Missing Feature Data Report

Reetahan Mukhopadhyay
Lamont-Doherty Earth Observatory, The Data Science Institute at Columbia University

July 3, 2022

Intro

This document serves to provide guidance and clarity regarding the process of understanding how missing data points from observed data collected by various instruments may impact the performance of our machine learning models. Our models predict simulated PM2.5 concentrations based on other simulated species generated by the GEOS-Chem global atmospheric chemistry model, specifically in a region over South Asia.

When working with simulated data, there is no need to worry about data availability, as data can be created on-demand to cover all desired timeframes and locations. In the real world, however, we do need to deal with missing data points as factors such as atmospheric conditions (like cloud cover), instrument failure, etc can cause entries for certain locations on certain timesteps to not be properly recorded. For our models to be more realistic, this notion must be taken into account.

We inspect observed data of predictive species under our consideration to determine the spatiotemporal location (identified by a timestamp and grid cell - let us refer to these as CDs, or cell-days) of missing data points. This is used to create a "mask" which is then applied to all of our simulated data: we intentionally remove data on our simulated features at the same CDs where we observed missing data in reality. With this reduced simulated dataset, we regenerate our training and testing datasets and re-run our models to understand the change in performance with reduced data availability. To reiterate, these masks are applied to simulated data, so we are not using any observed data yet — just the pattern of missing data that appears in observed data. The next steps from here, however, would be collecting and applying observed data to our model.

Methodology

In this section, we discuss the process of creating the missing data masks and applying them for use with the machine learning model.

Observed Data

For this study, we considered AOD (aerosol optical depth), NO_2 (nitrogen dioxide) and CO (carbon monoxide). Our temporal range is daily data from January 1, 2015 to December 31, 2015. The AOD data is obtained from the NASA Center for Climate Simulation (NCSS)'s datashare, by selecting MAIAC, CMG_0.05degree, AOT5km and 2015. Both the NO_2 and CO data is collected from NASA's Goddard Earth Sciences Data and Information Services Center (GES DISC) datasets. The long name for the NO_2 set is OMI/Aura NO2 Cloud-Screened Total and Tropospheric Column L3 Global Gridded 0.25 degree x 0.25 degree V3, and for CO it is Aqua/AIRS L3 Daily Support Product (AIRS-only) 1 degree x 1 degree V7.0 (AIRS3SPD). Both of these were downloaded using the OPeNDAP subsetting tool, where we set the date range to be in the aforementioned desired temporal range. We set the spatial range to be in the box bounded by the latitudes 6°N to 36°N and 68 °E to 98 °E. The data variable of interest in NO_2 and CO are ColumnAmountNO2 and COCDSup_A (Carbon monoxide layer column density in molecules/cm^2 on ascending orbit), respectively. The AOD data does not have a subsetting tool, so we subset it manually to the desired spatiotemporal range after downloading.

As it can be inferred, the spatial grid dimensions for the AOD, NO_2 and CO observed datasets are 0.05°by 0.05°, 0.25°by 0.25°, and 1.0°by 1.0°, respectively. All grids are rectilinear. The GEOS-Chem grid is 0.25°by 0.3125°, thus some regridding will be necessary, as discussed later.

We downloaded the NO_2 and CO data using a batch script provided by NCSS and the list of file links, and the AOD with a simple in-house batch script.

Generating Missing Masks

A missing mask starts with the same spatiotemporal grid as the original dataset. Then, for each CD with data available in the original set is marked as present in the mask, and each missing day is marked as missing in the mask.

The first step is to perform a regridding process to get the spatial grid of our observed data to be the same as the GEOS-Chem grid, in order to be able to apply the mask to our simulated data.

Existing Library

In order to perform regridding, we may consider existing libraries. One such library is the xESMF library, a Python package that provides a simple interface for regridding based on the Earth System Modeling Framework (ESMF) regridding utility, that supports several regridding algorithms and other options. Two methods of interest are bilinear regridding and patch regridding. The bilinear method follows a two dimensional linear interpolation method while the patch method will solve for least squares. Note that the xESMF bilinear interpolation is implemented to mark any target cell that has at least 1 missing source cell (marked as NaN) to be marked as missing entirely — as any arithmetic computation with a NaN value leads to a result of NaN.

Regridding NO_2 and CO were simple using this library, but AOD was not due to the file format of the raw data. Regardless of technical difficulties, we saw motivation for generating a regridding method that allowed for more flexibility in regards to handling missing values. Thus, we describe our own developed regridding method.

Our Method

On a high level, our method takes boxes consisting of several cells from the original grid, generating an aggregate value to represent that box — depending on whether there are too many missing values — and write the value to the desired cell on the target grid. We will refer to target cells as CDs on the target grid, original cells as CDs on the original grid, and boxes are groups of adjacent cells in the original grid, altogether shaped as a rectangle, that are used to form a singular value in the target grid.

First, we need to determine the dimensions of these boxes. The target array represents the data in the 68 – 98 °E and 6 – 36 °N region on our 0.25 by 0.3125 °GEOS-Chem grid. This comes out to an array shape of 121 by 96, on one spatial grid for each day of the 365 days, so the total shape is (365,121,96). Suppose the target array was of shape (m_2, n_2) and our original array is of shape (m_1, n_1). Then we want to solve the system of linear Diophantine equations: $\sum_{i=1}^{n} a_i x_i = m_1$, $\sum_{i=1}^{n} x_i = m_2$, $\sum_{i=1}^{k} b_i y_i = n_1$, $\sum_{i=1}^{k} y_i = n_2$, under the constraints $\forall i \mid a_i, b_i, x_i, y_i \in \mathbb{Z}_{>0}$, and additionally optimize $\forall\ a_i, a_j, b_i, b_j \in \{..., a_i, ..., a_j, ...\}, \{..., b_i, ..., b_j, ...\} \mid a_i, a_j = argmin_{a_i,a_j}|a_i - a_j|, b_i, b_j = argmin_{b_i,b_j}|b_i - b_j|$. Here, a_i represents a box length for box type i, b_i is a box width, x_i is the number of boxes with length a_i, and y_i is the number of boxes of width b_i. These constraints tell us that the each length times the number of boxes associated with it summed up should cover the original array length, while the total number of boxes should equal the target array length. The analogous situation occurs for width, original array width and target array width. The additional constraints say we want all our lengths and counts to be positive integers, and that for all the chosen lengths and widths, we want them to be close to each other as possible. So for any pair of lengths or widths, we want to select the set such that this difference is the smallest possible — this constraint is added as a choice so the regridding can maintain a semblance of consistency.

For AOD, whose original grid array was 600 by 600, we can use the aforementioned equations and conditions to select our box dimensions and counts. Let us start with the length dimension, with $m_1 = 600$ and $m_2 = 121$: $a_1x_1 + a_2x_2 + ... + a_nx_n = 600$ and $x_1 + x_2 + ... + x_n = 121$, where $a_i, x_i \in \mathbb{Z}_{>0}$; plus, the constraint that tries to minimize the difference in box lengths. So we want to consider the solutions to the

system of these two equations that meet our constraints of positive integers and being close to each other. We select the solution $a_1 = 5, a_2 = 3, a_2 = 2$ which gives us $x_1 = 119, x_2 = 1, x_3 = 1$, and overall we get $5 * 119 + 3 * 1 + 2 * 1 = 600, 119 + 1 + 1 = 121$. We can repeat this procedure for the width with $m_1 = 600$ and $m_2 = 96$ to get $b_1y_1 + b_2y_2 + ... + b_ny_n = 600$ and $y_1 + y_2 + ... + y_n = 96$. We select $a_1 = 6, a_2 = 12$, thus $x_1 = 92, x_2 = 4$ and have $6 * 92 + 12 * 4 = 600, 92 + 4 = 96$. So, most of the 600 by 600 array is cut up into boxes of 5 by 6, and at the end we get some 3 by 6, 2 by 6, 3 by 12, 2 by 12, 5 by 12.

With NO_2 we are given a 121 by 121 box, that already was on the same resolution lengthwise so nothing had to done there, and for the width we went with $2 * 25 + 1 * 71 = 121, 25 + 71 = 96$. To get the regridded target grid, we used the mean of each box. However, we still need to consider missing values - a key motivation for this methodology at all.

In our method, we make flexibility with missing values a controllable parameter, through the missing value threshold. We let this threshold be set somewhere in $(0, 1)$. For each box as previously discussed, we get the fraction of original cells that are missing out of all the cells in the box. We then check this fraction against our threshold. If the box's missing fraction is at or above the threshold, then the target cell is also marked as missing. If it's below, then instead of marking the target cell as empty, we just take the average of the remaining non-missing original cells in the box to get the target. Suppose we were considering a target cell that is mapped to by one of the 5 by 6 boxes in the original AOD grid and our threshold is set to 0.4. This means if at least 12 original cells of the 30 are not NaN, we take the mean of the non-missing data as the target cell value. In the case of the NO_2 grid, of course the 1 by 1 "boxes" will either be 0% or 100% missing, so they are going to be unaffected by the threshold. The 1 by 2 boxes will be 0,50 or 100% missing, and it happens to be in our regridding that every cell captured in the set of 1 by 2 boxes is either 0 or 100% missing. This means that the NO_2 is actually unaffected by the missing threshold in regridding.

Additionally, instead of doing a weighting scheme by giving a weight to each source cell in a given box based on say, the distance from the center of the source cell to the desired target location, we simply weight all source cells evenly (a weight of 1.0 for all source cells).

Note that we did not mention CO yet; this is because CO is on a 1°by 1°grid, which is coarser than our target grid. So regridding for CO to the target CO grid would be expansion, and we cannot use the same method as we described. We propose an alternate method for handling coarser grid transformations.

The simple paradigm would be to also compute dimensions for "boxes", but these boxes would represent boxes in the target grid whose progenitor are single cells in the original grid. The aforementioned equations and constraints still apply, but the meanings for m_1, n_1 are flipped with m_2, n_2. Then, the values in these boxes would simply be a copy of the value of their progenitor cell in the source grid, effectively making the target grid by repeating values from the source grid to hit the desired resolution. If the value in the source grid was NaN, then that missing value still gets repeated for its corresponding target box.

For CO, the 1°by 1°grid over our region translated to a 31 by 31 source array. We chose $4 * 28 + 3 * 3 = 121, 28 + 3 = 31$ and $3 * 28 + 4 * 3 = 96, 28 + 3 = 31$ for our boxes, giving us that 1 source CO value will be repeated 12 times in a 4 by 3 box in most cases, as well 4 by 4, 3 by 4, and 3 by 3 a few times.

While this is the main method we discuss in the results, let us also describe a similar alternate method that allows for more noise as well as missing value flexibility to be taken into account. Suppose we still compute our boxes as before. Instead of repeating the source value, let us instead select values from a normal distribution $\mathcal{N}(s[i][j], \sigma_s^2 b_l b_w)$, where $s[i][j]$ represents the original source value at index (i, j) to be the mean, and the variance to be the variance of the entire source grid σ_s^2 (for that day) divided by the number of cells in the box ($b_l * b_w$ being box length times width). This allows to take into account the notion of the variance being inversely related to the sample size, and allows our new grid to capture some of the variance from the original grid, and keeps the target cells similar to their source cells, just with some noise. To handle missing values, first, prior to regridding, for every missing value in the source grid, we find the nearest non-missing value to it in the array, and store it. Now, we define two missing value parameters that we can call missing threshold over (MTO), and missing threshold under (MTU). In the case we are considering a missing value $s[i][j]$ in the source grid, we look up its nearest non-missing neighbor $nnm(s[i][j])$, and then we select values from $\mathcal{N}(nnm(s[i][j]), \sigma_s^2 b_l b_w)$ to fill in its box. Next, we randomly select $MTO * b_l b_w$ indices from the box - so we pick a fraction MTO of the box to be marked as missing. For non-missing source values, we perform the filling in of the target with $\mathcal{N}(s[i][j], \sigma_s^2 b_l b_w)$ as stated. Next, we randomly select $MTU * b_l b_w$ indices from the box - so we pick a fraction MTU of the box to be marked as missing. The idea behind this is that in our scheme for AOD and NO_2, when a cell gets marked as missing

or not missing, that does not mean all the source cells were present or missing, just that the number did/did not hit the threshold. So here, we choose to black out MTU of values for a present box, such that if we did the reverse process and the threshold were set to lower than MTU, it would be marked as present. Similarly, we black out MTO of values for a missing box, such that if we did the reverse process and the threshold were set to higher than MTO, it would be marked as missing. Setting MTU to be in $(0, 0.5]$ and MTO to be in $[0.5, 1.0)$ would make practical sense. However, this procedure is more difficult to justify from a physical perspective, and more research would be needed to determine whether this would be a better procedure to utilize for upsampled regridding compared to the simpler method.

Our "simplified" version of this without the random variables and missing value procedure can actually be derived from this method, simply set $MTO = 1.0$, $MTU = 0.0$, and $\sigma_b^2 = 0$, which sets that all values in a box are marked for a missing source cell to be missing, does not replace any values derived from a present source cell with missing values, and setting the variance in a box to 0 makes all values simply equal the mean, which is the value copied from the source. We will consider this simplified method to be our default setting.

Note that all of the aforementioned regridding is applied to the array for each individual day; the regridding applied to a location on one day does not affect the regridding applied to the same location, or any other location, on a different day. So we iterate over each day, running this regridding procedure the day's grid. Our final output shape is $(365, 121, 96)$.

Once our regridding is complete, we make the mask by creating a new spatiotemporal grid with the same dimensions our target grid that is blank. For every value with an actual number in our regridded observed data, we mark it as present in our mask, and each NaN target cell is marked as missing in our mask.

Once this mask is created, we store it and then apply to all the other GEOS-Chem simulated features. This marks every simulated CD in each feature dataset that is marked as a missing CD in our mask as missing in the simulated dataset. Each simulated CD in each feature dataset that as marked as present in our mask is simply unaffected, its value is unchanged.

Automated Machine Learning

We now aim to supply this data to our machine learning framework. Effectively, we read in all the masked datasets and create one large dataset with each species to be a feature represented as a column, and each row corresponding to 1 CD (as opposed to one timestamp, or one location). Since each row represents 1 CD, and the mask applied to each species is the same, and the mask covers particular CDs, this means that it is impossible for only some values in a row to be missing — either the entire row is present, or the entire row is missing. These missing rows get dropped from the data. This is then split into a training and testing set, and can be fed into our model.

We utilize the Fast Lightweight Automated Machine Learning (FLAML) framework for AutoML, a machine learning framework that tries several tree-based models and employs Bayesian hyperparameter tuning to find the best models and hyperparameters given the training and testing set.

Results

In this section, we discuss and display the presence of the missing data itself, as well as present the new results obtained from the machine learning model.

Presence of Missing Data

Prior to feeding any data into machine learning, we wanted to do some exploratory data analysis (EDA) by inspecting and visualizing the missing data.

Method	AOD	NO_2	CO
Raw	59.06	30.15	23.99
Custom 1%	42.39	30.15	-
Custom: 10%	46.64	30.15	-
Custom 20%	50.60	30.15	-
Custom 30%	53.69	30.15	-
Custom 40%	56.39	30.15	-
Custom 50%	58.10	30.15	-
Custom 60%	61.37	30.15	-
Custom: 80%,10%	-	-	30.16
Custom: 65%, 25%	-	-	42.80
Custom: Trivial	-	-	23.87
XESMF: Bilinear	61.85	34.10	28.86
XESMF: Patch	71.09	38.85	38.47

Table 1: Percentage of cell-days marked as missing after applying regridding scheme for each species (raw is no scheme, and thus is on its original grid). Custom N % methods are our methods for AOD and NO_2 with a missing value threshold set to N%, the higher the threshold means a higher proportion of cells in a box must be present for it to be marked as present. Custom {MTO}%, {MTU}% refer to our method for CO with MTO and MTU referring to the two parameter missing threshold discussed earlier: missing threshold over (for when the original value is missing) and missing threshold under (for when the original value is present). Custom Trivial is equivalent to the copied and repeated value method as discussed earlier — and equivalent to Custom 100%, 0% with 0 variance. XESMF Bilinear and Patch are by applying the XESMF library with those methods.

In Table 1 we can look at the prevalence of missing data prior to as well as under each regridding methodology. As previously stated, NO_2 was laid out such that for every box that had 2 cells in them, either were all or nothing, and of course boxes with 1 cell are equivalent to the original grid, thus NO_2 was unaffected by our regridding. As we can note, the library functions yielded more missing values than our method for NO_2 and especially for AOD, as they do not have this sort of flexibility with missing values and are set to mark a chunk as missing when it comes across one. Our trivial regridding for CO also yielded less missing data than the library functions as well. Note that it is very similar although not exactly equal to the raw data, as it is mostly just maintaining the same proportions by copying present and missing values for the 4 by 3 boxes that make up most of the regridding, but it maybe that for the small amount of other boxes with other sizes - 9 (3 by 3) and 16 (4 by 4), more of the 4 by 4s may happen to be present values while more 3 by 3s happen to be missing values, which would make the final missing proportion slightly lower than the original. We also noticed with the nontrivial CO regriddings, since there are far more raw present than missing values, altering the MTU has a bigger effect than the MTO.

Clearly, AOD by far is the bottleneck in terms of availability, even under generous thresholds — so the harshest missing mask would be based on that species. As you make the threshold more and more generous (towards 0), a question arises of how representative really is using the average of a small fraction of cells to represent the mostly missing box. Based on understanding about species persistence in the atmosphere, we believe that 10% would be a valid threshold to use for a mask based on AOD. As an additional check, we picked a target cell over a location with generally high AOD coverage (over 70% of days with present data), them taking a sample of 10% cells in the box, getting an average and repeating this for each day and checking the time series, especially how it compares to selecting 30% or 50% of cells.

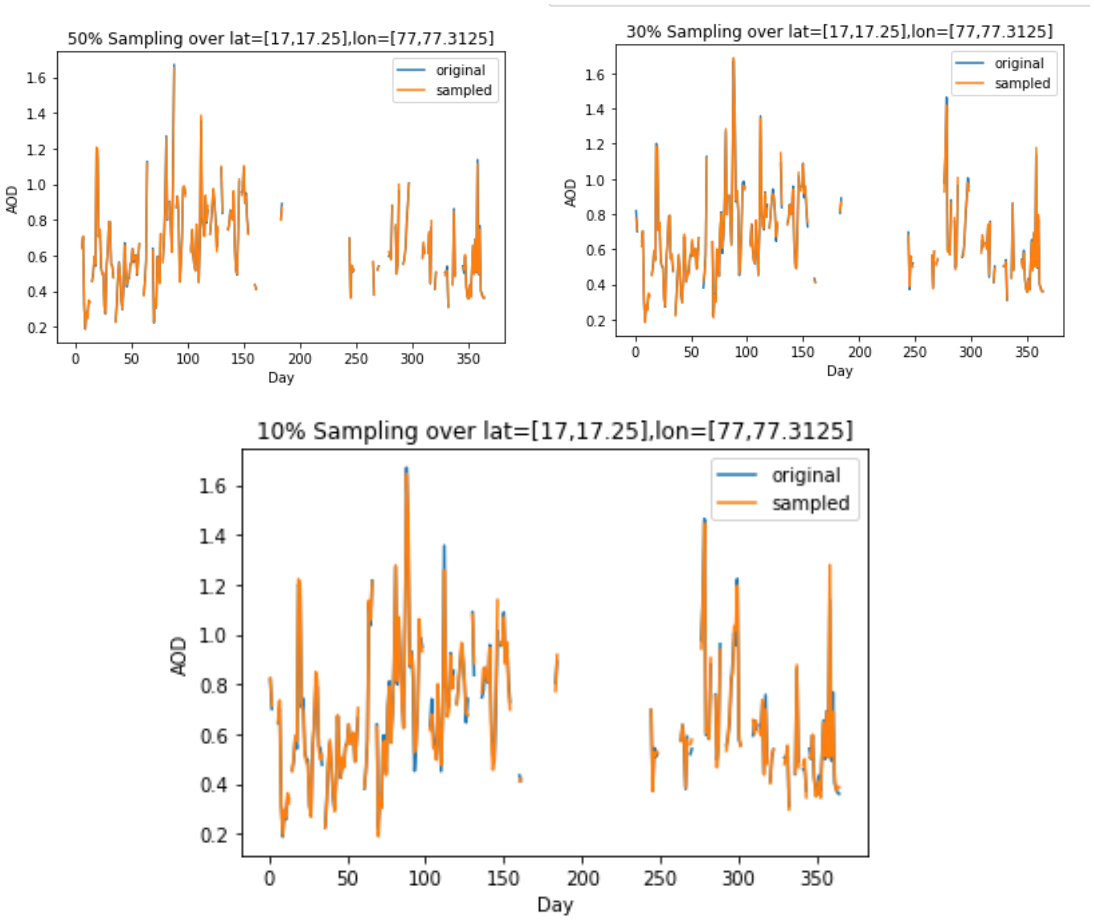

Figure 1: Time series over 2015 for sampling 50%, 30%, and 10% of original cells needing to present in our target cell enclosed by 17-17.25°N, 77-77.3125°E, versus the original time series with all present data

As we can see, the 10% sampling seemed to remain reasonably similar to the original time series. This gives some support to the notion that 10% as a threshold to use for an AOD mask would be reasonable, in the logic that for regions that do have low AOD availability, using 10% of possible points may still give a representative sample of what the true AOD value is. There are some issues with this approach, in that the conditions of regions with lower availability are inherently different than where we have higher availability (Himalayas vs South India), as well as that the missing cells in a target box may not be an entirely randomly distributed. Nevertheless, this test gives at least some support to using the 10% threshold.

As for a look at geographically, where the missing data is, we provide maps of data availability for our three species under different regridding algorithms. The percent refers to the fraction of days, out of 365, that the location had present data. For these plots, we do not plot data over water bodies, which of course includes the Bay of Bengal, the Arabian Sea and Indian Ocean, but also inland water bodies, so the missing patches of coloring inland are target cells that were deemed to be mostly lakes or other inland water bodies.

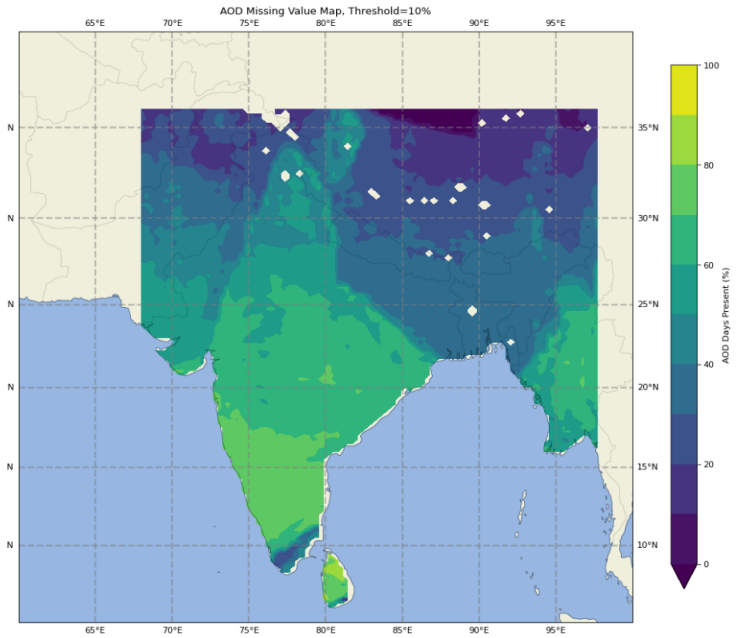

Figure 2: Percent of days in 2015 with present AOD data for locations in South Asia under 10% thresholds for missing data in custom regridding.

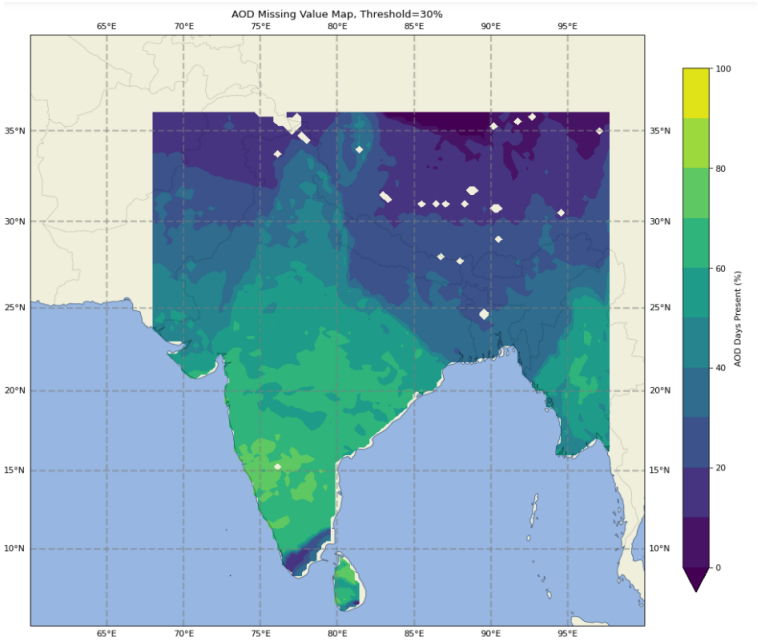

Figure 3: Percent of days in 2015 with present AOD data for locations in South Asia under 30% thresholds for missing data in custom regridding.

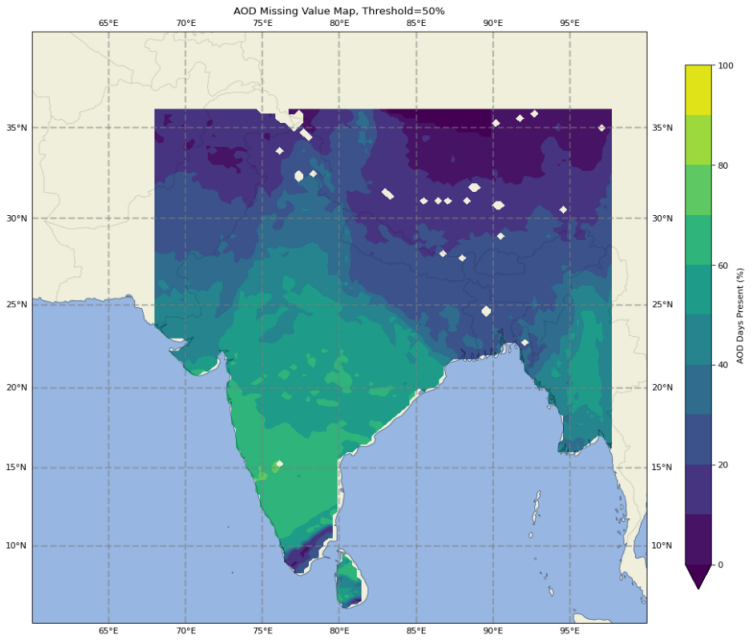

Figure 4: Percent of days in 2015 with present AOD data for locations in South Asia under 50% thresholds for missing data in custom regridding.

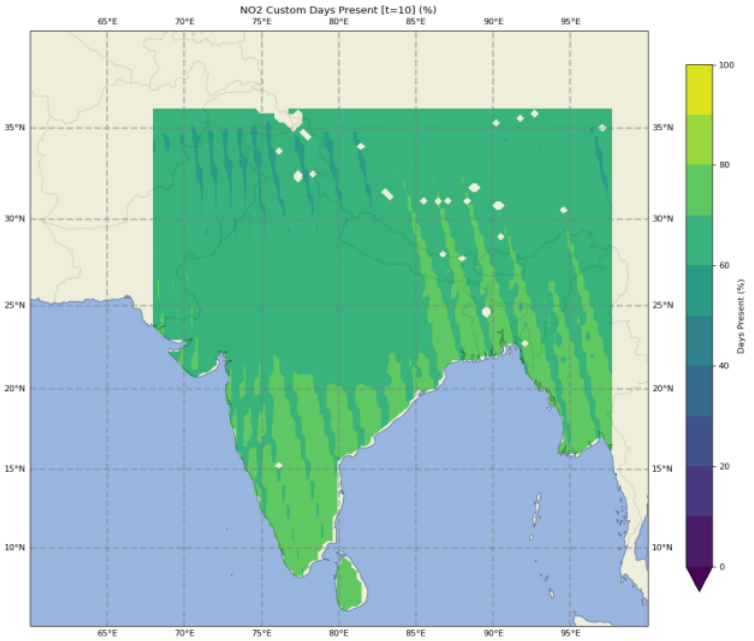

Figure 5: Percent of days in 2015 with present NO_2 data for locations in South Asia under 10% threshold in custom regridding.

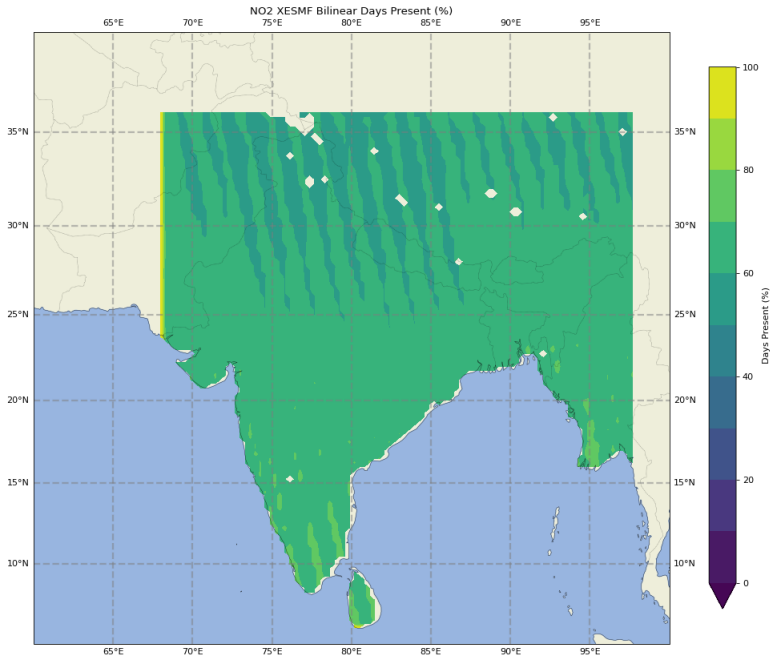

Figure 6: Percent of days in 2015 with present NO_2 data for locations in South Asia under XESMF's bilinear regridding scheme.

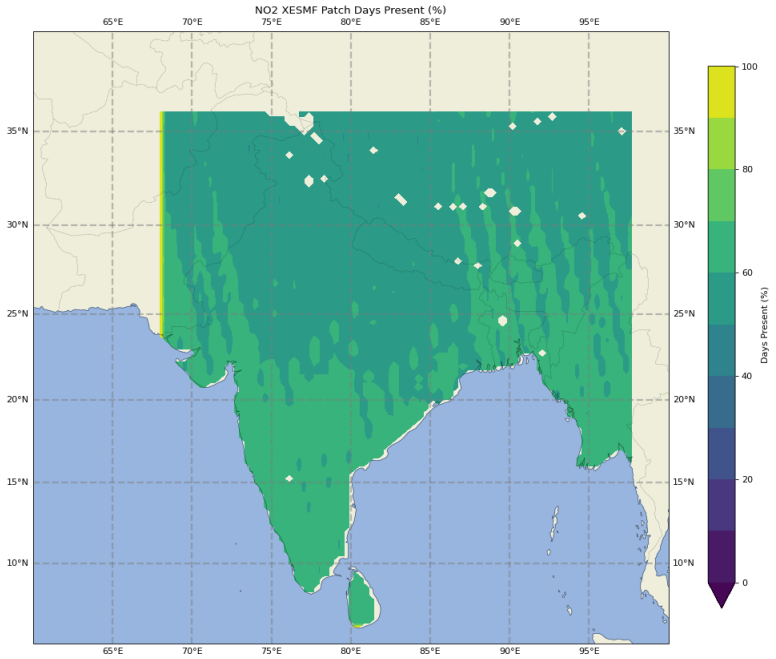

Figure 7: Percent of days in 2015 with present NO_2 data for locations in South Asia under XESMF's patch regridding scheme.

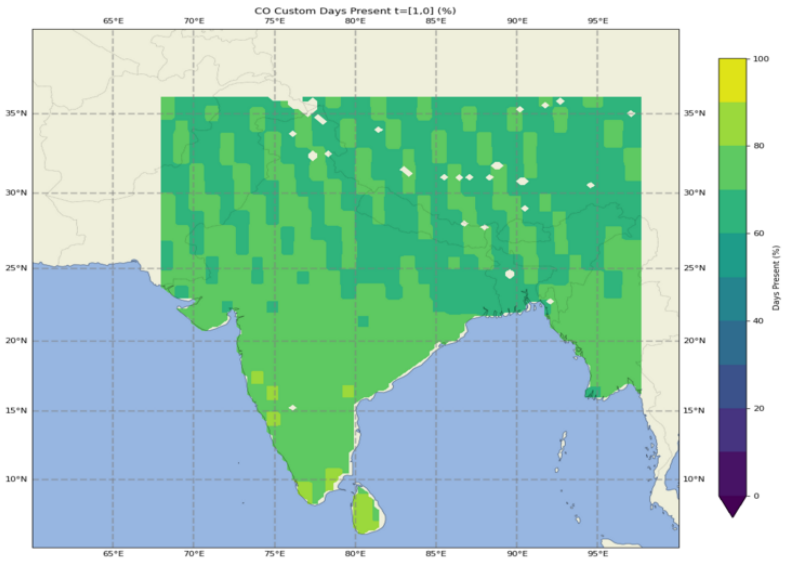

Figure 8: Percent of days in 2015 with present CO data for locations in South Asia under trivial custom regridding scheme.

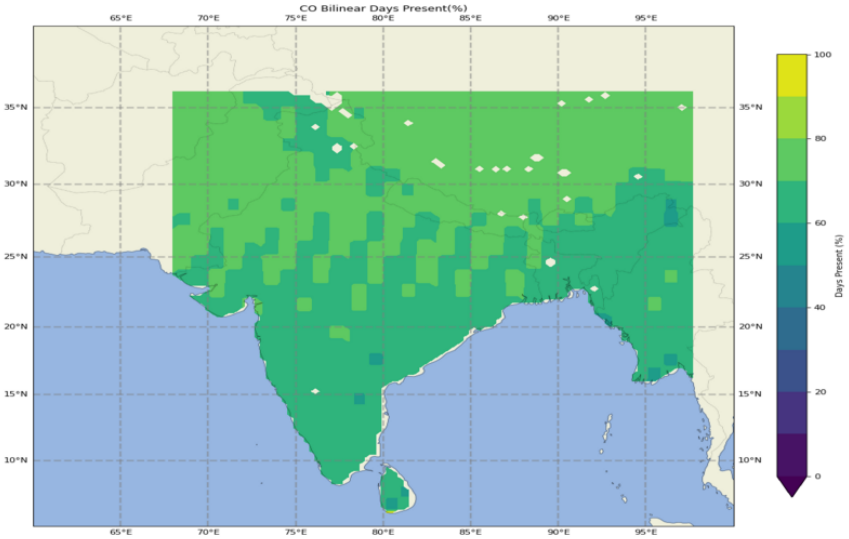

Figure 9: Percent of days in 2015 with present CO data for locations in South Asia under XESMF's bilinear regridding scheme.

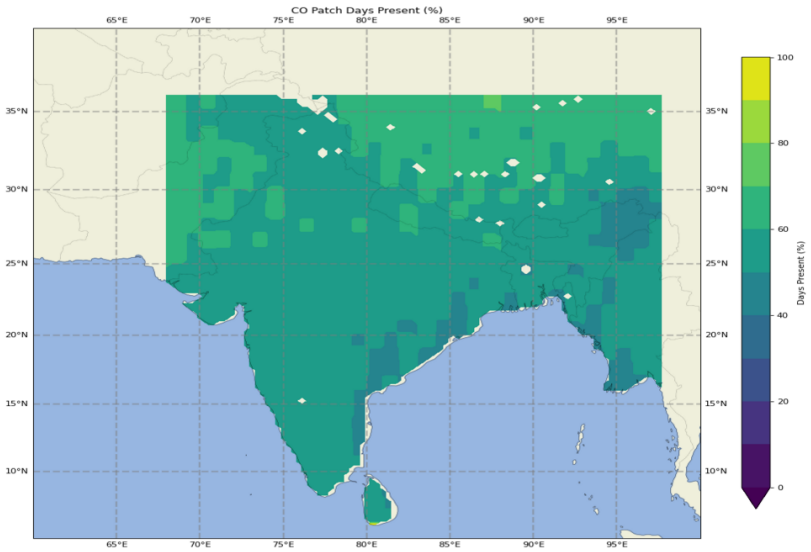

Figure 10: Percent of days in 2015 with present CO data for locations in South Asia under XESMF's patch regridding scheme.

For AOD, data availability clearly plummets towards higher elevation areas, such as around the Himalayas, and peaks in South India (with some unknown reason for low availability in the southernmost part of India, that does not extend into Sri Lanka). The darker colors diffusing across the map is apparent as the thresholds get more stringent. For NO_2 we get a "striped" pattern of sorts, with certain striped regions getting lower or higher availability. CO experiences a similar phenomenon, although its stripes seem to be more jagged. Like AOD, NO_2 showed more availability in South India, and CO seems to do so for our custom trivial method but this does not seem to follow for the library regridding methods. Further investigation would be required to determine the differences in the regridding pattern for CO.

Model Performance

Now let us take an actual look at the results of our exploration. We provided training and testing sets to AutoML in the format as discussed earlier. AutoML tries several models and uses Bayesian hyperparameter tuning to find optimal hyperparameters for its selected model. First, let us present the reference result for when there is zero missing data, in Table 2 .

R^2	RMSE	MAE	Model Type
0.999	2.61	0.73	XGBoost Limited Depth

Table 2: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set for the features with no missing masks, as well as the type of model AutoML selected.

Next, let us look at the results for applying the missing mask based on the AOD, NO_2 and CO observed data under different regridding methods.

	R^2	RMSE	MAE	Model Type
Custom - 10%	0.921	4.80	2.84	LGBM
Custom - 30%	0.931	4.63	2.81	XGBoost
Custom - 50%	0.922	4.70	2.86	LGBM

Table 3: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set for the features with the missing mask based on AOD with 10%, 30%, and 50% thresholds applied, as well as the type of model AutoML selected.

	R^2	RMSE	MAE	Model Type
Custom - 10%	0.923	4.52	2.79	Random Forest
Bilinear	0.915	4.79	3.03	Extra Trees

Table 4: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set for the features with the missing mask based on NO_2 under the custom regridding method (10%), the XESMF Bilinear, and XESMF Patch methods, as well as the type of model AutoML selected.

	R^2	RMSE	MAE	Model Type
Custom - Trivial	0.914	4.62	2.78	LGBM
Bilinear	0.914	4.62	2.80	XGBoost

Table 5: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set for the features with the missing mask based on CO under the custom trivial regridding method, the XESMF Bilinear, and XESMF Patch methods, as well as the type of model AutoML selected.

It appears that the answer to our key inquiry — can our machine learning modelling framework be robust to swaths of missing data, missing data in a pattern matching missing data in reality, is yes. The performance in general seems to definitely be worse than the non-missing data, but it is not a tremendous drop-off, we still achieve high performance. The performance of predicting PM2.5 levels seems to remain quite satisfactory, across the board, even on the harshest missing mask: the AOD 50% threshold. The performance differences between the different regridding strategies also appears to be fairly minimal, suggesting that regardless of the reasonable method we select to handle missing data and how stringent it may be, the model performance is capable of staying high. There also seems to be relatively little performance difference among the different feature masks. Of course, there does exist a point in which enough data can be taken away from the model training and testing process to cause a severe performance drop, but this study aimed to simply investigate whether it would be robust against realistic quantities and patterns of missing data.

Withholding Features

Additionally, we wanted to know how robust our model is not only to missing data, but also to missing species. Thus, we dropped chunks of features, re-trained the model and checked the new testing output. We grouped the species into AOD-based, gaseous and meteorological features and tried withholding each (one at a time). We defined these groups through Table 6.

Abbreviation	Name	Group
AOT_C	Aerosol optical thickness (or AOD) at 550 nm	AOD
AOT_DUST_C	Aerosol optical thickness (or AOD) of dust at 550 nm	AOD
CO_trop	tropospheric vertical column of CO	Gas
SO2_trop	tropospheric vertical column of SO2	Gas
NO2_trop	tropospheric vertical column of NO2	Gas
CH2O_trop	tropospheric vertical column of CH2O	Gas
NH3_trop	tropospheric vertical column of NH3	Gas
T2M	2-meter air temperature	Met
PBLH	Planetary boundary layer height	Met
U10M	10-meter eastward wind	Met
V10M	10-meter northward wind	Met
PRETCOT	Total precipitation	Met
RH	2-meter relative humidity	Met

Table 6: Select features by their full name, abbreviation and which grouping they belong to

First, we provide the reference result for no missing mask, but with the dropped feature groups as described, in Table 7.

	R^2	RMSE	MAE
No AOD	0.998	3.20	0.92
No Gas	0.998	3.56	1.02
No Met	0.998	3.47	0.94

Table 7: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set with no missing mask applied, while dropping the AOD-based, gaseous, and meteorlogical features from the training and testing sets.

We tried dropping each set of features in tandem with trying out our different regridding methods. We present the results of feature dropping for the Custom 50% AOD run, the Custom 50% NO_2 run, and the Custom Trivial CO run.

	R^2	RMSE	MAE
No AOD	0.869	6.72	4.25
No Gas	0.856	7.06	4.22
No Met	0.847	7.28	4.54

Table 8: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set under the Custom, 50% missing threshold AOD mask, while dropping the AOD-based, gaseous, and meteorlogical features from the training and testing sets.

	R^2	RMSE	MAE
No AOD	0.894	5.28	3.32
No Gas	0.888	5.44	3.26
No Met	0.885	5.50	3.42

Table 9: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set under the Custom, 50% missing threshold NO_2 mask, while dropping the AOD-based, gaseous, and meteorological features from the training and testing sets.

	R^2	RMSE	MAE
No AOD	0.868	5.71	3.53
No Gas	0.856	5.91	3.54
No Met	0.848	6.12	3.68

Table 10: R^2, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) on the testing set under the Custom, Trivial missing threshold CO mask, while dropping the AOD-based, gaseous, and meteorological features from the training and testing sets.

Clearly, there is a drop in performance when dropping features compared to the last subsection, in which all features were included. There is, of course, also an obvious drop compared to the non-missing data, but like with the full-feature data, the results remain satisfactorily high. However, the question of which set of features seems to carry the greatest importance in terms of causing the largest performance drop does not seem apparent; the performance when considering all three metrics for evaluation does not seem to make any one feature stand out, in any of the feature masks considered here. Once again, there also does not seem to be a big difference across the different feature masks.

The code for the regridding process and machine learning workflow is available at [1]

[1] https://github.com/reetahan/regridding-geos-chem