JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Detection of Air Pollution Hotspots and its Sources using Deep Learning

[1]Annette John, [2]Priyansh Jain, [3]Sharon Raju, [4]Vinit Shah

Department of Computer Engineering,
AISSMS Institute of Information Technology, Pune, India

Abstract: Air pollution is a critical global environmental hazard, which is estimated to cause seven million deaths across the globe annually. Earlier studies have only focused on PM2.5 pollutants and have used data from costly sensors having sparse spatial resolutions. The proposed method tackles this with an interdisciplinary approach using Earth and Health sciences that integrates different datasets to study the causes and effects of air pollution by using freely available satellite data that have finer spatial resolutions; and also considers gases like NO2, CO, etc. The primary objective is to find the probable sources of air pollution in a cost-effective manner, and produce future projections based on the temporal data. We achieve this by deploying Artificial Intelligence techniques such as DBSCAN, K-Means, Time Series Analysis, and LSTM - on public datasets from the ESA and WHO. We performed an analysis of the Nationwide Lockdown on AQI levels. Our model acquired an accuracy of 88% on the training set and 90.7% on the validation set, respectively.

IndexTerms **- Deep learning, Artificial Intelligence, Kaggle Dataset, LSTM, KMeans, DBSCAN, Geospatial, WHO, ESA, Crowdsourcing, Satellite data, Sentinel 5P, AQI**

I. INTRODUCTION

Our planet is plagued by environmental problems and if left unchecked, it can affect livelihoods. Air Pollution in particular has become a serious cause of concern owing to the massive increase in the number of upcoming industries, burning of fossil fuels, and open-field burning. The global toll of premature deaths caused due to the careless combustion of coal, diesel, and gasoline are increasingly high, and these numbers only keep rising with every passing year [1]. The existing monitoring systems have very few measurement stations that are able to accurately predict the air quality indices for cities. In order to increase the accuracy of these predictions, the installation of a large number of expensive sensors would be required [2]. This poses a tradeoff between cost and accuracy[3]; Moreover, currently India has merely 308 ground stations that are situated only in the urban areas [4] which makes it difficult to accurately predict all the probable hotspot locations. A feasible solution to all these problems would be to leverage free satellite data. Satellite provides columnar concentrations of pollutants on a daily basis. Challenges involved in the current statement are - mining the data sets from different satellite parameters and providing the final output with moderate spatial resolution on pollution information. Our solution will help identify the pollution hotspots, and their sources. This will help us to uncover patterns and trends, and also prove helpful to take corrective actions for major problematic areas. The remainder of our paper is structured in the following order - Section 2 lists the Related Work to our project, Section 3 elaborates on the proposed methodology adopted to execute our solution, Section 4 analyzes the Experimental Results of our model. In Section 5 we demonstrate the Visualization Tool developed by us followed by the Conclusion in Section 6, which is succeeded by the Acknowledgements and References.

II. RELATED WORK

G.Peter Zhang [5] proposed a hybrid system for forecasting time series that uses ARIMA & ANN. In this approach, future values of variables are presumed to be a linear function of multiple past observations & random errors. Both these models achieved favorable results in their linear and nonlinear areas of study respectively, but failed to capture the complete information. Although, the hybrid models are better equipped to pick up the patterns..

Wald et.al [6] investigated the potential capabilities of satellite imagery with the intention of mapping air quality parameters and the concentrations of various pollutants obtained from ground stations in Nantes - a city in Europe and compared this to data extracted from Landsat TM6. The ground instruments capture the data which can be locally accurate. However, the location of the instrument plays a very important role. Also, the instruments cannot cover the area as much as satellites can.

Nevin et.al [7], proposed a model on the Fuzzy Time Series - which is based on Fuzzy K-Medoid (FKM) clustering. Fuzzy time series models are majorly composed of 3 steps: Initially, values such as number of clusters (cl), fuzziness-index (fi > 1), termination criteria, and the number of iterations (itr) are decided. cl number of initial medoids are selected from the dataset. The paper shows the advantages of using the Fuzzy K-Medoid algorithm over models like Fuzzy C-Means and Gustafson-Kessel.

Ghude et.al [8] used a multi functional regression model to study the behaviors of the seasonal cycle over the NO2 emission hotspots located within India. The satellite measurements are able to measure the increase of tropospheric NO2 concentration. However, the emission inventories located in India are unpredictable owing to the dearth of accurate statistics & emission attributes. We can use satellite measurements that have a higher accuracy in order to rightly identify the larger hotspots of India.

P. Kumar et.al [9] illustrate the applications of low-cost meteorological sensors to manage air pollution in urban city areas. The sensors are calibrated in 2 steps, and these sensors are capable of operating for two years to detect hot spots, and perform necessary evaluations. However, costs involved in their installation, maintenance and data analysis are high, and the development of sensors that can measure nano-sized particles is still under research.

Howard H. Chang et.al [10] examine the interdependencies that relate asth-ma morbidity to air pollution. They implemented a time-series analysis of the number of admissions to an asthma hospital & pollution attributed to PM2.5 in the city of Jackson, Mississippi. However, this methodology can have anomalies and outliers. Also, the study focuses on a small area of Jackson Mississippi which might not con-cur to other places.

C. Yu [11] investigated the time series air quality monitoring data based on exploratory data analysis. They assessed 3 aspects of air quality: Trends detected in data, existing time changes in that data, and the combination of factors related to the weather data. Using this methodology, we can find patterns that are hidden in the air quality data, and with exploratory analysis, we can study the effects of pollutants on larger scaled areas.

Rahman et.al [12] forecasted the Air Pollution Index with Artificial Neural Networks based on Air Pollution Index data of monitoring stations situated in Malaysia. The Fuzzy time-series (FTS), Autoregressive-integrated moving average (ARIMA), & artificial neural network (ANN) were the models chosen for forecasting the values of API. ANNs were found to be the most effective, which was followed by FTS. ARIMA had a major downside as it was only capable of capturing linear time-series data.

Most of the previous studies have been focusing only on PM2.5 pollutants and have been using data from costly sensors and instruments or data that has a sparse spatial resolution. Also, they have taken into consideration small scale areas. Our proposed method aims to mitigate these shortcomings by considering other pollutant gases like NO2, CO, etc and eliminating the use of costly sensors by making use of freely available satellite data which will enable us to work with data having finer spatial resolution. We will be focusing on a larger scale of area, and averaging data over a shorter period of time to represent the outcome more effectively

III. Research Methodology

3.1 Data Collection

The data from Sentinel-5P(Sentinel 5-Precursor) [13] is available from 30th June, 2018 - present. The data from Sentinel-5P comes with a daily frequency. The spatial resolution for this data is 7x7 km[14]. For each day multiple files are available which cover parts of India. No single file covers all parts of India because usually the across track dimension is less than the distance between west and east ends of India. Since we are working on areas included in or near India, we have to set bounds on the coordinates, i.e. within the area marked by 5° – 40°N & 65° – 100°E, respectively. We eliminate the coordinates outside the said bounds. JSON Data is then plotted using basemap or Seaborn scripts on a daily basis, which is then visualized on a map

The portal provided by ESA gives us the near real time and offline data. However, the data is not scaled. We needed to scale the data to 50Km/px because we don't require the precision it provides. Hence, we used Earth Engine, which is a Google product that allows one to write scripts to scale the data to the required configuration and download it. We took the following data sets from Google Earth Engine [15]: • Sentinel 5p Carbon Monoxide [16] • Sentinel 5p Nitrogen Dioxide [17] We created a JavaScript script to fetch the spatially scaled data for the years 2018, 2019, 2020 and averaged it. We also downloaded the Monthly data and averaged it to get insights into climate trends and seasonal data. All of these files are available in GeoTIFF format which we then convert into CSV for further analysis and also into JSON for the purpose of plotting these data points. The data conversion was done using a Python script. The Rasterio[18] module of python allows us to read the raster data from the GeoTIFF files and return a numpy array. Fig. 1 shows the visualization of geotiff data using the basemap library and Fig. 2 shows the visualization of data which is scaled and converted to CSV format. After flattening the data from a raster to csv there are a total of 1,25,664 data points for a single week. The total number of weeks from July, 2018 to December 2020 considered is 131 and for each of these weeks there are said number of data points.

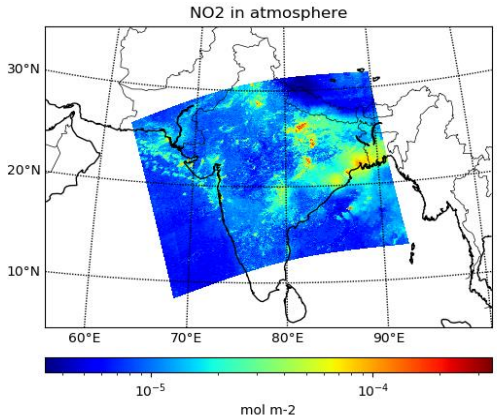

Figure 1 Visualization of GeoTIFF data from Sentinel-5p spanning multiple days. .

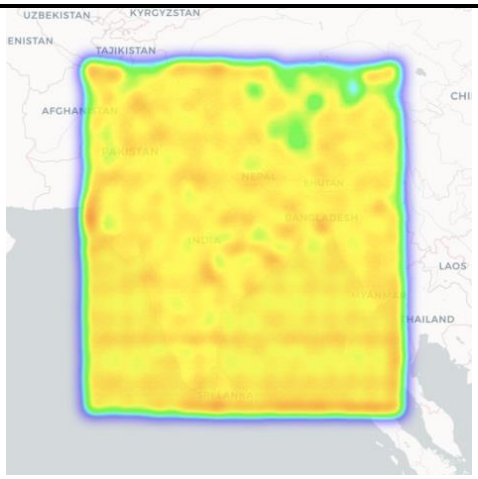

Figure 2 Visualization of CSV data from Sentinel-5p spanning data from Google Earth Engine using Folium HeatMap.

3.2 Data Preprocessing

Data Preprocessing consists of cleaning the data so that the machine learning models can find patterns and make more sense out of the data. There were a considerable amount of missing values in the data hence we leveraged the pandas library to perform data analysis & interpolation to fill those. There was an inconsistent behaviour that was exhibited by the indoor gases under consideration as they were showing much lower values than expected, because of which - we had to remove them from the dataset. We also made use of the date-time component to help us dig out new features that would be beneficial to uncover seasonal trends. The Normalization techniques that we used include MinMax Scaling and Percentile Linearization. The former is used to scale the data to a given range as shown in equation (1) whereas the latter is used when the data needs to be spaced uniformly as shown in equation (2).

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \times (R_{max} - R_{min}) + R_{min} \quad (1)$$

where R_{max} and R_{min} are the upper and lower bounds of the intended feature range.

$$x_{scaled} = \frac{x_{rank} - 1}{x_{total}} \times 100 \quad (2)$$

where x_{rank} is the rank of the given value of in the distribution and x_{total} is the total number of values.

After preprocessing, we get data that is free from missing values and outliers. This data contains the pollutant information like the pollutant type, the concentration, and the latitude & longitude information. This is then compared with the OpenAQ dataset [19] to accurately identify the location of the hotspot.

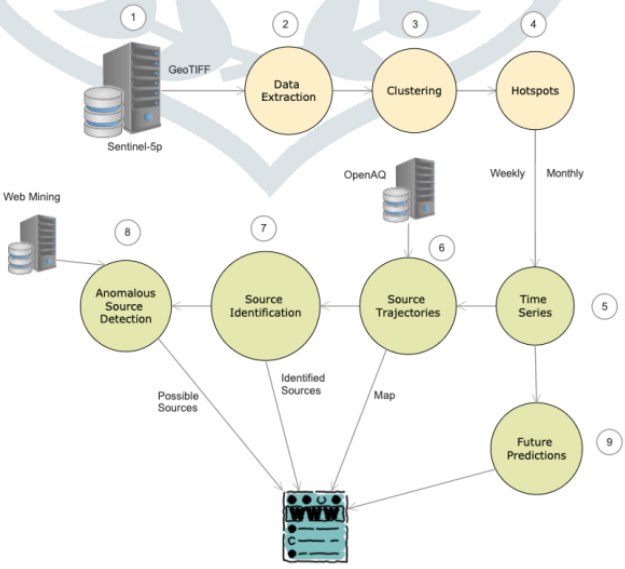

Figure 3 System Architecture

3.3 Clustering Algorithm for Pattern Discovery

DBSCAN [20] can be distinguished as a simple region-based & pixel-based image segmentation method as it is concerned with the selection of initial seed points. It usually provides favourable segmentations that correlates well with the observed edges. In the case of finer image-segmentation, we can leverage DBSCAN. After all the clusters for hotspots are

formed, we assign IDs to each cluster. Lastly, the information and labels corresponding to all the data points is output to a json file for the front-end.

K-Means Clustering [21] is a relatively simple but powerful unsupervised clustering technique where given the value of K, the data is clustered into K clusters.

In order to evaluate the clustering algorithms, Silhouette Score is used. It measures how well a given algorithm makes clusters and how close a point is to the cluster it belongs to, as compared to the foreign clusters. Its value ranges between -1 and +1. The closer the Silhouette score is to +1, the better is the performance of the given algorithm.

Figure 4 shows that the Silhouette score is the highest for K = 20, hence the value of K chosen for K-Means clustering was 20. The Silhouette score for K-Means clustering was found to be 0.3347869066552297, whereas it was -0.3700728037189192 for DBSCAN. Hence, it was concluded that K-Means Clustering performs better than DBSCAN on this data set.

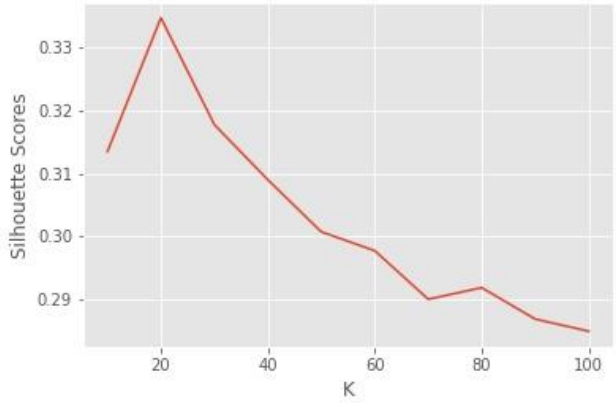

Figure 4 Silhouette Scores vs K

3.4 Time Series Analysis [22] on Weekly Data from Sentinel-5p

The weekly data was downloaded from the Sentinel project using Google Earth Engine. The period considered for this data source is from 2 July, 2018 - 31 Dec, 2020.

Figure 5 shows that the weekly clustered data is highly skewed, in terms of concentration of pollutants. This needs to be normalized in order to visualise it properly in the temporal as well as spatial dimension. In order to remove the skewness from the data, Percentile Linearization is used to normalize the values. However, one downside of this technique is that it gets rid of skewness so well that it may result in the loss of pattern information that the skew provides. But since our goal here is to just visualise the data and not perform any machine learning or deep learning algorithms on it, this technique is suitable in this case. Figure 6 shows the distribution of data after this normalization is applied and the skew in the data is completely removed.

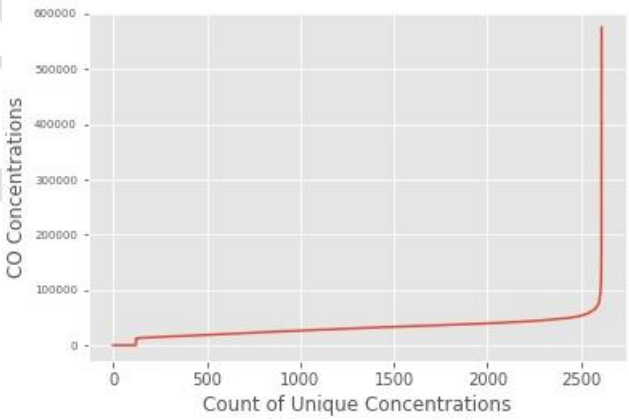

Figure 5 Skewed Distribution of CO concentrations

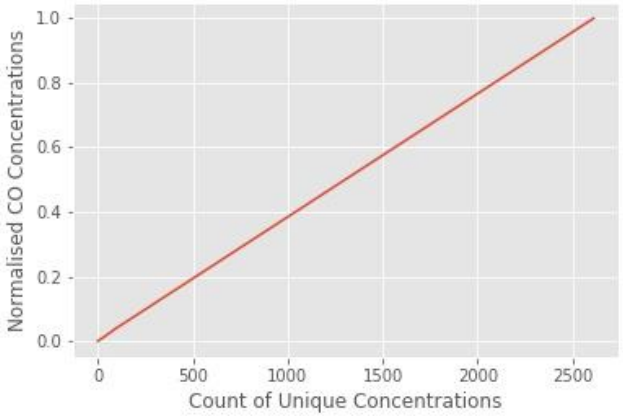

Figure 6 Normalized Distribution of CO concentrations

The normalized data can then easily be segregated into equally spaced bins based on the concentration. The Latitude, Longitude and Concentrations were then converted into GeoJson features and fed to the Timestamp GeoJson module of folium to create an interactive plot of clusters which provides a slider to adjust the temporal dimension of the visualization.
Figure 8 and Figure 9 show the clusters that are visualized on a map. The map uses the color palette given by Fig. 7 to depict different levels of pollution.

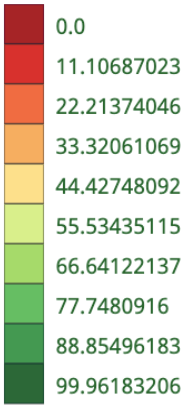

Figure 7 Color palette for clusters (µmol m-2)

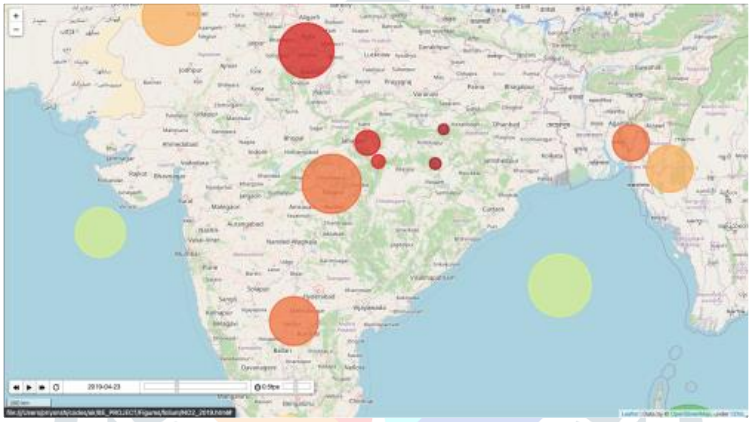

Figure 8 NO2 Clusters, April, 2019

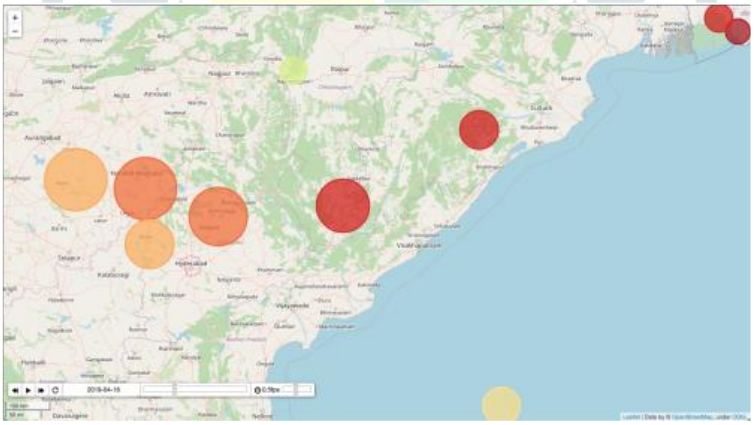

Figure 9 CO Clusters, April, 2019

Figure 10 shows the maximum concentrations of NO2 plotted against time. We observed that the maximum concentration clusters were formed with a sharp spike during winter months following a dip during the Monsoon and a gradual decline during the summer months.

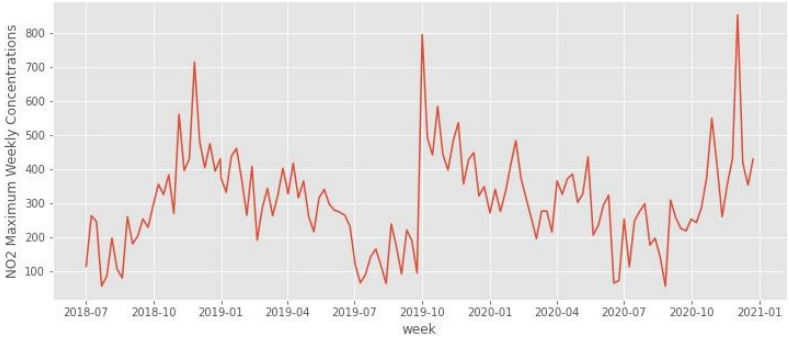

Figure 10 Maximum Weekly Concentrations of NO2 over the time period of July, 2018 - December,2020

Figure 11 shows the average concentrations of CO plotted against time. Here, as opposed to NO2, the concentration levels rise steeply during the summer months and take a dip during the monsoon and winter months.

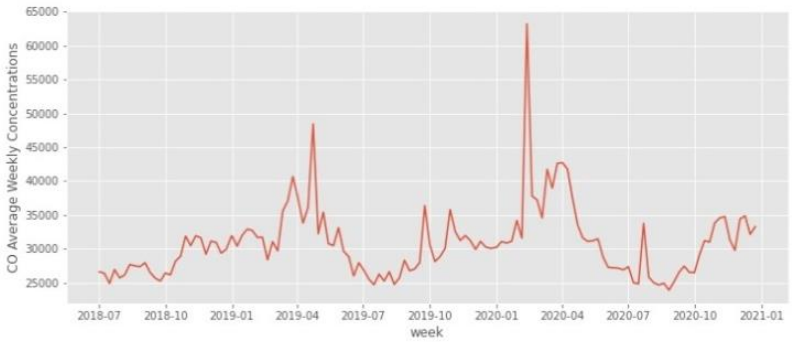

Figure 11 Average Weekly Concentrations of CO over the time period of July, 2018 - December,2020

3.5 Prediction of hotspot locations and concentrations using LSTM [24]

Recurrent Neural Networks (RNN) factor in the previous input along with the current input. However, the drawback of the RNN is that it is unable to distinguish between the recent information and the old information which can lead to unexpected results. Secondly, RNNs cannot retain the complete information derived by the previous states that go a long way back, due to the problem of vanishing gradient. In other words, it is suitable for scenarios where only the information from the past few inputs are necessary to predict the output. LSTMs solve this problem by maintaining two different states of the memory, Long term and Short term.

A cell in an RNN takes in an event Ev_t and a memory Mem_{t-1}, and the output of the cell Mem_t is calculated as given b equation (3),

$$Mem_t = \tanh(W[Mem_{t-1} \times Ev_t] + b \quad (3)$$

where, $W[]$ is the weight matrix and b is the bias and tanh is the activation function.

LSTMs take current input as well as long term and short term memory from the previous state as input and produce an output while simultaneously updating both the memories in context to the current input. A cell in an LSTM is similar to a cell in a RNN but with a lot more operations internally. It takes a Long Term Memory LTM_{t-1} , a Short Term Memory STM_{t-1} and an event Ev_t. This is accomplished by using a bunch of LSTM cells which contain the following 4 gates:

1. Forget gate: This is where the long term memory goes in order to forget everything that isn't considered important according to it. The forget factor ff_t, is calculated as given by the equation (4)

$$ff_t = \sigma(W_f[STM_{t-1}, Ev_t] + b_f \quad (4)$$

And the output of the forget gate is given by the equation (5),

$$F_{t=} LTM_{t-1} \times ff_t \quad (5)$$

2. Learn gate: The Short-term memory along with the current input are merged in the learn gate, which absorbs the information from the recent events. The ignore ig_t , is calculated as given by the equation (6),

$$ig_t = \sigma(W_i[STM_{t-1}, Ev_t]) + b_i \quad (6)$$

And the output of the Learn gate is given by (7),

$$L_t = \tanh(W_n[STM_{t-1}, Ev_t] + b) \times ig_t \quad (7)$$

3. Remember gate: The long term memory coming from the forget gate is combined with the new information learned from the Learn gate, and this is then updated in the long term memory which is calculated as given by the equation (8),

$$LTM_t = F_t \times L_t \quad (8)$$

4. Use gate: The use gate also combines the same information but instead updates the short term memory, this output also becomes the prediction output of the LSTM cell. It can be represented mathematically as given by the equations (9) and (10),

$$U_t = \tanh(W_u \times LTM_{t-1} \times ff_t + b_u) \quad (9)$$

$$V_{t=} \sigma(W_v[STM_{t-1}, Ev_t] + b_v) \quad (10)$$

The short term memory/output is given by the equation (11)

$$STM_t = U_t \times V_t \quad (11)$$

3.5.1 Algorithm

The model's algorithm is illustrated in Figure 12, which is elaborated as follows: The clusters from the clustering module are fed to the LSTM in order to generate future projections. The time step considered for the LSTM is 8, hence the LSTM will take sequential data from 8 weeks and predict the clusters` coordinates and concentration for the 9th week. The dimension of the input dataset can be understood as (number of weeks, number of clusters, number of fields) where, fields are latitude, longitude and concentration. Since the number of time steps is 8 we need to extend our data to account for it. This is done by simply sliding a window of 8 timesteps over 131 points and appending the timesteps in the window to an extended dataset using the numpy's append() function. Now ConvLSTM2D layer of the keras library is used which can essentially handle 2 dimensional timesteps in contrast to the LSTM layer which can only handle a single dimension timesteps. As a requirement of this layer the input dataset must be 5 dimensional which can be understood as *(batch_size, number of timesteps, number of channels in a single time step, number of clusters, number of fields).* The batch size is -1 here so as to generalize the model to handle any number of weeks and passing -1 in the reshape method handles the calculation of the batch size internally. Then a sequential model is initialized using *Sequential()* provided by keras and the ConvLSTM2D layer is added to the model. Mean absolute error is used as the loss function and the optimizer used is Adam optimizer. Then the model is compiled and ready to be trained. The data, which is a 5 dimensional array, is passed to the *TimeseriesGenerator* method of keras which structures the data in the form of input and output components according to the timesteps and the batch size for feeding it to the model. And then the model can be trained using the fit function of the model. Similar data transformations are required for validation as well as testing datasets, hence exactly the same series of transformations are performed on those datasets as well.

Algorithm 1 LSTM Model

Input: data set: dimension = (131, 20, 3)
Output predictions: dimensions = (batch_size, 20, 3)

Extend 131 data points to account for 8 time steps
for *i in (1, data set.shape[0]-8* **do**
    $append(extended\_dataset, dataset[i : i + 8, :, :], axis = 0)$
end for
Reshape extended_data set to (-1, 8, 1, 20, 3)
$model \leftarrow instantiateSequentialModel$
$model.add(ConvLSTM2D())$
$loss \leftarrow mean\_absolute\_error$
$optimizer \leftarrow adam$
Compile model
$data\_generator \leftarrow$
        $TimeseriesGenerator(extended\_dataset)$
model.fit(data_generator))

Figure 12 LSTM Model

3.6 Data Analysis Capabilities for studying effects of Lockdown on AQI levels

Air Pollution Dataset India from Kaggle which constituted hourly data of various gases, SO2, CO, NO2, PM2.5, Ozone, PM10, etc. and had 90531 observations. Figure 13, Figure 14, Figure 15 and Figure 16 show the visual representations of it. The concentrations of NO2 and SO2 are represented in µg/m3 .

As shown in Fig. 13, we observe that there is a coherent pattern that the pollution level of NO2 in India dips in Summer and starts rising and reaches its highest peak in Winter. During the monsoon season, it lies in the middle of its peak. As illustrated in Fig 14, NO2 level started increasing from 2017. Its median value in 2020 is comparatively lesser as compared to the previous years, thereby giving a clear indication that there might be a reduction of pollution level because of complete lockdown.

As seen in Fig 15, SO2 level falls in Summer and starts rising & then reaches their highest peak in winter. During the monsoon - it's in the middle of its peak. In Fig 16, we observe that SO2 levels started increasing from 2016 and its median value in 2020 is comparatively greater as compared to other gases in 2020, giving a clear indication that there might be an impact of SO2 on pollution level during complete lockdown.

It was observed that there is a non-linear distribution for the pollutants. There was a major chunk of observations which had missing values and hence were discarded. We plotted the trends of gases with intelligent data analysis capabilities and then rendered this into the visualization tool.

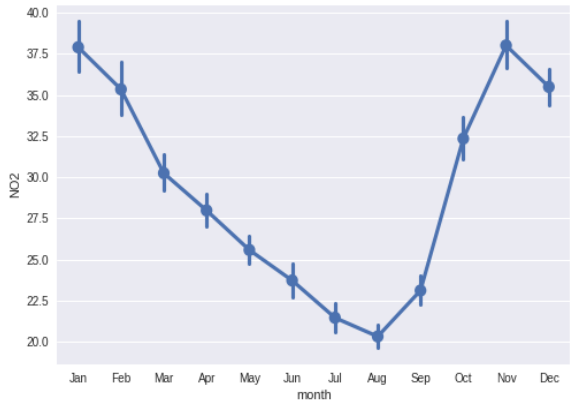

Figure 13 Monthly NO2 trend

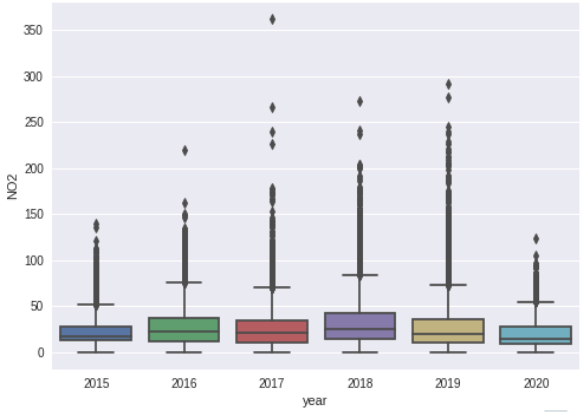

Figure 14 Yearly NO2 trend

Figure 15 Monthly SO2 trend

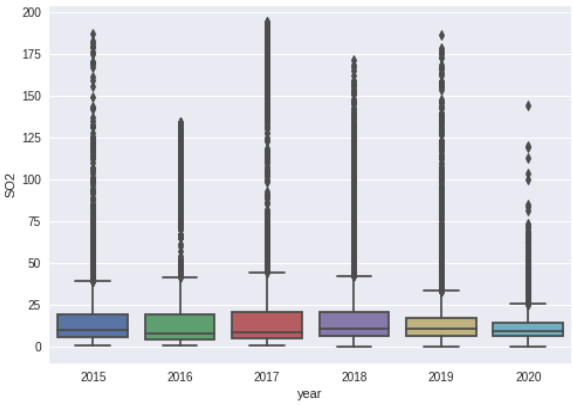

Figure 16 Yearly SO2 trend

3.7 Anomalous Hotspots and Health Science

Once the clusters for data points were formed, we integrated them with the available data in the source area of pollutants to find out the possible causes of hotspot generation. However, sometimes pollution levels spike owing to bursting fire crackers during festivals, concerts, etc. In this case we can leverage the power of crowdsourcing to point out the possible causes of pollution levels by reporting it on our platform.

We correlated our findings with health data from the WHO to study the implications of different pollutants' effects on humans of different demographics.

IV. EXPERIMENTAL RESULTS.

Kaggle Kernels were used for the processing with the following configuration:

- CPU Model: Intel(R) Xeon(R) CPU @ 2.00GHz
- Number of cores: 4
- RAM: 16GB

The model attained an accuracy of 88% on the training set and 90.8% on the validation set respectively.

The LSTM model was optimized using the Adam optimizer and was assessed on Mean Absolute Error. As depicted in Table 1 and Table 2, a total of 10 repetitions of the LSTM model were performed as these performance metrics are not exactly the same every time we train and evaluate the model. Hence, it was necessary, in order to establish that the results are reproducible. The Averages, Standard Deviations and Standard Errors were calculated for these iterations and are depicted in the respective tables.

Table 1. LSTM Loss (CO)

	Training Loss	Cross-Validation Loss	Testing Loss
1	0.1147464216	0.2841933072	0.2868733406
2	0.1104866639	0.2955319583	0.2974078953
3	0.1109537214	0.2976851463	0.3004445136
4	0.1109933034	0.2635736465	0.2664004266
5	0.1100764722	0.295383811	0.2983513474
6	0.1098883152	0.2975958884	0.2995250821
7	0.1096314415	0.2974168956	0.299515456
8	0.1124527454	0.2620214522	0.2606650293
9	0.1095826179	0.2695629895	0.2722676694
10	0.1113166288	0.2736761272	0.2765548527
Mean	0.110871	0.283664	0.285801
Standard Deviation	0.001564	0.014244	0.014701
Standard Error	0.000301	0.002741	0.002829

Table 2. LSTM Loss (NO2)

	Training Loss	Cross-Validation Loss	Testing Loss
1	0.1656883508	0.2825722694	0.2874326706
2	0.166384533	0.283087343	0.2883476615
3	0.1641630828	0.2766163051	0.2785299718
4	0.1642437577	0.2840788662	0.2874309719
5	0.1621614844	0.2700106204	0.2730641663
6	0.1648747027	0.2693461776	0.2742093205
7	0.1648140401	0.2784764171	0.283406049
8	0.1661850512	0.2643034756	0.2720349729
9	0.1685337126	0.2744047046	0.2810124159
10	0.163390696	0.2809629738	0.2828187346
Mean	0.165044	0.276386	0.280829
Standard Deviation	0.001680	0.006386	0.005840
Standard Error	0.000323	0.001229	0.001124

V. VISUALIZATION TOOL

Figure 17 shows a prototype of the user-friendly tool we have developed that demonstrates the outcome of the data analysis and visualization carried out by processing data obtained from existing ESA resources, WHO resources, and ground level data to simplify the public's understanding of satellite data and how it can be leveraged effectively. The visualization tool provides users with options such as to display temporal resolutions of a selected year and also displays pollutant levels, health impacts and other visualization insights. The air pollution hot-spot locations are plotted on a map with checker pins. The user can click on a particular hot-spot pin to view additional information about the pollutant concentrations, AQI levels, and health impacts pertaining to that particular pin's location.

Figure 17 Visualization Tool

VI. Conclusion

Our solution demonstrates intelligent data analysis capabilities using time series analysis and visualization by identifying sources of pollutants like NO2, CO, etc., their concentrations, source sector data (Industrial, Residential, Power Stations, Transport, Waste and Mobile), the reason of generation, along with latitude and longitude data - which helps us plot these hotspots and also provide related information about the pollution levels on a map. The yearly and monthly plots help us to identify climate trends and seasonal changes of indoor and outdoor pollutants. We also analyzed the effects of lockdown on pollution levels. This is then integrated with health science data to identify the health impacts of pollution on males and females of different age groups. We used freely available satellite data to find the coordinates of pollution hotspots and generated their probable sources. Our model also provides the future projections of the pollutant concentrations for efficient decision making.

VII. Acknowledgements

The authors would like to extend their gratitude to Dr. S.N Zaware, HOD Computer Department, AISSMS IOIT, Pune University for her guidance, and valuable inputs throughout the execution of all the methodologies mentioned in the paper.

.

References

[1] "Nokia corporation," https://www.nokia.com/, (Accessed on 05/28/2021).
[2] J. Y. Xu Luo, "A survey on pollution monitoring using sensor networks in environment protection," (Accessed on 05/24/2021).
[3] "India air quality index (aqi) and air pollution information | airvisual," https://www.iqair.com/india, (Accessed on 05/24/2021).
[4] G. Zhang, "Time series forecasting using a hybrid arima and neural net- work model," Neurocomputing, vol. 50, pp. 159–175, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231201007020
[5] L. Wald, L. Basly, and B. Jean-Michel, "Satellite data for the air pollution mapping," 05 1998.
[6] N. Güler Dincer and Özge Akkuş, "A new fuzzy time series model based on robust clustering for forecasting of air pollution,"
[7]Ecological Informatics, vol. 43, pp. 157–164, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574954117302030
[8] S. Ghude, S. Fadnavis, G. Beig, S. Polade, and R. A, "Detection of surface emission hot spots, trends, and seasonal cycle from satellite-retrieved no2 over india," J. Geophys. Res., vol. 113, p. D20305, 10 2008.
[9] P. Kumar, L. Morawska, C. Martani, G. Biskos, M. Neophytou, S. Di Sabatino, M. Bell,
[10] L. Norford, and R. Britter, "The rise of low-cost sensing for managing air pollution in cities," Environment International, vol. 75, pp. 199–205, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0160412014003547
[11] H. Chang, A. Pan, D. Lary, L. Waller, L. Zhang, B. Brackin, R. Finley, and F. Faruque, "Time- series analysis of satellite-derived fine particulate matter pollution and asthma morbidity in jackson, ms," Environmental Monitoring and Assessment, vol. 191, 06 2019.
[12] C. Yu, "Research of time series air quality data based on exploratory data analysis and representation," in 2016 Fifth International Conference on Agro-Geoinformatics (Agro- Geoinformatics), 2016, pp. 1–5.
[13] N. Rahman, M. H. Lee, M. T. Latif, and S. Suhartono, "Forecasting of air pollution index with artificial neural network," Jurnal Teknologi (Sciences and Engineering), vol. 63, pp. 59–64, 07 2013.
[14] "Sentinel-5p," https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-5p, (Accessed on 05/24/2021).
[15] "Openaq | kaggle," https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-5p/products- algorithms, (Accessed on 05/25/2021).
[16] "Google earth engine," https://earthengine.google.com/, (Accessed on 05/24/2021).
[17] "Sentinel-5p offl co:offline carbon monoxide," https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S5P_OFFL_L3_CO, (Accessed on 05/24/2021).
[18] "Sentinel-5p offl no2: Offline nitrogen dioxide," https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S5P_OFFL_L3_NO2, (Accessed on 05/24/2021).
[19] "Openaq | kaggle," https://rasterio.readthedocs.io/en/latest/, (Accessed on 05/25/2021).
[20] "Openaq | kaggle," https://www.kaggle.com/open-aq/openaq, (Accessed on 05/26/2021).
[21] "Dbscan - wikipedia," https://en.wikipedia.org/wiki/DBSCAN, (Accessed on 05/24/2021).
[22] "k-means clustering, wikipedia," https://en.wikipedia.org/wiki/K-means_clustering,
[23] "Time series, wikipedia," https://en.wikipedia.org/wiki/Time_series
[24] "Convlstm2d layer," https://keras.io/api/layers/recurrent_layers/conv_lstm2d/ convlstm2d-class, (Accessed on 05/24/2021).