# %%

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

from tensorflow.keras.layers import LSTM

from tensorflow.keras.layers import Dropout

from tensorflow.keras.models import load\_model

import numpy as np

import pandas as pd

import os

import tensorflow as tf

LookBackNum = 12 # LSTM往前看的筆數

ForecastNum = 48 # 預測筆數 (一天: 9:00 ~ 5:00 pm)

# 載入訓練資料

DataName = os.getcwd() + '\L1\_2024\_0102.csv'

SourceData = pd.read\_csv(DataName, encoding='utf-8')

target = ['Power(mW)']

AllOutPut = SourceData[target].values

X\_train = []

y\_train = []

# 設定每i-12筆資料(X\_train)就對應到第i筆資料(y\_train)

for i in range(LookBackNum, len(AllOutPut)):

X\_train.append(AllOutPut[i - LookBackNum:i, 0])

y\_train.append(AllOutPut[i, 0])

# 12 data is a batch?

X\_train = np.array(X\_train)

y\_train = np.array(y\_train)

# Reshaping

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

# %%

# ============================建置&訓練模型============================

# 建置LSTM模型

regressor = Sequential()

regressor.add(LSTM(units=128, return\_sequences=True, input\_shape=(X\_train.shape[1], 1)))

regressor.add(LSTM(units=64))

regressor.add(Dropout(0.2))

# output layer

regressor.add(Dense(units=1))

regressor.compile(optimizer='adam', loss='mean\_squared\_error')

# 開始訓練

regressor.fit(X\_train, y\_train, epochs=1, batch\_size=128)

print('Model Saved')

# 保存模型

from datetime import datetime

NowDateTime = datetime.now().strftime("%Y-%m-%dT%H\_%M\_%SZ")

regressor.save('WheatherLSTM\_Model' + '.h5')

print('Model Saved')

# %%

# ============================預測數據============================

# 載入模型

regressor = load\_model('WheatherLSTM\_Model.h5')

# 載入測試資料

# DataName = os.getcwd() + r'\upload.csv'

SourceData = pd.read\_csv('L1\_2024\_0102\_answer', encoding='utf-8')

target = ['序號']

EXquestion = SourceData[target].values

inputs = [] # 存放參考資料

PredictOutput = [] # 存放預測值

count = 0

# while(count < len(EXquestion)):

# print('count : ',count)

# LocationCode = int(EXquestion[count])

# strLocationCode = str(LocationCode)[-2:]

# print('strLocationCode = ', strLocationCode)

# if LocationCode < 10 :

# strLocationCode = '0'+LocationCode

inputs = np.append(inputs, AllOutPut[-12:])

# DataName = os.getcwd()+'/L1\_2024\_0102''.csv'

SourceData = pd.read\_csv(DataName, encoding='utf-8')

ReferTitle = SourceData[['Serial']].values

ReferData = SourceData[['Power(mW)']].values

# 用迴圈不斷使新的預測值塞入參考資料，並預測下一筆資料

for i in range(ForecastNum):

# print(i)

# 將新的預測值加入參考資料(用自己的預測值往前看)

if i > 0:

inputs = np.append(inputs, PredictOutput[i - 1])

# 切出新的參考資料12筆(往前看12筆)

X\_test = []

X\_test.append(inputs[0 + i:LookBackNum + i])

# Reshaping

NewTest = np.array(X\_test)

NewTest = np.reshape(NewTest, (NewTest.shape[0], NewTest.shape[1], 1))

predicted = regressor.predict(NewTest)

PredictOutput.append(round(predicted[0, 0], 2))

# 每次預測都要預測48個，因此加48個會切到下一天

# 0~47,48~95,96~143...

count += 48

df = pd.DataFrame(PredictOutput, columns=['答案'])

# 將 DataFrame 寫入 CSV 檔案

df.to\_csv('result.csv', index=False)

import os

print(os.getcwd())

print('Output CSV File Saved')

# %%