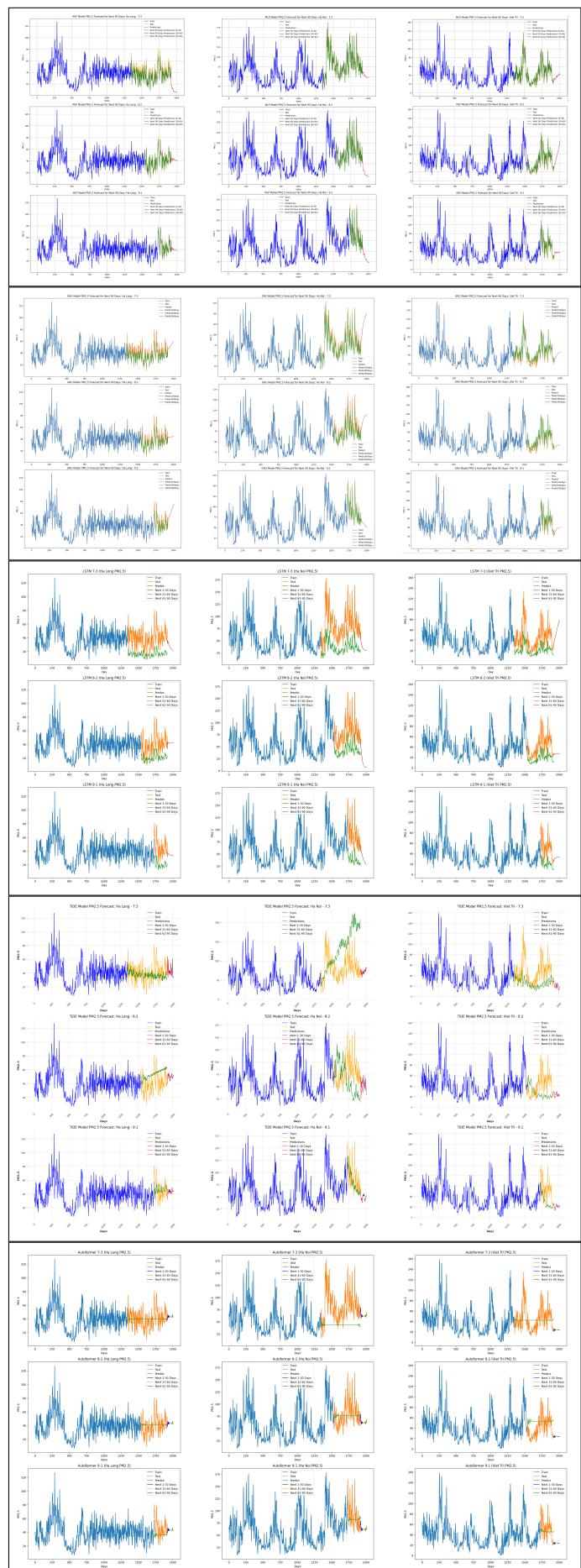
$$\begin{aligned} \hat{Y}^{(l)} &= \text{Decompose}(Z^{(l)}) \\ \hat{X}^{(l)} &= \text{AutoCorrelation}(\hat{Y}^{(l)}) \end{aligned}$$

V. KẾT QUẢ THÍ NGHIỆM

A. Cài đặt mô hình



Hình 3. Kết quả dự báo của mô hình LinearRegression, ARIMA, VAR, RNN

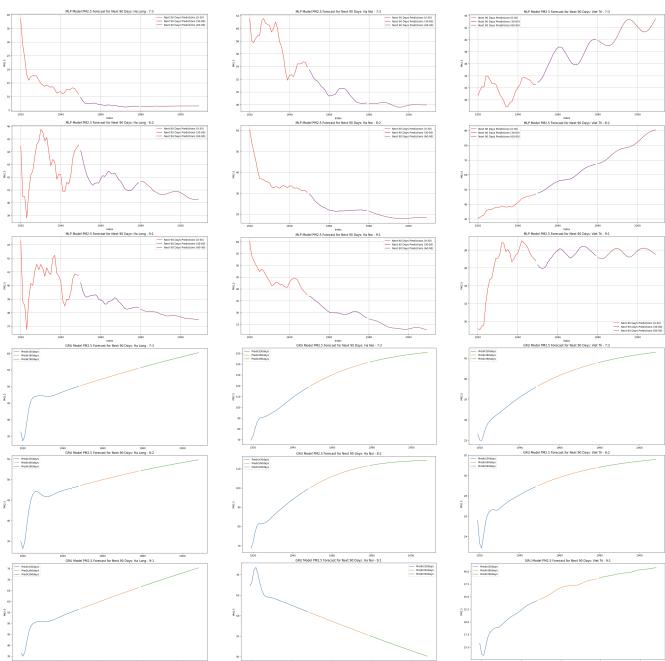


Hình 4. Kết quả dự báo của mô hình MLP, GRU, LSTM, Autoformer

B. Chỉ số Đánh giá và Đường Dự đoán PM.25 cho 90 ngày tiếp theo của hai mô hình tốt nhất

Bảng III
SỐ LIỆU HIỆU SUẤT TRÊN BỘ DỮ LIỆU PM2.5 Ở HÀ NỘI

Model	Ratio	RMSE	MAE	MAPE (%)
LINEAR REGRESSION	7-3	51.26	42.36	46.14
	8-2	30.01	21.71	23.65
	9-1	34.86	25.56	23.76
ARIMA	7-3	41.83	32.03	34.23
	8-2	36.78	28.35	30.36
	9-1	44.69	40.51	52.87
VAR	7-3	43.02	33.01	34.66
	8-2	33.68	25.09	26.72
	9-1	28.47	20.81	19.86
Random Forest	7-3	8.52	6.57	8.25
	8-2	8.23	6.39	7.91
	9-1	9.35	7.51	8.61
RNN	7-3	49.36	44.18	55.23
	8-2	52.56	48.51	62.20
	9-1	42.88	39.47	45.89
MLP	7-3	7.06	5.34	6.63
	8-2	7.63	5.75	7.11
	9-1	8.43	6.50	7.29
GRU	7-3	7.05	6.40	7.35
	8-2	9.43	8.76	7.89
	9-1	3.25	2.45	2.43
LSTM	7-3	50.99	46.98	55.33
	8-2	42.06	38.77	46.54
	9-1	56.05	52.83	56.89
TiDE	7-3	72.62	60.77	83.88
	8-2	49.92	40.00	54.50
	9-1	28.40	21.45	21.76
Autoformer	7-3	50.08	41.07	44.73
	8-2	25.18	18.85	23.51
	9-1	26.91	20.11	20.61



Hình 5. Đường dự đoán 90 ngày của mô hình MLP và GRU

Bảng IV
SỐ LIỆU HIỆU SUẤT TRÊN BỘ DỮ LIỆU PM2.5 Ở HÀ LONG

Model	Ratio	RMSE	MAE	MAPE (%)
LINEAR REGRESSION	7-3	12.48	9.62	26.55
	8-2	13.17	10.12	30.13
	9-1	15.28	11.29	21.79
ARIMA	7-3	14.27	11.53	38.34
	8-2	14.49	11.77	39.81
	9-1	18.50	16.46	42.62
VAR	7-3	12.06	9.36	28.23
	8-2	13.10	10.17	31.21
	9-1	12.44	9.14	18.77
Random Forest	7-3	5.29	4.20	11.60
	8-2	4.67	3.70	10.88
	9-1	4.71	3.74	8.61
RNN	7-3	8.76	6.94	19.54
	8-2	9.21	7.38	20.93
	9-1	13.58	12.08	28.23
MLP	7-3	7.04	5.80	14.10
	8-2	3.98	3.18	9.11
	9-1	4.00	3.17	7.52
GRU	7-3	7.09	5.97	13.57
	8-2	6.84	13.35	5.85
	9-1	5.20	4.38	8.66
LSTM	7-3	26.35	24.60	58.96
	8-2	22.14	20.02	47.18
	9-1	27.13	25.87	55.53
TiDE	7-3	12.67	9.67	26.33
	8-2	18.53	16.29	53.34
	9-1	13.14	10.23	22.95
Autoformer	7-3	12.14	9.36	27.45
	8-2	13.14	10.26	32.43
	9-1	15.67	11.66	22.43

Bảng V
SỐ LIỆU HIỆU SUẤT TRÊN BỘ DỮ LIỆU PM2.5 Ở VIỆT TRÌ

Model	Ratio	RMSE	MAE	MAPE (%)
LINEAR REGRESSION	7-3	28.91	21.07	39.62
	8-2	19.52	15.96	45.73
	9-1	25.77	21.05	36.69
ARIMA	7-3	22.51	17.96	53.78
	8-2	23.04	17.26	37.96
	9-1	20.45	16.52	44.11
VAR	7-3	22.72	17.04	42.66
	8-2	18.76	15.42	45.87
	9-1	17.35	14.54	30.51
Random Forest	7-3	6.31	4.58	10.88
	8-2	5.06	3.80	9.66
	9-1	5.90	4.64	8.39
RNN	7-3	19.97	16.70	41.66
	8-2	15.93	12.87	34.26
	9-1	14.03	11.57	25.25
MLP	7-3	5.31	4.00	9.77
	8-2	5.16	11.67	4.07
	9-1	5.87	4.52	9.69
GRU	7-3	4.32	12.92	3.80
	8-2	4.25	4.00	14.00
	9-1	5.29	5.24	12.22
LSTM	7-3	28.89	24.95	52.06
	8-2	26.87	23.65	54.81
	9-1	34.99	32.43	60.89
TiDE	7-3	30.28	22.28	41.96
	8-2	30.24	24.66	60.67
	9-1	28.36	24.03	42.76
Autoformer	7-3	22.47	16.91	43.82
	8-2	22.52	18.93	69.40
	9-1	19.12	15.51	32.48

VI. KẾT LUẬN

A. Tổng quan

Trong bộ dữ liệu PM2.5 ở Hạ Long, mô hình Random Forest và MLP cho thấy hiệu suất tốt nhất với các chỉ số RMSE, MAE và MAPE thấp nhất ở cả ba tỷ lệ (7-3, 8-2, 9-1). Mô hình LSTM có hiệu suất kém nhất với các chỉ số lỗi cao nhất, đặc biệt là ở tỷ lệ 7-3 và 9-1. Các mô hình khác như Linear Regression, ARIMA và Autoformer cho thấy hiệu suất trung bình, với RMSE, MAE và MAPE biến động tùy thuộc vào tỷ lệ chia dữ liệu.

Trên bộ dữ liệu PM2.5 ở Hà Nội, mô hình MLP và GRU cho thấy hiệu suất tốt nhất với các chỉ số RMSE, MAE và MAPE thấp nhất, đặc biệt ở tỷ lệ 7-3 và 9-1. Mô hình Random Forest cũng đạt hiệu suất tốt nhưng thấp hơn một chút so với MLP và GRU. Mô hình TiDE và LSTM có hiệu suất kém nhất với các chỉ số lỗi cao, đặc biệt là TiDE ở tỷ lệ 7-3 và LSTM ở tỷ lệ 9-1. Các mô hình khác như Linear Regression, ARIMA và Autoformer cho thấy hiệu suất trung bình.

Trên bộ dữ liệu PM2.5 ở Việt Trì, mô hình MLP và GRU tiếp tục thể hiện hiệu suất tốt nhất với các chỉ số RMSE, MAE và MAPE thấp, đặc biệt ở tỷ lệ 7-3 và 9-1. Mô hình Random Forest cũng đạt hiệu suất tốt, mặc dù thấp hơn một chút so với MLP và GRU. Các mô hình như Linear Regression, ARIMA và LSTM có hiệu suất kém hơn với các chỉ số lỗi cao, đặc biệt là mô hình LSTM ở cả ba tỷ lệ. Mô hình TiDE và Autoformer cũng cho thấy hiệu suất trung bình nhưng vẫn kém hơn so với các mô hình dựa trên học sâu như MLP và GRU.

B. Thách thức

Sự phức tạp trong việc xử lý dữ liệu, tính chất vốn có của dữ liệu không khí đòi hỏi phải áp dụng kỹ thuật xử lý dữ liệu chính xác để đảm bảo độ chính xác của các mô hình dự đoán. Mặc dù nhóm đã đi sâu vào khía cạnh lý thuyết của các mô hình, sử dụng các thuật toán để đánh giá hiệu quả của các mô hình dự đoán, tuy nhiên các kết quả cho thấy độ chính xác của một vài mô hình vẫn chưa đạt yêu cầu.

C. Hướng phát triển

Trong tương lai, nhóm em sẽ tiếp tục nghiên cứu các mô hình để dự đoán chỉ số PM2.5 chính xác hơn và nhằm xác định những mô hình phù hợp nhất cho tập dữ liệu của mình. Chúng em sẽ nâng cao việc lựa chọn các kỹ thuật xử lý dữ liệu và nghiên cứu sâu hơn các phương pháp tiên tiến như Deep Learning và Machine Learning mới. Những mô hình này hứa hẹn sẽ cải thiện đáng kể độ chính xác và hiệu quả trong việc dự đoán chỉ số không khí PM2.5. Ngoài ra sẽ áp dụng các mô hình vào các ứng dụng, phần mềm dự báo chất lượng không khí và các dự báo khác trong tương lai.

D. Lời cảm ơn

Nhóm em xin gửi lời cảm ơn chân thành đến PGS.TS Nguyễn Đình Thuân và trợ giảng Nguyễn Minh Nhựt vì đã nhiệt tình hỗ trợ, góp ý, cung cấp chi tiết những kiến thức và hướng dẫn tận tình cho nhóm trong suốt quá trình học và thực hiện đồ án môn học. Nhờ sự chỉ bảo tận tâm của các thầy, nhóm em đã vượt qua nhiều khó khăn trong quá trình thực hiện đồ án. Tuy nhiên,

chúng em không thể tránh khỏi những thiếu sót và rất mong nhận được những nhận xét, góp ý từ thầy để nhóm có thể rút kinh nghiệm và hoàn thiện hơn. Cuối cùng, nhóm em xin chúc thầy luôn mạnh khỏe để tiếp tục sứ mệnh cao cả của nghề giáo, truyền đạt kiến thức cho các thế hệ sinh viên.

TÀI LIỆU

- [1] E. Marinov, D. Petrova-Antanova, and S. Malinov, “Time Series Forecasting of Air Quality: A Case Study of Sofia City,” *Atmosphere*, vol. 13, p. 788, 2022. [Online]. Available:<https://www.mdpi.com/2073-4433/13/5/788>
- [2] K. N. Sh, I. Irfani, and U. 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)*, pp. 14-22, Atlantis Press, 2023. [Online]. Available:https://doi.org/10.2991/978-94-6463-332-0_3
- [3] K. H. Waseem, H. Mushtaq, F. Abid, A. M. Abu-Mahfouz, A. Shaikh, M. Turan, and J. Rasheed, “Forecasting of Air Quality Using an Optimized Recurrent Neural Network,” *Processes*, vol. 10, no. 10, p. 2117, 2022. [Online]. Available: <https://doi.org/10.3390/pr1010211>
- [4] Citation: Esager, M.W.M.; Ünlü, K.D. “Forecasting Air Quality in Tripoli: An Evaluation of Deep Learning Models for Hourly PM2.5 Surface Mass Concentrations. *Atmosphere* 2023, 14, 478. [Online]. Available: <https://doi.org/10.3390/atmos14030478>
- [5] A. Das, W. Kong, A. Leach, S. Mathur, R. Sen, and R. Yu, “Long-term Forecasting with Tide: Time-series Dense Encoder,” *arXiv:2304.08424 [stat.ML]*, April 2023. [Online]. Available: <https://doi.org/10.48550/arXiv.2304.08424>
- [6] B. T. Ong, K. Sugiura, and K. Zetsu, “Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting PM2.5,” *Neural Comput Appl*, vol. 27, no. 6, pp. 1553–1566, 2016. [Online]. Available: <https://doi.org/10.1007/s00521-015-1955-3>
- [7] Y. B. Lim, I. Aliyu, and C. G. Lim, “Air pollution matter prediction using recurrent neural networks with sequential data,” In: *Proceedings of the 2019 3rd International Conference on Intelligent Systems, Meta-heuristics & Swarm Intelligence. ISMSI 2019*, pp. 40–44, Association for Computing Machinery, New York, NY, USA, 2019. [Online]. Available: <https://doi.org/10.1145/3325773.3325788>
- [8] H. Wu, J. Xu, J. Wang, & M. Long, “Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting,” 2021.