**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**KHOA/VIỆN CÔNG NGHỆ THÔNG TIN**

**---🙠**🕮**🙢---**

Ảnh có chứa văn bản, ký hiệu

Mô tả được tạo tự động

**BÁO CÁO BÀI TẬP LỚN MÔN HỌC MÁY**

**Đề tài: Dự báo thời tiết**

**Giáo viên hướng dẫn: TS. Nguyễn Nhật Quang**

**Sinh viên thực hiện:**

**1. Nguyễn Hoàng Anh 20204511**

**2. Nghiêm Ngọc Phong 20204676**

**3. Nguyễn Thị Bạch Tuyết 20204621**

**4. Nguyễn Quốc Việt 20204704**

**5. Nguyễn Anh Vũ 20204707**

**Lớp: 128706**

**MỤC LỤC**

[I. MÔ TẢ BÀI TOÁN 4](#_Toc123072149)

[II. PHÂN TÍCH BÀI TOÁN 4](#_Toc123072150)

[1. Hướng giải quyết bài toán 4](#_Toc123072151)

[2. Cơ sở lý thuyết 4](#_Toc123072152)

[**2.1 Recurrent Neural Network** 4](#_Toc123072153)

[**2.2 Long Short-term memory** 6](#_Toc123072154)

[**2.3 Optimizer** 8](#_Toc123072155)

[**2.3 Initialization** 10](#_Toc123072156)

[III. XÂY DỰNG MÔ HÌNH 11](#_Toc123072157)

[1. Thu thập dữ liệu 11](#_Toc123072158)

[2. Tiền xử lý dữ liệu 11](#_Toc123072159)

[3. Xây dựng mô hình 16](#_Toc123072160)

[3.1. Xây dựng mạng RNN (Recurrent Neural Network) 17](#_Toc123072161)

[3.2 Xây dựng mạng LSTM (Long short term memory) 21](#_Toc123072162)

[4. Kết quả 23](#_Toc123072163)

[IV. ĐÁNH GIÁ 25](#_Toc123072164)

[V. Các khó khăn gặp phải và hướng phát triển trong tương lai 25](#_Toc123072165)

[VI. Tài liệu tham khảo 26](#_Toc123072166)

**BẢNG HÌNH MINH HỌA**

[Hình 1. Sơ đồ hoạt động của RNN 4](#_Toc122587525)

[Hình 2. Hoạt động xảy ra trong một step 5](#_Toc122587526)

[Hình 3. Các dạnh mô hình RNN 6](#_Toc122587527)

[Hình 4. Sơ đồ hoạt động của LSTM 7](#_Toc122587528)

[Hình 5. Tính toán trong 2 mô hình 7](#_Toc122587529)

[Hình 6. Gradient dưới góc nhìn vật lý (bên trái là GD, bên phải là GD with Momentum) 8](#_Toc122587530)

[Hình 7. Dữ liệu trong file csv ban đầu 11](#_Toc122587531)

[Hình 8. Dữ liệu bị thiếu giá trị ở cột 'R' 12](#_Toc122587532)

[Hình 9. Nhiệt độ trung bình trong ngày 12](#_Toc122587533)

[Hình 10. Nhiệt độ cao nhất trong ngày 13](#_Toc122587534)

[Hình 11. Nhiệt độ thấp nhất trong ngày 13](#_Toc122587535)

[Hình 12. Lượng mưa trong ngày 13](#_Toc122587536)

[Hình 13. Thống kê số ngày mưa và không mưa 14](file:///D:\Study\Machine%20learing%20and%20data%20mining\Báo%20cáo%20bài%20tập%20nhóm\Báo-cáo-1.docx#_Toc122587537)

[Hình 14.. Độ ẩm trung bình trong ngày 14](#_Toc122587538)

[Hình 15. Độ ẩm thấp nhất trong ngày 14](#_Toc122587539)

[Hình 16. Biểu đồ phân tích nhiệt độ trung bình 15](#_Toc122587540)

[Hình 17. Dữ liệu nhiệt độ trung bình trước và sau khi dùng bộ lọc trung vị 15](#_Toc122587541)

[Hình 18. Biểu đồ thể hiện các giá trị bất thường của giữ liệu 16](#_Toc122587542)

[Hình 19. Minh hoạ kiến trúc mô hình LSTM với 3 time steps, mỗi time step có 6 thuộc tính 16](#_Toc122587543)

[Hình 20. Mô hình chi tiết với n time step 17](#_Toc122587544)

[Hình 21. Thiết lập tham số 17](#_Toc122587545)

[Hình 22. Lấy dữ liệu đã được xử lý từ file csv 18](#_Toc122587546)

[Hình 23. Chia tập dữ liệu thành 3 tập train, valid và test 18](#_Toc122587547)

[Hình 24. Xây dựng ma trận dữ liệu 19](#_Toc122587548)

[Hình 25. Train mô hình (RNN) 20](#_Toc122587549)

[Hình 26. Màu xanh là giá trị train\_loss, màu đỏ là val\_loss (RNN) 20](#_Toc122587550)

[Hình 27. Nhiệt độ trung bình (RNN) 20](#_Toc122587551)

[Hình 28. Nhiệt độ cao nhất (RNN) 20](#_Toc122587552)

[Hình 29. Nhiệt độ thấp nhất (RNN) 20](#_Toc122587553)

[Hình 30. Độ ẩm trung bình (RNN) 21](#_Toc122587554)

[Hình 31. Độ ẩm thấp nhất (RNN) 21](#_Toc122587555)

[Hình 32. Dự đoán khả năng có mưa (RNN) 21](#_Toc122587556)

[Hình 33. Mô hình LSTM 21](#_Toc122587557)

[Hình 34. Train mô hình (LSTM) 22](#_Toc122587558)

[Hình 35. Đường màu xanh là thông số train\_loss, đường màu đỏ là val\_loss(LSTM) 22](#_Toc122587559)

[Hình 36. Nhiệt độ trung bình (LSTM) 22](#_Toc122587560)

[Hình 37. Nhiệt độ cao nhất (LSTM) 22](#_Toc122587561)

[Hình 38. Nhiệt độ thấp nhất (LSTM) 23](#_Toc122587562)

[Hình 39. Độ ẩm trung bình (LSTM) 23](#_Toc122587563)

[Hình 40. Độ ẩm thấp nhất (LSTM) 23](#_Toc122587564)

[Hình 41. Dự đoán khả năng có mưa (LSTM) 23](#_Toc122587565)

[Hình 42. Sai số của mô hình RNN khi dùng 3 ngày dự đoán 1 ngày 24](#_Toc122587566)

[Hình 43. Sai số của mô hình LSTM khi dùng 3 ngày dự đoán 1 ngày 24](#_Toc122587567)

[Hình 44. Sai số của mô hình LSTM khi dùng 15 ngày dự đoán 1 ngày 24](#_Toc122587568)

[Hình 45. Sai số của mô hình RNN khi dùng 3 ngày dự đoán 3 ngày 25](#_Toc122587569)

# MÔ TẢ BÀI TOÁN

Dự báo thời tiết là một bài toán hết sức quan trọng trong cuộc sống hằng ngày. Đầu vào của bài toán sẽ là số liệu về mây, mực nước, lượng mưa, nhiệt độ, độ ẩm, tốc độ gió,... theo ngày của một khu vực nhất định và đầu ra sẽ là các thông số tương ứng của ngày tiếp theo. Trong thực tế việc dự đoán gặp nhiều sai số, khó khăn khi có những ngày thời tiết thất thường, không theo quy luật hoặc thiên tai làm cho dữ liệu không còn đúng. Trong phạm vi bài tập lớn này, mô hình sẽ chỉ sử dụng các thông số khí tượng là nhiệt độ, độ ẩm, lượng mưa và sử dụng thuật toán RNN và LSTM để dự đoán các thông số nhiệt độ, độ ẩm, có mưa hay không cho ngày hôm sau.

# PHÂN TÍCH BÀI TOÁN

## Hướng giải quyết bài toán

Dữ liệu được lấy từ năm 2014 đến năm 2021 của trạm khí tượng tại một vùng núi phía Bắc. Dữ liệu ban đầu có một số chỗ bị NULL và một số chỗ dữ liệu thất thường (do thiên tai), vậy nên ta cần có bước tiền xử lý để bổ sung những chỗ dữ liệu bị thiếu và chuẩn hóa những chỗ dữ liệu thất thường. Ta sẽ trực quan hóa dữ liệu trên biểu đồ để kiểm tra sự tuần hoàn.

Sau khi làm sạch dữ liệu, ta sẽ lựa chọn Feature Vector chứa các số liệu khí tượng và chia thành 3 tập: training set, optimal set và test set.

Cuối cùng, chúng ta sẽ tiến hành xây dựng 2 mô hình máy học là RNN và LSTM để dự đoán và so sánh kết quả của 2 mô hình này.

## Cơ sở lý thuyết

### **2.1 Recurrent Neural Network**

Recurrent Neural Network (RNN) là một thuật toán cực kì quan trọng có khả năng xử lý thông tin dạng chuỗi. RNN có rất nhiều ứng dụng trong thực tế như: Machine Translation, mô hình hóa ngôn ngữ và sinh văn bản, nhận dạng giọng nói, mô tả hình ảnh: RNN kết hợp cùng CNN để sinh ra mô tả cho hình ảnh chưa được gán nhãn.

Với mạng Nơ-ron thông thường, dữ liệu sẽ được cho vào cùng một lúc. Nhưng trong thực tế sẽ có rất nhiều bài toán, dữ liệu của chúng ta mang ý nghĩa trình tự, nếu thay đổi trình tự kết quả sẽ khác. Mạng Nơ-ron RNN có khả năng giải quyết bài toán này.

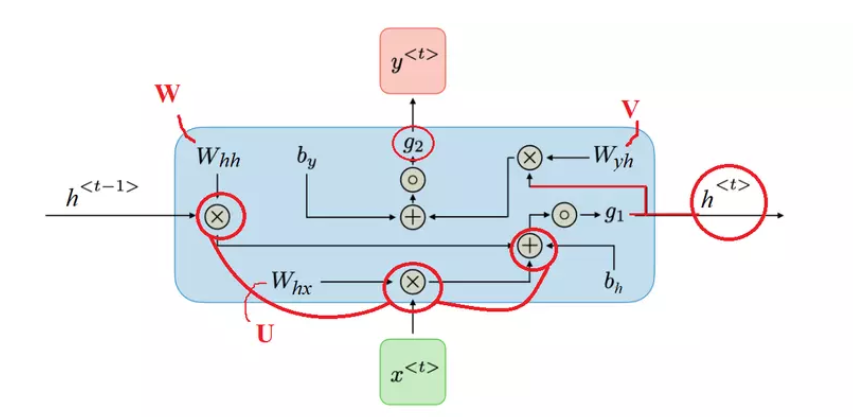
Hình ảnh dưới đây sẽ mô tả RNN hoạt động như thế nào.

Ảnh có chứa văn bản, đồng hồ, ký hiệu

Mô tả được tạo tự động

Hình 1. Sơ đồ hoạt động của RNN

Mỗi block RNN sẽ lấy thông tin từ các block trước và input hiện tại. Các x ở đây đại diện cho dữ liệu đầu vào lần lượt (được chia theo time step). ​ đại diện cho time step thứ t, và là output của một step.



Hình 2. Hoạt động xảy ra trong một step

Chúng ta sẽ cùng tìm hiểu các tham số trong hình trên. Đầu tiên là (hidden state) là trạng thái ẩn tại bước t, có thể gọi đây là bộ nhớ của mạng. Hidden state được tính toán dự trên các hidden state của các bước trước đó nên nó chứa thông tin của các hidden state trước đó và cộng với input hiện tại theo công thức = ( \* + \* + ) trong đó g1 là hàm tác động *tanh.*  được khởi tạo theo kỹ thuật [Glorot initialization](#glorot_uniform), còn là [Orthogonal initialization](#orthogonal_init) của từng timestep. Tại một block của mạng RNN có 2 đầu ra. Trong đó, là tổng hợp thông tin của các state trước để tiếp tục truyền đi trong chuỗi mạng, g2 là hàm tác động *softmax* và là output của từng time step một. được tính theo công thức sau:

( )

Nhằm đơn giản hoá, nhóm đã sử dụng layer SimpleRNN của Keras với được tínhtheo công thức:

()

Các dạng dự báo trong mô hình RNN được sử dụng là: many to one, many to many.

A picture containing text, clock, clipart

Description automatically generatedShape

Description automatically generated

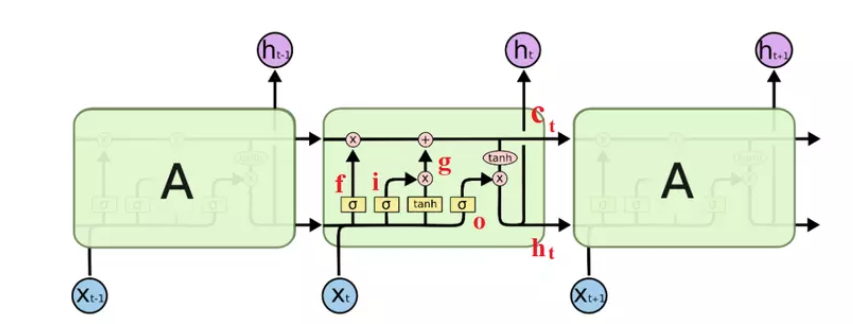
Hình 3. Các dạnh mô hình RNN

**Many to one:** là trường hợp đầu vào có *ո* time steps, đầu ra là vector output tại time step cuối cùng trong mạng.

**Many to many:** là trường hợp đầu vào có *ո* time steps, đầu ra là chuỗi các vector output tại mỗi time step.

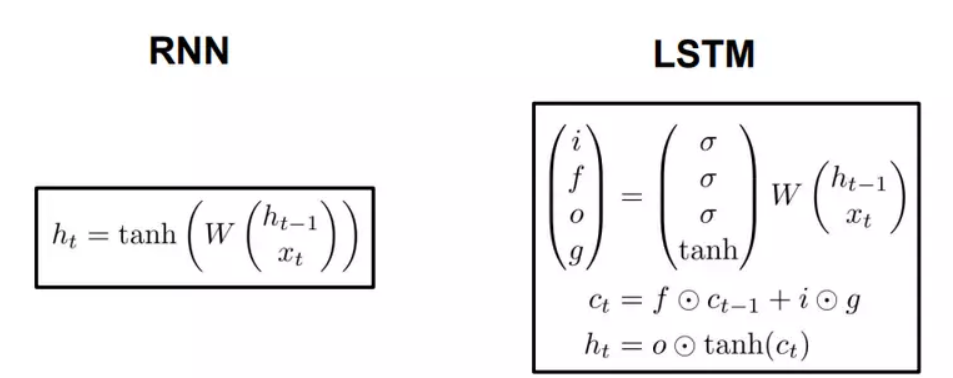
### **2.2 Long Short-term memory**

Có thể nói, Long Short-term memory là phiên bản cải tiến của Recurrent Neural Network. Về lý thuyết là 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 bị vanishing gradient, hay nói cách khác là model chỉ học được từ các state gần nó. Nhằm mục đích thử nghiệm, nhóm đã tìm đến LSTM với hi vọng mô hình có thể học được các phụ thuộc xa, và trên cơ sở đó để cải thiện kết quả so với RNN.



Hình 4. Sơ đồ hoạt động của LSTM

Về cơ bản, ý tưởng của RNN và LSTM không khác nhau là mấy. Chúng ta chỉ thêm một số tính toán ở mô hình của LSTM. Tất cả được tóm tắt trong hình sau.



Hình 5. Tính toán trong 2 mô hình

Đầu tiên, chúng ta có i, f, g có công thức gần giống hệt nhau và chỉ khác mỗi ma trận tham số. Chính ma trận này sẽ quyết định chức năng khác nhau của từng cổng. là ký hiệu của hàm sigmoid. Quan sát hình 4 để thấy rõ hơn vị trí các cổng.

**Input gate i** là cổng vào. Cổng vào giúp quyết định bao nhiêu lượng thông tin đầu vào sẽ ảnh hưởng đến trạng thái mới. Quyết định bằng cách nào, thông qua đặc điểm của hàm sigmoid (đầu ra nằm trong khoảng [0,1]), như vậy khi một vector thông tin đi qua đây, nếu nhân với 0, vector sẽ bị triệt tiêu hoàn toàn. Nếu nhân với 1, hầu hết thông tin sẽ được giữ lại.

Tương tự như vậy, **forget gate f** là cổng quên. Cổng quên sẽ quyết định sẽ bỏ đi bao nhiêu lượng thông tin đến từ trạng thái trước đó.

Cuối cùng, cổng **output gate o** là cổng ra. Cổng ra điều chỉnh lượng thông tin có thể ra ngoài và lượng thông tin truyền tới trạng thái tiếp theo.

Tiếp theo, g thực chất cũng chỉ là một trạng thái ẩn được tính dựa trên đầu vào hiện tại và trạng thái trước . Tính hoàn toàn tương tự như input gate, chỉ thay vì dùng sigmoid, ta dùng tanh. Kết hợp hai điều này lại để cập nhật trạng thái mới.

Ta có là bộ nhớ trong của LSTM. Nhìn vào công thức, có thể thấy nó là tổng hợp của bộ nhớ trước đã được lọc qua cổng quên **f**, cộng với trạng thái ẩn g đã được lọc bởi cổng vào **i**. Cell state sẽ mang thông tin nào quan trọng truyền đi xa hơn và sẽ được dùng khi cần. Đây chính là long term memory.

Sau khi có , ta sẽ đưa nó qua cổng ra để lọc thông tin một lần nữa, thu được trạng thái mới . Ta có thể thấy RNN truyền thống là dạng đặc biệt của LSTM. Nếu thay giá trị đầu ra của input gate là 1 và đầu ra forget gate là 0 (không nhớ trạng thái trước), ta được RNN thuần.

Tổng kết lại, LSTM giải quyết phần nào vanishing gradient so với RNN, nhưng chỉ một phần và với lượng tính toán như trên, RNN đã chậm, LSTM nay còn chậm hơn. Tuy vậy, với những cải tiến so với RNN thuần, LSTM đã và đang được sử dụng phổ biến. Trên thực tế, cách cài đặt LSTM cũng rất đa dạng và linh hoạt theo bài toán. Trong bài toán dự báo thời tiết, dữ liệu có dạng Time-series, vậy nên rất thích hợp với mô hình RNN và LSTM. Dưới đây, chúng ta sẽ cùng áp dụng hai mô hình này vào giải quyết bài toán dự báo thời tiết và so sánh kết quả giữa hai mô hình.

### **2.3 Optimizer**

Nhóm sử dụng giải thuật **Adaptive Moment Estimation** (*Adam*), là sự kết hợp của hai giải thuật **Gradient descent with Momentum** và **Root Mean Square Propagation** (*RMSP*).

**Momentum** được ra đời nhằm giải quyết vấn đề tối ưu cục bộ của **Gradient Descent** (*GD*). Dưới góc nhìn vật lý, giải thuật cung cấp động lượng giúp viên bi bắt đầu từ B có thể vượt qua được điểm tối ưu cục bộ D để tới điểm tối ưu toàn cục C.

A picture containing shape

Description automatically generated

Hình 6. Gradient dưới góc nhìn vật lý (bên trái là GD, bên phải là GD with Momentum)

Về mặt toán học, trọng số được cập nhật theo công thức:

với,

Trong đó:

* là tốc độ học
* là siêu tham số (thường được chọn là 0.9)
* là gradient tại

**Momentum** đã giải quyết được vấn đề tối ưu cục bộ nhưng lại sinh ra vấn đề mới là lúc gần tới được nghiệm toàn cục thì mất thời gian giao động qua lại trước khi dừng hẳn khiến thời gian huấn luyện bị kéo dài.

**Root Mean Square Propagation** (*RMSP*) là phiên bản cải tiến của **Adagrad**, giải thuật thay đổi tốc độ học theo thời gian. **RMSP** giải quyết vấn đề tốc độ học giảm dần của **Adagrad** bằng cách chia tốc độ học cho trung bình của bình phương gradient:

với,

Trong đó:

* + là tốc độ học tại thời điểm *t*
  + là hằng số dương, được thêm vào để tránh chia cho 0 (thường được chọn là 10-8)
  + là siêu tham số (thường được chọn là 0.9)
  + là gradient tại

Nhìn vào công thức cập nhật trọng số ở trên thì **RMSP** chỉ thay đổi tốc độ học để hội tụ nhanh hơn chứ chưa giải quyết được vấn đề tối ưu cục bộ.

Như đã nói ở trên, **Adam** là sự kết hợp của **Momentum** và **RMSP**. Giải thuật kiểm soát tốc độ học sao cho có giao động nhỏ nhất khi đạt đến điểm tối ưu toàn cục và đồng thời tạo ra động lượng đủ lớn để vượt qua điểm tối ưu cục bộ. Kế thừa hai phương trình sau:

Trọng số được cập nhật theo công thức:

Trong đó:

* là tốc độ học (thường được chọn là 0.001)
* là hằng số dương, được thêm vào để tránh chia cho 0 (thường được chọn là 10-8)
* , là các siêu tham số (thường được chọn lần lượt là 0.9 và 0.999)

Từ công thức toán học, ta có thể thấy **Adam** gồm có hai kỹ thuật:

* Tính exponential moving average của giá trị đạo hàm lưu vào biến và sử dụng nó là tử số của việc cập nhật hướng. Với ý nghĩa là nếu có giá trị lớn, thì việc descent đang đi đúng hướng và chúng ta cần bước nhảy lớn hơn để đi nhanh hơn. Tương tự, nếu giá trị nhỏ, phần descent có thể không đi về hướng tối tiểu và chúng ta nên đi 1 bước nhỏ để thăm dò. Đây là phần **Momentum** của thuật toán.
* Tính exponential moving average của bình phương gía trị đạo hàm lưu vào biến và sử dụng nó là phần mẫu số của việc cập nhật hướng. Với ý nghĩa như sau: Giả sử gradient mang các giá trị dương, âm lẫn lộn, thì khi cộng các giá trị lại theo công thức tính ta sẽ được giá trị gần số 0. Do âm dương lẫn lộn nên nó bị triệt tiêu lẫn nhau. Nhưng trong trường hợp này thì sẽ mang giá trị lớn. Do đó, trong trường hợp này, chúng ta sẽ không hướng tới cực tiểu, chúng ta sẽ không muốn đi theo hướng đạo hàm trong trường hợp này. Chúng ta để ở phần mẫu vì khi chia cho một giá trị cao, giá trị của các phần cập nhật sẽ nhỏ, và khi có giá trị thấp, phần cập nhật sẽ lớn. Đây chính là phần tối ưu **RMSP** của thuật toán.

Qua quá trình tìm hiểu các giải thuật tối ưu, với những ưu điểm mạnh của Adam, nhóm đã quyết định sử dụng nó để tối ưu mô hình.

### **2.3 Initialization**

**[Glorot initialization](https://www.tensorflow.org/api_docs/python/tf/keras/initializers/GlorotUniform)** hay còn gọi là **Xavier initialization** là kỹ thuật khởi tạo trọng số để tránh hiện tượng gradient không định trong mô hình có nhiều tầng ần. Cụ thể, kỹ thuật sử dụng phân phối đều trong khoảng với:

Trong đó: , là số đầu vào và đầu ra của layer tương ứng.

Không giống như [Glorot](#glorot_uniform), [**Orthogonal initialization**](https://smerity.com/articles/2016/orthogonal_init.html?fbclid=IwAR3Q0FqSXPl75z84w9NVrF4yEbLNQWtCC9vpdxb67RS_Fj_8ZHSGqs199hw)dùng ma trận trực giao để khởi tạo trọng số (ma trận có các trị riêng ).

Xét mô hình RNN đơn giản với = 0, không sử dụng **bias**, , ***n*** là số time step.

= ( \* + \* ) = \*

= () =

Không mất tính tổng quát, coi là ma trận vuông, khi đó ta có:

Khi đó,

với là ma trận đường chéo với các trị riêng được đặt trên đường chéo, là ma trận với các vector riêng tương ứng của . Từ kết quả trên, khi ***n*** đủ lớn ta thấy:

* biến mất (hiện tượng vanishing gradient) khi tất cả .
* duy trì ổn định khi tất cả .
* bùng nổ (hiện tượng exploding gradient) có một .

Do đó việc khởi tạo ma trận trọng số có các trị riêng hạn chế được phần nào hiện tượng vanishing, exploding gradient. Vì vậy nhóm sử dụng **Orthogonal initialization** để khởi tạo trọng sốtrong mô hình RNN mà LSTM**.**

# XÂY DỰNG MÔ HÌNH

## Thu thập dữ liệu

Dữ liệu là một tài sản vô cùng giá trị, có những dữ liệu chúng ta muốn lấy cũng không được. Rất may mắn, nhờ một mối quan hệ đặc biệt nên nhóm em đã xin được file dữ liệu khí tượng tại một trạm khí tượng của một vùng núi phía Bắc từ năm 2014 đến năm 2021.

Dữ liệu được lưu trữ trong các file mặt cắt theo từng năm gồm các thông số: Nhiệt độ trung bình, nhiệt độ thấp nhất, nhiệt độ cao nhất, độ ẩm trung bình, độ ẩm thấp nhất, độ ẩm cao nhất và lượng mưa. Để thuận tiện trong việc xử lý và lấy ra, số liệu trong các file mặt cắt này được tổng hợp lại trong file “weatherdataset.csv”

Ảnh có chứa bàn

Mô tả được tạo tự động

Hình 7. Dữ liệu trong file csv ban đầu

## Tiền xử lý dữ liệu

Dữ liệu ban đầu tại cột R (lượng mưa) chứa một số chỗ còn thiếu, một số chỗ dữ liệu thất thường, ta cần chuẩn hóa dữ liệu đầu vào này.

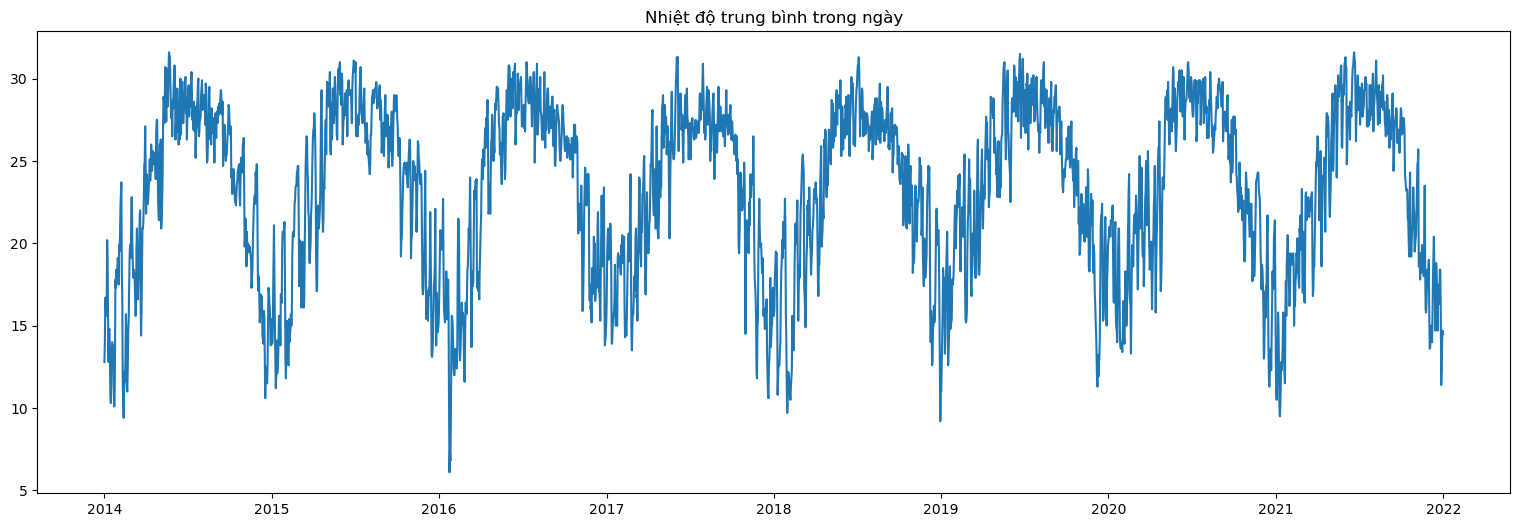
Ảnh có chứa văn bản

Mô tả được tạo tự động

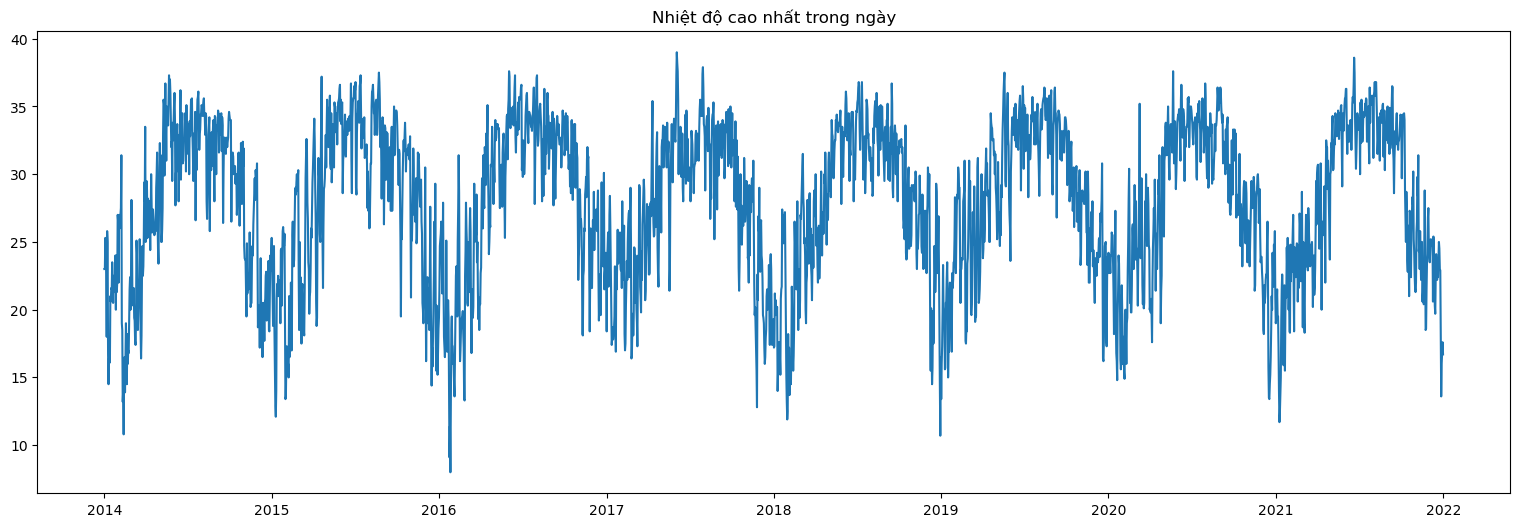
Hình 8. Dữ liệu bị thiếu giá trị ở cột 'R'

Đầu tiên, trong cột lượng mưa (R), các giá trị NaN chính là những hôm không mưa, còn những hôm lượng mưa = 0.0 có nghĩa là có mưa nhưng lượng mưa rất thấp. Vì vậy, ta sẽ thay thế những giá trị NaN bằng giá trị -1 để tránh nhầm lẫn với những hôm mưa bé.

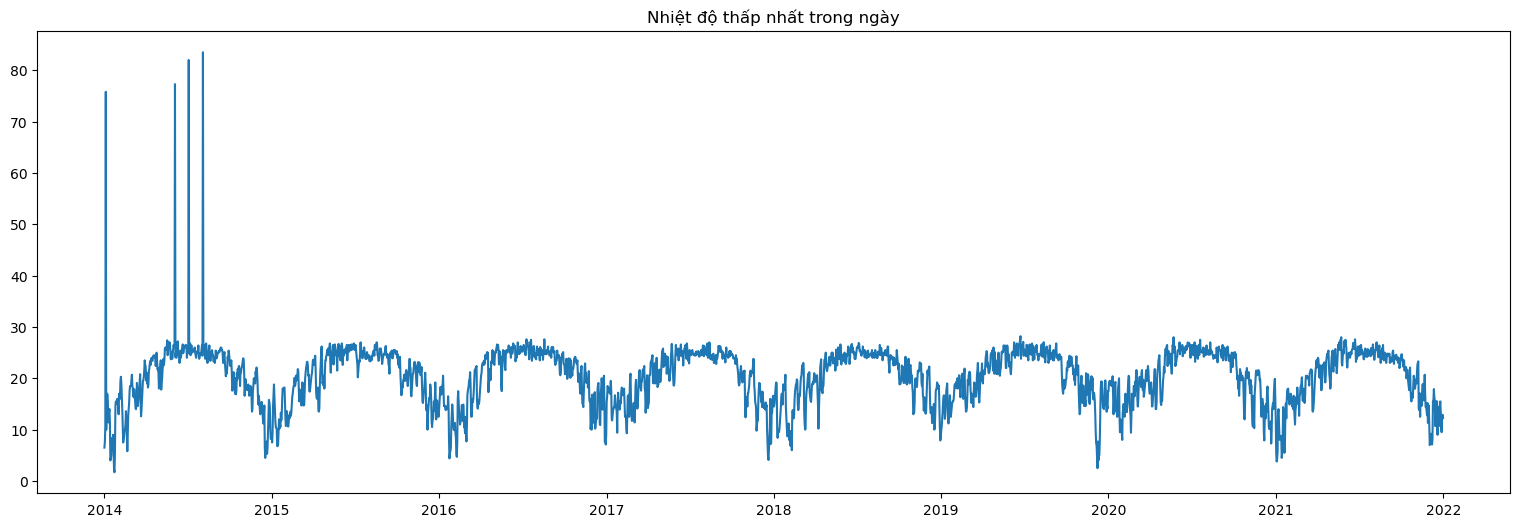
Để có cái nhìn trực quan về dữ liệu thì tôi sẽ vẽ biểu đồ từng cột dữ liệu bằng thư viện Matplotlib và Seaborn:



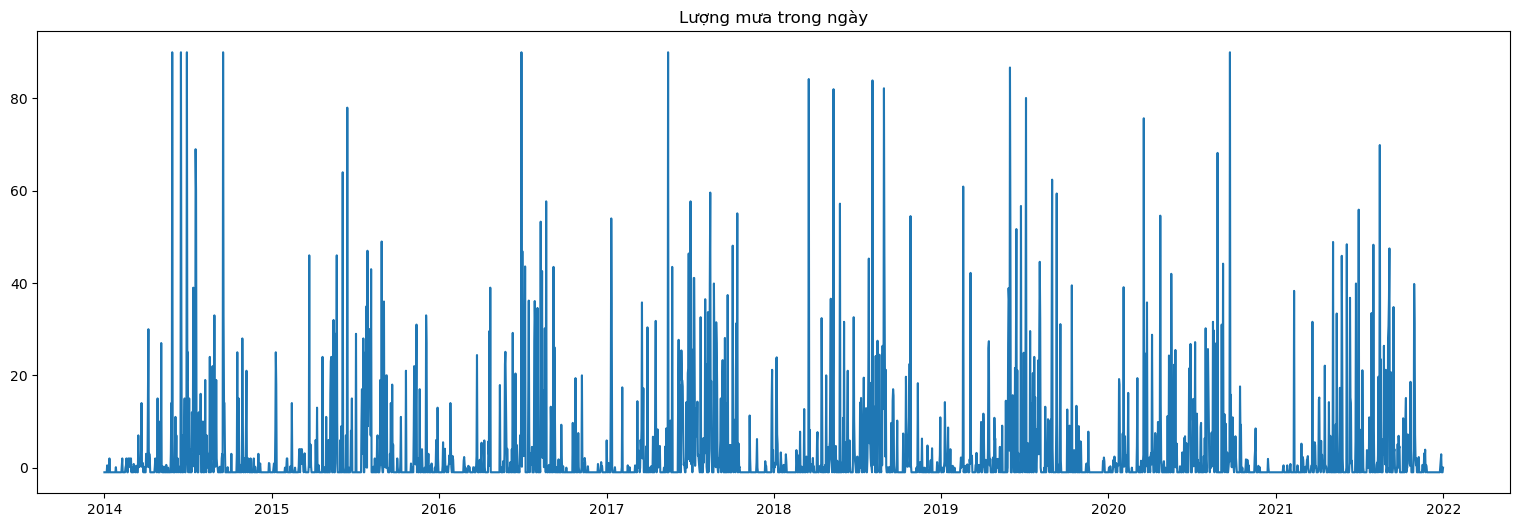
Hình 9. Nhiệt độ trung bình trong ngày



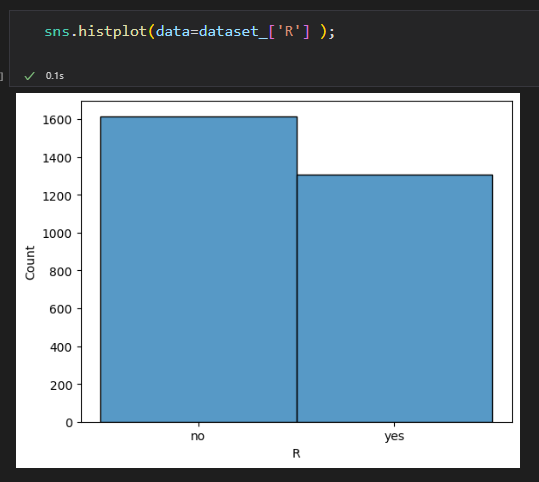
Hình 10. Nhiệt độ cao nhất trong ngày



Hình 11. Nhiệt độ thấp nhất trong ngày



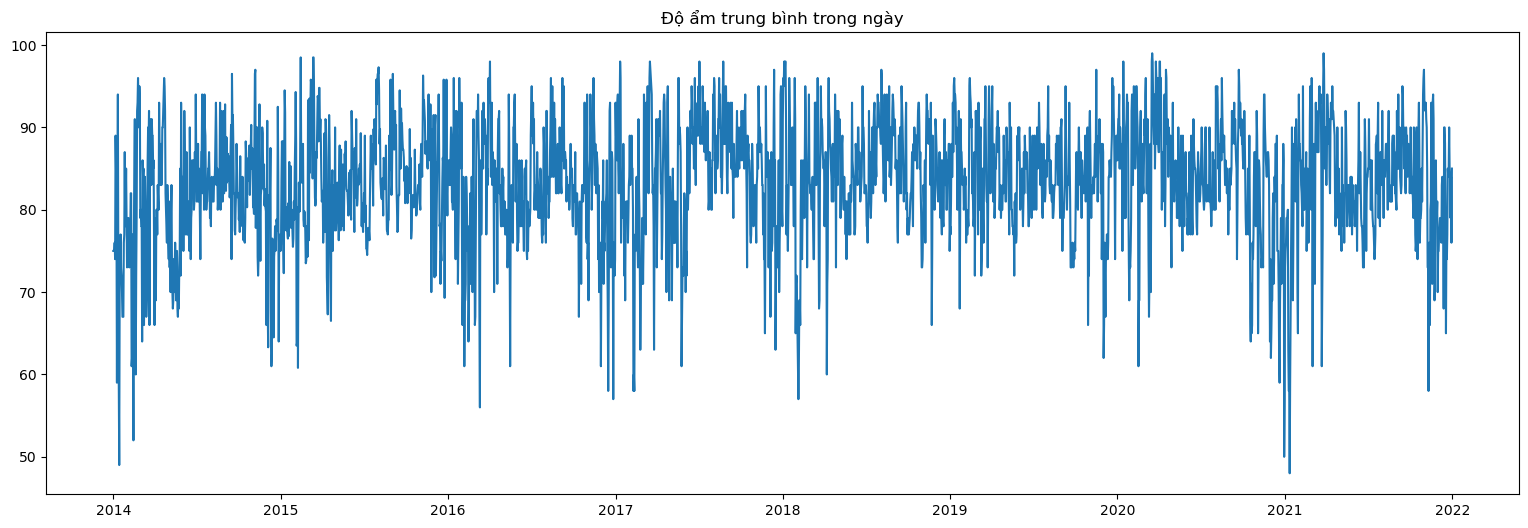
Hình 12. Lượng mưa trong ngày

Thống kê theo số ngày mưa và không mưa, ta có số ngày mưa chiếm 50%, luợng mưa thay đổi thất thường (ngày hôm qua không có mưa nhưng ngày hôm này có thể mưa rất nhiều), vì vậy ta thay đổi giá trị của cột mưa thành các nhãn 0 (không có mưa) và 1 (có mưa) nhằm đưa ra dự đoán chính xác hơn và tránh gây nhiễu đến việc dự đoán các cột khác.

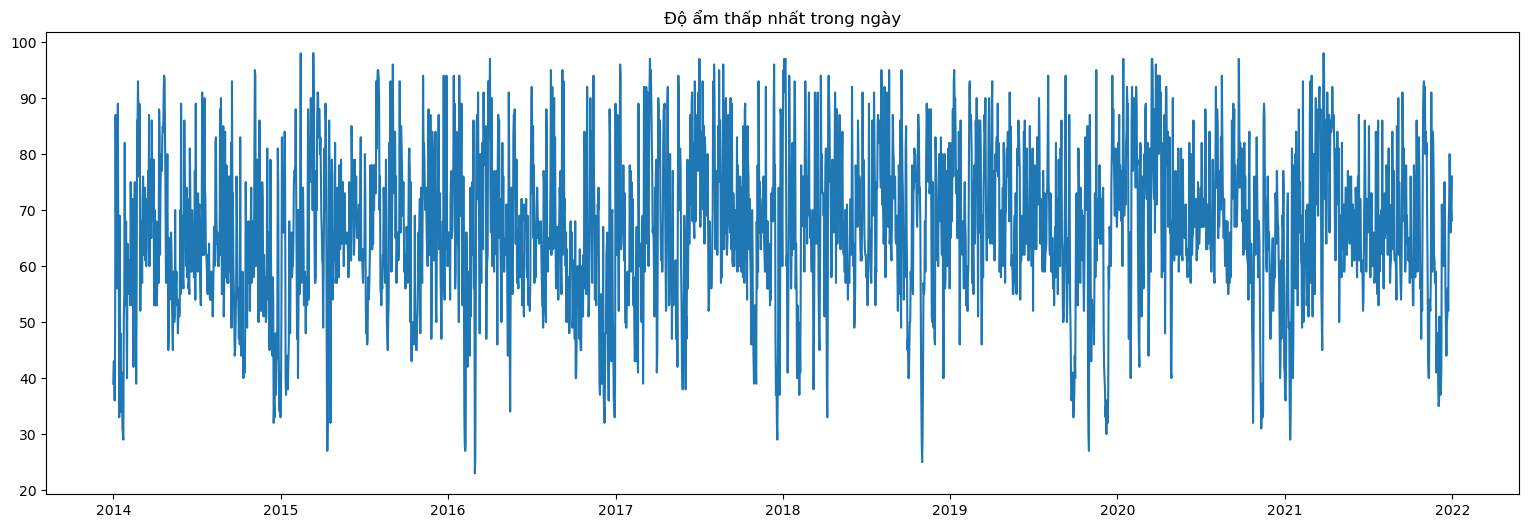
Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 13. Thống kê số ngày mưa và không mưa

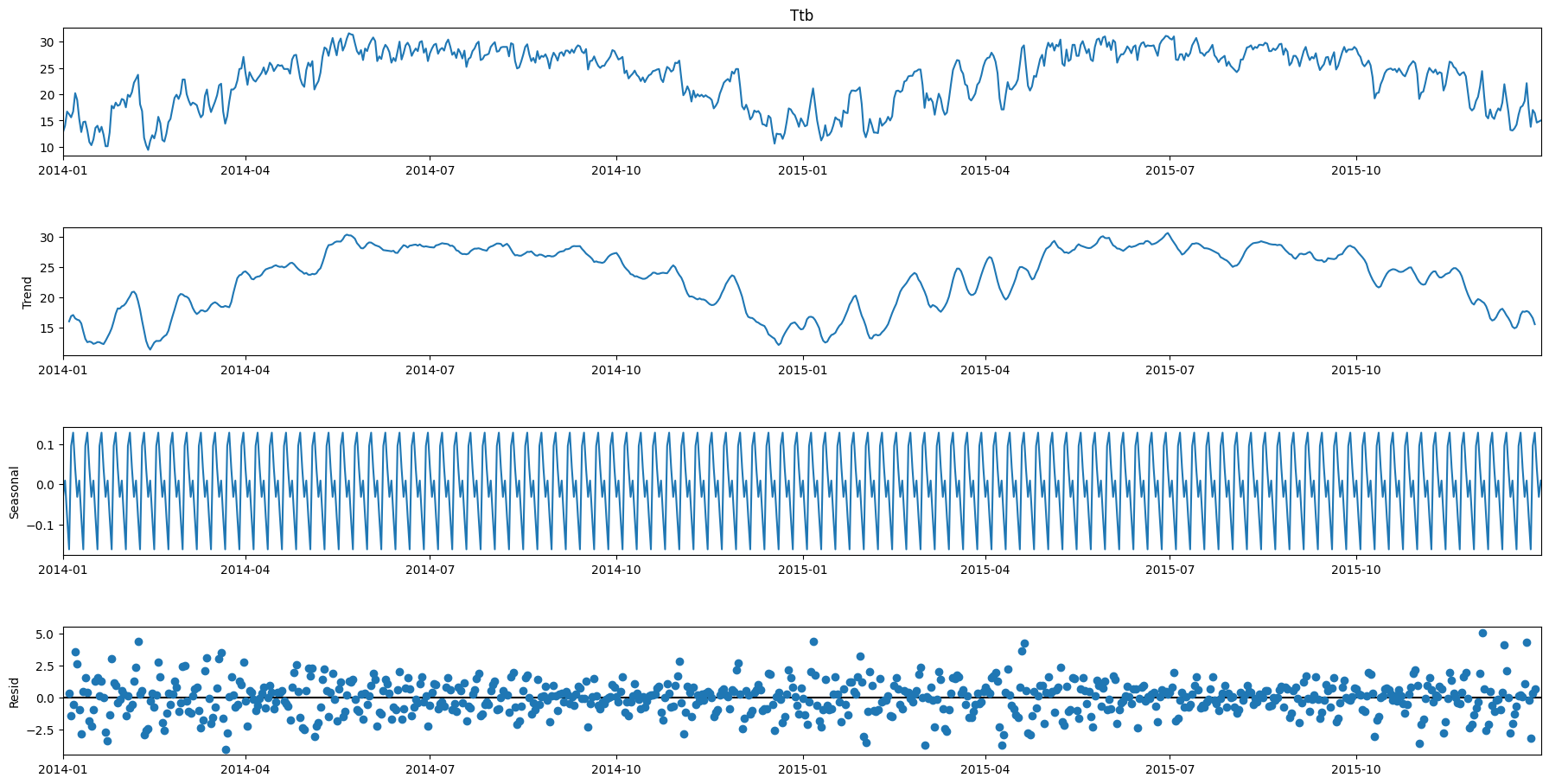


Hình 14.. Độ ẩm trung bình trong ngày



Hình 15. Độ ẩm thấp nhất trong ngày

Ta có thể thấy điều đặc biệt trong dữ liệu khí tượng là sự tương đối tuần hoàn qua các năm trừ một vài ngày có dữ liệu khác thường, có thể do thiên tai hoặc sai sót trong quá trình nhập liệu. Những dữ liệu bất thường này sẽ làm cho kết quả dự đoán bị sai lệch. Vì vậy, chúng ta phải giảm nhiễu dữ liệu. Dưới đây, chúng ta sẽ phân tích dữ liệu nhiệt độ trung bình:

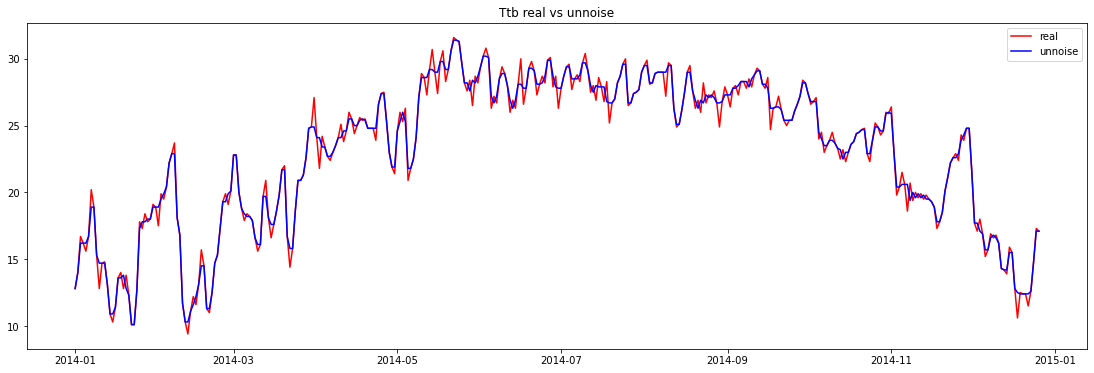


Hình 16. Biểu đồ phân tích nhiệt độ trung bình

Biểu đồ trên cùng là thống kê nhiệt độ trung bình theo ngày; biểu đồ thứ hai thể hiện xu hướng của dữ liệu theo thời gian: lên hay xuống, tăng hay giảm; biểu đồ thứ ba cho thấy sự tuần hoàn của dữ liệu trong từng khoảng thời gian cố định; biểu đồ cuối cùng là thành phần còn lại sau khi trích xuất yếu tố xu hướng (trend) và mùa vụ (seasonal) của dữ liệu, nó chỉ ra sự bất thường của dữ liệu so với hai yếu tố có quy luật trên nằm trong khoảng .

Để giảm sự bất thường của dữ liệu, ta sử dụng bộ lọc trung vị median filter kích thước 1x3 để loại bỏ các hôm có số liệu thay đổi bất thường x\_i\_new = median([x[i-1], x[i], x[i+1]])

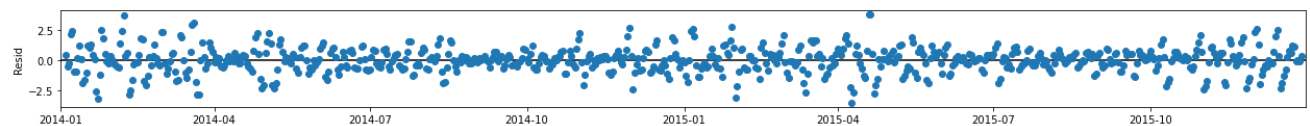
Ví dụ: nếu input[4,2,5,9,3,6,5] thì output sẽ là [4,4,5,5,6,5,5]



Hình 17. Dữ liệu nhiệt độ trung bình trước và sau khi dùng bộ lọc trung vị

Đường màu đỏ là dữ liệu trước khi dùng bộ lọc, tồn tại các đỉnh nhọn trên biểu đồ do các điểm dữ liệu bất thường gây ra, còn trên đường màu xanh, các đỉnh nhọn đã được làm mịn hơn.

Biểu đồ bên dưới cho thấy khoảng bất thường của dữ liệu đã thu hẹp lại về khoảng



Hình 18. Biểu đồ thể hiện các giá trị bất thường của giữ liệu

Như vậy bước bước tiền xử lý dữ liệu đã xong

## Xây dựng mô hình

Nhóm tiến hành xây dựng hai mô hình RNN và LSTM chung một cấu trúc như hình sau:

Table

Description automatically generated

Hình 19. Hình minh hoạ kiến trúc mô hình LSTM với 3 time steps, mỗi time step có 6 thuộc tính

Với ý tưởng xếp chồng layer RNN (hoặc LSTM) ‘many to one’ lên layer RNN (hoặc LSTM) ‘many to many’ nhằm phát huy được tối đa sức mạnh của mạng nơ ron hồi quy. Hình ảnh dưới đây sẽ mô tả rõ cách hoạt động của mô hình.

Diagram

Description automatically generated

Hình 20. Mô hình chi tiết với n time step

Trọng số ở mỗi layer được khởi tạo

### 3.1. Xây dựng mạng RNN (Recurrent Neural Network)

Đầu tiên ta thiết lập các tham số

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 21. Thiết lập tham số

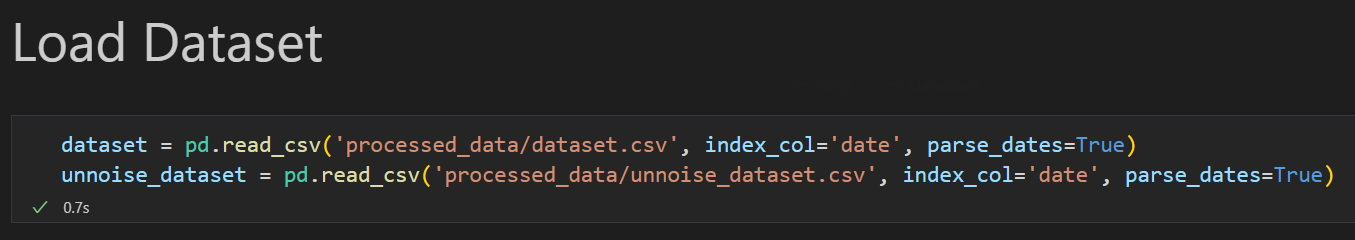
*n-timestep* là số ngày dùng để huấn luyện, dự đoán cho các ngày tiếp theo, ở đây ta chọn bằng 3 vì trong thực tế, thời tiết của một ngày sẽ có sự tương quan so với một số ngày trước đó.

*n-timepred* là số ngày dự đoán ra từ 3 ngày trước đó (n-timestep). Trong mô hình này, chúng ta sẽ sử dụng 3 ngày hôm trước để dự đoán 1 ngày tiếp theo.

*n-feature* là số thuộc tính của đầu vào, ở đâu 6 thuộc tính là: nhiệt độ trung bình, nhiệt độ thấp nhất, nhiệt độ cao nhất, độ ẩm trung bình, độ ẩm thấp nhất và nhãn có mưa hay không.

*train-val-test-rate* là tỉ lệ chia dữ liệu thành 3 tập: training, validation và test. Nghĩa là chúng ta sẽ có dữ liệu của 6 năm để training, 1 năm để tối ưu và 1 năm để test

Tiếp theo ta sẽ load dự liệu từ file csv đã tiền xử lý trước đó. Chúng ta sẽ sử dụng hai bộ dữ liệu, bộ dataset là bộ dữ liệu chưa giảm nhiễu, bộ unnoise là bộ dữ liệu đã giảm nhiễu để so sánh kết quả.



Hình 22. Lấy dữ liệu đã được xử lý từ file csv

Tiến hành scale dữ liệu về khoảng (0,1) theo công thức:

A picture containing text, watch

Description automatically generated

Sau khi scale, chúng ta sẽ tiến hành chia bộ dữ liệu theo tỉ lệ đã khai báo ở trên.

Ảnh có chứa văn bản, màn hình, ảnh chụp màn hình, trong nhà

Mô tả được tạo tự động

Hình 23. Chia tập dữ liệu thành 3 tập train, valid và test

Bước tiếp theo chúng ta sẽ chia sữ liệu theo time step cho traing-set, validation-set và test-set.

Ảnh có chứa văn bản

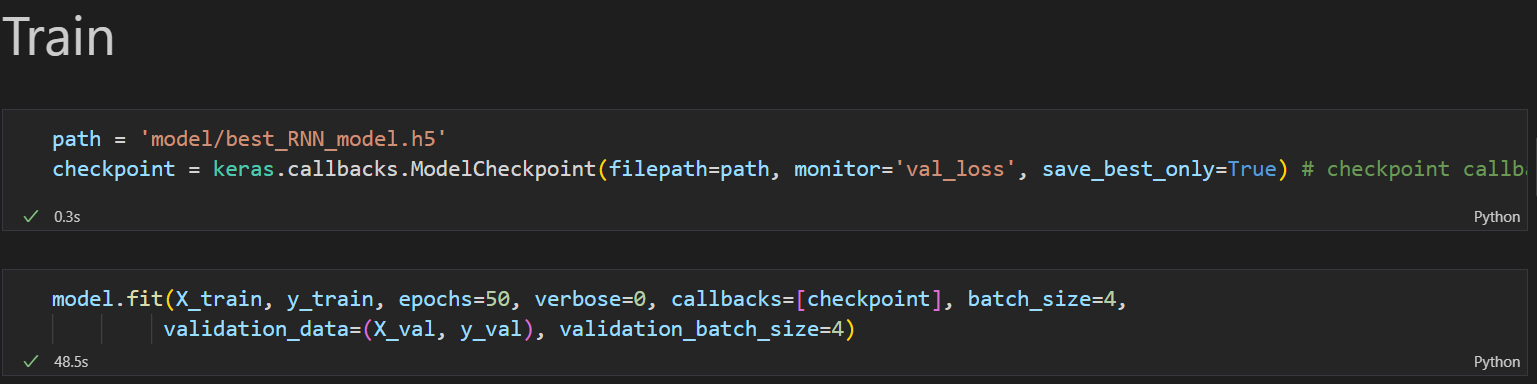
Mô tả được tạo tự động

Hình 24. Xây dựng ma trận dữ liệu

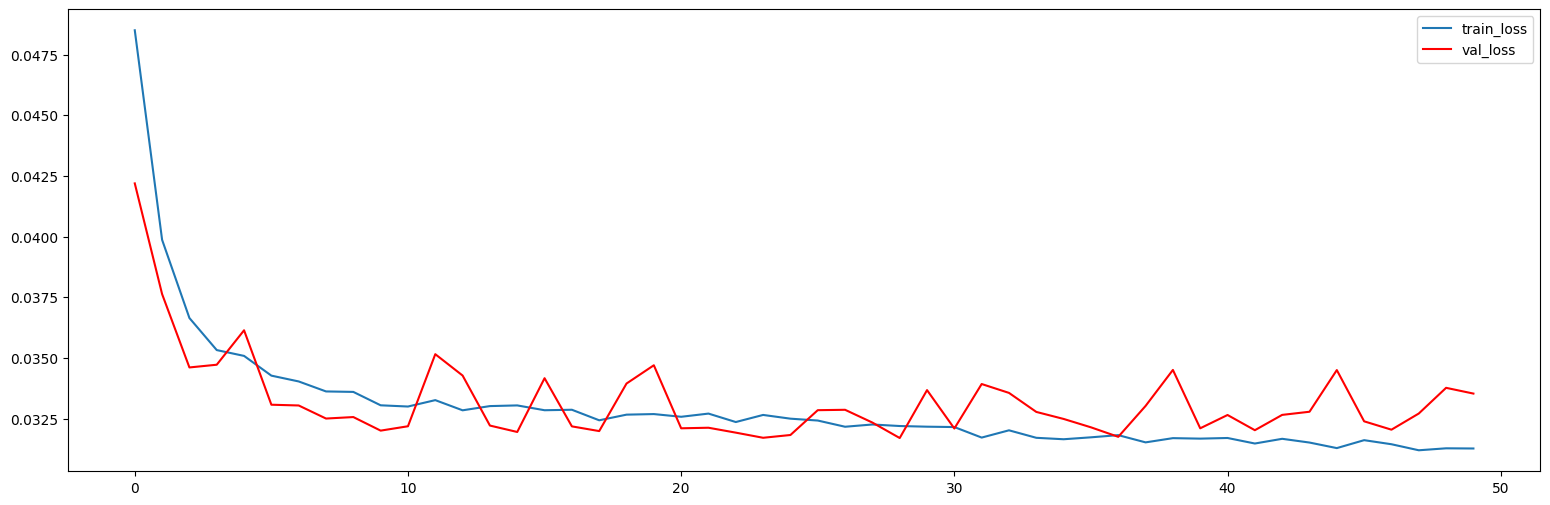
Chúng ta sẽ phân tích X-train để hiểu rõ hơn. Chiều đầu tiên của ma trận là tổng số ngày trong tập train, chiều thứ 2 là số ngày sử dụng để training, chiều thứ 3 là số thuộc tính dùng để training. Cách chia như thế để thuận tiện trong việc xây dựng mô hình, cắt dữ liệu ra để huấn luyện.

Các bước xử lý dữ liệu, chia tập dữ liệu đã xong, bước tiếp theo sẽ là xây dựng model học máy. Như đã trình bày ở đầu chương, mô hình bao gồm hai layer SimpleRNN xếp chồng lên nhau, sau đó nối với layer gồm 6 nơ ron liên kết đầy đủ (tương ứng với 6 thuộc tính cần dự đoán) đi qua hàm tác động *sigmoid* đóng vai trò là đầu ra của mô hình.

Tiếp theo ta thực hiện train model với 50 vòng lặp, batch size 4 và bộ dữ liệu tối ưu validation\_data đã chuẩn bị ở trên. Sau khi train xong, hệ thống sẽ lưu model có giá trị hàm mất mát thấp nhất trên tập tối ưu theo đường dẫn *‘model/best-RNN-model.h5’*.



Hình 25. Train mô hình (RNN)

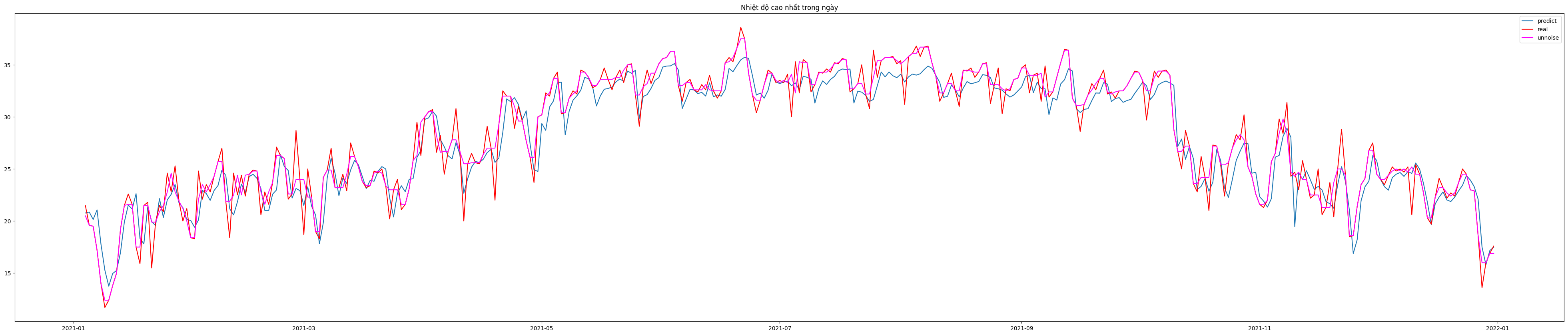


Hình 26. Màu xanh là giá trị train\_loss, màu đỏ là val\_loss (RNN)

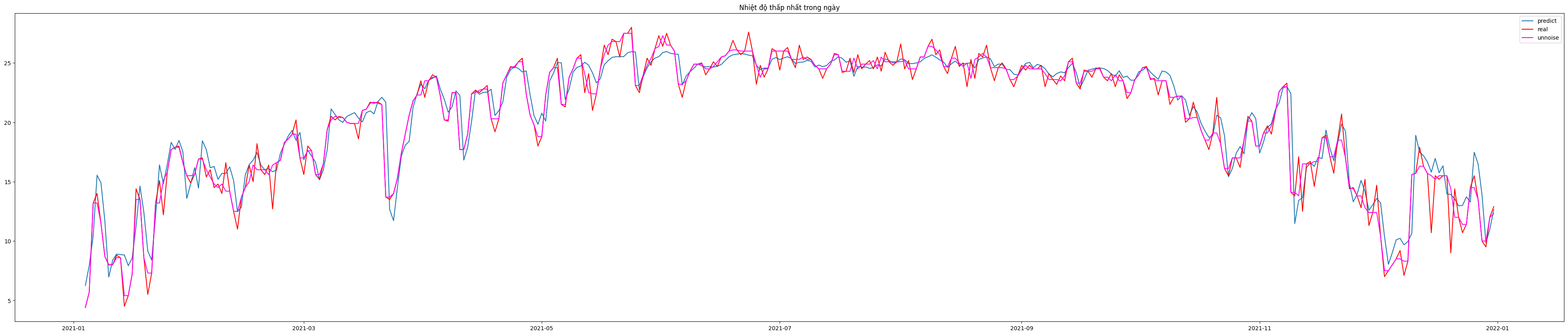
Sau khi huấn luyện và dự đoán, dưới đây là kết quả của số liệu dự đoán so với số liệu thực tế của các thông số khí tượng. Đường màu xanh là kết quả dự đoán, đường màu đỏ là số liệu thực tế, đường màu tím là số liệu đã giảm nhiễu.



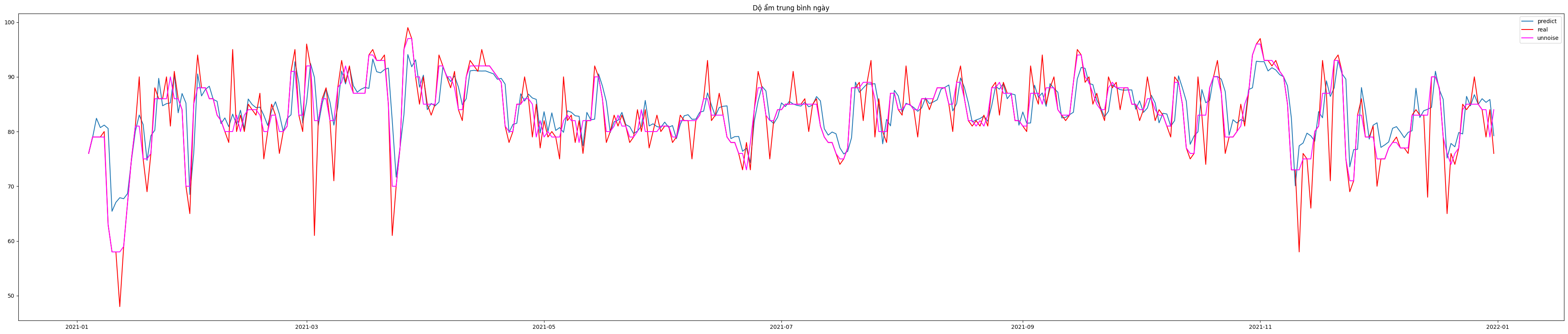
Hình 27. Nhiệt độ trung bình (RNN)



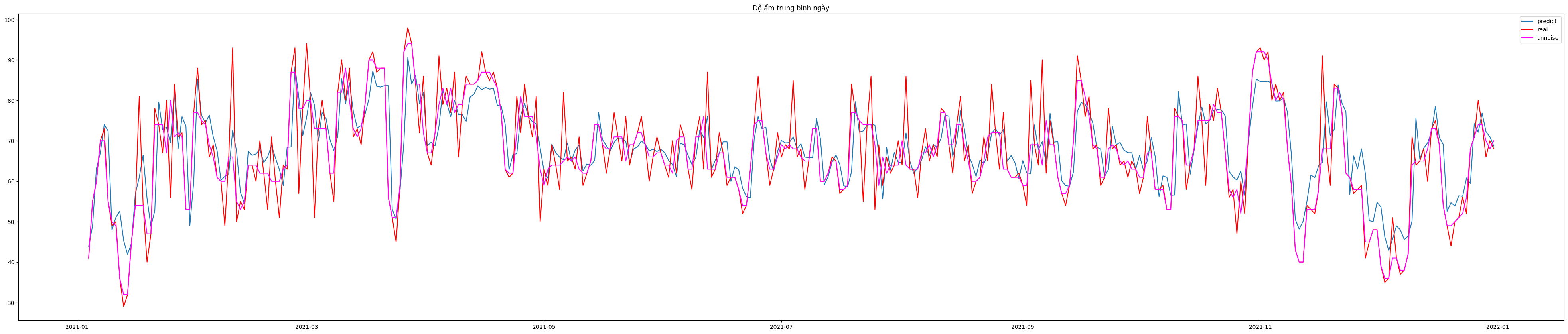
Hình 28. Nhiệt độ cao nhất (RNN)



Hình 29. Nhiệt độ thấp nhất (RNN)



Hình 30. Độ ẩm trung bình (RNN)



Hình 31. Độ ẩm thấp nhất (RNN)

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 32. Dự đoán khả năng có mưa (RNN)

### 3.2 Xây dựng mạng LSTM (Long short term memory)

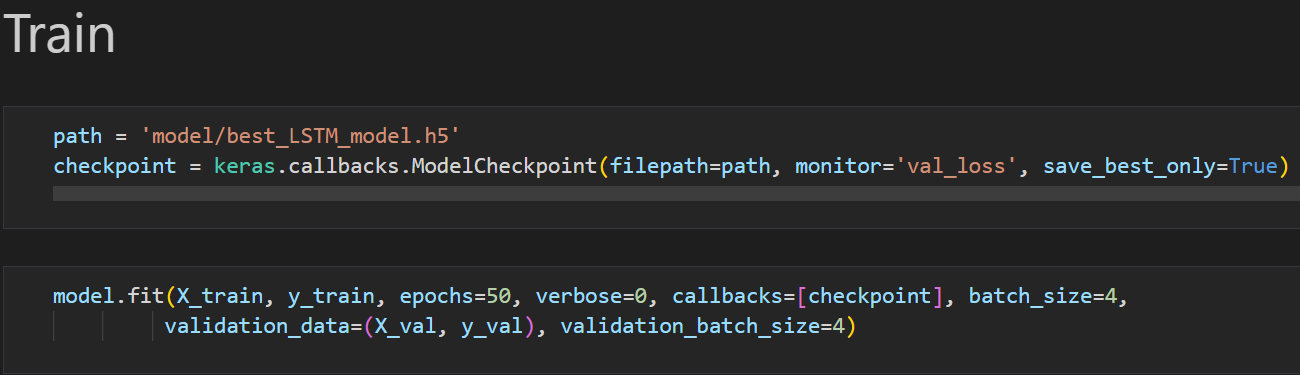
Về phần chuẩn bị dữ liệu, mô hình LSTM không khác so với RNN, chỉ có sự khác biệt nhỏ ở phần xây dựng model.

Ảnh có chứa văn bản

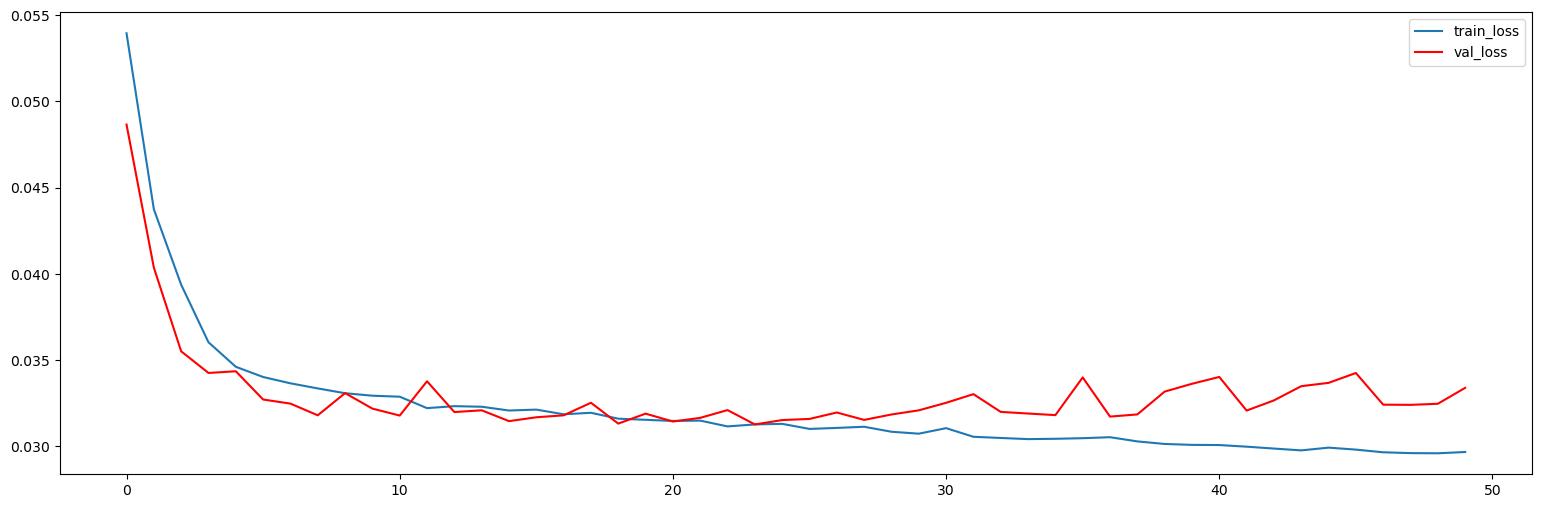
Mô tả được tạo tự động

Hình 33. Mô hình LSTM

Mô hình LSTM tương tự như mô hình RNN nêu trên, chỉ thay 2 layer đầu tiên bằng LSTM thay vì RNN. Chúng ta cũng train mô hình với 50 vòng lặp với thông số giống với khi train mô hình RNN ở trên



Hình 34. Train mô hình (LSTM)

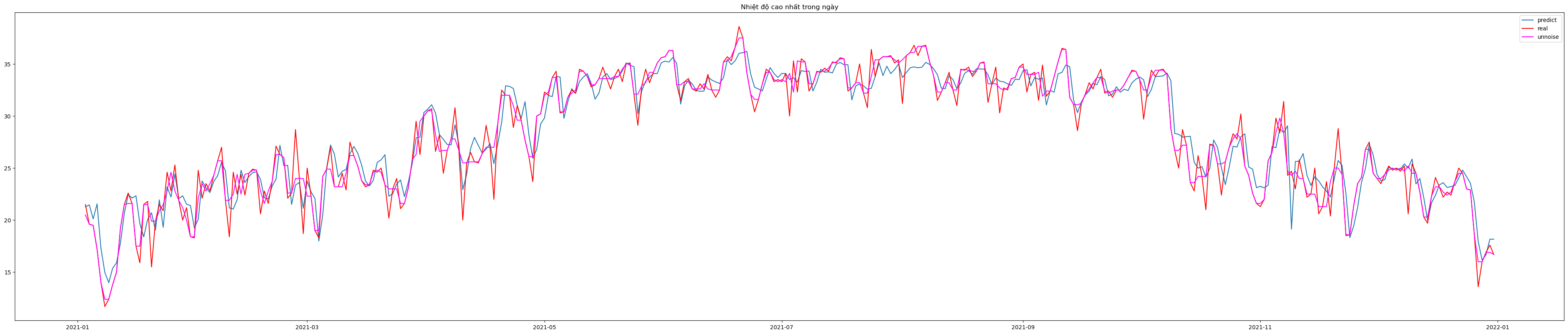


Hình 35. Đường màu xanh là thông số train\_loss, đường màu đỏ là val\_loss(LSTM)

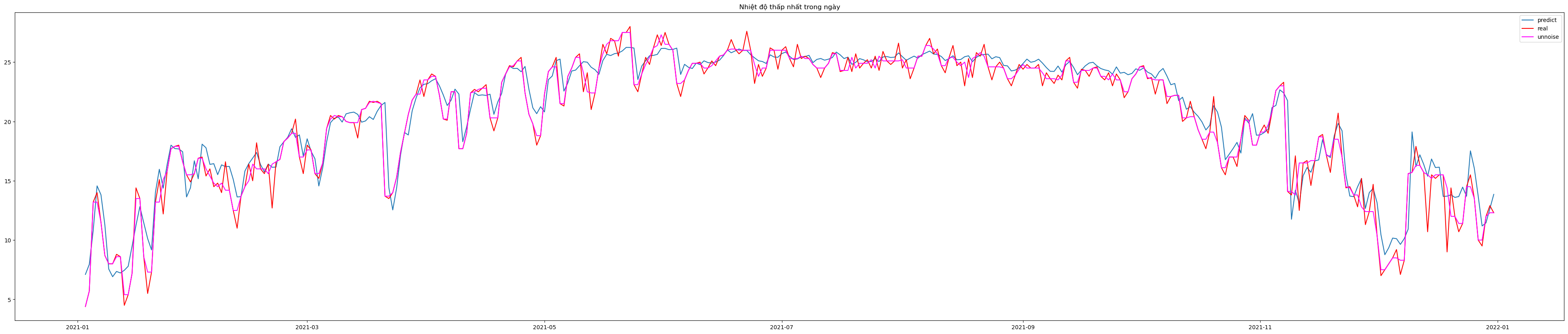
Sau khi huấn luyện và dự đoán, dưới đây là kết quả của số liệu dự đoán so với số liệu thực tế của các thông số khí tượng theo ngày. Đường màu xanh là kết quả dự đoán, đường màu đỏ là số liệu thực tế, đường màu tím là số liệu đã giảm nhiễu.



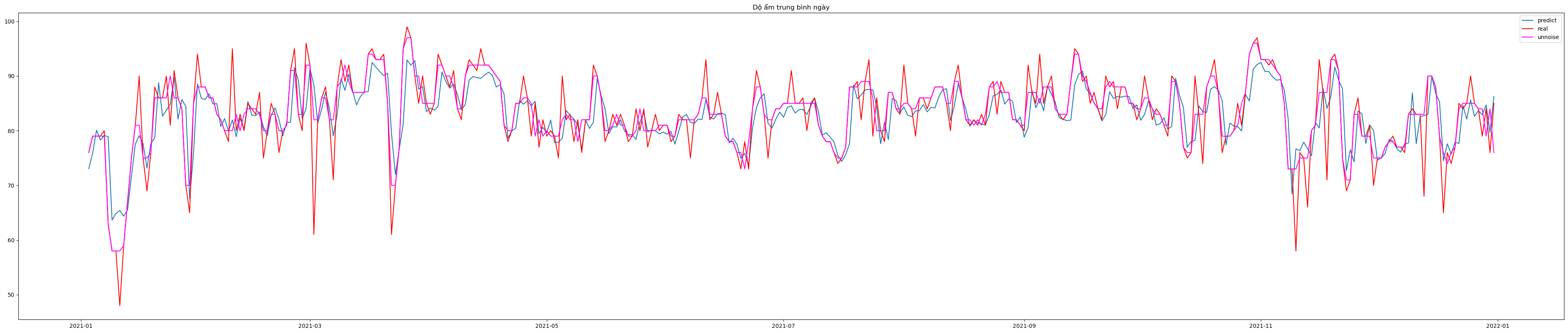
Hình 36. Nhiệt độ trung bình (LSTM)

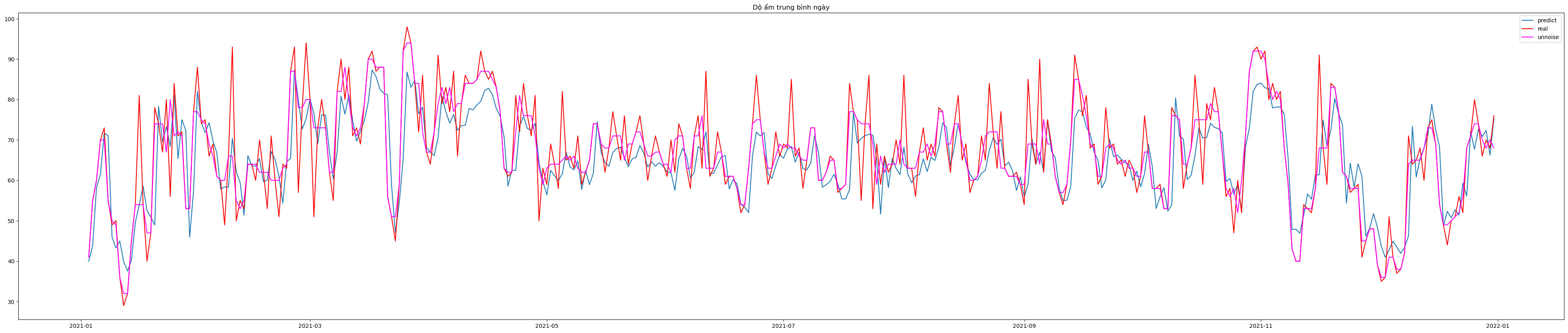


Hình 37. Nhiệt độ cao nhất (LSTM)



Hình 38. Nhiệt độ thấp nhất (LSTM)



Hình 39. Độ ẩm trung bình (LSTM) 

Hình 40. Độ ẩm thấp nhất (LSTM)

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 41. Dự đoán khả năng có mưa (LSTM)

## Kết quả

Dưới đây là bảng tổng hợp sai số dự đoán của mô hình RNN và LSTM, hàng cuối cùng là tỉ lệ dự đoán đúng khả năng có mưa.

Table

Description automatically generated

Hình 42. Sai số của mô hình RNN khi dùng 3 ngày dự đoán 1 ngày

Table

Description automatically generated

Hình 43. Sai số của mô hình LSTM khi dùng 3 ngày dự đoán 1 ngày

Table

Description automatically generated

Hình 44. Sai số của mô hình LSTM khi dùng 15 ngày dự đoán 1 ngày

Nhóm đã thực hiện dự đoán *‘n\_timepred’* ngày bằng cách cho kết quả dự đoán làm input để đoán ngày tiếp theo. Ví dụ: dùng ngày 1,2,3 để đoán ngày 4,5,6 bằng cách đưa ngày 1,2,3 vào mô hình thu được kết quả dự đoán của ngày 4. Sau đó tiếp tục đưa ngày 2,3 và 4 vừa thu được vào mô hình để lấy kết quả của ngày 5. Và như thế lặp lại cho đến ngày cuối cùng cần dự đoán. Hình dưới là kết quả của mô hình RNN sử dụng 3 ngày dự đoán 3 ngày, LSTM cũng cho kết quả tương tự.

Table

Description automatically generated

Hình 45. Sai số của mô hình RNN khi dùng 3 ngày dự đoán 3 ngày

# ĐÁNH GIÁ

Qua những biểu đồ trực quan cho các số liệu của RNN và LSTM, ta có thể thấy kết quả dự đoán không khác nhau nhiều. Dự báo thời tiết là vấn đề cần dùng time step nhỏ, trải qua quá trình thử nghiệm, nhóm em thấy với n\_timestep = 3 cho kết quả tối ưu nhất (để trực quan thì nhóm có so sánh 2 kết quả n\_timestep = 3 và n\_timestep = 15). LSTM được sinh ra để giải quyết vấn đề phụ thuộc xa, vanishing gradient của RNN khi số timestep lớn. Như đã trình bày ở trên, việc dự báo với số timestep nhỏ (n\_timestep = 3) dẫn đến kết quả gần giống nhau giữa RNN và LSTM do không thể phát huy được ưu điểm của LSTM ở long-term memory, ngược lại, việc tính toán nhiều hơn khiến tốc độ của LSTM trong quá trình huấn luyện mô hình lâu hơn RNN một chút.

Nhóm cũng đã thử dùng một số mô hình đơn giản khác, phân đoạn dữ liệu thành dạng low, medium, high để dự đoán có mưa hay không với mô hình Naïve Bayes, tuy nhiên kết quả không khả quan như khi dùng RNN và LSTM. Sau khi nhóm chúng em trao đổi đã kết luận nguyên nhân có thể do việc phân đoạn dữ liệu không hợp lý, hoặc dùng các trường dữ liệu không hoàn toàn độc lập với nhau, vì vậy nhóm đã bỏ mô hình Naïve Bayes.

# Các khó khăn gặp phải và hướng phát triển trong tương lai

Trong quá trình thực hiện đồ án môn học, nhóm đã gặp phải một số khó khăn trong việc tiếp cận những kiến thức mới, chưa có kinh nghiệm trong việc xử lý dữ liệu, xây dựng mô hình, đánh giá mô hình. Nhưng bằng việc cùng nhau trao đổi, đọc và tham khảo tài liệu bên ngoài, nghe tư vấn từ những người làm khí tượng trong thực tế, nhóm đã tiến hành thử nghiệm rất nhiều lần, tối ưu kết quả đạt được từng chút một. Cuối cùng, nhóm đã thu được kết quả với tỉ lệ chính xác khá cao, có thể ứng dụng trong thực tế.

Về hướng phát triển trong tương lai, nếu có điều kiện có được tập dữ liệu phong phú hơn về khí tượng như là có thêm dữ liệu về mực nước, mây, áp suất,... nhóm sẽ có thể xây dựng mô hình một cách chính xác hơn và có thể phát triển một ứng dụng nhỏ về dự báo thời tiết.

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | N. T. Huyen, “[Recurrent Neural Network: Từ RNN đến LSTM](https://viblo.asia/p/recurrent-neural-network-tu-rnn-den-lstm-gGJ597z1ZX2),” 24 6 2021. [Trực tuyến]. Available: https://viblo.asia/p/recurrent-neural-network-tu-rnn-den-lstm-gGJ597z1ZX2. |
| [2] | "[Intuition of Adam Optimizer](https://www.geeksforgeeks.org/intuition-of-adam-optimizer/?ref=rp)," [Online]. Available: https://www.geeksforgeeks.org/intuition-of-adam-optimizer/?ref=rp. |
| [3] | S. Merity, "[Explaining and illustrating orthogonal initialization for recurrent neural networks](https://smerity.com/articles/2016/orthogonal_init.html)," [Online]. Available: https://smerity.com/articles/2016/orthogonal\_init.html. |
| [4] | S. A. Afshine Amidi, "[Recurrent Neural Networks cheatsheet](https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks)," [Online]. Available: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks. |