import math

import warnings

import numpy as np

import pandas as pd

from data.data import process\_data

from keras.models import load\_model

from keras.utils.vis\_utils import plot\_model

import sklearn.metrics as metrics

import matplotlib as mpl

import matplotlib.pyplot as plt

warnings.filterwarnings("ignore")

def MAPE(y\_true, y\_pred):

y = [x for x in y\_true if x > 0]

y\_pred = [y\_pred[i] for i in range(len(y\_true)) if y\_true[i] > 0]

num = len(y\_pred)

sums = 0

for i in range(num):

tmp = abs(y[i] - y\_pred[i]) / y[i]

sums += tmp

mape = sums \* (100 / num)

return mape

def eva\_regress(y\_true, y\_pred):

mape = MAPE(y\_true, y\_pred)

vs = metrics.explained\_variance\_score(y\_true, y\_pred)

mae = metrics.mean\_absolute\_error(y\_true, y\_pred)

mse = metrics.mean\_squared\_error(y\_true, y\_pred)

r2 = metrics.r2\_score(y\_true, y\_pred)

print('explained\_variance\_score:%f' % vs)

print('mape:%f%%' % mape)

print('mae:%f' % mae)

print('mse:%f' % mse)

print('rmse:%f' % math.sqrt(mse))

print('r2:%f' % r2)

def plot\_results(y\_true, y\_preds, names):

d = '2016-3-4 00:00'

x = pd.date\_range(d, periods=288, freq='5min')

fig = plt.figure()

ax = fig.add\_subplot(111)

ax.plot(x, y\_true, label='True Data')

for name, y\_pred in zip(names, y\_preds):

ax.plot(x, y\_pred, label=name)

plt.legend()

plt.grid(True)

plt.xlabel('Time of Day')

plt.ylabel('Flow')

date\_format = mpl.dates.DateFormatter("%H:%M")

ax.xaxis.set\_major\_formatter(date\_format)

fig.autofmt\_xdate()

plt.show()

def main():

lstm = load\_model('C:/Users/DELL/Downloads/traffic flow prediction/TrafficFlowPrediction-master/model/lstm.h5')

gru = load\_model('C:/Users/DELL/Downloads/traffic flow prediction/TrafficFlowPrediction-master/model/gru.h5')

saes = load\_model('C:/Users/DELL/Downloads/traffic flow prediction/TrafficFlowPrediction-master/model/saes.h5')

models = [lstm, gru, saes]

names = ['LSTM', 'GRU', 'SAEs']

lag = 12

file1 = 'C:/Users/DELL/Downloads/traffic flow prediction/TrafficFlowPrediction-master/data/train.csv'

file2 = 'C:/Users/DELL/Downloads/traffic flow prediction/TrafficFlowPrediction-master/data/test.csv'

\_, \_, X\_test, y\_test, scaler = process\_data(file1, file2, lag)

y\_test = scaler.inverse\_transform(y\_test.reshape(-1, 1)).reshape(1, -1)[0]

y\_preds = []

for name, model in zip(names, models):

if name == 'SAEs':

X\_test = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1]))

else:

X\_test = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1], 1))

file = 'C:/Users/DELL/Downloads/traffic flow prediction/TrafficFlowPrediction-master/images/' + name + '.png'

plot\_model(model, to\_file=file, show\_shapes=True)

predicted = model.predict(X\_test)

predicted = scaler.inverse\_transform(predicted.reshape(-1, 1)).reshape(1, -1)[0]

y\_preds.append(predicted[:288])

print(name)

eva\_regress(y\_test, predicted)

plot\_results(y\_test[: 288], y\_preds, names)

if \_\_name\_\_ == '\_\_main\_\_':

main()