Supporting Information for:

An ensemble learning approach for estimating high spatiotemporal resolution of ground-level ozone in the contiguous United States

Weeberb J. Requia* [a, b]

Qian Di[a, c]

Rachel Silvern[d]

James T. Kelly[e]

Petros Koutrakis[a]

Loretta J. Mickley[d]

Melissa P. Sulprizio[d]

Heresh Amini[a, f]

Liuhua Shi[a, g]

Joel Schwartz[a]

* Corresponding Author: SGAN 602, Asa Norte, Brasília, DF, 70830-051, Brazil

weeberb.requia@fgv.br

a - Harvard University, Department of Environmental Health, TH Chan School of Public Health

Boston, Massachusetts, United States

b - School of Public Policy and Government, Fundação Getúlio Vargas

Brasília, Distrito Federal, Brazil

c - Research Center for Public Health, Tsinghua University, Beijing, China

d - Harvard University, John A. Paulson School of Engineering and Applied Sciences

Boston, Massachusetts, United States

e - U.S. Environmental Protection Agency, Office of Air Quality Planning & Standards

Research Triangle Park, NC, United States

f - Department of Public Health, Faculty of Health and Medical Sciences, University of Copenhagen

Copenhagen, Denmark

g - Emory University, Gangarosa Department of Environmental Health, Rollins School of Public Health

Atlanta, Georgia, United States

This file includes:

Pages S1-S36

Figure S1

Data source

References

R script used in the machine learning analyses

Tables S1 to S6

Figures S4-S8

Figure S1

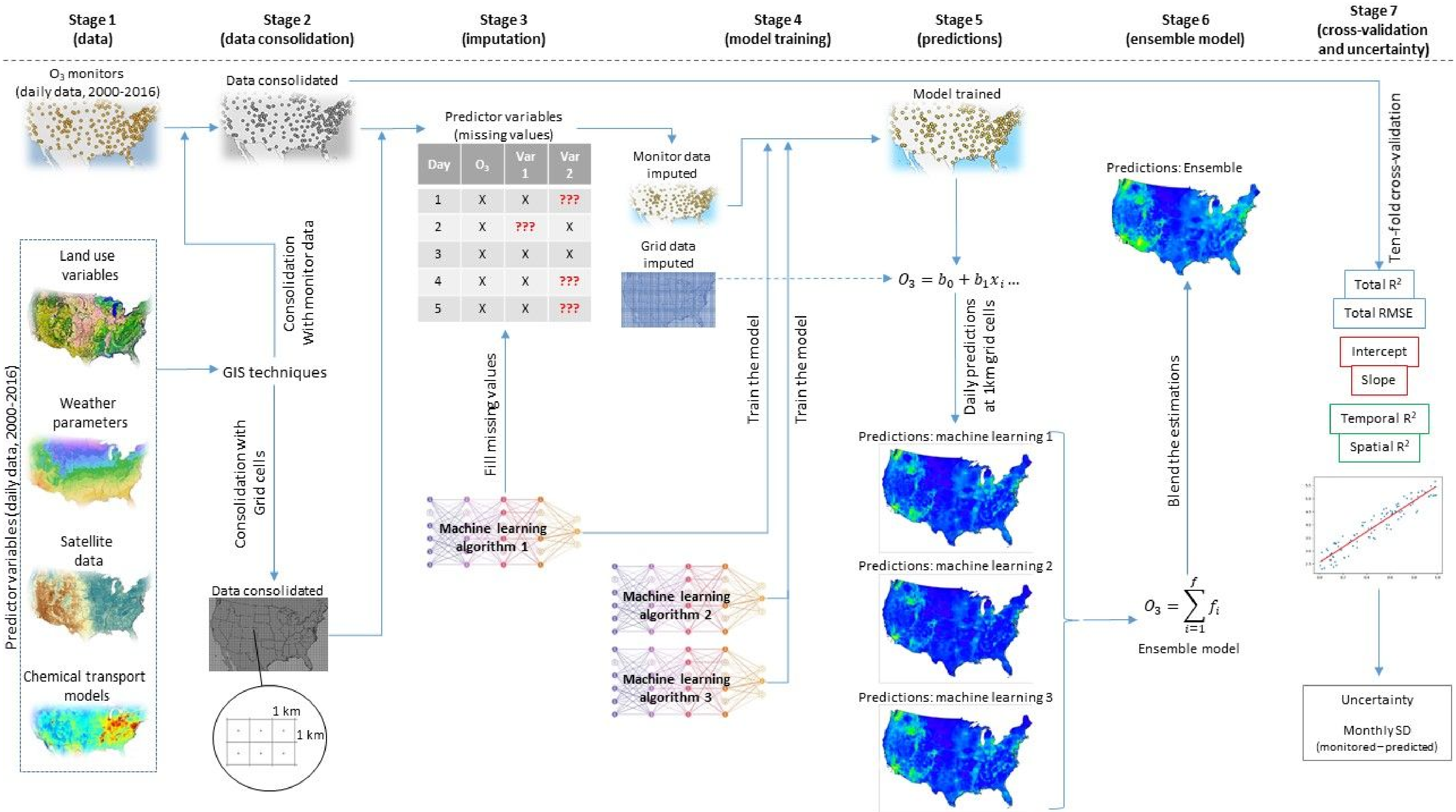

Figure S1 – Study design

SUPPLEMENTARY MATERIALS 2 (S2)

As we describe in the following sections, we accounted for multi spatial scale 100m, 1km, 10km strategy to capture the predictors. The 1km scale captures predictors on the same scale that we are predicting O_3. We include the 10 km scale for some predictors because O_3 is a regional pollutant and predictors elsewhere than the grid cell being predicted may be relevant. For example, NO_2 and VOC emissions 10km may be relevant, and land use and meteorology predictors are surrogates for such things. Finally, we used a finer scale because some predictors are related to NO emissions which quench O_3, and these vary on a fine spatial scale.

S2. Data source (first stage)

S2.1. O_3 ground measurements

We obtained daily maximum 8-hr O_3 data from the Environmental Protection Agency (EPA), including the sites from the Air Quality System (AQS) and the sites from the Clean Air Status and Trends Network (CASTNET). In addition to these EPA sites, we also collected data from other regional monitoring datasets. In total, we obtained 2,279 monitoring sites available within the study area during the study period. Note that some monitoring sites did not operate during the entire study period, especially during the winter season. The monitoring sites are not homogenous over the study area. Sites are more densely located in the eastern United States, the industrial Midwest, and the western coast. Figure S2.1 shows the location of the monitoring sites considered in our analysis.

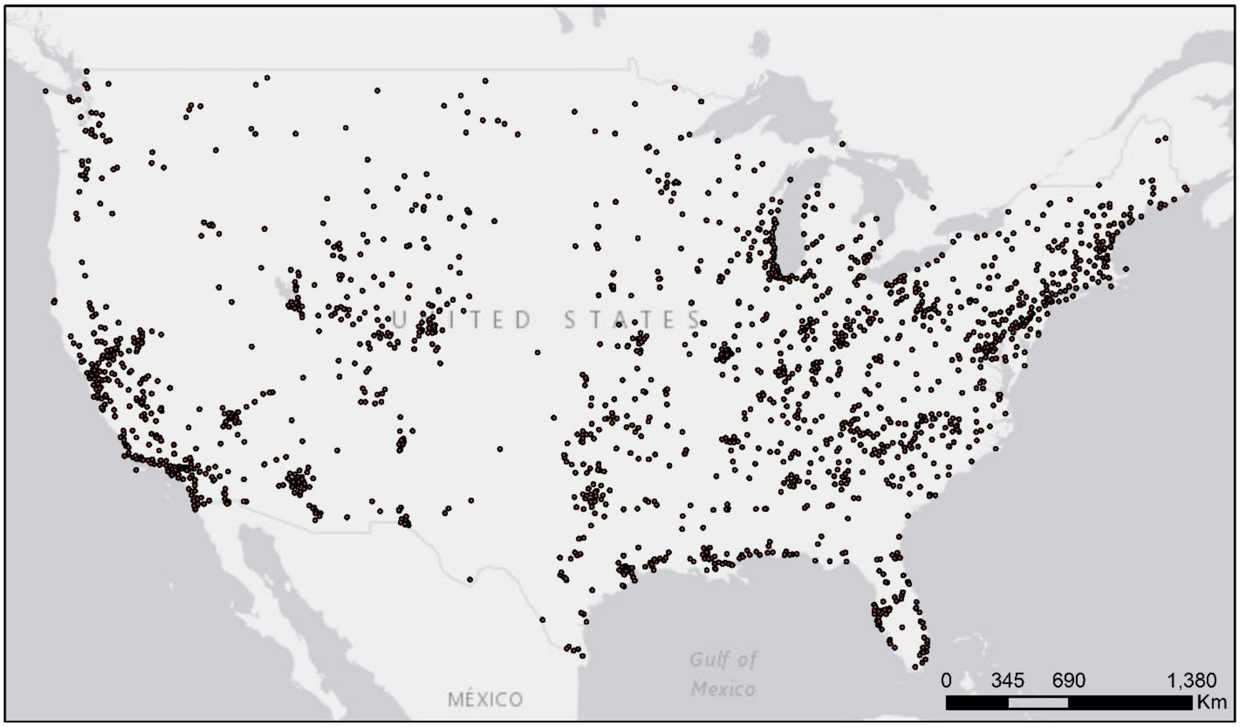

Figure S2.1 – O_3 monitoring sites in the United States

S2.2. Meteorological data

Meteorological data were provided by the National Centers for Environmental Prediction (NCEP). The NCEP data is composed by reanalysis datasets from multiple sources, including land-surface monitors, ship, radiosonde, pibal, aircraft, satellite, and other sources. This reanalysis data from NCEP data has high spatio-temporal resolution, which includes daily data with a spatial resolution of 32 km × 32 km over the U.S. The proportion of missing values is relatively low in the dataset [1]. We included 12 groups of meteorological variables, including surface air temperature, accumulated total precipitation, downward shortwave radiation flux at the surface, cloud area fractions, surface albedo, accumulated total evaporation at the surface, planetary boundary layer height, column precipitable water through the troposphere, pressure, specific humidity at 2 m, visibility, and wind speed, which was computed as the vector sum of u-wind (east-west component of the wind) at 10 m and v-wind (north-south component) at 10 m.

S2.3. Chemical transport model and remote sensing data

We used simulation results from two Chemical Transport Models (CTMs) to account for O_3 formation, dispersion, and deposition. We also considered CTM predictions for other pollutants in order to represent O_3 precursors. Previous studies have used such data to improve the performance of air pollution predictions. Chemical transport simulations represent emissions, transport, chemical reactions, and deposition of pollutants based on state-of-the-science understanding of each of these processes. These mechanistic treatments make CTMs uniquely suited to simulating future and policy scenarios under altered emission conditions in addition to informing statistical modeling studies [2,3]. We also used data based on Remote Sensing (RS) techniques that provide top-down observational constraints for the total column of O_3 and its precursors. Note that some chemical transport models also incorporate data from remote sensing to develop model inputs. We describe below the chemical transport and remote sensing data used in our analysis. First, we detail the data used for O_3, and then the data used for the O_3 precursors.

S2.3.1. CTM and RS data for O_3

a) GEOS-Chem data:

We used daily simulations of O_3 from the GEOS-Chem chemical transport model. This is a global three-dimensional model of tropospheric chemistry based on integrated weather variables from the Goddard Earth Observing System (GEOS) developed by NASA. Full details of the methodology of this model is found in [4]. We performed a nested grid simulation at 0.500° × 0.667°

for North America using boundary conditions from a global model simulation. GEOS-Chem simulates O_3 concentrations at different layers through the troposphere. Therefore, to calibrate the tropospheric column of O_3 to ground-level O_3, we calculated scaling factors as the percentage of surface-level O_3 in the total tropospheric column. This approach is similar to that used in modeling $PM_{2.5}$, where aerosol optical depth (AOD) is a column measurement of aerosol and researchers used the vertical profile from a chemical transport model to calibrate AOD to ground-based $PM_{2.5}$ [5,6].

The retrieval algorithm of satellite-based O_3 is affected by certain atmospheric factors, including aerosol abundance, surface reflectance, surface albedo, and cloud contamination [7]. To correct possible errors in O_3 retrieval, we included in our model variables related to aerosol concentration/aerosol types, cloud coverage, and surface albedo/surface reflectance. We obtained GEOS-Chem variables related to aerosol concentration and aerosol types, which include simulated elemental carbon, organic carbon, sulfate, nitrate, and aerosol mass. The remaining variables used to correct errors in O_3 retrieval were obtained from other CTM and RS sources, as described in the next sections. Note that cloud coverage and surface albedo were obtained from the NCEP/NCAR reanalysis dataset, as described above.

b) GEMS data:

We obtained GEMS (Geostationary Environment Monitoring Spectrometer, a satellite-based instrument) O_3 data from European Centre for Medium-Range Weather Forecasts (ECMWF). This data is from Copernicus Atmosphere Monitoring Service (CAMS) products that derives GEMS total column O_3 at 0.125-degree resolution.

c) OMI data:

We also used tropospheric O_3 columns from the Ozone Monitoring Instrument (OMI), an instrument on board the Earth Observing System (EOS)-Aura satellite. The OMI O_3 data product is available every day at 13 km × 48 km grid cells. To relate OMI O_3 column retrievals to surface-level O_3, we used the CTM scaling factors described above.

We also obtained OMI data related to absorbing aerosol index in the ultraviolet and visible ranges (OMAERUVd and OMAEROe). These data are added to the aerosol output from GEOS-Chem to correct possible aerosol-related errors in satellite-based O_3 retrieval.

d) CMAQ data:

Daily simulations of O_3 were also accessed from the Community Multiscale Air Quality Modeling System (CMAQ). This CTM is a numerical air quality model developed by EPA that simulates the emissions, chemistry and physics of the atmosphere on a 12 km grid. As with GEOS-Chem, we obtained model output from CMAQ related to aerosol concentration and aerosol type, including simulated elemental carbon, organic carbon, sulfate, and nitrate. Note that this was the same set of aerosol data obtained from GEOS-Chem, as described above.

e) MERRA-2 data:

We used total column O_3 estimates from Modern-Era Retrospective Analysis for Research and Applications, Version 2 (MERRA-2). To supplement the aerosol data from GEOS-Chem, OMI, and CMAQ, we also accessed MERRA-2 surface concentrations of sulphate aerosol, hydrophilic black carbon, hydrophobic black carbon, hydrophilic organic carbon, and hydrophobic organic carbon.

f) MODIS data:

We used surface reflectance estimates from MODIS - MOD09A1 [8], which provide estimates of the surface spectral reflectance of TERRA MODIS Bands 1-7 corrected for atmospheric conditions such as gases, aerosols, and Rayleigh scattering. As we described previously, surface reflectance is used to correct possible errors in the O_3 retrieval.

S2.3.2. CTM and RS data for O_3 precursors

As mentioned in the introduction section, O_3 formation is based on mechanisms involving photochemical reactions of O_3 precursors, including NO_x, VOCs, and CO. These precursors are incorporated into the chemical transport simulations through emission inventories. However, the temporal resolution of emissions is limited. To address this limitation, we first used AQS data from U.S. EPA ground monitors (same source used for the O_3 data at monitors) to represent daily measurements of SO_2, NO_2, NOx, and VOCs. Then, we used some chemical and remote sensing data to characterize ozone precursors. These data were accessed from the same sources as described above. We used NO_2 concentration from GEOS-Chem model, NO_2 column measurement from the OMI satellite instrument (with spatial resolution of 13 km × 24 km), NO_2 simulations from CMAQ, and NO_2 column concentration simulations from CAMS (Copernicus Atmosphere Monitoring Service), another reanalysis data set.

S2.4. Other predictor variables

The chemical simulation models and data from remote sensing may not capture the very fine spatio-temporal resolution of the atmospheric mechanisms related to O_3 formation or removal. Therefore, we considered a set of land use, temporal terms, and some extra variables to represent proxies of the O_3 formation or removal. Previous studies have shown that these proxy variables improve the ability to capture the local variation of O_3 concentration [2,3,9–11]. The description of these set of variables used in our model is provided below.

S2.4.1. Land use terms

We accessed land use variables at 30 m resolution from the National Land Cover Database – NLCD [12]. This database includes water bodies, developed areas, urban areas, barren land, forest, shrub land, herbaceous land, planted/cultivated land, and wetlands. We calculated the proportion of each land-use type in grid cells with 100 m, 1 km, and 10 km horizontal resolution. As a complement for vegetation areas, we also accounted for Normalized Difference Vegetation Index (NDVI). We accessed NDVI data from the MODIS data product MOD13A2 at 1 km × 1 km level (https://cmr.earthdata.nasa.gov/search/concepts/C194001238-LPDAAC_ECS.html). Finally, as an additional proxy variable for local air pollution emissions, we included restaurant density in the model. We obtained the location of restaurants from the U.S. historical business data [13], and then we calculated weighted restaurant density in each 1 km × 1 km grid cell. The weight was based on the amount of emissions, approximated by the number of seats.

S2.4.2. Elevation

We accounted for different metrics of elevation, including minimum elevation, maximum elevation, mean elevation, median elevation, standard deviation of elevation, and break line emphasis. We aggregated the data from its original 7.5-arc-second spatial resolution to three different spatial resolution – 100 m × 100 m, 1 km × 1 km, and 10 km × 10 km. These three spatial resolutions were included in the model as separate predictor variables. This data was provided by the Global Multi-Resolution Terrain Elevation Dataset [14].

S2.4.3. Transportation

Traffic emissions are important sources of O_3 precursors [15,16]. We considered two variables as traffic emission proxies – road density and traffic count. Road density was obtained from the US Census Bureau. The data accessed includes shapefiles representing all roads in the USA. We calculated the spatial density (total length of road in each grid cell) for each 100 m × 100 m, 1 km × 1 km, and 10 km × 10 km grid cell (as for elevation, these three different spatial scales were included in the model as separate predictors). Annual average traffic count data for the contiguous U.S. was provided by ArcGIS Online. We interpolated the original data to 100 m, 1 km, and 10 km spatial resolution.

S2.4.4. Temporal terms

To improve the detection of the temporal variation of O_3, we accounted for 26 temporal predictor variables. We included in the model 16 dummy variables representing the years (2000-2016), 6 dummy variables for the weekdays, 1 variable representing the Julian days, and 2 variables for the season trends. The variables representing the seasonal patterns were estimated based on sine and cosine functions [17], in which: *sine season = sin(2 x π x doy / 365.24)*; and, *cosine season = cos(2 x π x doy / 365.24)*; where *doy* is the day of year (e.g., 1:365).

S2.4.5. Spatio-temporal lag of O_3 measurements

We assumed that O_3 concentration from nearby monitoring sites are more correlated than from faraway sites, and O_3 concentration from neighboring days are more correlated than long ago. These assumptions are based on the spatial and temporal autocorrelation of O_3 distributions. Therefore, we included spatially and temporally lagged O_3 measurements in the model. We estimated the spatially lagged terms as inverse distance weighted O_3 measurements at other locations, as well as their one-day, three-day and five-day lagged moving average values.

S2.4.6. Temporal lag of several O_3 predictors

We also accounted for temporal lag (1-day lagged moving values) of meteorological variables, including air temperature, total precipitation accumulation, pressure, humidity, and wind speed.

REFERENCES

(1) Collins, W.; Deaven, D.; Gandin, L.; Iredell, M.; Jenne, R.; Joseph, D. The NCEP NCAR 40-Year Reanalysis Project. *Bull. Am. Meteorol. Soc.* **1996**, *77* (3), 437–472.
(2) Di, Q.; Rowland, S.; Koutrakis, P.; Schwartz, J. A Hybrid Model for Spatially and Temporally Resolved Ozone Exposures in the Continental United States. *J. Air Waste Manag. Assoc.* **2017**, *67* (1), 39–52. https://doi.org/10.1080/10962247.2016.1200159.
(3) Di, Q.; Amini, H.; Shi, L.; Kloog, I.; Silvern, R.; Kelly, J.; Sabath, M. B.; Choirat, C.; Koutrakis, P.; Lyapustin, A.; Wang, Y.; Mickley, L. J.; Schwartz, J. An Ensemble-Based Model of PM2.5 Concentration across the Contiguous United States with High Spatiotemporal Resolution. *Environ. Int.* **2019**, *130* (July), 104909. https://doi.org/10.1016/j.envint.2019.104909.
(4) Bey, I.; Jacob, D. J.; Yantosca, R. M.; Logan, J. A.; Field, B. D.; Fiore, A. M.; Li, Q.; Liu, H.; Mickley, L. J.; Schultz, M. G. Global Modeling of Tropospheric Chemistry with Assimilated Meteorology: Model Description and Evaluation. *J. Geophys. Res.* **2001**, *106* (D19), 23073–23095. https://doi.org/10.1029/2001jd000807.
(5) Liu, Y.; Park, R. J.; Jacob, D. J.; Li, Q.; Kilaru, V.; Sarnat, J. A. Mapping Annual Mean Ground-Level PM2.5 Concentrations Using Multiangle Imaging Spectroradiometer Aerosol Optical Thickness over the Contiguous United States. *J. Geophys. Res. D Atmos.* **2004**, *109* (22), 1–10. https://doi.org/10.1029/2004JD005025.

(6) van Donkelaar, A.; Martin, R. V; Brauer, M.; Kahn, R.; Levy, R.; Verduzco, C.; Villeneuve, P. J. Global Estimates of Ambient Fine Particulate Matter Concentrations from Satellite-Based Aerosol Optical Depth: Development and Application. *Environ. Health Perspect.* **2010**, *118* (6), 847–855. https://doi.org/10.1289/ehp.0901623.
(7) Lin, J. T.; Martin, R. V.; Boersma, K. F.; Sneep, M.; Stammes, P.; Spurr, R.; Wang, P.; Van Roozendael, M.; Clémer, K.; Irie, H. Retrieving Tropospheric Nitrogen Dioxide from the Ozone Monitoring Instrument: Effects of Aerosols, Surface Reflectance Anisotropy, and Vertical Profile of Nitrogen Dioxide. *Atmos. Chem. Phys.* **2014**, *14* (3), 1441–1461. https://doi.org/10.5194/acp-14-1441-2014.
(8) NASA. MOD09A1 MODIS/Terra Surface Reflectance 8-Day L3 Global 500m SIN Grid V006. U.S. Government, LP DAAC 2018. https://doi.org/https://doi.org/10.5067/MODIS/MOD09A1.006.
(9) Di, Q.; Koutrakis, P.; Schwartz, J. A Hybrid Prediction Model for PM2.5 Mass and Components Using a Chemical Transport Model and Land Use Regression. *Atmos. Environ.* **2016**, *131*, 390–399. https://doi.org/10.1016/j.atmosenv.2016.02.002.
(10) Ramos, Y.; Requia, W. J.; St-Onge, B.; Blanchet, J.-P.; Kestens, Y.; Smargiassi, A. Spatial Modeling of Daily Concentrations of Ground-Level Ozone in Montreal, Canada: A Comparison of Geostatistical Approaches. *Environ. Res.* **2018**, *166*, 487–496. https://doi.org/10.1016/j.envres.2018.06.036.
(11) Watson, G. L.; Telesca, D.; Reid, C. E.; Pfister, G. G.; Jerrett, M. Machine Learning Models Accurately Predict Ozone Exposure during Wildfire Events. *Environ. Pollut.* **2019**, *254*, 112792. https://doi.org/10.1016/j.envpol.2019.06.088.
(12) Collin H. Homer, Joyce A. Fry, C. A. B. *The National Land Cover Database*; 2012.
(13) Caracuzzo, A. H. B. S. Infogroup US Historical Business Data 1997-2016. In Harvard Dataverse: 2016. 2016.
(14) Danielson, J. J.; Gesch, D. B. *Global Multi-Resolution Terrain Elevation Data 2010 (GMTED2010)*; 2011. https://doi.org/10.3133/ofr20111073.
(15) Uherek, E.; Halenka, T.; Borken-Kleefeld, J.; Balkanski, Y.; Berntsen, T.; Borrego, C.; Gauss, M.; Hoor, P.; Juda-Rezler, K.; Lelieveld, J.; Melas, D.; Rypdal, K.; Schmid, S. Transport Impacts on Atmosphere and Climate: Land Transport. *Atmos. Environ.* **2010**, *44* (37), 4772–4816. https://doi.org/10.1016/j.atmosenv.2009.04.044.
(16) Alhajeri, N. S.; McDonald-Buller, E. C.; Allen, D. T. Comparisons of Air Quality Impacts of Fleet Electrification and Increased Use of Biofuels. *Environ. Res. Lett.* **2011**, *6* (2), 024011. https://doi.org/10.1088/1748-9326/6/2/024011.
(17) Stolwijk, A. M.; Straatman, H.; Zielhuis, G. A. Studying Seasonality by Using Sine and Cosine Functions in Regression Analysis. *J. Epidemiol. Community Health* **1999**, *53* (4), 235–238. https://doi.org/10.1136/jech.53.4.235.

SUPPLEMENTARY MATERIALS 3 (S3)

R script used in the machine learning analyses

```
########### Ozone model###############
########################################
### Model training - Neural Network ###
########################################
library(h2o)
library(mgcv)
library(parallel)

####################################################################
### Step 1: Set working directory and prepare the dataset ########
####################################################################
##### Set working directory:
setwd("/media/gate/Weeberb/Ozone_model")

##### Open dataset:
InputData <- readRDS("//media/InputData_O3_model.rds")

####################################################################
### Step 2: Grid search - Choose the best model ##################
####################################################################
library(e1071)

##### Define the parameters:
X_Var = c(2:87,94:124,127:length(InputData2))
data1 = InputData2[!is.na(InputData2[,1]),]
set.seed(123)
train_ind <- sample(seq_len(nrow(data1)), size = round(nrow(data1)*0.9))
train_data= data1[train_ind,]

##### Starting H2O:
h2o.init(min_mem_size = "120g",nthreads = 5)

##### Open the data using H2O:
train_h2o<-as.h2o(train_data)
test_data= data1[-train_ind,]
test_h2o<-as.h2o(test_data)

##### Define the parameters
hyper_params=list(epochs=c(40,50,75),hidden=list(c(210,210),c(350,350),c(250,250)),l1=c(1e-
4,1e-5),activation = c("Rectifier"))
```

```
##### Run the model and save the results
modgrid<-h2o.grid("deeplearning",x=X_Var,y=1,epsilon = 1e-08,training_frame =
train_h2o,hyper_params = hyper_params)
saveRDS(modgrid,"grid_search_Neural_Network.rds")

##### Get a list of the highest R2 value:
model_ids <- modgrid@model_ids
models <- lapply(model_ids, function(id) { h2o.getModel(id)})

MaxR2 = 0
MaxID = 0
for(i in 1:18)
{
  TempR = models[[i]]@model$training_metrics@metrics$r2
  if(TempR>MaxR2)
  {
    MaxR2 = TempR
    MaxID = i
  }
  cat(sprintf("%d %f\n",i,TempR))
}



######################################################################
### Step 3: Run the model - Neural Network #######################
######################################################################
##### Start H2O:
h2o.init(min_mem_size = "120g",nthreads = 5)
#h2o.init(nthreads=-1,max_mem_size = "400G",port = (54321))    #### To start H2O on
Odyssey

##### Open the data using H2O:
InputData_h2o <- as.h2o(InputData)

### Set "X" variables:
#X_Var_1 = c(2:87,94:124,127:128,130:137,139:length(InputData2))
X_Var_1 = c(2:112, 114:121, 123:length(InputData))

names(InputData)

## Run the model and save the results:
mod_nn_1 <- h2o.deeplearning(x = X_Var_1, # column numbers for predictors
                             y = 1, # column number for label
                             training_frame = InputData_h2o,nfolds=10,standardize = FALSE,
```

```
                    fold_assignment="Modulo",seed=271828,keep_cross_validation_predictions
= TRUE,
                    activation="Rectifier",hidden=c(250,250),epochs=50,
                    epsilon = 1e-08,l1=1e-05,distribution="AUTO")

h2o.saveModel(mod_nn_1,force = TRUE,
"2_Outcome/1_ModelTraining_Neural_Network/2_Model")

summary(mod_nn_1)



########### Ozone model################
##########################################
### Model training - Random Forest ###
##########################################
library(h2o)
library(mgcv)
library(parallel)

######################################################################
### Step 1: Set working directory and prepare the dataset ########
######################################################################
##### Set working directory:
setwd("/media/gate/Weeberb/Ozone_model")

##### Open dataset:
InputData <- readRDS("//media/InputData_O3_model.rds")

######################################################################
### Step 2: Grid search - Choose the best model ##################
######################################################################
##### Define the parameters:
X_Var = c(2:87,94:124,127:length(InputData2))
data1 = InputData2[!is.na(InputData2[,1]),]
set.seed(123)
train_ind <- sample(seq_len(nrow(data1)), size = round(nrow(data1)*0.9))
train_data= data1[train_ind,]

##### Starting H2O:
h2o.init(min_mem_size = "120g",nthreads = 5)

##### Open the data using H2O:
train_h2o<-as.h2o(train_data)
test_data= data1[-train_ind,]
test_h2o<-as.h2o(test_data)
```

```
##### Define the parameters
grid_space <- list()
grid_space$ntrees <- c(800,1000,1200)
grid_space$max_depth <- c(7,9,11)
grid_space$nbins <- c(15,20,24)
grid_space$nbins_cats <- c(400,449,500)
grid_space$sample_rate <- c(0.4,0.5)#0.415836

##### Run the model and save the results
modgrid_rf <- h2o.grid("randomForest", grid_id="drf_grid_cars_test", x=X_Var, y=1,
                       training_frame=train_h2o,validation_frame = test_h2o,hyper_params=grid_space)

saveRDS(modgrid_rf,"grid_search_Random_Forest.rds")

##### Get a list of the highest R2 value:
model_ids <- modgrid_rf@model_ids
models <- lapply(model_ids, function(id) { h2o.getModel(id)})

MaxR2 = 0
MaxID = 0
for(i in 1:162)
{
  TempR = models[[i]]@model$training_metrics@metrics$r2
  if(TempR>MaxR2)
  {
    MaxR2 = TempR
    MaxID = i
  }
  cat(sprintf("%d %f ntrees:%d max_depth:%d nbins:%d nbins_cats:%d
sample_rate:%f\n",i,TempR,models[[i]]@parameters$ntrees,models[[i]]@parameters$max_dept
h,models[[i]]@parameters$nbins,models[[i]]@parameters$nbins_cats,models[[i]]@parameters$s
ample_rate))
}


######################################################################
### Step 3: Run the model - Random Forest ########################
######################################################################
##### Start H2O:
h2o.init(min_mem_size = "120g",nthreads = 5)
#h2o.init(nthreads=-1,max_mem_size = "400G",port = (54321))    #### To start H2O on
Odyssey

##### Open the data using H2O:
InputData_h2o <- as.h2o(InputData)
```

```
### Set "X" variables:
#X_Var_1 = c(2:87,94:124,127:128,130:137,139:length(InputData2))
X_Var_1 = c(2:112, 114:121, 123:length(InputData))
names(InputData)

## Run the model and save the results:
mod_rf_1 <-h2o.randomForest(x = X_Var_1,
                    y = 1,
                    training_frame = InputData_h2o,nfolds=10,
                    fold_assignment="Modulo",seed=271828,keep_cross_validation_predictions =
TRUE,
                    ntrees=800,max_depth = 9,nbins = 15,nbins_cats = 449,sample_rate = 0.5)

h2o.saveModel(mod_rf_1,force = TRUE,
"2_Outcome/2_ModelTraining_Random_Forest/2_Model")

summary(mod_rf_1)


########### Ozone model##################
############################################
### Model training - Gradient Boosting ###
############################################
library(h2o)
library(mgcv)
library(parallel)


####################################################################
### Step 1: Set working directory and prepare the dataset ########
####################################################################
##### Set working directory:
setwd("/media/gate/Weeberb/Ozone_model")

##### Open dataset:
InputData <- readRDS("//media/InputData_O3_model.rds")

####################################################################
### Step 2: Grid search - Choose the best model ##################
####################################################################
##### Define the parameters:
X_Var = c(2:87,94:124,127:length(InputData2))
data1 = InputData2[!is.na(InputData2[,1]),]
set.seed(123)
train_ind <- sample(seq_len(nrow(data1)), size = round(nrow(data1)*0.9))
train_data= data1[train_ind,]
```

```
##### Starting H2O:
h2o.init(min_mem_size = "120g",nthreads = 5)

##### Open the data using H2O:
train_h2o<-as.h2o(train_data)
test_data= data1[-train_ind,]
test_h2o<-as.h2o(test_data)

##### Define the parameters
xgb_params1 <- list(learn_rate = c(0.01,0.005,0.007),
                    max_depth = c(6,7,8),
                    sample_rate = c(1.0),
                    col_sample_rate = c(0.4,0.5,0.6),
                    ntrees = c(175,200,250))

##### Run the model and save the results
modgrid_xgb <- h2o.grid("gbm", x = X_Var, y = 1,
                        grid_id = "xgb_params1",
                        training_frame = train_h2o,validation_frame = test_h2o,
                        seed = 1,
                        hyper_params = xgb_params1)

saveRDS(modgrid_xgb,"grid_search_Gradient_boosting.rds")

##### Get a list of the highest R2 value:
model_ids <- modgrid_xgb@model_ids
models <- lapply(model_ids, function(id) { h2o.getModel(id)})
MaxR2 = 0
MaxID = 0
for(i in 1:81)
{
  TempR = models[[i]]@model$training_metrics@metrics$r2
  if(TempR>MaxR2)
  {
    MaxR2 = TempR
    MaxID = i
  }
  cat(sprintf("%d %f\n",i,TempR))
}
```

```
######################################################################
### Step 3: Run the model - Gradient Boosting#####################
######################################################################

##### Start H2O:
h2o.init(min_mem_size = "120g",nthreads = 5)
#h2o.init(nthreads=-1,max_mem_size = "400G",port = (54321))    #### To start H2O on
Odyssey

##### Open the data using H2O:
InputData_h2o <- as.h2o(InputData2)

### Set "X" variables:
#X_Var_1 = c(2:87,94:124,127:128,130:137,139:length(InputData2))
X_Var_1 = c(2:112, 114:121, 123:length(InputData))
names(InputData)

## Run the model and save the results:
mod_gbm_1=h2o.gbm(x = X_Var_1,
            y = 1,
            training_frame = InputData_h2o,nfolds=10,
            fold_assignment="Modulo", seed=271828,keep_cross_validation_predictions =
TRUE,
            ntrees=200,learn_rate = 0.007,max_depth = 7,sample_rate = 1,col_sample_rate = 0.5)

h2o.saveModel(mod_gbm_1,force = TRUE,
"2_Outcome/3_ModelTraining_Gradient_Boosting/2_Model")

summary(mod_gbm_1)
```

```
############################################
### Cross-validation     #####################
############################################


library(h2o)
library(caret)
library(mgcv)
library(parallel)


for(YEAR in 2015:2016)
{
########### for training ########################
#DirPath = "D:\\Google Drive\\Research\\USTemperature\\"
 DirPath = "/nfs/bigdata_nobackup/a/airpred_d_scratch/"
GCS = "North_America_Equidistant_Conic"
Sep = "/"
VariableID = 99939
NAME = "O3"
EPLISON = 1/1000# change for ozone, NO2.
STARTDATE = as.Date(paste0(YEAR,"-01-01"))
ENDDATE = as.Date(paste0(YEAR,"-12-31"))
N_Core = 61
OPTION = "training"#"prediction"#"training"
SiteName_Train = "AQRVO3"
SiteName_Predict = "AQRVO3"

################ code
source("ModelFunctions.R")

## path
DirPath_Assembled = paste0(DirPath,"assembled_data",Sep)
DirPath_Processed = paste0(DirPath,"processed_data",Sep)
DirPath_Model =
paste0(DirPath,"assembled_data",Sep,"training",Sep,NAME,"_",VariableID,"_CV",Sep)
dir.create(DirPath_Model)
## read location
SiteData_Train<-
ReadLocation(paste0(DirPath_Processed,SiteName_Train,Sep,"Location",Sep,SiteName_Train,"
Site_",GCS))
N_Site_Train <- nrow(SiteData_Train)
SiteData_Predict<-
ReadLocation(paste0(DirPath_Processed,SiteName_Predict,Sep,"Location",Sep,SiteName_Predi
ct,"Site_",GCS))
N_Site_Predict <- nrow(SiteData_Predict)
```

```
## time
N_Day = as.numeric(ENDDATE - STARTDATE + 1)

## read weight
Weight1 =
h5read_robust(paste0(DirPath_Processed,SiteName_Predict,Sep,"Temp",Sep,"SpatialLaggedWe
ightPeak41_",SiteName_Train,"_",SiteName_Predict,".h5"),name = "Weight")
Weight2 =
h5read_robust(paste0(DirPath_Processed,SiteName_Predict,Sep,"Temp",Sep,"SpatialLaggedWe
ightPeak42_",SiteName_Train,"_",SiteName_Predict,".h5"),name = "Weight")
Weight3 =
h5read_robust(paste0(DirPath_Processed,SiteName_Predict,Sep,"Temp",Sep,"SpatialLaggedWe
ightPeak43_",SiteName_Train,"_",SiteName_Predict,".h5"),name = "Weight")

## start h2o vm
h2o.init(min_mem_size = "300g")



## read imputed data
if(file.exists(paste0(DirPath_Model,"InputDataImputed.rds")))
{
  InputData = readRDS(paste0(DirPath_Model,"InputDataImputed.rds"))
  InputData_h2o <- as.h2o(InputData)
}else
{
  ## if not, read non-imputed data
  if(file.exists(paste0(DirPath_Model,"InputData.rds")))
  {
    InputData = readRDS(paste0(DirPath_Model,"InputData.rds"))
  }else
  {
    ## if not again, read data from scratch
    # ##read csv configuration file
    col = c(rep("character",6))
    col[c(2,3)] = "logical"
    col[c(7,8)] = "numeric"
    VariableList =
read.csv(paste0(DirPath,"assembled_data",Sep,"VariableList_",VariableID,".csv"),colClasses =
col)
    VariableList = VariableList[!is.na(VariableList$READ),]
    InputData =
ReadData(DirPath,Sep,VariableID,NAME,SiteName_Train,STARTDATE,ENDDATE,OPTION
,SiteData_Train,VariableList)
    saveRDS(InputData,paste0(DirPath_Model,"InputData_Original.rds"))
```

```
    InputData <- StandardData(DirPath,InputData,VariableID)
    InputData$MonitorData = log(InputData$MonitorData+EPLISON)
    saveRDS(InputData,paste0(DirPath_Model,"InputData.rds"))

  }
  ## imputation
  InputData_h2o <- as.h2o(InputData)
  InputData_h2o =
ImputeData(DirPath,Sep,InputData_h2o,InputData,OPTION,VariableID,SiteData_Train)
  InputData = as.data.frame(InputData_h2o)
  saveRDS(InputData,paste0(DirPath_Model,"InputDataImputed.rds"))
}

## for CV
set.seed(123)
flds <- createFolds(seq(1:nrow(SiteData_Predict)), k = 10, list = TRUE, returnTrain = FALSE)
saveRDS(flds,paste0(DirPath_Model,"CV.rds"))

sink(file=paste0(DirPath_Model,paste0("testH2o_train_output_CV_"),STARTDATE,"_",ENDD
ATE,".txt"),append=T,split=F)

#######################
for(m in 1:10)
{
  if(file.exists(paste0(DirPath_Model,"OutputData","_round",m,".rds")))
  {
    next
  }

  Index_train = which(is.element(data$SiteCode,SiteData[-flds[[m]],"SITECODE"]))
  Index_test = which(is.element(data$SiteCode,SiteData[flds[[m]],"SITECODE"]))

  # variables used in step 1
  X_Var_1 = which(!names(InputData_h2o) %in% c("MonitorData","SiteCode",
"CalendarDay","PM25_Region","NO2_Region","Ozone_Region","Temporal_Lagged_1","Temp
oral_Lagged_2","Temporal_Lagged_3","Spatial_Lagged_1","Spatial_Lagged_2","Spatial_Lagg
ed_3"))

  #step 1: neural network
  TempDir = paste0(paste0(DirPath_Model,"NeuralNetwork_Step1","_round",m,Sep))
  if(length(list.files(TempDir))>0)
  {
    if(length(list.files(TempDir))>1)
    {
      stop("more than one model here exist!",TempDir)
    }
```

```
    Model <- list.files(TempDir)
    mod_nn_1 <- h2o.loadModel(paste0(TempDir,Model[1]))
  }else
  {
    mod_nn_1 <- h2o.deeplearning(x = X_Var_1, # column numbers for predictors
                                  y = 1, # column number for label
                                  training_frame = InputData_h2o[Index_train,],nfolds=10,standardize =
FALSE,

fold_assignment="Modulo",seed=271828,keep_cross_validation_predictions = TRUE,
                                  activation="Rectifier",hidden=c(210,210),epochs=50,
                                  epsilon = 1e-08,l1=1e-05,distribution="AUTO")
    h2o.saveModel(mod_nn_1,force = TRUE,TempDir)
  }
  #step 1: random forest
  TempDir = paste0(paste0(DirPath_Model,"RandomForest_Step1","_round",m,Sep))
  if(length(list.files(TempDir))>0)
  {
    if(length(list.files(TempDir))>1)
    {
      stop("more than one model here exist!",TempDir)
    }
    Model <- list.files(TempDir)
    mod_rf_1 <- h2o.loadModel(paste0(TempDir,Model[1]))
  }else
  {
    mod_rf_1=h2o.randomForest(x = X_Var_1,
                               y = 1,
                               training_frame = InputData_h2o[Index_train,],nfolds=10,
                               fold_assignment="Modulo",seed=271828,keep_cross_validation_predictions
= TRUE,
                               ntrees=1000,max_depth = 9,nbins = 20,nbins_cats = 449,sample_rate =
0.41536)
    h2o.saveModel(mod_rf_1,force = TRUE,TempDir)
  }
  ## Step 1: gradient boosting
  TempDir = paste0(paste0(DirPath_Model,"GradientBoosting_Step1","_round",m,Sep))
  if(length(list.files(TempDir))>0)
  {
    if(length(list.files(TempDir))>1)
    {
      stop("more than one model here exist!",TempDir)
    }
    Model <- list.files(TempDir)
    mod_gbm_1 <- h2o.loadModel(paste0(TempDir,Model[1]))
  }else
```

```
  {
    mod_gbm_1=h2o.gbm(x = X_Var_1,
                 y = 1,
                 training_frame = InputData_h2o[Index_train,],nfolds=10,
                 fold_assignment="Modulo", seed=271828,keep_cross_validation_predictions =
TRUE,
                 ntrees=200,learn_rate = 0.007,max_depth = 7,sample_rate = 1,col_sample_rate =
0.5)
    h2o.saveModel(mod_gbm_1,force = TRUE,TempDir)
  }

  ## ensemble
  InputData$pred_nn_1<-as.vector(h2o.predict(mod_nn_1,newdata=InputData_h2o)$predict)
  InputData$pred_gbm_1<-as.vector(h2o.predict(mod_gbm_1,newdata=InputData_h2o)$predict)
  InputData$pred_rf_1<-as.vector(h2o.predict(mod_rf_1,newdata=InputData_h2o)$predict)

  if(file.exists(paste0(DirPath_Model,"Ensemble_Step1","_round",m,".rds")))
  {
    mod_ensemble_1 <- readRDS(paste0(DirPath_Model,"Ensemble_Step1","_round",m,".rds"))
  }else
  {
    cl <- makeCluster(N_Core)
    mod_ensemble_1<-bam(MonitorData ~ s(Other_Lat, Other_Lon,
by=pred_nn_1)+s(Other_Lat, Other_Lon, by=pred_gbm_1)+s(Other_Lat, Other_Lon,
by=pred_rf_1),data=InputData[Index_train,],cluster=cl)
    stopCluster(cl)
    saveRDS(mod_ensemble_1,paste0(DirPath_Model,"Ensemble_Step1","_round",m,".rds"))
  }

  ## show results
  Pred<-predict(mod_ensemble_1,newdata = InputData)
  InputData$pred_ensemble_1 = Pred
  A = summary(lm(MonitorData~pred_ensemble_1,data=InputData[Index_test,]))
  print(sprintf("Step1: Ensemble:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))
  A = summary(lm(MonitorData~pred_rf_1,data=InputData[Index_test,]))
  print(sprintf("Step1: Random Forest:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))
  A = summary(lm(MonitorData~pred_gbm_1,data=InputData[Index_test,]))
  print(sprintf("Step1: Gradient Boosting:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))
  A = summary(lm(MonitorData~pred_nn_1,data=InputData[Index_test,]))
  print(sprintf("Step1: Neural Network:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))

  ##########################################
  #### two step modeling...
  dim(Pred)<-c(N_Day,N_Site_Train)
  Pred_1 = apply(Pred,2,function(x) as.numeric(filter(x, rep(1/7,7),sides = 2,circular = TRUE)))
```

```
  Pred_2 = apply(Pred,2,function(x) as.numeric(filter(x,
c(1/16,2/16,3/16,4/16,3/16,2/16,1/16),sides = 2,circular = TRUE)))
  Pred_3 = apply(Pred,2,function(x) as.numeric(filter(x,
c(1/44,4/44,9/44,16/44,9/44,4/44,1/44),sides = 2,circular = TRUE)))
  Pred_4 = Pred%*%Weight1
  Pred_5 = Pred%*%Weight2
  Pred_6 = Pred%*%Weight3
  dim(Pred_1)<-c(N_Day*N_Site_Train)
  dim(Pred_2)<-c(N_Day*N_Site_Train)
  dim(Pred_3)<-c(N_Day*N_Site_Train)
  dim(Pred_4)<-c(N_Day*N_Site_Train)
  dim(Pred_5)<-c(N_Day*N_Site_Train)
  dim(Pred_6)<-c(N_Day*N_Site_Train)

  Temp_h2o = as.h2o(as.data.frame(cbind(Pred_1,Pred_2,Pred_3,Pred_4,Pred_5,Pred_6)))
  colnames(Temp_h2o)<-
c("Temporal_Lagged_1","Temporal_Lagged_2","Temporal_Lagged_3","Spatial_Lagged_1","Sp
atial_Lagged_2","Spatial_Lagged_3")
  InputData_h2o = as.h2o(InputData)
  InputData_h2o = h2o.cbind(InputData_h2o,Temp_h2o)



  ##########################################################
  ### STEP 2
  X_Var_2 = which(!names(InputData_h2o) %in% c("MonitorData","SiteCode",
"CalendarDay","PM25_Region","NO2_Region","Ozone_Region"))

  ## neural network, best choice
  TempDir = paste0(paste0(DirPath_Model,"NeuralNetwork_Step2","_round",m,Sep))
  if(length(list.files(TempDir))>0)
  {
   if(length(list.files(TempDir))>1)
   {
    stop("more than one model here exist!",TempDir)
   }
   Model <- list.files(TempDir)
   mod_nn_2 <- h2o.loadModel(paste0(TempDir,Model[1]))
  }else
  {
   mod_nn_2 <- h2o.deeplearning(x = X_Var_2, # column numbers for predictors
                              y = 1, # column number for label
                              training_frame = InputData_h2o[Index_train,],nfolds=10,standardize =
FALSE,

fold_assignment="Modulo",seed=271828,keep_cross_validation_predictions = TRUE,
```

```
                        activation="Rectifier",hidden=c(210,210),epochs=50,
                        epsilon = 1e-08,l1=1e-05,distribution="AUTO")
  h2o.saveModel(mod_nn_2,force = TRUE,TempDir)
 }

 ## random forest --- the best model
 TempDir = paste0(paste0(DirPath_Model,"RandomForest_Step2","_round",m,Sep))
 if(length(list.files(TempDir))>0)
 {
  if(length(list.files(TempDir))>1)
  {
   stop("more than one model here exist!",TempDir)
  }
  Model <- list.files(TempDir)
  mod_rf_2 <- h2o.loadModel(paste0(TempDir,Model[1]))
 }else
 {
  mod_rf_2 =h2o.randomForest(x = X_Var_2,
                       y = 1,
                       training_frame = InputData_h2o[Index_train,],nfolds=10,
                       fold_assignment="Modulo",seed=271828,keep_cross_validation_predictions
= TRUE,
                       ntrees=1000,max_depth = 9,nbins = 20,nbins_cats = 449,sample_rate =
0.41536)
  h2o.saveModel(mod_rf_2,force = TRUE,TempDir)
 }

 ## Step 2 gradient boosting
 TempDir = paste0(paste0(DirPath_Model,"GradientBoosting_Step2","_round",m,Sep))
 if(length(list.files(TempDir))>0)
 {
  if(length(list.files(TempDir))>1)
  {
   stop("more than one model here exist!",TempDir)
  }
  Model <- list.files(TempDir)
  mod_gbm_2 <- h2o.loadModel(paste0(TempDir,Model[1]))
 }else
 {
  mod_gbm_2 =h2o.gbm(x = X_Var_2,
                 y = 1,
                 training_frame = InputData_h2o[Index_train,],nfolds=10,
                 fold_assignment="Modulo", seed=271828,keep_cross_validation_predictions =
TRUE,
                 ntrees=200,learn_rate = 0.007,max_depth = 7,sample_rate = 1,col_sample_rate =
0.5)
```

```
    h2o.saveModel(mod_gbm_2,force = TRUE,TempDir)
  }

  ## ensemble
  InputData$pred_nn_2<-as.vector(h2o.predict(mod_nn_2,newdata=InputData_h2o)$predict)
  InputData$pred_gbm_2<-as.vector(h2o.predict(mod_gbm_2,newdata=InputData_h2o)$predict)
  InputData$pred_rf_2<-as.vector(h2o.predict(mod_rf_2,newdata=InputData_h2o)$predict)

  if(file.exists(paste0(DirPath_Model,"Ensemble_Step2","_round",m,".rds")))
  {
    mod_ensemble_2 <- readRDS(paste0(DirPath_Model,"Ensemble_Step2","_round",m,".rds"))
  }else
  {
    cl <- makeCluster(N_Core)
    mod_ensemble_2<-bam(MonitorData ~ s(Other_Lat, Other_Lon,
by=pred_nn_2)+s(Other_Lat, Other_Lon, by=pred_gbm_2)+s(Other_Lat, Other_Lon,
by=pred_rf_2),data=InputData[Index_train,],cluster=cl)
    stopCluster(cl)
    saveRDS(mod_ensemble_2,paste0(DirPath_Model,"Ensemble_Step2","_round",m,".rds"))
  }

  ## show results
  Pred<-predict(mod_ensemble_2,newdata = InputData)
  InputData$pred_ensemble_2 = Pred
  A = summary(lm(MonitorData~pred_ensemble_2,data=InputData[Index_test,]))
  print(sprintf("Step2: Ensemble:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))
  A = summary(lm(MonitorData~pred_rf_2,data=InputData[Index_test,]))
  print(sprintf("Step2: Random Forest:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))
  A = summary(lm(MonitorData~pred_gbm_2,data=InputData[Index_test,]))
  print(sprintf("Step2: Gradient Boosting:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))
  A = summary(lm(MonitorData~pred_nn_2,data=InputData[Index_test,]))
  print(sprintf("Step2: Neural Network:%f,%f\n",A$r.squared,sqrt(mean(A$residuals^2))))

  ## save input data and output data
  OutputData =
InputData[,c("CalendarDay","pred_nn_1","pred_gbm_1","pred_rf_1","pred_ensemble_1","pred_
nn_2","pred_gbm_2","pred_rf_2","pred_ensemble_2")]
  saveRDS(OutputData,paste0(DirPath_Model,"OutputData","_round",m,".rds"))
}
saveRDS(OutputData,paste0(DirPath_Model,"OutputData.rds"))
sink()
}
```

Table S1

Table S1 – List of predictor variables

Meteorological variables
Accumulated total precipitation
Downward Shortwave Radiation on Flux
Downward Shortwave Radiation on Flux
Accumulated total Evaporation
High Cloud Area Fraction
Planetary Boundary Layer Height
Low Cloud Area Fraction
Medium Cloud Area Fraction
Precipitable Water for entire atmosphere
Visibility
Air temperature (surface)
Pressure (surface)
Specific Humidity at 2m
U-wind at 10 m
V-wind at 10 m
Precipitation rate
Latent Heat Flux
Sensible Heat Flux
Snow Cover
Soil Moisture Content
Forecast of Total Cloud Cover
Upward Longwave Radiation on Flux
Omega: A term used to describe vertical motion in the atmosphere
Accumulated Snow
Cloud coverage
Surface albedo
GEOS-Chem data
Surface-level O3, simulated by GEOS-Chem
Surface-level NO2, simulated by GEOS-Chem
Elemental carbon - GEOS-Chem variable related to aerosol concentration and aerosol type
Organic carbon - GEOS-Chem variable related to aerosol concentration and aerosol type
Sulfate - GEOS-Chem variable related to aerosol concentration and aerosol type
Nitrate - GEOS-Chem variable related to aerosol concentration and aerosol type
Aerosol mass - GEOS-Chem variable related to aerosol concentration and aerosol type
GEMS data
GEMS total column O3 at 0.125-degree resolution
OMI satellite data
OMAERUVd_UVA - absorbing aerosol index in the ultraviolet range
OMAEROe_UVA - absorbing aerosol index in the ultraviolet range
OMAEROe_VISA - absorbing aerosol index in the visible range
Satellite-measured column O3 concentration
Satellite-measured column SO2 concentration
Satellite-measured column NO2 concentration
Satellite-measured UV index

CMAQ data
Surface-level NO2, simulated by CMAQ
Percentage of surface-level NO2 at the total column NO2, simulated by CMAQ
Surface-level ozone, simulated by CMAQ
Percentage of surface-level ozone at the total column ozone, simulated by CMAQ
Surface-level PM2.5 nitrate, simulated by CMAQ
Surface-level PM2.5 sulfate, simulated by CMAQ
Surface-level PM2.5 elemental carbon, simulated by CMAQ
Surface-level PM2.5 organic carbon, simulated by CMAQ
MERRA
Hydrophilic Black Carbon
Hydrophobic Black Carbon
Hydrophilic Organic Carbon
Hydrophobic Organic Carbon
Sulphate aerosol
Total column O3
MODIS Satellite data
Surface temperature during the day, mean of nearby grid cells
Surface temperature at night, mean of nearby grid cells
MODIS-measured cloud coverage during the day, mean of nearby grid cells
MODIS-measured cloud coverage at night, mean of nearby grid cells
Surface temperature during the day, data from the nearest grid cell
Surface temperature at night, data from the nearest grid cell
MODIS-measured cloud coverage during the day, data from the nearest grid cell
MODIS-measured cloud coverage at night, data from the nearest grid cell
MAIAC aerosol (aerosol optical depth) data at 470 nm wavelength, from Aqua satellite, retrieved from the nearest grid cell
MAIAC aerosol (aerosol optical depth) data at 550 nm wavelength, from Aqua satellite, retrieved from the nearest grid cell
MAIAC aerosol (aerosol optical depth) data at 470 nm wavelength, from Terra satellite, retrieved from the nearest grid cell
MAIAC aerosol (aerosol optical depth) data at 550 nm wavelength, from Terra satellite, retrieved from the nearest grid cell
Viewing angle of the sensor at the Terra satellite
Viewing angle of the sensor at the Aqua satellite
NDVI value from MODIS MOD13A2, 1 km spatial resolution and 16-day temporal resolution
CAMS data
NO2 column concentration simulations from CAMS
Additional air quality data from EPA
Daily measurements of SO2
Daily measurements of NO2
Daily measurements of NOx
Daily measurements of VOCs
NLCD landuse database
Wetland coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests
Water coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests

Planted coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests
Herbaceous coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests
Shrubland coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests
Barren coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests
Developed area coverage from NLCD data set, the original 30 meter data were aggregated to 10000 meter raster, then we interpolated the 10000-meter raster to locations of interests
Wetland coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Water coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Planted coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Herbaceous coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Shrubland coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Barren coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Developed area coverage from NLCD data set, the original 30 meter data were aggregated to 100 meter raster, then we interpolated the 100-meter raster to locations of interests
Road density obtained from the US Census Bureau
Primary road density; we converted primary road (lines shapefile format) into raster with 100 meter pixel width; then we interpolated them to locations of interests
Primary road density; we converted primary road (lines shapefile format) into raster with 1000 meter pixel width; then we interpolated them to locations of interests
Primary road density; we converted primary road (lines shapefile format) into raster with 10000 meter pixel width; then we interpolated them to locations of interests
Primary and secondary road density; we converted primary and secondary road (lines shapefile format) into raster with 100 meter pixel width; then we interpolated them to locations of interests
Primary and secondary road density; we converted primary and secondary road (lines shapefile format) into raster with 1000 meter pixel width; then we interpolated them to locations of interests
Primary and secondary road density; we converted primary and secondary road (lines shapefile format) into raster with 10000 meter pixel width; then we interpolated them to locations of interests
All road (primary, secondary and terciary road) density; we converted all road (lines shapefile format) into raster with 100 meter pixel width; then we interpolated them to locations of interests
All road (primary, secondary and terciary road) density; we converted all road (lines shapefile format) into raster with 1000 meter pixel width; then we interpolated them to locations of interests
All road (primary, secondary and terciary road) density; we converted all road (lines shapefile format) into raster with 10000 meter pixel width; then we interpolated them to locations of interests
Global Multi-Resolution Terrain Elevation Dataset
Maximal elevation, original data was at 7.5 arc-seconds, then aggregated to 100 meter resolution
Minimal elevation, original data was at 7.5 arc-seconds, then aggregated to 100 meter resolution
Median elevation, original data was at 7.5 arc-seconds, then aggregated to 100 meter resolution
Mean elevation, original data was at 7.5 arc-seconds, then aggregated to 100 meter resolution
Systematic subsample, original data was at 7.5 arcseconds, then aggregated to 100 meter resolution
Breakline emphasis, original data was at 7.5 arc-seconds, then aggregated to 100 meter resolution
Standard deviation, original data was at 7.5 arc-seconds, then aggregated to 100 meter resolution
Maximal elevation, original data was at 7.5 arc-seconds, then aggregated to 1000 meter resolution

Minimal elevation, original data was at 7.5 arc-seconds, then aggregated to 1000 meter resolution
Median elevation, original data was at 7.5 arc-seconds, then aggregated to 1000 meter resolution
Mean elevation, original data was at 7.5 arc-seconds, then aggregated to 1000 meter resolution
Systematic subsample, original data was at 7.5 arcseconds, then aggregated to 1000 meter resolution
Breakline emphasis, original data was at 7.5 arc-seconds, then aggregated to 1000 meter resolution
Standard deviation, original data was at 7.5 arc-seconds, then aggregated to 1000 meter resolution
Maximal elevation, original data was at 7.5 arc-seconds, then aggregated to 10000 meter resolution
Minimal elevation, original data was at 7.5 arc-seconds, then aggregated to 10000 meter resolution
Median elevation, original data was at 7.5 arc-seconds, then aggregated to 10000 meter resolution
Mean elevation, original data was at 7.5 arc-seconds, then aggregated to 10000 meter resolution
Systematic subsample, original data was at 7.5 arcseconds, then aggregated to 10000 meter resolution
Breakline emphasis, original data was at 7.5 arc-seconds, then aggregated to 10000 meter resolution
Standard deviation, original data was at 7.5 arc-seconds, then aggregated to 10000 meter resolution
MCD12Q1: a satellite-based landuse types
Water
Evergreen Needleleaf forest
Evergreen Broadleaf forest
Deciduous Needleleaf forest
Deciduous Broadleaf forest
Mixed forest
Closed shrublands
Open shrublands
Woody savannas
Savannas
Grasslands
Permanent wetlands
Croplands
Urban and built-up
Cropland/Natural Vegetation mosaic
Snow and ice
Barren or sparsely vegetated
Unclassified
Miscellaneous
Restaurant density
Annual average traffic count data for the contiguous U.S. interpolated 100 m
Annual average traffic count data for the contiguous U.S. interpolated 1000 m
Annual average traffic count data for the contiguous U.S. interpolated 10000 m
Temporal terms
Dummy variable representing the year 2000
Dummy variable representing the year 2001
Dummy variable representing the year 2002
Dummy variable representing the year 2003
Dummy variable representing the year 2004
Dummy variable representing the year 2005
Dummy variable representing the year 2006
Dummy variable representing the year 2007
Dummy variable representing the year 2008
Dummy variable representing the year 2009
Dummy variable representing the year 2010
Dummy variable representing the year 2011

Dummy variable representing the year 2012
Dummy variable representing the year 2013
Dummy variable representing the year 2014
Dummy variable representing the year 2015
Dummy variable representing the year 2016
Dummy variable representing the weekday 1
Dummy variable representing the weekday 2
Dummy variable representing the weekday 3
Dummy variable representing the weekday 4
Dummy variable representing the weekday 5
Dummy variable representing the weekday 6
Dummy variable representing the Julian days
Variable representing the seasonal pattern – sine season
Variable representing the seasonal pattern – cosine season
Spatio-temporal lag of O3 measurements
Spatially lagged terms as inverse distance weighted O3 measurements at other locations, as well as their 1-day
Spatially lagged terms as inverse distance weighted O3 measurements at other locations, as well as their 3-day
Spatially lagged terms as inverse distance weighted O3 measurements at other locations, as well as their 5-day
Temporal lag of O3 predictors
1-day lagged moving values of air temperature
1-day lagged moving values of total precipitation accumulation
1-day lagged moving values of pressure
1-day lagged moving values of humidity
1-day lagged moving values of wind speed

Table S2 - Parameters Tuned for Base Learners

Neural Network		Random Forest		Gradient Boosting	
Parameter Name	Final Value after Tuning	Parameter Name	Final Value after Tuning	Parameter Name	Final Value after Tuning
Epochs	50	Number of trees	1200	Learning rate	0.007
Hidden layer and the number of neurons	2 hidden layers with 275 neurons in each layer	number of bins for numerical columns	20	Number of trees	200
L1 regularization	10^{-4}	number of bins for categorical columns	449	Column sample rate	0.5
Activation function	Rectifier	Maximum tree depth	9	Maximum tree depth	7
		Sample rate	0.42	Sample rate	1

Note: We used grid search to find optimal value for above parameters and used the final values for model training and model prediction. Take neural network as an example, to do grid search, we tried a series of parameter combinations in a parameter space (i.e., grid), fit neural networks, calculated cross-validated R^2, and chose the parameter combination that yielded the best model performance. If the chosen parameter combination was on the edge of parameter space, we slightly expanded the parameter space and repeated the above process.

Tables S3-S5

Table S3 – Cross-validation results by region

Region	Ensemble model						Neural Network	Random Forest	Gradient Boosting
	R^2	RMSE (ppb)	Intercept	Slope	Spatial R^2	Temporal R^2	R^2	R^2	R^2
East North Central	0.928	4.030	0.946	0.989	0.846	0.934	0.927	0.924	0.927
East South Central	0.912	4.161	0.383	0.986	0.779	0.924	0.909	0.909	0.911
Middle Atlantic	0.908	4.601	3.865	0.943	0.847	0.931	0.911	0.913	0.915
Mountain	0.862	4.642	1.594	0.969	0.789	0.891	0.855	0.855	0.857
New England	0.867	4.811	1.961	0.979	0.773	0.908	0.859	0.863	0.867
Pacific	0.891	5.467	1.295	0.970	0.879	0.896	0.881	0.884	0.886
South Atlantic	0.912	4.283	0.667	0.991	0.881	0.915	0.910	0.906	0.908
West North Central	0.920	3.699	1.089	1.028	0.843	0.929	0.917	0.917	0.919
West South Central	0.920	4.136	0.332	1.001	0.809	0.933	0.920	0.917	0.920

Note: Region division was based on U.S. Census Bureau. New England: Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island, Vermont; Middle Atlantic: New Jersey, New York, Pennsylvania; East North Central: Indiana, Illinois, Michigan, Ohio, Wisconsin; West North Central: Iowa, Nebraska, Kansas, North Dakota, Minnesota, South Dakota, Missouri; South Atlantic: Delaware, District of Columbia, Florida, Georgia, Maryland, North Carolina, South Carolina, Virginia, West Virginia; East South Central: Alabama, Kentucky, Mississippi, Tennessee; West South Central: Arkansas, Louisiana, Oklahoma, Texas; Mountain: Arizona, Colorado, Idaho, New Mexico, Montana, Utah, Nevada, Wyoming; Pacific: Alaska, California, Hawaii, Oregon, Washington. Although the Pacific Region includes Alaska and Hawaii, both states were not included in our modeling.

Table S4 – Cross-validation results by season

Season	Ensemble model						Neural Network	Random Forest	Gradient Boosting
	R^2	RMSE (ppb)	Intercept	Slope	Spatial R^2	Temporal R^2	R^2	R^2	R^2
Summer	0.885	5.320	0.168	1.004	0.891	0.903	0.884	0.877	0.880
Fall	0.863	4.375	0.814	0.976	0.848	0.894	0.863	0.853	0.857
Winter	0.853	4.313	0.539	0.991	0.688	0.885	0.853	0.844	0.848
Spring	0.879	4.682	0.526	0.989	0.819	0.904	0.878	0.871	0.874

Note: The seasons were defined as follows: summer (July – September), fall (October – December), winter (January – March), and spring (April – June).

Table S5 – Cross-validation results by population density

Season	Ensemble model						Neural Network	Random Forest	Gradient Boosting
	R^2	RMSE (ppb)	Intercept	Slope	Spatial R^2	Temporal R^2	R^2	R^2	R^2
Quartile 1	0.888	4.794	0.284	0.993	0.849	0.900	0.883	0.882	0.885
Quartile 2	0.911	4.388	0.018	1.002	0.875	0.924	0.908	0.903	0.907
Quartile 3	0.902	4.538	0.645	0.987	0.863	0.915	0.900	0.895	0.898
Quartile 4	0.911	4.643	0.249	1.005	0.864	0.925	0.900	0.899	0.903

Table S6 – Variables sorted by % of missing values.

Variables sorted by number of missings	% of missing values
MAIACUS_Optical_Depth_047_Terra_Nearest4	78.657
MAIACUS_Optical_Depth_055_Terra_Nearest4	78.657
MOD04L2_550	64.888
MOD11A1_LST_Day_1km_Nearest4	63.548
MOD11A1_Clear_day_cov_Nearest4	63.548
MOD11A1_LST_Night_1km_Nearest4	59.446
MOD11A1_Clear_night_cov_Nearest4	59.446
MAIACUS_cosVZA_Terra_Nearest	27.130
REANALYSIS_gflux_DailyMean	14.703
REANALYSIS_soilm_DailyMean	14.703
MOD13A2_Nearest4	3.334
CMAQ_NO2	3.108
CMAQ_NO2_Vertical	3.108
CMAQ_Ozone	3.108
CMAQ_Ozone_Vertical	3.108
CMAQ_PM25_TOT	3.108
CMAQ_PM25_Vertical	3.108
CMAQ_PM25_NO3	3.108
CMAQ_PM25_SO4	3.108
CMAQ_PM25_EC	3.108
CMAQ_PM25_OC	3.108
MERRA2aer_SO4	3.061
MERRA2aer_OCPHOBIC	3.061
MERRA2aer_OCPHILIC	3.061
MERRA2aer_BCPHOBIC	3.061
MERRA2aer_BCPHILIC	3.061
MOD09A1	2.696
RoadDensity_prisecroads1000	1.345
RoadDensity_prisecroads10000	1.345
RoadDensity_roads1000	1.252

USElevation_min100	0.232
USElevation_mea100	0.186
USElevation_bln100	0.186
USElevation_med100	0.139
NLCD_Barren100	0.139
NLCD_Developed100	0.139
NLCD_Herbaceous100	0.139
NLCD_Planted100	0.139
NLCD_Shrubland100	0.139
NLCD_Water100	0.139
NLCD_Wetlands100	0.139
USElevation_dsc10000	0.093
USElevation_max100	0.093
USElevation_max10000	0.093
USElevation_mea10000	0.093
USElevation_med10000	0.093
USElevation_min10000	0.093
USElevation_std100	0.093
USElevation_std10000	0.093
USElevation_bln10000	0.093
REANALYSIS_hpbl_DailyMax	0.093
REANALYSIS_shum_2m_DailyMax	0.093
REANALYSIS_prate_DailyMax	0.093
REANALYSIS_vis_DailyMax	0.093
REANALYSIS_apcp_DailyMean	0.093
REANALYSIS_dlwrf_DailyMean	0.093
REANALYSIS_dswrf_DailyMean	0.093
REANALYSIS_evap_DailyMean	0.093
REANALYSIS_hpbl_DailyMean	0.093
REANALYSIS_lhtfl_DailyMean	0.093
REANALYSIS_shtfl_DailyMean	0.093
REANALYSIS_shum_2m_DailyMean	0.093
REANALYSIS_snowc_DailyMean	0.093
REANALYSIS_tcdc_DailyMean	0.093
REANALYSIS_ulwrf_DailyMean	0.093

REANALYSIS_omega_DailyMean	0.093
REANALYSIS_weasd_DailyMean	0.093
REANALYSIS_prate_DailyMean	0.093
REANALYSIS_vis_DailyMean	0.093
REANALYSIS_hpbl_DailyMin	0.093
REANALYSIS_shum_2m_DailyMin	0.093
REANALYSIS_prate_DailyMin	0.093
REANALYSIS_vis_DailyMin	0.093
REANALYSIS_hpbl_1Day	0.093
REANALYSIS_shum_2m_1Day	0.093
REANALYSIS_prate_1Day	0.093
REANALYSIS_vis_1Day	0.093
REANALYSIS_air_sfc_DailyMin	0.093
REANALYSIS_air_sfc_DailyMean	0.093
REANALYSIS_air_sfc_DailyMax	0.093
REANALYSIS_air_sfc_1Day	0.093
REANALYSIS_windspeed_10m_DailyMax	0.093
REANALYSIS_windspeed_10m_DailyMean	0.093
REANALYSIS_windspeed_10m_DailyMin	0.093
REANALYSIS_windspeed_10m_1Day	0.093
NLCD_Barren10000	0.046
NLCD_Developed10000	0.046
NLCD_Herbaceous10000	0.046
NLCD_Planted10000	0.046
NLCD_Shrubland10000	0.046
NLCD_Water10000	0.046
NLCD_Wetlands10000	0.046

Figures S4-S8

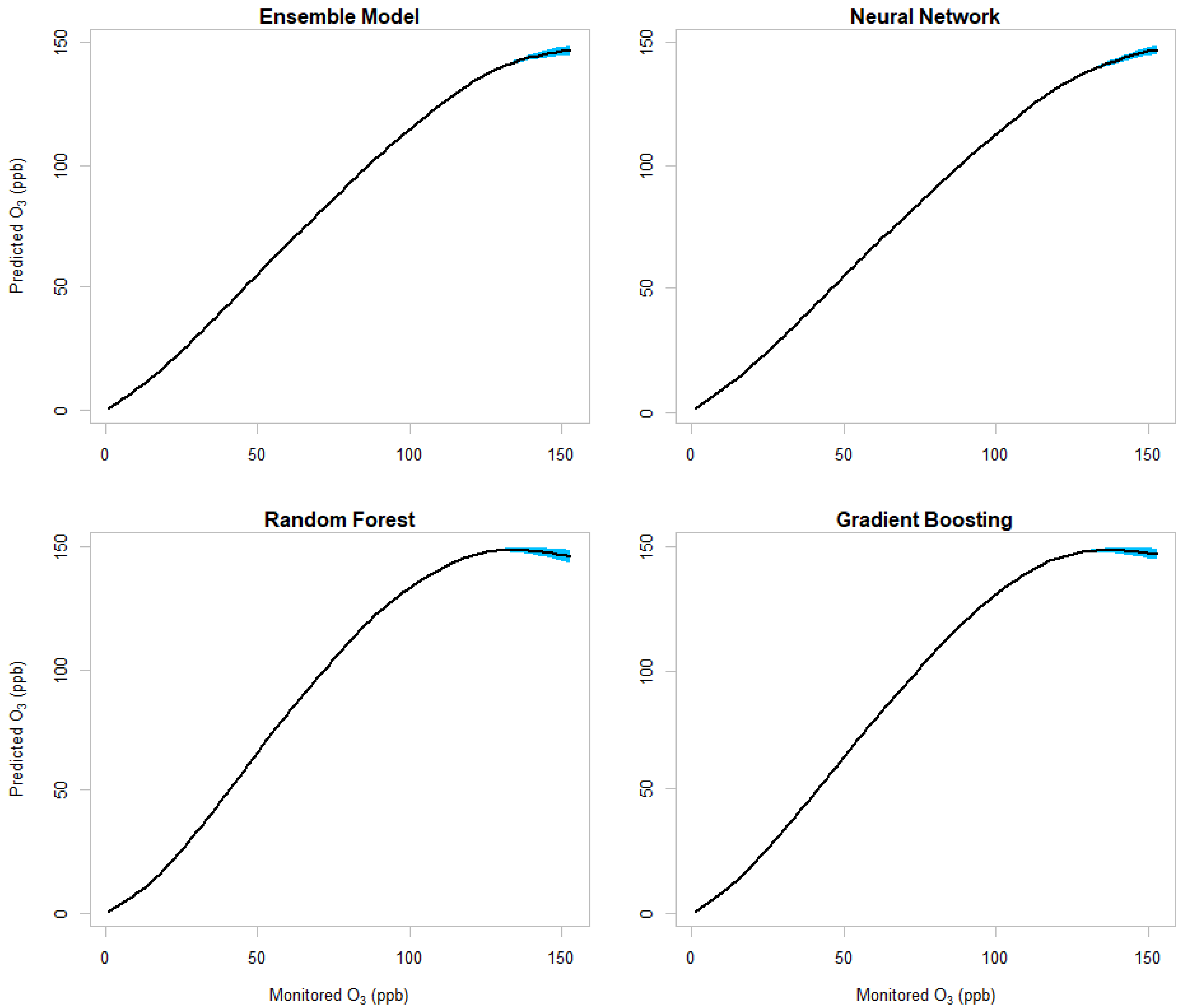

Figure S4 – O_3 levels predicted versus measured for the ensemble model and the three machine learning algorithms.

Note: We regressed daily predicted O_3 from each model (ensemble, neural network, random forest, and gradient boosting) against monitored O_3 using a GAM model with spline on the monitored O_3. Blue color represents 95% confidence interval.

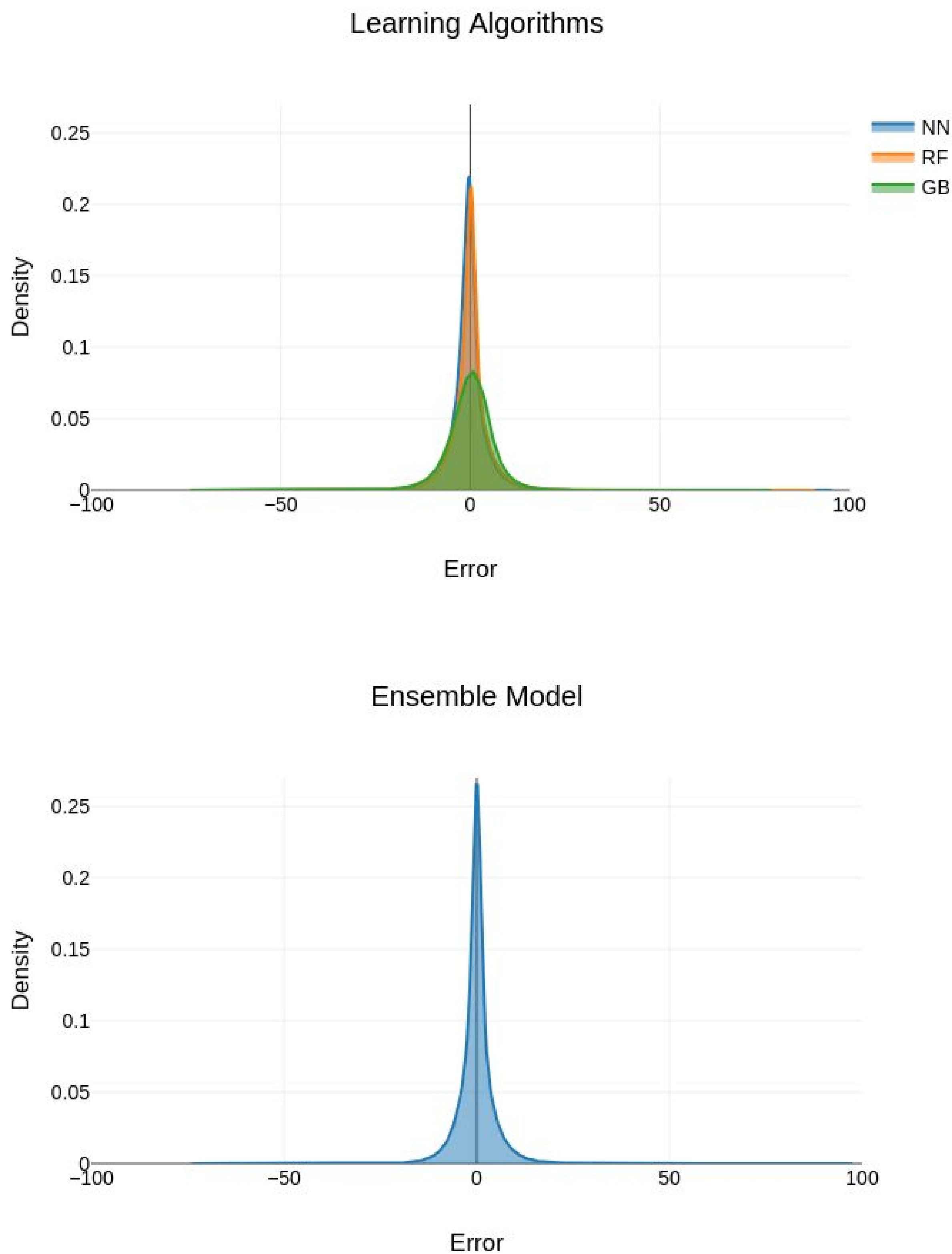

Figure S5 – O_3 mapping error estimates (ppb) from cross validation for ensemble model and three machine learning algorithms, where error = predicted – observed values at each site.

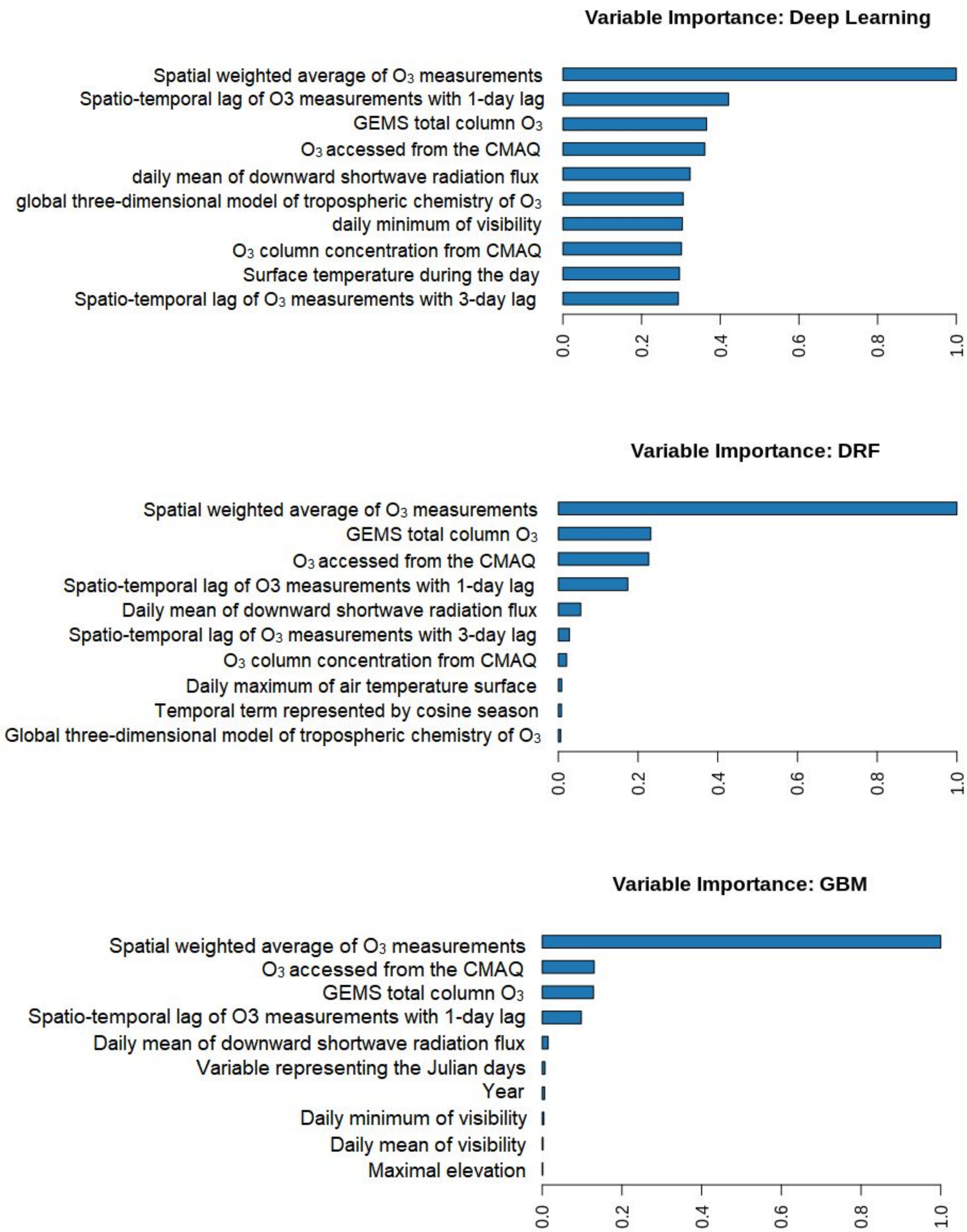

Figure S6 – Relative contribution of predictor variables for the three machine models.

Note: neural network (deep learning), random forest (DRF), and gradient boosting (GBM).

Note 2: We used the H2O package in R to run the three machine learning models. The command "h2o.varimp" extracts the list of variable importance. Some H2O algorithm class has its own methodology for computing variable importance. For random forest and gradient boosting, variable importance is determined by looking at whether a variable was selected to split on during the building process, and how much the squared error (over all trees) improved (decreased) as a result. For neural network, H2O uses the Gedeon method (Gedeon, 1997) - http://users.cecs.anu.edu.au/~Tom.Gedeon/pdfs/ContribDataMinv2.pdf.

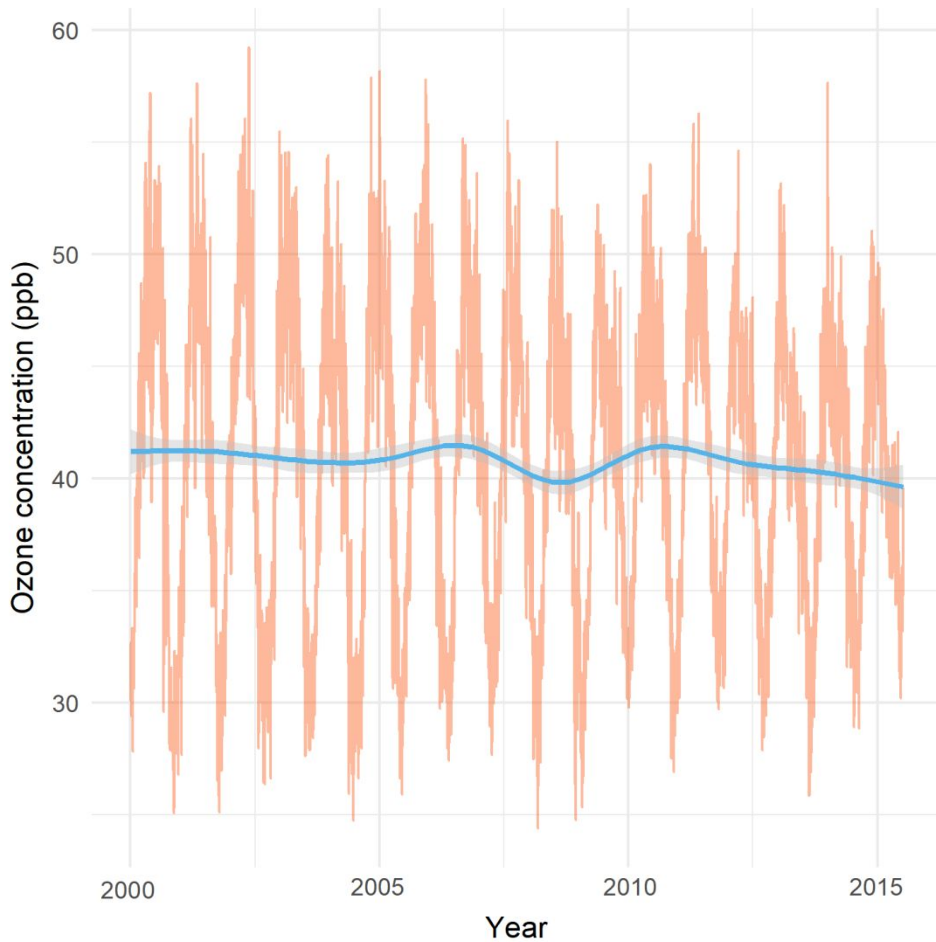

Figure S7 – Temporal trends of O_3.

Note: daily nationwide averages (orange line), smoothed conditional means function (blue line).

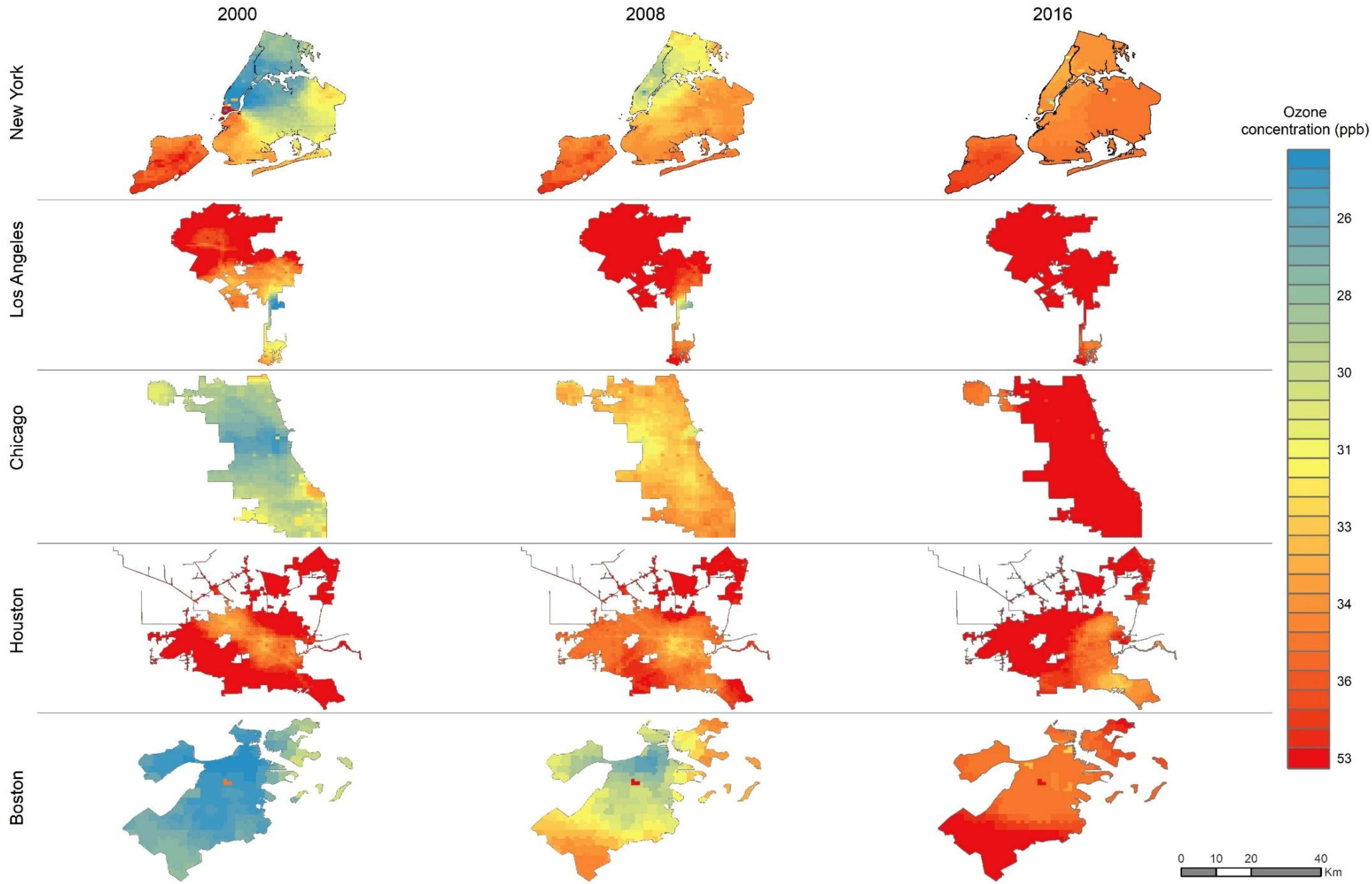

Figure S8 – Spatial distribution of the predicted levels of O_3 by the ensemble model for the major cities in the USA.

Note 1: We considered the top 4 cities in the US in terms of population – New York, Los Angeles, Chicago, Houston + Boston.

Note 2: in order to be possible the comparison of different levels of O_3 (represented by the legend with a color bar varying from blue [lowest concentration] to red [highest concentration]) over the cities and years, we standardized the symbolization (spatial distribution of the colors representing the ozone variation over space) based on the city and year with the lowest O3 concentration (New York, year 2000).