Electronic Supplementary Material (ESI) for Environmental Science: Water Research & Technology.
This journal is © The Royal Society of Chemistry 2019

Supporting Information for

Deep learning based dynamic COD prediction model for urban sewage

Zifei Wang[1,3], Yi Man*[,1], Yusha Hu*[,1] Jigeng Li[1], Mengna Hong[1], Peizhe Cui[2,4]

1. State Key Laboratory of Pulp and Papermaking Engineering, South China University of Technology, Guangzhou, 510640, China

2. College of Chemical Engineering, Qingdao University of Science and Technology, Qingdao, 266042, China

3. National Demonstration Center for Experimental Light Industry and Food Education, South China University of Technology, Guangzhou, 510640, China

4. Shandong Collaborative Innovation Center of Eco-Chemical Engineering, Qingdao, 266042, China

Content

1.Table of model key parameters selection

Table S1. The evaluation indicator of different model layers

Convolution layers	RMSE (mg)	MAPE
1(1×5)	**1034.39**	**0.95**
2(1×4+1×2)	1144.68	0.99
2(1×2+1×4)	1060.34	0.99
2(1×3)	1192.89	1.08
3(1×3+(1×2) ×2)	1182.61	1.05
3(1×2+1×3+1×2)	1058.84	0.97
3(1×2+1×2+1×3)	1095.85	0.97
4((1×2) ×4)	1149.11	1.05

Table S2. The evaluation indicator of different Dropout

Dropout	RMSE	MAPE
0.1	1083.76	0.95
0.2	1122.06	1.00
0.3	1105.95	0.96
0.4	1034.35	0.91
0.5	1116.42	0.97
0.6	**1012.10**	**0.90**
0.7	1071.29	0.97
0.8	1066.84	0.96
0.9	1099.62	1.03

Table S3. The evaluation indicator of different filters

filters	RMSE	MAPE
16	1112.43	1.04
32	1037.86	0.94
64	**1032.48**	**0.93**
128	1118.43	0.97
256	1151.47	0.99

Table S4. The evaluation indicator of different Batch_size

Batch_size	RMSE	MAPE
32	1102.54	0.97
64	1059.25	0.95
128	**1056.96**	**0.93**
256	1087.19	0.98

Table S5. The evaluation indicator of different epochs

epochs	RMSE	MAPE
1	1036.43	0.98
10	1159.91	1.05
30	1143.24	1.00
50	**1050.06**	**0.93**
70	1120.61	0.98
100	1138.78	1.00

2. Figure of parameter adjustment

The optimization for the model parameters has four steps (Fig. S1.):

The first step is to adjust the number of convolution layers. Convolution layers are determined by the size of the convolution kernel. Convolution kernel is the core of the whole network.

The second step is to adjust the Dropout parameters to ensure that the model does not overfit.

The third step is to adjust the filters parameter. Filter is an important parameter that determines the dimension of the output space and has a significant impact on the prediction effect.

The fourth step is to adjust the batch_size parameter to determine the number of each training samples. The last step is to adjust the epochs parameter. Epochs is the iteration

cycle of the model, and the number of iterations affects the learning degree of the model.

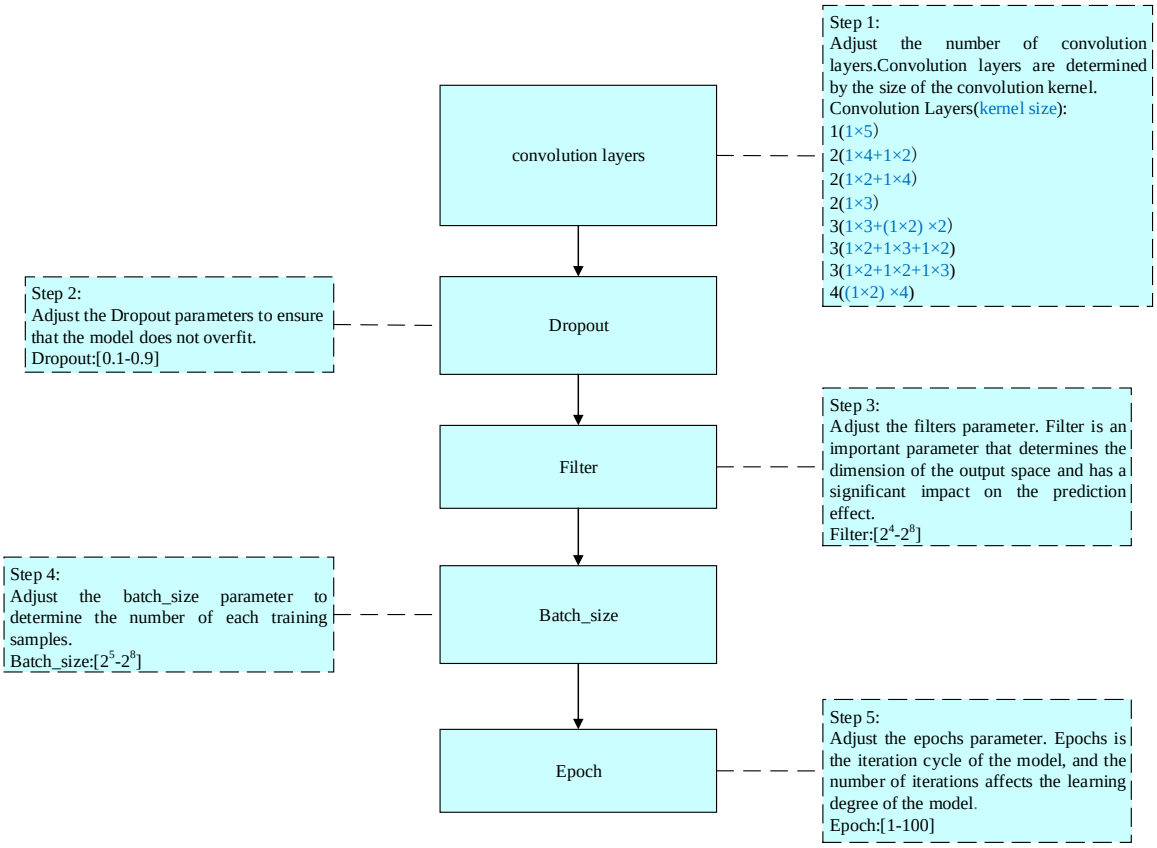

Fig. S1. The parameter adjustment sequence

3. The program code

In this paper, PYTHON version 3.6 is used to program the model. The dynamic forecasting code of the CNN-LSTM is shown as follows.

```
import pickle
import numpy as np
import pandas as pd
import xlrd
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt
import math
from keras.models import Sequential
from keras.layers import Dense
```

```
from keras.layers import LSTM
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
from math import sqrt
from sklearn.metrics import    mean_absolute_error

path="C:\\Users\\a\\Desktop\\sewage.xlsx"
data = pd.read_excel(path,header=0)
dataset=data.values
dataset1=dataset[:,-1]
dataset=dataset.astype('float32')


scaler=MinMaxScaler(feature_range=(0,1))

a=[]
b=[]

for i in    range(2):
      N=0
      J=720
      M=22750
      M=M+J*i
      N=N+J*i
      train=data.values[N:M,:-1]
      train=scaler.fit_transform(train)
      test=data.values[M:M+J,:-1]
      test=scaler.fit_transform(test)
      train_t = data.values[N:M,5:6]
      train_1=scaler.fit_transform(train_t)
      test_t=data.values[M:M+J,5:6]
      test_1=scaler.fit_transform(test_t)
      from keras.models import Sequential
      from keras.layers import Dense, Dropout
      from keras.layers import Embedding
      from keras.layers import Conv1D, GlobalAveragePooling1D,
MaxPooling1D,GlobalMaxPooling1D
      from keras.layers.normalization import BatchNormalization
      from keras.layers.core import Flatten

      x_train = train.reshape(train.shape[0], train.shape[1], 1)
```

```
    x_test = test.reshape(test.shape[0], test.shape[1], 1)


    model = Sequential()
    model.add(Conv1D(64, 5, activation='relu', input_shape=(5, 1)))
    model.add(MaxPooling1D(1))
    model.add(LSTM(64))
    model.add(Dropout(0.6))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy',
                  optimizer='rmsprop',
                  metrics=['accuracy'])

    model.fit(x_train,train_1, batch_size=128, epochs=50)
    score = model.evaluate(x_test, test_1, batch_size=128)
    pr=model.predict(x_test,steps=1)
    s=scaler.inverse_transform(pr)
    a.extend(s)
    b.extend(test_t)
a = np.array(a)
b = np.array(b)
RMSE_valid = sqrt(mean_squared_error(b, a))
def MAPE(b, a):
    result = 0
    for i in range(len(b)):
        result += abs((b[i] - a[i]) / b[i])
    result /= len(b)
    result *= 100
    return result

print(RMSE_valid)

print(MAPE(b,a))
np.savetxt("predict.xls",a,delimiter=",")
#np.savetxt("true.xls",b,delimiter=",")
squares =a
c=b
plt.plot(b, linewidth=1)
plt.plot(a,color="r", linewidth=1)
plt.title("CNN", fontsize=24)
plt.xlabel("Value", fontsize=14)
```

```
plt.ylabel("Square of Value", fontsize=14)
plt.tick_params(axis='both', which='major', labelsize=14)
plt.show()
```