**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**KHOA CÔNG NGHỆ THỒNG TIN**



**ĐỒ ÁN CƠ SỞ**

**Môn: Cơ Sở Trí Tuệ Nhân Tạo**

**ĐỀ TÀI: Time Series Forecasting for Stock Prices**

Sinh viên thực hiện : **3122410496 CHU THỊ BẢO YẾN**

**3122410183 DƯ NGUYỄN ĐĂNG KHOA**

**3122410220 NGUYỄN BẠCH PHÚ LỘC**

**3122410332** **TRẦN THÁI THANH PHƯƠNG**

Giảng viên hướng dẫn : **TS.** **HUỲNH MINH TRÍ**

Lớp : **841110 – NHÓM LỚP 2**

***Thành phố Hồ Chí Minh, tháng 5 năm 2025***

LỜI MỞ ĐẦU

Trong lĩnh vực trí tuệ nhân tạo và học máy, các mô hình chuỗi thời gian đóng vai trò quan trọng trong việc dự đoán các giá trị tương lai dựa trên dữ liệu lịch sử. Mô hình LSTM (Long Short-Term Memory), một loại mạng nơ-ron hồi quy (RNN) tiên tiến, đã chứng minh hiệu quả vượt trội trong việc xử lý dữ liệu chuỗi thời gian, đặc biệt trong dự đoán giá cổ phiếu. Báo cáo này trình bày một nghiên cứu toàn diện về mô hình LSTM, tập trung vào việc áp dụng để dự đoán giá cổ phiếu dựa trên dữ liệu lịch sử.

LỜI CẢM ƠN

Chúng em xin chân thành gửi lời cảm ơn đến thầy Huỳnh Minh Trí, giảng viên bộ môn “Cơ sở trí tuệ nhân tạo” thuộc Khoa Công Nghệ Thông Tin, trường Đại học Sài Gòn đã tin tưởng giao đề tài cho nhóm.

Với vốn kiến thức hạn hẹp và còn thiếu kinh nghiệm trong các lĩnh vực liên quan đến báo cáo, nên báo cáo còn chưa đầy đủ thông tin và các sai sót trong việc xây dựng báo cáo. Rất mong nhận được sự quan tâm và chỉ bảo của thầy để báo cáo này được hoàn thiện hơn.

Xin kính chúc thầy Huỳnh Minh Trí thật nhiều sức khỏe để tiếp tục sự nghiệp dạy bảo thế hệ tương lai đất nước.

NHẬN XÉT

MỤC LỤC

**Giới thiệu**

**Cấu trúc báo cáo**

**Tổng quan**

**Nghiên cứu tổng quan  
4.1. Các nghiên cứu ban đầu và cơ bản  
4.2. Cảm hứng kỹ thuật  
4.3. Các biến thể và cải tiến của LSTM  
4.4. Ứng dụng của LSTM trong các lĩnh vực**

**Phân tích thiết kế hệ thống  
5.1. Mô hình tổng quan của hệ thống nghiên cứu  
5.2. Toán học hóa các thành phần chính  
5.3. Mã giả  
5.4. Các tham số của mô hình  
5.5. Ưu điểm và nhược điểm của LSTM**

**Triển khai xây dựng  
6.1. Các bước triển khai  
6.2. Môi trường và công cụ lập trình  
6.3. Hiệu suất và độ phức tạp**

**Kết luận và hướng phát triển  
7.1. Tóm tắt  
7.2. Hướng phát triển  
7.3. Kết luận**

**Danh mục tài liệu tham khảo**

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

DANH MỤC CỤM TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Cụm từ | Viết tắt |
| 1 | Long Short-Term Memory | LSTM |
| 2 | Recurrent Neural Network | RNN |
| 3 | Công nghệ thông tin | CNTT |
| 4 | Root Mean Square Error | RMSE |
| 5 | AutoRegressive Integrated Moving Average | ARIMA |
| 6 | Convolutional Neural Network | CNN |

CHƯƠNG 1 GIỚI THIỆU

## **1.1 Cấu trúc báo cáo**

Báo cáo này được tổ chức để cung cấp cái nhìn toàn diện về mô hình LSTM trong dự đoán giá cổ phiếu, bao gồm các nội dung sau:

* Giới thiệu: Giới thiệu bối cảnh của các mô hình chuỗi thời gian và vai trò của LSTM.
* Nghiên cứu tổng quan: Trình bày các nghiên cứu ban đầu về LSTM, cảm hứng kỹ thuật, các biến thể, và ứng dụng trong nhiều lĩnh vực.
* Phân tích thiết kế hệ thống: Phân tích cấu trúc của LSTM, toán học hóa các thành phần chính, mã giả, tham số, và ưu/nhược điểm.
* Triển khai xây dựng: Mô tả các bước triển khai, công cụ lập trình, và phân tích hiệu suất.
* Kết luận và hướng phát triển: Tóm tắt các điểm chính và đề xuất hướng nghiên cứu tương lai.

## **1.2 Tổng quan**

Mô hình LSTM (Long Short-Term Memory), một dạng mạng nơ-ron hồi quy (RNN) tiên tiến, được thiết kế để xử lý và dự báo dữ liệu chuỗi thời gian với các mối quan hệ dài hạn. Được giới thiệu bởi Hochreiter và Schmidhuber (1997), LSTM khắc phục hạn chế “vanishing gradient” của RNN truyền thống nhờ cấu trúc ô nhớ (memory cells) và ba cổng điều chỉnh thông tin: cổng quên (forget gate), cổng đầu vào (input gate), và cổng đầu ra (output gate). Các cổng này cho phép LSTM lưu giữ và truy xuất thông tin quan trọng qua nhiều bước thời gian, giúp mô hình hóa hiệu quả các xu hướng phức tạp và tính thời vụ trong dữ liệu.

Trong dự báo chuỗi thời gian giá cổ phiếu, LSTM tỏ ra vượt trội nhờ khả năng nắm bắt các mô hình phi tuyến và biến động thị trường, vốn khó xử lý bởi các phương pháp thống kê truyền thống như ARIMA hay PROPHET. Bằng cách phân tích dữ liệu giá lịch sử, LSTM có thể dự đoán giá đóng cửa hoặc xu hướng tương lai với độ chính xác cao. Mô hình thường được triển khai thông qua các thư viện như KERAS, sử dụng dữ liệu được tiền xử lý (chuẩn hóa, chia thành chuỗi) và huấn luyện với bộ tối ưu hóa như Adam. Hiệu suất của LSTM thường được đánh giá bằng sai số bình phương trung bình (RMSE), cho thấy khả năng cạnh tranh trong các bài toán tài chính.

Ứng dụng của LSTM trong dự báo giá cổ phiếu không chỉ giới hạn ở việc dự đoán giá trị mà còn hỗ trợ phân tích rủi ro và ra quyết định đầu tư. Tuy nhiên, mô hình yêu cầu tiền xử lý dữ liệu cẩn thận, tối ưu hóa tham số kỹ lưỡng, và chi phí tính toán cao hơn so với các phương pháp đơn giản. Với sự phát triển của các biến thể như GRU hay Bidirectional LSTM, cùng với các kỹ thuật cải tiến như attention mechanisms, LSTM tiếp tục là một công cụ mạnh mẽ và đầy tiềm năng trong lĩnh vực dự báo chuỗi thời gian tài chính.

# **CHƯƠNG 2** **NGHIÊN CỨU TỔNG QUAN**

## **2.1 Chi tiết thuật toán**

LSTM bao gồm các ô nhớ được kết nối tuần tự, mỗi ô xử lý một bước thời gian trong chuỗi dữ liệu. Mỗi ô có ba cổng chính:  
- Cổng quên (Forget Gate): Quyết định thông tin nào từ trạng thái ô trước đó cần loại bỏ.  
- Cổng đầu vào (Input Gate): Quyết định thông tin mới nào được thêm vào trạng thái ô.  
- Cổng đầu ra (Output Gate): Quyết định thông tin nào từ trạng thái ô được sử dụng làm đầu ra.  
Cấu trúc này cho phép LSTM duy trì và cập nhật trạng thái ô (cell state) và trạng thái ẩn (hidden state) qua nhiều bước thời gian.

// thiếu cái cổng quên

**2.1.2 Đầu vào và đầu ra**

Đầu vào: Một chuỗi giá cổ phiếu (thường là giá đóng cửa) với độ dài cố định (ví dụ: 60 ngày), định dạng thành tensor có kích thước [số mẫu, độ dài chuỗi, 1].  
Đầu ra: Giá trị dự đoán cho bước thời gian tiếp theo (giá cổ phiếu ngày kế tiếp).

**2.1.3 Toán học hóa Các Thành phần Chính**

Các phương trình dưới đây mô tả quá trình xử lý dữ liệu tại mỗi bước thời gian t trong LSTM:

**Cổng quên (Forget Gate)**

fₜ = σ(W\_f · [hₜ₋₁, xₜ] + b\_f)

**Cổng đầu vào (Input Gate)**

iₜ = σ(W\_i · [hₜ₋₁, xₜ] + b\_i)  
C̃ₜ = tanh(W\_C · [hₜ₋₁, xₜ] + b\_C)

**Cập nhật trạng thái ô**

Cₜ = fₜ \* Cₜ₋₁ + iₜ \* C̃ₜ

**Cổng đầu ra (Output Gate)**

oₜ = σ(W\_o · [hₜ₋₁, xₜ] + b\_o)  
hₜ = oₜ \* tanh(Cₜ)

**Dự đoán cuối cùng**

ŷₜ = W\_d · hₜ + b\_d

// giải thích các thuật ngữ

**2.1.4 Quy trình Thuật toán**

Bước 1: Thu thập dữ liệu giá cổ phiếu lịch sử.  
Bước 2: Chuẩn hóa dữ liệu bằng MinMaxScaler.  
Bước 3: Tạo chuỗi thời gian với độ dài cố định.  
Bước 4: Chia dữ liệu thành tập huấn luyện và kiểm tra.  
Bước 5: Xây dựng mô hình với các lớp LSTM và Dense.  
Bước 6: Cấu hình huấn luyện (loss=MSE, optimizer=Adam).  
Bước 7: Huấn luyện mô hình.  
Bước 8: Dự đoán và đánh giá bằng RMSE.

## **2.2 Ưu và nhược điểm**

**Ưu điểm của LSTM:**

1. **Ghi nhớ dài hạn hiệu quả**  
   LSTM có khả năng lưu giữ thông tin từ các bước thời gian xa trong quá khứ nhờ trạng thái ô (cell state), phù hợp với dữ liệu có tính tuần hoàn hoặc xu hướng dài hạn như giá cổ phiếu.
2. **Xử lý dữ liệu phi tuyến tốt**  
   Với các hàm kích hoạt như sigmoid và tanh, LSTM có thể học và mô hình hóa các quan hệ phi tuyến trong chuỗi thời gian.
3. **Giảm thiểu vấn đề mất mát thông tin (vanishing gradient)**  
   So với RNN truyền thống, LSTM giải quyết tốt hơn vấn đề gradient bị tiêu biến khi lan truyền ngược qua nhiều bước thời gian.
4. **Linh hoạt và mở rộng dễ dàng**  
   Có thể mở rộng bằng cách xếp chồng nhiều lớp LSTM, kết hợp với các tầng Dense, Dropout hoặc thậm chí các mạng CNN để cải thiện hiệu suất.
5. **Ứng dụng rộng rãi và đã được kiểm chứng**  
   LSTM được sử dụng thành công trong nhiều bài toán chuỗi thời gian khác như dự đoán nhu cầu, phân tích ngữ cảnh, dịch máy,…

**Nhược điểm của LSTM:**

1. **Chi phí tính toán cao**  
   Do cấu trúc phức tạp với nhiều ma trận trọng số, LSTM yêu cầu nhiều tài nguyên phần cứng và thời gian huấn luyện, đặc biệt với dữ liệu lớn.
2. **Cần nhiều dữ liệu huấn luyện**  
   Mô hình dễ bị overfitting nếu dữ liệu huấn luyện không đủ lớn hoặc không đại diện cho các mẫu biến động thực tế.
3. **Khó giải thích**  
   LSTM là mô hình hộp đen (black-box), khó diễn giải và không trực quan để lý giải vì sao mô hình đưa ra một dự đoán cụ thể.
4. **Nhạy cảm với siêu tham số**  
   Các tham số như số đơn vị ẩn, learning rate, batch size,... cần được tinh chỉnh cẩn thận. Việc lựa chọn không phù hợp có thể khiến mô hình hoạt động kém.
5. **Không tận dụng tốt mối liên hệ không tuần tự**  
   Nếu mối quan hệ giữa các thời điểm không liên tiếp là quan trọng (ví dụ sự kiện thị trường bất ngờ), LSTM có thể không nhận diện tốt bằng mô hình attention như Transformer.

# **CHƯƠNG 3 THỰC NGHIỆM**

## **3.2 Mã giả**

// 1. Import libraries

IMPORT keras, numpy, pandas, sklearn.preprocessing

// 2. Load and preprocess data

FUNCTION load\_data(file\_path):

    data = READ\_CSV(file\_path)

    prices = data['Close']

    RETURN prices

// 3. Prepare time series data

FUNCTION prepare\_data(prices, sequence\_length):

    X, y = EMPTY\_ARRAYS

    FOR i FROM 0 TO length(prices) - sequence\_length:

        X[i] = prices[i:i+sequence\_length]

        y[i] = prices[i+sequence\_length]

    scaler = MinMaxScaler()

    X\_normalized = scaler.fit\_transform(X)

    y\_normalized = scaler.transform(y)

    X\_reshaped = RESHAPE(X\_normalized, [samples, sequence\_length, 1])

    RETURN X\_reshaped, y\_normalized, scaler

// 4. Split data

FUNCTION split\_data(X, y, train\_ratio):

    split\_idx = INTEGER(length(X) \* train\_ratio)

    X\_train = X[:split\_idx]

    y\_train = y[:split\_idx]

    X\_test = X[split\_idx:]

    y\_test = y[split\_idx:]

    RETURN X\_train, y\_train, X\_test, y\_test

// 5. Build LSTM model

FUNCTION build\_lstm\_model(sequence\_length, units):

    model = Sequential()

    model.add(LSTM(units=units, input\_shape=(sequence\_length, 1), return\_sequences=False))

    model.add(Dense(units=1))

    model.compile(optimizer='adam', loss='mean\_squared\_error')

    RETURN model

// 6. Train model

FUNCTION train\_model(model, X\_train, y\_train, epochs, batch\_size):

    history = model.fit(X\_train, y\_train, epochs=epochs, batch\_size=batch\_size, validation\_split=0.2, shuffle=False)

    RETURN history

// 7. Evaluate model

FUNCTION evaluate\_model(model, X\_test, y\_test, scaler):

    y\_pred\_normalized = model.predict(X\_test)

    y\_pred = scaler.inverse\_transform(y\_pred\_normalized)

    y\_test = scaler.inverse\_transform(y\_test)

    rmse = SQRT(MEAN((y\_pred - y\_test)^2))

    RETURN rmse, y\_pred, y\_test

// 8. Main function

FUNCTION main():

    file\_path = 'stock\_data.csv'

    sequence\_length = 60

    units = 50

    epochs = 50

    batch\_size = 32

    train\_ratio = 0.8

    prices = load\_data(file\_path)

    X, y, scaler = prepare\_data(prices, sequence\_length)

    X\_train, y\_train, X\_test, y\_test = split\_data(X, y, train\_ratio)

    model = build\_lstm\_model(sequence\_length, units)

    history = train\_model(model, X\_train, y\_train, epochs, batch\_size)

    rmse, y\_pred, y\_test = evaluate\_model(model, X\_test, y\_test, scaler)

    PRINT("Test RMSE:", rmse)

    model.save('lstm\_stock\_model.h5')

**CHƯƠNG 6 TRIỂN KHAI VÀ XÂY DỰNG**

**6.1 Các bước triển khai**

Ta sẽ có quy trình sau:

| **Bước** | | **Mô tả** |
| --- | --- | --- |
| **1** | Lấy dữ liệu cổ phiếu | |
| **2** | Chuẩn hóa và tạo tập huấn luyện | |
| **3** | Xây dựng mô hình LSTM | |
| **4** | Huấn luyện mô hình | |
| **5** | Dự đoán và vẽ biểu đồ | |

**6.2 Môi trường và công cụ lập trình**

**Cài đặt thư viện:**

**Trên máy tính Window , mở Command Prompt và gõ :**

“ pip install yfinance numpy pandas matplotlib scikit-learn tensorflow ”

**Giải thích từng thư viện:**

| **Thư viện** | **Vai trò** |
| --- | --- |
| **yfinance** | Lấy dữ liệu giá cổ phiếu từ Yahoo Finance |
| **numpy** | Xử lý dữ liệu dạng mảng, ma trận |
| **pandas** | Xử lý dữ liệu dạng bảng (DataFrame) |
| **matplotlib** | Vẽ biểu đồ kết quả |
| **scikit-learn** | Tiền xử lý dữ liệu (MinMaxScaler) |
| **tensorflow** | Xây dựng và huấn luyện mô hình LSTM |

**Lưu ý : cài bản Python từ 3.11.x đổ xuống vì tenserflow 2.x chưa hỗ trợ những bản Python mới hơn.**

**Môi trường và công cụ lập trình**

Tùy vào mục đích và khả năng máy tính của bạn, dưới đây là so sánh các môi trường tốt nhất để triển khai mô hình LSTM dự đoán giá cổ phiếu:

**1/ . Google Colab – Khuyên dùng cho người mới & máy yếu**

| **Ưu điểm** | **Nhược điểm** |
| --- | --- |
| - Miễn phí GPU/TPU- Không cần cài Python | - Giới hạn thời gian chạy (~12h) - Không lưu file lâu dài |
| - Có thể chia sẻ notebook dễ dàng | - Phụ thuộc vào internet |
| - Hỗ trợ tốt các thư viện ML | - Có thể bị ngắt nếu để lâu không hoạt động |

**2/ Jupyter Notebook (cài trên máy)**

| **Ưu điểm** | **Nhược điểm** |
| --- | --- |
| - Linh hoạt, chạy offline | - Cần cấu hình môi trường thủ công |
| - Giao diện tương tác trực quan | - Máy yếu chạy mô hình chậm |
| - Tốt cho debug và trình bày từng bước |  |

**3/ VS Code + Python (code thuần, không notebook)**

| **Ưu điểm** | **Nhược điểm** |
| --- | --- |
| - Linh hoạt nhất cho dự án lớn | - Không trực quan như notebook |
| - Tích hợp Git, debug mạnh | - Mất thời gian setup môi trường ban đầu |
| - Phù hợp để triển khai API, ứng dụng backend |  |

Dựa trên các đặc điểm của các môi trường lập trình trên, ta sẽ dùng môi trường Google Colab để có thể dễ dàng demo ví dụ

**Tập dữ liệu:**

Dữ liệu ta thu thập từ trang web <https://finance.yahoo.com/quote/AAPL/history?period1=1325376000&period2=1578614400&interval=1d&filter=history&frequency=1d> , ta lấy từ thời gian 03/01/2012 đến 09/01/2020. Hãy xem dữ liệu của chúng ta. Mở tập tin Apple stock price chứa dữ liệu trong 8 năm. Bạn sẽ thấy nó chứa 7 cột: date, open, high, low, low, close, adj và volume. Ta sẽ dự đoán giá cổ phiếu Open, vì vậy ta không quan tâm đến những cột còn lại.

**Data Train:**

Về bộ dữ liệu để cho training thì ta lấy 80% từ bộ dữ liệu ta tải về.

**Data Test:**

Về bộ test thì ta sẽ lấy 20% từ bộ dữ liệu ta tải về.

**Môi trường lập trình :**

Hiện tại ta dùng google colab để lập trình, cùng với sử dụng thư viện keras.

**Thực hiện lập trình và giải thích chi tiết.**

Đầu tiên chúng ta sẽ xem xu hướng giá trị cổ phiểu của công ty APPLE COMPANY như thế nào đã nhé A graph with a line

AI-generated content may be incorrect.

Table 1 -Giá trị cổ phiếu của công ty APPLE COMPANY

**Kết nối với tài khoản google driver:**

Do tải file dữ liệu lên google driver nên ta cần liên kết google colab với google driver của ta để tí nữa ta có thể lấy dữ liệu ra từ google driver và tiến hành xử lý dữ liệu trước khi cho vào mô hình mạng.

A close-up of a text

AI-generated content may be incorrect.

**Import Libraries**A screen shot of a computer

AI-generated content may be incorrect.

Chúng ta import 1 số thư viện quan trọng, để tí nữa chúng ta sử dụng. Ở đây ta sử dụng các thư viện:

* numpy: Giúp xử lý số liệu
* panda: dùng để đọc dữ liệu
* pyplot: Dùng để vẽ biểu đồ
* MinMaxScaler: Dùng để chuẩn hóa dữ liệu
* Và một số thư viện để lập trình cho mô hình mạng dựa trên thư viện keras

**Import Dataset**

Thực hiện lấy data bằng thư viện panda, sau đó tiến hành chia dữ liệu thành 2 bộ, một bộ train chiếm 80% dữ liệu chính, và một bộ test chiếm 20% còn lại của dữ liệu.

A computer screen shot of a code

AI-generated content may be incorrect.

**Xử lý dữ liệu:**

A computer code with many black and white text

AI-generated content may be incorrect.

Trước hết để giải thích hàm trên thì ta xin giải thích một số điều như sau:

* time\_step: Trong bài toán này thì bạn hiểu là nếu bạn muốn cứ 30 giá trị của Open thì đoán 1 giá trị open tiếp theo thì time\_step ở đây bằng 30. Còn num\_predict là 1. Từ đó ta thấy hàm get\_data ở trên mục đích là định dạng lại dữ liệu để tí có thể đưa vào mạng. Ví dụ, sau khi qua hàm get\_data thì: x\_train = [[1,2,3,4,5,6,7,8,9,10],[2,3,4,5,6,7,8,9,10,11]] và y\_train = [11,12] Bản chất là ta muốn dùng [1,2,3,4,5,6,7,8,9,10] để đoán ra 11, [2,3,4,5,6,7,8,9,10,11] để đoán ra 12

A screenshot of a computer program

AI-generated content may be incorrect.

Ở đây, ta đã code theo kiểu lấy 30 dữ liệu để đoán 1 dữ liệu tiếp theo. Sau đó ta chuẩn hóa dữ liệu về dạng từ 0 đến 1, theo hàm MinMaxScaler() cho bộ train và test. Mục đích của chuẩn hóa là để tí nữa vào mô hình mạng nó tối ưu tốt hơn. Tiếp theo, chúng ta sẽ reshape lại cho x\_train và y\_train :

A white background with black and blue text

AI-generated content may be incorrect.

Tại sao phải reshape vậy, bởi vị theo quy chuẩn của keras thì đầu vào của LSTM có dạng (batch\_size, time\_steps, feature) như sau:A diagram of a computer network

AI-generated content may be incorrect.

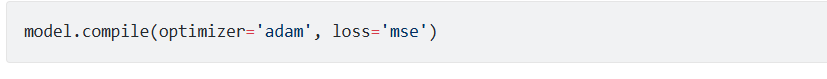
Batch size: Cứ hiểu là có bao nhiêu cặp (time\_steps, feature) ấy time\_steps: Như trình bày ở trên rồi feature: là có bao nhiêu thuộc tính của mỗi phần tử trong time\_step. Ví dụ : time\_step có 10 giá trị (mỗi giá trị là một vector), mỗi vector là một giá trị 2 chiều chẳng hạn, thì feature ở đây là 2 (tức 2 chiều đó ). Tóm cái váy lại, thì feature cứ hiểu là số thuộc tính của mỗi phần tử time\_step. Còn reshape đầu ra mục đích là tí cho hợp với shape đầu ra của mô hình mạng. Ở trên ta thấy dùng 30 để đoán 1, nên đầu ra ở đây phải reshape theo (-1,1).

**Xây dựng mô hình mạng:**

A screenshot of a computer code

AI-generated content may be incorrect.

Bước đầu tiên, chúng ta cần khởi tạo lớp Sequential. Đây sẽ là lớp mô hình của chúng tôi và chúng tôi sẽ thêm các lớp LSTM, Dropout và Dense cho mô hình này. Ở trên ta thêm vào 3 lớp LSTM liên tiếp, và cứ qua 1 lớp là có 1 dropout 0.3 . Cuối cùng ta cho qua một tầng Dense với đầu ra là 1 chiều.



Ta sử dụng hàm loss bình phương trung bình là hàm mất mát và để tối ưu hóa thuật toán, chúng tôi sử dụng trình tối ưu hóa adam.

**Huấn luyện mạng**

A math equation with black text

AI-generated content may be incorrect.

Thực hiện huấn luyện mạng với validation là 0.2, batch\_size =30 và tiến hành lưu lại mô hình

**Chạy kết quả test**

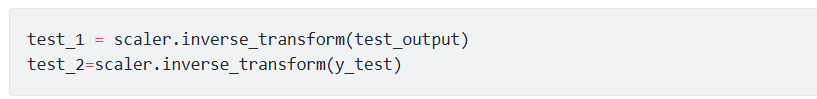
A computer screen shot of a program

AI-generated content may be incorrect.

A graph with red and blue lines

AI-generated content may be incorrect.

Ta thực hiện lấy model đã lưu ra và tiến hành dự đoán bằng test\_output = model.predict(x\_test) Sau khi dự đoán xong thì chúng ta thực hiện inverse transform lại, do lúc nãy ta đã chuẩn hóa chúng nằm trong 0->1, bây giờ ta chuyển nó về đúng giá trị thật của chúng bằng đoạn mã sau:



Và chúng ta có thể thấy mô hình mạng của ta dự đoán giá trị cổ phiếu khá chính xác phải không nào.

**6.3 Hiệu suất và độ phức tạp**

**6.3.1 Sự phức tạp**

**a. Phức tạp về mô hình (LSTM)**

* LSTM là một loại mạng nơ-ron hồi tiếp (RNN) với kiến trúc phức tạp hơn so với các mô hình học máy truyền thống. Chúng có 3 cổng chính (forget, input, output) để kiểm soát thông tin, điều này giúp mô hình "ghi nhớ" thông tin dài hạn trong chuỗi thời gian.
* Mô hình này có nhiều tham số cần tối ưu, đặc biệt là khi sử dụng nhiều lớp LSTM như trong ví dụ trên. Mỗi lớp LSTM có thể có hàng ngàn tham số cần huấn luyện.

**b. Khối lượng tính toán**

* Mô hình LSTM yêu cầu tính toán phức tạp và thời gian huấn luyện lâu, đặc biệt khi số lượng dữ liệu lớn hoặc mô hình có nhiều tham số.
* Xử lý chuỗi dài là điểm mạnh của LSTM, nhưng nó cũng có thể dẫn đến sự chậm trễ trong việc huấn luyện và dự đoán nếu không được tối ưu hóa tốt.

**c. Tiền xử lý dữ liệu**

* Dữ liệu cần được chuẩn hóa để các giá trị trong dãy thời gian không có độ lệch quá lớn, đảm bảo quá trình huấn luyện ổn định.
* Quá trình tạo sliding window và chia nhỏ dữ liệu thành các chuỗi nhỏ (60 ngày) có thể làm tăng độ phức tạp khi số lượng dữ liệu quá lớn.

**6.3.2 Hiệu suất mô hình (Model Performance)**

**a. Độ chính xác (Accuracy)**

* Độ chính xác của mô hình LSTM phụ thuộc vào nhiều yếu tố như:
* Chất lượng dữ liệu: Nếu dữ liệu có nhiều nhiễu hoặc thiếu dữ liệu quan trọng, mô hình sẽ dự đoán không chính xác.
* Sự lựa chọn tham số: Số lớp, số đơn vị trong mỗi lớp, số epoch và batch size đều ảnh hưởng lớn đến kết quả.
* Dự đoán giá cổ phiếu là một bài toán phức tạp vì giá cổ phiếu có sự biến động không thể đoán trước do các yếu tố như tin tức, sự kiện toàn cầu, và tâm lý thị trường. Mặc dù LSTM có thể đạt được độ chính xác cao với dữ liệu lịch sử, nhưng nó không thể hoàn toàn dự đoán chính xác trong môi trường đầy biến động.

**b. Tốc độ huấn luyện và dự đoán**

* Mặc dù LSTM là mạnh mẽ, nhưng tốc độ huấn luyện của nó có thể khá chậm, đặc biệt nếu bạn không có phần cứng tốt (GPU).
* Sử dụng GPU có thể làm tăng hiệu suất huấn luyện rất nhiều, vì các phép toán ma trận trong LSTM có thể được xử lý song song hiệu quả.
* Nếu mô hình được huấn luyện trên CPU, thời gian huấn luyện có thể kéo dài từ vài giờ đến vài ngày tùy thuộc vào khối lượng dữ liệu.

**c. Overfitting**

* LSTM dễ bị overfitting nếu số lượng tham số quá lớn và dữ liệu huấn luyện không đủ đa dạng.
* Regularization (ví dụ: Dropout) và early stopping có thể giúp giảm thiểu overfitting.

**6.3.3 Các yếu tố khác ảnh hưởng đến hiệu suất**

* Dữ liệu lịch sử: LSTM chỉ học được từ dữ liệu lịch sử, do đó nếu các sự kiện ngoài tầm kiểm soát (như khủng hoảng kinh tế hoặc sự kiện chính trị lớn) xảy ra, mô hình có thể không phản ánh đúng biến động.
* Tuning mô hình: Tuning các tham số như số lượng layers, số units trong mỗi layer, số lượng epoch, và batch size là rất quan trọng để đạt được hiệu suất tối ưu.
* Sử dụng các kỹ thuật bổ sung: Các mô hình ensemble, chẳng hạn như kết hợp LSTM với mô hình ARIMA hay các thuật toán học máy khác, có thể cải thiện độ chính xác.

**6.4 Kết luận và hướng phát triển**

**6.4.1 Hướng phát triển của LSTM**

* Mặc dù LSTM là một mô hình mạnh mẽ, nó vẫn có những giới hạn. Trong tương lai (và hiện tại), LSTM đang được mở rộng và thay thế/đi kèm với các kỹ thuật mới:

**6.4.1.1 Kết hợp với các mô hình khác (Hybrid Models)**

* LSTM + ARIMA: Kết hợp mô hình thống kê truyền thống với LSTM để tận dụng cả tuyến tính và phi tuyến tính.
* LSTM + Attention: Thêm cơ chế "chú ý" (Attention) giúp mô hình tập trung vào các thời điểm quan trọng trong chuỗi thời gian.
* LSTM + CNN: Sử dụngmạng tích chập để trích xuất đặc trưng trước khi đưa vào LSTM.

**6.4.1.2 Thay thế bằng các kiến trúc tiên tiến hơn**

* GRU (Gated Recurrent Unit): Đơn giản hơn LSTM, ít tham số, đôi khi cho kết quả tương đương hoặc tốt hơn.
* Transformer/Attention-based Models: Được chứng minh vượt trội trong nhiều bài toán chuỗi thời gian, đặc biệt là mô hình như:
* Informer, Autoformer, TimesNet
* Đây là xu hướng mới vì mô hình này huấn luyện nhanh hơn và xử lý chuỗi dài tốt hơn so với LSTM.

**6.4.1.3 Tăng cường dữ liệu và học tập chuyển giao (Transfer Learning)**

* Dùng dữ liệu từ nhiều cổ phiếu/ngành khác nhau để cải thiện khả năng tổng quát.
* Ứng dụng pretrained LSTM để giảm chi phí huấn luyện trên dữ liệu mới.

**6.4.2 Kết luận**

| **Ưu điểm** | **Giải thích** |
| --- | --- |
| **Xử lý dữ liệu tuần tự tốt** | Ghi nhớ thông tin trong chuỗi dài (long-term dependencies) |
| **Linh hoạt** | Áp dụng được cho nhiều dạng dữ liệu thời gian (tài chính, khí hậu, IoT, v.v.) |
| **Mạnh mẽ hơn RNN truyền thống** | Giảm hiện tượng mất nhớ (vanishing gradient) |

| **Nhược điểm** | **Giải thích** |
| --- | --- |
| **Chậm và nặng** | Huấn luyện tốn thời gian, không tối ưu cho chuỗi cực dài |
| **Dễ overfitting** | Nếu không có đủ dữ liệu hoặc không dùng regularization |
| **Không thể hiện tốt tính toàn cục (global)** | Mỗi bước chỉ nhìn thấy một phần chuỗi thời gian, không thấy toàn cảnh như Transformer |

**6.4.2.1 Kết luận tổng quát:**

- LSTM vẫn là một công cụ mạnh mẽ cho các bài toán chuỗi thời gian như dự đoán giá cổ phiếu, đặc biệt khi dữ liệu có tính phụ thuộc theo thời gian rõ rệt. Tuy nhiên, trong thời đại mô hình Attention và Transformer đang phát triển mạnh mẽ, LSTM nên được xem là một bước đệm quan trọng – hữu ích trong nhiều trường hợp, nhưng cần được mở rộng hoặc thay thế khi yêu cầu tính hiệu quả, tốc độ và khả năng xử lý chuỗi dài cao hơn.

**Tham khảo**

<https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/>

<https://nttuan8.com/bai-14-long-short-term-memory-lstm/>

<https://viblo.asia/s/su-dung-mang-lstm-long-short-term-memory-de-du-doan-co-phieu-24lJDz06KPM>

<https://finance.yahoo.com/quote/AAPL/history/?period1=1325376000&period2=1578614400&interval=1d&filter=history&frequency=1d&guce_referrer=aHR0cHM6Ly92aWJsby5hc2lhLw&guce_referrer_sig=AQAAAJZGBqV59AkCTmWl48bXn_lJ6AkmZT4xU_FuLvev6aFoq7FqegOoTa9ESA0hA50AmjOGDq-WnFWMyV86au20p8TcpMEyuroxM08NwlK9LZzSB7vazZWFqPbUawnDfGLZtsH90Npccvz6L30uJUbbUhV1YG93t9Zbwkw-x-Oi9D7N> **( lấy dữ liệu chứng khoán )**