TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Ứng dụng dự đoán năng lượng điện gió

DOÃN ANH TUẤN HUY
huy.dat183929@sis.hust.edu.vn

Ngành Công nghệ thông tin
Chuyên ngành Công nghệ thông tin

Giảng viên hướng dẫn: TS. Trịnh Anh Phúc _____

Bộ môn: Khoa học máy tính
Trường: Công nghệ thông tin – Truyền thông

HÀ NỘI, 7/2022

Lời cảm ơn

Lời đầu tiên, em xin được gửi lời cảm ơn sâu sắc đến TS. Trịnh Anh Phúc. Thầy đã hướng dẫn, giúp đỡ em nhiều trong quá trình nghiên cứu, thực hiện và hoàn thiện DATN này.

Em cũng xin cảm ơn tới tất cả thầy cô đã chỉ dạy cho em những kiến thức quý báu trong suốt quá trình học tập, cung cấp một hành trang thiết thực cho tương lai sắp tới.

Cuối cùng, em xin cảm ơn gia đình, bạn bè, anh chị đồng nghiệp- nhưng người đã luôn bên canh, cổ vũ, động viên tinh thần, giúp em vượt qua những khó khăn trong suốt quá trình học tập tại đại học Bách Khoa Hà Nội.

Do thời gian có hạn, cũng như kiến thức chuyên môn còn chưa tốt, DATN của em không tránh khỏi những thiếu sót và hạn chế. Em rất mong nhận được những đánh giá, nhận xét từ thầy cô và các bạn để có thể rút ra được những kinh nghiệm làm việc một cách vững vàng hơn.

Em xin chân thành cảm ơn !

Tóm tắt nội dung đồ án

Năng lượng gió được coi là nguồn năng lượng xanh, sạch, dồi dào và phong phú nhất hiện nay, nó có mặt ở mọi nơi, cả ban ngày lẫn ban đêm. Nó được sử dụng từ hàng trăm năm nay. Con người đã sử dụng năng lượng gió để di chuyển thuyền buồm hay khinh khí cầu, ngoài ra năng lượng gió còn được sử dụng để tạo công cơ học nhờ vào các cối xay gió. Ý tưởng dùng năng lượng gió để sản xuất điện hình thành ngay sau khi phát minh ra điện và máy phát điện. Tuy vậy năng lượng gió không thể sản xuất được 1 cách nhất quán dẫn đến khả năng thay đổi cao. Sự biến đổi như vậy có thể gây ra những thách thức lớn đối với việc kết hợp năng lượng gió vào hệ thống lưới điện. Để duy trì sự cân bằng giữa sản xuất và tiêu thụ điện, sự biến động của năng lượng gió đòi hỏi phải thay thế điện từ các nguồn khác mà có thể không có sẵn trong thời gian ngắn.

Nắm bắt được điều đó, đồ án tốt nghiệp này hướng tới việc có thể ước lượng chính xác được nguồn cung cấp năng lượng gió trong 1 trang trại ở các quy mô thời gian khác nhau.

Mục lục

Danh mục hình vẽ

Danh mục bảng

Danh mục công thức

Danh mục các từ viết tắt

GCN	Graph Convolutional Network Mạng đồ thị tích chập
LSTM	Long short term memory Bộ nhớ dài ngắn
MLP	Multi layer perceptron Mạng nơ ron truyền thẳng nhiều lớp
CNN	Convolution neural network Mạng nơ ro tích chập
ML	Machine learning Máy học
DL	Deep learning Học sâu

Danh mục thuật ngữ

Model	Mô hình
Dataset	Tập dữ liệu
Train	“ dạy học”

Chương 1 . Giới thiệu đề tài

Trong quá trình xây dựng và phát triển đề tài, bước khởi đầu sẽ là xác định được những vẫn đề, xác định được mục tiêu đề tài và định hướng phát triển của dự án. Chương 1 sẽ làm rõ các yêu cầu đặt ra cũng như các tính năng sẽ xây dựng được.

1.1 Đặt vấn đề

Hiện nay xu hướng toàn cầu hóa đang tập trung hướng tới mục tiêu bảo vệ môi trường. Khi mà tình trạng sử dụng các nguồn nguyên liệu từ than đá, dầu mỏ, khí đốt,… đang gây ra những ảnh hưởng vô cùng nghiêm trọng tới môi trường. Đặc biệt là những hiện tượng hiệu ứng nhà kính, mưa axit, ô nhiễm không khí, ô nhiễm nước, đất,… đã và đang tác động trực tiếp đến đời sống sức khỏe của người dân. Theo đó, một dang năng lượng mới được phát hiện ra. Đó chính là, năng lượng sạch. Chính vì thế năng lượng sạch đang không ngừng được chú trọng và sử dụng rộng rãi. Đặc biệt, Việt Nam là một trong các khu vực địa lý có điều kiện tự nhiên thuận lợi nhất để phát triển mạnh các nguồn năng lượng sạch. Vậy năng lượng sạch là gì?

Năng lượng sạch là dạng năng lượng mà trong quá trình sinh công bản thân nó không tạo ra những chất thải độc hại gây ảnh hưởng cho môi trường xung quanh. Thông thường, các nguồn năng lượng sạch đều có sẵn từ thiên nhiên hoặc là chế phẩm của các sản phẩm tự nhiên nên không gây ô nhiễm, ít bị cạn kiệt.Năng lượng sạch thường được cung cấp và sản xuất từ năng lượng hóa thạch (than đá, dầu mỏ, khí đốt) và năng lượng hạt nhân. Các nguồn năng lượng phải dựa trên cơ sở sử dụng công nghệ chuyển hóa năng lượng sạch. Đảm bảo thân thiệt đối với môi trường trong suốt quá trình sản xuất. Đồng thời, các nơi sản xuất năng lượng sạch cũng phải đảm bảo quy trình thực hiện đúng với quy định bảo vệ môi trường.

Hiện nay trên thế giới, ngành công nghiệp năng lượng sạch vô cùng phát triển. Ngày càng có nhiều loại năng lượng sạch được phát hiện và thử nghiệm. Điển hình như: Pin nhiên liệu, năng lượng mặt trời, nước, dầu thực vật phế thải, năng lượng tuyết, năng lượng lên men, năng lượng địa nhiệt, khí Metan Hydrate,…Trong đó có cả năng lượng gió.

Năng lượng gió là động năng của không khí di chuyển trong bầu khí quyển của trái đất. Năng lượng gió là một hình thức gián tiếp của năng lượng mặt trời. Năng lượng gió được mô tả như một quá trình , nó được sử dụng để phát ra năng lượng cơ hoặc điện. Tuabin gió sẽ chuyển đổi từ động lực gió thành năng lượng cơ.

Năng lượng gió được coi là nguồn năng lượng xanh dồi dào và phong phú nhất hiện nay, nó có mặt ở mọi nơi, cả ban ngày lẫn ban đêm. Người ta sử dụng sức gió để quay các tua bin phát điện để sử dụng trong cuộc sống. Đây là nguồn năng lượng với rất nhiều ưu điểm: là năng lượng tái tạo sạch,ít gây ô nhiễm môi trường, nguồn nhiên liệu miễn phí, có hiệu suất cao, hiệu quả về mặt chi phí, tốn ít diện tích xây dựng,… Ngoài ra nó còn góp phần giảm thiểu các tác động tiêu cực đến môi trường. Theo đó, khi sử dụng loại năng lượng này sẽ giúp con người giảm bớt lượng khí thải ảnh hưởng xấu đến môi trường sống xung quanh ta, đồng thời đảm bảo sức khỏe của con người. Đối với đất nước: sử dụng nguồn năng lượng sạch như trên liên quan trực tiếp đến sự hình thành lối sống văn minh và phát triển bền vững của quốc gia. Ngoài ra, nó còn tạo tiềm năng kinh tế vùng và phát triển mạnh mẽ an ninh năng lượng của cả đất nước.

Hiện nay tại Việt Nam, với điều kiện địa lý thuận lợi bờ biển dài, lượng gió nhiều và phân bổ đều quanh năm, đây sẽ là một dạng năng lượng được chú trọng phát triển ở hiện tại và tương lai.

Dù là một loại nguồn năng lượng tái tạo sạch và an toàn, nhưng năng lượng gió không thể được sản xuất một cách nhất quán, dẫn đến khả năng thay đổi cao. Sự biến đổi như vậy có thể gây ra những thách thức lớn đối với việc kết hợp năng lượng gió vào hệ thống lưới điện. Để duy trì sự cân bằng giữa sản xuất và tiêu thụ điện, sự biến động của năng lượng gió đòi hỏi phải thay thế điện từ các nguồn khác mà có thể không có sẵn trong thời gian ngắn.

Vì vậy, dự báo năng lượng gió (Wind Power Forecast - WPF) đã được công nhận rộng rãi là một trong những vấn đề quan trọng nhất trong tích hợp và vận hành điện gió.

Trong số 20 thị trường lớn nhất trên thế giới, chỉ riêng châu Âu đã có 13 nước với Đức là nước dẫn đầu về công suất của các nhà máy dùng năng lượng gió với khoảng cách xa so với các nước còn lại. Tại Đức, Đan Mạch và Tây Ban Nha việc phát triển năng lượng gió liên tục trong nhiều năm qua được nâng đỡ bằng quyết tâm chính trị. Nhờ vào đó mà một ngành công nghiệp mới đã phát triển tại 3 quốc gia này. Công nghệ Đức (bên cạnh các phát triển mới từ Đan Mạch và Tây Ban Nha) đã được sử dụng trên thị trường nhiều hơn trong những năm vừa qua.

Dưới đây là hình mẫu cỗ tua bin điện gió lớn nhất thế giới (năm 2014) V164-8.0 MW xây dựng ở trung tâm thử nghiệm quốc gia Đan Mạch. Cỗ máy dài 20m, rộng 8m, cao 8m, trọng lượng khoảng 390 Tấn. Mỗi cỗ V164-8.0MW có thể cung cấp cho7500 căn hộ dân châu Âu.

Hình 1.1 Ảnh thực tuabin gió

Ở Việt Nam, điện gió thuộc nhóm công nghiệp năng lượng mới nổi, được nhập cuộc theo sự phát triển nguồn năng lượng tái tạo chung của thế giới, sự nhập khẩu khoa học kỹ thuật, đồng thời đáp ứng nhu cầu phát triển nguồn năng lượng khi các nguồn thủy điện lớn đã khai thác hết, các thủy điện nhỏ không đảm bảo lợi ích mang lại so với thiệt hại môi trường mà nó gây ra. Mặt khác Việt Nam có tiềm năng lớn về năng lượng mặt trời và gió, do ở gần xích đạo và tồn tại những vùng khô nắng nhiều và gió có hướng tương đối ổn định như các tỉnh nam Trung Bộ. Vì thế điện gió cùng với điện mặt trời đang được nhà nước Việt Nam khuyến khích phát triển.

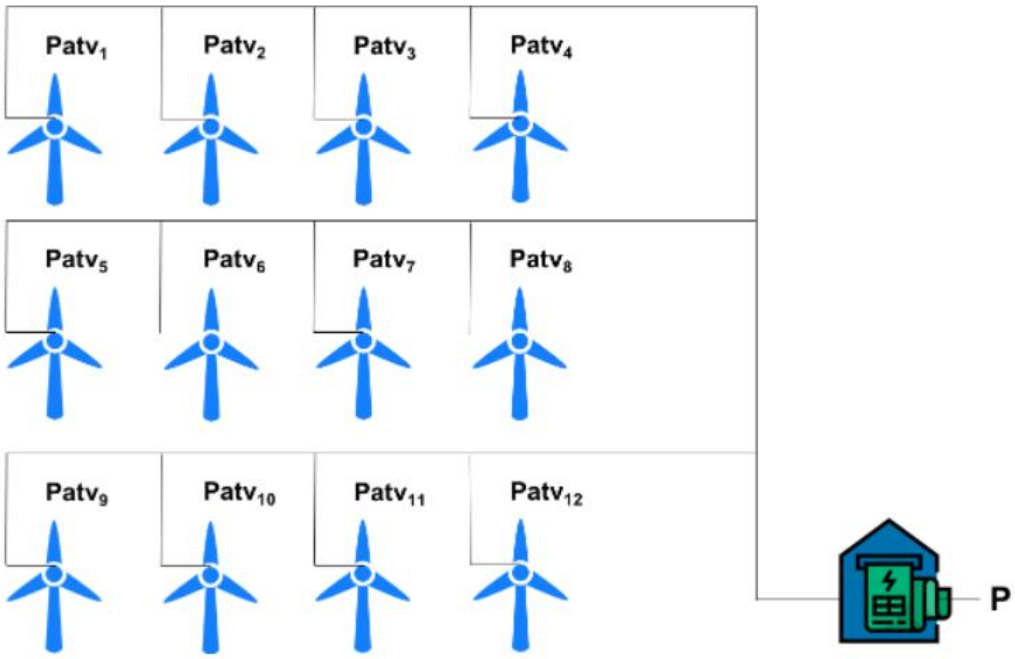
1 số tỉnh thành nổi tiếng về điện gió như : Ninh Thuận, Bình Thuận, Quảng Trị, Bạc Liêu….. Dưới đây là hình ảnh về điện gió Đầm Nại thuộc huyện Thuận Bắc, Ninh Thuận , 02/2022:

Hình 1.2 Điện gió Đầm Nại

1.2 Mục tiêu và phạm vi đề tài

Với những gì đã trình bày trong phần 1.1, em đặt ra mục tiêu xây dựng được các model đưa ra được Target variable và tiến hành so sánh.

Dưới đây là mô hình minh họa cho bài toán đề ra:



Hình 1.3: Mô hình dự án

Theo hình vẽ, mỗi tua bin gió sẽ tạo ra được năng lượng gió riêng biệt và công suất đầu ra của trang trại gió là tổng của tất cả các tuabin gió. Nói cách khác, tại thời điểm t, công suất đầu ra của trang trại điện gió là :

$$P = \sum_{i}^{\infty} Patv_i$$

1.3.Định hướng giải pháp

Sau khi xác định rõ được mục tiêu, em đã đưa ra giải pháp là sẽ xây dựng các model mà có thể train cùng 1 lúc các tuabin hoặc mỗi tuabin sẽ sử dụng 1 model.

Để train tất cả tuabin cùng 1 lúc, ta sẽ sử dụng model Graph

Với việc train mỗi tuabin 1 model,ta sẽ sử dụng các model khác như CNN, LSTM..

1.4.Bố cục đồ án

Nội dung của báo cáo đồ án tốt nghiệp được chia thành 6 chương chính,với nội dung được tóm tắt như sau:

Chương 1: Giới thiệu đề tài

Ở chương này, các yêu cầu ban đầu của bài toán được nêu lên, bao gồm những vấn đề, các chức năng chính, định hướng giải pháp

Chương 2 : Thu thập,tiền xử lí và phân tích dataset

Chương này sẽ tập trung vào dataset mà ta sẽ train trong suốt quá trình.

Chương 3: Xây dựng mô hình

Sau khi khảo sát qua dataset đã có, cần đưa ra 1 vài model để có thể train với bộ dataset đó .

Chương 4: Công nghệ sử dụng

Ở chương này tổng hợp các công nghệ được sử dụng trong dự án: các framework, các library…

Chương 5: Đánh giá kết quả

Trong chương này sẽ đưa ra kết quả thu được và tiến hành so sánh

Chương 6: Kết luận và hướng phát triển

Cuối cùng, phần kết luận sẽ tổng kết lại quá trình xây dựng đồ án, những ưu nhược điểm của hệ thống, những gì đã và chưa làm được cũng như hướng phát triển mở rộng trong tương lai.

Chương 2 . Thu thập, tiền xử lí và phân tích dữ liệu

Chương này sẽ tập trung vào dataset mà ta sẽ train trong suốt quá trình.

Dataset là một tập hợp dữ liệu. Nói cách khác, dataset tương ứng với nội dung của một bảng cơ sở dữ liệu hoặc một ma trận dữ liệu thống kê, trong đó mỗi cột của bảng đại diện cho một biến cụ thể và mỗi hàng tương ứng với một thành viên nhất định của tập dữ liệu được đề cập.

Trong các dự án ML,DL, chúng ta cần một tập dữ liệu đào tạo. Đây là tập dữ liệu thực tế được sử dụng để huấn luyện mô hình thực hiện các hành động khác nhau.

Trong quá trình phát triển AI, chúng ta luôn dựa vào dữ liệu. Từ đào tạo, điều chỉnh, lựa chọn mô hình đến kiểm tra, chúng ta sử dụng ba bộ dữ liệu khác nhau: bộ đào tạo (training set), bộ xác thực (validation set) và bộ thử nghiệm (testing set). Validation set được sử dụng để chọn và điều chỉnh mô hình machine learning cuối cùng.

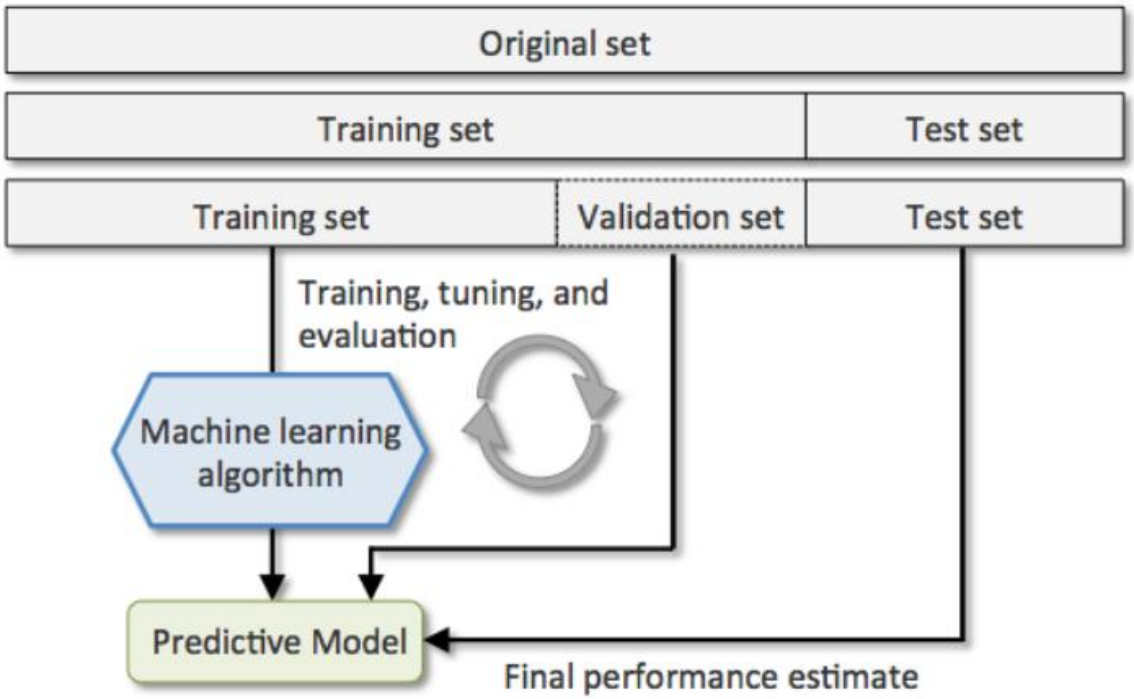
Có 3 loại dataset cho ML, DL:

- Training set: là tập dùng để huấn luyện thuật toán hiểu cách áp dụng các khái niệm như mạng nơ-ron, để học và tạo ra kết quả. Nó bao gồm cả dữ liệu đầu vào và đầu ra dự kiến. Tập hợp đào tạo chiếm phần lớn trong tổng số dữ liệu, khoảng 60%. Trong thử nghiệm, các mô hình phù hợp với các thông số trong một quá trình được gọi là điều chỉnh trọng lượng (adjusting weights).

- Validation set: Để mô hình được đào tạo, nó cần được đánh giá định kỳ và đó chính xác là mục đích của bộ xác thực (valiation set). Thông qua việc tính toán tổn thất (tức là tỷ lệ lỗi) mà mô hình mang lại trên bộ xác thực tại bất kỳ điểm nào đã cho, chúng ta có thể biết độ chính xác của nó. Đây là bản chất của đào tạo. Sau đó, mô hình sẽ điều chỉnh các tham số của nó dựa trên kết quả đánh giá thường xuyên trên valiation set. Thông thường, validation set chiếm 20% phần lớn dữ liệu được sử dụng

- Test set: Tập dữ liệu thử nghiệm được sử dụng để đánh giá thuật toán của bạn được đào tạo tốt như thế nào với tập dữ liệu đào tạo. Trong các dự án AI, chúng ta không thể sử dụng tập dữ liệu đào tạo trong giai đoạn thử nghiệm vì thuật toán sẽ biết trước kết quả mong đợi không phải là mục tiêu của chúng tôi. Test set chiếm khoảng 20% dữ liệu.

Dưới đây là việc sử dụng dataset trong ML,DL:



Hình 2.1 Phân bố dataset

2.1.Thu thập dữ liệu

Ở đây dataset được em thu thập từ cuộc thi thường niên: https://aistudio.baidu.com/aistudio/competition/detail/152/0/datasets

Mô tả : Dữ liệu giám sát theo ngữ cảnh từ hệ thống Kiểm soát Giám sát và Thu thập Dữ liệu (SCADA) trên các tuabin gió của một trang trại điện gió sẽ được công bố.Dữ liệu SCADA được lấy mẫu cứ 10 phút một lần từ mỗi tuabin gió trong trang trại điện gió (bao gồm 134 tuabin gió) thuộc sở hữu của Tập đoàn Longyuan Power. Ngoài ra, vị trí tương đối của tất cả các tuabin gió trong trang trại gió sẽ được đưa ra để mô tả sự phân bố theo không gian của các tuabin gió.

Kích thước dữ liệu: là 1 file csv khoảng hơn 300MB

2.2.Tiền xử lí dữ liệu

Tiền xử lý dữ liệu là một bước rất quan trọng trong việc giải quyết bất kỳ vấn đề nào trong lĩnh vực Học Máy. Hầu hết các bộ dữ liệu được sử dụng trong các vấn đề liên quan đến Học Máy cần được xử lý, làm sạch và biến đổi trước khi một thuật toán Học Máy có thể được huấn luyện trên những bộ dữ liệu này.

Hiện nay có rất nhiều kỹ thuật tiền xử lí dữ liệu phổ biến:

- Xử lí dữ liệu bị khuyết (missing data): khi gặp missing data, ta thường có các cách giải quyết sau:
    - Tìm kiếm missing trong dataset
    - Xác định giá trị missing trong dataset
    - Removing data
    - Data imputation (có thể thay bằng các giá trị -1,-99,.. mean, medium,..) tùy trường hợp
- Co giãn dữ liệu (Scaling data): thông thường có 2 cách là Normalization và Standardization:
    - Normalization: là phương pháp scale dữ liệu từ miền giá trị bất kì sang miền giá trị nằm trong khoảng 0 đến 1. Phương pháp này yêu cầu chúng ta cần xác định được giá trị lớn nhất (max) và giá trị nhỏ nhất (min) của dữ liệu. Giá trị được normalize theo công thức sau: $y = (\mathrm{x} - \min)/(\max - \min)$ với y là biến sau normalize, x là biến trước normalize.
    - Standardization: là việc scale dữ liệu về một phân bố trong đó giá trị trung bình của các quan sát bằng 0 và độ lệch chuẩn = 1. Kỹ thuật này còn được gọi là "whitening.". Nhờ việc chuẩn hóa, các thuật toán như linear regression, logistic regression được cải thiện. Công thức như sau (x ngang, theta lần lượt là kỳ vọng và phương sai của thành phần đó trên toàn bộ training data)

$$x' = \frac{x - \bar{x}}{\sigma}$$

Công thức 2.1. Data Standardization

- Mã hóa các biến nhóm (encoding categorical variables) : đây là trường hợp dữ liệu không ở dạng số (not a number), máy tính sẽ không hiểu và làm việc được với categorical variable, do vậy ta phải encoding để đưa nó về dạng khác để có thể đưa vào model.

Trong đồ án nàu, do là dataset được cung cấp bởi nhà tổ chức cuộc thi nên dữ liệu phần nào khá rõ ràng, tuy nhiên ta vẫn phải xử lí, cụ thể:

Trong dataset, có 1 số thời điểm bị thiếu ví dụ như:

30	176	11:30	0	18.89	-258.02	-258.02	365.57	0	0	0	0	0
30	176	11:40	0	19	-259.54	-259.54	367.72	0	0	0	0	0
30	176	11:50	0	19.96	-272.67	-272.67	386.31	0	0	0	0	0
30	176	12:00										
30	176	12:10										
30	176	12:20										
30	176	12:30										
30	176	12:40										
30	176	12:50										
30	176	13:00										
30	176	13:10										
30	176	13:20										
30	176	13:30										
30	176	13:40										
30	176	13:50										
30	176	14:00										
30	176	14:10										
30	176	14:20										
30	176	14:30	6.95	14.63	20.62	21.67	387.02	90.17	90.17	90.2	0	0
30	176	14:40	6.36	19.99	20.71	21.91	387.02	90.17	90.17	90.2	0	0
30	176	14:50	6.43	12.28	20.77	22.47	397.31	90.17	90.17	90.2	0	0
30	176	15:00	6.96	-7.06	20.77	23.08	406.16	90.17	90.17	90.2	0	0
30	176	15:10	7.17	-11.67	20.67	23.49	406.16	90.17	90.17	90.2	0	0

Hình 2.2 . Dữ liệu cần xử lí

Cách xử lý: xử lý đơn giản bằng cách ô nào không phải là số (not a number) thì ta sẽ duplicate ô trước hoặc ô sau nó (chính là việc missing data và t sử dụng data imputation).

Dưới đây là dòng code đơn giản thực hiện nhiệm vụ này:

```
1  import os
2  import pandas as pd
3
4  class TestData:
5      def __init__(self, path):
6          self.data = pd.read_csv(path)
7      def get_data(self, tid):
8          return self.data[self.data['TurbID'] == tid]['Patv']
```

Hình 2.3. Dòng xử lí data

2.3.Phân tích dữ liệu

Đây là hình ảnh về những cột và hàng đầu tiên của dataset:

TurbID	Day	Tmstamp	Wspd	Wdir	Etmp	Itmp	Ndir	Pab1	Pab2	Pab3	Prtv	Patv
1	1	0:00										
1	1	0:10	6.17	-3.99	30.73	41.8	25.92	1	1	1	-0.25	494.66
1	1	0:20	6.27	-2.18	30.6	41.63	20.91	1	1	1	-0.24	509.76
1	1	0:30	6.42	-0.73	30.52	41.52	20.91	1	1	1	-0.26	542.53
1	1	0:40	6.25	0.89	30.49	41.38	20.91	1	1	1	-0.23	509.36
1	1	0:50	6.1	-1.03	30.47	41.22	20.91	1	1	1	-0.27	482.21
1	1	1:00	6.77	1.07	30.31	41.19	20.91	1	1	1	-0.23	584.75
1	1	1:10	6.7	-2.8	30.24	41	20.91	1	1	1	-0.23	557.98
1	1	1:20	6.44	-3.46	30.13	40.91	20.91	1	1	1	-0.21	503.94
1	1	1:30	6.25	-3.15	29.97	40.72	20.91	1	1	1	-0.26	463.37
1	1	1:40	6.87	-4.58	29.87	40.7	20.91	1	1	1	-0.25	577.47
1	1	1:50	6.6	-2.17	29.75	40.62	16.87	1	1	1	-0.25	538.85
1	1	2:00	6.71	-1.42	29.68	40.47	13.32	1	1	1	-0.26	559.37
1	1	2:10	6.86	-3.85	29.61	40.44	13.32	1	1	1	-0.23	585.51
1	1	2:20	6.88	-1.31	29.68	40.49	13.32	1	1	1	-0.26	608.83
1	1	2:30	7.1	-0.57	29.7	40.44	13.32	1	1	1	-0.28	675.43
1	1	2:40	7.16	-2.52	29.7	40.4	7.92	1	1	1	-0.25	716.68
1	1	2:50	6.8	-3.16	29.7	40.48	2.87	1	1	1	-0.3	629.83
1	1	3:00	6.85	3.03	29.7	40.53	-1.86	1	1	1	-0.26	651.13
1	1	3:10	6.85	-0.64	29.75	40.6	2.66	1	1	1	-0.27	677.56
1	1	3:20	6.57	-3.02	29.89	40.76	5.73	1	1	1	-0.27	619.05
1	1	3:30	6.36	-1.6	29.83	40.86	5.73	1	1	1	-0.31	576.73
1	1	3:40	6.88	-1.41	29.79	40.83	1.42	1	1	1	-0.28	665.23
1	1	3:50	7.32	-0.7	29.79	40.93	-1.86	1	1	1	-0.26	761.74
1	1	4:00	7.92	-1.04	29.81	40.8	-7.79	1	1	1	-0.25	898.34
1	1	4:10	7.73	2.44	29.8	40.82	-9.46	1	1	1	-0.32	867.22

Hình 2.4. Dataset

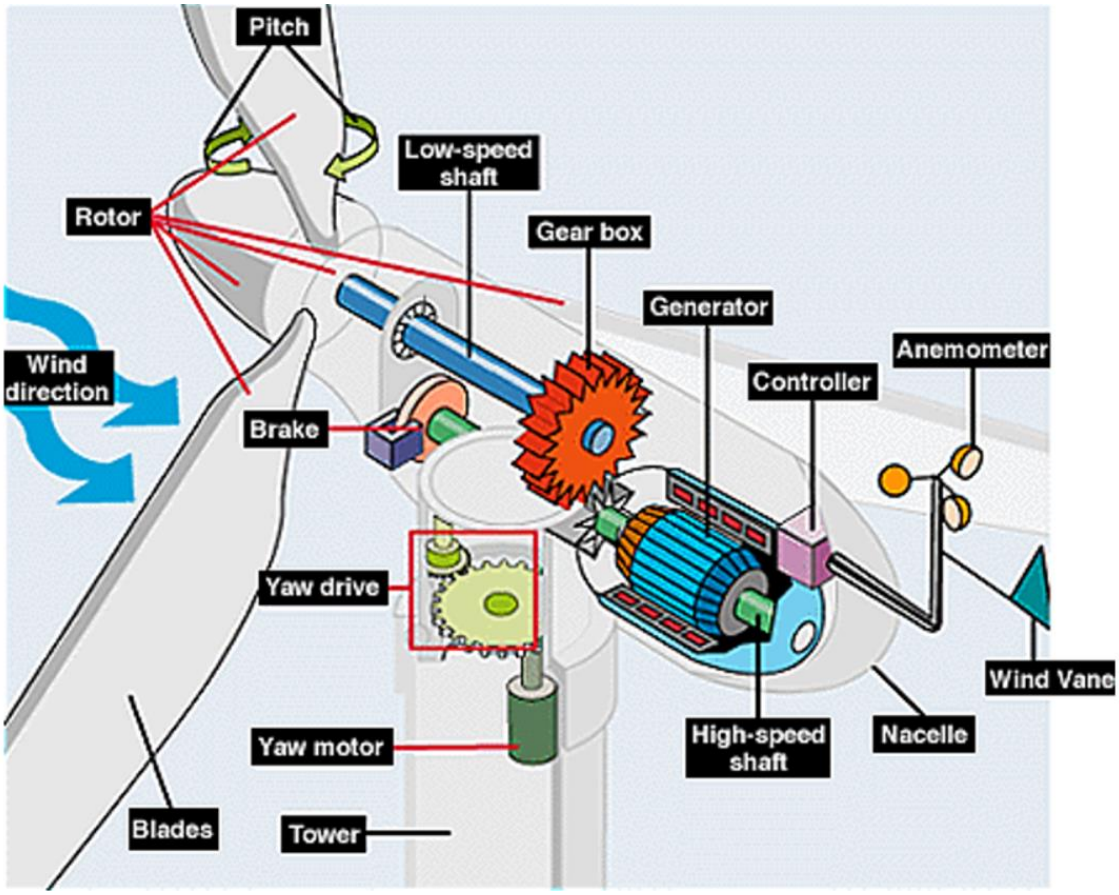
Đây là bảng ý nghĩa về cột dữ liệu trong dataset:

STT	Tên	Ý nghĩa
1	TurbID	ID của tuabin (wind turbine ID)
2	Day	Số thứ tự ngày (day of the record)
3	Tmstamp	Mốc thời gian khởi tạo (created time of the record)

STT	Tên	Ý nghĩa
4	Wspd (m/s)	Tốc độ gió được đo bởi máy đo gió (The wind speed recorded by the anemometer)
5	Wdir (°)	Góc giữa hướng gió và ví trí của nacelle (The angle between the wind direction and the position of turbine nacelle)
6	Etmp (℃)	Nhiệt độ của môi trường xung quanh(Temperature of the surounding environment)
7	Itmp (℃)	Nhiệt độ bên trong nacelle của tuabin(Temperature inside the turbine nacelle)
8	Ndir (°)	Hướng nacelle, tức là góc yaw của tuabin （ Nacelle direction,i.e., the yaw angle of the nacelle ）
9	Pab1 (°)	Góc pitch của cánh quạt 1 (Pitch angle of blade 1)
10	Pab2 (°)	Góc pitch của cánh quạt 2 (Pitch angle of blade 2)
11	Pab3 (°)	Góc pitch của cánh quạt 3(Pitch angle of blade 3)
12	Prtv (kW)	Công suất phản kháng (Reactive power)
13	Patv (kW)	Công suất hữu dụng – biến mục tiêu (Active power (target variable))

Bảng 2.1. Ý nghĩa các cột data

Dưới đây là hình ảnh minh họa tuabin gió:



Hình 2.5 Cấu trúc tuabin

Trong đó:

- **Anemometer**: Bộ đo lường tốc độ gió. Chúng có trách nhiệm truyền dữ liệu của tốc độ gió đi tới bộ phận điều khiển.
- **Blades**: Đây là cánh quạt, khi gió thổi sẽ tạo lực vào cánh quạt. Làm quay trục của động cơ tuabin và sau đó là dẫn tới các chuyển động liên hoàn của hệ thống tuabin điện gió.
- **Brake**: Bộ hãm (hay còn được gọi là phanh), chúng dùng để dừng hoạt động motor trong trường hợp khẩn cấp.
- **Rotor**: Bộ phận này bao gồm các cánh quạt và trục.
- **Controller**: Bộ điều khiển.
- **Gear box**: Bộ phận hộp số. Trong bộ phần này, phần bánh răng của hệ thống sẽ được nối với trục tốc độ cao và trục tốc độ thấp. Bánh răng này không thể thiếu và chúng khá đắt tiền.
- **Generator**: Bộ phận máy phát để phát ra nguồn điện.
- **High – speed shaft**: Là trục chuyển động tốc độ cao của một máy phát.

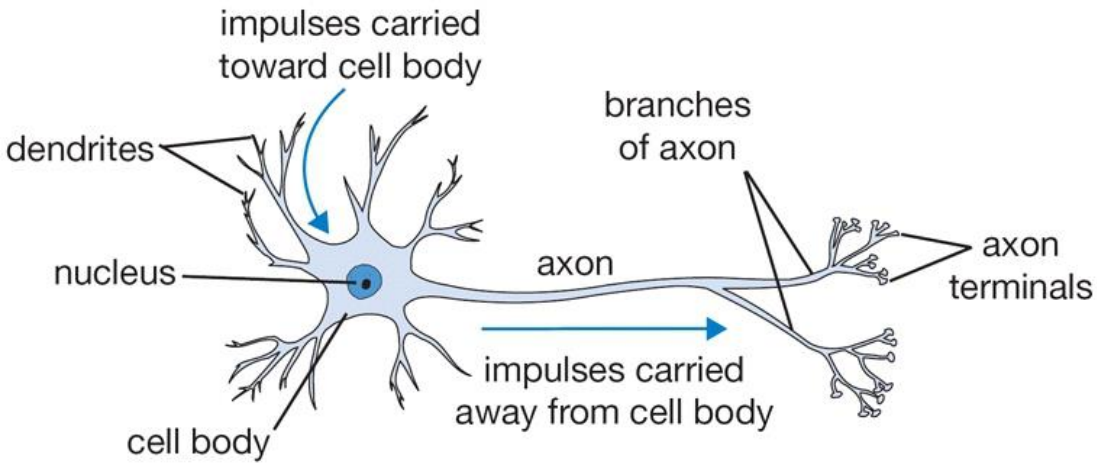
- **Low – speed shaft**: Ngược với High – speed shaft đó là trục chuyển động tốc độ thấp.
- **Nacelle**: Đây là phần vỏ của động cơ. Bao gồm lớp vỏ bọc ngoài và vỏ của Rotor. Được dùng để làm lớp bảo vệ, che chở cho các thành phần chi tiết cấu tạo bên trong của động cơ.
- **Pitch**: Đây là bộ phận giữ cho rotor có thể tạo ra điện khi chúng quay trong gió..

Chương 3 . Xây dựng mô hình

Sau đây là một số model, architecture ta sẽ sử dụng trong quá trình train với dataset.

3.1.Multilayer Perceptron (MLP)

Perceptron cơ bản: một mạng Neural được cấu thành bởi các neural đơn lẻ được gọi là các perceptron. Neural nhân tạo được lấy cảm hứng từ Neural sinh học như hình sau:



Hình 3.1 Neuron sinh trắc học

Quan sát hình ảnh trên, ta có thể thấy một Neural có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất. Mô hình của perceptron cũng tương tự như vậy.

Một perceptron sẽ nhận một hoặc nhiều đầu **x** vào dạng nhị phân và cho ra một kết quả **oo** dạng nhị phân duy nhất. Các đầu vào được điều phối tầm ảnh hưởng bởi các tham số trọng lượng tương ứng **w** của nó, còn kết quả đầu ra được quyết định dựa vào một ngưỡng quyết định **b** nào đó

$$o = \begin{cases} 0 & \text{if } \sum_i w_i x_i \leq \text{threshold} \\ 1 & \text{if } \sum_i w_i x_i > \text{threshold} \end{cases}$$

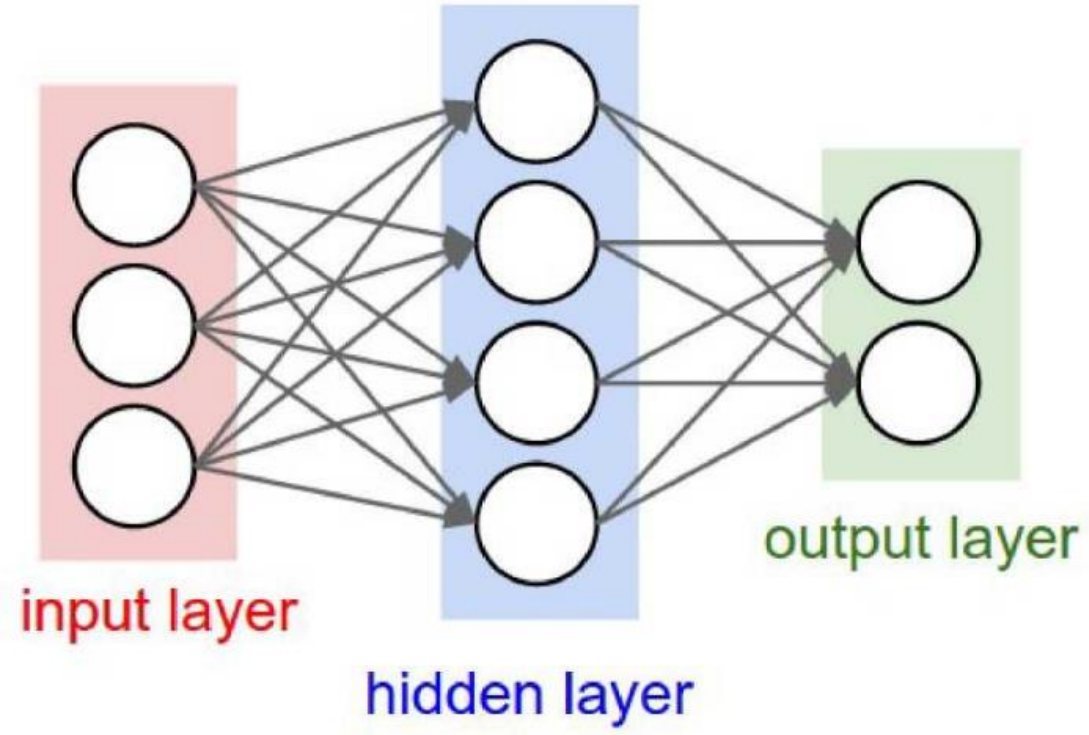
Đặt $b = -\text{threshold}$, ta có thể viết lại thành:

$$o = \begin{cases} 0 & \text{if } \sum_i w_i x_i + b \leq 0 \\ 1 & \text{if } \sum_i w_i x_i + b > 0 \end{cases}$$

Công thức 3.1 Perceptron

Kiến trúc mạng nơ ron nhân tạo:

Mạng Neural là sự kết hợp của của các tầng perceptron hay còn được gọi là perceptron đa tầng (*multilayer perceptron*) như hình vẽ bên dưới:



Hình 3.2. Mạng neuron nhân tạo

Một mạng Neural sẽ có 3 kiểu tầng:

- Tầng vào (input layer): Là tầng bên trái cùng của mạng thể hiện cho các đầu vào của mạng.

- Tầng ra (output layer): Là tầng bên phải cùng của mạng thể hiện cho các đầu ra của mạng.
- Tầng ẩn (hidden layer): Là tầng nằm giữa tầng vào và tầng ra thể hiện cho việc suy luận logic của mạng.

Lưu ý rằng, một Neural chỉ có 1 tầng vào và 1 tầng ra nhưng có thể có nhiều tầng ẩn.

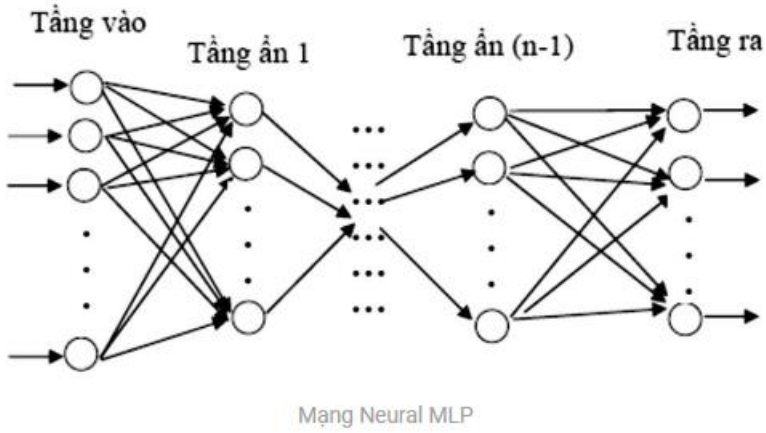
Trong mạng Neural, mỗi nút mạng là một sigmoid Neural nhưng hàm kích hoạt của chúng có thể khác nhau. Tuy nhiên trong thực tế người ta thường để chúng cùng dạng với nhau để tính toán cho thuận lợi.

Ở mỗi tầng, số lượng các nút mạng (Neural) có thể khác nhau tuỳ thuộc vào bài toán và cách giải quyết. Nhưng thường khi làm việc người ta để các tầng ẩn có số lượng Neural bằng nhau. Ngoài ra, các Neural ở các tầng thường được liên kết đôi một với nhau tạo thành mạng kết nối đầy đủ (*full- connected network*).

Kiến trúc mạng Neural MLP (Multi layer perceptron)

Multi layer perceptron (MLP) là mô hình mạng nơ ron được sử dụng rộng rãi nhất.

Một mạng MLP tổng quát là mạng có n (n≥2) tầng (thông thường tầng đầu vào(input layer) không được tính đến): trong đó gồm một tầng đầu ra-output layer (tầng thứ n) và (n-1) tầng ẩn-hidden layer.



Hình 3.3. Mô hình mạng MLP

Kiến trúc của một mạng MLP tổng quát có thể mô tả như sau:

- Đầu vào là các vector (x1, x2, …, xp) trong không gian p chiều, đầu ra là các vector (y1, y2, …, yq) trong không gian q chiều. Đối với các bài toán phân loại, p chính là kích thước của mẫu đầu vào, q chính là số lớp cần phân loại.
- Mỗi neural thuộc tầng sau liên kết với tất cả các noron thuộc tầng liền trước nó.

- Đầu ra của neural tầng trước là đầu vào của nơron thuộc tầng liền sau nó.

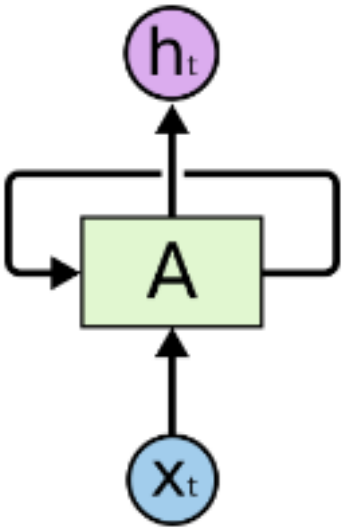
Hoạt động của mạng MLP như sau: tại tầng đầu vào các neural nhận tín hiệu vào xử lý (tính tổng trọng số, gửi tới hàm truyền) rồi cho ra kết quả (là kết quả của hàm truyền); kết quả này sẽ được truyền tới các neural thuộc tầng ẩn thứ nhất; các nơron tại đây tiếp nhận như là tín hiệu đầu vào, xử lý và gửi kết quả đến tầng ẩn thứ 2. Quá trình tiếp tục cho đến khi các neural thuộc tầng ra cho kết quả.

Một số kết quả đã được chứng minh:

- Bất kì một hàm Boolean nào cũng có thể biểu diễn được bởi một mạng MLP 2 tầng trong đó các neural sử dụng hàm truyền sigmoid.
- Tất cả các hàm liên tục đều có thể xấp xỉ bởi một mạng MLP 2 tầng sử dụng hàm truyền sigmoid cho các neural tầng ẩn và hàm truyền tuyến tính cho các nơron tầng ra với sai số nhỏ tùy ý.
- Mọi hàm bất kỳ đều có thể xấp xỉ bởi một mạng MLP 3 tầng sử dụng hàm truyền sigmoid cho các neural tầng ẩn và hàm truyền tuyến tính cho các neural tầng ra.

3.2.Long Short Term Memory (LSTM)

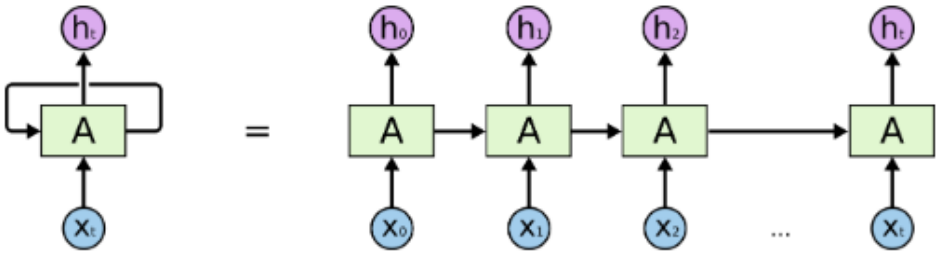
Trước tiên, giới thiệu 1 chút về RNN (Recurrent neural network) -mạng nơ ron truy hồi. Dưới đây là hình biểu diễn về mạng RNN:



Hình 3.4 Mạng RNN

Hình trên biểu diễn kiến trúc của một mạng nơ ron truy hồi. Trong kiến trúc này mạng nơ ron sử dụng một đầu vào là một véc tơ Xt và trả ra đầu ra là một giá trị ẩn ht. Đầu vào được đấu với một thân mạng nơ ron A có tính chất truy hồi và thân này được đấu tới đầu ra ht.

Vòng lặp A ở thân mạng nơ ron là điểm mấu chốt trong nguyên lý hoạt động của mạng nơ ron truy hồi. Đây là chuỗi sao chép nhiều lần của cùng một kiến trúc nhằm cho phép các thành phần có thể kết nối liền mạch với nhau theo mô hình chuỗi. Đầu ra của vòng lặp trước chính là đầu vào của vòng lặp sau. Nếu trải phẳng thân mạng nơ ron A ta sẽ thu được một mô hình dạng:



Hình 3.5. Cấu trúc trải phẳng của RNN

Một trong những điểm đặc biệt của RNN đó là nó có khả năng kết nối các thông tin liền trước với nhiệm vụ hiện tại tuy nhiên không thể liên kết được với những thông tin trước nó tức RNN không có khả năng học trong dài hạn. Vì thế LSTM là giải pháp được nghĩ tới ở bài toán này bởi :

Việc dự đoán phải dựa vào các thời điểm khác nhau chứ không thể chỉ dựa vào mốc thời gian liền kề trước nó.

LSTM là 1 kiến trúc đặc biệt của RNN có khả năng học được sự phụ thuộc trong dài hạn, khắc phục được nhiều điểm yếu của RNN.

Ở đây ta 2 lần train model này:

Lần 1 với các tham số:

Dataset					Model				
Version	No. train	No. val	Path	Note	Architecture	Epoch	Batchsize	Pretrained	Others
	24883	6221	wtbdata	using first 220 days. create dataset => train_test_split(test size = 0.2, random_state = 0.2) . input_shape = (1,288,6)	LSTM(units=64, input_shape=(288,6), return_sequences= True) BatchNormalization() Dropout(0.2) LSTM(units=64, input_shape=(288,6), return_sequences= True) BatchNormalization() Dense(units=1)	50	128	None	

Hình 3.6. LSTM_1

Lần 2 với các tham số:

Dataset					Model				
Version	No. train	No. val	Path	Note	Architecture	Epoch	Batchsize	Pretrained	Others
	24998	6250	wtbdata	using first 220 days. create dataset => train_test_split(test size = 0.2, random_state = 0.2) . input_shape = (1,144,6)	LSTM(units=64, input_shape=(144,6), return_sequences= True) BatchNormalization() Dropout(0.2) LSTM(units=64, input_shape=(144,6), return_sequences=True) Reshape((-1, 144*64)) BatchNormalization() Dropout(0.2) Dense(units=288)	50	128	None	

Hình 3.7. LTSM_2

3.3.Convolutional neural network (CNN)

CNN là 1 trong những mô hình Deep learning tiên tiến. Nó giúp ta xây dựng được những hệ thống thông minh với độ chính xác cao.

Cấu trúc mạng CNN:

Mạng CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo.

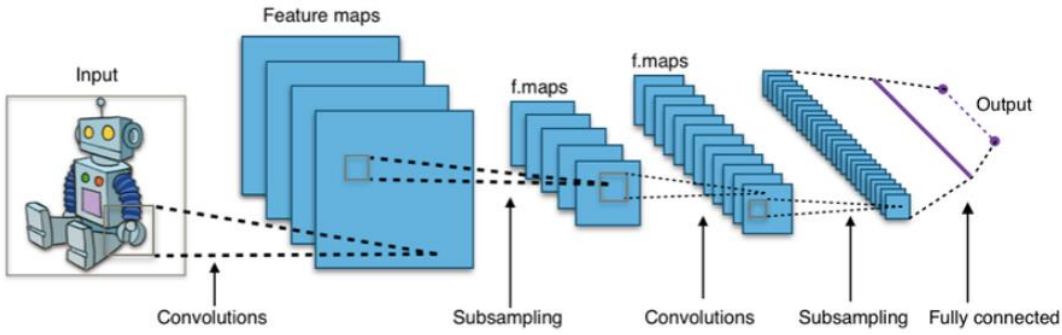
Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Trong mô hình mạng truyền ngược (feedforward neural network) thì mỗi neural đầu vào (input node) cho mỗi neural đầu ra trong các lớp tiếp theo.

Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer) hay mạng toàn vẹn (affine layer). Còn trong mô hình CNNs thì ngược lại. Các layer liên kết được với nhau thông qua cơ chế convolution.

Layer tiếp theo là kết quả convolution từ layer trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó.

Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số layer khác như pooling/subsampling layer dùng để chắt lọc lại các thông tin hữu ích hơn (loại bỏ các thông tin nhiễu).

Trong quá trình huấn luyện mạng (traning) CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà bạn thực hiện. Ví dụ trong tác vụ phân lớp ảnh, CNNs sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features. Layer cuối cùng được dùng để phân lớp ảnh.



Hình 3.8. Cấu trúc CNN

Các tham số được sử dụng trong model này bao gồm :

Dataset					Model				
Version	No. train	No. val	Path	Note	Architecture	Epoch	Batchsize	Pretrained	Others
	24883	6221	wtbdata	using first 220 days=> create dataset => train_test_split(test size = 0.2, random_state = 0.2) . input_shape = (1,288,6)	Conv1D(filters= 32, kernel_size = 2, activation='relu', input_shape=(288,6)) MaxPooling1D(pool_size=2) Flatten() Dense(600, activation='relu') Dropout(0.2) Dense(288)	50	128	None	

Hình 3.9. 1D_CNN

3.4.Graph convolutional network (GCN)

Graph convolutional nerwork(GCN) là 1 cách tiếp cận để học bán bám sát trên dữ liệu có cấu trúc đồ thị. Nó dựa trên biến thể hiệu quả của mạng CNN hoạt động trực tiếp trên đồ thị.

Một mô hình GCN được định nghĩa như sau:

- 1 đồ thị G = (V, E)
- 1 ma trận adjacency matrix A nxn
- *X ∈ R^(nxd)* là ma trận feature ứng với các nút của đồ thị, với n*n* là tổng số lượng nút và d*d* là số chiều của node feature. Node embedding liên quan đến các thông tin của nút đó.

Ở đây ta train với các tham số sau:

Dataset					Model				
Version	No. train	No. val	Path	Note	Architecture	Epoch	Batchsize	Pretrained	Others
	27648	3168		train/val/test: 195/25/25 x: 144x5, y: 288	2	30	512	None	

Hình 3.10 .GCN

3.5. Activation Function

Trên thực tế có rất nhiều activation function có thể được sử dụng trong quá trình train, ta có thể kể đến như:

- Sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Công thức 3.2. Sigmoid

- Tanh

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

Công thức 3.3 Tanh

- Leaky ReLU

$$f(x) = \mathbb{1}(x < 0)(\alpha x) + \mathbb{1}(x >= 0)(x)$$

với α là hằng số nhỏ.

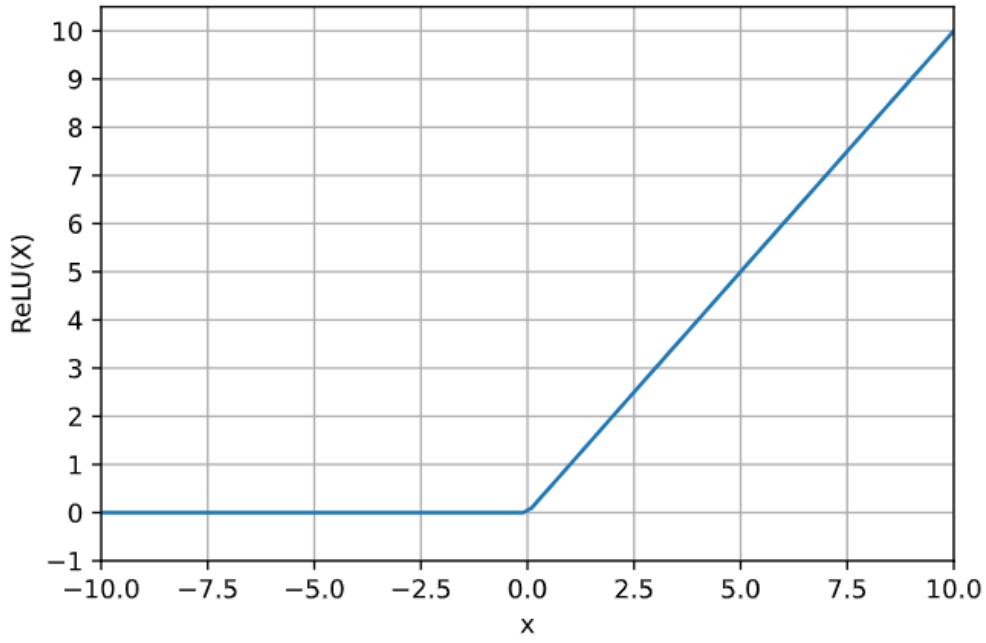
Công thức 3.4 LeakyReLU

Tuy nhiên ở bài toán này, ta sử dụng hàm activation là ReLU có công thức là:

$$f(x) = max(0, x)$$

Công thức 3.5 . ReLU

Phân tích:



Hình 3.11. đồ thị ReLU

Hàm ReLU đang được sử dụng khá nhiều trong những năm gần đây khi huấn luyện các mạng neuron. ReLU đơn giản lọc các giá trị < 0. Nhìn vào công thức chúng ta dễ dàng hiểu được cách hoạt động của nó. Một số ưu điểm khá vượt trội của nó so với Sigmoid và Tanh:

- Tốc độ hội tụ nhanh hơn hẳn. ReLU có tốc độ hội tụ nhanh gấp 6 lần Tanh (Krizhevsky et al.). Điều này có thể do ReLU không bị bão hoà ở 2 đầu như Sigmoid và Tanh.
- Tính toán nhanh hơn. Tanh và Sigmoid sử dụng hàm exp và công thức phức tạp hơn ReLU rất nhiều do vậy sẽ tốn nhiều chi phí hơn để tính toán.

Chương 4. Công nghệ sử dụng

4.1 Ngôn ngữ lập trình: Python

Hình 4.1. Logo python

Python là 1 ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt năm 1991. Python được thiết kế vs ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình và là ngôn ngữ lập trình dễ học; được dùng rộng rãi trong phát triển trí tuệ nhân tạo. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu. Vào tháng 7 năm 2018, van Rossum đã từ chức lãnh đạo trong cộng đồng ngôn ngữ Python sau 30 năm làm việc

Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động,do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, Tcl. Python được phát triển trong 1 dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

Ban đầu, python được phát triển để chạy trên nền unix. Theo thời gian, python dần mở rộng sang mọi hệ điều hành từ MS_DOS đến Mac OS, OS/2, Windows, Linux.

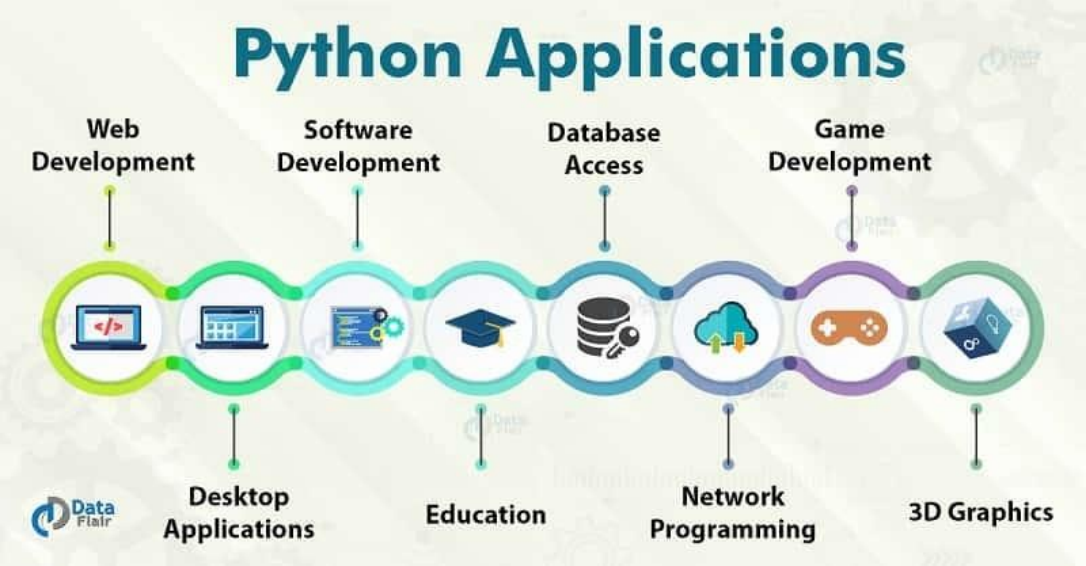
Python luôn được xếp hạng và những ngôn ngữ lập trình phổ biến nhất.

Các tính năng chính của python:

- ngôn ngữ lập trình đơn giản,dễ học
- miễn phí,mã nguồn mở
- khả năng di chuyển

- khả năng mở rộng và có thể nhúng
- ngôn ngữ thông dịch cao cấp
- cải thiện năng suất làm việc của nhà phát triển vì với python họ có thể dùng ít dòng mã hơn để viết 1 chương trình
- python có thể được sử dụng trên nhiều hệ điều hành máy tính khác nhau(windows, macOS, Linux, Unix)

Tóm lại, Python là một ngôn ngữ lập trình thông dịch (interpreted), hướng đối tượng (object-oriented), và là một ngôn ngữ bậc cao (high-level) ngữ nghĩa động (dynamic semantics). Python hỗ trợ các module và gói (packages), khuyến khích chương trình module hóa và tái sử dụng mã. Trình thông dịch Python và thư viện chuẩn mở rộng có sẵn dưới dạng mã nguồn hoặc dạng nhị phân miễn phí cho tất cả các nền tảng chính và có thể được phân phối tự do.



Hình 4.2 Python Applications

4.2 Pytorch

Hình 4.3 Logo Pytorch

Pytorch là open source ML framework dựa trên thư viện Torch, được sử dụng cho các ứng dụng như thị giác máy tính (Computer Vision) hay xử lí ngôn ngữ tự nhiên (Natural language Processing) chủ yếu được phát triển bởi MetaAI. Đây là phần mềm mã nguồn mở miễn phí được phát hành theo giấy phép BSD sửa đổi. Mặc dù giao diện python được trau chuốt hơn và là trọng tâm phát triển chính, pytorch cũng có giao diện C++.

Một số phần mềm học sâu được xây dựng trên Pytorch bao gồm Tesla Autopilot, Uber's Pyro, Hugging Face's Transformers,Pytorch Lightning và Catalyst.

Trong số những framework hỗ trợ deeplearning thì pytorch là một trong những framework được ưa chuộng nhiều nhất (cùng với tensorflow và keras), có lượng người dùng đông đảo, cộng đồng lớn mạnh. Vào năm 2019 framework này đã vươn lên vị trí thứ 2 về số lượng người dùng trong những framework hỗ trợ deeplearning (chỉ sau tensorflow). Đây là package sử dụng các thư viện của CUDA và C/C++ hỗ trợ các tính toán trên GPU nhằm gia tăng tốc độ xử lý của mô hình. 2 mục tiêu chủ đạo của package này hướng tới là:

- Thay thế kiến trúc của numpy để tính toán được trên GPU.
- Deep learning platform cung cấp các xử lý tốc độ và linh hoạt.

Pytorch là framework được phát triển bởi Facebook. Pytorch được phát triển với giấy phép mã nguồn mở do đó nó tạo được cho mình một cộng đồng rất lớn.

1 số ưu điểm của Pytorch:

- Mang lại khả năng debug dễ dàng hơn theo hướng interactively, rất nhiều nhà nghiên cứu và engineer đã dùng cả pytorch và tensorflow đều đánh giá cáo pytorch hơn trong vấn đề debug và visualize.
- Hỗ trợ tốt dynamic graphs
- Được phát triển bởi đội ngũ facebook
- Kết hợp cả các API cấp cao và cấp thấp

Trong đó ưu điểm dễ thấy nhất của pytorch đó là việc nó rất gần với ngôn ngữ Python mà không phải đưa vào các khái niệm về Graph, Session phức tạp. Điều này khiến cho các bạn chuyển đổi từ các package khác (nhất là bạn nào quen sử dụng Numpy) sang Pytorch một cách dễ dàng hơn. Do không còn dựa trên Graph nên việc debug có thể thực hiện trực tiếp trong mô hình mà không cần phải khởi tạo một Session giống như trong TF 1.X. Và đó cũng chính là hướng tiếp cận mà TF 2.X đang hướng tới. Thực sự giúp cho các developer thuận tiện hơn trong quá trình học và sử dụng Framework.

Bên cạnh đó là 1 số nhược điểm:

- Vẫn chưa được hoàn thiện trong việc deploy, áp dụng cho các hệ thống lớn ,... được như framework ra đời trước nó như tensorflow.
- Ngoài document chính từ pytorch thì vẫn còn khá hạn chế các nguồn tài liệu bên ngoài như các tutorials hay các câu hỏi trên stackoverflow.

Sau đây là 1 chút sự so sánh khác biệt chính giữa 2 framework nổi tiếng của DL là Pytorch và Tensorflow:

Pytorch	Tensorflow
PyTorch có liên quan chặt chẽ với khung Torch dựa trên lua-based được phát triển bởi Facebook.	TensorFlow được phát triển bởi Google Brain và được phát triển bởi Google.
PyTorch tương đối mới so với các công nghệ cạnh tranh khác.	TensorFlow không phải là mới và được nhiều nhà nghiên cứu và các chuyên gia trong ngành coi là một công cụ cần thiết.
PyTorch bao gồm mọi thứ theo cách bắt buộc và năng động.	TensorFlow bao gồm các đồ thị tĩnh và động dưới dạng kết hợp.
Đồ thị tính toán trong PyTorch được xác định trong thời gian chạy.	TensorFlow không bao gồm bất kỳ tùy chọn thời gian chạy nào.
PyTorch bao gồm triển khai đặc trưng cho các khuôn khổ di động và nhúng.	TensorFlow hoạt động tốt hơn cho các frameworks nhúng.

Bảng 4.1.Pytorch-Tensorflow

1 module hay sử dụng trong Pytorch đó là Neural Network (nn). Pytorch hỗ trợ thư viện torch.nn để xây dựng neural network. Nó bao gồm các khối cần thiết để xây dựng nên 1 mạng neural network hoàn chỉnh. Mỗi layer trong mạng gọi là một module và được kế thừa từ nn.Module. Mỗi module sẽ có thuộc tính Parameter (ví dụ W, b trong Linear Regression) để được tối ưu trong quá trình mô hình học.

4.3 Library

Ở đây em có sử dụng 1 số thư viện để phục vụ quá trình làm đồ án, cụ thể như sau

4.3.1 Matplotlib

Hình 4.4. Matplotlib

Matplotlib là một trong những thư viện Python phổ biến nhất được sử dụng để trực quan hóa dữ liệu. Nó là một thư viện đa nền tảng để tạo các đồ thị 2D từ dữ liệu trong các mảng. Matplotlib được viết bằng Python và sử dụng NumPy, phần mở rộng toán học của Python. Nó cung cấp một API hướng đối tượng giúp nhúng các plot trong các ứng dụng và sử dụng bộ công cụ GUI Python như PyQt, WxPythonotTkinter. Ngoài ra có thể được sử dụng trong Python và IPython shell, Jupyter Notebook và các máy chủ web.

Matplotlib có giao diện được đặt tên là Pylab, được thiết kế giống với MATLAB - ngôn ngữ lập trình độc quyền được phát triển bởi MathWorks. Matplotlib cùng với NumPy có thể được coi là mã nguồn mở tương đương với MATLAB.

Matplotlib ban đầu được viết bởi John D. Hunter vào năm 2003. Phiên bản ổn định hiện tại là 2.2.0 được phát hành vào tháng 1 năm 2018.

Để thực hiện các suy luận thống kê cần thiết, cần phải trực quan hóa dữ liệu của bạn và Matplotlib là một trong những giải pháp như vậy cho người dùng Python. Nó là một thư viện vẽ đồ thị rất mạnh mẽ hữu ích cho những người làm việc với Python và NumPy. Module được sử dụng nhiều nhất của Matplotib là Pyplot cung cấp giao diện như MATLAB nhưng thay vào đó, nó sử dụng Python và nó là nguồn mở.

Một Matplotlib figure có thể được phân loại thành nhiều phần như dưới đây:

- **Figure:** Như một cái cửa sổ chứa tất cả những gì bạn sẽ vẽ trên đó.
- **Axes:** Thành phần chính của một figure là các axes (những khung nhỏ hơn để vẽ hình lên đó). Một figure có thể chứa một hoặc nhiều axes. Nói cách

khác, figure chỉ là khung chứa, chính các axes mới thật sự là nơi các hình vẽ được vẽ lên.

- **Axis:** Chúng là dòng số giống như các đối tượng và đảm nhiệm việc tạo các giới hạn biểu đồ.
- **Artist:** Mọi thứ mà bạn có thể nhìn thấy trên figure là một artist như Text objects, Line2D objects, collection objects. Hầu hết các Artists được gắn với Axes.

Ưu điểm của Matplotlib:

- Matplotlib là một thư viện giống như GNUplot. Ưu điểm chính so với GNUplot đó là Matplotlib là một module của Python. Do mức độ phổ biến của python ngày càng tăng, nên matplotlib cũng nhận được sự quan tâm tương tự.

- Một lý do khác cho sự hấp dẫn của Matplotlib nằm ở chỗ nó được xem là một sự lựa chọn hoàn hảo thay thế cho MATLAB, nếu nó được sử dụng kết hợp với Numpy và Scipy. Trong khi MATLAB đắt đỏ và mã nguồn đóng, Matplotlib lại miễn phí và mã nguồn mở. Nó cũng là ngôn ngữ hướng đối tượng. Hơn nữa, nó có thể được sử dụng với bộ công cụ GUI mục đích chung như wxPython, Qt, và GTK+. Cũng có một thủ tục "pylab", được thiết kế để giống với MATLAB. Điều này có thể làm cho những người quen sử dụng MATLAB dễ dàng chuyển sang dùng matplotlib.

- Matplotlib có thể được sử dụng để tạo ra những figures đủ chất lượng cho một loạt các định dạng hardcopy và môi trường tương tác trên nền tảng.

- Một đặc điểm khác của matplotlib là tốc độ lĩnh hội, có nghĩa là người dùng thường đạt được tiến bộ nhanh chóng sau khi bắt đầu. Các trang web chính thức có thể nói những điều sau đây: "matplotlib cố gắng làm những điều khó khăn, phức tạp trở lên dễ dàng nhất có thể. Bạn có thể tạo ra các hình vẽ, histograms, phổ, biểu đồ thanh, errorcharts, scatterplots, vv, với chỉ một vài dòng mã."

- Matplotlib có một số interfaces để tương tác với thư viện matplotlib: Object-Oriented API, The Scripting Interface (pyplot), The MATLAB Interface (pylab). Pyplot và pylab đều là lightweight interfaces, tuy nhiên Pyplot cung cấp một giao diện thủ tục các thư viện vẽ hướng đối tượng trong matplotlib. Các lệnh vẽ của nó được thiết kế tương tự với Matlab cả về cách đặt tên và ý nghĩa các đối số. Cách thiết kế này đã giúp cho việc sử dụng pyplot dễ dàng và dễ hiểu .

Pyplot là một module của Matplotlib cung cấp các hàm đơn giản để thêm các thành phần plot như lines, images, text, v.v. vào các axes trong figure.

4.3.2 OS

Module OS trong Python cung cấp các chức năng được sử dụng để tương tác với hệ điều hành và cũng có được thông tin liên quan về nó. OS đi theo các Module tiện ích tiêu chuẩn của Python. Module này cung cấp một cách linh động sử dụng chức năng phụ thuộc vào hệ điều hành.

Module OS trong python cho phép chúng ta làm việc với các tập tin và thư mục. Sau đây là 1 vài phương thức:

Hàm	Ý nghĩa
os.getcwd()	trả về thư mục làm việc hiện tại của một tiến trình.
os.listdir()	liệt kê tất cả các tệp, thư mục của đường dẫn được truyền vào
os.chdir()	thay đổi thư mục làm việc hiện tại
os.mkdir()	tạo thư mục/tệp ở đường dẫn được truyền
os.rmdir()	thực hiện việc xóa thư mục hiện tại
os.walk()	Phương thức này sẽ duyệt qua hệ thống tệp

Bảng 4.2. OS

4.3.3 Pandas

Hình 4.5.Pandas

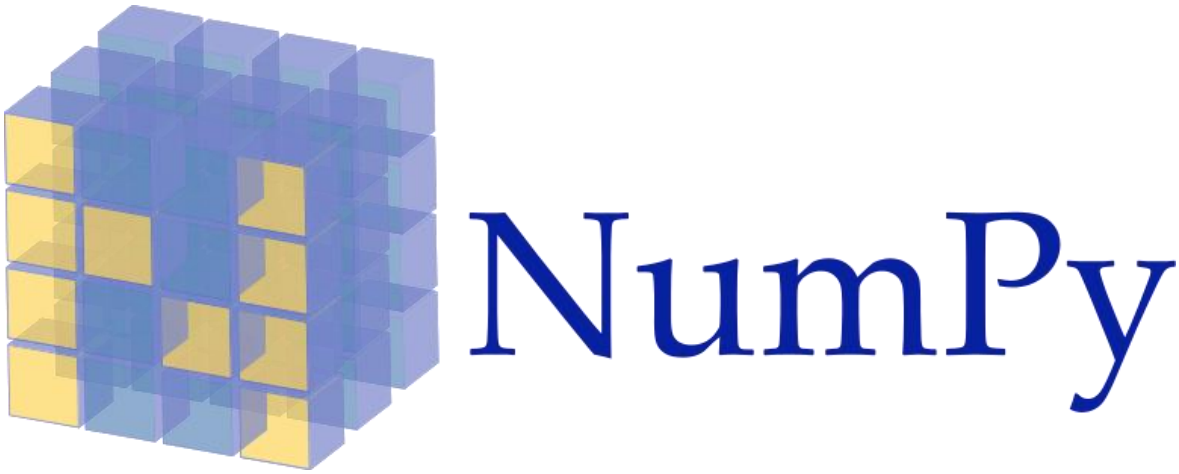
Pandas là một thư viện mã nguồn mở được xây dựng dựa trên NumPy, sử dụng thao tác và phân tích dữ liệu, được thiết kế để cho phép bạn làm việc với dữ liệu được gắn nhãn hoặc quan hệ theo cách trực quan hơn

Đây cũng là bộ công cụ phân tích và xử lý dữ liệu mạnh mẽ của ngôn ngữ lập trình python. Thư viện này được sử dụng rộng rãi trong cả nghiên cứu lẫn phát triển các ứng dụng về khoa học dữ liệu. Thư viện này sử dụng một cấu trúc dữ liệu riêng là Dataframe. Pandas cung cấp rất nhiều chức năng xử lý và làm việc trên cấu trúc dữ liệu này. Chính sự linh hoạt và hiệu quả đã khiến cho pandas được sử dụng rộng rãi.

- DataFrame đem lại sự linh hoạt và hiệu quả trong thao tác dữ liệu và lập chỉ mục;
- Là một công cụ cho phép đọc/ ghi dữ liệu giữa bộ nhớ và nhiều định dạng file: csv, text, excel, sql database, hdf5;
- Liên kết dữ liệu thông minh, xử lý được trường hợp dữ liệu bị thiếu. Tự động đưa dữ liệu lộn xộn về dạng có cấu trúc;
- Dễ dàng thay đổi bố cục của dữ liệu;
- Tích hợp cơ chế trượt, lập chỉ mục, lấy ra tập con từ tập dữ liệu lớn.
- Có thể thêm, xóa các cột dữ liệu;
- Tập hợp hoặc thay đổi dữ liệu với *group by* cho phép bạn thực hiện các toán tử trên tập dữ liệu;
- Hiệu quả cao trong trộn và kết hợp các tập dữ liệu;

- Lập chỉ mục theo các chiều của dữ liệu giúp thao tác giữa dữ liệu cao chiều và dữ liệu thấp chiều;
- Tối ưu về hiệu năng;
- Pandas được sử dụng rộng rãi trong cả học thuật và thương mại. Bao gồm thống kê, thương mại, phân tích, quảng cáo,…
- Có thể xử lý tập dữ liệu khác nhau về định dạng: chuỗi thời gian, bảng không đồng nhất, ma trận dữ liệu
- Khả năng import dữ liệu từ nhiều nguồn khác nhau như CSV, DB/SQL
- Có thể xử lý vô số phép toán cho tập dữ liệu: subsetting, slicing, filtering, merging, groupBy, re-ordering, and re-shaping,..
- Xử lý dữ liệu mất mát theo ý người dùng mong muốn: bỏ qua hoặc chuyển sang 0
- Xử lý, phân tích dữ liệu tốt như mô hình hoá và thống kê
- Tích hợp tốt với các thư viện khác của python
- Cung cấp hiệu suất tốt

4.3.4 Numpy



Hình 4.6.Numpy

Numpy là là một thư viện toán học rất phổ biến và mạnh mẽ của Python. NumPy được trang bị các hàm số đã được tối ưu, cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng Python đơn thuần.

Đây là một trong những thư viện hữu ích nhất của python, đặc biệt là nếu bạn đang tìm hiểu về các con số. Vì phần lớn Khoa học Dữ liệu và Máy học xoay quanh Thống kê, nên việc thực hành trở nên quan trọng hơn nhiều.

NumPy được phát triển bởi Jim Hugunin. Phiên bản ban đầu là Numarray được phát triển, có một số chức năng bổ sung. Năm 2005, Travis Oliphant đã tạo ra gói NumPy bằng cách kết hợp các tính năng của Numarray và gói Numeric.

Sử dụng NumPy, lập trình viên có thể thực hiện các thao tác sau:

- Các phép toán toán học và logic trên mảng.
- Các biến đổi Fourier và các quy trình để thao tác shape.
- Các phép toán liên quan đến đại số tuyến tính. NumPy tích hợp sẵn các hàm cho đại số tuyến tính và tạo số ngẫu nhiên.

NumPy thường được sử dụng cùng với các gói như SciPy (Python Scientific) và Mat-plotlib (thư viện vẽ đồ thị). Sự kết hợp này được sử dụng rộng rãi để thay thế cho MatLab, một nền tảng phổ biến cho tính toán kỹ thuật. Tuy nhiên, Python thay thế cho MatLab hiện được xem như một ngôn ngữ lập trình hoàn thiện và hiện đại hơn.Điều quan trọng hơn cả là Numpy là một thư viện mã nguồn mở, miễn phí so với MatLab là một thư viện mã nguồn đóng và phải trả phí .

Numpy là một thư viện lõi phục vụ cho khoa học máy tính của Python, hỗ trợ cho việc tính toán các mảng nhiều chiều, có kích thước lớn với các hàm đã được tối ưu áp dụng lên các mảng nhiều chiều đó. Numpy đặc biệt hữu ích khi thực hiện các hàm liên quan tới Đại Số Tuyến Tính.

4.3.5 Tensorflow

Hình 4.7. Tensorflow

Tensorflow chính là thư viện mã nguồn mở cho machine learning nổi tiếng nhất thế giới, được phát triển bởi các nhà nghiên cứu từ Google. Việc hỗ trợ mạnh mẽ các phép toán học để tính toán trong machine learning và deep learning đã giúp việc tiếp cận các bài toán trở nên đơn giản, nhanh chóng và tiện lợi hơn nhiều.

Các hàm được dựng sẵn trong thư viện cho từng bài toán cho phép TensorFlow xây dựng được nhiều neural network. Nó còn cho phép ta tính toán song song trên nhiều máy tính khác nhau, thậm chí trên nhiều CPU, GPU trong cùng 1 máy hay tạo ra các dataflow graph – đồ thị luồng dữ liệu để dựng nên các model.

Được viết bằng C++ và thao tác interface bằng Python nên phần performance của TensorFlow cực kỳ tốt. Đối tượng sử dụng nó cũng đa dạng không kém: từ các nhà nghiên cứu, nhà khoa học dữ liệu và dĩ nhiên không thể thiếu các lập trình viên.

Kiến trúc TensorFlow hoạt động được chia thành 3 phần:

- Tiền xử lí dữ liệu
- Dựng model
- Train và ước tính model

Các component của tensorflow:

- Tensor:
    - Tên của TensorFlow được đưa ra trực tiếp là nhờ vào framework cốt lõi của nó: Tensor. Trong TensorFlow, tất cả các tính toán đều liên quan tới các tensor. 1 tensor là 1 **vector** hay **ma trận** của n-chiều không gian đại diện cho tất cả loại dữ liệu. Tất cả giá trị trong 1

tensor chứa đựng loại dữ liệu giống hệt nhau với 1 **shape** đã biết (hoặc đã biết 1 phần). Shape của dữ liệu chính là chiều của ma trận hay mảng.

- 1 tensor có thể được bắt nguồn từ dữ liệu input hay kết quả của 1 tính toán. Trong TensorFlow, tất cả các hoạt động được tiến hành bên trong 1 **graph** – biểu đồ. Biểu đồ là 1 tập hợp tính toán được diễn ra liên tiếp. Mỗi operation được gọi là 1 **op node** (operation node) và được kết nối với nhau.
- Biểu đồ phát thảo các op và kết nối giữa các node. Tuy nhiên, nó không hiển thị các giá trị. Phần edge của các node chính là tensor, 1 cách để nhập operation với dữ liệu.

- Graph:
  - TensorFlow sử dụng framework dạng biểu đồ. Biểu đồ tập hợp và mô tả tất cả các chuỗi tính toán được thực hiện trong quá trình training.
  - Tất cả tính toán trong biểu đồ được thực hiện bằng cách kết nối các tensor lại với nhau. 1 **tensor** có 1 **node** và 1 **edge**. Node mang operation toán học và sản xuất các output ở đầu cuối. Các edge giải thích mối quan hệ input/output giữa các node.

Danh sách các thuật toán nổi bật được hỗ trợ bởi tensorflow. Tính tới thời điểm phiên bản ra mắt TensorFlow 1.10, nó đã sở hữu built-in API cho:

- **Linear regression:** tf.estimator.LinearRegressor
- **Classification:** tf.estimator.LinearClassifier
- **Deep learning classification:** tf.estimator.DNNClassifier
- **Deep learning wipe and deep:** tf.estimator.DNNLinearCombinedClassifier
- **Booster tree regression:** tf.estimator.BoostedTreesRegressor
- **Boosted tree classification:** tf.estimator.BoostedTreesClassifier

Chương 5. Đánh giá kết quả

5.1 Hàm đánh giá

Để đánh giá chất lượng model, có thể sử dụng nhiều hàm metric khác nhau

1. RMSE (Root Mean Squared Error):

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

Công thức 5.1. RMSE

2. MAE (Mean Absolute Error):

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

Công thức 5.2. MAE

3. MAPE (Mean Absolute Percentage Error):

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

Công thức 5.3. MAPE

4. R2-Score:

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

Công thức 5.4. R2-Score

RMSE và MAE là hai cách khác nhau để đánh giá độ sai lệch trung bình của mô hình với giá trị thực tế. MAPE phản ánh giá trị dự báo sai khác bao nhiêu phần trăm so với giá trị trung bình. R2-Score thể hiện mức độ phù hợp giữa các giá trị dự đoán và các giá trị thực tế.

Ở đây em sử dụng hàm đánh giá MSE (Mean Squared Error) với công thức :

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Công thức 5.5 Công thức MSE

Đây là hàm đánh giá thường dùng cho mô hình hồi quy. MSE dùng để đánh giá độ sai lệch trung bình của mô hình so với thực tế.

5.2 Kết quả

Sau quá trình training và áp dụng hàm đánh giá MSE trên 2 tập validation và test set, em thu được bảng sau :

Epoch	Batchsize	Model	Val MSE	Test MSE
30	512	GCN(512) + LSTM(1024) + Dense(288)	210660	143250
50	128	1D_CNN	46403.156	203128.179
50	128	LSTM	26400.916	413446.834
50	128	LSTM	26586.567	324925.856

Bảng 5.1 Kết quả đánh giá

Từ bảng trên ta thấy được,với số lượng batchsize, epoch như trên.

- Với model graph, sai số qua hàm MSE thu được thấp hơn nhiều so với khi ta train với các model CNN, LSTM.
- Test MSE trên model Graph thấp chỉ bằng gần 1/3 so với model LSTM_1
- Với model CNN thì Val MSE cao vượt trội
- Với model LSTM_2, cả Val MSE và Test MSE đều nằm ở mức giữa

Theo quan điểm cá nhân của em, sở dĩ sai số qua model graph thấp đến vậy là bởi vì:

- Graph tận dụng được quan hệ tương quan giữa vị trí các tuabin với nhau trong trang trại (vì trong trang trại, có thể có 2 turbine đứng gần thẳng nhau theo hướng gió, sẽ chắn nhau, 2 turbine cạnh nhau có thể có Patv giống nhau)
- Còn với các model còn lại, vì chỉ train trên mỗi 1 turbine 1 model, nên các turbine độc lập với nhau

Từ đó em rút ra nhận xét:

- Model Graph tận dụng được quan hệ giữa các turbin
- Model LSTM thì tốt cho dữ liệu time series
- Model CNN thì kém hơn

Chương 6. Kết luận và hướng phát triển

6.1 Kết luận

Cùng với sự hướng dẫn của TS. Trịnh Anh Phúc, em đã xây dựng được những model để đánh giá được năng lượng gió của 1 trang trại. Trong quá trình xây dựng, em cũng đã tìm hiểu và biết thêm được những model mới, cách train và đánh giá kết quả.

Những ưu điểm nổi bật của đề tài bao gồm:

- Đa dạng phương pháp
- Có thể so sánh được các phương pháp để tìm ra được cách tối ưu nhất
- Giảm gánh nặng tính toán của con người trong vấn đề ước tính năng lượng gió thu được…

6.2 Hướng phát triển

Trong tương lai, khi ứng dụng được đầu tư thời gian cũng như kinh phí xây dựng,em dự định sẽ mở rộng theo các hướng sau:

- Tích hợp hệ thống thành website hoặc app để dễ bề tính toán
- Tiếp tục tìm tòi nghiên cứu thêm các model để tìm kiếm kết quả tối ưu
- Khảo sát, thu thập dữ liệu từ các nhà máy điện gió ở Việt Nam
- Áp dụng mô hình tối ưu để đưa ra được kết quả chính xác nhất, giảm chi phí, gia tăng lợi nhuận cho các nhà máy điện gió…

Tài liệu tham khảo

[1] Thomas N.Kipf, Max Welling, Semi-Supervised Classification with Graph Convolutional Networks, https://arxiv.org/abs/1609.02907, last visited August 2022.

[2] Time series forecasting, https://www.tensorflow.org/tutorials/structured_data/time_series, last visited August 2022.

[3] PaddlePaddle, https://github.com/PaddlePaddle/PaddleSpatial, last visited 2022.

[4] Web traffic time series forecasting, https://www.kaggle.com/c/web-traffic-time-series-forecasting/discussion/43795, last visited August 2022.