**Dự đoán giá cổ phiếu sử dụng mô hình thuật toán Transformer và LSTM**

Võ Mạnh Thường, 19119224  
Trường Đại học Sư phạm Kỹ Thuật Thành Phố Hồ Chí Minh

**1. Giới thiệu:**

**1.1. Machine learning ( máy học):**

Máy học là một ngành của trí tuệ nhân tạo và khoa học dữ liệu, nó được sử dụng để xây dựng các mô hình và thuật toán để máy tự học từ dữ liệu và dự đoán kết quả. Máy học được sử dụng rộng rãi trong các lĩnh vực như xử lý ngôn ngữ tự nhiên, nhận dạng đối tượng, dự báo thị trường, và điều khiển tự động.

Có hai loại chính của máy học là học cụ thể và học tự động. Trong học cụ thể, người ta sẽ xây dựng mô hình và chỉ định các quy tắc cho máy học, trong khi học tự động, máy sẽ tự học và tìm ra các quy tắc từ dữ liệu. Học tự động là một trong những ngành chính của máy học, nó bao gồm các thuật toán và mô hình máy học mà không cần phải được chỉ định quy tắc hoặc giám sát. Trong học tự động, máy sẽ học từ dữ liệu và tìm ra các quy tắc tự động. Các thuật toán học tự động phổ biến bao gồm hồi quy tuyến tính, cây quyết định, mô hình phân loại và học sâu. Học tự động có thể được chia thành hai loại chính là học có giám sát và học không giám sát, trong học có giám sát, máy sẽ được cung cấp với cả dữ liệu và nhãn (kết quả muốn dự đoán) để học, trong học không giám sát máy sẽ chỉ được cung cấp với dữ liệu.

**1.2.Máy học trong dự đoán tài chính:**

Máy học là một kỹ thuật trí tuệ nhân tạo đang được sử dụng rộng rãi trong các lĩnh vực, trong đó có tài chính. Trong ngành tài chính, máy học được sử dụng để dự đoán giá cổ phiếu, dự báo tỷ giá, dự báo lãi suất và rủi ro giao dịch.

Máy học có khả năng học và dự đoán dữ liệu một cách chính xác hơn so với các phương pháp truyền thống. Các mô hình máy học được sử dụng trong tài chính bao gồm hồi quy tuyến tính, cây quyết định, mô hình phân loại và học sâu.

Học sâu là một trong những thuật toán máy học mới nhất và hiệu quả nhất để dự đoán tài chính. Nó sử dụng các mô hình mạng neural để học và dự đoán dữ liệu. Học sâu có khả năng tự học và cải thiện khả năng dự đoán của chúng khi được cung cấp thêm dữ liệu.

**2.Xây dựng mô hình:**

**2.1.Tập dữ liệu Dataset:**

Tím kiếu dữ liệu dataset trong mô hình máy học là một trong những bước quan trọng nhất trong quá trình nghiên cứu, phải chọn được dữ liệu đủ dài và hợp lý thì mô hình mới có thể đưa ra kết quả tối ưu và chính xác nhất.   
Trong đề tài này sử dụng tập dữ liệu giá cổ phiều hằng ngày của các công ty Google (GOOG), Amazon (AMZN), Apple (AAPL), Invesco QQQ Trust (QQQ) để nghiên cứu. Các tập dữ liệu mà đề tài sử dụng có thời gian khác nhau GOOG (7 tháng), AMZN (2 năm), AAPL (2 năm), QQQ ( 4 tháng và 13 năm).

Các tập dữ liệu này được lấy từ Yahoo Finance. Yahoo Finance là một trang web cung cấp thông tin tài chính và kinh tế miễn phí cho người dùng. Trang web cung cấp các tính năng như xem giá cổ phiếu, biểu đồ giá, thông tin tài chính và kinh tế, tài khoản chứng khoán, và các bài viết về tài chính và kinh tế. Yahoo Finance cũng cung cấp các dịch vụ như theo dõi giá cổ phiếu trong thời gian thực, theo dõi tài khoản chứng khoán, và tạo các bảng tài chính. Trang web cũng cung cấp các chuyên mục chuyên sâu về các chủ đề như tài chính cá nhân, chứng khoán, tiền gửi và sổ tiết kiệm, và các chủ đề kinh tế quốc tế.  
Tập dữ liệu trong đề tài này này sẽ được chia thành 2 tập con: train data và test data. Tập train data sẽ được sử dụng để huấn luyện cho mô hình và tập test data được sử dụng để dự đoán và đánh giá độ chính xác của mô hình.**2.2.Data analysis:**

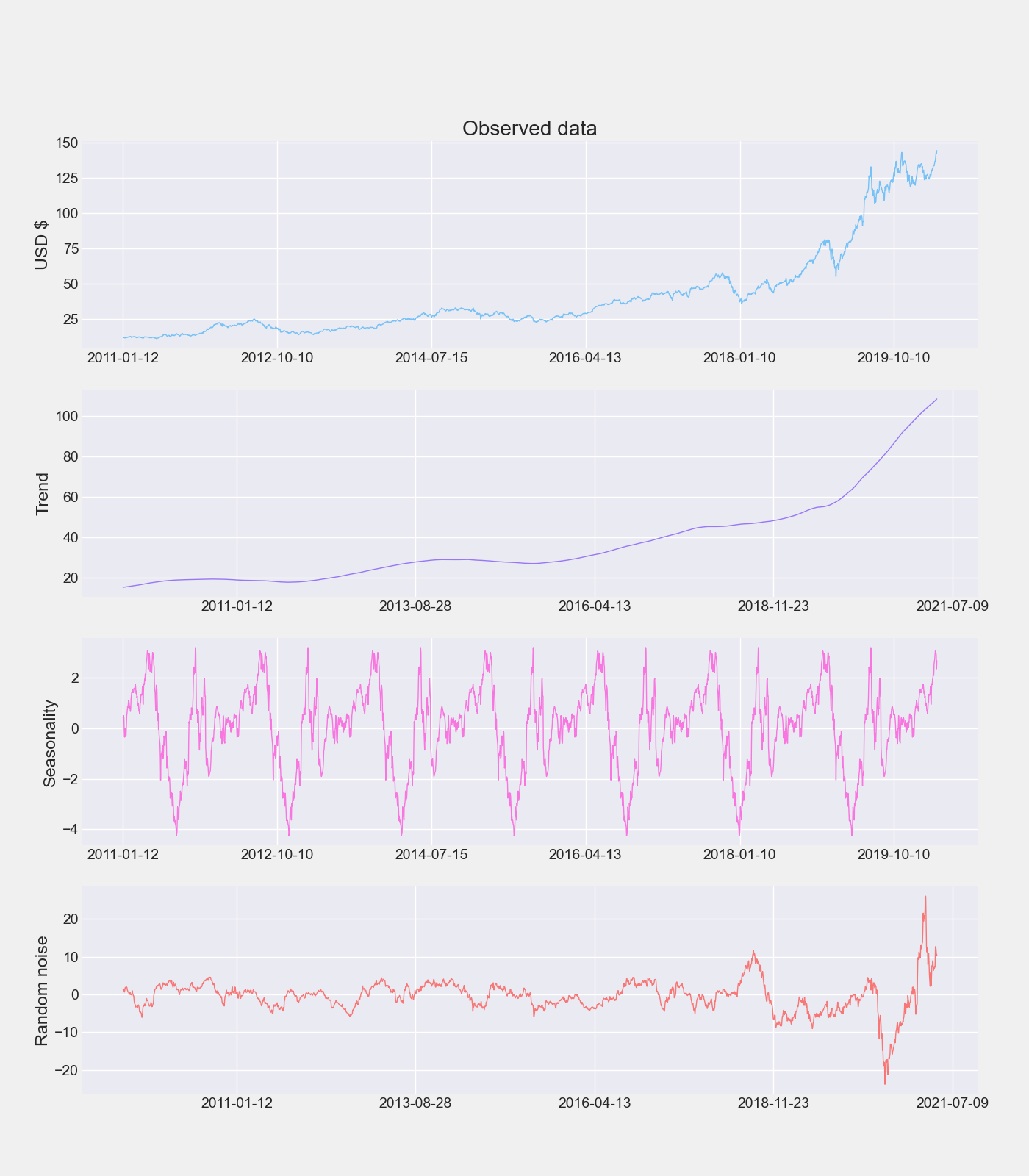
Một số câu hỏi quan tâm trong phân tích dữ liệu thời gian là:

- Dòng xu hướng tổng thể của dữ liệu là gì?

- Có có sự lặp lại về mùa hay không?

- Mức độ nhiễu của dữ liệu là bao nhiêu?

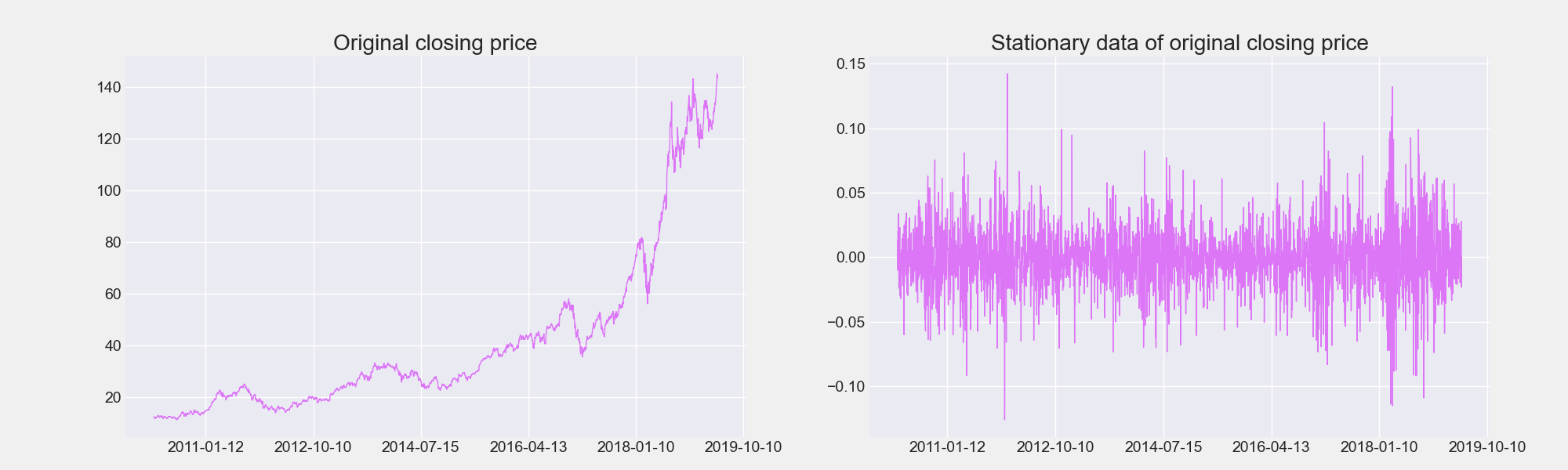
Trong Analysis.py, đề tài sử dụng API của statsmodels để phân tích các câu hỏi này, kết quả của việc này cho giá trị cổ phiếu được thể hiện qua hình sau:



Như đã thấy, có một số hành vi mùa rõ ràng trong dữ liệu. Điều này là không mong muốn và dữ liệu đã được chuyển thành stationary. Điều này được thực hiện bởi thuật toán:



Trong đó i là phạm vi khác biệt ( differencing range), được đặt là 12. Để kiểm tra dữ liệu có stationary sau khi chuyển đổi hay không, Phương pháp kiểm tra Augmented Dickey-Fuller được thực hiện. Kết quả của việc chuyển đổi được thể hiện như sau:



Trong đó, có thể thấy rang dữ liệu cả giá trị trung bình và phương sai stationary.

Sau đó việc chuyển đổi ngược (inverse transformation) để lấy lại giá trị ban đầu:



**2.3.Chỉ số indicator:**

Một trong những yếu tố giúp bài này dự đoán chính xác hơn giá cổ phiếu nhờ vào các chỉ số indicator.

Trong dự đoán tài chính, các chỉ số indicator là các giá trị số hóa được sử dụng để đo lường và đánh giá tình hình kinh tế và tình hình tài chính của một công ty hoặc thị trường. Chúng bao gồm các chỉ số như tỷ suất lãi suất, tỷ suất lợi nhuận, tỷ lệ nợ, và tỷ lệ cổ tức. Các chỉ số này được sử dụng để đánh giá tình hình tài chính của một công ty và để dự đoán tương lai của công ty hoặc thị trường.

Đề tài này sử dụng 8 indicator bao gồm: daily return, Roc indicator, Williams, money flow index, ulcer index, average true range, simple moving average, exponential.

Các chỉ số này đều được đưa vào 2 mô hình LSTM và transformer để thực hiện dự đoán giá cổ phiếu.

**2.4.Mô hình thuật toán LSTM:**

LSTM (Long Short-Term Memory) là một loại mô hình máy học được sử dụng để xử lý và dự đoán dữ liệu thời gian. Nó là một phiên bản nâng cao của mô hình RNN (Recurrent Neural Network) với khả năng ghi nhớ dài hạn hơn.

LSTM có một bộ nhớ cục bộ được gọi là cell state, nó cho phép mô hình ghi nhớ và quản lý thông tin trong thời gian dài. LSTM còn có các cổng đầu vào, đầu ra và quên, cho phép mô hình chọn thông tin cần thiết để ghi nhớ và quên thông tin không cần thiết.

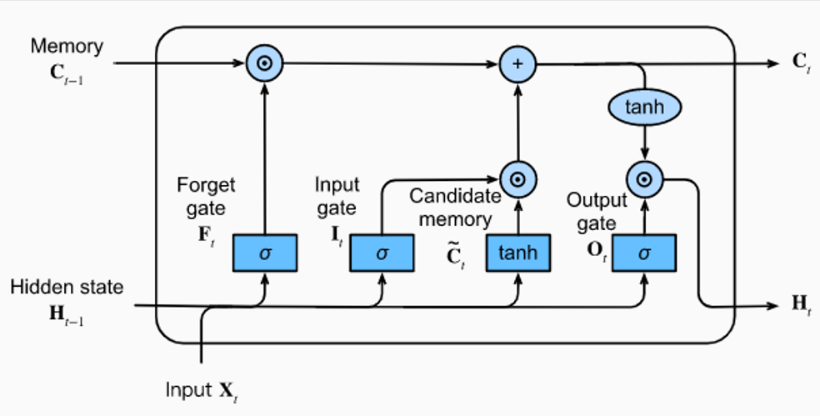
LSTM được sử dụng rộng rãi trong các bài toán xử lý ngôn ngữ tự nhiên, dự đoán thời gian và dự báo tiền tệ. Nó cũng được sử dụng trong các lĩnh vực khác như âm nhạc, video và tài chính.

Nguyên lý hoạt động của mô hình trong bài toán dự đoán tài chính như sau:

LSTM là một loại mạng neural được sử dụng trong bài toán dự đoán tài chính để giải quyết vấn đề "vanishing gradient" trong các mạng neural truyền thống. Vấn đề này xảy ra khi mạng neural truyền thống khó khăn trong việc học các quan hệ giữa các sự kiện xa xưa và hiện tại, khiến cho dự đoán trong tương lai không chính xác. LSTM có thể học các quan hệ thời gian dài và cập nhật trạng thái dữ liệu cho từng thời điểm, giúp cho dự đoán trong tương lai được chính xác hơn. LSTM có một cấu trúc gồm các "cell" và "gate", các gate này sẽ quản lý việc cập nhật và truyền thông thông tin qua các cell. Cụ thể hơn, có 3 loại gate: "input gate", "forget gate" và "output gate".

"Input gate" sẽ quản lý việc cập nhật dữ liệu từ input vào cell. "Forget gate" sẽ quản lý việc xóa bớt các thông tin không cần thiết trong cell. "Output gate" sẽ quản lý việc truyền thông thông tin từ cell ra output.

Cấu trúc của LSTM như sau:



Với cấu trúc này, LSTM có thể học được các quan hệ thời gian dài và cập nhật trạng thái dữ liệu cho từng thời điểm, giúp cho dự đoán trong tương lai được chính xác hơn.

Mô hình LSTM trong đề tài được lấy từ thư viện TORCHRNN với các tham số đầu vào:

class rnn\_params:

    rnn\_type = 'lstm'

    input\_dim = 16

    hidden\_dim = 2048

    num\_layers = 1

    output\_dim = 1

    n\_epochs = 100

    lr = 0.00001

**Trong đó:**

* rnn\_type: Loại RNN sử dụng trong mô hình. Trong trường hợp này, được chỉ định là 'lstm' (kiểu Long Short-Term Memory)
* input\_dim: Kích thước đầu vào cho mô hình RNN
* hidden\_dim: Kích thước của tầng ẩn trong mô hình RNN
* num\_layers: Số tầng RNN trong mô hình
* output\_dim: Kích thước của tầng đầu ra trong mô hình RNN
* n\_epochs: Số lần lặp (epochs) cho việc huấn luyện mô hình
* lr: Tốc độ học (learning rate) cho thuật toán huấn luyện.

**2.5. Mô hình thuật toán transformer:**

Transformer là một kiểu mô hình máy học được sử dụng để xử lý dữ liệu văn bản. Nó được giới thiệu bởi Google năm 2017 trong bài báo "Attention Is All You Need" và đã đạt được sự thành công trong các bài toán xử lý ngôn ngữ tự nhiên.

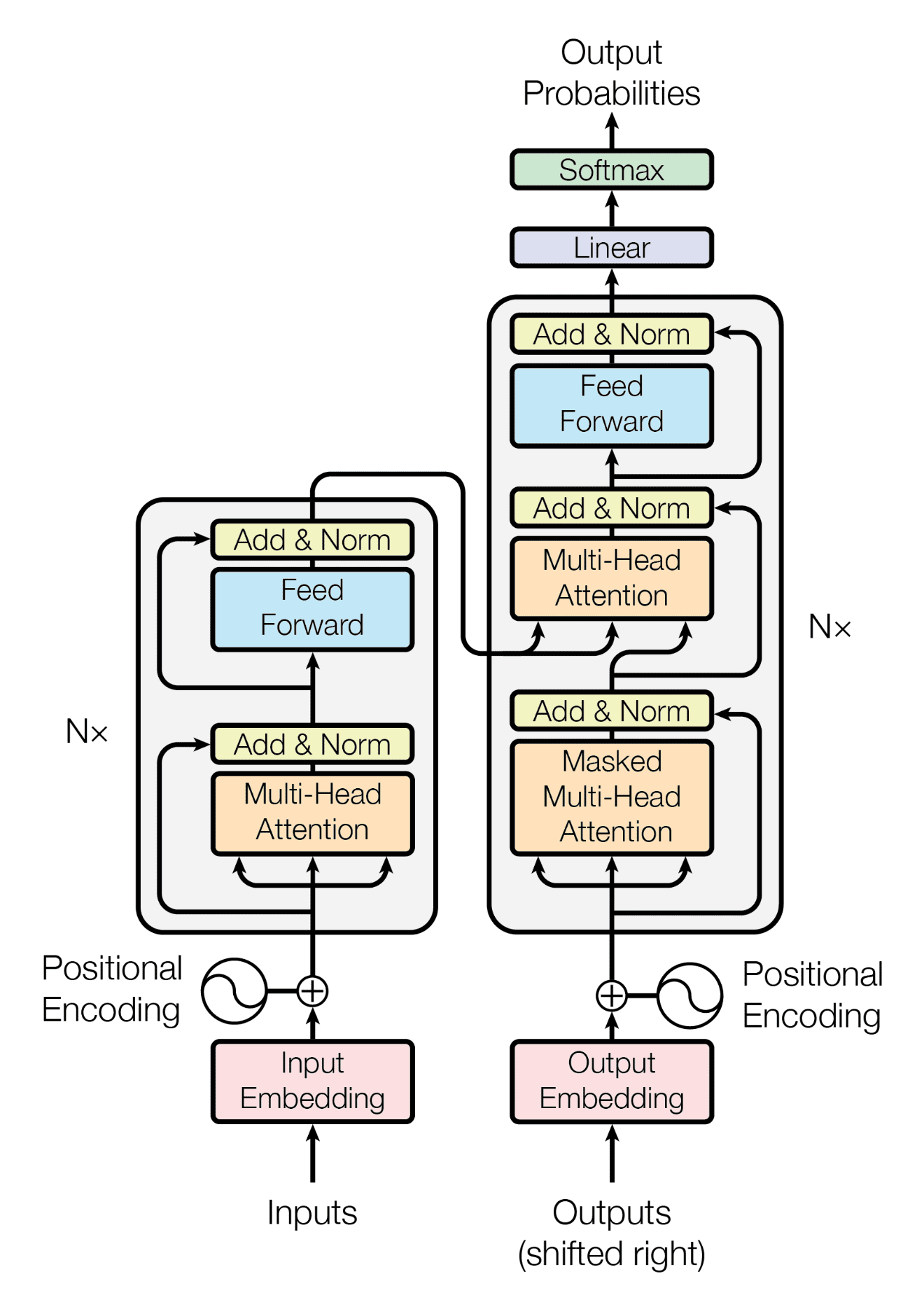
Mô hình sử dụng một kiểu attention mechanism để tạo ra mối quan hệ giữa từng từ trong câu. Nó cho phép mô hình tìm ra các quan hệ quan trọng giữa các từ trong câu và sử dụng chúng để dự đoán từ tiếp theo hoặc dự đoán nghĩa của câu.

Transformer cũng có khả năng xử lý dữ liệu văn bản có độ dài không giới hạn, điều này cho phép nó áp dụng cho các bài toán mà các mô hình truyền thống không thể giải quyết được.

Mô hình đã được áp dụng rộng rãi trong các bài toán xử lý ngôn ngữ tự nhiên, chuyển ngôn ngữ, truy vấn tìm kiếm, và các bài toán khác liên quan đến xử lý văn bản.

Transformer là một mô hình mạng neural được sử dụng trong bài toán dự đoán tài chính để giải quyết vấn đề khó khăn trong việc học các quan hệ giữa các sự kiện trong dữ liệu thời gian. Trong Transformer, các input được xử lý theo cách khác so với các mô hình truyền thống như LSTM, nó sử dụng một mạng neural attention để tìm ra các quan hệ giữa các input.

Cấu trúc của mô hình transformer sẽ gồm một lớp mã hóa và giải mã:



Trong Transformer, mỗi input được biểu diễn dưới dạng một vector, và các vector này được truyền qua một mạng neural attention để tìm ra các quan hệ giữa các input. Các quan hệ này được sử dụng để tính toán một vector mới cho mỗi input, và vector mới này sẽ được sử dụng để dự đoán kết quả.

Với mạng neural attention, Transformer có thể học được các quan hệ giữa các input mà không cần phải theo thứ tự của chúng, giúp cho dự đoán trong tương lai được chính xác hơn. Điều này cực kỳ hữu ích trong các bài toán dự đoán tài chính, vì dữ liệu tài chính thường có nhiều quan hệ phức tạp giữa các sự kiện.

Trong đề tài, mô hình transformer được khởi tạo với các tham số sau:

class transf\_params:

    n\_layers = 11

    num\_heads = 12

    model\_dim = 16  # nr of features

    forward\_dim = 2048

    output\_dim = 1

    dropout = 0

    n\_epochs = 100

    lr = 0.01

Trong đó:

* n\_layers: số lớp trong mô hình transformer.
* num\_heads: số lượng heads trong mỗi lớp multi-head attention.
* model\_dim: số chiều của mỗi vector đầu vào và đầu ra trong mô hình.
* forward\_dim: số chiều của fully connected layer trong mô hình.
* output\_dim: số chiều của vector đầu ra cuối cùng.
* dropout: tỷ lệ dropout được sử dụng trong mô hình.
* n\_epochs: số lần lặp qua tập dữ liệu trong quá trình huấn luyện.
* lr: tốc độ học (learning rate) được sử dụng trong quá trình huấn luyện.

**2.6. Trainning:**

Trong quá trình trainning, các tham số cơ bản được khởi tạo như sau:

Số lượng epoch: 100.

Loss: Mean square error ( MSE)

Learning rate: 0.01 đối với mô hình transformer, 0.00001 đối với mô hình lstm.

**3. Lưu đồ thuật toán:**

Cách hoạt động của mô hình dự đoán tài chính sử dụng mô hình LSTM hoặc transformer được thể hiện ở lưu đồ sau:



Đầu tiên, ta lấy dataset từ yahoo finance về tập dữ liệu mà muốn mô hình học. Sau đó thực hiện tính toán 8 chỉ số indicator bao gồm: daily return, Roc indicator, Williams, money flow index, ulcer index, average true range, simple moving average, exponential và gộp vào tập dataset.

Tiến hành phân đoạn dữ liệu thành 30 ngày để dự đoán ngày thứ 31 và chia tập dữ liệu theo tỉ lệ 8:2, trong đó 80% dữ liệu sẽ được dùng để trainning và 20% dữ liệu còn lại để dự đoán.

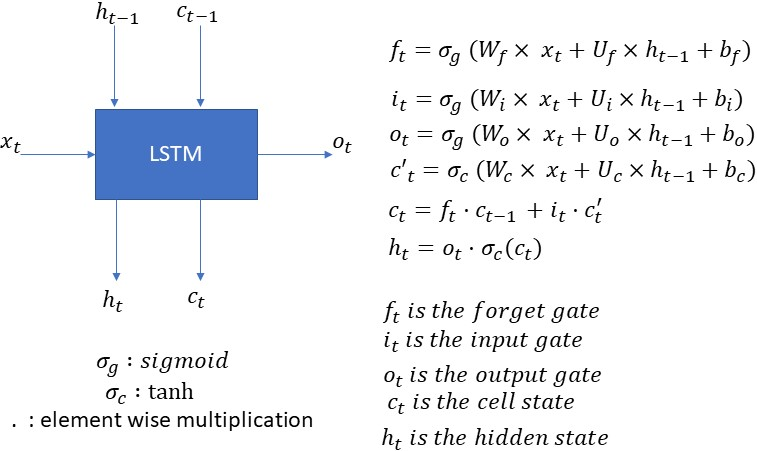
Sau khi chia dữ liệu ta tiến hành trainning trước cho 1 trong 2 mô hình LSTM hoặc Transformer. Sau khi trainning ta sẽ thu được chỉ số MSE tối ưu nhất ( thấp nhất).

Để dự đoán và đánh giá độ chính xác của mô hình ta sử dụng 20% dữ liệu còn lại để tính toán. Sau khi train xong ta lấy mô hình đã train đó tiếp tục với tập dữ liệu còn lại. Thực hành dự đoán dựa trên dataset phân đoạn dữ liệu 30 ngày gần nhất để dự đoán ngày thứ 31 và cứ như thế tiếp tục cho đến hết. Sau khi thu được kết quả dự đoán ta so sánh giá predict với giá target và tính các chỉ số RMSE, MAE, MAPE để đánh giá mô hình.

Trong mô hình LSTM quá trình trainning diễn ra như sau:



Đầu tiên mô hình đọc dữ liệu data đã qua chuẩn hóa và tiền xử lý, khởi tạo các tham số ban đầu như đã nói ở mục 2.4. Tiến hành tính forward propagation, forward propagation là quá trình xử lý dữ liệu từ input đến output. Nó bao gồm các bước như tính toán các giá trị cho các điểm thời gian trong mạng LSTM, tính toán các giá trị output và tính toán các giá trị mất mát cho mạng để huấn luyện. Bắt đầu quá trình trainning từ epoch 0 đến 100, trong mỗi bước trainning mô hình sẽ tự học được các tham số p,q,w. Trong đó p, q, v là các tham số được sử dụng để điều chỉnh hoạt động của từng bước thời gian. p là tham số forget gate, q là tham số input gate, v là tham số output gate. Chúng được sử dụng để điều chỉnh việc quên hoặc ghi nhớ thông tin trong từng bước thời gian, quản lý những thông tin mới được nhập vào và xác định những thông tin được truyền tiếp đến bước thời gian tiếp theo.

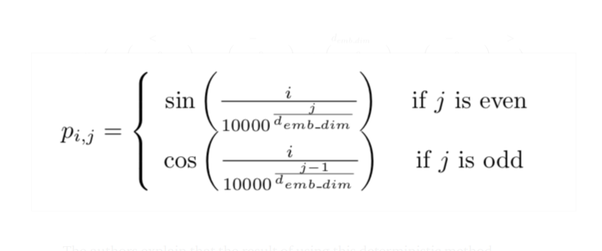
  
Sau khi tính toán xong nó sẽ thực hiện cập nhật lại trọng số weight (w), là một tham số được sử dụng trong hầu hết các mô hình machine learning, đặc biệt là trong các mô hình neural network. Weight là một số thực mà được gán cho mỗi kết nối giữa các neuron trong mạng. Nó định nghĩa mức độ quan trọng của mỗi kết nối trong việc truyền tín hiệu từ neuron nguồn đến neuron đích. Trong quá trình huấn luyện, weight được cập nhật để tối ưu hóa độ chính xác của mô hình. Weight trong mô hình này được tính dựa trên thuật toán Adam. Weight được sử dụng để điều chỉnh hoạt động của các tham số q, p, v trong quá trình huấn luyện mô hình, xác định mức độ quan trọng của các thông tin được nhập vào trong quá trình huấn luyện mô hình. Sau đó nó trải qua thêm một bước forward nữa và cập nhật lại chỉ số MSE. Sau khi thực hiện xong 100 epochs ta sẽ thu được chỉ số MSE và weight tối ưu nhất và kết thúc quá trình trainning. Mô hình sau khi trainning xong sẽ được đưa vào để dự đoán giá tiếp theo.

Trong mô hình Transformer quá trình trainning diễn ra như sau:



Quá trình huấn luyện của mô hình của Transformer giống với LSTM nhưng nó sẽ khác ở các thành phần như feed forward và nó không tính các tham số q,p,v như mô hình LSTM mà thay vào đó là tính toán các tham số queries (Q), Keys (K), value (V). Chi tiết như sau:

Đầu tiên, mô hình LSTM huấn luyện bằng cách đưa các dữ liệu vào tuần tự nên sẽ không cần quan tâm vị trí của nó trong tập dữ liệu là bao nhiêu vì nó sẽ đánh dấu vị trí của input đó sau mỗi lần đưa vào giá trị tiếp theo. Nhưng ngược lại, ở mô hình transformer, nó đưa vào song song đồng thời tất cả các dữ liệu input nên chúng ta cần phải có 1 lớp mã hóa vị trí để xác định được vị trí của nó ở trong tập input. Lúc này lớp positional encoding được thêm vào để nhúng thêm một vector vị trí vào vector của tập dữ liệu. Công thức của lớp positional encoding như sau:



Điểm mạnh trong mô hình transformer là lớp multi-head attention. Multi-head attention bao gồm nhiều lớp attention mà model có thể tập trung vào nhiều khía cạnh của dữ liệu cùng lúc. Thông thường, một lớp attention chỉ tập trung vào một khía cạnh của dữ liệu (ví dụ như một từ trong câu) nhưng multi-head attention cho phép model tập trung vào nhiều khía cạnh khác nhau của dữ liệu cùng lúc. Điều này giúp model học được nhiều thông tin hơn và có thể tốt hơn trong việc xử lý dữ liệu phức tạp.

Trong lớp attention, mô hình sẽ học mối quan hệ giữa giá cổ phiếu của ngày này so với các ngày còn lại trong tập dữ liệu cộng thêm quan hệ tăng hay giảm của giá cổ phiếu với các chỉ số indicators nhờ vào lớp attention. lớp attention tính toán các chỉ số q, k, v bằng cách sử dụng ma trận trọng số Wq, Wk, Wv để tính toán tích vô hướng của các vector đầu vào với các ma trận trọng số đó. Cụ thể, q, k, v được tính bằng:

q = Wq \* input

k = Wk \* input

v = Wv \* input

Trong đó, input là vector đầu vào của mô hình, Wq, Wk, Wv là các ma trận trọng số khác nhau.

Sau khi tính toán các chỉ số trên nó sẽ sử dụng một phép tính dot product (tích vô hướng) giữa các vector q và k để xác định mức độ quan hệ giữa các vector đầu vào. Phép tính này được sử dụng để tính toán một ma trận attention weights, mỗi phần tử của ma trận này là một giá trị đại diện cho mức độ quan hệ giữa vector q và vector k tương ứng.

Sau đó, lớp attention sử dụng ma trận attention weights đó để tính toán một ma trận attention output bằng cách thực hiện phép nhân ma trận với ma trận v. Kết quả cuối cùng là một vector mới được tạo ra bởi các giá trị quan hệ giữa các vector đầu vào.. Và trong quá trình huấn luyện, mô hình sẽ học các ma trận Wk, Wq, Wv và ma trận weight sao cho tối ưu nhất. Sau khi hoàn tất quá trình huấn luyện, ta sẽ thu được các vector input mới đã được nhúng giá trị mức độ quan hệ với input khác.

Ở phần feed forward, nó được sử dụng để tính toán các biểu diễn đặc trưng của 1 giá trị stock. Nó bao gồm một lớp fully connected (hoặc có thể là một lớp có kích thước khác) và một hàm kích hoạt, đề tài này sử dụng ReLU. Kết quả của feedforward được sử dụng để tăng cường hoặc giảm sự quan tâm của mô hình giữa các giá stock trong dataset. Feedforward của mô hình Transformer khác với forward của mô hình LSTM. Trong LSTM, forward được thực hiện bởi các tính toán trên các gate (input, forget, output) và các tính toán hồi quy (recurrent) để tính toán các trạng thái hidden và cell. Trong khi đó, trong mô hình Transformer, feedforward chỉ là một lớp fully connected và một hàm kích hoạt được sử dụng để tính toán các biểu diễn đặc trưng của giá stock trong dataset mà không có tính toán hồi quy.

Sau khi chạy xong 100 epoch, ta sẽ thu được weight và MSE tối ưu nhất và có thể dùng nó để dự đoán giá cổ phiếu ở tập test data. Để dự đoán cổ phiếu tiếp theo, nó sẽ đưa qua một lớp linear và softmax để đưa về một ma trận 1 hàng có độ dài vocab\_size. Lúc này khi đưa vào một input, nó sẽ tìm đến vector output vừa tính được có mức độ quan hệ cao nhất và dự đoán.

**3.Kết quả:**

**3.1.Các phương pháp đánh giá:**

**3.1.1. RMSE:**

RMSE (Root Mean Squared Error) là một chỉ số đo lường sai số giữa dự đoán và giá trị thực tế. Nó được tính bằng cách lấy trung bình bình phương của các sai số giữa dự đoán và giá trị thực tế. RMSE được sử dụng rộng rãi trong các bài toán dự đoán và học máy, vì nó cho phép đo lường sai số trung bình và có thể dễ dàng so sánh với các mô hình khác. Nó thường được sử dụng để đo lường chính xác của mô hình dự đoán.

**3.1.2. MAE:**

MAE (Mean Absolute Error) là một chỉ số đo lường sai số giữa dự đoán và giá trị thực tế. Nó được tính bằng cách lấy trung bình của tổng các giá trị tuyệt đối của sai số giữa dự đoán và giá trị thực tế. MAE là một chỉ số đo lường sai số trung bình và thường được sử dụng để đánh giá chính xác của mô hình dự đoán trong các bài toán học máy và dự đoán. Nó có điểm mạnh là có thể chịu được các giá trị ngoại lệ (outliers) mà không bị ảnh hưởng quá nhiều.

**3.1.3. MAPE:**

MAPE (Mean Absolute Percentage Error) là một chỉ số đo lường sai số giữa dự đoán và giá trị thực tế. Nó được tính bằng cách lấy trung bình của tỷ lệ sai số tuyệt đối giữa dự đoán và giá trị thực tế. MAPE được sử dụng rất nhiều trong các bài toán dự đoán và học máy, đặc biệt là trong các bài toán dự đoán tài chính và kinh tế. Nó cho phép đo lường sai số tuyệt đối trong tỷ lệ % và có thể dễ dàng so sánh với các mô hình khác. Nhưng cần lưu ý rằng MAPE không thích hợp khi có giá trị thực tế bằng 0.

* + 1. **Memory consumed:**

Đề tài cũng thực hiện tính toán bộ nhớ tiêu hao của GPU trong quá trình training.

* 1. **Dataset:**

Các thông số trong hàm shape of train data và shape of test có ý nghĩa như sau:

+ x\_train: dữ liệu để train

+ y\_train: dữ liệu kiểm thử x\_train

+ x\_test: dữ liệu để test trong quá trình training

+ y\_test: dữ liệu kiểm thử test

Các thông số của shape sẽ là (x,y,z) của tập x và (a,b) của tập y trong đó:

* (x,y,z) là kích thước của tập x, là một ma trận 3 chiều với với số hàng là x, số cột là y, và số chiều là z.
* (a,b) là kích thước của dữ liệu y, chúng là một ma trận 2 chiều với số hàng là a, số cột là b.
* y là số lượng ngõ vào và z là số lượng ngõ ra.

Kết quả của các tập như sau:

* **AMZN:**

Shape of train data: (x, y) = ((2081, 30, 16), (2081, 1))

Shape of test data: (x, y) = ((527, 30, 16), (528, 1))

* **GOOG:**

Shape of train data: (x, y) = ((574, 30, 16), (574, 1))

Shape of test data: (x, y) = ((150, 30, 16), (151, 1))

* **AAPL:**

Shape of train data: (x, y) = ((2081, 30, 16), (2081, 1))

Shape of test data: (x, y) = ((527, 30, 16), (528, 1))

* **QQQ ( 4 tháng):**

Shape of train data: (x, y) = ((71, 30, 16), (71, 1))

Shape of test data: (x, y) = ((25, 30, 16), (26, 1))

* **QQQ ( 13 năm ):**

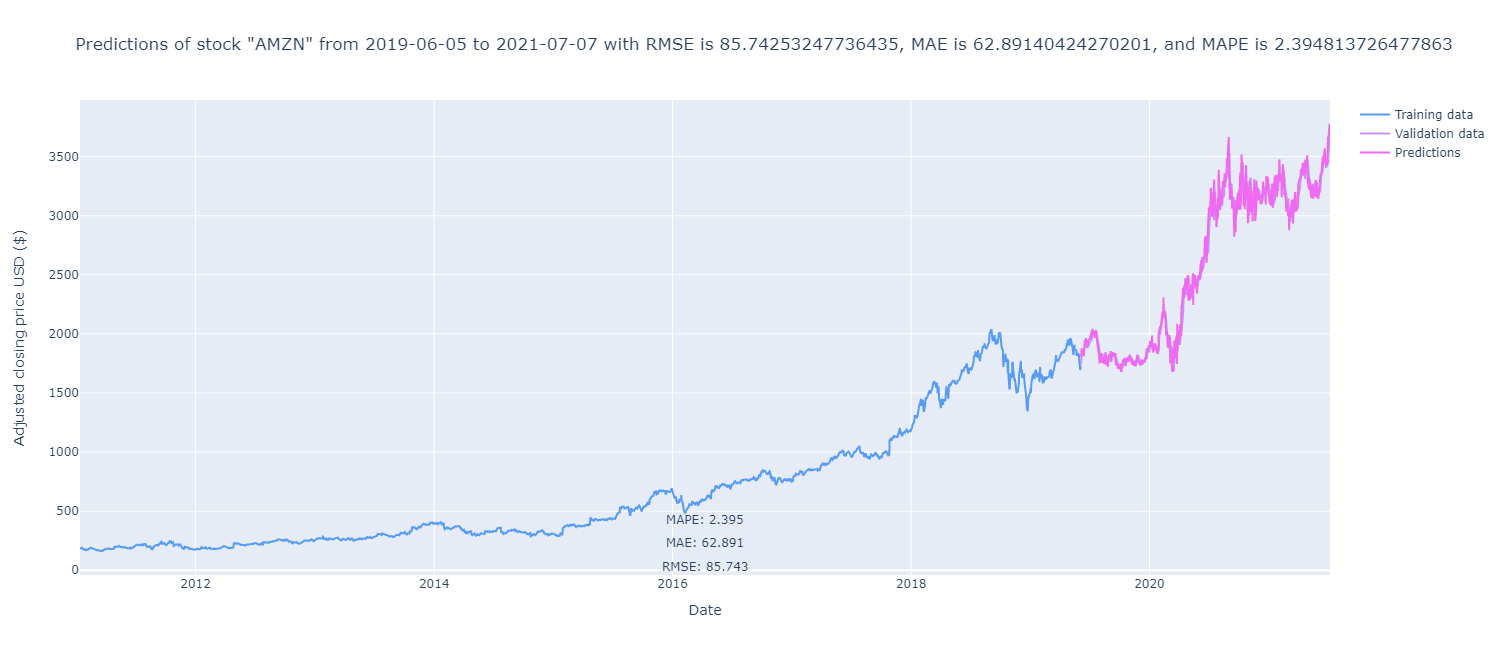
Shape of train data: (x, y) = ((2487, 30, 16), (2487, 1))

Shape of test data: (x, y) = ((629, 30, 16), (630, 1))

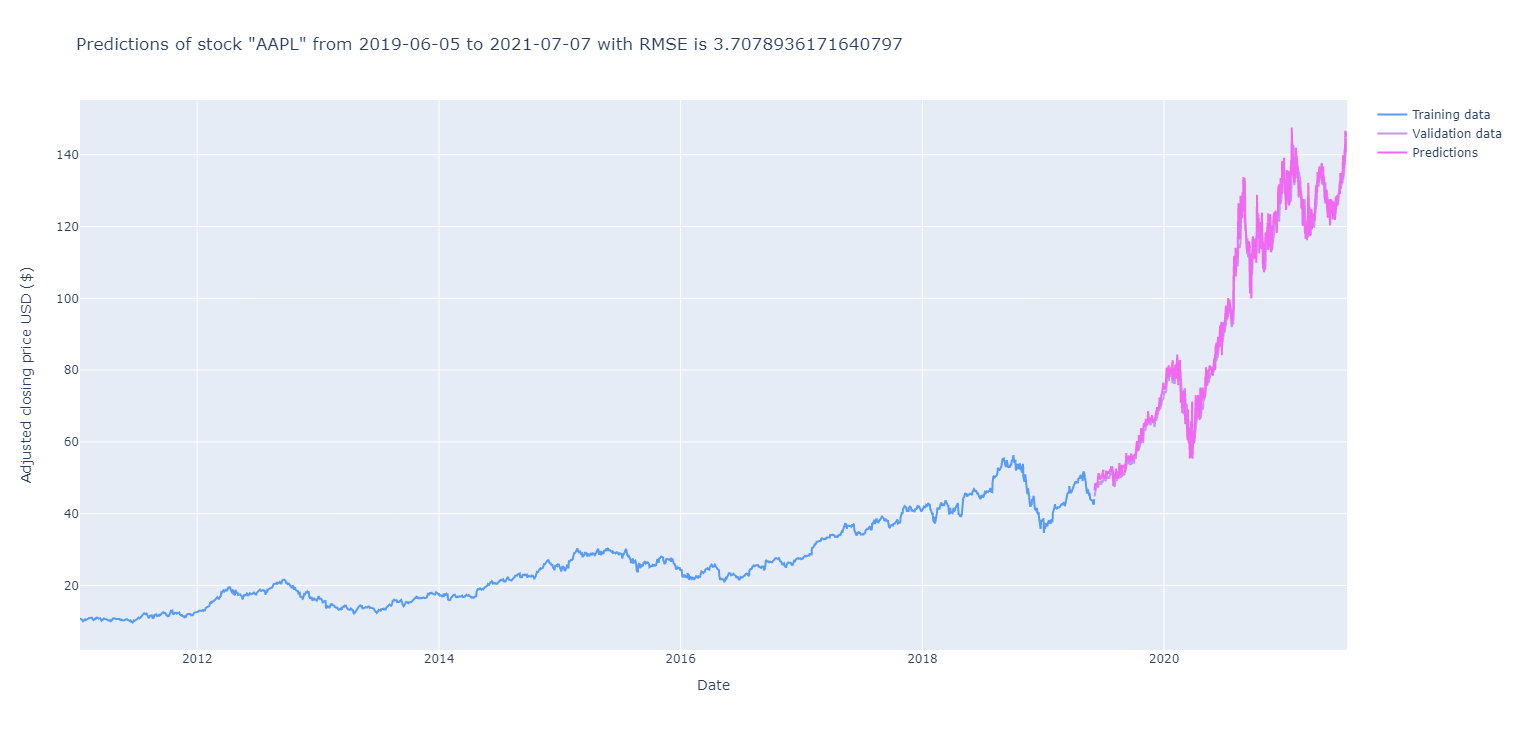
* 1. **LSTM:**

Kết quả dự đoán mô hình của LSTM trong 3 tập dữ liệu như sau:

1. **AMZN:**



1. **AAPL:**

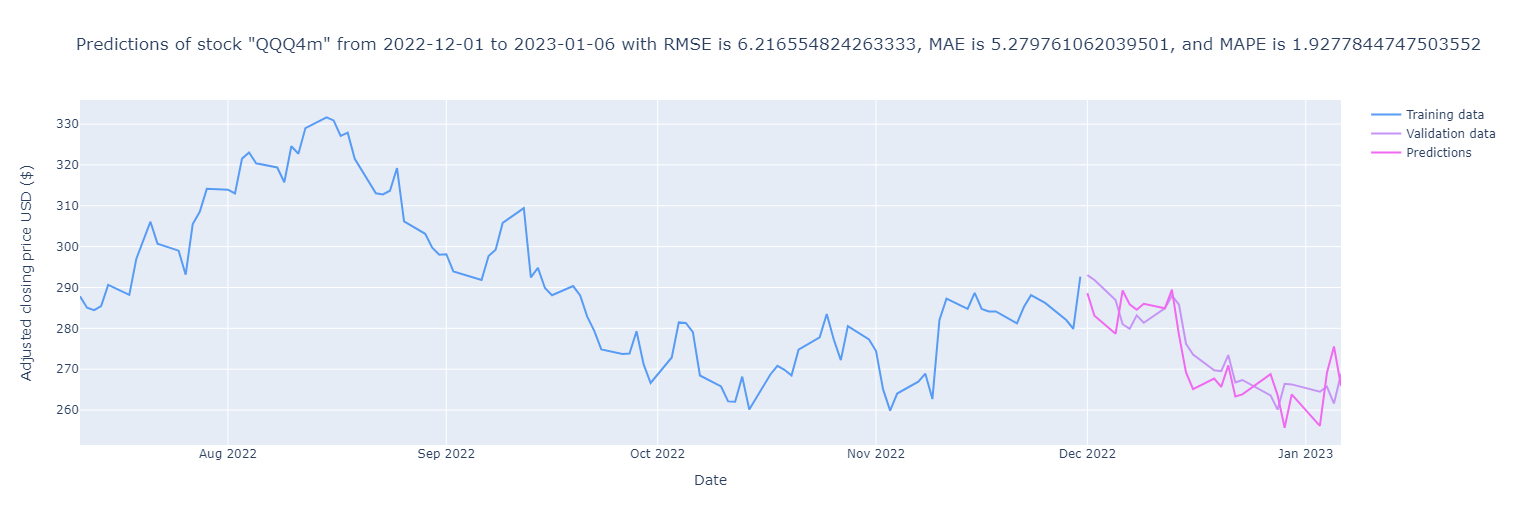


1. **GOOG:**

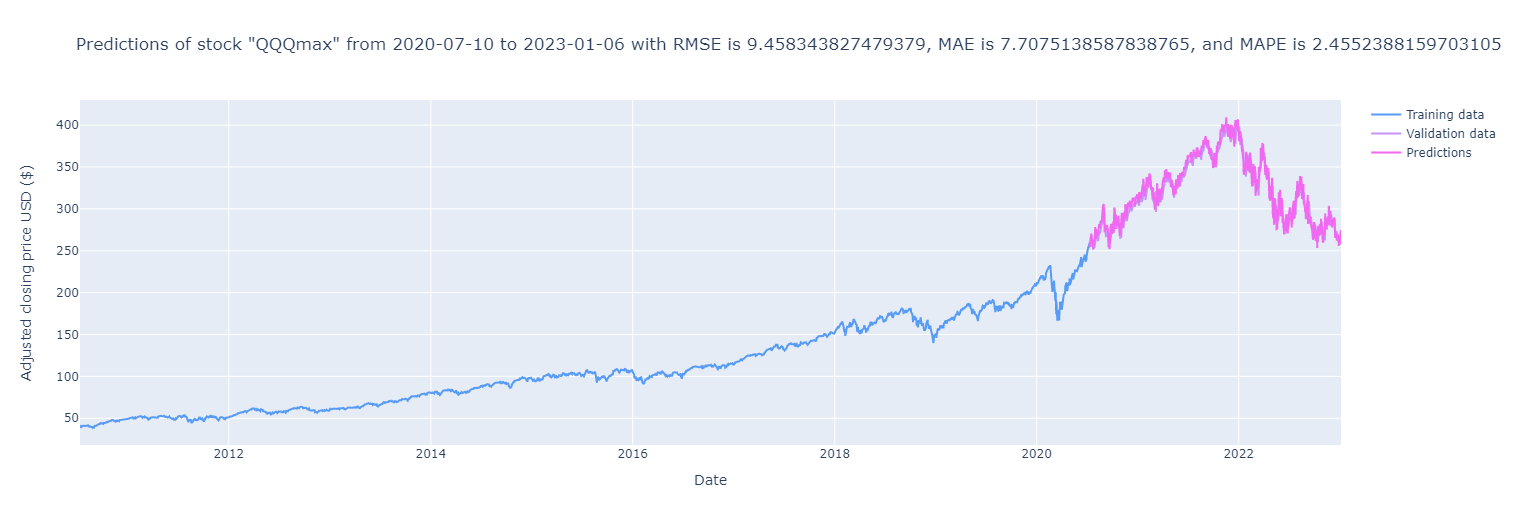


1. **QQQ:**

* Tập dữ liệu 4 tháng:



* Tập dữ liệu 13 năm:



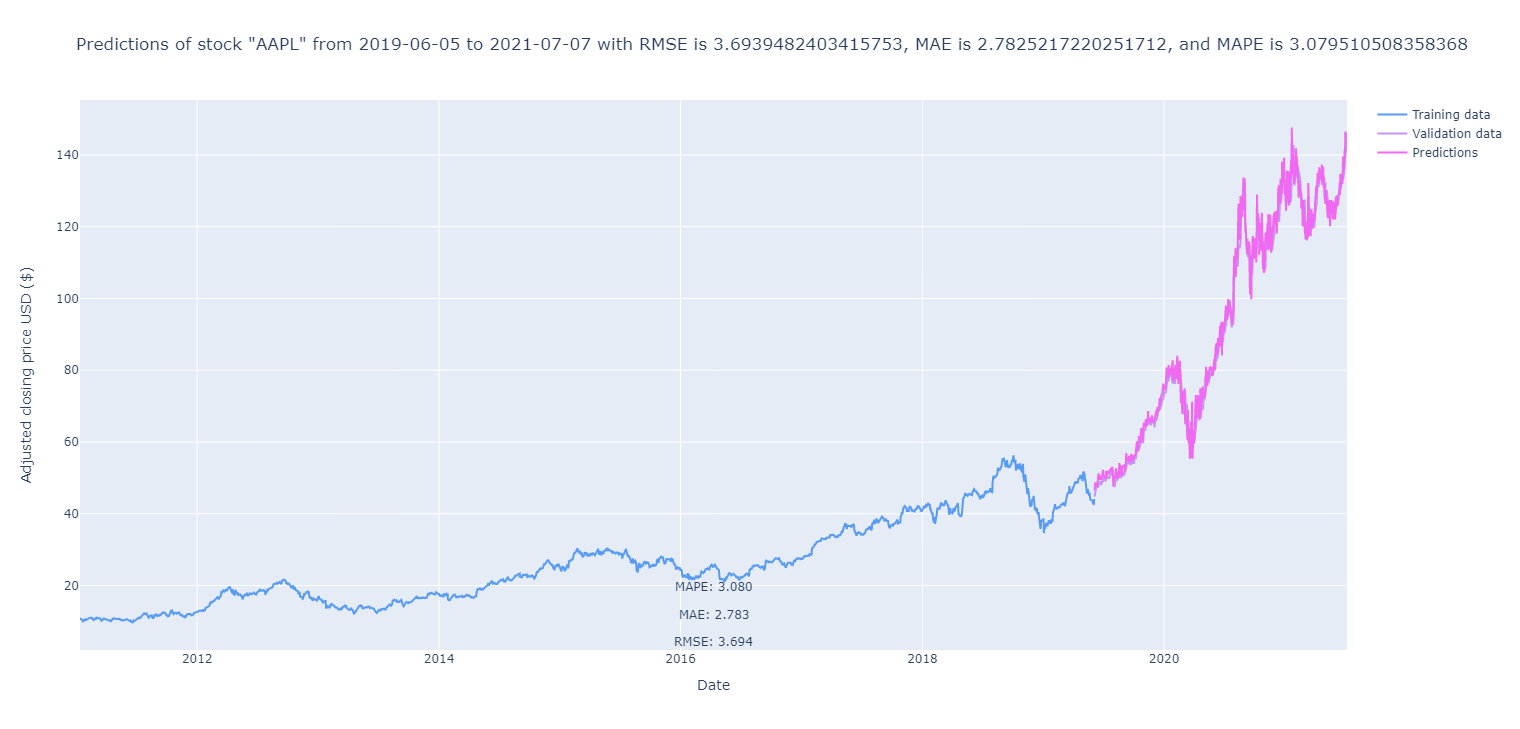
* 1. **Transformer:**

Kết quả của mô hình transformer trong 3 tập dữ liệu như sau:

1. **AMZN:**



1. **AAPL:**



1. **GOOG:**

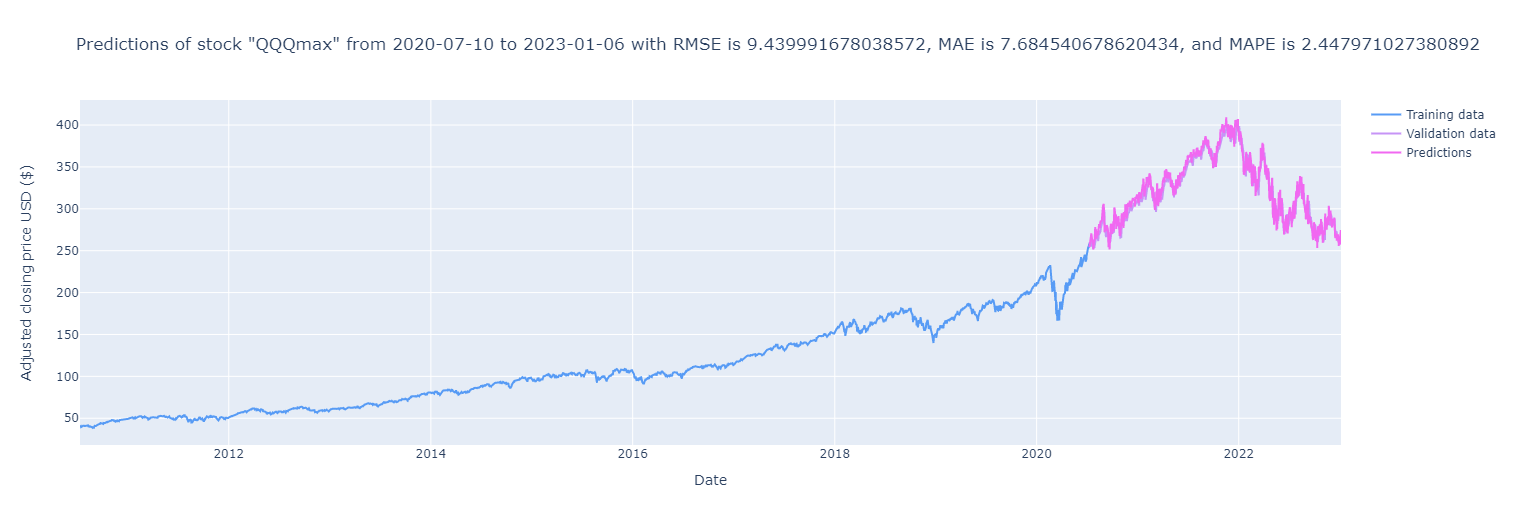


1. **QQQ:**

* Tập dữ liệu 4 tháng:



* Tập dữ liệu 13 năm:



1. **So sánh giữa 2 mô hình:**
2. **Đánh giá mô hình:**

Sau khi training và predict của 2 mô hình LSTM và transformer ta được bảng kết quả như sau:

|  |  |  |
| --- | --- | --- |
| STOCK:GOOG | LSTM | Transformer |
| RMSE | 85.7 | 86.4 |
| MAE | 62.9 | 63.3 |
| MAPE | 2.4 | 2.4 |
| Trainning time (s) | 46s | 12 |
| Inference time (s) | 0.05 | 0.03 |
| Memory GPU consumed (GB) | 3.1 | 2.9 |
| STOCK:AMZN |  |  |
| RMSE | 60.3 | 59.6 |
| MAE | 45.95 | 45.57 |
| MAPE | 2.176 | 2.16 |
| Trainning time (s) | 187 | 59 |
| Inference time (s) | 0.14 | 0.0453 |
| Memory GPU consumed (GB) | 8.1 | 8.65 |
| STOCK:AAPL |  |  |
| RMSE | 3.707 | 3.69 |
| MAE | 2.806 | 2.78 |
| MAPE | 3.112 | 3.08 |
| Trainning time (s) | 187 | 58 |
| Inference time (s) | 0.14 | 0.0453 |
| Memory GPU consumed (GB) | 8.1 | 8.65 |

Dựa vào bảng so sánh trên ta thấy được, đối với các tập có dữ liệu ngắn hơn thì sự khác biệt giữa LSTM và transformer chênh lệch không nhiều, thậm chí với các tập dữ liệu có thời lượng ngắn hơn tầm 4 tháng thì LSTM có kết quả tốt hơn so với transformer. Lí do là vì LSTM có ít tham số hơn so với transformer cộng thêm việc transformer phải sử dụng các lớp self-attention để xử lý và tính toán nên sẽ cần thời gian nhiều hơn vì vậy trong trường hợp ngắn hạn thì các trình tự dữ liệu có thể được xử lý đủ bằng LSTM chứ không cần đến các layer self-attention của transformer.  
Tuy nhiên ngược lại, ở các tập dữ liệu lớn hơn, transformer thể hiện rõ ưu điểm về mặt thời gian training và cả inference time do điểm mạnh hơn của transformer so với LSTM là khả năng xử lý song song thay vì tuần tự như LSTM.

Về độ chính xác thì hầu như không có mô hình nào thể hiện rõ mạnh hơn mô hình còn lại. Do đó, qua các trường hợp trên ta có thể thấy được transformer chỉ mạnh hơn LSTM về khoảng thời gian training và inference time.

Về phần tiêu hao bộ nhớ, LSTM thường tiêu tốn nhiều bộ nhớ GPU hơn Transformer trong các tập dữ liệu ngắn hạn, vì LSTM có thể cần sử dụng nhiều tầng và nhiều tham số hơn Transformer để xử lý trình tự. Trong khi đó, Transformer tiêu tốn nhiều bộ nhớ GPU hơn LSTM trong các tập dữ liệu dài hơn, do Transformer sử dụng các self-attention layers cần tính toán nhiều tham số và có thể cần sử dụng nhiều tầng hơn so với LSTM để xử lý quan hệ giữa các input.

Để thấy rõ sự khác biệt của 2 mô hình dựa trên tập dữ liệu ngắn và dài, đề tài đã thực hiện chạy 2 mô hình trên 2 tập dữ liệu QQQ có thời gian là 4 tháng và 12 năm, kết quả như sau:

|  |  |  |
| --- | --- | --- |
| STOCK: | QQQ | 4 tháng |
|  | LSTM | Transformer |
| RMSE | 6.22 | 6.38 |
| MAPE | 1.93 | 1.97 |
| MAE | 5.28 | 5.4 |
| Trainning time (s) | 8s | 9s |
| Inference time (s) | 0.0155 | 0.0293 |
| Memory GPU consumed (GB) | 1.66 | 1.07 |
| STOCK: | QQQ | 13 năm |
| RMSE | 9.46 | 9.44 |
| MAPE | 2.46 | 2.45 |
| MAE | 7.7 | 7.69 |
| Trainning time (s) | 220s | 70s |
| Inference time (s) | 0.17 | 0.05 |
| Memory GPU consumed (GB) | 9.45 | 10.19 |

Rõ ràng ở 2 tập dữ liệu cùng 1 stock có thời lượng khác nhau, LSTM tốt hơn so với transformer ở tập dữ liệu QQQ 4 tháng, nhưng ở tập có thời lượng rất lớn 13 năm, transformer tốt hơn LSTM ở mọi mặt kể cả độ chính xác trong dự đoán và đặc biệt là thời gian trainning nhanh gần gấp 3 lần so với LSTM. Từ đó ta có thể thấy, ở các tập dữ liệu càng dài thì transformer sẽ càng tốt hơn LSTM. Do giới hạn phần cứng nên đề tài chỉ có thể thử nghiệm ở thời gian 13 năm, nhưng khi có phần cứng đủ mạnh ta có thể cho mô hình transformer học ở thời gian rất lớn (có thể sử dụng data từ năm 1962 ) lúc này transformer sẽ vượt trội và bỏ xa LSTM ở mọi chỉ số ( Độ chính xác và thời gian). Và ở các tập dữ liệu cực lớn thì LSTM sẽ gặp vấn đề vanishing gradient problem, gây ra sự không chính xác trong dự đoán nhưng ngược lại thì transformer lại có thể học tốt mà không quan tâm dữ liệu lớn đến bao nhiêu vì nhờ vào cơ chế self-attention, tránh việc bị mất mát gradient.

1. **Ưu điểm của transformer so với LSTM:**

Transformer có một số ưu điểm so với LSTM trong bài toán dự đoán giá cổ phiếu:

* Transformer có khả năng xử lý các tập dữ liệu liên tục nhanh hơn so với LSTM.
* Transformer có khả năng tự động học các quan hệ giữa các input, giúp cho việc dự đoán giá cổ phiếu tốt hơn.
* Transformer có khả năng học độc lập với thời gian, giúp cho việc dự đoán giá cổ phiếu tốt hơn trong trường hợp dữ liệu có nhiễu.
* Transformer tỏ ra ưu thế hơn LSTM khi xử lý các tập dữ liệu lớn. Ngược lại, LSTM vẫn tốt hơn transformer một chút ở các tập dữ liệu ngắn hơn.
* Với khả năng học song song của transformer, tốc độ training và inference nhanh hơn nhiều so với LSTM chỉ xử lý tuần tự.

1. **Kết luận:**

Mô hình LSTM là một trong những mô hình deep learning phổ biến được sử dụng trong bài toán dự đoán giá cổ phiếu. Nó có khả năng xử lý dữ liệu thời gian liên tục và học các quan hệ giữa các bước thời gian. Tuy nhiên, mô hình LSTM cũng có một số hạn chế trong việc dự đoán giá cổ phiếu, như khó khăn trong việc xử lý dữ liệu liên tục và không tự động học các quan hệ giữa các input.

Mô hình transformer là một trong những mô hình deep learning mới và đang được sử dụng rộng rãi trong bài toán dự đoán giá cổ phiếu. Nó có khả năng xử lý dữ liệu liên tục nhanh hơn so với LSTM và có khả năng tự động học các quan hệ giữa các từ trong câu. Nó còn có khả năng học độc lập với thời gian, giúp cho việc dự đoán giá cổ phiếu tốt hơn trong trường hợp dữ liệu có nhiễu. Tuy nhiên, nó có chi phí tính toán cao hơn và yêu cầu dữ liệu lớn hơn so với LSTM.

Khi sử dụng 2 mô hình để dự đoán, khi lượng dữ liệu ngắn thì ta có thể sử dụng LSTM hoặc Transformer nhưng ở các tập dữ liệu lớn thì ta nên dùng transformer vì nó thể hiện tốt hơn LSTM ở mặt thời gian, độ chính xác và đặc biệt là tránh được các vấn đề bùng nổ hoặc mất mát gradient.

1. **Tài liệu tham khảo:**

[1]. Ashish Vaswani, “Attention Is All You Need”, 2017.

[2]. Tashreef Muhammad, Anika Bintee Aftab, Md. Mainul Ahsan, Maishameem Meherin Muhu, Muhammad Ibrahim, Shahidul Islam Khan, Mohammad Shafiul Alam, “Transformer-Based Deep Learning Model for Stock Price Prediction: A Case Study on Bangladesh Stock Market”, 2022.

[3]. Shaswat Mohanty, Anirudh Vijay, Nandagopan Gopakumar, “StockBot: Using LSTMs to Predict Stock Prices” 2022.

[4]. Chen Zhang, “Asset Pricing and Deep Learning”, 2022.

[5]. Zong-Yu Peng và Pei-Chang Guo, “A Data Organization Method for LSTM and Transformer When Predicting Chinese Banking Stock Prices”, 2022.

[6]. Huicheng Liu, “Leveraging Financial News for Stock Trend Prediction with Attention-Based Recurrent Neural Network”, 2018.