ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO TUẦN 6

Môn học: Project II

Chủ đề: Xây dựng mô hình dự đoán giá cổ phiếu bằng mạng Transformer

Giáo viên hướng dẫn:	**Đỗ Tuấn Anh**
Sinh viên thực hiện:	***Lê Minh Triết***
Mã số sinh viên:	**20220045**

Hà Nội - 2025

MỤC LỤC

Contents

MỤC LỤC 3

1. Công việc đã hoàn thành trong tuần 4

1.1. Xây dựng kiến trúc mô hình Transformer với Multi-Head Attention và Positional Encoding 4

1.2. Tối ưu siêu tham số mô hình, huấn luyện và lấy kết quả 6

2. Dự kiến các công việc tuần tới 6

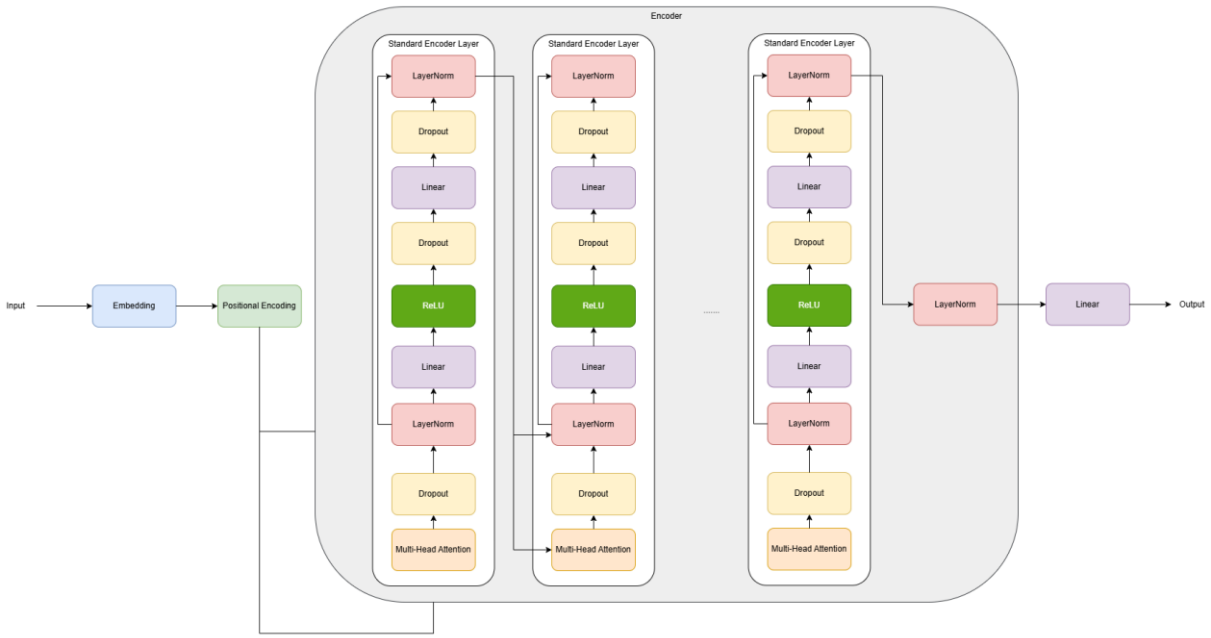
1. Công việc đã hoàn thành trong tuần

1.1. Xây dựng kiến trúc mô hình Transformer với Multi-Head Attention và Positional Encoding

Từ báo cáo tuần trước, em xây dựng mô hình Transformer với Multi-Head Attention với các thành phần chính sau:

1. **Positional Encoding:** Mã hóa vị trí
2. **Standard Encoder Layer:** Đại diện cho 1 lớp Encoder theo chuẩn
3. **Transformer Encoder:** Tập hợp các encoder.
4. **Standard Transformer Model:** Kết hợp Standard Transformer Encoder và một lớp Fully Connected cuối cùng

Kiến trúc chính có trong ảnh sau



Hình 1. Kiến trúc mạng Transformer cho bài toán Stock Prediction

Trước hết, em xây dựng ***Positinal Encoding***, một lớp giúp Transformer hiểu thứ tự các phần tử trong chuỗi thời gian bằng cách thêm thông tin vị trí của vector đầu vào.

Mỗi vị trí pos được tính bằng **hàm sin** (cho chỉ số chẵn) và **hàm cos** (cho chỉ số lẻ), với tần số giảm dần theo công thức:

$$PE(pos, 2i) = sin(pos/10^{(4i/d\_model)})$$

$$PE(pos, 2i+1) = cos(pos/10^{(4i/d\_model)})$$

Điều này giúp các vị trí gần nhau có biểu diễn tương tự nhưng vẫn giữ tính duy nhất. Giá trị của PE sau đó sẽ được cộng trực tiếp vào đầu vào (sau Embedding) của Transformer.

Các tham số của Positional Encoding bao gồm

- d_model: Số chiều của vector đầu vào cho Positional Encoding (thường là số chiều của các đặc trưng đầu vào đã qua Embedding).

- dropout: Tỉ lệ Dropout, sẽ sử dụng sau khi PE được cộng vào để tránh overfitting.
- max_len: Độ dài lớn nhất mà một chuỗi đầu vào có thể có.

Tiếp theo là xây dựng kiến trúc cho một ***Standard Encoder Layer***. Một Layer theo chuẩn mà em nói tới ở bài báo cáo trước có các thành phần theo thứ tự sau:

1. Lớp Self-Attetion, ở đây dùng Multi-head Attention.
2. Lớp Dropout thứ nhất. (Có Residual)
3. Lớp LayerNorm thứ nhất.
4. Lớp Linear thứ nhất.
5. Lớp Activation. (Sử dụng ReLU)
6. Lớp Dropout thứ hai.
7. Lớp Linear thứ hai.
8. Lớp Dropout thứ ba. (Có Residual)
9. Lớp LayerNorm thứ hai.

Đầu ra cuối cùng chính là đầu ra sau khi qua lớp LayerNorm thứ hai.

Thành phần này sẽ bao gồm các tham số

- d_model: Số chiều vector đầu vào.
- nhead: Số Head cho Multi-Head Attention.
- dim_ff: Số chiều trung gian cho 2 lớp Linear.
- dropout: Tỉ lệ dropout.

Tiếp theo là ***Transformer Encoder***. Nó sẽ là 1 lớp tổng quát cho các Encoder, gồm các tham số đầu vào

- encoder_layer: Một Encoder Layer được định nghĩa sẵn (như Encoder Layer ở trên).
- num_layers: Số lần duplicate lại encoder_layer để tạo thành 1 bộ encoder.

Cuối cùng là lớp ***Standard Transformer Model***. Nó sẽ kết hợp encoder và một lớp Fully Connected cuối cùng. Nó bao gồm các siêu tham số

- input_dim: Số đặc trưng đầu vào ban đầu.
- d_model: Số chiều vector đầu vào sau khi embedding.
- nhead: Số head dùng cho Multi-Head Attention.
- num_encoder_layers: Số lượng Encoder Layers được sử dụng.
- dim_ff: Số chiều Linear trung gian.
- dropout: Tỉ lệ dropout.
- max_len: Độ dài lớn nhất mà chuỗi đầu vào có thể có.

- output_dim: Số chiều đầu ra. (Ở đây đặt là 1 vì chỉ dự đoán Close Price).
- ndays: Số ngày trong tương lai cần dự đoán

Source code cho phần này ở

https://github.com/trietp1253201581/StockPrediction/blob/main/model/transformer.py

1.2. Tối ưu siêu tham số mô hình, huấn luyện và lấy kết quả

Em sử dụng Optuna để tối ưu các siêu tham số của mô hình, cụ thể danh sách các tham số và giá trị để thử là

- d_model: 64,128
- nhead: 2, 4, 8
- num_encoder_layers: 2, 4, 6
- dim_ff: 256, 512, 1024
- dropout: 0.1, 0.2, 0.3
- lr (dùng trong hàm train): 1e-5 -> 1e-3

Chi tiết xem ở Kaggle NoteBook

https://www.kaggle.com/code/trietp1253201581/stock-prediction-test-model

(version 2)

Sau khi chạy 30 lần thử, em thu được bộ tham số tốt nhất là

```
{'d_model': 128, 'nhead': 4, 'num_encoder_layers': 4, 'dim_ff': 256, 'dropout': 0.1, 'lr': 0.00014157174523138512}
```

Sau đó em huấn luyện mô hình với bộ tham số này và đạt được sai số MSE trên tập test là 5603 (trong khi Baseline là 3048). Chi tiết xem ở version 3 của notebook

https://www.kaggle.com/code/trietp1253201581/stock-prediction-test-model

Điều này dẫn tới phải cải thiện mô hình Transformer này.

2. Dự kiến các công việc tuần tới

Thay thế thành phần MultiHead Attention bằng Local Attention và Causal Attention, tối ưu tham số và đánh giá