به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین چهارم

نام و نام خانوادگی	سارا رستمی – محمدامین شاهچراغی
شماره دانشجویی	۸۱۰۱۰۰۳۵۵ - ۸۱۰۱۹۹۱۹۶
تاریخ ارسال گزارش	۰۶.۱۰.۱۴۰۱

فهرست

شکل‌ها

جدول‌ها

پاسخ ۱. تخمین آلودگی هوا

۱-۱. سوالات تشریحی

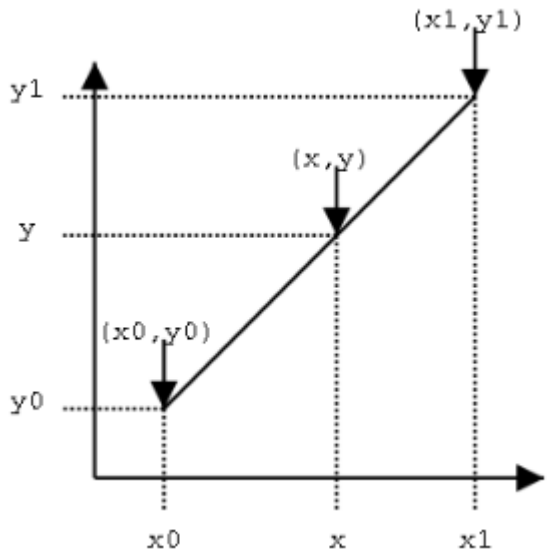
: Linear interpolation method

روش درونیابی خطی یا Linear Interpolation Method روشی است که با استفاده از آن میتوان داده هایی که در یک مجکوعه داده احیانا غایب هستند را تخمین زد معمولا برای تخمین missing values از روش میانگین گیری استفاده میشود اما در داده هایی که فرمت سری زمانی دارند باید از روش هایی مانند Linear Interpolation استفاده برد .

این روش با داشتن ۲ نقطه و نرخ تغییرات بین آنها نقطه سومی که بین آنها قرار دارد ولی مقدارش نامشخص است را تخمین میزند به اینصورت که آنها را با یک خط به یکدیگر متصل میکند اگر داده ها خطی نباشند و بین آنها تغییرات بزرگ اتفاق بیافتد تخمین این روش ممکن است دقیق نباشد .

$$SL(x) = f(x_{i-1})\frac{x - x_i}{x_{i-1} - x_i} + f(x_i)\frac{x - x_{i-1}}{x_i - x_{i-1}} \quad x \in [x_{i-1}, x_i], i = 1, 2, 3, ..., n$$

شکل ۱- فرمول روش Linear Interpolation



شکل ۲- روش Linear Interpolation

: Pearson Correlation

معیاری برای اندازه گیری همبستگی خطی بین ۲ مجموعه داده است . در این روش کواریانس دو مجوعه تقسیم بر ضرب انحراف معیار آنها میشود . در واقع کواریانسی است که نرمالسازی شده است . بنابراین همیشه مقدارش بین -۱ و ۱ است و هرقدر به ۱ نزدیکتر باشد یعنی ارتباط و همبستگی قویتری بین دو مجوعه داده وجود دارد .

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y}$$

شکل ۳- فرمول pearson correlation

: R^2

معیار آماری است که نشان میدهد چه نسبتی از واریانس وتغییرات یک متغیر وابسته توسط متغیر یا متغیر های غیروابسته از طریق رابطه رگرسیون توضیح داده میشود . هر چقدر به ۱ نزدیکتر یاشد متغیر یا متغیر های غیروابسته ، متغیر وابسته را بهتر توصیف میکنند .

$$\mathrm{R}^2 = 1 - \frac{\text{Unexplained Variation}}{\text{Total Variation}}$$

شکل ۴- فرمول R2

۲-۱. دیتاست

ابتدا همه ی ۱۲ فایل در گوگل درایو بارگذاری میکنیم و سپس از آنجا فرا میخوانیم . در اینجا موقع خواندن داده ها همه ستون های مربوط به نشان دادن زمان و تاریخ را در یک ستون تجمیع میکنیم چون میخواهیم داده ها بر اساس ترتیب زمانی شان مرتب باشند و همه ی داده های مربوط به ساعت و تاریخ

همین مساله را نشان میدهند . سپس ۷۷ سطر ابتدایی داده های مربوط به ایستگاه Aotizhongxin را نشان میدهیم . در اینجا چندین خانه داده ها که مقدار ندارند (missing value) مشهود است .

year_month_day_hour	No	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	wd	WSPM	station
2013-03-01 00:00:00	1	4.0	4.0	4.0	7.0	300.0	77.0	-0.7	1023.0	-18.8	0.0	NNW	4.4	Aotizhongxin
2013-03-01 01:00:00	2	8.0	8.0	4.0	7.0	300.0	77.0	-1.1	1023.2	-18.2	0.0	N	4.7	Aotizhongxin
2013-03-01 02:00:00	3	7.0	7.0	5.0	10.0	300.0	73.0	-1.1	1023.5	-18.2	0.0	NNW	5.6	Aotizhongxin
2013-03-01 03:00:00	4	6.0	6.0	11.0	11.0	300.0	72.0	-1.4	1024.5	-19.4	0.0	NW	3.1	Aotizhongxin
2013-03-01 04:00:00	5	3.0	3.0	12.0	12.0	300.0	72.0	-2.0	1025.2	-19.5	0.0	N	2.0	Aotizhongxin
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2013-03-04 00:00:00	73	42.0	83.0	51.0	86.0	1300.0	4.0	7.7	1015.7	-11.1	0.0	N	2.6	Aotizhongxin
2013-03-04 01:00:00	74	49.0	80.0	39.0	64.0	1000.0	25.0	8.2	1016.7	-11.7	0.0	N	2.8	Aotizhongxin
2013-03-04 02:00:00	75	34.0	62.0	NaN	14.0	300.0	68.0	8.1	1016.7	-11.8	0.0	N	4.3	Aotizhongxin
2013-03-04 03:00:00	76	12.0	34.0	6.0	12.0	NaN	77.0	7.2	1016.9	-11.6	0.0	N	2.8	Aotizhongxin
2013-03-04 04:00:00	77	7.0	18.0	14.0	NaN	400.0	42.0	6.0	1018.0	-11.6	0.0	NNW	1.0	Aotizhongxin

77 rows × 14 columns

شکل ۵- داده های ایستگاه Aotizhongxin

۳-۱. پیش‌پردازش

۱-۳-۱. Missing Value

داده های غایب را با متد df.interpolate() جایگزین میکنیم . برای دیتا فریم مربوط ایستگاه Aotizhongxin اینکار را میکنیم و ۷۷ سطر اول را نمایش میدهیم تا جایگذاری مقادیر غایب که در شکل ۳ با رنگ زرد مشخص شده دیده شود . سپس برای ادامه کارمان کل دیتا فریم که شامل ستون های pm2.5 استگاه های دیگر نیز است را نیز interpolate میکنیم .

year_month_day_hour	No	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	wd	WSPM	station
2013-03-01 00:00:00	1	4.0	4.0	4.0	7.0	300.0	77.0	-0.7	1023.0	-18.8	0.0	NNW	4.4	Aotizhongxin
2013-03-01 01:00:00	2	8.0	8.0	4.0	7.0	300.0	77.0	-1.1	1023.2	-18.2	0.0	N	4.7	Aotizhongxin
2013-03-01 02:00:00	3	7.0	7.0	5.0	10.0	300.0	73.0	-1.1	1023.5	-18.2	0.0	NNW	5.6	Aotizhongxin
2013-03-01 03:00:00	4	6.0	6.0	11.0	11.0	300.0	72.0	-1.4	1024.5	-19.4	0.0	NW	3.1	Aotizhongxin
2013-03-01 04:00:00	5	3.0	3.0	12.0	12.0	300.0	72.0	-2.0	1025.2	-19.5	0.0	N	2.0	Aotizhongxin
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2013-03-04 00:00:00	73	42.0	83.0	51.0	86.0	1300.0	4.0	7.7	1015.7	-11.1	0.0	N	2.6	Aotizhongxin
2013-03-04 01:00:00	74	49.0	80.0	39.0	64.0	1000.0	25.0	8.2	1016.7	-11.7	0.0	N	2.8	Aotizhongxin
2013-03-04 02:00:00	75	34.0	62.0	22.5	14.0	300.0	68.0	8.1	1016.7	-11.8	0.0	N	4.3	Aotizhongxin
2013-03-04 03:00:00	76	12.0	34.0	6.0	12.0	350.0	77.0	7.2	1016.9	-11.6	0.0	N	2.8	Aotizhongxin
2013-03-04 04:00:00	77	7.0	18.0	14.0	28.5	400.0	42.0	6.0	1018.0	-11.6	0.0	NNW	1.0	Aotizhongxin

77 rows × 14 columns

شکل ۶- جایگذاری مقادیر Nan ایستگاه Aotizhongxin

year_month_day_hour	No	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	...	PM2.5_Dingling	PM2.5_Dongsi	PM2.5_Guanyuan	PM2.5_Gucheng	PM2.5_Huairou
2013-03-01 00:00:00	1	4.0	4.0	4.0	7.0	300.0	77.0	-0.7	1023.0	-18.8	...	4.0	9.0	4.0	6.0	7.0
2013-03-01 01:00:00	2	8.0	8.0	4.0	7.0	300.0	77.0	-1.1	1023.2	-18.2	...	7.0	4.0	4.0	6.0	4.0
2013-03-01 02:00:00	3	7.0	7.0	5.0	10.0	300.0	73.0	-1.1	1023.5	-18.2	...	5.0	7.0	3.0	5.0	4.0
2013-03-01 03:00:00	4	6.0	6.0	11.0	11.0	300.0	72.0	-1.4	1024.5	-19.4	...	6.0	3.0	3.0	6.0	3.0
2013-03-01 04:00:00	5	3.0	3.0	12.0	12.0	300.0	72.0	-2.0	1025.2	-19.5	...	5.0	3.0	3.0	5.0	3.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2017-02-28 19:00:00	35060	12.0	29.0	5.0	35.0	400.0	95.0	12.5	1013.5	-16.2	...	11.0	16.0	13.0	14.0	16.0
2017-02-28 20:00:00	35061	13.0	37.0	7.0	45.0	500.0	81.0	11.6	1013.6	-15.1	...	13.0	18.0	20.0	27.0	21.0
2017-02-28 21:00:00	35062	16.0	37.0	10.0	66.0	700.0	58.0	10.8	1014.2	-13.3	...	9.0	23.0	16.0	22.0	17.0
2017-02-28 22:00:00	35063	21.0	44.0	12.0	87.0	700.0	35.0	10.5	1014.4	-12.9	...	10.0	23.0	11.0	9.0	11.0
2017-02-28 23:00:00	35064	19.0	31.0	10.0	79.0	600.0	42.0	8.6	1014.1	-15.9	...	13.0	30.0	15.0	12.0	11.0

شکل ۷- جایگذاری مقادیر Nan ایستگاه در دیتافریم شامل همه ایستگاه ها

۲-۳-۱. Encoding Categorical Variable

ستون wd را بر اساس مقاله با اضافه کردن مقادیر ۲۲,۵ درجه به عدد تبدیل میکنیم .

```
labels={
          'N':0,
         'NNE':22.5,
         "NE":45,
         "ENE":67.5,
         'E':90,
         'ESE':112.5,
         'SE':135,
         'SSE':157.5,
         'S':180,
         'SSW':202.5,
         'SW':225,
         'WSW':247.5,
         'W':270,
         'WNW':292.5,
         'NW':315,
         'NNW':337.5,
         'N':0,
         'North':0,
         'East':90,
         'West':270,
         'South':180
       }
```

شکل ۸- مقادیر تبدیل جهت به زاویه

year_month_day_hour	No	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	wd	WSPM	station
2013-03-01 00:00:00	1	4.0	4.0	4.0	7.0	300.0	77.0	-0.7	1023.0	-18.8	0.0	337.5	4.4	Aotizhongxin
2013-03-01 01:00:00	2	8.0	8.0	4.0	7.0	300.0	77.0	-1.1	1023.2	-18.2	0.0	0.0	4.7	Aotizhongxin
2013-03-01 02:00:00	3	7.0	7.0	5.0	10.0	300.0	73.0	-1.1	1023.5	-18.2	0.0	337.5	5.6	Aotizhongxin
2013-03-01 03:00:00	4	6.0	6.0	11.0	11.0	300.0	72.0	-1.4	1024.5	-19.4	0.0	315.0	3.1	Aotizhongxin
2013-03-01 04:00:00	5	3.0	3.0	12.0	12.0	300.0	72.0	-2.0	1025.2	-19.5	0.0	0.0	2.0	Aotizhongxin
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2017-02-28 19:00:00	35060	12.0	29.0	5.0	35.0	400.0	95.0	12.5	1013.5	-16.2	0.0	315.0	2.4	Aotizhongxin
2017-02-28 20:00:00	35061	13.0	37.0	7.0	45.0	500.0	81.0	11.6	1013.6	-15.1	0.0	292.5	0.9	Aotizhongxin
2017-02-28 21:00:00	35062	16.0	37.0	10.0	66.0	700.0	58.0	10.8	1014.2	-13.3	0.0	315.0	1.1	Aotizhongxin
2017-02-28 22:00:00	35063	21.0	44.0	12.0	87.0	700.0	35.0	10.5	1014.4	-12.9	0.0	337.5	1.2	Aotizhongxin
2017-02-28 23:00:00	35064	19.0	31.0	10.0	79.0	600.0	42.0	8.6	1014.1	-15.9	0.0	22.5	1.3	Aotizhongxin

35064 rows × 14 columns

شکل ۹- تبدیل جهت به زاویه

۱-۳-۳. Normalization

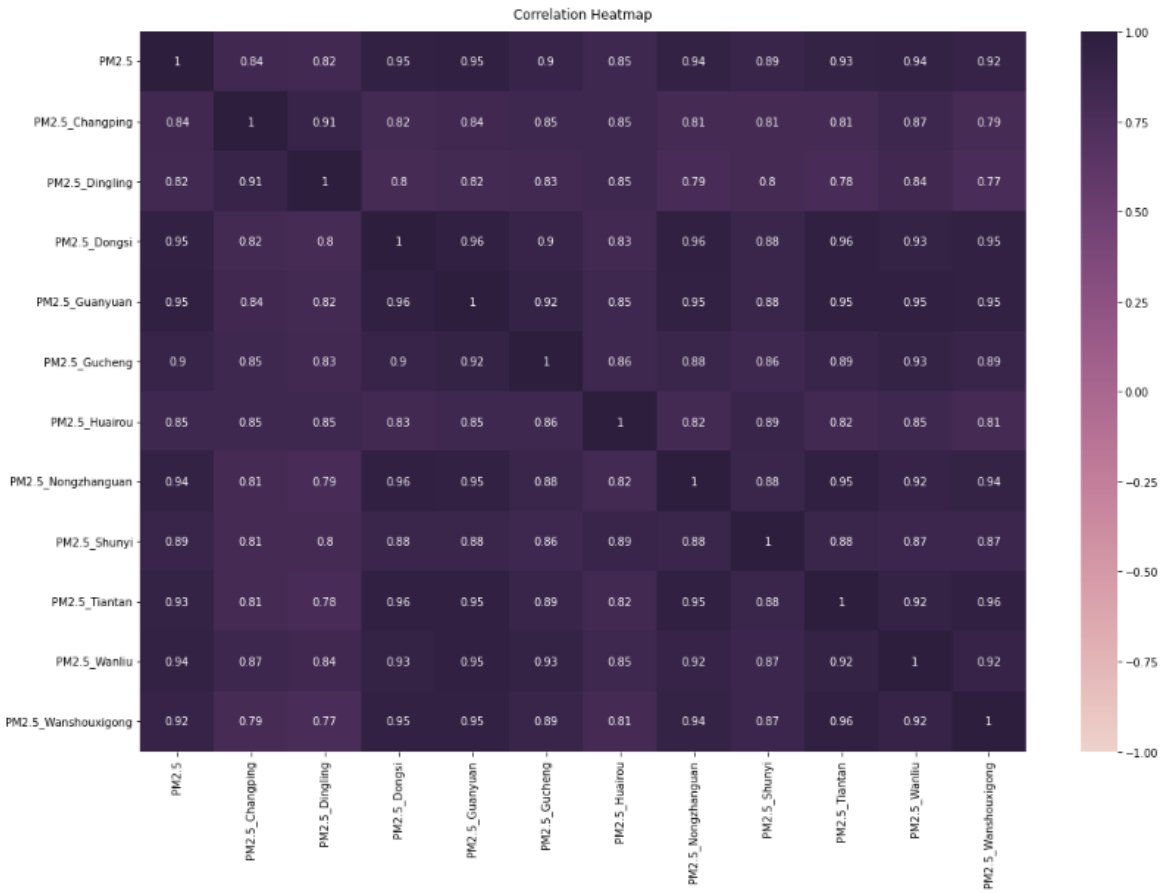
داده ها را با ابتدا به فرمت آرایه در می آوریم به کمک df.values سپس آنها را به کمک MinMaxScaler نرمالسازی میکنیم که نتایج آن را در زیر میبینید .

```
array([[0.00111732, 0.00203666, 0.02020202, ..., 0.00366748, 0.00628272,
        0.0060241 ],
       [0.00558659, 0.00610998, 0.02020202, ..., 0.00366748, 0.00732984,
        0.00803213],
       [0.00446927, 0.00509165, 0.02020202, ..., 0.00366748, 0.00104712,
        0.00502008],
       ...,
       [0.01452514, 0.03564155, 0.06060606, ..., 0.01833741, 0.01151832,
        0.01104418],
       [0.02011173, 0.04276986, 0.06060606, ..., 0.01466993, 0.0104712 ,
        0.00903614],
       [0.01787709, 0.02953157, 0.05050505, ..., 0.01466993, 0.0052356 ,
        0.01004016]])
```

شکل ۱۰- داده های نرمالسازی شده

۱-۳-۴. Pearson Correlation

همبستگی مربوط به Pm2.5 ایستگاه ها را نسبت به هم محاسبه میکنیم و نقشه حرارتی آنرا به نمایش در می آوریم همانطور که دیده میشود مقادیر همبستگی بالایی دارند .



شکل ۱۱- نقشه حرارتی همبستگی داده ها

۱-۳-۵. Feature Selection

مقادیر ستون های خواسته شده و ستون pm2.5 مربوط به همه ایستگاه ها را در یک دیتافریم میریزیم که تعداد ۲۰ ویژگی میشود و از آن برای آموزش مدل استفاده میکنیم . این فایل در یک اکسل جداگانه نیز در پوشه تمرین فرستاده شده است .

۱-۳-۶. Supervised dataset

داده ها را به به حالت supervised در می آوریم یکبار با مقدار lookback = 1*24 ,یکبار با 7*24 یعنی ۱۶۸ چون داده ها بر اساس ساعت دسته بندی شده اند این مقادیر lag ۱ و ۷ روز را برای ما ایجاد میکند . همچنین برچسب ها را که همان ستون ابتدایی است یعنی مقدار pm2.5 جدا میکنیم . سپس

داده ها را به نسبت ۲۰ به ۸۰ برای تست و آموزش جدا میکنیم ابعاد داده های تست و آموزش برای lag ۱ روز در زیر آمده است .

```
X_train (28052, 24, 20)
X_test (6988, 24, 20)
Y_train (28052, 1)
Y_test (6988, 1)
```

شکل ۱۲- ابعاد داده ها برای lag ۱ روز

اینکار را برای آموزش و تست مدل با lag ۷ روز نیز باید انجام دهیم که ابعاد داد ها برای lag ۷ روز در زیر آمده است .

```
X_train (28052, 168, 20)
X_test (6844, 168, 20)
Y_train (28052, 1)
Y_test (6844, 1)
```

شکل ۱۳- ابعاد داده ها برای lag ۷ روز

۴-۱. آموزش شبکه

مدل را مطابق مقاله به کمک keras میسازیم ابتدا لایه های مربوط به cnn سپس lstm و سپس لایه dense1

```
model = Sequential()

model.add(Conv1D(filters=64, kernel_size=3, padding="causal", activation='relu',input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(BatchNormalization())

model.add(Conv1D(filters=64, kernel_size=3, padding="causal", activation='relu'))
model.add(BatchNormalization())

model.add(Conv1D(filters=32, kernel_size=3, padding="causal", activation='relu'))

model.add(MaxPooling1D(pool_size = 3))

model.add(LSTM(100, return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(50))
model.add(Dropout(0.3))

model.add(Dense(1, activation="relu"))

adam = keras.optimizers.Adam(learning_rate=0.0001, decay=0.00001)

model.compile(loss='mean_squared_error',optimizer= adam)

model.summary()
```

شکل ۱۴- کد ساخت مدل

Summary مدل برای lag ۱ روز در زیر آمده است .

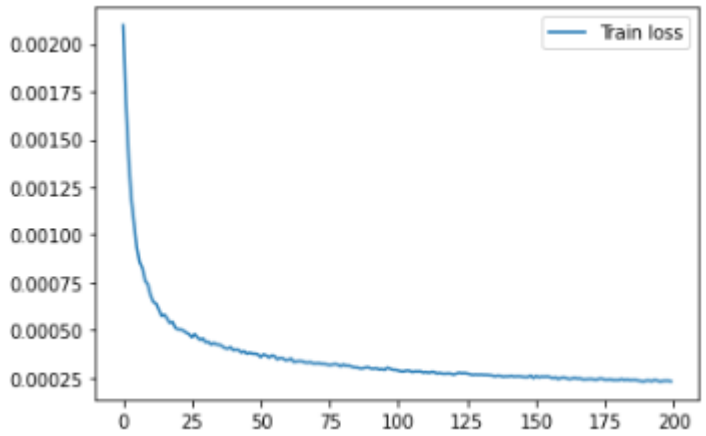
```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_9 (Conv1D)           (None, 24, 64)            3904

 batch_normalization_6 (Batc  (None, 24, 64)           256
 hNormalization)

 conv1d_10 (Conv1D)          (None, 24, 64)            12352

 batch_normalization_7 (Batc  (None, 24, 64)           256
 hNormalization)

 conv1d_11 (Conv1D)          (None, 24, 32)            6176

 max_pooling1d_3 (MaxPooling  (None, 24, 10)           0
 1D)

 lstm_6 (LSTM)               (None, 24, 100)           44400

 dropout_6 (Dropout)         (None, 24, 100)           0

 lstm_7 (LSTM)               (None, 50)                30200

 dropout_7 (Dropout)         (None, 50)                0

 dense_3 (Dense)             (None, 1)                 51

=================================================================
Total params: 97,595
Trainable params: 97,339
Non-trainable params: 256
```

شکل ۱۵- لایه های مدل برای ورودی با lag ۱روز

Learning_rate و decay باید پایین تنظیم شوند تا مدل به درستی آموزش ببیند .مدل را در ۲۰۰ ایپاک و batch_size = 32 آموزش میدهیم و مقدار loss گام به گام مطابق زیر کاهش پیدا میکند .

```
Epoch 1/200
877/877 [==============================] - 10s 8ms/step - loss: 0.0058
Epoch 2/200
877/877 [==============================] - 7s 8ms/step - loss: 0.0028
Epoch 3/200
877/877 [==============================] - 7s 8ms/step - loss: 0.0020

…

…

Epoch 198/200
877/877 [==============================] - 8s 9ms/step - loss: 2.3763e-04
Epoch 199/200
877/877 [==============================] - 7s 8ms/step - loss: 2.3540e-04
Epoch 200/200
877/877 [==============================] - 7s 8ms/step - loss: 2.3553e-04
```

نمودار loss در هر ایپاک را رسم میکنیم روند کاهشی است .



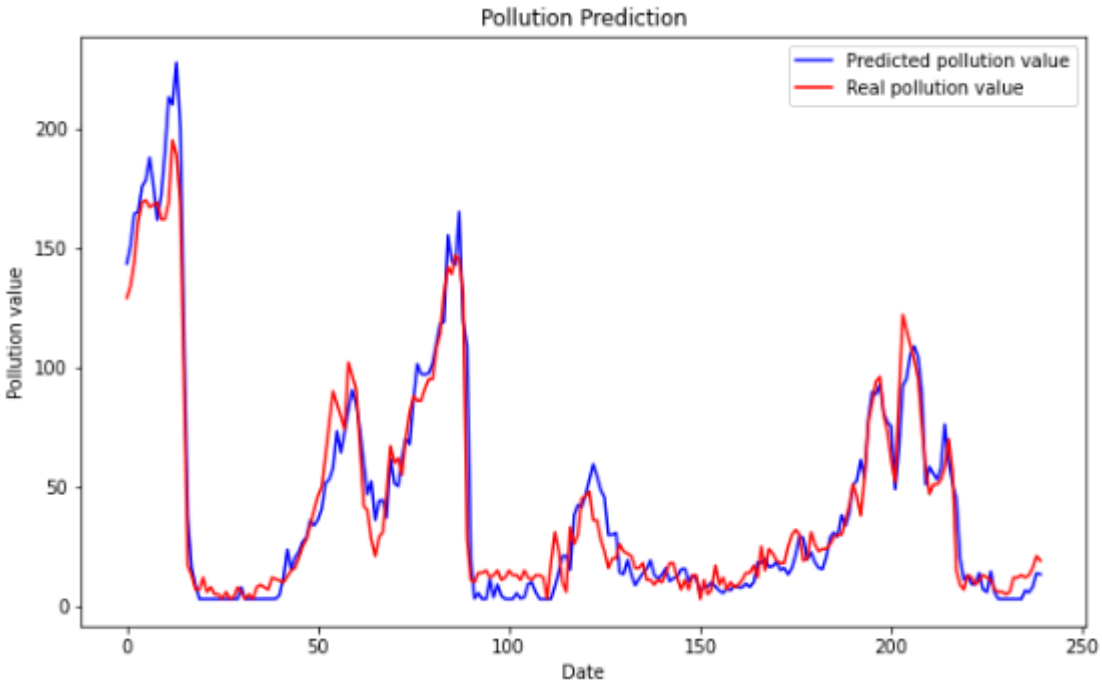
شکل ۱۶- loss در هر ایپاک با lag ۱ روز

مدل به شکل مناسبی آموزش میبیند و loss آن کاهش میابد . سپس به کمک مدل آموزش دیده مقادیر را پیشبینی میکنیم و آماره های مورد نظر را برای lag ۱روز محاسبه میکنیم که در زیر درج شده است

```
mae: 12.124641135182502
RMSE: 19.50350009928017
R2: 0.9456964876799547
```

شکل ۱۷- آماره ها برای lag ۱ روز

سپس نمودار مربوط به ۱۰ روز از داده های تست که مقدار pm2.5 آنها توسط مدل پیشبینی شده است را رسم میکنیم تا کارایی مدل برای پیشبینی آلودگی را مشاهده کنیم .



شکل ۱۸- پیشبینی آلودگی توسط مدل با lag ۱

سپس ورودی ها را برای lag ۷ روز supervised و تنظیم میکنیم و مدل را دوباره میسازیم که summary آن در زیر آمده است .

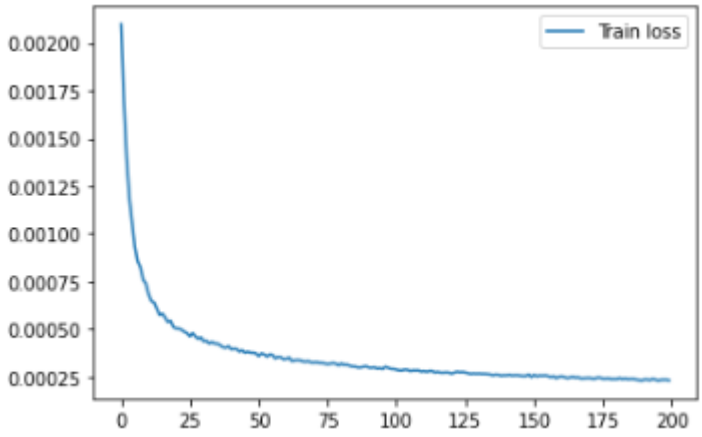
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 168, 64)           3904

 batch_normalization (BatchN  (None, 168, 64)          256
 ormalization)

 conv1d_1 (Conv1D)           (None, 168, 64)           12352

 batch_normalization_1 (Batc  (None, 168, 64)          256
 hNormalization)

 conv1d_2 (Conv1D)           (None, 168, 32)           6176

 max_pooling1d (MaxPooling1D  (None, 168, 10)          0
 )

 lstm (LSTM)                 (None, 168, 100)          44400

 dropout (Dropout)           (None, 168, 100)          0

 lstm_1 (LSTM)               (None, 50)                30200

 dropout_1 (Dropout)         (None, 50)                0

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 97,595
Trainable params: 97,339
Non-trainable params: 256
_________________________________________________________________
```

شکل ۱۹- summary مدل برای ورودی با lag ۷ روز

مدل را در ۲۰۰ ایپاک و batch_size = 32 آموزش میدهیم و مقدار loss گام به گام مطابق زیر کاهش پیدا میکند .

```
Epoch 1/200
877/877 [==============================] - 10s 11ms/step - loss: 0.0021
Epoch 2/200
877/877 [==============================] - 10s 11ms/step - loss: 0.0017
Epoch 3/200
877/877 [==============================] - 10s 11ms/step - loss: 0.0014

…

…

Epoch 198/200
877/877 [==============================] - 10s 11ms/step - loss: 2.3465e-04
Epoch 199/200
877/877 [==============================] - 10s 11ms/step - loss: 2.3554e-04
Epoch 200/200
877/877 [==============================] - 10s 11ms/step - loss: 2.3242e-04
```

نمودار loss در هر ایپاک را رسم میکنیم تا روند آنرا ببینیم .



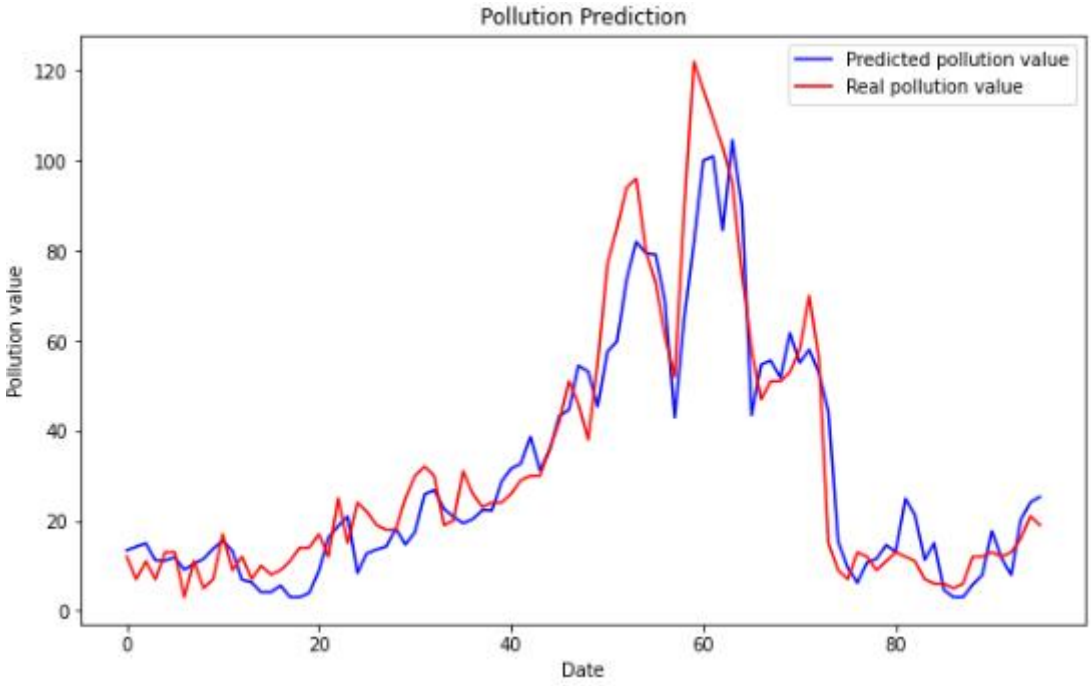
شکل ۲۰- loss در هر ایپاک با lag ۷ روز

میبینیم که مدل به خوبی آموزش میبیند . سپس به کمک مدل آموزش دیده مقادیر را پیشبینی میکنیم و آماره های مورد نظر را برای lag ۷روز محاسبه میکنیم که در زیر درج شده است .

```
mae: 12.8569274381768
RMSE: 20.957371707660158
R2: 0.9383186915182291
```

شکل ۲۱- آماره ها برای lag ۷ روز

سپس نمودار مربوط به ۱۰ روز از داده های تست که مقدار pm2.5 آنها توسط مدل با لگ ۷ روز پیشبینی شده است را رسم میکنیم تا نحوه ی پیشبینی را روی نمودار ببینیم .



شکل ۲۲- پیشبینی آلودگی توسط مدل با lag ۷

lag ۱ روز بهتر از lag ۷ روز عمل میکند . با بالا رفتن مقدار lag عملکرد مدل تضعیف میشود . و این نشان میدهد که باید از داده هایی که به لحاظ زمانی نزدیکترند برای پیشبینی آلودگی استفاده نمود و داده های قبلی به دلیل تغییر شرایط جوی کمتر برای پیشبینی الودگی مفید اند در کل مدل عملکرد مناسبی برای پیشبینی مقدار pm2.5 دارد و روند کلی میزان این آلاینده را مناسب پیشبینی میکند .

پاسخ ۲ . تشخیص اخبار جعلی

۲-۱. توضیحات مدل‌ها

تفاوت RNN و LSTM :

شبکه های حافظه طولانی کوتاه مدت (Long Short-Term Memory) یک نسخه بهبود یافته از شبکه‌های عصبی بازگشتی(Recurrent Neural Network) هستند که به داده‌ها، «وزن‌هایی» را اختصاص می‌دهند و به RNN این امکان را می‌دهند تا اطلاعات جدید را وارد کند، اطلاعات را فراموش کند و یا به آن ها اهمیت کافی دهد تا روی خروجی اثر بگذارد. در واقع آن ها شبکه‌های RNN را قادر می‌سازند تا ورودی‌ها را در مدت زمان طولانی به خاطر بسپرند. چرا داده های متنی به ویژگی بازگشتی بودن نیاز دارند؟

به این دلیل که در متن ما نیاز به حفظ وابستگی بلند مدت بین کلمات داریم. به این صورت که کلمه اول جمله با کلمه آخر جمله که فعل آن می باشد وابستگی دارد و این وابستگی باید حفظ بشود به همین دلیل از شبکه های بازگشتی استفاده می کنیم که اطلاعات گذشته را حفظ می کنند.

مدل پیشنهادی از CNN برای استخراج ویژگی های محلی(local features) و از LSTM برای یادگیری وابستگی‌های بلندمدت استفاده می‌کند. ابتدا، یک لایه CNN از Conv1D برای پردازش بردارهای ورودی و استخراج ویژگی های محلی که در سطح متن قرار دارند استفاده می شود. خروجی لایه CNN (feature maps ها) ورودی لایه RNN هستند. در واقع لایه RNN از ویژگی های محلی که توسط CNN استخراج شده استفاده می کند و وابستگی های بلندمدت بین کلمات را می آموزد.

۲-۲. ورودی مدل

Word embedding ها بردار های عددی هستند که نمایانگر کلمات یک لغت نامه اند. سه تکنیک embedding با نام‌های Word2vec، GloVe و fastText معرفی شده است که در تمام این تکنیک‌ها، شباهت معنایی (semantic similarity) بین کلمات حفظ می‌شود؛ به عبارتی دیگر، با بردارهای به‌دست‌آمده از این تکنیک‌ها، می‌توان معنای کلمات را تشخیص داد و میزان شباهت کلمات مختلف را با یکدیگر به دست آورد. من از روش word2vec استفاده کردم. در این روش به کمک شبکه عصبی یک بردار با اندازه کوچک و ثابت برای نمایش تمام لغات و متون در نظرگرفته شده و با اعداد مناسب در فاز آموزش مدل یا training برای هر لغت این بردار محاسبه می شود برای افزایش دقت این روش، مجموعه

داده اولیه که برای آموزش مدل مورد نیاز است، باید حدود چند میلیارد لغت را که درون چندین میلیون سند یا متن به کار رفته اند، در برگیرد. بعد از ایجاد بردارهای مرتبط با هر لغت، برای نمایش برداری هر متن یا خبر ، می توان بردار تک تک کلمات به کار رفته در آنرا یافته و میانگین اعداد هر ستون را به دست آورد که نتیجه آن یک بردار برای هر متن یا سند خواهد بود.

۲-۳. پیاده‌سازی مدل

۲-۳-۱. پیش‌پردازش

طبق آنچه در مقاله آمده است ابتدا داده را خواندم و جملات متن مقاله هارا را پاکسازی کردم به این معنی که تمام URLها و stop word ها را حذف کردم و سپس داده توکنایز شده را به دو دسته آموزش و تست با نسبت ۰,۲ تقسیم کردم.

```
new_df = pd.read_csv('/content/drive/My Drive/Q2/FA-KES-Dataset.csv', encoding= 'unicode_escape')
```

```
column_n = ['unit_id',  'article_title',  'article_content',  'source', 'date'  ,'location',  'labels']
remove_c = ['unit_id',  'article_title','source', 'date'  ,'location']
categorical_features = []
target_col = ['labels']
text_f = ['article_content']
```

شکل ۲۳- خواندن داده‌ها و حذف ستون‌های غیرمرتبط

توابع مربوط به تمیزسازی متن را در شکل زیر مشاهده می‌کنید.

```
def clean_dataset(df):
    # remove unused column
    df = remove_unused_c(df)
    #impute null values
    df = null_process(df)
    return df


def clean_text(text):
    text = str(text).replace(r'http[\w:/\.]+', ' ')  # removing urls
    text = str(text).replace(r'[^\.\w\s]', ' ')
    return text


def nltk_preprocess(text):
    text = clean_text(text)
    wordlist = re.sub(r'[^\w\s]', '', text).split()
    text = ' '.join([wnl.lemmatize(word) for word in wordlist if word not in stopwords_dict])
    return  text
```

شکل ۲۴- توابع مربوط به تمیزسازی متن

۲-۳-۲. آموزش مدل‌ها

داده های پیش پردازش شده را با دو مدل آموزش دادم:

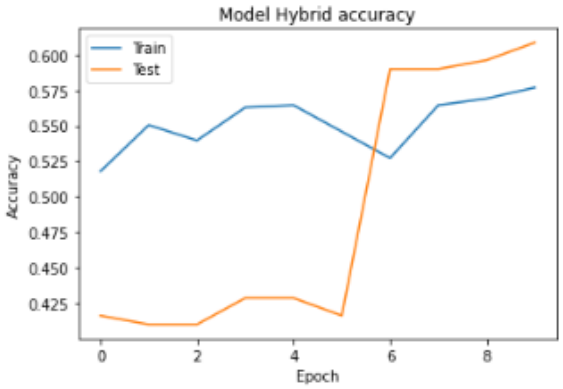
مدل اول : Hybrid(CNN-RNN)

تمام لایه ها و پارامتر ها منطبق بر مقاله پیاده سازی شد.

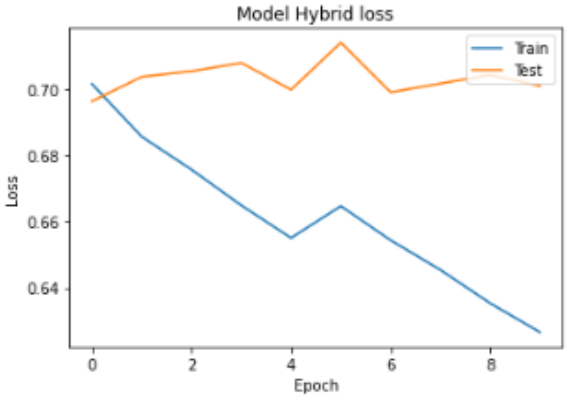
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_6 (Embedding)      (None, None, 100)         1041700

conv1d_2 (Conv1D)            (None, None, 128)         64128

max_pooling1d_2 (MaxPooling  (None, None, 128)         0
1D)

lstm_5 (LSTM)                (None, 32)                20608

dense_6 (Dense)              (None, 1)                 33

=================================================================
Total params: 1,126,469
Trainable params: 84,769
Non-trainable params: 1,041,700
```

شکل ۲۵- مشخصات مدل **Hybrid**

مدل نوشته شده را طی ۱۰ ایپاک آموزش دادم که نمودار های دقت و خطا و نتایج مدل در شکل‌های زیر قابل مشاهده می باشد.



شکل ۲۶- نمودار دقت مدل **Hybrid**



شکل ۲۷- نمودار **loss** مدل **Hybrid**

همچنین نتایج F1_score، recall، Precision و Accuracy این مدل را در شکل زیر مشاهده می‌کنید.

	precision	recall	f1-score	support
0	0.42	1.00	0.59	67
1	0.00	0.00	0.00	94
accuracy			0.42	161
macro avg	0.21	0.50	0.29	161
weighted avg	0.17	0.42	0.24	161

شکل ۲۸ – خلاصه‌ی عملکرد مدل **Hybrid** روی داده‌های تست

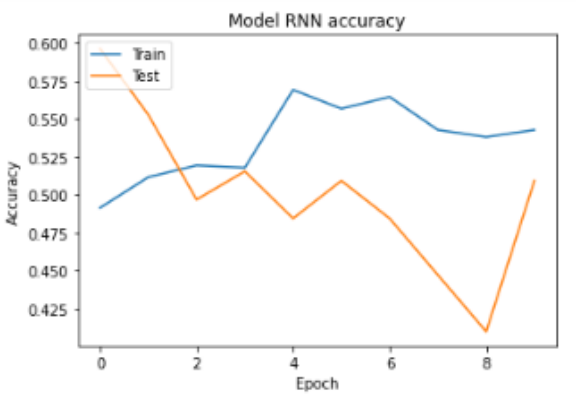
مدل دوم: RNN

مشخصات این مدل را در شکل زیر مشاهده می‌کنید.

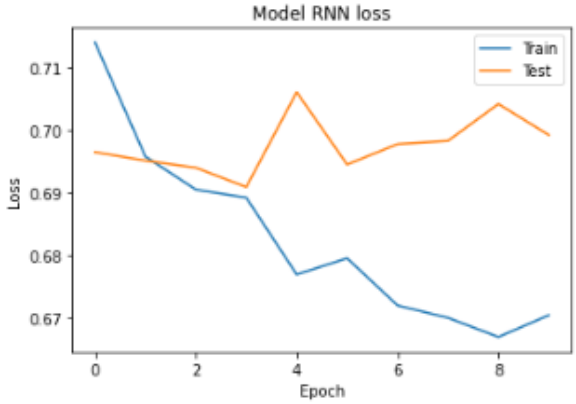
```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_5 (Embedding)     (None, None, 100)         1041700

 simple_rnn_1 (SimpleRNN)    (None, 32)                4256

 dense_5 (Dense)             (None, 1)                 33

=================================================================
Total params: 1,045,989
Trainable params: 4,289
Non-trainable params: 1,041,700
_________________________________________________________________
Epoch 1/10
```

شکل۲۹ - مشخصات مدل **RNN**

این مدل را هم برای ۱۰ مرحله روی داده های پیش پردازش شده آموزش دادم. نمودار accuracy و loss این مدل را در شکل‌های زیر مشاهده می‌کنید.



شکل ۳۰- نمودار دقت مدل **RNN**



شکل ۳۱- نمودار **loss** مدل **RNN**

همچنین نتایج F1_score، recall، Precision و Accuracy این مدل را در شکل زیر مشاهده می‌کنید.

	precision	recall	f1-score	support
0	0.42	1.00	0.59	67
1	0.00	0.00	0.00	94
accuracy			0.42	161
macro avg	0.21	0.50	0.29	161
weighted avg	0.17	0.42	0.24	161

شکل ۳۲- خلاصه‌ی عملکرد مدل روی داده‌های تست

همانطور که مشاهده کردید دقت و خطای مدل Hybrid نسبت به مدل RNN ساده بهبود داشته است و این به دلیل استفاده از شبکه CNN می باشد.

۴-۲. تحلیل نتایج

فکر می کنم بزرگترین مشکل این پیاده سازی در کم بودن تعداد داده ها می باشد و اگر از دیتاست بزرگتری استفاده کنیم نتایج بسیار بهتری خواهیم داشت . بهترین نتیجه زمانی پدیدار می شود که دیتاست استفاده شده حجم بالایی داشته باشد و بالانس نیز باشد. تعداد داده های کلاس ها متناسب باشد. اگر تعداد داده های داده ۰ بیشتر باشد مدل به سمتی می رود که همه داده هارا ۰ پاسخ دهد در صورتی که در این نوع مسائل recall اهمیت بالایی دارد و نمیخواهیم متن فیک تشخیص داده نشده بماند.