TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO KẾT THÚC MÔN: KIẾN TẬP CÔNG NGHIỆP

TÌM HIỂU VỀ TIME SERIES FORECASTING TRONG DEEP LEARNING

Người hướng dẫn: **Nguyễn Thanh Hiên**
Người thực hiện: **Trần Quốc Lĩnh - 51703124**
Lớp: **17050301**
Khoá: **21**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020

LỜI CẢM ƠN

Cảm ơn <bằng cấp> thầy Nguyễn Thanh Hiên đã hướng dẫn và tận tình chỉ bảo em trong suốt thời gian em thực tập ở công ty TNHH Tin học Đại Phát, cũng như trong khoảng thời gian em thực hiện bài báo cáo này.

BÀI BÁO CÁO ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Em xin cam đoan đây là sản phẩm nghiên cứu của riêng em. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu ,hình ảnh được chính em thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong bài tiểu luận còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào em xin hoàn toàn chịu trách nhiệm về nội dung bài tập lớn của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do em gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 20 tháng 07 năm 2020

Tác giả

(Đã ký)

Trần Quốc Lĩnh

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

Tp. Hồ Chí Minh, ngày 20 tháng 07 năm 2020
(kí và ghi họ tên)

Phần đánh giá của GV giám sát

Tp. Hồ Chí Minh, ngày 20 tháng 07 năm 2020
(kí và ghi họ tên)

TÓM TẮT

Đây là một bài báo cáo, một bài tóm lược tổng quát và cơ bản nhất về time series forecasting trong deeplearning. Kèm theo đó bài báo cáo này cũng trình bày các phương pháp, các bài toán thực tế sử dụng time series forecasting.

MỤC LỤC

DANH MỤC CHÚ THÍCH

Hình ảnh

CHƯƠNG I: BỐI CẢNH HIỆN TẠI VÀ TỔNG QUAN VỀ TIME SERIES FORECASTING

Từ trước đến nay, việc xem xét, đánh giá những khả năng có thể xảy ra và tác động của nó đến các công ty, tổ chức, bộ máy là một điều hết sức quan trọng. Điều này là nhân tố quyết định cho sự tồn tại và phát triển của các cơ quan, đoàn thể. Dĩ nhiên nó đòi hỏi có một sự tỉ mỉ, một cái nhìn bao quát, một sự tính toán trong nhiều năm trời, và chỉ dành cho những chuyên gia thâm niên, những nhà nghiên cứu tài ba trong chính lĩnh vực đó. Một công việc không ít tốn kém cả về sức người lẫn sức của.

Hình 1: Bảng thống kê sự tăng trưởng cổ phiếu (hình minh họa)

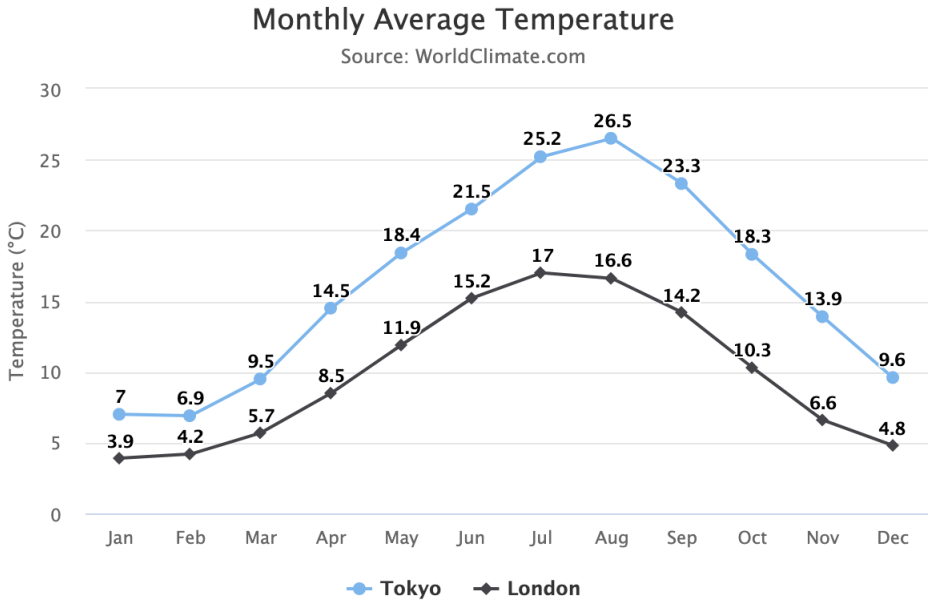
Bài toán về việc sử dụng máy tính để tính toán, dự đoán tương lai, dự đoán kết quả có thể xảy ra dựa trên những dữ kiện đã ghi nhận từ quá khứ đến hiện tại, không phải là một điều gì mới mẻ. Nhưng trước đó do hiệu quả và khả năng dự đoán kém, nên bài toán này không được cộng đồng xã hội chú ý đến. Tuy nhiên, trong những năm trở lại đây, như một quả bom, nó đã bùng nổ và lan rộng khắp thế giới, đón nhận được sự quan tâm nhiệt tình của không chỉ cộng đồng khoa học

mà còn cả các ngành nghề, lĩnh vực khác (kinh tế, y học, khí tượng thủy văn,...). Ấy là nhờ vào sự xuất hiện của deep learning và việc ứng dụng deep learning vào việc tính toán, đánh giá, thay đổi và cải thiện tính chính xác của bài toán.

Time series

Một time series là một chuỗi dữ liệu được đánh chỉ mục (một danh sách hoặc đồ thị) theo thứ tự thời gian. Thông thường các giá trị giữ liệu được xác định/đo đạt tại các mốc thời gian cách đều nhau. Chính vì thế, time series là một chuỗi dữ liệu rời rạc. Cụ thể là các thông tin về độ ẩm, hướng gió, lực gió, lượng mây,... được đo đạt mỗi giờ tại các trạm khí tượng thủy văn.

Chuỗi thời gian thường được biểu diễn dưới dạng biểu đồ đường. Chúng được sử dụng trong thống kê, xử lý tín hiệu, nhận dạng mẫu, kinh tế, tài chính, toán học, dự báo thời tiết, dự báo động đất, điện não đồ, kỹ thuật điều khiển, thiên văn học, kỹ thuật truyền thông và trong bất kỳ lĩnh vực khoa học ứng dụng - kỹ thuật nào liên quan đến các phép đo thời gian.



Hình 2: Biểu đồ ghi lại sự thay đổi nhiệt độ của hai thành phố Tokyo và Lodon qua các tháng

Phân tích chuỗi thời gian nhằm mục đích phát hiện và tập hợp lại các yếu tố có ảnh hưởng đến giá trị của đối tượng quan sát theo thời gian.

Trong Time-series Data, có hai loại chính:
- Chuỗi thời gian thông thường (regular time series), loại thông thường được gọi là số liệu.
- Chuỗi thời gian bất thường (events) là những sự kiện.

Time series forecasting

Như đã trình bày ở trên, dự đoán dựa trên chuỗi thời gian là việc dùng những dữ liệu đã có để dự đoán sự kiện hoặc một chuỗi sự kiện có thể xảy đến trong tương lai. Tuy nhiên, dự đoán thời gian là một việc khó khăn. Không như các bài toán phân lớp và hồi quy, bài toán chuỗi thời gian làm tăng độ phức tạp của thứ tự các quan sát hay sự phụ thuộc giữa các quan sát.

Người ta cũng đã dành không ít thời gian để xây dựng và điều chỉnh các mô hình, các phương pháp dự đoán thời gian sao cho sự kiện được dự đoán sai lệch thấp nhất có thể. Các phương pháp được trình bày bên dưới đây là những phương pháp phổ biến và có hiệu quả tốt nhất đã được kiểm chứng bằng thực nghiệm

Những phương pháp được sử dụng trong time series forecasting

Phương pháp truyền thống

Autoregressive integrated moving average (ARIMA)

Mô hình trung bình trượt, đồng liên kết, tự hồi quy ARIMA sử dụng chính các giá trị trong quá khứ của Y để giải thích cho Y hiện tại. Vì việc xác định mỗi một mô hình dựa trên sự phân tích dữ liệu cụ thể cho từng trường hợp mà không phụ thuộc hay bám lấy bất kỳ một lý thuyết tổng quát nào. Nên ARIMA có được tính chất linh hoạt và tiết kiệm chi phí.

Hạn chế, rào cản của phương pháp truyền thống

- **Chỉ hiệu quả với dữ liệu hoàn chỉnh:** Một vài giá trị bị thiếu hụt có thể ảnh hưởng nặng nề dến mô hình.

- **Phụ thuộc vào mối quan hệ tuyến tính:** Các mô hình truyền thống không thích hợp với phân phối phức tạp.
- **Chỉ dùng được với dữ liệu đơn biến:** Trong khi thực tế, dữ liệu đầu vào hầu hết là đa biến.
- **Không chạy tốt với những dự đoán có chu kỳ dài:** Chúng thường chỉ tập trung vào việc dự đoán duy nhất một sự kiện (one-step).
- **Thời gian phụ thuộc phải là cố định:** Mối quan hệ giữa các quan sát thường luôn khác nhau tại các thời điểm khác nhau, do đó cần có sự cố định giữa các thời gian quan sát.

Các phương pháp mới sử dụng deep learning

1. Multilayer Perceptrons (MLP)

MLP có những đặt trưng nổi trội như:

- Khả năng xấp xỉ các hàm phi tuyến tính. Mang lại một kết quả đầy hứa hẹn.
- Khả năng chịu lỗi. Học tốt kể cả khi bị thiết sót dữ liệu.
- Có thể xử lý dữ liệu đầu vào là đa biến.
- Có khả năng dự đoán nhiều bước (multi-step).

2. Convolutional Neural Networks (CNN)

CNN thường được dùng trong việc phân lớp hình ảnh, bởi hiệu quả tuyệt vời của nó. Bên cạnh đó CNN cũng có khả năng trích xuất đặc trưng của dữ liệu đầu vào trong time series forecasting. CNN còn sở hữu đầy đủ những ưu điểm của MLP.

3. Recurrent Neural Networks(RNN)

RNN cung cấp khả năng xử lý trật tự của các quan sát và cho phép quan sát thể hiện sự phụ thuộc theo gian của nó. Khác với MLP, RNN sử dụng chính bộ nhớ của mình để xử lý chuỗi dữ liệu đầu vào. Nhờ đó mà RNN có thể xử lý các quan sát có độ trễ cố định về thời gian.

So sánh, đánh giá các phương pháp

Phương pháp	**Ưu điểm**	**Hạn chế**
Truyền thống	- Đơn giản - Dễ hiểu	- Đòi hỏi dữ liệu hoàn chỉnh - Chỉ có mối quan hệ tuyến tính - Hiệu quả với đơn biến - Kém hiệu quả với chu kỳ dài
MLP	- Xấp xỉ phi tuyến - Chịu được nhiễu (Noise) - Xử lý được đầu vào đa biến - Dự đoán được nhiều bước	- Require meaningful mapping between inputs and outputs - Rely on lag observations - Static mapping function and fixed outputs & inputs
CNN	- Tất cả ưu điểm của MLP - Kết xuất đặt trưng	- Không thể học temporal dependence
RNN	- Tất cả ưu điểm của CNN - Học được temporal tpdependence	- Còn đang tranh cãi

.

CHƯƠNG II: MỘT SỐ ỨNG DỤNG THỰC TẾ

Dự đoán mức tăng trưởng của một loại cổ phiếu

Đây mà một lĩnh vực mà không phải ai cũng am hiểu, không phải ai cũng thích hợp để tham gia vào. Nó đòi hỏi không chỉ tài năng nghiên cứu thị trường của người chơi cổ phiếu mà còn là những hành động quyết đoán dựa trên các tính toán của mình.

Để có thể dự đoán được giá trị của cổ phiếu trong khoảng thời gian nào đó sắp tới, người đầu tư chuyên nghiệp cần phải điều tra vô số các yếu tố về thị trường, chính trị, xã hội,... các yếu tố này sẽ ảnh hưởng trực tiếp đến giá của cổ phiếu. Tuy nhiên các yếu tố vừa kể không hề đơn giản để thu thập và tính toán.

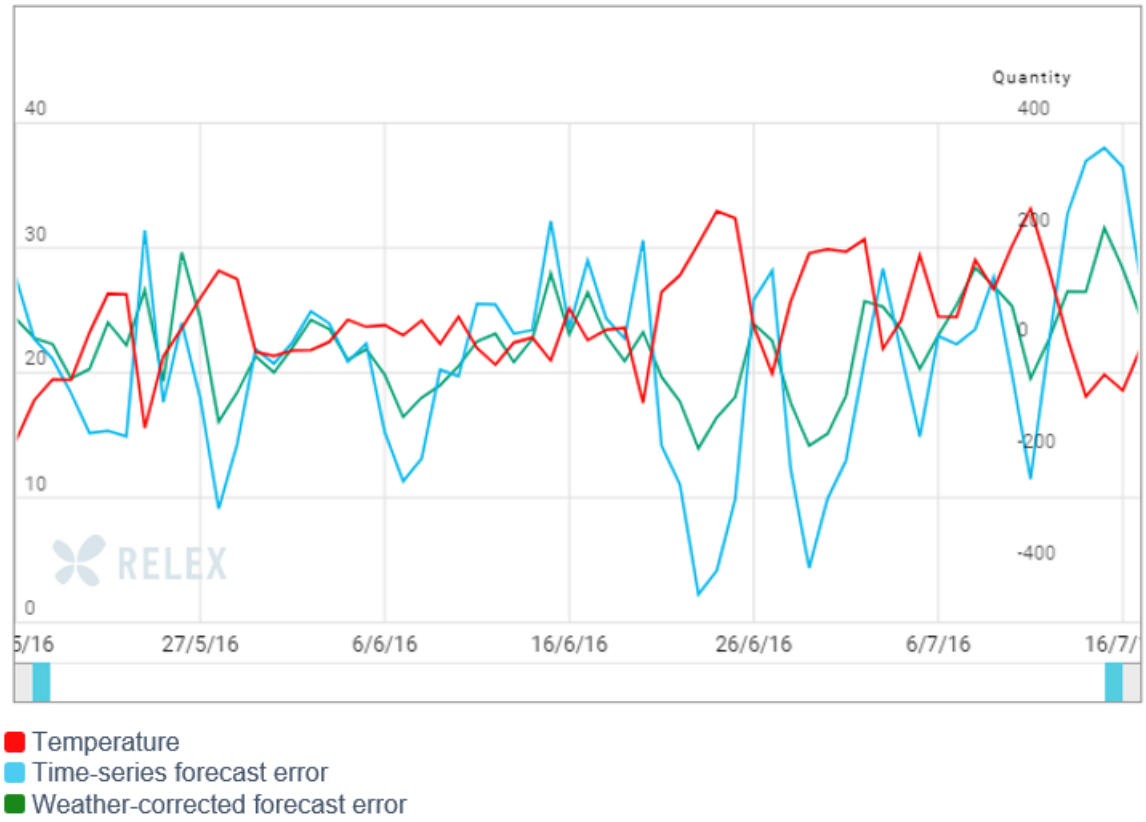
Hình 3: Bảng dự đoán mức tăng trưởng của cổ phiếu (Hình minh họa)

Nắm bắt được nhu cầu thực tế đó, người ta mới đưa, chỉnh sửa, phân tích các dữ liệu về thị trường, chính trị, xã hội sang dạng số liệu theo thời gian và áp dụng vào bài toán time series forecasting. Mặc dù vậy, có lẽ là do số liệu thu thập vẫn còn thiếu sót hoặc mô hình dự đoán vẫn chưa hoàn chỉnh nên kết quả dự đoán vẫn còn nhiều thiếu sót, không chiếm lấy được sự tin tưởng của người sử dụng. Tuy nhiên, các phương pháp mới vẫn được đang nghiên cứu sao cho kết quả ngày một chính xác hơn. Điều này mang đến một tương lai đầy hứa hẹn cho các nhà chơi cổ phiếu,

và cũng có thể là thu hút thêm nhiều người chơi cổ phiếu hơn nữa do tính đơn giản mà nó mang lại.

Dự báo thời tiết

Dự báo thời tiết chính xác đem lại nhiều lợi ích cho không ít các ngành nghề, lĩnh vực trong xã hội. Theo phương pháp truyền thống, quá trình dự báo thời tiết thường trải qua các giai đoạn: Quan trắc, trao đổi số liệu, phân tích, dự báo và phân phát sản phẩm dự báo. Với dữ liệu đã có, người ta sẽ phải thực hiện hàng tỉ phép tính trước khi cho ra dữ liệu cuối cùng, do đó vô cùng tiêu tốn thời gian và chi phí. Đây cũng là quy trình cơ bản của các loại hình dịch vụ dự báo thời tiết.



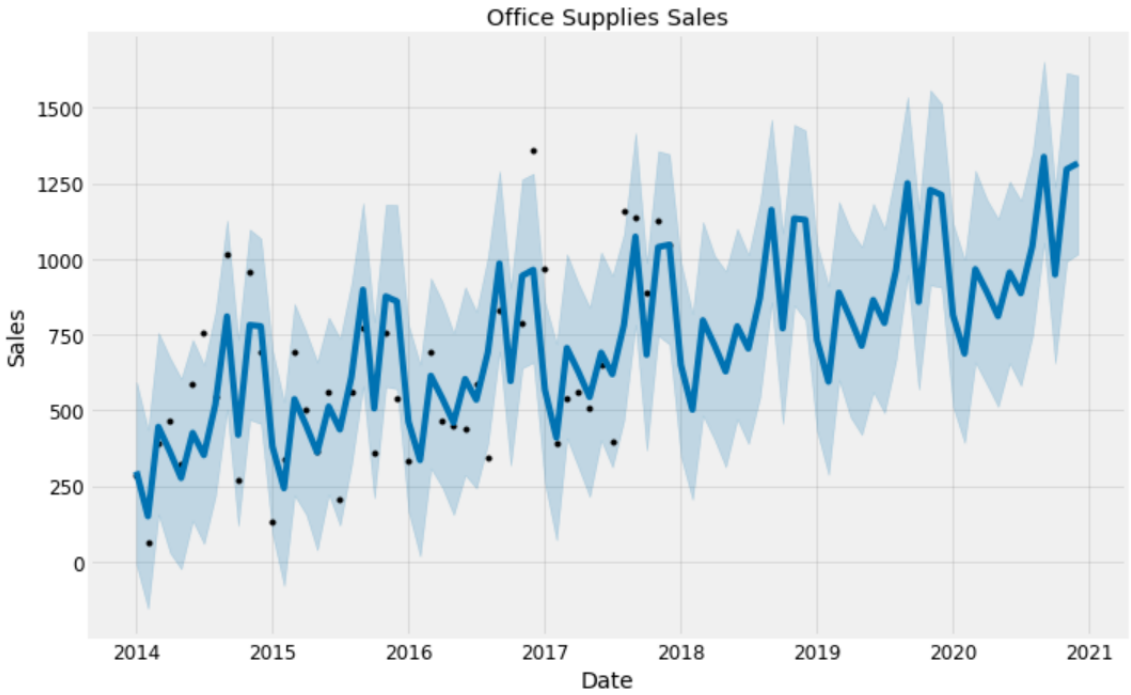
Hình 4: Bảng minh họa kết quả dự báo thời tiết (Hình minh họa)

Với việc thu thập, ghi lại các thông tin về trạng thái của khí quyển, nhiệt độ, độ ẩm, gió,... theo các mốc thời gian cụ thể. Đã đem lại tính tương thích cao cho bài toán series forecasting. Và đúng như mong đợi, hiệu quả mà time series forecasting

mang đến cho ngành dự báo thời tiết như một làn sóng mới về sự đơn giản, chính xác, tiết kiệm chi phí và nhanh chóng.

Dự đoán doanh thu

Có thể nói việc tính toán doanh thu trong một sự kiện hoặc một giai đoạn thời gian sắp tới đối với các ngành nghề liên quan đến sản xuất, mua bán, dịch vụ,... là một trong những công việc đòi hỏi sự tỉ mỉ và khó khăn nhất. Bởi nó giúp nhà đầu tư biết rằng, liệu họ có nên đầu tư vào cái này thay vì cái kia không?, liệu họ có nên mở rộng cơ sở trong tháng sắp tới hay không? nếu mở thì nên mở thêm bao nhiêu cửa hàng? Tổng quát hơn, nó ảnh hưởng trực tiếp đến hành động và quyết định của nhà đầu tư, của người sở hữu cửa hàng, sở hữu doanh nghiệp,...



Hình 5: Bảng dự đoán hạn mức doanh thu theo từng năm (Hình minh họa)

Từ ngàn xưa, từ trước khi người kinh doanh bắt đầu sử dụng máy tính để dự đoán mức doanh thu của mình, để tính toán mức doanh thu hì họ thường xem xét các vấn đề loại mặt hàng nào đang yêu thích? đang bán chạy? Có bao nhiêu cửa hàng đang kinh doanh loại mặt hàng này trong khu vực? Họ có bán chạy không? Vô số câu hỏi như thế sẽ được đặt ra, và sau khi trả lời tất thảy các câu hỏi đó, người ta

sẽ dự đoán (phần lớn các trường hợp là ước lượng) mức doanh thu mà họ thu được nếu người ta đầu tư vào loại mặt hàng, lĩnh vực đó. Nghe có vẻ đơn giản, nhưng để tính toán tốt, ít sai sót nhất thì số câu hỏi đặt ra phải lên tới hàng trăm, có khi hàng nghìn câu hỏi, và với mỗi câu hỏi người ta phải có câu trả lời dưới dạng số liệu (dĩ nhiên số liệu này phải gần với thực tế nhất có thể). Tuy nhiên phương pháp vừa nêu chỉ dành cho những doanh nghiệp khổng lồ, đa phần các doanh nghiệp vừa và nhỏ chỉ xem xét và đặt ra một vài câu hỏi trọng yếu, do đó mà các kết quả thu được cũng khác nhau.

Cho đến khi người ta bắt đầu nghĩ đến việc dùng máy tính để thay mình tính toán doanh thu, họ cũng dùng các phương pháp, các công cụ công nghệ cao thay họ thu thập dữ liệu, sau đó sắp xếp thành các chuỗi dữ liệu theo thời gian. Vâng, rất phù hợp với bài toán time series forecasting. Tương tự như trên, time series forecasting cũng cho ra các kết quả dự đoán có độ chính xác đến bất ngờ (ở một vài trường hợp hầu như trùng khớp). Nhờ vào điều này, không chỉ có các công ty, doanh nghiệp lớn mới có thể đưa ra các quyết định đúng đắn, tầm cỡ, mà còn mở rộng cơ hội cho các doanh nghiệp, công ty nhỏ lẽ khác. Góp phần cho sự phát triển chung của toàn xã hội.

CHƯƠNG III: VÍ DỤ MINH HỌA DỰ BÁO THỜI TIẾT

Chuẩn bị

Môi trường

- Sử dụng python 3.8
- Trình soạn thảo Jupyter notebook
- Package: tensorflow, matplotlib, numpy, os, pandas

Dữ liệu

- Tập dữ liệu thời tiết được cung cấp bởi Max Planck Institute for Biogeochemistry.

- Tập dữ liệu này chứa 14 loại đặc điểm khác nhau của thời tiết như: nhiệt độ (temperature), áp suất không khí (atmospheric), độ ẩm (humidity),... Các giá trị này sẽ được ghi lại cứ mỗi 10 phút, bắt đầu từ năm 2009 đến 2016. Phần dữ liệu này đã được chỉnh sửa lại bởi François Chollet.

Mục tiêu

Với dữ liệu đã có, sử dụng mô hình RNN để dự đoán nhiệt độ của 6 giờ kế tiếp qua hai trường hợp: đơn biến (univariate) và đa biến (multivariate)

Code

Khai báo các package cần thiết

```
import tensorflow as tf

import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd

mpl.rcParams['figure.figsize'] = (8, 6)
mpl.rcParams['axes.grid'] = False
```

Tải dữ liệu

```
zip_path = tf.keras.utils.get_file(
origin='https://storage.googleapis.com/tensorflow/tf-keras-datasets/jena_climate_2009_2016.csv.zip',
fname='jena_climate_2009_2016.csv.zip',
extract=True)
csv_path, _ = os.path.splitext(zip_path)
df = pd.read_csv(csv_path)
```

Xem thử bảng dữ liệu

```
df.head()
```

Nếu kết quả in ra có dạng như thế này, thì xem như việc tải dữ liệu thành công.

	Date Time	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	wv (m/s)	ma w (m/
0	01.01.2009 00:10:00	996.52	-8.02	265.40	-8.90	93.3	3.33	3.11	0.22	1.94	3.12	1307.75	1.03	1.75
1	01.01.2009 00:20:00	996.57	-8.41	265.01	-9.28	93.4	3.23	3.02	0.21	1.89	3.03	1309.80	0.72	1.5(
2	01.01.2009 00:30:00	996.53	-8.51	264.91	-9.31	93.9	3.21	3.01	0.20	1.88	3.02	1310.24	0.19	0.63
3	01.01.2009 00:40:00	996.51	-8.31	265.12	-9.07	94.2	3.26	3.07	0.19	1.92	3.08	1309.19	0.34	0.5(
4	01.01.2009 00:50:00	996.51	-8.27	265.15	-9.04	94.1	3.27	3.08	0.19	1.92	3.09	1309.00	0.32	0.63

Từ bảng trên ta thấy, cứ mỗi giờ thì có 6 quan sát được ghi lại. Vậy nên mỗi ngày sẽ có 144 (6×24) quan sát.

Với mục tiêu dự đoán nhiệt độ của 6 giờ kế tiếp. Ta thử chọn ra 5 ngày quan sát trước đó. Tức là ta sẽ lấy 720 (5×144) quan sát. Tương ứng ta sẽ có 2 tham số 'history_size' biểu diễn cho số quan sát trước đó và 'target_size' là số quan sát cần dự đoán.

Ta dùng 300,000 hàng đầu tiên của tập dữ liệu để làm tập dữ liệu huấn luyện, phần còn lại sẽ làm dữ liệu kiểm thử.

```
TRAIN_SPLIT = 300000
tf.random.set_seed(13) #Setting seed to ensure reproducibility.
```

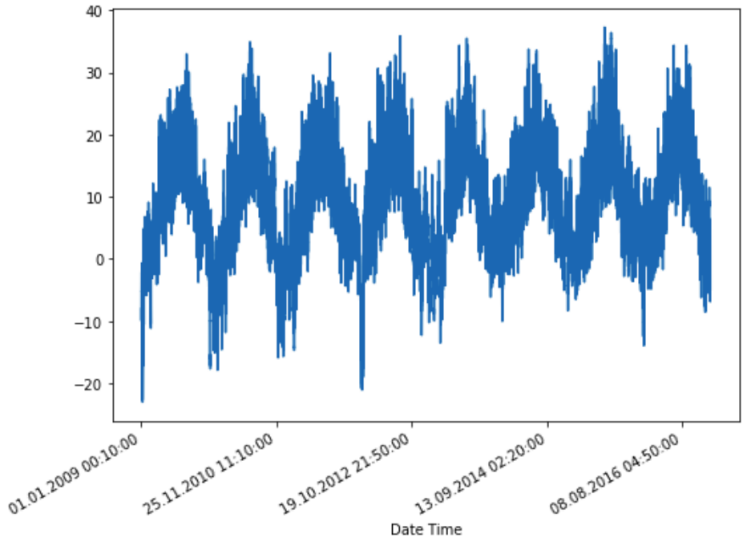
Dự đoán với đơn biến

Với trường hợp này ta sẽ chỉ dùng một giá trị duy nhất là nhiệt độ, để dự đoán nhiệt độ trong tương lai.

```
uni_data = df['T (degC)']
uni_data.index = df['Date Time']
uni_data.head()
```

Xem thử dữ liệu

```
uni_data.plot(subplots=True)
uni_data = uni_data.values
```



Để cho mô hình hoạt động hiệu quả, chính xác hơn, ta cần chuẩn hóa dữ liệu trước khi cho đi huấn luyện.

```
uni_train_mean = uni_data[:TRAIN_SPLIT].mean()
uni_train_std = uni_data[:TRAIN_SPLIT].std()
uni_data = (uni_data-uni_train_mean)/uni_train_std
```

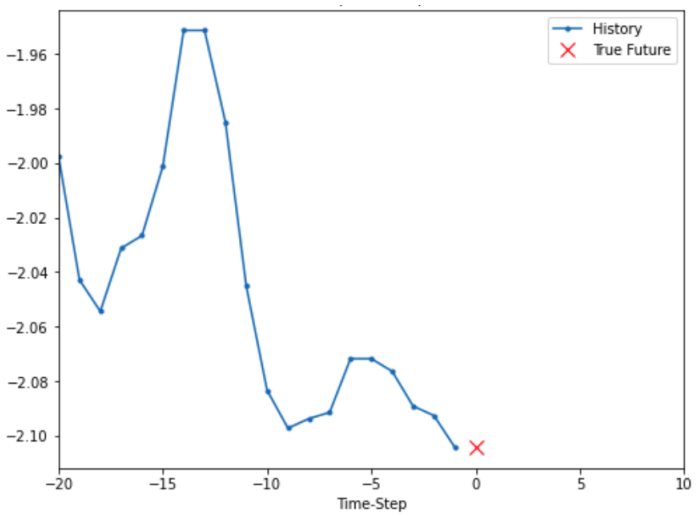
Chế biến dữ liệu lại thành tập dữ liệu huấn luyện

```
def univariate_data(dataset, start_index, end_index, history_size,
    target_size):
  data = []
  labels = []

  start_index = start_index + history_size
  if end_index is None:
    end_index = len(dataset) - target_size

```

```
  for i in range(start_index, end_index):
    indices = range(i-history_size, i)
    # Reshape data from (history_size,) to (history_size, 1)
    data.append(np.reshape(dataset[indices], (history_size, 1)))
    labels.append(dataset[i+target_size])
  return np.array(data), np.array(labels)


univariate_past_history = 20
univariate_future_target = 0

x_train_uni, y_train_uni = univariate_data(uni_data, 0, TRAIN_SPLIT
   , univariate_past_history, univariate_future_target)
x_val_uni, y_val_uni = univariate_data(uni_data, TRAIN_SPLIT, None,
univariate_past_history,
univariate_future_target)
```

Vậy là ta đã chuẩn bị xong dữ liệu, hãy xem thử nó trông như thế nào.

```
def create_time_steps(length):
  return list(range(-length, 0))

def show_plot(plot_data, delta, title):
  labels = ['History', 'True Future', 'Model Prediction']
  marker = ['.-', 'rx', 'go']
  time_steps = create_time_steps(plot_data[0].shape[0])
  if delta:
    future = delta
  else:
    future = 0

  plt.title(title)
  for i, x in enumerate(plot_data):
    if i:
      plt.plot(future, plot_data[i], marker[i], markersize=10,
      label=labels[i])
    else:
      plt.plot(time_steps, plot_data[i].flatten(), marker[i], label
   =labels[i])
  plt.legend()
  plt.xlim([time_steps[0], (future+5)*2])
  plt.xlabel('Time-Step')
  return plt

show_plot([x_train_uni[0], y_train_uni[0]], 0, 'Sample Example')
```



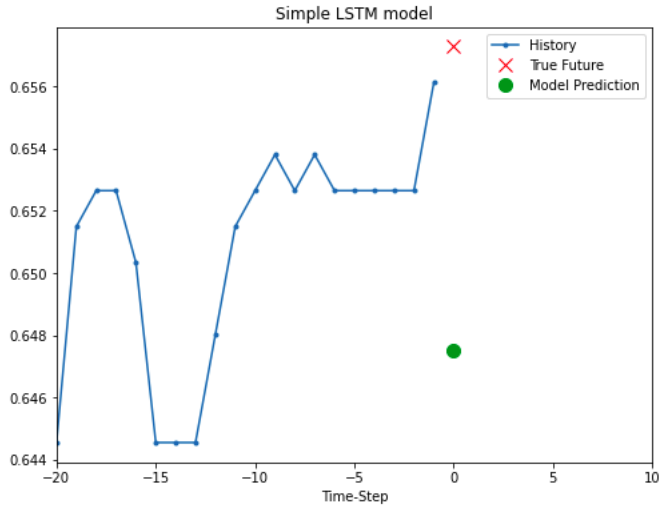
Tiến hành huấn luyện

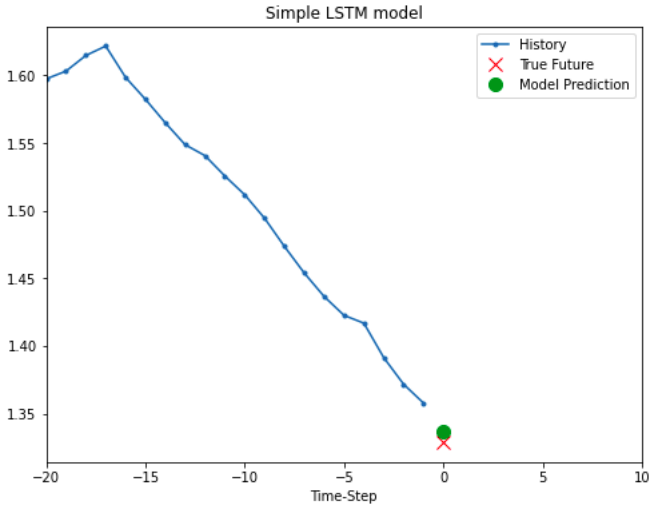
```
BATCH_SIZE = 256
BUFFER_SIZE = 10000

train_univariate = tf.data.Dataset.from_tensor_slices((x_train_uni,
    y_train_uni))
train_univariate = train_univariate.cache().shuffle(BUFFER_SIZE).
   batch(BATCH_SIZE).repeat()

val_univariate = tf.data.Dataset.from_tensor_slices((x_val_uni,
   y_val_uni))
val_univariate = val_univariate.batch(BATCH_SIZE).repeat()

simple_lstm_model = tf.keras.models.Sequential([
tf.keras.layers.LSTM(8, input_shape=x_train_uni.shape[-2:]),
tf.keras.layers.Dense(1)
])

simple_lstm_model.compile(optimizer='adam', loss='mae')

simple_lstm_model = tf.keras.models.Sequential([
tf.keras.layers.LSTM(8, input_shape=x_train_uni.shape[-2:]),
tf.keras.layers.Dense(1)
])

simple_lstm_model.compile(optimizer='adam', loss='mae')

EVALUATION_INTERVAL = 200
EPOCHS = 10
```

```

simple_lstm_model.fit(train_univariate, epochs=EPOCHS,
steps_per_epoch=EVALUATION_INTERVAL,
validation_data=val_univariate, validation_steps=50)
```

Thử một vài dự đoán

```
for x, y in val_univariate.take(3):
plot = show_plot([x[0].numpy(), y[0].numpy(),
simple_lstm_model.predict(x)[0]], 0, 'Simple LSTM model')
plot.show()
```





Dự đoán với đa biến

Dự đoán một bước (one-step)

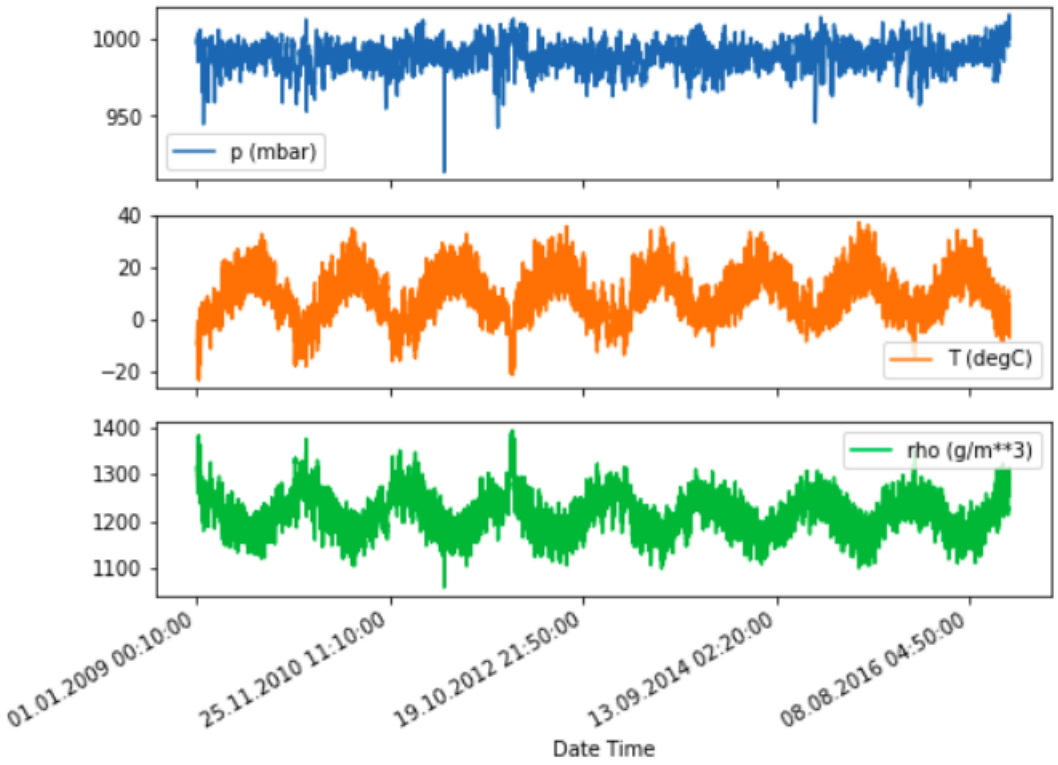
Với trường hợp này ta sẽ chỉ dùng 3 đặc trưng trong 14 đặc trưng trong tập dữ liệu, để dự đoán nhiệt độ trong tương lai.

```
features_considered = ['p (mbar)', 'T (degC)', 'rho (g/m**3)']

features = df[features_considered]
features.index = df['Date Time']
```

Xem thử dữ liệu

```
features.head()
features.plot(subplots=True)
```

Date Time	p (mbar)	T (degC)	rho (g/m**3)
01.01.2009 00:10:00	996.52	-8.02	1307.75
01.01.2009 00:20:00	996.57	-8.41	1309.80
01.01.2009 00:30:00	996.53	-8.51	1310.24
01.01.2009 00:40:00	996.51	-8.31	1309.19
01.01.2009 00:50:00	996.51	-8.27	1309.00



Tương tự như trên, ta cũng chuẩn hóa dữ liệu trước khi cho đi huấn luyện.

```
dataset = features.values
data_mean = dataset[:TRAIN_SPLIT].mean(axis=0)
data_std = dataset[:TRAIN_SPLIT].std(axis=0)

dataset = (dataset-data_mean)/data_std
```

Chế biến dữ liệu lại thành tập dữ liệu huấn luyện

```
def multivariate_data(dataset, target, start_index, end_index,
    history_size,
  target_size, step, single_step=False):
  data = []
  labels = []

  start_index = start_index + history_size
  if end_index is None:
    end_index = len(dataset) - target_size

  for i in range(start_index, end_index):
    indices = range(i-history_size, i, step)
    data.append(dataset[indices])
```

```
13
14    if single_step:
15      labels.append(target[i+target_size])
16    else:
17      labels.append(target[i:i+target_size])
18
19    return np.array(data), np.array(labels)
20
21  past_history = 720
22  future_target = 72
23  STEP = 6
24
25  x_train_single, y_train_single = multivariate_data(dataset, dataset
        [:, 1], 0,
26  TRAIN_SPLIT, past_history,
27  future_target, STEP,
28  single_step=True)
29  x_val_single, y_val_single = multivariate_data(dataset, dataset[:,
        1],
30  TRAIN_SPLIT, None, past_history,
31  future_target, STEP,
32  single_step=True)
```

Tiến hành huấn luyện

```
1  train_data_single = tf.data.Dataset.from_tensor_slices((
       x_train_single, y_train_single))
2  train_data_single = train_data_single.cache().shuffle(BUFFER_SIZE).
       batch(BATCH_SIZE).repeat()
3
4  val_data_single = tf.data.Dataset.from_tensor_slices((x_val_single,
        y_val_single))
5  val_data_single = val_data_single.batch(BATCH_SIZE).repeat()
6
7  single_step_model = tf.keras.models.Sequential()
8  single_step_model.add(tf.keras.layers.LSTM(32,
9  input_shape=x_train_single.shape[-2:]))
10 single_step_model.add(tf.keras.layers.Dense(1))
11
12 single_step_model.compile(optimizer=tf.keras.optimizers.RMSprop(),
       loss='mae')
13
14 single_step_history = single_step_model.fit(train_data_single,
       epochs=EPOCHS,
15 steps_per_epoch=EVALUATION_INTERVAL, validation_data=
       val_data_single, validation_steps=50)
```

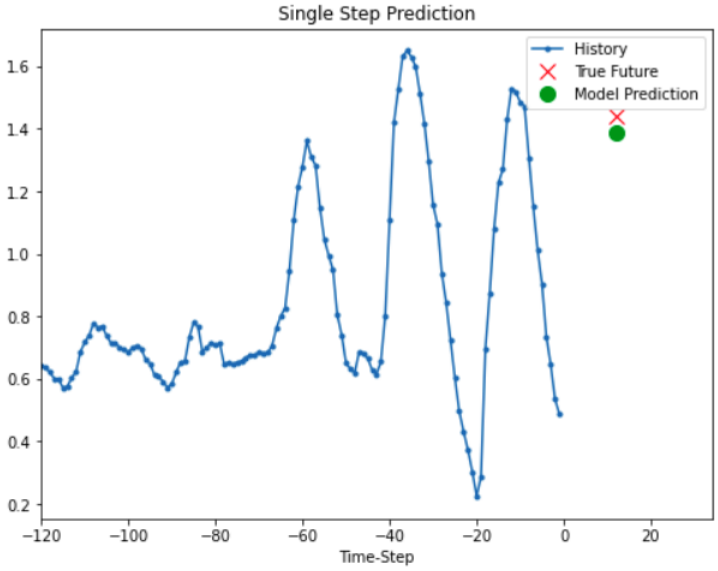
Xem quá biểu đồ biểu diễn độ lỗi

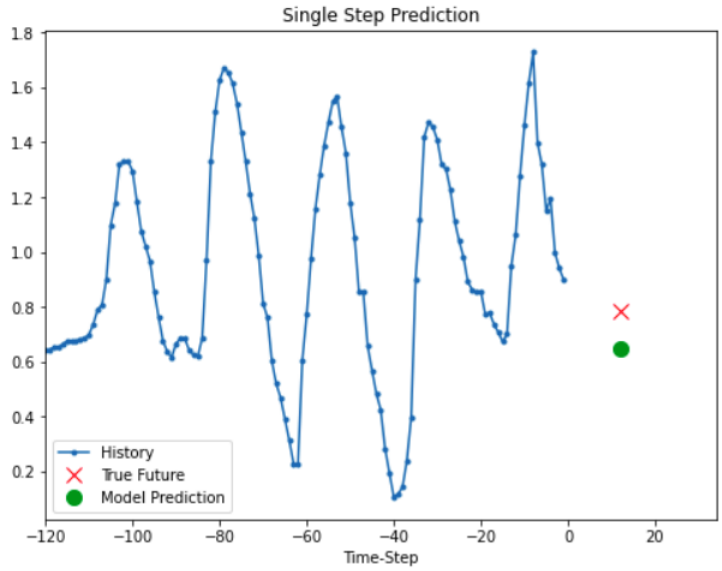
```
def plot_train_history(history, title):
  loss = history.history['loss']
  val_loss = history.history['val_loss']

  epochs = range(len(loss))

  plt.figure()

  plt.plot(epochs, loss, 'b', label='Training loss')
  plt.plot(epochs, val_loss, 'r', label='Validation loss')
  plt.title(title)
  plt.legend()

  plt.show()

plot_train_history(single_step_history,
'Single Step Training and validation loss')
```

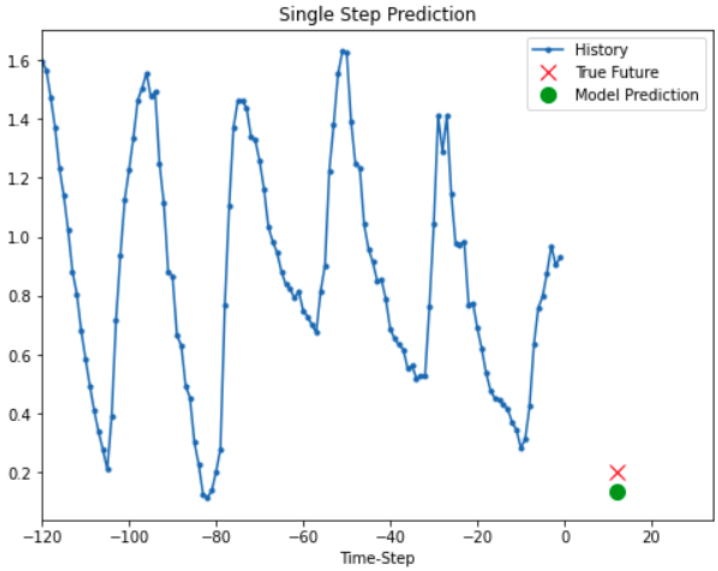
Thử một vài dự đoán

```
for x, y in val_data_single.take(3):
  plot = show_plot([x[0][:, 1].numpy(), y[0].numpy(),
```

```
  single_step_model.predict(x)[0]], 12,
  'Single Step Prediction')
plot.show()
```







Dự đoán nhiều bước (Multi-step)

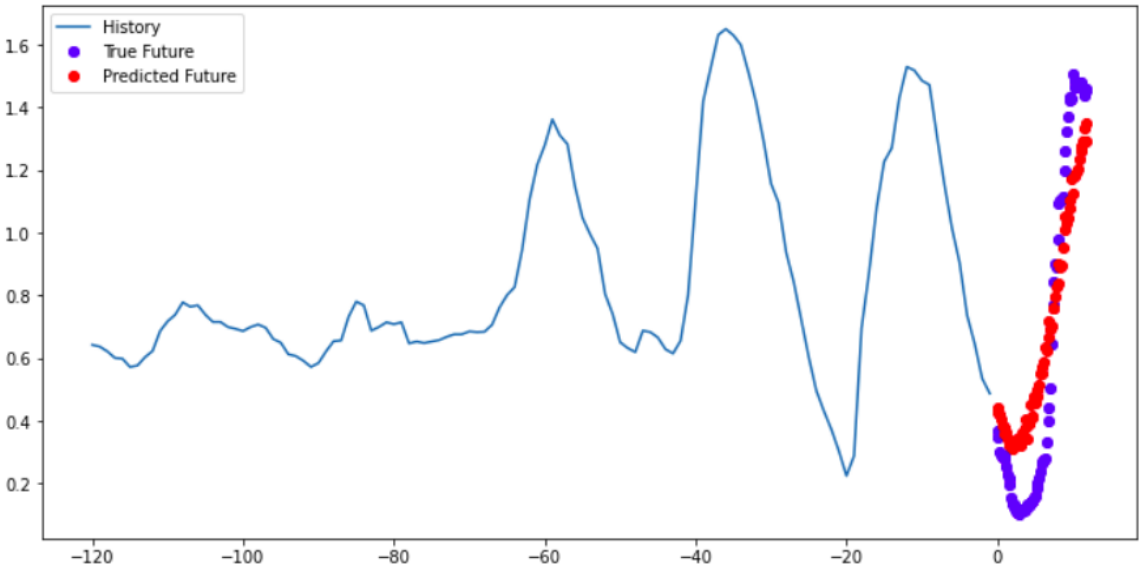
Các bước bên dưới thực hiện tương tự như trên.

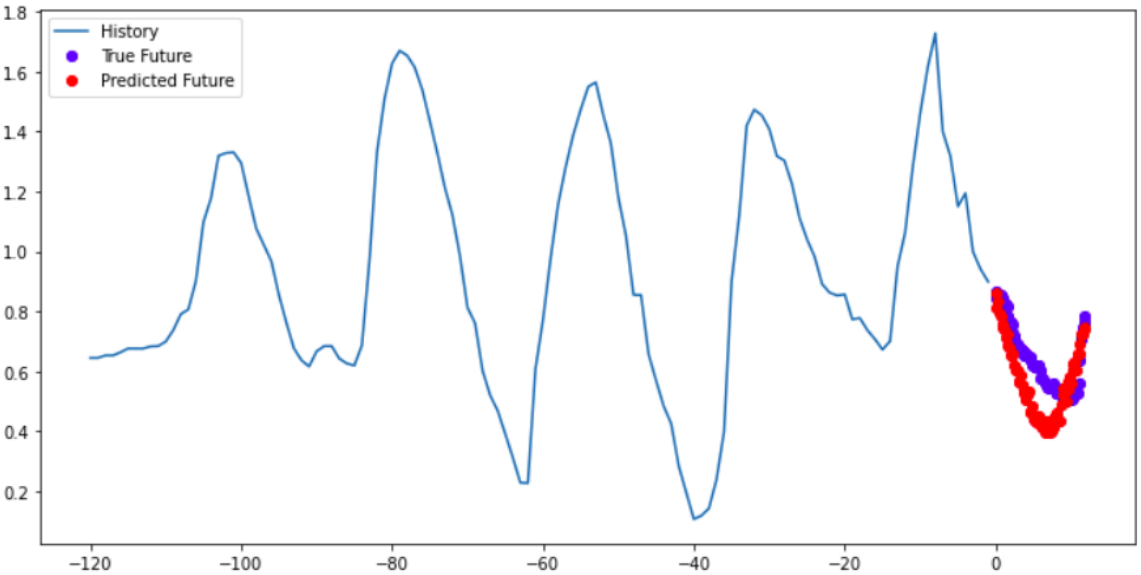
```
future_target = 72
x_train_multi, y_train_multi = multivariate_data(dataset,
    dataset[:, 1], 0,
TRAIN_SPLIT, past_history,
future_target, STEP)
x_val_multi, y_val_multi = multivariate_data(dataset, dataset
    [:, 1],
TRAIN_SPLIT, None, past_history,
future_target, STEP)

train_data_multi = tf.data.Dataset.from_tensor_slices((
    x_train_multi, y_train_multi))
train_data_multi = train_data_multi.cache().shuffle(BUFFER_SIZE
    ).batch(BATCH_SIZE).repeat()

val_data_multi = tf.data.Dataset.from_tensor_slices((
    x_val_multi, y_val_multi))
val_data_multi = val_data_multi.batch(BATCH_SIZE).repeat()

multi_step_model = tf.keras.models.Sequential()
multi_step_model.add(tf.keras.layers.LSTM(32,
return_sequences=True,
input_shape=x_train_multi.shape[-2:]))
```

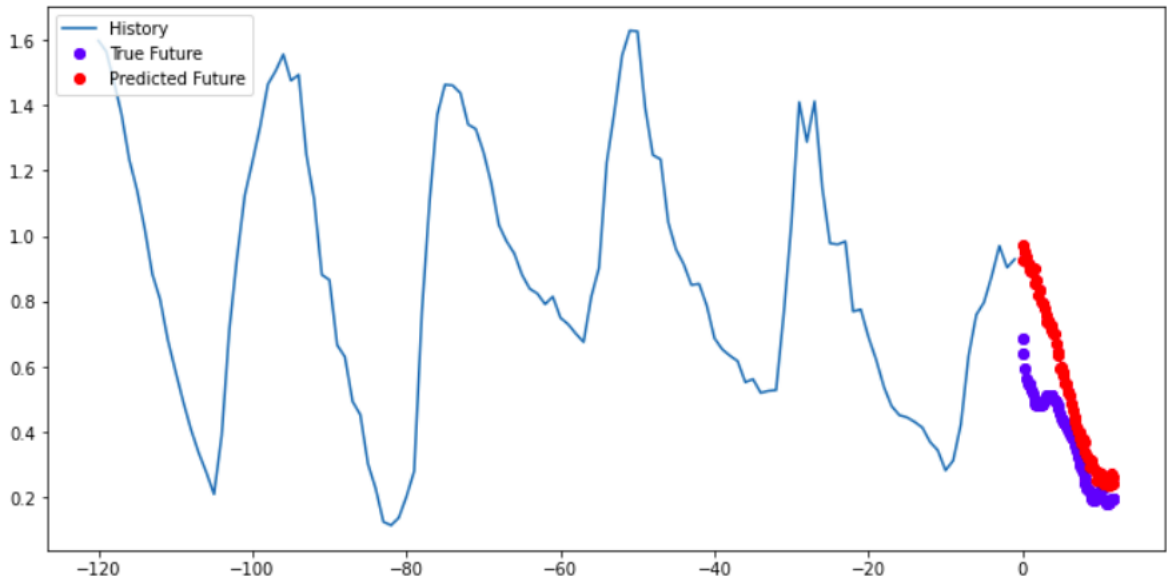
```
multi_step_model.add(tf.keras.layers.LSTM(16, activation='relu'))
multi_step_model.add(tf.keras.layers.Dense(72))

multi_step_model.compile(optimizer=tf.keras.optimizers.RMSprop(clipvalue=1.0), loss='mae')

multi_step_history = multi_step_model.fit(train_data_multi, epochs=EPOCHS,
steps_per_epoch=EVALUATION_INTERVAL,
validation_data=val_data_multi,
validation_steps=50)
```

Xem vài kết quả dự đoán

```
def multi_step_plot(history, true_future, prediction):
  plt.figure(figsize=(12, 6))
  num_in = create_time_steps(len(history))
  num_out = len(true_future)

  plt.plot(num_in, np.array(history[:, 1]), label='History')
  plt.plot(np.arange(num_out)/STEP, np.array(true_future), 'bo',
  label='True Future')
  if prediction.any():
    plt.plot(np.arange(num_out)/STEP, np.array(prediction), 'ro', label='Predicted Future')
  plt.legend(loc='upper left')
  plt.show()

for x, y in train_data_multi.take(1):
  multi_step_plot(x[0], y[0], np.array([0]))
```


History
True Future
Predicted Future
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
−120
−100
−80
−60
−40
−20
0


History
True Future
Predicted Future
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
−120
−100
−80
−60
−40
−20
0


History
True Future
Predicted Future
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
−120
−100
−80
−60
−40
−20
0

.

Chương 4: TỔNG KẾT

Kết quả.

Kết quả đạt được.

Qua những mội dung được trình bày ở trên, ta có thể thấy bài báo cáo đã trình bày một cách cơ bản và đầy đủ nhất chủ đề tìm hiểu về time series forecasting. Qua đó ta có thể hiểu được thế nào là time series, ứng dụng và cách triển khai thuật toán. Nó mang nhiều giá trị hiện thực không chỉ trong nghiên cứu khoa học mà còn có thể áp dụng vào thực tiễn của cuộc sống.

Hạn chế.

Bài báo cáo tuy đã trình bày tốt nhất có thể nhưng vẫn còn tồn tại một vài thiếu xót nhỏ. Các phần cần nêu theo mục đích của đề tài đã được hoàn thành nhưng chỉ ở mức là tổng quát, chung nhất, chưa đi vào chi tiết và trình bày cụ thể.

Hướng phát triển.

Có nhiều hướng phát triển khác nhau cho các bài time series forecasting. Ngoài việc xây dựng thuật toán theo các phương pháp trên thì còn có nhiều phương pháp khác, được thiết kế lại để tối ưu hóa hiệu năng, độ chính xác,... cho những trường hợp và mục đích riêng biệt. Ngành công nghệ thông tin ngày càng phát triển, đặc biệt là các lĩnh vực trí tuệ nhân tạo, học máy, học sâu. Nên trong tương lai, chắc chắn sẽ có thêm nhiều cách tiếp cận mới, hiệu quả và chính xác hơn. Không chỉ hữu ích trong bài toán time series forecasting mà còn nhiều bài toán khác nữa.

TÀI LIỆU THAM KHẢO

[1]. https://towardsdatascience.com/deep-learning-for-time
-series-and-why-deep-learning-a6120b147d60

[2]. https://www.slideshare.net/BeriDang/m-hnh-arima-103586847

[3]. https://viblo.asia/p/time-series-data-gDVK2Qbv5Lj

[4]. https://en.wikipedia.org/wiki/Time_series

[5]. https://arxiv.org/pdf/2004.13408.pdf

[6]. Deep Learning for Time Series Forecasting - Jason Brownlee

[7]. http://www.ijmlc.org/vol7/632-P17.pdf

[8]. https://www.tensorflow.org/tutorials/structured_data/time_series