no2-2501980690

July 8, 2023

1 FINAL EXAM - UAS DEEP LEARNING No 2

Nama : Agnes Calista

NIM : 2501980690

Link Video : https://youtu.be/DpIhEsz2bsQ

1.1 import libraries

[ ]:
```
pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-
packages (2.12.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (23.5.26)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.56.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (3.8.0)
Requirement already satisfied: jax>=0.3.15 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (0.4.10)
Requirement already satisfied: keras<2.13,>=2.12.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: libclang>=13.0.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.0)
Requirement already satisfied: numpy<1.24,>=1.22 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.22.4)
```

```
Requirement already satisfied: opt-einsum>=2.3.2 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (23.1)
Requirement already satisfied:
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3
in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.20.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.16.0)
Requirement already satisfied: tensorboard<2.13,>=2.12 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.3)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: termcolor>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (4.6.3)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.32.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in
/usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow)
(0.40.0)
Requirement already satisfied: ml-dtypes>=0.1.0 in
/usr/local/lib/python3.10/dist-packages (from jax>=0.3.15->tensorflow) (0.2.0)
Requirement already satisfied: scipy>=1.7 in /usr/local/lib/python3.10/dist-
packages (from jax>=0.3.15->tensorflow) (1.10.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow) (2.17.3)
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow) (1.0.0)
Requirement already satisfied: markdown>=2.6.8 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow) (3.4.3)
Requirement already satisfied: requests<3,>=2.21.0 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow) (2.27.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow) (0.7.1)
Requirement already satisfied: werkzeug>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.13,>=2.12->tensorflow) (2.3.6)
```

```
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (5.3.1)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (0.3.0)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-
packages (from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from google-auth-
oauthlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow) (1.3.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (2023.5.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (2.0.12)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow) (3.4)
Requirement already satisfied: MarkupSafe>=2.1.1 in
/usr/local/lib/python3.10/dist-packages (from
werkzeug>=1.0.1->tensorboard<2.13,>=2.12->tensorflow) (2.1.3)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in
/usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow) (0.5.0)
Requirement already satisfied: oauthlib>=3.0.0 in
/usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow) (3.2.2)
```

[ ]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Flatten,LSTM, Dense, Flatten, Embedding

import re
```

```python
import keras
from keras import Model
from tensorflow.keras.layers import Flatten,LSTM, Dense, Flatten, Embedding
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential

from keras.initializers import glorot_uniform
from sklearn import model_selection
import keras.layers as layers

import csv
from datetime import datetime
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score
import math
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_squared_error, mean_absolute_error,␣
 ↪mean_absolute_percentage_error
```

1.2 import dataset & preprocessing

[ ]:
```python
dfAMZN = pd.read_csv("AMZN.csv",
                 parse_dates=["Date"],
                 index_col=["Date"])
dfAMZN.head()
```

[ ]:

Date	Open	High	Low	Close	Adj Close	Volume
1997-05-15	2.437500	2.500000	1.927083	1.958333	1.958333	72156000
1997-05-16	1.968750	1.979167	1.708333	1.729167	1.729167	14700000
1997-05-19	1.760417	1.770833	1.625000	1.708333	1.708333	6106800
1997-05-20	1.729167	1.750000	1.635417	1.635417	1.635417	5467200
1997-05-21	1.635417	1.645833	1.375000	1.427083	1.427083	18853200

[ ]:
```python
# Buat variabel baru yang berisi
pricesAmazon = pd.DataFrame(dfAMZN["Close"]).rename(columns={"Close": "Price"})
pricesAmazon.head()
```

[ ]:

Date	Price
1997-05-15	1.958333
1997-05-16	1.729167
1997-05-19	1.708333
1997-05-20	1.635417
1997-05-21	1.427083

[ ]:
```
WINDOW_SIZE = 5

HORIZON = 5
```

Memisahkan data n data time series tersebut menjadi dua bagian input dan output dengan window size = 5 [dari hari senin s.d jumat] dan horizon = 5 [dari hari senin s.d jumat].

[ ]:
```
# Get AMZN date array
timesteps1 = pricesAmazon.index.to_numpy()
prices1 = pricesAmazon["Price"].to_numpy()

timesteps1[:10], prices1[:10]
```

[ ]:
```
(array(['1997-05-15T00:00:00.000000000', '1997-05-16T00:00:00.000000000',
        '1997-05-19T00:00:00.000000000', '1997-05-20T00:00:00.000000000',
        '1997-05-21T00:00:00.000000000', '1997-05-22T00:00:00.000000000',
        '1997-05-23T00:00:00.000000000', '1997-05-27T00:00:00.000000000',
        '1997-05-28T00:00:00.000000000', '1997-05-29T00:00:00.000000000'],
       dtype='datetime64[ns]'),
 array([1.95833337, 1.72916663, 1.70833337, 1.63541663, 1.42708337,
        1.39583337, 1.5       , 1.58333337, 1.53125   , 1.50520837]))
```

code diatas untuk mempersiapkan timestamps dan price dalam format yang sesuai untuk proses selanjutnya. Code diatas untuk mengambil timesteps dan harga saham dari dataframe pricesAmazon kemudian mengubahnya dalam numpy array.

[ ]:
```
def get_labelled_windows(x, horizon=1):
  return x[:, :-horizon], x[:, -horizon:]

test_window, test_label = get_labelled_windows(tf.expand_dims(tf.range(6)+1,␣
 ↪axis=0), horizon=HORIZON)
print(f"Window: {tf.squeeze(test_window).numpy()} -> Label: {tf.
 ↪squeeze(test_label).numpy()}")
```

```
Window: [1 2 3 4 5] -> Label: 6
```

[ ]:
```
def make_windows(x, window_size=5, horizon=1):

  window_step = np.expand_dims(np.arange(window_size+horizon), axis=0)
  window_indexes = window_step + np.expand_dims(np.
↪arange(len(x)-(window_size+horizon-1)), axis=0).T
  windowed_array = x[window_indexes]
  windows, labels = get_labelled_windows(windowed_array, horizon=horizon)

  return windows, labels
```

```
[ ]: full_windows1, full_labels1 = make_windows(prices1, window_size=WINDOW_SIZE,␣
      ↪horizon=HORIZON)
     len(full_windows1), len(full_labels1)
```

```
[ ]: (5753, 5753)
```

1.3 Create train tes val split

```
[ ]: def make_train_val_test_splits(windows, labels, val_split=0.1, test_split=0.1):

         total_size = len(windows)
         train_size = int(total_size * 0.8)
         val_size = int(total_size * 0.1)
         test_size = total_size - train_size - val_size

         train_windows = windows[:train_size]
         train_labels = labels[:train_size]
         val_windows = windows[train_size:train_size+val_size]
         val_labels = labels[train_size:train_size+val_size]
         test_windows = windows[train_size+val_size:]
         test_labels = labels[train_size+val_size:]

         return train_windows, val_windows, test_windows, train_labels, val_labels,␣
      ↪test_labels
```

```
[ ]: train_windows1, val_windows1, test_windows1, train_labels1, val_labels1,␣
      ↪test_labels1= make_train_val_test_splits(full_windows1, full_labels1)
     print("Train set length:", len(train_windows1))
     print("Validation set length:", len(val_windows1))
     print("Test set length:", len(test_windows1))
     print("Train labels length:", len(train_labels1))
     print("Validation labels length:", len(val_labels1))
     print("Test labels length:", len(test_labels1))
```

```
Train set length: 4602
Validation set length: 575
Test set length: 576
Train labels length: 4602
Validation labels length: 575
Test labels length: 576
```

```
[ ]: print('Train set: {} baris , {} kolom'.format(train_windows1.shape[0],␣
      ↪train_windows1.shape[1]))
     print('Test set: {} baris , {} kolom'.format(test_windows1.shape[0],␣
      ↪test_windows1.shape[1]))
     print('Validation set: {} baris , {} kolom'.format(val_windows1.shape[0],␣
      ↪val_windows1.shape[1]))
```

```
Train set: 4602 baris , 5 kolom
Test set: 576 baris , 5 kolom
Validation set: 575 baris , 5 kolom
```

[ ]:
```
# Scaling training set
scaled = MinMaxScaler(feature_range=(0,1))
training_set_scaled = scaled.fit_transform(train_windows1)
test_set_scaled = scaled.fit_transform(test_windows1)
val_set_scaled = scaled.fit_transform(val_windows1)
```

- Scaling. Pada proses scaling saya menggunakan MinMaxScaler karena dalam paper Deepa et.al., MinMaxScaler dapat mengubah data kedalam rentang yang telah ditentukan. contohnya 0 sampai 1. dengan memakai MinMaxScaler rentang data yang sama dapat memperbaiki stabilitas algoritma machine learning. selain itu dengan MinMaxScaler data yang memiliki tingkat range yang jauh akan menghasilkan nilai yang tidak akurat, jadi dengan menggunakan MinMaxScaler akan mengubah skala menjadi rentang yang sama sehingga model yang dihasilkan akan lebih konsisten.

referensi Deepa, B., & Ramesh, K. (2022). Epileptic seizure detection using deep learning through min max scaler normalization. Int. J. Health Sci, 6, 10981-10996.

[ ]:
```
timesteps = 8

x_train = []
y_train = []

x_test = []
y_test = []

x_val = []
y_val = []

for i in range(timesteps,train_windows1.shape[0]):
    x_train.append(training_set_scaled[i-timesteps:i,0])
    y_train.append(training_set_scaled[i,0])
x_train, y_train = np.array(x_train), np.array(y_train)

for i in range(timesteps,test_windows1.shape[0]):
    x_test.append(test_set_scaled[i-timesteps:i,0])
    y_test.append(test_set_scaled[i,0])
x_test, y_test = np.array(x_test), np.array(y_test)

for i in range(timesteps,val_windows1.shape[0]):
    x_val.append(val_set_scaled[i-timesteps:i,0])
    y_val.append(val_set_scaled[i,0])
x_val, y_val = np.array(x_val), np.array(y_val)
```

[ ]:
```
print(x_train[0], y_train[0])
print(x_train[1], y_train[1])

print(x_test[0], y_test[0])
print(x_test[1], y_test[1])

print(x_val[0], y_val[0])
print(x_val[1], y_val[1])

print("Train Shape : ")
print(x_train.shape, y_train.shape)
x_train = x_train.reshape((x_train.shape[0], x_train.shape[1], 1))
print(x_train.shape, y_train.shape)
print("")

print("Test Shape : ")
print(x_test.shape, y_test.shape)
x_test = x_test.reshape((x_test.shape[0], x_test.shape[1], 1))
print(x_test.shape, y_test.shape)
print("")

print("Val Shape : ")
print(x_val.shape, y_val.shape)
x_val = x_val.reshape((x_val.shape[0], x_val.shape[1], 1))
print(x_val.shape, y_val.shape)
print("")

print("Train shape : ")
print(x_train.shape, y_train.shape)
idx = np.random.permutation(len(x_train))
x_train = x_train[idx]
y_train = y_train[idx]

print("Test shape : ")
print(x_test.shape, y_test.shape)
idx = np.random.permutation(len(x_test))
x_test = x_test[idx]
y_test = y_test[idx]

print("Val shape : ")
print(x_val.shape, y_val.shape)
idx = np.random.permutation(len(x_val))
x_val = x_val[idx]
y_val = y_val[idx]
```

```
[[1.05019625e-03]
 [6.22338371e-04]
```

```
 [5.83442362e-04]
 [4.47305662e-04]
 [5.83442362e-05]
 [0.00000000e+00]
 [1.94480713e-04]
 [3.50065417e-04]] 0.00025282494918691095
[[6.22338371e-04]
 [5.83442362e-04]
 [4.47305662e-04]
 [5.83442362e-05]
 [0.00000000e+00]
 [1.94480713e-04]
 [3.50065417e-04]
 [2.52824949e-04]] 0.0002042048265724736
[[0.        ]
 [0.00686363]
 [0.00305485]
 [0.00211264]
 [0.01216005]
 [0.01700027]
 [0.02834694]
 [0.02517308]] 0.01549264602078293
[[0.00686363]
 [0.00305485]
 [0.00211264]
 [0.01216005]
 [0.01700027]
 [0.02834694]
 [0.02517308]
 [0.01549265]] 0.012655977997268142
[[0.05035307]
 [0.04317979]
 [0.02027292]
 [0.03990134]
 [0.03173335]
 [0.02371945]
 [0.04969454]
 [0.04878392]] 0.05627940175349522
[[0.04317979]
 [0.02027292]
 [0.03990134]
 [0.03173335]
 [0.02371945]
 [0.04969454]
 [0.04878392]
 [0.0562794 ]] 0.06636684392482695
Train Shape :
(4594, 8, 1) (4594,)
```

```
(4594, 8, 1) (4594,)

Test Shape :
(568, 8, 1) (568,)
(568, 8, 1) (568,)

Val Shape :
(567, 8, 1) (567,)
(567, 8, 1) (567,)

Train shape :
(4594, 8, 1) (4594,)
Test shape :
(568, 8, 1) (568,)
Val shape :
(567, 8, 1) (567,)
```

1.4 baseline

[ ]:
```python
def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)

    # "ATTENTION LAYER"
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res = x + inputs

    # FEED FORWARD Part
    x = layers.LayerNormalization(epsilon=1e-6)(res)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation = "relu")(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=inputs.shape[-1], kernel_size=1)(x)

    return x + res
```

baseline dari arsitektur Transformer menggunakan satu layer Conv1D pada bagian Feed Forward dengan Activation function menggunakan ReLU dan bagian node perceptron pada output disesuaikan dengan horizon datanya. Baseline juga terdapat layer multi-Head Attention digunkan dalam lapisan encoder dan decoder untuk memproses lebih baik dari input dan memperoleh hasil yang lebih baik. Pada Baseline Model didapat hasil yang sedikit lebih tinggi daripada tuning model, itu berarti Tuning model sudah berjalan dengan baik dan mempunyai arsitektur yang lebih baik daripada baseline. karena Tuning dapat mengalahkan Baseline model Arsitektur Attention.

Model baseline Transformer untuk dataset Amazon terdiri dari lapisan sesuai gambar diatas : 1. Lapisan Embedding digunakan untuk menromalkan lapisan input menggunakan LayerNormalization dan lapisan pertama dalam model. 2. Lapisan Attention menggunakan MultiHeadAttention untuk menerima input x dan melakukan operasi attention. 3. Lapisan Add & Norm, pada lapisan ini attention layer atau nilai x ditambahkan dengan input. 4. Feed forward yang melakukan operasi feed forward pada nilai x menggunakan satu lapisan Conv1D dan fungsi aktivasi ReLU. 5. Lapisan terakhir adalah Add & Norm. Layer ini menambahkan hasil dengan layer add & norm sebelumnya. kemudian melakukan normalisasi data dengan LayerNormalization.

Referensi : *Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.

Narang, S., Chung, H. W., Tay, Y., Fedus, W., Fevry, T., Matena, M., … & Raffel, C. (2021). Do transformer modifications transfer across implementations and applications?. arXiv preprint arXiv:2102.11972. *

[ ]:
```
def build_model_x(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=0,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diatas digunakan untuk membangun model Transformer dari lapisan Encoder dan untuk membuat model transfomer yang bisa di setting untuk jumlah, ukuran dan lainnya.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
 ↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
```

```python
        pct = epoch / warmup_epochs
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```python
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```python
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```python
modelBaseline = build_model_x(
    input_shape,
    head_size=46,
    num_heads=60,
    ff_dim=55,
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

modelBaseline.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=1e-4),
    metrics=["mae"],
)

modelBaseline.summary()
```

```
Model: "model"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_1 (InputLayer)           [(None, 8, 1)]       0           []
```

```
 layer_normalization (LayerNorm  (None, 8, 1)        2
['input_1[0][0]']
 alization)

 multi_head_attention (MultiHea  (None, 8, 1)        19321
['layer_normalization[0][0]',
 dAttention)
'layer_normalization[0][0]']

 dropout (Dropout)             (None, 8, 1)        0
['multi_head_attention[0][0]']

 tf.__operators__.add (TFOpLamb  (None, 8, 1)        0
['dropout[0][0]',
 da)
'input_1[0][0]']

 layer_normalization_1 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add[0][0]']
 rmalization)

 conv1d (Conv1D)               (None, 8, 55)       110
['layer_normalization_1[0][0]']

 dropout_1 (Dropout)           (None, 8, 55)       0
['conv1d[0][0]']

 conv1d_1 (Conv1D)             (None, 8, 1)        56
['dropout_1[0][0]']

 tf.__operators__.add_1 (TFOpLa  (None, 8, 1)        0
['conv1d_1[0][0]',
 mbda)
'tf.__operators__.add[0][0]']

 layer_normalization_2 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_1[0][0]']
 rmalization)

 multi_head_attention_1 (MultiH  (None, 8, 1)        19321
['layer_normalization_2[0][0]',
 eadAttention)
'layer_normalization_2[0][0]']

 dropout_2 (Dropout)           (None, 8, 1)        0
['multi_head_attention_1[0][0]']

 tf.__operators__.add_2 (TFOpLa  (None, 8, 1)        0
```

```
['dropout_2[0][0]',
 mbda)
'tf.__operators__.add_1[0][0]']

 layer_normalization_3 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_2[0][0]']
 rmalization)

 conv1d_2 (Conv1D)              (None, 8, 55)        110
['layer_normalization_3[0][0]']

 dropout_3 (Dropout)            (None, 8, 55)        0
['conv1d_2[0][0]']

 conv1d_3 (Conv1D)              (None, 8, 1)         56
['dropout_3[0][0]']

 tf.__operators__.add_3 (TFOpLa  (None, 8, 1)        0
['conv1d_3[0][0]',
 mbda)
'tf.__operators__.add_2[0][0]']

 layer_normalization_4 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_3[0][0]']
 rmalization)

 multi_head_attention_2 (MultiH  (None, 8, 1)        19321
['layer_normalization_4[0][0]',
 eadAttention)
'layer_normalization_4[0][0]']

 dropout_4 (Dropout)            (None, 8, 1)         0
['multi_head_attention_2[0][0]']

 tf.__operators__.add_4 (TFOpLa  (None, 8, 1)        0
['dropout_4[0][0]',
 mbda)
'tf.__operators__.add_3[0][0]']

 layer_normalization_5 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_4[0][0]']
 rmalization)

 conv1d_4 (Conv1D)              (None, 8, 55)        110
['layer_normalization_5[0][0]']

 dropout_5 (Dropout)            (None, 8, 55)        0
['conv1d_4[0][0]']
```

```
 conv1d_5 (Conv1D)              (None, 8, 1)         56
['dropout_5[0][0]']

 tf.__operators__.add_5 (TFOpLa  (None, 8, 1)        0
['conv1d_5[0][0]',
 mbda)
'tf.__operators__.add_4[0][0]']

 layer_normalization_6 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_5[0][0]']
 rmalization)

 multi_head_attention_3 (MultiH  (None, 8, 1)        19321
['layer_normalization_6[0][0]',
 eadAttention)
'layer_normalization_6[0][0]']

 dropout_6 (Dropout)            (None, 8, 1)         0
['multi_head_attention_3[0][0]']

 tf.__operators__.add_6 (TFOpLa  (None, 8, 1)        0
['dropout_6[0][0]',
 mbda)
'tf.__operators__.add_5[0][0]']

 layer_normalization_7 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_6[0][0]']
 rmalization)

 conv1d_6 (Conv1D)              (None, 8, 55)        110
['layer_normalization_7[0][0]']

 dropout_7 (Dropout)            (None, 8, 55)        0
['conv1d_6[0][0]']

 conv1d_7 (Conv1D)              (None, 8, 1)         56
['dropout_7[0][0]']

 tf.__operators__.add_7 (TFOpLa  (None, 8, 1)        0
['conv1d_7[0][0]',
 mbda)
'tf.__operators__.add_6[0][0]']

 layer_normalization_8 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_7[0][0]']
 rmalization)
```

```
 multi_head_attention_4 (MultiH  (None, 8, 1)        19321
['layer_normalization_8[0][0]',
 eadAttention)
'layer_normalization_8[0][0]']

 dropout_8 (Dropout)            (None, 8, 1)         0
['multi_head_attention_4[0][0]']

 tf.__operators__.add_8 (TFOpLa  (None, 8, 1)        0
['dropout_8[0][0]',
 mbda)
'tf.__operators__.add_7[0][0]']

 layer_normalization_9 (LayerNo  (None, 8, 1)        2
['tf.__operators__.add_8[0][0]']
 rmalization)

 conv1d_8 (Conv1D)              (None, 8, 55)        110
['layer_normalization_9[0][0]']

 dropout_9 (Dropout)            (None, 8, 55)        0
['conv1d_8[0][0]']

 conv1d_9 (Conv1D)              (None, 8, 1)         56
['dropout_9[0][0]']

 tf.__operators__.add_9 (TFOpLa  (None, 8, 1)        0
['conv1d_9[0][0]',
 mbda)
'tf.__operators__.add_8[0][0]']

 global_average_pooling1d (Glob  (None, 8)           0
['tf.__operators__.add_9[0][0]']
 alAveragePooling1D)

 dense (Dense)                  (None, 256)          2304
['global_average_pooling1d[0][0]'
                                                                 ]

 dropout_10 (Dropout)           (None, 256)          0            ['dense[0][0]']

 dense_1 (Dense)                (None, 1)            257
['dropout_10[0][0]']

================================================================================
==================
Total params: 100,016
Trainable params: 100,016
```

```
Non-trainable params: 0
_________________________________________________________________________________
__________________
```

[ ]:
```
history = modelBaseline.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=5,
    batch_size=20,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/5
230/230 [==============================] - 49s 175ms/step - loss: 0.1974 - mae:
0.1974 - val_loss: 0.4160 - val_mae: 0.4160 - lr: 1.0000e-06
Epoch 2/5
230/230 [==============================] - 32s 140ms/step - loss: 0.0804 - mae:
0.0804 - val_loss: 0.0289 - val_mae: 0.0289 - lr: 3.4300e-05
Epoch 3/5
230/230 [==============================] - 30s 131ms/step - loss: 0.0367 - mae:
0.0367 - val_loss: 0.0184 - val_mae: 0.0184 - lr: 6.7600e-05
Epoch 4/5
230/230 [==============================] - 32s 137ms/step - loss: 0.0347 - mae:
0.0347 - val_loss: 0.0262 - val_mae: 0.0262 - lr: 1.0090e-04
Epoch 5/5
230/230 [==============================] - 33s 142ms/step - loss: 0.0330 - mae:
0.0330 - val_loss: 0.0241 - val_mae: 0.0241 - lr: 1.3420e-04
```

[ ]:
```
# Evaluate the model on the test set
TestLoss = modelBaseline.evaluate(x_test, y_test)
print("Test Loss:", TestLoss)
```

```
18/18 [==============================] - 1s 70ms/step - loss: 0.0429 - mae:
0.0429
Test Loss: [0.04291554540395737, 0.04291554540395737]
```

[ ]:
```
# Evaluate the model on the test set
TestLoss = modelBaseline.evaluate(x_test, y_test)
print("Test Loss:", TestLoss)
```

```
18/18 [==============================] - 3s 172ms/step - loss: 0.0374 - mae:
0.0374
Test Loss: [0.037373557686805725, 0.037373557686805725]
```

[ ]:
```
BaselineAMZN = modelBaseline.predict(x_test)
actualAMZN = np.argmax(x_test, axis=1)
```

```
baseline = np.argmax(BaselineAMZN, axis=1)
print("Ground Truth:", actualAMZN)
print("Predicted Result:", baseline)
```

Ini merupakan Ground Truth dan Predicted Result menggunakan model

Didapatkan hasil Baseline dari arsitektur Transformer, yaitu: - RMSE 60% - MAE 57% - MAPE 10%

hasil masih belum dapat dikatakan baik. sekarang kita akan membuat modifikasi dari arsitektur tersebut.

2 C. Modifikasi Aesitektur

3 Dataset Amazon

3.1 Modifikasi 1

[ ]:
```
def transformerModif(inputs, head_size, num_heads, ff_dim, dropout=0):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)

    # "ATTENTION LAYER"
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res1 = x + inputs

    # Additional Attention Layer
    x = layers.LayerNormalization(epsilon=1e-6)(res1)
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res2 = x + res1

    # FEED FORWARD Part
    x = layers.LayerNormalization(epsilon=1e-6)(res2)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation="relu")(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation="relu")(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=inputs.shape[-1], kernel_size=1)(x)

    return x + res2
```

Pada arsitektur modifikasi pertama saya menambahkan attention layer untuk agar model lebih fokus pada bagian input. pada pemprosesan attention juga memberikan representasi lebih real. kemudian dengan ini diharapkan model dapat memahami lebih baik tentang relasi titik di input.

kemudian, saya juga menambakan layer dropout dan convolution 1D di bagian feed forward.

Dropout saya gunakan untuk menghilangkan node dari layer sebelumnya, dengan tujuan mengurangi overfitting cara ini terbukti pada paper Srivastava et.al.

Referensi : Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.

[ ]:
```
def build_modif1(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=0,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformerModif(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diata digunakan untuk memodifikasi model transformer. pada modifikasi ini saya tidak memakai dropout sehingga dituliskan 0. Adapun parameter-parameter yang digunakan seperti - 'input_shape' untuk menentukkan shape input data. - 'head_size', 'num_heads', 'ff_dim' parameter-parameter ini dipakai untuk mengatur konfigurasi dari layer Transformator. - 'mlp_units' adalah besar dari setiap lapisan dense untuk mengatur jumlah neuron dari lapisan dense. - 'num_transformer_blocks' menentukan jumlah blok Transformer.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
 ↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
        return ((base_lr - initial_lr) * pct) + initial_lr
```

```
    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
model_modif1 = build_modif1(
    input_shape,
    head_size=46,
    num_heads=60,
    ff_dim=55,
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

model_modif1.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=1e-4),
    metrics=["mae"],
)

model_modif1.summary()
```

```
Model: "model_2"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_4 (InputLayer)           [(None, 8, 1)]       0           []

 layer_normalization_20 (LayerN  (None, 8, 1)        2
['input_4[0][0]']
 ormalization)
```

```
 multi_head_attention_10 (Multi  (None, 8, 1)        19321
['layer_normalization_20[0][0]',
 HeadAttention)
'layer_normalization_20[0][0]']

 dropout_22 (Dropout)           (None, 8, 1)         0
['multi_head_attention_10[0][0]']

 tf.__operators__.add_20 (TFOpL  (None, 8, 1)        0
['dropout_22[0][0]',
 ambda)
'input_4[0][0]']

 layer_normalization_21 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_20[0][0]']
 ormalization)

 multi_head_attention_11 (Multi  (None, 8, 1)        19321
['layer_normalization_21[0][0]',
 HeadAttention)
'layer_normalization_21[0][0]']

 dropout_23 (Dropout)           (None, 8, 1)         0
['multi_head_attention_11[0][0]']

 tf.__operators__.add_21 (TFOpL  (None, 8, 1)        0
['dropout_23[0][0]',
 ambda)
'tf.__operators__.add_20[0][0]']

 layer_normalization_22 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_21[0][0]']
 ormalization)

 conv1d_20 (Conv1D)             (None, 8, 55)        110
['layer_normalization_22[0][0]']

 dropout_24 (Dropout)           (None, 8, 55)        0
['conv1d_20[0][0]']

 conv1d_21 (Conv1D)             (None, 8, 55)        3080
['dropout_24[0][0]']

 dropout_25 (Dropout)           (None, 8, 55)        0
['conv1d_21[0][0]']

 conv1d_22 (Conv1D)             (None, 8, 1)         56
```

```
['dropout_25[0][0]']

 tf.__operators__.add_22 (TFOpL  (None, 8, 1)        0
['conv1d_22[0][0]',
 ambda)
'tf.__operators__.add_21[0][0]']

 layer_normalization_23 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_22[0][0]']
 ormalization)

 multi_head_attention_12 (Multi  (None, 8, 1)        19321
['layer_normalization_23[0][0]',
 HeadAttention)
'layer_normalization_23[0][0]']

 dropout_26 (Dropout)           (None, 8, 1)         0
['multi_head_attention_12[0][0]']

 tf.__operators__.add_23 (TFOpL  (None, 8, 1)        0
['dropout_26[0][0]',
 ambda)
'tf.__operators__.add_22[0][0]']

 layer_normalization_24 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_23[0][0]']
 ormalization)

 multi_head_attention_13 (Multi  (None, 8, 1)        19321
['layer_normalization_24[0][0]',
 HeadAttention)
'layer_normalization_24[0][0]']

 dropout_27 (Dropout)           (None, 8, 1)         0
['multi_head_attention_13[0][0]']

 tf.__operators__.add_24 (TFOpL  (None, 8, 1)        0
['dropout_27[0][0]',
 ambda)
'tf.__operators__.add_23[0][0]']

 layer_normalization_25 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_24[0][0]']
 ormalization)

 conv1d_23 (Conv1D)             (None, 8, 55)        110
['layer_normalization_25[0][0]']
```

```
 dropout_28 (Dropout)           (None, 8, 55)        0
['conv1d_23[0][0]']

 conv1d_24 (Conv1D)             (None, 8, 55)        3080
['dropout_28[0][0]']

 dropout_29 (Dropout)           (None, 8, 55)        0
['conv1d_24[0][0]']

 conv1d_25 (Conv1D)             (None, 8, 1)         56
['dropout_29[0][0]']

 tf.__operators__.add_25 (TFOpL  (None, 8, 1)        0
['conv1d_25[0][0]',
 ambda)
'tf.__operators__.add_24[0][0]']

 layer_normalization_26 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_25[0][0]']
 ormalization)

 multi_head_attention_14 (Multi  (None, 8, 1)        19321
['layer_normalization_26[0][0]',
 HeadAttention)
'layer_normalization_26[0][0]']

 dropout_30 (Dropout)           (None, 8, 1)         0
['multi_head_attention_14[0][0]']

 tf.__operators__.add_26 (TFOpL  (None, 8, 1)        0
['dropout_30[0][0]',
 ambda)
'tf.__operators__.add_25[0][0]']

 layer_normalization_27 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_26[0][0]']
 ormalization)

 multi_head_attention_15 (Multi  (None, 8, 1)        19321
['layer_normalization_27[0][0]',
 HeadAttention)
'layer_normalization_27[0][0]']

 dropout_31 (Dropout)           (None, 8, 1)         0
['multi_head_attention_15[0][0]']

 tf.__operators__.add_27 (TFOpL  (None, 8, 1)        0
['dropout_31[0][0]',
```

```
 ambda)
'tf.__operators__.add_26[0][0]']

 layer_normalization_28 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_27[0][0]']
 ormalization)

 conv1d_26 (Conv1D)             (None, 8, 55)       110
['layer_normalization_28[0][0]']

 dropout_32 (Dropout)           (None, 8, 55)       0
['conv1d_26[0][0]']

 conv1d_27 (Conv1D)             (None, 8, 55)       3080
['dropout_32[0][0]']

 dropout_33 (Dropout)           (None, 8, 55)       0
['conv1d_27[0][0]']

 conv1d_28 (Conv1D)             (None, 8, 1)        56
['dropout_33[0][0]']

 tf.__operators__.add_28 (TFOpL  (None, 8, 1)        0
['conv1d_28[0][0]',
 ambda)
'tf.__operators__.add_27[0][0]']

 layer_normalization_29 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_28[0][0]']
 ormalization)

 multi_head_attention_16 (Multi  (None, 8, 1)        19321
['layer_normalization_29[0][0]',
 HeadAttention)
'layer_normalization_29[0][0]']

 dropout_34 (Dropout)           (None, 8, 1)        0
['multi_head_attention_16[0][0]']

 tf.__operators__.add_29 (TFOpL  (None, 8, 1)        0
['dropout_34[0][0]',
 ambda)
'tf.__operators__.add_28[0][0]']

 layer_normalization_30 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_29[0][0]']
 ormalization)
```

```
 multi_head_attention_17 (Multi  (None, 8, 1)        19321
['layer_normalization_30[0][0]',
 HeadAttention)
'layer_normalization_30[0][0]']

 dropout_35 (Dropout)           (None, 8, 1)         0
['multi_head_attention_17[0][0]']

 tf.__operators__.add_30 (TFOpL  (None, 8, 1)        0
['dropout_35[0][0]',
 ambda)
'tf.__operators__.add_29[0][0]']

 layer_normalization_31 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_30[0][0]']
 ormalization)

 conv1d_29 (Conv1D)             (None, 8, 55)        110
['layer_normalization_31[0][0]']

 dropout_36 (Dropout)           (None, 8, 55)        0
['conv1d_29[0][0]']

 conv1d_30 (Conv1D)             (None, 8, 55)        3080
['dropout_36[0][0]']

 dropout_37 (Dropout)           (None, 8, 55)        0
['conv1d_30[0][0]']

 conv1d_31 (Conv1D)             (None, 8, 1)         56
['dropout_37[0][0]']

 tf.__operators__.add_31 (TFOpL  (None, 8, 1)        0
['conv1d_31[0][0]',
 ambda)
'tf.__operators__.add_30[0][0]']

 layer_normalization_32 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_31[0][0]']
 ormalization)

 multi_head_attention_18 (Multi  (None, 8, 1)        19321
['layer_normalization_32[0][0]',
 HeadAttention)
'layer_normalization_32[0][0]']

 dropout_38 (Dropout)           (None, 8, 1)         0
['multi_head_attention_18[0][0]']
```

```
 tf.__operators__.add_32 (TFOpL  (None, 8, 1)        0
['dropout_38[0][0]',
 ambda)
'tf.__operators__.add_31[0][0]']

 layer_normalization_33 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_32[0][0]']
 ormalization)

 multi_head_attention_19 (Multi  (None, 8, 1)        19321
['layer_normalization_33[0][0]',
 HeadAttention)
'layer_normalization_33[0][0]']

 dropout_39 (Dropout)           (None, 8, 1)        0
['multi_head_attention_19[0][0]']

 tf.__operators__.add_33 (TFOpL  (None, 8, 1)        0
['dropout_39[0][0]',
 ambda)
'tf.__operators__.add_32[0][0]']

 layer_normalization_34 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_33[0][0]']
 ormalization)

 conv1d_32 (Conv1D)             (None, 8, 55)       110
['layer_normalization_34[0][0]']

 dropout_40 (Dropout)           (None, 8, 55)       0
['conv1d_32[0][0]']

 conv1d_33 (Conv1D)             (None, 8, 55)       3080
['dropout_40[0][0]']

 dropout_41 (Dropout)           (None, 8, 55)       0
['conv1d_33[0][0]']

 conv1d_34 (Conv1D)             (None, 8, 1)        56
['dropout_41[0][0]']

 tf.__operators__.add_34 (TFOpL  (None, 8, 1)        0
['conv1d_34[0][0]',
 ambda)
'tf.__operators__.add_33[0][0]']

 global_average_pooling1d_2 (Gl  (None, 8)           0
```

```
['tf.__operators__.add_34[0][0]']
 obalAveragePooling1D)

 dense_4 (Dense)                (None, 256)          2304
['global_average_pooling1d_2[0][0
                                                                   ]']

 dropout_42 (Dropout)           (None, 256)          0
['dense_4[0][0]']

 dense_5 (Dense)                (None, 1)            257
['dropout_42[0][0]']

================================================================================
==================
Total params: 212,031
Trainable params: 212,031
Non-trainable params: 0
--------------------------------------------------------------------------------
------------------
```

[ ]:
```python
historyModif1 = model_modif1.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=5,
    batch_size=20,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/5
230/230 [==============================] - 86s 323ms/step - loss: 0.1990 - mae:
0.1990 - val_loss: 0.4248 - val_mae: 0.4248 - lr: 1.0000e-06
Epoch 2/5
230/230 [==============================] - 61s 264ms/step - loss: 0.0805 - mae:
0.0805 - val_loss: 0.0358 - val_mae: 0.0358 - lr: 3.4300e-05
Epoch 3/5
230/230 [==============================] - 64s 278ms/step - loss: 0.0417 - mae:
0.0417 - val_loss: 0.0271 - val_mae: 0.0271 - lr: 6.7600e-05
Epoch 4/5
230/230 [==============================] - 58s 254ms/step - loss: 0.0378 - mae:
0.0378 - val_loss: 0.0274 - val_mae: 0.0274 - lr: 1.0090e-04
Epoch 5/5
230/230 [==============================] - 61s 266ms/step - loss: 0.0362 - mae:
0.0362 - val_loss: 0.0251 - val_mae: 0.0251 - lr: 1.3420e-04
```

[ ]:
```
# Evaluate the model on the test set
TessLossModif1 = model_modif1.evaluate(x_test, y_test)
print("Test Loss:", TessLossModif1)
```

```
18/18 [==============================] - 3s 154ms/step - loss: 0.0453 - mae:
0.0453
Test Loss: [0.045295681804418564, 0.045295681804418564]
```

Dalam versi modifikasi ini, saya menambahkan lapisan attention tambahan setelah lapisan attention pertama. Hal ini memungkinkan interaksi yang lebih kompleks dan berpotensi meningkatkan kinerja model. Selain itu, lapisan konvolusi tambahan dengan aktivasi ReLU ditambahkan di bagian feed-forward untuk menangkap lebih jauh pola non-linear. Namun memang saya tidak memodifikasi pada bagian hyperparameter. karena saya ingin bereksperimen apakah dengan menambahkan lapisan attention, lapisan konvolusi (ReLu) di bagian feed-forward akan memberikan hasil yang lebih bagus atau tidak. ternyata dari percobaan tidak begitu mempengaruhi hasil akhirnya.

referensi :
Narang, S., Chung, H. W., Tay, Y., Fedus, W., Fevry, T., Matena, M., … & Raffel, C. (2021). Do transformer modifications transfer across implementations and applications?. arXiv preprint arXiv:2102.11972.

3.2 Modifikasi 2

[ ]:
```
def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0.1,␣
↪activation="relu"):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)

    # Attention Layer
    attention_output = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    attention_output = layers.Dropout(dropout)(attention_output)
    attention_output = layers.LayerNormalization(epsilon=1e-6)(attention_output␣
↪+ x)

    # Feed Forward Part
    ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
↪activation=activation)(attention_output)
    ff_output = layers.BatchNormalization()(ff_output)
    ff_output = layers.Dropout(dropout)(ff_output)

    ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
↪activation=activation)(ff_output)
    ff_output = layers.BatchNormalization()(ff_output)
    ff_output = layers.Dropout(dropout)(ff_output)
```

```
    transformer_output = layers.Add()([ff_output, attention_output])
    classification_output = layers.Dense(1,␣
↪activation="sigmoid")(transformer_output)

    return classification_output
```

Pada Modifikasi yang ke-2 ada beberapa modifikasi yang dilakukan pada function transfomer_encoder, yaitu:
1. attention_output = layers.LayerNormalization(epsilon=1e-6)(attention_output + x): Hasil Attention ditambahkan dengan input awal, kemudian dilakukan normalisasi menggunakan LayerNormalization. 2. melakukan normalisasi setelah layer attention dengan menggunakan LayerNormalization. 3. setelah feed-forward menambahkan residual connected. Tujuannya adalah untuk membantu memudahkan aliran gradien dan agar informasi yang terkait dapat bertahan dalam pemrosesan pembelajaran. 4. pada layer feed forward juga ditambahkan normalisasi batch pada output Conv1D menggunakan BatchNormalization. 5. pada 'transformer_output' dilakukan penjumlahan antara output dari lapisan Feed Forward dan output dari lapisan Attention.

dengan modifikasi ini diharapkan mendapatkan hasil yang lebih rendah daripada baseline.

Referensi : *Lin, T., Wang, Y., Liu, X., & Qiu, X. (2022). A survey of transformers. AI Open.*

[ ]:
```
def modif2_model(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=0,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diata digunakan untuk memodifikasi model transformer. pada modifikasi ini saya tidak memakai dropout sehingga dituliskan 0. Adapun parameter-parameter yang digunakan seperti - 'input_shape' untuk menentukkan shape input data. - 'head_size', 'num_heads', 'ff_dim' parameter-parameter ini dipakai untuk mengatur konfigurasi dari layer Transformator.

- ‘mlp__units’ adalah besar dari setiap lapisan dense untuk mengatur jumlah neuron dari lapisan dense. - ‘num__transformer__blocks’ menentukan jumlah blok Transformer.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
model_2 = modif2_model(
    input_shape,
    head_size=46, # Embedding size for attention
    num_heads=60, # Number of attention heads
    ff_dim=55, # Hidden layer size in feed forward network inside transformer
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

model_2.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=1e-4),
    metrics=["mae"],
)
model_2.summary()
```

```
Model: "model_5"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape          Param #     Connected to
```

```
================================================================================
==================
 input_19 (InputLayer)          [(None, 8, 1)]       0           []

 layer_normalization_60 (LayerN  (None, 8, 1)        2
['input_19[0][0]']
 ormalization)

 multi_head_attention_30 (Multi  (None, 8, 1)        19321
['layer_normalization_60[0][0]',
 HeadAttention)
'layer_normalization_60[0][0]']

 dropout_70 (Dropout)           (None, 8, 1)         0
['multi_head_attention_30[0][0]']

 tf.__operators__.add_50 (TFOpL  (None, 8, 1)        0
['dropout_70[0][0]',
 ambda)
'layer_normalization_60[0][0]']

 layer_normalization_61 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_50[0][0]']
 ormalization)

 conv1d_50 (Conv1D)             (None, 8, 55)        110
['layer_normalization_61[0][0]']

 batch_normalization_10 (BatchN  (None, 8, 55)       220
['conv1d_50[0][0]']
 ormalization)

 dropout_71 (Dropout)           (None, 8, 55)        0
['batch_normalization_10[0][0]']

 conv1d_51 (Conv1D)             (None, 8, 55)        3080
['dropout_71[0][0]']

 batch_normalization_11 (BatchN  (None, 8, 55)       220
['conv1d_51[0][0]']
 ormalization)

 dropout_72 (Dropout)           (None, 8, 55)        0
['batch_normalization_11[0][0]']

 add_5 (Add)                    (None, 8, 55)        0
['dropout_72[0][0]',
'layer_normalization_61[0][0]']
```

```
 dense_15 (Dense)               (None, 8, 1)         56          ['add_5[0][0]']

 layer_normalization_62 (LayerN  (None, 8, 1)        2
['dense_15[0][0]']
 ormalization)

 multi_head_attention_31 (Multi  (None, 8, 1)        19321
['layer_normalization_62[0][0]',
 HeadAttention)
'layer_normalization_62[0][0]']

 dropout_73 (Dropout)           (None, 8, 1)         0
['multi_head_attention_31[0][0]']

 tf.__operators__.add_51 (TFOpL  (None, 8, 1)        0
['dropout_73[0][0]',
 ambda)
'layer_normalization_62[0][0]']

 layer_normalization_63 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_51[0][0]']
 ormalization)

 conv1d_52 (Conv1D)             (None, 8, 55)        110
['layer_normalization_63[0][0]']

 batch_normalization_12 (BatchN  (None, 8, 55)       220
['conv1d_52[0][0]']
 ormalization)

 dropout_74 (Dropout)           (None, 8, 55)        0
['batch_normalization_12[0][0]']

 conv1d_53 (Conv1D)             (None, 8, 55)        3080
['dropout_74[0][0]']

 batch_normalization_13 (BatchN  (None, 8, 55)       220
['conv1d_53[0][0]']
 ormalization)

 dropout_75 (Dropout)           (None, 8, 55)        0
['batch_normalization_13[0][0]']

 add_6 (Add)                    (None, 8, 55)        0
['dropout_75[0][0]',
'layer_normalization_63[0][0]']
```

```
 dense_16 (Dense)               (None, 8, 1)         56          ['add_6[0][0]']

 layer_normalization_64 (LayerN  (None, 8, 1)        2
['dense_16[0][0]']
 ormalization)

 multi_head_attention_32 (Multi  (None, 8, 1)        19321
['layer_normalization_64[0][0]',
 HeadAttention)
'layer_normalization_64[0][0]']

 dropout_76 (Dropout)           (None, 8, 1)         0
['multi_head_attention_32[0][0]']

 tf.__operators__.add_52 (TFOpL  (None, 8, 1)        0
['dropout_76[0][0]',
 ambda)
'layer_normalization_64[0][0]']

 layer_normalization_65 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_52[0][0]']
 ormalization)

 conv1d_54 (Conv1D)             (None, 8, 55)        110
['layer_normalization_65[0][0]']

 batch_normalization_14 (BatchN  (None, 8, 55)       220
['conv1d_54[0][0]']
 ormalization)

 dropout_77 (Dropout)           (None, 8, 55)        0
['batch_normalization_14[0][0]']

 conv1d_55 (Conv1D)             (None, 8, 55)        3080
['dropout_77[0][0]']

 batch_normalization_15 (BatchN  (None, 8, 55)       220
['conv1d_55[0][0]']
 ormalization)

 dropout_78 (Dropout)           (None, 8, 55)        0
['batch_normalization_15[0][0]']

 add_7 (Add)                    (None, 8, 55)        0
['dropout_78[0][0]',
'layer_normalization_65[0][0]']

 dense_17 (Dense)               (None, 8, 1)         56          ['add_7[0][0]']
```

```
 layer_normalization_66 (LayerN  (None, 8, 1)        2
['dense_17[0][0]']
 ormalization)

 multi_head_attention_33 (Multi  (None, 8, 1)        19321
['layer_normalization_66[0][0]',
 HeadAttention)
'layer_normalization_66[0][0]']

 dropout_79 (Dropout)           (None, 8, 1)         0
['multi_head_attention_33[0][0]']

 tf.__operators__.add_53 (TFOpL  (None, 8, 1)        0
['dropout_79[0][0]',
 ambda)
'layer_normalization_66[0][0]']

 layer_normalization_67 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_53[0][0]']
 ormalization)

 conv1d_56 (Conv1D)             (None, 8, 55)        110
['layer_normalization_67[0][0]']

 batch_normalization_16 (BatchN  (None, 8, 55)       220
['conv1d_56[0][0]']
 ormalization)

 dropout_80 (Dropout)           (None, 8, 55)        0
['batch_normalization_16[0][0]']

 conv1d_57 (Conv1D)             (None, 8, 55)        3080
['dropout_80[0][0]']

 batch_normalization_17 (BatchN  (None, 8, 55)       220
['conv1d_57[0][0]']
 ormalization)

 dropout_81 (Dropout)           (None, 8, 55)        0
['batch_normalization_17[0][0]']

 add_8 (Add)                    (None, 8, 55)        0
['dropout_81[0][0]',
'layer_normalization_67[0][0]']

 dense_18 (Dense)               (None, 8, 1)         56          ['add_8[0][0]']
```

```
 layer_normalization_68 (LayerN  (None, 8, 1)        2
['dense_18[0][0]']
 ormalization)

 multi_head_attention_34 (Multi  (None, 8, 1)        19321
['layer_normalization_68[0][0]',
 HeadAttention)
'layer_normalization_68[0][0]']

 dropout_82 (Dropout)           (None, 8, 1)        0
['multi_head_attention_34[0][0]']

 tf.__operators__.add_54 (TFOpL  (None, 8, 1)        0
['dropout_82[0][0]',
 ambda)
'layer_normalization_68[0][0]']

 layer_normalization_69 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_54[0][0]']
 ormalization)

 conv1d_58 (Conv1D)             (None, 8, 55)       110
['layer_normalization_69[0][0]']

 batch_normalization_18 (BatchN  (None, 8, 55)       220
['conv1d_58[0][0]']
 ormalization)

 dropout_83 (Dropout)           (None, 8, 55)       0
['batch_normalization_18[0][0]']

 conv1d_59 (Conv1D)             (None, 8, 55)       3080
['dropout_83[0][0]']

 batch_normalization_19 (BatchN  (None, 8, 55)       220
['conv1d_59[0][0]']
 ormalization)

 dropout_84 (Dropout)           (None, 8, 55)       0
['batch_normalization_19[0][0]']

 add_9 (Add)                    (None, 8, 55)       0
['dropout_84[0][0]',
'layer_normalization_69[0][0]']

 dense_19 (Dense)               (None, 8, 1)        56          ['add_9[0][0]']

 global_average_pooling1d_5 (Gl  (None, 8)          0
```

```
['dense_19[0][0]']
 obalAveragePooling1D)

 dense_20 (Dense)              (None, 256)         2304
['global_average_pooling1d_5[0][0
                                                                 ]']

 dropout_85 (Dropout)          (None, 256)         0
['dense_20[0][0]']

 dense_21 (Dense)              (None, 1)           257
['dropout_85[0][0]']

================================================================================
==================
Total params: 117,616
Trainable params: 116,516
Non-trainable params: 1,100
--------------------------------------------------------------------------------
------------------
```

[ ]:
```
history_Modif2 = model_2.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=3,
    batch_size=20,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/3
230/230 [==============================] - 39s 142ms/step - loss: 0.2141 - mae:
0.2141 - val_loss: 0.3698 - val_mae: 0.3698 - lr: 1.0000e-06
Epoch 2/3
230/230 [==============================] - 33s 145ms/step - loss: 0.1929 - mae:
0.1929 - val_loss: 0.3054 - val_mae: 0.3054 - lr: 3.4300e-05
Epoch 3/3
230/230 [==============================] - 32s 139ms/step - loss: 0.1881 - mae:
0.1881 - val_loss: 0.3449 - val_mae: 0.3449 - lr: 6.7600e-05
```

- Model ditraining dengan 3 iterasi karena memakan banyak komputasi untuk itu, serta batch_size yang dipakai sebesar 30.

[ ]:
```
# Evaluate the model on the test set
Modif2 = model_2.evaluate(x_test, y_test)
print("Test Loss:", TessLossModif1)
```

```
18/18 [==============================] - 1s 66ms/step - loss: 0.4716 - mae:
```

```
0.4716
Test Loss: [0.045295681804418564, 0.045295681804418564]
```

Test Loss merupakan pengukuran seberapa jauh nilai sebenanrya dengan nilai prediksi. semakin kecil nilainya akan semakin bagus pengukurannya. untuk output diatas sudah terbilang kecil yaitu 0,4% yang artinya tes loss sendiri sudah bagus. Nilai loss pada test set digunakan untuk mengevaluasi model.

referensi :

Oh, H. W., Yoon, E. S., & Chung, M. K. (1997). An optimum set of loss models for performance prediction of centrifugal compressors. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, 211(4), 331-338.

[ ]:
```
modifikasipredictions2 = model_2.predict(x_test)
```

```
18/18 [==============================] - 2s 63ms/step
```

[ ]:
```
rmse = np.sqrt(mean_squared_error(y_test, modifikasipredictions2))
mae = mean_absolute_error(y_test, modifikasipredictions2)
mape = mean_absolute_percentage_error(y_test, modifikasipredictions2)

print("Modify Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
Modify Model:
RMSE: 0.5431465073262999
MAE: 0.510573599563848
MAPE: 0.9194575680559335
```

Pada Modifikasi yang ke-2 ada beberapa modifikasi yang dilakukan pada function transfomer_encoder, yaitu:
1. penambahan residual connected 2. melakukan normalisasi setelah layer attention 3. setelah feed-forward menambahkan residual connected. 4. terakhir melakukan normalisasi setelah feed forward, dengan tujuan menjaga konsistensi distribusi data yang ada.

dengan modifikasi ini diharapkan mendapatkan hasil yang lebih rendah daripada baseline.

Namun hasil yang didapat tidaklah sesuai harapan. modifikasi ke-2 tidak lebih rendah daripada arsitektur baseline maupun modifikasi pertama.

3.3 Modifikasi 3

[ ]:
```
def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0.1,␣
 ↪activation="relu"):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)
```

```
    # Attention Layer
    attention_output = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    attention_output = layers.Dropout(dropout)(attention_output)
    attention_output = layers.LayerNormalization(epsilon=1e-6)(attention_output␣
↪+ x)

    # Feed Forward Part
    ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
↪activation=activation)(attention_output)
    ff_output = layers.BatchNormalization()(ff_output)
    ff_output = layers.Dropout(dropout)(ff_output)

    ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
↪activation=activation)(ff_output)
    ff_output = layers.BatchNormalization()(ff_output)
    ff_output = layers.Dropout(dropout)(ff_output)

    transformer_output = layers.Add()([ff_output, attention_output])
    classification_output = layers.Dense(1,␣
↪activation="sigmoid")(transformer_output)

    return classification_output
```

Pada Arsitektur yang ke-3 menambahkan beberapa layer tambahan, seperti 1. lapisan attention dan juga output attention ditambahkan dengan input asli 'attention_output + x'. 2. selain itu residual connection digunakan dalam penjumlahan akhir dengan output dari bagian feed-forward menggunakan 'layer.Add(). 3. saya juga menambahkan normalisasi batch dengan tujuan mengurangi ketergantungan pada distribusi input dan mempercepat proses pelatihan 4. pada bagian layer feed forward menambahkan 1 dropout dan juga menambahkan 1 transformasi linear dan activation sigmoid.

dalam paper Svozil et.al., dibahas bahwa beberapa masalah dalam memilih model ataupun tambahan arsitektur adalah berapa banyak layer yang ada. dalam jaringan saraf sendiri jaringan dengan fungsi aktivasi merupakan jumlah unit yang besar.

referensi : Svozil, D., Kvasnicka, V., & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. Chemometrics and intelligent laboratory systems, 39(1), 43-62.

[ ]:
```
def modify_3(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
```

```
    dropout=0,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diata digunakan untuk memodifikasi model transformer. pada modifikasi ini saya tidak memakai dropout sehingga ditulis 0. alasan saya menghilangkan dropout adalah dalam paper yang saya baca yaitu Bell et.al., dengan memakainya dropout bisa mengalami bias itulah salah satu alasan say tidak mengambahkan bias karena ingin berekxperimen apakah akan mendapatkan evaluasi model yang lebih rendah. Adapun parameter-parameter yang digunakan seperti - 'input_shape' untuk menentukkan shape input data. - 'head_size', 'num_heads', 'ff_dim' parameter-parameter ini dipakai untuk mengatur konfigurasi dari layer Transformator. - 'mlp_units' adalah besar dari setiap lapisan dense untuk mengatur jumlah neuron dari lapisan dense. - 'num_transformer_blocks' menentukan jumlah blok Transformer.

Bell, M. L., Kenward, M. G., Fairclough, D. L., & Horton, N. J. (2013). Differential dropout and bias in randomised controlled trials: when it matters and when it may not. Bmj, 346.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,
 ↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
model3 = modify_3(
    input_shape,
    head_size=46,
    num_heads=60,
    ff_dim=55,
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

model3.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=0.001),
    metrics=["mae"],
)

model3.summary()
```

```
Model: "model_8"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_22 (InputLayer)          [(None, 8, 1)]       0           []

 layer_normalization_90 (LayerN  (None, 8, 1)        2
['input_22[0][0]']
 ormalization)

 multi_head_attention_45 (Multi  (None, 8, 1)        19321
['layer_normalization_90[0][0]',
 HeadAttention)
'layer_normalization_90[0][0]']

 dropout_118 (Dropout)          (None, 8, 1)         0
['multi_head_attention_45[0][0]']

 tf.__operators__.add_65 (TFOpL  (None, 8, 1)        0
['dropout_118[0][0]',
 ambda)
'layer_normalization_90[0][0]']
```

```
 layer_normalization_91 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_65[0][0]']
 ormalization)

 conv1d_80 (Conv1D)             (None, 8, 55)        110
['layer_normalization_91[0][0]']

 batch_normalization_40 (BatchN  (None, 8, 55)       220
['conv1d_80[0][0]']
 ormalization)

 dropout_119 (Dropout)          (None, 8, 55)        0
['batch_normalization_40[0][0]']

 conv1d_81 (Conv1D)             (None, 8, 55)        3080
['dropout_119[0][0]']

 batch_normalization_41 (BatchN  (None, 8, 55)       220
['conv1d_81[0][0]']
 ormalization)

 dropout_120 (Dropout)          (None, 8, 55)        0
['batch_normalization_41[0][0]']

 add_20 (Add)                   (None, 8, 55)        0
['dropout_120[0][0]',
'layer_normalization_91[0][0]']

 dense_36 (Dense)               (None, 8, 1)         56
['add_20[0][0]']

 layer_normalization_92 (LayerN  (None, 8, 1)        2
['dense_36[0][0]']
 ormalization)

 multi_head_attention_46 (Multi  (None, 8, 1)        19321
['layer_normalization_92[0][0]',
 HeadAttention)
'layer_normalization_92[0][0]']

 dropout_121 (Dropout)          (None, 8, 1)         0
['multi_head_attention_46[0][0]']

 tf.__operators__.add_66 (TFOpL  (None, 8, 1)        0
['dropout_121[0][0]',
 ambda)
'layer_normalization_92[0][0]']
```

```
 layer_normalization_93 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_66[0][0]']
 ormalization)

 conv1d_82 (Conv1D)             (None, 8, 55)       110
['layer_normalization_93[0][0]']

 batch_normalization_42 (BatchN  (None, 8, 55)       220
['conv1d_82[0][0]']
 ormalization)

 dropout_122 (Dropout)          (None, 8, 55)       0
['batch_normalization_42[0][0]']

 conv1d_83 (Conv1D)             (None, 8, 55)       3080
['dropout_122[0][0]']

 batch_normalization_43 (BatchN  (None, 8, 55)       220
['conv1d_83[0][0]']
 ormalization)

 dropout_123 (Dropout)          (None, 8, 55)       0
['batch_normalization_43[0][0]']

 add_21 (Add)                   (None, 8, 55)       0
['dropout_123[0][0]',
'layer_normalization_93[0][0]']

 dense_37 (Dense)               (None, 8, 1)        56
['add_21[0][0]']

 layer_normalization_94 (LayerN  (None, 8, 1)        2
['dense_37[0][0]']
 ormalization)

 multi_head_attention_47 (Multi  (None, 8, 1)        19321
['layer_normalization_94[0][0]',
 HeadAttention)
'layer_normalization_94[0][0]']

 dropout_124 (Dropout)          (None, 8, 1)        0
['multi_head_attention_47[0][0]']

 tf.__operators__.add_67 (TFOpL  (None, 8, 1)        0
['dropout_124[0][0]',
 ambda)
'layer_normalization_94[0][0]']
```

```
 layer_normalization_95 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_67[0][0]']
 ormalization)

 conv1d_84 (Conv1D)             (None, 8, 55)        110
['layer_normalization_95[0][0]']

 batch_normalization_44 (BatchN  (None, 8, 55)       220
['conv1d_84[0][0]']
 ormalization)

 dropout_125 (Dropout)          (None, 8, 55)        0
['batch_normalization_44[0][0]']

 conv1d_85 (Conv1D)             (None, 8, 55)        3080
['dropout_125[0][0]']

 batch_normalization_45 (BatchN  (None, 8, 55)       220
['conv1d_85[0][0]']
 ormalization)

 dropout_126 (Dropout)          (None, 8, 55)        0
['batch_normalization_45[0][0]']

 add_22 (Add)                   (None, 8, 55)        0
['dropout_126[0][0]',
'layer_normalization_95[0][0]']

 dense_38 (Dense)               (None, 8, 1)         56
['add_22[0][0]']

 layer_normalization_96 (LayerN  (None, 8, 1)        2
['dense_38[0][0]']
 ormalization)

 multi_head_attention_48 (Multi  (None, 8, 1)        19321
['layer_normalization_96[0][0]',
 HeadAttention)
'layer_normalization_96[0][0]']

 dropout_127 (Dropout)          (None, 8, 1)         0
['multi_head_attention_48[0][0]']

 tf.__operators__.add_68 (TFOpL  (None, 8, 1)        0
['dropout_127[0][0]',
 ambda)
'layer_normalization_96[0][0]']
```

```
 layer_normalization_97 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_68[0][0]']
 ormalization)

 conv1d_86 (Conv1D)             (None, 8, 55)        110
['layer_normalization_97[0][0]']

 batch_normalization_46 (BatchN  (None, 8, 55)       220
['conv1d_86[0][0]']
 ormalization)

 dropout_128 (Dropout)          (None, 8, 55)        0
['batch_normalization_46[0][0]']

 conv1d_87 (Conv1D)             (None, 8, 55)        3080
['dropout_128[0][0]']

 batch_normalization_47 (BatchN  (None, 8, 55)       220
['conv1d_87[0][0]']
 ormalization)

 dropout_129 (Dropout)          (None, 8, 55)        0
['batch_normalization_47[0][0]']

 add_23 (Add)                   (None, 8, 55)        0
['dropout_129[0][0]',
'layer_normalization_97[0][0]']

 dense_39 (Dense)               (None, 8, 1)         56
['add_23[0][0]']

 layer_normalization_98 (LayerN  (None, 8, 1)        2
['dense_39[0][0]']
 ormalization)

 multi_head_attention_49 (Multi  (None, 8, 1)        19321
['layer_normalization_98[0][0]',
 HeadAttention)
'layer_normalization_98[0][0]']

 dropout_130 (Dropout)          (None, 8, 1)         0
['multi_head_attention_49[0][0]']

 tf.__operators__.add_69 (TFOpL  (None, 8, 1)        0
['dropout_130[0][0]',
 ambda)
'layer_normalization_98[0][0]']
```

```
 layer_normalization_99 (LayerN  (None, 8, 1)        2
['tf.__operators__.add_69[0][0]']
 ormalization)

 conv1d_88 (Conv1D)             (None, 8, 55)        110
['layer_normalization_99[0][0]']

 batch_normalization_48 (BatchN  (None, 8, 55)       220
['conv1d_88[0][0]']
 ormalization)

 dropout_131 (Dropout)          (None, 8, 55)        0
['batch_normalization_48[0][0]']

 conv1d_89 (Conv1D)             (None, 8, 55)        3080
['dropout_131[0][0]']

 batch_normalization_49 (BatchN  (None, 8, 55)       220
['conv1d_89[0][0]']
 ormalization)

 dropout_132 (Dropout)          (None, 8, 55)        0
['batch_normalization_49[0][0]']

 add_24 (Add)                   (None, 8, 55)        0
['dropout_132[0][0]',
'layer_normalization_99[0][0]']

 dense_40 (Dense)               (None, 8, 1)         56
['add_24[0][0]']

 global_average_pooling1d_8 (Gl  (None, 8)           0
['dense_40[0][0]']
 obalAveragePooling1D)

 dense_41 (Dense)               (None, 256)          2304
['global_average_pooling1d_8[0][0
                                                                 ]']

 dropout_133 (Dropout)          (None, 256)          0
['dense_41[0][0]']

 dense_42 (Dense)               (None, 1)            257
['dropout_133[0][0]']

================================================================================
==================
```

```
Total params: 117,616
Trainable params: 116,516
Non-trainable params: 1,100
________________________________________________________________________________
__________________
```

[ ]:
```
history_3 = model3.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=2,
    batch_size=32,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/2
144/144 [==============================] - 32s 220ms/step - loss: 0.3105 - mae:
0.3105 - val_loss: 0.5236 - val_mae: 0.5236 - lr: 1.0000e-06
Epoch 2/2
144/144 [==============================] - 32s 224ms/step - loss: 0.2146 - mae:
0.2146 - val_loss: 0.3034 - val_mae: 0.3034 - lr: 3.4300e-05
```

[ ]:
```
# Evaluate the model on the test set
TestLoss = model3.evaluate(x_test, y_test)
print("Test Loss:", TestLoss)
```

```
18/18 [==============================] - 5s 74ms/step - loss: 0.6788 - mae:
0.6788
Test Loss: [0.678811252117157, 0.678811252117157]
```

Test Loss merupakan pengukuran seberapa jauh nilai sebenanrya dengan nilai prediksi. semakin kecil nilainya akan semakin bagus pengukurannya. untuk output diatas sudah terbilang kecil yaitu 6.7%% lebih besar daripada modifikasi ke-2 yang artinya pada model modifikasi ke-3 masih dibawah modifikasi ke-2.

referensi :

Oh, H. W., Yoon, E. S., & Chung, M. K. (1997). An optimum set of loss models for performance prediction of centrifugal compressors. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, 211(4), 331-338.

[ ]:
```
# Matric Eval dalam test set

# Predict on the test set
baseline_predictions1 = model3.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, baseline_predictions1))
```

```
baseline_mae = mean_absolute_error(y_test, baseline_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, baseline_predictions1)

print("Baseline Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
18/18 [==============================] - 3s 111ms/step
Baseline Model:
RMSE: 0.457036004688076
MAE: 0.4220837581296884
MAPE: 0.857550506092802
```

Pada Arsitektur yang ke-3 menambahkan beberapa layer tambahan, seperti 1. lapisan attention dan juga output attention ditambahkan dengan input asli 'attention_output + x'. 2. selain itu residual connection digunakan dalam penjumlahan akhir dengan output dari bagian feed-forward menggunakan 'layer.Add(). 3. saya juga menambahkan normalisasi batch dengan tujuan mengurangi ketergantungan pada distribusi input dan mempercepat proses pelatihan 4. pada bagian layer feed forward menambahkan 1 dropout dan juga menambahkan 1 transformasi linear dan activation sigmoid.

Hasil dari Arsitektur modify ke-3 sebesar - RMSE 45% - MAE 42% - MAPE 85% tidak cukup baik daripada sebelumnya. sehingga dalam modifikasi untuk data Amazon yang paling bagu adalah modifikasi pertama. dengan arsitektur Pada arsitektur modifikasi pertama saya menambahkan attention layer untuk agar model lebih fokus pada bagian input. pada pemprosesan attention juga memberikan representasi lebih real. kemudian dengan ini diharapkan model dapat memahami lebih baik tentang relasi titik di inpput.

kemudian, saya juga menambakan layer dropout dan convolution 1D di bagian feed forward.

Dropout saya gunakan untuk menghilangkan node dari layer sebelumnya, dengan tujuan mengurangi overfitting cara ini terbukti pada paper Srivastava et.al.

Referensi : Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.

[ ]:
```
# Matric Eval dalam test set

# Predict on the test set
baseline_predictions1 = modelBaseline.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, baseline_predictions1))
baseline_mae = mean_absolute_error(y_test, baseline_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, baseline_predictions1)

print("Baseline Model:")
print("RMSE:", baseline_rmse)
```

```
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
18/18 [==============================] - 2s 66ms/step
Baseline Model:
RMSE: 0.05925470901130825
MAE: 0.042915541423656285
MAPE: 0.10178843711006433
```

4 Dataset CISCO

[ ]:
```
dfAMZN = pd.read_csv("CSCO.csv",
                 parse_dates=["Date"],
                 index_col=["Date"])
dfAMZN.head()
```

[ ]:

Date	Open	High	Low	Close	Adj Close	Volume
1990-02-16	0.0	0.079861	0.073785	0.077257	0.059806	940636800
1990-02-20	0.0	0.079861	0.074653	0.079861	0.061822	151862400
1990-02-21	0.0	0.078993	0.075521	0.078125	0.060478	70531200
1990-02-22	0.0	0.081597	0.078993	0.078993	0.061150	45216000
1990-02-23	0.0	0.079861	0.078125	0.078559	0.060814	44697600

[ ]:
```
# Buat variabel baru yang berisi
pricesCisco = pd.DataFrame(dfAMZN["Close"]).rename(columns={"Close": "Price"})
pricesCisco.head()
```

[ ]:

Date	Price
1990-02-16	0.077257
1990-02-20	0.079861
1990-02-21	0.078125
1990-02-22	0.078993
1990-02-23	0.078559

[ ]:
```
WINDOW_SIZE = 5
HORIZON = 1
```

- Window size : rentang dari senin hingga jumat
- Horizon hanya hari senin saja

[ ]:
```
# Get AMZN date array
timesteps = pricesCisco.index.to_numpy()
prices1 = pricesCisco["Price"].to_numpy()

timesteps[:10], prices1[:10]
```

```
[ ]: (array(['1990-02-16T00:00:00.000000000', '1990-02-20T00:00:00.000000000',
             '1990-02-21T00:00:00.000000000', '1990-02-22T00:00:00.000000000',
             '1990-02-23T00:00:00.000000000', '1990-02-26T00:00:00.000000000',
             '1990-02-27T00:00:00.000000000', '1990-02-28T00:00:00.000000000',
             '1990-03-01T00:00:00.000000000', '1990-03-02T00:00:00.000000000'],
            dtype='datetime64[ns]'),
      array([0.07725695, 0.07986111, 0.078125  , 0.07899305, 0.07855903,
             0.07638889, 0.078125  , 0.08072916, 0.07986111, 0.08072916]))
```

```
[ ]: def get_labelled_windows(x, horizon=1):
       return x[:, :-horizon], x[:, -horizon:]

     test_window, test_label = get_labelled_windows(tf.expand_dims(tf.range(6)+1,␣
      ↪axis=0), horizon=HORIZON)
     print(f"Window: {tf.squeeze(test_window).numpy()} -> Label: {tf.
      ↪squeeze(test_label).numpy()}")
```

```
Window: [1 2 3 4 5] -> Label: 6
```

```
[ ]: def make_windows(x, window_size=5, horizon=1):

       window_step = np.expand_dims(np.arange(window_size+horizon), axis=0)
       window_indexes = window_step + np.expand_dims(np.
      ↪arange(len(x)-(window_size+horizon-1)), axis=0).T
       windowed_array = x[window_indexes]
       windows, labels = get_labelled_windows(windowed_array, horizon=horizon)

       return windows, labels
```

```
[ ]: full_windows1, full_labels1 = make_windows(prices1, window_size=WINDOW_SIZE,␣
      ↪horizon=HORIZON)
     len(full_windows1), len(full_labels1)
```

```
[ ]: (7584, 7584)
```

4.1 Create split train tes val

```
[ ]: def make_train_val_test_splits(windows, labels, val_split=0.1, test_split=0.1):

         total_size = len(windows)
         train_size = int(total_size * 0.8)
         val_size = int(total_size * 0.1)
         test_size = total_size - train_size - val_size

         train_windows = windows[:train_size]
         train_labels = labels[:train_size]
         val_windows = windows[train_size:train_size+val_size]
```

```
    val_labels = labels[train_size:train_size+val_size]
    test_windows = windows[train_size+val_size:]
    test_labels = labels[train_size+val_size:]

    return train_windows, val_windows, test_windows, train_labels, val_labels,␣
 ↪test_labels
```

[ ]:
```
train_windows1, val_windows1, test_windows1, train_labels1, val_labels1,␣
 ↪test_labels1= make_train_val_test_splits(full_windows1, full_labels1)
print("Train set length:", len(train_windows1))
print("Validation set length:", len(val_windows1))
print("Test set length:", len(test_windows1))
print("Train labels length:", len(train_labels1))
print("Validation labels length:", len(val_labels1))
print("Test labels length:", len(test_labels1))
```

```
Train set length: 6067
Validation set length: 758
Test set length: 759
Train labels length: 6067
Validation labels length: 758
Test labels length: 759
```

[ ]:
```
print('Train set: {} baris x {} kolom'.format(train_windows1.shape[0],␣
 ↪train_windows1.shape[1]))
print('Test set: {} baris x {} kolom'.format(test_windows1.shape[0],␣
 ↪test_windows1.shape[1]))
print('Validation set: {} baris x {} kolom'.format(val_windows1.shape[0],␣
 ↪val_windows1.shape[1]))
```

```
Train set: 6067 baris x 5 kolom
Test set: 759 baris x 5 kolom
Validation set: 758 baris x 5 kolom
```

[ ]:
```
# Scaling training set
scaled = MinMaxScaler(feature_range=(0,1))
training_set_scaled = scaled.fit_transform(train_windows1)
test_set_scaled = scaled.fit_transform(test_windows1)
val_set_scaled = scaled.fit_transform(val_windows1)
```

[ ]:
```
timesteps = 8

x_train = []
y_train = []

x_test = []
y_test = []
```

```
x_val = []
y_val = []

for i in range(timesteps,train_windows1.shape[0]):
    x_train.append(training_set_scaled[i-timesteps:i,0])
    y_train.append(training_set_scaled[i,0])
x_train, y_train = np.array(x_train), np.array(y_train)

for i in range(timesteps,test_windows1.shape[0]):
    x_test.append(test_set_scaled[i-timesteps:i,0])
    y_test.append(test_set_scaled[i,0])
x_test, y_test = np.array(x_test), np.array(y_test)

for i in range(timesteps,val_windows1.shape[0]):
    x_val.append(val_set_scaled[i-timesteps:i,0])
    y_val.append(val_set_scaled[i,0])
x_val, y_val = np.array(x_val), np.array(y_val)


print(x_train[0], y_train[0])
print(x_train[1], y_train[1])

print(x_test[0], y_test[0])
print(x_test[1], y_test[1])

print(x_val[0], y_val[0])
print(x_val[1], y_val[1])
```

```
[7.59631864e-05 1.08518771e-04 8.68150170e-05 9.76668475e-05
 9.22409323e-05 6.51112627e-05 8.68150170e-05 1.19370602e-04]
0.0001085187712203757
[1.08518771e-04 8.68150170e-05 9.76668475e-05 9.22409323e-05
 6.51112627e-05 8.68150170e-05 1.19370602e-04 1.08518771e-04]
0.0001193706017712313
[0.12680637 0.13475426 0.12969648 0.13403184 0.13078038 0.13186416
 0.12174859 0.12174859] 0.12391613426164061
[0.13475426 0.12969648 0.13403184 0.13078038 0.13186416 0.12174859
 0.12174859 0.12391613] 0.11596825063664995
[0.00466196 0.00466196 0.02020204 0.00543898 0.         0.0598291
 0.05827506 0.0349651 ] 0.059052081737083206
[0.00466196 0.02020204 0.00543898 0.         0.0598291  0.05827506
 0.0349651  0.05905208] 0.0660451011665506
```

[ ]:
```
print("Train Shape : ")
print(x_train.shape, y_train.shape)
x_train = x_train.reshape((x_train.shape[0], x_train.shape[1], 1))
```

```
print(x_train.shape, y_train.shape)
print("")

print("Test Shape : ")
print(x_test.shape, y_test.shape)
x_test = x_test.reshape((x_test.shape[0], x_test.shape[1], 1))
print(x_test.shape, y_test.shape)
print("")

print("Val Shape : ")
print(x_val.shape, y_val.shape)
x_val = x_val.reshape((x_val.shape[0], x_val.shape[1], 1))
print(x_val.shape, y_val.shape)
print("")



print("Train shape : ")
print(x_train.shape, y_train.shape)
idx = np.random.permutation(len(x_train))
x_train = x_train[idx]
y_train = y_train[idx]

print("Test shape : ")
print(x_test.shape, y_test.shape)
idx = np.random.permutation(len(x_test))
x_test = x_test[idx]
y_test = y_test[idx]

print("Val shape : ")
print(x_val.shape, y_val.shape)
idx = np.random.permutation(len(x_val))
x_val = x_val[idx]
y_val = y_val[idx]
```

```
Train Shape :
(6059, 8, 1) (6059,)
(6059, 8, 1) (6059,)

Test Shape :
(751, 8, 1) (751,)
(751, 8, 1) (751,)

Val Shape :
(750, 8, 1) (750,)
(750, 8, 1) (750,)
```

```
Train shape :
(6059, 8, 1) (6059,)
Test shape :
(751, 8, 1) (751,)
Val shape :
(750, 8, 1) (750,)
```

4.2 Baseline Arsitektur Cisco

[ ]:
```python
def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)

    # "ATTENTION LAYER"
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res = x + inputs

    # FEED FORWARD Part
    x = layers.LayerNormalization(epsilon=1e-6)(res)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation = "relu")(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=inputs.shape[-1], kernel_size=1)(x)

    return x + res
```

Model baseline Transformer untuk dataset Amazon terdiri dari lapisan sesuai gambar diatas : 1. Lapisan Embedding digunakan untuk menromalkan lapisan input menggunakan LayerNormalization dan lapisan pertama dalam model. 2. Lapisan Attention menggunakan MultiHeadAttention untuk menerima input x dan melakukan operasi attention. 3. Lapisan Add & Norm, pada lapisan ini attention layer atau nilai x ditambahkan dengan input. 4. Feed forward yang melakukan operasi feed forward pada nilai x menggunakan satu lapisan Conv1D dan fungsi aktivasi ReLU. 5. Lapisan terakhir adalah Add & Norm. Layer ini menambahkan hasil dengan layer add & norm sebelumnya. kemudian melakukan normalisasi data dengan LayerNormalization.

[ ]:
```python
def build_model_x(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=0,
    mlp_dropout=0,
```

```
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
 ↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
modelBaseline = build_model_x(
    input_shape,
    head_size=46,
    num_heads=60,
    ff_dim=55,
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)
```

```python
modelBaseline.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=1e-4),
    metrics=["mae"],
)

modelBaseline.summary()
```

```
Model: "model_9"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_23 (InputLayer)          [(None, 8, 1)]       0           []

 layer_normalization_100 (Layer  (None, 8, 1)        2
['input_23[0][0]']
 Normalization)

 multi_head_attention_50 (Multi  (None, 8, 1)        19321
['layer_normalization_100[0][0]',
 HeadAttention)
'layer_normalization_100[0][0]']

 dropout_134 (Dropout)          (None, 8, 1)         0
['multi_head_attention_50[0][0]']

 tf.__operators__.add_70 (TFOpL  (None, 8, 1)        0
['dropout_134[0][0]',
 ambda)
'input_23[0][0]']

 layer_normalization_101 (Layer  (None, 8, 1)        2
['tf.__operators__.add_70[0][0]']
 Normalization)

 conv1d_90 (Conv1D)             (None, 8, 55)        110
['layer_normalization_101[0][0]']

 dropout_135 (Dropout)          (None, 8, 55)        0
['conv1d_90[0][0]']

 conv1d_91 (Conv1D)             (None, 8, 1)         56
['dropout_135[0][0]']
```

```
 tf.__operators__.add_71 (TFOpL  (None, 8, 1)        0
['conv1d_91[0][0]',
 ambda)
'tf.__operators__.add_70[0][0]']

 layer_normalization_102 (Layer  (None, 8, 1)        2
['tf.__operators__.add_71[0][0]']
 Normalization)

 multi_head_attention_51 (Multi  (None, 8, 1)        19321
['layer_normalization_102[0][0]',
 HeadAttention)
'layer_normalization_102[0][0]']

 dropout_136 (Dropout)          (None, 8, 1)         0
['multi_head_attention_51[0][0]']

 tf.__operators__.add_72 (TFOpL  (None, 8, 1)        0
['dropout_136[0][0]',
 ambda)
'tf.__operators__.add_71[0][0]']

 layer_normalization_103 (Layer  (None, 8, 1)        2
['tf.__operators__.add_72[0][0]']
 Normalization)

 conv1d_92 (Conv1D)             (None, 8, 55)        110
['layer_normalization_103[0][0]']

 dropout_137 (Dropout)          (None, 8, 55)        0
['conv1d_92[0][0]']

 conv1d_93 (Conv1D)             (None, 8, 1)         56
['dropout_137[0][0]']

 tf.__operators__.add_73 (TFOpL  (None, 8, 1)        0
['conv1d_93[0][0]',
 ambda)
'tf.__operators__.add_72[0][0]']

 layer_normalization_104 (Layer  (None, 8, 1)        2
['tf.__operators__.add_73[0][0]']
 Normalization)

 multi_head_attention_52 (Multi  (None, 8, 1)        19321
['layer_normalization_104[0][0]',
 HeadAttention)
'layer_normalization_104[0][0]']
```

```
 dropout_138 (Dropout)          (None, 8, 1)        0
['multi_head_attention_52[0][0]']

 tf.__operators__.add_74 (TFOpL  (None, 8, 1)        0
['dropout_138[0][0]',
 ambda)
'tf.__operators__.add_73[0][0]']

 layer_normalization_105 (Layer  (None, 8, 1)        2
['tf.__operators__.add_74[0][0]']
 Normalization)

 conv1d_94 (Conv1D)             (None, 8, 55)       110
['layer_normalization_105[0][0]']

 dropout_139 (Dropout)          (None, 8, 55)       0
['conv1d_94[0][0]']

 conv1d_95 (Conv1D)             (None, 8, 1)        56
['dropout_139[0][0]']

 tf.__operators__.add_75 (TFOpL  (None, 8, 1)        0
['conv1d_95[0][0]',
 ambda)
'tf.__operators__.add_74[0][0]']

 layer_normalization_106 (Layer  (None, 8, 1)        2
['tf.__operators__.add_75[0][0]']
 Normalization)

 multi_head_attention_53 (Multi  (None, 8, 1)        19321
['layer_normalization_106[0][0]',
 HeadAttention)
'layer_normalization_106[0][0]']

 dropout_140 (Dropout)          (None, 8, 1)        0
['multi_head_attention_53[0][0]']

 tf.__operators__.add_76 (TFOpL  (None, 8, 1)        0
['dropout_140[0][0]',
 ambda)
'tf.__operators__.add_75[0][0]']

 layer_normalization_107 (Layer  (None, 8, 1)        2
['tf.__operators__.add_76[0][0]']
 Normalization)
```

```
 conv1d_96 (Conv1D)             (None, 8, 55)        110
['layer_normalization_107[0][0]']

 dropout_141 (Dropout)          (None, 8, 55)        0
['conv1d_96[0][0]']

 conv1d_97 (Conv1D)             (None, 8, 1)         56
['dropout_141[0][0]']

 tf.__operators__.add_77 (TFOpL  (None, 8, 1)        0
['conv1d_97[0][0]',
 ambda)
'tf.__operators__.add_76[0][0]']

 layer_normalization_108 (Layer  (None, 8, 1)        2
['tf.__operators__.add_77[0][0]']
 Normalization)

 multi_head_attention_54 (Multi  (None, 8, 1)        19321
['layer_normalization_108[0][0]',
 HeadAttention)
'layer_normalization_108[0][0]']

 dropout_142 (Dropout)          (None, 8, 1)         0
['multi_head_attention_54[0][0]']

 tf.__operators__.add_78 (TFOpL  (None, 8, 1)        0
['dropout_142[0][0]',
 ambda)
'tf.__operators__.add_77[0][0]']

 layer_normalization_109 (Layer  (None, 8, 1)        2
['tf.__operators__.add_78[0][0]']
 Normalization)

 conv1d_98 (Conv1D)             (None, 8, 55)        110
['layer_normalization_109[0][0]']

 dropout_143 (Dropout)          (None, 8, 55)        0
['conv1d_98[0][0]']

 conv1d_99 (Conv1D)             (None, 8, 1)         56
['dropout_143[0][0]']

 tf.__operators__.add_79 (TFOpL  (None, 8, 1)        0
['conv1d_99[0][0]',
 ambda)
'tf.__operators__.add_78[0][0]']
```

```
 global_average_pooling1d_9 (Gl  (None, 8)           0
['tf.__operators__.add_79[0][0]']
 obalAveragePooling1D)

 dense_43 (Dense)               (None, 256)          2304
['global_average_pooling1d_9[0][0
                                                                 ]']

 dropout_144 (Dropout)          (None, 256)          0
['dense_43[0][0]']

 dense_44 (Dense)               (None, 1)            257
['dropout_144[0][0]']

================================================================================
==================
Total params: 100,016
Trainable params: 100,016
Non-trainable params: 0
--------------------------------------------------------------------------------
------------------
```

[ ]:
```
history = modelBaseline.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=5,
    batch_size=20,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/5
303/303 [==============================] - 121s 371ms/step - loss: 0.1049 - mae:
0.1049 - val_loss: 0.2263 - val_mae: 0.2263 - lr: 1.0000e-06
Epoch 2/5
303/303 [==============================] - 44s 146ms/step - loss: 0.0520 - mae:
0.0520 - val_loss: 0.0467 - val_mae: 0.0467 - lr: 3.4300e-05
Epoch 3/5
303/303 [==============================] - 43s 141ms/step - loss: 0.0390 - mae:
0.0390 - val_loss: 0.0523 - val_mae: 0.0523 - lr: 6.7600e-05
Epoch 4/5
303/303 [==============================] - 45s 150ms/step - loss: 0.0352 - mae:
0.0352 - val_loss: 0.0515 - val_mae: 0.0515 - lr: 1.0090e-04
Epoch 5/5
303/303 [==============================] - 44s 146ms/step - loss: 0.0325 - mae:
0.0325 - val_loss: 0.0393 - val_mae: 0.0393 - lr: 1.3420e-04
```

[ ]:
```python
# Evaluate the model on the test set
TestLoss = modelBaseline.evaluate(x_test, y_test)
print("Test Loss:", TestLoss)
```

```
24/24 [==============================] - 2s 67ms/step - loss: 0.0352 - mae:
0.0352
Test Loss: [0.0352291576564312, 0.0352291576564312]
```

Test Loss merupakan pengukuran seberapa jauh nilai sebenanrya dengan nilai prediksi. semakin kecil nilainya akan semakin bagus pengukurannya. untuk output diatas sudah terbilang kecil yaitu 0,3% yang artinya tes loss sendiri sudah bagus. Nilai loss pada test set digunakan untuk mengevaluasi model.

referensi :

Oh, H. W., Yoon, E. S., & Chung, M. K. (1997). An optimum set of loss models for performance prediction of centrifugal compressors. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, 211(4), 331-338.

4.3 Modifikasi ke-1

[ ]:
```python
def transformerModif(inputs, head_size, num_heads, ff_dim, dropout=0):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)

    # "ATTENTION LAYER"
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res1 = x + inputs

    # Additional Attention Layer
    x = layers.LayerNormalization(epsilon=1e-6)(res1)
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res2 = x + res1

    # FEED FORWARD Part
    x = layers.LayerNormalization(epsilon=1e-6)(res2)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation="relu")(x)
    #x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation="relu")(x)
    #x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=inputs.shape[-1], kernel_size=1)(x)
```

```
    return x + res2
```

Pada percobaan modifikasi data Cisco yang pertama, perubahan ini yang saya lakukan :
- saya membuang dropout pada feed forward - menambahkan attention layer berupa: 1. dengan normalisasi dan embedding layer untuk menjaga konsistensi distribusi data. 2. menambahkan Attention Layer, kemudian output tersebut dimasukkan ke dropout layer. 3. saya juga menambahkan residual connection menggunakan `+` untuk menyimpan data asli dan membantu aliran gradien selama proses pembelajaran.

referensi Narang, S., Chung, H. W., Tay, Y., Fedus, W., Fevry, T., Matena, M., ... & Raffel, C. (2021). Do transformer modifications transfer across implementations and applications?. arXiv preprint arXiv:2102.11972.

[ ]:
```
def build_modif1(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=2,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformerModif(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="relu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diata digunakan untuk memodifikasi model transformer. pada modifikasi ini saya memakai dropout sebesar 2. Adapun parameter-parameter yang digunakan seperti - 'input_shape' untuk menentukkan shape input data. - 'head_size', 'num_heads', 'ff_dim' parameter-parameter ini dipakai untuk mengatur konfigurasi dari layer Transformator. - 'mlp_units' adalah besar dari setiap lapisan dense untuk mengatur jumlah neuron dari lapisan dense. - 'num_transformer_blocks' menentukan jumlah blok Transformer.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
 ↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
```

```
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
model_modif1 = build_modif1(
    input_shape,
    head_size=46,
    num_heads=60,
    ff_dim=55,
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

model_modif1.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=1e-4),
    metrics=["mae"],
)

model_modif1.summary()
```

```
Model: "model_13"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape          Param #     Connected to
================================================================================
==================
 input_27 (InputLayer)          [(None, 8, 1)]        0           []

 layer_normalization_145 (Layer  (None, 8, 1)         2
```

```
['input_27[0][0]']
 Normalization)

 multi_head_attention_75 (Multi  (None, 8, 1)        19321
['layer_normalization_145[0][0]',
 HeadAttention)
'layer_normalization_145[0][0]']

 dropout_198 (Dropout)          (None, 8, 1)        0
['multi_head_attention_75[0][0]']

 tf.__operators__.add_105 (TFOp  (None, 8, 1)        0
['dropout_198[0][0]',
 Lambda)
'input_27[0][0]']

 layer_normalization_146 (Layer  (None, 8, 1)        2
['tf.__operators__.add_105[0][0]'
 Normalization)                                                   ]

 multi_head_attention_76 (Multi  (None, 8, 1)        19321
['layer_normalization_146[0][0]',
 HeadAttention)
'layer_normalization_146[0][0]']

 dropout_199 (Dropout)          (None, 8, 1)        0
['multi_head_attention_76[0][0]']

 tf.__operators__.add_106 (TFOp  (None, 8, 1)        0
['dropout_199[0][0]',
 Lambda)
'tf.__operators__.add_105[0][0]'
                                                                  ]

 layer_normalization_147 (Layer  (None, 8, 1)        2
['tf.__operators__.add_106[0][0]'
 Normalization)                                                   ]

 conv1d_135 (Conv1D)            (None, 8, 55)       110
['layer_normalization_147[0][0]']

 conv1d_136 (Conv1D)            (None, 8, 55)       3080
['conv1d_135[0][0]']

 conv1d_137 (Conv1D)            (None, 8, 1)        56
['conv1d_136[0][0]']

 tf.__operators__.add_107 (TFOp  (None, 8, 1)        0
```

```
['conv1d_137[0][0]',
 Lambda)
'tf.__operators__.add_106[0][0]'
                                                                ]

 layer_normalization_148 (Layer  (None, 8, 1)        2
['tf.__operators__.add_107[0][0]'
 Normalization)                                                 ]

 multi_head_attention_77 (Multi  (None, 8, 1)        19321
['layer_normalization_148[0][0]',
 HeadAttention)
'layer_normalization_148[0][0]']

 dropout_200 (Dropout)          (None, 8, 1)         0
['multi_head_attention_77[0][0]']

 tf.__operators__.add_108 (TFOp  (None, 8, 1)        0
['dropout_200[0][0]',
 Lambda)
'tf.__operators__.add_107[0][0]'
                                                                ]

 layer_normalization_149 (Layer  (None, 8, 1)        2
['tf.__operators__.add_108[0][0]'
 Normalization)                                                 ]

 multi_head_attention_78 (Multi  (None, 8, 1)        19321
['layer_normalization_149[0][0]',
 HeadAttention)
'layer_normalization_149[0][0]']

 dropout_201 (Dropout)          (None, 8, 1)         0
['multi_head_attention_78[0][0]']

 tf.__operators__.add_109 (TFOp  (None, 8, 1)        0
['dropout_201[0][0]',
 Lambda)
'tf.__operators__.add_108[0][0]'
                                                                ]

 layer_normalization_150 (Layer  (None, 8, 1)        2
['tf.__operators__.add_109[0][0]'
 Normalization)                                                 ]

 conv1d_138 (Conv1D)            (None, 8, 55)        110
['layer_normalization_150[0][0]']
```

```
 conv1d_139 (Conv1D)           (None, 8, 55)        3080
['conv1d_138[0][0]']

 conv1d_140 (Conv1D)           (None, 8, 1)         56
['conv1d_139[0][0]']

 tf.__operators__.add_110 (TFOp  (None, 8, 1)        0
['conv1d_140[0][0]',
 Lambda)
'tf.__operators__.add_109[0][0]'
                                                                  ]

 layer_normalization_151 (Layer  (None, 8, 1)        2
['tf.__operators__.add_110[0][0]'
 Normalization)                                                   ]

 multi_head_attention_79 (Multi  (None, 8, 1)        19321
['layer_normalization_151[0][0]',
 HeadAttention)
'layer_normalization_151[0][0]']

 dropout_202 (Dropout)         (None, 8, 1)         0
['multi_head_attention_79[0][0]']

 tf.__operators__.add_111 (TFOp  (None, 8, 1)        0
['dropout_202[0][0]',
 Lambda)
'tf.__operators__.add_110[0][0]'
                                                                  ]

 layer_normalization_152 (Layer  (None, 8, 1)        2
['tf.__operators__.add_111[0][0]'
 Normalization)                                                   ]

 multi_head_attention_80 (Multi  (None, 8, 1)        19321
['layer_normalization_152[0][0]',
 HeadAttention)
'layer_normalization_152[0][0]']

 dropout_203 (Dropout)         (None, 8, 1)         0
['multi_head_attention_80[0][0]']

 tf.__operators__.add_112 (TFOp  (None, 8, 1)        0
['dropout_203[0][0]',
 Lambda)
'tf.__operators__.add_111[0][0]'
                                                                  ]
```

```
 layer_normalization_153 (Layer  (None, 8, 1)        2
['tf.__operators__.add_112[0][0]'
 Normalization)                                                  ]

 conv1d_141 (Conv1D)            (None, 8, 55)        110
['layer_normalization_153[0][0]']

 conv1d_142 (Conv1D)            (None, 8, 55)        3080
['conv1d_141[0][0]']

 conv1d_143 (Conv1D)            (None, 8, 1)         56
['conv1d_142[0][0]']

 tf.__operators__.add_113 (TFOp  (None, 8, 1)        0
['conv1d_143[0][0]',
 Lambda)
'tf.__operators__.add_112[0][0]'
                                                                 ]

 layer_normalization_154 (Layer  (None, 8, 1)        2
['tf.__operators__.add_113[0][0]'
 Normalization)                                                  ]

 multi_head_attention_81 (Multi  (None, 8, 1)        19321
['layer_normalization_154[0][0]',
 HeadAttention)
'layer_normalization_154[0][0]']

 dropout_204 (Dropout)          (None, 8, 1)         0
['multi_head_attention_81[0][0]']

 tf.__operators__.add_114 (TFOp  (None, 8, 1)        0
['dropout_204[0][0]',
 Lambda)
'tf.__operators__.add_113[0][0]'
                                                                 ]

 layer_normalization_155 (Layer  (None, 8, 1)        2
['tf.__operators__.add_114[0][0]'
 Normalization)                                                  ]

 multi_head_attention_82 (Multi  (None, 8, 1)        19321
['layer_normalization_155[0][0]',
 HeadAttention)
'layer_normalization_155[0][0]']

 dropout_205 (Dropout)          (None, 8, 1)         0
['multi_head_attention_82[0][0]']
```

```
 tf.__operators__.add_115 (TFOp  (None, 8, 1)        0
['dropout_205[0][0]',
 Lambda)
'tf.__operators__.add_114[0][0]'
                                                                   ]

 layer_normalization_156 (Layer  (None, 8, 1)        2
['tf.__operators__.add_115[0][0]'
 Normalization)                                                    ]

 conv1d_144 (Conv1D)            (None, 8, 55)        110
['layer_normalization_156[0][0]']

 conv1d_145 (Conv1D)            (None, 8, 55)        3080
['conv1d_144[0][0]']

 conv1d_146 (Conv1D)            (None, 8, 1)         56
['conv1d_145[0][0]']

 tf.__operators__.add_116 (TFOp  (None, 8, 1)        0
['conv1d_146[0][0]',
 Lambda)
'tf.__operators__.add_115[0][0]'
                                                                   ]

 layer_normalization_157 (Layer  (None, 8, 1)        2
['tf.__operators__.add_116[0][0]'
 Normalization)                                                    ]

 multi_head_attention_83 (Multi  (None, 8, 1)        19321
['layer_normalization_157[0][0]',
 HeadAttention)
'layer_normalization_157[0][0]']

 dropout_206 (Dropout)          (None, 8, 1)         0
['multi_head_attention_83[0][0]']

 tf.__operators__.add_117 (TFOp  (None, 8, 1)        0
['dropout_206[0][0]',
 Lambda)
'tf.__operators__.add_116[0][0]'
                                                                   ]

 layer_normalization_158 (Layer  (None, 8, 1)        2
['tf.__operators__.add_117[0][0]'
 Normalization)                                                    ]
```

```
 multi_head_attention_84 (Multi  (None, 8, 1)        19321
['layer_normalization_158[0][0]',
 HeadAttention)
'layer_normalization_158[0][0]']

 dropout_207 (Dropout)          (None, 8, 1)        0
['multi_head_attention_84[0][0]']

 tf.__operators__.add_118 (TFOp  (None, 8, 1)        0
['dropout_207[0][0]',
 Lambda)
'tf.__operators__.add_117[0][0]'
                                                                ]

 layer_normalization_159 (Layer  (None, 8, 1)        2
['tf.__operators__.add_118[0][0]'
 Normalization)                                                 ]

 conv1d_147 (Conv1D)            (None, 8, 55)       110
['layer_normalization_159[0][0]']

 conv1d_148 (Conv1D)            (None, 8, 55)       3080
['conv1d_147[0][0]']

 conv1d_149 (Conv1D)            (None, 8, 1)        56
['conv1d_148[0][0]']

 tf.__operators__.add_119 (TFOp  (None, 8, 1)        0
['conv1d_149[0][0]',
 Lambda)
'tf.__operators__.add_118[0][0]'
                                                                ]

 global_average_pooling1d_13 (G  (None, 8)           0
['tf.__operators__.add_119[0][0]'
 lobalAveragePooling1D)                                         ]

 dense_61 (Dense)               (None, 256)         2304
['global_average_pooling1d_13[0][
                                                                0]']

 dropout_208 (Dropout)          (None, 256)         0
['dense_61[0][0]']

 dense_62 (Dense)               (None, 1)           257
['dropout_208[0][0]']

================================================================================
```

```
==================
Total params: 212,031
Trainable params: 212,031
Non-trainable params: 0
________________________________________________________________________________
__________________
```

[ ]:
```
historyModif1 = model_modif1.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=3,
    batch_size=20,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/3
303/303 [==============================] - 92s 272ms/step - loss: 0.2126 - mae:
0.2126 - val_loss: 0.4831 - val_mae: 0.4831 - lr: 1.0000e-06
Epoch 2/3
303/303 [==============================] - 80s 265ms/step - loss: 0.0956 - mae:
0.0956 - val_loss: 0.0875 - val_mae: 0.0875 - lr: 3.4300e-05
Epoch 3/3
303/303 [==============================] - 78s 257ms/step - loss: 0.0338 - mae:
0.0338 - val_loss: 0.0337 - val_mae: 0.0337 - lr: 6.7600e-05
```

[ ]:
```
# Evaluate the model on the test set
TessLossModif1 = model_modif1.evaluate(x_test, y_test)
print("Test Loss:", TessLossModif1)
```

```
24/24 [==============================] - 3s 118ms/step - loss: 0.0297 - mae:
0.0297
Test Loss: [0.029742585495114326, 0.029742585495114326]
```

Test Loss merupakan pengukuran seberapa jauh nilai sebenanrya dengan nilai prediksi. semakin kecil nilainya akan semakin bagus pengukurannya. untuk output diatas sudah terbilang kecil yaitu 0,2% yang artinya tes loss sendiri sudah bagus. Nilai loss pada test set digunakan untuk mengevaluasi model.

referensi :

Oh, H. W., Yoon, E. S., & Chung, M. K. (1997). An optimum set of loss models for performance prediction of centrifugal compressors. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, 211(4), 331-338.

4.4 Modifikasi ke-2

```
[ ]: def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0.1,␣
     ↪activation="relu"):

         # Normalization and Attention
         # "EMBEDDING LAYER"
         x = layers.LayerNormalization(epsilon=1e-6)(inputs)

         # Attention Layer
         attention_output = layers.MultiHeadAttention(
             key_dim=head_size, num_heads=num_heads, dropout=dropout
         )(x, x)
         attention_output = layers.Dropout(dropout)(attention_output)
         attention_output = layers.LayerNormalization(epsilon=1e-6)(attention_output␣
     ↪+ x)

         # Feed Forward Part
         ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
     ↪activation=activation)(attention_output)
         ff_output = layers.BatchNormalization()(ff_output)
         ff_output = layers.Dropout(dropout)(ff_output)

         ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
     ↪activation=activation)(ff_output)
         ff_output = layers.BatchNormalization()(ff_output)
         ff_output = layers.Dropout(dropout)(ff_output)

         transformer_output = layers.Add()([ff_output, attention_output])
         classification_output = layers.Dense(1,␣
     ↪activation="sigmoid")(transformer_output)

         return classification_output
```

Pada Modifikasi yang ke-2 ada beberapa modifikasi yang dilakukan pada function transfomer_encoder, yaitu:
1. attention_output = layers.LayerNormalization(epsilon=1e-6)(attention_output + x): Hasil Attention ditambahkan dengan input awal, kemudian dilakukan normalisasi menggunakan LayerNormalization. 2. melakukan normalisasi setelah layer attention dengan menggunakan LayerNormalization. 3. setelah feed-forward menambahkan residual connected. Tujuannya adalah untuk membantu memudahkan aliran gradien dan agar informasi yang terkait dapat bertahan dalam pemrosesan pembelajaran. 4. pada layer feed forward juga ditambahkan normalisasi batch pada output Conv1D menggunakan BatchNormalization. 5. pada 'transformer_output' dilakukan penjumlahan antara output dari lapisan Feed Forward dan output dari lapisan Attention.

dengan modifikasi ini diharapkan mendapatkan hasil yang lebih rendah daripada baseline.

Referensi : *Lin, T., Wang, Y., Liu, X., & Qiu, X. (2022). A survey of transformers. AI Open.*

[ ]:
```
def modif2_model(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=0,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diata digunakan untuk memodifikasi model transformer. pada modifikasi ini saya tidak memakai dropout sehingga ditulis 0. Adapun parameter-parameter yang digunakan seperti - 'input_shape' untuk menentukkan shape input data. - 'head_size', 'num_heads', 'ff_dim' parameter-parameter ini dipakai untuk mengatur konfigurasi dari layer Transformator. - 'mlp_units' adalah besar dari setiap lapisan dense untuk mengatur jumlah neuron dari lapisan dense. - 'num_transformer_blocks' menentukan jumlah blok Transformer.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
 ↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
model_2 = modif2_model(
    input_shape,
    head_size=46, # Embedding size for attention
    num_heads=60, # Number of attention heads
    ff_dim=55, # Hidden layer size in feed forward network inside transformer
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

model_2.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=1e-4),
    metrics=["mae"],
)
model_2.summary()
```

```
Model: "model_11"
________________________________________________________________________________
__________________
 Layer (type)                Output Shape         Param #     Connected to
================================================================================
==================
 input_25 (InputLayer)       [(None, 8, 1)]       0           []

 layer_normalization_125 (Layer  (None, 8, 1)     2
['input_25[0][0]']
 Normalization)

 multi_head_attention_65 (Multi  (None, 8, 1)     19321
['layer_normalization_125[0][0]',
 HeadAttention)
'layer_normalization_125[0][0]']

 dropout_166 (Dropout)       (None, 8, 1)         0
['multi_head_attention_65[0][0]']

 tf.__operators__.add_95 (TFOpL  (None, 8, 1)     0
['dropout_166[0][0]',
 ambda)
'layer_normalization_125[0][0]']
```

```
 layer_normalization_126 (Layer  (None, 8, 1)        2
['tf.__operators__.add_95[0][0]']
 Normalization)

 conv1d_115 (Conv1D)           (None, 8, 55)        110
['layer_normalization_126[0][0]']

 batch_normalization_50 (BatchN  (None, 8, 55)       220
['conv1d_115[0][0]']
 ormalization)

 dropout_167 (Dropout)         (None, 8, 55)        0
['batch_normalization_50[0][0]']

 conv1d_116 (Conv1D)           (None, 8, 55)        3080
['dropout_167[0][0]']

 batch_normalization_51 (BatchN  (None, 8, 55)       220
['conv1d_116[0][0]']
 ormalization)

 dropout_168 (Dropout)         (None, 8, 55)        0
['batch_normalization_51[0][0]']

 add_25 (Add)                  (None, 8, 55)        0
['dropout_168[0][0]',
'layer_normalization_126[0][0]']

 dense_47 (Dense)              (None, 8, 1)         56
['add_25[0][0]']

 layer_normalization_127 (Layer  (None, 8, 1)        2
['dense_47[0][0]']
 Normalization)

 multi_head_attention_66 (Multi  (None, 8, 1)        19321
['layer_normalization_127[0][0]',
 HeadAttention)
'layer_normalization_127[0][0]']

 dropout_169 (Dropout)         (None, 8, 1)         0
['multi_head_attention_66[0][0]']

 tf.__operators__.add_96 (TFOpL  (None, 8, 1)        0
['dropout_169[0][0]',
 ambda)
'layer_normalization_127[0][0]']
```

```
 layer_normalization_128 (Layer  (None, 8, 1)        2
['tf.__operators__.add_96[0][0]']
 Normalization)

 conv1d_117 (Conv1D)            (None, 8, 55)        110
['layer_normalization_128[0][0]']

 batch_normalization_52 (BatchN  (None, 8, 55)       220
['conv1d_117[0][0]']
 ormalization)

 dropout_170 (Dropout)          (None, 8, 55)        0
['batch_normalization_52[0][0]']

 conv1d_118 (Conv1D)            (None, 8, 55)        3080
['dropout_170[0][0]']

 batch_normalization_53 (BatchN  (None, 8, 55)       220
['conv1d_118[0][0]']
 ormalization)

 dropout_171 (Dropout)          (None, 8, 55)        0
['batch_normalization_53[0][0]']

 add_26 (Add)                   (None, 8, 55)        0
['dropout_171[0][0]',
'layer_normalization_128[0][0]']

 dense_48 (Dense)               (None, 8, 1)         56
['add_26[0][0]']

 layer_normalization_129 (Layer  (None, 8, 1)        2
['dense_48[0][0]']
 Normalization)

 multi_head_attention_67 (Multi  (None, 8, 1)        19321
['layer_normalization_129[0][0]',
 HeadAttention)
'layer_normalization_129[0][0]']

 dropout_172 (Dropout)          (None, 8, 1)         0
['multi_head_attention_67[0][0]']

 tf.__operators__.add_97 (TFOpL  (None, 8, 1)        0
['dropout_172[0][0]',
 ambda)
'layer_normalization_129[0][0]']
```

```
 layer_normalization_130 (Layer  (None, 8, 1)        2
['tf.__operators__.add_97[0][0]']
 Normalization)

 conv1d_119 (Conv1D)            (None, 8, 55)        110
['layer_normalization_130[0][0]']

 batch_normalization_54 (BatchN  (None, 8, 55)       220
['conv1d_119[0][0]']
 ormalization)

 dropout_173 (Dropout)          (None, 8, 55)        0
['batch_normalization_54[0][0]']

 conv1d_120 (Conv1D)            (None, 8, 55)        3080
['dropout_173[0][0]']

 batch_normalization_55 (BatchN  (None, 8, 55)       220
['conv1d_120[0][0]']
 ormalization)

 dropout_174 (Dropout)          (None, 8, 55)        0
['batch_normalization_55[0][0]']

 add_27 (Add)                   (None, 8, 55)        0
['dropout_174[0][0]',
'layer_normalization_130[0][0]']

 dense_49 (Dense)               (None, 8, 1)         56
['add_27[0][0]']

 layer_normalization_131 (Layer  (None, 8, 1)        2
['dense_49[0][0]']
 Normalization)

 multi_head_attention_68 (Multi  (None, 8, 1)        19321
['layer_normalization_131[0][0]',
 HeadAttention)
'layer_normalization_131[0][0]']

 dropout_175 (Dropout)          (None, 8, 1)         0
['multi_head_attention_68[0][0]']

 tf.__operators__.add_98 (TFOpL  (None, 8, 1)        0
['dropout_175[0][0]',
 ambda)
'layer_normalization_131[0][0]']
```

```
 layer_normalization_132 (Layer  (None, 8, 1)        2
['tf.__operators__.add_98[0][0]']
 Normalization)

 conv1d_121 (Conv1D)            (None, 8, 55)        110
['layer_normalization_132[0][0]']

 batch_normalization_56 (BatchN  (None, 8, 55)       220
['conv1d_121[0][0]']
 ormalization)

 dropout_176 (Dropout)          (None, 8, 55)        0
['batch_normalization_56[0][0]']

 conv1d_122 (Conv1D)            (None, 8, 55)        3080
['dropout_176[0][0]']

 batch_normalization_57 (BatchN  (None, 8, 55)       220
['conv1d_122[0][0]']
 ormalization)

 dropout_177 (Dropout)          (None, 8, 55)        0
['batch_normalization_57[0][0]']

 add_28 (Add)                   (None, 8, 55)        0
['dropout_177[0][0]',
'layer_normalization_132[0][0]']

 dense_50 (Dense)               (None, 8, 1)         56
['add_28[0][0]']

 layer_normalization_133 (Layer  (None, 8, 1)        2
['dense_50[0][0]']
 Normalization)

 multi_head_attention_69 (Multi  (None, 8, 1)        19321
['layer_normalization_133[0][0]',
 HeadAttention)
'layer_normalization_133[0][0]']

 dropout_178 (Dropout)          (None, 8, 1)         0
['multi_head_attention_69[0][0]']

 tf.__operators__.add_99 (TFOpL  (None, 8, 1)        0
['dropout_178[0][0]',
 ambda)
'layer_normalization_133[0][0]']
```

```
 layer_normalization_134 (Layer  (None, 8, 1)        2
['tf.__operators__.add_99[0][0]']
 Normalization)

 conv1d_123 (Conv1D)            (None, 8, 55)        110
['layer_normalization_134[0][0]']

 batch_normalization_58 (BatchN  (None, 8, 55)       220
['conv1d_123[0][0]']
 ormalization)

 dropout_179 (Dropout)          (None, 8, 55)        0
['batch_normalization_58[0][0]']

 conv1d_124 (Conv1D)            (None, 8, 55)        3080
['dropout_179[0][0]']

 batch_normalization_59 (BatchN  (None, 8, 55)       220
['conv1d_124[0][0]']
 ormalization)

 dropout_180 (Dropout)          (None, 8, 55)        0
['batch_normalization_59[0][0]']

 add_29 (Add)                   (None, 8, 55)        0
['dropout_180[0][0]',
'layer_normalization_134[0][0]']

 dense_51 (Dense)               (None, 8, 1)         56
['add_29[0][0]']

 global_average_pooling1d_11 (G  (None, 8)           0
['dense_51[0][0]']
 lobalAveragePooling1D)

 dense_52 (Dense)               (None, 256)          2304
['global_average_pooling1d_11[0][
                                                                 0]']

 dropout_181 (Dropout)          (None, 256)          0
['dense_52[0][0]']

 dense_53 (Dense)               (None, 1)            257
['dropout_181[0][0]']

================================================================================
==================
Total params: 117,616
```

```
Trainable params: 116,516
Non-trainable params: 1,100
________________________________________________________________________________
_________________
```

[ ]:
```
history_Modif2 = model_2.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=3,
    batch_size=20,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/3
303/303 [==============================] - 57s 157ms/step - loss: 0.3425 - mae:
0.3425 - val_loss: 0.5890 - val_mae: 0.5890 - lr: 1.0000e-06
Epoch 2/3
303/303 [==============================] - 43s 141ms/step - loss: 0.1814 - mae:
0.1814 - val_loss: 0.3104 - val_mae: 0.3104 - lr: 3.4300e-05
Epoch 3/3
303/303 [==============================] - 44s 145ms/step - loss: 0.1522 - mae:
0.1522 - val_loss: 0.2978 - val_mae: 0.2978 - lr: 6.7600e-05
```

[ ]:
```
# Evaluate the model on the test set
Modif2 = model_2.evaluate(x_test, y_test)
print("Test Loss:", TessLossModif1)
```

```
24/24 [==============================] - 2s 66ms/step - loss: 0.3344 - mae:
0.3344
Test Loss: [0.03982545807957649, 0.03982545807957649]
```

Test Loss merupakan pengukuran seberapa jauh nilai sebenanrya dengan nilai prediksi. semakin kecil nilainya akan semakin bagus pengukurannya. untuk output diatas sudah terbilang kecil yaitu 0,3% yang artinya tes loss sendiri sudah bagus. Nilai loss pada test set digunakan untuk mengevaluasi model.

referensi :

Oh, H. W., Yoon, E. S., & Chung, M. K. (1997). An optimum set of loss models for performance prediction of centrifugal compressors. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, 211(4), 331-338.

[ ]:
```
#modifiikasi data cisco
modifikasipredictions2 = model_2.predict(x_test)



rmse = np.sqrt(mean_squared_error(y_test, modifikasipredictions2))
```

```
mae = mean_absolute_error(y_test, modifikasipredictions2)
mape = mean_absolute_percentage_error(y_test, modifikasipredictions2)

print("Modify Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
24/24 [==============================] - 3s 73ms/step
Modify Model:
RMSE: 0.052564227652386976
MAE: 0.039825460310895004
MAPE: 247285641777.9027
```

Pada modifikasi ke-2 arsitektur Transformer didapatkan hasil :
- RMSE 0,5 % yang artinya sudah terbilang kecil - MAE 0,3% MAE juga dapat dikatakan kecil. - MAPE 24% artinya dapat dibilang lumayan besar.

4.5 modifikasi ke 3

[ ]:
```
def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0.1,␣
 ↪activation="relu"):

    # Normalization and Attention
    # "EMBEDDING LAYER"
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)

    # Attention Layer
    attention_output = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    attention_output = layers.Dropout(dropout)(attention_output)
    attention_output = layers.LayerNormalization(epsilon=1e-6)(attention_output␣
 ↪+ x)

    # Feed Forward Part
    ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
 ↪activation=activation)(attention_output)
    ff_output = layers.BatchNormalization()(ff_output)
    ff_output = layers.Dropout(dropout)(ff_output)

    ff_output = layers.Conv1D(filters=ff_dim, kernel_size=1,␣
 ↪activation=activation)(ff_output)
    ff_output = layers.BatchNormalization()(ff_output)
    ff_output = layers.Dropout(dropout)(ff_output)

    transformer_output = layers.Add()([ff_output, attention_output])
```

```
    classification_output = layers.Dense(1,␣
↪activation="sigmoid")(transformer_output)

    return classification_output
```

Pada Arsitektur yang ke-3 menambahkan beberapa layer tambahan, seperti 1. lapisan attention dan juga output attention ditambahkan dengan input asli 'attention_output + x'. 2. selain itu residual connection digunakan dalam penjumlahan akhir dengan output dari bagian feed-forward menggunakan 'layer.Add(). 3. saya juga menambahkan normalisasi batch dengan tujuan mengurangi ketergantungan pada distribusi input dan mempercepat proses pelatihan 4. pada bagian layer feed forward menambahkan 1 dropout dan juga menambahkan 1 transformasi linear dan activation sigmoid.

[ ]:
```
def modify_3(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
    num_transformer_blocks,
    mlp_units,
    dropout=0,
    mlp_dropout=0,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs

    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="elu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1, activation="linear")(x)
    return keras.Model(inputs, outputs)
```

Function diata digunakan untuk memodifikasi model transformer. pada modifikasi ini saya tidak memakai dropout sehingga ditulis 0. Adapun parameter-parameter yang digunakan seperti - 'input_shape' untuk menentukkan shape input data. - 'head_size', 'num_heads', 'ff_dim' parameter-parameter ini dipakai untuk mengatur konfigurasi dari layer Transformator. - 'mlp_units' adalah besar dari setiap lapisan dense untuk mengatur jumlah neuron dari lapisan dense. - 'num_transformer_blocks' menentukan jumlah blok Transformer.

[ ]:
```
def lr_scheduler(epoch, lr, warmup_epochs=30, decay_epochs=100,␣
↪initial_lr=1e-6, base_lr=1e-3, min_lr=5e-5):
    if epoch <= warmup_epochs:
        pct = epoch / warmup_epochs
```

```
        return ((base_lr - initial_lr) * pct) + initial_lr

    if epoch > warmup_epochs and epoch < warmup_epochs+decay_epochs:
        pct = 1 - ((epoch - warmup_epochs) / decay_epochs)
        return ((base_lr - min_lr) * pct) + min_lr

    return min_lr
```

[ ]:
```
callbacks = [
            keras.callbacks.EarlyStopping(patience=10,␣
 ↪restore_best_weights=True),
            keras.callbacks.LearningRateScheduler(lr_scheduler)
            ]
```

[ ]:
```
input_shape = x_train.shape[1:]
print(input_shape)
```

```
(8, 1)
```

[ ]:
```
model3 = modify_3(
    input_shape,
    head_size=46,
    num_heads=60,
    ff_dim=55,
    num_transformer_blocks=5,
    mlp_units=[256],
    mlp_dropout=0.4,
    dropout=0.14,
)

model3.compile(
    loss="mae",
    optimizer=keras.optimizers.Adam(learning_rate=0.001),
    metrics=["mae"],
)

model3.summary()
```

```
Model: "model_12"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_26 (InputLayer)          [(None, 8, 1)]       0           []

 layer_normalization_135 (Layer  (None, 8, 1)        2
```

```
['input_26[0][0]']
 Normalization)

 multi_head_attention_70 (Multi  (None, 8, 1)        19321
['layer_normalization_135[0][0]',
 HeadAttention)
'layer_normalization_135[0][0]']

 dropout_182 (Dropout)          (None, 8, 1)         0
['multi_head_attention_70[0][0]']

 tf.__operators__.add_100 (TFOp  (None, 8, 1)        0
['dropout_182[0][0]',
 Lambda)
'layer_normalization_135[0][0]']

 layer_normalization_136 (Layer  (None, 8, 1)        2
['tf.__operators__.add_100[0][0]'
 Normalization)                                                ]

 conv1d_125 (Conv1D)            (None, 8, 55)        110
['layer_normalization_136[0][0]']

 batch_normalization_60 (BatchN  (None, 8, 55)       220
['conv1d_125[0][0]']
 ormalization)

 dropout_183 (Dropout)          (None, 8, 55)        0
['batch_normalization_60[0][0]']

 conv1d_126 (Conv1D)            (None, 8, 55)        3080
['dropout_183[0][0]']

 batch_normalization_61 (BatchN  (None, 8, 55)       220
['conv1d_126[0][0]']
 ormalization)

 dropout_184 (Dropout)          (None, 8, 55)        0
['batch_normalization_61[0][0]']

 add_30 (Add)                   (None, 8, 55)        0
['dropout_184[0][0]',
'layer_normalization_136[0][0]']

 dense_54 (Dense)               (None, 8, 1)         56
['add_30[0][0]']

 layer_normalization_137 (Layer  (None, 8, 1)        2
```

```
['dense_54[0][0]']
 Normalization)

 multi_head_attention_71 (Multi  (None, 8, 1)        19321
['layer_normalization_137[0][0]',
 HeadAttention)
'layer_normalization_137[0][0]']

 dropout_185 (Dropout)          (None, 8, 1)         0
['multi_head_attention_71[0][0]']

 tf.__operators__.add_101 (TFOp  (None, 8, 1)        0
['dropout_185[0][0]',
 Lambda)
'layer_normalization_137[0][0]']

 layer_normalization_138 (Layer  (None, 8, 1)        2
['tf.__operators__.add_101[0][0]'
 Normalization)                                                      ]

 conv1d_127 (Conv1D)            (None, 8, 55)        110
['layer_normalization_138[0][0]']

 batch_normalization_62 (BatchN  (None, 8, 55)       220
['conv1d_127[0][0]']
 ormalization)

 dropout_186 (Dropout)          (None, 8, 55)        0
['batch_normalization_62[0][0]']

 conv1d_128 (Conv1D)            (None, 8, 55)        3080
['dropout_186[0][0]']

 batch_normalization_63 (BatchN  (None, 8, 55)       220
['conv1d_128[0][0]']
 ormalization)

 dropout_187 (Dropout)          (None, 8, 55)        0
['batch_normalization_63[0][0]']

 add_31 (Add)                   (None, 8, 55)        0
['dropout_187[0][0]',
'layer_normalization_138[0][0]']

 dense_55 (Dense)               (None, 8, 1)         56
['add_31[0][0]']

 layer_normalization_139 (Layer  (None, 8, 1)        2
```

```
['dense_55[0][0]']
 Normalization)

 multi_head_attention_72 (Multi  (None, 8, 1)        19321
['layer_normalization_139[0][0]',
 HeadAttention)
'layer_normalization_139[0][0]']

 dropout_188 (Dropout)          (None, 8, 1)        0
['multi_head_attention_72[0][0]']

 tf.__operators__.add_102 (TFOp  (None, 8, 1)        0
['dropout_188[0][0]',
 Lambda)
'layer_normalization_139[0][0]']

 layer_normalization_140 (Layer  (None, 8, 1)        2
['tf.__operators__.add_102[0][0]'
 Normalization)                                                ]

 conv1d_129 (Conv1D)            (None, 8, 55)       110
['layer_normalization_140[0][0]']

 batch_normalization_64 (BatchN  (None, 8, 55)       220
['conv1d_129[0][0]']
 ormalization)

 dropout_189 (Dropout)          (None, 8, 55)       0
['batch_normalization_64[0][0]']

 conv1d_130 (Conv1D)            (None, 8, 55)       3080
['dropout_189[0][0]']

 batch_normalization_65 (BatchN  (None, 8, 55)       220
['conv1d_130[0][0]']
 ormalization)

 dropout_190 (Dropout)          (None, 8, 55)       0
['batch_normalization_65[0][0]']

 add_32 (Add)                   (None, 8, 55)       0
['dropout_190[0][0]',
'layer_normalization_140[0][0]']

 dense_56 (Dense)               (None, 8, 1)        56
['add_32[0][0]']

 layer_normalization_141 (Layer  (None, 8, 1)        2
```

```
['dense_56[0][0]']
 Normalization)

 multi_head_attention_73 (Multi  (None, 8, 1)        19321
['layer_normalization_141[0][0]',
 HeadAttention)
'layer_normalization_141[0][0]']

 dropout_191 (Dropout)          (None, 8, 1)         0
['multi_head_attention_73[0][0]']

 tf.__operators__.add_103 (TFOp  (None, 8, 1)        0
['dropout_191[0][0]',
 Lambda)
'layer_normalization_141[0][0]']

 layer_normalization_142 (Layer  (None, 8, 1)        2
['tf.__operators__.add_103[0][0]'
 Normalization)                                                   ]

 conv1d_131 (Conv1D)            (None, 8, 55)        110
['layer_normalization_142[0][0]']

 batch_normalization_66 (BatchN  (None, 8, 55)       220
['conv1d_131[0][0]']
 ormalization)

 dropout_192 (Dropout)          (None, 8, 55)        0
['batch_normalization_66[0][0]']

 conv1d_132 (Conv1D)            (None, 8, 55)        3080
['dropout_192[0][0]']

 batch_normalization_67 (BatchN  (None, 8, 55)       220
['conv1d_132[0][0]']
 ormalization)

 dropout_193 (Dropout)          (None, 8, 55)        0
['batch_normalization_67[0][0]']

 add_33 (Add)                   (None, 8, 55)        0
['dropout_193[0][0]',
'layer_normalization_142[0][0]']

 dense_57 (Dense)               (None, 8, 1)         56
['add_33[0][0]']

 layer_normalization_143 (Layer  (None, 8, 1)        2
```

```
['dense_57[0][0]']
 Normalization)

 multi_head_attention_74 (Multi  (None, 8, 1)        19321
['layer_normalization_143[0][0]',
 HeadAttention)
'layer_normalization_143[0][0]']

 dropout_194 (Dropout)          (None, 8, 1)         0
['multi_head_attention_74[0][0]']

 tf.__operators__.add_104 (TFOp  (None, 8, 1)        0
['dropout_194[0][0]',
 Lambda)
'layer_normalization_143[0][0]']

 layer_normalization_144 (Layer  (None, 8, 1)        2
['tf.__operators__.add_104[0][0]'
 Normalization)                                                 ]

 conv1d_133 (Conv1D)            (None, 8, 55)        110
['layer_normalization_144[0][0]']

 batch_normalization_68 (BatchN  (None, 8, 55)       220
['conv1d_133[0][0]']
 ormalization)

 dropout_195 (Dropout)          (None, 8, 55)        0
['batch_normalization_68[0][0]']

 conv1d_134 (Conv1D)            (None, 8, 55)        3080
['dropout_195[0][0]']

 batch_normalization_69 (BatchN  (None, 8, 55)       220
['conv1d_134[0][0]']
 ormalization)

 dropout_196 (Dropout)          (None, 8, 55)        0
['batch_normalization_69[0][0]']

 add_34 (Add)                   (None, 8, 55)        0
['dropout_196[0][0]',
'layer_normalization_144[0][0]']

 dense_58 (Dense)               (None, 8, 1)         56
['add_34[0][0]']

 global_average_pooling1d_12 (G  (None, 8)           0
```

```
['dense_58[0][0]']
 lobalAveragePooling1D)

 dense_59 (Dense)              (None, 256)          2304
['global_average_pooling1d_12[0][
                                                                0]']

 dropout_197 (Dropout)         (None, 256)          0
['dense_59[0][0]']

 dense_60 (Dense)              (None, 1)            257
['dropout_197[0][0]']

================================================================================
==================
Total params: 117,616
Trainable params: 116,516
Non-trainable params: 1,100
________________________________________________________________________________
__________________
```

[ ]:
```
history_3 = model3.fit(
    x_train,
    y_train,
    validation_split=0.2,
    epochs=2,
    batch_size=32,
    callbacks=callbacks,
    validation_data = (x_val, y_val)
)
```

```
Epoch 1/2
190/190 [==============================] - 60s 252ms/step - loss: 0.1859 - mae:
0.1859 - val_loss: 0.3748 - val_mae: 0.3748 - lr: 1.0000e-06
Epoch 2/2
190/190 [==============================] - 42s 223ms/step - loss: 0.1646 - mae:
0.1646 - val_loss: 0.3038 - val_mae: 0.3038 - lr: 3.4300e-05
```

[ ]:
```
# Evaluate the model on the test set
TestLoss = model3.evaluate(x_test, y_test)
print("Test Loss:", TestLoss)
```

```
24/24 [==============================] - 2s 74ms/step - loss: 0.3384 - mae:
0.3384
Test Loss: [0.3384099006652832, 0.3384099006652832]
```

Test Loss merupakan pengukuran seberapa jauh nilai sebenanrya dengan nilai prediksi. semakin kecil nilainya akan semakin bagus pengukurannya. untuk output diatas sudah terbilang kecil yaitu

3,3%% tetapi tidak lebih bagus daripada modifikasi pertama yaitu 0,2%.

referensi :

Oh, H. W., Yoon, E. S., & Chung, M. K. (1997). An optimum set of loss models for performance prediction of centrifugal compressors. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, 211(4), 331-338.

[ ]:
```
# Matric Eval dalam test set

# Predict on the test set
baseline_predictions1 = model3.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, baseline_predictions1))
baseline_mae = mean_absolute_error(y_test, baseline_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, baseline_predictions1)

print("Baseline Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
24/24 [==============================] - 3s 69ms/step
Baseline Model:
RMSE: 0.39577612657544126
MAE: 0.33840987619753704
MAPE: 1109053864850.5317
```

pada performa pengukuran modifikasi ke-3 dilihat hasil MAPE sudah lebih baik dibandingkan sebelumnya yaitu 11% namun RMSE dan MAE meningkat tajam yaitu sebesar 39% dan juga 33%. itu artinya model modifikasi yang paling baik ada pada model modifikasi arsitektur ke-1 yang akan saya bahas lebih lanjut pada bagian evaluasi model itu sendiri.

5 D. Evaluasi Model Transformer

5.0.1 Dataset Amazon

Baseline

[ ]:
```
# Matric Eval dalam test set

# Predict on the test set
baseline_predictions1 = modelBaseline.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, baseline_predictions1))
baseline_mae = mean_absolute_error(y_test, baseline_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, baseline_predictions1)
```

```python
print("Baseline Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
18/18 [==============================] - 2s 87ms/step
Baseline Model:
RMSE: 0.6009641199513225
MAE: 0.5706344838451348
MAPE: 1.0188866419093743
```

Tuning Model

[ ]:
```python
#Tuning Model
modifikasi_predictions1 = model_modif1.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, modifikasi_predictions1))
baseline_mae = mean_absolute_error(y_test, modifikasi_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, modifikasi_predictions1)

print("Modify Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
18/18 [==============================] - 3s 119ms/step
Modify Model:
RMSE: 0.06235842225245711
MAE: 0.045295680700345016
MAPE: 0.10750118924609536
```

[ ]:
```python
import numpy as np

# Persiapkan data uji
x_test = np.array(test_windows1)  # Isi dengan testing set yang sesuai

# Ground Truth
ground_truth = np.array(test_labels1)  # Isi dengan nilai Ground Truth yang␣
 ↪sesuai dengan testing set

# Prediksi nilai
print("Shape of x_test:", x_test.shape)

print("Ground Truth:", ground_truth)
print("Predicted Result:", predicted_result)
```

```
Shape of x_test: (576, 5)
```

```
Ground Truth: [[1179.14001465]
 [1190.57995605]
 [1187.38000488]
 [1177.61999512]
 [1174.76000977]
 [1168.35998535]
 [1176.76000977]
 [1182.26000977]
 [1186.09997559]
 [1169.4699707 ]
 [1189.01000977]
 [1204.19995117]
 [1209.58996582]
 [1229.14001465]
 [1246.86999512]
 [1252.69995117]
 [1254.32995605]
 [1276.68005371]
 [1305.19995117]
 [1304.85998535]
 [1295.        ]
 [1293.31994629]
 [1294.57995605]
 [1327.31005859]
 [1362.54003906]
 [1357.51000977]
 [1377.94995117]
 [1402.05004883]
 [1417.68005371]
 [1437.81994629]
 [1450.89001465]
 [1390.        ]
 [1429.94995117]
 [1390.        ]
 [1442.83996582]
 [1416.7800293 ]
 [1350.5       ]
 [1339.59997559]
 [1386.22998047]
 [1414.51000977]
 [1451.05004883]
 [1461.76000977]
 [1448.68994141]
 [1468.34997559]
 [1482.92004395]
 [1485.33996582]
 [1500.        ]
 [1521.94995117]
```

```
[1511.97998047]
[1512.44995117]
[1493.44995117]
[1500.25      ]
[1523.60998535]
[1537.64001465]
[1545.        ]
[1551.85998535]
[1578.89001465]
[1598.39001465]
[1588.18005371]
[1591.        ]
[1582.31994629]
[1571.68005371]
[1544.93005371]
[1586.51000977]
[1581.85998535]
[1544.92004395]
[1495.56005859]
[1555.85998535]
[1497.05004883]
[1431.42004395]
[1447.33996582]
[1371.98999023]
[1392.05004883]
[1410.56994629]
[1451.75      ]
[1405.22998047]
[1406.07995605]
[1436.2199707 ]
[1427.05004883]
[1448.5       ]
[1430.79003906]
[1441.5       ]
[1503.82995605]
[1527.83996582]
[1556.91003418]
[1527.48999023]
[1517.85998535]
[1460.08996582]
[1460.17004395]
[1517.95996094]
[1572.61999512]
[1566.13000488]
[1582.26000977]
[1569.68005371]
[1572.07995605]
[1580.94995117]
```

```
[1600.14001465]
[1592.39001465]
[1608.        ]
[1609.07995605]
[1602.91003418]
[1601.54003906]
[1576.11999512]
[1587.2800293 ]
[1581.76000977]
[1574.36999512]
[1585.45996094]
[1581.40002441]
[1601.85998535]
[1603.06994629]
[1610.15002441]
[1612.86999512]
[1624.89001465]
[1629.61999512]
[1641.54003906]
[1665.27001953]
[1696.34997559]
[1695.75      ]
[1689.30004883]
[1683.98999023]
[1689.11999512]
[1698.75      ]
[1704.85998535]
[1723.85998535]
[1715.9699707 ]
[1723.79003906]
[1734.7800293 ]
[1750.07995605]
[1730.2199707 ]
[1715.67004395]
[1663.15002441]
[1691.08996582]
[1660.51000977]
[1701.44995117]
[1699.80004883]
[1713.7800293 ]
[1693.95996094]
[1699.72998047]
[1710.63000488]
[1739.02001953]
[1743.06994629]
[1755.        ]
[1796.61999512]
[1813.0300293 ]
```

```
[1822.48999023]
[1843.93005371]
[1842.92004395]
[1812.9699707 ]
[1813.69995117]
[1802.        ]
[1829.23999023]
[1863.60998535]
[1808.        ]
[1817.27001953]
[1779.2199707 ]
[1777.43994141]
[1797.17004395]
[1834.32995605]
[1823.29003906]
[1847.75      ]
[1862.47998047]
[1886.52001953]
[1898.52001953]
[1886.30004883]
[1896.19995117]
[1919.65002441]
[1882.61999512]
[1886.52001953]
[1882.2199707 ]
[1876.70996094]
[1883.42004395]
[1904.90002441]
[1902.90002441]
[1905.39001465]
[1927.68005371]
[1932.81994629]
[1998.09997559]
[2002.38000488]
[2012.70996094]
[2039.51000977]
[1994.81994629]
[1958.31005859]
[1952.06994629]
[1939.01000977]
[1987.15002441]
[1990.        ]
[1989.86999512]
[1970.18994141]
[1908.0300293 ]
[1941.05004883]
[1926.42004395]
[1944.30004883]
```

```
[1915.01000977]
[1934.35998535]
[1974.55004883]
[1974.84997559]
[2012.97998047]
[2003.        ]
[2004.35998535]
[1971.31005859]
[1952.76000977]
[1909.42004395]
[1889.65002441]
[1864.42004395]
[1870.31994629]
[1755.25      ]
[1719.35998535]
[1788.60998535]
[1760.94995117]
[1819.95996094]
[1831.72998047]
[1770.7199707 ]
[1764.0300293 ]
[1789.30004883]
[1768.69995117]
[1664.19995117]
[1782.17004395]
[1642.81005859]
[1538.88000488]
[1530.42004395]
[1598.01000977]
[1665.5300293 ]
[1665.5300293 ]
[1627.80004883]
[1642.81005859]
[1755.48999023]
[1754.91003418]
[1712.43005371]
[1636.84997559]
[1631.17004395]
[1599.01000977]
[1619.43994141]
[1593.41003418]
[1512.29003906]
[1495.45996094]
[1516.72998047]
[1502.06005859]
[1581.32995605]
[1581.42004395]
[1677.75      ]
```

```
[1673.56994629]
[1690.17004395]
[1772.35998535]
[1668.40002441]
[1699.18994141]
[1629.13000488]
[1641.0300293 ]
[1643.23999023]
[1663.54003906]
[1658.38000488]
[1591.91003418]
[1520.91003418]
[1551.47998047]
[1495.07995605]
[1460.82995605]
[1377.44995117]
[1343.95996094]
[1470.90002441]
[1461.64001465]
[1478.02001953]
[1501.9699707 ]
[1539.13000488]
[1500.2800293 ]
[1575.39001465]
[1629.51000977]
[1656.57995605]
[1659.42004395]
[1656.2199707 ]
[1640.56005859]
[1617.20996094]
[1674.56005859]
[1683.7800293 ]
[1693.2199707 ]
[1696.19995117]
[1632.17004395]
[1640.02001953]
[1654.93005371]
[1670.56994629]
[1637.89001465]
[1593.88000488]
[1670.43005371]
[1718.72998047]
[1626.22998047]
[1633.31005859]
[1658.81005859]
[1640.26000977]
[1614.36999512]
[1588.2199707 ]
```

```
[1591.        ]
[1638.01000977]
[1640.        ]
[1622.65002441]
[1607.94995117]
[1627.57995605]
[1622.09997559]
[1619.43994141]
[1631.56005859]
[1633.        ]
[1636.40002441]
[1641.08996582]
[1639.82995605]
[1671.72998047]
[1696.17004395]
[1692.43005371]
[1668.94995117]
[1625.94995117]
[1620.80004883]
[1670.61999512]
[1673.09997559]
[1690.81005859]
[1686.2199707 ]
[1712.35998535]
[1742.15002441]
[1761.84997559]
[1797.27001953]
[1819.26000977]
[1764.77001953]
[1774.26000977]
[1783.76000977]
[1765.69995117]
[1773.42004395]
[1780.75      ]
[1814.18994141]
[1813.97998047]
[1820.69995117]
[1818.85998535]
[1837.2800293 ]
[1849.85998535]
[1835.83996582]
[1847.32995605]
[1844.06994629]
[1843.06005859]
[1844.86999512]
[1863.04003906]
[1864.81994629]
[1861.68994141]
```

```
[1887.31005859]
[1923.77001953]
[1901.75      ]
[1902.25      ]
[1950.63000488]
[1938.43005371]
[1926.52001953]
[1911.52001953]
[1900.81994629]
[1962.45996094]
[1950.55004883]
[1921.        ]
[1917.77001953]
[1899.86999512]
[1889.97998047]
[1822.68005371]
[1840.11999512]
[1871.15002441]
[1907.56994629]
[1869.        ]
[1858.9699707 ]
[1857.52001953]
[1859.68005371]
[1815.47998047]
[1823.2800293 ]
[1836.43005371]
[1819.18994141]
[1816.31994629]
[1775.06994629]
[1692.68994141]
[1729.56005859]
[1738.5       ]
[1754.35998535]
[1804.0300293 ]
[1860.63000488]
[1863.69995117]
[1855.31994629]
[1870.30004883]
[1869.67004395]
[1886.0300293 ]
[1901.36999512]
[1908.79003906]
[1918.18994141]
[1911.30004883]
[1913.90002441]
[1878.27001953]
[1897.82995605]
[1904.2800293 ]
```

```
[1893.63000488]
[1922.18994141]
[1934.31005859]
[1939.        ]
[1942.91003418]
[1952.31994629]
[1988.30004883]
[2017.41003418]
[2001.06994629]
[2011.        ]
[2020.98999023]
[2009.90002441]
[1992.0300293 ]
[1977.90002441]
[1964.52001953]
[1985.63000488]
[1994.48999023]
[2000.81005859]
[1973.81994629]
[1943.05004883]
[1912.44995117]
[1898.5300293 ]
[1866.7800293 ]
[1855.31994629]
[1823.23999023]
[1765.13000488]
[1787.82995605]
[1793.40002441]
[1832.89001465]
[1807.57995605]
[1784.92004395]
[1824.33996582]
[1762.95996094]
[1776.11999512]
[1792.56994629]
[1816.11999512]
[1801.38000488]
[1823.54003906]
[1804.66003418]
[1749.61999512]
[1768.86999512]
[1761.82995605]
[1764.25      ]
[1786.40002441]
[1776.29003906]
[1789.83996582]
[1800.61999512]
[1840.7199707 ]
```

```
[1833.51000977]
[1831.34997559]
[1820.55004883]
[1822.98999023]
[1843.55004883]
[1839.33996582]
[1807.83996582]
[1822.55004883]
[1817.45996094]
[1821.5       ]
[1794.16003418]
[1785.30004883]
[1741.60998535]
[1768.32995605]
[1739.83996582]
[1725.44995117]
[1735.91003418]
[1735.65002441]
[1713.22998047]
[1724.42004395]
[1739.65002441]
[1732.66003418]
[1705.51000977]
[1721.98999023]
[1720.26000977]
[1731.92004395]
[1736.43005371]
[1767.38000488]
[1777.43005371]
[1787.47998047]
[1757.51000977]
[1785.66003418]
[1765.72998047]
[1762.17004395]
[1780.7800293 ]
[1761.32995605]
[1777.07995605]
[1762.70996094]
[1779.98999023]
[1776.66003418]
[1791.43994141]
[1804.66003418]
[1801.70996094]
[1795.77001953]
[1788.19995117]
[1785.88000488]
[1771.65002441]
[1778.        ]
```

```
[1753.10998535]
[1754.59997559]
[1739.48999023]
[1752.5300293 ]
[1752.79003906]
[1745.5300293 ]
[1734.70996094]
[1745.7199707 ]
[1773.83996582]
[1796.93994141]
[1818.51000977]
[1800.80004883]
[1781.59997559]
[1769.95996094]
[1760.68994141]
[1740.47998047]
[1751.59997559]
[1749.51000977]
[1739.20996094]
[1748.7199707 ]
[1760.32995605]
[1760.93994141]
[1769.20996094]
[1790.66003418]
[1784.0300293 ]
[1792.2800293 ]
[1786.5       ]
[1793.        ]
[1789.20996094]
[1868.77001953]
[1869.80004883]
[1846.89001465]
[1847.83996582]
[1898.01000977]
[1874.9699707 ]
[1902.88000488]
[1906.85998535]
[1891.9699707 ]
[1901.05004883]
[1883.16003418]
[1891.30004883]
[1869.43994141]
[1862.02001953]
[1877.93994141]
[1864.7199707 ]
[1892.        ]
[1887.45996094]
[1884.57995605]
```

```
[1861.64001465]
[1828.33996582]
[1853.25      ]
[1858.        ]
[1870.68005371]
[2008.7199707 ]
[2004.19995117]
[2049.66992188]
[2039.86999512]
[2050.22998047]
[2079.2800293 ]
[2133.90991211]
[2150.80004883]
[2160.        ]
[2149.87011719]
[2134.87011719]
[2155.66992188]
[2170.2199707 ]
[2153.10009766]
[2095.9699707 ]
[2009.29003906]
[1972.73999023]
[1979.58996582]
[1884.30004883]
[1883.75      ]
[1953.94995117]
[1908.98999023]
[1975.82995605]
[1924.0300293 ]
[1901.08996582]
[1800.60998535]
[1891.81994629]
[1820.85998535]
[1676.60998535]
[1785.        ]
[1689.15002441]
[1807.83996582]
[1830.        ]
[1880.93005371]
[1846.08996582]
[1902.82995605]
[1940.09997559]
[1885.83996582]
[1955.48999023]
[1900.09997559]
[1963.94995117]
[1949.7199707 ]
[1907.69995117]]
```

```
Predicted Result: [[0.2855832 ]
 [0.6932517 ]
 [0.11486241]
 [0.9657291 ]
 [0.31072176]
 [0.55184937]
 [0.4082673 ]
 [0.6724903 ]
 [0.7733001 ]
 [0.5327494 ]
 [0.48186168]
 [0.8179988 ]
 [0.3870478 ]
 [0.43497628]
 [0.6879024 ]
 [0.64359266]
 [0.78955585]
 [0.68042445]
 [0.8323975 ]
 [0.743834  ]
 [0.7754506 ]
 [0.5105114 ]
 [0.6403891 ]
 [0.67630565]
 [0.627643  ]
 [0.83540535]
 [0.64972615]
 [0.6139797 ]
 [0.43290508]
 [0.56504965]
 [0.3572133 ]
 [0.654306  ]
 [0.8610606 ]
 [0.72745436]
 [0.6959712 ]
 [0.36353353]
 [0.5425327 ]
 [0.82767665]
 [0.62131584]
 [0.5144887 ]
 [0.83751565]
 [0.5182816 ]
 [0.3536912 ]
 [0.2728471 ]
 [0.46613446]
 [0.54008496]
 [0.74025464]
 [0.39936724]
```

```
[0.62650985]
[0.6361414 ]
[0.8204444 ]
[0.6603745 ]
[0.8314741 ]
[0.5950399 ]
[0.968202  ]
[0.64540637]
[0.36033395]
[0.433249  ]
[0.61360246]
[0.5676944 ]
[0.49052152]
[0.96280336]
[0.65609086]
[0.49717805]
[0.5294765 ]
[0.7286799 ]
[0.6338401 ]
[0.66594386]
[0.6357785 ]
[0.5702662 ]
[0.03351967]
[0.6497111 ]
[0.79523104]
[0.51862943]
[0.6279361 ]
[0.76464164]
[0.6429674 ]
[0.4111479 ]
[0.55673313]
[0.5425962 ]
[0.8279141 ]
[0.49815127]
[0.03476203]
[0.5900992 ]
[0.8864521 ]
[0.09863628]
[0.7126748 ]
[0.65746206]
[0.74258155]
[0.90729547]
[0.6200556 ]
[0.66974586]
[0.5727356 ]
[0.27494574]
[0.5266695 ]
[0.29566348]
```

```
[0.7073396 ]
[0.48245564]
[0.638438  ]
[0.66750765]
[0.33308324]
[0.26134804]
[0.5853533 ]
[0.64819765]
[0.53951174]
[0.74776924]
[0.6149692 ]
[0.7312823 ]
[0.50326824]
[0.72094554]
[0.4249293 ]
[0.6920904 ]
[0.8230093 ]
[0.47613034]
[0.6252593 ]
[0.44434765]
[0.79942536]
[0.6544408 ]
[0.74022096]
[0.7024729 ]
[0.71770906]
[0.60662913]
[0.6442797 ]
[0.66768706]
[0.83527493]
[0.68372095]
[0.5481428 ]
[0.266255  ]
[0.67510855]
[0.48929402]
[0.7081455 ]
[0.74785197]
[0.51226413]
[0.4660859 ]
[0.5656697 ]
[0.50678515]
[0.75979525]
[0.6774312 ]
[0.07190609]
[0.75918   ]
[0.5301541 ]
[0.8072131 ]
[0.6802184 ]
[0.8392882 ]
```

```
[0.5229217 ]
[0.63754916]
[0.6749865 ]
[0.74048555]
[0.6879262 ]
[0.79205674]
[0.4010593 ]
[0.54450774]
[0.43771818]
[0.6268635 ]
[0.41053638]
[0.75996625]
[0.49385333]
[0.28816196]
[0.7532164 ]
[0.56865513]
[0.6892993 ]
[0.2809228 ]
[0.8245723 ]
[0.03365146]
[0.39626718]
[0.3561926 ]
[0.23806381]
[0.8430318 ]
[0.3809031 ]
[0.5642816 ]
[0.5115568 ]
[0.24014309]
[0.77747333]
[0.73828566]
[0.6454984 ]
[0.3926893 ]
[0.36707324]
[0.41859576]
[0.41956702]
[0.6915831 ]
[0.820791  ]
[0.46012166]
[0.41362828]
[0.6313999 ]
[0.52806234]
[0.8426588 ]
[0.60659134]
[0.53508055]
[0.8510951 ]
[0.45261952]
[0.7112566 ]
[0.83965105]
```

```
[0.7277943 ]
[0.2962789 ]
[0.7373495 ]
[0.6193105 ]
[0.40314916]
[0.611792  ]
[0.5012904 ]
[0.7174273 ]
[0.20423019]
[0.1396431 ]
[0.68559206]
[0.68420464]
[0.29016855]
[0.543121  ]
[0.5181025 ]
[0.6832544 ]
[0.48341754]
[0.42218408]
[0.8290101 ]
[0.5304892 ]
[0.66135633]
[0.62741363]
[0.3048719 ]
[0.5599156 ]
[0.7402242 ]
[0.4813547 ]
[0.6850115 ]
[0.610384  ]
[0.46984687]
[0.53171694]
[0.6876874 ]
[0.45528957]
[0.5850327 ]
[0.33423963]
[0.6949742 ]
[0.6705627 ]
[0.40776846]
[0.5729445 ]
[0.5536664 ]
[0.29641682]
[0.675075  ]
[0.28049913]
[0.77778363]
[0.5046068 ]
[0.4410262 ]
[0.8320779 ]
[0.35679325]
[0.4253748 ]
```

```
[0.6192654 ]
[0.48877773]
[0.65173817]
[0.56613827]
[0.7530913 ]
[0.36346298]
[0.34326872]
[0.03271864]
[0.69766146]
[0.28252858]
[0.67084813]
[0.72746956]
[0.29704812]
[0.31984004]
[0.46020266]
[0.96040106]
[0.66835535]
[0.8312869 ]
[0.99994314]
[0.08431134]
[0.6161076 ]
[0.43116346]
[0.6143327 ]
[0.43357217]
[0.5059161 ]
[0.5173664 ]
[0.74746394]
[0.98746014]
[0.66158164]
[0.68462265]
[0.75420964]
[0.78796136]
[0.73522353]
[0.47708437]
[0.77128255]
[0.1783234 ]
[0.69776845]
[0.6511602 ]
[0.62074614]
[0.64799905]
[0.49474305]
[0.2967616 ]
[0.7431356 ]
[0.74487454]
[0.4901115 ]
[0.27181947]
[0.03882899]
[0.61800885]
```

```
[0.6165613 ]
[0.81917286]
[0.7420243 ]
[0.66470194]
[0.45284575]
[0.45551774]
[0.49335894]
[0.5916759 ]
[0.44901386]
[0.21313827]
[0.78726524]
[0.3640695 ]
[0.5231787 ]
[0.7679124 ]
[0.63115066]
[0.74548113]
[0.54403174]
[0.37392822]
[0.4592506 ]
[0.6407714 ]
[0.5121368 ]
[0.77215594]
[0.62855893]
[0.8071449 ]
[0.5052591 ]
[0.72372055]
[0.554204  ]
[0.6175524 ]
[0.82507014]
[0.62710875]
[0.7832123 ]
[0.61792254]
[0.77428526]
[0.19468331]
[0.63204384]
[0.537225  ]
[0.82301545]
[0.69425523]
[0.79769874]
[0.679635  ]
[0.6288698 ]
[0.36582807]
[0.59450823]
[0.85387015]
[0.45066968]
[0.8035729 ]
[0.84241766]
[0.57800794]
```

```
[0.7434015 ]
[0.6211376 ]
[0.6259845 ]
[0.6436701 ]
[0.6233378 ]
[0.7391274 ]
[0.39862457]
[0.77377486]
[0.5854329 ]
[0.84408116]
[0.7028254 ]
[0.29057187]
[0.45108685]
[0.73474723]
[0.40534562]
[0.43936995]
[0.72896415]
[0.6658514 ]
[0.40959337]
[0.3004585 ]
[0.65894705]
[0.6891302 ]
[0.3244134 ]
[0.5076535 ]
[0.7911937 ]
[0.73439825]
[0.336502  ]
[0.83899975]
[0.7150979 ]
[0.49467248]
[0.64277875]
[0.7667155 ]
[0.695913  ]
[0.6341802 ]
[0.5281434 ]
[0.7302926 ]
[0.5245414 ]
[0.7670664 ]
[0.7001156 ]
[0.63375   ]
[0.6432688 ]
[0.7029854 ]
[0.69058675]
[0.03302487]
[0.03789589]
[0.69234943]
[0.31514862]
[0.51671803]
```

```
[0.6856072 ]
[0.3905798 ]
[0.66609925]
[0.4378145 ]
[0.4362541 ]
[0.6479782 ]
[0.12942426]
[0.24554   ]
[0.479668  ]
[0.62845266]
[0.76507133]
[0.8216566 ]
[0.34690818]
[0.50619316]
[0.7451006 ]
[0.74475145]
[0.35787442]
[0.3174025 ]
[0.4818124 ]
[0.6257971 ]
[0.49329054]
[0.49541482]
[0.7923345 ]
[0.7401109 ]
[0.7391014 ]
[0.7098096 ]
[0.66529685]
[0.49390063]
[0.2502883 ]
[0.8462622 ]
[0.52813536]
[0.32649347]
[0.6177884 ]
[0.59463555]
[0.7942935 ]
[0.7534044 ]
[0.6625885 ]
[0.58335626]
[0.8998822 ]
[0.7452588 ]
[0.69192827]
[0.25796077]
[0.61589986]
[0.5238477 ]
[0.5310627 ]
[0.541687  ]
[0.15230486]
[0.67416   ]
```

```
[0.4562101 ]
[0.64385676]
[0.60931635]
[0.49613377]
[0.02863361]
[0.4362521 ]
[0.66406333]
[0.32177195]
[0.35414615]
[0.74098873]
[0.66322863]
[0.64249146]
[0.6898375 ]
[0.7388159 ]
[0.84683394]
[0.4668781 ]
[0.62975395]
[0.2785108 ]
[0.6402438 ]
[0.67883337]
[0.6956201 ]
[0.62847745]
[0.84540224]
[0.7522449 ]
[0.5076015 ]
[0.4485077 ]
[0.4977571 ]
[0.5850896 ]
[0.9347395 ]
[0.26895973]
[0.98765475]
[0.6907323 ]
[0.6143328 ]
[0.512661  ]
[0.6769247 ]
[0.13883156]
[0.0281591 ]
[0.7016674 ]
[0.59540796]
[0.7860996 ]
[0.63817763]
[0.6250788 ]
[0.6403114 ]
[0.04751558]
[0.7565215 ]
[0.22652052]
[0.7757872 ]
[0.5254519 ]
```

```
[0.7702571 ]
[0.65841633]
[0.7436378 ]
[0.67981815]
[0.35096237]
[0.8487401 ]
[0.56294024]
[0.70194435]
[0.647908  ]
[0.80754876]
[0.6754747 ]
[0.45034817]
[0.4598984 ]
[0.545737  ]
[0.8441521 ]
[0.0777718 ]
[0.4104322 ]
[0.67503685]
[0.81409097]
[0.72481817]
[0.8090874 ]
[0.44912818]
[0.5828578 ]
[0.6648059 ]
[0.64114934]
[0.4707133 ]
[0.6347388 ]
[0.639168  ]
[0.6327965 ]
[0.4751758 ]
[0.6247164 ]
[0.70876217]
[0.7972896 ]
[0.05888995]
[0.6366228 ]
[0.66864324]
[0.61669457]
[0.1382289 ]
[0.6082579 ]
[0.6457496 ]
[0.6461952 ]
[0.4674068 ]
[0.63969517]
[0.639392  ]
[0.5209262 ]
[0.81260693]
[0.5980431 ]
[0.34071648]
```

```
 [0.71412873]
 [0.45417082]
 [0.02600437]
 [0.8456366 ]
 [0.7943667 ]
 [0.48782352]
 [0.6336312 ]
 [0.604554  ]
 [0.33808568]
 [0.6370814 ]
 [0.51194   ]
 [0.5736482 ]
 [0.7656176 ]
 [0.7440399 ]
 [0.61957246]
 [0.8029887 ]
 [0.75594795]
 [0.80369574]
 [0.6059953 ]
 [0.49216273]
 [0.5810938 ]
 [0.7095976 ]
 [0.6712855 ]
 [0.6613779 ]
 [0.5090179 ]
 [0.7129188 ]
 [0.753641  ]
 [0.6651983 ]
 [0.95012164]
 [0.48626083]
 [0.7415267 ]
 [0.6289097 ]
 [0.80254245]
 [0.749521  ]
 [0.63031524]
 [0.45758507]
 [0.6757301 ]
 [0.58425546]
 [0.7136602 ]
 [0.6161767 ]]
```

pada hasil Evaluasi terdapat baseline dan juga Tuning model. baseline dari arsitektur Transformer menggunakan satu layer Conv1D pada bagian Feed Forward dengan Activation function menggunakan ReLU dan bagian node perceptron pada output disesuaikan dengan horizon datanya. Baseline juga terdapat layer multi-Head Attention digunkan dalam lapisan encoder dan decoder untuk memproses lebih baik dari input dan memperoleh hasil yang lebih baik. Pada Baseline Model didapat hasil yang sedikit lebih tinggi daripada tuning model, itu berarti Tuning model sudah berjalan dengan baik dan mempunyai arsitektur yang lebih baik daripada baseline. karena Tuning dapat mengalahkan Baseline model Arsitektur Attention.

Dengan Tuning berhasil pada modifikasi pertama yaitu dengan pemodifikasian menambahkan attention layer untuk agar model lebih fokus pada bagian input. pada pemprosesan attention juga memberikan representasi lebih real. kemudian dengan ini diharapkan model dapat memahami lebih baik tentang relasi titik di inpput.

kemudian, saya juga menambakan layer dropout dan convolution 1D di bagian feed forward.

Dropout saya gunakan untuk menghilangkan node dari layer sebelumnya, dengan tujuan mengurangi overfitting cara ini terbukti pada paper Srivastava et.al.

Referensi : Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.

5.0.2 Dataset Cisco

Baseline Model

[ ]:
```
# Matric Eval dalam test set

# Predict on the test set
baseline_predictions1 = modelBaseline.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, baseline_predictions1))
baseline_mae = mean_absolute_error(y_test, baseline_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, baseline_predictions1)

print("Baseline Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
24/24 [==============================] - 2s 67ms/step
Baseline Model:
RMSE: 0.04936246652260908
MAE: 0.035229160721132316
MAPE: 256365377249.79178
```

[ ]:
```
import numpy as np

# Persiapkan data uji
x_test = np.array(test_windows1)  # Isi dengan testing set yang sesuai

# Ground Truth
ground_truth = np.array(test_labels1)  # Isi dengan nilai Ground Truth yang␣
↪sesuai dengan testing set

# Prediksi nilai
```

```python
print("Shape of x_test:", x_test.shape)

print("Ground Truth:", ground_truth)
print("Predicted Result:", predicted_result)
```

```
Shape of x_test: (576, 5)
Ground Truth: [[1179.14001465]
 [1190.57995605]
 [1187.38000488]
 [1177.61999512]
 [1174.76000977]
 [1168.35998535]
 [1176.76000977]
 [1182.26000977]
 [1186.09997559]
 [1169.4699707 ]
 [1189.01000977]
 [1204.19995117]
 [1209.58996582]
 [1229.14001465]
 [1246.86999512]
 [1252.69995117]
 [1254.32995605]
 [1276.68005371]
 [1305.19995117]
 [1304.85998535]
 [1295.        ]
 [1293.31994629]
 [1294.57995605]
 [1327.31005859]
 [1362.54003906]
 [1357.51000977]
 [1377.94995117]
 [1402.05004883]
 [1417.68005371]
 [1437.81994629]
 [1450.89001465]
 [1390.        ]
 [1429.94995117]
 [1390.        ]
 [1442.83996582]
 [1416.7800293 ]
 [1350.5       ]
 [1339.59997559]
 [1386.22998047]
 [1414.51000977]
 [1451.05004883]
```

```
[1461.76000977]
[1448.68994141]
[1468.34997559]
[1482.92004395]
[1485.33996582]
[1500.        ]
[1521.94995117]
[1511.97998047]
[1512.44995117]
[1493.44995117]
[1500.25      ]
[1523.60998535]
[1537.64001465]
[1545.        ]
[1551.85998535]
[1578.89001465]
[1598.39001465]
[1588.18005371]
[1591.        ]
[1582.31994629]
[1571.68005371]
[1544.93005371]
[1586.51000977]
[1581.85998535]
[1544.92004395]
[1495.56005859]
[1555.85998535]
[1497.05004883]
[1431.42004395]
[1447.33996582]
[1371.98999023]
[1392.05004883]
[1410.56994629]
[1451.75      ]
[1405.22998047]
[1406.07995605]
[1436.2199707 ]
[1427.05004883]
[1448.5       ]
[1430.79003906]
[1441.5       ]
[1503.82995605]
[1527.83996582]
[1556.91003418]
[1527.48999023]
[1517.85998535]
[1460.08996582]
[1460.17004395]
```

```
[1517.95996094]
[1572.61999512]
[1566.13000488]
[1582.26000977]
[1569.68005371]
[1572.07995605]
[1580.94995117]
[1600.14001465]
[1592.39001465]
[1608.        ]
[1609.07995605]
[1602.91003418]
[1601.54003906]
[1576.11999512]
[1587.2800293 ]
[1581.76000977]
[1574.36999512]
[1585.45996094]
[1581.40002441]
[1601.85998535]
[1603.06994629]
[1610.15002441]
[1612.86999512]
[1624.89001465]
[1629.61999512]
[1641.54003906]
[1665.27001953]
[1696.34997559]
[1695.75      ]
[1689.30004883]
[1683.98999023]
[1689.11999512]
[1698.75      ]
[1704.85998535]
[1723.85998535]
[1715.9699707 ]
[1723.79003906]
[1734.7800293 ]
[1750.07995605]
[1730.2199707 ]
[1715.67004395]
[1663.15002441]
[1691.08996582]
[1660.51000977]
[1701.44995117]
[1699.80004883]
[1713.7800293 ]
[1693.95996094]
```

```
[1699.72998047]
[1710.63000488]
[1739.02001953]
[1743.06994629]
[1755.        ]
[1796.61999512]
[1813.0300293 ]
[1822.48999023]
[1843.93005371]
[1842.92004395]
[1812.9699707 ]
[1813.69995117]
[1802.        ]
[1829.23999023]
[1863.60998535]
[1808.        ]
[1817.27001953]
[1779.2199707 ]
[1777.43994141]
[1797.17004395]
[1834.32995605]
[1823.29003906]
[1847.75      ]
[1862.47998047]
[1886.52001953]
[1898.52001953]
[1886.30004883]
[1896.19995117]
[1919.65002441]
[1882.61999512]
[1886.52001953]
[1882.2199707 ]
[1876.70996094]
[1883.42004395]
[1904.90002441]
[1902.90002441]
[1905.39001465]
[1927.68005371]
[1932.81994629]
[1998.09997559]
[2002.38000488]
[2012.70996094]
[2039.51000977]
[1994.81994629]
[1958.31005859]
[1952.06994629]
[1939.01000977]
[1987.15002441]
```

```
[1990.        ]
[1989.86999512]
[1970.18994141]
[1908.0300293 ]
[1941.05004883]
[1926.42004395]
[1944.30004883]
[1915.01000977]
[1934.35998535]
[1974.55004883]
[1974.84997559]
[2012.97998047]
[2003.        ]
[2004.35998535]
[1971.31005859]
[1952.76000977]
[1909.42004395]
[1889.65002441]
[1864.42004395]
[1870.31994629]
[1755.25      ]
[1719.35998535]
[1788.60998535]
[1760.94995117]
[1819.95996094]
[1831.72998047]
[1770.7199707 ]
[1764.0300293 ]
[1789.30004883]
[1768.69995117]
[1664.19995117]
[1782.17004395]
[1642.81005859]
[1538.88000488]
[1530.42004395]
[1598.01000977]
[1665.5300293 ]
[1665.5300293 ]
[1627.80004883]
[1642.81005859]
[1755.48999023]
[1754.91003418]
[1712.43005371]
[1636.84997559]
[1631.17004395]
[1599.01000977]
[1619.43994141]
[1593.41003418]
```

```
[1512.29003906]
[1495.45996094]
[1516.72998047]
[1502.06005859]
[1581.32995605]
[1581.42004395]
[1677.75      ]
[1673.56994629]
[1690.17004395]
[1772.35998535]
[1668.40002441]
[1699.18994141]
[1629.13000488]
[1641.0300293 ]
[1643.23999023]
[1663.54003906]
[1658.38000488]
[1591.91003418]
[1520.91003418]
[1551.47998047]
[1495.07995605]
[1460.82995605]
[1377.44995117]
[1343.95996094]
[1470.90002441]
[1461.64001465]
[1478.02001953]
[1501.9699707 ]
[1539.13000488]
[1500.2800293 ]
[1575.39001465]
[1629.51000977]
[1656.57995605]
[1659.42004395]
[1656.2199707 ]
[1640.56005859]
[1617.20996094]
[1674.56005859]
[1683.7800293 ]
[1693.2199707 ]
[1696.19995117]
[1632.17004395]
[1640.02001953]
[1654.93005371]
[1670.56994629]
[1637.89001465]
[1593.88000488]
[1670.43005371]
```

```
[1718.72998047]
[1626.22998047]
[1633.31005859]
[1658.81005859]
[1640.26000977]
[1614.36999512]
[1588.2199707 ]
[1591.        ]
[1638.01000977]
[1640.        ]
[1622.65002441]
[1607.94995117]
[1627.57995605]
[1622.09997559]
[1619.43994141]
[1631.56005859]
[1633.        ]
[1636.40002441]
[1641.08996582]
[1639.82995605]
[1671.72998047]
[1696.17004395]
[1692.43005371]
[1668.94995117]
[1625.94995117]
[1620.80004883]
[1670.61999512]
[1673.09997559]
[1690.81005859]
[1686.2199707 ]
[1712.35998535]
[1742.15002441]
[1761.84997559]
[1797.27001953]
[1819.26000977]
[1764.77001953]
[1774.26000977]
[1783.76000977]
[1765.69995117]
[1773.42004395]
[1780.75      ]
[1814.18994141]
[1813.97998047]
[1820.69995117]
[1818.85998535]
[1837.2800293 ]
[1849.85998535]
[1835.83996582]
```

```
[1847.32995605]
[1844.06994629]
[1843.06005859]
[1844.86999512]
[1863.04003906]
[1864.81994629]
[1861.68994141]
[1887.31005859]
[1923.77001953]
[1901.75      ]
[1902.25      ]
[1950.63000488]
[1938.43005371]
[1926.52001953]
[1911.52001953]
[1900.81994629]
[1962.45996094]
[1950.55004883]
[1921.        ]
[1917.77001953]
[1899.86999512]
[1889.97998047]
[1822.68005371]
[1840.11999512]
[1871.15002441]
[1907.56994629]
[1869.        ]
[1858.9699707 ]
[1857.52001953]
[1859.68005371]
[1815.47998047]
[1823.2800293 ]
[1836.43005371]
[1819.18994141]
[1816.31994629]
[1775.06994629]
[1692.68994141]
[1729.56005859]
[1738.5       ]
[1754.35998535]
[1804.0300293 ]
[1860.63000488]
[1863.69995117]
[1855.31994629]
[1870.30004883]
[1869.67004395]
[1886.0300293 ]
[1901.36999512]
```

```
[1908.79003906]
[1918.18994141]
[1911.30004883]
[1913.90002441]
[1878.27001953]
[1897.82995605]
[1904.2800293 ]
[1893.63000488]
[1922.18994141]
[1934.31005859]
[1939.        ]
[1942.91003418]
[1952.31994629]
[1988.30004883]
[2017.41003418]
[2001.06994629]
[2011.        ]
[2020.98999023]
[2009.90002441]
[1992.0300293 ]
[1977.90002441]
[1964.52001953]
[1985.63000488]
[1994.48999023]
[2000.81005859]
[1973.81994629]
[1943.05004883]
[1912.44995117]
[1898.5300293 ]
[1866.7800293 ]
[1855.31994629]
[1823.23999023]
[1765.13000488]
[1787.82995605]
[1793.40002441]
[1832.89001465]
[1807.57995605]
[1784.92004395]
[1824.33996582]
[1762.95996094]
[1776.11999512]
[1792.56994629]
[1816.11999512]
[1801.38000488]
[1823.54003906]
[1804.66003418]
[1749.61999512]
[1768.86999512]
```

```
[1761.82995605]
[1764.25      ]
[1786.40002441]
[1776.29003906]
[1789.83996582]
[1800.61999512]
[1840.7199707 ]
[1833.51000977]
[1831.34997559]
[1820.55004883]
[1822.98999023]
[1843.55004883]
[1839.33996582]
[1807.83996582]
[1822.55004883]
[1817.45996094]
[1821.5       ]
[1794.16003418]
[1785.30004883]
[1741.60998535]
[1768.32995605]
[1739.83996582]
[1725.44995117]
[1735.91003418]
[1735.65002441]
[1713.22998047]
[1724.42004395]
[1739.65002441]
[1732.66003418]
[1705.51000977]
[1721.98999023]
[1720.26000977]
[1731.92004395]
[1736.43005371]
[1767.38000488]
[1777.43005371]
[1787.47998047]
[1757.51000977]
[1785.66003418]
[1765.72998047]
[1762.17004395]
[1780.7800293 ]
[1761.32995605]
[1777.07995605]
[1762.70996094]
[1779.98999023]
[1776.66003418]
[1791.43994141]
```

```
[1804.66003418]
[1801.70996094]
[1795.77001953]
[1788.19995117]
[1785.88000488]
[1771.65002441]
[1778.        ]
[1753.10998535]
[1754.59997559]
[1739.48999023]
[1752.5300293 ]
[1752.79003906]
[1745.5300293 ]
[1734.70996094]
[1745.7199707 ]
[1773.83996582]
[1796.93994141]
[1818.51000977]
[1800.80004883]
[1781.59997559]
[1769.95996094]
[1760.68994141]
[1740.47998047]
[1751.59997559]
[1749.51000977]
[1739.20996094]
[1748.7199707 ]
[1760.32995605]
[1760.93994141]
[1769.20996094]
[1790.66003418]
[1784.0300293 ]
[1792.2800293 ]
[1786.5       ]
[1793.        ]
[1789.20996094]
[1868.77001953]
[1869.80004883]
[1846.89001465]
[1847.83996582]
[1898.01000977]
[1874.9699707 ]
[1902.88000488]
[1906.85998535]
[1891.9699707 ]
[1901.05004883]
[1883.16003418]
[1891.30004883]
```

```
[1869.43994141]
[1862.02001953]
[1877.93994141]
[1864.7199707 ]
[1892.        ]
[1887.45996094]
[1884.57995605]
[1861.64001465]
[1828.33996582]
[1853.25      ]
[1858.        ]
[1870.68005371]
[2008.7199707 ]
[2004.19995117]
[2049.66992188]
[2039.86999512]
[2050.22998047]
[2079.2800293 ]
[2133.90991211]
[2150.80004883]
[2160.        ]
[2149.87011719]
[2134.87011719]
[2155.66992188]
[2170.2199707 ]
[2153.10009766]
[2095.9699707 ]
[2009.29003906]
[1972.73999023]
[1979.58996582]
[1884.30004883]
[1883.75      ]
[1953.94995117]
[1908.98999023]
[1975.82995605]
[1924.0300293 ]
[1901.08996582]
[1800.60998535]
[1891.81994629]
[1820.85998535]
[1676.60998535]
[1785.        ]
[1689.15002441]
[1807.83996582]
[1830.        ]
[1880.93005371]
[1846.08996582]
[1902.82995605]
```

```
 [1940.09997559]
 [1885.83996582]
 [1955.48999023]
 [1900.09997559]
 [1963.94995117]
 [1949.7199707 ]
 [1907.69995117]]
Predicted Result: [[0.2855832 ]
 [0.6932517 ]
 [0.11486241]
 [0.9657291 ]
 [0.31072176]
 [0.55184937]
 [0.4082673 ]
 [0.6724903 ]
 [0.7733001 ]
 [0.5327494 ]
 [0.48186168]
 [0.8179988 ]
 [0.3870478 ]
 [0.43497628]
 [0.6879024 ]
 [0.64359266]
 [0.78955585]
 [0.68042445]
 [0.8323975 ]
 [0.743834  ]
 [0.7754506 ]
 [0.5105114 ]
 [0.6403891 ]
 [0.67630565]
 [0.627643  ]
 [0.83540535]
 [0.64972615]
 [0.6139797 ]
 [0.43290508]
 [0.56504965]
 [0.3572133 ]
 [0.654306  ]
 [0.8610606 ]
 [0.72745436]
 [0.6959712 ]
 [0.36353353]
 [0.5425327 ]
 [0.82767665]
 [0.62131584]
 [0.5144887 ]
 [0.83751565]
```

```
[0.5182816 ]
[0.3536912 ]
[0.2728471 ]
[0.46613446]
[0.54008496]
[0.74025464]
[0.39936724]
[0.62650985]
[0.6361414 ]
[0.8204444 ]
[0.6603745 ]
[0.8314741 ]
[0.5950399 ]
[0.968202  ]
[0.64540637]
[0.36033395]
[0.433249  ]
[0.61360246]
[0.5676944 ]
[0.49052152]
[0.96280336]
[0.65609086]
[0.49717805]
[0.5294765 ]
[0.7286799 ]
[0.6338401 ]
[0.66594386]
[0.6357785 ]
[0.5702662 ]
[0.03351967]
[0.6497111 ]
[0.79523104]
[0.51862943]
[0.6279361 ]
[0.76464164]
[0.6429674 ]
[0.4111479 ]
[0.55673313]
[0.5425962 ]
[0.8279141 ]
[0.49815127]
[0.03476203]
[0.5900992 ]
[0.8864521 ]
[0.09863628]
[0.7126748 ]
[0.65746206]
[0.74258155]
```

```
[0.90729547]
[0.6200556 ]
[0.66974586]
[0.5727356 ]
[0.27494574]
[0.5266695 ]
[0.29566348]
[0.7073396 ]
[0.48245564]
[0.638438  ]
[0.66750765]
[0.33308324]
[0.26134804]
[0.5853533 ]
[0.64819765]
[0.53951174]
[0.74776924]
[0.6149692 ]
[0.7312823 ]
[0.50326824]
[0.72094554]
[0.4249293 ]
[0.6920904 ]
[0.8230093 ]
[0.47613034]
[0.6252593 ]
[0.44434765]
[0.79942536]
[0.6544408 ]
[0.74022096]
[0.7024729 ]
[0.71770906]
[0.60662913]
[0.6442797 ]
[0.66768706]
[0.83527493]
[0.68372095]
[0.5481428 ]
[0.266255  ]
[0.67510855]
[0.48929402]
[0.7081455 ]
[0.74785197]
[0.51226413]
[0.4660859 ]
[0.5656697 ]
[0.50678515]
[0.75979525]
```

```
[0.6774312 ]
[0.07190609]
[0.75918   ]
[0.5301541 ]
[0.8072131 ]
[0.6802184 ]
[0.8392882 ]
[0.5229217 ]
[0.63754916]
[0.6749865 ]
[0.74048555]
[0.6879262 ]
[0.79205674]
[0.4010593 ]
[0.54450774]
[0.43771818]
[0.6268635 ]
[0.41053638]
[0.75996625]
[0.49385333]
[0.28816196]
[0.7532164 ]
[0.56865513]
[0.6892993 ]
[0.2809228 ]
[0.8245723 ]
[0.03365146]
[0.39626718]
[0.3561926 ]
[0.23806381]
[0.8430318 ]
[0.3809031 ]
[0.5642816 ]
[0.5115568 ]
[0.24014309]
[0.77747333]
[0.73828566]
[0.6454984 ]
[0.3926893 ]
[0.36707324]
[0.41859576]
[0.41956702]
[0.6915831 ]
[0.820791  ]
[0.46012166]
[0.41362828]
[0.6313999 ]
[0.52806234]
```

```
[0.8426588 ]
[0.60659134]
[0.53508055]
[0.8510951 ]
[0.45261952]
[0.7112566 ]
[0.83965105]
[0.7277943 ]
[0.2962789 ]
[0.7373495 ]
[0.6193105 ]
[0.40314916]
[0.611792  ]
[0.5012904 ]
[0.7174273 ]
[0.20423019]
[0.1396431 ]
[0.68559206]
[0.68420464]
[0.29016855]
[0.543121  ]
[0.5181025 ]
[0.6832544 ]
[0.48341754]
[0.42218408]
[0.8290101 ]
[0.5304892 ]
[0.66135633]
[0.62741363]
[0.3048719 ]
[0.5599156 ]
[0.7402242 ]
[0.4813547 ]
[0.6850115 ]
[0.610384  ]
[0.46984687]
[0.53171694]
[0.6876874 ]
[0.45528957]
[0.5850327 ]
[0.33423963]
[0.6949742 ]
[0.6705627 ]
[0.40776846]
[0.5729445 ]
[0.5536664 ]
[0.29641682]
[0.675075  ]
```

```
[0.28049913]
[0.77778363]
[0.5046068 ]
[0.4410262 ]
[0.8320779 ]
[0.35679325]
[0.4253748 ]
[0.6192654 ]
[0.48877773]
[0.65173817]
[0.56613827]
[0.7530913 ]
[0.36346298]
[0.34326872]
[0.03271864]
[0.69766146]
[0.28252858]
[0.67084813]
[0.72746956]
[0.29704812]
[0.31984004]
[0.46020266]
[0.96040106]
[0.66835535]
[0.8312869 ]
[0.99994314]
[0.08431134]
[0.6161076 ]
[0.43116346]
[0.6143327 ]
[0.43357217]
[0.5059161 ]
[0.5173664 ]
[0.74746394]
[0.98746014]
[0.66158164]
[0.68462265]
[0.75420964]
[0.78796136]
[0.73522353]
[0.47708437]
[0.77128255]
[0.1783234 ]
[0.69776845]
[0.6511602 ]
[0.62074614]
[0.64799905]
[0.49474305]
```

```
[0.2967616 ]
[0.7431356 ]
[0.74487454]
[0.4901115 ]
[0.27181947]
[0.03882899]
[0.61800885]
[0.6165613 ]
[0.81917286]
[0.7420243 ]
[0.66470194]
[0.45284575]
[0.45551774]
[0.49335894]
[0.5916759 ]
[0.44901386]
[0.21313827]
[0.78726524]
[0.3640695 ]
[0.5231787 ]
[0.7679124 ]
[0.63115066]
[0.74548113]
[0.54403174]
[0.37392822]
[0.4592506 ]
[0.6407714 ]
[0.5121368 ]
[0.77215594]
[0.62855893]
[0.8071449 ]
[0.5052591 ]
[0.72372055]
[0.554204  ]
[0.6175524 ]
[0.82507014]
[0.62710875]
[0.7832123 ]
[0.61792254]
[0.77428526]
[0.19468331]
[0.63204384]
[0.537225  ]
[0.82301545]
[0.69425523]
[0.79769874]
[0.679635  ]
[0.6288698 ]
```

```
[0.36582807]
[0.59450823]
[0.85387015]
[0.45066968]
[0.8035729 ]
[0.84241766]
[0.57800794]
[0.7434015 ]
[0.6211376 ]
[0.6259845 ]
[0.6436701 ]
[0.6233378 ]
[0.7391274 ]
[0.39862457]
[0.77377486]
[0.5854329 ]
[0.84408116]
[0.7028254 ]
[0.29057187]
[0.45108685]
[0.73474723]
[0.40534562]
[0.43936995]
[0.72896415]
[0.6658514 ]
[0.40959337]
[0.3004585 ]
[0.65894705]
[0.6891302 ]
[0.3244134 ]
[0.5076535 ]
[0.7911937 ]
[0.73439825]
[0.336502  ]
[0.83899975]
[0.7150979 ]
[0.49467248]
[0.64277875]
[0.7667155 ]
[0.695913  ]
[0.6341802 ]
[0.5281434 ]
[0.7302926 ]
[0.5245414 ]
[0.7670664 ]
[0.7001156 ]
[0.63375   ]
[0.6432688 ]
```

```
[0.7029854 ]
[0.69058675]
[0.03302487]
[0.03789589]
[0.69234943]
[0.31514862]
[0.51671803]
[0.6856072 ]
[0.3905798 ]
[0.66609925]
[0.4378145 ]
[0.4362541 ]
[0.6479782 ]
[0.12942426]
[0.24554   ]
[0.479668  ]
[0.62845266]
[0.76507133]
[0.8216566 ]
[0.34690818]
[0.50619316]
[0.7451006 ]
[0.74475145]
[0.35787442]
[0.3174025 ]
[0.4818124 ]
[0.6257971 ]
[0.49329054]
[0.49541482]
[0.7923345 ]
[0.7401109 ]
[0.7391014 ]
[0.7098096 ]
[0.66529685]
[0.49390063]
[0.2502883 ]
[0.8462622 ]
[0.52813536]
[0.32649347]
[0.6177884 ]
[0.59463555]
[0.7942935 ]
[0.7534044 ]
[0.6625885 ]
[0.58335626]
[0.8998822 ]
[0.7452588 ]
[0.69192827]
```

```
[0.25796077]
[0.61589986]
[0.5238477 ]
[0.5310627 ]
[0.541687  ]
[0.15230486]
[0.67416   ]
[0.4562101 ]
[0.64385676]
[0.60931635]
[0.49613377]
[0.02863361]
[0.4362521 ]
[0.66406333]
[0.32177195]
[0.35414615]
[0.74098873]
[0.66322863]
[0.64249146]
[0.6898375 ]
[0.7388159 ]
[0.84683394]
[0.4668781 ]
[0.62975395]
[0.2785108 ]
[0.6402438 ]
[0.67883337]
[0.6956201 ]
[0.62847745]
[0.84540224]
[0.7522449 ]
[0.5076015 ]
[0.4485077 ]
[0.4977571 ]
[0.5850896 ]
[0.9347395 ]
[0.26895973]
[0.98765475]
[0.6907323 ]
[0.6143328 ]
[0.512661  ]
[0.6769247 ]
[0.13883156]
[0.0281591 ]
[0.7016674 ]
[0.59540796]
[0.7860996 ]
[0.63817763]
```

```
[0.6250788 ]
[0.6403114 ]
[0.04751558]
[0.7565215 ]
[0.22652052]
[0.7757872 ]
[0.5254519 ]
[0.7702571 ]
[0.65841633]
[0.7436378 ]
[0.67981815]
[0.35096237]
[0.8487401 ]
[0.56294024]
[0.70194435]
[0.647908  ]
[0.80754876]
[0.6754747 ]
[0.45034817]
[0.4598984 ]
[0.545737  ]
[0.8441521 ]
[0.0777718 ]
[0.4104322 ]
[0.67503685]
[0.81409097]
[0.72481817]
[0.8090874 ]
[0.44912818]
[0.5828578 ]
[0.6648059 ]
[0.64114934]
[0.4707133 ]
[0.6347388 ]
[0.639168  ]
[0.6327965 ]
[0.4751758 ]
[0.6247164 ]
[0.70876217]
[0.7972896 ]
[0.05888995]
[0.6366228 ]
[0.66864324]
[0.61669457]
[0.1382289 ]
[0.6082579 ]
[0.6457496 ]
[0.6461952 ]
```

```
[0.4674068 ]
[0.63969517]
[0.639392  ]
[0.5209262 ]
[0.81260693]
[0.5980431 ]
[0.34071648]
[0.71412873]
[0.45417082]
[0.02600437]
[0.8456366 ]
[0.7943667 ]
[0.48782352]
[0.6336312 ]
[0.604554  ]
[0.33808568]
[0.6370814 ]
[0.51194   ]
[0.5736482 ]
[0.7656176 ]
[0.7440399 ]
[0.61957246]
[0.8029887 ]
[0.75594795]
[0.80369574]
[0.6059953 ]
[0.49216273]
[0.5810938 ]
[0.7095976 ]
[0.6712855 ]
[0.6613779 ]
[0.5090179 ]
[0.7129188 ]
[0.753641  ]
[0.6651983 ]
[0.95012164]
[0.48626083]
[0.7415267 ]
[0.6289097 ]
[0.80254245]
[0.749521  ]
[0.63031524]
[0.45758507]
[0.6757301 ]
[0.58425546]
[0.7136602 ]
[0.6161767 ]]
```

ini merupakan nilai ground dan prediksi resultnya.

Tuning Model

```
[ ]: #tuning model
modifikasi_predictions1 = model_modif1.predict(x_test)

# Matric Eval
baseline_rmse = np.sqrt(mean_squared_error(y_test, modifikasi_predictions1))
baseline_mae = mean_absolute_error(y_test, modifikasi_predictions1)
baseline_mape = mean_absolute_percentage_error(y_test, modifikasi_predictions1)

print("Modify Model:")
print("RMSE:", baseline_rmse)
print("MAE:", baseline_mae)
print("MAPE:", baseline_mape)
```

```
24/24 [==============================] - 4s 116ms/step
Modify Model:
RMSE: 0.0406121328878552
MAE: 0.029742581463814494
MAPE: 218715763097.16583
```

pada hasil Evaluasi terdapat baseline dan juga Tuning model. baseline dari arsitektur Transformer menggunakan satu layer Conv1D pada bagian Feed Forward dengan Activation function menggunakan ReLU dan bagian node perceptron pada output disesuaikan dengan horizon datanya. Baseline juga terdapat layer multi-Head Attention digunkan dalam lapisan encoder dan decoder untuk memproses lebih baik dari input dan memperoleh hasil yang lebih baik. Pada Baseline Model didapat hasil yang sedikit lebih tinggi daripada tuning model, itu berarti Tuning model sudah berjalan dengan baik dan mempunyai arsitektur yang lebih baik daripada baseline. karena Tuning dapat mengalahkan Baseline model Arsitektur Attention.

Dengan Tuning model mendapatkan hasil yang lebih bagus. Model tuning berhasil pada percobaan modifikasi ke-1 dengan mengubah Pada percobaan modifikasi data Cisco yang pertama, perubahan ini yang saya lakukan :
- saya membuang dropout pada feed forward - menambahkan attention layer berupa: 1. dengan normalisasi dan embedding layer untuk menjaga konsistensi distribusi data. 2. menambahkan Attention Layer, kemudian output tersebut dimasukkan ke dropout layer. 3. saya juga menambahkan residual connection menggunakan + untuk menyimpan data asli dan membantu aliran gradien selama proses pembelajaran.

referensi

Narang, S., Chung, H. W., Tay, Y., Fedus, W., Fevry, T., Matena, M., ... & Raffel, C. (2021). Do transformer modifications transfer across implementations and applications?. arXiv preprint arXiv:2102.11972.

Baseline dengan modifikasi arsitektur pertama mendapatkan hasil evaluasi yang lebih baik dibandingkan dengan baseline. ada beberapa teori yang membuktikan bahwa modifikasi ke-1 mendapatkan evaluasi terbaik dalam kedua dataset tersebut.

pada paper Hernández et.al., dikatakan dengan jaringan dengan enam lapisan identik yang memiliki perhatian sendiri (self-attention) dan jaringan feed-forward, dengan sub-lapisan ketiga yang melakukan perhatian terhadap output encoder, ini menggambarkan model Transformer yang populer dalam pemrosesan bahasa alami (natural language processing).

Dengan menggunakan Attention, jaringan dapat menyesuaikan bobot atau relevansi setiap elemen dalam urutan data inputnya. Ini memungkinkan jaringan untuk memberikan lebih banyak perhatian pada informasi yang penting dan mengabaikan informasi yang kurang relevan atau noise. Dengan itu penggunaan attention dalam jaringan saraf dapat meningkatkan kemampuan jaringan untuk menangkap pola-pola yang lebih kompleks dan mendasar dalam data inputnya.

rujukan : Hernández, A., & Amigó, J. M. (2021). Attention mechanisms and their applications to complex systems. Entropy, 23(3), 283.

5.0.3 E. Link Video : https://youtu.be/DpIhEsz2bsQ