**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**KHOA CÔNG NGHỆ THỒNG TIN**



**ĐỒ ÁN CƠ SỞ**

**Môn: Cơ Sở Trí Tuệ Nhân Tạo**

**ĐỀ TÀI: DỰ ĐOÁN GIÁ CỔ PHIẾU BĂNG   
MÔ HÌNH CHUỖI THỜI GIAN**

**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.

**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

**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')