

ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN CHUỖI THỜI GIAN

**Xây dựng mô hình dự báo
chuỗi thời gian đa biến
bằng các mô hình Học sâu**

Nguyễn Anh Minh - 20194417

Nguyễn Đình Nhật - 20190080

Lê Đức Tài - 20195163

Ngành: Toán Tin

Giảng viên hướng dẫn: TS. Nguyễn Thị Ngọc Anh
Viện: Toán ứng dụng và Tin học

Chữ ký của GVHD

HÀ NỘI, 3/2023

Mục lục

1 Giới thiệu	3
1.1 Tổng quan đề tài	3
1.2 Cấu trúc bài báo cáo	4
2 Lý thuyết cơ bản chuỗi thời gian	9
2.1 Ví dụ chuỗi thời gian	9
2.1.1 Chuỗi thời gian đa biến	11
2.2 Tính chất cơ bản của chuỗi thời gian	13
2.3 Các mô hình dự báo chuỗi thời gian đơn biến cơ bản	15
2.3.1 Mô hình tự hồi quy AR	15
2.3.2 Mô hình trung bình trượt MA	17
2.3.3 Mô hình ARMA	18
2.3.4 Mô hình ARIMA	19
3 Giới thiệu về một số mạng trong Học sâu	20
3.1 Tổng quan mạng Neural	20
3.2 Sequence Modeling: Mạng LSTM và GRU	21
3.3 Mạng Convolution	27
3.4 Mạng nơ ron đồ thị thời gian quang phổ StemGNN	30
3.4.1 Lớp tương quan tiềm ẩn	31
3.4.2 Mạng StemGNN	31
4 Khảo sát các bài báo về chủ đề dự báo chuỗi thời gian	33
4.1 Mô hình dự báo dựa trên mô hình thống kê	33
4.2 Mô hình dự báo dựa trên mô hình học máy	34
4.3 Mô hình dự báo dựa trên mạng Deep Neural	34

5 Phương pháp luận	36
5.1 Dự báo chuỗi thời gian dựa trên các mô hình thống kê truyền thống	36
5.1.1 Mô hình VAR	36
5.1.2 Mô hình VARMA	37
5.2 Dự báo chuỗi thời gian dựa trên các mô hình học sâu	38
5.2.1 Mô hình dự báo chuỗi thời gian dựa vào mạng tích chập với các chuỗi thời gian tương quan	38
5.2.2 Mạng Hồi quy hai lớp kết hợp phương pháp Attention [1]	43
5.3 Dự báo chuỗi thời gian dựa trên các mô hình Hybrid	47
5.3.1 Mô hình hóa các mẫu thời gian dài và ngắn hạn bằng Mạng nơ-ron .	47
6 Kết quả thực nghiệm	50
6.1 Tiền xử lý dữ liệu	50
6.2 Khai phá dữ liệu	52
6.3 Kết quả chạy mô hình	59
6.3.1 VAR	60
6.3.2 VARMA	62
6.3.3 XGBoost	63
6.3.4 LSTM	65
6.3.5 LSTM AutoEncoder	66
6.3.6 LSTM Autoencoder kết hợp kỹ thuật Attention	68
6.3.7 Mô hình Dilated Convolution	70
6.3.8 Dilated Convolution kết hợp khử trend	72
6.3.9 StemGNN	74
6.3.10 Convolution kết hợp LSTM và khử trend	76
6.3.11 Nhận xét	78
7 Kết luận	80

1 Giới thiệu

1.1 *Tổng quan về tài*

Dự báo chuỗi thời gian đa biến là quá trình ước tính giá trị tương lai của một chuỗi dữ liệu thời gian, trong đó có nhiều biến đầu vào được ảnh hưởng lẫn nhau. Nó là một phương pháp phân tích dữ liệu mạnh mẽ được sử dụng trong nhiều lĩnh vực, từ kinh tế học đến kỹ thuật, khoa học xã hội, v.v.

Dự báo chuỗi thời gian đa biến là một phương pháp mô hình hóa các mối quan hệ giữa nhiều biến đầu vào, đồng thời sử dụng các giá trị lịch sử của các biến để dự đoán các giá trị trong tương lai. Các mô hình dự báo chuỗi thời gian đa biến thường được phát triển bằng cách sử dụng các phương pháp thống kê và học máy để tìm ra các mối quan hệ giữa các biến đầu vào và dự đoán các giá trị đầu ra.

Trong báo cáo này, nhóm tác giả sẽ sử dụng đồng thời các mô hình thống kê truyền thống như VAR, VARMA, các mô hình học sâu như mạng LSTM, mạng GNN, mạng Convolution và các mô hình Hybrid như Convolution kết hợp phương pháp tách trend bằng Linear Filter để dự báo chuỗi thời gian đa biến chất lượng nước thải.

Tính mới của nhóm được trình bày cụ thể trong Phần 6.3 này như sau:

1. Hiểu ý tưởng được sử dụng trong các bài báo xây dựng mô hình dự báo chuỗi thời gian đa biến và hoàn toàn tự đề xuất và xây dựng lại các mô hình bằng framework Tensorflow.
2. Khai phá dữ liệu và nhận thấy các đặc trưng của dữ liệu để xây dựng các mô hình phù hợp.
3. Hyper-parameter tuning và xây dựng kiến trúc phù hợp với dữ liệu để dự báo đạt kết quả tốt nhất.

1.2 Cấu trúc bài báo cáo

Nội dung của báo cáo được chia thành 7 phần như sau:

1. **Giới thiệu:** giới thiệu tổng quan đề tài.
2. **Lý thuyết cơ bản chuỗi thời gian:** Trình bày một số kiến thức cơ bản cần có để dự báo chuỗi thời gian đa biến
3. **Giới thiệu về một số mạng trong Học sâu:** Trình bày kiến trúc của các layer trong mô hình học sâu như LSTM, GNN, Convolution.
4. **Khảo sát các bài báo về chủ đề dự báo chuỗi thời gian:** Survey một số bài báo về dự báo chuỗi thời gian đơn biến cũng như đa biến và rút ra ý tưởng về phương pháp xây dựng các mô hình dự báo.
5. **Phương pháp luận:** Trình bày chi tiết cách xây dựng các mô hình dự báo chuỗi thời gian đa biến bằng nhóm mô hình truyền thống, Học sâu, Học máy, Hybrid.
6. **Kết quả thực nghiệm:** Trình bày chi tiết cách khai phá dữ liệu được cung cấp và cách xử lý dữ liệu. Đồng thời trình bày cách xây dựng các mô hình dự báo chuỗi thời gian đa biến và đánh giá kết quả.
7. **Kết luận.**

Trong báo cáo này, nhóm tác giả sử dụng một số tên viết tắt cho các thuật ngữ như sau:

LSTM	Long-Short Term Memory
GNN	Graph Neural Network
Conv	Convolution layer
GRU	Gated Recurrent Unit
MAE	Mean Absolute Error
RMSE	Root Mean Square Error

Bảng 1: Bảng thuật ngữ viết tắt

Dù đã rất cố gắng xong báo cáo này vẫn không tránh khỏi những hạn chế cần khắc phục. Vì vậy, tác giả rất mong quý thầy cô đưa ra những ý kiến góp ý bổ ích để đồ án này tiếp tục được phát triển và có những kết quả mới tốt hơn.

Lời cảm ơn

Báo cáo này được thực hiện và hoàn thành tại Đại học Bách Khoa Hà Nội, nằm trong nội dung học phần *Chuỗi thời gian* của kì học 2022-1.

Tác giả xin được dành lời cảm ơn chân thành tới TS. Nguyễn Thị Ngọc Anh, là giảng viên đã trực tiếp hướng dẫn và khuyến nghị cho tác giả để tài rất thú vị này, đồng thời cô cũng đã giúp đỡ tận tình và có những đóng góp bổ ích để tác giả có thể hoàn thành báo cáo này một cách tốt nhất.

Danh sách hình vẽ

1	Doanh số hàng tháng của cửa hàng	9
2	Ví dụ về một random walk đơn giản $\{S_t, t = 0, 1, 2, \dots, 200\}$	11
3	Đồ thị về tính mùa trong chuỗi thời gian doanh số bán kem của một cửa hàng	13
4	Đồ thị về tính xu hướng trong chuỗi thời gian của chuỗi giá.	14
5	Kiến trúc cơ bản của mạng MLP	20
6	Computational Graph	21
7	Computational graph biểu diễn RNN	22
8	Kiến trúc Encoder-Decoder	23
9	Kiến trúc tế bào của mạng LSTM	24
10	So sánh mạng Neural và mạng Convolution	29
11	Mô hình StemGNN	30
12	Mạng nơ ron tích chập dilated với ba lớp	39
13	Kiến trúc mạng Condition Convolution	42
14	Cơ chế input attention	44
15	Cơ chế Temporal Attention	45
16	Hình ảnh data ban đầu	50
17	Tỷ lệ null	51
18	Chu kỳ của biến dow	52
19	Histogram của các trường dữ liệu	53
20	Correlation giữa các biến	54
21	Kiểm định tính nhân quả Granger	55
22	Kiểm định tính Cointegration	55
23	Hình ảnh dữ liệu	56
24	Tách trend, seasonal của biến pH và DO	57
25	ACF và PACF của DO, pH	58

26	Đặc trưng dữ liệu của tập Test	59
27	Hệ số mô hình VAR	60
28	Đồ thị so sánh kết quả	61
29	Đồ thị dự báo mô hình VARMA	62
30	Dạng dữ liệu đầu vào	63
31	Đồ thị dự báo của mô hình XGBoost	64
32	Mô hình LSTM	65
33	Đồ thị dự báo LSTM	66
34	Mô hình LSTM AutoEncoder	67
35	Ý tưởng của mô hình AutoEncoder	67
36	Đồ thị dự báo của mô hình LSTM AutoEncoder	68
37	Mô hình LSTM Autoencoder kết hợp kỹ thuật Attention	69
38	Đồ thị dự báo của mô hình LSTM Autoencoder Attention	70
39	Mô hình Dilated Convolution	71
40	Đồ thị dự báo của mô hình	72
41	Mô hình Dilated Convolution kết hợp khử trend	73
42	Đồ thị dự báo của mô hình Dilated Convolution kết hợp Khử trend	74
43	Mô hình StemGNN	75
44	Đồ thị dự báo của mô hình StemGNN	76
45	Mô hình Convolution kết hợp LSTM và khử trend	77
46	Đồ thị dự báo của mô hình Conv + LSTM + Trend Decomposition	78

Danh sách bảng

1	Bảng thuật ngữ viết tắt	4
2	Kết quả chạy mô hình VAR	61
3	Kết quả chạy mô hình VARMA	63
4	Kết quả chạy mô hình XGBoost	64
5	Kết quả chạy mô hình LSTM	65
6	Kết quả LSTM AutoEncoder	67
7	Kết quả chạy mô hình LSTM AutoEncoder kết hợp kỹ thuật Attention . . .	69
8	Kết quả mô hình Dilated Convolution	71
9	Kết quả chạy mô hình Dilated Convolution kết hợp khử trend	73
10	Kết quả chạy mô hình StemGNN	75
11	Kết quả chạy mô hình Convolution kết hợp LSTM và khử trend	77

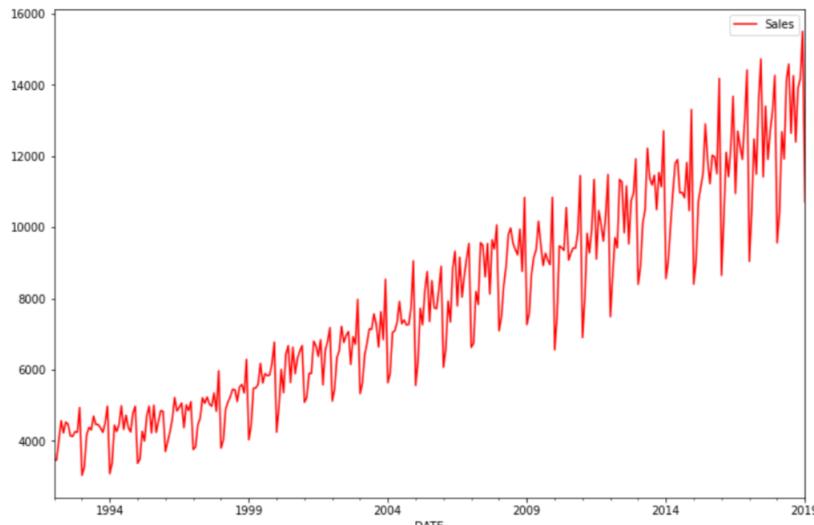
2 Lý thuyết cơ bản chuỗi thời gian

Trong phần này, nhóm tác giả trình bày một số kiến thức cơ bản cần có để dự báo chuỗi thời gian đa biến.

2.1 Ví dụ chuỗi thời gian

Chuỗi thời gian là tập hợp x_t , mỗi quan sát được ghi lại tại một thời điểm cụ thể t . Chuỗi thời gian có thể liên tục hoặc rời rạc. Chuỗi thời gian rời rạc là tập hợp các quan sát được ghi lại tại các khoảng thời gian cố định. Nếu các quan sát được ghi lại liên tục trong khoảng thời gian thì được gọi là chuỗi thời gian liên tục.

Ví dụ 1: Hình 1 thể hiện doanh số bán hàng rượu Alcohol của cửa hàng từ tháng 1 năm 1992 đến tháng 1 năm 2019. Trong bộ dữ liệu gồm có 325 quan sát $\{(Tháng 1. 1992), (Tháng 2. 1992), \dots, (Tháng 1. 2019)\}$. Biểu đồ doanh số bán hàng có xu hướng tăng và có một mô hình theo mùa với điểm cao nhất vào tháng 6 và điểm thấp nhất vào tháng 1.



Hình 1: Doanh số hàng tháng của cửa hàng

Phân tích chuỗi thời gian là các phương pháp để phân tích dữ liệu chuỗi thời gian nhằm trích xuất các thông tin thống kê có ý nghĩa và các đặc trưng khác của dữ liệu. Dự báo chuỗi thời gian là việc sử dụng mô hình để dự đoán các giá trị tương lai dựa trên các giá trị quan sát trước đó.

Mô hình chuỗi thời gian cho các quan sát $\{x_t\}$ là xác định phân phối đồng thời (hoặc có thể là giá trị trung bình và hiệp phương sai) của chuỗi các biến ngẫu nhiên $\{X_t\}$. Mô hình

chuỗi thời gian xác suất cho chuỗi các biến ngẫu nhiên $\{X_1, X_2, \dots\}$ xác định phân phối xác suất đồng thời của vector ngẫu nhiên $(X_1, X_2, \dots, X_n)^T; n = 1, 2, \dots$ hoặc tương đương với xác suất

$$P(X_1 \leq x_1, \dots, X_n \leq x_n), x_i \in \mathbb{R}, n = 1, 2, \dots$$

IID noise

Mô hình đơn giản nhất của một chuỗi thời gian là mô hình không có xu hướng và không có thành phần mùa, các quan sát đơn giản chỉ là các biến ngẫu nhiên độc lập, có cùng phân phối và có trung bình bằng không. Ta gọi chuỗi các biến ngẫu nhiên X_1, X_2, \dots như vậy là *IID noise*. Với bất kỳ số nguyên dương n và các số thực x_1, x_2, \dots, x_n , ta có thể viết

$$P[X_1 \leq x_1, \dots, X_n \leq x_n] = P[X_1 \leq x_1] \cdots P[X_n \leq x_n] = F(x_1) \cdots F(x_n),$$

trong đó $F(\cdot)$ là hàm phân phối của các biến ngẫu nhiên X_1, X_2, \dots

Các phần tử trong chuỗi IID noise có cùng phân phối xác suất và không có mối tương quan với nhau. Không có sự thay đổi theo thời gian hay xu hướng trong chuỗi này. Việc sử dụng chuỗi IID noise trong mô hình chuỗi thời gian cho phép giả định rằng các quan sát đều quan trọng và không có ảnh hưởng của quá khứ lên tương lai.

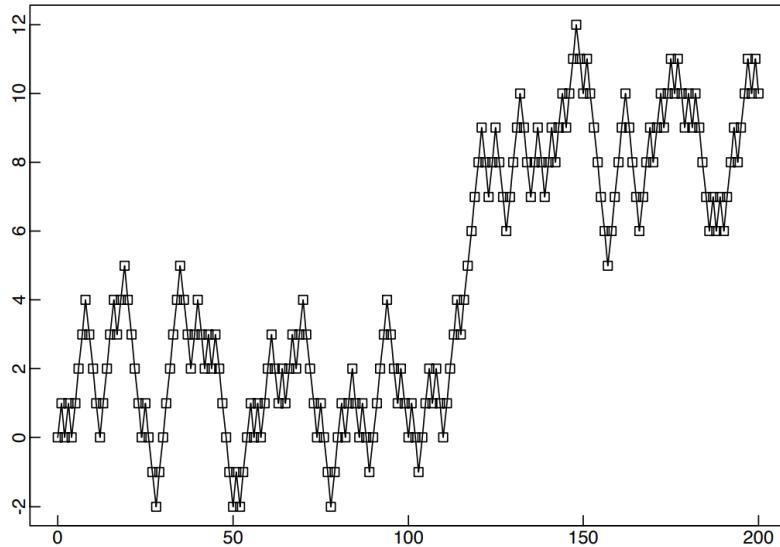
Random Walk

Quá trình random walk $\{S_t, t = 0, 1, 2, \dots\}$ (bắt đầu từ 0) được thu được bằng cách tích lũy các biến ngẫu nhiên iid. Do đó, quá trình Random Walk có trung bình bằng 0, được xác định bằng cách đặt $S_0 = 0$ và

$$S_t = X_1 + X_2 + \dots + X_t, \quad \text{với } t = 1, 2, \dots$$

trong đó $\{X_t\}$ là iid noise. Quá trình này có thể được xem như vị trí của một người đi bộ bắt đầu tại vị trí 0 tại . thời điểm 0 và tại mỗi thời điểm nguyên, tung đồng xu công bằng, bước một đơn vị sang phải mỗi khi xuất hiện mặt ngửa và một đơn vị sang trái cho mỗi mặt sấp. Kết quả của các lần tung đồng xu có thể được khôi phục từ $\{S_t, t = 0, 1, \dots\}$ bằng cách lấy hiệu. Do đó, kết quả của lần tung thứ t có thể được tìm thấy từ $S_t - S_{t-1} = X_t$.

Hình 2 thể hiện một ví dụ cụ thể của một random walk với độ dài là 200.



Hình 2: Ví dụ về một random walk đơn giản $\{S_t, t = 0, 1, 2, \dots, 200\}$.

Random walk có thể được sử dụng để dự đoán giá cổ phiếu hoặc các biến thay đổi khác trong tương lai, tuy nhiên, cần phải chú ý đến tính chất ngẫu nhiên của mô hình và rủi ro trong việc dự đoán.

White noise

Nếu $\{X_t\}$ là một chuỗi các biến ngẫu nhiên không tương quan, mỗi biến có trung bình bằng không và phương sai là σ^2 thì $\{X_t\}$ được gọi là white noise. Ta có thể biểu diễn điều kiện trên dưới dạng:

- $E[X_t] = 0$.
- $E[X_t^2] = \sigma^2 = Var[X_t]$.
- $E[X_t X_{t+h}] = 0 \quad \forall h \neq 0$.

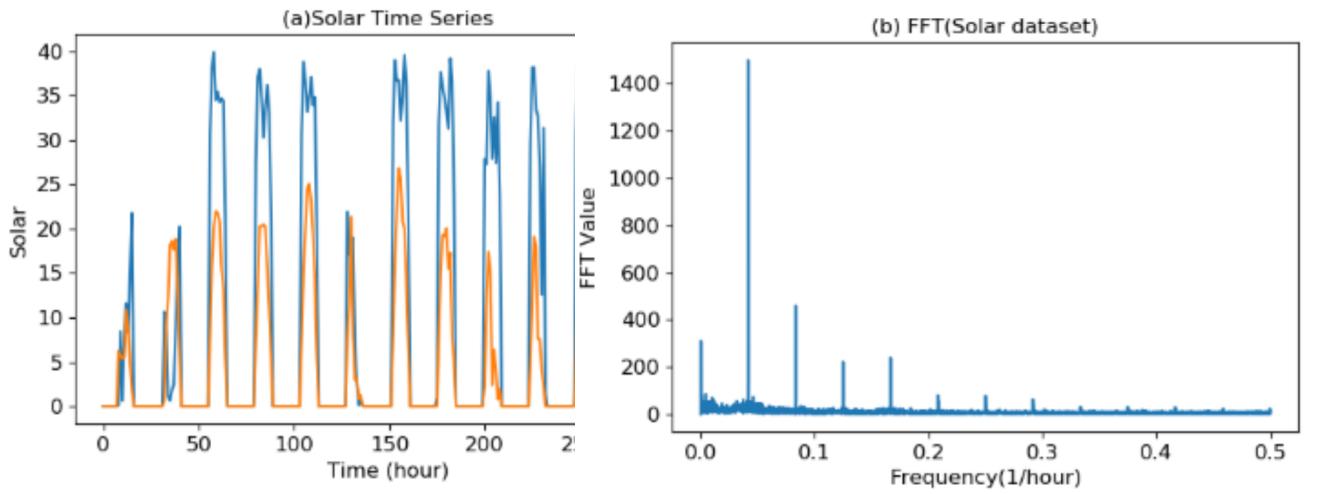
Khi đó $\{X_t\}$ được ký hiệu như sau:

$$\{X_t\} \sim WN(0, \sigma^2)$$

2.1.1 Chuỗi thời gian đa biến

Dự báo chuỗi thời gian đa biến là quá trình dự báo giá trị tương lai của một tập hợp các biến trong chuỗi thời gian. Cách giải quyết dự báo chuỗi thời gian đa biến hiệu quả là

phải tìm cách trích xuất các mẫu thời gian phức tạp và các mối quan hệ phụ thuộc lẫn nhau từ chuỗi thời gian một cách chính xác. Hình 2.1.1(a) minh họa một ví dụ về chuỗi năng lượng mặt trời được tạo ra bởi hai nhà máy điện phát điện từ năng lượng mặt trời tại Alabama, Hoa Kỳ. Chúng ta có thể dễ dàng nhận thấy rằng hai chuỗi này liên quan đến nhau. Theo kỹ thuật [2], trong Hình 2.1.1(b), chúng ta vẽ giá trị biến đổi Fourier (FFT) của chuỗi năng lượng mặt trời, cho thấy tính chu kỳ của chuỗi trong 24 giờ, 12 giờ và 365 ngày. Đặc biệt, khi chuỗi thời gian đa biến thể hiện các mẫu phụ thuộc lẫn nhau và chu kỳ thời gian không ổn định, việc dự báo chính xác sẽ rất khó khăn.



Mô hình dự báo với chuỗi thời gian đa biến

Xét chuỗi thời gian với N biến, ta biểu diễn chuỗi thời gian thứ i là $\mathbf{x}^i = (\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \dots, \mathbf{x}_{i,T})^T \in \mathbb{R}^T$, trong đó T là độ dài của chuỗi thời gian hay số lượng quan sát trong chuỗi. $\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N) \in \mathbb{R}^{T \times N}$ là tập hợp N chuỗi thời gian. Sử dụng một cửa sổ trượt thời gian (time window) có kích thước τ để chia các chuỗi thời gian thành các phần có độ dài bằng nhau. Dự báo chuỗi thời gian đa biến là xây dựng một hàm $f : \mathbb{R}^{N \times \tau} \rightarrow \mathbb{R}^N$ ánh xạ các quan sát lịch sử của tất cả các chuỗi sang giá trị tại thời điểm tiếp theo.

Dự báo chuỗi thời gian bao gồm dự báo ngắn hạn, trung hạn hoặc dài hạn tùy vào nhu cầu thực tế. Ví dụ, trong dự báo lưu lượng giao thông khoảng thời gian dự báo thường từ vài giờ đến một ngày; còn trong trường hợp dự báo hàng hóa trong các siêu thị thì khoảng thời gian dự báo có thể là vài ngày, vài tuần hoặc vài tháng. Do đó, chúng ta cần thực hiện dự báo đa bước bằng cách lặp lại dự báo một bước hoặc ước lượng trực tiếp một mô hình riêng cho mỗi dự báo. Mô hình ước lượng ước lượng đê quy thường được ký hiệu là

$$Y_{t+1} = f(Z_t) + \epsilon_{t+1}$$

trong đó $Z_t = (X_{t-\tau}, X_{t-\tau+1}, \dots, X_{t-1})$ là các giá trị đầu vào được tạo thành bởi các quan sát

được thu thập trong quá khứ, $f(\cdot)$ là mô hình dự báo, ϵ_{t+1} là lỗi tại thời điểm $t + 1$. Dự báo nhiều bước được thực hiện bởi $\hat{Y}_{t+h} = f(\hat{Z}_{t+h-1})$. Mô hình đệ quy h lần để dự báo \hat{Y}_{t+h} . Ví dụ, $\hat{Y}_{t+2} = f(\hat{Z}_{t+1})$, trong đó đầu vào $\hat{Z}_{t+1} = (X_{t-\tau+1}, \dots, \hat{X}_{t+1})$ sử dụng giá trị dự báo để thay thế các quan sát hiện tại chưa được biết đến. Ưu điểm của việc huấn luyện này đơn giản chỉ cần huấn luyện một mô hình. Tuy nhiên, sai số dự báo sẽ tăng dần khi thời gian dự báo tăng lên.

Dự báo nhiều bước trực tiếp là huấn luyện h mô hình dự báo độc lập cho h các bước dự đoán tương ứng (horizons). Việc dự báo này được biểu diễn

$$Y_{t+h} = f_h(Z_t) + \epsilon_{t+h,h}$$

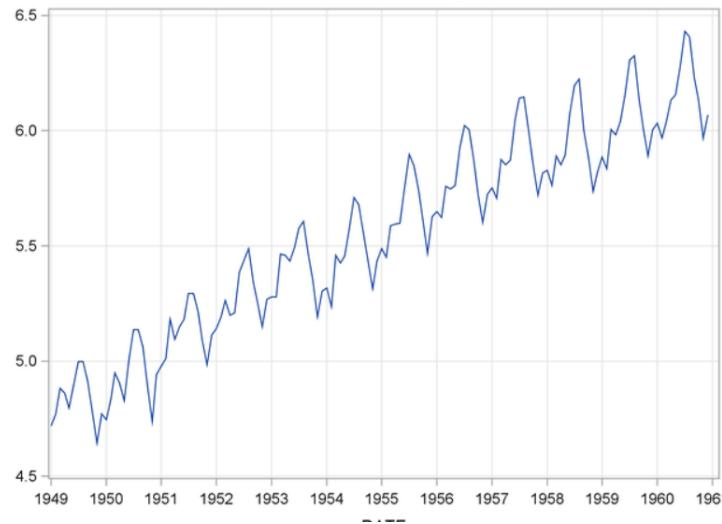
trong đó $f_h(\cdot)$ là mô hình dự báo h bước. Việc dự báo nhiều bước trực tiếp này tránh được vấn đề về tích lũy sai số nhưng đòi hỏi phải huấn luyện nhiều mô hình hơn.

$$(X_{t-\tau+1}, X_{t-\tau+2}, \dots, X_t) \xrightarrow{f(\cdot)} X_{t+1+h}$$

2.2 Tính chất cơ bản của chuỗi thời gian

Tính mùa trong chuỗi thời gian

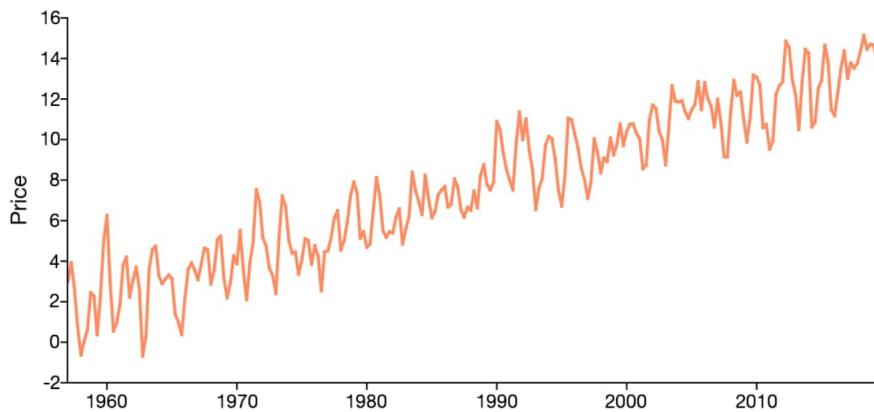
Tính mùa (seasonal) trong chuỗi thời gian là sự lặp lại chu kỳ của mẫu giá trị trong chuỗi thời gian trong một khoảng thời gian cố định. Các chu kỳ này có thể do yếu tố thời tiết, sự thay đổi trong kinh doanh hay các yếu tố khác. Ví dụ, chúng ta dự đoán doanh số kem sẽ cao hơn trong những tháng hè và thấp hơn trong những tháng mùa đông.



Hình 3: Đồ thị về tính mùa trong chuỗi thời gian doanh số bán kem của một cửa hàng

Tính xu hướng trong chuỗi thời gian

Tính xu hướng (trend) trong chuỗi thời gian là một biến đổi dài hạn của chuỗi thời gian, thể hiện xu hướng tăng hoặc giảm của dữ liệu theo thời gian. Xu hướng có thể được phân tích và mô hình hóa để đưa ra dự báo cho tương lai, và là một trong những yếu tố quan trọng trong việc phân tích chuỗi thời gian. Ví dụ, nếu chuỗi thời gian cho thấy xu hướng tăng dần, ta có thể dự đoán rằng dữ liệu sẽ tiếp tục tăng trong tương lai, và ngược lại nếu xu hướng là giảm dần. Tuy nhiên, xu hướng có thể bị ảnh hưởng bởi nhiều và sự biến động ngắn hạn của chuỗi thời gian, do đó cần phân tích kỹ lưỡng và xác định một cách chính xác.



Hình 4: Đồ thị về tính xu hướng trong chuỗi thời gian của chuỗi giá.

Quá trình dừng

Chuỗi thời gian $\{X_t\}$ được gọi là quá trình **dừng chặt** nếu với mọi h, t_i và n , ta có:

$$P(X_{t_1} \leq x_{t_1}, X_{t_2} \leq x_{t_2}, \dots, X_{t_n} \leq x_{t_n}) = P(X_{t_1+h} \leq x_{t_1}, X_{t_2+h} \leq x_{t_2}, \dots, X_{t_n+h} \leq x_{t_n})$$

Chuỗi thời gian $\{X_t\}$ được gọi là **dừng yếu** nếu:

- (i) $E[X_t] = \mu_x(t)$ độc lập với t .
- (ii) Tương quan của của $\{X_t\}$

$$\gamma_X(t+h, t) = E(X_t X_{t+h}) \text{ độc lập với } t \text{ với } \forall h$$

Nhận xét:

- (i) Chuỗi thời gian $\{X_t\}$ là dừng chặt thì sẽ là dừng yếu.
- (ii) Khi nói tới quá trình dừng thì hiểu là dừng yếu.

Các hàm giá trị đặc trưng

Xét chuỗi thời gian $\{X_t\}$ với $E(X_t^2) < \infty$. Khi đó hàm trung bình của $\{X_t\}$ là:

$$\mu_X(t) = E(X_t)$$

Và hàm hiệp phương sai của $\{X_t\}$ là:

$$\gamma_X(r, s) = Cov(X_r, X_s) = E[(X_r - \mu_X(r))(X_s - \mu_X(s))]$$

với mọi số nguyên r và s .

Chuỗi thời gian $\{X_t\}$ là chuỗi dừng. **Hàm hiệp phương sai** (ACVF) của $\{X_t\}$ với độ trễ (lag) h là:

$$\gamma_x(h) = Cov(X_{t+h}, X_t)$$

Hàm tự tương quan(ACF) của $\{X_t\}$ với độ trễ h là:

$$\rho_x(h) = \frac{\gamma_x(h)}{\gamma_x(0)} = Cor(X_{t+h}, X_t)$$

2.3 Các mô hình dự báo chuỗi thời gian đơn biến cơ bản

Trong phần tiếp theo, ta sẽ cùng tìm hiểu một số mô hình chuỗi thời gian cơ bản, các mô hình phổ biến được giới thiệu trong phần này bao gồm: AR, MA, ARMA, ARIMA và SARIMA. Đối với mỗi mô hình, ta sẽ xem xét dưới góc độ xem xét công thức chung mà mô hình tuân theo, tính dừng của chuỗi, đánh giá hàm ACF, ACVF, PACF. Đồng thời đánh giá hàm dự báo mà mỗi mô hình sử dụng. Nội dung lý thuyết sẽ được tham khảo từ sách [3].

2.3.1 Mô hình tự hồi quy AR

Quá trình tự hồi quy (autoregression, AR) với bậc q được ký hiệu là AR(q) là quá trình dừng thỏa mãn công thức như sau:

$$X_t - \phi_1 X_{t-1} - \cdots - \phi_p X_{t-p} = Z_t$$

với $\{Z_t\} \sim WN(0, \sigma^2)$.

Hay ký hiệu đơn giản hơn, $\phi(B)X_t = W_t$ với đa thức $\phi(B) = 1 - \phi_1 B - \cdots - \phi_p B^p$.

Ta xét trường hợp $p = 1$ (AR(1)), lúc này $\phi(B) = 1 - \phi_1 B$ được xem là mô hình AR(1) nếu tồn tại nghiệm dừng thỏa mãn $\phi(B)X_t = W_t$, tương đương với nghiệm $\phi_1 \neq 1$ không nằm trên đường tròn đơn vị. Điều này tương đương với $\forall z \in \mathbb{R}, \phi(Z) = 0 \Rightarrow z \neq 1$

Một quá trình AR(p) được gọi là có tính nhân quả nếu:

$$\phi(z) = 1 - \phi_1 z - \cdots - \phi_p z^p \Rightarrow |z| \neq 1$$

hay nói cách khác, mọi nghiệm của đa thức $\phi(z)$ đều nằm ngoài đường tròn đơn vị.

Giả sử X_t là chuỗi dừng và $|\phi| < 1$. Từ đó, ta có:

$$E[X_t] = 0, \quad E[X_t^2] = \frac{\sigma^2}{1 - \phi^2}$$

Giá trị hàm ACVF tại độ trễ h là

$$\begin{aligned} \gamma_X(h) &= \text{Cov}(\phi X_{t+h-1} + W_{t+h}, X_t) \\ &= \phi \text{Cov}(X_{t+h-1}, X_t) \\ &= \phi \gamma_X(h-1) \\ &= \phi^{|h|} \gamma_X(0) \quad (\text{kiểm tra } h > 0 \text{ và } h < 0) \\ &= \frac{\phi^{|h|} \sigma^2}{1 - \phi^2}. \end{aligned}$$

Giá trị hàm ACF là:

$$\rho(h) = \frac{\gamma(h)}{\gamma(0)} = \phi^{|h|}$$

Hàm Partial AutoCorrelation Function (PACF) của chuỗi dừng $\{X_t\}$ là:

$$\begin{aligned} \phi_{11} &= \text{Corr}(X_1, X_0) = \rho(1) \\ \phi_{hh} &= \text{Corr}(X_h - X_h^{h-1}, X_0 - X_0^{h-1}) \quad \text{for } h = 2, 3, \dots \end{aligned}$$

Áp dụng để tính PACF của chuỗi AR(p), ta có: $X_t = \sum_{i=1}^p \phi_i X_{t-i} + W_t$,

$$X_{n+1}^n = \sum_{i=1}^p \phi_i X_{n+1-i}.$$

$$\text{Do đó, } \phi_{hh} = \begin{cases} \phi_h & \text{nếu } 1 \leq h \leq p \\ 0 & \text{ngược lại.} \end{cases}$$

2.3.2 Mô hình trung bình trượt MA

Quá trình trung bình trượt (moving-average process, MA) với bậc q hay còn được ký hiệu là $MA(q)$ là quá trình có dạng sau:

$$X_t = Z_t + \theta_1 Z_{t-1} + \cdots + \theta_q Z_{t-q}$$

với $\{Z_t\} \sim WN(0, \sigma^2)$ và các hệ số $\theta_1, \dots, \theta_q$.

Giá trị của hàm ACVF đối với quá trình trung bình trượt:

$$\gamma(h) = \begin{cases} \sigma^2 \sum_{j=0}^{q-|h|} \theta_j \theta_{j+|h|}, & \text{nếu } |h| \leq q, \\ 0, & \text{nếu } |h| > q, \end{cases}$$

với $\theta_0 = 1$. Hàm ACVF của $MA(q)$ do đó có giá trị bằng 0 khi độ trễ có giá trị lớn hơn q .

Dễ dàng suy ra được giá trị của hàm ACF của chuỗi $MA(q)$

$$\rho(q) = \frac{\gamma(h)}{\gamma(0)}$$

Một chuỗi thời gian $MA(q)$ được gọi là có tính khả nghịch nếu mọi nghiệm của đa thức θ đều nằm ngoài đường tròn đơn vị

$$\theta(z) = 1 + \theta_1 z + \cdots + \theta_q z^q = 0 \Rightarrow |z| > 1$$

Tiếp theo ta cùng xem xét hàm PACF của một chuỗi $MA(q)$ khả nghịch. Từ chuỗi $MA(q)$ khả nghịch, với $X_t = \sum_{i=1}^q \theta_i W_{t-i} + W_t$, $X_t = -\sum_{i=1}^q \pi_i X_{t-i} + W_t$, ta có:

$$\begin{aligned} X_{n+1}^n &= P(X_{n+1} \mid X_1, \dots, X_n) \\ &= P\left(-\sum_{i=1}^q \pi_i X_{n+1-i} + W_{n+1} \mid X_1, \dots, X_n\right) \\ &= -\sum_{i=1}^q \pi_i P(X_{n+1-i} \mid X_1, \dots, X_n) \\ &= -\sum_{i=1}^q \pi_i X_{n+1-i} - \sum_{i=n+1}^{\infty} \pi_i P(X_{n+1-i} \mid X_1, \dots, X_n). \end{aligned}$$

Hàm PACF của quá trình $MA(1)$ tại độ trễ h là:

$$\alpha(h) = \phi_{hh} = -(-\theta)^h / (1 + \theta^2 + \cdots + \theta^{2h})$$

2.3.3 Mô hình ARMA

Mô hình ARMA là mô hình kết hợp cả hai quá trình AR và MA. Cụ thể, mô hình ARMA(p, q) là quá trình dừng có công thức định nghĩa như sau:

$$\phi(B)X_t = \theta(B)Z_t \quad (1)$$

với $\phi(z) = 1 - \phi_1z - \cdots - \phi_pz^p$ và $\theta(z) = 1 + \theta_1z + \cdots + \theta_qz^q$

Tương tự như quá trình AR và MA, quá trình ARMA cũng có tính nhân quả và khả nghịch, định nghĩa của tính nhân quả tương tự như đối với hai chuỗi ban đầu. Chuỗi được gọi là có tính nhân quả nếu $\phi(z)$ là đa thức có mọi nghiệm nằm ngoài đường tròn đơn vị, và được gọi là có tính khả nghịch nếu như $\theta(z)$ có nghiệm nằm ngoài đường tròn đơn vị.

$$X_t = \sum_{j=0}^{\infty} \psi_j Z_{t-j}$$

với $\sum_{j=0}^{\infty} \psi_j z^j = \theta(z)/\phi(z), |z| \leq 1$.

Trong phần này, tác giả sẽ chỉ ra các hàm ACF, PACF của các quá trình ARMA đơn giản. Lấy ví dụ từ chuỗi ARMA(1,1). Cụ thể công thức biểu diễn ARMA(1,1) được chỉ ra ở phương trình 2

$$X_t - \phi X_{t-1} = Z_t + \theta Z_{t-1}, \quad \{Z_t\} \sim WN(0, \sigma^2) \quad (2)$$

Với giá trị $|\phi| < 1$ hay chuỗi có tính nhân quả, ta suy ra được:

$$\begin{aligned} \gamma(0) &= \sigma^2 \sum_{j=0}^{\infty} \psi_j^2 \\ &= \sigma^2 \left[1 + (\theta + \phi)^2 \sum_{j=0}^{\infty} \phi^{2j} \right] \\ &= \sigma^2 \left[1 + \frac{(\theta + \phi)^2}{1 - \phi^2} \right], \\ \gamma(1) &= \sigma^2 \sum_{j=0}^{\infty} \psi_{j+1} \psi_j \\ &= \sigma^2 \left[\theta + \phi + (\theta + \phi)^2 \phi \sum_{j=0}^{\infty} \phi^{2j} \right] \\ &= \sigma^2 \left[\theta + \phi + \frac{(\theta + \phi)^2 \phi}{1 - \phi^2} \right], \end{aligned}$$

từ đó dựa vào truy hồi ta tính ra được giá trị $\gamma(h)$

$$\gamma(h) = \phi^{h-1} \gamma(1), \quad h \geq 2$$

$$\text{Giá trị hàm ACF tại độ trễ } h \rho(h) = \frac{\gamma(h)}{\gamma(0)} = \phi^{h-1} \frac{\gamma(1)}{\gamma(0)}$$

2.3.4 Mô hình ARIMA

Nếu d là số nguyên không âm, $\{X_t\}$ được gọi là quá trình $ARIMA(p, s, q)$ nếu $Y_t := (1 - B)^d X_t$ là quá trình ARMA(p, q) nhân quả. Ta có thể viết $\phi(B)(1 - B)^d X_t = \theta(B)W_t$. Điều này đồng nghĩa với việc $\{X_t\}$ thỏa mãn phương trình sai phân

$$\phi^*(B)X_t \equiv \phi(B)(1 - B)^d X_t = \theta(B)Z_t, \quad \{Z_t\} \sim WN(0, \sigma^2)$$

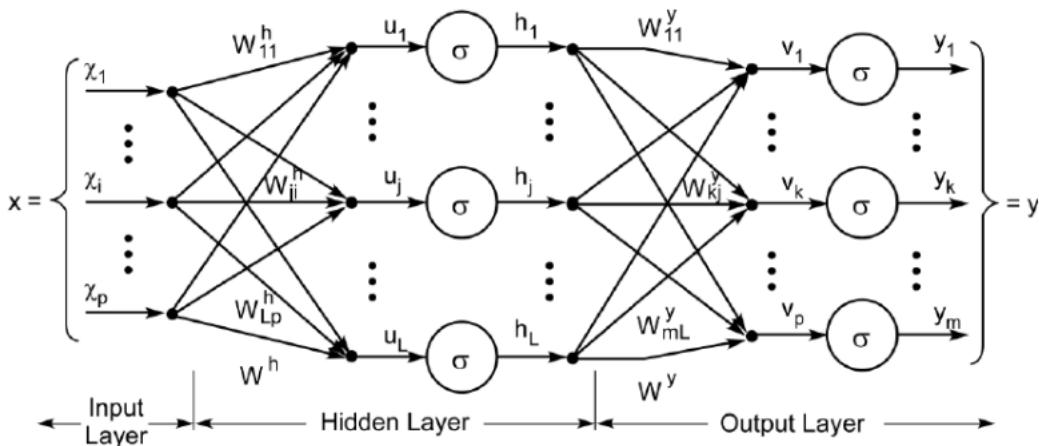
với $\phi(z)$ và $\theta(z)$ là đa thức với bậc là p và q , và các nghiệm của hai đa thức trên đều nằm ngoài đường tròn đơn vị. Đa thức $\phi^*(z)$ có nghiệm bậc d tại $z = 1$. Quá trình $\{X_t\}$ là dừng khi và chỉ khi $d = 0$, trong trường hợp này có thể chuyển mô hình về quá trình ARMA(p, q).

3 Giới thiệu về một số mạng trong Học sâu

Trong phần này, tác giả sẽ trình bày các kiến thức cơ bản về mạng Neural và kiến trúc của mạng Recurrent Neural. Các kiến thức trong phần này được tham khảo chính trong sách [4]. Các ký hiệu in đậm như \mathbf{v} biểu thị cho các vector.

3.1 Tổng quan mạng Neural

Mô hình mạng Neural được xây dựng dựa trên mô hình tổng quát Multilayer Perceptron-MLP (theo [4]). Một MLP là ánh xạ $y = f * (x, \theta)$ với input là dữ liệu và output là giá trị dự đoán. Hàm f được xây dựng từ các hàm tuyến tính, hàm tanh, Bằng việc lựa chọn các hàm thích hợp và dữ liệu đầu vào mà mô hình MLP có thể xấp xỉ được tham số θ và từ đó giúp máy tính xác định đầu ra.



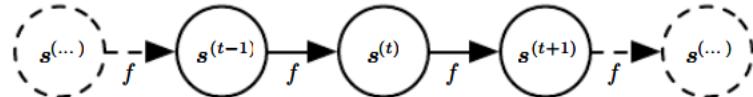
Hình 5: Kiến trúc cơ bản của mạng MLP

Kiến trúc của mạng Neural được xây dựng tương tự như ở Hình 5 với 3 thành phần chính là:

1. Input Layer: Lớp input dữ liệu.
2. Hidden Layer: Lớp tính toán, lưu trữ thông tin trung gian gồm nhiều các *node* mạng với các *activation function* f khác nhau.
3. Output Layer: Lớp đầu ra dự đoán của mô hình.

Mỗi *layer* là một tầng trong mạng Neural, một *Hidden Layer* và *Output Layer* có thể gồm nhiều *node* (tương ứng các σ trong Hình 5) là thành phần lưu trữ thông tin trong mạng, mỗi *node* sẽ liên kết với toàn bộ các *node* của *layer* trước đó với các hệ số w riêng và ở mỗi *node* sẽ diễn ra 2 bước tính toán: tính tổng tuyến tính với các hệ số w và sau đó sử dụng *activation function* riêng để tính toán ra đầu ra dùng cho *layer* tiếp theo.

Việc mô hình hóa tính toán của mạng Neural được biểu diễn thông qua *computational graph* như sau:



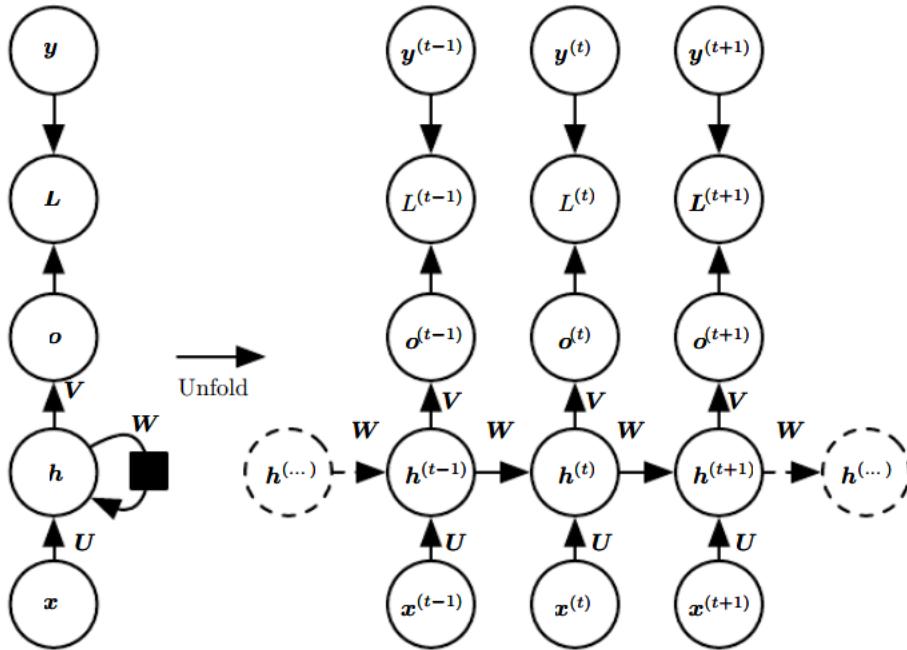
Hình 6: Computational Graph

Mỗi một *node* có một *activation function* f riêng (ví dụ như hàm tuyến tính, hàm *relu*, hàm *tanh*, hàm *softmax*,...) và được minh họa trong *computational graph* như trên. Ví dụ trong hình 6, ta có biểu diễn cách tính giá trị *state* $s^{(t)}$ của *layer* sau được tính thông qua *layer* trước:

$$s^{(t)} = f(s^{(t-1)}, x^{(t)}; \theta)$$

3.2 Sequence Modeling: Mạng LSTM và GRU

Mạng *Long-short term memory LSTM* và mạng *Gated Recurrent Unit GRU* đều có kiến trúc chung là mạng *Recurrent Neural RNN*, chúng được sử dụng rộng rãi trong các mô hình *Natural Language Processing NLP*, cụ thể là dịch và xử lý các đoạn văn bản. Kiến trúc mạng RNN tổng quát có thể được biểu diễn như sau:



Hình 7: Computational graph biểu diễn RNN

Việc tính toán *forward* trong mạng RNN được tổng quát thông qua hàm sau:

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \boldsymbol{\theta})$$

Như trong hình 7 biểu diễn, mạng RNN sẽ gồm các ánh xạ nối chuỗi input \mathbf{x} đến các giá trị output tương ứng của từng *node* là \mathbf{o} . Một hàm mất mát *loss* L sẽ tính toán độ sai khác giữa mỗi \mathbf{o} với giá trị *training* \mathbf{y} thông qua đại lượng dự đoán của mạng là $\hat{\mathbf{y}} = \text{softmax}(\mathbf{o})$. Mạng RNN gồm các ma trận trọng số sau: ma trận trọng số \mathbf{U} tính toán từ input \mathbf{x} tạo ra các giá trị ẩn *hidden* \mathbf{h} của mạng, các giá trị \mathbf{h} sẽ được dùng làm input cho bước lặp tiếp theo trong *forward* mạng RNN và các giá trị \mathbf{h} đó được kết nối với nhau thông qua ma trận trọng số \mathbf{W} , việc tính toán từ \mathbf{h} tới các giá trị output \mathbf{o} của mạng thông qua ma trận trọng số \mathbf{V} . Cụ thể, quá trình tính toán trong mạng RNN sẽ tiến hành như sau:

$$\begin{aligned}\mathbf{a}^{(t)} &= \mathbf{b} + \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} \\ \mathbf{h}^{(t)} &= \tanh(\mathbf{a}^{(t)}) \\ \mathbf{o}^{(t)} &= \mathbf{c} + \mathbf{V}\mathbf{h}^{(t)} \\ \hat{\mathbf{y}}^{(t)} &= \text{softmax}(\mathbf{o}^{(t)})\end{aligned}$$

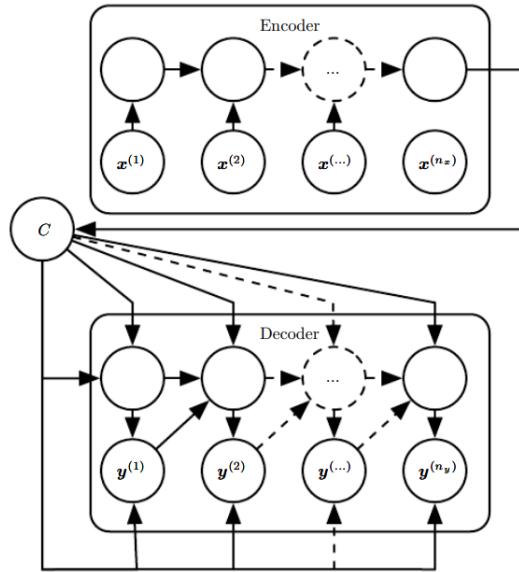
Các tham số *bias* của mạng được biểu diễn thông qua vectors \mathbf{b} và \mathbf{c} . Mô hình trên được sử dụng trong trường hợp chuỗi input \mathbf{x} và chuỗi output \mathbf{y} có cùng độ dài. Tổng giá trị hàm

mất mát khi biết trước chuỗi input \mathbf{x} và chuỗi output \mathbf{y} là tổng các hàm mất mát của tất cả các bước lặp. Ví dụ $L^{(t)}$ là hàm mất mát *negative log-likelihood* của $y^{(t)}$ cho trước chuỗi $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}$ thì:

$$\begin{aligned} & L\left(\left\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\tau)}\right\}, \left\{\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(\tau)}\right\}\right) \\ &= \sum_t L^{(t)} \\ &= - \sum_t \log p_{\text{model}}\left(y^{(t)} \mid \left\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}\right\}\right) \end{aligned}$$

với $p_{\text{model}}\left(y^{(t)} \mid \left\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}\right\}\right)$ được tính thông qua mô hình khi so sánh giá trị của $y^{(t)}$ với giá trị dự đoán của mô hình là $\hat{y}^{(t)}$.

Kiến trúc Encoder-Decoder Sequence-to-Sequence

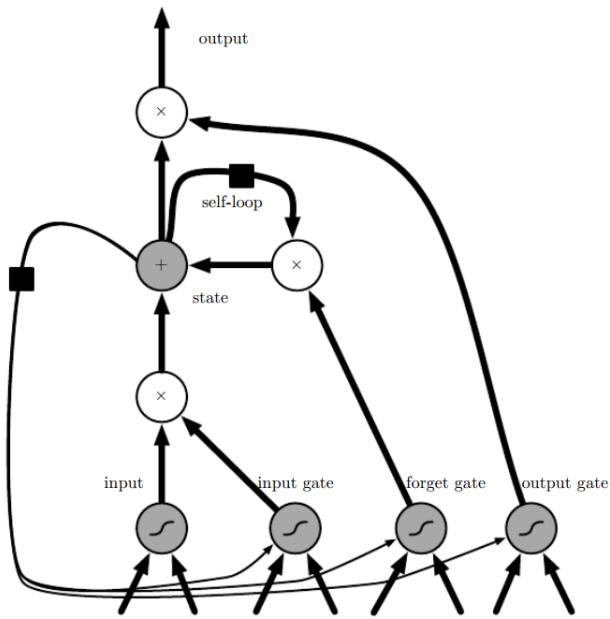


Hình 8: Kiến trúc Encoder-Decoder

Hình 8 miêu tả kiến trúc ²sequence-to-sequence của mạng RNN trong việc học tạo ra chuỗi output $(\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(n_y)})$ khi biết trước chuỗi input $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(n_x)})$. Nó là sự kết hợp của một khối Encoder RNN đọc chuỗi dữ liệu đầu vào và một khối Decoder RNN tạo ra chuỗi output đầu ra (hoặc tính xác suất xảy ra một sự kiện nào đó (bài toán phân loại) khi cho trước chuỗi đầu vào). Lớp ẩn cuối cùng của Encoder RNN được dùng để tính một biến "ngữ cảnh" (context) cố định C biểu diễn thông tin được trích xuất ra từ chuỗi đầu vào để là đầu vào cho khối Decoder RNN.

Mạng LSTM

Về mặt lý thuyết mạng RNN có thể mang thông tin từ các *layer* trước đến các *layer* sau, nhưng thực tế là thông tin chỉ mang được qua một số lượng *state* nhất định, sau đó thì sẽ xảy ra hiện tượng *vanishing gradient* (tức đạo hàm hàm *loss* theo tham số của mạng xấp xỉ bằng 0) hay nói cách khác là mạng RNN chỉ học được từ các *state* gần nó, đây là hiện tượng *short term memory*. Với bài toán dịch văn bản trong xử lý ngôn ngữ tự nhiên thì input của mạng là những câu văn dài và ta cần thông tin của những câu văn trước đó để output đầu ra dịch đúng ngữ cảnh nhất. Chính vì vậy, để khắc phục hiện tượng *vanishing gradient* một mạng RNN cải tiến ra đời chính là mạng *Long-short term memory* LSTM.



Hình 9: Kiến trúc tế bào của mạng LSTM

Điểm đặc biệt trong cấu trúc tế bào của mạng LSTM là nó có cơ chế *self-loop* để làm giảm hiện tượng *vanishing gradient*, tế bào LSTM sẽ quyết định tỷ lệ lưu giữ thông tin của các *state* trước đó, các tham số của *self-loop* được điều khiển thông qua cổng "quên" (forget gate) $f_i^{(t)}$ (ứng với bước lặp thứ t và tế bào thứ i), cổng này sẽ điều chỉnh tỷ lệ quên thông tin trước trong khoảng từ 0 tới 1 thông qua hàm *sigmoid*:

$$f_i^{(t)} = \sigma \left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)} \right)$$

trong đó $\mathbf{x}^{(t)}$ là vector input hiện tại và $\mathbf{h}^{(t)}$ là vector *hidden layer* hiện tại chứa output của toàn bộ tế bào LSTM trước đó. $b^f, \mathbf{U}^f, \mathbf{W}^f$ lần lượt là tham số *bias* của mạng, trọng số của

dữ liệu đầu vào và trọng số cho tầng cổng quên. Thành phần quan trọng nhất của tế bào là *state unit* $s_i^{(t)}$ lưu trữ thông tin của mạng. Khi đó quá trình cập nhật giá trị trong mạng diễn ra như sau:

$$s_i^{(t)} = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} \sigma \left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)} \right)$$

với \mathbf{b} , \mathbf{U} và \mathbf{W} lần lượt ký hiệu cho hệ số *bias*, tham số của dữ liệu input và tham số *recurrent* của tế bào LSTM. Ta có thể thấy giá trị của s_i^t được cập nhật kết hợp với giá trị của tầng cổng quên $f_i^{(t)}$. Giá trị *external input gate* $g_i^{(t)}$ được tính toán tương tự như tầng cổng quên với các tham số riêng để tính toán tỷ lệ ảnh hưởng của dữ liệu đầu vào ở bước thứ t là $x_j^{(t)}$ và thông tin từ *state* trước đó là $h_j^{(t-1)}$:

$$g_i^{(t)} = \sigma \left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)} \right).$$

Giá trị output dự đoán $h_i^{(t)}$ của tế bào LSTM có thể bị bỏ qua thông qua cổng *output* $q_i^{(t)}$ cũng được tính thông qua hàm *sigmoid*:

$$\begin{aligned} h_i^{(t)} &= \tanh \left(s_i^{(t)} \right) q_i^{(t)} \\ q_i^{(t)} &= \sigma \left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)} \right) \end{aligned}$$

có chứa các vector tham số \mathbf{b}^o , \mathbf{U}^o , \mathbf{W}^o lần lượt ứng với *bias*, trọng số với dữ liệu input, trọng số của tầng cổng quên đối với kết quả output dự đoán của các *state* trước đó.

Mạng GRU

Mạng GRU *Gated Recurrent Unit* là một loại mạng RNN cũng được thiết kế để khắc phục hiện tượng *vanishing gradient*. Mạng GRU có kiến trúc tương tự mạng LSTM nhưng có sự thay đổi trong cách thiết kế tế bào *cell*, cụ thể là thay đổi cách thiết lập hàm để "quên" thông tin từ các *state* trước đó cũng như cách tính trọng số trong việc sử dụng thông tin từ các *state* trước đó:

$$h_i^{(t)} = u_i^{(t-1)} h_i^{(t-1)} + \left(1 - u_i^{(t-1)} \right) \sigma \left(b_i + \sum_j U_{i,j} x_j^{(t-1)} + \sum_j W_{i,j} r_j^{(t-1)} h_j^{(t-1)} \right)$$

với u biểu thị cho *cổng cập nhật* (*update gate*) và r biểu thị cho *cổng reset*. Các cổng này được thiết lập tính toán như sau:

$$u_i^{(t)} = \sigma \left(b_i^u + \sum_j U_{i,j}^u x_j^{(t)} + \sum_j W_{i,j}^u h_j^{(t)} \right)$$

và

$$r_i^{(t)} = \sigma \left(b_i^r + \sum_j U_{i,j}^r x_j^{(t)} + \sum_j W_{i,j}^r h_j^{(t)} \right)$$

Cổng *reset* cũng như cổng *cập nhật* có thể bỏ qua thông tin của các *state* trước đó, cổng *cập nhật* có cơ chế tương tự như tầng cổng quên, cổng *reset* sẽ điều khiển phần thông tin nào của *state* trước đó được dùng để sử dụng trong việc tính toán của *state* này.

Mạng Recurrent-skip

Các lớp Recurrent với GRU [5] và LSTM [6] được thiết kế để ghi thông tin lịch sử nên nó có thể nhận biết được sự phụ thuộc dài. Tuy nhiên do không dùng tối gradient, GRU và LSTM thường không thể nắm được các sự phụ thuộc có thời gian rất dài trong thực tế. Vì thế, Guokun Lai cùng các cộng sự [7] đã đề xuất giảm thiểu vấn đề này thông qua một thành phần lặp mới, tận dụng mô hình chu kỳ trong các tập dữ liệu thực tế. Ví dụ, mức tiêu thụ điện và mức độ giao thông đều có mẫu rõ ràng theo ngày. Nếu chúng ta muốn dự báo mức tiêu thụ điện vào thời điểm t trong ngày, thì có thể dùng mô hình dự báo theo mùa là tận dụng các bản ghi vào thời điểm t giờ hàng ngày, bên cạnh các bản ghi gần đây nhất. Loại phụ thuộc này khó có thể nắm bắt các đơn vị tuần hoàn sẵn do chu kỳ dài (24 giờ) và các vấn đề tối ưu tương ứng. Lấy cảm hứng từ tính hiệu quả của thủ thuật này Guokun Lai cùng các cộng sự [7] đã phát triển một cấu trúc tuần hoàn với các skip connect để mở rộng phạm vi thời gian của luồng thông tin và giúp quá trình tối ưu trở nên dễ dàng hơn. Cụ thể, các liên kết bỏ qua được vào giữa ô ẩn (hidden cell) hiện tại và các ô ẩn trong cùng giai đoạn ở các chu kỳ kế tiếp. Quá trình cập nhật có thể được xây dựng dưới dạng:

$$\begin{aligned} r_t &= \sigma(x_t W_{xr} + h_{t-p} W_{hr} + b_r) \\ u_t &= \sigma(x_t W_{xu} + h_{t-p} W_{hu} + b_u) \\ c_t &= \text{RELU}(x_t W_{xc} + r_t \odot (h_{t-p} W_{hc}) + b_c) \\ h_t &= (1 - u_t) \odot h_{t-p} + u_t \odot c_t \end{aligned}$$

trong đó đầu vòng của mỗi lớp là kết quả đầu ra của lớp tích chập và p là lượng ô ẩn (hidden cell) bị bỏ qua. Giá trị p dễ dàng xác định cho các bộ dữ liệu có chu kỳ rõ ràng và cần phải

điều chỉnh đều không có. Trong các thử nghiệm, nhóm tác giả [7] đã tìm thấy rằng p được điều chỉnh tốt để có thể cải thiện đáng kể hiệu suất của mô hình. Hơn nữa, LSTNet có thể dễ dàng mở rộng để chứa các độ dài skip p .

Guokun Lai cùng các cộng sự [7] đã sử dụng lớp dense để kết hợp các đầu ra của thành phần Recurrent và Recurrent-skip. Đầu vào của lớp dense bao gồm các trạng thái ẩn (hidden state) của thành phần Recurrent tại mốc thời gian t , được ký hiệu là h_t^R và p trạng thái ẩn của thành phần Recurrent-skip từ mốc thời gian thứ $t-p+1$ đến t được ký hiệu là $h_{t-p+1}^S, h_{t-p+2}^S, \dots, h_t^S$. Đầu ra của lớp dense được tính là

$$h_t^D = W^R h_t^R + \sum_{i=0}^{p-1} W_i^S h_{t-i}^S + b$$

trong đó h_t^D là kết quả của mạng nơ ron tại mốc thời gian t .

Temporal Attention

Lớp Recurrent-skip yêu cầu một siêu tham số p được xác định trước điều này không thuận lợi cho việc dự báo chuỗi thời gian không có tính mùa hoặc chu kỳ thay đổi theo thời gian. Để giảm thiểu vấn đề này, Guokun Lai cùng các cộng sự [7] đã đề xuất một phương án thay thế khác, cơ chế attention [8], để học cách kết hợp trọng số của ẩn tại mỗi vị trí cửa sổ ma trận đầu vào. Cụ thể, trọng số attention $\alpha_t \in \mathbb{R}^q$ tại thời điểm t hiện tại được tính như sau:

$$\alpha_t = \text{AttnScore}(H_t^R, h_{t-1}^R)$$

trong đó $H_t^R = [h_{t-q}^R, \dots, h_{t-1}^R]$ là ma trận được tạo thành bằng cách xếp chồng các biểu diễn ẩn của RNN theo cột và AttnScore là một số hàm tương đồng như tích vô hướng, cosin hoặc được tham số hóa bằng một mạng perceptron nhiều tầng đơn giản.

Kết quả đầu ra cuối cùng của lớp attention thời gian là sự nối chuỗi của vector trọng số context $\mathbf{c}_t = H_t \mathbf{f}_t$ và đại diện cửa sổ ẩn (window hidden representation) cuối cùng h_{t-1}^R , bên cạnh đó phép chiếu tuyến tính được biểu diễn dưới dạng

$$h_t^D = W[\mathbf{c}_t; h_{t-1}^R] + b$$

3.3 Mang Convolution

Tích chập

Tích chập rời rạc của hai tín hiệu (signals) một chiều f và g , viết là $f * g$ được định nghĩa

$$(f * g)(i) = \sum_{j=-\infty}^{\infty} f(j)g(i-j)$$

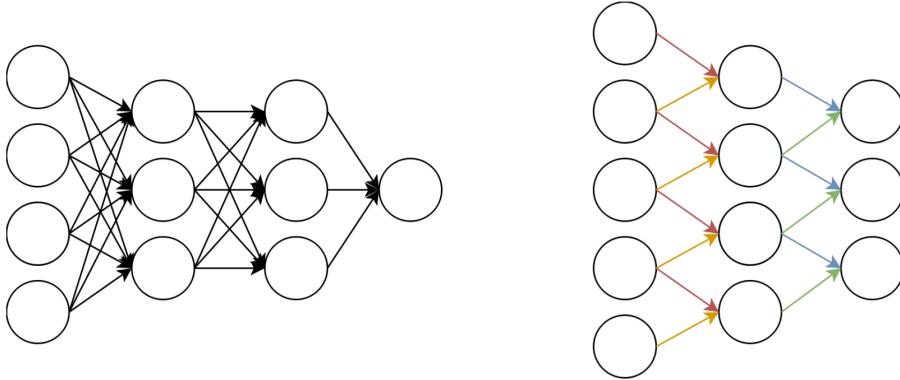
Giả sử $f = [f(0), \dots, f(N-1)]$ và $g = [g(0), \dots, g(M-1)]$, thì tích chập của của f và g là

$$(f * g)(i) = \sum_{j=0}^{M-1} f(j)g(i-j)$$

Cách xử lý các mẫu không xác định (undefined samples) ảnh hưởng đến kích thước đầu ra của phép tích chập. Nếu số lượng mẫu không xác định được đặt bằng 0, thì điều này được gọi là zero padding. Nếu không áp dụng zero padding, đầu ra của phép tích chập sẽ có kích thước $N - M + 1$ (với $i = 0, \dots, N - M$). Nếu padding với p số 0 ở hai bên của tín hiệu đầu vào f thì đầu ra sẽ có kích thước là $N - M + 2p + 1$. Việc zero padding cho phép điều khiển kích đầu đầu ra của phép tích chập, có thể giảm, giữ nguyên hoặc tăng so với kích thước đầu vào. Phép tính tích chập tại điểm i được tính bằng cách di chuyển tín hiệu g qua tín hiệu đầu vào f theo trục j và tính tổng có trọng số của hai tín hiệu này.

Mạng nơ ron tích chập

Mạng nơ-ron tích chập được phát triển với ý tưởng về kết nối cục bộ. Mỗi nút chỉ kết nối với một vùng cục bộ trên đầu vào, xem hình 10. Phạm vi không gian của kết nối được gọi là vùng tiếp nhận nút. Kết nối cục bộ được thực hiện bằng cách thay thế tổng có số từ mạng nơ ron với phép tích chập. Trong mỗi tầng của mạng nơ ron tích chập, đầu vào được tích chập với ma trận trọng số (còn gọi là bộ lọc) để tạo ra một feature map. Khác với mạng nơ ron thông thường, tất cả các giá trị của đầu ra feature map đều có cùng trọng số. Điều này có nghĩa là tất cả các đầu ra phát hiện chính xác cùng một mẫu. Khả năng kết nối cục bộ và trọng số chia sẻ (shared weights) của CNN giảm tổng số lượng tham số có thể học được, dẫn đến việc huấn luyện hiệu quả hơn. Bản chất của mạng nơ ron tích chập là học trong mỗi lớp một ma trận trọng số sẽ có thể trích xuất các đặc trưng cần thiết không đổi theo dịch chuyển từ đầu vào.



Hình 10: So sánh mạng Neural và mạng Convolution

Mạng Neural cơ bản với 3 lớp (Trái) vs. mạng nơ ron tích chập với 2 lớp và kích thước bộ lọc 1×2 , sao cho vùng tiếp nhận của mỗi nút bao gồm hai nơ ron đầu vào từ lớp trước và các trọng số được chia sẻ trên các lớp được biểu thị bằng các màu giống hệt nhau (Phải).

Đầu vào của một lớp tích chập thường có 3 chiều: the height, weight và số lượng channels. Trong lớp đầu tiên, đầu vào sẽ được tích chập với tập M_1 bộ lọc 3 chiều áp dụng trên tất cả các channels đầu vào để tạo các map features đầu ra. Giả sử có đầu vào 1 chiều $x = (x_t)_{t=0}^{N-1}$ có kích thước N không Đầu ra feature map từ lớp đầu tiên được tạo ra bằng cách tích chập mỗi bộ lọc (filter) w_h^1 với $h = 1, \dots, M_1$ với đầu vào

$$a^1(i, h) = (w_h^1 * x)(i) = \sum_{j=-\infty}^{\infty} w_h^1(j)x(i-j)$$

trong đó $w_h^1 \in \mathbb{R}^{1 \times k \times 1}$ và $a^1 \in \mathbb{R}^{1 \times N-k+1 \times M_1}$.

Trong mỗi lớp tiếp theo $l = 2, \dots, L$ đầu vào feature map $f^{l-1} \in \mathbb{R}^{1 \times N_{l-1} \times M_{l-1}}$, trong đó $1 \times N_{l-1} \times M_{l-1}$ là kích thước của feature map của lớp trước với $N_{l-1} = N_{l-2} - k + 1$, là tích chập với M_l bộ lọc $w_h^l \in \mathbb{R}^{1 \times k \times M_{l-1}}$, $h = 1, \dots, M_l$ để tạo feature map $a^l \in \mathbb{R}^{1 \times N_l \times M_l}$:

$$a^l(i, h) = (w_h^l * f^{l-1})(i) = \sum_{j=-\infty}^{\infty} \sum_{m=1}^{M_{l-1}} w_h^l(j, m)f^{l-1}(i-j, m)$$

Sau đó, đầu ra của quá trình tích chập được đưa qua hàm phi tuyến $f^l = h(a^l)$. Tham số kích thước bộ lọc (fillter size) quyết định phạm vi tiếp nhận (receptive feild) của mỗi nút đầu ra. Nếu không có zero padding, đầu ra của quá trình tích chập ở mỗi lớp sẽ có chiều rộng (width) $N_l = N_{l-1} - k + 1$ với $l = 1, \dots, L$. Đầu ra của mạng nơ ron sau L lớp tích chập sẽ là ma trận f^L , có kích thước phụ thuộc vào kích thước bộ lọc và số lượng bộ lọc sử dụng

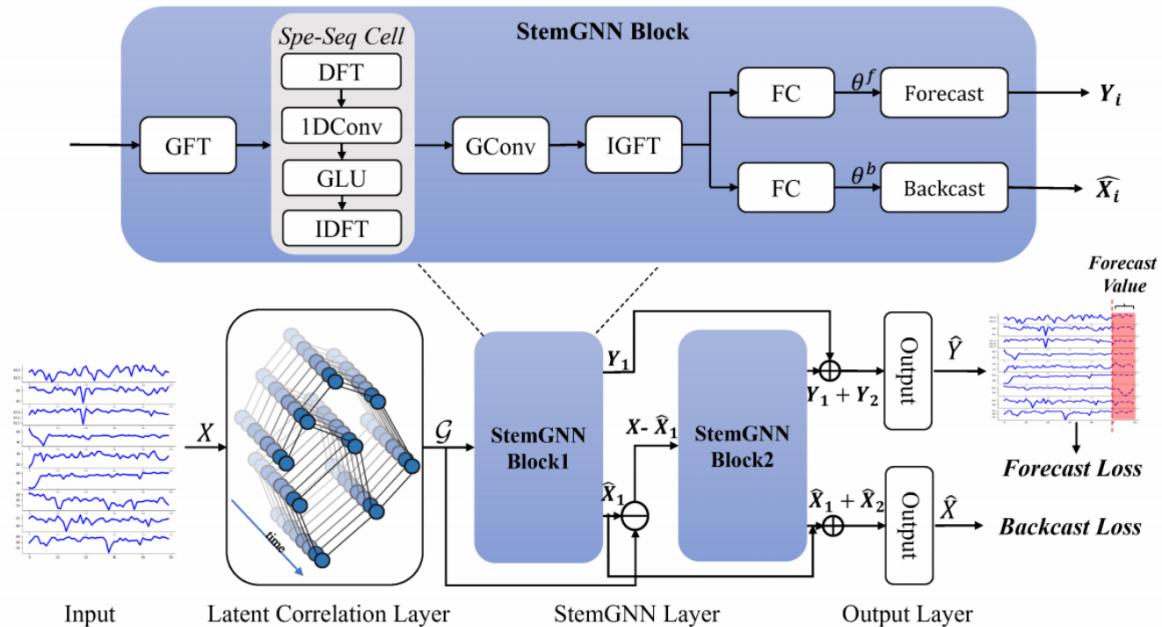
ở lớp cuối. Tùy vào những gì chúng ta muốn mô hình học, các trọng số trong mô hình sẽ được huấn luyện để giảm thiểu sai số giữa đầu ra của mạng f^L và đầu ra thực sự mà chúng ta quan tâm.

3.4 Mang nơ ron đồ thị thời gian quang phổ StemGNN

StemGNN (Spectral temporal graph Neural network) là một trong các phương pháp được Defu Cao và cộng sự [9] đề xuất để dự báo chuỗi thời gian.

Mô hình tổng quát được biểu diễn trong biểu đồ 11

- Dữ liệu đầu vào X được đưa vào dưới dạng lớp tương quan tiềm ẩn (Latent Correlation Layer), đầu ra là đồ thị với ma trận trọng số W .
- Đồ thị $\mathcal{G} = (X, W)$ đóng vai trò là đầu vào cho khối StemGNN.
- Đầu ra của khối StemGNN sẽ đi qua GLU và các lớp FC. Có 2 loại đầu ra của mạng là dự báo xuôi Y_i và dự báo ngược \hat{X}_i .



Hình 11: Mô hình StemGNN

3.4.1 Lớp tương quan tiềm ẩn

Mục đích của lớp tương quan tiềm ẩn là thông qua dữ liệu đầu vào, có thể trích xuất ra được mối tương quan liên quan tới chuỗi trong dữ liệu, từ đó làm đầu vào cho các lớp tiếp theo. Với cơ sở mạng GNN, ta cần xây dựng cấu trúc đồ thị trong mô hình chuỗi thời gian đa biến.

Với đầu vào là $X \in \mathbb{R}^{N \times T}$ đẩy vào lớp GRU (Gated Recurrent Unit), mô hình thực hiện tính toán các trạng thái ẩn tương ứng với thời điểm t . Sau đó, ta sử dụng trạng thái ẩn R cuối cùng để biểu diễn cho toàn bộ chuỗi thời gian và tính toán ma trận trọng số W bằng cơ chế tự chú ý (self-attention) cho dưới dạng sau:

$$Q = RW^Q, K = RW^K, W = \text{Softmax} \left(\frac{QK^T}{\sqrt{d}} \right)$$

với Q và K ký hiệu cho biểu diễn của truy vấn và khóa, được tính toán bởi phép chiếu tuyến tính với tham số có thể học W^Q và W^K trong phương pháp attention, và d là kích thước cõi ẩn của Q và K . Ma trận đầu ra $W \in \mathbb{R}^{N \times N}$ được xem là ma trận liền kề của đồ thị \mathcal{G} . Độ phức tạp thời gian của phương pháp self-attention là $O(N^2d)$.

3.4.2 Mạng StemGNN

Mạng StemGNN được xây dựng bằng cách xếp nhiều khối StemGNN mà bỏ qua việc kết nối. Một khối StemGNN được thiết kế bằng việc nhúng chuỗi phỏ (Spectral Sequential) vào mô đun Convolution Spectral. Trong phần này, ta sẽ tìm hiểu về kiến trúc của khối StemGNN, và mô tả khái quát mô đun Spe-Seq và tích chập đồ thị phỏ.

Spectral Sequential Cell (Spe-Seq Cell)

Phần tử Spe-Seq \mathcal{S} hướng tới việc phân rã mỗi thành phần chuỗi thời gian sau khi áp dụng GFT vào frequency basis và học các đặc trưng biểu diễn trên chúng. Mỗi phần tử chứa bốn thành phần: Discrete Fourier Transform (DFT, \mathcal{F}), 1D convolution, GLU và Inverse Discrete Fourier Transform (IDFT, \mathcal{F}^{-1}), với DFT và IDFT chuyển đổi dữ liệu chuỗi thời gian giữa miền tạm thời và miền frequency. Đặc biệt, đầu ra của DFT có phần thực (\hat{X}_u^r) và phần ảo (\hat{X}_u^i) được tính toán bởi cùng một công thức với các tham số khác nhau. Công thức được biểu diễn ở:

$$M^* (\hat{X}_u^*) = \text{GLU} (\theta_\tau^* (\hat{X}_u^*), \theta_\tau^* (\hat{X}_u^*)) = \theta_\tau^* (\hat{X}_u^*) \odot \sigma^* (\theta_\tau^* (\hat{X}_u^*)), * \in \{r, i\} \quad (3)$$

với θ_τ^* là nhân tích chập với kích thước là 3 trong phép tính, \odot là phép nhân Hadamard và cỗng sigmoid phi tuyến tính σ^* xác định số lượng thông tin trong đầu vào liên quan tới pattern chuỗi. Cuối cùng, kết quả thu được bởi $M^r(\hat{x}_u^r) + iM^i(\hat{x}_u^i)$ và ta tính toán đầu ra bằng cách áp dụng IGFT.

Khối StemGNN

Tích chập đồ thị phổ được sử dụng rộng rãi trong các mô hình dự báo chuỗi thời gian bởi tính nổi trội trong việc học các biểu diễn ẩn của chuỗi thời gian đa biến trong miền phổ. Chìa khóa cốt lõi là áp dụng Đồ thị biến đổi Fourier (Graph Fourier Transform, GFT) để thu được các mối quan hệ. Lưu ý rằng đầu ra của GFT là chuỗi thời gian đa biến trong khi GFT không tìm hiểu các mối quan hệ trong chuỗi thời gian một cách rõ ràng, do đó mà ta có thể tận dụng Biến đổi Fourier rời rạc (DFT) để học cách biểu diễn của chuỗi thời gian đầu vào trong miền chu kỳ với cơ sở lượng giác, nhận dạng được các dấu hiệu lặp lại trong dữ liệu có tính chu kỳ hoặc các đặc trưng tự tương quan giữa các mốc thời gian khác nhau. Khi đó đầu ra của Spe-Seq Cell được tiến hành bởi toàn bộ các thành phần của tích chập đồ thị phổ.

Mô hình này có thể được mở rộng trên các channel khác nhau. Áp dụng GFT và Spe-Seq Cell trên mỗi channel X_i của dữ liệu đầu vào và tổng hợp kết quả sau khi dùng tích chập đồ thị với kernel Θ_{ij} . Tiếp theo, ta áp dụng phép biến đổi ngược đồ thị Fourier trên tổng để thu được channel thứ j Z_j của đầu ra, có thể viết dưới dạng sau

$$Z_j = \mathcal{G}\mathcal{F}^{-1} \left(\sum_i g_{\Theta_{ij}}(\Lambda_i) \mathcal{S}(\mathcal{G}\mathcal{F}(X_i)) \right)$$

với ký hiệu $\mathcal{G}\mathcal{F}, \mathcal{G}\mathcal{F}^{-1}$ và \mathcal{S} ký hiệu cho GFT, IGFT và Spe-Seq Cell, Θ_{ij} là đồ thị tích chập kernel tương ứng với đầu vào thứ i và channel thứ j , và Λ_i là ma trận giá trị riêng của Laplace đã chuẩn hóa và một số véc tơ riêng được sử dụng trong GFT tương đương với biến nhiều chiều (N) mà không giảm số chiều. Sau đó ta gộp mỗi đầu ra channel Z_j để thu được kết quả cuối cùng Z .

Tích chập đồ thị phổ

Tích chập đồ thị phổ được phân tách thành 3 bước:

- Các chuỗi thời gian đầu vào được chiếu lên miền phổ bởi GFT.
- Biểu diễn phổ được lọc bằng toán tử tích chập trên đồ thị với nhân có thể học được.
- Thuật toán IGFT được áp dụng đối với biểu diễn phổ để thu được đầu ra cuối.

4 Khảo sát các bài báo về chủ đề dự báo chuỗi thời gian

Việc xây dựng các mô hình dự báo chuỗi thời gian đã và đang là một vấn đề quan trọng trong ngành toán học và khoa học máy tính nói chung và đặc biệt có nhiều ứng dụng trong các vấn đề như dự báo khí hậu [10], phân tích tài chính [11], đưa ra các quyết định thương mại trong kinh tế [12], ... Các phương pháp dự báo chuỗi thời gian truyền thống tập trung vào các mô hình có tham số được xây dựng từ dữ liệu và kiến thức chuyên ngày như mô hình AR [13], mô hình *exponential smoothing* [14], ... Các mô hình học máy hiện đại trong dự báo chuỗi thời gian được xây dựng dựa trên tính động (liên tục cập nhật) của dữ liệu và chỉ dùng dữ liệu để xây dựng mô hình [15]. Với sự tăng cường khả năng thu thập dữ liệu và tính toán trong thời gian gần đây, học máy đã trở thành một phần cần thiết của thế hệ tiếp theo của các mô hình dự báo chuỗi thời gian. Trong đó, học sâu đặc biệt đã được phổ biến trong thời gian gần đây, được truyền cảm hứng bởi các thành tựu đáng kể trong phân loại hình ảnh [16], xử lý ngôn ngữ tự nhiên [17] và học tăng cường [18]. Bằng cách kết hợp các giả thiết kiến trúc tùy chỉnh - hoặc các động lực suy luận [18] - phản ánh các chi tiết nhỏ của bộ dữ liệu cơ bản, các mạng nơ-ron sâu có thể học các biểu diễn dữ liệu phức tạp [19], từ đó giảm thiểu nhu cầu thiết kế đặc trưng và mô hình thủ công. Sự có sẵn của các framework lan truyền ngược mã nguồn mở [?] cũng đã đơn giản hóa việc huấn luyện mạng, cho phép tùy chỉnh các thành phần và hàm mất mát của mạng.

4.1 Mô hình dự báo dựa trên mô hình thống kê

Mô hình thống kê thường xuyên được sử dụng trong việc dự báo chuỗi thời gian. Mô hình AR và ARIMA [20] là hai trong đó. Mô hình ARIMA là mô hình chuỗi thời gian có dạng tổ hợp tuyến tính của các quan sát trong quá khứ cùng chuỗi white noise. Phương pháp Box-Jenkins có thể giúp tìm được bậc tối ưu cho mô hình. Tuy nhiên, mô hình này cần giả thiết chuỗi thời gian có tính dừng, trong khi dữ liệu thực tế thường không thỏa mãn ràng buộc này. Hơn nữa, các mô hình này chỉ có thể sử dụng cho chuỗi thời gian đơn biến và bỏ qua sự phụ thuộc giữa các biến. Một số mô hình biến thể như VAR [21] giúp mở rộng mô hình ban đầu trên không gian đa biến, tuy nhiên do hạn chế của mô hình với phép tìm kiếm trên lưới nhằm lựa chọn bậc mô hình, độ phức tạp thời gian tăng lên đáng kể.

4.2 Mô hình dự báo dựa trên mô hình học máy

Dự báo chuỗi thời gian sử dụng một phần quan sát cùng các đặc trưng để dự báo tương lai, do đó mà các mô hình hồi quy tuyến tính cũng như phi tuyến được sử dụng để dự báo chuỗi thời gian. So với hồi quy tuyến tính, hồi quy Ridge thêm L2 regularization vào trong hàm mất mát. Mô hình SVR có mục tiêu tìm siêu phẳng có tổng khoảng cách tới các điểm dữ liệu là cực tiểu [22]. KNN được mở rộng để dự báo chuỗi thời gian dựa trên các hàng xóm gần nhất của chuỗi đầu vào [23]. Một lợi thế của mô hình dự báo chuỗi thời gian dựa trên học máy là tính linh hoạt trong việc chọn các đặc trưng để dự đoán. Tuy nhiên, trích chọn đặc trưng trong các phương pháp này đòi hỏi kỹ năng của người dùng và thường tốn nhiều thời gian.

4.3 Mô hình dự báo dựa trên mạng Deep Neural

Trong các năm gần đây, mạng Deep Neural được ứng dụng thành công trong nhiều lĩnh vực khác nhau. Ví dụ, mô hình mạng CNN thể hiện hiệu suất nổi trội hơn hẳn trong các công việc liên quan tới thị giác máy tính bởi khả năng phân tích các đặc trưng cục bộ từ dữ liệu ảnh đầu vào. RNN được sử dụng cho các mô hình chuỗi và có ứng dụng sâu rộng trong quá trình xử lý ngôn ngữ tự nhiên, ví dụ như dịch máy và nhận dạng giọng nói. Hiệu năng của RNN cơ bản giảm mạnh nếu độ dài chuỗi tăng lên. Hai biến thể của RNN, LSTM và GRU, giải quyết vấn đề này với nhiều thiết kế mạng phức tạp hơn.

Cả CNN và RNN đều chỉ có thể phân tích được các biểu diễn tiềm ẩn từ dữ liệu dạng Euclid như hình ảnh, video, văn bản, trong khi một số dạng dữ liệu khác không có dạng Euclid: một số dữ liệu với cấu trúc đồ thị không thể xử lý nếu chỉ dùng trực tiếp CNN hoặc RNN. Gần đây, các nhà nghiên cứu quan tâm nhiều hơn tới việc mở rộng cách tiếp cận học sâu cho đồ thị dạng dữ liệu và có nhiều mô hình khác nhau được đề xuất. Mô hình GCN [24] là một trong các ví dụ nổi bật.

Mạng Neural thường được áp dụng trong việc dự báo chuỗi thời gian đa biến do chúng có thể biểu diễn mối quan hệ phi tuyến và học các pattern phức tạp từ tín hiệu đầu vào. Các nhà nghiên cứu đã áp dụng nhiều mạng Neural khác nhau trong việc dự báo mô hình để huấn luyện và khám phá các phụ thuộc ý nghĩa và các mối quan hệ tự phụ thuộc.

Dầu tiên, ta áp dụng mô hình RNN cơ bản và một vài các mô hình chuỗi [24] được sử dụng rộng rãi để huấn luyện các mối quan hệ phụ thuộc tạm thời. Kiến trúc encoder-decoder thường xuyên được sử dụng. Encoder là chuỗi các RNN dùng để encode chuỗi đầu vào thành vec tơ ngữ cảnh, và decoder (cũng là chuỗi RNN) tạo ra chuỗi độ dài biến đổi dần ra.

Độ dài của đầu vào và đầu ra có thể khác nhau. Với công việc dự báo chuỗi thời gian, ra sử dụng chuỗi thời gian trong lịch sử là chuỗi đầu vào cho encoder, và decoder trả về giá trị dự báo của chuỗi thời gian tại thời điểm tiếp theo [24].

Tiếp theo, có nhiều kỹ thuật khác nhau được phát triển để trích xuất quan hệ phụ thuộc giữa các chuỗi thời gian đa biến. Lấy ví dụ, trong mô hình LSTNet model, quan hệ phụ thuộc được huấn luyện bởi 2-D CNN thông qua chuỗi thời gian dưới dạng trọng số của CNN. Một số kết quả khác sử dụng cơ chế attention để huấn luyện quan hệ tự phục thuộc [24]. Ngoài ra, các kết quả gần đây cố gắng để mô hình hóa các thông tin bởi mạng đồ thị Neural [25].

Ngoài ra, cả quan hệ phụ thuộc tạm thời lẫn quan hệ tự phụ thuộc đều quan trọng trong việc đánh giá dự báo, một số mô hình hybrid cho ra kết quả tốt hơn. Lấy ví dụ, quá trình R2N2 [26] biểu diễn chuỗi thời gian đa biến với mô hình tuyến tính đơn giản và lõi phần dư với mô hình RNN. Mô hình LSTNet hybrid khác [7] phân tích các đặc trưng ngắn hạn bởi RNN, và sử dụng RNN để học các quan hệ phụ thuộc tạm thời. Thành phần tuyến tính cũng được tích hợp vào mô hình LSTNet để vượt qua nhược điểm của mạng Deep Neural, ví dụ như kích cỡ của đầu ra không nhạy trên kích cỡ đầu vào. Lấy ví dụ, trong một số dữ liệu thực tế, kích thước của đầu vào có thể thay đổi linh hoạt khi một số trường không được quan sát liên tục, lúc này độ chính xác của mô hình mạng Neural bị giảm đi đáng kể.

5 Phương pháp luận

5.1 Dự báo chuỗi thời gian dựa trên các mô hình thống kê truyền thống

5.1.1 Mô hình VAR

Nhóm tác giả sẽ trình bày lý thuyết về mô hình VAR trong [27].

Quá trình tự tương quan với véc tơ m chiều với bậc là p , ký hiệu là $\text{VAR}(p)$, là quá trình thỏa mãn

$$\mathbf{X}_t = \boldsymbol{\theta}_0 + \Phi_1 \mathbf{X}_{t-1} + \cdots + \Phi_p \mathbf{X}_{t-p} + \mathbf{Z}_t \quad (4)$$

hoặc

$$\Phi_p(B) \mathbf{X}_t = \boldsymbol{\theta}_0 + \mathbf{Z}_t$$

với \mathbf{Z}_t là chuỗi véc tơ white noise m chiều, $\text{VWN}(\mathbf{0}, \Sigma)$, và

$$\Phi_p(B) = \mathbf{I} - \Phi_1 B - \cdots - \Phi_p B^p$$

Dễ chứng minh rằng quá trình rõ ràng có tính khả nghịch. Ngoài ra quá trình có tính dừng nếu nghiệm của $|\mathbf{I} - \Phi_1 B - \cdots - \Phi_p B^p|$ nằm ngoài đường tròn đơn vị, cụ thể hơn thì ta cần mọi nghiệm của phương trình

$$|\lambda^p \mathbf{I} - \lambda^{p-1} \Phi_1 - \cdots - \Phi_p| = 0$$

nằm ngoài đường tròn đơn vị. Trong trường hợp này, kỳ vọng của chuỗi là véc tơ $E(\mathbf{X}_t) = \boldsymbol{\mu}$ có thể tính toán thông qua công thức

$$\begin{aligned} \boldsymbol{\mu} &= E(\mathbf{X}_t) = E(\boldsymbol{\theta}_0 + \Phi_1 \mathbf{X}_{t-1} + \cdots + \Phi_p \mathbf{X}_{t-p} + \mathbf{Z}_t) \\ &= \boldsymbol{\theta}_0 + \Phi_1 \boldsymbol{\mu} + \cdots + \Phi_p \boldsymbol{\mu} \end{aligned}$$

và do đó

$$\boldsymbol{\mu} = (\mathbf{I} - \Phi_1 - \cdots - \Phi_p)^{-1} \boldsymbol{\theta}_0$$

Ta có

$$\boldsymbol{\theta}_0 = (\mathbf{I} - \Phi_1 - \cdots - \Phi_p) \boldsymbol{\mu} = (\mathbf{I} - \Phi_1 B - \cdots - \Phi_p B^p) \boldsymbol{\mu},$$

nên phương trình 4 có thể viết dưới dạng

$$\Phi_p(B) \dot{\mathbf{X}}_t = \mathbf{Z}_t$$

hoặc

$$\dot{\mathbf{X}}_t = \Phi_1 \dot{\mathbf{X}}_{t-1} + \cdots + \Phi_p \dot{\mathbf{X}}_{t-p} + \mathbf{Z}_t$$

với $\dot{\mathbf{Z}}_t = \mathbf{Z}_t - \boldsymbol{\mu}$.

5.1.2 Mô hình VARMA

Mô hình véc tơ m chiều được gọi là quá trình VARMA với bậc là p và q , được ký hiệu là VARMA(p, q) là mô hình thỏa mãn:

$$\mathbf{X}_t = \boldsymbol{\theta}_0 + \Phi_1 \mathbf{X}_{t-1} + \cdots + \Phi_p \mathbf{X}_{t-p} + \mathbf{Z}_t - \Theta_1 \mathbf{Z}_{t-1} - \cdots - \Theta_q \mathbf{Z}_{t-q} \quad (5)$$

hoặc

$$\Phi_p(B) \mathbf{X}_t = \boldsymbol{\theta}_0 + \Theta_q(B) \mathbf{Z}_t \quad (6)$$

với \mathbf{Z}_t là véc tơ quá trình nhiễu m chiều, VWN($\mathbf{0}, \Sigma$), và

$$\begin{aligned} \Phi_p(B) &= \mathbf{I} - \Phi_1 B - \cdots - \Phi_p B^p, \\ \Theta_q(B) &= \mathbf{I} - \Theta_1 B - \cdots - \Theta_q B^q \end{aligned}$$

Mô hình có tính dừng nếu các nghiệm của ma trận định thức $|\Phi_p(B)|$ nằm ngoài đường tròn đơn vị, do đó

$$\begin{aligned} \dot{\mathbf{X}}_t &= [\Phi_p(B)]^{-1} \Theta_q(B) \mathbf{Z}_t \\ &= \sum_{j=0}^{\infty} \Psi_j \mathbf{Z}_{t-j} \end{aligned}$$

Mô hình được gọi là có tính khả nghịch nếu mọi nghiệm của $|\Theta_q(B)|$ nằm ngoài đường tròn đơn vị do đó

$$\Pi(B) \dot{\mathbf{X}}_t = \mathbf{Z}_t$$

với

$$\Pi(B) = [\Theta_q(B)]^{-1} \Phi_p(B) = \mathbf{I} - \sum_{j=0}^{\infty} \Pi_j B^j$$

Đối với n quan sát trên véc tơ chuỗi thời gian, \mathbf{X}_t , $t = 1, 2, \dots, n$, khi mô hình VARMA(p, q) được định nghĩa bởi

$$\mathbf{X}_t - \Phi_1 \mathbf{X}_{t-1} - \cdots - \Phi_p \mathbf{X}_{t-p} = \boldsymbol{\theta}_0 + \mathbf{Z}_t - \Theta_1 \mathbf{Z}_{t-1} - \cdots - \Theta_q \mathbf{Z}_{t-q} \quad (7)$$

Ước lượng hiệu quả của các tham số, $\Phi = (\Phi_1, \dots, \Phi_p)$, $\boldsymbol{\theta}_0$, $\Theta = (\Theta_1, \dots, \Theta_q)$, và Σ , được tính toán bằng phương pháp ước lượng hợp lý cực đại. Đặc biệt, giả sử rằng véc tơ nhiễu là Gauss, hàm log hợp lý cho bởi công thức:

$$\begin{aligned} \ln L(\Phi, \boldsymbol{\theta}_0, \Theta, \Sigma | \mathbf{X}) &= -\frac{nm}{2} \ln 2\pi - \frac{n}{2} |\Sigma| - \frac{1}{2} \sum_{t=1}^n \mathbf{Z}'_t \Sigma^{-1} \mathbf{Z}_t \\ &= -\frac{nm}{2} \ln 2\pi - \frac{n}{2} |\Sigma| - \frac{1}{2} \text{tr } \Sigma^{-1} S(\Phi, \boldsymbol{\theta}_0, \Theta), \end{aligned}$$

với

$$\mathbf{Z}_t = \mathbf{X}_t - \Phi_1 \mathbf{X}_{t-1} - \cdots - \Phi_p \mathbf{X}_{t-p} - \boldsymbol{\theta}_0 + \Theta_1 \mathbf{Z}_{t-1} + \cdots + \Theta_q \mathbf{Z}_{t-q}$$

và

$$S(\Phi, \boldsymbol{\theta}_0, \Theta) = \sum_{t=1}^n \mathbf{Z}_t \mathbf{Z}'_t$$

Sau khi ước lượng các tham số, cần phải kiểm tra tính phù hợp của mô hình thông qua việc phân tích giá trị phần dư

$$\hat{\mathbf{Z}}_t = \mathbf{X}_t - \hat{\Phi}_1 \mathbf{X}_{t-1} - \cdots - \hat{\Phi}_p \mathbf{X}_{t-p} - \hat{\boldsymbol{\theta}}_0 + \hat{\Theta}_1 \hat{\mathbf{Z}}_{t-1} + \cdots + \hat{\Theta}_q \hat{\mathbf{Z}}_{t-q}$$

Với mô hình phù hợp, chuỗi véc tơ phần dư sẽ có dạng chuỗi White noise.

Sau khi phân tích phần dư, nếu mô hình phù hợp, có thể sử dụng để dự đoán các giá trị tương lai. Với mô hình tổng quát trong 7, dự đoán l bước tại thời điểm n được cho bởi:

$$\hat{\mathbf{X}}_n(\ell) = \hat{\Phi}_1 \hat{\mathbf{X}}_n(\ell-1) + \cdots + \hat{\Phi}_p \hat{\mathbf{X}}_n(\ell-p) + \hat{\boldsymbol{\theta}}_0 + \hat{\mathbf{Z}}_n(\ell) - \hat{\Theta}_1 \hat{\mathbf{Z}}_n(\ell-1) - \cdots - \hat{\Theta}_q \hat{\mathbf{Z}}_n(\ell-q),$$

với $\hat{\mathbf{X}}_n(j) = \mathbf{X}_{n+j}$ ($j \leq 0$), $\hat{\mathbf{Z}}_{n+j} = \mathbf{0}$, $j > 0$, và $\hat{\mathbf{Z}}_{n+j} = \mathbf{Z}_{n+j}$ khi $j \leq 0$. Phương trình trên con có thể dùng để ước lượng các tham số và mối quan hệ nằm trong véc tơ.

5.2 Dự báo chuỗi thời gian dựa trên các mô hình học sâu

5.2.1 Mô hình dự báo chuỗi thời gian dựa vào mạng tích chập với các chuỗi thời gian tương quan

Xét với chuỗi thời gian một chiều $x = (x_t)_{t=0}^{N-1}$. Ta đưa ra một mô hình tham số θ với nhiệm vụ dự báo ra giá trị tiếp theo $\hat{x}(t+1)$ dựa trên lịch sử của chuỗi $x(0), \dots, x(t)$. Hay nói cách khác là cực đại hóa hàm hợp lý

$$p(x \mid \theta) = \prod_{t=0}^{N-1} p(x(t+1) \mid x(0), \dots, x(t), \theta)$$

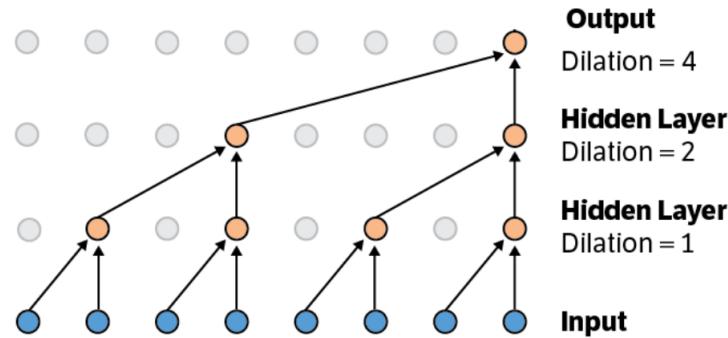
Dể cực đại hóa hàm hợp lý này thì Anastasia Borovykh và cộng sự [28] dùng mạng nơ ron tích chập ở dạng kiến trúc WaveNet [29] để cải tiến kiến trúc mạng nơ ron để áp dụng thành công cho việc dự đoán chuỗi thời gian.

Các chuỗi thời gian thường hiển thị sự tương quan dài hạn, vì vậy để cho phép mạng nơ ron học được phụ thuộc dài hạn này, Anastasia Borovykh và cộng sự sử dụng các lớp

tích chập dialated xếp chồng. Như được giới thiệu trong [30], tích chập dilated sẽ tạo ra một ngăn xếp feature map M_l được biểu diễn

$$(w_h *_d f^{l-1})(i) = \sum_{j=-\infty}^{\infty} \sum_{m=1}^{M_{l-1}} w_h^l(j, m) f^{l-1}(i - d \cdot j, m),$$

trong đó d là dialation factor và M_l là số lượng channels. Nói cách khác, trong tích chập dilated, bộ lọc được áp dụng vào mỗi phần tử thứ d trong vectơ đầu vào, cho phép mô hình hiểu quả học được các mối quan hệ giữa các điểm dữ liệu. Kiến trúc mạng sử dụng tương tự [30] với L lớp tích chập dilated $l = 1, \dots, L$ và với dilations tăng theo cấp số nhân: $d \in [2^0, 2^1, \dots, 2^{L-1}]$. Bộ lọc w có kích thước $1 \times k := 1 \times 2$. Ví dụ mạng tích chập dilated ba lớp được biểu diễn ở hình 12. Tại mỗi lớp, áp dụng tích chập dilated, theo sau là hàm phi tuyến để đưa ra feature map $f^t, t = 1, \dots, L$. Sau đó, L lớp tích chập dilated trong mô hình được tiếp theo bởi một lớp tích chập 1×1 giúp giảm số lượng channels về một, để mô hình đưa ra kết quả dưới dạng vector một chiều. Vì chúng ta quan tâm đến việc huấn luyện mô hình để dự đoán chuỗi thời gian $\hat{x} = (\hat{x}_t)_{t=0}^{N-1}$.



Hình 12: Mạng nơ ron tích chập dilated với ba lớp

Vùng tiếp nhận (receptive field) của mạng nơ ron được định nghĩa là tập hợp các phần tử đầu vào của điều chỉnh đầu ra của mạng nơ ron đó. Anastasia Borovykh và cộng sự đã định nghĩa vùng tiếp nhận r của mô hình là số lượng neurons trong đầu vào của lớp đầu tiên, tức là chuỗi thời gian, có thể sửa đổi đầu ra ở lớp cuối cùng, tức là dự báo chuỗi thời gian. Điều này phụ thuộc vào số lớp L và kích thước bộ lọc k , được tính bởi

$$r := 2^{L-1}k$$

Trong hình 12, vùng tiếp nhận $r = 8$, giá trị đầu ra bị ảnh hưởng bởi 8 neurons đầu vào.

Việc sử dụng zero padding có thể điều chỉnh được kích thước của đầu ra. nastasia Borovykh và cộng sự sử dụng tích chập nhân quả (causal convolutions), trong đó nhân quả

(causal) chỉ ra rằng đầu ra của tích chập không phụ thuộc vào đầu vào trong tương lai. Trong chuỗi thời gian, điều này tương đương với việc thêm một vector 0 có kích thước bằng kích thước vùng tiếp nhận, để đầu vào được xác định bởi:

$$[0, \dots, 0, x(0), \dots, x(N-1)] \in \mathbb{R}^{N+r}$$

và đầu ra của L lớp Wavenet là

$$[\hat{x}(0), \dots, \hat{x}(N)] \in \mathbb{R}^{N+1}$$

Tại thời điểm huấn luyện, dự đoán của $x(1), \dots, x(N)$ được tính bằng tích chập đầu vào với kernels tại mỗi lớp $l = 1, \dots, L$ và cuối cùng là tích chập 1×1 . Tại thời điểm thử nghiệm, dự đoán một bước $\hat{x}(t+1)$ với $t+1 \geq r$ được tính bởi $[x(t+1-r), \dots, x(t)]$ trong quá trình huấn luyện mô hình. Dự báo n bước được thực hiện theo cách tuần tự bằng cách từng dự đoán vào trong mạng tại bước kế tiếp ví dụ như dự báo hai bước $\hat{x}(t+2)$ được dự báo từ $[x(t+2-r), \dots, \hat{x}(t+1)]$. Ý tưởng của mạng là sử dụng các khả năng của mạng nơ ron tích chập như tự hồi quy (autoregressive). Trong mô hình tự hồi quy đơn giản bậc p dự báo giá trị $x(t+1)$ được tính bởi $\hat{x}(t+1) = \sum_{i=1}^p \alpha_i x_{t-i} + \epsilon(t)$, trong đó $\alpha_i, i = 1, \dots, p$ là hệ số (learnable weights) và $\epsilon(t)$ là white noise. Với mô hình WaveNet, kỳ vọng của dự báo với mọi $t \in \{0, \dots, N\}$ là

$$\mathbb{E}[x(t+1) | x(t), \dots, x(t-r)] = \beta_1(x(t-r)) + \dots + \beta_r(x(t)),$$

trong đó hàm $\beta_i, i = 1, \dots, r$ là thông số phụ thuộc vào dữ liệu và được tối ưu thông qua mạng tích chập.

Hàm mục tiêu:

Trọng số mạng, bộ lọc w_h^l được tối huấn luyện để cực tiểu MAE để tránh hiện tượng overfitting, Anastasia Borovykh và cộng sự sử dụng L2 regularization, khi đó hàm mục tiêu là:

$$E(w) = \frac{1}{N} \sum_{t=0}^{N-1} |\hat{x}(t+1) - x(t+1)| + \frac{\gamma}{2} \sum_{l=0}^L \sum_{h=1}^{M_{l+1}} (w_h^l)^2,$$

trong đó $\hat{x}(t+1)$ là giá trị dự báo của $x(t+1)$ sử dụng $x(0), x(1), \dots, x(t)$.

Tối ưu hóa trọng số

Mục đích của huấn luyện mô hình là tìm trọng số để cực tiểu hóa hàm mục tiêu. Tối ưu hóa trọng số dựa trên gradient descent, cập nhật trọng số dựa trên gradient của hàm lỗi

$$w_h^l(\tau+1) = w_h^l(\tau) - \eta \nabla E(w(\tau))$$

với $\tau = 1, \dots, T$, trong đó T là số lần huấn luyện và ∇ là learning rate. Vector gradient được tính thông qua backpropagation, điều này tương đương với áp dụng quy tắc chuỗi lặp đi lặp lại từ hàm sai số được tính toán trong lớp cuối cùng đến khi gradient theo trọng số lớp cần thiết được tính được w_l^h tính được

$$\frac{\partial E(w(\tau))}{\partial w_h^l(j, m)} = \sum_{i=1}^{N_l} \frac{\partial E(w(\tau))}{\partial f^l(i, h)} \frac{\partial f^l(i, h)}{\partial a^l(i, h)} \frac{\partial a^l(i, m)}{\partial w_h^l(j, m)}$$

trong đó tính tổng trên toàn bộ các nút chứa trọng số cần tính.

Hàm kích hoạt

Trong mỗi lớp, Anastasia Borovykh và cộng sự sử dụng hàm phi tuyến hoặc hàm kích hoạt để biến đổi đầu ra từ tích chập, điều này cho phép mô hình học các biểu diễn phi tuyến của dữ liệu. Trong mô hình, hàm phi tuyến được thể hiện dưới dạng hàm ReLU với $ReLU(x) := \max(x, 0)$, kết quả đầu ra ở lớp l là

$$f^l = [\text{ReLU}(w_1^l *_d f^{l-1}) + b, \dots, \text{ReLU}(w_{M_l}^l *_d f^{l-1} + b)]$$

trong đó $b \in \mathbb{R}$ là bias làm đổi đầu vào thành phi tuyến, $*_d$ là tích chập dilation và $f^t \in \mathbb{R}^{1 \times N_t \times M_{t+1}}$ là kết quả đầu ra của tích chập bộ lọc w_l^h , $h = 1, \dots, M_l$ trong lớp l .

Residual learning

Trong mạng nơ ron, để giải quyết vấn đề giảm độ chính xác khi thêm nhiều lớp mạng, chúng ta sử dụng kết nối dư (residual connections) sau mỗi phép tích chập được thực hiện từ đầu vào tới đầu ra của mạng. Khi $M_l > 1$, đầu ra từ phép phi tuyến được đi qua một phép tích chập 1×1 trước khi được thêm vào kết nối dư. Việc này giúp đảm bảo kết nối dư và đầu ra của phép tích chập có cùng số kênh. Nhờ vậy, chúng ta có thể xếp nhiều lớp mạng trong khi vẫn giữ được khả năng của mạng neural để chính xác ánh xạ các phụ thuộc học được trong các lớp ban đầu.

Conditioning

Khi dự báo chuỗi thời gian $x = (x_t)_t^{N-1} = 0$ dựa trên chuỗi thời gian $y = (y_t)_t^{N-1} = 0$, Anastasia Borovykh và cộng sự hướng tới cực đại hóa hàm hợp lý

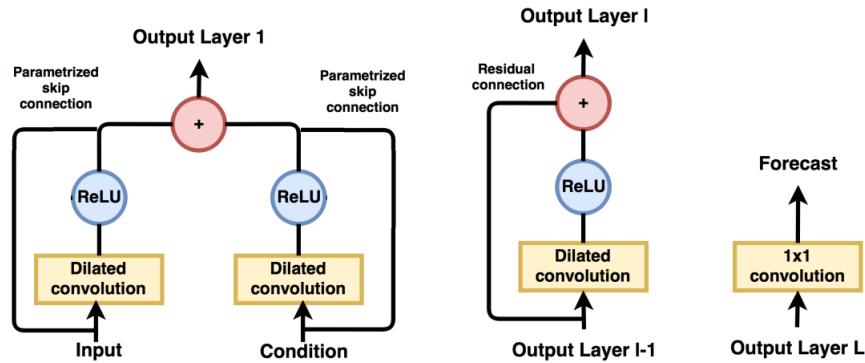
$$p(x \mid y, \theta) = \prod_{t=0}^{N-1} p(x(t+1) \mid x(0), \dots, x(t), y(0), \dots, y(t), \theta)$$

Điều kiện trên chuỗi thời gian y được áp dụng bằng cách tính toán hàm kích hoạt của tích chập với bộ lọc w_h^l và v_h^l trong lớp đầu tiên là

$$\text{ReLU}(w_h^l *_d x + b) + \text{ReLU}(v_h^l *_d y + b)$$

tại mỗi lần lọc $h = 1, \dots, M - 1$. Khi dự báo $x(t+1)$, vùng tiếp nhận của mạng chưa $x(0), \dots, x(t)$ và $y(0), \dots, y(t)$. Tương tự như đầu, để đảm bảo tính nhân quả, vectơ điều kiện được mở rộng với một vector 0 có cùng kích thước với vùng tiếp nhận. Trong [29], tác giả đề xuất sử dụng v_h^1 là bộ lọc 1×1 . Với cửa sổ đầu vào ngắn (short input window), loại điều kiện này không có khả năng bắt được tất cả các phụ thuộc giữa các chuỗi thời gian. Vì vậy, Anastasia Borovykh và cộng sự sử dụng tích chập $1 \times k$, tăng khả năng học được các phụ thuộc với số lớp ít hơn. Do đó, vùng tiếp nhận của mạng chứa $2^{L-1}k$ phần tử của cả đầu vào và các điều kiện (conditions).

Thay vì residual connection trong lớp đầu tiên, Anastasia Borovykh và cộng sự thêm các kết nối bỏ qua được tham số hóa bằng tích chập 1×1 từ cả đầu vào cũng như condition đến kết quả của tích chập dilated. Conditioning có thể dễ dàng được mở rộng thành chuỗi thời gian đa biến $M \times N$ bằng cách sử dụng M tích chập dilated từ mỗi condition riêng biệt và thêm chúng vào tích chập với đầu vào. Các kết nối skip trong mô hình được thiết kế để đảm bảo rằng mô hình có thể học được các mối quan hệ cần thiết giữa dự báo và cả đầu vào và condition một cách chính xác. Nếu một condition cụ thể không đóng góp cho dự báo, mô hình có thể loại bỏ nó bằng cách đặt trọng số trong kết nối skip về 0. Phương pháp này cho phép điều kiện cải thiện dự báo một cách có chọn lọc. Khi số lượng bộ lọc M_l lớn hơn một, kết nối skip tham số hóa sử dụng một tích chập 1×1 với bộ lọc M_l để đảm bảo rằng tổng của kết nối skip và tích chập gốc là hợp lệ. Kiến trúc mạng được minh họa trong hình 13.



Hình 13: Kiến trúc mạng Condition Convolution

Trong lớp đầu tiên (L) đầu vào và condition (với zero padding) được tích chập, truyền qua hàm phi tuyến. Kết quả từ lớp đầu tiên này là đầu vào trong lớp tích chập kế tiếp với

residual connection từ đầu vào đến đầu ra của tích chập. Quá trình này được lặp lại cho các tầng khác, cho đến khi chúng ta thu được đầu ra từ tầng L (M). Đầu ra này được truyền qua một tích chập 1×1 , dẫn đến đầu ra cuối cùng: chuỗi thời gian dự báo (R).

5.2.2 Mạng Hồi quy hai lớp kết hợp phương pháp Attention [1]

Mục tiêu của mô hình Dual-Stage Attention-Based Recurrent Neural Network (DA-RNN) với bản chất là mô hình ngoại sinh tự hồi quy phi tuyến (Nonlinear autoregressive exogenous, NARX), ta cần dự đoán giá trị tương lai dựa vào các giá trị đã quan sát. Ngoài ra mô hình trên còn có thể học được các yếu tố ảnh hưởng lâu dài trong chuỗi thời gian cũng như trích xuất các thành phần liên quan nhiều hơn trong mô hình dự báo.

Ký hiệu và phát biểu bài toán

Với chuỗi thời gian gồm n quan sát, $\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^n)^\top = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T) \in \mathbb{R}^{n \times T}$, với T là giá trị cửa sổ trượt, ta ký hiệu $\mathbf{x}^k = (x_1^k, x_2^k, \dots, x_T^k)^\top \in \mathbb{R}^T$ là đầu vào dự báo kích thước T và chuỗi $\mathbf{x}_t = (x_t^1, x_t^2, \dots, x_t^n)^\top \in \mathbb{R}^n$ ký hiệu cho véc tơ ngoại sinh n phần tử tại thời điểm t .

Khi huấn luyện mô hình, chuỗi đầu vào dự báo $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$ với $\mathbf{x}_t \in \mathbb{R}^n$ đi kèm với véc tơ mục tiêu $(y_1, y_2, \dots, y_{T-1})$ với $y_t \in \mathbb{R}$, đây được xem là giá trị hiện tại và quá khứ gồm trong chuỗi gồm n trường quan sát, các mô hình NARX được huấn luyện để học cách tạo một ánh xạ phi tuyến tới giá trị trong tương lai của chuỗi y_T .

$$\hat{y}_T = F(y_1, \dots, y_{T-1}, \mathbf{x}_1, \dots, \mathbf{x}_T).$$

với $F(\cdot)$ là ánh xạ phi tuyến cần huấn luyện.

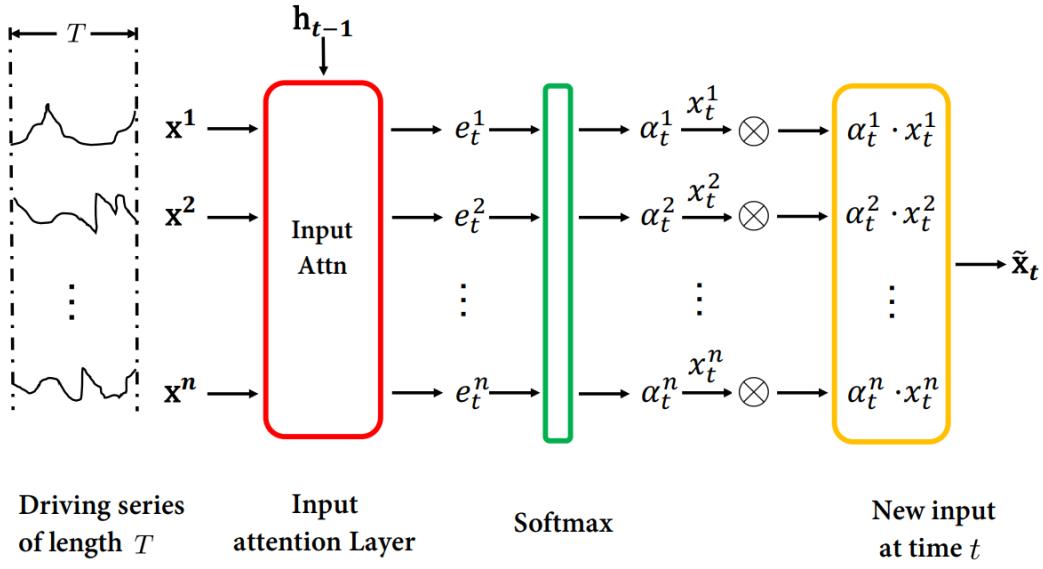
Tiếp theo ta sẽ tìm hiểu cấu trúc mô hình DA-RNN. Tổng quan cấu trúc được mô tả trong hình 14. Cơ chế input attention tính toán trọng số attention α_t^k cho chuỗi đầu vào đa biến $\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^n$ với giả thiết trạng thái ẩn là \mathbf{h}_{t-1} đặt trong encoder và sau đó được cho vào tính toán giá trị $\tilde{\mathbf{x}}_t = (\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \dots, \alpha_t^n x_t^n)^\top$ ở LSTM encoder.

Cơ chế input attention

Với chuỗi đầu vào dự báo $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$ with $\mathbf{x}_t \in \mathbb{R}^n$, n là số lượng quan sát trên mỗi trường, ta áp dụng encoder để tìm ra ánh xạ từ \mathbf{x}_t tới \mathbf{h}_t (tại thời điểm t) với

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \mathbf{x}_t)$$

giá trị $\mathbf{h}_t \in \mathbb{R}^m$ là trạng thái ẩn của encoder tại thời điểm t , m là kích thước trạng thái ẩn, và f_1 hàm activation phi tuyến, có thể là LSTM [6] hoặc GRU [31]. Trong khuôn khổ báo cáo,



Hình 14: Cơ chế input attention

ta sử dụng mô hình LSTM cho f_1 để bắt được các ảnh hưởng lâu dài. Mỗi đơn vị LSTM có một phần tử nhớ với trạng thái s_t tại thời điểm t . Phần tử nhớ sẽ được kiểm soát bởi ba cổng sigmoid: cổng quên f_t , cổng đầu vào i_t và cổng đầu ra o_t . Cách cập nhập trạng thái của LSTM như sau:

$$f_t = \sigma(\mathbf{W}_f [\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_f)$$

$$i_t = \sigma(\mathbf{W}_i [\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_i)$$

$$o_t = \sigma(\mathbf{W}_o [\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_o)$$

$$\mathbf{s}_t = f_t \odot \mathbf{s}_{t-1} + i_t \odot \tanh(\mathbf{W}_s [\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_s)$$

$$\mathbf{h}_t = o_t \odot \tanh(\mathbf{s}_t)$$

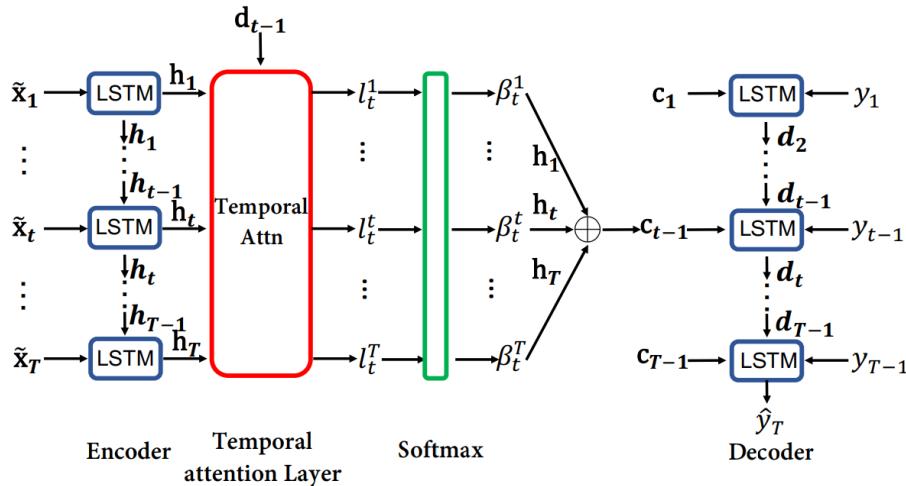
với $[\mathbf{h}_{t-1}; \mathbf{x}_t] \in \mathbb{R}^{m+n}$ là sự ghép nối trạng thái ẩn \mathbf{h}_{t-1} với đầu vào hiện tại \mathbf{x}_t . $\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_s \in \mathbb{R}^{m \times (m+n)}$, và $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o, \mathbf{b}_s \in \mathbb{R}^m$ là các tham số cần học. σ và \odot lần lượt là hàm logistic sigmoid và phép nhân phần tử (elementwise multiplication).

Với trường quan sát thứ k là $\mathbf{x}^k = (x_1^k, x_2^k, \dots, x_T^k)^\top \in \mathbb{R}^T$, ta có thể xây dựng cơ chế input attention thông qua mô hình attention xác định, ví dụ như perceptron đa lớp, sử dụng trạng thái ẩn quá khứ \mathbf{h}_{t-1} và phần tử trạng thái s_{t-1} trong đơn vị encoder LSTM:

$$e_t^k = \mathbf{v}_e^\top \tanh(\mathbf{W}_e [\mathbf{h}_{t-1}; \mathbf{s}_{t-1}] + \mathbf{U}_e \mathbf{x}^k) \quad (8)$$

và

$$\alpha_t^k = \frac{\exp(e_t^k)}{\sum_{i=1}^n \exp(e_t^i)},$$



Hình 15: Cơ chế Temporal Attention

với $\mathbf{v}_e \in \mathbb{R}^T$, $\mathbf{W}_e \in \mathbb{R}^{T \times 2m}$ và $\mathbf{U}_e \in \mathbb{R}^{T \times T}$ là các tham số cần học. Để ngắn gọn, ta bỏ qua thành phần bias trong 8. α_t^k là trọng số attention đánh giá độ quan trọng của trường thứ k trong đầu vào tại thời điểm t . Hàm softmax được áp dụng trong e_t^k để đảm bảo tổng các trọng số bằng 1. Cơ chế đầu vào attention được xem là để áp dụng vào mạng tiếp theo để có thể huấn luyện cùng với các thành phần khác của RNN. Với các trọng số đánh giá, ta có thể trích xuất chuỗi mới

$$\tilde{\mathbf{x}}_t = (\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \dots, \alpha_t^n x_t^n)^\top$$

Khi đó trạng thái ẩn tại thời điểm t có thể cập nhật bằng công thức:

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \tilde{\mathbf{x}}_t)$$

với f_1 là đơn vị LSTM có thể tính toán dựa trên các phương trình đã có ở trên với \mathbf{x}_t thay thế bởi chuỗi mới $\tilde{\mathbf{x}}_t$. Với cơ chế đầu vào attention đề xuất, encoder có thể chủ động tập trung vào các chuỗi quan trọng hơn là xem các trường đầu vào bình đẳng nhau.

Decoder với attention tạm thời

Để dự báo đầu ra \hat{y}_T , ta sử dụng mô hình RNN dựa trên LSTM để decode thông tin đầu vào đã encode. Biểu đồ 15 mô tả luồng hoạt động của cơ chế trên. Hệ thống temporal attention tính toán trọng số attention β_t^k dựa trên các decoder trạng thái ẩn ở quá khứ \mathbf{d}_{d-1} và biểu diễn thông tin đầu vào dưới dạng tổng trọng số của encoder trạng thái ẩn qua tất cả các bước. Véc tơ khởi tạo \mathbf{c}_t được sử dụng dưới dạng đầu vào của đơn vị decoder LSTM. Đầu ra \hat{y}_T của đơn vị decoder LSTM cuối chính là kết quả dự báo.

Để dự báo đầu ra \hat{y}_T , ta dựa trên mạng RNN với cơ sở LSTM để decode các thông tin đầu vào đã encode. Tuy nhiên, khi số lượng chuỗi đầu vào tăng lên thì hiệu suất của mạng

encoder-decoder trở nên giảm đi rõ rệt. Do đó, dựa trên encoder với đầu vào attention, cơ chế temporal attention được sử dụng trong decoder đáp ứng việc lựa chọn các encoder trạng thái ẩn liên quan qua tất cả các bước. Đặc biệt, trọng số attention của mỗi encoder trạng thái ẩn tại thời điểm t được tính toán dựa trên các decoder trạng thái ẩn trước đó $\mathbf{d}_{t-1} \in \mathbb{R}^p$ và phần tử trạng thái của đơn vị LSTM $\mathbf{s}'_{t-1} \in \mathbb{R}^p$ với

$$l_t^i = \mathbf{v}_d^\top \tanh(\mathbf{W}_d [\mathbf{d}_{t-1}; \mathbf{s}'_{t-1}] + \mathbf{U}_d \mathbf{h}_i), \quad 1 \leq i \leq T$$

và

$$\beta_t^i = \frac{\exp(l_t^i)}{\sum_{j=1}^T \exp(l_t^j)},$$

với $[\mathbf{d}_{t-1}; \mathbf{s}'_{t-1}] \in \mathbb{R}^{2p}$ là phép ghép nối của trạng thái ẩn trước đó và phần tử trạng thái của đơn vị LSTM. $\mathbf{v}_d \in \mathbb{R}^m$, $\mathbf{W}_d \in \mathbb{R}^{m \times 2p}$ và $\mathbf{U}_d \in \mathbb{R}^{m \times m}$ là các tham số có thể học. Hệ số bias được bỏ qua. Trọng số attention β_t^i đặc trưng cho độ quan trọng của encoder trạng thái ẩn thứ i cho việc dự đoán. Do mỗi encoder trạng thái ẩn \mathbf{h}_i được ánh xạ tới các thành phần temporal của đầu vào, cơ chế attention tính toán véc tơ \mathbf{c}_t và xem đây là trọng số tổng của mọi encoder trạng thái ẩn $\{\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T\}$

$$\mathbf{c}_t = \sum_{i=1}^T \beta_t^i \mathbf{h}_i.$$

\mathbf{c}_t có giá trị phân biệt tại mỗi bước.

Khi tính toán xong véc tơ tổng trọng số, ta tổng hợp chúng với chuỗi mục tiêu $(y_1, y_2, \dots, y_{T-1})$

$$\tilde{y}_{t-1} = \tilde{\mathbf{w}}^\top [y_{t-1}; \mathbf{c}_{t-1}] + \tilde{b},$$

với $[y_{t-1}; \mathbf{c}_{t-1}] \in \mathbb{R}^{m+1}$ được ghép bởi decoder đầu vào y_{t-1} và véc tơ \mathbf{c}_{t-1} . Tham số $\tilde{\mathbf{w}} \in \mathbb{R}^{m+1}$ và $\tilde{b} \in \mathbb{R}$ ánh xạ véc tơ đã ghép nối tới kích thước của decoder đầu vào. Giá trị mới \tilde{y}_{t-1} được sử dụng để cập nhật decoder trạng thái ẩn tại thời điểm t :

$$\mathbf{d}_t = f_2(\mathbf{d}_{t-1}, \tilde{y}_{t-1}).$$

Ta chọn hàm phi tuyến f_2 là một đơn vị LSTM, là một mô hình phổ biến để mô hình hóa các phụ thuộc dài hạn. Khi đó \mathbf{d}_t có thể được cập nhật:

$$\begin{aligned} \mathbf{f}'_t &= \sigma(\mathbf{W}'_f [\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}'_f) \\ \mathbf{i}'_t &= \sigma(\mathbf{W}'_i [\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}'_i) \\ \mathbf{o}'_t &= \sigma(\mathbf{W}'_o [\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}'_o) \\ \mathbf{s}'_t &= \mathbf{f}'_t \odot \mathbf{s}'_{t-1} + \mathbf{i}'_t \odot \tanh(\mathbf{W}'_s [\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}'_s) \\ \mathbf{d}_t &= \mathbf{o}'_t \odot \tanh(\mathbf{s}'_t) \end{aligned}$$

với $[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] \in \mathbb{R}^{p+1}$ là phép ghép nối giữa \mathbf{d}_{t-1} và $\tilde{y}_{t-1} \cdot \mathbf{W}'_f, \mathbf{W}'_i, \mathbf{W}'_o, \mathbf{W}'_s \in \mathbb{R}^{p \times (p+1)}$, và $\mathbf{b}'_f, \mathbf{b}'_i, \mathbf{b}'_o, \mathbf{b}'_s \in \mathbb{R}^p$ là các tham số có thể học. σ và \odot là các hàm logistic sigmoid và phép nhân elementwise multiplication.

Với mô hình NARX, ta cần sử dụng DA-RNN để xấp xỉ hàm F và thu được xấp xỉ của đầu ra \hat{y}_T với mọi quan sát đầu vào và mọi đầu ra trước đó. Đặc biệt, \hat{y}_T có thể thu được bởi

$$\begin{aligned}\hat{y}_T &= F(y_1, \dots, y_{T-1}, \mathbf{x}_1, \dots, \mathbf{x}_T) \\ &= \mathbf{v}_y^\top (\mathbf{W}_y [\mathbf{d}_T; \mathbf{c}_T] + \mathbf{b}_w) + b_v,\end{aligned}$$

Các tham số $\mathbf{W}_y \in \mathbb{R}^{p \times (p+m)}$ and $\mathbf{b}_w \in \mathbb{R}^p$ ánh xạ véc tơ ghép nối tối kinh thước của decoder trạng thái ẩn. Hàm tuyến tính có trọng số $\mathbf{v}_y \in \mathbb{R}^p$ và bias $b_v \in \mathbb{R}$ giúp ta tính toán giá trị dự đoán cuối cùng.

5.3 Dự báo chuỗi thời gian dựa trên các mô hình Hybrid

5.3.1 Mô hình hóa các mẫu thời gian dài và ngắn hạn bằng Mạng nơ-ron

Mô hình LSTNet được đề xuất bởi Lai Et al [7] vào năm 2018 dựa trên ý tưởng tiếp cận việc dự báo chuỗi thời gian tương tự việc sinh text cho ảnh tức dùng mô hình Convolution kết hợp với LSTM để dự báo chuỗi thời gian, đồng thời để tăng hiệu suất dự báo họ đề xuất khử trend của dữ liệu.

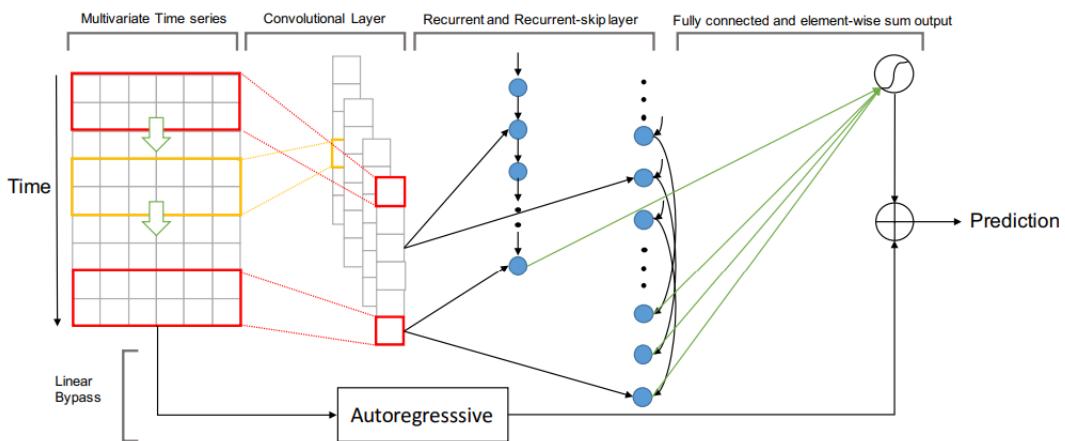


Figure 2: An overview of the Long- and Short-term Time-series network (LSTNet)

Khối AutoRegressive

Mặt hạn chế của các mô hình Học sâu như Convolution hay LSTM là tính khái quát hóa dữ liệu chuỗi thời gian quá mức dẫn tới việc không thích ứng với các thay đổi trong dữ liệu chuỗi thời gian. Trên các bộ dữ liệu thực tế, giá trị chuỗi thời gian thay đổi liên tục trong một chu kỳ rất nhỏ (chẳng hạn 5 phút) nên khi khái quát hóa dữ liệu quá mức có thể làm giảm chất lượng dự báo. Do đó, để tăng hiệu suất dự báo, Lai.et.al đã dùng mô hình AR để mô hình hóa khuynh của dữ liệu kết hợp với phần khái quát hóa dữ liệu thông qua các hàm phi tuyến của mạng Neural.

Ký hiệu kết quả dự báo của thành phần AR là $h_t^L \in \mathbb{R}^n$ và các hệ số của mô hình AR là $W^{ar} \in \mathbb{R}^{q^{ar}}$ và $b^{ar} \in \mathbb{R}$, với q^{ar} là kích thước của cửa sổ đầu vào trên ma trận đầu vào, ta có công thức sau:

$$h_{t,i}^L = \sum_{k=0}^{q^{ar}-1} W_k^{ar} y_{t-k,i} + b^{ar}$$

Dự đoán cuối cùng của LSTNet được thu được bằng cách tích hợp các đầu ra của phần mạng nơ-ron và thành phần AR:

$$\hat{Y}_t = h_t^D + h_t^L$$

với \hat{Y}_t là kết quả dự báo cuối cùng của mô hình ở bước t .

Hàm mục tiêu

Hàm lỗi bình phương là hàm mặc định được sử dụng cho nhiều tác vụ dự báo, và mục tiêu tối ưu tương ứng được định nghĩa như sau:

$$\underset{\Theta}{\text{minimize}} \sum_{t \in \Omega_T \text{ Train}} \|Y_t - \hat{Y}_{t-h}\|_F^2$$

Trong đó, Θ đại diện cho tập hợp các tham số của mô hình, Ω_{Train} là tập các điểm thời gian được sử dụng để huấn luyện, $\|\cdot\|_F$ là norm Frobenius, và h là khoảng thời gian dự báo như. Mô hình hồi quy tuyến tính truyền thống với hàm mất mát bình phương được gọi là Linear Ridge, tương đương với mô hình vector tự hồi quy với chính quy hóa ridge. Hàm mục tiêu

lúc đó là:

$$\underset{\Theta}{\text{minimize}} \frac{1}{2} \|\Theta\|_F^2 + C \sum_{t \in \Omega_{\text{Train}}} \sum_{i=0}^{n-1} \xi_{t,i}$$

v.d.k $|\hat{Y}_{t-h,i} - Y_{t,i}| \leq \xi_{t,i} + \epsilon, t \in \Omega_{\text{Train}}$

$$\xi_{t,i} \geq 0$$

6 Kết quả thực nghiệm

Trong phần này, nhóm tác giả sẽ tiến hành mô tả các bước khai phá dữ liệu, tiền xử lý dữ liệu và xây dựng các mô hình dự báo trong báo cáo gồm nhóm các mô hình Học máy, các mô hình Học sâu, các mô hình Hybrid để dự báo chuỗi thời gian đa biến.

Tính mới của nhóm được trình bày cụ thể trong Phần 6.3 này như sau:

- Hiểu ý tưởng được sử dụng trong các bài báo xây dựng mô hình dự báo chuỗi thời gian đa biến và hoàn toàn tự đề xuất và xây dựng lại các mô hình bằng framework Tensorflow.
- Khai phá dữ liệu và nhận thấy các đặc trưng của dữ liệu để xây dựng các mô hình phù hợp.
- Hyper-parameter tuning và xây dựng kiến trúc phù hợp với dữ liệu để dự báo đạt kết quả tốt nhất.

Code báo cáo: <https://github.com/orgs/TimeSeriesCK/repositories>

6.1 Tiền xử lý dữ liệu

Dữ liệu được cung cấp bao gồm 11 trường chuỗi thời gian và 4 trường liên quan đến chỉ số thời gian. Các trường chuỗi thời gian bao gồm: pH (đo độ pH của nước), EC, DO, TSS, TN, TP, TOC, ORP, Temp, Temp (Nhiệt độ nước), dow.

D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
pH	EC	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP	year	month	dow	hour	datehour	
7.29	1000	0.01	37.72	2.3826	118.3018	20.6403	257.62	32.64	28.20479	2017	2017-07	1	14	2017-07-11 14	
7.29	1000	0.01	37.18	7.284571	23.18288	14.90099	260.5	32.56	22.11204	2017	2017-07	1	14	2017-07-11 14	
7.29	1000	0.01	36.64	4.668972	11.3631	21.68547	255.54	32.54	33.1165	2017	2017-07	1	14	2017-07-11 14	
7.3	1000	0.01	36.25	2.14674	83.47461	24.29929	255.06	32.47	27.30068	2017	2017-07	1	14	2017-07-11 14	
7.31	1000	0.01	36.08	2.934312	12.72559	36.73038	258.62	32.42	33.86626	2017	2017-07	1	14	2017-07-11 14	
7.31	1000	0.01	35.83	5.594663	79.63951	11.19079	260.1	32.42	38.53592	2017	2017-07	1	14	2017-07-11 14	
7.31	1000	0.01	35.63	1.8919	92.43837	28.06502	258.92	32.42	24.90146	2017	2017-07	1	14	2017-07-11 14	
7.31	1000	0.01	35.52	3.548026	26.13504	25.17564	255.42	32.35	26.98888	2017	2017-07	1	14	2017-07-11 14	
7.31	1000	0.01	35.35	3.075528	130.6204	8.040837	252.52	32.3	29.32008	2017	2017-07	1	14	2017-07-11 14	
7.31	1000	0.01	35.26	2.058291	235.7047	8.77974	252.12	32.27	29.27418	2017	2017-07	1	14	2017-07-11 14	
7.35	1000	0.01	37.75	5.14182	137.1077	23.77295	191.75	32.42	33.56262	2017	2017-07	1	14	2017-07-11 14	
7.37	1000	0.01	39.44	3.445046	205.4531	19.36902	223.65	32.57	28.61158	2017	2017-07	1	15	2017-07-11 15	
7.38	1000	0.01	40.54	3.587051	73.60091	23.87226	242.94	32.81	25.68352	2017	2017-07	1	15	2017-07-11 15	
7.38	1000	0.01	40.45	6.362624	10.65344	16.94229	246.56	32.82	26.57401	2017	2017-07	1	15	2017-07-11 15	
7.38	1000	0.01	40.76	3.149138	43.70732	32.68112	248.9	32.82	26.05485	2017	2017-07	1	15	2017-07-11 15	
7.37	1000	0.01	39.26	2.627215	139.0557	16.326	252.12	32.82	29.27576	2017	2017-07	1	15	2017-07-11 15	
7.36	1000	0.01	38.24	2.753338	24.77645	10.63164	252.16	32.7	31.54065	2017	2017-07	1	15	2017-07-11 15	
7.37	1000	0.01	38.58	5.221459	49.45931	11.28356	252.25	32.68	25.30238	2017	2017-07	1	15	2017-07-11 15	
7.34	1000	0.01	37.48	4.924476	43.92275	10.16065	252.17	32.68	31.60849	2017	2017-07	1	15	2017-07-11 15	

Hình 16: Hình ảnh data ban đầu

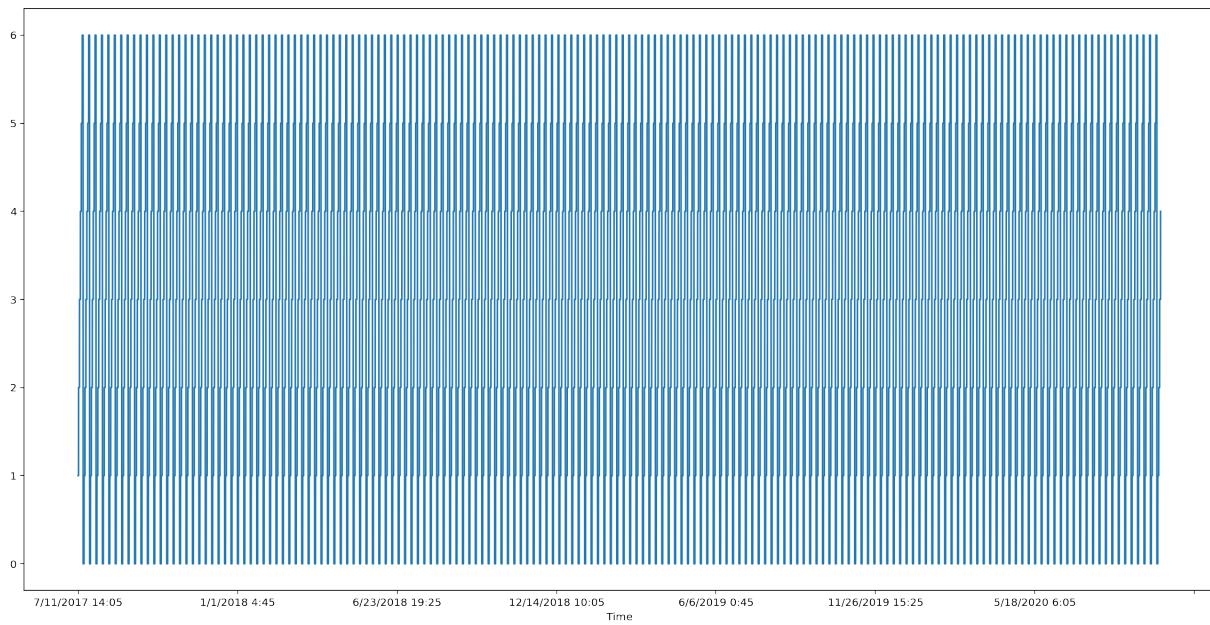
Ta có 339397 bản ghi và cứ sau 5 phút sẽ có một bản ghi dữ liệu. Tiến hành xem tỷ lệ Null của dữ liệu, ta có

```
Time      0.000000
pH        0.142895
EC        0.203111
DO        0.139085
TSS       0.000000
TN        0.000000
TP        0.000000
TOC       0.000000
ORP        0.131528
Temp      0.145240
TEMP      0.000000
year      0.000000
month     0.000000
dow       0.000000
hour      0.000000
datehour   0.000000
dtype: float64
```

Hình 17: Tỷ lệ null

Ta thấy trường pH, EC, ORP, Temp đều có tỷ lệ null trên 10% nên ta cần có cách tiến hành điền dữ liệu thiếu để dự báo chuỗi thời gian. Sau khi tiến hành kiểm thử các phương pháp điền dữ liệu thiếu như điền toàn bộ bằng 0, bằng median, bằng mean của dữ liệu thì nhóm tác giả nhận thấy phương pháp điền dữ liệu bằng mô hình KNN (K Nearest Neighbor) cho kết quả đánh giá trên tập Test tốt nhất.

Đồng thời, nhóm tác giả nhận thấy việc bỏ biến EC ra khỏi mô hình để dự báo riêng cho kết quả tốt hơn. Mặt khác, nhóm tác giả nhận thấy sau 24h giá trị của biến dow sẽ tăng lên 1 đơn vị và có miền giá trị từ 0 về 6, khi dow tăng đến 6 nó sẽ quay về 0.

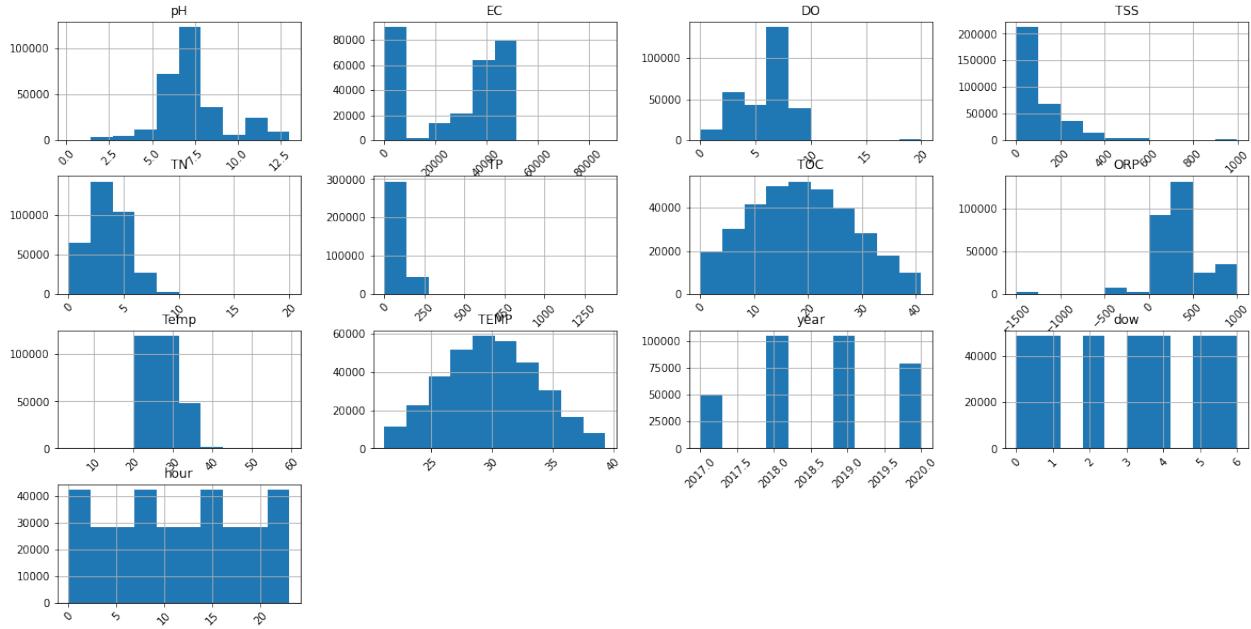


Hình 18: Chu kỳ của biến dow

Do đó, nhóm tác giả chỉ sử dụng 9 trường gồm 'pH', 'DO', 'TSS', 'TN', 'TP', 'TOC', 'ORP', 'Temp' để làm biến xây dựng mô hình dự báo.

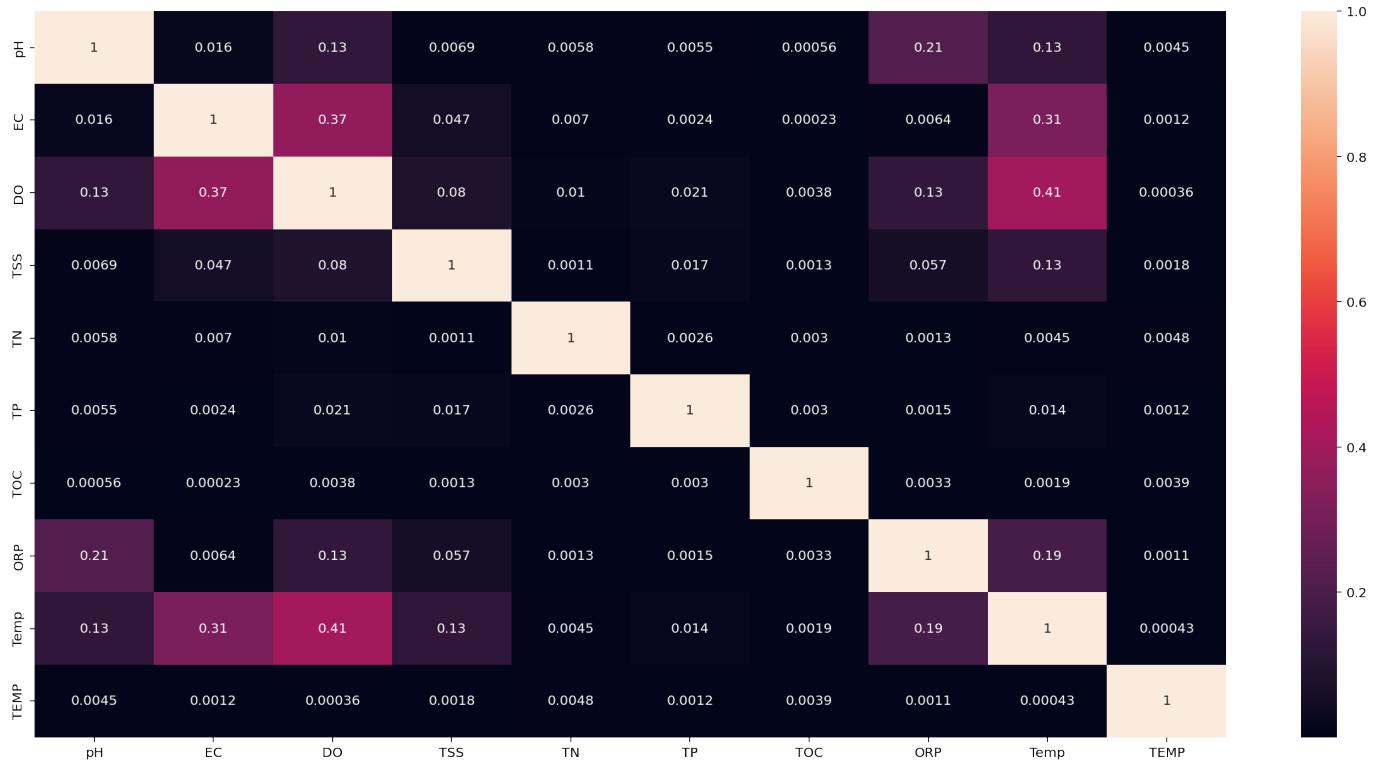
6.2 Khai phá dữ liệu

Ta tiến hành vẽ đồ thị histogram của các trường dữ liệu để xem phân bố dữ liệu thu được kết quả:



Hình 19: Histogram của các trường dữ liệu

Ta có thể thấy phân bố của TEMP, TOC tiệm cận phân phối chuẩn. Phân bố của EC khá bất thường do tỷ lệ null nhiều nên sau khi điền dữ liệu bị khuyết thì các giá trị tập trung từ 0-20000, do đó nhóm tác giả bỏ qua việc dự báo trường EC. Sau khi điền dữ liệu bị khuyết bằng KNN, ta sẽ có thông tin về correlation giữa các biến như sau:



Hình 20: Correlation giữa các biến

Ta có thể thấy tương quan giữa biến TEMP và DO khá cao. Các biến khác có tương quan không đáng kể. Để xem xét tới tính nhân quả tương tác giữa các biến hay nói cách khác xem độ ảnh hưởng của các biến tới nhau trong quá trình dự báo chuỗi thời gian đa biến ta sẽ sử dụng kiểm định tính nhân quả Granger. Granger's causality là một kiểm định thống kê được sử dụng để xác định liệu một chuỗi thời gian có thể được sử dụng để dự đoán một chuỗi thời gian khác hay không. Giả thuyết không có sự tương quan trong kiểm định Granger là các giá trị quá khứ của một chuỗi thời gian không giúp đỡ trong việc dự đoán các giá trị tương lai của một chuỗi thời gian khác. Nếu giá trị p được thu được từ kiểm định nhỏ hơn mức ý nghĩa 0,05, nó cho thấy có bằng chứng để bác bỏ giả thuyết không và chúng ta có thể kết luận rằng có một mối quan hệ nhân quả giữa hai chuỗi thời gian và do đó ta không thể dự báo đơn lẻ từng biến bằng các mô hình dự báo đơn biến mà bắt buộc phải dùng các mô hình dự báo đa biến.

	pH_x	DO_x	TSS_x	TN_x	TP_x	TOC_x	ORP_x	Temp_x	TEMP_x
pH_y	1.0000	0.0000	0.0003	0.0117	0.0776	0.0536	0.0000	0.0000	0.2320
DO_y	0.0000	1.0000	0.0000	0.4995	0.0033	0.2360	0.0000	0.0000	0.1360
TSS_y	0.3688	0.0000	1.0000	0.0931	0.0049	0.5127	0.0000	0.0000	0.5028
TN_y	0.0021	0.0000	0.1310	1.0000	0.0000	0.7307	0.5439	0.4172	0.7640
TP_y	0.0102	0.0000	0.0000	0.0000	1.0000	0.4224	0.0909	0.0000	0.1125
TOC_y	0.6425	0.0205	0.1789	0.6399	0.3240	1.0000	0.0746	0.1095	0.2771
ORP_y	0.0000	0.0001	0.0000	0.0378	0.3459	0.0006	1.0000	0.0000	0.2059
Temp_y	0.0000	0.0000	0.0000	0.1915	0.0019	0.1557	0.0000	1.0000	0.0000
TEMP_y	0.0030	0.6440	0.0507	0.4323	0.0156	0.3137	0.1441	0.0000	1.0000

Hình 21: Kiểm định tính nhân quả Granger

Dựa vào luật kiểm định Granger, ta có thể xây dựng mô hình dự báo tách riêng biến DO với biến TN (do có $p = 0.4995$) hay biến TSS và TEMP ($p = 0.5028$).

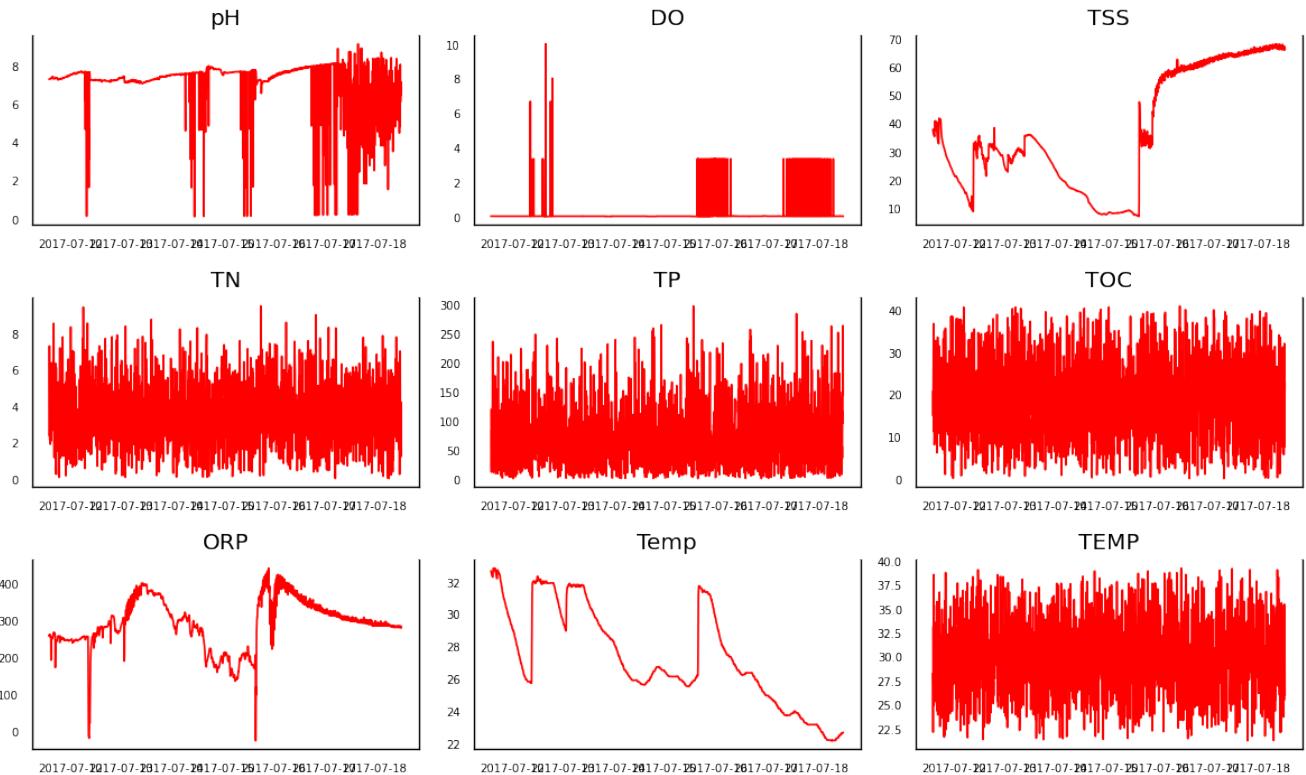
Để kiểm tra tính tương quan dài hạn giữa các biến, ta sử dụng luật kiểm định Johanssen. Với mức ý nghĩa 5%, ta thấy các biến có tương quan dài hạn với nhau:

Name	::	Test Stat > C(95%)	=>	Signif

pH	::	149121.97 > 179.5199	=>	True
DO	::	97506.1 > 143.6691	=>	True
TSS	::	49692.73 > 111.7797	=>	True
TN	::	18978.25 > 83.9383	=>	True
TP	::	7548.58 > 60.0627	=>	True
TOC	::	2996.11 > 40.1749	=>	True
ORP	::	1338.1 > 24.2761	=>	True
Temp	::	543.16 > 12.3212	=>	True
TEMP	::	7.71 > 4.1296	=>	True

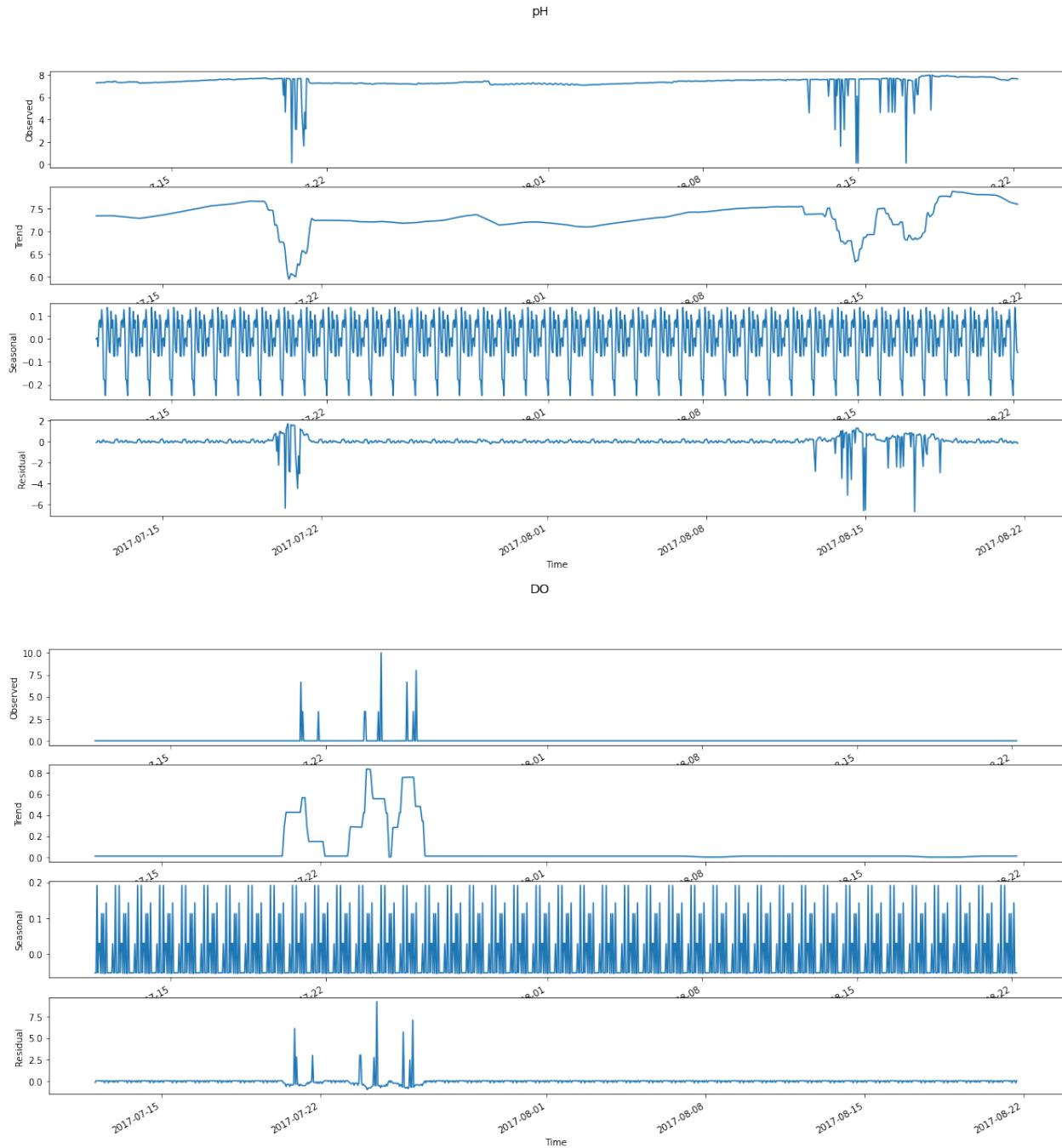
Hình 22: Kiểm định tính Cointegration

Dựa vào 2 luật kiểm định trên, ta rút ra kết luận nên xây dựng mô hình dự báo đồng thời 9 biến chuỗi thời gian. Tiến hành vẽ đồ thị trực quan hóa dữ liệu với 2000 điểm dữ liệu đầu tiên, ta có:



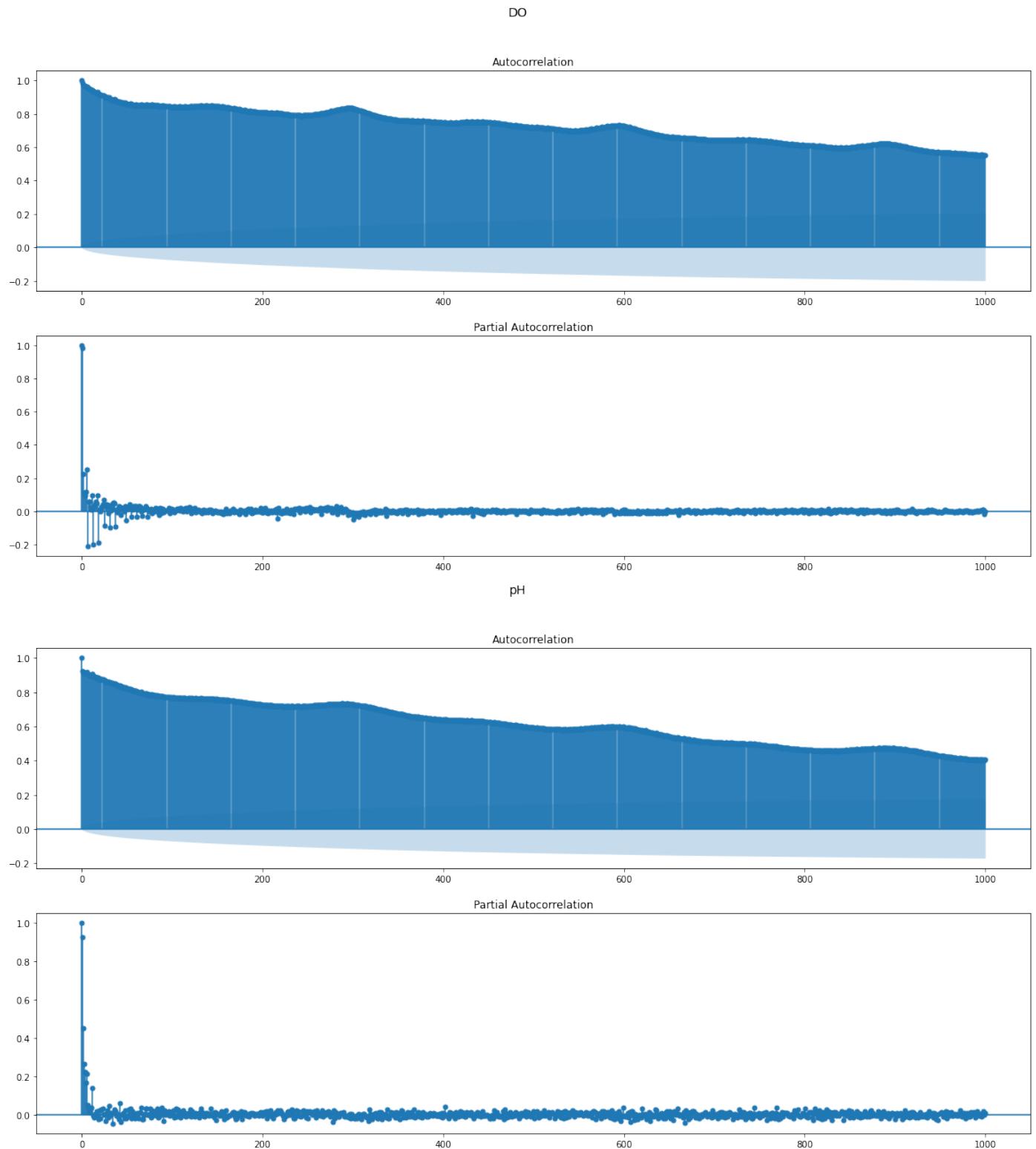
Hình 23: Hình ảnh dữ liệu

Dựa vào đồ thị trên ta có thể thấy tính đa mùa rõ ràng ở các trường TN, TP, TOC, TEMP. Sử dụng phương pháp time series decomposition để tách trend, seasonal ta thu được kết quả của trường DO, pH như sau:



Hình 24: Tách trend, seasonal của biến pH và DO

Ta thấy rằng các biến đều có tính mùa rất mạnh. Tiến hành vẽ hàm ACF, PACF của các biến, ta thu được kết quả như sau:



Hình 25: ACF và PACF của DO, pH

Lấy ví dụ hai biến pH và DO, ta thấy hàm ACF có xu hướng giảm dần và hàm PACF

có xu hướng xấp xỉ 0 sau một cỡ lag nhất định nên các mô hình thống kê cơ bản như VAR hay VARMA có thể được dùng để dự báo.

Ta sẽ xây dựng mô hình dự báo 10 bước tiếp theo dựa vào 20 điểm dữ liệu. Tiến hành xây dựng bộ dữ liệu dạng chuỗi với tập Train gồm 339197 dòng đầu tiên, tập Validation gồm 100 dòng tiếp và tập Test để đánh giá gồm 100 dòng cuối cùng. Tiến hành tính các đặc trưng kỳ vọng, phương sai, độ lệch chuẩn, ... trên tập Test thu được kết quả như sau:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
count	100.000000	100.00000	100.000000	100.000000	100.000000	100.000000	100.0000	100.000000	100.000000
mean	12.526600	8.22470	154.238292	3.738409	72.207370	19.042988	386.3854	25.576500	29.975073
std	0.185021	0.06469	104.641409	1.881318	57.148063	9.737441	1.7411	0.732801	4.052737
min	12.240000	8.14000	2.999879	0.050655	0.362308	1.105357	382.6900	24.200000	23.003170
25%	12.385000	8.17000	66.034878	2.351146	32.555134	12.082982	384.8000	24.760000	26.709929
50%	12.485000	8.20000	143.039817	3.782773	56.718114	19.631822	386.8450	25.950000	29.898861
75%	12.680000	8.29000	225.141185	5.134521	86.750603	25.267442	387.8350	26.137500	33.302709
max	12.880000	8.37000	511.772846	9.122997	284.417964	40.357630	388.8100	26.430000	39.204101

Hình 26: Đặc trưng dữ liệu của tập Test

6.3 Kết quả chạy mô hình

Trong phần này, nhóm tác giả sẽ trình bày chi tiết cách xây dựng các mô hình dự báo và áp dụng với bộ dữ liệu được giao. Các chỉ số đánh giá chính gồm RMSE và MAE. Hai metric này được tính như sau:

- Root Mean Squared Error (RMSE)

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

với y_i là giá trị thực sự cần dự đoán, và \hat{y}_i là giá trị mô hình dự đoán, n là kích thước của dữ liệu cần dự đoán.

- Mean Absolute Error (MAE) MAE là một phương pháp đo lường sự khác biệt giữa hai biến liên tục. Giả sử rằng \hat{y}_i và y_i là hai biến liên tục thể hiện kết quả dự đoán của mô

hình và kết quả thực tế. chúng ta có độ đo MAE được tính theo công thức sau:

$$MAE = \frac{1}{n} \sum_{i=0}^n |y_i - \hat{y}_i|$$

6.3.1 VAR

Ta tiến hành xây dựng mô hình VAR với order = 20 tức mô hình có dạng:

$$\mathbf{X}_t = c + A_1 \mathbf{X}_{t-1} + A_2 \mathbf{X}_{t-2} + \dots + A_{20} \mathbf{X}_{t-20} + \mathbf{W}_t$$

Sau khi dùng phương pháp OLS để tìm bộ hệ số tối ưu, ta có:

=====				
Model:		VAR		
Method:		OLS		
Date:		Mon, 06, Mar, 2023		
Time:		15:07:46		

No. of Equations:	9.00000	BIC:	31.2200	
Nobs:	339276.	HQIC:	31.1833	
Log likelihood:	-9.61843e+06	FPE:	3.43776e+13	
AIC:	31.1684	Det(Omega_mle):	3.42130e+13	

Results for equation pH				
=====				
	coefficient	std. error	t-stat	prob
const	0.059509	0.028284	2.104	0.035
L1.pH	0.192829	0.001749	110.239	0.000
L1.DO	0.004624	0.001325	3.490	0.000
L1.TSS	-0.000021	0.000009	-2.451	0.014
L1.TN	-0.001059	0.000428	-2.477	0.013
L1.TP	-0.000016	0.000013	-1.185	0.236
L1.TOC	-0.000140	0.000079	-1.778	0.075
L1.ORP	0.000079	0.000011	7.401	0.000
L1.Temp	-0.001472	0.000691	-2.129	0.033
L1.TEMP	-0.000053	0.000193	-0.272	0.785
L2.pH	0.149516	0.001782	83.888	0.000
L2.DO	-0.005748	0.001463	-3.930	0.000
L2.TSS	-0.000022	0.000009	-2.579	0.010
L2.TN	0.000248	0.000428	0.579	0.563
L2.TP	-0.000017	0.000013	-1.334	0.182

Hình 27: Hệ số mô hình VAR

Tiến hành dự báo 10 bước tiếp theo, ta có đồ thị so sánh kết quả dự báo như sau:



Hình 28: Đồ thị so sánh kết quả

Ta có thể thấy mô hình dự báo khá tốt đáng điệu của chuỗi thời gian ở trường ORP, Temp, DO. Kết quả đánh giá trên metrics RMSE và MAE như sau

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	0.19	0.10	107.89	1.73	59.62	9.40	2.71	0.51	4.03
MAE	0.16	0.07	90.48	1.42	45.28	7.53	2.12	0.36	3.38

Bảng 2: Kết quả chạy mô hình VAR

6.3.2 VARMA

Mô hình VARMA sẽ sử dụng phương pháp MLE để ước lượng tham số cho mô hình do đó khi dùng toàn bộ tập dữ liệu các tham số sẽ khó hội tụ, do đó ta chỉ sử dụng 4000 điểm dữ liệu cuối của tập train để xây dựng bộ tham số cho mô hình VARMA có bậc (2,2). Khi đó, công thức của mô hình có dạng:

$$\mathbf{X}_t = \boldsymbol{\mu} + \sum_{i=1}^2 \boldsymbol{\Phi}_i \mathbf{X}_{t-i} + \sum_{j=1}^2 \boldsymbol{\Theta}_j \mathbf{W}_{t-j} + \mathbf{W}_t$$

Tiến hành dự báo 10 bước bằng mô hình VARMA vừa xây dựng, ta thu được kết quả



Hình 29: Đồ thị dự báo mô hình VARMA

Dựa vào đồ thị có thể cho thấy mô hình dự báo chưa tốt cụ thể ở việc dự báo dâng điệu các chuỗi thời gian. Khi tiến hành tăng bậc của phần AR hoặc MA thì ước lượng MLE

không hội tụ do đó nhóm tác giả chỉ dừng lại ở bậc (2,2). Cụ thể, kết quả đánh giá các metric như sau:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	0.33	0.11	106.36	1.73	58.92	9.38	2.84	0.96	4.03
MAE	0.29	0.08	89.99	1.42	44.70	7.56	2.31	0.69	3.38

Bảng 3: Kết quả chạy mô hình VARMA

6.3.3 XGBoost

Nhận thấy các bản ghi các nhau 5 phút và dữ liệu có tính đa mùa nên nhóm tác giả quyết định tiến hình Feature Engineering bằng cách thêm trường *Date – Time* bằng cách Embedding cột Ngày và Thời gian bản ghi được cập nhật. Xây dựng mô hình XGBoost dự báo chuỗi thời gian dưới dạng mô hình Regression với các trường dữ liệu là các chuỗi thời gian và cột *Date – Time*. Dữ liệu X dùng để làm feature và dữ liệu y dùng để làm nhãn có dạng

Time	y_lag_1	y_lag_2	y_lag_3	y_lag_4	y_lag_5	y_lag_6	y_lag_7	y_lag_8	y_lag_9	y_lag_10	...
2017-07-11 14:15:00	37.180000	37.720000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
2017-07-11 14:15:00	7.284571	2.382600	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
2017-07-11 14:15:00	23.182876	118.301766	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
2017-07-11 14:15:00	14.900992	20.640303	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
2017-07-11 14:15:00	260.500000	257.620000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Time Industries	y_step_1	y_step_2	y_step_3	y_step_4	y_step_5	y_step_6	y_step_7	y_step_8	y_step_9
2017-07-11 14:05:00	pH	7.290000	7.290000	7.290000	7.300000	7.310000	7.310000	7.310000	7.310000	7.310000	...
	DO	0.010000	0.010000	0.010000	0.010000	0.010000	0.010000	0.010000	0.010000	0.010000	...
	TSS	37.720000	37.180000	36.640000	36.250000	36.080000	35.830000	35.630000	35.520000	35.350000	...
	TN	2.382600	7.284571	4.668972	2.146710	2.934312	5.594663	1.891900	3.548026	3.075528	...
	TP	118.301766	23.182876	11.363099	83.474613	12.725587	79.639509	92.438369	26.135042	130.620427	...

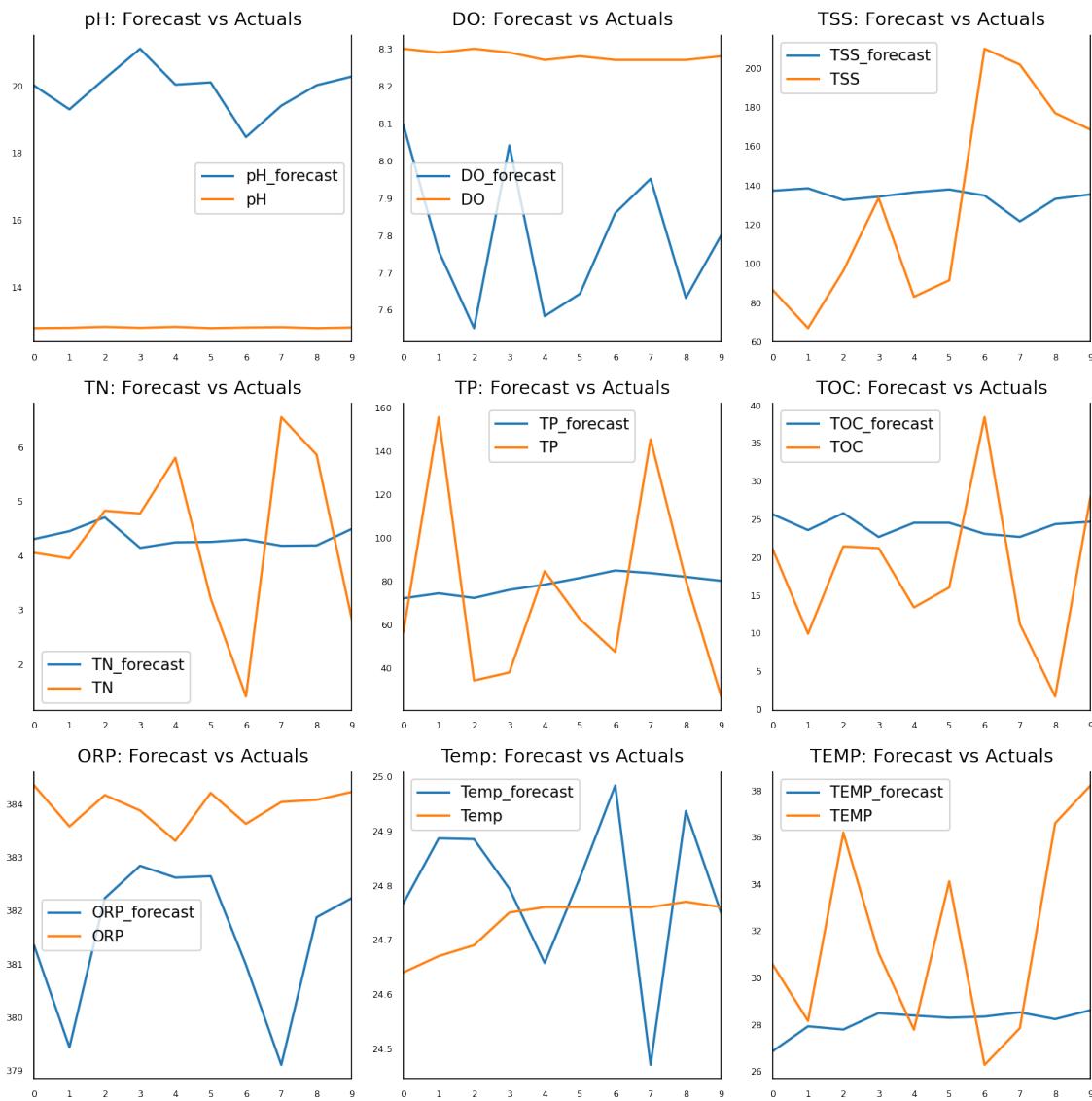
Hình 30: Dạng dữ liệu đầu vào

Chiều của X có dạng $nsample * timestep * nfeature$ tương ứng với $timestep = 20$ dùng 20 điểm dữ liệu và $nfeature = 9$. Chiều của y có dạng $nsample * targetstep * nfeature$ với $targetstep = 10$ tức ta đi dự báo 10 bước tiếp theo. Sau khi huấn luyện mô hình, ta được kết quả

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	$4.14 \pm .01$	$1.10 \pm .01$	$114.03 \pm .01$	$2.12 \pm .01$	$59.54 \pm .01$	$10.38 \pm .01$	$83.57 \pm .01$	$4.13 \pm .01$	$6.58 \pm .01$
MAE	$3.78 \pm .01$	$0.72 \pm .01$	$93.48 \pm .01$	$1.70 \pm .01$	$46.95 \pm .01$	$8.16 \pm .01$	$33.29 \pm .01$	$1.98 \pm .01$	$4.63 \pm .01$

Bảng 4: Kết quả chạy mô hình XGBoost

So sánh đáng điệu đồ thị của kết quả dự đoán và trường dữ liệu, ta có:



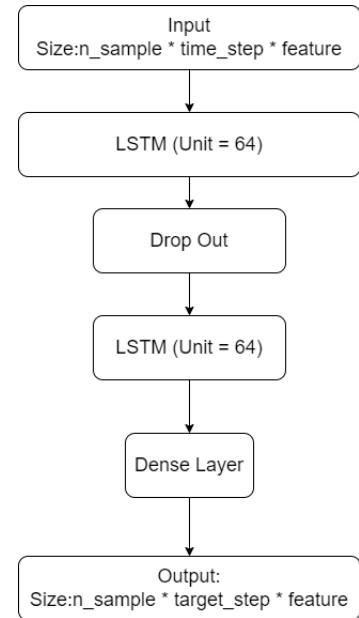
Hình 31: Đồ thị dự báo của mô hình XGBoost

Mô hình XGBoost dự báo khá tốt ở biến TOC, ORP về đáng điệu.

6.3.4 LSTM

Tiến hành xây dựng mô hình LSTM gồm 2 lớp LSTM xếp chồng lên nhau như sau:

Model: "sequential_1"		
Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 20, 64)	18944
dropout_2 (Dropout)	(None, 20, 64)	0
lstm_4 (LSTM)	(None, 32)	12416
repeat_vector_1 (RepeatVector or)	(None, 10, 32)	0
time_distributed_1 (TimeDistributed)	(None, 10, 9)	297
<hr/>		
Total params: 31,657		
Trainable params: 31,657		
Non-trainable params: 0		



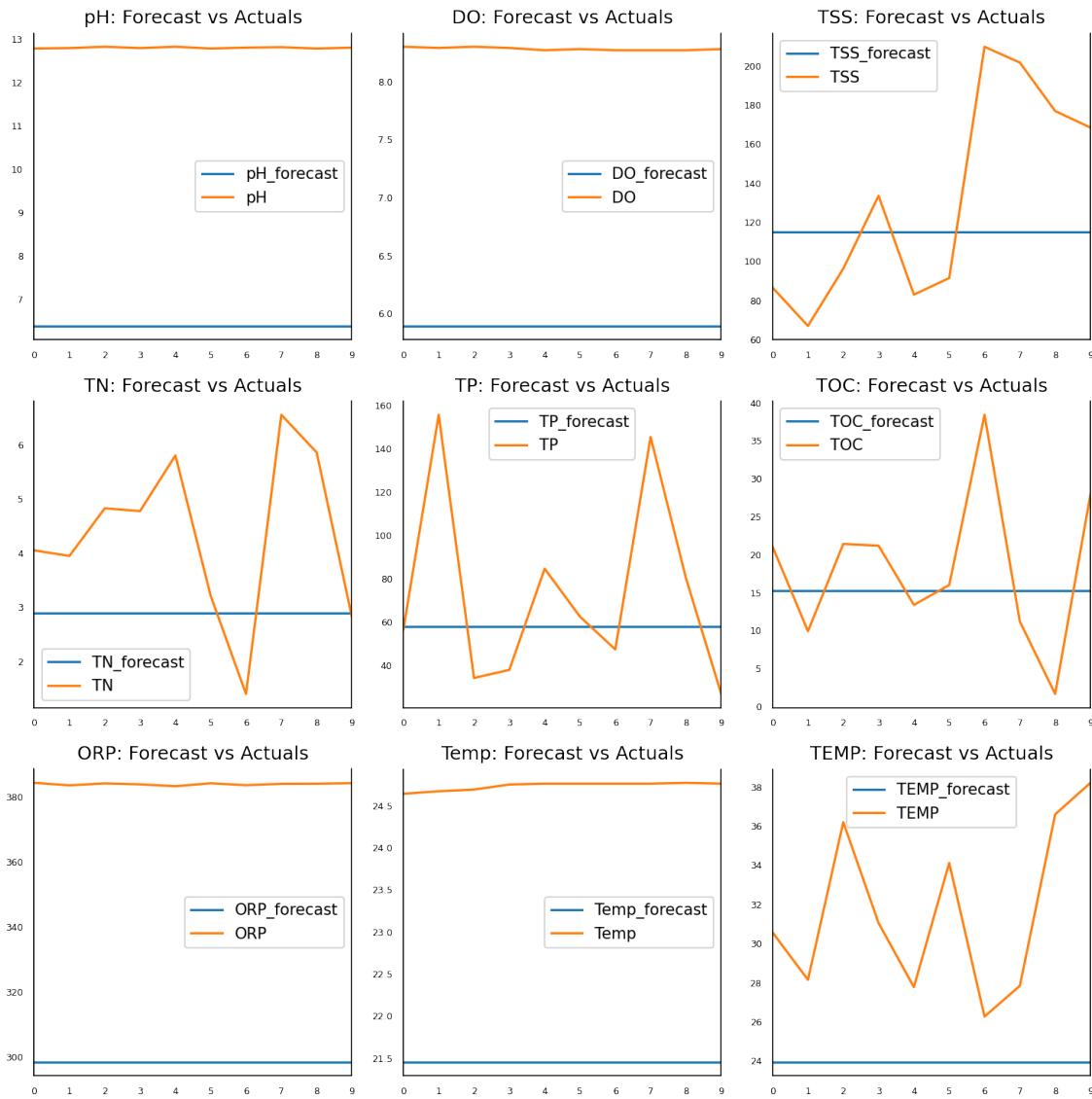
Hình 32: Mô hình LSTM

Mô hình sẽ encode dữ liệu qua 2 lớp LSTM với 64 và 32 đơn vị rồi cuối cùng cho qua 1 lớp Dense Layer để tiến hành tính loss function theo MSE. Kết quả đánh giá trên các metric thu được:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	6.55 ± .01	2.88 ± .01	111.26 ± .01	1.98 ± .01	61.28 ± .01	10.32 ± .01	75.44 ± .01	5.36 ± .01	9.03 ± .01
MAE	6.53 ± .01	2.85 ± .01	89.33 ± .01	1.67 ± .01	41.75 ± .01	8.27 ± .01	72.33 ± .01	5.01 ± .01	7.77 ± .01

Bảng 5: Kết quả chạy mô hình LSTM

Dáng điệu của đồ thị dự báo và dữ liệu thật như sau:



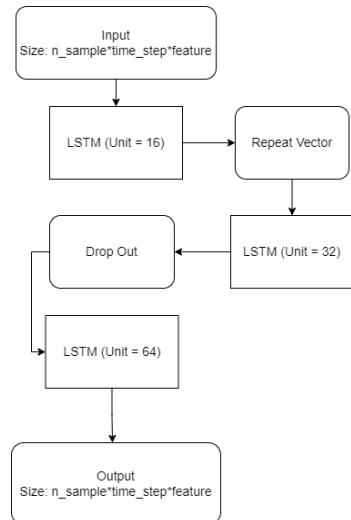
Hình 33: Đồ thị dự báo LSTM

Mô hình không học được đáng điệu của dữ liệu mà chỉ đưa ra giá trị trung bình cho các trường dữ liệu.

6.3.5 LSTM AutoEncoder

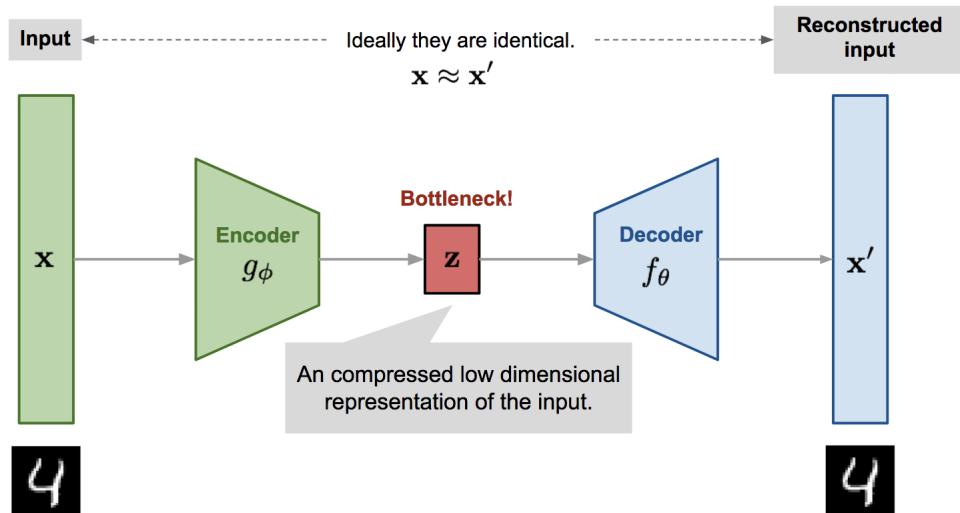
Với ý tưởng dùng các lớp LSTM để Encode thông tin của chuỗi thời gian và Decode lại thông tin để tiến hành so sánh với chuỗi thực, nhóm tác giả xây dựng mô hình LSTM AutoEncoder như sau: Ta sẽ tiến hành Encode thông tin dựa vào một lớp LSTM 16 đơn vị và Decode thông tin dựa vào 2 lớp LSTM 32 và 64 đơn vị. Giữa các khối là các lớp Drop

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 20, 9)]	0
lstm (LSTM)	(None, 16)	1664
repeat_vector (RepeatVector (None, 10, 16))		0
lstm_1 (LSTM)	(None, 10, 32)	6272
dropout (Dropout)	(None, 10, 32)	0
lstm_2 (LSTM)	(None, 10, 64)	24832
time_distributed (TimeDistr ibuted)	(None, 10, 9)	585



Hình 34: Mô hình LSTM AutoEncoder

Out giúp ngăn chặn Overfitting. Ý tưởng này dựa trên việc coi chuỗi thời gian cũng là một chuỗi số tương tự với ảnh và dùng mô hình AutoEncoder của Thị giác máy tính để xây dựng mô hình.



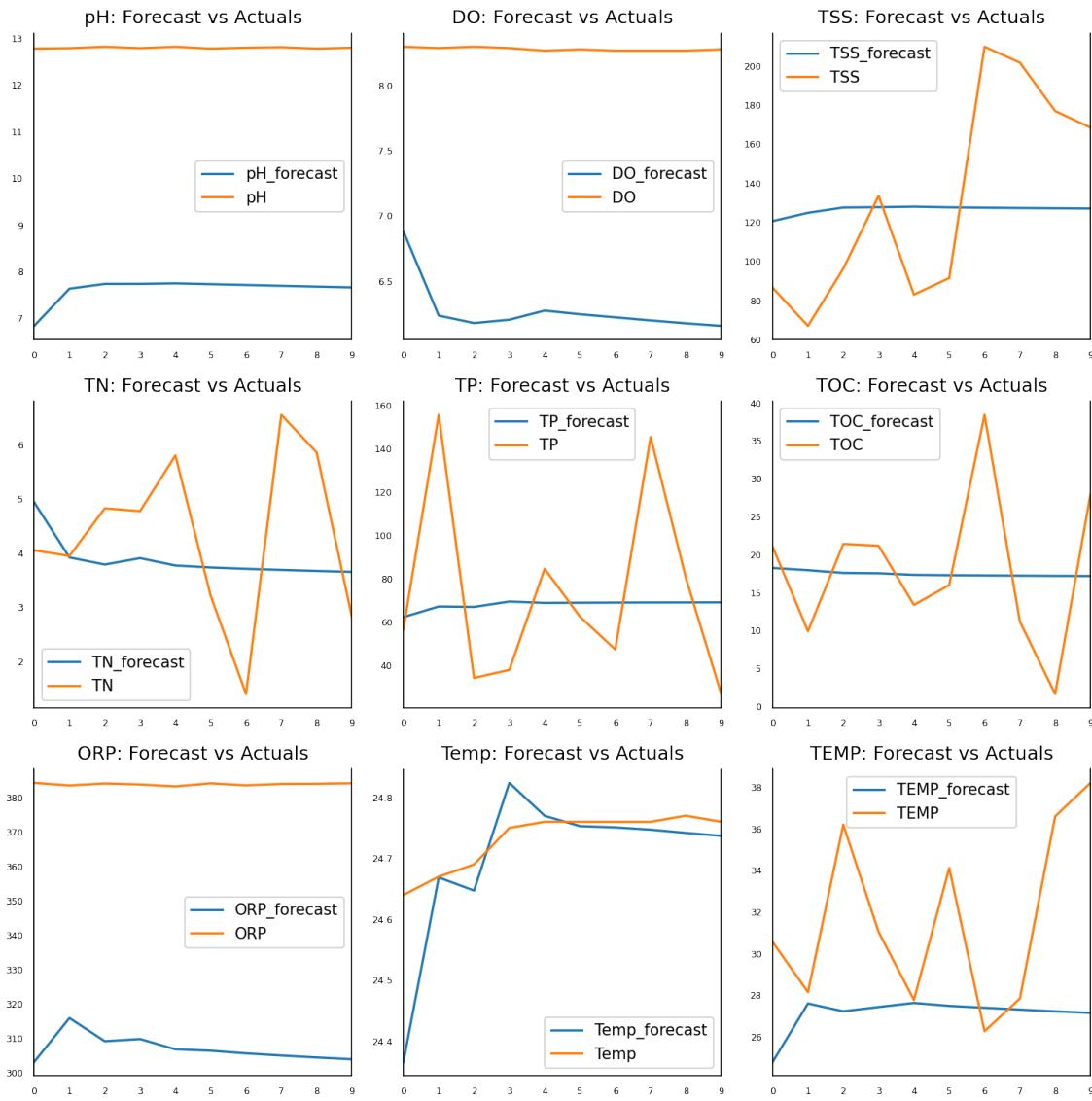
Hình 35: Ý tưởng của mô hình AutoEncoder

Khi đó, kết quả thu được như sau:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	$5.10 \pm .01$	$1.96 \pm .01$	$108.14 \pm .01$	$1.82 \pm .01$	$59.02 \pm .01$	$9.42 \pm .01$	$81.54 \pm .01$	$1.17 \pm .01$	$5.24 \pm .01$
MAE	$5.09 \pm .01$	$1.94 \pm .01$	$89.16 \pm .01$	$1.48 \pm .01$	$43.08 \pm .01$	$7.60 \pm .01$	$81.00 \pm .01$	$0.96 \pm .01$	$4.18 \pm .01$

Bảng 6: Kết quả LSTM AutoEncoder

Ta thấy các chỉ số đánh giá tương đối tốt nhưng chất lượng dự báo cho biến ORP tệ hơn hẳn các mô hình khác. Sau đây là đáng điệu chuỗi thời gian dự báo:



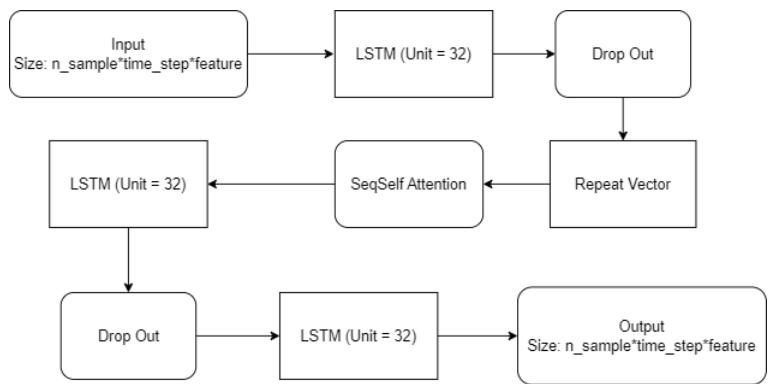
Hình 36: Đồ thị dự báo của mô hình LSTM AutoEncoder

6.3.6 LSTM Autoencoder kết hợp kỹ thuật Attention

Dựa trên ý tưởng của bài báo [1], nhóm tác giả tự xây dựng bộ Encode bằng 1 lớp LSTM 32 Unit để Encode thông tin của chuỗi thời gian đầu vào. Sau đó sử dụng kỹ thuật Self Attention để đánh trọng số chú ý vào các phần trong chuỗi thời gian. Sau đó qua bộ Decoder gồm 2 lớp LSTM xếp chồng lên nhau và đầu ra cho qua một lớp Dense để đưa ra giá trị dự báo. Đồng thời giữa các lớp sẽ có lớp Drop Out để tránh overfitting.

Model: "sequential"		
Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 32)	5376
dropout (Dropout)	(None, 32)	0
repeat_vector (RepeatVector (None, 10, 32))		0
seq_self_attention (SeqSelf Attention)	(None, 10, 32)	2113
lstm_1 (LSTM)	(None, 10, 32)	8320
dropout_1 (Dropout)	(None, 10, 32)	0
lstm_2 (LSTM)	(None, 10, 64)	24832
time_distributed (TimeDistr ibuted)	(None, 10, 9)	585

Total params: 41,226
Trainable params: 41,226
Non-trainable params: 0



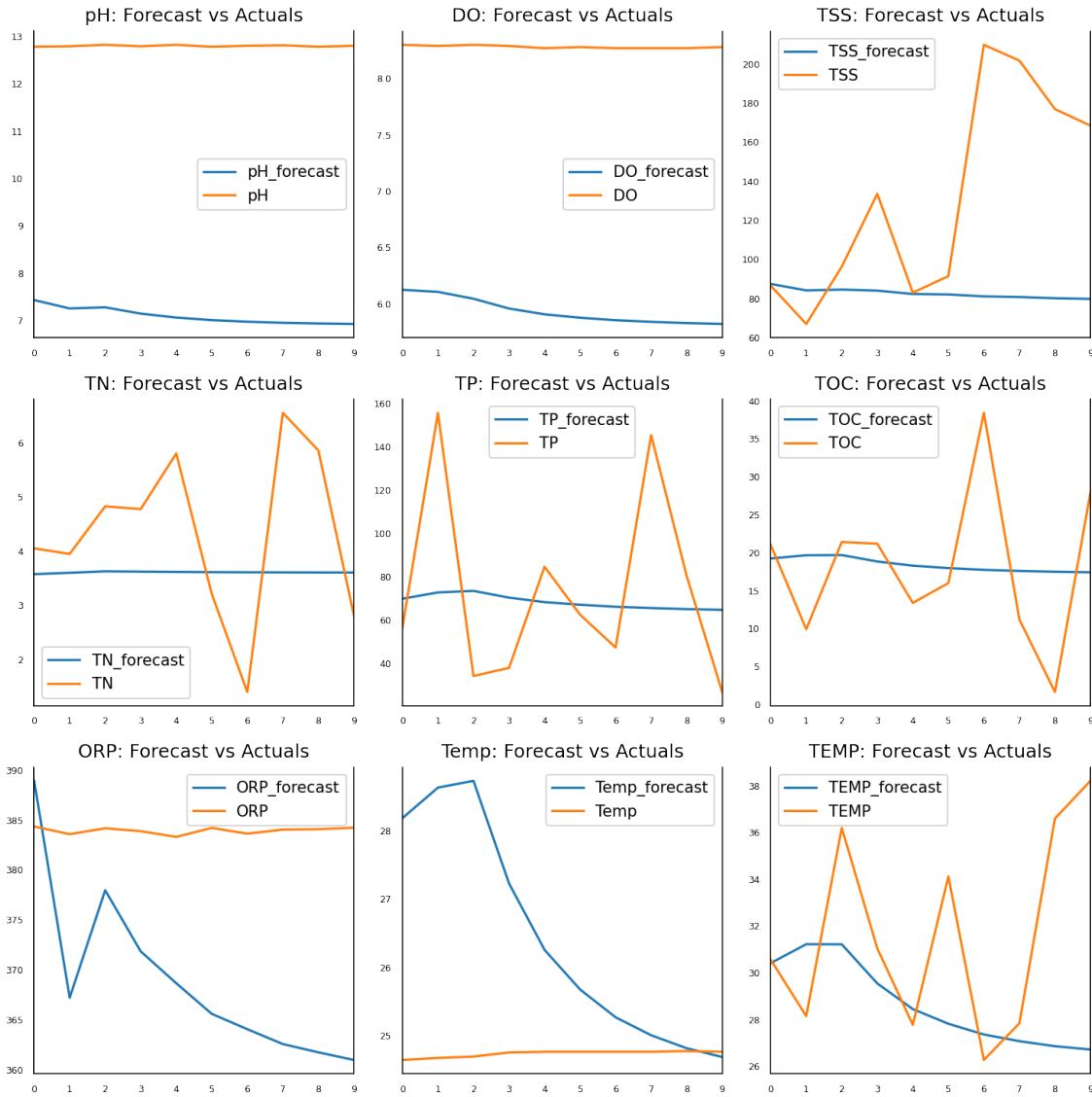
Hình 37: Mô hình LSTM Autoencoder kết hợp kỹ thuật Attention

Khi đó, kết quả thu được như sau:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	$5.27 \pm .01$	$2.01 \pm .01$	$113.43 \pm .01$	$1.73 \pm .01$	$59.13 \pm .01$	$9.38 \pm .01$	$53.14 \pm .01$	$2.36 \pm .01$	$4.61 \pm .01$
MAE	$5.26 \pm .01$	$2.00 \pm .01$	$90.77 \pm .01$	$1.41 \pm .01$	$44.81 \pm .01$	$7.49 \pm .01$	$47.19 \pm .01$	$1.98 \pm .01$	$3.76 \pm .01$

Bảng 7: Kết quả chạy mô hình LSTM AutoEncoder kết hợp kỹ thuật Attention

Ta có thể thấy các thông số đánh giá tương đối tốt. Cụ thể, hình ảnh đồ thị dự báo chuỗi thời gian như sau:

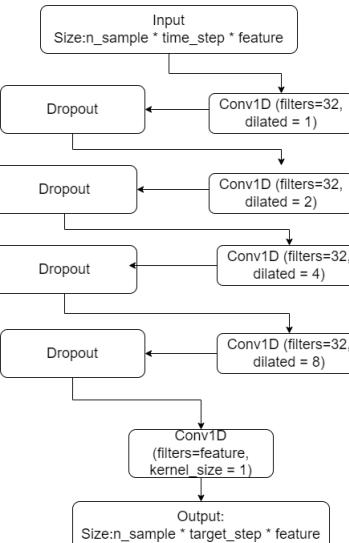


Hình 38: Đồ thị dự báo của mô hình LSTM Autoencoder Attention

6.3.7 Mô hình Dilated Convolution

Dựa vào ý tưởng của [28], ta coi chuỗi thời gian là dữ liệu dạng chuỗi tương tự ảnh. Để trích xuất được đặc trưng trong các phần của chuỗi thời gian ta sẽ sử dụng mạng Dilated Convolution là biến thể của mạng tích chập giúp trích xuất thông tin trên một khoảng rộng hơn.

Layer (type)	Output Shape	Param #
<hr/>		
input_2 (InputLayer)	[None, 20, 9]	0
conv1d_4 (Conv1D)	(None, 20, 32)	896
dropout_4 (Dropout)	(None, 20, 32)	0
conv1d_5 (Conv1D)	(None, 20, 32)	3104
dropout_5 (Dropout)	(None, 20, 32)	0
conv1d_6 (Conv1D)	(None, 20, 32)	3104
dropout_6 (Dropout)	(None, 20, 32)	0
conv1d_7 (Conv1D)	(None, 10, 9)	3177
conv1d_8 (Conv1D)	(None, 10, 9)	90
<hr/>		



Hình 39: Mô hình Dilated Convolution

Khi đó, kết quả thu được như sau:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	4.82 ± .01	1.82 ± .01	111.29 ± .01	1.78 ± .01	59.12 ± .01	9.36 ± .01	113.89 ± .01	0.89 ± .01	4.19 ± .01
MAE	4.82 ± .01	1.82 ± .01	90.50 ± .01	1.48 ± .01	42.77 ± .01	7.54 ± .01	113.82 ± .01	0.75 ± .01	3.48 ± .01

Bảng 8: Kết quả mô hình Dilated Convolution

Ta nhận thấy mô hình chỉ dự báo tốt trên một số biến như TN, TP, TOC. Đồ thị chuỗi dự báo như sau:



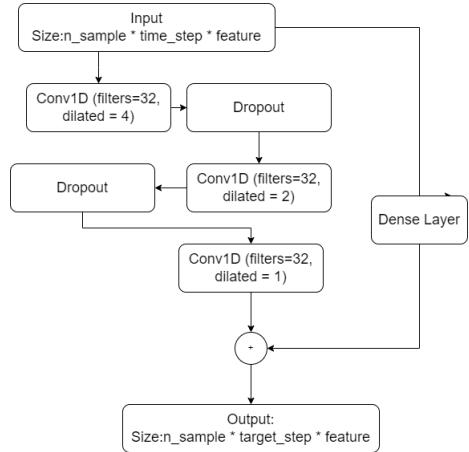
Hình 40: Đồ thị dự báo của mô hình

6.3.8 Dilated Convolution kết hợp khử trend

Để khử trend cho dữ liệu, nhóm tác giả đề xuất sử dụng một lớp Dense Layer với activation linear để ước lượng trend của dữ liệu. Đầu ra của lớp Dense Layer này sẽ được nối với đầu ra của mạng Dilated Convolution tương tự như phần 6.3.7.

Khi đó, kết quả thu được như sau:

Layer (type)	Output Shape	Param #	Connected to
input_3 (InputLayer)	[(None, 20, 9)]	0	[]
conv1d_6 (Conv1D)	(None, 20, 32)	896	['input_3[0][0]']
dropout_5 (Dropout)	(None, 20, 32)	0	['conv1d_6[0][0]']
conv1d_7 (Conv1D)	(None, 20, 32)	3104	['dropout_5[0][0]']
dropout_6 (Dropout)	(None, 20, 32)	0	['conv1d_7[0][0]']
conv1d_8 (Conv1D)	(None, 20, 32)	3104	['dropout_6[0][0]']
dropout_7 (Dropout)	(None, 20, 32)	0	['conv1d_8[0][0]']
reshape (Reshape)	(None, 180)	0	['input_3[0][0]']
conv1d_9 (Conv1D)	(None, 10, 9)	3177	['dropout_7[0][0]']
dense_1 (Dense)	(None, 1)	181	['reshape[0][0]']
add (Add)	(None, 10, 9)	0	['conv1d_9[0][0]', 'dense_1[0][0]']

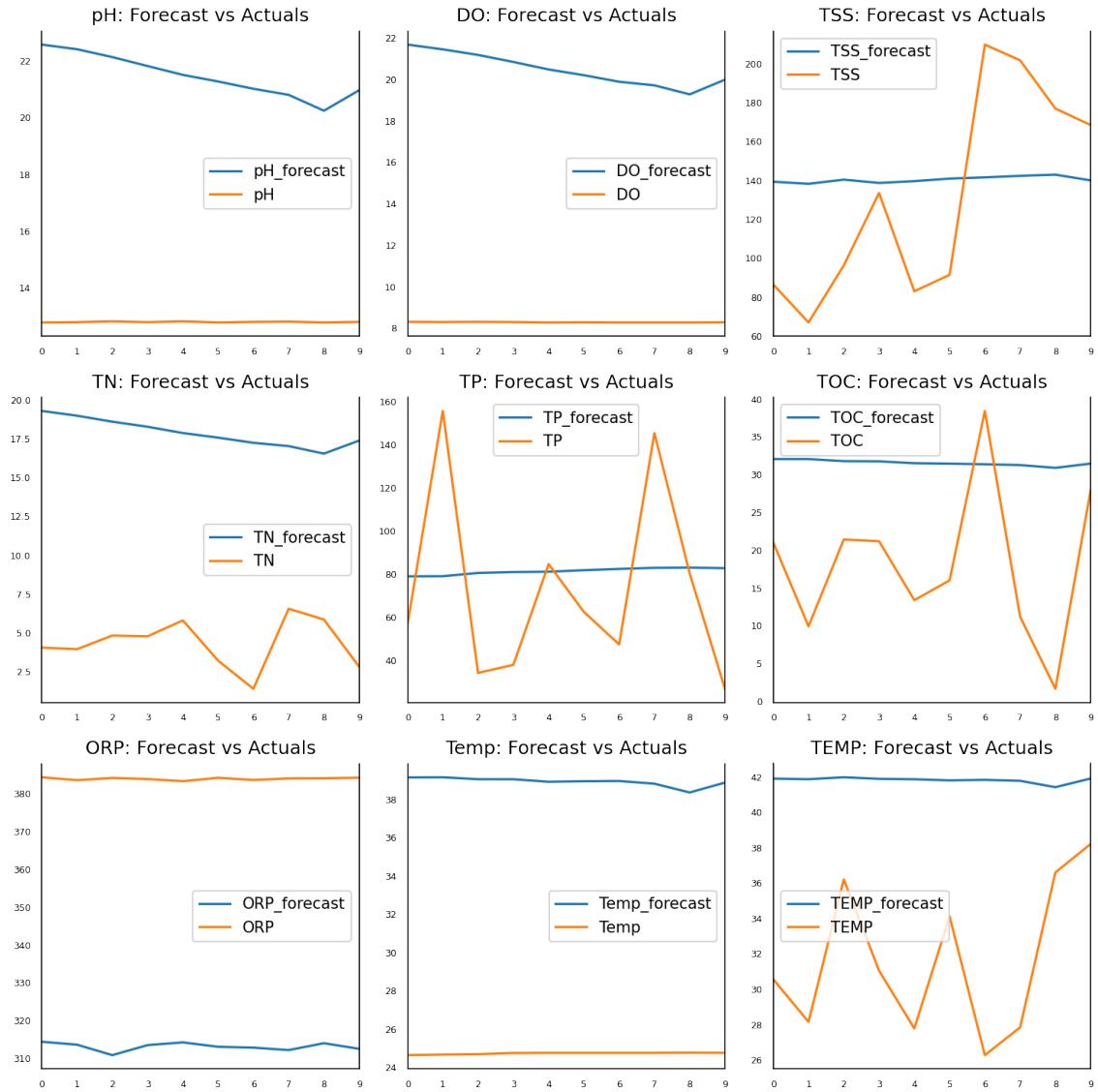


Hình 41: Mô hình Dilated Convolution kết hợp khử trend

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	6.08 ± .01	9.62 ± .01	109.84 ± .01	11.88 ± .01	59.30 ± .01	13.46 ± .01	71.42 ± .01	9.32 ± .01	8.47 ± .01
MAE	5.96 ± .01	9.55 ± .01	89.10 ± .01	11.71 ± .01	45.60 ± .01	11.31 ± .01	71.35 ± .01	9.20 ± .01	7.41 ± .01

Bảng 9: Kết quả chạy mô hình Dilated Convolution kết hợp khử trend

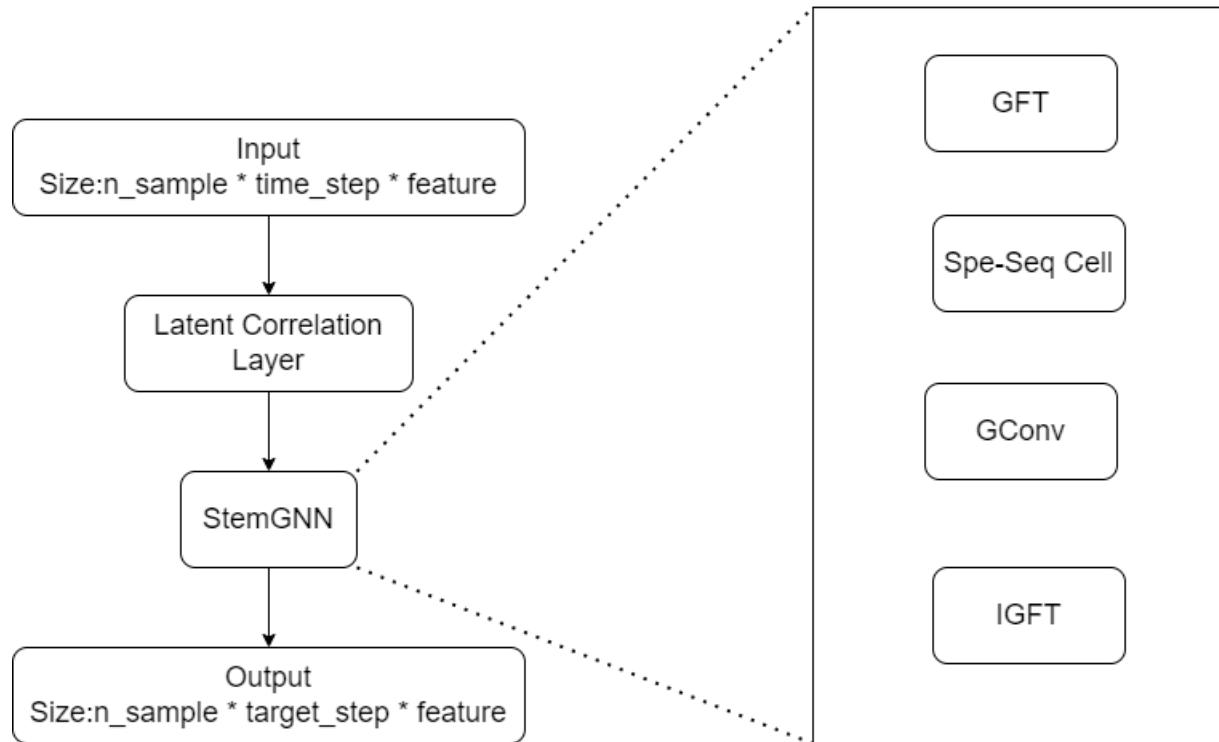
Từ kết quả thực nghiệm ta thấy việc khử trend không giúp cải thiện khả năng dự báo của mô hình quá nhiều.



Hình 42: Đồ thị dự báo của mô hình Dilated Convolution kết hợp Khử trend

6.3.9 StemGNN

Xây dựng mô hình tương tự như trong paper [9] nhưng để tăng tốc độ huấn luyện mô hình nhóm tác giả chỉ sử dụng Data gồm 4200 dòng cuối cùng. Tập train lúc này gồm 4000 dòng dữ liệu, tập Validation và Tập Test giữ nguyên.



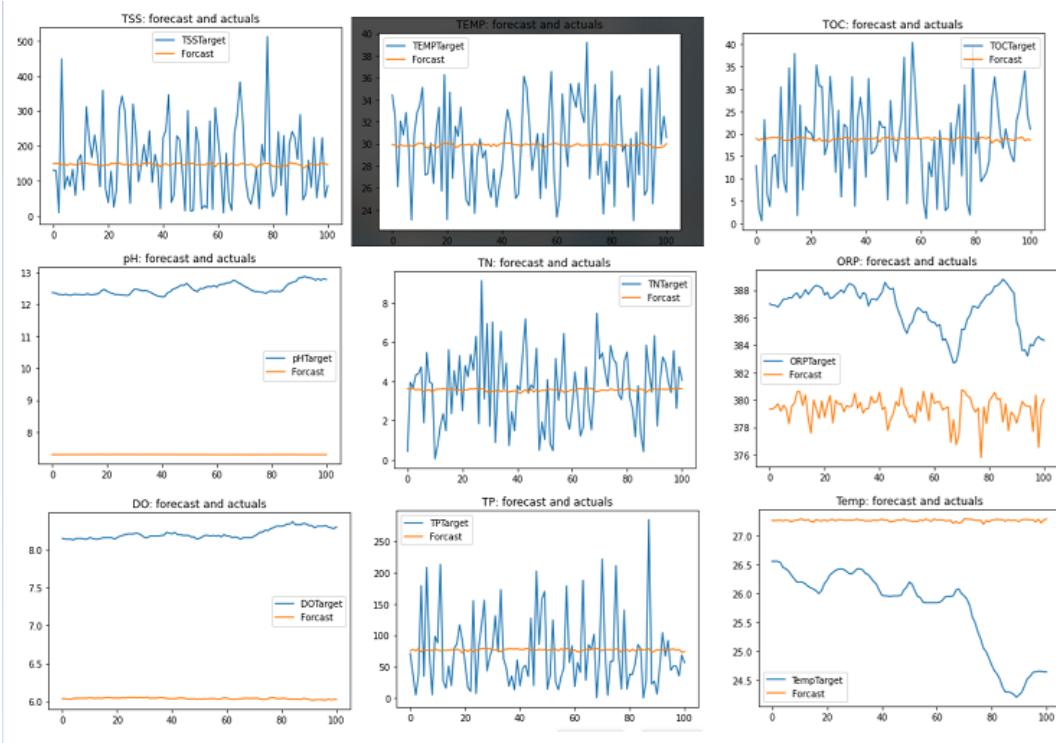
Hình 43: Mô hình StemGNN

Tiến hành huấn luyện mô hình ta thu được kết quả:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	$5.86 \pm .01$	$2.04 \pm .01$	$1.09 \pm .01$	$1.08 \pm .01$	$1.04 \pm .01$	$1.07 \pm .01$	$0.05 \pm .01$	$0.86 \pm .01$	$1.02 \pm .01$
MAE	$5.86 \pm .01$	$2.04 \pm .01$	$0.89 \pm .01$	$0.88 \pm .01$	$0.81 \pm .01$	$0.88 \pm .01$	$0.05 \pm .01$	$0.78 \pm .01$	$0.86 \pm .01$

Bảng 10: Kết quả chạy mô hình StemGNN

Có thể thấy kết quả dự báo của mô hình StemGNN cho kết quả tốt nhất. Hình ảnh chuỗi thời gian dự báo được mô tả như sau:



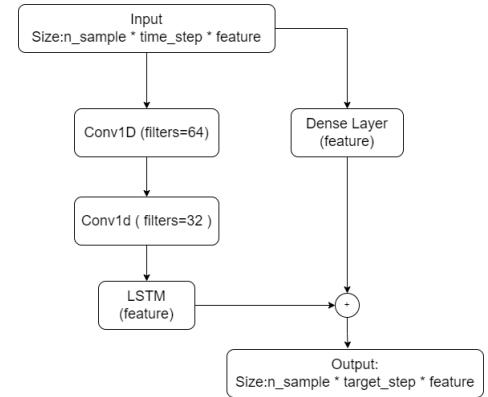
Hình 44: Đồ thị dự báo của mô hình StemGNN

Tuy không bắt chước được dáng điệu của chuỗi thời gian nhưng kết quả đánh giá theo 2 metric cho kết quả tốt nhất.

6.3.10 Convolution kết hợp LSTM và khử trend

Dựa trên ý tưởng của bài báo [7], ta xây dựng mạng Neural gồm các lớp Convolution để trích chọn đặc trưng của chuỗi thời gian đầu vào và cho qua mạng LSTM để dự báo. Đồng thời dùng một lớp Dense Layer để tách trend của dữ liệu. Cụ thể mô hình được xây dựng như sau:

Layer (type)	Output Shape	Param #	Connected to
input_4 (InputLayer)	[None, 20, 9]	0	[]
conv1d_10 (Conv1D)	(None, 20, 64)	3520	['input_4[0][0]']
dropout_8 (Dropout)	(None, 20, 64)	0	['conv1d_10[0][0]']
conv1d_11 (Conv1D)	(None, 20, 32)	12320	['dropout_8[0][0]']
dropout_9 (Dropout)	(None, 20, 32)	0	['conv1d_11[0][0]']
conv1d_12 (Conv1D)	(None, 10, 9)	3177	['dropout_9[0][0]']
reshape_1 (Reshape)	(None, 180)	0	['input_4[0][0]']
lstm_3 (LSTM)	(None, 10, 9)	684	['conv1d_12[0][0]']
dense_2 (Dense)	(None, 1)	181	['reshape_1[0][0]']
add_1 (Add)	(None, 10, 9)	0	['lstm_3[0][0]', 'dense_2[0][0]']



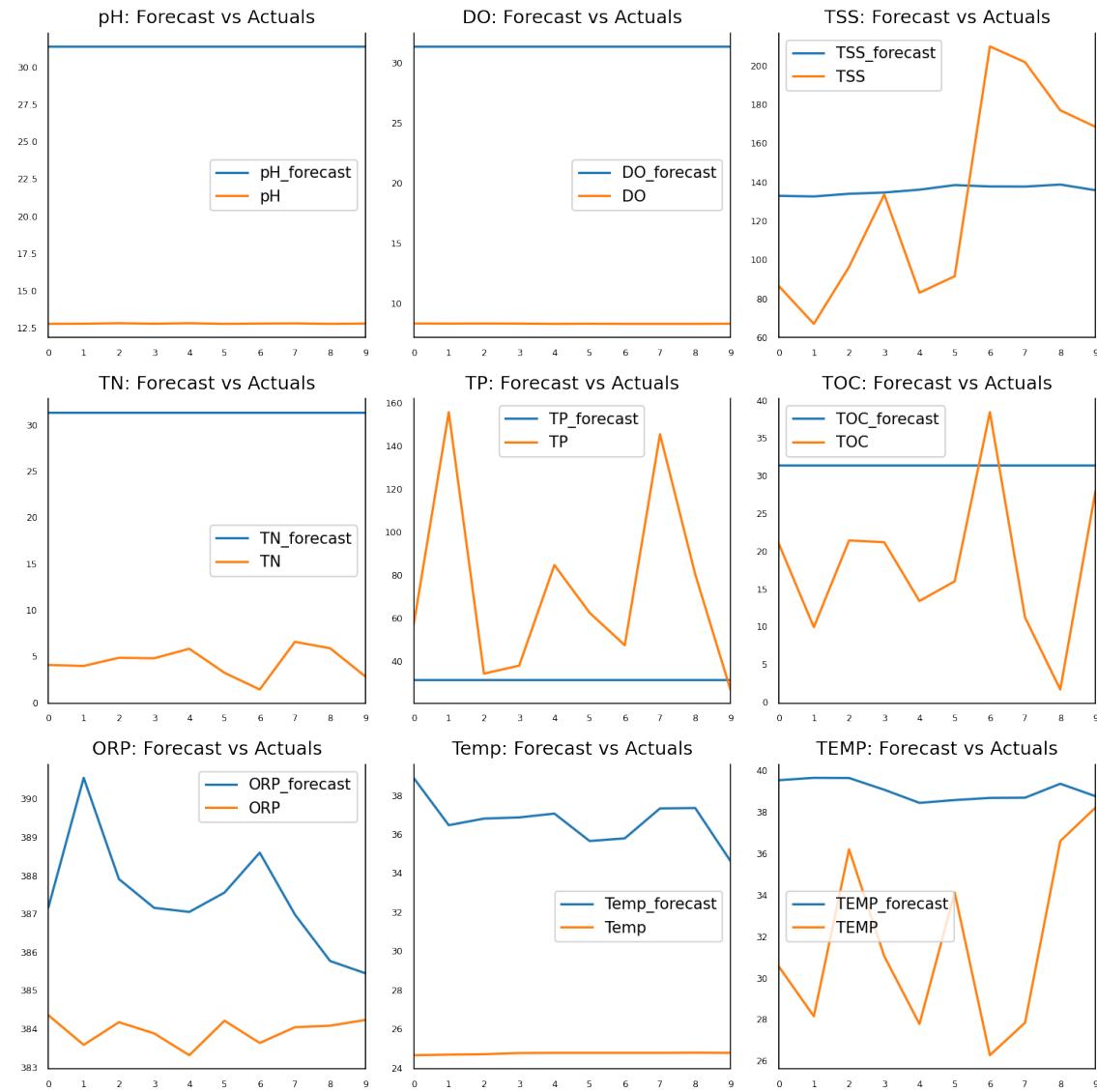
Hình 45: Mô hình Convolution kết hợp LSTM và khử trend

Khi đó, thu được kết quả như sau:

	pH	DO	TSS	TN	TP	TOC	ORP	Temp	TEMP
RMSE	$19.64 \pm .01$	$23.96 \pm .01$	$106.46 \pm .01$	$28.57 \pm .01$	$70.32 \pm .01$	$16.54 \pm .01$	$5.81 \pm .01$	$12.75 \pm .01$	$11.34 \pm .01$
MAE	$19.60 \pm .01$	$23.93 \pm .01$	$88.96 \pm .01$	$28.48 \pm .01$	$46.37 \pm .01$	$14.32 \pm .01$	$4.76 \pm .01$	$12.62 \pm .01$	$10.36 \pm .01$

Bảng 11: Kết quả chạy mô hình Convolution kết hợp LSTM và khử trend

Mô hình chỉ dự báo tốt ở biến ORP còn lại các biến khác cho kết quả dự báo kém hơn so với các mô hình khác trong báo cáo. Dáng điệu chuỗi thời gian dự báo như sau:



Hình 46: Đồ thị dự báo của mô hình Conv + LSTM + Trend Decomposition

6.3.11 Nhận xét

Dựa vào việc xây dựng các mô hình dự báo dựa trên các mạng Học sâu, Học máy, Thống kê và Hybrid, nhóm tác giả rút ra những nhận xét sau:

- Mô hình StemGNN cho kết quả dự báo tốt nhất theo chỉ số đánh giá RMSE và MAE.
- Mô hình thống kê truyền thống VAR và VARMA cho kết quả dự báo tốt dựa trên chỉ số đánh giá RMSE, MAE.

- Cách tiếp cận mô hình dưới dạng một bài toán hồi quy và dùng mô hình Học máy XGBoost để dự báo cho kết quả tương đối tốt.
- Việc khử khuynh cho dữ liệu không đem lại nhiều hiệu quả trong việc nâng cao hiệu suất dự báo.
- Các mô hình dựa trên lớp LSTM cho kết quả tương đồng với các mô hình xây dựng dựa trên lớp Convolution.

7 Kết luận

Trong báo cáo trên, nhóm tác giả đã đạt được một số kết quả sau:

- Trình bày được khái quát lý thuyết của chuỗi thời gian, các mô hình thống kê dự báo chuỗi thời gian, các mô hình Học sâu dự báo chuỗi thời gian, các mô hình Học máy dự báo chuỗi thời gian và mô hình Hybrid dự báo chuỗi thời gian.
- Tiến hành khảo sát các kỹ thuật xây dựng mô hình dự báo chuỗi thời gian.
- Tiến xử lý và khai phá dữ liệu được giao.
- Tự xây dựng lại các mô hình dự báo dựa vào ý tưởng của các bài báo. Đạt được kết quả dự báo 10 bước tương đối tốt.

Tài liệu tham khảo

- [1] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017.
- [2] MathWorks. Find periodicity using frequency analysis, Accessed March 6, 2023.
- [3] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. Springer, 2002.
- [4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [6] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [7] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.
- [8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [9] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in neural information processing systems*, 33:17766–17778, 2020.
- [10] Manfred Mudelsee. Trend analysis of climate time series: A review of methods. *Earth-science reviews*, 190:310–322, 2019.
- [11] Joos-Hendrik Böse, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Dustin Lange, David Salinas, Sebastian Schelter, Matthias Seeger, and Yuyang Wang. Probabilistic demand forecasting at scale. *Proceedings of the VLDB Endowment*, 10(12):1694–1705, 2017.

- [12] Torben G Andersen, Tim Bollerslev, Peter Christoffersen, and Francis X Diebold. Volatility forecasting, 2005.
- [13] George EP Box and Gwilym M Jenkins. Time series analysis: Forecasting and control san francisco. *Calif: Holden-Day*, 1976.
- [14] Everette S Gardner Jr. Exponential smoothing: The state of the art. *Journal of forecasting*, 4(1):1–28, 1985.
- [15] Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric reviews*, 29(5-6):594–621, 2010.
- [16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
- [17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [18] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.
- [19] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
- [20] Ratnadip Adhikari and Ramesh K Agrawal. An introductory study on time series modeling and forecasting. *arXiv preprint arXiv:1302.6613*, 2013.
- [21] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.
- [22] Kyoung-jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.
- [23] Adela Sasu. K-nearest neighbor algorithm for univariate time series prediction. *Bulletin of the Transilvania University of Brasov• Vol*, 5(54), 2012.

- [24] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.
- [25] He Lyu, Ningyu Sha, Shuyang Qin, Ming Yan, Yuying Xie, and Rongrong Wang. Advances in neural information processing systems. *Advances in neural information processing systems*, 32, 2019.
- [26] Hardik Goel, Igor Melnyk, and Arindam Banerjee. R2n2: Residual recurrent neural networks for multivariate time series forecasting. *arXiv preprint arXiv:1709.03159*, 2017.
- [27] William WS Wei. *Multivariate time series analysis and applications*. John Wiley & Sons, 2018.
- [28] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.
- [29] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
- [30] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [31] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.