Attention based Model

February 19, 2021

[1]:
```
import numpy as np
import pandas as pd
import os, datetime
import tensorflow as tf
from tensorflow.keras.models import *
from tensorflow.keras.layers import *
from sklearn.preprocessing import MinMaxScaler

import matplotlib.pyplot as plt
plt.style.use('seaborn')

import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
batch_size = 128
seq_len = 128

d_k = 64
d_v = 64
h = 8
d_ff = 2048
```

[3]:
```
stock = pd.read_csv('C:\Jupyter_Project\Hanyang_Securities_F.csv')
df = stock.dropna()

df['Volume'].replace(to_replace=0, method='ffill', inplace=True)
df.sort_values('Date', inplace=True)
df.tail()
```

[3]:

	Date	Open	High	Low	Close	Adj Close	Volume
5181	2021-02-01	9200	9480	9100	9380	9380.0	81355
5182	2021-02-02	9460	9810	9460	9700	9700.0	105755
5183	2021-02-03	9850	10200	9800	9990	9990.0	170966
5184	2021-02-04	10100	10200	9940	10150	10150.0	133504
5185	2021-02-05	10200	10800	10150	10650	10650.0	247224

[4]:
```
ratio = df['Adj Close']/df['Close']
ratio
```

[4]:
```
0       0.231324
1       0.231324
2       0.231324
3       0.231324
4       0.231324
          ...
5181    1.000000
5182    1.000000
5183    1.000000
5184    1.000000
5185    1.000000
Length: 5186, dtype: float64
```

[5]:
```
df['Adj Open'] = df['Open']*ratio
df['Adj High'] = df['High']*ratio
df['Adj Low'] = df['Low']*ratio
```

[6]:
```
df.drop(['Open','High','Low','Close'], axis=1, inplace=True)

df
```

[6]:

	Date	Adj Close	Volume	Adj Open	Adj High	Adj Low
0	2000-01-04	1619.266357	56800	1457.339721	1642.398734	1457.339721
1	2000-01-05	1549.868774	52100	1549.868774	1642.398253	1529.049641
2	2000-01-06	1457.339844	64900	1619.266493	1619.266493	1445.773655
3	2000-01-07	1473.532349	61800	1468.905874	1526.736814	1457.339686
4	2000-01-10	1503.603882	56100	1529.049486	1549.868617	1457.339147
...	...	...	...	...	...	...
5181	2021-02-01	9380.000000	81355	9200.000000	9480.000000	9100.000000
5182	2021-02-02	9700.000000	105755	9460.000000	9810.000000	9460.000000
5183	2021-02-03	9990.000000	170966	9850.000000	10200.000000	9800.000000
5184	2021-02-04	10150.000000	133504	10100.000000	10200.000000	9940.000000
5185	2021-02-05	10650.000000	247224	10200.000000	10800.000000	10150.000000

```
[5186 rows x 6 columns]
```

[7]:
```
df.rename(columns={'Date':'Date','Adj Open':'Open', 'Adj High':'High', 'Adj Low':
 ↪'Low', 'Adj Close':'Close'}, inplace=True)
```

[8]:
```
df = df[['Date', 'Open', 'High', 'Low', 'Close', 'Volume']]

df.head()
```

[8]:

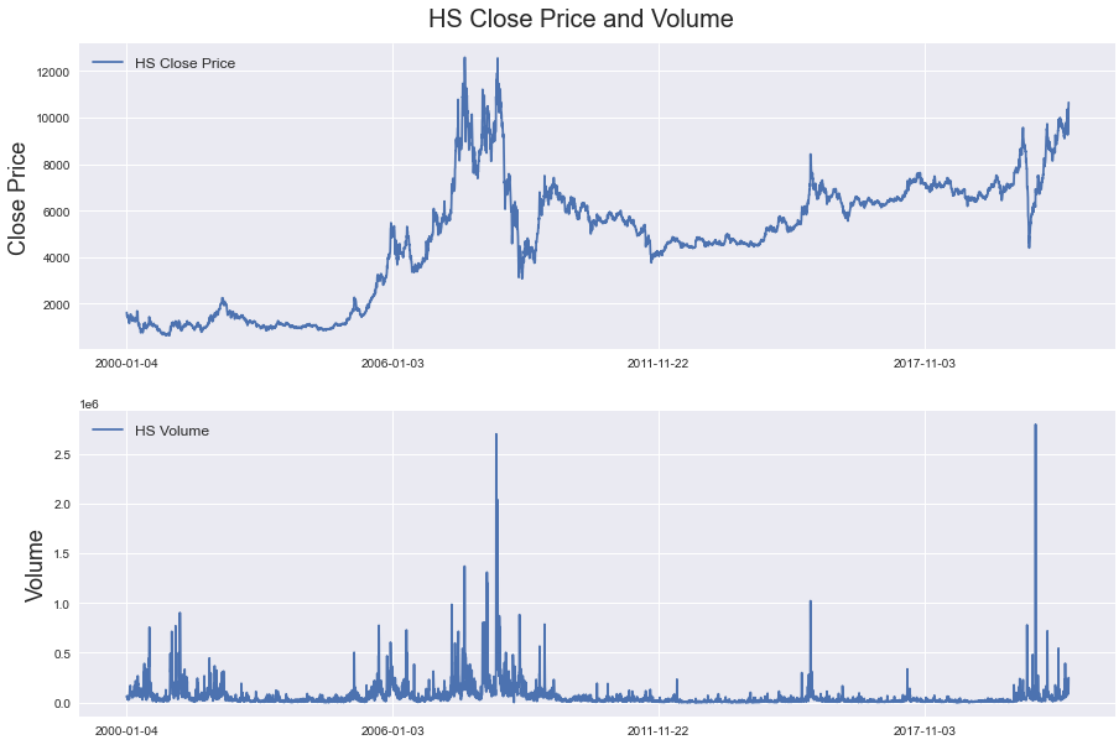
	Date	Open	High	Low	Close	Volume
0	2000-01-04	1457.339721	1642.398734	1457.339721	1619.266357	56800
1	2000-01-05	1549.868774	1642.398253	1529.049641	1549.868774	52100
2	2000-01-06	1619.266493	1619.266493	1445.773655	1457.339844	64900
3	2000-01-07	1468.905874	1526.736814	1457.339686	1473.532349	61800
4	2000-01-10	1529.049486	1549.868617	1457.339147	1503.603882	56100

[9]:
```
df.index.values
```

[9]:
```
array([   0,    1,    2, ..., 5183, 5184, 5185], dtype=int64)
```

[10]:
```
fig = plt.figure(figsize=(15,10))
st = fig.suptitle("HS Close Price and Volume", fontsize=20)
st.set_y(0.92)

ax1 = fig.add_subplot(211)
ax1.plot(df['Close'], label='HS Close Price')
ax1.set_xticks(range(0, df.shape[0], 1464))
ax1.set_xticklabels(df['Date'].loc[::1464])
ax1.set_ylabel('Close Price', fontsize=18)
ax1.legend(loc="upper left", fontsize=12)

ax2 = fig.add_subplot(212)
ax2.plot(df['Volume'], label='HS Volume')
ax2.set_xticks(range(0, df.shape[0], 1464))
ax2.set_xticklabels(df['Date'].loc[::1464])
ax2.set_ylabel('Volume', fontsize=18)
ax2.legend(loc="upper left", fontsize=12)
```

[10]:
```
<matplotlib.legend.Legend at 0x19e3362d7c0>
```



[11]:
```
df.head()
```

[11]:

	Date	Open	High	Low	Close	Volume
0	2000-01-04	1457.339721	1642.398734	1457.339721	1619.266357	56800
1	2000-01-05	1549.868774	1642.398253	1529.049641	1549.868774	52100
2	2000-01-06	1619.266493	1619.266493	1445.773655	1457.339844	64900
3	2000-01-07	1468.905874	1526.736814	1457.339686	1473.532349	61800
4	2000-01-10	1529.049486	1549.868617	1457.339147	1503.603882	56100

[12]:
```
scaler = MinMaxScaler()
scale_cols = ['Open', 'High', 'Low', 'Close', 'Volume']
df_scaled = scaler.fit_transform(df[scale_cols])

# 정규화가 완료된 데이터들은 pandas dataframe으로 변환합니다
# pandas는 시계열 자료에 대한 다양한 기능을 제공하여 LSTM에서 사용하는 window를 만들
때 유용합니다

df_scaled = pd.DataFrame(df_scaled)
df_scaled.columns = scale_cols

print(df_scaled)
```

	Open	High	Low	Close	Volume
0	0.069093	0.078420	0.072692	0.082280	0.020301

```
1     0.076891  0.078420  0.078924  0.076473  0.018620
2     0.082740  0.076587  0.071686  0.068730  0.023197
3     0.070068  0.069256  0.072692  0.070085  0.022088
4     0.075136  0.071089  0.072692  0.072601  0.020050
...        ...       ...       ...       ...       ...
5181  0.721622  0.699387  0.736878  0.731697  0.029080
5182  0.743534  0.725532  0.768164  0.758474  0.037804
5183  0.776402  0.756432  0.797711  0.782742  0.061119
5184  0.797472  0.756432  0.809878  0.796130  0.047725
5185  0.805899  0.803969  0.828128  0.837970  0.088383

[5186 rows x 5 columns]
```

[13]:
```
df_scaled.describe()
```

[13]:

	Open	High	Low	Close	Volume
count	5186.000000	5186.000000	5186.000000	5186.000000	5186.000000
mean	0.353266	0.336558	0.359212	0.350731	0.021491
std	0.216670	0.207100	0.219397	0.214807	0.043042
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.112288	0.107498	0.113192	0.111441	0.003833
50%	0.388059	0.367230	0.394341	0.384704	0.009001
75%	0.504751	0.478909	0.514688	0.500902	0.022008
max	1.000000	1.000000	1.000000	1.000000	1.000000

[14]:
```
times = sorted(df_scaled.index.values)
last_20pct = sorted(df_scaled.index.values)[-int(0.2*len(times))]
last_40pct = sorted(df_scaled.index.values)[-int(0.4*len(times))]
```

[15]:
```
df_train = df_scaled[(df_scaled.index < last_40pct)]  # Training data are 80% of␣
 ↪total data
df_valid = df_scaled[(df_scaled.index >= last_40pct) & (df_scaled.index <␣
 ↪last_20pct)]
df_test = df_scaled[(df_scaled.index >= last_20pct)]

# print proportions
print('train: {}% | validation: {}% | test {}%'.format(round(len(df_train)/
 ↪len(df_scaled),2),
                                                       round(len(df_valid)/
 ↪len(df_scaled),2),
                                                       round(len(df_test)/
 ↪len(df_scaled),2)))
```

```
train: 0.6% | validation: 0.2% | test 0.2%
```

[16]:
```
train_data = df_train.values
valid_data = df_valid.values
test_data = df_test.values
```

```
print('Training data shape: {}'.format(train_data.shape))
print('Validation data shape: {}'.format(valid_data.shape))
print('Test data shape: {}'.format(test_data.shape))

df_train.head()
```

```
Training data shape: (3112, 5)
Validation data shape: (1037, 5)
Test data shape: (1037, 5)
```

[16]:

	Open	High	Low	Close	Volume
0	0.069093	0.078420	0.072692	0.082280	0.020301
1	0.076891	0.078420	0.078924	0.076473	0.018620
2	0.082740	0.076587	0.071686	0.068730	0.023197
3	0.070068	0.069256	0.072692	0.070085	0.022088
4	0.075136	0.071089	0.072692	0.072601	0.020050

[17]:
```
fig = plt.figure(figsize=(15,12))
st = fig.suptitle("Data Separation", fontsize=20)
st.set_y(0.95)

##############################################################################

ax1 = fig.add_subplot(211)
ax1.plot(np.arange(train_data.shape[0]), df_train['Close'], label='Training␣
 ↪data')

ax1.plot(np.arange(train_data.shape[0],
                   train_data.shape[0]+valid_data.shape[0]), df_valid['Close'],␣
 ↪label='Validation data')

ax1.plot(np.arange(train_data.shape[0]+valid_data.shape[0],
                   train_data.shape[0]+valid_data.shape[0]+test_data.shape[0]),␣
 ↪df_test['Close'], label='Test data')
ax1.set_xlabel('Date')
ax1.set_ylabel('Normalized Closing Returns')
ax1.set_title("Close Price", fontsize=18)
ax1.legend(loc="best", fontsize=12)

##############################################################################

ax2 = fig.add_subplot(212)
ax2.plot(np.arange(train_data.shape[0]), df_train['Volume'], label='Training␣
 ↪data')

ax2.plot(np.arange(train_data.shape[0],
```

```
                    train_data.shape[0]+valid_data.shape[0]), df_valid['Volume'],␣
 ↪label='Validation data')

ax2.plot(np.arange(train_data.shape[0]+valid_data.shape[0],
                    train_data.shape[0]+valid_data.shape[0]+test_data.shape[0]),␣
 ↪df_test['Volume'], label='Test data')
ax2.set_xlabel('Date')
ax2.set_ylabel('Normalized Volume Changes')
ax2.set_title("Volume", fontsize=18)
ax2.legend(loc="best", fontsize=12)
```

[17]: <matplotlib.legend.Legend at 0x19e33745fd0>

[18]:
```
# Training data
x_train, y_train = [], []
for i in range(seq_len, len(train_data)):
```

```
  x_train.append(train_data[i-seq_len:i]) # Chunks of training data with a␣
 ↪length of 128 df-rows
  y_train.append(train_data[:, 3][i]) #Value of 4th column (Close Price) of␣
 ↪df-row 128+1
x_train, y_train = np.array(x_train), np.array(y_train)

###############################################################################

# Validation data
x_valid, y_valid = [], []
for i in range(seq_len, len(valid_data)):
    x_valid.append(valid_data[i-seq_len:i])
    y_valid.append(valid_data[:, 3][i])
x_valid, y_valid = np.array(x_valid), np.array(y_valid)

###############################################################################

# Test data
x_test, y_test = [], []
for i in range(seq_len, len(test_data)):
    x_test.append(test_data[i-seq_len:i])
    y_test.append(test_data[:, 3][i])
x_test, y_test = np.array(x_test), np.array(y_test)

print('Training set shape', x_train.shape, y_train.shape)
print('Validation set shape', x_valid.shape, y_valid.shape)
print('Testing set shape' ,x_test.shape, y_test.shape)
```

```
Training set shape (2984, 128, 5) (2984,)
Validation set shape (909, 128, 5) (909,)
Testing set shape (909, 128, 5) (909,)
```

[19]:
```
class Time2Vector(Layer):
  def __init__(self, seq_len, **kwargs):
    super(Time2Vector, self).__init__()
    self.seq_len = seq_len

  def build(self, input_shape):
    '''Initialize weights and biases with shape (batch, seq_len)'''
    self.weights_linear = self.add_weight(name='weight_linear',
                                shape=(int(self.seq_len),),
                                initializer='uniform',
                                trainable=True)

    self.bias_linear = self.add_weight(name='bias_linear',
                                shape=(int(self.seq_len),),
                                initializer='uniform',
```

```
                                trainable=True)

    self.weights_periodic = self.add_weight(name='weight_periodic',
                                shape=(int(self.seq_len),),
                                initializer='uniform',
                                trainable=True)

    self.bias_periodic = self.add_weight(name='bias_periodic',
                                shape=(int(self.seq_len),),
                                initializer='uniform',
                                trainable=True)

  def call(self, x):
    '''Calculate linear and periodic time features'''
    x = tf.math.reduce_mean(x[:,:,:4], axis=-1)
    time_linear = self.weights_linear * x + self.bias_linear # Linear time␣
↪feature
    time_linear = tf.expand_dims(time_linear, axis=-1) # Add dimension (batch,␣
↪seq_len, 1)

    time_periodic = tf.math.sin(tf.multiply(x, self.weights_periodic) + self.
↪bias_periodic)
    time_periodic = tf.expand_dims(time_periodic, axis=-1) # Add dimension␣
↪(batch, seq_len, 1)
    return tf.concat([time_linear, time_periodic], axis=-1) # shape = (batch,␣
↪seq_len, 2)

  def get_config(self): # Needed for saving and loading model with custom layer
    config = super().get_config().copy()
    config.update({'seq_len': self.seq_len})
    return config
```

[20]:
```
class Scaled_Dot_Product_Attention(Layer):
  def __init__(self, d_k, d_v):
    super(Scaled_Dot_Product_Attention, self).__init__()
    self.d_k = d_k
    self.d_v = d_v

  def build(self, input_shape):
    self.query = Dense(self.d_k,
                       input_shape=input_shape,
                       kernel_initializer='glorot_uniform',
                       bias_initializer='glorot_uniform')

    self.key = Dense(self.d_k,
                     input_shape=input_shape,
                     kernel_initializer='glorot_uniform',
```

```
                    bias_initializer='glorot_uniform')

    self.value = Dense(self.d_v,
                       input_shape=input_shape,
                       kernel_initializer='glorot_uniform',
                       bias_initializer='glorot_uniform')

  def call(self, inputs): # inputs = (in_seq, in_seq, in_seq)
    q = self.query(inputs[0])
    k = self.key(inputs[1])

    attn_weights = tf.matmul(q, k, transpose_b=True)
    attn_weights = tf.map_fn(lambda x: x/np.sqrt(self.d_k), attn_weights)
    attn_weights = tf.nn.softmax(attn_weights, axis=-1)

    v = self.value(inputs[2])
    attn_out = tf.matmul(attn_weights, v)
    return attn_out

################################################################################

class Multi_Head_Attention(Layer):
  def __init__(self, d_k, d_v, h):
    super(Multi_Head_Attention, self).__init__()
    self.d_k = d_k
    self.d_v = d_v
    self.h = h
    self.attn_heads = list()

  def build(self, input_shape):
    for n in range(self.h):
      self.attn_heads.append(Scaled_Dot_Product_Attention(self.d_k, self.d_v))

    # input_shape[0]=(batch, seq_len, 7), input_shape[0][-1]=7
    self.linear = Dense(input_shape[0][-1],
                        input_shape=input_shape,
                        kernel_initializer='glorot_uniform',
                        bias_initializer='glorot_uniform')

  def call(self, inputs):
    attn = [self.attn_heads[i](inputs) for i in range(self.h)]
    concat_attn = tf.concat(attn, axis=-1)
    multi_linear = self.linear(concat_attn)
    return multi_linear

################################################################################
```

```
class TransformerEncoder(Layer):
  def __init__(self, d_k, d_v, h, d_ff, dropout=0.1, **kwargs):
    super(TransformerEncoder, self).__init__()
    self.d_k = d_k
    self.d_v = d_v
    self.h = h
    self.d_ff = d_ff
    self.attn_heads = list()
    self.dropout_rate = dropout

  def build(self, input_shape):
    self.attn_multi = Multi_Head_Attention(self.d_k, self.d_v, self.h)
    self.attn_dropout = Dropout(self.dropout_rate)
    self.attn_normalize = LayerNormalization(input_shape=input_shape,␣
 ↪epsilon=1e-6)

    self.ff_conv1D_1 = Conv1D(filters=self.d_ff, kernel_size=1,␣
 ↪activation='relu')
    # input_shape[0]=(batch, seq_len, 7), input_shape[0][-1] = 7
    self.ff_conv1D_2 = Conv1D(filters=input_shape[0][-1], kernel_size=1)
    self.ff_dropout = Dropout(self.dropout_rate)
    self.ff_normalize = LayerNormalization(input_shape=input_shape, epsilon=1e-6)

  def call(self, inputs): # inputs = (in_seq, in_seq, in_seq)
    attn_layer = self.attn_multi(inputs)
    attn_layer = self.attn_dropout(attn_layer)
    attn_layer = self.attn_normalize(inputs[0] + attn_layer)

    ff_layer = self.ff_conv1D_1(attn_layer)
    ff_layer = self.ff_conv1D_2(ff_layer)
    ff_layer = self.ff_dropout(ff_layer)
    ff_layer = self.ff_normalize(inputs[0] + ff_layer)
    return ff_layer

  def get_config(self): # Needed for saving and loading model with custom layer
    config = super().get_config().copy()
    config.update({'d_k': self.d_k,
                   'd_v': self.d_v,
                   'h': self.h,
                   'd_ff': self.d_ff,
                   'attn_heads': self.attn_heads,
                   'dropout_rate': self.dropout_rate})
    return config
```

[21]:
```
# val_loss가 10회 같을 시 early_stop, batch_size(=K)는 K문제 풀고 답보고 하는 식
# 위에서 모델을 구성한 후 compile 메서드를 호출하여 학습과정을 설정합니다
# optimizer : 훈련 과정을 설정한다
```

```
# loss : 최적화 과정에서 최소화될 손실 함수(loss function)을 설정합니다
# metrics : 훈련을 모니터링하기 위해 사용됩니다
# validation_data = 검증 데이터를 사용합니다. 각 에포크마다 정확도도 함께 출력됩니다
# 이 정확도는 훈련이 잘 되고 있는지를 보여줄 뿐이며 실제로 모델이 검증데이터를 학습하지는 않습니다
# 검증 데이터의 loss가 낮아지다가 높아지기 시작하면 overfitting의 신호입니다
# verbose / 0 : 출력 없음 / 1 : 훈련 진행도 보여주는 진행 막대 보여줌 / 2 : 미니 배치마다 손실 정보 출력

from numpy import array
from keras.models import Sequential
from keras.layers import Dense
from keras import backend as K

def RMSE(y_true, y_pred):
    return K.sqrt(K.mean(K.square(y_pred - y_true)))

def soft_acc(y_true, y_pred):
    return K.mean(K.equal(K.round(y_true), K.round(y_pred)))

def MPE(y_true, y_pred):
    return K.mean((y_true - y_pred) / y_true) * 100

def MSLE(y_true, y_pred):
    return K.mean(K.square(K.log(y_true+1) - K.log(y_pred+1)), axis=-1)

def RMSLE(y_true, y_pred):
    return K.sqrt(K.mean(K.square(K.log(y_true+1) - K.log(y_pred+1)), axis=-1))

def R2(y_true, y_pred):
    SS_res = K.sum(K.square(y_true - y_pred))
    SS_tot = K.sum(K.square(y_true - K.mean(y_true)))
    return ( 1 - SS_res/(SS_tot + K.epsilon()))
```

[22]:
```
from keras.models import Sequential
from keras.layers import Dense
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras import optimizers
from keras.optimizers import Adam




def create_model():
  '''Initialize time and transformer layers'''
  time_embedding = Time2Vector(seq_len)
  attn_layer1 = TransformerEncoder(d_k, d_v, h, d_ff)
```

```
  attn_layer2 = TransformerEncoder(d_k, d_v, h, d_ff)
  attn_layer3 = TransformerEncoder(d_k, d_v, h, d_ff)
  attn_layer4 = TransformerEncoder(d_k, d_v, h, d_ff)
  attn_layer5 = TransformerEncoder(d_k, d_v, h, d_ff)
  attn_layer6 = TransformerEncoder(d_k, d_v, h, d_ff)

  '''Construct model'''
  in_seq = Input(shape=(seq_len, 5))
  x = time_embedding(in_seq)
  x = Concatenate(axis=-1)([in_seq, x])
  x = attn_layer1((x, x, x))
  x = attn_layer2((x, x, x))
  x = attn_layer3((x, x, x))
  x = attn_layer4((x, x, x))
  x = attn_layer5((x, x, x))
  x = attn_layer6((x, x, x))
  x = GlobalAveragePooling1D(data_format='channels_first')(x)
  x = Dropout(0.1)(x)
  x = Dense(64, activation='relu')(x)
  x = Dropout(0.1)(x)
  out = Dense(1, activation='linear')(x)



  model = Model(inputs=in_seq, outputs=out)
  model.compile(loss = RMSE, optimizer=Adam(lr=0.001, beta_1=0.9, beta_2=0.999),␣
 ↪metrics=[soft_acc, 'mse', 'mae', RMSE, 'mape', MPE, MSLE, RMSLE, R2])
  return model


model = create_model()
model.summary()
filename = os.path.join('tmp', 'checkpointer.ckpt')

callback = tf.keras.callbacks.ModelCheckpoint(filename,
                                              monitor='val_loss',
                                              save_best_only=True, verbose=1)
early_stop = EarlyStopping(monitor='val_loss', patience=10)

history = model.fit(x_train, y_train,
                    batch_size=batch_size,
                    epochs=200,
                    callbacks=[callback, early_stop],
                    validation_data=(x_valid, y_valid))
```

```
#################################################################################
'''Calculate predictions and metrics'''

#Calculate predication for training, validation and test data
train_pred = model.predict(x_train)
valid_pred = model.predict(x_valid)
test_pred = model.predict(x_test)

#Print evaluation metrics for all datasets
train_evaluate = model.evaluate(x_train, y_train, verbose=0)
valid_evaluate = model.evaluate(x_valid, y_valid, verbose=0)
test_evaluate = model.evaluate(x_test, y_test, verbose=0)
```

```
WARNING:tensorflow:AutoGraph could not transform <bound method Time2Vector.call
of <__main__.Time2Vector object at 0x0000019E3380EC10>> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <bound method Time2Vector.call of
<__main__.Time2Vector object at 0x0000019E3380EC10>> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING:tensorflow:AutoGraph could not transform <bound method
TransformerEncoder.call of <__main__.TransformerEncoder object at
0x0000019E337E0700>> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <bound method TransformerEncoder.call of
<__main__.TransformerEncoder object at 0x0000019E337E0700>> and will run it as-
is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING:tensorflow:AutoGraph could not transform <bound method
```

```
Multi_Head_Attention.call of <__main__.Multi_Head_Attention object at
0x0000019E338910A0>> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <bound method Multi_Head_Attention.call
of <__main__.Multi_Head_Attention object at 0x0000019E338910A0>> and will run it
as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING:tensorflow:AutoGraph could not transform <bound method
Scaled_Dot_Product_Attention.call of <__main__.Scaled_Dot_Product_Attention
object at 0x0000019E338F0BE0>> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <bound method
Scaled_Dot_Product_Attention.call of <__main__.Scaled_Dot_Product_Attention
object at 0x0000019E338F0BE0>> and will run it as-is.
Please report this to the TensorFlow team. When filing the bug, set the
verbosity to 10 (on Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full
output.
Cause: module 'gast' has no attribute 'Index'
To silence this warning, decorate the function with
@tf.autograph.experimental.do_not_convert
Model: "functional_1"
__________________________________________________________________________________
__________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================
==================
input_1 (InputLayer)            [(None, 128, 5)]     0
__________________________________________________________________________________
__________________
time2_vector (Time2Vector)      (None, 128, 2)       512         input_1[0][0]
__________________________________________________________________________________
__________________
concatenate (Concatenate)       (None, 128, 7)       0           input_1[0][0]
```

```
time2_vector[0][0]
__________________________________________________________________________________
__________________
transformer_encoder (Transforme (None, 128, 7)       46634
concatenate[0][0]
concatenate[0][0]
concatenate[0][0]
__________________________________________________________________________________
__________________
transformer_encoder_1 (Transfor (None, 128, 7)       46634
transformer_encoder[0][0]
transformer_encoder[0][0]
transformer_encoder[0][0]
__________________________________________________________________________________
__________________
transformer_encoder_2 (Transfor (None, 128, 7)       46634
transformer_encoder_1[0][0]
transformer_encoder_1[0][0]
transformer_encoder_1[0][0]
__________________________________________________________________________________
__________________
transformer_encoder_3 (Transfor (None, 128, 7)       46634
transformer_encoder_2[0][0]
transformer_encoder_2[0][0]
transformer_encoder_2[0][0]
__________________________________________________________________________________
__________________
transformer_encoder_4 (Transfor (None, 128, 7)       46634
transformer_encoder_3[0][0]
transformer_encoder_3[0][0]
transformer_encoder_3[0][0]
__________________________________________________________________________________
__________________
transformer_encoder_5 (Transfor (None, 128, 7)       46634
transformer_encoder_4[0][0]
transformer_encoder_4[0][0]
transformer_encoder_4[0][0]
__________________________________________________________________________________
__________________
global_average_pooling1d (Globa (None, 128)          0
transformer_encoder_5[0][0]
__________________________________________________________________________________
__________________
dropout (Dropout)               (None, 128)          0
global_average_pooling1d[0][0]
__________________________________________________________________________________
__________________
dense (Dense)                   (None, 64)           8256        dropout[0][0]
```

```
_________________________________________________________________________________
__________________
dropout_1 (Dropout)             (None, 64)           0           dense[0][0]
_________________________________________________________________________________
__________________
dense_1 (Dense)                 (None, 1)            65          dropout_1[0][0]
=================================================================================
==================
Total params: 288,637
Trainable params: 288,637
Non-trainable params: 0
_________________________________________________________________________________
__________________
Epoch 1/200
24/24 [==============================] - ETA: 0s - loss: 0.2268 - soft_acc:
0.8617 - mse: 0.0587 - mae: 0.1750 - RMSE: 0.2236 - mape: 2623.0007 - MPE: -inf
- MSLE: 0.0348 - RMSLE: 0.1388 - R2: -0.1431
Epoch 00001: val_loss improved from inf to 0.03710, saving model to
tmp\checkpointer.ckpt
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\muiiya\lib\site-
packages\tensorflow\python\training\tracking\tracking.py:111:
Model.state_updates (from tensorflow.python.keras.engine.training) is deprecated
and will be removed in a future version.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied
automatically.
WARNING:tensorflow:From C:\ProgramData\Anaconda3\envs\muiiya\lib\site-
packages\tensorflow\python\training\tracking\tracking.py:111: Layer.updates
(from tensorflow.python.keras.engine.base_layer) is deprecated and will be
removed in a future version.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied
automatically.
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 289s 12s/step - loss: 0.2268 -
soft_acc: 0.8617 - mse: 0.0587 - mae: 0.1750 - RMSE: 0.2236 - mape: 2623.0007 -
MPE: -inf - MSLE: 0.0348 - RMSLE: 0.1388 - R2: -0.1431 - val_loss: 0.0371 -
val_soft_acc: 0.6260 - val_mse: 0.0016 - val_mae: 0.0325 - val_RMSE: 0.0368 -
val_mape: 7.9381 - val_MPE: -6.0414 - val_MSLE: 7.5475e-04 - val_RMSLE: 0.0227 -
val_R2: -16.0035
Epoch 2/200
24/24 [==============================] - ETA: 0s - loss: 0.0937 - soft_acc:
0.8973 - mse: 0.0090 - mae: 0.0629 - RMSE: 0.0936 - mape: 12672.7148 - MPE: -inf
- MSLE: 0.0042 - RMSLE: 0.0455 - R2: 0.8194
Epoch 00002: val_loss did not improve from 0.03710
24/24 [==============================] - 238s 10s/step - loss: 0.0937 -
soft_acc: 0.8973 - mse: 0.0090 - mae: 0.0629 - RMSE: 0.0936 - mape: 12672.7148 -
MPE: -inf - MSLE: 0.0042 - RMSLE: 0.0455 - R2: 0.8194 - val_loss: 0.0679 -
```

```
val_soft_acc: 0.6172 - val_mse: 0.0050 - val_mae: 0.0642 - val_RMSE: 0.0660 -
val_mape: 16.6072 - val_MPE: -15.1866 - val_MSLE: 0.0025 - val_RMSLE: 0.0448 -
val_R2: -70.8496
Epoch 3/200
24/24 [==============================] - ETA: 0s - loss: 0.0877 - soft_acc:
0.9121 - mse: 0.0078 - mae: 0.0596 - RMSE: 0.0867 - mape: 16032.8916 - MPE: -inf
- MSLE: 0.0038 - RMSLE: 0.0439 - R2: 0.8460
Epoch 00003: val_loss improved from 0.03710 to 0.03306, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 382s 16s/step - loss: 0.0877 -
soft_acc: 0.9121 - mse: 0.0078 - mae: 0.0596 - RMSE: 0.0867 - mape: 16032.8916 -
MPE: -inf - MSLE: 0.0038 - RMSLE: 0.0439 - R2: 0.8460 - val_loss: 0.0331 -
val_soft_acc: 0.6807 - val_mse: 0.0014 - val_mae: 0.0283 - val_RMSE: 0.0326 -
val_mape: 6.5088 - val_MPE: -2.8766 - val_MSLE: 6.6346e-04 - val_RMSLE: 0.0195 -
val_R2: -7.8659
Epoch 4/200
24/24 [==============================] - ETA: 0s - loss: 0.0834 - soft_acc:
0.9193 - mse: 0.0070 - mae: 0.0558 - RMSE: 0.0834 - mape: 18504.8145 - MPE: -inf
- MSLE: 0.0033 - RMSLE: 0.0410 - R2: 0.8598
Epoch 00004: val_loss improved from 0.03306 to 0.03266, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 641s 27s/step - loss: 0.0834 -
soft_acc: 0.9193 - mse: 0.0070 - mae: 0.0558 - RMSE: 0.0834 - mape: 18504.8145 -
MPE: -inf - MSLE: 0.0033 - RMSLE: 0.0410 - R2: 0.8598 - val_loss: 0.0327 -
val_soft_acc: 0.8760 - val_mse: 0.0014 - val_mae: 0.0274 - val_RMSE: 0.0308 -
val_mape: 6.5035 - val_MPE: -2.4595 - val_MSLE: 6.4578e-04 - val_RMSLE: 0.0191 -
val_R2: -6.6208
Epoch 5/200
24/24 [==============================] - ETA: 0s - loss: 0.0755 - soft_acc:
0.9178 - mse: 0.0058 - mae: 0.0508 - RMSE: 0.0755 - mape: 16281.3096 - MPE: -inf
- MSLE: 0.0027 - RMSLE: 0.0372 - R2: 0.8831
Epoch 00005: val_loss did not improve from 0.03266
24/24 [==============================] - 600s 25s/step - loss: 0.0755 -
soft_acc: 0.9178 - mse: 0.0058 - mae: 0.0508 - RMSE: 0.0755 - mape: 16281.3096 -
MPE: -inf - MSLE: 0.0027 - RMSLE: 0.0372 - R2: 0.8831 - val_loss: 0.0871 -
val_soft_acc: 0.8672 - val_mse: 0.0094 - val_mae: 0.0845 - val_RMSE: 0.0907 -
val_mape: 18.7043 - val_MPE: 19.4065 - val_MSLE: 0.0047 - val_RMSLE: 0.0602 -
val_R2: -114.6790
Epoch 6/200
24/24 [==============================] - ETA: 0s - loss: 0.0705 - soft_acc:
0.9426 - mse: 0.0052 - mae: 0.0488 - RMSE: 0.0700 - mape: 17005.1465 - MPE: -inf
- MSLE: 0.0024 - RMSLE: 0.0358 - R2: 0.8996
Epoch 00006: val_loss did not improve from 0.03266
24/24 [==============================] - 635s 26s/step - loss: 0.0705 -
soft_acc: 0.9426 - mse: 0.0052 - mae: 0.0488 - RMSE: 0.0700 - mape: 17005.1465 -
MPE: -inf - MSLE: 0.0024 - RMSLE: 0.0358 - R2: 0.8996 - val_loss: 0.0753 -
```

```
val_soft_acc: 0.9189 - val_mse: 0.0059 - val_mae: 0.0703 - val_RMSE: 0.0730 -
val_mape: 17.3822 - val_MPE: 15.1897 - val_MSLE: 0.0031 - val_RMSLE: 0.0512 -
val_R2: -51.8530
Epoch 7/200
24/24 [==============================] - ETA: 0s - loss: 0.0551 - soft_acc:
0.9549 - mse: 0.0031 - mae: 0.0382 - RMSE: 0.0553 - mape: 17110.3535 - MPE: -inf
- MSLE: 0.0015 - RMSLE: 0.0282 - R2: 0.9382
Epoch 00007: val_loss did not improve from 0.03266
24/24 [==============================] - 687s 29s/step - loss: 0.0551 -
soft_acc: 0.9549 - mse: 0.0031 - mae: 0.0382 - RMSE: 0.0553 - mape: 17110.3535 -
MPE: -inf - MSLE: 0.0015 - RMSLE: 0.0282 - R2: 0.9382 - val_loss: 0.0508 -
val_soft_acc: 0.9297 - val_mse: 0.0028 - val_mae: 0.0466 - val_RMSE: 0.0472 -
val_mape: 11.4651 - val_MPE: 8.0948 - val_MSLE: 0.0014 - val_RMSLE: 0.0334 -
val_R2: -16.2001
Epoch 8/200
24/24 [==============================] - ETA: 0s - loss: 0.0580 - soft_acc:
0.9551 - mse: 0.0034 - mae: 0.0405 - RMSE: 0.0583 - mape: 13254.2939 - MPE: -inf
- MSLE: 0.0016 - RMSLE: 0.0297 - R2: 0.9319
Epoch 00008: val_loss did not improve from 0.03266
24/24 [==============================] - 656s 27s/step - loss: 0.0580 -
soft_acc: 0.9551 - mse: 0.0034 - mae: 0.0405 - RMSE: 0.0583 - mape: 13254.2939 -
MPE: -inf - MSLE: 0.0016 - RMSLE: 0.0297 - R2: 0.9319 - val_loss: 0.0740 -
val_soft_acc: 0.8672 - val_mse: 0.0065 - val_mae: 0.0713 - val_RMSE: 0.0771 -
val_mape: 16.0316 - val_MPE: 16.6119 - val_MSLE: 0.0032 - val_RMSLE: 0.0507 -
val_R2: -83.1511
Epoch 9/200
24/24 [==============================] - ETA: 0s - loss: 0.0520 - soft_acc:
0.9629 - mse: 0.0027 - mae: 0.0368 - RMSE: 0.0515 - mape: 12823.4424 - MPE: -inf
- MSLE: 0.0013 - RMSLE: 0.0272 - R2: 0.9461
Epoch 00009: val_loss did not improve from 0.03266
24/24 [==============================] - 668s 28s/step - loss: 0.0520 -
soft_acc: 0.9629 - mse: 0.0027 - mae: 0.0368 - RMSE: 0.0515 - mape: 12823.4424 -
MPE: -inf - MSLE: 0.0013 - RMSLE: 0.0272 - R2: 0.9461 - val_loss: 0.0830 -
val_soft_acc: 0.8955 - val_mse: 0.0070 - val_mae: 0.0788 - val_RMSE: 0.0831 -
val_mape: 19.1709 - val_MPE: 18.9563 - val_MSLE: 0.0037 - val_RMSLE: 0.0574 -
val_R2: -81.1436
Epoch 10/200
24/24 [==============================] - ETA: 0s - loss: 0.0487 - soft_acc:
0.9652 - mse: 0.0024 - mae: 0.0339 - RMSE: 0.0482 - mape: 16307.9580 - MPE: -inf
- MSLE: 0.0012 - RMSLE: 0.0251 - R2: 0.9526
Epoch 00010: val_loss improved from 0.03266 to 0.03162, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 769s 32s/step - loss: 0.0487 -
soft_acc: 0.9652 - mse: 0.0024 - mae: 0.0339 - RMSE: 0.0482 - mape: 16307.9580 -
MPE: -inf - MSLE: 0.0012 - RMSLE: 0.0251 - R2: 0.9526 - val_loss: 0.0316 -
val_soft_acc: 0.9316 - val_mse: 0.0012 - val_mae: 0.0285 - val_RMSE: 0.0294 -
val_mape: 6.5946 - val_MPE: 1.3856 - val_MSLE: 5.7546e-04 - val_RMSLE: 0.0198 -
```

```
val_R2: -3.0768
Epoch 11/200
24/24 [==============================] - ETA: 0s - loss: 0.0475 - soft_acc:
0.9674 - mse: 0.0023 - mae: 0.0334 - RMSE: 0.0470 - mape: 13459.5137 - MPE: -inf
- MSLE: 0.0011 - RMSLE: 0.0248 - R2: 0.9552
Epoch 00011: val_loss improved from 0.03162 to 0.02894, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 814s 34s/step - loss: 0.0475 -
soft_acc: 0.9674 - mse: 0.0023 - mae: 0.0334 - RMSE: 0.0470 - mape: 13459.5137 -
MPE: -inf - MSLE: 0.0011 - RMSLE: 0.0248 - R2: 0.9552 - val_loss: 0.0289 -
val_soft_acc: 0.6895 - val_mse: 0.0013 - val_mae: 0.0258 - val_RMSE: 0.0299 -
val_mape: 5.6156 - val_MPE: -4.6685 - val_MSLE: 5.6265e-04 - val_RMSLE: 0.0173 -
val_R2: -10.0029
Epoch 12/200
24/24 [==============================] - ETA: 0s - loss: 0.0484 - soft_acc:
0.9654 - mse: 0.0024 - mae: 0.0340 - RMSE: 0.0483 - mape: 17144.3965 - MPE: -inf
- MSLE: 0.0012 - RMSLE: 0.0253 - R2: 0.9526
Epoch 00012: val_loss improved from 0.02894 to 0.02590, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 824s 34s/step - loss: 0.0484 -
soft_acc: 0.9654 - mse: 0.0024 - mae: 0.0340 - RMSE: 0.0483 - mape: 17144.3965 -
MPE: -inf - MSLE: 0.0012 - RMSLE: 0.0253 - R2: 0.9526 - val_loss: 0.0259 -
val_soft_acc: 0.9414 - val_mse: 7.8856e-04 - val_mae: 0.0228 - val_RMSE: 0.0238
- val_mape: 5.4102 - val_MPE: 2.0389 - val_MSLE: 3.7617e-04 - val_RMSLE: 0.0160
- val_R2: -1.8360
Epoch 13/200
24/24 [==============================] - ETA: 0s - loss: 0.0451 - soft_acc:
0.9633 - mse: 0.0021 - mae: 0.0317 - RMSE: 0.0451 - mape: 14512.6484 - MPE: -inf
- MSLE: 0.0010 - RMSLE: 0.0236 - R2: 0.9581
Epoch 00013: val_loss did not improve from 0.02590
24/24 [==============================] - 695s 29s/step - loss: 0.0451 -
soft_acc: 0.9633 - mse: 0.0021 - mae: 0.0317 - RMSE: 0.0451 - mape: 14512.6484 -
MPE: -inf - MSLE: 0.0010 - RMSLE: 0.0236 - R2: 0.9581 - val_loss: 0.0379 -
val_soft_acc: 0.9277 - val_mse: 0.0015 - val_mae: 0.0347 - val_RMSE: 0.0368 -
val_mape: 8.3771 - val_MPE: 7.8419 - val_MSLE: 7.4104e-04 - val_RMSLE: 0.0248 -
val_R2: -11.0599
Epoch 14/200
24/24 [==============================] - ETA: 0s - loss: 0.0481 - soft_acc:
0.9608 - mse: 0.0023 - mae: 0.0337 - RMSE: 0.0483 - mape: 11775.7695 - MPE: -inf
- MSLE: 0.0012 - RMSLE: 0.0250 - R2: 0.9525
Epoch 00014: val_loss did not improve from 0.02590
24/24 [==============================] - 710s 30s/step - loss: 0.0481 -
soft_acc: 0.9608 - mse: 0.0023 - mae: 0.0337 - RMSE: 0.0483 - mape: 11775.7695 -
MPE: -inf - MSLE: 0.0012 - RMSLE: 0.0250 - R2: 0.9525 - val_loss: 0.0774 -
val_soft_acc: 0.6113 - val_mse: 0.0070 - val_mae: 0.0752 - val_RMSE: 0.0802 -
val_mape: 17.3402 - val_MPE: -17.7820 - val_MSLE: 0.0030 - val_RMSLE: 0.0506 -
```

```
val_R2: -91.7494
Epoch 15/200
24/24 [==============================] - ETA: 0s - loss: 0.0478 - soft_acc:
0.9599 - mse: 0.0023 - mae: 0.0343 - RMSE: 0.0476 - mape: 14198.7881 - MPE: -inf
- MSLE: 0.0011 - RMSLE: 0.0255 - R2: 0.9541
Epoch 00015: val_loss improved from 0.02590 to 0.02304, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 821s 34s/step - loss: 0.0478 -
soft_acc: 0.9599 - mse: 0.0023 - mae: 0.0343 - RMSE: 0.0476 - mape: 14198.7881 -
MPE: -inf - MSLE: 0.0011 - RMSLE: 0.0255 - R2: 0.9541 - val_loss: 0.0230 -
val_soft_acc: 0.9561 - val_mse: 5.8588e-04 - val_mae: 0.0197 - val_RMSE: 0.0212
- val_mape: 4.6564 - val_MPE: 3.2517 - val_MSLE: 2.8183e-04 - val_RMSLE: 0.0138
- val_R2: -1.3618
Epoch 16/200
24/24 [==============================] - ETA: 0s - loss: 0.0473 - soft_acc:
0.9622 - mse: 0.0023 - mae: 0.0340 - RMSE: 0.0467 - mape: 13438.7305 - MPE: -inf
- MSLE: 0.0012 - RMSLE: 0.0255 - R2: 0.9552
Epoch 00016: val_loss did not improve from 0.02304
24/24 [==============================] - 838s 35s/step - loss: 0.0473 -
soft_acc: 0.9622 - mse: 0.0023 - mae: 0.0340 - RMSE: 0.0467 - mape: 13438.7305 -
MPE: -inf - MSLE: 0.0012 - RMSLE: 0.0255 - R2: 0.9552 - val_loss: 0.0516 -
val_soft_acc: 0.9463 - val_mse: 0.0028 - val_mae: 0.0483 - val_RMSE: 0.0487 -
val_mape: 12.2618 - val_MPE: 10.6667 - val_MSLE: 0.0015 - val_RMSLE: 0.0352 -
val_R2: -25.9089
Epoch 17/200
24/24 [==============================] - ETA: 0s - loss: 0.0412 - soft_acc:
0.9627 - mse: 0.0017 - mae: 0.0293 - RMSE: 0.0416 - mape: 14570.4717 - MPE: -inf
- MSLE: 8.7093e-04 - RMSLE: 0.0220 - R2: 0.9652
Epoch 00017: val_loss did not improve from 0.02304
24/24 [==============================] - 863s 36s/step - loss: 0.0412 -
soft_acc: 0.9627 - mse: 0.0017 - mae: 0.0293 - RMSE: 0.0416 - mape: 14570.4717 -
MPE: -inf - MSLE: 8.7093e-04 - RMSLE: 0.0220 - R2: 0.9652 - val_loss: 0.0393 -
val_soft_acc: 0.6719 - val_mse: 0.0025 - val_mae: 0.0358 - val_RMSE: 0.0418 -
val_mape: 7.6163 - val_MPE: -6.2300 - val_MSLE: 0.0011 - val_RMSLE: 0.0237 -
val_R2: -27.8420
Epoch 18/200
24/24 [==============================] - ETA: 0s - loss: 0.0423 - soft_acc:
0.9688 - mse: 0.0018 - mae: 0.0298 - RMSE: 0.0420 - mape: 13042.1807 - MPE: -inf
- MSLE: 8.9842e-04 - RMSLE: 0.0222 - R2: 0.9643
Epoch 00018: val_loss did not improve from 0.02304
24/24 [==============================] - 848s 35s/step - loss: 0.0423 -
soft_acc: 0.9688 - mse: 0.0018 - mae: 0.0298 - RMSE: 0.0420 - mape: 13042.1807 -
MPE: -inf - MSLE: 8.9842e-04 - RMSLE: 0.0222 - R2: 0.9643 - val_loss: 0.0482 -
val_soft_acc: 0.6582 - val_mse: 0.0034 - val_mae: 0.0453 - val_RMSE: 0.0510 -
val_mape: 9.9457 - val_MPE: -10.4411 - val_MSLE: 0.0014 - val_RMSLE: 0.0303 -
val_R2: -40.5891
Epoch 19/200
```

```
24/24 [==============================] - ETA: 0s - loss: 0.0416 - soft_acc:
0.9600 - mse: 0.0018 - mae: 0.0290 - RMSE: 0.0413 - mape: 11454.5098 - MPE: -inf
- MSLE: 8.5614e-04 - RMSLE: 0.0216 - R2: 0.9654
Epoch 00019: val_loss did not improve from 0.02304
24/24 [==============================] - 846s 35s/step - loss: 0.0416 -
soft_acc: 0.9600 - mse: 0.0018 - mae: 0.0290 - RMSE: 0.0413 - mape: 11454.5098 -
MPE: -inf - MSLE: 8.5614e-04 - RMSLE: 0.0216 - R2: 0.9654 - val_loss: 0.0378 -
val_soft_acc: 0.6719 - val_mse: 0.0024 - val_mae: 0.0347 - val_RMSE: 0.0406 -
val_mape: 7.3492 - val_MPE: -7.3014 - val_MSLE: 0.0010 - val_RMSLE: 0.0230 -
val_R2: -27.5816
Epoch 20/200
24/24 [==============================] - ETA: 0s - loss: 0.0394 - soft_acc:
0.9703 - mse: 0.0016 - mae: 0.0276 - RMSE: 0.0395 - mape: 10799.9668 - MPE: -inf
- MSLE: 7.5906e-04 - RMSLE: 0.0205 - R2: 0.9686
Epoch 00020: val_loss improved from 0.02304 to 0.01637, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
24/24 [==============================] - 950s 40s/step - loss: 0.0394 -
soft_acc: 0.9703 - mse: 0.0016 - mae: 0.0276 - RMSE: 0.0395 - mape: 10799.9668 -
MPE: -inf - MSLE: 7.5906e-04 - RMSLE: 0.0205 - R2: 0.9686 - val_loss: 0.0164 -
val_soft_acc: 0.9463 - val_mse: 3.4236e-04 - val_mae: 0.0131 - val_RMSE: 0.0160
- val_mape: 2.9517 - val_MPE: -0.1186 - val_MSLE: 1.5413e-04 - val_RMSLE: 0.0090
- val_R2: -0.8066
Epoch 21/200
24/24 [==============================] - ETA: 0s - loss: 0.0380 - soft_acc:
0.9733 - mse: 0.0015 - mae: 0.0268 - RMSE: 0.0378 - mape: 13139.5977 - MPE: -inf
- MSLE: 7.2898e-04 - RMSLE: 0.0201 - R2: 0.9709
Epoch 00021: val_loss did not improve from 0.01637
24/24 [==============================] - 928s 39s/step - loss: 0.0380 -
soft_acc: 0.9733 - mse: 0.0015 - mae: 0.0268 - RMSE: 0.0378 - mape: 13139.5977 -
MPE: -inf - MSLE: 7.2898e-04 - RMSLE: 0.0201 - R2: 0.9709 - val_loss: 0.0204 -
val_soft_acc: 0.7432 - val_mse: 5.0265e-04 - val_mae: 0.0172 - val_RMSE: 0.0212
- val_mape: 3.9007 - val_MPE: -1.9439 - val_MSLE: 2.2791e-04 - val_RMSLE: 0.0118
- val_R2: -5.0200
Epoch 22/200
24/24 [==============================] - ETA: 0s - loss: 0.0369 - soft_acc:
0.9680 - mse: 0.0014 - mae: 0.0257 - RMSE: 0.0365 - mape: 15602.7832 - MPE: -inf
- MSLE: 6.7898e-04 - RMSLE: 0.0192 - R2: 0.9730
Epoch 00022: val_loss did not improve from 0.01637
24/24 [==============================] - 942s 39s/step - loss: 0.0369 -
soft_acc: 0.9680 - mse: 0.0014 - mae: 0.0257 - RMSE: 0.0365 - mape: 15602.7832 -
MPE: -inf - MSLE: 6.7898e-04 - RMSLE: 0.0192 - R2: 0.9730 - val_loss: 0.1281 -
val_soft_acc: 0.5898 - val_mse: 0.0203 - val_mae: 0.1263 - val_RMSE: 0.1358 -
val_mape: 28.5643 - val_MPE: -30.0079 - val_MSLE: 0.0083 - val_RMSLE: 0.0828 -
val_R2: -311.4964
Epoch 23/200
24/24 [==============================] - ETA: 0s - loss: 0.0451 - soft_acc:
0.9593 - mse: 0.0021 - mae: 0.0327 - RMSE: 0.0449 - mape: 12537.0996 - MPE: -inf
```

```
- MSLE: 0.0010 - RMSLE: 0.0243 - R2: 0.9588
Epoch 00023: val_loss did not improve from 0.01637
24/24 [==============================] - 958s 40s/step - loss: 0.0451 -
soft_acc: 0.9593 - mse: 0.0021 - mae: 0.0327 - RMSE: 0.0449 - mape: 12537.0996 -
MPE: -inf - MSLE: 0.0010 - RMSLE: 0.0243 - R2: 0.9588 - val_loss: 0.0172 -
val_soft_acc: 0.9590 - val_mse: 3.3874e-04 - val_mae: 0.0147 - val_RMSE: 0.0165
- val_mape: 3.5813 - val_MPE: -2.6811 - val_MSLE: 1.6031e-04 - val_RMSLE: 0.0103
- val_R2: -1.8794
Epoch 24/200
24/24 [==============================] - ETA: 0s - loss: 0.0381 - soft_acc:
0.9697 - mse: 0.0015 - mae: 0.0263 - RMSE: 0.0379 - mape: 12730.5645 - MPE: -inf
- MSLE: 7.2096e-04 - RMSLE: 0.0197 - R2: 0.9710
Epoch 00024: val_loss did not improve from 0.01637
24/24 [==============================] - 960s 40s/step - loss: 0.0381 -
soft_acc: 0.9697 - mse: 0.0015 - mae: 0.0263 - RMSE: 0.0379 - mape: 12730.5645 -
MPE: -inf - MSLE: 7.2096e-04 - RMSLE: 0.0197 - R2: 0.9710 - val_loss: 0.0312 -
val_soft_acc: 0.6729 - val_mse: 0.0015 - val_mae: 0.0290 - val_RMSE: 0.0335 -
val_mape: 6.2244 - val_MPE: -6.3531 - val_MSLE: 6.5340e-04 - val_RMSLE: 0.0194 -
val_R2: -18.9229
Epoch 25/200
24/24 [==============================] - ETA: 0s - loss: 0.0403 - soft_acc:
0.9689 - mse: 0.0017 - mae: 0.0280 - RMSE: 0.0405 - mape: 11772.6104 - MPE: -inf
- MSLE: 7.9748e-04 - RMSLE: 0.0208 - R2: 0.9660
Epoch 00025: val_loss did not improve from 0.01637
24/24 [==============================] - 965s 40s/step - loss: 0.0403 -
soft_acc: 0.9689 - mse: 0.0017 - mae: 0.0280 - RMSE: 0.0405 - mape: 11772.6104 -
MPE: -inf - MSLE: 7.9748e-04 - RMSLE: 0.0208 - R2: 0.9660 - val_loss: 0.0205 -
val_soft_acc: 0.7383 - val_mse: 6.3244e-04 - val_mae: 0.0177 - val_RMSE: 0.0215
- val_mape: 3.8094 - val_MPE: -3.1036 - val_MSLE: 2.7618e-04 - val_RMSLE: 0.0119
- val_R2: -5.4607
Epoch 26/200
24/24 [==============================] - ETA: 0s - loss: 0.0379 - soft_acc:
0.9680 - mse: 0.0015 - mae: 0.0265 - RMSE: 0.0378 - mape: 8102.5537 - MPE: -inf
- MSLE: 7.0651e-04 - RMSLE: 0.0197 - R2: 0.9709
Epoch 00026: val_loss did not improve from 0.01637
24/24 [==============================] - 967s 40s/step - loss: 0.0379 -
soft_acc: 0.9680 - mse: 0.0015 - mae: 0.0265 - RMSE: 0.0378 - mape: 8102.5537 -
MPE: -inf - MSLE: 7.0651e-04 - RMSLE: 0.0197 - R2: 0.9709 - val_loss: 0.0440 -
val_soft_acc: 0.6514 - val_mse: 0.0025 - val_mae: 0.0414 - val_RMSE: 0.0461 -
val_mape: 9.2293 - val_MPE: -9.6486 - val_MSLE: 0.0011 - val_RMSLE: 0.0279 -
val_R2: -30.3202
Epoch 27/200
24/24 [==============================] - ETA: 0s - loss: 0.0366 - soft_acc:
0.9688 - mse: 0.0014 - mae: 0.0254 - RMSE: 0.0366 - mape: 11164.5898 - MPE: -inf
- MSLE: 6.7093e-04 - RMSLE: 0.0190 - R2: 0.9728
Epoch 00027: val_loss improved from 0.01637 to 0.01518, saving model to
tmp\checkpointer.ckpt
INFO:tensorflow:Assets written to: tmp\checkpointer.ckpt\assets
```

```
24/24 [==============================] - 1075s 45s/step - loss: 0.0366 -
soft_acc: 0.9688 - mse: 0.0014 - mae: 0.0254 - RMSE: 0.0366 - mape: 11164.5898 -
MPE: -inf - MSLE: 6.7093e-04 - RMSLE: 0.0190 - R2: 0.9728 - val_loss: 0.0152 -
val_soft_acc: 0.9580 - val_mse: 2.6983e-04 - val_mae: 0.0120 - val_RMSE: 0.0139
- val_mape: 2.7685 - val_MPE: 1.6264 - val_MSLE: 1.2521e-04 - val_RMSLE: 0.0084
- val_R2: 0.0577
Epoch 28/200
24/24 [==============================] - ETA: 0s - loss: 0.0367 - soft_acc:
0.9678 - mse: 0.0014 - mae: 0.0257 - RMSE: 0.0367 - mape: 7949.6260 - MPE: -inf
- MSLE: 6.6257e-04 - RMSLE: 0.0190 - R2: 0.9724
Epoch 00028: val_loss did not improve from 0.01518
24/24 [==============================] - 879s 37s/step - loss: 0.0367 -
soft_acc: 0.9678 - mse: 0.0014 - mae: 0.0257 - RMSE: 0.0367 - mape: 7949.6260 -
MPE: -inf - MSLE: 6.6257e-04 - RMSLE: 0.0190 - R2: 0.9724 - val_loss: 0.0218 -
val_soft_acc: 0.9248 - val_mse: 5.3634e-04 - val_mae: 0.0191 - val_RMSE: 0.0215
- val_mape: 4.5644 - val_MPE: -4.2005 - val_MSLE: 2.4826e-04 - val_RMSLE: 0.0133
- val_R2: -3.7372
Epoch 29/200
24/24 [==============================] - ETA: 0s - loss: 0.0357 - soft_acc:
0.9669 - mse: 0.0013 - mae: 0.0247 - RMSE: 0.0356 - mape: 9120.7588 - MPE: -inf
- MSLE: 6.1619e-04 - RMSLE: 0.0183 - R2: 0.9740
Epoch 00029: val_loss did not improve from 0.01518
24/24 [==============================] - 885s 37s/step - loss: 0.0357 -
soft_acc: 0.9669 - mse: 0.0013 - mae: 0.0247 - RMSE: 0.0356 - mape: 9120.7588 -
MPE: -inf - MSLE: 6.1619e-04 - RMSLE: 0.0183 - R2: 0.9740 - val_loss: 0.0279 -
val_soft_acc: 0.9541 - val_mse: 7.9550e-04 - val_mae: 0.0253 - val_RMSE: 0.0264
- val_mape: 6.2962 - val_MPE: 5.2235 - val_MSLE: 4.0745e-04 - val_RMSLE: 0.0181
- val_R2: -6.9264
Epoch 30/200
24/24 [==============================] - ETA: 0s - loss: 0.0367 - soft_acc:
0.9691 - mse: 0.0014 - mae: 0.0259 - RMSE: 0.0364 - mape: 12049.0713 - MPE: -inf
- MSLE: 6.7839e-04 - RMSLE: 0.0193 - R2: 0.9727
Epoch 00030: val_loss did not improve from 0.01518
24/24 [==============================] - 892s 37s/step - loss: 0.0367 -
soft_acc: 0.9691 - mse: 0.0014 - mae: 0.0259 - RMSE: 0.0364 - mape: 12049.0713 -
MPE: -inf - MSLE: 6.7839e-04 - RMSLE: 0.0193 - R2: 0.9727 - val_loss: 0.0684 -
val_soft_acc: 0.6094 - val_mse: 0.0056 - val_mae: 0.0666 - val_RMSE: 0.0712 -
val_mape: 15.1718 - val_MPE: -15.6417 - val_MSLE: 0.0024 - val_RMSLE: 0.0448 -
val_R2: -70.2989
Epoch 31/200
24/24 [==============================] - ETA: 0s - loss: 0.0350 - soft_acc:
0.9703 - mse: 0.0012 - mae: 0.0239 - RMSE: 0.0352 - mape: 8415.7598 - MPE: -inf
- MSLE: 5.8728e-04 - RMSLE: 0.0176 - R2: 0.9748
Epoch 00031: val_loss did not improve from 0.01518
24/24 [==============================] - 897s 37s/step - loss: 0.0350 -
soft_acc: 0.9703 - mse: 0.0012 - mae: 0.0239 - RMSE: 0.0352 - mape: 8415.7598 -
MPE: -inf - MSLE: 5.8728e-04 - RMSLE: 0.0176 - R2: 0.9748 - val_loss: 0.0269 -
val_soft_acc: 0.6943 - val_mse: 8.7694e-04 - val_mae: 0.0247 - val_RMSE: 0.0277
```

```
- val_mape: 5.6379 - val_MPE: -5.6211 - val_MSLE: 3.9557e-04 - val_RMSLE: 0.0169
- val_R2: -8.6698
Epoch 32/200
24/24 [==============================] - ETA: 0s - loss: 0.0358 - soft_acc:
0.9688 - mse: 0.0013 - mae: 0.0251 - RMSE: 0.0355 - mape: 11513.2695 - MPE: -inf
- MSLE: 6.3224e-04 - RMSLE: 0.0187 - R2: 0.9745
Epoch 00032: val_loss did not improve from 0.01518
24/24 [==============================] - 901s 38s/step - loss: 0.0358 -
soft_acc: 0.9688 - mse: 0.0013 - mae: 0.0251 - RMSE: 0.0355 - mape: 11513.2695 -
MPE: -inf - MSLE: 6.3224e-04 - RMSLE: 0.0187 - R2: 0.9745 - val_loss: 0.0535 -
val_soft_acc: 0.6436 - val_mse: 0.0035 - val_mae: 0.0513 - val_RMSE: 0.0554 -
val_mape: 11.6073 - val_MPE: -11.9369 - val_MSLE: 0.0015 - val_RMSLE: 0.0346 -
val_R2: -39.1862
Epoch 33/200
24/24 [==============================] - ETA: 0s - loss: 0.0355 - soft_acc:
0.9646 - mse: 0.0013 - mae: 0.0251 - RMSE: 0.0352 - mape: 9756.0449 - MPE: -inf
- MSLE: 6.3342e-04 - RMSLE: 0.0188 - R2: 0.9748
Epoch 00033: val_loss did not improve from 0.01518
24/24 [==============================] - 906s 38s/step - loss: 0.0355 -
soft_acc: 0.9646 - mse: 0.0013 - mae: 0.0251 - RMSE: 0.0352 - mape: 9756.0449 -
MPE: -inf - MSLE: 6.3342e-04 - RMSLE: 0.0188 - R2: 0.9748 - val_loss: 0.0523 -
val_soft_acc: 0.6436 - val_mse: 0.0042 - val_mae: 0.0494 - val_RMSE: 0.0567 -
val_mape: 10.7163 - val_MPE: -8.3161 - val_MSLE: 0.0018 - val_RMSLE: 0.0328 -
val_R2: -61.4931
Epoch 34/200
24/24 [==============================] - ETA: 0s - loss: 0.0350 - soft_acc:
0.9723 - mse: 0.0012 - mae: 0.0246 - RMSE: 0.0350 - mape: 9540.5791 - MPE: -inf
- MSLE: 6.0294e-04 - RMSLE: 0.0183 - R2: 0.9753
Epoch 00034: val_loss did not improve from 0.01518
24/24 [==============================] - 906s 38s/step - loss: 0.0350 -
soft_acc: 0.9723 - mse: 0.0012 - mae: 0.0246 - RMSE: 0.0350 - mape: 9540.5791 -
MPE: -inf - MSLE: 6.0294e-04 - RMSLE: 0.0183 - R2: 0.9753 - val_loss: 0.0588 -
val_soft_acc: 0.6250 - val_mse: 0.0048 - val_mae: 0.0560 - val_RMSE: 0.0620 -
val_mape: 12.2761 - val_MPE: -12.9355 - val_MSLE: 0.0020 - val_RMSLE: 0.0373 -
val_R2: -56.4068
Epoch 35/200
24/24 [==============================] - ETA: 0s - loss: 0.0368 - soft_acc:
0.9673 - mse: 0.0014 - mae: 0.0253 - RMSE: 0.0368 - mape: 7831.2246 - MPE: -inf
- MSLE: 6.5561e-04 - RMSLE: 0.0187 - R2: 0.9722
Epoch 00035: val_loss did not improve from 0.01518
24/24 [==============================] - 911s 38s/step - loss: 0.0368 -
soft_acc: 0.9673 - mse: 0.0014 - mae: 0.0253 - RMSE: 0.0368 - mape: 7831.2246 -
MPE: -inf - MSLE: 6.5561e-04 - RMSLE: 0.0187 - R2: 0.9722 - val_loss: 0.0358 -
val_soft_acc: 0.6787 - val_mse: 0.0016 - val_mae: 0.0335 - val_RMSE: 0.0370 -
val_mape: 7.5351 - val_MPE: -7.7474 - val_MSLE: 7.0735e-04 - val_RMSLE: 0.0227 -
val_R2: -16.9758
Epoch 36/200
24/24 [==============================] - ETA: 0s - loss: 0.0332 - soft_acc:
```

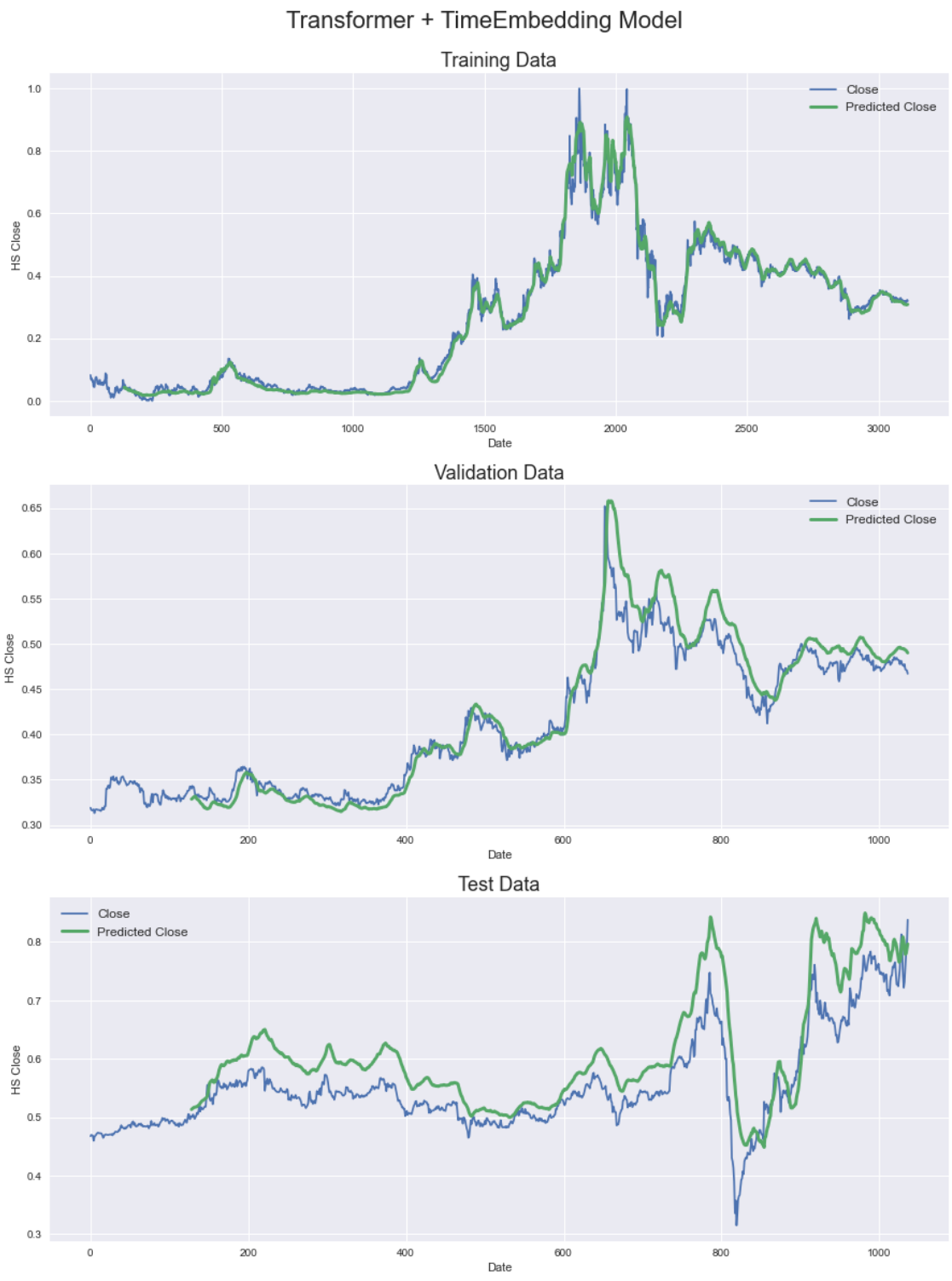
```
0.9689 - mse: 0.0011 - mae: 0.0229 - RMSE: 0.0333 - mape: 9187.8701 - MPE: -inf
- MSLE: 5.3030e-04 - RMSLE: 0.0170 - R2: 0.9778
Epoch 00036: val_loss did not improve from 0.01518
24/24 [==============================] - 909s 38s/step - loss: 0.0332 -
soft_acc: 0.9689 - mse: 0.0011 - mae: 0.0229 - RMSE: 0.0333 - mape: 9187.8701 -
MPE: -inf - MSLE: 5.3030e-04 - RMSLE: 0.0170 - R2: 0.9778 - val_loss: 0.0389 -
val_soft_acc: 0.6562 - val_mse: 0.0025 - val_mae: 0.0364 - val_RMSE: 0.0418 -
val_mape: 7.7517 - val_MPE: -8.0860 - val_MSLE: 0.0011 - val_RMSLE: 0.0242 -
val_R2: -30.0132
Epoch 37/200
24/24 [==============================] - ETA: 0s - loss: 0.0331 - soft_acc:
0.9712 - mse: 0.0011 - mae: 0.0228 - RMSE: 0.0332 - mape: 10606.1631 - MPE: -inf
- MSLE: 5.2473e-04 - RMSLE: 0.0168 - R2: 0.9779
Epoch 00037: val_loss did not improve from 0.01518
24/24 [==============================] - 905s 38s/step - loss: 0.0331 -
soft_acc: 0.9712 - mse: 0.0011 - mae: 0.0228 - RMSE: 0.0332 - mape: 10606.1631 -
MPE: -inf - MSLE: 5.2473e-04 - RMSLE: 0.0168 - R2: 0.9779 - val_loss: 0.0174 -
val_soft_acc: 0.9277 - val_mse: 4.3164e-04 - val_mae: 0.0142 - val_RMSE: 0.0175
- val_mape: 3.1616 - val_MPE: -1.3748 - val_MSLE: 1.8782e-04 - val_RMSLE: 0.0097
- val_R2: -2.0044
```

[23]:

```
'''Display results'''

fig = plt.figure(figsize=(15,20))
st = fig.suptitle("Transformer + TimeEmbedding Model", fontsize=22)
st.set_y(0.92)

#Plot training data results
ax11 = fig.add_subplot(311)
ax11.plot(train_data[:, 3], label='Close')
ax11.plot(np.arange(seq_len, train_pred.shape[0]+seq_len), train_pred,␣
 ↪linewidth=3, label='Predicted Close')
ax11.set_title("Training Data", fontsize=18)
ax11.set_xlabel('Date')
ax11.set_ylabel('HS Close')
ax11.legend(loc="best", fontsize=12)

#Plot validation data results
ax21 = fig.add_subplot(312)
ax21.plot(valid_data[:, 3], label='Close')
ax21.plot(np.arange(seq_len, valid_pred.shape[0]+seq_len), valid_pred,␣
 ↪linewidth=3, label='Predicted Close')
ax21.set_title("Validation Data", fontsize=18)
ax21.set_xlabel('Date')
ax21.set_ylabel('HS Close')
ax21.legend(loc="best", fontsize=12)
```

```
#Plot test data results
ax31 = fig.add_subplot(313)
ax31.plot(test_data[:, 3], label='Close')
ax31.plot(np.arange(seq_len, test_pred.shape[0]+seq_len), test_pred,␣
 ↪linewidth=3, label='Predicted Close')
ax31.set_title("Test Data", fontsize=18)
ax31.set_xlabel('Date')
ax31.set_ylabel('HS Close')
ax31.legend(loc="best", fontsize=12)
```

[23]: <matplotlib.legend.Legend at 0x19ea319db50>

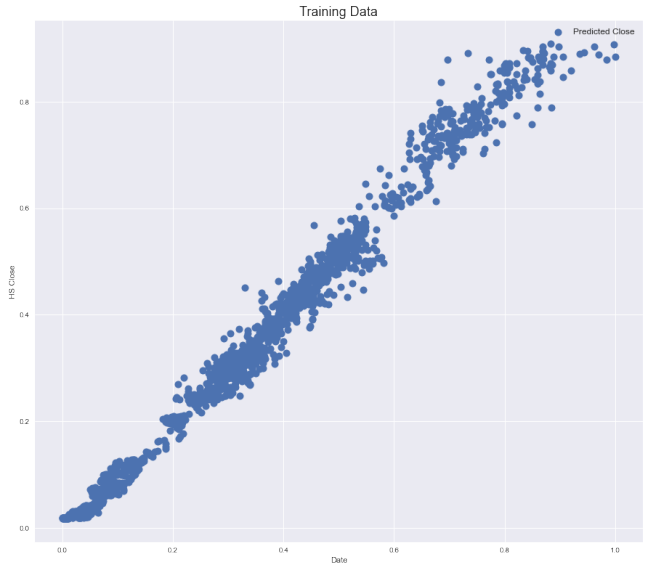


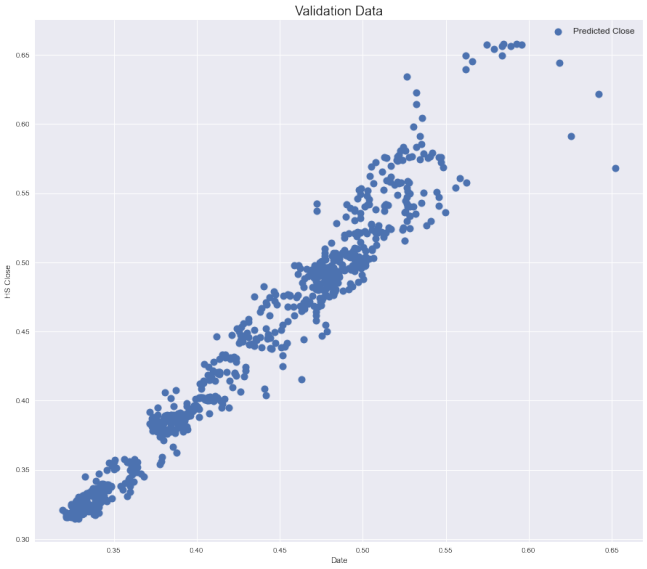
[24]:
```
# 원래값과 예측 값이 일치하면 직선에 가깝게 분포가 된다
```

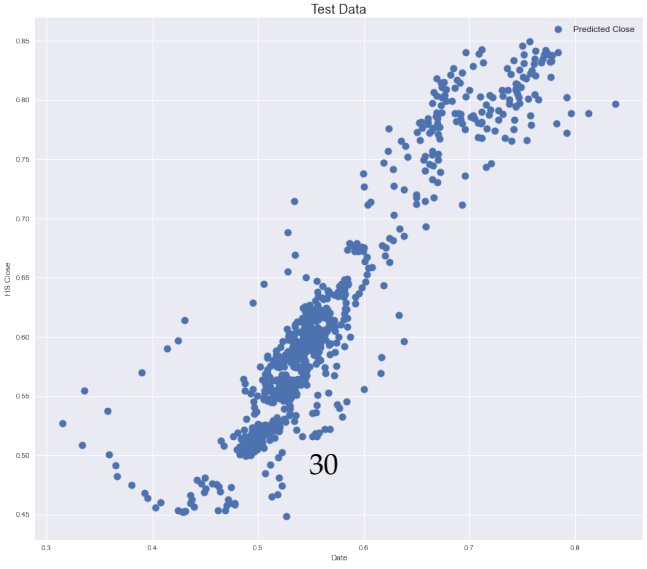
```
%matplotlib inline
import matplotlib.pyplot as plt

'''Display results'''

fig = plt.figure(figsize=(15,45))
st = fig.suptitle("Transformer + TimeEmbedding Model", fontsize=22)
st.set_y(0.92)

#Plot training data results
ax11 = fig.add_subplot(311)
plt.scatter(np.asarray(y_train), train_pred, linewidth=3, label='Predicted␣
 ↪Close')
ax11.set_title("Training Data", fontsize=18)
ax11.set_xlabel('Date')
ax11.set_ylabel('HS Close')
ax11.legend(loc="best", fontsize=12)

#Plot validation data results
ax21 = fig.add_subplot(312)
plt.scatter(np.asarray(y_valid), valid_pred, linewidth=3, label='Predicted␣
 ↪Close')
ax21.set_title("Validation Data", fontsize=18)
ax21.set_xlabel('Date')
ax21.set_ylabel('HS Close')
ax21.legend(loc="best", fontsize=12)

#Plot test data results
ax31 = fig.add_subplot(313)
plt.scatter(np.asarray(y_test), test_pred, linewidth=3, label='Predicted Close')
ax31.set_title("Test Data", fontsize=18)
ax31.set_xlabel('Date')
ax31.set_ylabel('HS Close')
ax31.legend(loc="best", fontsize=12)
```

[24]: <matplotlib.legend.Legend at 0x19e0d331e20>

Transformer + TimeEmbedding Model

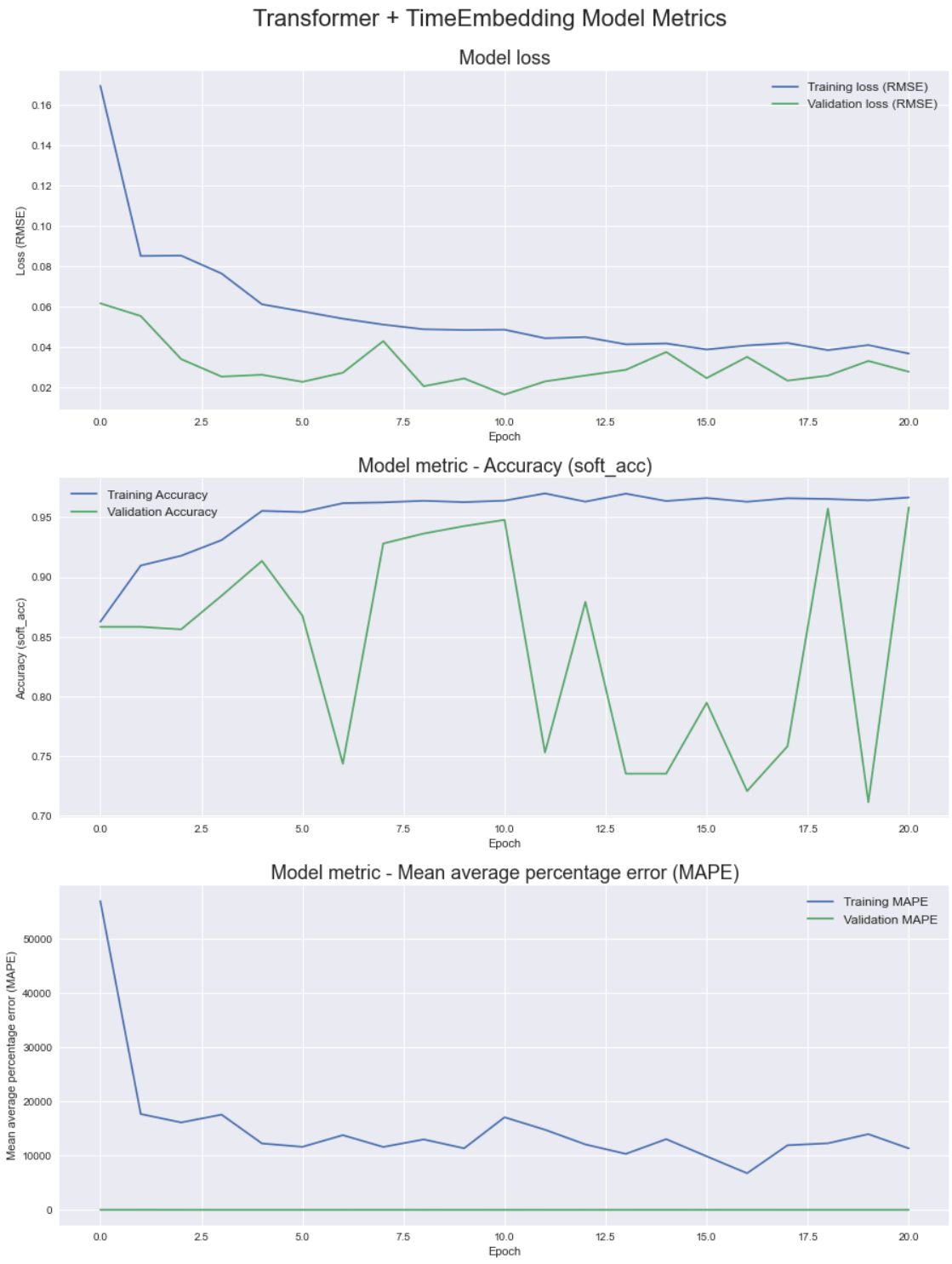






[26]:
```
'''Display model metrics'''

fig = plt.figure(figsize=(15,20))
st = fig.suptitle("Transformer + TimeEmbedding Model Metrics", fontsize=22)
st.set_y(0.92)

#Plot Model Loss
ax1 = fig.add_subplot(311)
ax1.plot(history.history['loss'], label='Training loss (RMSE)')
ax1.plot(history.history['val_loss'], label='Validation loss (RMSE)')
ax1.set_title("Model loss", fontsize=18)
ax1.set_xlabel('Epoch')
ax1.set_ylabel('Loss (RMSE)')
ax1.legend(loc="best", fontsize=12)

#Plot Model Acurracy
ax2 = fig.add_subplot(312)
ax2.plot(history.history['soft_acc'], label='Training Accuracy')
ax2.plot(history.history['val_soft_acc'], label='Validation Accuracy')
ax2.set_title("Model metric - Accuracy (soft_acc)", fontsize=18)
ax2.set_xlabel('Epoch')
ax2.set_ylabel('Accuracy (soft_acc)')
ax2.legend(loc="best", fontsize=12)

#Plot MAPE
ax3 = fig.add_subplot(313)
ax3.plot(history.history['mape'], label='Training MAPE')
ax3.plot(history.history['val_mape'], label='Validation MAPE')
ax3.set_title("Model metric - Mean average percentage error (MAPE)", fontsize=18)
ax3.set_xlabel('Epoch')
ax3.set_ylabel('Mean average percentage error (MAPE)')
ax3.legend(loc="best", fontsize=12)
```

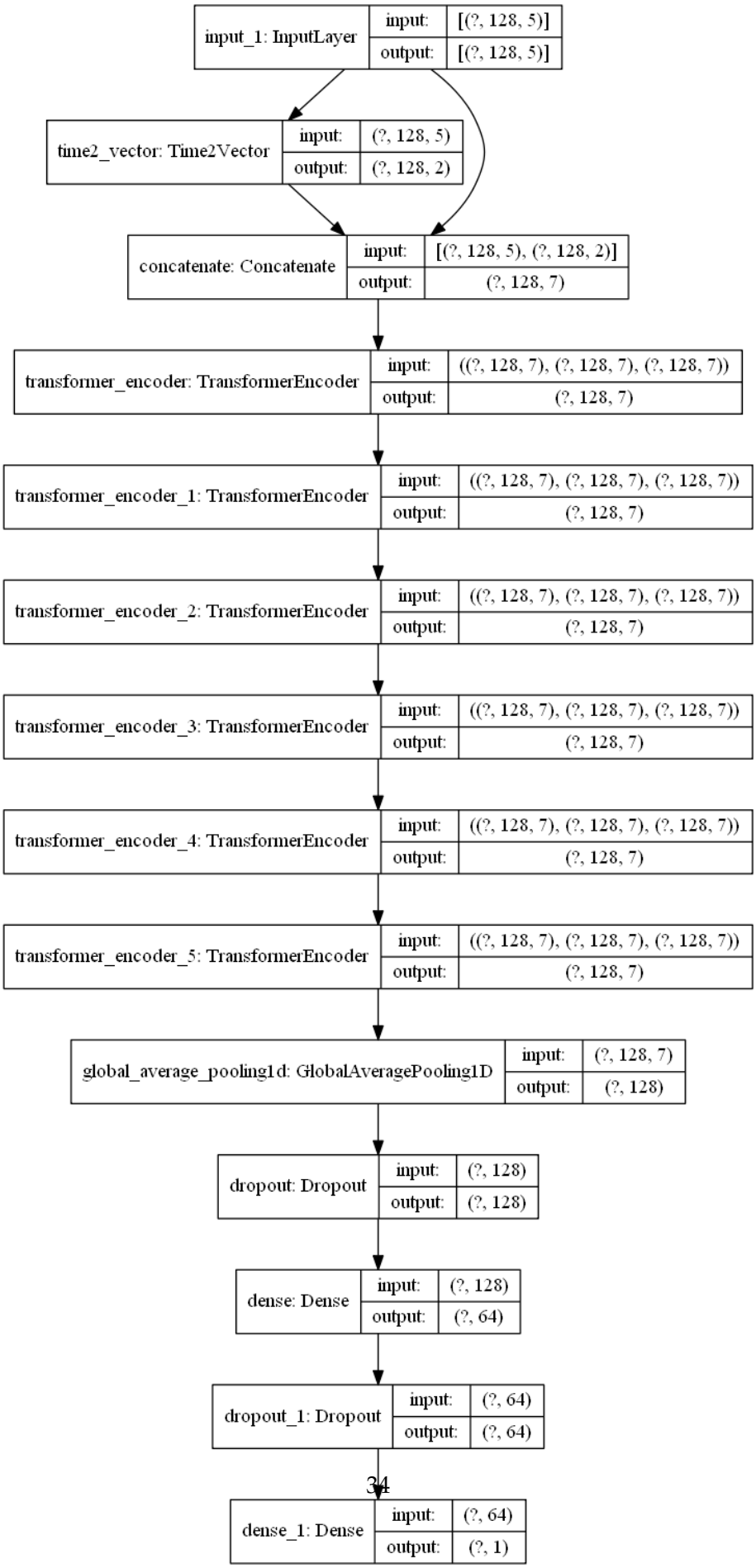
[26]:
```
<matplotlib.legend.Legend at 0x2690438f040>
```



[25]:
```
tf.keras.utils.plot_model(
    model,
    to_file="HS_Transformer+TimeEmbedding.png",
```

```
    show_shapes=True,
    show_layer_names=True,
    expand_nested=True,
    dpi=96,)
```

[25]:


input_1: InputLayer
input: [(?, 128, 5)]
output: [(?, 128, 5)]
time2_vector: Time2Vector
input: (?, 128, 5)
output: (?, 128, 2)
concatenate: Concatenate
input: [(?, 128, 5), (?, 128, 2)]
output: (?, 128, 7)
transformer_encoder: TransformerEncoder
input: ((?, 128, 7), (?, 128, 7), (?, 128, 7))
output: (?, 128, 7)
transformer_encoder_1: TransformerEncoder
input: ((?, 128, 7), (?, 128, 7), (?, 128, 7))
output: (?, 128, 7)
transformer_encoder_2: TransformerEncoder
input: ((?, 128, 7), (?, 128, 7), (?, 128, 7))
output: (?, 128, 7)
transformer_encoder_3: TransformerEncoder
input: ((?, 128, 7), (?, 128, 7), (?, 128, 7))
output: (?, 128, 7)
transformer_encoder_4: TransformerEncoder
input: ((?, 128, 7), (?, 128, 7), (?, 128, 7))
output: (?, 128, 7)
transformer_encoder_5: TransformerEncoder
input: ((?, 128, 7), (?, 128, 7), (?, 128, 7))
output: (?, 128, 7)
global_average_pooling1d: GlobalAveragePooling1D
input: (?, 128, 7)
output: (?, 128)
dropout: Dropout
input: (?, 128)
output: (?, 128)
dense: Dense
input: (?, 128)
output: (?, 64)
dropout_1: Dropout
input: (?, 64)
output: (?, 64)
dense_1: Dense
input: (?, 64)
output: (?, 1)

[26]:
```
model.summary()
```

```
Model: "functional_1"
_________________________________________________________________________________
__________________
Layer (type)                    Output Shape         Param #     Connected to
=================================================================================
==================
input_1 (InputLayer)            [(None, 128, 5)]     0
_________________________________________________________________________________
__________________
time2_vector (Time2Vector)      (None, 128, 2)       512         input_1[0][0]
_________________________________________________________________________________
__________________
concatenate (Concatenate)       (None, 128, 7)       0           input_1[0][0]
time2_vector[0][0]
_________________________________________________________________________________
__________________
transformer_encoder (Transforme (None, 128, 7)       46634
concatenate[0][0]
concatenate[0][0]
concatenate[0][0]
_________________________________________________________________________________
__________________
transformer_encoder_1 (Transfor (None, 128, 7)       46634
transformer_encoder[0][0]
transformer_encoder[0][0]
transformer_encoder[0][0]
_________________________________________________________________________________
__________________
transformer_encoder_2 (Transfor (None, 128, 7)       46634
transformer_encoder_1[0][0]
transformer_encoder_1[0][0]
transformer_encoder_1[0][0]
_________________________________________________________________________________
__________________
transformer_encoder_3 (Transfor (None, 128, 7)       46634
transformer_encoder_2[0][0]
transformer_encoder_2[0][0]
transformer_encoder_2[0][0]
_________________________________________________________________________________
__________________
transformer_encoder_4 (Transfor (None, 128, 7)       46634
transformer_encoder_3[0][0]
transformer_encoder_3[0][0]
```

```
transformer_encoder_3[0][0]
__________________________________________________________________________________
__________________
transformer_encoder_5 (Transfor (None, 128, 7)       46634
transformer_encoder_4[0][0]
transformer_encoder_4[0][0]
transformer_encoder_4[0][0]
__________________________________________________________________________________
__________________
global_average_pooling1d (Globa (None, 128)          0
transformer_encoder_5[0][0]
__________________________________________________________________________________
__________________
dropout (Dropout)               (None, 128)          0
global_average_pooling1d[0][0]
__________________________________________________________________________________
__________________
dense (Dense)                   (None, 64)           8256        dropout[0][0]
__________________________________________________________________________________
__________________
dropout_1 (Dropout)             (None, 64)           0           dense[0][0]
__________________________________________________________________________________
__________________
dense_1 (Dense)                 (None, 1)            65          dropout_1[0][0]
==================================================================================
==================
Total params: 288,637
Trainable params: 288,637
Non-trainable params: 0
__________________________________________________________________________________
__________________
```

[27]:
```
import numpy as np


print('R2_Score')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[9], valid_evaluate[9], test_evaluate[9]))

print('Mean Squared Error')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[2], valid_evaluate[2], test_evaluate[2]))

print('Mean Absolute Error')
print('-' * 40)
```

```
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[3], valid_evaluate[3], test_evaluate[3]))

print('Root Mean Squared Error')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[4], valid_evaluate[4], test_evaluate[4]))

print('Mean Squared Logarithmic Error')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[7], valid_evaluate[7], test_evaluate[7]))

print('Root Mean Squared Logarithmic Error')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[8], valid_evaluate[8], test_evaluate[8]))

print('Mean Absolute Percentage Error')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[5], valid_evaluate[5], test_evaluate[5]))

print('Mean Percentage Error')
print('-' * 40)
print('train error: {} |\nvalid error: {} |\ntest error : {}\n'.
 ↪format(train_evaluate[6], valid_evaluate[6], test_evaluate[6]))
```

```
R2_Score
----------------------------------------
train error: -2.2048556804656982 |
valid error: -3.0671796798706055 |
test error : -22.03276824951172

Mean Squared Error
----------------------------------------
train error: 0.0005313938017934561 |
valid error: 0.00043164295493625104 |
test error : 0.0036397164221853018

Mean Absolute Error
----------------------------------------
train error: 0.01543228980153799 |
valid error: 0.014203246682882309 |
test error : 0.05052703246474266

Root Mean Squared Error
```

```
---------------------------------------
train error: 0.018304595723748207 |
valid error: 0.01682080328464508 |
test error : 0.05263758823275566

Mean Squared Logarithmic Error
---------------------------------------
train error: 0.00025315783568657935 |
valid error: 0.0001878163602668792 |
test error : 0.0013876954326406121

Root Mean Squared Logarithmic Error
---------------------------------------
train error: 0.011533037759363651 |
valid error: 0.00967154186218977 |
test error : 0.0314338281750679

Mean Absolute Percentage Error
---------------------------------------
train error: 6202.07373046875 |
valid error: 3.1616315841674805 |
test error : 8.97009563446045

Mean Percentage Error
---------------------------------------
train error: -inf |
valid error: -1.1282840967178345 |
test error : -8.362812995910645
```

[ ]: