Deep Air: Forecasting Air Pollution in Beijing, China

Vikram Reddy
vikram_reddy@berkeley.edu

Pavan Yedavalli
pavyedav@berkeley.edu

Shrestha Mohanty
shrestha_mohanty@berkeley.edu

Udit Nakhat
udit.nakhat@berkeley.edu

1 Abstract

Air pollution in urban environments has risen steadily in the last several decades. Such cities as Beijing and Delhi have experienced rises to dangerous levels for citizens. As a growing and urgent public health concern, cities and environmental agencies have been exploring methods to forecast future air pollution, hoping to enact policies and provide incentives and services to benefit their citizenry. Much research is being conducted in environmental science to generate deterministic models of air pollutant behavior; however, this is both complex, as the underlying molecular interactions in the atmosphere need to be simulated, and often inaccurate. As a result, with greater computing power in the twenty-first century, using machine learning methods for forecasting air pollution has become more popular. This paper investigates the use of the LSTM recurrent neural network (RNN) as a framework for forecasting in the future, based on time series data of pollution and meteorological information in Beijing. Due to the sequence dependencies associated with large-scale and longer time series datasets, RNNs, and in particular LSTM models, are well-suited. Our results show that the LSTM framework produces equivalent accuracy when predicting future timesteps compared to the baseline support vector regression for a single timestep. Using our LSTM framework, we can now extend the prediction from a single timestep out to 5 to 10 hours into the future. This is promising in the quest for forecasting urban air quality and leveraging that insight to enact beneficial policy.

2 Introduction

Due to rapid industrialization, the twentieth century was the most environmentally unfriendly period of time in the history of humanity. The proliferation of nonrenewable energy sources, such as coal, as well as the global rise of the gasoline-powered automobile, created dangerously high levels of greenhouse gas emissions. The United Nations states that today over 54% of the world's population lives in urban areas, and it is anticipated to increase to 66% by 2050 [8]. This increased urbanization has brought rising air pollution to the forefront of public health, as such major cities as Beijing and Delhi are presently experiencing dangerous levels of pollution. The primary air pollution metric is $PM_{2.5}$, or particulate matter that is up to 2.5 microns in diameter. These particles are small and light, which allows them to stay in the atmosphere for longer periods of time. Their diminutive size also allows them to bypass the filters of the nose and throat, and thus they can penetrate into the lungs and even the circulatory system very quickly [18]. In fact, $PM_{2.5}$ has been associated with 4 to 8% increases in cardiopulmonary diseases and lung cancer [3].

To tackle this rising public health concern, cities have explored policies and services to help their citizens, such as subsidizing the use of public transit, automobile congestion pricing, offering free protective facial masks, and subsidizing medical checkups for asthma and bronchitis. However, these benefits are contingent on knowing pollution in the future. Accordingly, forecasting air pollution has been a major research area across several different fields, from environmental science to computer science and statistics. Deterministic models in environmental science seek to understand air pollutant behavior at the molecular level, simulating diffusion and dispersion patterns based on size and type of molecule [7, 15]. In computer science and statistics, linear machine learning models have been leveraged in a more data-driven approach, specifically using multiple linear regression [9] and auto regression moving average [2]. Nonetheless, the limitation of linearity hinders the prediction accuracies, as much pollutant behavior is nonlinear. As a result, support vector regressions have been proposed [13, 12, 14]; these methods, however, have mainly predicted current time-step pollution rather than future air pollution [16]. In order to forecast future air pollution, using longer timespan data from previous years to find patterns in pollution would increase accuracy.

3 Theoretical Background

In this paper, we propose a scheme that uses a specific RNN, long short-term memory (LSTM), to analyze time series pollution data in Beijing. LSTMs take as inputs not only the current input, but also what they have "perceived" previously in time, essentially using the output at time $t-1$ as an input to time t, along with the new input at time t [5]. Given this, the network effectively has 'memory,' unlike feedforward networks. This characteristic is important because there is often information in the sequence itself, and not just the outputs [1]. Because air pollution varies temporally and since health risks are due to long-term exposures to $PM_{2.5}$, it is understood that the best predictor of future air pollution is previous air pollution over long time periods [6, 10]. We obtained time series pollution and meteorological data from 2015 to 2017 from Dr. Xiaojing Yao of the Institute of Remote Sensing Applications at the Chinese Academy of Sciences [11], as well as similar data from the Beijing International Airport from 2010 to 2014, and found LSTM RNNs to be most appropriately suited for the data.

Simple RNNs that need to find connections between the final output and data several timesteps before are limited, since there are many multiplications (an exponential number) that occur within the hidden layers of the net. This creates derivatives that will vanish, which makes it difficult for computers to compute and for networks to learn [5]. For this reason, LSTMs are a good model given the scope of the data we obtained, since they preserve the errors in a gated cell. A simple RNN would have had poor accuracy and major computational bottlenecks, since our data have a high number of samples over several thousand timesteps. The comparison between a simple RNN and an LSTM RNN is shown in Figures 1 and 2, respectively.

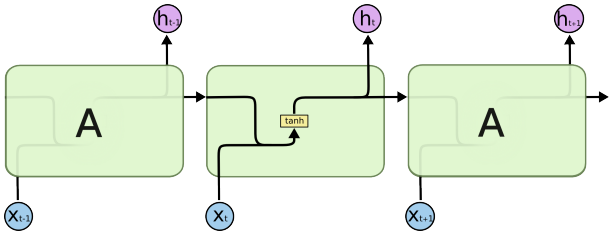

Figure 1: From *Colah blog*: A simple RNN with one layer and no gated memory cells

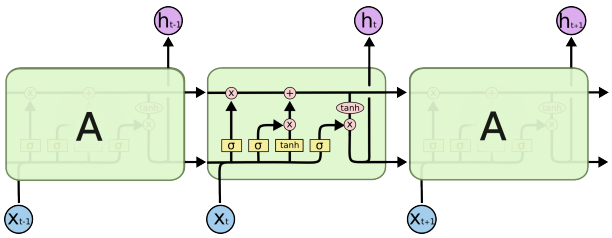

Figure 2: From *Colah blog*: An LSTM RNN with four layers, gated memory, and sigmoid activation functions

This paper will have the following structure: Section IV will describe the data, Section V the methods, Section VI the results, and Section VII will conclude the paper with high-level insights and future work.

4 Data

4.1 Data Description

The data were obtained from three sources. First, we gathered meteorological and air pollution data from 2010 to 2014 from Li et. al (2014), published as a UCI dataset. In order to expand this dataset to 2017, we received air pollution data from Dr. Xiaojing Yao, as mentioned above, from 2015 to 2017. We then got meteorological data from 2015 to 2017 from the US NOAA (National Oceanic and Atmospheric Administration). We queried their FTP server for the Global Integrated Surface Data archives. Finally, we built a parser in Python to extract weather data from their archived format.

4.2 Preprocessing and Features

We preprocessed and converted each dataset, which had hourly information, to a time series so that it can be used towards solving a supervised learning problem. The dataset we received from Dr. Yao contained pollution information from 35 specific stations. We took the average value of all the stations at each hour and joined this dataset with the meteorological dataset for the years 2015 to 2017. We also preprocessed the 2015 to 2017 weather data by first extracting the hourly timesteps from the larger set of 30-minute timesteps. We then combined the resulting dataset with the pollution and meteorological data for the years 2010 to 2014. We cleaned the missing values and unified the format of the data taken from the NOAA archives. For example, we changed the units of Wind Direction from "Angular degrees" to a categorical variable of "NE", "NW", "SE", or "SW". This matched the format of the UCI dataset.

The features of our dataset are cumulative rain hours, cumulative snow hours, wind speed, wind direction, dew point, air temperature, and air pressure, date, year, month, day, and hour. Figure 3 shows the summary statistics for these features. We later took out date, hour, day, month, and year, and instead built consecutive sequences (windows) from the time series. The target variable is pollution measured in $PM_{2.5}$. The resulting dimensions are 8 by 68,870.

	air_pressure	air_temp	cumulative_rain_hours	cumulative_snow_hours	day	dew_point	hour	month	pm2.5	wind_speed	year
count	68970.00	68970.00	68970.00	68970.00	68970.00	68970.00	68970.00	68970.00	68970.00	68970.00	68970.00
mean	4352.02	60.88	2.86	0.76	15.69	20.15	11.50	6.44	89.31	54.62	2013.44
std	4403.63	147.48	6.39	4.92	8.80	262.95	6.92	3.41	83.94	544.56	2.27
min	991.00	-160.00	0.00	0.00	1.00	-341.00	0.00	1.00	0.00	0.00	2010.00
25%	1013.00	4.00	0.00	0.00	8.00	-14.00	6.00	3.00	28.00	3.13	2011.00
50%	1026.00	20.00	0.00	0.00	16.00	5.00	12.00	6.00	68.58	10.29	2013.00
75%	10165.20	70.00	6.00	0.76	23.00	21.00	18.00	9.00	117.77	30.00	2015.00
max	10465.00	9999.00	81.00	99.00	31.00	9999.00	23.00	12.00	994.00	9999.00	2017.00

Figure 3: Summary Statistics for our feature set and target

We cleaned several values found in the summary table of Figure 3. For example, the 9999 values as max were transformed

as NaNs during preprocessing and handled accordingly. In NOAA's data dictionary, they are identified as "missing values."

4.3 Data Partitioning

We partitioned the data into four sections: train, dev, and two leave-out test sets. The training set consisted of years 2010 through 2012. The development set, in which we tuned our hyperparameters and model configurations, was the year 2013. Finally, the two leave-out sets were the year 2014 and the years 2015 to 2017. We created multiple leave-out sets in order to test how the model performs on one year versus multiple years of data.

5 Methods

5.1 Persistent Model Baseline

We used two persistent models to serve as baselines for the sequence-to-sequence forecast. A persistent model uses one or more previous timesteps to forecast the target variable at the next timestep. The conceptual motivation (and assumption) behind this baseline is that the predicted timestep would not be very different from data collected in past timesteps very close to it. That is, the data will *persist* across short periods of time. To forecast the pollution level at the next hour or future sequence of hours, we take into account the meteorological and pollution features set to a specific time lag value. For instance, if we want to understand how the past 10 hours of meteorological and pollution data affect pollution level at the next hour, we take a time lag value of 10 and the prediction hour as 1. We vary the time lag and prediction hours in our models and measure their performance with the RMSE.

5.1.1 Nonlinear Regression

The first persistent model we used is a Nonlinear Regression. Specifically, we used a Support Vector Regressor with a radial basis kernel. As the data exhibited cyclical patterns in our exploratory analysis, we believed the nonlinear method would capture some of these relationships effectively.

5.1.2 LSTM Sequence to Scalar

Next, we built an LSTM sequence-to-scalar model, in the hopes to attain a similar persistent model. We altered the number of timesteps used for the previous sequence, using 1, 3, 4, 6, 8, 10, 15, and 25 timesteps. We also experimented with various network configurations, including number of layers, types of layers, number of nodes per layer, loss functions, batch size, and number of epochs.

5.2 Forecasting Model

We then designed a model that would forecast air pollution hours or days into the future.

5.2.1 LSTM Sequence to Sequence

We used a LSTM Encoder and Decoder Architecture that learns a fixed representation of the output. Future timesteps is the quantity that is fixed. We again altered several network configurations, including number of layers, number of nodes per layer, and batch size.

According to studies by Geman, Bienenstock, and Doursat, altering the number of hidden units might reduce overfitting and increase model generalization. One possible rule of thumb is to take a hidden dimension size of two-thirds times the sum of the input and output layers [4]. The LSTM Layer may have an input dimension of 30, a hidden layer of $(2/3) * 42 = 14$, and an output layer of 12. In terms of hours, this translates to using 30 hours in the past to predict 12 hours in the future. Similarly, for a Deep LSTM with an encoder layer of size 30 and a decoder layer of size 5, we predict that a hidden layer of $(2/3) * 36 = 24$ would be most effective. The same source tells us that we should use a hidden dimension size of less than the input dimension and greater than the output dimension. We tested various other configurations as well to confirm these studies. Figure 4 shows the architecture of our Deep LSTM.

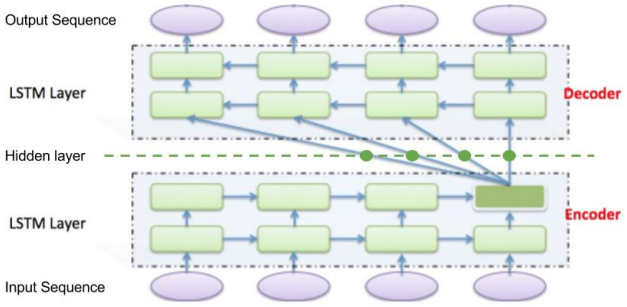

Figure 4: Deep Architecture with 2 LSTM layers and a Hidden Layer in between.

We also used early stopping techniques, such as limiting the number of epochs, in order to reduce overfitting.

6 Experiments and Results

We designed our models with various Python packages, including Scikit-Learn, Keras, and native TensorFlow. For hardware, we first benchmarked our tests on a CPU. We then ran our heavier workloads on an AWS p2.xlarge instance, which housed NVIDIA's Tesla K80 GPU. For several sequence-to-sequence LSTMs, we experienced a 4x speedup when moving from CPU to GPU.

6.1 Development Set Results

6.1.1 Tuning SVR Sequence to Scalar and LSTM sequence to Scalar

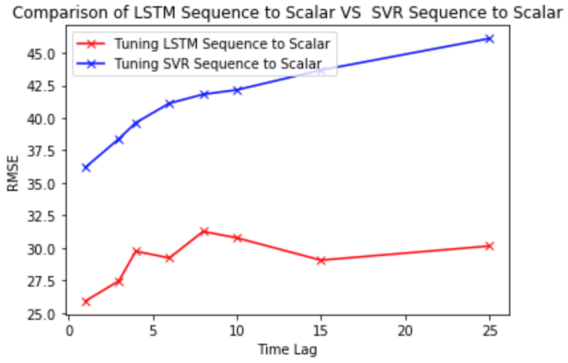

Figure 5: Comparison of RMSE for a Support Vector Regression persistent model VS LSTM model.

From Figure 5, it can be seen that increasing the time lag in the SVR model is positively correlated with RMSE. According to this model, the pollution level at any given hour is best determined by the meteorological and air pollution data of the past hour. For the LSTM model, we can see that the RMSE is minimized at the lowest time lag, similar to SVR. We also see that RMSE increases as the time lag increases. But the important trend is that with the LSTM model, the RMSE increases much more slowly over time lag relative to SVR. It also has a lower error than SVR for each of the time lags. This shows that the LSTM model is a better predictor over longer time lags than other nonlinear methods like SVR.

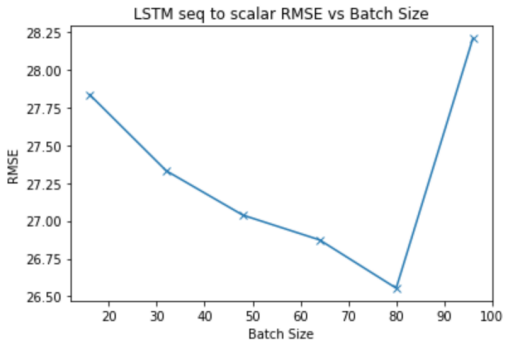

Figure 6: RMSE vs.Batch Size for a LSTM model

From Figure 6, it can be seen that as the batch size increases, the RMSE reduces until the batch size equals 80, at which point the RMSE increases. This shows that the batch size that produces the lowest RMSE is 80.

6.1.2 Tuning LSTM Sequence to Sequence

From Figure 7, we see that the LSTMs with the lowest RMSE on the Dev set have a hidden layer dimension of less than or equal to the input dimension of 30 (shown with the red line). Moreover, the model with the lowest RMSE occurs when the number of hidden dimensions is 24, which is two-thirds times the sum of the input and output dimensions, as Geman et al. advised.

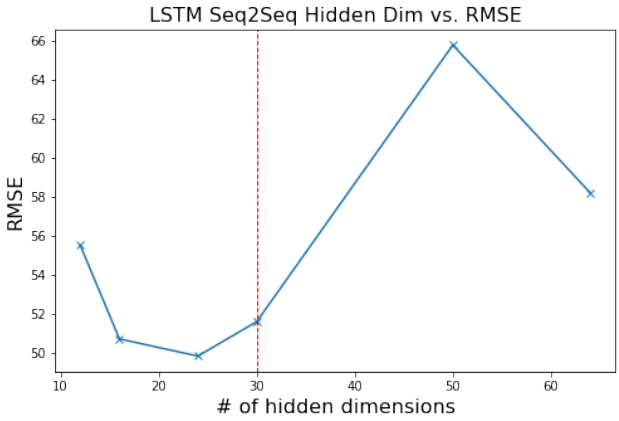

Figure 7: RMSE vs. # of hidden dimensions in a single hidden layer.

Next, we measured the runtime as a function of batch size, hypothesizing that a smaller batch size will incur greater costs at runtime. We can see from Figure 8, however, that the smallest batch size of 16 incurred a slightly slower runtime of 16 seconds versus 11 seconds, so this difference was not large.

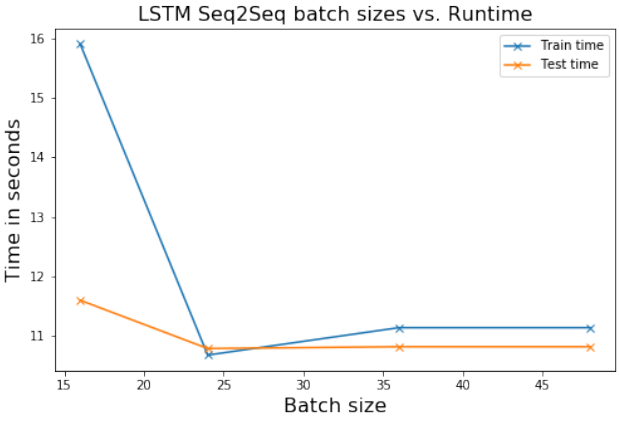

Figure 8: LSTM Seq2Seq Batch Sizes vs. Runtime.

The minimum runtime is when the batch size is 24. The following results thus used 24 for the leave-out sets.

6.2 Test Set Results

In Tables 1, 2, and 3, we use a model with the best hyperparameters as tuned on the dev set to measure RMSE and R^2 values

on the test set. The tuned model has a hidden dimension of 24, and a batch size of 24. It is a coincidence that the hidden dimension and the batch size are equal, but it would require further testing to see if these two parameters have correlated effects on model performance.

Table 1: Test RMSE for 2014 Leave-Out Set

Model	2014 RMSE	2014 R^2
SVR	44.03	0.7985
LSTM Seq2Scalar	24.9	0.92
LSTM Seq2Seq	44.15	0.689

Table 2: Test RMSE for 2015-2017 Leave-Out Set

Model	2015-2017 RMSE	2015-2017 R^2
SVR	46.01	0.749
LSTM Seq2Scalar	12.78	0.96
LSTM Seq2Seq	48.73	0.513

Table 3: Seq2Seq Test RMSE for Various Future Time Lags

Model	2014 RMSE	2014 R^2
5 future hours	44.15	0.689
10 future hours	74.8	0.588
120 future hours	108.18	-0.328

Table 4: Seq2Seq Test RMSE for Various Past Time Lags

Model	2014 RMSE
20 past, 5 future	48.2
30 past, 5 future	44.15
40 past, 5 future	45.2
50 past, 5 future	47.3
60 past, 5 future	46.2

The runtime increases as the time steps get larger. Runtime for 30 past hours to 10 future hours was 21 seconds while 5 days past to 5 days future was 161 seconds.

Figure 9 shows the SVR sequence-to-scalar, 4 hours previous to 1 future hour, run on the 2014 leave-out. From Table 1 above, the RMSE was 44.03.

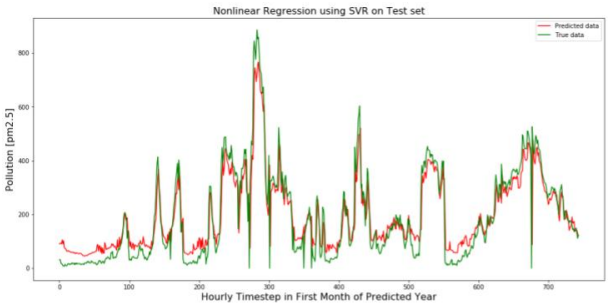

Figure 9: SVR Seq2Scalar Prediction vs. Actual.

Figure 10 shows the LSTM sequence-to-scalar, 4 hours previous to 1 future hour, run on the 2014 leave-out. From Table 1, the RMSE was 24.09.

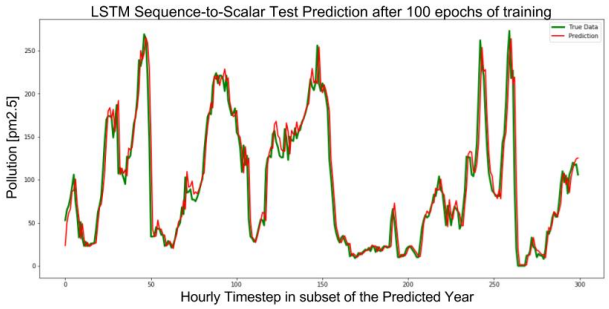

Figure 10: LSTM Seq2Scalar Prediction vs. Actual.

Figure 11 shows the LSTM sequence-to-sequence, 30 hours previous to 5 future hours. From Table 3, the RMSE was 44.15.

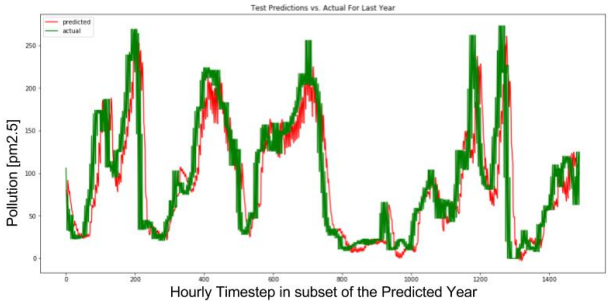

Figure 11: LSTM Seq2Seq 30 Past hours to 5 future hours.

Figure 12 shows the LSTM sequence-to-sequence, with 30 time steps previous to 10 future time steps. From Table 3, the RMSE was 74.8. As you can see, the model starts to degrade.

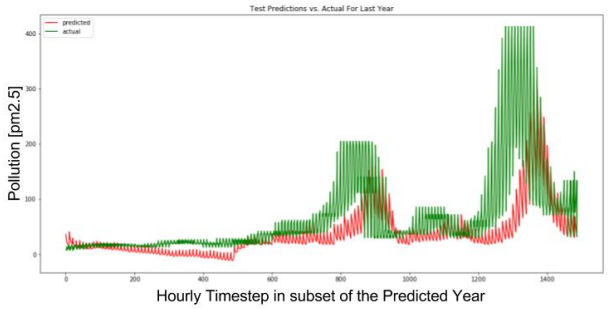

Figure 12: LSTM Seq2Seq 30 Past hours to 10 future hours.

Finally, we pushed the model beyond the within-day measures. Figure 13 shows the prediction of 5 days into the future, using 5 days of the past. From Table 3, the RMSE for this was 108.18.

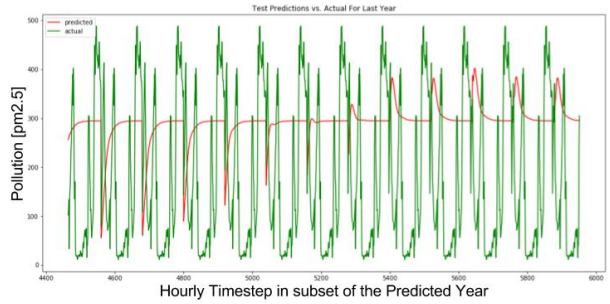

Figure 13: LSTM Seq2Seq 120 Past hours (5 days) to 120 future hours.

7 Conclusion

7.1 Insights

From this project, we see that we are able to forecast pollution for many future hours within the same accuracy as prediction for a single future hour. In particular, we notice that the error is minimized when the future timesteps range from 2 to 10 hours. From Figure 4, if we keep future hours constant, and vary past hours, the RMSE stays about the same (within 5). This means that the granularity of the previous day (short-term data) is significant enough to capture changes in air pollution. The results and the LSTM framework have predictive robustness, but only until a certain future timestep. Then, the results degrade considerably.

Predicting air pollution accurately up to ten hours in advance can be extremely useful for city policy, and more specifically, for dynamic public transit pricing. If such a city as Beijing could predict in the morning that air pollution would be dangerously high in the evening, then it could encourage public transit use by decreasing fares for that day or by dynamically enacting road congestion pricing for the evening rush hour to discourage driving. This timestep of six to ten hours for future forecasting is an encouraging stepping stone, with hopes that further optimization would increase the predicting timestep to days, which would help cities to offer longer-term services such as medical checkups and free masks. It would also help cities to prepare transit for a larger influx of travelers on high pollution days, which would require more workers and more frequency for their trains and buses.

In addition, the results of this paper can help in the management of deployed sensors, primarily in decreasing power consumption. We can now help cities save money by keeping their sensors on only in six to ten hour increments, rather than 24/7, since it was shown that short-term data is significant in prediction and can effectively create a pollution trajectory without constant monitoring. This will reduce the power consumption of these sensors, and thus save cities millions of dollars per year in electricity costs [17].

7.2 Future Work

Multiscale predictions can also be conducted to understand *how* historical predictions affect future time lags. We could combine all the data that lie within particular time lags into one input for a multiscale prediction task, such as 6-12, 12-18, and 18-24 hours, and train separate models to forecast the air pollution from these separate lag periods [11].

Because long-term prediction tasks are naturally more difficult, they require more relevant historical data, including optimum time lags, which adds another layer of optimization of the LSTM model. This means that many hyperparameters, such as batch size and number of LSTM cells, may still be optimized to return a lower RMSE for longer future forecasting. The hope is that by leveraging as much time series data as possible we can create stronger weights in the RNN, based on the sequence dependencies. As mentioned above, longer prediction times can help cities in policymaking and resource allocation.

8 Appendix

8.1 Source Code and Website

Our source code can be found at Deep Air. The project website can be found at Deep Air at Berkeley I-School.

8.2 Training Set Visualization

In Figure 14 we present a series of time series plots for certain features in our training set. We discovered cyclical patterns within our data, thus motivating our use of nonlinear models to fit the data.

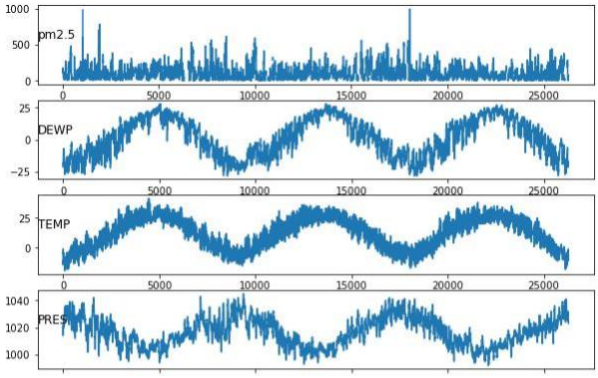

Figure 14: Cyclical patterns of Dew point, Temperature, and Pressure in the Training set. The target pollution is shown above.

8.3 PCA for feature selection

We also ran PCA to see which features explained the most variance of the data. Table 5 below shows the spread, excluding the categorical variable of wind direction. We kept rain due to its conceptual significance as a known agent to mitigate air pollution. Since the next lowest had a significant percent-explained variance (snow was 11.7%), we determined that keeping all features was important. Specifically, without rain, the rest of the features explained 93.5% of the variance.

Table 5: PCA for explained variance of our feature set

Feature	% Variance Explained
dew point	32.0%
air temp	17.4%
air press	16.7%
wind speed	15.7%
snow	11.7%
rain	6.6%

References

[1] Abdelhadi Azzouni and Guy Pujolle. "A Long Short-Term Memory Recurrent Neural Network Framework for Network Traffic Matrix Prediction". In: *CoRR* abs/1705.05690 (2017). arXiv: 1705.05690. URL: http://arxiv.org/abs/1705.05690.

[2] George Edward Pelham Box and Gwilym Jenkins. *Time Series Analysis, Forecasting and Control*. Holden-Day, Incorporated, 1990. ISBN: 0816211043.

[3] Pope III C et al. "Lung cancer, cardiopulmonary mortality, and long-term exposure to fine particulate air pollution". In: *JAMA* 287.9 (2002), pp. 1132–1141. DOI: 10.1001/jama.287.9.1132. eprint: /data/journals/jama/4822/joc11435.pdf. URL: +%20http://dx.doi.org/10.1001/jama.287.9.1132.

[4] Stuart Geman, Elie Bienenstock, and René Doursat. "Neural Networks and the Bias/Variance Dilemma". In: *Neural Comput.* 4.1 (Jan. 1992), pp. 1–58. ISSN: 0899-7667. DOI: 10.1162/neco.1992.4.1.1. URL: http://dx.doi.org/10.1162/neco.1992.4.1.1.

[5] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[6] Gerard Hoek et al. "Long-term air pollution exposure and cardio- respiratory mortality: a review". In: *Environmental Health* 12.1 (May 2013), p. 43. ISSN: 1476-069X. DOI: 10.1186/1476-069X-12-43. URL: https://doi.org/10.1186/1476-069X-12-43.

[7] Carlie J. Coats. "High Performance Algorithms In The Sparse Matrix Operator Kernel Emissions (smoke) Modeling System". In: (Mar. 1996).

[8] David A Leon. "Cities, urbanization and health". In: *International Journal of Epidemiology* 37.1 (2008), pp. 4–8. DOI: 10.1093/ije/dym271. eprint: /oup/backfile/content_public/journal/ije/37/1/10.1093/ije/dym271/2/dym271.pdf. URL: +%20http://dx.doi.org/10.1093/ije/dym271.

[9] Can Li, N. Christina Hsu, and Si-Chee Tsay. "A study on the potential applications of satellite data in air quality monitoring and forecasting". In: *Atmospheric Environment* 45.22 (2011), pp. 3663–3675. ISSN: 1352-2310. DOI: https://doi.org/10.1016/j.atmosenv.2011.04.032. URL: http://www.sciencedirect.com/science/article/pii/S1352231011004109.

[10] Xiang Li et al. "Deep learning architecture for air quality predictions". In: 23 (Oct. 2016).

[11] Xiang Li et al. "Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation". In: *Environmental Pollution* 231.Part 1 (2017), pp. 997–1004. ISSN: 0269-7491. DOI: https://doi.org/10.1016/j.envpol.2017.08.114. URL: http://www.sciencedirect.com/science/article/pii/S0269749117307534.

[12] Bing-Chun Liu et al. “Urban air quality forecasting based on multi-dimensional collaborative Support Vector Regression (SVR): A case study of Beijing-Tianjin-Shijiazhuang”. In: *PLOS ONE* 12.7 (July 2017), pp. 1–17. DOI: 10.1371/journal.pone.0179763. URL: https://doi.org/10.1371/journal.pone.0179763.

[13] Wei-Zhen Lu and Dong Wang. “Ground-level ozone prediction by support vector machine approach with a cost-sensitive classification scheme”. In: *Science of The Total Environment* 395.2 (2008), pp. 109–116. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2008.01.035. URL: http://www.sciencedirect.com/science/article/pii/S0048969708000776.

[14] Weizhen Lu et al. “Air pollutant parameter forecasting using support vector machines”. In: *Neural Networks, 2002. IJCNN ’02. Proceedings of the 2002 International Joint Conference on*. Vol. 1. 2002, pp. 630–635. DOI: 10.1109/IJCNN.2002.1005545.

[15] R. O. Olatinwo et al. “The Weather Research and Forecasting (WRF) model: application in prediction of TSWV-vectors populations”. In: *Journal of Applied Entomology* 135.1-2 (2011), pp. 81–90. ISSN: 1439-0418. DOI: 10.1111/j.1439-0418.2010.01539.x. URL: http://dx.doi.org/10.1111/j.1439-0418.2010.01539.x.

[16] Zhongang Qi et al. “Deep Air Learning: Interpolation, Prediction, and Feature Analysis of Fine-grained Air Quality”. In: *CoRR* abs/1711.00939 (2017). arXiv: 1711.00939. URL: http://arxiv.org/abs/1711.00939.

[17] Samira Shamsir et al. *Applications of Sensing Technology for Smart Cities*. Aug. 2017.

[18] K. Yang et al. “Urban air pollution study based on GIS”. In: *2009 Joint Urban Remote Sensing Event*. May 2009, pp. 1–5. DOI: 10.1109/URS.2009.5137606.