目次

- 発表の流れ
- 実験設定
  - Transformerモデル
  - 特徴量
  - ハイパラ調整
- 結果/考察
- アプリケーション
- 感想

発表の内容

- 時系列データにおけるTransformerのアーキテクチャを再確認する(訓練結果の考察・改善のために必要な理解！！)
- Transformer を時系列データ解析へ適応し, ベースラインモデル、従来のモデル(LSTM)と性能を比較する.
- Transformerモデルのハイパラ調整による推測性能変化を実験的に検証する.

モデル開発の目標

＜問題設定＞株式銘柄FANUCの直近６０日間の株価の終値から次の５日間の予測をする.

今回は設定を単純化するため以下の条件下で開発する.

・**FANUCのみ**の過去の**終値のデータのみ**が利用可能(他の銘柄は使用しない)
・テストデータは過去2003-2024のうち**過去15%**を使用するため, 利用不可
・転移学習を含む, 基盤モデルの**ファインチューニングは禁止**
・**スクラッチ実装**(Torchベースで可)

Transformerとは　　“Attention is all you need”

Vaswani,Aらにより2017年Attention is all you needで発表されたself-attention層を積層した新しい深層学習機構.

部分的に改変した様々なモデルが開発されている. Vanilla Transformerは元論文に示されているTransformerを指す.

X: 入力ベクトル$\mathbb{R}^{is\times bs\times d}$ (is: 入力系列サイズ, bs: バッチサイズ, d: 埋め込み次元)

フィードフォーワード

$$Linear(X) = XW^T_{decoder} + b_{decoder} \quad (\boldsymbol{b}, \boldsymbol{W_{decoder}}: \text{重み} \mathbb{R}^{1\times d}, \mathbb{R}^1)$$

$$FeedForward(X) = \boldsymbol{W_2} ReLu(\boldsymbol{W_1} x + \boldsymbol{b_1}) + \boldsymbol{b_2}$$

Self-Attentionブロック

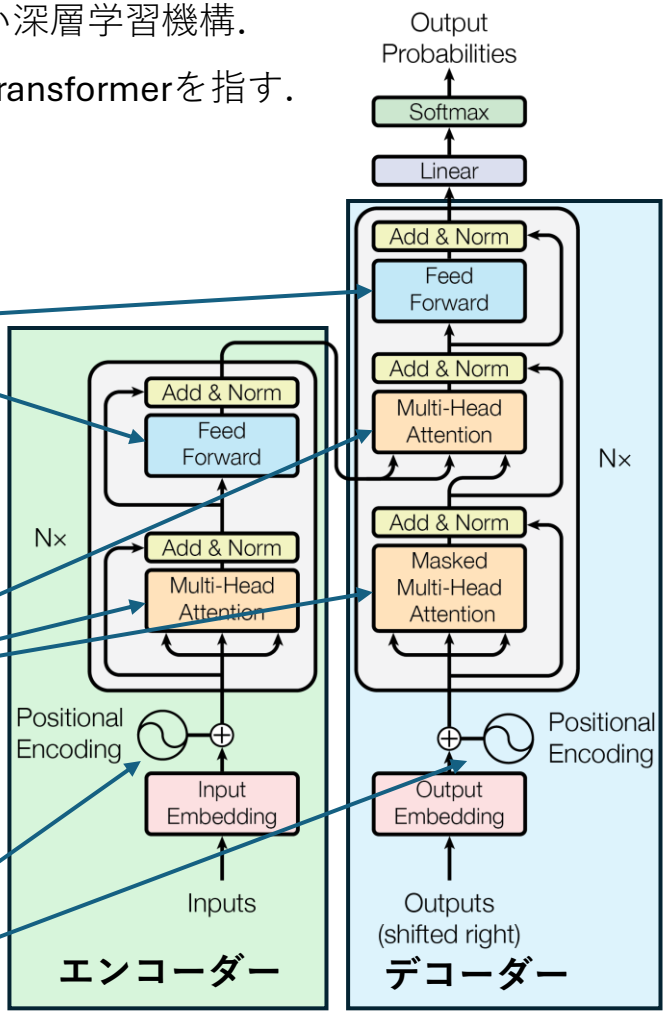
$$Q = X\boldsymbol{W_Q}, K = X\boldsymbol{W_K}, V = X\boldsymbol{W_V} \quad (\boldsymbol{W_{Q,K,V}}: \text{重み} \mathbb{R}^{d\times d_k}, d_k = \frac{d}{nhead}, nhead: \text{ヘッド数})$$

$$\mathrm{Z} = \mathrm{Attention(Q, K, V)} = Softmax(\frac{QK^T}{\sqrt{d_k}} + M_{ij})V \quad (M_{ij}\text{マスク}: \{0 \; if \; i \geq j, -\infty \; if \; i < j\})$$

$$\mathrm{MultiHeadAttention(Q, K, V)} = Concat(Z_i)\boldsymbol{W^O} \quad (i = 0 \ldots nhead, \boldsymbol{W^O}: \text{重み} \mathbb{R}^{d\times d})$$

位置エンコーディング

$$\mathrm{Positional Encoding}(X) = X + POS_l \left(\begin{array}{c} POS_l = \sin\left(\frac{k}{10000^{\frac{l}{d}}}\right) if \; l = 2i, \cos\left(\frac{k}{10000^{\frac{l}{d}}}\right) \\ if \; l = 2i+1, k = 1 \ldots is, l = 1 \ldots d \end{array} \right)$$



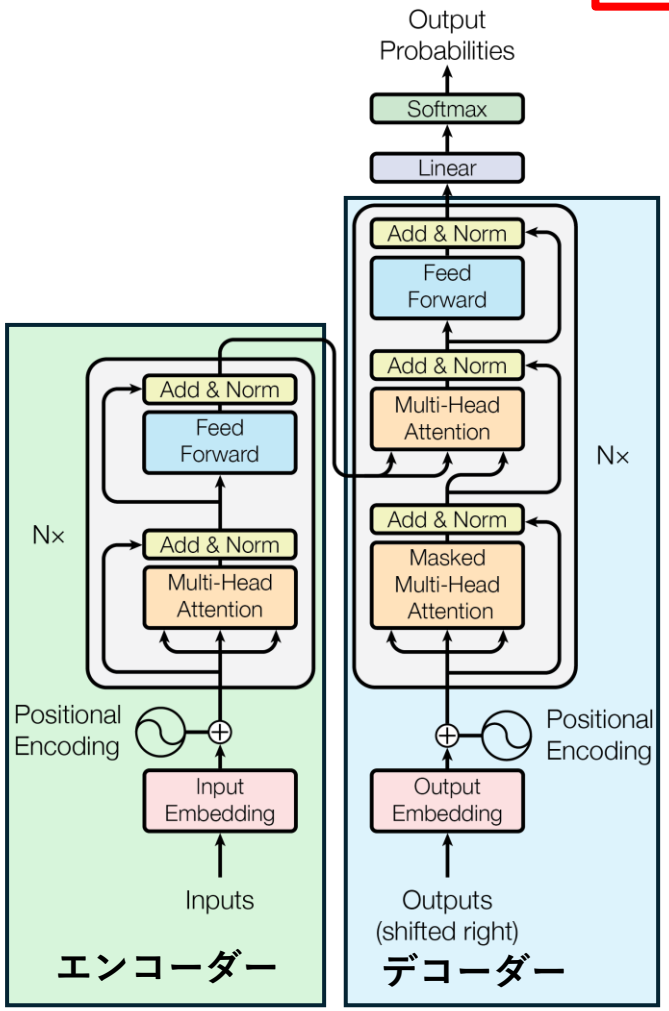
★今回実装するTransformer はVanillaではない

[図]Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017, June 12). Attention is all you need. arXiv.org.

時系列データ解析におけるTransformer　ー概要ー

Transformerとは？

Self-Attention機構を含むエンコーダーとデコーダーからなる深層学習モデル.



デコーダー, エンコーダー単体で使われることも多く, それぞれ役割が異なる.

NLPにおいて,

★エンコーダーモデル
Attention層+フィードフォーワード層で構成. Attention層でtoken前後との関連性を計算し文章を理解する.
＜代表的なモデル＞ BERT

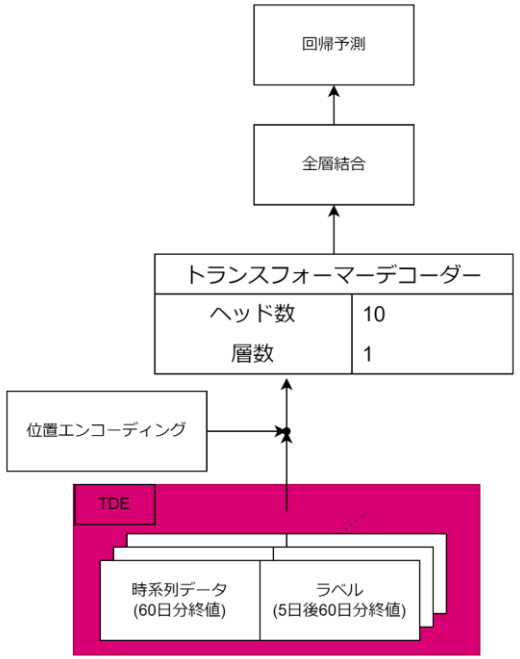
★デコーダーモデル(今回実装するモデル)
Masked-Attention層+ Cross-Attention層+フィードフォーワード層で構成. Masked-Attention層でtokenの因果関係を保持するために予測対象はMask. 続く単語を推測するタスクに適する.
＜代表的なモデル＞GPT

時系列データ解析においては, 現在の値が過去のデータに基づいて決定されるという性質上, **自己回帰的な性質をもつデコーダーモデルが適切である.**

[図]Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017, June 12). Attention is all you need. arXiv.org.

時系列データ解析におけるTransformerの実装　ー特徴量ー

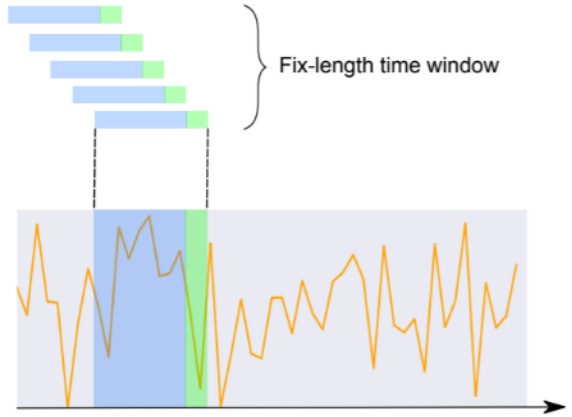
NLPと異なり，時系列データ解析におけるTransformerではEmbeddingは行わずTime Delay Embedding(TDE)を施す[1]



$$TDE_{d\tau}(x_t) = \left(x_t, x_{t-\tau} \dots, x_{t-(d-1)\tau}\right)$$

d: 埋め込み次元, τ: スライド数

TDEは過去の値を一定の遅延（タイムラグ）を持つ形で取り出し，現在の状態や未来の予測に利用できるデータセットを作成する．



↑TDEの作成イメージ[1]

＜自然言語処理＞

「I love NLP」をモデルに入力する場合

「I」　$\overrightarrow{X_1} = (0,0,1,\dots)$
「love」　$\overrightarrow{X_2} = (0,0,0\ \dots)$
「NLP」　$\overrightarrow{X_3} = (1,1,0,\dots)$

Token化した後，one-hotやword2vecなどで 単語をベクトル表現に変換．

＜時系列解析＞

$d = 10, \tau = 1$として，2日後の予測をする場合

$\overrightarrow{X_1} = (x_1, x_2, \dots x_{10})$ $\overrightarrow{Y_1} = (x_3, x_4, \dots x_{12})$
$\overrightarrow{X_2} = (x_2, x_3, \dots x_{11})$ $\overrightarrow{Y_2} = (x_4, x_5, \dots x_{13})$
$\overrightarrow{X_3} = (x_3, x_4, \dots x_{12})$ $\overrightarrow{Y_3} = (x_5, x_6, \dots x_{14})$
…
$\overrightarrow{X_i} = (x_i, x_{i+1}, \dots x_{i+9})$ $\overrightarrow{Y_i} = (x_{i+2}, x_{i+3}, \dots x_{i+11})$

X:入力，Y:ラベル，$i = 1$〜利用可能なデータ数

★時系列解析は自然言語処理と異なり，初めから数値表現のデータを扱うため，特殊な変換が不要．一方，一定の長さで分割する，ある種の「埋め込み」を行う．

TDE 定性的性質

埋め込み次元数dによる影響

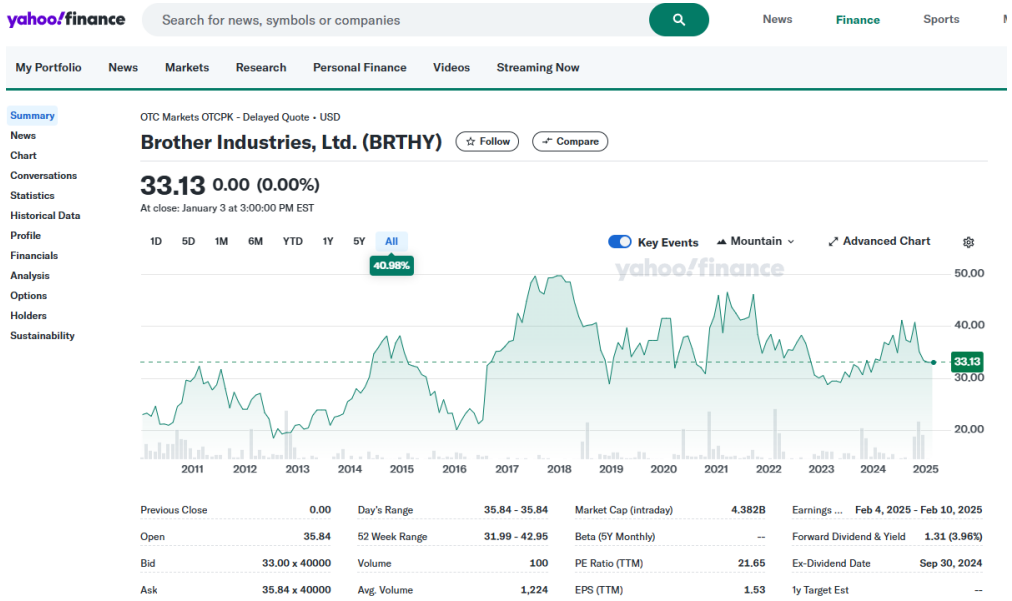
- 季節性(データに即したdに決定)
- 計算コスト(dが大きいと計算コスト大)

Transformerは長距離依存性を捉えるのが得意であるため，埋め込み次元は長くてもLSTMなどのRNN系モデルより高い性能を出すことが可能ではないか．

[1] Wu, N., Green, B., Ben, X., & O'Banion, S. (2020, January 23). Deep transformer models for time series forecasting: The influenza prevalence case. arXiv.org.

時系列データ解析におけるTransformerの実装　ー特徴量ー

①Yahoo!FinanceからBrotherの株式の終値を取得



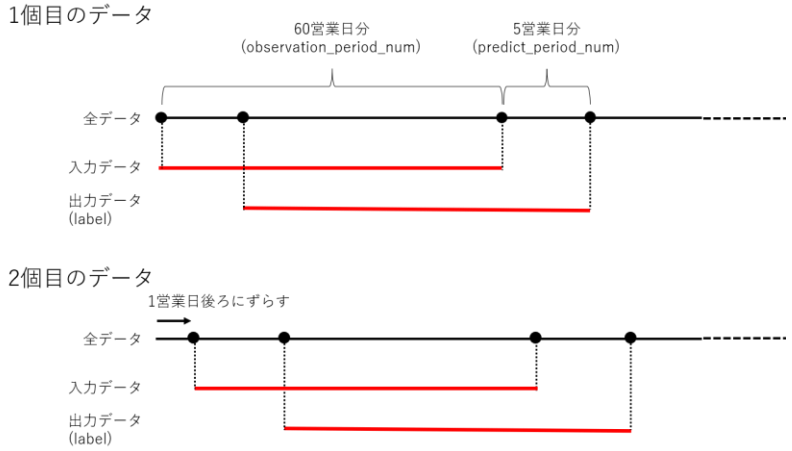
[図1]Yahoo!Financeの株式データ

2003年1月1日～2024年12月31日まで.
東証が閉まっている日や取引をしていない日があり, 約3700日分取得した。

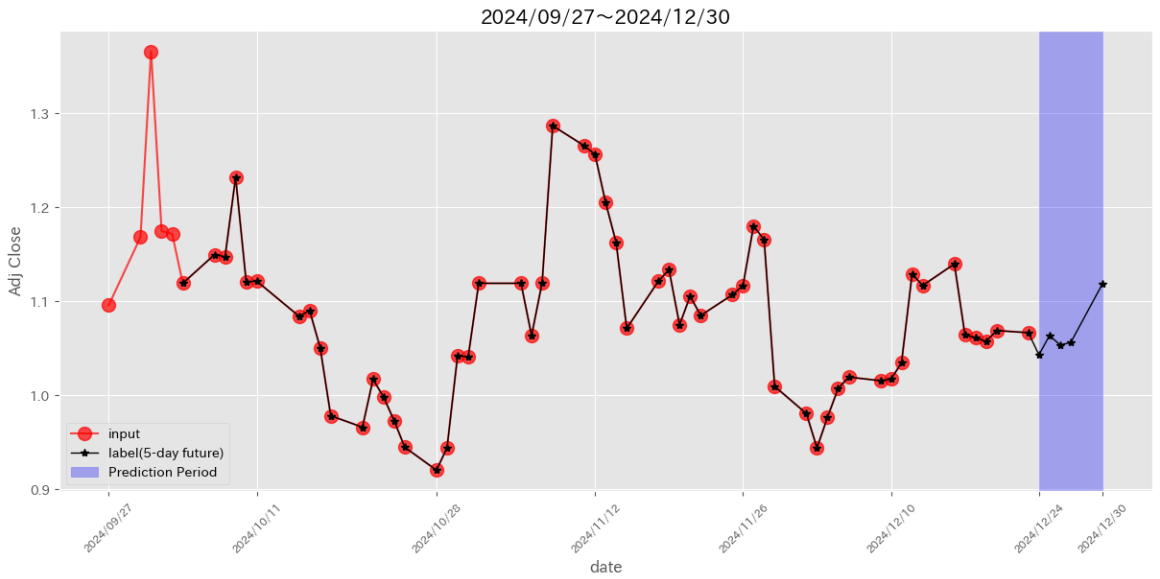
X入力データ　　Y出力データ

$$\begin{pmatrix} (2003年1月1日～2003年3月1日) & (2003年1月6日～2003年3月6日) \\ (2003年1月2日～2003年3月2日) & (2003年1月7日～2003年3月7日) \\ (2003年1月3日～2003年3月3日) & (2003年1月8日～2003年3月8日) \\ \cdots & \cdots \\ (2024年10月28日～2024年12月26日) & (2003年11月2日～2024年12月31日) \end{pmatrix}$$

②データを学習用に整理する。



[図2]今回のTDE設定
区間:60日
スライド幅:1日
予測:5日後



[図3] 入力データと出力データの対応, 予測区間

時系列データ解析におけるTransformerの実装　ー特徴量ー

＜データ分割＞
Train:Validation:Test= 70:15:15で分割. 時系列はTrain, Validation, Testの順に新しく, リークを防ぐためシャッフルは行っていない.

2003-2018のデータに基づいて2022-2024の推測をすることになる。

2019-2021のデータが十分に活用できていない.
株式価格は短距離依存性も十分にあるため, validationに使ったデータも訓練させたい

今回はデータ数(約3000)が十分ではないため, 上の分割プリセットでハイパラ調整. 最近のトレンドを学習するためValidation もTrainに入れた以下の分割プリセットで再学習して最終的な推測をした.

この時は, すでにハイパラが調整済であるため固定でループ

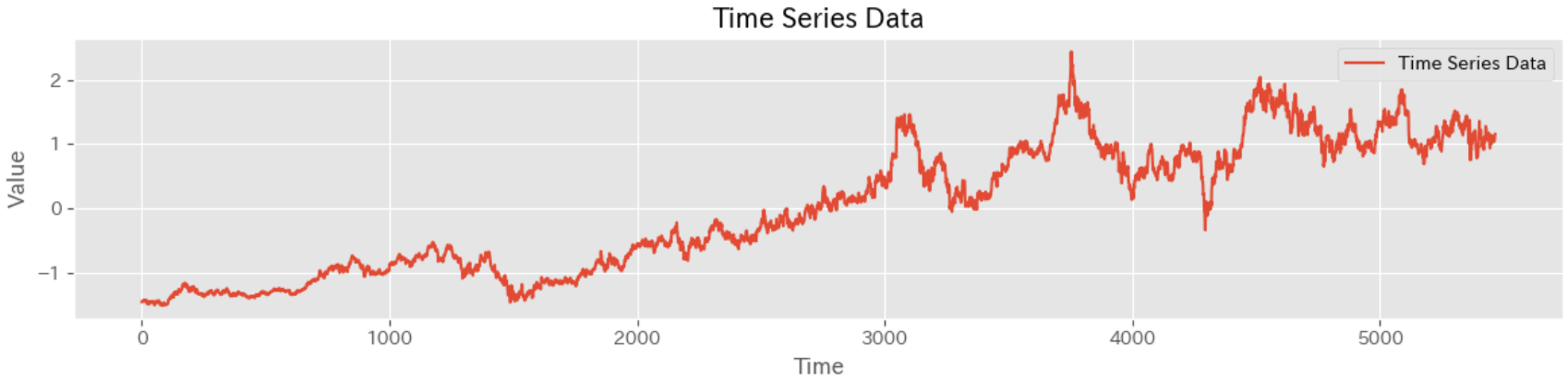
時系列データ解析におけるTransformerの実装　ー特徴量ー

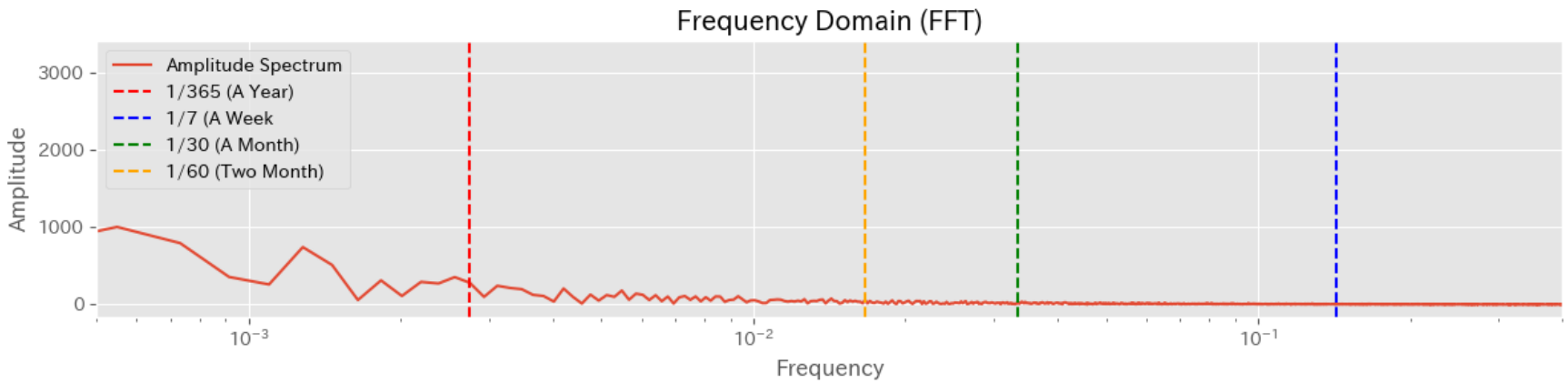
季節性とは？　時系列データにおいて特定の周期で繰り返されるパターンや変動.

<季節性分析>　学習モデルが季節性トレンドを捉えるために,時系列データを適切に分割して入力する必要がある.ここでは取り扱う株式価格推移データがどのような季節性を内包しているか調査する.

有名な株式市場の季節性

- １月効果
- セルインメイ
- ８月の夏枯れ相場
- 掉尾の一振





FFTを用いてデータの表現方法を変換

・**時間軸表現**(図上側)
データが時間経過に対してどのように変動するか

FFT変換

・**周波数表現**(図上側)
角周波数がデータ内でどれほど強く表れているか

存在しそうな季節性の候補
・年間周期（1/365 Hz）
・2か月周期（1/60 Hz）
↑今回のデータ分割設定
・月間周期（1/30 Hz）
・週間周期（1/7 Hz）

季節性が隠れている可能性があったが,FANUC株には特に目立ったスペクトルは見られなかった.

事業がイベントに大きく依存せず,年間を通して安定した需要があるため.銘柄によっては季節性があるのでは.

[図1] 上:時系列データ全体,
下:FFT後のスペクトルの分布

時系列データ解析におけるTransformerの実装　ーハイパラ調整ー

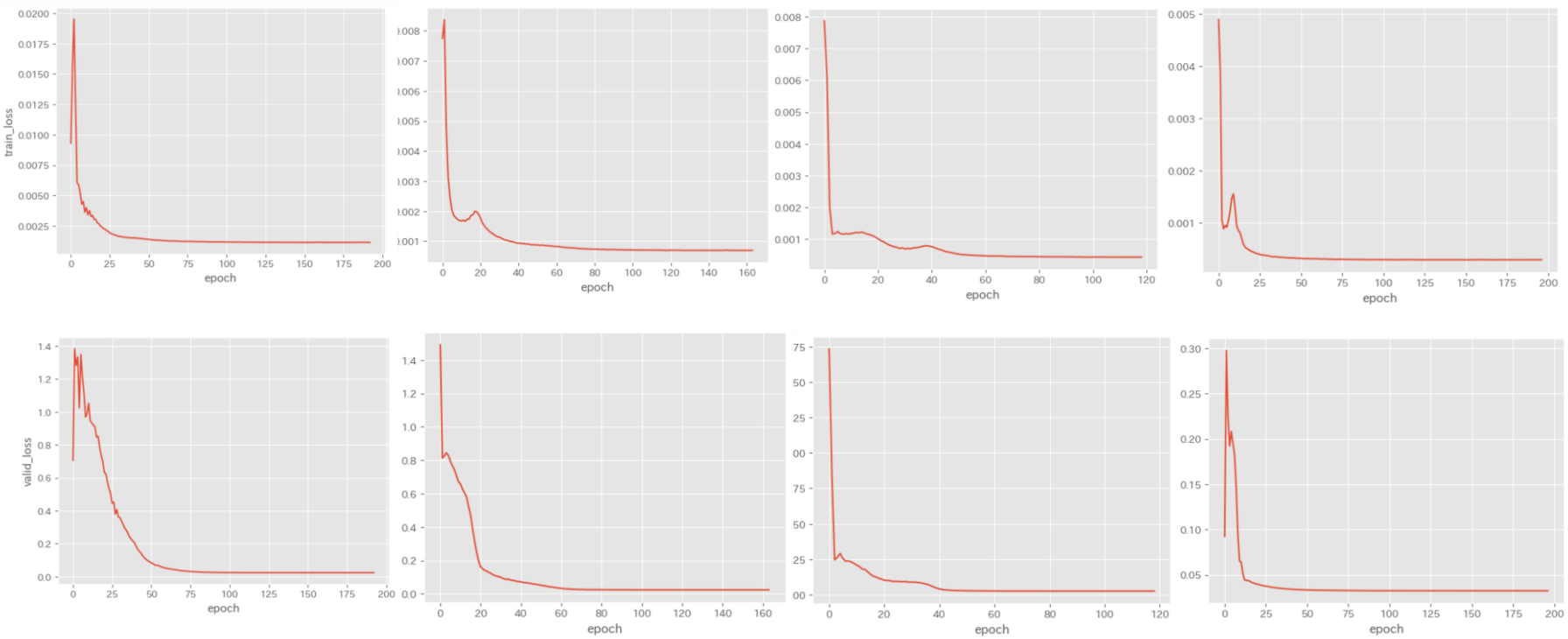
バッチサイズの大きさはどれが適切であるか？

base /2	
バッチサイズ	8
学習率	0.00005
エポック	1000(193)
層数	1

base	
バッチサイズ	16
学習率	0.00005
エポック	1000(160)
層数	1

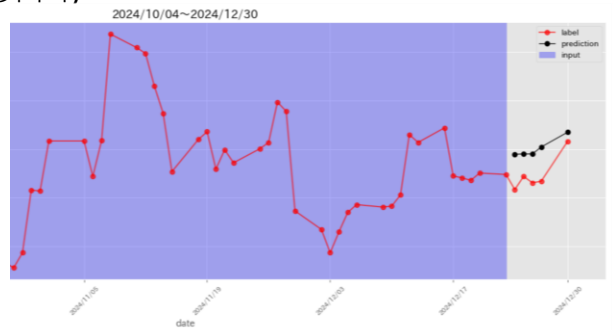
base x2	
バッチサイズ	32
学習率	0.00005
エポック	1000(120)
層数	1

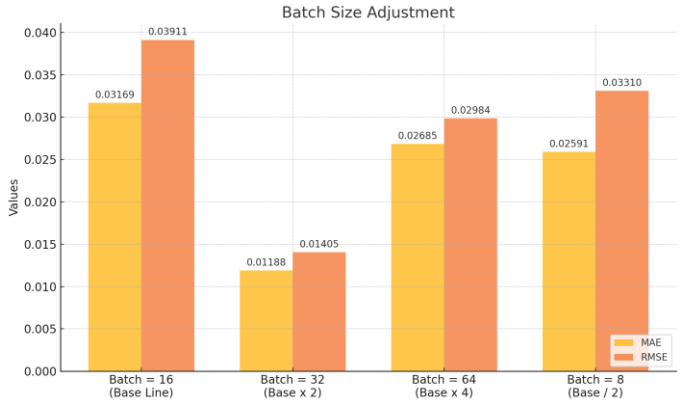
base x4	
バッチサイズ	64
学習率	0.00005
エポック	1000(200)
層数	1



[図1]バッチサイズごとのtrain, validation lossの推移

[図2]↓MAE(Mean Absolute Error), RMSE(Root Mean Square Error)による評価結果
(2024-12-24から5日営業日予測と実際データから算出)





バッチサイズ=32で最も良い性能を示すことが分かった.

時系列データ解析におけるTransformerの実装　ーハイパラ調整ー

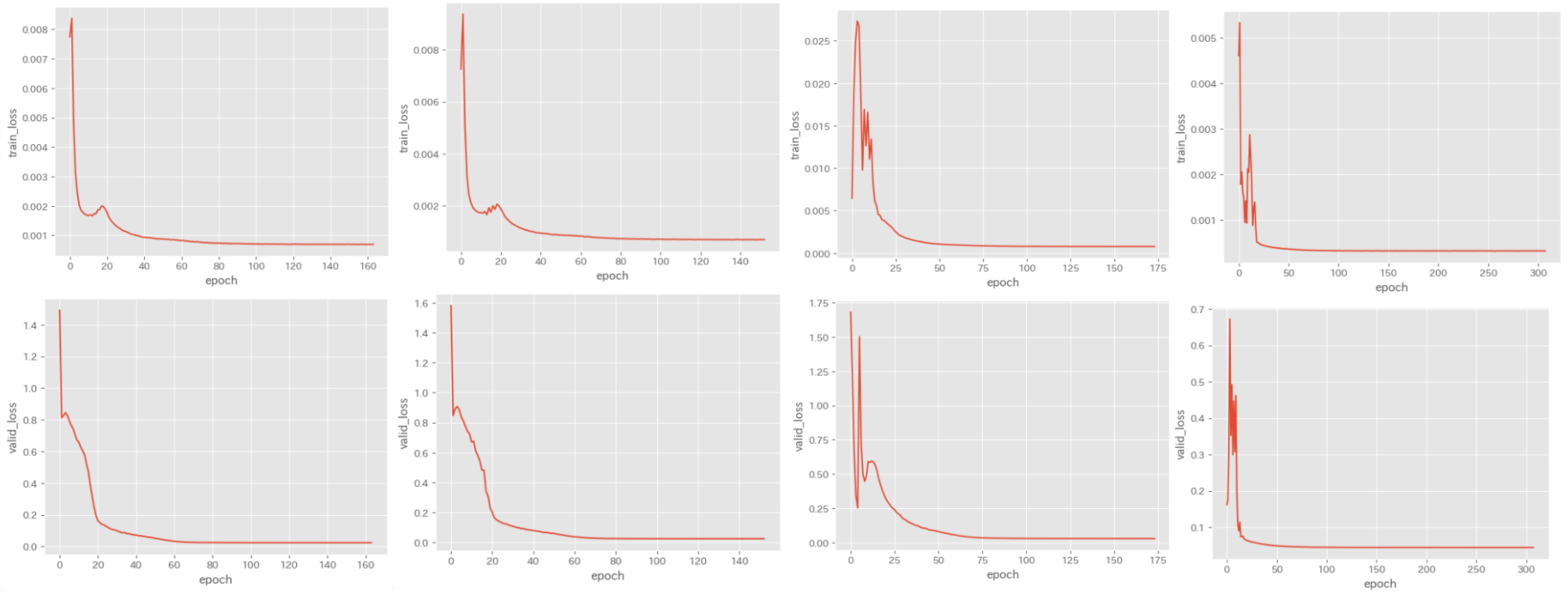
Transformer層の数は何層が適切であるか？

base	
バッチサイズ	16
学習率	0.00005
エポック	1000(160)
層数	1

base x2	
バッチサイズ	16
学習率	0.00005
エポック	1000(150)
層数	2

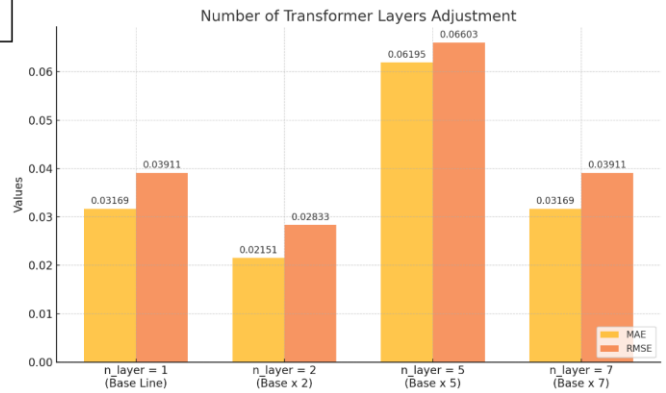
base x 5	
バッチサイズ	16
学習率	0.00005
エポック	1000(175)
層数	5

base x7	
バッチサイズ	64
学習率	0.00005
エポック	1000(300)
層数	7



[図1]Transformer層数ごとのtrain, validation lossの推移

[図2]↓MAE(Mean Absolute Error), RMSE(Root Mean Square Error)による評価結果



層数=2で最も良い性能を示すことが分かった.

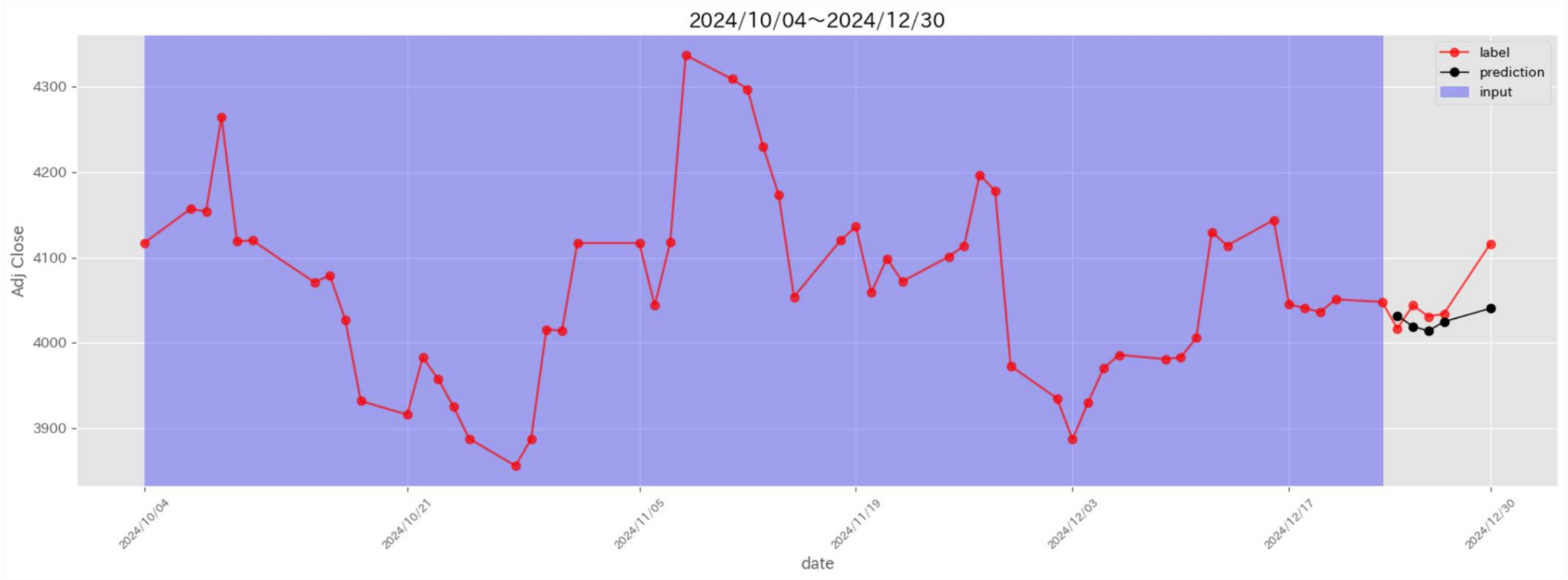
以上より,

- **バッチサイズ=32**
- **層数=2**

ー結果/考察ー　　5日間予測(短期予測)

本モデルの開発目標である, 5日間未来の株価推移の予測結果を示す. y軸はUSDである.

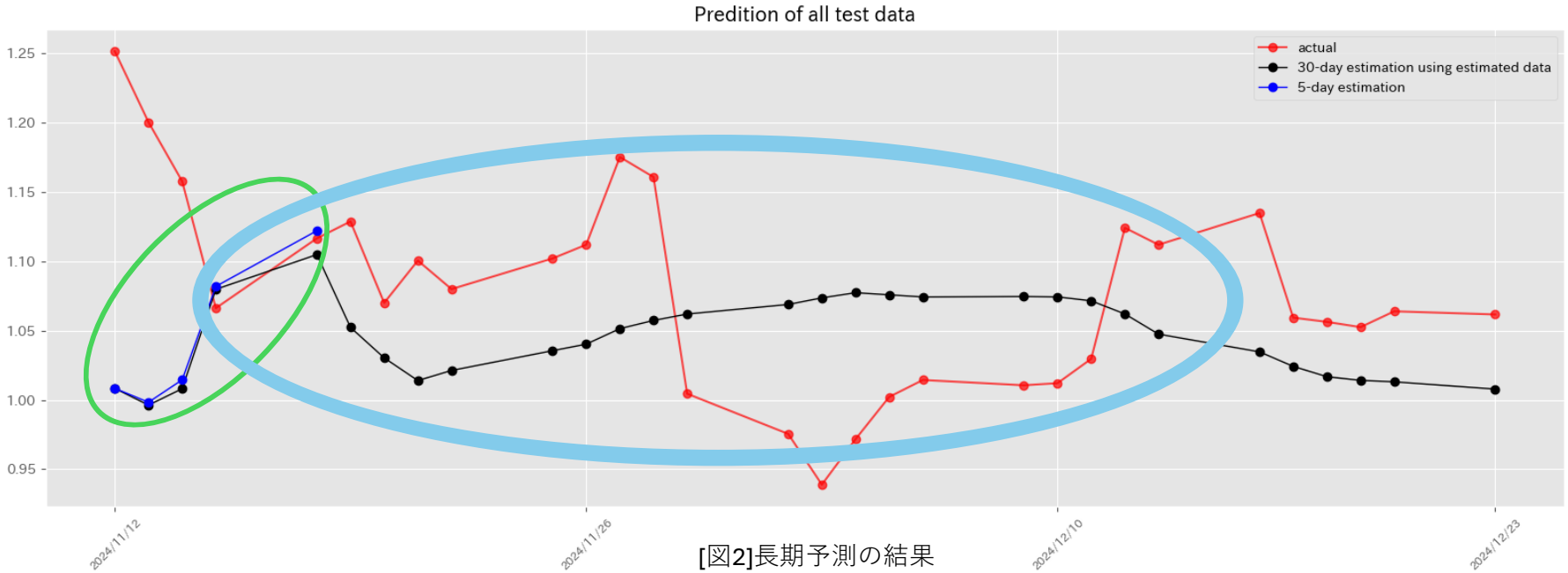
12/24からの5日間予測のMAPE(Mean Absolute Percentage Error)は約1%であり, 十分小さい誤差で良い予測ができていることが分かる.



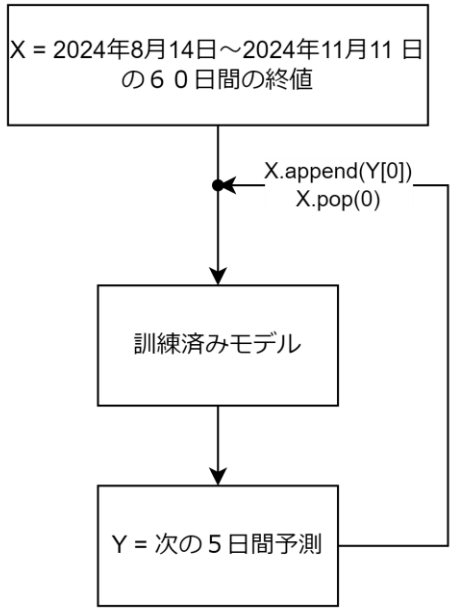
[図1]短期予測の結果

ー結果/考察ー　30日間予測(逐次予測)

2024年8月14日～2024年11月11 日までのデータを入力し, 得られた5日間予測の内1日目を次の入力に使用し, 繰り返し予測, 30日間続ける.



[図2]長期予測の結果



[図2]長期予測の方法

・5日間予測と30日間予測が極めて一致している. **モデルが入力の傾向を捉えている良い現象. ハイパラ調整の成果**

・降下した後再び上昇する傾向がつかめている. **ノイズのような動きは追従できないため, 期待通りの挙動**

LSTMとの比較

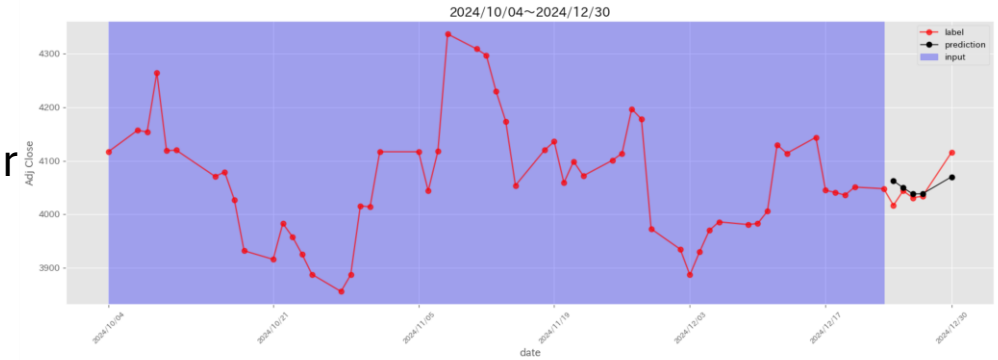
5日間予測の場合

MAPE

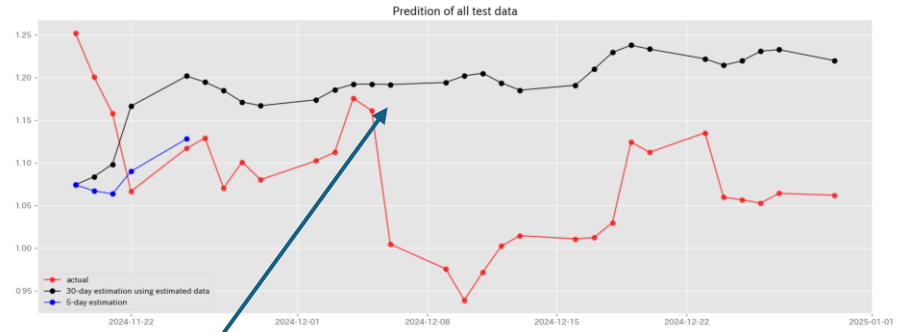
30日予測の場合

MAE

Transformer

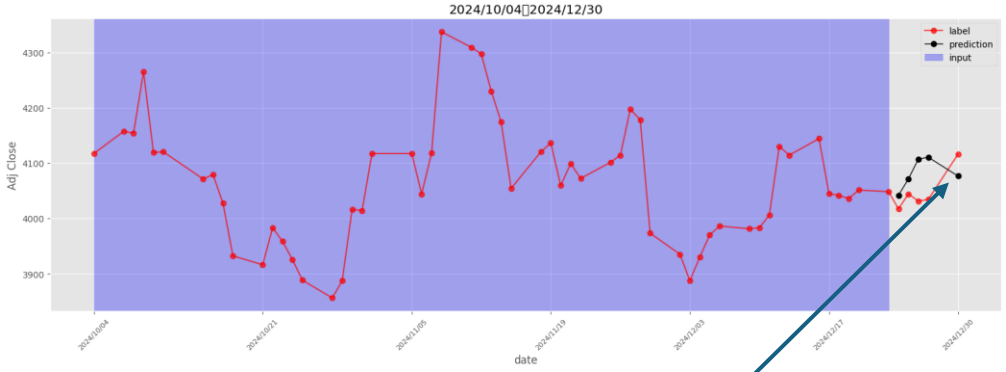


0.84%

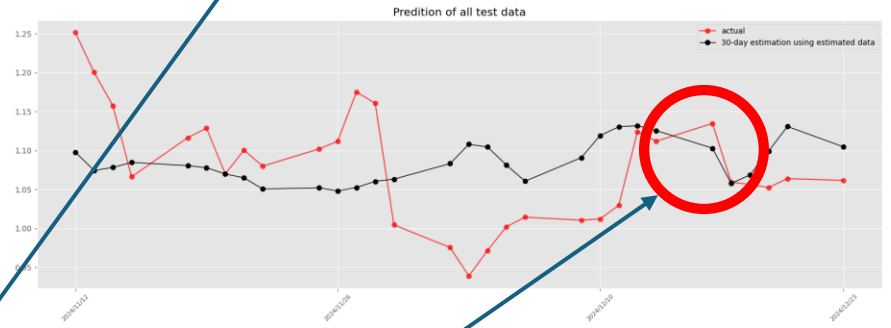


0.065

LSTM



2.8%



0.07

・Transformer は 5 日間の上昇傾向がつかめていたのに対し, LSTMは最後の一日で大きく誤った傾向を予測してしまった.

・Transformer の方は30日間予測で遠い未来がLSTMと比べて q 予測できなくなっているように見える.

段のような挙動が予測できている可能性がある.

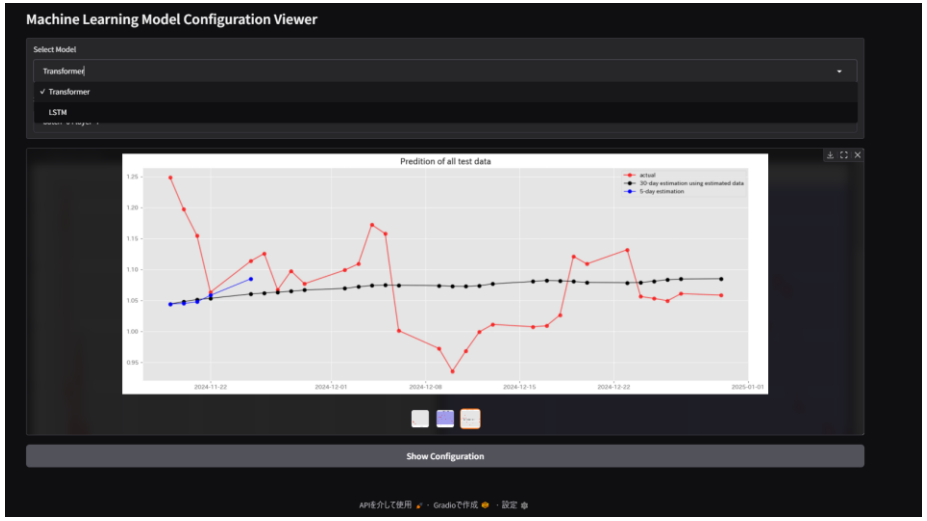
アプリケーション開発

“Machine Learning Model Configuration Viewer”

ハイパラ調整の過程で得られたモデルによる様々な予測のショーケースアプリケーションを開発しました.

gradio

Gradio上で公開しています.



←知りたいモデルを選択(現在はLSTMとTransformer の 2 つのみ)

→更に細かい設定を選択して, 予測済みのデータが表示される