REPORT ON

Efficiency of different ML/DL models in predicting Air Quality

BY

Name of the Student	ID Number
Lakshay Munjal	***2019A7PS0047P***

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

April 2021

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE PILANI (RAJASTHAN)

Abstract:

Owing to global industrialization and urbanisation, air pollution has been one of the most serious environmental problems in the twenty-first century. Accurate air quality forecasts are needed for its mitigation. Deep learning has recently emerged as a general-purpose technology for extracting complex information from vast amounts of data and very large networks of neurons, and hence has the ability to push air quality forecasting to new heights. We provide a quick rundown of recent attempts to use deep learning approaches in air quality forecasting. First we define various methods and architectures of deep networks and their relevance to explore the non-linear spatio-temporal features across multiple scales of air pollution. We then compare these with various supervised machine learning models to see how they hold up against traditional regression approaches. We finally give a conclusion and discuss the viability of these networks in Air Quality Forecasting.

TABLE OF CONTENTS

1. Introduction

Air Pollution is a major environmental problem that is gaining worldwide notice and is a major issue in many developing countries. Obtaining real-time air quality data is critical for air pollution management and shielding us from the negative health effects of pollution. As a result, air quality estimation is needed to better represent the evolving pattern of air emissions, provide timely and full environmental quality information for environmental management decisions, and prevent severe air pollution incidents.

Recently, deep learning, a new potential machine learning methodology, has attracted considerable academic and industrial attention and has been successfully applied to image classification, natural language processing, prediction task, object detection, artificial intelligence, motion modeling, etc.

Deep learning algorithms use multilayered frameworks to extricate the inherent properties of the data layer by layer from the lowest to the highest level and to identify emblematic structures in the data. The innate complexity of the air quality process causes its spatial distribution and over-the-time trends to be influenced by numerous factors , air pollutant emissions and deposits, weather conditions, traffic flow, human activities being a few of many. This situation has created complications in using conventional flat models, when used particularly to provide a decent rendition of air quality characteristics. These algorithms can extract representative air quality features without prior knowledge, leading to a more efficient interpretation of air quality predictions.

2. Literature Survey

Multiple linear regression (MLR) (Li et al. 2011) model and the auto regression moving average (ARMA) model are commonly used for air quality prediction. However, these methods usually yield limited accuracy due to their inability to model nonlinear patterns. A promising alternative is ANNs like SVR (Nieto et al. 2013).

Deep networks' architectures assess their ability to derive complex nonlinear features from high-dimensional datasets across scales. Deep artificial neural networks are distinguished from shallow artificial neural networks by the presence of several layers of neurons. Qi Liao et al. 2020 uses Simple RNN, LSTM and a GRU network to build the temporal portion of the spatiotemporal model. Xiang Li et al. 2016 uses an LR paired with a 4 layer stacked autoencoder to create a spatiotemporal model.

3. Dataset

The concentration of **PM2.5, PM10, NO, NO2, NOx, NH3, SO2, CO, Ozone** and **Benzene** for the city of New Delhi was downloaded from the official website of the Central Pollution Control Board. The Dataset is specifically from the station at **Dwarka-Sector 8, Delhi**, under

the DPCC. The data is recorded at every **15 min interval** from **1st January 2019** to **1st January 2021**, equating to **70176 lines** of data.

From Date	To Date	PM2.5	PM10	NO	NO2	NOx	NH3	SO2	CO	Ozone	Benzene
01-01-2019 00:00	01-01-2019 00:15	509	838	192.9	58.9	188.7	62.5	18.2	5.9	1.8	14.8
01-01-2019 00:15	01-01-2019 00:30	509	838	223.8	57	213.1	61.1	16.2	6	2.3	14.1
01-01-2019 00:30	01-01-2019 00:45	509	838	192.4	64.7	191.2	67	16.1	5.2	2.3	13.8
01-01-2019 00:45	01-01-2019 01:00	417	659	220	63.1	213.1	66.5	15.5	5.4	2.3	13.6
01-01-2019 01:00	01-01-2019 01:15	417	659	211	56.2	202.1	65.8	15.3	5	None	12.6
01-01-2019 01:15	01-01-2019 01:30	417	659	196.4	62.5	193.4	66.5	14.8	4.5	None	12.7

The gaps in the data are filled by linear interpolation so as the preserve the fluctuations in the concentration. Finally, the data is normalized by using minmax normalization.

From Date	PM2.5	PM10	NO	NO2	NOx	NH3	SO2	CO	Ozone	Benzene
2019-01-01 00:00:00	0.548004	0.837675	0.386063	0.183578	0.377457	0.311067	0.099945	0.59	0.008504	0.391534
2019-01-01 00:15:00	0.548004	0.837675	0.447938	0.177646	0.426394	0.304088	0.088901	0.6	0.011006	0.373016
2019-01-01 00:30:00	0.548004	0.837675	0.385062	0.201686	0.382471	0.3335	0.088349	0.52	0.011006	0.365079
2019-01-01 00:45:00	0.448759	0.658317	0.440328	0.196691	0.426394	0.331007	0.085036	0.54	0.011006	0.359788
2019-01-01 01:00:00	0.448759	0.658317	0.422307	0.175148	0.404332	0.327517	0.083932	0.5	0.011131	0.333333
2019-01-01 01:15:00	0.448759	0.658317	0.393072	0.194817	0.386883	0.331007	0.081171	0.45	0.011256	0.335979
2019-01-01 01:30:00	0.448759	0.658317	0.272928	0.222292	0.296831	0.357428	0.081171	0.3	0.011381	0.309524
2019-01-01 01:45:00	0.382956	0.591182	0.204646	0.211364	0.238067	0.322532	0.075097	0.27	0.011506	0.272487
2019-01-01 02:00:00	0.382956	0.591182	0.184622	0.209179	0.221119	0.303838	0.06778	0.25	0.009005	0.21164

4. Various Models

K-Nearest Neighbor method assign a predicted value to a new observation based on the plurality or mean (sometimes weighted) of its k "Nearest Neighbors" in the training set. Given an infinite amount of data, any observation will have many "neighbors" that are arbitrarily near with respect to all measured characteristics, and the variability of their outcomes will provide as precise a prediction as is theoretically possible barring a perfectly and completely specified model. However, given that we never have an infinite amount of data, the actual utility of this asymptotic property is questionable, especially for modest datasets.

SVR uses the same basic idea as Support Vector Machine (SVM), a classification algorithm, but applies it to predict real values rather than a class. SVR acknowledges the presence of non-linearity in the data and provides a proficient prediction model.

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

Recurrent Neural Networks are variants of feedforward neural networks but have self-connection of neurons cyclic structure into the network. Thus, input data can be memorized, and sequences of data can influence network outputs through self-connected neurons. Taking advantage of their memory characteristics, RNNs outperform FNNs in many applications. However, RNNs may fail to capture long time dependencies in input data, and it may face the problems of vanishing and exploding gradients when the time of training is too long.

Long Short-Term Memory Networks are enhanced RNNs. They introduce memory blocks to overcome the vanishing gradient problem. The memory blocks consist of three types of nonlinear multiplicative gates: the input gate, output gate, and forget gate. The multiplicative gates control the memory block operation and determine whether the input information needs to be remembered. The input gate controls the flow of cell activation from input into a memory cell, while output gate controls the flow of output from a memory cell into other nodes.

Gated Recurrent Unit networks are simplified versions of the LSTM networks. They only consist of update and reset gates but can still balance the data flows inside the unit [10]. The update gate replaces the input and forgets gates in LSTM, which determines whether information needs to be remembered. The advantage of using GRU compared with LSTM is that GRU have fewer parameters and thus less computational loads for training.

5. Model definitions and Observations

To preserve the temporal relationships in the ML models, the data from t-16 to t-1 units is included as the features of the t^{th} unit. This way the context formed by the recent history of an input can be used to more accurately predict its future. Data appears of the form:

t-12	t-11	t-10	t-9	t-8	t-7	t-6	t-5
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	687.0
NaN	NaN	NaN	NaN	NaN	NaN	687.0	646.0
NaN	NaN	NaN	NaN	NaN	687.0	646.0	-189.0
NaN	NaN	NaN	NaN	687.0	646.0	-189.0	-611.0
NaN	NaN	NaN	687.0	646.0	-189.0	-611.0	1339.0
NaN	NaN	687.0	646.0	-189.0	-611.0	1339.0	30.0
NaN	687.0	646.0	-189.0	-611.0	1339.0	30.0	1645.0
687.0	646.0	-189.0	-611.0	1339.0	30.0	1645.0	-276.0

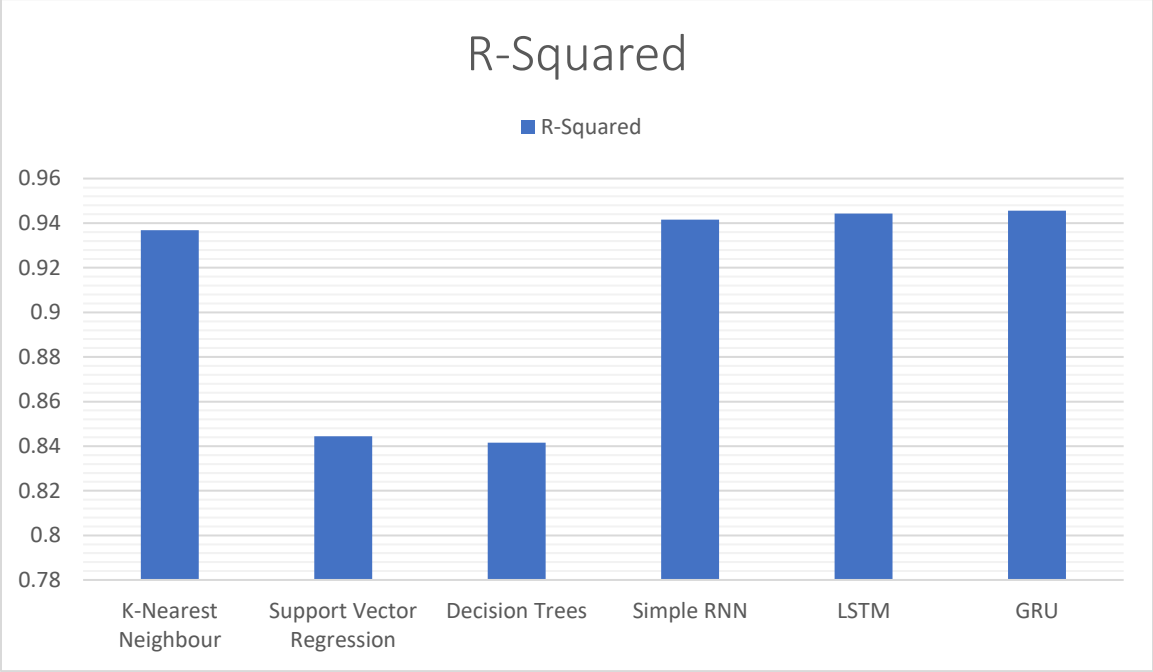
t-4	t-3	t-2	t-1	t
NaN	NaN	NaN	NaN	687.0
NaN	NaN	NaN	687.0	646.0
NaN	NaN	687.0	646.0	-189.0
NaN	687.0	646.0	-189.0	-611.0
687.0	646.0	-189.0	-611.0	1339.0
646.0	-189.0	-611.0	1339.0	30.0
-189.0	-611.0	1339.0	30.0	1645.0
-611.0	1339.0	30.0	1645.0	-276.0
1339.0	30.0	1645.0	-276.0	561.0
30.0	1645.0	-276.0	561.0	470.0
1645.0	-276.0	561.0	470.0	3395.0
-276.0	561.0	470.0	3395.0	360.0
561.0	470.0	3395.0	360.0	3440.0

SVR being can only provide single outputs, therefore, to obtain a multi-featured output, the model is wrapped in a Regressor Chain.

The RNN, LSTM and GRU models each contain a single recurrent segment of 32 units followed by a Dense layer of size 10. They are trained for 30 epochs each in batches of 32.

The Dataset is split into train and test sets in the ratio 3:1. Metric for accuracy is R-Squared: $R^2 = 1 - \frac{n \times MSE}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$.

Method	R-Squared
K-Nearest Neighbour	0.9369093648907126
Support Vector Regression	0.8444477518904032
Decision Trees	0.8416144196041311
Simple RNN	0.9416489590067089
LSTM	0.944267206509914
GRU	0.9456603784676176



The RNN, LSTM and GRU models perform consistently better than SVR and D-trees, even without pre-trained weights and only 15 epochs of training.

5. Conclusion

Machine Learning models, although efficient enough on their own, don't extract high dimensional information from the data nearly as efficiently as the Deep Learning models. Deep learning is appealing in its potential to improve air quality forecasts in diverse aspects. Its use in air quality forecasting is only in its early stages, and it faces several obstacles.

For starters, there is a data discrepancy that prevents systematic research that lead to breakthroughs in forecasting efficiency improvements. To date, there are no general datasets for testing deep learning algorithms such that benchmarking can be done to gather information for better predictions.

Secondly, given comprehensive datasets, a variety of deep networks and learning algorithms must be compared and improved in order to retrieve complex nonlinear features across scales concealed within such datasets.

Deep networks are often criticized for being opaque and difficult to understand but learning the dynamics of pollutants will allow the underlying deep networks to catch some cause-and-effect interactions as forecasting output improves.

6. References

- Recurrent Neural Networks (RNN) with Keras | TensorFlow Core
- 1.4. Support Vector Machines — scikit-learn 0.24.2 documentation
- 1.6. Nearest Neighbors — scikit-learn 0.24.2 documentation
- 1.10. Decision Trees — scikit-learn 0.24.2 documentation
- Li et al. 2011
- Qi Liao el al. 2020
- Nieto et al. 2013
- Xiang Li et al. 2016
- https://machinelearningmastery.com/feature-selection-time-series-forecasting-python/