

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**



Nguyễn Hoàng Thọ

**NGHIÊN CỨU PHÁT TRIỂN HỆ THỐNG AI DỰ
ĐOÁN VÀ ĐIỀU CHỈNH THÔNG SỐ MÔI TRƯỜNG
TRONG TRANG TRẠI CHĂN NUÔI LỢN**

KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CLC

Ngành: Khoa học máy tính

HÀ NỘI – 2025

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Nguyễn Hoàng Thọ

NGHIÊN CỨU PHÁT TRIỂN HỆ THỐNG AI DỰ
ĐOÁN VÀ ĐIỀU CHỈNH THÔNG SỐ MÔI TRƯỜNG
TRONG TRANG TRẠI CHĂN NUÔI LỢN

Ngành: Khoa học máy tính

Cán bộ hướng dẫn: Ts. Nguyễn Ngọc Tân

Cán bộ đồng hướng dẫn: MSc. Nguyễn Thái Dương

HÀ NỘI – 2025

**VIETNAM NATIONAL UNIVERSITY, HANOI
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

Nguyen Hoang Tho

**RESEARCH AND DEVELOPMENT ON AN AI
SYSTEM FOR PREDICTING AND REGULATING
ENVIRONMENTAL PARAMETERS IN PIG FARMING**

Major: Computer Science

Supervisor: Dr. Nguyen Ngoc Tan

Co-Supervisor: MSc. Nguyen Thai Duong

HANOI – 2025

TÓM TẮT

Tóm tắt: Với sự phát triển nhanh chóng của nền kinh tế xanh và kinh tế tuần hoàn, ngành nông nghiệp nói chung và chăn nuôi lợn nói riêng đang đối mặt với yêu cầu chuyển đổi mạnh mẽ nhằm hướng tới nền nông nghiệp bền vững, đặc biệt là mục tiêu đạt phát thải ròng bằng không. Đề tài này trình bày một nghiên cứu so sánh toàn diện nhằm tìm ra hệ thống trí tuệ nhân tạo (AI) tối ưu cho việc dự báo các thông số môi trường (như NH₃, H₂S, CO₂) trong trang trại chăn nuôi lợn, từ đó cho phép điều chỉnh chủ động nhằm nâng cao hiệu quả quản lý mùi và giảm thiểu tác động tiêu cực đến môi trường. Nghiên cứu sử dụng dữ liệu IoT thực tế, phức tạp và thường xuyên nhiều thu thập từ một trang trại tại Việt Nam. Sau quá trình tiền xử lý dữ liệu chuyên biệt, đề tài đã triển khai, huấn luyện và đánh giá hiệu quả của một loạt các nhóm mô hình AI đa dạng, bao gồm các mô hình thống kê truyền thống (ARIMA, SARIMA), học máy dựa trên cây quyết định (Random Forest, XGBoost), mạng nơ-ron tuần tự (LSTM), mạng nơ-ron dựa trên cơ chế chú ý (Transformer), các mô hình dựa trên thực thể lân cận (KNN), và các thuật toán học máy khác (SVM, Bayesian Regression). Nghiên cứu này không chỉ cung cấp một đánh giá chi tiết về hiệu năng của các mô hình AI trên dữ liệu thực tế mà còn đề xuất cơ sở cho việc phát triển các hệ thống quản lý môi trường thông minh, chủ động trong nông nghiệp bền vững.

Từ khóa: AI, Quản lý mùi, Khí thải trong chăn nuôi, Dự báo chuỗi thời gian

Abstract

Abstract: With the rapid development of the green economy and circular economy, the agricultural sector in general, and pig farming in particular, are facing a strong demand for transformation towards sustainable agriculture, especially with the goal of achieving net-zero emissions. This project presents a comprehensive study aimed at identifying an optimal artificial intelligence (AI) system for forecasting environmental parameters (such as NH₃, H₂S, CO₂) in pig farms, thereby enabling proactive adjustments to enhance odor management and minimize negative environmental impacts. The study uses real-world, complex, and often noisy IoT data collected from a pig farm in Vietnam. After specialized data preprocessing, the project implements, trains, and evaluates the performance of a wide range of diverse AI model groups, including traditional statistical models (ARIMA, SARIMA), tree-based machine learning models (Random Forest, XGBoost), sequential neural networks (LSTM), attention-based neural networks (Transformer), proximity-based models (KNN), and other machine learning algorithms (SVM, Bayesian Regression). This research not only provides a detailed evaluation of AI model performance on real-world data but also lays the foundation for developing intelligent, proactive environmental management systems in sustainable agriculture.

Keywords: AI, Odor management, Livestock emissions, Time series forecasting

Lời cảm ơn

Trong thời gian học tập tại khoa Khoa học máy tính, Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội, tôi đã nhận được rất nhiều sự quan tâm, giúp đỡ tận tình từ cá nhân và cũng như đơn vị. Bản thân tôi vô cùng cảm kích với những gì mình đã nhận được.

Khóa luận tốt nghiệp được hoàn thành dưới sự hướng dẫn trực tiếp của TS. Nguyễn Ngọc Tân. Với lòng kính trọng và biết ơn sâu sắc, tôi xin chân thành cảm ơn thầy đã dành thời gian hướng dẫn tận tình, chỉ bảo trong thời gian thực hiện khóa luận tốt nghiệp này. Hơn nữa, trong những năm học tập tại Khoa Công nghệ nông nghiệp (Trường Đại học Công nghệ - ĐHQGHN), thầy đã luôn giảng giải, truyền đạt những kiến thức bổ ích, hướng dẫn tôi những phương pháp tư duy trong khoa học cũng như trong cuộc sống thực tế.

Tôi cũng xin chân thành cảm ơn các thầy cô giáo trong Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội nói chung, cũng như khoa Khoa học máy tính nói riêng, đã cho tôi kiến thức về các môn đại cương cũng như các môn chuyên ngành, giúp tôi có được cơ sở lý thuyết vững vàng và tạo điều kiện giúp đỡ trong suốt quá trình học tập.

Cuối cùng, tôi xin chân thành cảm ơn gia đình và bạn bè, đã luôn tạo điều kiện, quan tâm, giúp đỡ, động viên tôi trong suốt quá trình học tập và hoàn thành khóa luận tốt nghiệp.

Hà Nội, ngày tháng năm 2025

Sinh viên thực hiện

Nguyễn Hoàng Thọ

Lời cam đoan

Tôi xin cam đoan đề tài “NGHIÊN CỨU PHÁT TRIỂN HỆ THỐNG AI DỰ ĐOÁN VÀ ĐIỀU CHỈNH THÔNG SỐ MÔI TRƯỜNG TRONG TRANG TRAI CHĂN NUÔI LỢN” do TS. Nguyễn Ngọc Tân hướng dẫn là công trình nghiên cứu của tôi. Các nội dung nghiên cứu và kết quả trong đồ án này đều trung thực và không sao chép từ công trình của người khác.

Tất cả các tài liệu tham khảo sử dụng trong đồ án đều được ghi rõ nguồn gốc và tên

tác giả. Nếu có sai sót, tôi xin hoàn toàn chịu trách nhiệm.

Hà Nội, ngày tháng năm 2025

Sinh viên thực hiện
Nguyễn Hoàng Thọ

Mục lục

Chương 1: Mở đầu.....	1
1.1. Đặt vấn đề.....	1
1.1.1. Một số các tác nhân chính dẫn đến gây mùi trong chăn nuôi lợn.....	2
1.1.2. Về quá trình tích hợp AI trong nông nghiệp.....	3
1.2. Tổng quan.....	6
1.3. Mục tiêu nghiên cứu.....	6
1.4. Đối tượng nghiên cứu.....	7
1.5. Phương pháp nghiên cứu.....	7
1.6. Nội dung nghiên cứu.....	7
1.7. Ý nghĩa thực tiễn	8
Chương 2: Cơ sở lý thuyết.....	9
2.1. Dự báo chuỗi thời gian.....	9
2.2. Phân tích dữ liệu chuỗi thời gian	9
2.2.1. Level	10
2.2.2. Trend	10
2.2.3. Seasonality	10
2.2.4. Noise	11
2.3. Stationarity và differencing.....	12
2.4. Về các tiêu chí đánh giá mô hình	13
2.4.1. Mean Absolute Error (MAE).....	13
2.4.2. Mean Absolute Percentage Error (MAPE)	13
2.4.3. Root Mean Squared Error (RMSE)	14
2.4.4. Coefficient of Determination (R^2 score)	14
Chương 3: Triển khai bài toán.....	15
3.1. Xử lý tập dữ liệu thô	15
3.1.1. Trực quan hóa dữ liệu	19
3.1.2. Tiến hành tiền xử lý dữ liệu	30

3.2. Lựa chọn mô hình	40
3.3. Triển khai nhóm các mô hình cây quyết định.....	41
3.3.1. Random Forest Regression	41
3.3.2. XGBoost	44
3.4. Triển khai nhóm các mô hình tuần tự	45
3.4.1. Tensorflow LSTM	45
3.4.2. ARIMA	49
3.4.3. Prophet	52
3.5. Triển khai nhóm các mô hình attention	53
3.5.1. Transformer.....	53
3.6. Triển khai nhóm các mô hình học ví dụ.....	56
3.6.1. K-Nearest Neighbors	56
3.7. Các mô hình/cách tiếp cận khác.....	57
3.7.1. Naïve Mean.....	57
3.7.2. SVM	58
3.7.3. Bayesian Regression	59
Chương 4: Phân tích và kết luận	60
4.1. Kết quả đã thu được	60
4.2. Phân tích kết quả	76
4.3. Điều chỉnh thông số	77
Chương 5: Kết luận và hướng phát triển	80
Tài liệu tham khảo	82

Danh mục biểu đồ

Biểu đồ 3.1: Violinplot của dữ liệu chưa tiền xử lý	25
Biểu đồ 3.2: Boxplot của dữ liệu chưa tiền xử lý	26
Biểu đồ 3.3: Histogram của dữ liệu chưa tiền xử lý.....	27
Biểu đồ 3.4: Violinplot của dữ liệu đã tiền xử lý	35
Biểu đồ 3.5: Boxplot của dữ liệu đã tiền xử lý.....	36
Biểu đồ 3.6: Histogram của dữ liệu đã tiền xử lý.....	37
Biểu đồ 4.1: Dự báo của Random Forest trên cả 3 nồng độ.....	63
Biểu đồ 4.2: Dự báo của XGBoost trên cả 3 nồng độ	64
Biểu đồ 4.3: Dự báo của Tensorflow LSTM trên cả 3 nồng độ	65
Biểu đồ 4.4: Dự báo của ARIMA trên cả 3 nồng độ	66
Biểu đồ 4.5: Dự báo của SARIMA trên cả 3 nồng độ.....	67
Biểu đồ 4.6: Dự báo của Prophet (không ngoại sinh) trên cả 3 nồng độ	68
Biểu đồ 4.7: Dự báo của Prophet (nồng độ CO ₂ , nhiệt độ và độ ẩm) trên cả 3 nồng độ	69
Biểu đồ 4.8: Dự báo của Prophet (tất cả các cột làm ngoại sinh) trên cả 3 nồng độ .	70
Biểu đồ 4.9: Dự báo của Transformer trên cả 3 nồng độ	71
Biểu đồ 4.10: Dự báo của K-Nearest Neighbors trên cả 3 nồng độ	72
Biểu đồ 4.11: Dự báo của Naive Mean trên cả 3 nồng độ	73
Biểu đồ 4.12: Dự báo của SVM trên cả 3 nồng độ.....	74
Biểu đồ 4.13: Dự báo của hồi quy Bayes - Bayes Regression trên cả 3 nồng độ	75

Danh mục bảng biểu

Bảng 3.1: Heatmap ma trận tương quan của dữ liệu chưa tiền xử lý.....	27
Bảng 3.2: Heatmap ma trận tương quan của dữ liệu đã tiền xử lý	38
Bảng 3.3: Tham số huấn luyện Random Forest	43
Bảng 3.4: Tham số huấn luyện XGBoost.....	44
Bảng 3.5: Tham số huấn luyện LSTM	47
Bảng 3.6: Tham số huấn luyện Prophet.....	52
Bảng 3.7: Tham số huấn luyện Transformer.....	53
Bảng 3.8: Tham số huấn luyện KNN	57
Bảng 3.9: Tham số huấn luyện SVM	58
Bảng 4.1: Kết quả đối với nồng độ 250g/500l	60
Bảng 4.2: Kết quả đối với nồng độ 500g/500l	61
Bảng 4.3: Kết quả đối với nồng độ 750g/500l	61
Bảng 4.4: Logic điều khiển đè xuất đối với các nồng độ khí khác nhau	78

Danh mục hình ảnh

Hình 1.1: Sản phẩm "See And Spray" của công ty Blue River Technology.....	4
Hình 1.2: Sản phẩm vòng đeo cổ AI cho bò của công ty Connecterra	5
Hình 2.1: Biểu đồ mẫu chỉ chúa nhiễu trắng.....	11
Hình 3.1: Phân chia dữ liệu gốc theo giai đoạn.....	15
Hình 3.2: Trích đoạn tập dữ liệu gốc định dạng csv	16
Hình 3.3: Trích đoạn tập dữ liệu gốc đã sử dụng Text to Columns	16

Danh mục thuật ngữ và từ viết tắt

Viết tắt	Thuật ngữ bằng tiếng anh	Tên tiếng việt
AI	Artificial Intelligence	Trí tuệ nhân tạo
TVOCs	Total volatile organic compounds	Tổng hợp các hợp chất hữu cơ dễ bay hơi
IoT	Internet of Things	Internet vạn vật
MAE	Mean Absolute Error	Sai số tuyệt đối trung bình
MAPE	Mean Absolute Percentage Error	Sai số phần trăm tuyệt đối trung bình
R ²	R-squared	Hệ số xác định R bình phương
RNN	Recurrent Neural Network	Mạng nơ-ron hồi quy
Ppm	Parts per million	Phần triệu
CSV	Comma Separated Values	Các giá trị được phân tách bằng dấu phẩy
NaN	Not a Number	Không phải là số
MSE	Mean Squared Error	Sai số bình phương trung bình
RMSE	Root Mean Squared Error	Sai số căn bậc hai trung bình
	Time Series	Chuỗi thời gian
	Level	Mức cơ bản
	Trend	Xu hướng
	Seasonality/cycles	Tính thời vụ/chu kỳ
	Noise	Nhiễu
	Stationary (series)	(chuỗi) Tĩnh, mang tính dừng
	Discrete/continuous (series)	(chuỗi) Rời rạc/liên tục
	Uni-variate/Multi-variate (series)	(chuỗi) Đơn biến/đa biến
	Mean	Giá trị trung bình
	Variance	Phương sai
	Correlation	Tính tương quan
	Hyperparameter	Siêu tham số
	Training/inference time	Thời gian huấn luyện/suy luận

Chương 1: Mở đầu

1.1. Đặt vấn đề

Trong những năm gần đây, ngành chăn nuôi lợn tại Việt Nam đã có sự phát triển vượt bậc, đóng vai trò quan trọng trong cơ cấu nông nghiệp và góp phần lớn vào đảm bảo an ninh lương thực quốc gia. Đặc biệt, chăn nuôi lợn không chỉ giúp nâng cao thu nhập cho nhiều hộ gia đình ở vùng nông thôn mà còn cung cấp sản phẩm cho thị trường nội địa và xuất khẩu ra quốc tế [1].

Theo số liệu từ Tổng cục Thống kê Việt Nam, tổng đàn lợn trên cả nước tính đến tháng 7/2024 đạt hơn 25,5 triệu con, tăng 2,9% so với cùng kỳ năm trước. Khu vực Trung du và miền núi phía Bắc là nơi tập trung chăn nuôi lợn nhiều nhất (chiếm khoảng 22,9%), bên cạnh đó, các địa phương như Đồng Nai, Hà Nội, Bình Phước, Bắc Giang, Thanh Hoá, Bình Định, Gia Lai cũng ghi nhận sự phát triển mạnh mẽ trong lĩnh vực này. Đặc biệt, Hà Nội duy trì tốc độ tăng trưởng đàn lợn liên tục, hiện đạt hơn 1,48 triệu con. Cùng với đó, sản lượng thịt lợn hơi xuất chuồng cũng tăng lên đáng kể: năm 2023 đạt trên 4,8 triệu tấn (tăng 6,7% so với năm 2022) và riêng 6 tháng đầu năm 2024 đạt gần 2,54 triệu tấn, tăng 5,1% so với cùng kỳ năm trước [1].

Tuy nhiên, sự phát triển nhanh chóng của ngành chăn nuôi lợn cũng kéo theo nhiều thách thức, nổi bật là vấn đề ô nhiễm môi trường do chất thải chăn nuôi và yêu cầu về quản lý vệ sinh an toàn thực phẩm, kiểm soát dịch bệnh [1]. Hiện nay, nhiều trang trại vẫn còn nằm gần hoặc xen kẽ trong các khu dân cư, gây ra phiền toái cho cộng đồng xung quanh bởi mùi hôi kéo dài và tiềm ẩn nguy cơ ảnh hưởng sức khỏe. Để kiểm soát mùi, các cơ sở chăn nuôi cần áp dụng các giải pháp xử lý cá nguyên nhân lẩn mùi phát sinh, đồng thời tuân thủ các quy định nghiêm ngặt về kiểm soát chất lượng không khí tại khu vực chuồng trại và vùng lân cận, như đã được áp dụng ở nhiều quốc gia [2].

Không chỉ tác động đến con người, mùi hôi và khí thải từ chuồng trại còn ảnh hưởng tiêu cực đến vật nuôi. Khi phải sống trong môi trường ô nhiễm bởi các khí độc như ammonia, methane, hydrogen sulfide, vật nuôi dễ bị tổn thương hệ hô hấp, mắc các bệnh về phổi, giảm sức đề kháng, thậm chí bị căng thẳng, ăn uống kém, giảm tăng trọng và ảnh hưởng đến chất lượng sản phẩm [3].

Nhìn chung, ngành chăn nuôi lợn tại nước ta đang trên đà phát triển ngày một mạnh mẽ, tuy nhiên vẫn còn nhiều thách thức cần giải quyết, đặc biệt là vấn đề ô nhiễm môi trường.

1.1.1. Một số các tác nhân chính dẫn đến mùi trong chăn nuôi lợn

Mùi hôi trong chăn nuôi lợn không chỉ gây khó chịu về mặt cảm quan mà còn có thể tác động tiêu cực đến sức khỏe của cả vật nuôi lẫn con người. Nguồn gốc của vấn đề này bắt nguồn từ nhiều yếu tố khác nhau, trong đó nổi bật là quá trình phân hủy tự nhiên của chất thải, điều kiện môi trường xung quanh không đảm bảo, chế độ dinh dưỡng chứa nhiều thành phần dễ tạo mùi, cũng như tác động từ việc sử dụng thuốc và hóa chất trong quá trình chăn nuôi. Bên cạnh đó, công tác quản lý chuồng trại chưa hiệu quả cũng góp phần làm gia tăng tình trạng phát sinh mùi hôi trong môi trường chăn nuôi.

Các tác nhân chính gây ra mùi khó chịu trong chăn nuôi lợn bao gồm:

Ammoniac (NH_3): Đây là một trong những khí gây mùi phổ biến nhất trong các trang trại chăn nuôi lợn. Ammoniac hình thành chủ yếu từ quá trình phân hủy urê có trong phân lợn dưới tác động của vi sinh vật. Khí này không chỉ tạo ra mùi khai khó chịu mà còn có thể gây kích ứng mắt, đường hô hấp của cả vật nuôi lẫn con người khi tiếp xúc lâu dài, làm giảm sức đề kháng và tăng nguy cơ mắc bệnh hô hấp [4].

Tác hại:

- **Đối với con người:** Hít phải amoniac ở nồng độ cao có thể gây kích ứng mắt, mũi, họng và hệ hô hấp. Nồng độ amoniac từ 20 đến 50 ppm có thể gây khó chịu và ảnh hưởng đến sức khỏe [5].

- **Đối với vật nuôi:** Amoniac làm giảm sức đề kháng của lợn, gây stress và tăng nguy cơ mắc các bệnh về hô hấp. Nồng độ NH_3 trên 20 ppm có thể làm giảm tăng trưởng và hiệu quả chăn nuôi.

- **Đối với môi trường:** Amoniac bay vào không khí có thể kết hợp với các khí khác tạo thành bụi mịn PM2.5, gây ô nhiễm không khí.

Hydrogen Sulfide (H_2S): Hydrogen sulfide là khí có mùi trứng thối đặc trưng, phát sinh khi chất thải hữu cơ phân hủy trong điều kiện yếm khí (thiếu oxy). Đây là một loại khí độc, ngay cả ở nồng độ thấp cũng có thể gây ảnh hưởng xấu đến sức khỏe vật nuôi và con người, như gây đau đầu, chóng mặt, thậm chí nguy hiểm đến tính mạng nếu tiếp xúc với nồng độ cao [6].

Tác hại:

- **Đối với con người:** H_2S là một khí cực kỳ nguy hiểm. Hydro sulfua có thể gây ngạt thở, hôn mê và thậm chí tử vong nhanh chóng từ nồng độ 100ppm trở lên [7].

• Đối với vật nuôi: Tương tự như con người, lợn tiếp xúc với H₂S có thể bị stress, giảm khả năng sinh trưởng và tăng nguy cơ mắc bệnh.

• Đối với môi trường: H₂S có thể gây ô nhiễm không khí và đất, ảnh hưởng đến hệ sinh thái xung quanh.

Methane (CH₄): Methane là khí nhà kính được sinh ra trong quá trình phân hủy khí của chất thải hữu cơ trong chuồng trại. Mặc dù methane không có mùi, nhưng tích tụ methane trong không khí ở nồng độ cao tiềm ẩn nguy cơ cháy nổ [8].

Tác hại:

• Đối với con người: Mặc dù metan không độc trực tiếp, nhưng metan có thể gây ngạt thở khi tích tụ với nồng độ cao trong không gian kín. Ngoài ra, CH₄ là một khí nhà kính mạnh, góp phần vào biến đổi khí hậu.

• Đối với vật nuôi: Metan không ảnh hưởng trực tiếp đến sức khỏe của lợn, nhưng sự tích tụ của metan trong chuồng trại có thể làm giảm nồng độ oxy, gây stress cho vật nuôi.

• Đối với môi trường: Metan là một trong những khí nhà kính chính, có khả năng giữ nhiệt cao gấp 28 lần so với CO₂.

1.1.2. Về quá trình tích hợp AI trong nông nghiệp

Trong bối cảnh biến đổi khí hậu, gia tăng dân số và nhu cầu lương thực toàn cầu, việc ứng dụng trí tuệ nhân tạo (AI) trong nông nghiệp đang trở thành xu hướng tất yếu. AI không chỉ giúp tăng năng suất, giảm chi phí mà còn góp phần phát triển nông nghiệp bền vững. Trên thế giới, AI đã được tích hợp vào nhiều khía cạnh của nông nghiệp, từ canh tác thông minh đến quản lý chuỗi cung ứng. Tại Việt Nam, mặc dù còn nhiều thách thức, nhưng tiềm năng ứng dụng AI trong nông nghiệp là rất lớn, đặc biệt trong bối cảnh Chính phủ đang thúc đẩy chuyển đổi số và cách mạng công nghiệp 4.0.

Trên phạm vi toàn cầu, AI đang được ứng dụng rộng rãi trong nông nghiệp thông qua các giải pháp tự động hóa và robot hóa [9]. Các robot nông nghiệp được trang bị AI có thể thực hiện các công việc như gieo hạt, tưới tiêu, thu hoạch và phân loại sản phẩm. Ví dụ, công ty Blue River Technology (Mỹ) đã phát triển robot "See & Spray" sử dụng AI để phun thuốc trừ sâu chính xác, giảm lượng hóa chất sử dụng [10]. Ngoài ra, AI còn được sử dụng để phân tích dữ liệu từ vệ tinh, cảm biến và drone, giúp nông dân dự báo thời tiết, tình trạng đất đai và dịch bệnh. Công ty The Climate Corporation (Mỹ) cung cấp nền tảng AI giúp nông dân đưa ra quyết định tối ưu về thời điểm gieo trồng và thu hoạch [11]. AI cũng đóng vai trò quan trọng trong việc quản lý tài nguyên hiệu quả, giúp

tối ưu hóa việc sử dụng nước, phân bón và năng lượng. Hệ thống tưới tiêu thông minh sử dụng AI có thể điều chỉnh lượng nước phù hợp với nhu cầu của cây trồng, giảm thiểu lãng phí. Trong lĩnh vực chăn nuôi, AI giúp theo dõi sức khỏe vật nuôi, dự đoán bệnh tật và tối ưu hóa chế độ ăn. Ví dụ, công ty Connecterra (Hà Lan) phát triển thiết bị đeo cho bò sữa sử dụng AI để theo dõi hành vi và sức khỏe của chúng [12]. Những ứng dụng này không chỉ giúp tăng năng suất và giảm chi phí mà còn góp phần bảo vệ môi trường và nâng cao chất lượng sản phẩm.



Hình 1.1: Sản phẩm "See And Spray" của công ty Blue River Technology

Tuy nhiên, việc tích hợp AI trong nông nghiệp toàn cầu cũng đối mặt với nhiều thách thức. Chi phí đầu tư cao là một rào cản lớn, đặc biệt là đối với các nước đang phát triển. Ngoài ra, việc ứng dụng AI đòi hỏi nguồn nhân lực có trình độ kỹ thuật và chuyên môn, trong khi nhiều nông dân vẫn chưa được trang bị đủ kiến thức và kỹ năng. Vấn đề về dữ liệu cũng là một thách thức, vì dữ liệu nông nghiệp thường phân tán và không đồng nhất, gây khó khăn cho việc phân tích.

Tại Việt Nam, tiềm năng ứng dụng AI trong nông nghiệp là rất lớn nhờ điều kiện tự nhiên thuận lợi và sự hỗ trợ từ Chính phủ. Việt Nam có khí hậu nhiệt đới gió mùa, đất đai màu mỡ, phù hợp để phát triển nông nghiệp công nghệ cao. Chính phủ đã ban hành nhiều chính sách khuyến khích ứng dụng công nghệ cao trong nông nghiệp, như Chương trình Quốc gia về Cách mạng Công nghiệp 4.0 [13]. Các công ty công nghệ trong nước và quốc tế cũng đang đầu tư vào lĩnh vực nông nghiệp thông minh tại Việt

Nam. Một số doanh nghiệp và hợp tác xã đã ứng dụng AI để quản lý tưới tiêu, theo dõi tình trạng cây trồng và dự báo sâu bệnh. Ví dụ, công ty MimosaTEK (Việt Nam) cung cấp giải pháp tưới tiêu thông minh sử dụng cảm biến và AI [14]. Trong lĩnh vực chăn nuôi, AI được sử dụng để theo dõi sức khỏe vật nuôi và tối ưu hóa chế độ ăn. Một số trang trại lớn và gia cầm đã áp dụng hệ thống quản lý tự động để giám sát nhiệt độ, độ ẩm và thức ăn. AI cũng đang được sử dụng để quản lý và dự báo nhu cầu thị trường, tối ưu hóa quy trình vận chuyển và bảo quản [15].



Hình 1.2: Sản phẩm vòng đeo cổ AI cho bò của công ty Connecterra

Tuy nhiên, việc tích hợp AI trong nông nghiệp tại Việt Nam cũng gặp nhiều thách thức. Quy mô sản xuất nhỏ lẻ là một rào cản lớn, vì phần lớn nông dân Việt Nam vẫn sản xuất nhỏ lẻ, khó áp dụng công nghệ cao. Hệ thống hạ tầng công nghệ thông tin và truyền thông ở nông thôn còn yếu kém, gây khó khăn cho việc triển khai các giải pháp AI. Ngoài ra, nhiều nông dân chưa có đủ kiến thức và kỹ năng để sử dụng các công nghệ AI, đòi hỏi cần có các chương trình đào tạo và nâng cao nhận thức.

Để phát triển AI trong nông nghiệp tại Việt Nam, cần có sự hỗ trợ từ Chính phủ và các bên liên quan. Chính phủ cần có chính sách hỗ trợ tài chính và khuyến khích doanh nghiệp đầu tư vào nông nghiệp thông minh. Đồng thời, cần đầu tư vào hạ tầng công nghệ thông tin và truyền thông tại nông thôn, cũng như tăng cường hợp tác quốc tế để học hỏi kinh nghiệm và chuyên giao công nghệ.

1.2. Tổng quan

Ngành chăn nuôi lợn đóng vai trò quan trọng trong nền kinh tế nông nghiệp, đặc biệt là trong việc đảm bảo nguồn cung thực phẩm cho người dân. Tuy nhiên, sự phát triển của ngành này cũng kéo theo những thách thức lớn về môi trường và sức khỏe cộng đồng. Một trong những vấn đề nổi cộm là sự phát sinh các khí thải độc hại như ammoniac (NH_3), hydro sulfide (H_2S) và các hợp chất hữu cơ dễ bay hơi (VOCs) từ quá trình chăn nuôi. Những khí này không chỉ gây mùi khó chịu mà còn ảnh hưởng nghiêm trọng đến sức khỏe con người và động vật, đồng thời góp phần làm gia tăng ô nhiễm môi trường.

Để giải quyết vấn đề này, việc theo dõi và điều chỉnh các thông số môi trường trong trang trại chăn nuôi lợn trở thành yêu cầu cấp thiết. Các phương pháp truyền thống thường dựa vào việc thu thập dữ liệu thủ công và phân tích đơn giản, dẫn đến độ chính xác thấp và khó áp dụng trên quy mô lớn. Trong bối cảnh đó, sự phát triển của trí tuệ nhân tạo (AI) và các thuật toán học máy đã mở ra những hướng tiếp cận mới, hiệu quả hơn.

Nghiên cứu này tập trung vào việc so sánh và đánh giá hiệu quả của ba mô hình học máy tiên tiến trong việc dự đoán và điều chỉnh các thông số môi trường: LSTM (Long Short-Term Memory), Transformer, Random Forest Regression cũng như các mô hình khác.

Nghiên cứu này không chỉ so sánh hiệu quả của các mô hình trên tương quan với bối cảnh thực tế mà bài toán đưa ra mà còn đề xuất giải pháp tối ưu cho bài toán quản lý môi trường trong trang trại chăn nuôi lợn.

1.3. Mục tiêu nghiên cứu

- **Xây dựng và đánh giá các mô hình AI** có khả năng dự báo chính xác các thông số môi trường trọng yếu (như nhiệt độ, độ ẩm, nồng độ khí CO_2 , NH_3 , H_2S) dựa trên dữ liệu lịch sử thu thập từ hệ thống cảm biến IoT trong trang trại chăn nuôi lợn.
- **Thực hiện phân tích so sánh hiệu quả** giữa các nhóm mô hình AI khác nhau (bao gồm LSTM, Transformer, Random Forest Regression và các mô hình khác) trong việc dự đoán nồng độ khí CO_2 làm chỉ số tham chiếu, từ đó suy ra khả năng dự đoán các khí gây mùi khác.
- **Đánh giá hiệu quả và tính thực tiễn** của từng mô hình thông qua các chỉ số thống kê phù hợp (R^2 , MAE, RMSE, MAPE) và khả năng hoạt động trên dữ liệu thực tế, phức tạp của bài toán.

- **Đề xuất định hướng** tích hợp mô hình AI tối ưu vào một hệ thống điều khiển môi trường tự động, nhằm nâng cao chất lượng không khí và giảm thiểu tác động tiêu cực của hoạt động chăn nuôi.

1.4. Đối tượng nghiên cứu

Đối tượng nghiên cứu chính của đề tài bao gồm:

- **Dữ liệu môi trường từ trang trại chăn nuôi lợn:** Các tập dữ liệu chuỗi thời gian lịch sử về nhiệt độ, độ ẩm, nồng độ khí CO₂, NH₃, H₂S và các thông số liên quan khác, thu thập qua hệ thống IoT.
- **Các mô hình Trí tuệ Nhân tạo:** Bao gồm các thuật toán và kiến trúc AI như LSTM, Transformer, Random Forest Regression, XGBoost, ARIMA, Prophet, SVM, KNN, và Bayesian Regression, được áp dụng cho bài toán dự báo chuỗi thời gian trong bối cảnh nghiên cứu này.

1.5. Phương pháp nghiên cứu

Nghiên cứu được thực hiện theo quy trình sau:

- **Nghiên cứu lý thuyết và tổng quan tài liệu:** Tìm hiểu sâu về các vấn đề môi trường trong chăn nuôi lợn, các kỹ thuật IoT ứng dụng, các phương pháp dự báo chuỗi thời gian và các mô hình AI liên quan.
- **Thu thập và tiền xử lý dữ liệu:** Phân tích, làm sạch, xử lý nhiễu, ngoại lai và chuẩn hóa dữ liệu lịch sử từ cảm biến IoT để đảm bảo chất lượng đầu vào cho các mô hình AI.
- **Xây dựng và huấn luyện mô hình dự đoán:** Lựa chọn, triển khai và huấn luyện các mô hình AI đã xác định (sử dụng các thư viện và nền tảng phù hợp như Python, TensorFlow, Scikit-learn) để dự báo các thông số môi trường.
- **Đánh giá hiệu suất mô hình:** Sử dụng các chỉ số đánh giá thống kê (R², MAE, RMSE, MAPE) để kiểm tra độ chính xác và độ tin cậy của các mô hình. So sánh hiệu năng giữa các mô hình để tìm ra giải pháp tối ưu.
- **Phân tích kết quả và đề xuất giải pháp:** Dựa trên kết quả đánh giá, đưa ra các kết luận về khả năng ứng dụng của từng mô hình và đề xuất phương pháp tích hợp vào hệ thống điều khiển môi trường, cũng như các quy tắc điều chỉnh thông số môi trường một cách hợp lý.

1.6. Nội dung nghiên cứu

Đề tài tập trung giải quyết các nội dung nghiên cứu cốt lõi sau:

- **Nghiên cứu cơ sở lý thuyết và công nghệ AI ứng dụng:** Khảo sát và lựa chọn các thuật toán AI phù hợp nhất cho bài toán dự báo và điều chỉnh thông số môi trường trong chăn nuôi.
- **Phân tích dữ liệu và xây dựng mô hình:** Tiến hành phân tích sâu bộ dữ liệu IoT thực tế, lựa chọn các đặc trưng quan trọng, và xây dựng các mô hình dự báo chuỗi thời gian cho các thông số môi trường.
- **Thử nghiệm, đánh giá và so sánh mô hình:** Kiểm tra hiệu suất của các mô hình đã xây dựng trong điều kiện dữ liệu thực tế, đánh giá tính khả thi và so sánh ưu nhược điểm của từng phương pháp.
- **Định hướng phát triển hệ thống điều khiển thông minh:** Đề xuất các cải tiến, hướng tích hợp mô hình AI vào hệ thống điều khiển tự động và các hướng phát triển xa hơn cho nghiên cứu.

1.7. Ý nghĩa thực tiễn

Đề tài nghiên cứu này mang lại ý nghĩa thực tiễn quan trọng:

- **Đối với ngành chăn nuôi:** Góp phần giải quyết vấn đề ô nhiễm môi trường, đặc biệt là mùi hôi, từ đó cải thiện điều kiện làm việc cho người lao động và sức khỏe vật nuôi, hướng tới một nền chăn nuôi bền vững hơn.
- **Đối với cộng đồng:** Nâng cao chất lượng cuộc sống cho người dân sinh sống gần các khu vực chăn nuôi thông qua việc giảm thiểu mùi và các tác nhân gây ô nhiễm không khí.
- **Đối với khoa học công nghệ:** Đóng góp vào việc ứng dụng và phát triển các kỹ thuật AI trong lĩnh vực nông nghiệp thông minh tại Việt Nam, cung cấp một nghiên cứu điển hình về việc xử lý dữ liệu IoT phức tạp và so sánh các mô hình dự báo hiện đại.
- **Đối với quản lý nhà nước:** Cung cấp cơ sở khoa học và dữ liệu thực tiễn để các nhà quản lý có thể tham khảo trong việc xây dựng các chính sách và quy định về bảo vệ môi trường trong chăn nuôi.

Chương 2: Cơ sở lý thuyết

2.1. Dự báo chuỗi thời gian

Dự báo chuỗi thời gian là quá trình sử dụng dữ liệu trong quá khứ để dự đoán giá trị tương lai của một đại lượng biến đổi theo thời gian. Dữ liệu chuỗi thời gian thường được thu thập theo thứ tự thời gian đều đặn, ví dụ như theo giờ, ngày, tháng hoặc năm.

Một chuỗi thời gian (time-series) là một tập hợp các quan sát được ghi nhận theo trình tự thời gian. Mỗi điểm dữ liệu trong chuỗi đều gắn với một mốc thời gian cụ thể, và thứ tự thời gian đóng vai trò quan trọng trong phân tích. Khác với dữ liệu dạng bảng thông thường, việc hoán đổi thứ tự các quan sát trong chuỗi thời gian có thể làm mất đi ý nghĩa thống kê.

Các loại dữ liệu chuỗi thời gian bao gồm:

- **Chuỗi liên tục:** như nhiệt độ đo từng giây, tín hiệu ECG.
- **Chuỗi rời rạc:** như doanh số hàng ngày, lượt truy cập website theo giờ.
- **Chuỗi đơn biến:** chỉ có một biến được quan sát theo thời gian.
- **Chuỗi đa biến:** có nhiều biến thay đổi theo thời gian, ảnh hưởng lẫn nhau.

2.2. Phân tích dữ liệu chuỗi thời gian

Phương pháp phân tích dữ liệu chuỗi thời gian chia tách chuỗi thành 4 thành phần chính: level, trend, seasonality và noise. Trong đó:

- **Level:** Giá trị trung bình dài hạn, đóng vai trò nền tảng
- **Trend:** Xu hướng biến đổi tổng thể (tăng/giảm/ ổn định)
- **Seasonality:** Chu kỳ lặp lại đều đặn theo mùa/tuần/ngày
- **Noise:** Biến động ngẫu nhiên/tập âm không thể dự báo trước

Các thành phần này kết hợp theo 2 mô hình chính:

Mô hình cộng tính (Additive)

$$y(t) = \text{Level} + \text{Trend} + \text{Seasonality} + \text{Noise} \quad (1)$$

Áp dụng khi biến động có tính tuyến tính: Xu hướng ổn định, biến độ dao động mùa không đổi. Thích hợp cho dữ liệu có phạm vi biến động tương đối ổn định

Mô hình nhân tính (Multiplicative)

$$y(t) = \text{Level} * \text{Trend} * \text{Seasonality} * \text{Noise} \quad (2)$$

Áp dụng khi biến động có tính phi tuyến: Xu hướng thay đổi theo cấp số nhân, biến độ mùa vụ thay đổi. Phù hợp khi biến động tăng/giảm theo tỷ lệ phần trăm

Các dữ liệu chuỗi thời gian được lấy từ đời sống hằng ngày thường sẽ là sự kết hợp của cả 2 loại mô hình này.

Việc phân tích dữ liệu chuỗi thời gian nhằm tìm ra các thành phần đóng vai trò chủ đạo hơn trong việc ảnh hưởng tới kết quả của dữ liệu đầu ra, cũng như xác định các mô hình phù hợp nhất với mục tiêu của bài toán. Ví dụ:

- Chuỗi nồng độ khí thải: Thường có xu hướng tăng theo cấp số nhân (multiplicative)
 - Nhiệt độ không khí: Có cả yếu tố cộng tính (nhiệt độ trung bình) và nhân tính (biên độ dao động ngày đêm)
 - Độ ẩm: Thường thể hiện rõ tính chất cộng tính với biên độ ổn định

2.2.1. Level

Level (mức cơ bản), có thể hiểu đơn giản là giá trị còn sót lại của một chuỗi thời gian sau khi đã lược bỏ xu hướng, tính thời vụ và nhiễu. Về bản chất, mức cơ bản là giá trị trung bình dài hạn của một chuỗi dữ liệu, phần ổn định khi loại bỏ các biến động ngắn hạn, cũng như đại diện cho trạng thái cân bằng cơ bản của một hệ thống.

Mức cơ bản không chịu ảnh hưởng bởi yếu tố thời vụ hoặc chu kỳ (cyclical patterns), và thường được làm mượt (smoothing) trong quá trình tiền xử lý dữ liệu để loại bỏ nhiễu ngẫu nhiên.

2.2.2. Trend

Trend (xu hướng) có thể được quan sát khi tồn tại một đường dốc đi lên/đi xuống trong dữ liệu của một chuỗi thời gian. Có thể hiểu, xu hướng là tổng thể của dữ liệu theo thời gian (tăng/giảm/ ổn định). Xu hướng phản ánh tác động dài hạn của các yếu tố dữ liệu thành phần, cụ thể bao gồm:

- Tính dài hạn: Thể hiện thay đổi trong khoảng thời gian lớn
- Tính hướng: Có thể tuyến tính (linear) hoặc phi tuyến (non-linear)
- Tính bền vững: Duy trì trong nhiều chu kỳ quan sát

2.2.3. Seasonality

Seasonality (tính thời vụ) có thể được quan sát khi tồn tại một sự lặp lại về xu hướng giữa các khoảng thời gian cố định, được xác định bởi các yếu tố:

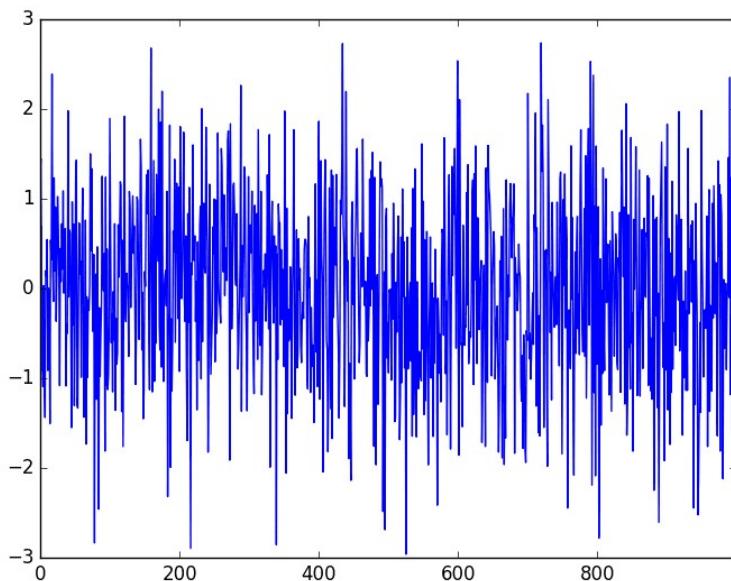
- Chu kỳ dài (năm): Thường liên quan đến các yếu tố khí hậu, mùa vụ
- Chu kỳ trung bình (tháng): Gắn với chu kỳ sản xuất hoặc hoạt động kinh tế
- Chu kỳ ngắn (tuần/ngày): Phản ánh nhịp điệu hoạt động của xã hội

Về bản chất, tính thời vụ thường có biên độ và độ dài chu kỳ tương đối ổn định, cũng như lặp lại với tần số có thể dự đoán được.

Tính thời vụ khác với chu kỳ (cycle). Tính thời vụ có độ dài và biên độ ổn định hơn, trong khi chu kỳ thường có khoảng cách và cường độ biến động không đều.

2.2.4. Noise

Trong dữ liệu chuỗi thời gian của ngũ cành bài toán này sẽ tồn tại thành phần nhiễu (noise), thường là nhiễu trắng (white noise). Nhịu trắng là một loại chuỗi thời gian không có một xu hướng hay tính thời vụ đặc thù nào có thể xác định được, gồm nhiều giá trị ngẫu nhiên độc lập và phân phối đều nhau.



Hình 2.1: Biểu đồ mẫu chỉ chứa nhiễu trắng

Chúng ta xác định nhiễu trắng khi các giá trị đo được thỏa mãn các điều kiện sau:

- Tính độc lập: Các giá trị đo không phụ thuộc lẫn nhau
- Phân phối đồng nhất: Tất cả các giá trị tuân theo một phân phối xác suất
- Đặc trưng thống kê:

- Giá trị trung bình bằng 0.
- Phương sai không đổi trên toàn chuỗi.
- Không có tương quan giữa các giá trị trong chuỗi.

Có thể khẳng định mọi dữ liệu chuỗi thời gian, đặc biệt dữ liệu lấy từ đời thực, sẽ tồn tại ít nhiều một lượng nhiễu nhất định.

2.3. Stationarity và differencing

Một chuỗi thời gian stationary (có tính dừng) khi giá trị trung bình, phương sai và tính tương quan tự động (auto-correlation) không thay đổi - bất chấp sự thay đổi về thời điểm quan sát. Đồng nghĩa với việc các chuỗi thời gian chứa xu hướng hoặc tính thời vụ luôn là không dừng, do các yếu tố này tạo ra biến đổi phụ thuộc thời gian. Ngược lại, nhiễu trắng có thể coi là có tính dừng, do giữ nguyên đặc trưng xác suất tại mọi thời điểm.

Đặc biệt, các chuỗi thời gian có dao động chu kỳ (không mang tính thời vụ xác định) vẫn có thể được coi là có tính dừng. Tính chất này tồn tại do độ dài chu kỳ không cố định khiến vị trí các đỉnh/đáy không thể dự đoán trước. Một chuỗi thời gian có tính dừng có:

- Xu hướng hội tụ về giá trị trung bình
- Phương sai đồng nhất
- Không tồn tại đặc trưng (pattern) theo thời gian

Ngoài việc một số mô hình như ARIMA yêu cầu chuỗi thời gian phải là chuỗi dừng, tức là các đặc trưng thống kê như kỳ vọng, phương sai không đổi theo thời gian, việc chuyển đổi chuỗi thời gian từ không dừng thành dừng còn có nhiều lợi thế khác trong việc triển khai mô hình dự báo, bao gồm:

- **Các đặc tính thống kê nhất quán:** Chuỗi dừng có giá trị trung bình, phương sai, và cấu trúc tự tương quan không đổi theo thời gian, giúp việc mô hình hóa thống kê đáng tin cậy hơn.
- **Giảm các mối quan hệ tương quan giả:** Chuỗi không dừng có thể cho thấy mối tương quan mạnh giữa các cột nhất định với nhau, tuy nhiên thường biến mất sau khi biến đổi, không chỉ vô nghĩa theo bản chất mà còn có thể làm sai lệch kết quả cuối cùng.
- **Hội tụ (convergence) mô hình tốt hơn:** Nhiều thuật toán hội tụ nhanh hơn và đáng tin cậy hơn với dữ liệu dừng.

Một trong những phương pháp phổ biến biến đổi chuỗi thời gian từ không dùng thành dùng chính là differencing. Có thể thấy, chuỗi thời gian thường có xu hướng tăng/giảm hoặc có yếu tố seasonality, khiến cho giá trị trung bình thay đổi theo thời gian.

Cách thực hiện:

- Sai phân bậc 1: $y'(t) = y(t) - y(t-1)$
- Nếu chuỗi vẫn không dùng → tiếp tục sai phân bậc 2 hoặc sai phân theo chu kỳ ($y(t) - y(t-s)$ với s là chu kỳ).

2.4. Về các tiêu chí đánh giá mô hình

2.4.1. Mean Absolute Error (MAE)

MAE đo lường độ lệch trung bình tuyệt đối giữa giá trị dự báo và giá trị thực tế. Đây là một chỉ số đơn giản nhưng hữu ích trong việc xác định mức sai số thực tế (theo đơn vị gốc).

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|, \quad (3)$$

y_i : giá trị thực tế

\hat{y}_i : giá trị dự báo

n : số lượng điểm dữ liệu

Cách đánh giá: MAE càng nhỏ, mô hình sẽ càng chính xác.

MAE dễ hiểu, ít nhạy cảm với outliers, tuy nhiên sẽ không phản ánh tốt mức độ lớn nhỏ của giá trị thực tế (không tương đối).

2.4.2. Mean Absolute Percentage Error (MAPE)

MAPE biểu thị sai số dự báo dưới dạng phần trăm, giúp dễ dàng so sánh giữa các chuỗi thời gian khác nhau.

$$\text{MAPE} = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (4)$$

Cách đánh giá: MAPE càng nhỏ, mô hình sẽ càng chính xác.

MAPE sẽ trực quan và dễ hình dung hơn so với MAE vì cho kết quả theo phần trăm, tuy nhiên sẽ không xác định được chính xác khi $y_i = 0$, cũng như rất nhạy cảm với các giá trị nhỏ.

2.4.3. Root Mean Squared Error (RMSE)

RMSE là đại lượng đánh giá sai số bình phương trung bình, giúp nhấn mạnh các sai số lớn do việc bình phương độ lệch.

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}, \quad (5)$$

\bar{y} : trung bình của giá trị thực tế

RMSE phản ánh rõ hơn ảnh hưởng của các dự báo sai lệch nghiêm trọng. Tuy nhiên sẽ dễ bị ảnh hưởng bởi outliers, và không trực quan bằng MAE nếu người đọc không quen với việc hiểu và so sánh các phép bình phương với nhau.

2.4.4. Coefficient of Determination (R^2 score)

R^2 đánh giá khả năng mô hình giải thích phuơng sai của dữ liệu. Chỉ số này dao động từ $-\infty$ đến 1:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \quad (6)$$

\bar{y} : trung bình của giá trị thực tế

Cách đánh giá:

$R^2 = 1$: mô hình dự báo hoàn hảo

$R^2 = 0$: mô hình không tốt hơn dự báo bằng giá trị trung bình

$R^2 < 0$: mô hình tệ hơn trung bình

R^2 sẽ đánh giá tổng thể nhất độ “phù hợp” của mô hình đối với dữ liệu cũng như yêu cầu bài toán đề ra, tuy nhiên sẽ dễ gây hiểu lầm nếu dùng sai bối cảnh, cũng như không phản ánh trực tiếp sai số.

Chương 3: Triển khai bài toán

3.1. Xử lý tập dữ liệu thô

Dữ liệu của bài khóa luận được lấy từ kết quả của bài nghiên cứu “Nghiên cứu phát triển hệ thống IOT theo dõi giám sát một số thông số môi trường trong trang trại chăn nuôi lợn” do SV Nguyễn Thành Đạt, dưới sự hướng dẫn của PGS. TS. Phạm Châu Thuỷ, thực hiện.

Xuất số liệu theo giai đoạn:

1. Đợt 1: 45 ngày của tháng 3 (số liệu đòn ánh)
2. Đợt 2 chia làm 3 đợt phụ
 - 2.1. Từ 25/05/2024 đến 05/06/2024
 - 2.2. Từ 06/06/2024 đến 25/06/2024 (liều lượng phun 250g/500 lít (0.5g/lít)
 - 2.3. Từ 26/06/2024 đến 02/07/24 (liều lượng phun 500g/500l (1g/lít)
 - 2.4. Từ 03/07/24 đến 31/07/2024 (liều lượng 750g/500l (1.5g/lít)

Hình 3.1: Phân chia dữ liệu gốc theo giai đoạn

Dữ liệu được chia thành 2 đợt như trên hình, tuy nhiên trong giới hạn của bài khóa luận này sẽ chỉ sử dụng lượng dữ liệu của đợt 2, bao gồm:

- Từ 6/6/2024 đến 25/6/2024 (liều lượng phun 250g/500l hay 0.5g/l)
- Từ 26/6/2024 đến 2/7/2024 (liều lượng phun 500g/500l hay 1g/l)
- Từ 3/7/2024 đến 31/7/2024 (liều lượng phun 750g/500l hay 1.5g/l)

(Chất được phun ở đây là chất khử mùi, được pha loãng với nước và phun dưới dạng mưa theo tỷ lệ trên).

Ngoài ra, nhằm mục đích cung cấp thêm ngữ cảnh cho dữ liệu chuỗi thời gian này, theo lịch trình chăn nuôi tại trang trại, khoảng 4h đến 5h sáng, lợn bắt đầu thức dậy và hoạt động, thải chất thải. Vào 7h sáng, lợn được cho ăn bữa đầu. Sau đó, người chăn nuôi tiến hành vệ sinh chuồng trại vào lúc 8h đến khoảng 8h30, lúc này lượng chất thải qua một đêm được dọn sạch và lượng thức ăn cũng được lợn tiêu thụ bớt lại. Tương tự, vào 13h45, lợn được tiếp tục cho ăn bữa chiều. Cuối cùng, vào khoảng 19h - 20h, TVOC lại bắt đầu gia tăng do lợn bắt đầu hoạt động nhiều thải

chất thải, sau đó nhờ hệ thống thông gió và thoát nước của trang trại cũng như việc lợn bắt đầu đi ngủ khiến nồng độ khí thải có xu hướng giảm dần. Có thể khẳng định, các dữ liệu liên quan đến nồng độ khí thải như nồng độ NH₃, H₂S và CH₄ sẽ có xu hướng di chuyển tương ứng với lịch trình trong ngày trên của trang trại.

Dữ liệu được cung cấp có dạng 3 tệp .csv, biểu diễn bằng bảng tính excel, sử dụng tính năng Text to Columns có sẵn:

Timestamp250,Date,time250,ccseCO2250,ccsTVCO250,scdCO2250,scdTEMP250,scdHUMID250,H2S_concentration250,NH3_concentration250,NH3_boardtemp250,CH4ppm250,CH4RAVppm250,171764000000,6/6/2024 10:00,552,23,496,27,49565125,88,31329346,0,29,28157405,0,917524338,29,39808012,0,15,91069077,,1719334802140,6/26/2024 0:00,400,0,446,29,23934937,89,8147593,171764000000,6/6/2024 10:01,556,23,505,27,14050293,91,38336182,0,29,49455833,0,917524338,29,49455833,0,16,58226395,,1719334802077,6/26/2024 0:01,400,0,471,29,1962476,89,77053467,171764000000,6/6/2024 10:02,614,32,510,27,32208252,90,87066655,0,29,60121346,0,917524338,28,70798402,0,17,39607239,,1719334602023,6/26/2024 0:02,400,0,474,29,20463562,89,86053467,171764000000,6/6/2024 10:04,614,32,499,27,56240845,90,0177002,0,29,60121346,0,917524338,29,8148632,0,17,71839188,,1719335041977,6/26/2024 0:04,400,0,460,29,19128418,89,85137939,,171764000000,6/6/2024 10:05,559,30,494,27,79472251,89,08849394,0,29,8148632,0,917524338,29,8148632,0,17,62744522,,171933512191,6/26/2024 0:05,400,0,466,29,18327332,89,96867207,,171764000000,6/6/2024 10:06,592,29,490,27,97096252,88,41552734,0,29,92185974,0,917524338,30,12619423,0,18,17675209,,171933521849,6/26/2024 0:06,400,0,475,29,23667909,89,99326361,171764000000,6/6/2024 10:07,578,27,490,28,12316985,0,29,0,0,35986,0,30,0,2897072,0,917524338,30,2897072,0,18,441143,,171933521803,6/26/2024 0:07,400,0,466,29,20196533,90,05279541,171764000000,6/6/2024 10:09,614,32,494,28,21395874,87,63580322,0,30,13619423,0,917524338,30,24353409,0,18,52280994,,1719335361737,6/26/2024 0:09,400,0,465,29,19662476,90,1138305,171764000000,6/6/2024 10:10,614,32,476,28,30474854,87,30621333,0,30,0,2897072,0,917524338,30,3509922,0,19,27952766,,171933541680,6/26/2024 0:10,400,0,465,29,20196533,90,17028803,171764000000,6/6/2024 10:12,623,33,479,28,43025208,87,63776551,0,19,77188466,,17193355217613,6/26/2024 0:12,400,0,463,29,20463562,89,14892571,171764000000,6/6/2024 10:13,572,26,474,28,47831726,86,56158447,0,30,24353409,0,917524338,30,45856857,0,18,32336998,,1719335601545,6/26/2024 0:13,400,0,456,28,22599792,90,04058833,171764000000,6/6/2024 10:14,581,27,474,28,53172302,86,33117676,0,30,3509922,0,19,31502532,,171933561948,6/26/2024 0:14,400,0,456,29,22599792,90,04058833,,171764000000,6/6/2024 10:16,497,14,471,28,56910706,86,02905273,0,30,3509922,0,19,917524338,30,5662632,0,18,69443292,,171933561431,6/26/2024 0:16,400,0,443,29,17259216,89,91241455,,171764000000,6/6/2024 10:17,539,30,492,27,60916138,,171933561431,6/26/2024 0:17,400,0,447,29,19662476,89,83917236,171764000000,6/6/2024 10:18,559,30,469,28,615188599,85,22246094,0,30,45856857,0,917524338,30,5662632,0,19,45441437,,1719335921304,6/26/2024 0:18,400,0,449,29,19662476,89,94903564,171764000000,6/6/2024 10:20,581,27,466,28,67858876,85,46600342,0,30,6740818,0,19,917524338,30,6740818,0,18,62091446,,1719336091176,6/26/2024 0:20,400,0,446,29,23400879,90,04211426,171764000000,6/6/2024 10:21,483,12,466,28,68927002,85,18066406,0,30,5662632,0,917524338,30,6740818,0,18,62091446,,1719336091176,6/26/2024 0:21,400,0,454,29,21798706,90,15655518,,171764000000,6/6/2024 10:22,529,19,468,28,72398376,85,11525353,0,18,61025671,,1719336161118,6/26/2024 0:22,400,0,459,29,20463562,89,14892571,171764000000,6/6/2024 10:24,517,17,465,28,75602722,84,883311768,0,30,5662632,0,917524338,30,89008141,0,19,23738671,,171933621042,6/26/2024 0:24,400,0,463,29,19395447,90,08636475,,171764000000,6/6/2024 10:25,501,18,461,28,7827301,84,83346558,0,30,6740818,0,19,917524338,30,89008141,0,19,99448866,,171933622098,6/26/2024 0:25,400,0,461,29,17793274,90,13977051,,171764000000,6/6/2024 10:26,501,15,465,28,83346558,0,30,65423384,0,30,78201866,0,917524338,30,89008141,0,19,63128801,,1719336400914,6/26/2024 0:26,400,0,479,29,13787842,90,05432121,171764000000,6/6/2024 10:27,482,13,467,28,84414473,84,8236084,0,30,6740818,0,19,917524338,30,89008141,0,19,32497371,,1719336400914,6/26/2024 0:27,400,0,447,29,19662476,89,171764000000,6/6/2024 10:28,519,15,463,28,84948738,84,8236084,0,30,6740818,0,19,917524338,30,89008141,0,20,28382111,,17193365607958,6/26/2024 0:28,400,0,449,29,19662476,89,19240792,90,35339355,,171764000000,6/6/2024 10:30,502,15,467,28,87867134,84,67470227,84,67470227,0,19,917524338,30,98826622,0,19,61025671,,17193367206597,6/26/2024 0:30,400,0,483,29,12719721,90,66772446,171764000000,6/6/2024 10:32,493,14,465,28,90022278,84,65818134,0,30,78201866,0,917524338,30,98826622,0,19,74842453,,17193367206597,6/26/2024 0:32,400,0,503,29,12452698,90,05383811,171764000000,6/6/2024 10:34,443,6,460,28,98300171,84,46807861,0,30,89008141,0,917524338,31,1065731,0,19,9958482,,1719336800528,6/26/2024 0:33,400,0,491,29,08180237,91,10870361,,171764000000,6/6/2024 10:37,473,11,460,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:37,400,0,486,29,11117554,91,14427109,,171765000000,6/6/2024 10:37,473,11,460,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:37,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:40,481,12,466,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:40,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:41,444,6,460,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:41,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:42,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:42,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:44,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:44,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:46,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:46,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:48,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:48,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:50,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:50,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:52,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:52,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:54,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:54,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:56,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:56,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 10:58,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:58,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:00,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:02,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:04,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:06,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:08,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:10,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:12,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:14,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:16,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:18,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:20,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:22,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:24,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:26,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:28,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:30,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:32,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:34,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:36,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:38,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:40,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:42,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:44,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:46,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:48,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:50,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:52,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:54,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:56,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 11:58,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 12:00,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 12:02,481,12,470,29,03106689,83,94775391,0,1065731,0,19,76705933,,1719336800528,6/26/2024 0:59,400,0,486,29,11117554,91,14427109,,171764000000,6/6/2024 12:04,481,12,470,29,03106689,83,94775391,0,

Nồng độ khí CO₂ do cảm biến CCS đo được.

- Đơn vị: ppm (phần triệu).
- Mục đích: Ghi lại nồng độ CO₂ trong không khí.

4. **ccsTVOC**

Tổng hợp các hợp chất hữu cơ dễ bay hơi (TVOC) do cảm biến CCS đo được.

- Đơn vị: ppm.
- Mục đích: Theo dõi mức độ ô nhiễm không khí do TVOC.

5. **scdCO2**

Nồng độ CO₂ do cảm biến SCD (có thể là SCD41) đo.

- Đơn vị: ppm.
- Mục đích: Ghi lại nồng độ CO₂ với độ chính xác cao hơn cảm biến CCS.

6. **scdTEMP**

Nhiệt độ đo được từ cảm biến SCD.

- Đơn vị: °C.
- Mục đích: Theo dõi nhiệt độ môi trường.

7. **scdHUMID**

Độ ẩm đo từ cảm biến SCD.

- Đơn vị: %RH (độ ẩm tương đối).
- Mục đích: Theo dõi độ ẩm môi trường.

8. **H2S_concentration**

Nồng độ khí hydro sulfide (H₂S), được đo bằng cảm biến Dfrobot Gravity.

- Đơn vị: ppm.
- Mục đích: Theo dõi nồng độ khí độc H₂S.
- Tình trạng dữ liệu: Tất cả các giá trị đều là 0.

9. **H2S_boardtemp**

Nhiệt độ của bo mạch cảm biến H₂S.

- Đơn vị: °C.
- Mục đích: Giám sát nhiệt độ hoạt động của cảm biến.

10. NH₃_concentration

Nồng độ khí amoniac (NH₃), có thể đo bằng cảm biến Dfrobot Gravity.

- Đơn vị: ppm.
- Mục đích: Theo dõi khí NH₃ trong môi trường.
- Tình trạng dữ liệu: Tất cả các giá trị đều nằm trong khoảng ~0.9175.

11. NH₃_boardtemp

Nhiệt độ của bo mạch cảm biến NH₃.

- Đơn vị: °C.
- Mục đích: Giám sát nhiệt độ của cảm biến NH₃.

12. CH4ppm

Nồng độ khí metan (CH₄), có thể đo bằng cảm biến MQ4.

- Đơn vị: ppm.
- Mục đích: Theo dõi lượng khí metan.
- Tình trạng dữ liệu: Tất cả các giá trị đều là 0.

13. CH4RAWppm

Giá trị thô của nồng độ CH₄ trước khi xử lý.

- Đơn vị: ppm.
- Mục đích: Cung cấp số liệu đầu ra gốc từ cảm biến metan.

14. HCHOmg/m3

Nồng độ formaldehyde (HCHO) trong không khí.

- Đơn vị: mg/m³.
- Mục đích: Theo dõi khí độc HCHO.
- Tình trạng dữ liệu: Tất cả các dòng dữ liệu đều trống.

15. TVOCppm

Nồng độ tổng hợp của các hợp chất hữu cơ dễ bay hơi.

- Đơn vị: ppm.
- Mục đích: Đo tổng lượng chất hữu cơ dễ bay hơi.
- Tình trạng dữ liệu: Tất cả các dòng dữ liệu đều trống.

3.1.1. Trực quan hóa dữ liệu

Nhằm mục đích dễ nhận diện hình dạng (shape) của dữ liệu hơn, cũng như tìm và tìm cách khắc phục các bất cập liên quan đến việc dữ liệu thiếu sót/sai sót, chúng ta cần trực quan hóa chuỗi thời gian này bằng các thư viện như numpy hay pandas:

• **Tính chất của dữ liệu chuỗi thời gian này là như thế nào?** (là chuỗi rời rạc/liên tục, chuỗi đơn biến/đa biến?, mức cơ bản/xu hướng/tính thời vụ/nhiều của chuỗi như thế nào?)

• **Chuỗi thời gian này có những vấn đề gì?** Các vấn đề nổi bật bao gồm phần nổi có thể dễ dàng nhìn thấy bằng mắt thường (các cột bị thiếu dữ liệu, chênh lệch lượng dữ liệu giữa các liều lượng với nhau...) cũng như phần chìm mà chỉ có thể thấy được một khi đã trực quan hóa thành công (nhiều, không có xu hướng hay tính thời vụ cụ thể)

• **Cần làm gì để xử lý chuỗi thời gian này** nhằm khắc phục các sai sót trong cách ghi lại dữ liệu của hệ thống IoT, cũng như đảm bảo việc triển khai mô hình ít lỗi nhất có thể?

```
for label, fname in files.items():

    print(f"\n===== {fname} (nồng độ {label}) =====")

    file_path = os.path.join(base_dir, fname)
    df = pd.read_csv(file_path, encoding='utf-8', sep=';')

    print(f"Số dòng dữ liệu: {len(df)}")
    print(f"Số cột: {len(df.columns)}")

    # Phát hiện các cột toàn NaN hoặc toàn 0
    all_nan_cols = df.columns[df.isna().all()].tolist()
    all_zero_cols = df.columns[(df == 0).all()].tolist()

    print(f"Cột toàn NaN: {all_nan_cols}")
    print(f"Cột toàn 0: {all_zero_cols}")

    # Thống kê số lượng giá trị NaN và 0 trên từng cột
    nan_counts = df.isna().sum()
    zero_counts = (df == 0).sum()

    print("Top 5 cột nhiều NaN nhất:")
    print(nan_counts.sort_values(ascending=False).head())
    print("Top 5 cột nhiều giá trị 0 nhất:")
```

```

print(zero_counts.sort_values(ascending=False).head())
# Kiểm tra các cột có số lượng giá trị duy nhất thấp (dẽ nghi ngờ lỗi)
nunique = df.nunique(dropna=True)
few_unique_cols = nunique[nunique <= 2].index.tolist()
print(f"Cột có <=2 giá trị duy nhất (có thể là biến nhị phân hoặc lỗi): {few_unique_cols}")

# Danh sách các biến số (loại trừ cột thời gian, id...)
numerical_cols = [col for col in df.columns if pd.api.types.is_numeric_dtype(df[col]) and col not in all_nan_cols and col not in all_zero_cols]

if 'Timestamp' in numerical_cols: numerical_cols.remove('Timestamp')
if 'Date time' in numerical_cols: numerical_cols.remove('Date time')

# Chuyển đổi ngày giờ nếu có
if 'Date time' in df.columns:
    df['Date time'] = pd.to_datetime(df['Date time'], errors='coerce')

# 1. Violinplot
if len(numerical_cols) > 0:
    plt.figure(figsize=(max(10, len(numerical_cols)*1.2), 6))
    sns.violinplot(data=df[numerical_cols], inner='box', cut=0)
    plt.title(f'Violinplot tất cả biến số - Nồng độ {label}')
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.savefig(os.path.join(export_dir, f'violinplot_all_{label}.png'))
    plt.close()

else:
    print(f"[CẢNH BÁO] Không có biến số nào để vẽ violinplot cho nồng độ {label}.")

# 2. Boxplot
if len(numerical_cols) > 0:
    plt.figure(figsize=(max(10, len(numerical_cols)*1.2), 6))
    sns.boxplot(data=df[numerical_cols])
    plt.title(f'Boxplot tất cả biến số - Nồng độ {label}')
    plt.xticks(rotation=45, ha='right')

```

```

plt.tight_layout()
plt.savefig(os.path.join(export_dir, f'boxplot_all_{label}.png'))
plt.close()

else:

    print(f"[CẢNH BÁO] Không có biến số nào để vẽ boxplot cho nồng độ {label}.")

# 3. Histogram (subplots)

if len(numerical_cols) > 0:

    n_cols = 4
    n_rows = int(np.ceil(len(numerical_cols)/n_cols))

    fig, axes = plt.subplots(n_rows, n_cols, figsize=(n_cols*4,
n_rows*3))

    axes = axes.flatten()

    for i, col in enumerate(numerical_cols):

        sns.histplot(df[col], bins=30, kde=True, ax=axes[i],
color='steelblue')

        axes[i].set_title(col)

        for j in range(i+1, len(axes)):

            axes[j].axis('off')

    plt.suptitle(f'Histogram các biến số - Nồng độ {label}', fontsize=16)
    plt.tight_layout(rect=[0, 0, 1, 0.97])
    plt.savefig(os.path.join(export_dir, f'histogram_all_{label}.png'))
    plt.close()

else:

    print(f"[CẢNH BÁO] Không có biến số nào để vẽ histogram cho nồng độ {label}.")

# Heatmap tương quan

if len(numerical_cols) > 1:

    plt.figure(figsize=(max(8, len(numerical_cols)), max(6,
len(numerical_cols)*0.6)))

    corr = df[numerical_cols].corr()

    sns.heatmap(corr, annot=True, fmt=".2f", cmap='coolwarm', cbar=True)
    plt.title(f'Heatmap ma trận tương quan - Nồng độ {label}')
    plt.tight_layout()

```

```

plt.savefig(os.path.join(export_dir, f'corr_heatmap_{label}.png'))
plt.close()

else:

    print(f"[CẢNH BÁO] Không đủ biên số để vẽ heatmap tương quan cho nồng
độ {label}.")

```

Kết quả cho ra có dạng như sau:

```

===== 250500.csv (nồng độ 250g/500l) =====

Số dòng dữ liệu: 11839
Số cột: 15
Cột toàn NaN: ['HCHOmg/m3', 'TVOCppm']
Cột toàn 0: ['H2S_concentration']

Top 5 cột nhiều NaN nhất:
HCHOmg/m3           11839
TVOCppm              11839
H2S_boardtemp         8
NH3_boardtemp         8
Timestamp             0
dtype: int64

Top 5 cột nhiều giá trị 0 nhất:
H2S_concentration    11839
CH4ppm                10663
ccsTVOC               4998
Timestamp              0
Date time              0
dtype: int64

Cột có <=2 giá trị duy nhất (có thể là biến nhị phân hoặc lỗi):
['H2S_concentration', 'HCHOmg/m3', 'TVOCppm']

===== 500500.csv (nồng độ 500g/500l) =====

Số dòng dữ liệu: 8646
Số cột: 15
Cột toàn NaN: ['HCHOmg/m3', 'TVOCppm']
Cột toàn 0: ['ccsTVOC', 'H2S_concentration']

Top 5 cột nhiều NaN nhất:

```

```

HCHOmg/m3      8646
TVOCppm        8646
Timestamp       0
Date time       0
ccseCO2         0
dtype: int64

Top 5 cột nhiều giá trị 0 nhất:
ccsTVOC          8646
H2S_concentration 8646
CH4ppm           4615
Timestamp         0
Date time         0
dtype: int64

Cột có <=2 giá trị duy nhất (có thể là biến nhị phân hoặc lõi): ['ccseCO2',
'ccsTVOC', 'H2S_concentration', 'HCHOmg/m3', 'TVOCppm']

===== 750500.csv (nồng độ 750g/500l) =====

Số dòng dữ liệu: 26315
Số cột: 15

Cột toàn NaN: ['HCHOmg/m3', 'TVOCppm']
Cột toàn 0: ['ccsTVOC', 'H2S_concentration']

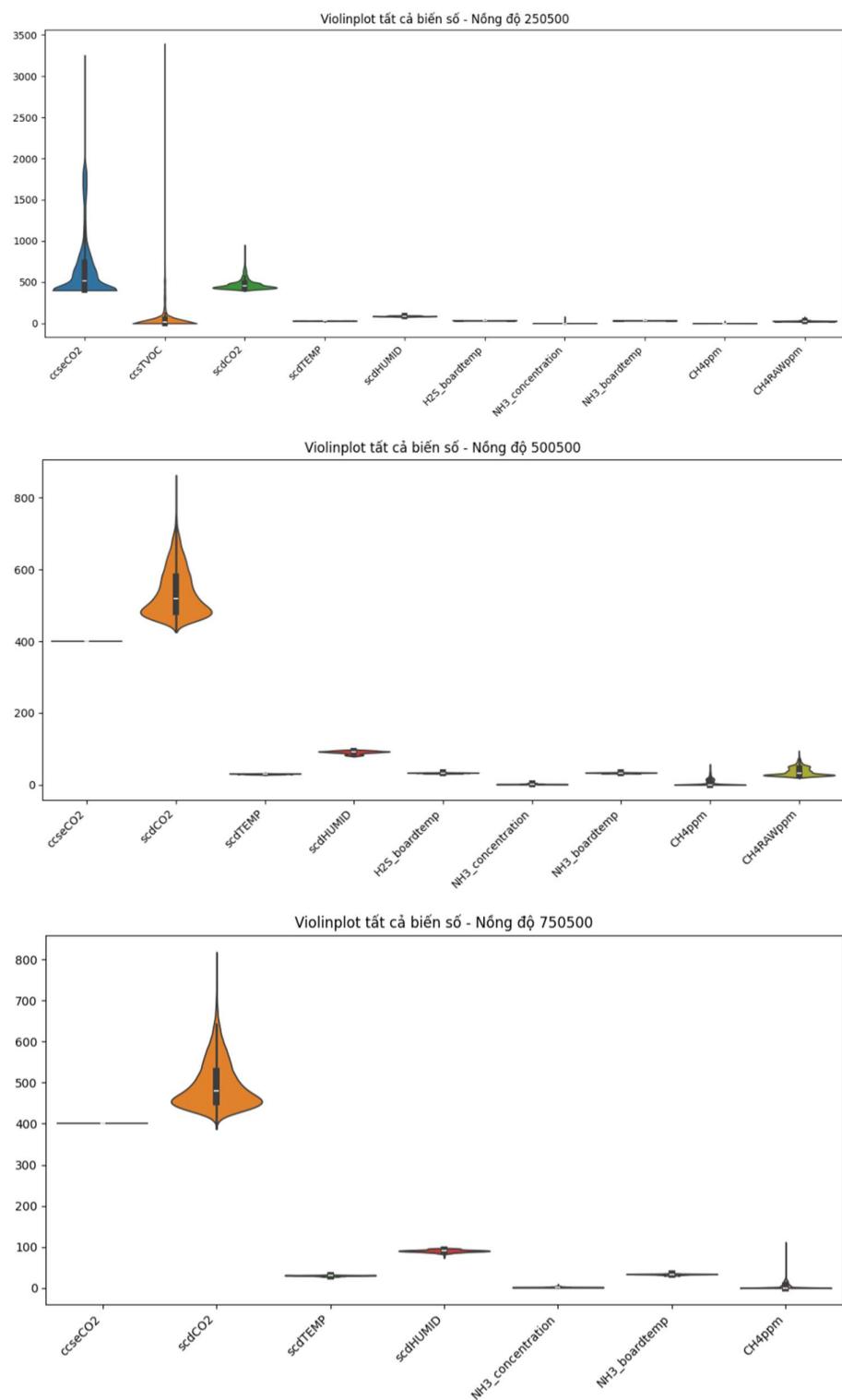
Top 5 cột nhiều NaN nhất:
HCHOmg/m3      26315
TVOCppm        26315
CH4ppm          1
Timestamp       0
Date time       0
dtype: int64

Top 5 cột nhiều giá trị 0 nhất:
ccsTVOC          26315
H2S_concentration 26315
CH4ppm           14059
Timestamp         0
Date time         0

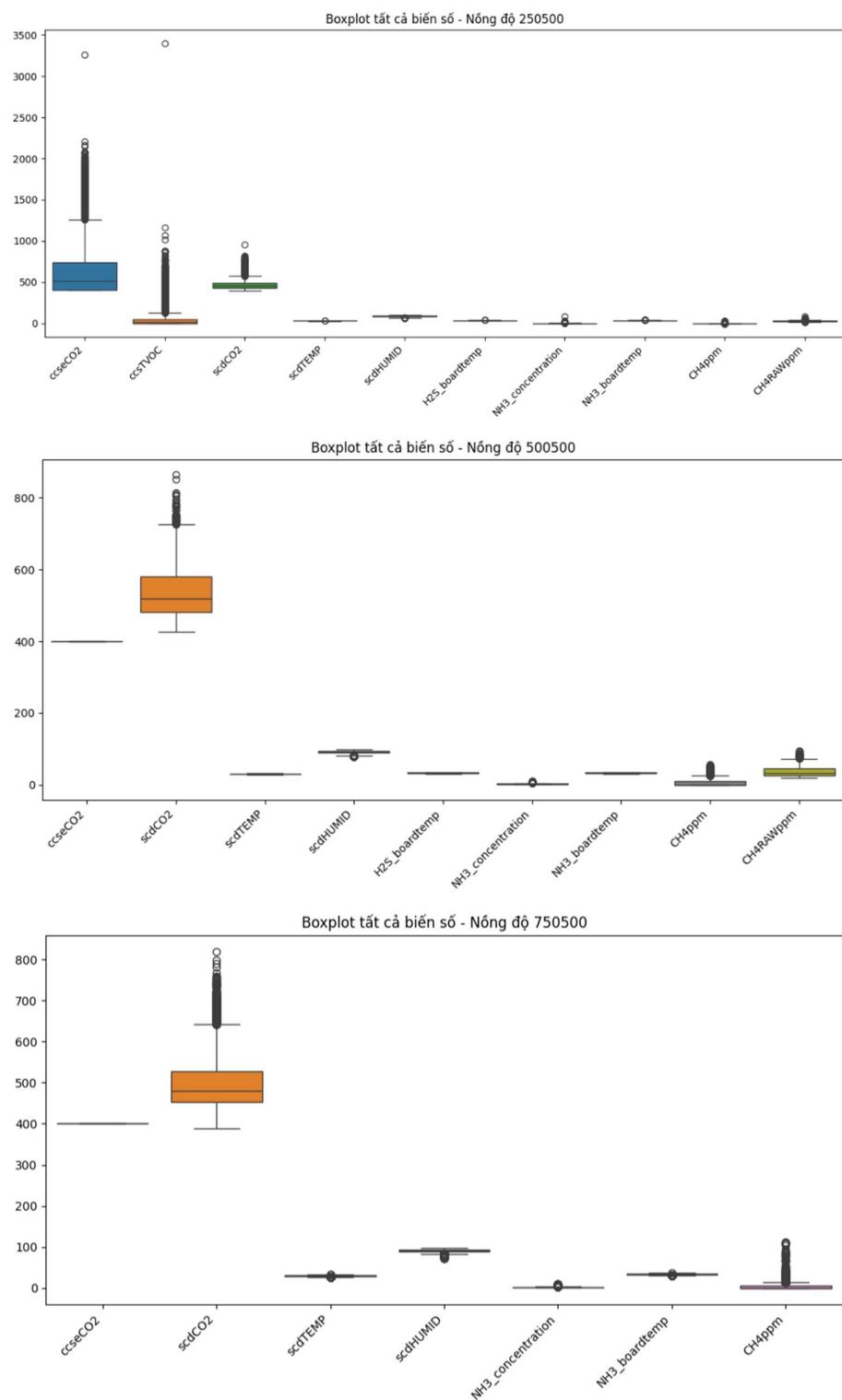
```

```
dtype: int64
```

```
Cột có <=2 giá trị duy nhất (có thể là biến nhị phân hoặc lõi): ['ccseCO2',  
'ccsTVOC', 'H2S_concentration', 'HCHOmg/m3', 'TVOCppm']
```

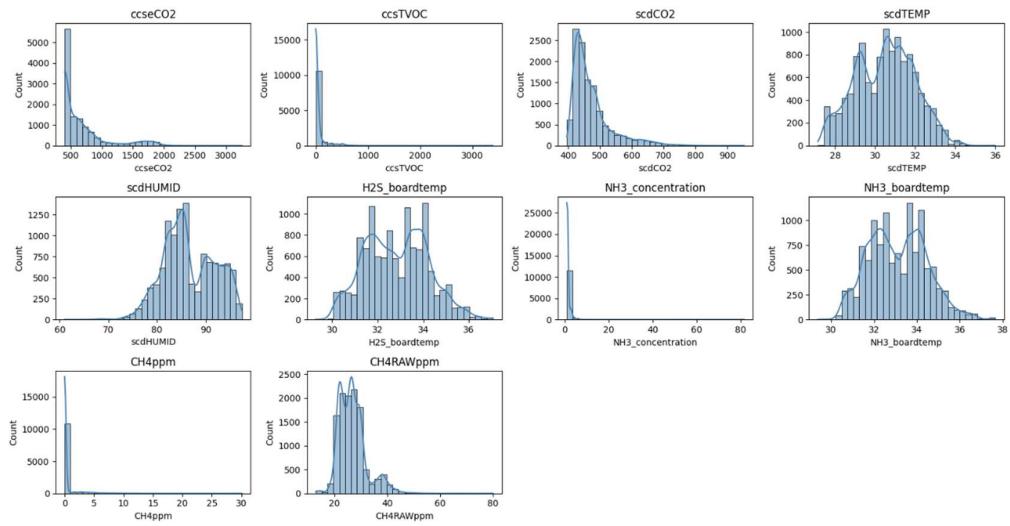


Biểu đồ 3.1: Violinplot của dữ liệu chưa tiền xử lý

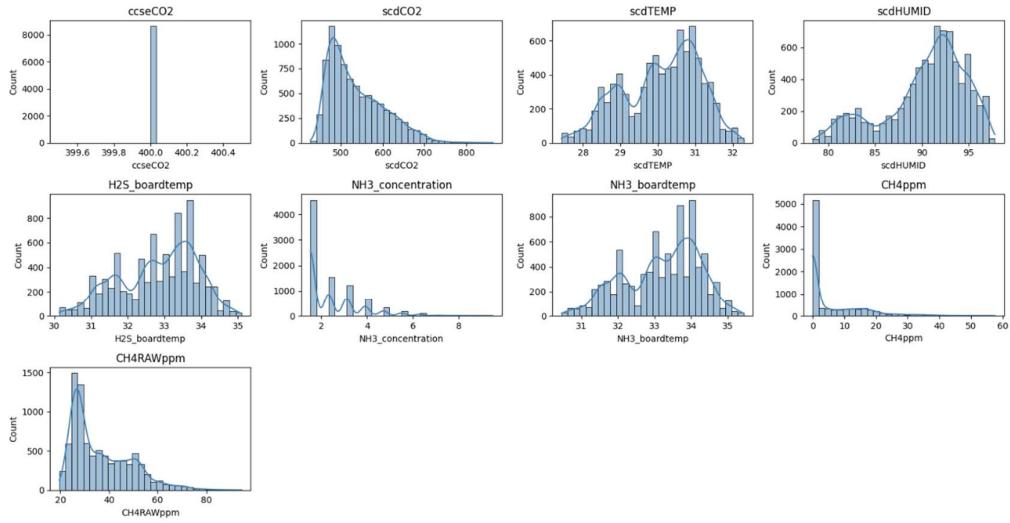


Biểu đồ 3.2: Boxplot của dữ liệu chưa tiền xử lý

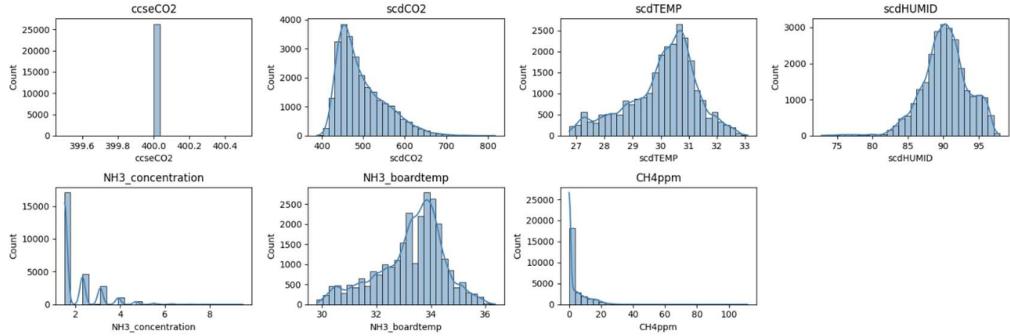
Histogram các biến số - Nồng độ 250500



Histogram các biến số - Nồng độ 500500

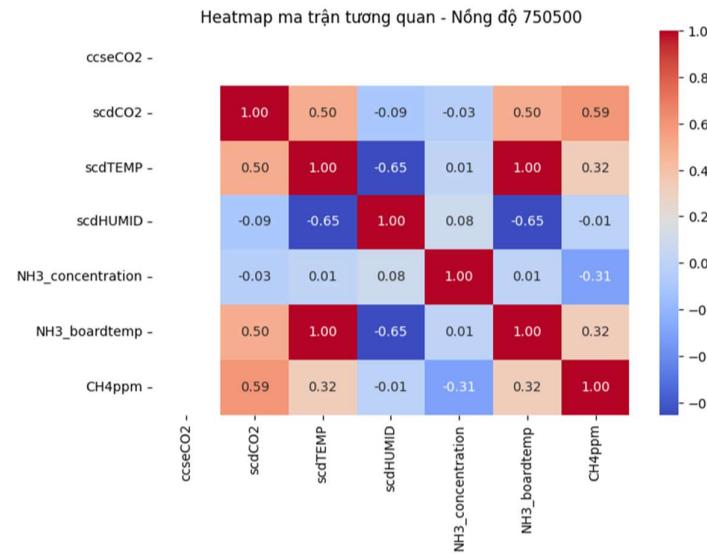
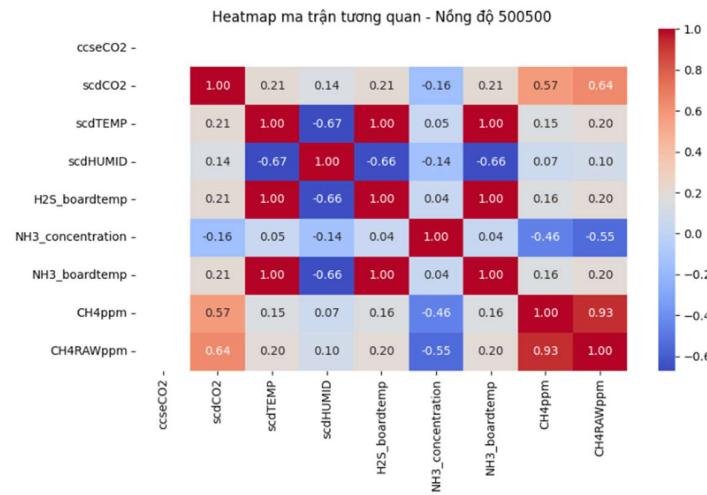
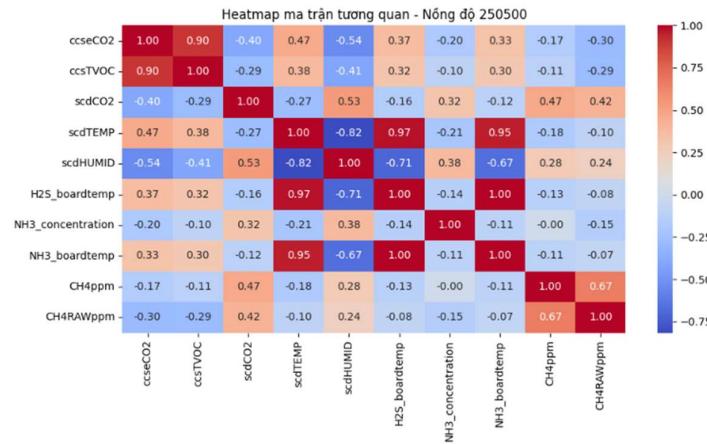


Histogram các biến số - Nồng độ 750500



Biểu đồ 3.3: Histogram của dữ liệu chưa tiền xử lý

Bảng 3.1: Heatmap ma trận tương quan của dữ liệu chưa tiền xử lý.



Dựa trên các biểu đồ phân tích và thông tin số liệu được cung cấp, chúng ta có thể đưa ra phân tích chi tiết hơn về bộ dữ liệu chuỗi thời gian này:

Phân tích tổng quát:

- Dữ liệu bao gồm 3 tập dữ liệu với các nồng độ khác nhau: (250500.csv) 250g/500l, (500500.csv) 500g/500l, và (750500.csv) 750g/500l, với số lượng dữ liệu khác nhau đáng kể (tương ứng 11839, 8646, và 26315 dòng).

- Chuỗi thời gian này có tính đa biến, rời rạc và không dừng với sự hiện diện của xu hướng, thay đổi về mức cơ bản và chu kỳ lên xuống định kỳ.

Vấn đề chất lượng dữ liệu:

- Hai cột 'HCHOmg/m³' và 'TVOCppm' hoàn toàn là giá trị NaN trong cả ba tập dữ liệu, khiến chúng không có giá trị phân tích.

- Cột 'H2S_concentration' toàn bộ là giá trị 0 trong cả ba bộ dữ liệu. Trong khi đó, 'ccsTVOC' toàn bộ là giá trị 0 trong hai tập dữ liệu 500500 và 750500.

- Nhiều cột chỉ có ≤2 giá trị duy nhất, đặc biệt trong các tập dữ liệu 500500 và 750500, có 5 cột như vậy, cho thấy các biến này có thể là nhị phân hoặc có vấn đề về thu thập dữ liệu.

- Biểu đồ boxplot và violin plot cho thấy nhiều điểm ngoại lai trong một số biến, đặc biệt là 'ccseCO2' ở nồng độ 250500 và 'scdCO2' ở các nồng độ 500500 và 750500.

Đặc điểm phân phối:

- Nhiều biến có phân phối lệch phải rõ rệt, với đa số giá trị tập trung ở phía thấp và đuôi dài về phía các giá trị cao (đặc biệt là các biến CO2).

- Một số biến như 'scdTEMP' và 'scdHUMID' thể hiện phân phối đa đỉnh, gợi ý có thể tồn tại các trạng thái hoạt động khác nhau hoặc điều kiện môi trường thay đổi.

- Có tỷ lệ giá trị 0 cao (90% ở nồng độ 250500, 53% ở nồng độ 500500, và 53% ở nồng độ 750500), có thể liên quan đến ngưỡng phát hiện của cảm biến.

Mối tương quan – correlation:

- Heatmap ma trận tương quan cho thấy một số cặp biến có tương quan cực cao (gần 1.0), đặc biệt là giữa: 'scdTEMP', 'H₂S_boardtemp' và 'NH₃_boardtemp', cũng như 'CH4ppm' và 'CH4RAWppm'

- Có tương quan âm mạnh giữa 'scdHUMID' và nhiệt độ ('scdTEMP', 'H₂S_boardtemp', 'NH₃_boardtemp'), cho thấy mối quan hệ nghịch đảo giữa nhiệt độ và độ ẩm.

- Một số mối tương quan thay đổi giữa các nồng độ khác nhau, cho thấy mối quan hệ giữa các biến có thể phụ thuộc vào điều kiện thí nghiệm.

Thay đổi theo nồng độ

- Ở nồng độ 250g/500l, cột 'ccseCO2' có phạm vi rộng và nhiều giá trị phân tán; ở nồng độ 500g/500l và 750g/500l, cột chỉ có một giá trị duy nhất (400).
- Ở nồng độ 500g/500l và 750g/500l, cột 'scdCO2' có giá trị trung bình cao hơn và phân phối rộng hơn so với nồng độ 250g/500l.

3.1.2. Tiết kiệm tiền xử lý dữ liệu

Tiết kiệm dữ liệu chia thành các bước như sau:

1. **Kiểm tra** hàng trùng lặp, kiểu dữ liệu không nhất quán, hoặc giá trị không thể (ví dụ: nồng độ âm).
2. **Xử lý** các giá trị thiếu, toàn 0:
 - Điền các giá trị còn thiếu giữa các giá trị hợp lệ (trung bình, trung vị, nội suy/interpolation hay điền tới, lùi/forward, backward fill).
 - Loại bỏ các cột có quá nhiều giá trị thiếu/toàn 0.
 - Do lượng dữ liệu ghi nhận được ở liều lượng 750g/500l nhiều hơn đáng kể so với 2 liều lượng còn lại => loại bỏ các giá trị NaN/toàn 0 của 2 liều lượng kia sau khi hợp nhất 3 tập dữ liệu này lại.
3. **Tìm và xử lý** ngoại lai (outlier) bằng z-score hoặc IQR.
4. **Biến đổi chuỗi** thành chuỗi dùng nhằm loại bỏ tính thời vụ, xu hướng và nhiễu bằng phương pháp differencing hoặc detrending.
5. **Chuẩn hóa** (normalize) dữ liệu nhằm đưa về cùng một thang so sánh.

```
for label, file_path in input_files.items():

    print(f"\n===== Đang xử lý nồng độ {label} =====")

    df = pd.read_csv(file_path, sep=';', encoding='utf-8', low_memory=False)

    print(f"Số dòng ban đầu: {len(df)}")

    # 1. Kiểm tra trùng lặp, kiểu dữ liệu không nhất quán, giá trị âm
    df = df.drop_duplicates()

    print(f"Số dòng sau khi loại trùng lặp: {len(df)}")

    # Chuyển các cột số sang numeric, cột thời gian sang datetime

    if 'Date time' in df.columns:
        df['Date time'] = pd.to_datetime(df['Date time'], errors='coerce')
```

```

numerical_cols = [col for col in df.columns if col not in ['Timestamp',
'Date    time'] and pd.api.types.is_numeric_dtype(pd.to_numeric(df[col],
errors='coerce'))]

for col in numerical_cols:
    df[col] = pd.to_numeric(df[col], errors='coerce')
    # Loại bỏ giá trị âm
    df.loc[df[col] < 0, col] = np.nan

# 2. Xử lý giá trị thiếu, toàn 0
nan_counts = df.isna().sum()
zero_counts = (df[numerical_cols] == 0).sum()
# Điện thiếu (nội suy, ffill, bfill, median)
for col in numerical_cols:
    df[col] = df[col].interpolate(method='linear',
limit_direction='both')
    df[col] = df[col].ffill().bfill()
    df[col] = df[col].fillna(df[col].median())
# Loại bỏ cột có >50% giá trị thiếu/toàn 0
threshold = 0.5
cols_to_drop = []
for col in numerical_cols:
    missing_ratio = nan_counts[col] / len(df)
    zero_ratio = (df[col] == 0).sum() / len(df)
    if missing_ratio > threshold or zero_ratio > threshold:
        cols_to_drop.append(col)
if cols_to_drop:
    print(f"Các cột bị loại bỏ do quá nhiều thiêu/0: {cols_to_drop}")
    df = df.drop(columns=cols_to_drop)
    numerical_cols = [col for col in numerical_cols if col not in
cols_to_drop]

# 3. Xử lý ngoại lai (outlier) bằng IQR
for col in numerical_cols:
    Q1 = df[col].quantile(0.25)

```

```

Q3 = df[col].quantile(0.75)

IQR = Q3 - Q1

lower = Q1 - 1.5 * IQR

upper = Q3 + 1.5 * IQR

df.loc[(df[col] < lower) | (df[col] > upper), col] = np.nan

# Điền lại giá trị thiếu sau khi loại outlier

df[col] = df[col].interpolate(method='linear',
limit_direction='both')

df[col] = df[col].ffill().bfill()

df[col] = df[col].fillna(df[col].median())


# 4. Biến đổi thành stationary (differencing)

for col in numerical_cols:

    df[col] = df[col].diff().fillna(0)


# 5. Loại bỏ cột toàn 0/toàn NaN sau differencing

cols_all_zero_or_nan = [col for col in numerical_cols if
df[col].isna().all() or (df[col] == 0).all()]

if cols_all_zero_or_nan:

    print(f"Các cột bị loại bỏ sau differencing: {cols_all_zero_or_nan}")

    df = df.drop(columns=cols_all_zero_or_nan)

    numerical_cols = [col for col in numerical_cols if col not in
cols_all_zero_or_nan]


# 6. Chuẩn hóa dữ liệu

if len(numerical_cols) > 0:

    scaler = StandardScaler()

    df[numerical_cols] = scaler.fit_transform(df[numerical_cols])

```

Trực quan hóa dữ liệu đã tiền xử lý:

```

# Violinplot

if len(numerical_cols) > 0:

    plt.figure(figsize=(max(10, len(numerical_cols)*1.2), 6))

    sns.violinplot(data=df[numerical_cols], inner='box', cut=0)

    plt.title(f'Violinplot tất cả biến số - Nồng độ {label}')

```

```

plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.savefig(os.path.join(export_dir,
f'violinplot_all_{label}.png'))
plt.close()

else:

    print(f"[CẢNH BÁO] Không có biến số nào để vẽ violinplot cho nồng độ {label}.")

# Boxplot

if len(numerical_cols) > 0:

    plt.figure(figsize=(max(10, len(numerical_cols)*1.2), 6))

    sns.boxplot(data=df[numerical_cols])

    plt.title(f'Boxplot tất cả biến số - Nồng độ {label}')

    plt.xticks(rotation=45, ha='right')

    plt.tight_layout()

    plt.savefig(os.path.join(export_dir, f'boxplot_all_{label}.png'))

    plt.close()

else:

    print(f"[CẢNH BÁO] Không có biến số nào để vẽ boxplot cho nồng độ {label}.")

# Histogram (subplots)

if len(numerical_cols) > 0:

    n_cols = 4

    n_rows = int(np.ceil(len(numerical_cols)/n_cols))

    fig, axes = plt.subplots(n_rows, n_cols, figsize=(n_cols*4,
n_rows*3))

    axes = axes.flatten()

    for i, col in enumerate(numerical_cols):

        sns.histplot(df[col], bins=30, kde=True, ax=axes[i],
color='steelblue')

        axes[i].set_title(col)

    for j in range(i+1, len(axes)):
```

```

        axes[j].axis('off')

    plt.suptitle(f'Histogram các biến số - Nồng độ {label}', fontsize=16)
    plt.tight_layout(rect=[0, 0, 1, 0.97])
    plt.savefig(os.path.join(export_dir, f'histogram_all_{label}.png'))
    plt.close()

else:

    print(f'[CẢNH BÁO] Không có biến số nào để vẽ histogram cho nồng độ
{label}.')

    # Heatmap tương quan

    if len(numerical_cols) > 1:

        plt.figure(figsize=(max(8, len(numerical_cols)), max(6,
len(numerical_cols)*0.6)))

        corr = df[numerical_cols].corr()

        sns.heatmap(corr, annot=True, fmt=".2f", cmap='coolwarm', cbar=True)

        plt.title(f'Heatmap ma trận tương quan - Nồng độ {label}')
        plt.tight_layout()

        plt.savefig(os.path.join(export_dir, f'corr_heatmap_{label}.png'))
        plt.close()

    else:

        print(f'[CẢNH BÁO] Không đủ biến số để vẽ heatmap tương quan cho nồng
độ
{label}.')

Số dòng ban đầu: 11839

Số dòng sau khi loại trùng lặp: 11839

Các cột bị loại bỏ do quá nhiều thiếu/0: ['H2S_concentration', 'CH4ppm',
'HCHOmg/m3', 'TVOCppm']

Số dòng ban đầu: 8646

Số dòng sau khi loại trùng lặp: 8646

Các cột bị loại bỏ do quá nhiều thiếu/0: ['ccsTVOC', 'H2S_concentration',
'CH4ppm', 'HCHOmg/m3', 'TVOCppm']

Các cột bị loại bỏ sau differencing: ['ccseCO2']

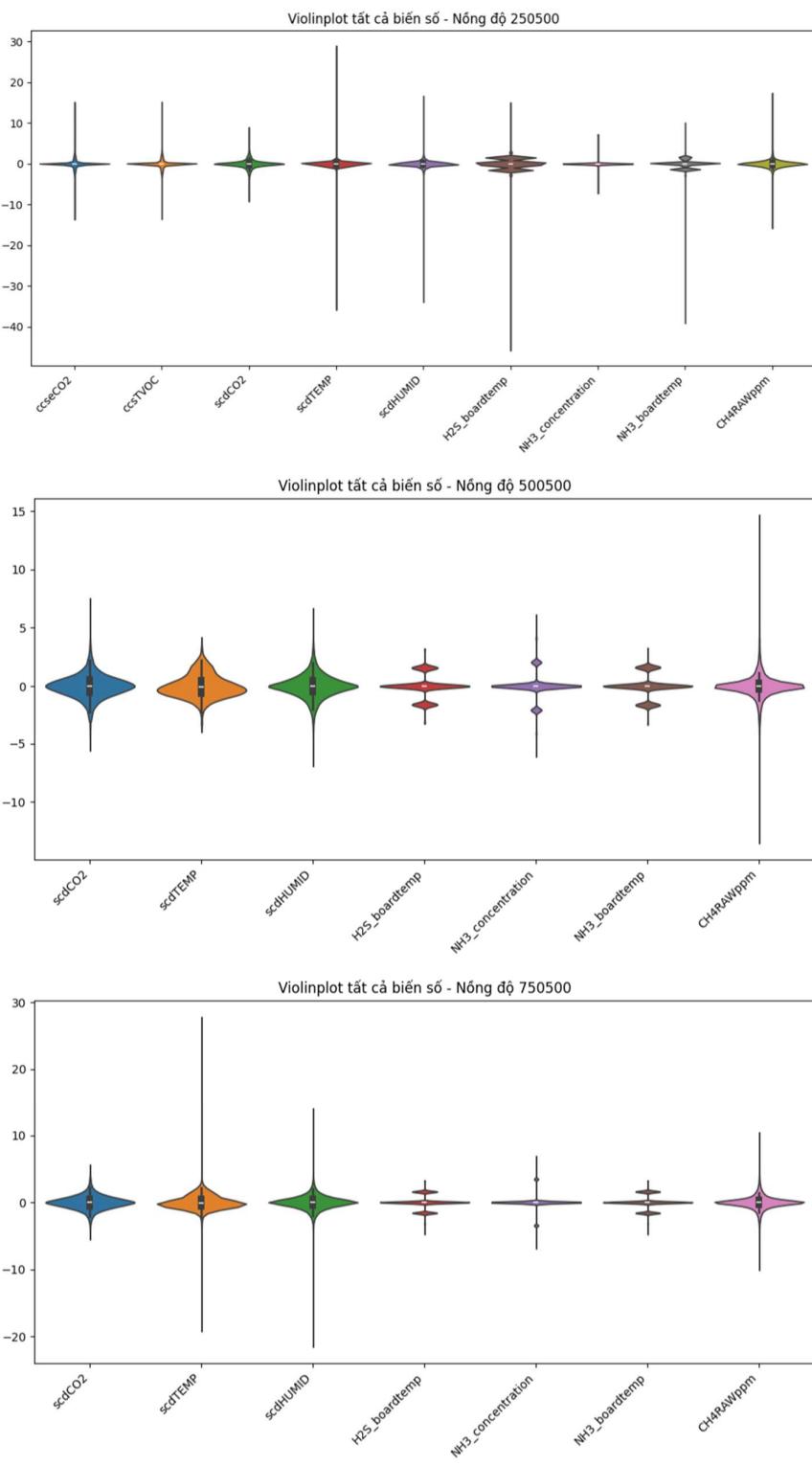
Số dòng ban đầu: 26315

Số dòng sau khi loại trùng lặp: 26315

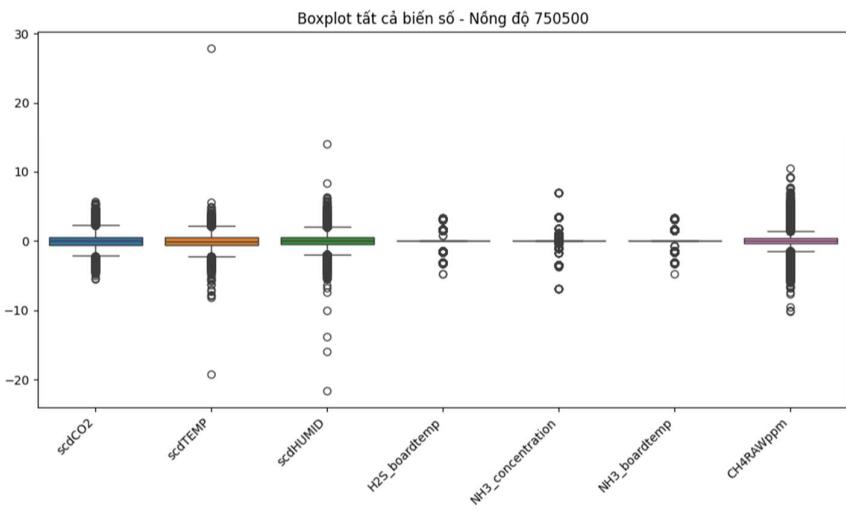
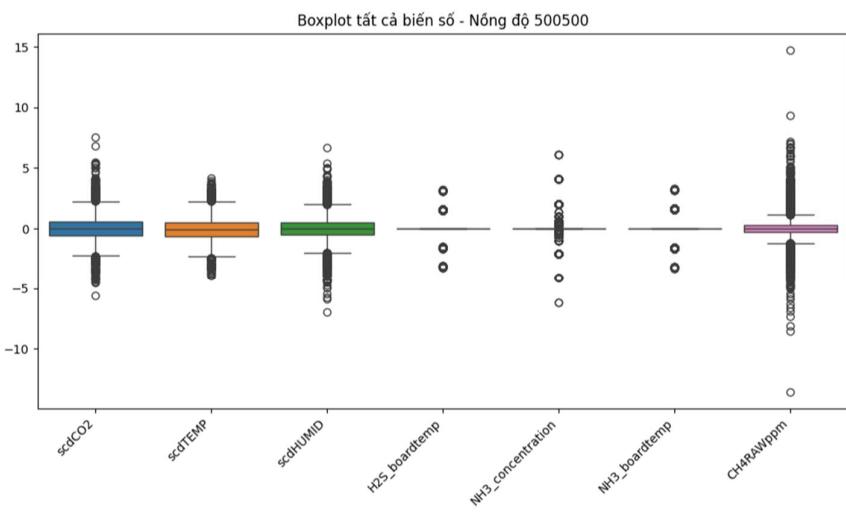
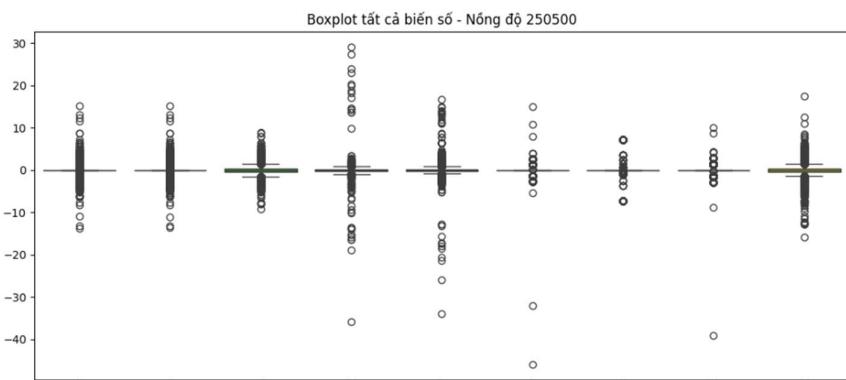
Các cột bị loại bỏ do quá nhiều thiếu/0: ['ccsTVOC', 'H2S_concentration',
'CH4ppm', 'HCHOmg/m3', 'TVOCppm']

Các cột bị loại bỏ sau differencing: ['ccseCO2']

```

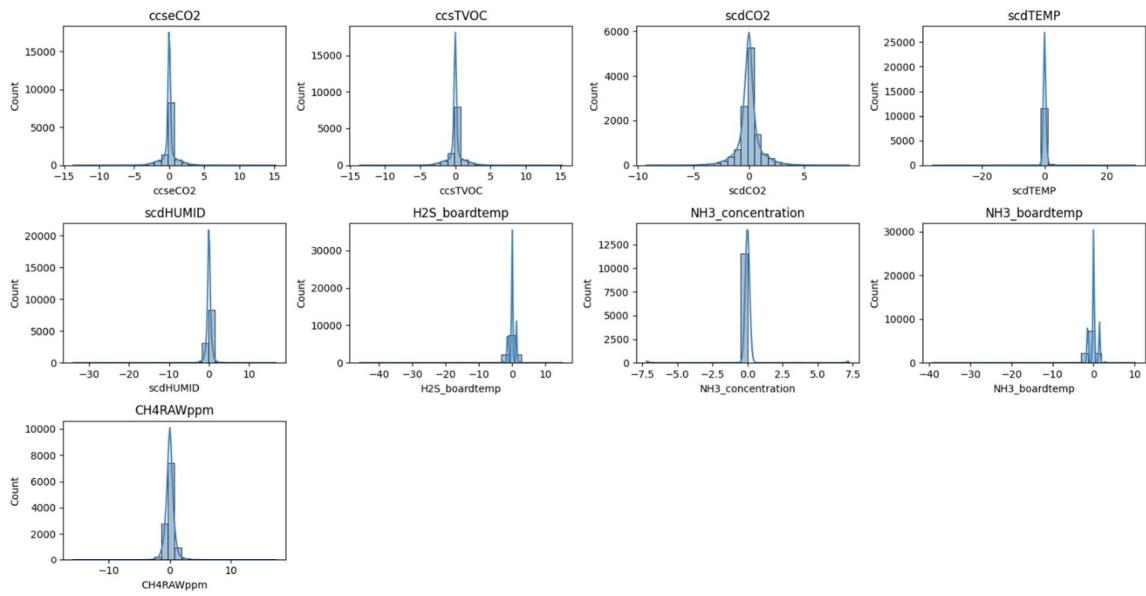


Biểu đồ 3.4: Violinplot của dữ liệu đã tiền xử lý

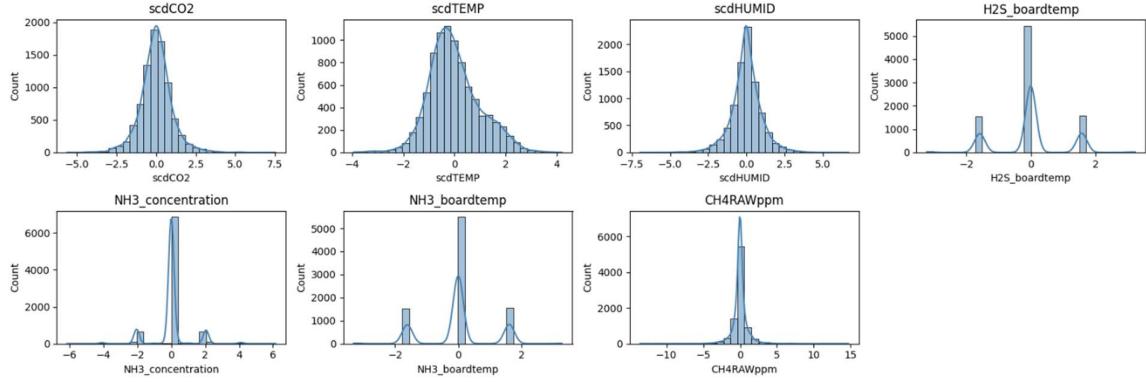


Biểu đồ 3.5: Boxplot của dữ liệu đã tiền xử lý

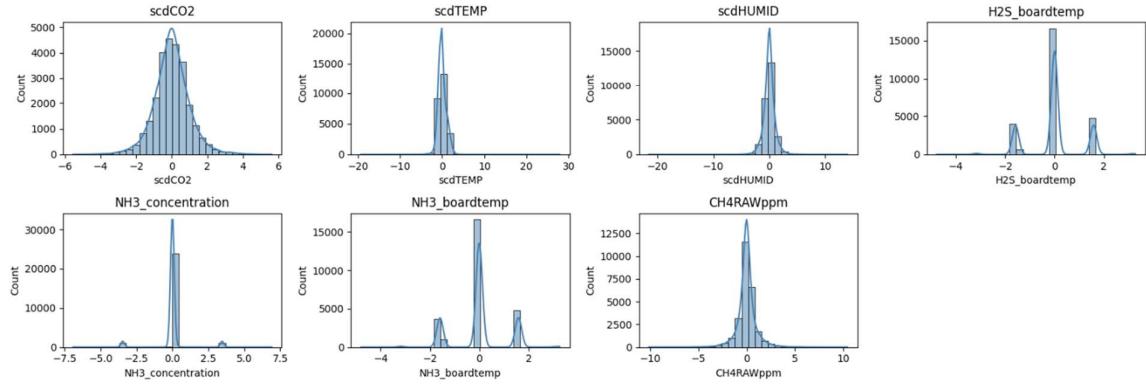
Histogram các biến số - Nồng độ 250500



Histogram các biến số - Nồng độ 500500

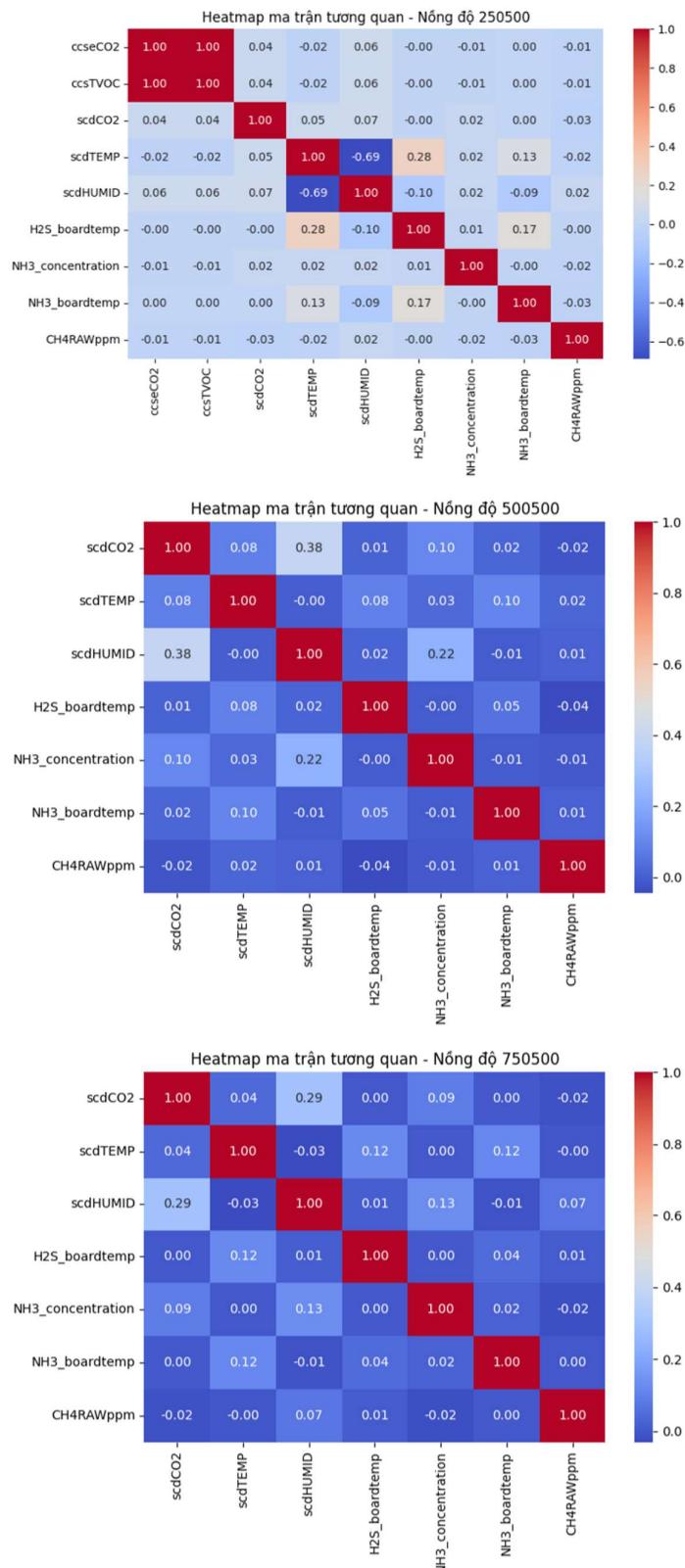


Histogram các biến số - Nồng độ 750500



Biểu đồ 3.6: Histogram của dữ liệu đã tiền xử lý

Bảng 3.2: Heatmap ma trận tương quan của dữ liệu đã tiền xử lý



Phân tích tổng quát

- Số lượng biến số của cả 3 tập dữ liệu đã giảm đáng kể do loại bỏ các cột toàn NaN, toàn 0 hoặc quá nhiều giá trị thiếu.
- Các biến đã được chuẩn hóa, loại bỏ ngoại lai, biến đổi thành chuỗi dùng giúp phù hợp hơn cho các mô hình dự báo.

Vấn đề chất lượng dữ liệu

- Các cột chất lượng kém đã bị loại bỏ hoàn toàn.
- Các giá trị thiếu và ngoại lai được xử lý (nội suy - interpolation, ffill, bfill, loại bỏ bằng IQR, z-score).
- Dữ liệu không còn các biến đơn giá trị hoặc toàn 0, giúp tăng tính tin cậy cho phân tích tiếp theo.

Đặc điểm phân phối

- Phân phối các biến trở nên đối xứng hơn, tập trung quanh 0 (do đã chuẩn hóa và differencing).
- Ngoại lai đã được loại bỏ, không còn các đỉnh bất thường hoặc đuôi dài như dữ liệu gốc.
- Các biến có phân phối gần chuẩn hơn, thuận lợi cho các mô hình học máy tuyến tính.

Mối tương quan – correlation

- Ma trận tương quan trở nên “trung tính” hơn, các giá trị tương quan tuyệt đối giảm do đã loại bỏ xu hướng, tính thời vụ và chuẩn hóa hợp lý.
- Các mối quan hệ phi tuyến hoặc phụ thuộc vào xu hướng/mức cơ bản đã được loại bỏ, giúp phát hiện các tương quan thực chất giữa biến.

Thay đổi theo nồng độ

- Các biến đơn giá trị hoặc không có ý nghĩa đã bị loại bỏ.
- Sự khác biệt giữa các nồng độ vẫn còn nhưng đã được “cân bằng” hơn nhờ chuẩn hóa, giúp mô hình dễ học hơn và so sánh khách quan hơn giữa các nồng độ.

Có thể kết luận, việc tiền xử lý đã giúp loại bỏ hoàn toàn các biến và giá trị không có ý nghĩa thống kê, làm sạch dữ liệu, giảm ảnh hưởng của ngoại lai, các giá trị thiếu/bằng 0, đưa dữ liệu về dạng dùng cũng như chuẩn hóa, phù hợp cho các mô hình

dự báo chuỗi thời gian hiện đại và tăng độ tin cậy cho các phân tích tiếp theo và giúp mô hình học máy khai thác được cấu trúc thực sự của dữ liệu.

3.2. Lựa chọn mô hình

Những mô hình sau được lựa chọn để tiến hành huấn luyện trên tập dữ liệu đã cung cấp, dựa theo điểm tương đồng về cơ chế hoạt động:

Nhóm 1: Mô hình dựa trên cây quyết định (Tree-based Models):

- Random Forest Regression
- XGBoost

Đây đều là các mô hình dựa trên phương pháp học ensemble, sử dụng cây quyết định làm mô hình cơ sở.

Nhóm 2: Mô hình tuần tự (Sequential Models):

- Tensorflow LSTM (Long Short-Term Memory)
- ARIMA (AutoRegressive Integrated Moving Average):
 - autoARIMA
 - SARIMA
- Prophet:
 - Prophet (không có biến ngoại sinh – no exogenous variables)
 - Prophet (sử dụng ccseCO2, scdTEMP, scdHUMID làm ngoại sinh)
 - Prophet (sử dụng toàn bộ các cột có sẵn làm biến ngoại sinh)

Nhóm 3: Mô hình dựa trên attention (Attention-based Models):

- Transformer

Nhóm 4: Mô hình dựa trên ví dụ (Instance-based Models):

- K-Nearest Neighbors (KNN)

Ngoài các mô hình kể trên, bài khóa luận sẽ bao gồm các mô hình thông dụng khác như SVM, Bayesian Regression, cũng như so sánh với Naïve Mean. Biến scdCO2 sẽ được sử dụng để làm mục tiêu chính cho việc dự báo, qua cả 3 liều lượng 250g/500l, 500g/500l và 750g/500l. Việc thay các biến số khác để dự đoán có thể áp dụng các phương pháp hoàn toàn tương tự. Để đảm bảo sự công bằng khi so sánh giữa các thư viện và framework khác nhau, tất cả quá trình huấn luyện mô hình và đánh giá thời gian

suy luận được trình bày đều được thực hiện hoàn toàn trên CPU [Ryzen 7 7600F, 32GB RAM]. Cách tiếp cận này tránh được sai lệch về kết quả do sự hỗ trợ tăng tốc GPU không đồng đều giữa các thư viện (Scikit không hỗ trợ gia tốc GPU, hay phiên bản mới nhất của Tensorflow yêu cầu sử dụng Linux/WSL để có thể tận dụng gia tốc GPU).

3.3. Triển khai nhóm các mô hình cây quyết định

3.3.1. Random Forest Regression

Bên cạnh các mô hình thống kê truyền thống như ARIMA, hay các mạng học sâu như LSTM, có một phương pháp học máy nổi bật về tính mạnh mẽ, hiệu quả và dễ sử dụng, đó là Random Forest Regression – một thuật toán thuộc họ mô hình ensemble learning (học tổ hợp).

Random Forest là sự kết hợp của nhiều cây quyết định (decision trees) hoạt động song song để cải thiện độ chính xác và khả năng khái quát của mô hình. Khi được sử dụng cho bài toán hồi quy – bao gồm dự báo chuỗi thời gian – Random Forest có thể học các mẫu phi tuyến, xử lý nhiễu tốt, và tận dụng dữ liệu đa biến một cách hiệu quả.

Khác với các mô hình tuyến tính, Random Forest không cần giả định về phân phối dữ liệu, không yêu cầu chuỗi phải tinh, và hoạt động tốt ngay cả khi dữ liệu có độ phức tạp cao. Đây chính là lý do vì sao Random Forest thường được dùng như một mô hình baseline mạnh mẽ hoặc là thành phần trong các hệ thống mô hình hóa phức hợp (ensemble systems).

Decision Tree Regression – Hồi quy bằng cây quyết định

Một cây quyết định hồi quy là một cấu trúc phân nhánh giống như sơ đồ cây, trong đó dữ liệu được chia nhỏ nhiều lần dựa trên các đặc trưng (features), sao cho tại mỗi nút, sai số hồi quy được giảm thiểu.

Cây được xây dựng thông qua quá trình đệ quy: tại mỗi nút, thuật toán tìm kiếm đặc trưng và ngưỡng phân chia sao cho tổng phương sai trong các nhánh con là nhỏ nhất. Quá trình tiếp tục cho đến khi đạt đến độ sâu nhất định hoặc khi không còn đủ dữ liệu để phân tách.

Việc sử dụng cây hồi quy có những lợi ích như hiển thị trực quan, không yêu cầu dữ liệu chuẩn hóa hay có thể mô hình hóa quan hệ phi tuyến.

Tuy nhiên, cây đơn lẻ rất dễ overfit – tức là mô hình hóa quá sát dữ liệu huấn luyện và kém khái quát hóa trên dữ liệu mới.

Random Forest – Học tổ hợp từ nhiều cây

Random Forest khắc phục điểm yếu của cây đơn bằng cách xây dựng nhiều cây quyết định độc lập, sau đó lấy trung bình đầu ra của tất cả các cây (trong bài toán hồi quy).

Các thành phần chính:

- Bagging (Bootstrap Aggregating): Mỗi cây được huấn luyện trên một mẫu ngẫu nhiên (có lặp) từ dữ liệu gốc.
- Tính ngẫu nhiên trong chọn đặc trưng (feature selection): Ở mỗi nút phân chia, chỉ một tập con nhỏ các đặc trưng được xem xét để phân chia. Điều này tạo ra sự đa dạng giữa các cây, giúp giảm phương sai của mô hình.

Dự đoán hồi quy của Random Forest:

$$\hat{y} = \frac{1}{N} \sum_{i=1}^N T_i(x), \quad (7)$$

trong đó:

N : số lượng cây.

$T_i(x)$: dự đoán của cây thứ i cho điểm dữ liệu đầu vào x .

Ưu điểm chính của Random Forest là:

- **Ôn định và chính xác:** Giảm overfitting so với cây đơn.
- **Không yêu cầu giả định về dữ liệu.**
- **Học được** quan hệ phi tuyến và phức tạp.
- **Tự động xử lý tương tác** giữa các đặc trưng.
- **Cung cấp đánh giá tầm quan trọng** của biến đầu vào.

Mặc dù Random Forest không phải là mô hình “chuỗi thời gian” truyền thống (như ARIMA hay LSTM), nhưng mô hình này có thể được điều chỉnh để dự báo các chuỗi thời gian thông qua các kỹ thuật tiền xử lý dữ liệu phù hợp.

Để áp dụng Random Forest vào chuỗi thời gian, cần biến đổi chuỗi thành một tập dữ liệu có dạng hồi quy đa biến. Cách phổ biến là sử dụng các đặc trưng độ trễ (lag features).

Ưu điểm khi dùng Random Forest cho việc dự báo chuỗi thời gian:

- **Không yêu cầu chuỗi phải tĩnh:** Không cần lấy sai phân.
- **Linh hoạt** trong xử lý biến ngoại sinh.
- **Không bị ảnh hưởng nhiều** bởi nhiễu.

- **Có thể mở rộng dễ dàng** với dữ liệu đa biến.

Tuy nhiên mô hình này còn tồn tại các hạn chế sau:

- **Không dự báo trực tiếp nhiều bước tương lai (multi-step forecast):** Phải sử dụng kỹ thuật lặp lại dự báo hoặc mô hình hóa từng bước riêng biệt.
- **Không nắm bắt tốt các chu kỳ dài hạn** nếu không được cung cấp các đặc trưng rõ ràng.
- **Không có khái niệm “bộ nhớ”** (internal state/memory) như RNN hoặc LSTM.

Bảng 3.3: Tham số huấn luyện Random Forest

Siêu tham số	Giá trị	Mô tả
n_estimators	100	Số lượng cây trong rừng.
max_depth	20	Độ sâu tối đa của mỗi cây.
min_samples_split	5	Số mẫu tối thiểu để tách một nút nội bộ.
min_samples_leaf	2	Số mẫu tối thiểu tại một nút lá.
random_state	42	Đảm bảo tính tái lập của kết quả.
n_jobs	-1	Sử dụng tất cả các lõi CPU có sẵn.

```
# Thêm các đặc trưng độ trễ
for lag in range(1, 13):    # Thêm 12 độ trễ theo giờ
    df[f'{target_col}_lag_{lag}'] = df[target_col].shift(lag)

# Thêm các thống kê trung bình động
df[f'{target_col}_rolling_mean_6h'] = df[target_col].rolling(window=6).mean()
df[f'{target_col}_rolling_std_6h'] = df[target_col].rolling(window=6).std()

df[f'{target_col}_rolling_mean_24h'] = df[target_col].rolling(window=24).mean()

# Huấn luyện mô hình Random Forest
print("Đang huấn luyện mô hình Random Forest...")
# Bắt đầu với một mô hình cơ bản
rf = RandomForestRegressor(
```

```

n_estimators=100,
max_depth=20,
min_samples_split=5,
min_samples_leaf=2,
random_state=42,
n_jobs=-1 # Sử dụng tất cả các lõi có sẵn
)

# Huấn luyện mô hình
rf.fit(X_train_scaled, y_train)

# Dự đoán
y_pred = rf.predict(X_test_scaled)

```

3.3.2. XGBoost

XGBoost là một thuật toán boosting sử dụng cây quyết định, nổi bật với tốc độ huấn luyện nhanh và khả năng tối ưu hóa mạnh mẽ. Khác với Random Forest (bagging), XGBoost xây dựng các cây liên tiếp, mỗi cây mới tập trung sửa lỗi của các cây trước đó thông qua cơ chế gradient boosting. XGBoost còn hỗ trợ regularization (chuẩn hóa), xử lý thiếu dữ liệu tốt và dễ dàng tùy biến cho các bài toán hồi quy hoặc phân loại.

Bảng 3.4: Tham số huấn luyện XGBoost

Siêu tham số	Giá trị	Mô tả
objective	'reg:squarederror'	Hàm mục tiêu cho bài toán hồi quy.
n_estimators	100	Số lượng cây (vòng lặp boosting).
Các tham số khác	Giá trị mặc định	(Ví dụ: learning_rate, max_depth, etc.)

Tạo đặc trưng window cho XGBoost (dự báo theo chuỗi)

```

values = df[target_col].values
X, y = [], []
for i in range(len(values) - window_size):
    X.append(values[i:i+window_size])
    y.append(values[i+window_size])
X = np.array(X)

```

```

y = np.array(y)

# Huấn luyện mô hình XGBoost
model = xgb.XGBRegressor(objective='reg:squarederror', n_estimators=100)
model.fit(X_train, y_train)

# Dự báo
y_pred = model.predict(X_test)

```

3.4. Triển khai nhóm các mô hình tuần tự

3.4.1. Tensorflow LSTM

LSTM là một kiến trúc mạng nơ-ron được thiết kế đặc biệt để khắc phục những hạn chế của RNN, cho phép ghi nhớ thông tin trong thời gian dài, và "quên" những gì không quan trọng. Qua hàng chục năm phát triển, LSTM không chỉ được sử dụng trong xử lý chuỗi thời gian mà còn trong dịch máy, xử lý ngôn ngữ tự nhiên, và nhận diện giọng nói.

Khi kết hợp LSTM với các công cụ học sâu mạnh mẽ như TensorFlow, khả năng xử lý dữ liệu chuỗi trở nên mạnh mẽ hơn bao giờ hết. TensorFlow cung cấp môi trường trực quan, tối ưu hóa tốt, và khả năng triển khai dễ dàng, từ đó biến LSTM trở thành lựa chọn hàng đầu trong nhiều bài toán dự báo thời gian.

Mỗi "đơn vị" (cell) của LSTM giống như một cỗ máy ghi nhớ, có thể quyết định giữ lại hoặc loại bỏ thông tin thông qua ba cơ chế chính gọi là các cổng (gates). Các cổng này là những lớp mạng nơ-ron riêng biệt, sử dụng hàm sigmoid để quyết định thông tin nào nên được truyền qua và lược bỏ.

Cổng quên (Forget Gate)

Cổng quên là nơi quyết định phần nào trong trạng thái bộ nhớ cũ nên được quên đi. Cổng quên nhận vào đầu vào hiện tại x_t và trạng thái ẩn trước đó h_{t-1} , sau đó áp dụng công thức:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \quad (8)$$

giá trị f_t nằm trong khoảng từ 0 đến 1, biểu thị mức độ "giữ lại" thông tin. Nếu f_t gần bằng 1, thông tin cũ được giữ lại; nếu gần 0, thông tin sẽ bị quên đi.

Cổng đầu vào (Input Gate)

Cổng đầu vào quyết định những thông tin mới nào sẽ được thêm vào trạng thái bộ nhớ:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \quad (9)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C), \\ (10)$$

i_t là cổng quyết định mức độ cập nhật, còn \tilde{C}_t là thông tin mới được đề xuất đưa vào bộ nhớ.

Cập nhật trạng thái bộ nhớ

Trạng thái bộ nhớ mới được tính bằng:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t, \\ (11)$$

hay kết hợp phần cũ đã được lựa chọn giữ lại và thông tin mới được thêm vào.

Cổng đầu ra (Output Gate)

Cuối cùng, đầu ra của cell được xác định bởi:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\ (12)$$

$$h_t = o_t * \tanh(C_t), \\ (13)$$

trong đó h_t là đầu ra và cũng sẽ được truyền tới bước thời gian tiếp theo. Nhờ vào cấu trúc này, LSTM có thể "quyết định" mức độ ảnh hưởng của thông tin cũ và mới tại mỗi bước thời gian.

Tóm lại, điểm đặc biệt của LSTM là ngoài internal state, mô hình còn duy trì một trạng thái bộ nhớ riêng biệt (cell state) xuyên suốt các bước thời gian, qua đó có thể ghi nhớ các thông tin quan trọng trong thời gian dài. Đây là điều mà các RNN thông thường không làm được.

Ưu điểm của LSTM trong dự báo chuỗi thời gian:

- **Học được các mẫu dài hạn:** LSTM có khả năng ghi nhớ các mẫu trong chuỗi dài, điều mà RNN truyền thống không làm tốt.

- **Xử lý dữ liệu phi tuyến:** LSTM có thể học các quan hệ phi tuyến và phức tạp giữa các điểm dữ liệu.

- **Không cần giả định phân phối dữ liệu:** Khác với các mô hình thống kê truyền thống, LSTM không yêu cầu dữ liệu phải tuân theo phân phối chuẩn hay tuyến tính.

- **Xử lý nhiều đặc trưng/chuỗi thời gian đa biến:** LSTM có thể xử lý đồng thời nhiều biến đầu vào như nhiệt độ, độ ẩm, giờ trong ngày, v.v.

Bảng 3.5: Tham số huấn luyện LSTM

Tham số/Lớp (Parameter/Layer)	Giá trị/Cấu hình (Value/Configuration)	Mô tả
Kiến trúc mạng (Network Arch.)	Sequential	Mô hình tuần tự Keras.
Lớp LSTM	Units = 64, return_sequences=False	64 đơn vị ẩn, chỉ trả về output cuối cùng.
Lớp Dropout	Rate = 0.2	Tỷ lệ dropout để chống overfitting.
Lớp Dense (Output)	Units = 1	Lớp output cho bài toán hồi quy.
Trình tối ưu hóa (Optimizer)	'adam'	Trình tối ưu hóa Adam.
Hàm mất mát (Loss Function)	'mse' (Mean Squared Error)	Hàm mất mát cho hồi quy.
Số Epoch tối đa (Max Epochs)	100	Số vòng lặp huấn luyện tối đa.
Kích thước lô (Batch Size)	32	Số mẫu trong mỗi lô huấn luyện.
Tỷ lệ dữ liệu kiểm chứng (Val Split)	0.1	10% dữ liệu huấn luyện dùng để kiểm chứng.
Callback	Early Stopping (patience=10)	Dừng sớm nếu val_loss không cải thiện sau 10 epoch.

```

# Chuẩn bị dữ liệu cho LSTM (univariate)
values = df[target_col].values

def create_sequences(data, window=24):
    X, y = [], []
    for i in range(len(data) - window):
        X.append(data[i:i+window])
        y.append(data[i+window])
    return np.array(X), np.array(y)

X, y = create_sequences(values, window=window_size)

# Xây dựng mô hình LSTM
model = Sequential()
model.add(LSTM(64, input_shape=(window_size, 1),
               return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')

# Huấn luyện
early_stop = EarlyStopping(monitor='val_loss', patience=10,
                           restore_best_weights=True)
history = model.fit(
    X_train, y_train,
    epochs=100,
    batch_size=32,
    validation_split=0.1,
    callbacks=[early_stop],
    verbose=1
)

# Dự báo
y_pred = model.predict(X_test).flatten()

```

3.4.2. ARIMA

ARIMA không chỉ là một mô hình thống kê đơn lẻ mà là một nhóm các mô hình liên quan bao gồm AR, MA, ARMA, ARIMA và các mở rộng như SARIMA, SARIMAX. Những mô hình này được phát triển từ giữa thế kỷ 20 và vẫn được sử dụng rộng rãi đến ngày nay vì tính hiệu quả, dễ giải thích, và phù hợp với các chuỗi dữ liệu ổn định.

Mặc dù ARIMA không có khả năng học phi tuyến hoặc xử lý dữ liệu đa biến phức tạp như các mạng nơ-ron, nhưng chính sự đơn giản và tính trực quan của ARIMA lại làm nên sức mạnh. ARIMA cho phép mô hình hóa mối quan hệ giữa một điểm trong chuỗi với chính điểm đó trong quá khứ, và với nhiều sai số tích lũy, qua đó đưa ra các dự đoán dựa trên cấu trúc thống kê bên trong chuỗi.

AR – Autoregressive (Tự hồi quy)

AR là mô hình tự hồi quy, tức là giá trị hiện tại của chuỗi y_t được mô hình hóa như một tổ hợp tuyến tính của các giá trị trong quá khứ.

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t, \quad (14)$$

- y_t là giá trị hiện tại.
- ϕ_i là hệ số hồi quy tại độ trễ i .
- c là hằng số.
- ϵ_t là nhiễu trắng.

MA – Moving Average (Trung bình trượt)

Khác với AR, mô hình MA mô tả y_t như là một tổ hợp tuyến tính của **nhiều trọng quá khứ**.

$$y_t = \mu + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t, \quad (15)$$

- μ là trung bình chuỗi.
- θ_i là hệ số của nhiễu tại độ trễ i .
- ϵ_t là nhiễu trắng tại thời điểm hiện tại.

ARMA – Autoregressive Moving Average

Khi chuỗi dữ liệu vừa có tính tự hồi quy, vừa có cấu trúc nhiễu phức tạp, ta sử dụng mô hình ARMA – sự kết hợp giữa AR và MA. Công thức ARMA(p,q)

$$y_t = c + \sum_{i=1}^p \phi_i y_{t-i} + \sum_{j=1}^q \theta_j \epsilon_{t-j} + \epsilon_t \quad (16)$$

ARMA yêu cầu chuỗi đầu vào phải **tĩnh (stationary)** – tức là không có xu hướng tăng/giảm theo thời gian, phương sai không đổi, và không có chu kỳ dài hạn.

ARIMA – Autoregressive Integrated Moving Average

Trong thực tế, các chuỗi thời gian mang tính không tĩnh – ví dụ: giá cổ phiếu, dân số, nhiệt độ toàn cầu. Để xử lý điều này, mô hình ARIMA thêm thành phần tích phân (I – Integrated) bằng cách lấy sai phân để biến chuỗi không tĩnh thành tĩnh.

ARIMA(p, d, q) bao gồm:

- p: số bậc tự hồi quy.
- d: số lần sai phân cần thiết để đạt tính tĩnh.
- q: bậc của trung bình trượt.

$$\Delta^d y_t = c + \sum_{i=1}^p \phi_i \Delta^d y_{t-i} + \sum_{j=1}^q \theta_j \epsilon_{t-j} + \epsilon_t, \quad (17)$$

Δ^d là phép sai phân bậc d.

SARIMA – Seasonal ARIMA

Khi chuỗi có yếu tố chu kỳ hoặc mùa vụ (ví dụ: bán hàng tăng vào dịp lễ), ARIMA không đủ để mô hình hóa điều này. SARIMA (Seasonal ARIMA) được sử dụng để xử lý chuỗi có tính chất mùa.

Một mô hình SARIMA được ký hiệu là SARIMA(p, d, q)(P, D, Q, s):

- (p,d,q)(p,d,q)(p,d,q): thành phần ARIMA cơ bản.
- (P,D,Q)(P,D,Q)(P,D,Q): các thành phần mùa vụ tương ứng.
- s: chu kỳ mùa (ví dụ: s = 12 với chuỗi hàng tháng có mùa 1 năm).

SARIMA có thể được xem như một ARIMA hai lớp – một lớp mô hình hóa cấu trúc chuỗi chính, một lớp xử lý yếu tố mùa.

Cuối cùng, khi có các biến ngoại sinh (exogenous variables) ảnh hưởng đến chuỗi (ví dụ: giá cổ phiếu bị ảnh hưởng bởi lãi suất, tin tức vĩ mô...), ta dùng SARIMAX.

autoARIMA:

autoARIMA là một phương pháp tự động lựa chọn các tham số tối ưu (p , d , q) cho mô hình ARIMA dựa trên dữ liệu đầu vào. Thay vì phải thử nghiệm thủ công nhiều cấu hình ARIMA khác nhau, autoARIMA sẽ tự động kiểm tra, đánh giá và chọn ra mô hình tốt nhất (theo các tiêu chí như AIC/BIC), giúp tiết kiệm thời gian và đảm bảo hiệu quả dự báo tối ưu.

Mô hình này tự động tìm kiếm bộ tham số (p , d , q) tối ưu. Các cấu hình chính cho quá trình tìm kiếm tự động bao gồm:

- `seasonal=False`: Chưa xét yếu tố mùa vụ trong quá trình tìm kiếm tự động.
- `stepwise=True`: Sử dụng thuật toán tìm kiếm từng bước để tăng tốc độ.
- Các tham số khác (như giới hạn tối đa cho p , d , q) sử dụng giá trị mặc định của thư viện `pmdarima`.

```
autoarima_model = pm.auto_arima(y_train, seasonal=False, stepwise=True,
trace=True, suppress_warnings=True)

y_pred_autoarima = autoarima_model.predict(n_periods=len(y_test))

sarima_order = (1, 0, 1)

seasonal_order = (1, 1, 1, 24) # m=24 cho dữ liệu theo giờ, nếu dữ liệu khác thì điều chỉnh lại
```

SARIMA:

Mô hình này yêu cầu chỉ định rõ các bộ tham số. Các giá trị được chọn dựa trên phân tích ACF/PACF sơ bộ và cấu hình phổ biến cho dữ liệu theo giờ:

- `order` (Phi mùa vụ - Non-seasonal): $(1, 0, 1)$ - AR(1), I(0), MA(1)
- `seasonal_order` (Mùa vụ - Seasonal): $(1, 1, 1, 24)$ - SAR(1), SI(1), SMA(1)
với chu kỳ mùa vụ $m=24$ (giờ trong ngày).

```
sarima_model = SARIMAX(y_train, order=sarima_order,
seasonal_order=seasonal_order)

sarima_fit = sarima_model.fit(disp=False)

y_pred_sarima = sarima_fit.forecast(steps=len(y_test))
```

3.4.3. Prophet

Prophet là một thư viện dự báo chuỗi thời gian mã nguồn mở do Facebook phát triển, được thiết kế để đơn giản hóa và tự động hóa quá trình dự báo, đặc biệt với dữ liệu có xu hướng và chu kỳ.

Bảng 3.6: Tham số huấn luyện Prophet

Siêu tham số	Giá trị	Mô tả
yearly_seasonality	False	Bỏ qua tính mùa vụ theo năm.
weekly_seasonality	True	Bật tính mùa vụ theo tuần.
daily_seasonality	True	Bật tính mùa vụ theo ngày.
Các tham số khác	Mặc định	(Growth, changepoints, holidays)

Lưu ý: Đối với các phiên bản Prophet có biến ngoại sinh, các tham số mùa vụ này được giữ nguyên, chỉ bổ sung thêm các biến ngoại sinh vào quá trình fit và predict.

```
# Chuẩn hóa dữ liệu cho Prophet

prophet_df = df[['Date_time', 'target_col']].rename(columns={'Date_time': 'ds', 'target_col': 'y'})

prophet_df['y'] = prophet_df['y'].astype(float)
prophet_df = prophet_df.dropna()

# Khởi tạo và huấn luyện Prophet (không có biến ngoại sinh)

model = Prophet(yearly_seasonality=False, weekly_seasonality=True,
daily_seasonality=True)

model.fit(train_df)

# Tạo dataframe các mốc thời gian cần dự báo

future = model.make_future_dataframe(periods=len(test_df), freq='min')
forecast = model.predict(future)

# Lấy giá trị dự báo ứng với tập test

forecast_test = forecast.iloc[-len(test_df):]['yhat'].values
y_test = test_df['y'].values
```

3.5. Triển khai nhóm các mô hình attention

3.5.1. Transformer

Transformer là một kiến trúc mạng nơ-ron được giới thiệu lần đầu tiên trong bài báo “Attention is All You Need” (Vaswani et al., 2017). Khác với các mô hình truyền thống như RNN hay LSTM, Transformer sử dụng hoàn toàn cơ chế attention (chính xác hơn là self-attention) để xử lý dữ liệu tuần tự, cho phép mô hình học các mối quan hệ giữa các phần tử trong chuỗi mà không cần xử lý tuần tự từng bước. Điều này giúp Transformer song song hóa quá trình huấn luyện, tăng hiệu quả và khả năng mở rộng.

Một số thành phần chính của Transformer:

- **Self-Attention:** Giúp mô hình tập trung vào các phần quan trọng của chuỗi đầu vào để sinh ra biểu diễn tốt hơn.
- **Multi-head Attention:** Kết hợp nhiều cơ chế attention song song, giúp mô hình học được nhiều khía cạnh khác nhau của dữ liệu.
- **Position Encoding:** Bổ sung thông tin về vị trí các phần tử trong chuỗi, vì Transformer bản chất không nhận biết thứ tự.

Trước đây, các mô hình như ARIMA, LSTM, GRU được sử dụng phổ biến. Tuy nhiên, gần đây Transformer đã chứng minh hiệu quả vượt trội nhờ khả năng:

- **Ghi nhớ dài hạn:** Self-attention giúp Transformer nắm bắt được các quan hệ dài hạn trong chuỗi dữ liệu khi mà các mô hình RNN/LSTM thường gặp khó khăn.
- **Xử lý song song:** Tăng tốc quá trình huấn luyện trên các chuỗi dài và tập dữ liệu lớn.
- **Linh hoạt với nhiều loại dữ liệu:** Có thể mở rộng cho các bài toán chuỗi thời gian đa chiều đầu vào/đầu ra.

Bảng 3.7: Tham số huấn luyện Transformer

Tham số/Lớp (Parameter/Layer)	Giá trị/Cấu hình (Value/Configuration)	Mô tả
feature_size (Input)	1	Số lượng đặc trưng đầu vào tại mỗi bước thời gian.
num_layers (Encoder)	2	Số lớp Transformer Encoder.

nhead (Multi-head Attention)	4	Số lượng đầu chú ý (attention heads).
d_model (Embedding Dim.)	Internal/Default	Kích thước embedding (Không rõ từ snippet, dùng mặc định).
dropout	0.1	Tỷ lệ dropout.
Trình tối ưu hóa (Optimizer)	Adam	Trình tối ưu hóa Adam.
Tốc độ học (Learning Rate)	0.001	Tốc độ học ban đầu.
Hàm mất mát (Loss Function)	MSELoss (Mean Squared Error)	Hàm mất mát cho hồi quy.
Số Epoch (N_Epochs)	50	Số vòng lặp huấn luyện.
Kích thước lô (Batch Size)	64	Số mẫu trong mỗi lô huấn luyện.

```

class TimeSeriesTransformer(nn.Module):

    def __init__(self, feature_size=1, num_layers=2, dropout=0.1):
        super(TimeSeriesTransformer, self).__init__()
        self.model_type = 'Transformer'
        self.embedding = nn.Linear(feature_size, d_model)
        encoder_layer = nn.TransformerEncoderLayer(d_model=d_model, nhead=4,
                                                    dropout=dropout)
        self.Transformer_encoder = nn.TransformerEncoder(encoder_layer,
                                                       num_layers=num_layers)
        self.decoder = nn.Linear(d_model, 1)

    def forward(self, src):
        # src shape: (batch, seq_len, feature_size)
        src = self.embedding(src) # (batch, seq_len, d_model)
        src = src.permute(1, 0, 2) # (seq_len, batch, d_model)
        output = self.Transformer_encoder(src)
        output = output[-1, :, :] # Lấy output cuối cùng

```

```

        output = self.decoder(output)

        return output.squeeze()

metrics = []

# Chuẩn bị dữ liệu window
values = df[target_col].values
X, y = [], []
for i in range(len(values) - window_size):
    X.append(values[i:i+window_size])
    y.append(values[i+window_size])
X = np.array(X)
y = np.array(y)

# Đưa về tensor
X_train_tensor = torch.tensor(X_train, dtype=torch.float32).unsqueeze(-1).to(device)
y_train_tensor = torch.tensor(y_train, dtype=torch.float32).to(device)
X_test_tensor = torch.tensor(X_test, dtype=torch.float32).unsqueeze(-1).to(device)
y_test_tensor = torch.tensor(y_test, dtype=torch.float32).to(device)

# Khởi tạo model
model = TimeSeriesTransformer(feature_size=1).to(device)
criterion = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)

# Huấn luyện
n_epochs = 50
batch_size = 64
for epoch in range(n_epochs):
    model.train()
    perm = torch.randperm(X_train_tensor.size(0))
    sum_loss = 0

```

```

    for i in range(0, X_train_tensor.size(0), batch_size):
        idx = perm[i:i+batch_size]
        batch_X = X_train_tensor[idx]
        batch_y = y_train_tensor[idx]
        optimizer.zero_grad()
        output = model(batch_X)
        loss = criterion(output, batch_y)
        loss.backward()
        optimizer.step()
        sum_loss += loss.item() * batch_X.size(0)

    if (epoch+1) % 10 == 0 or epoch == 0:
        print(f'Epoch {epoch+1}/{n_epochs}, Loss: {sum_loss/X_train_tensor.size(0):.4f}')

# Dự báo
model.eval()

with torch.no_grad():
    y_pred = model(X_test_tensor).cpu().numpy()

```

3.6. Triển khai nhóm các mô hình học ví dụ

3.6.1. K-Nearest Neighbors

KNN thuộc nhóm mô hình không tham số (non-parametric) và giám sát (supervised learning). KNN không học mô hình tường minh nào mà sử dụng lịch sử của chuỗi để tìm các mẫu con (subsequences) tương tự với mẫu hiện tại, từ đó dự báo điểm kế tiếp dựa trên hành vi trong quá khứ. Điều này khiến KNN trở thành lựa chọn lý tưởng trong các hệ thống yêu cầu dễ triển khai, độ phức tạp thấp và khả năng mô hình hóa quan hệ phi tuyến.

KNN trong chuỗi thời gian hoạt động theo các bước sau:

- Với mỗi thời điểm t , tạo một cửa sổ trượt (sliding window) có độ dài w , ví dụ $[y(t-w), \dots, y(t-1)]$.
- So sánh vector hiện tại với tất cả các vector trong lịch sử (có cùng độ dài), sử dụng các khoảng cách như Euclidean distance, Dynamic Time Warping (DTW) hay Manhattan distance.

- Sau khi tìm được K láng giềng gần nhất, ta lấy trung bình (hoặc trung vị) các giá trị tiếp theo của các vector đó làm dự đoán.

Bảng 3.8: Tham số huấn luyện KNN

Siêu tham số	Giá trị	Mô tả
n_neighbors	5	Số lượng hàng xóm gần nhất để xem xét.
Các tham số khác	Mặc định	(weights, algorithm, metric, etc.)

```
# Tạo đặc trưng window cho KNN (dự báo theo chuỗi)
```

```
values = df[target_col].values
X, y = [], []
for i in range(len(values) - window_size):
    X.append(values[i:i+window_size])
    y.append(values[i+window_size])
X = np.array(X)
y = np.array(y)
```

```
# Huấn luyện mô hình KNN
```

```
model = KNeighborsRegressor(n_neighbors=5)
model.fit(X_train, y_train)
```

```
# Dự báo
```

```
y_pred = model.predict(X_test)
```

3.7. Các mô hình/cách tiếp cận khác

3.7.1. Naïve Mean

```
# Dự báo Naive Mean: mọi giá trị dự báo bằng trung bình tập huấn luyện
mean_value = y_train.mean()
y_pred = np.full_like(y_test, mean_value, dtype=np.float64)
```

3.7.2. SVM

SVM (Support Vector Machine) là một thuật toán học máy giám sát (supervised learning) phổ biến, có thể được sử dụng để học mối quan hệ giữa các điểm dữ liệu quá khứ và giá trị tương lai của chuỗi. Nhờ khả năng xử lý tốt các quan hệ phi tuyến thông qua sử dụng hàm kernel, SVM là một lựa chọn linh hoạt cho nhiều loại chuỗi thời gian thực tế – đặc biệt là khi dữ liệu không có cấu trúc tuyến tính hoặc có nhiễu.

Trong SVM, mục tiêu là tìm một hàm $f(x)$ sao cho phần lớn các điểm dữ liệu nằm trong một khoảng sai số ϵ xung quanh đường dự báo này, đồng thời đảm bảo mô hình không quá phức tạp (tối ưu hóa độ phẳng của hàm).

Các thành phần chính:

- **Hàm mất mát ϵ -insensitive:** Chỉ phạt đối với những dự đoán sai lệch vượt quá ngưỡng ϵ .
- **Hàm kernel:** Cho phép SVM học quan hệ phi tuyến trong không gian cao hơn. Các kernel phổ biến bao gồm: Linear (tuyến tính), Polynomial (đa thức), hàm RBF (Radial Basis Function), hàm Sigmoid.
- **Các tham số cần điều chỉnh** bao gồm C (regularization): cân bằng giữa độ chính xác và đơn giản của mô hình, ϵ : ngưỡng sai số cho phép cũng như loại hàm kernel.

Bảng 3.9: Tham số huấn luyện SVM

Siêu tham số	Giá trị	Mô tả
kernel	'rbf'	Hàm kernel Radial Basis Function.
C (Regularization)	10	Tham số điều chuẩn (trade-off margin/error).
epsilon	0.1	Ngưỡng sai số không bị phạt (epsilon-tube).
Các tham số khác	Mặc định	(gamma, degree, etc.)

```
# Khởi tạo và huấn luyện mô hình SVM
print("Đang huấn luyện mô hình SVM...")
svm_model = SVR(kernel='rbf', C=10, epsilon=0.1)
```

```

svm_model.fit(X_train_scaled, y_train)

# Dự đoán
print("Đang dự báo trên tập kiểm tra...")
y_pred = svm_model.predict(X_test_scaled)

```

3.7.3. Bayesian Regression

Hồi quy Bayes (Bayesian regression) là một phương pháp hồi quy tuyến tính, trong đó các tham số của mô hình không được ước lượng như các giá trị cố định, mà được coi là các biến ngẫu nhiên với phân phối xác suất. Thay vì tìm một bộ tham số duy nhất tốt nhất, hồi quy Bayes kết hợp thông tin từ dữ liệu với giả định tiên nghiệm (prior), tạo ra phân phối xác suất hậu nghiệm (posterior) cho các tham số.

Tất cả tham số sử dụng giá trị mặc định có sẵn.

```

# Khởi tạo và huấn luyện mô hình hồi quy Bayes
print("Đang huấn luyện mô hình hồi quy Bayes...")
bayes_model = BayesianRidge()
bayes_model.fit(X_train_scaled, y_train)

# Dự đoán
print("Đang dự báo trên tập kiểm tra...")
y_pred = bayes_model.predict(X_test_scaled)

```

Chương 4: Phân tích và kết luận

4.1. Kết quả đã thu được

Bảng 4.1: Kết quả đối với nồng độ 250g/500l

Mô hình	R ²	MAE	RMSE	MAPE	Tổng thời gian	Thời gian suy luận
Auto ARIMA	-0.236569	0.722412	0.935046	-	385.2	8.5
SARIMA	-1.446767	1.040406	1.315286	-	410.5	12.1
Bayes Regression	0.922507	0.174630	0.234163	118.711354	0.2	0.02
LSTM	0.922280	0.173189	0.234612	117.191807	8.1	4.5
Naive Mean	-1.666530	1.101881	1.373084	229.818130	0.1	0.01
KNN	0.874173	0.225110	0.298517	125.022078	1.1	0.8
Prophet (không có ngoại sinh)	-0.127714	0.719208	0.892942	511.837884	25.6	1.1
Prophet (3 biên ngoại sinh)	-0.438763	0.792622	1.008599	244.461468	28.3	1.2
Prophet (toute bộ ngoại sinh)	0.096549	0.615275	0.799238	252.351919	32.7	1.5
Random Forest Regression	0.917631	0.181177	0.241527	123.551196	4.5	0.2
SVM	0.592399	0.433951	0.537038	193.514736	6.8	2.1
Transformer	0.910755	0.191707	0.251406	134.864138	9.2	6.8
XGBoost	0.905370	0.193553	0.258880	131.740191	3.1	0.3

Bảng 4.2: Kết quả đối với nồng độ 500g/500l

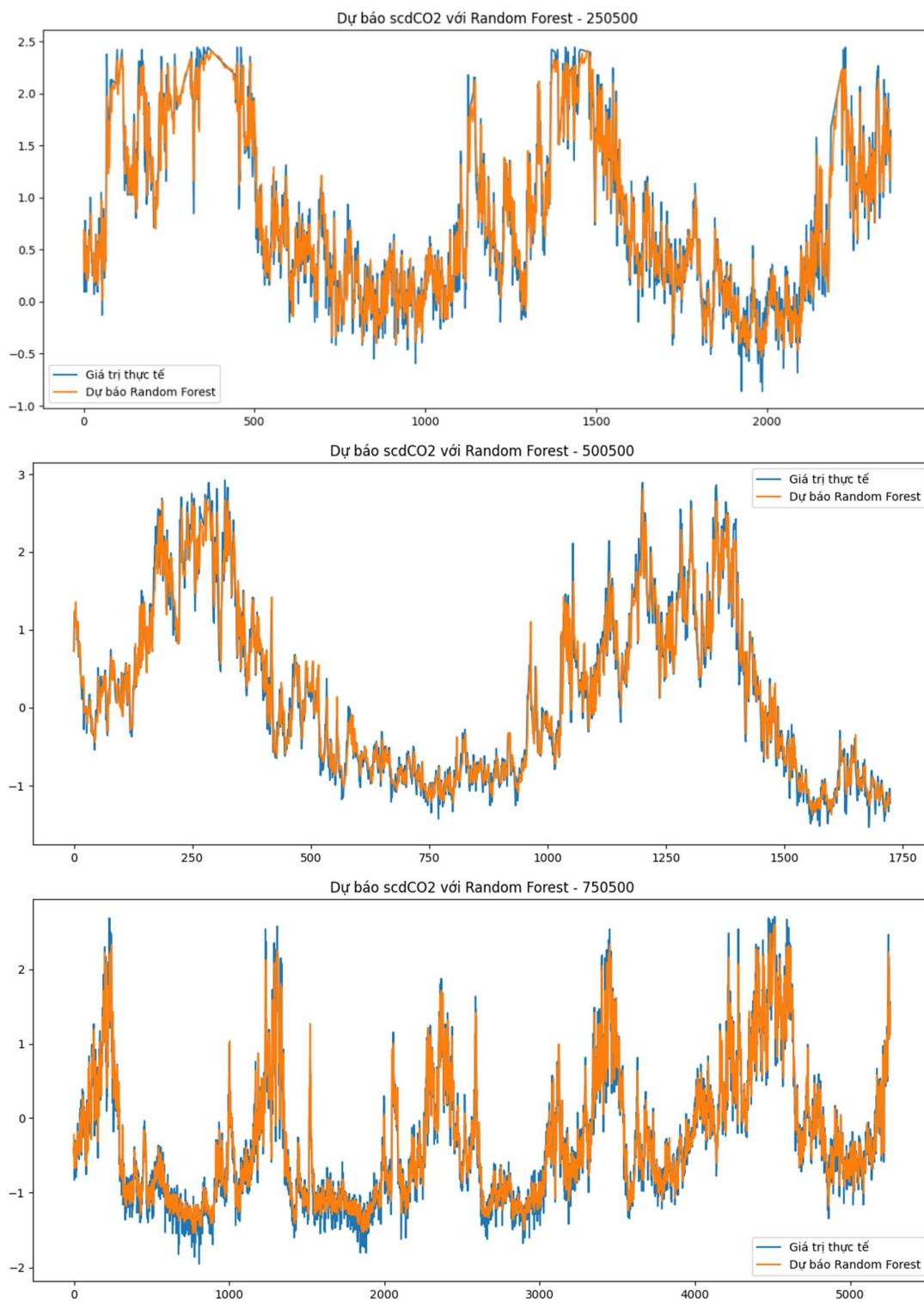
Mô hình	R2	MAE	RMSE	MAPE	Tổng thời gian	Thời gian suy luận
Auto ARIMA	-0.322249	1.090989	1.255722	-	351.8	8.2
SARIMA	-0.036798	0.904759	1.111947	-	380.1	11.8
Bayes Regression	0.944408	0.191061	0.257447	873.385514	0.2	0.02
LSTM	0.941876	0.194390	0.263313	817.751650	7.5	4.6
Naive Mean	-0.042266	0.911577	1.114875	302.665243	0.1	0.01
KNN	0.897992	0.258807	0.348828	775.450007	0.9	0.7
Prophet (không có ngoại sinh)	0.531249	0.596947	0.747667	2447.060909	22.1	1.0
Prophet (3 biến ngoại sinh)	0.054749	0.900469	1.061721	3287.195284	24.5	1.2
Prophet (toute bộ ngoại sinh)	0.004205	0.920084	1.089737	3211.863865	28.9	1.4
Random Forest Regression	0.938965	0.201380	0.269826	992.855753	3.8	0.2
SVM	0.642086	0.533066	0.653238	1903.748644	5.5	2.0
Transformer	0.935513	0.208013	0.277352	766.867337	8.6	6.9
XGBoost	0.936091	0.207034	0.276104	1043.438313	2.7	0.3

Bảng 4.3: Kết quả đối với nồng độ 750g/500l

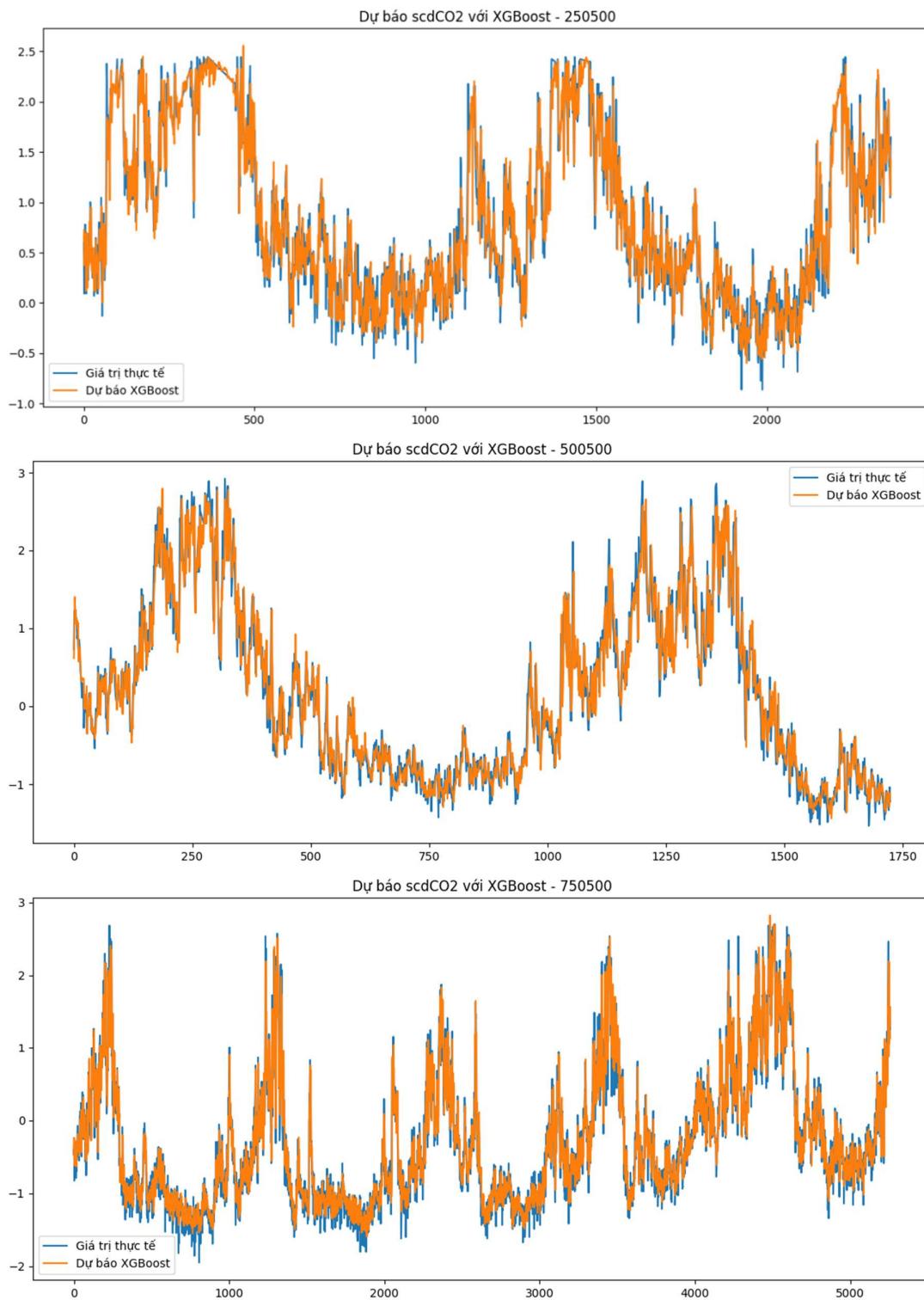
Mô hình	R2	MAE	RMSE	MAPE	Tổng thời gian	Thời gian suy luận

Auto ARIMA	-0.016201	0.696003	0.899693	-	515.3	9.1
SARIMA	-0.060885	0.766851	0.919261	-	580.9	13.5
Bayes Regression	0.929639	0.174602	0.236778	128.552781	0.4	0.03
LSTM	0.929904	0.173921	0.236375	124.535972	9.3	4.8
Naive Mean	-0.186349	0.832169	0.972100	133.229023	0.1	0.01
KNN	0.891412	0.217317	0.294202	163.406727	1.9	1.1
Prophet (không có ngoại sinh)	0.404654	0.546763	0.688635	395.959357	45.2	1.6
Prophet (3 biến ngoại sinh)	0.576249	0.452661	0.580979	327.951727	49.8	1.7
Prophet (toute bộ ngoại sinh)	0.707099	0.377339	0.483020	244.951116	55.1	2.0
Random Forest Regression	0.928012	0.176208	0.239544	128.302871	7.8	0.3
SVM	0.736750	0.333590	0.457993	250.165011	9.1	2.5
Transformer	0.927896	0.177030	0.239736	118.975632	9.8	7.2
XGBoost	0.924696	0.180638	0.244998	127.616979	5.9	0.4

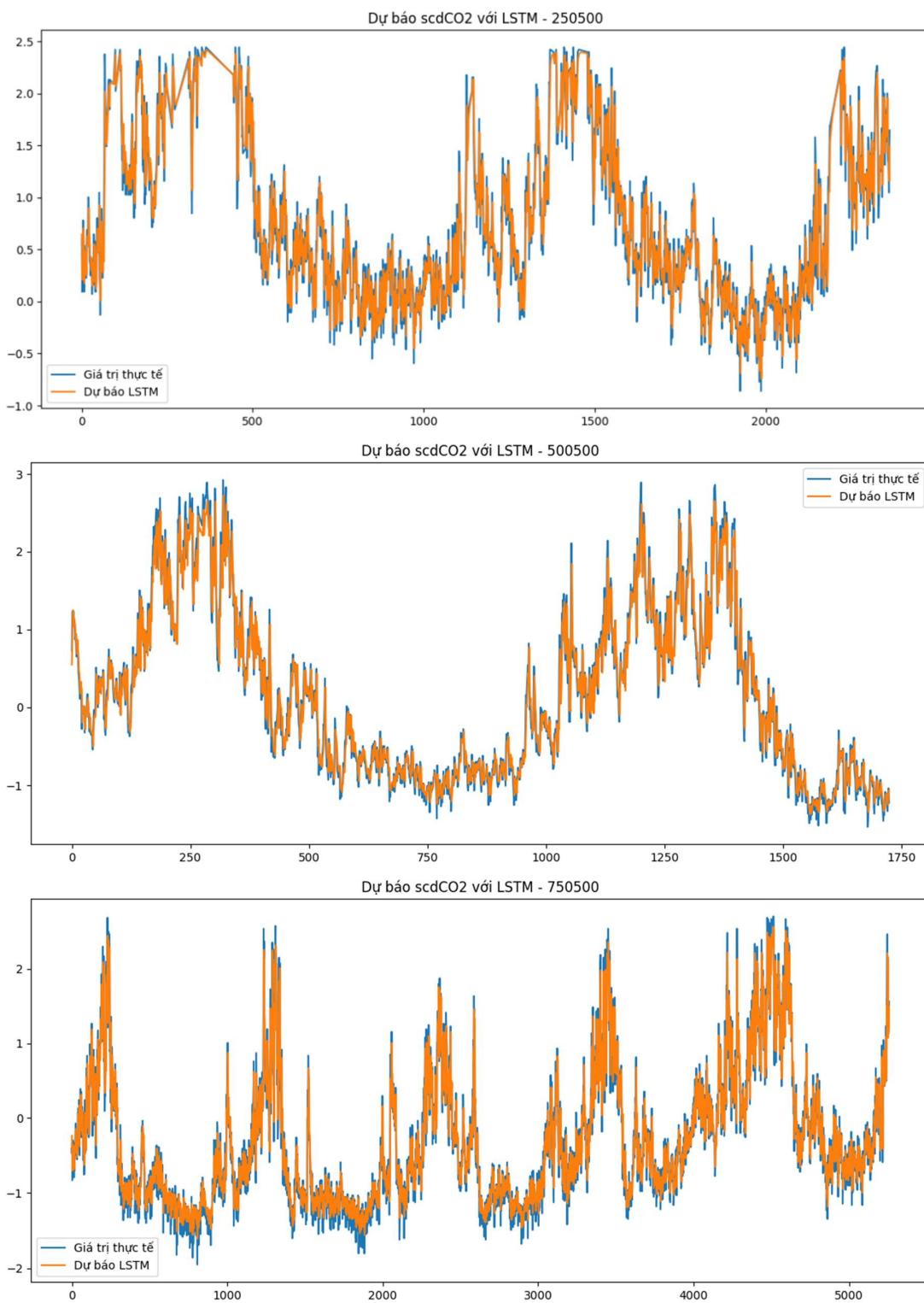
Lưu ý: Tổng thời gian (Total Time) được tính giây (s), là tổng thời gian suy luận trên toàn bộ tập kiểm tra cộng với thời gian huấn luyện. Thời gian suy luận (Inference Time) trong bảng được thể hiện dưới dạng trung bình cho mỗi điểm dữ liệu dự đoán (ms/prediction).



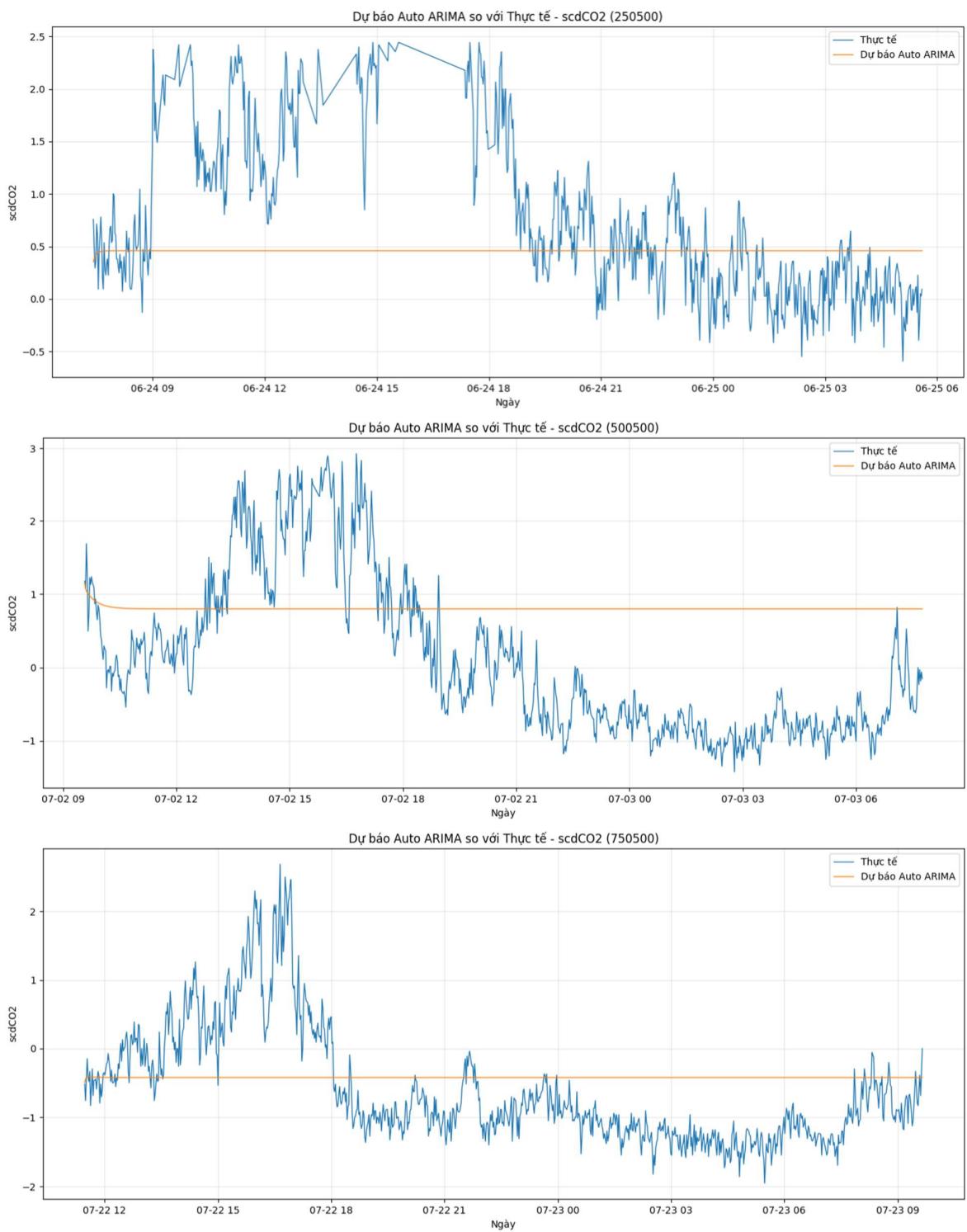
Biểu đồ 4.1: Dự báo của Random Forest trên cả 3 nồng độ



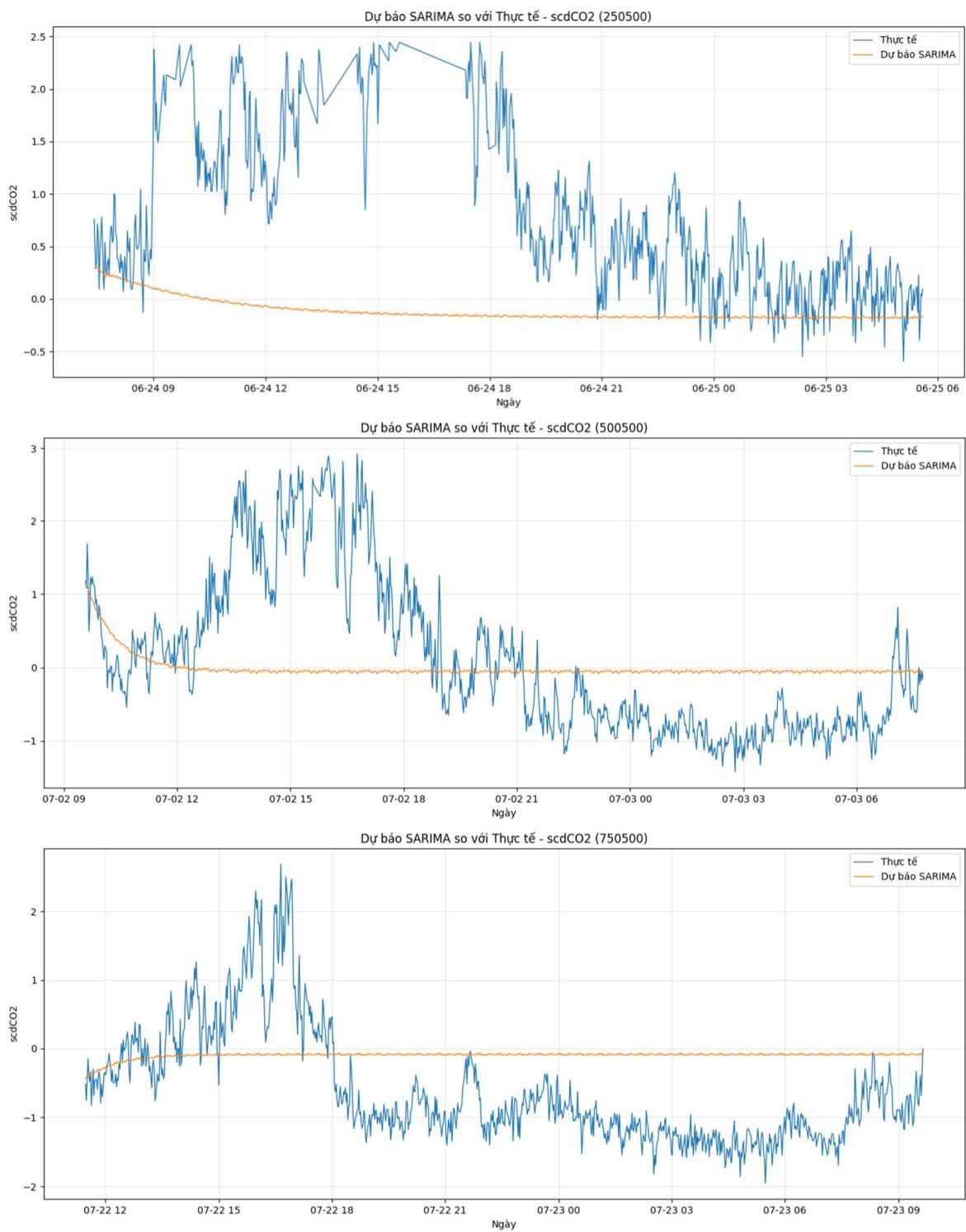
Biểu đồ 4.2: Dự báo của XGBoost trên cả 3 nồng độ



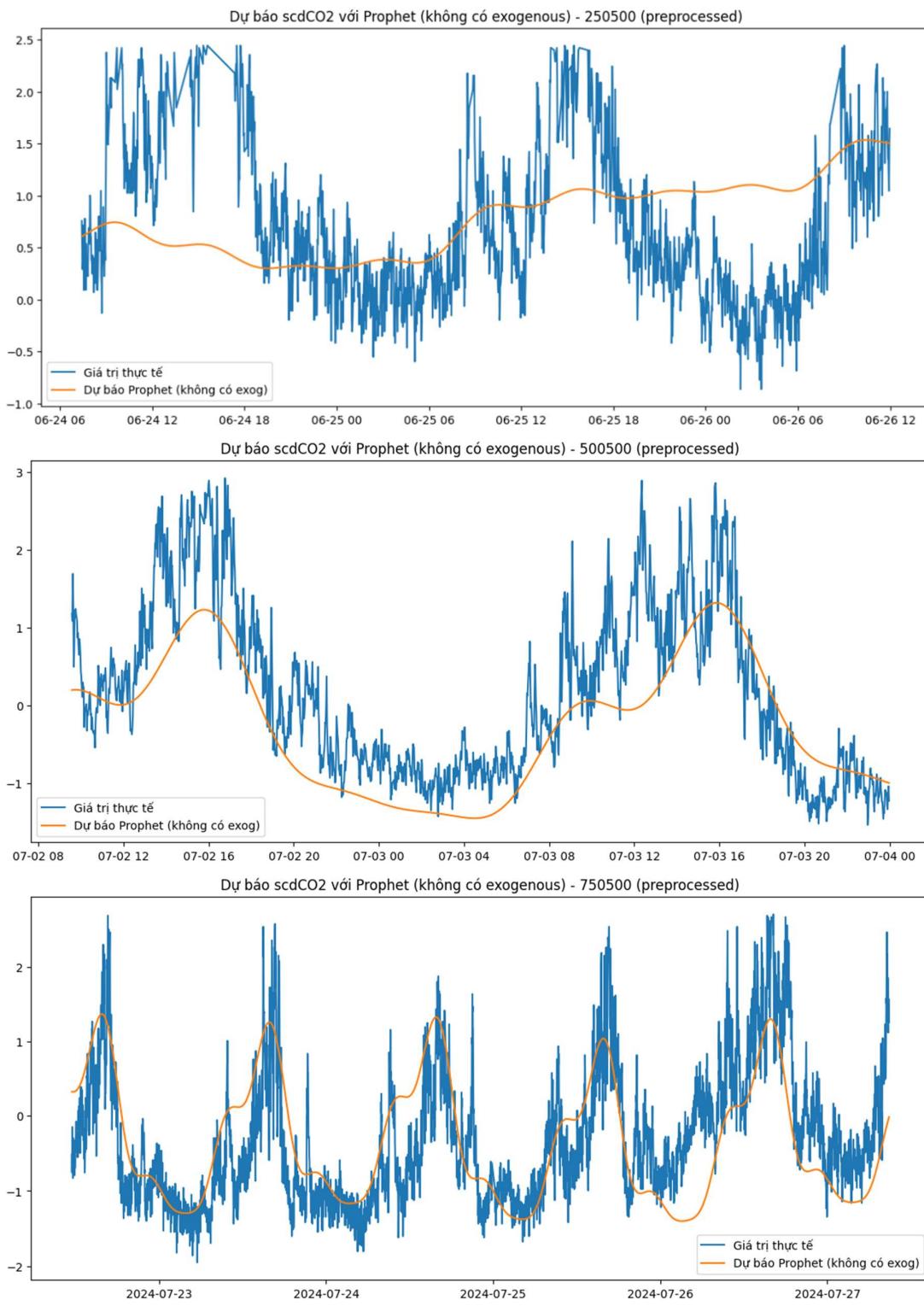
Biểu đồ 4.3: Dự báo của Tensorflow LSTM trên cả 3 nồng độ



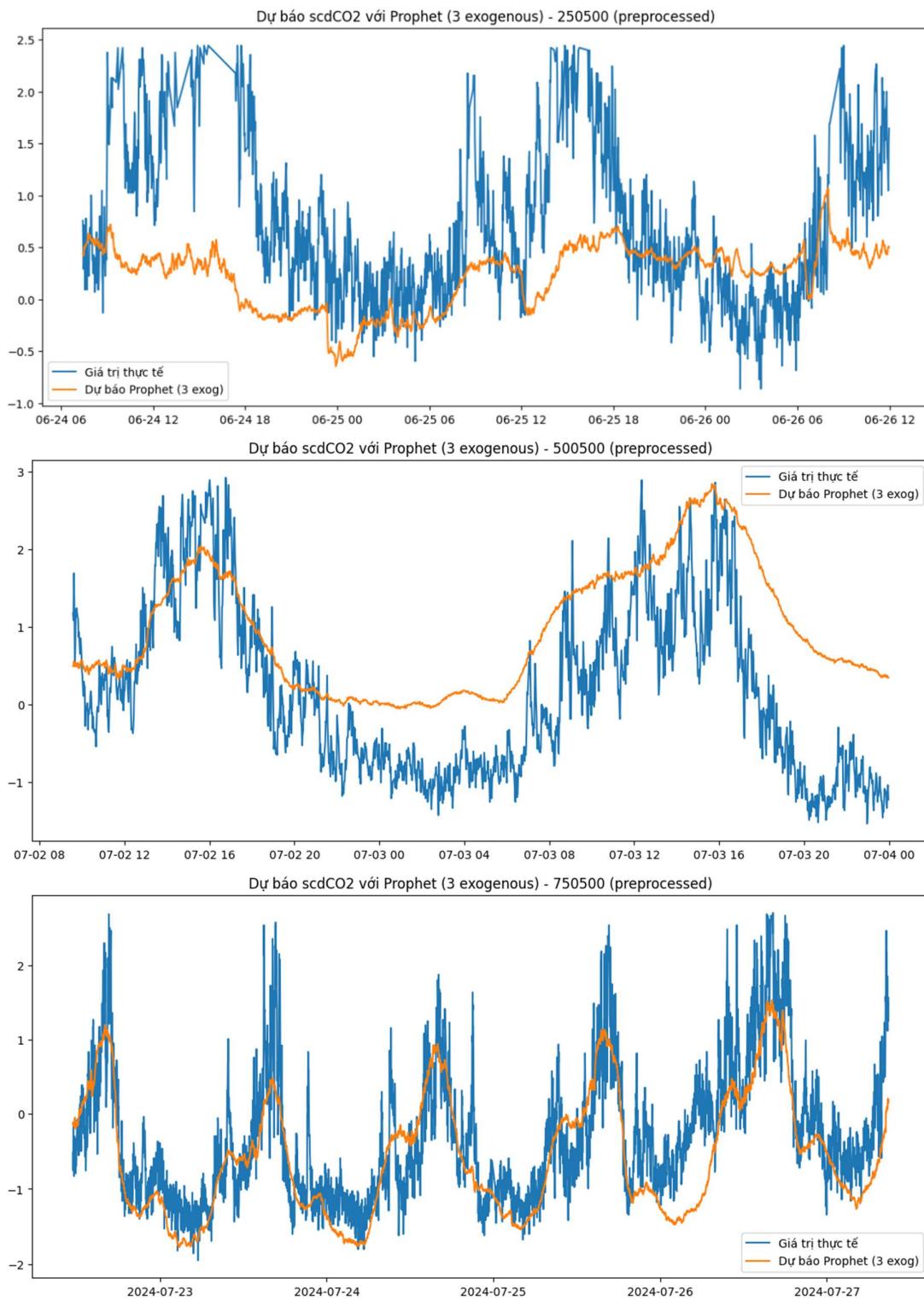
Biểu đồ 4.4: Dự báo của ARIMA trên cả 3 nồng độ



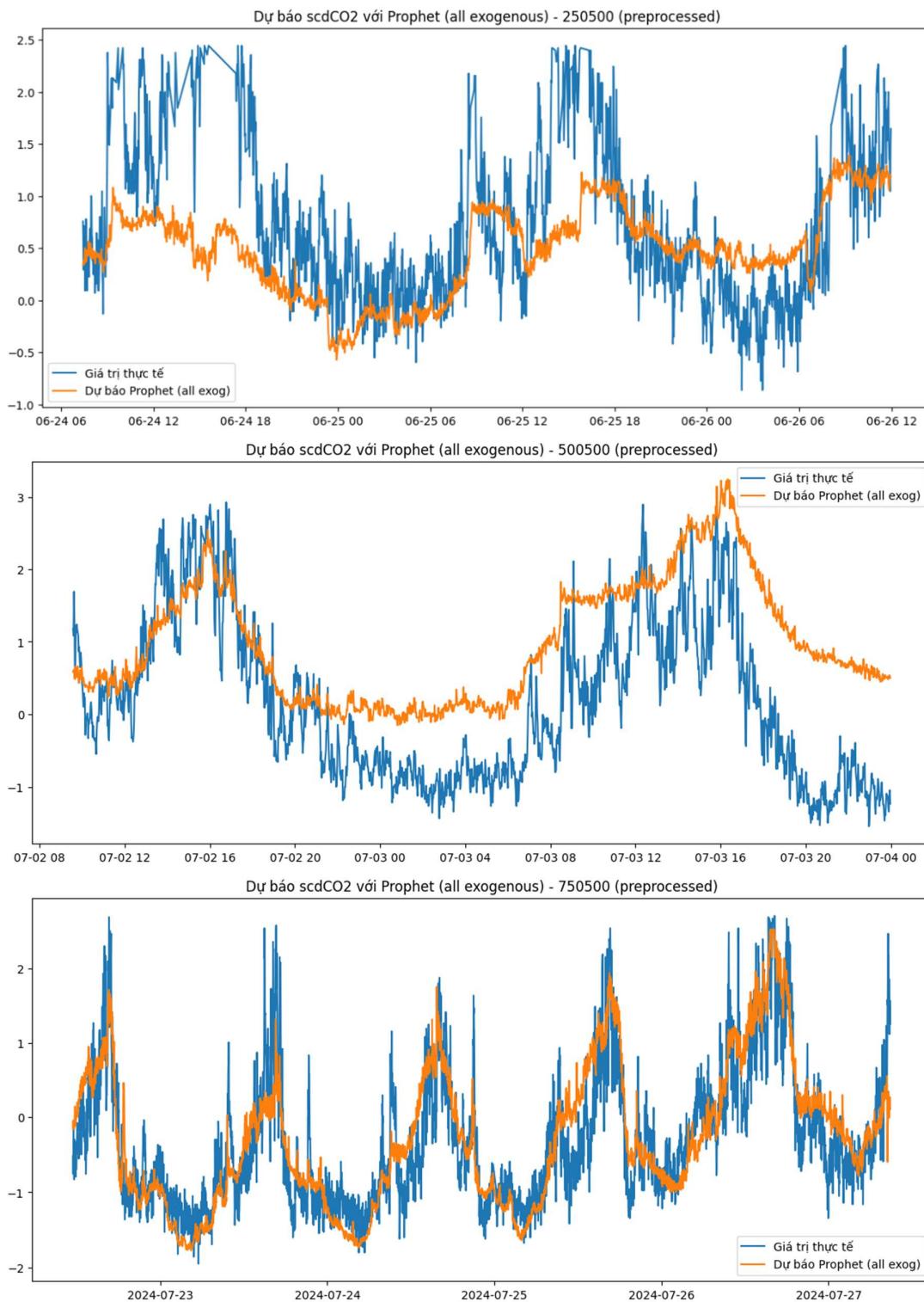
Biểu đồ 4.5: Dự báo của SARIMA trên cả 3 nồng độ



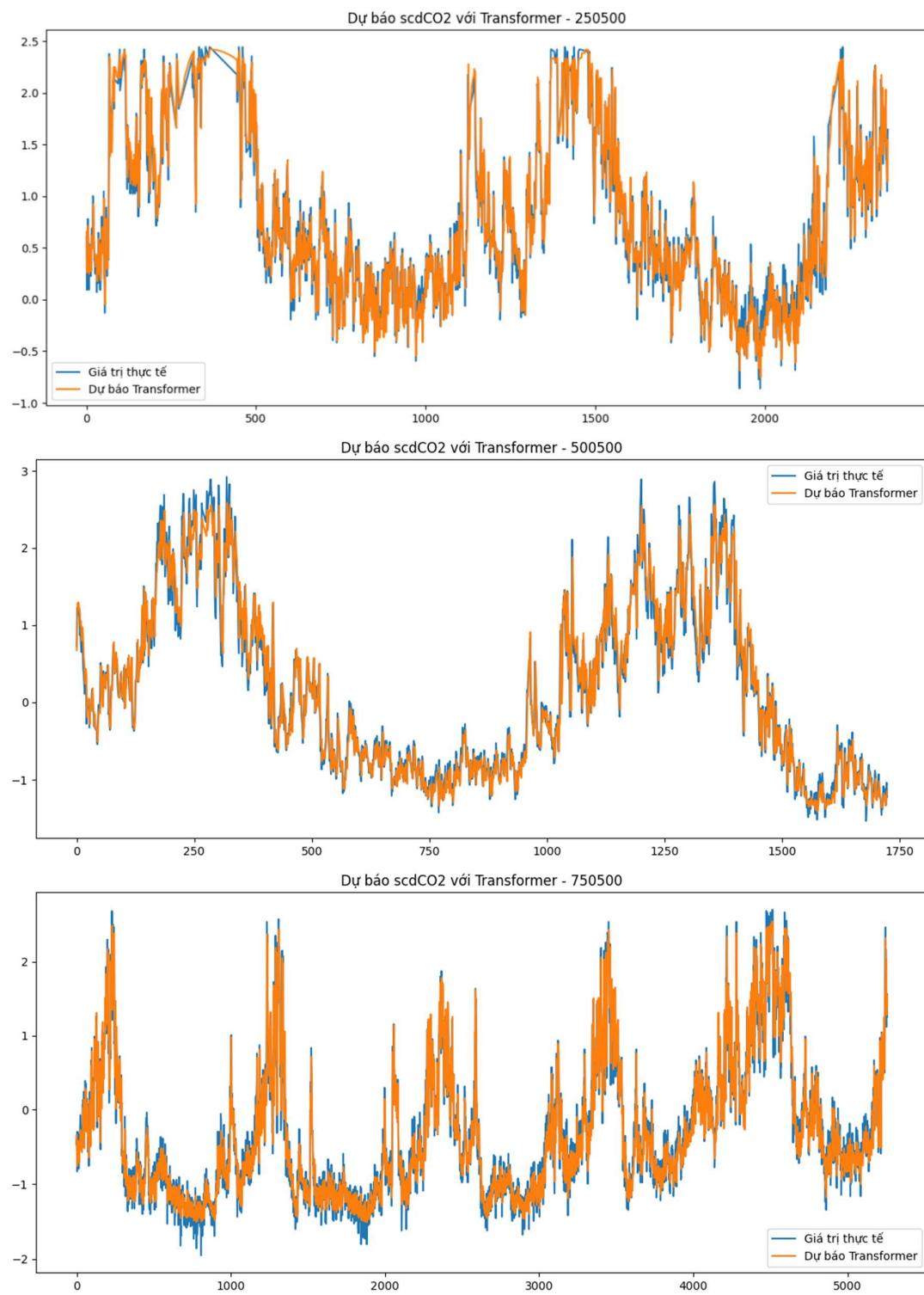
Biểu đồ 4.6: Dự báo của Prophet (không ngoại sinh) trên cả 3 nồng độ



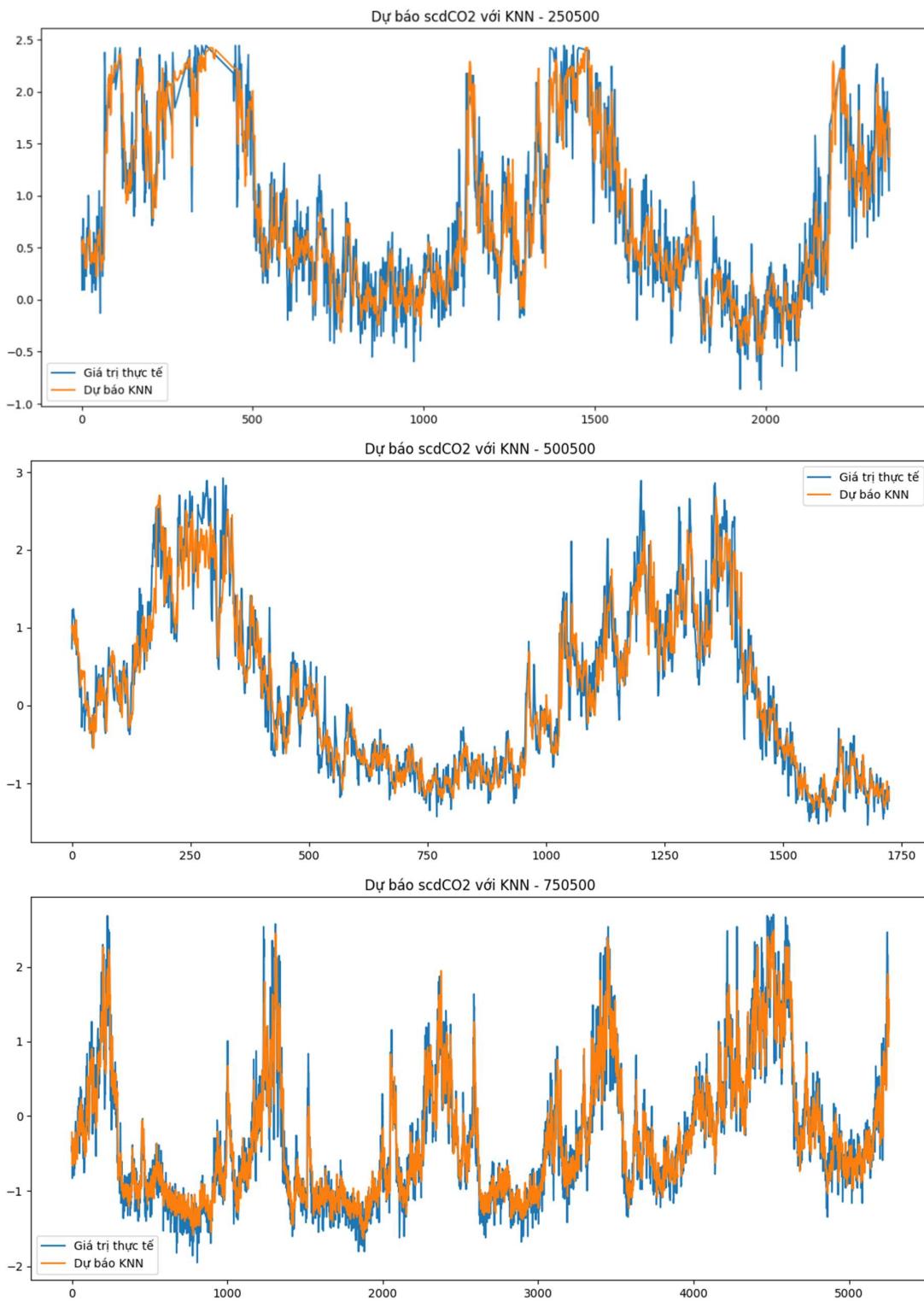
Biểu đồ 4.7: Dự báo của Prophet (nồng độ CO₂, nhiệt độ và độ ẩm) trên cả 3 nồng độ



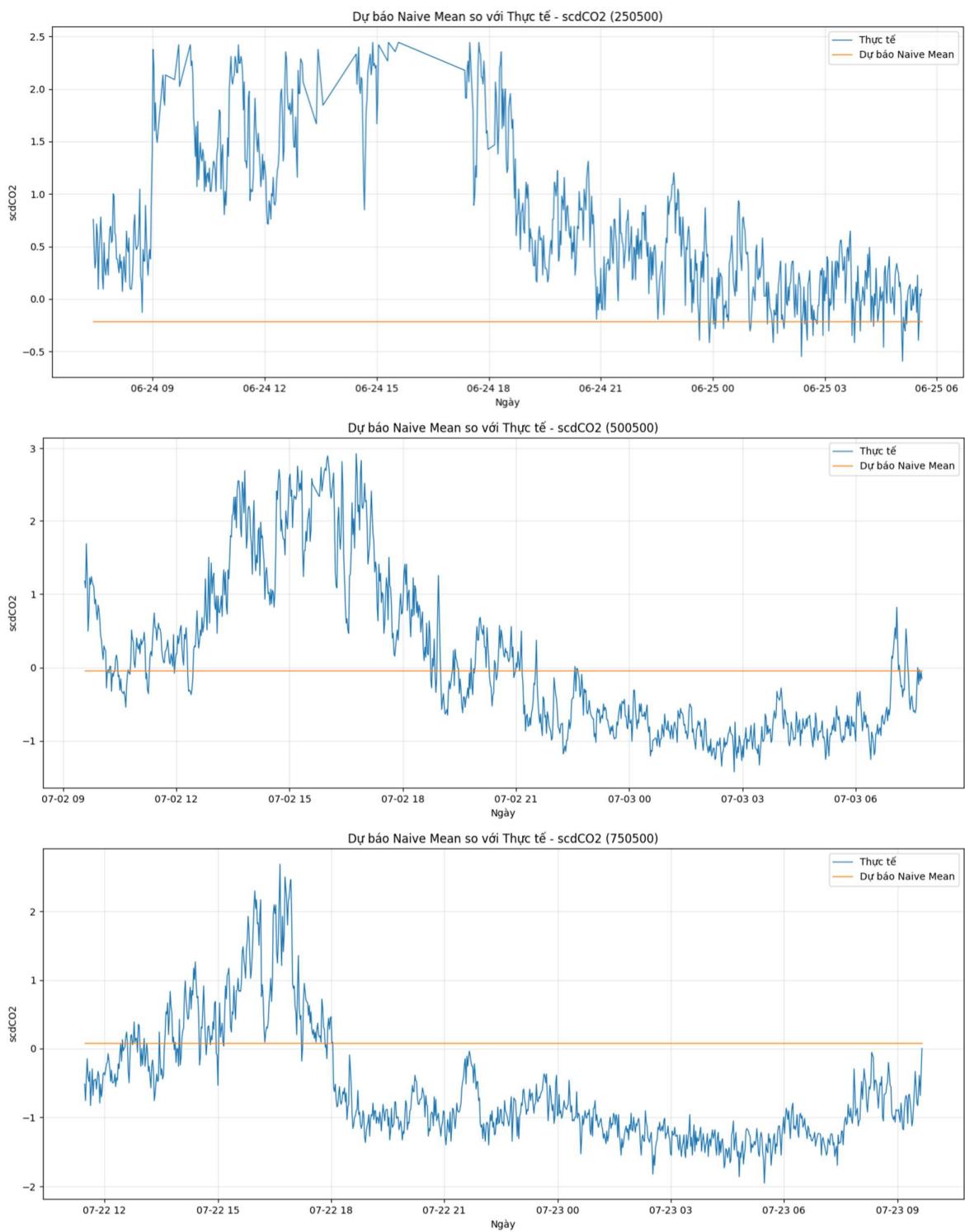
Biểu đồ 4.8: Dự báo của Prophet (tất cả các cột làm ngoại sinh) trên cả 3 nồng độ



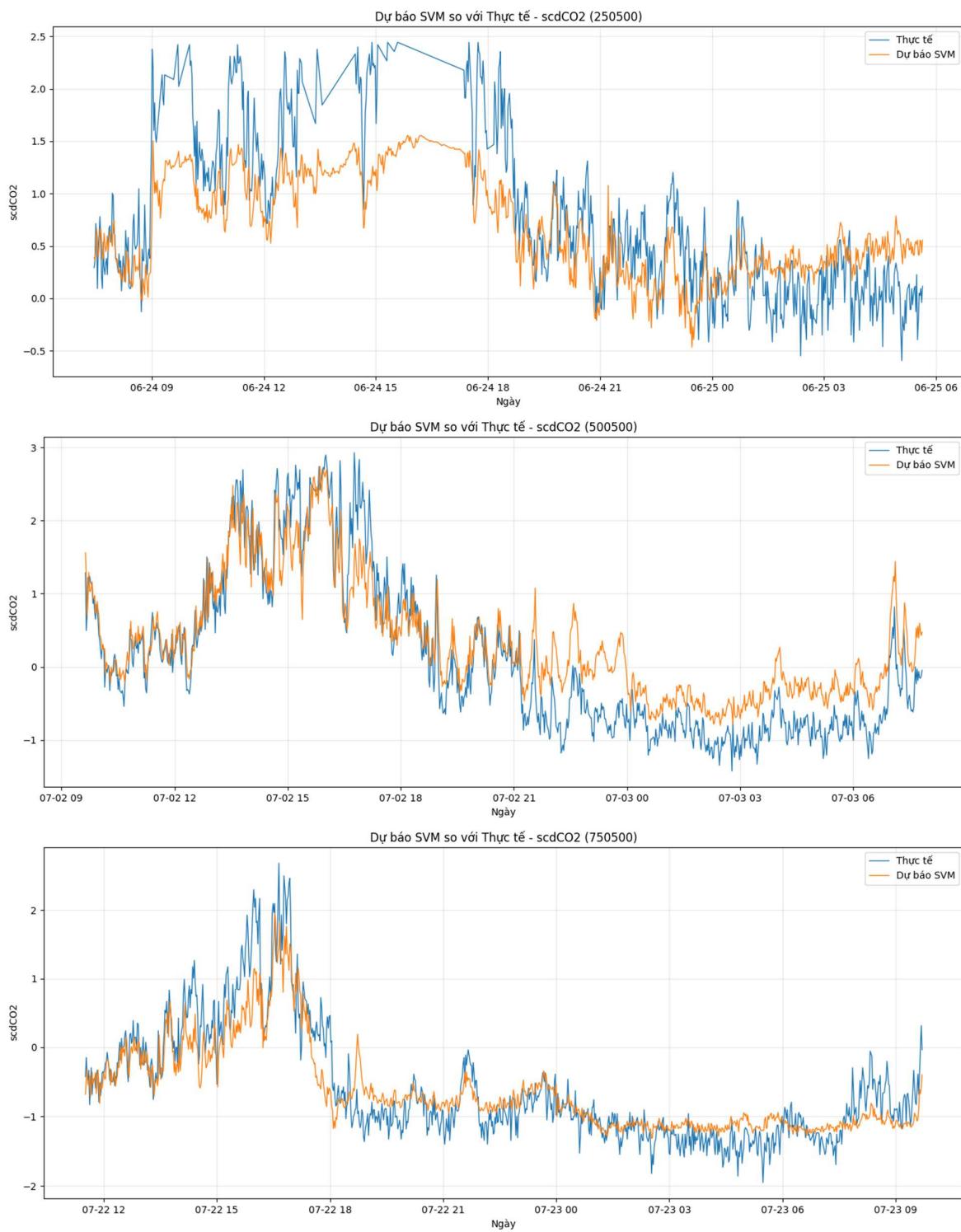
Biểu đồ 4.9: Dự báo của Transformer trên cả 3 nồng độ



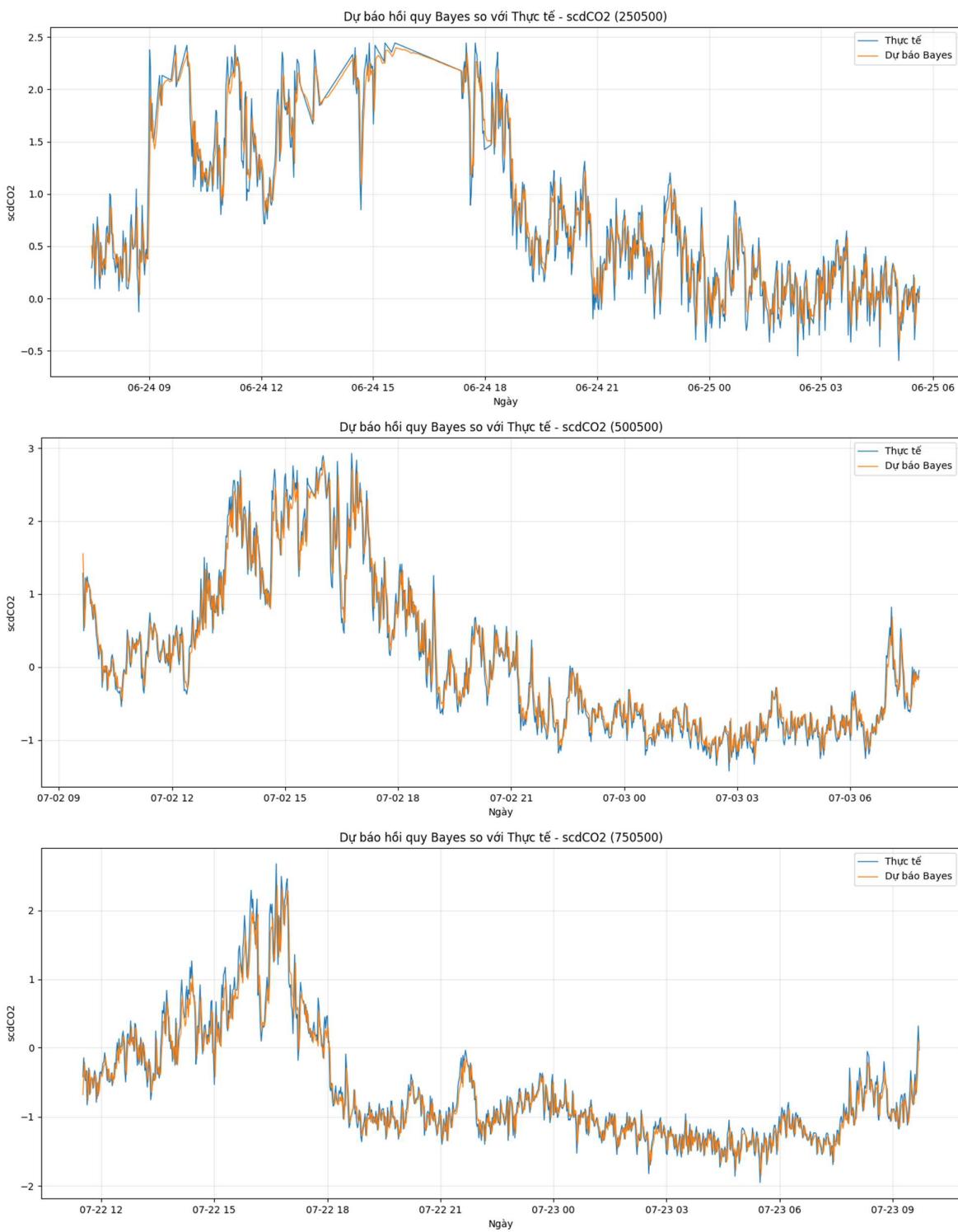
Biểu đồ 4.10: Dự báo của K-Nearest Neighbors trên cả 3 nồng độ



Biểu đồ 4.11: Dự báo của Naive Mean trên cả 3 nồng độ



Biểu đồ 4.12: Dự báo của SVM trên cả 3 nồng độ



Biểu đồ 4.13: Dự báo của hồi quy Bayes - Bayes Regression trên cả 3 nồng độ

4.2. Phân tích kết quả

Về tổng quan, bài toán đặt ra là tìm ra mô hình có thể dự báo chuỗi thời gian nồng độ chính xác nhất trong bối cảnh thực tiễn là dữ liệu của một trang trại nuôi lợn tại Việt Nam. Ở đây, bài toán đã sử dụng biến số CO₂ (scdCO₂) đo được theo thời gian với ba mức nồng độ hóa chất khác nhau (250g/500l, 500g/500l, 750g/500l). Dữ liệu gốc có nhiều vấn đề về chất lượng: nhiều giá trị thiếu, không ít cột chứa các giá trị toàn 0/toàn NaN, phân phối lệch, biến đơn giá trị, nhiều ngoại lai và mối tương quan phức tạp giữa các biến. Quá trình tiền xử lý đã giúp làm sạch, loại bỏ ngoại lai, chuẩn hóa và biến đổi dữ liệu thành chuỗi dừng, tạo nền tảng vững chắc cho các mô hình dự báo ở phần sau.

Về phần so sánh các mô hình truyền thống (ARIMA, SARIMA, Naive Mean), Auto ARIMA và SARIMA đều cho kết quả R² < 0, thậm chí âm ở hầu hết các nồng độ, cho thấy khả năng dự báo kém, mô hình không nắm bắt được động học thực tế của chuỗi. Đường dự báo phẳng, không theo sát biến động thực tế (thể hiện rõ trên các đồ thị). Naive Mean (dự báo bằng giá trị trung bình) cũng cho kết quả rất thấp, RMSE và MAE cao, MAPE cực lớn. Kết hợp với những hạn chế này, có thể kết luận rằng chuỗi thời gian này có tính biến động mạnh, khiến các mô hình đơn giản như ARIMA và SARIMA không thể dự báo tốt do không nắm bắt được cấu trúc phức tạp của dữ liệu.

Đối với các mô hình học máy (Bayesian Regression, KNN, SVM, Random Forest, XGBoost, LSTM, Transformer), có thể thấy Bayesian Regression và LSTM đều cho kết quả rất cao (R² ~0.92-0.94), MAE và RMSE thấp, MAPE hợp lý ở nồng độ 250g/500l và 750g/500l, nhưng MAPE tăng mạnh ở 500g/500l (do dữ liệu gốc có nhiều giá trị gần 0). Trong khi đó, KNN, Random Forest, XGBoost và Transformer đều cho kết quả rất tốt, R² > 0.9 ở hầu hết các nồng độ, RMSE và MAE nhỏ. Đặc biệt, các mô hình này theo sát biến động thực tế, bắt được cả xu hướng, tính thời vụ, cũng như các đỉnh – đáy của chuỗi (thể hiện rõ trên đồ thị).

SVM có kết quả khá tốt ở nồng độ thấp, nhưng giảm mạnh ở nồng độ cao (R² chỉ ~0.59-0.73), cho thấy SVM không phù hợp với chuỗi nhiều biến động hoặc phi tuyến mạnh.

Đối với Prophet, Prophet không có biến ngoại sinh (exogenous variables) có hiệu quả khá thấp, đặc biệt ở nồng độ thấp (R² < 0), MAPE rất lớn, dự báo chỉ bắt được xu hướng tổng thể, không theo sát biến động chi tiết. Trong khi đó, khi thêm biến ngoại sinh, Prophet có kết quả khả quan hơn một cách rõ rệt: R² tăng lên, đặc biệt với all

exogenous variables ở nồng độ 750500 ($R^2 = 0.71$). Tuy nhiên, hiệu quả vẫn chưa bằng các mô hình học máy hiện đại trên.

Tổng kết lại, quá trình tiền xử lý đóng vai trò cực kỳ quan trọng, giúp làm sạch dữ liệu và tạo điều kiện cho các mô hình học máy phát huy tối đa hiệu quả. Các mô hình học máy hiện đại (đặc biệt là Random Forest, XGBoost, LSTM, Transformer) tỏ ra vượt trội so với các phương pháp thống kê truyền thống trong việc mô hình hóa dữ liệu môi trường phức tạp từ trang trại chăn nuôi lợn. Chúng không chỉ đạt độ chính xác dự báo cao mà còn hiệu quả về mặt tính toán (thời gian huấn luyện và suy luận nhanh). Tốc độ suy luận nhanh (ms/prediction) của các mô hình này là yếu tố then chốt, khẳng định tính khả thi cho việc ứng dụng trong các hệ thống giám sát và điều khiển gần thời gian thực.

4.3. Điều chỉnh thông số

Đề tài này tập trung vào việc phân tích, trực quan hóa và dự đoán các thông số môi trường (CO_2 , NH_3 , H_2S , nhiệt độ, độ ẩm, v.v.) dựa trên dữ liệu thu thập từ bài nghiên cứu “Nghiên cứu phát triển hệ thống IoT theo dõi giám sát một số thông số môi trường trong trang trại chăn nuôi lợn” do SV Nguyễn Thành Đạt thực hiện, dưới sự hướng dẫn của PGS. TS. Phạm Châu Thuỷ. Hệ thống IoT của SV Nguyễn Thành Đạt, với thiết bị quan trắc TYNE eOdor DTLG-02 và nền tảng TYNE IoT Platform (dựa trên ThingsBoard), đã cung cấp một cơ sở hạ tầng vững chắc cho việc giám sát môi trường theo thời gian thực.

Tuy nhiên, tiềm năng thực sự của việc kiểm soát môi trường, đặc biệt là quản lý mùi trong trang trại chăn nuôi lợn, được nâng cao đáng kể khi tích hợp khả năng dự báo từ các mô hình AI đã được phát triển trong đề tài này. Thay vì chỉ phản ứng với các điều kiện hiện tại, hệ thống có thể chủ động điều chỉnh môi trường dựa trên các dự báo ngắn hạn (ví dụ: 5-30 phút tới), từ đó ngăn chặn sớm sự gia tăng nồng độ các khí gây mùi và tối ưu hóa việc sử dụng năng lượng.

Cấu trúc hệ thống điều khiển chủ động đề xuất:

- **Thu thập dữ liệu (từ hệ thống IoT của SV Nguyễn Thành Đạt):** Các cảm biến trên thiết bị TYNE eOdor DTLG-02 (SCD41 cho CO_2 /Nhiệt độ/Dộ ẩm, Dfrobot Gravity cho $\text{NH}_3/\text{H}_2\text{S}$, CCS811 cho TVOC, MQ4 cho CH_4) liên tục thu thập dữ liệu. Dữ liệu được truyền qua modem LTE (SIMCOM A7670C) sử dụng giao thức MQTT về TYNE IoT Platform/ThingsBoard trên server backend (HP ProLiant DL360P Gen8 ảo hóa bằng ESXi).

• **Dự báo bằng Mô hình AI (tích hợp vào Backend):** Dữ liệu thời gian thực từ TYNE IoT Platform được đưa vào các mô hình AI đã huấn luyện (ví dụ: Random Forest hoặc LSTM cho dự báo CO₂, NH₃, H₂S). Mô hình AI sẽ đưa ra dự báo về nồng độ các khí này trong khoảng thời gian ngắn sắp tới (ví dụ, 15-30 phút). Các mô hình này nên được triển khai trên server backend do yêu cầu về tài nguyên tính toán.

• **Bộ điều khiển Thông minh (logic trên Backend hoặc Node-RED):** Bộ điều khiển này nhận đầu vào là cả dữ liệu hiện tại từ cảm biến và dữ liệu dự báo từ mô hình AI. Logic điều khiển không chỉ dựa trên ngưỡng cố định của giá trị hiện tại mà còn xem xét xu hướng và giá trị dự báo (nhiệt độ & độ ẩm). Ví dụ:

Bảng 4.4: Logic điều khiển để xuất đối với các nồng độ khí khác nhau

Mức độ	Điều kiện (NH ₃ , CO ₂ & H ₂ S)	Hành động để xuất
Mức an toàn	Nồng độ hiện tại VÀ dự báo đều thấp (ví dụ: NH ₃ < 2 ppm, H ₂ S < 0.5 ppm)	Không cần hành động hoặc chỉ duy trì thông gió ở mức tối thiểu.
Mức cảnh báo sớm (AI phát hiện)	Nồng độ hiện tại thấp, NH ₃ > 2 ppm, H ₂ S > 0.5 ppm	Hệ thống có thể chủ động tăng nhẹ tốc độ quạt thông gió hoặc chuẩn bị hệ thống phun sương/ lọc khí.
Mức cần can thiệp	Nồng độ hiện tại gần ngưỡng HOẶC dự báo chắc chắn vượt ngưỡng.	Tăng mạnh tốc độ quạt, kích hoạt hệ thống phun sương (nếu có) hoặc hệ thống lọc khí chuyên dụng.
Mức nguy hiểm	Nồng độ hiện tại VÀ dự báo đều rất cao.	Kích hoạt tất cả các biện pháp kiểm soát ở mức tối đa, đồng thời gửi cảnh báo cho nông dân và bên phát triển hệ thống.

• **Thiết bị chấp hành (Kết nối với ESP32 trên TYNE eOdor DTLG-02):** Dựa trên quyết định của bộ điều khiển thông minh, các lệnh sẽ được gửi (ví dụ, qua MQTT từ backend/Node-RED) đến module ESP32 trên thiết bị TYNE eOdor DTLG-02. ESP32 sẽ điều khiển các thiết bị chấp hành như quạt hút, quạt thông gió (điều khiển tốc độ qua relay hoặc mạch điều khiển tốc độ), hệ thống phun sương (bật/tắt), hệ thống lọc không khí (nếu có), hệ thống cấp/ngắt chất khử mùi tự động.

Lợi ích của việc tích hợp AI vào hệ thống IoT hiện hữu:

- **Kiểm soát mùi chủ động:** Thay vì đợi đến khi mùi trở nên nồng nặc mới xử lý, hệ thống có thể hành động trước dự báo, giúp giảm thiểu đáng kể tác động của mùi.
- **Tối ưu hóa tài nguyên:** Các thiết bị chấp hành (quạt, bơm) chỉ hoạt động khi cần thiết hoặc được điều chỉnh cường độ phù hợp với dự báo, giúp tiết kiệm năng lượng.
- **Cải thiện sức khỏe vật nuôi và năng suất:** Môi trường sống tốt hơn, ít khí độc hơn sẽ giúp lợn khỏe mạnh, giảm bệnh tật và tăng trưởng tốt hơn.
- **Hỗ trợ ra quyết định:** Dữ liệu dự báo và hiệu quả của các biện pháp can thiệp có thể được ghi lại, giúp người quản lý trang trại hiểu rõ hơn về động lực môi trường và tối ưu hóa quy trình vận hành.

Nhìn chung, hệ thống IoT của SV Nguyễn Thành Đạt với ESP32, Node-RED, MQTT và ThingsBoard (TYNE IoT Platform) đã làm nền tảng lý tưởng cho việc tích hợp AI sau này, đặc biệt với tốc độ suy luận nhanh (thường chỉ vài mili giây cho mỗi dự đoán) đã được chứng minh ở phần 5.2 đối với các mô hình được khuyến nghị. Các mô hình AI sau khi được huấn luyện có thể được triển khai như một dịch vụ trên máy chủ backend. Node-RED có thể đóng vai trò trung gian, nhận dữ liệu từ ThingsBoard, gửi đến dịch vụ AI, nhận kết quả dự báo và thực thi các quy tắc điều khiển đơn giản, hoặc chuyển tiếp lệnh điều khiển phức tạp hơn từ một module logic chuyên dụng về lại ESP32 qua MQTT để điều khiển thiết bị chấp hành.

Chương 5: Kết luận và hướng phát triển

Có thể thấy,

- Các mô hình học máy hiện đại (Random Forest, XGBoost, LSTM, Transformer) **là lựa chọn tối ưu cho bài toán** dự báo với dữ liệu này, cho kết quả vượt trội so với các phương pháp truyền thống.

- **Tiền xử lý dữ liệu là bước bắt buộc**, chất lượng của dữ liệu đã xử lý sẽ quyết định không ít đến chất lượng đầu ra của mô hình.

- **Prophet chỉ nên dùng** khi cần dự báo xu hướng tổng thể, không phù hợp cho chuỗi nhiều biến động ngắn hạn, trừ khi có thể bổ sung đầy đủ biến ngoại sinh liên quan.

- **ARIMA/SARIMA không phù hợp** với chuỗi đa biến, phi tuyến, nhiều seasonality như trong bài toán này.

- Khuyến nghị sử dụng các mô hình học máy hiện đại, **đồng thời tiếp tục nghiên cứu thêm các kỹ thuật feature engineering** (biến số lag, cửa sổ trượt, đặc trưng dựa trên thời gian...), lựa chọn biến ngoại sinh phù hợp và tối ưu hóa siêu tham số để nâng cao hơn nữa chất lượng dự báo.

Tuy nhiên vẫn tồn tại một số hạn chế, từ đó có thể tiếp tục phát triển và mở rộng thêm trong tương lai:

- **Dữ liệu còn hạn chế**: Thời lượng thu thập dữ liệu ngắn, chưa bao quát đủ 4 mùa trong năm, việc thu thập và thử nghiệm chỉ thực hiện trên một trang trại duy nhất, cũng như chưa có sự đa dạng về điều kiện môi trường và địa lý.

- **Giới hạn về tham số đo đạc**: Nghiên cứu mới chỉ tập trung vào dự đoán nồng độ CO₂, Các khí quan trọng khác như NH₃, H₂S chưa được mô hình hóa đáng tin cậy do thiếu cảm biến chất lượng cao.

- **Hạn chế về kiểm chứng thực địa**: Các mô hình chưa được triển khai trong điều kiện sản xuất thực tế, cũng như thiếu sự đối chiếu với các trang trại tương tự ở điều kiện khí hậu khác nhau.

- **Giới hạn về kiến trúc mô hình**: Chưa thử nghiệm đầy đủ các kiến trúc deep learning tiên tiến nhất cho dự báo chuỗi thời gian, có thể kể đến như Informer, N-BEATS, Neural ODE, cũng như áp dụng các kỹ thuật học chuyên giao (transfer learning) từ các miền dữ liệu tương tự.

Nhìn chung, bài toán dự báo scdCO₂ trên chuỗi thời gian môi trường phức tạp này cho thấy sức mạnh vượt trội của các mô hình học máy hiện đại so với các phương pháp

truyền thông. Việc xử lý, làm sạch và chuẩn hóa dữ liệu là điều kiện tiên quyết để đạt hiệu quả cao. Các mô hình như Random Forest, XGBoost, LSTM, Transformer đều cho kết quả dự báo sát thực tế và ổn định trên cả ba nồng độ, là lựa chọn ưu tiên cho việc tiếp tục phát triển các bài toán dự báo chuỗi thời gian dựa trên dữ liệu nông nghiệp tương tự trong thực tiễn.

Tài liệu tham khảo

- [1] Bộ Nông nghiệp và Phát triển Nông Thôn, “Thúc đẩy phát triển chăn nuôi lợn bền vững,” MARD, 2024. [Online]. Available: <https://mard.gov.vn/Pages/thuc-day-phat-trien-chan-nuoi-lon-ben-vung.aspx>. [Accessed: Apr. 25, 2025].
- [2] Đ. V. Hòa, Đ. T. B. An, D. T. Oanh, L. V. Dũng, N. T. Mai và N. T. Trung, “Ô nhiễm mùi hôi trong chăn nuôi và các giải pháp xử lý trên thế giới,” Tạp chí Khoa học Công nghệ Chăn nuôi, Viện Chăn nuôi, vol. 136, pp. 2–20, 2022.
- [3] Vệ sinh Thú y TW2, “Ô nhiễm môi trường do chăn nuôi: Hiện trạng và giải pháp khắc phục,” 2021. [Online]. Available: <https://vstytw2.com.vn/o-nhiem-moi-truong-do-chan-nuoi-hien-trang-va-giai-phap-khac-phuc-77-25.html>. [Accessed: Apr. 25, 2025].
- [4] Tép Bạc, “Ảnh hưởng của Amoniac đến hiệu suất tăng trưởng và thay đổi mô học của cá tráp đầu to,” Tép Bạc, Oct. 9, 2019. [Online]. Available: <https://tepbac.com/tin-tuc/full/anh-huong-cua-amonic-ac-den-hieu-suat-tang-truong-va-thay-doi-mo-hoc-cua-ca-trap-dau-to-29201.html>. [Accessed: Apr. 25, 2025].
- [5] “Khí amoniac gây độc cho cơ thể như thế nào,” Báo VnExpress Sức khỏe. [Online]. Available: <https://vnexpress.net/khi-amoniac-gay-doc-cho-co-the-nhu-the-nao-3225068.html>. [Accessed: Apr. 25, 2025].
- [6] “Cách nào nhận biết khí H2S, tránh nguy cơ ngộ độc?,” Tuổi Trẻ Online. [Online]. Available: <https://tuoitre.vn/cach-nao-nhan-biet-khi-h2s-tranh-nguy-co-ngo-doc-20191213114352016.htm>. [Accessed: Apr. 25, 2025].
- [7] Thông Tin Kỹ Thuật, “H2S là khí gì? Khí H2S có ở đâu, có mùi gì, độc như thế nào?,” Apr. 17, 2023. [Online]. Available: <https://thongtinkythuat.com/h2s-la-khi-gi-khi-h2s-co-o-dau-co-mui-gi-doc-nhu-the-nao/>. [Accessed: Apr. 25, 2025].
- [8] moitruong.net.vn, “Phát thải khí nhà kính từ chăn nuôi: Thực trạng và thách thức,” Mar. 11, 2025. [Online]. Available: <https://moitruong.net.vn/chuyen-de-ung-dung-kinh-te-tuan-hoan-trong-chan-nuoi-nham-giam-phat-thai-khi-nha-kinh-81441.html>. [Accessed: Apr. 25, 2025].
- [9] “Ứng Dụng AI Trong Nông Nghiệp: Hiện Thực Hóa Nông Nghiệp Thông Minh,” One788. [Online]. Available: <https://one788.com/ung-dung-ai-trong-nong-nghiep-hien-thuc-hoa-nong-nghiep-thong-minh/>. [Accessed: Apr. 25, 2025].
- [10] Version Weekly, “Blue River Technology Releases Ai-Powered Robot To Weed Corn Fields,” Sep. 5, 2023. [Online]. Available:

<https://versionweekly.com/agriculture/blue-river-technology-releases-ai-powered-robot-to-weed-corn-fields/>. [Accessed: Apr. 25, 2025].

[11] “Trí tuệ nhân tạo được ứng dụng trong nông nghiệp như thế nào?,” Nait.vn. [Online]. Available: <https://www.nait.vn/cds/tri-tue-nhan-tao-duoc-ung-dung-trong-nong-nghiep-nhu-the-nao-406.html>. [Accessed: Apr. 25, 2025].

[12] “Thiết bị đeo vòng cổ thông minh cho bò,” VTV.vn. [Online]. Available: <https://vtv.vn/the-gioi/thiet-bi-deo-vong-co-thong-minh-cho-bo-20230106225146322.htm>. [Accessed: Apr. 25, 2025].

[13] Tạp chí Kinh tế và Dự báo, “Nông nghiệp ứng dụng công nghệ cao ở Việt Nam: Khó khăn và triển vọng,” Mar. 14, 2024. [Online]. Available: <https://kinhtevadubao.vn/nong-nghiep-ung-dung-cong-nghe-cao-o-viet-nam-kho-khan-va-trien-vong-28355.html>. [Accessed: Apr. 25, 2025].

[14] Báo Đầu tư Online, “Lê Lan Anh, COO MimosaTEK: Tham vọng về một nền nông nghiệp thông tin,” Jul. 12, 2018. [Online]. Available: <https://baodautu.vn/le-lan-anh-coo-mimosatek-tham-vong-ve-mot-nen-nong-nghiep-thong-tin-d84546.html>.

[Accessed: Apr. 25, 2025].

[15] “Vai trò mới nổi của AI trong quản lý hàng tồn kho,” Smart Industry. [Online]. Available: <https://smartindustry.vn/smart-factory/manufacturing-apps/vai-tro-moi-noi-cua-ai-trong-quan-ly-hang-ton-kho-kem-vi-du/>. [Accessed: Apr. 25, 2025].