Accurate and Scalable Gaussian Processes for Fine-Grained Air Quality Inference

Zeel B Patel, Palak Purohit*, **Harsh M Patel***, **Shivam Sahni***, **Nipun Batra**
IIT Gandhinagar
{patel_zeel, palak.purohit, harsh.patel, shivam.sahni, nipun.batra}@iitgn.ac.in

Abstract

Air pollution is a global problem and severely impacts human health. Fine-grained air quality (AQ) monitoring is important in mitigating air pollution. However, existing AQ station deployments are sparse. Conventional interpolation techniques fail to learn the complex AQ phenomena. Physics-based models require domain knowledge and pollution source data for AQ modeling. In this work, we propose a Gaussian processes based approach for estimating AQ. The important features of our approach are: a) a non-stationary (NS) kernel to allow input depended smoothness of fit; b) a Hamming distance-based kernel for categorical features; and c) a locally periodic kernel to capture temporal periodicity. We leverage batch-wise training to scale our approach to a large amount of data. Our approach outperforms the conventional baselines and a state-of-the-art neural attention-based approach.

Introduction

Today, 91% of the global population lives under unsafe levels of AQ (WHO 2021). Long-term exposure to $PM_{2.5}$ increases cardiopulmonary mortality by 6–13% per 10 µg/m^3 of $PM_{2.5}$, which causes yearly 8 million deaths worldwide (Kloog et al. 2013). Air quality (AQ) is affected by multiple factors, including but not limited to physicochemical processes, meteorological variables, and the geography of a place. Primary air pollution sources are solid fuels used in domestic cooking, industrial plants, vehicular emissions, roadside dust, and construction activities (Balakrishnan et al. 2019). Thus, air pollution is a complex spatio-temporal phenomenon, and fine-grained AQ monitoring is essential to make informed decisions towards air pollution mitigation.

Nations across the globe have sparse and non-uniform AQ station deployments. Existing techniques to generate AQ maps rely on interpolation approaches like Kriging, Spline, and Trend. Some recent approaches have taken a data-driven approach using deep learning to generate AQ maps (Cheng et al. 2018; Xu and Zhu 2016; Zheng, Liu, and Hsieh 2013). These methods: i) do not quantify uncertainty which may help policymakers make informed decisions; and ii) do not incorporate domain knowledge in modeling.

*These authors contributed equally.

Gaussian processes (GPs) are non-parametric Bayesian models often used in environment modeling (Sampson and Guttorp 1992) (aka Kriging (Krige 1951)). GPs implicitly provide uncertainty estimates along with predictions that can be useful for policymakers. The key component of GPs is covariance functions (aka kernels), which govern the characteristics of resultant fit on the data. Based on domain knowledge, the covariance functions can be combined to approximate underlying phenomena efficiently.

Widely used GP packages (Gardner et al. 2018; GPy since 2012) use a single length scale (a parameter governing smoothness of fit) for multiple features by default. In practice, features have a diverse relationship with observations, including variable or negligible prediction power. To neglect non-useful features and have better predictions with useful features, we use the ARD (Automatic Relevance Determination), enabling GPs to learn the length scale parameter individually for each feature.

Often, restrictive assumptions are made in GPs such as stationarity (Guizilini and Ramos 2015). Stationarity means that the covariance between two locations depends only on the Euclidean distance between them. For AQ, it is possible to have variable covariance for the same distance apart locations due to geographical heterogeneity and complex chemical reactions. To address this challenge, we utilize an approach given by (Plagemann, Kersting, and Burgard 2008) to induce non-stationarity by allowing input-depended smoothness in model fit. Our dataset contains categorical features such as wind direction (West, East,..), which are difficult to model because standard kernels are designed for continuous features. To address this problem, we use a Hamming distance-based kernel (Hutter et al. 2014) that is well suited for categorical features. Environmental processes often are periodic over time. Air pollution may have periodicity due to specific reasons such as: i) diurnal patterns of traffic; ii) yearly patterns of seasons and periodic nature of other meteorological features that affect AQ. We utilize periodic kernels combined with other kernels to encode this information.

We use the hourly AQ and meteorological data from Beijing (Cheng et al. 2018) and London over a month to evaluate our approaches. We compare our approach against: i) conventional spatial interpolation baselines such as Inverse Distance Weighting, K-Nearest Neighbors; ii) standard machine learning models such as XGBoost and Random Forest; and iii) a state-of-the-art neural attention based model ADAIN (Cheng et al. 2018). Our approach outperforms the baselines for both datasets in a cross-validation setting. We analyze the effects of using various proposed techniques and point out the strengths and weaknesses of each method.

We believe that our approach will make uncertainty-aware fine-grained AQ inference accurate and help policymakers make informed decisions to reduce air pollution. Our work is fully reproducible, and the code, data, and appendix are hosted at https://github.com/patel-zeel/AAAI22.

Related Work

We now discuss the related work across: i) Dispersion models; ii) forecasting; and iii) spatio-temporal inference.

Dispersion Models

Dispersion models are used to model AQ by mathematically approximating the physicochemical processes governing air pollution dynamics. Widely used dispersion models include Gaussian plumes, Street canyon models (Fallah-Shorshani, Shekarrizfard, and Hatzopoulou 2017) and computational fluid dynamics. These methods model AQ as a function of meteorology, traffic volume, and emission factors based on several empirical assumptions. These models require deep domain expertise and it is non-trivial to collect and update the required data. In our approach, we instead use publicly available and easily measurable data.

AQ Forecasting

AQ forecasting is a problem where a model predicts AQ t timestamps ahead in the future, leveraging available information till the current time. It is a well-explored problem due to the rise of neural network-based time-series modeling. Recently, (Luo et al. 2019) proposed a KDD-18 cup winning solution for AQ forecasting. The authors use a novel combination of LightGBM, Gated-DNN, and Seq2Seq models. Recent work (Yi et al. 2018) proposed a deep distributed fusion network-based method to forecast AQ on a large scale (300+ Chinese cities). Earlier work (Zheng et al. 2015) used a combination of linear regression for temporal dimension and neural network-based spatial predictor. We aim to solve a related but different problem to infer AQ at unmonitored locations at a given timestamp instead of forecasting in the future at the monitored locations.

AQ Inference

AQ inference is a problem of modeling AQ as a function of several features (meteorology, traffic, and other features). Previous work (Zheng, Liu, and Hsieh 2013) proposed a co-training based method on top of a neural network and linear-chain conditional random field (CRF) method for AQ inference. The authors perform classification based on standard ranges of AQ levels decided by the United States Environmental Protection Agency. The authors have used meteorological features, POIs (Points of Interests), road networks, traffic-related features, and mobility features. We use fewer features due to public data availability and focus on the regression task instead of classification. Recent state-of-the-art work (Cheng et al. 2018) proposes a neural attention-based approach to incorporate time-invariant and time-series features together. They also learn the effect of individual train stations on a test location via an attention net. We use similar features as the authors except road networks and POIs due to the unavailability of data.

Problem Statement

Given a set of AQ monitors $\mathcal{S}$, timestamps $\mathcal{T}$, features (latitude, longitude, temperature, humidity, weather, wind speed and wind direction) and corresponding $\text{PM}_{2.5}$ observations, the aim is to predict $\text{PM}_{2.5}$ at a new set of locations $\mathcal{S}^*$ for the same $\mathcal{T}$ timestamps.

Our Approach

We desire to have two main characteristics in AQ modeling: i) uncertainty along with AQ predictions and ii) incorporating domain-inspired information into the model. We propose Gaussian processes (GPs) based models which can implicitly provide uncertainty and incorporate domain-specific information via an appropriate kernel design. We first introduce the basics of GPs, then we discuss limitations of standard GPs and ways to overcome them with non-stationary GPs. Furthermore, we discuss feature-specific kernel design, and finally, batch training of GPs for scalability.

Stationary GPs

GPs assume a prior distribution of functions over the $\text{PM}_{2.5}$ observations $\mathbf{y}$.

$$f(\mathbf{x}) \sim \mathcal{GP}_y(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{1}$$

$$y = f(\mathbf{x}) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{2}$$

where, $\mathbf{x} \in \mathbb{R}^d$ is a feature vector, $y \in \mathbb{R}$ is observation, and ϵ is noise in observations with variance σ^2. $m(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$ is the prior mean function and $k(\mathbf{x}, \mathbf{x}') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the prior covariance function. In practice, we assume constant mean function without loss of generality. A well-known covariance function (kernel) is Radial Basis Function.

$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||_2^2}{2\ell^2}\right) \tag{3}$$

Where σ_f^2 is kernel variance and ℓ is length scale. In the current setting, $\boldsymbol{\theta} = \langle \sigma_f, \sigma, \ell \rangle$ are the model hyperparameters. GP hyperparameters are learned during the training process. The negative log marginal likelihood $\mathcal{L}(\boldsymbol{\theta})$ is minimized with respect to training data points $(X_n, \mathbf{y}_n)$ to optimize the hyperparameters.

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\left[\mathbf{y}^T K_y^{-1} \mathbf{y} + \log|K_y| + n \log(2\pi)\right] \tag{4}$$

Where $K_y = k(X_n, X_n) + \sigma^2 I_n$. Post training, predictive distribution of observations for test points X_t is given as

$$\mathbf{f}^* \sim \mathcal{N}(K^{*T} K_y^{-1} \mathbf{y}, K^{**} - K^{*T} K_y^{-1} K^*) \tag{5}$$

where K^* is $n \times t$ covariance matrix between train and test points and K^{**} is $t \times t$ covariance matrix among test points.

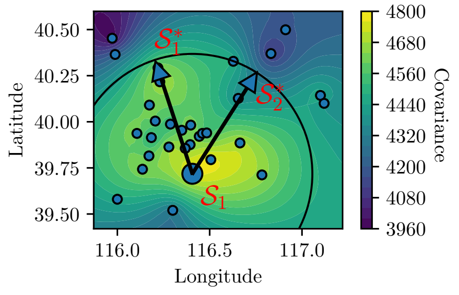

Figure 1: Interpolated empirical covariance between a target station ($\mathcal{S}_1$) and all other stations in Beijing dataset. Two locations ($\mathcal{S}_1^*$ and $\mathcal{S}_2^*$) with the same distance away from $\mathcal{S}_1$ have significantly different covariance. Thus, stationary kernels may fail to capture such a phenomenon. We may need a non-stationary kernel for such scenarios.

Non-Stationary GPs (NSGP)

Environmental processes, in general, are highly dynamic in nature depending on other environmental variables, the geography of a place and various other factors. For example, temperature differences at a small distance apart would be much higher in the hilly area than in a plain area.

We visualize the empirical covariance between a target AQ station ($\mathcal{S}_1$) and all other stations from Beijing dataset in Figure 1 to investigate the need for non-stationarity in our model. Empirical covariance was generated by computing covariance $\text{cov}(\mathcal{S}_1, \mathcal{S}_i), 1 < i < |S|$ from corresponding PM$_{2.5}$ values along the temporal dimension. We interpolated these covariance values with kriging (spherical variogram) to generate a fine-grained map. If we focus on a target station $\mathcal{S}_1$ and two locations in space $\mathcal{S}_1^*$ and $\mathcal{S}_2^*$, notice that $\text{cov}(\mathcal{S}_1, \mathcal{S}_1^*)$ is significantly different from $\text{cov}(\mathcal{S}_1, \mathcal{S}_2^*)$ despite $\mathcal{S}_1^*$ and $\mathcal{S}_2^*$ are same distance apart from $\mathcal{S}_1$.

Thus, the underlying covariance structure is hard to model efficiently with stationary kernels of the form $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{0}, \mathbf{x}-\mathbf{x}')$. Or, the covariance between two inputs depends only on the distance between the two inputs and not on their absolute values. RBF kernel in Eq. 3 is a stationary kernel. To model a complex phenomena, we may need a non-stationary kernel where the covariance evaluated at $\mathbf{x}$ and $\mathbf{x}'$ given as: $k(\mathbf{x}, \mathbf{x}')$ may be unequal to $k(\mathbf{x}+\Delta\mathbf{x}, \mathbf{x}'+\Delta\mathbf{x})$.

There are multiple ways of inducing non-stationarity in GPs. In this paper, we study non-stationarity based on length scales. We first quickly study the effect of length scales on model fit (functions generated from a GP).

Length scale parameter in most kernels governs the smoothness of functions drawn from the distribution. Figure 2 shows the effect of length scale on functions drawn from a GP prior. Large length scales imply more smoothness in functions. In reality, a function may have variable smoothness as shown by the curve corresponding to $\text{LS}(x) = 0.2x$. In such cases, it might be better to model the length scale as a function of input. Now, we formally introduce a non-stationary kernel known as Gibbs kernel.

Stationary kernels can be transformed to a non-stationary

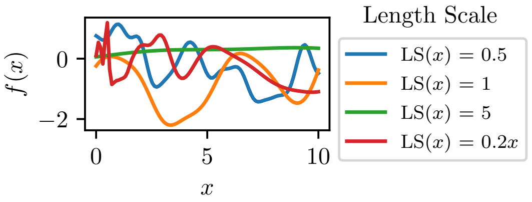

Figure 2: A larger length scale allows more smoothness in a drawn function. We also show a sample function corresponding to variable length scale uniformly varying from 0.01 to 2. Stationary GPs may underfit (have high bias) if dataset has such a distribution.

kernel with Gibbs kernel (Paciorek and Schervish 2003):

$$k_{NS}(x, x') = \sqrt{\frac{2\ell\ell'}{\ell^2 + \ell'^2}} k_S(x, x') \tag{6}$$

Where k_{NS} is a non-stationary kernel built on a stationary kernel k_S. Note that for a stationary kernel in Eq. 3, we had a single length scale parameter ℓ for all inputs. In k_{NS}, we have a separate ℓ associated with each input x. While plugging a stationary kernel k_S in Eq. 6, we need to replace ℓ^2 with $(\ell^2 + \ell'^2)/2$. This formulation is only possible if we can compute ℓ at any arbitrary data point x. Assuming smooth variation in length scales, we can use another GP to model this smoothness. However, learning length scales at all the input points may become computationally expensive.

Thus, (Plagemann, Kersting, and Burgard 2008) propose to learn $\overline{\boldsymbol{\ell}}_m$ at a set of inducing points $\overline{\mathbf{x}}_m^T, m << n$ and compute ℓ at any input location x with posterior distribution $p\left(\boldsymbol{\ell}|\mathbf{X}, \overline{\boldsymbol{\ell}}, \overline{\mathbf{x}}_m^T\right)$. The second level GP (length scale GP) for the length scale process is formulated as:

$$f_\ell(x) \sim \mathcal{GP}_\ell(m_\ell(x), k_\ell(x, x')) \tag{7}$$
$$\ell = f(x) + \epsilon_\ell, \;\; \epsilon_\ell \sim \mathcal{N}(\mathbf{0}, \overline{\sigma}^2) \tag{8}$$

where, prior $m_l(x)$ is constant 0 function and $k_l(x, x')$ can be any suitable kernel for the length scale modeling.

The hyperparameters of both GPs can be tuned by maximizing log of a-posteriority probability $\log p(\boldsymbol{\ell}|X, \mathbf{y})$ of latent length scales. In interest of space, we provide the equations of objective function (Plagemann, Kersting, and Burgard 2008) while leaving the derivations to the appendix.

$$\log p(\boldsymbol{\ell}|\overline{\mathbf{x}}^T, \overline{\boldsymbol{\ell}}) = -\frac{1}{2}\left[\log|\overline{K}^{**} - \overline{K}^{*T}\overline{K}_\ell^{-1}\overline{K}^*| + c\right] \tag{9}$$

$$\log p(\mathbf{y}|X, \boldsymbol{\ell}) = \frac{-1}{2}\left[\mathbf{y}^T K_y^{-1}\mathbf{y} + \log|K_y| + c\right] \tag{10}$$

$$\log p(\boldsymbol{\ell}|X, \mathbf{y}) = \log p(\mathbf{y}|X, \boldsymbol{\ell}) + \sum_{j=1}^{d} \log p(\boldsymbol{\ell}_j|\overline{\mathbf{x}}_j^T, \overline{\boldsymbol{\ell}}_j) \tag{11}$$

Where, $\overline{K}_\ell = \overline{K}_{f_\ell} + \overline{\sigma}^2 I_m$ is $m \times m$ noisy covariance matrix from $\mathcal{GP}_\ell$ computed among $\overline{\mathbf{x}}^T$. Similarly, $\overline{K}^*$ is $m \times n$ covariance matrix between inducing points and train points and $\overline{K}^{**}$ is $n \times n$ covariance matrix among train points. Eq. 6 to Eq. 9 are defined for one-dimensional data points because we have a separate $\mathcal{GP}_\ell$ for each feature. Finally, for multi-dimensional data with d features, we can optimize combined hyperparameters with Eq. 11. We use K-Means clustering separately on j^{th} feature with m_j number of clusters and choose the cluster centers as inducing points.

ARD (Automatic Relevance Determination)

AQ datasets contain numerous types of features which may require a separate treatment based on their range. GPs allow learning separate length scales for individual features with ARD functionality (automatic relevance determination). ARD is helpful in automatically choosing the useful features and ignoring non-informative features by setting the corresponding length scale values too high or too low during the optimization (Rasmussen and Williams 2005).

Hamming Distance Based Kernel for Categorical Features

We have a set of categorical features present in our dataset (wind direction and weather). Such features are often transformed by one-hot-encoding before training machine learning models. For GPs, widely used kernels such as RBF are not directly applicable to one-hot-encoded features due to the binary nature of features. Because RBF and similar kernels are designed to encode distance-based smoothness, but we do not have a continuous feature space to make RBF kernel effective for categorical features. Thus, we utilize a Hamming distance-based kernel (Hutter et al. 2014), which returns maximum correlation when categories are the same and returns a lower correlation regulated by length scale l.

$$k_{cat}(x, x') = \exp\left(-\frac{\mathbb{I}_{x \neq x'}}{2\ell^2}\right) \tag{12}$$

This kernel avoids high dimensional feature space generated by one-hot-encoding, which may significantly affect the computational resources required during the training.

Local Periodic Kernel for Temporal Feature

Most environmental phenomena are periodic in the temporal dimension. Air pollution also exhibits periodic behavior due to the nature of its sources and predictors. For example, traffic has diurnal patterns, and meteorological variables have daily and seasonal patterns. However, an exact periodicity may not be exist in practice, but it may vary smoothly in the space. This phenomenon is known as local periodicity (Duvenaud 2014). In GPs, the multiplication of two kernels incorporates the abilities of both kernels. Thus, we use the RBF kernel to incorporate smoothness in the period and the Periodic kernel to model the period in the dataset. Multiplication of these kernels is known as Local Periodic kernel

$$k_{Periodic}(x, x') = \exp\left(-\frac{2\sin^2(\pi|x - x'|/\sigma_p)}{\ell^2}\right) \tag{13}$$

$$k_{LocalPer}(x, x') = k_{Periodic}(x, x') \circ k_{RBF}(x, x') \tag{14}$$

Where, σ_p is the period learned by the periodic kernel.

Final Model

The final kernel is of the following form:

$$K = \sigma_f^2 \circ K_{Matern/RBF} \circ K_{cat} \circ K_{LocalPer} \tag{15}$$

Where, $K_{Matern/RBF}, K_{cat}$ and $K_{LocalPer}$ are kernels computed with continuous, categorical, and temporal features, respectively. We now show the process of training and inference with our model.

Initialization Given the training data $\langle X, \mathbf{y} \rangle$, fix the inducing points $\overline{\mathbf{x}}_j^T$ for j^{th} feature with any suitable heuristics. We use K-Means clustering separately on j^{th} feature with m_j number of clusters and choose the cluster centers as inducing points. Initialize $\overline{\boldsymbol{\ell}}_j$ for each $j \in \{1, 2, ..., d\}$. Initialize kernel hyperparameters for $\mathcal{GP}_\ell$ and $\mathcal{GP}_y$.

Training For each iteration of training, we would execute the following steps: 1) Evaluate $\boldsymbol{\ell}_j$ at $\mathbf{x}_j^T$ as predictive mean of corresponding $\mathcal{GP}_\ell$ using $\overline{\mathbf{x}}_j^T$ for each $j \in \{1, 2, ..., d\}$; 2) Calculate loss (Eq. 11) after evaluating required covariance matrices from $\mathcal{GP}_\ell$ and $\mathcal{GP}_y$; 3) Jointly update $\mathcal{GP}_\ell$, $\mathcal{GP}_y$ kernel hyperparameters and $\overline{\boldsymbol{\ell}}_j$ for each $j \in \{1, 2, ..., d\}$.

Prediction/Inference Evaluate $\boldsymbol{\ell}_j^*$ at $\mathbf{x}^{*T}{}_j$ as predictive mean of corresponding $\mathcal{GP}_\ell$ using $\overline{\mathbf{x}}_j^T$ for each $j \in \{1, 2, ..., d\}$ and use standard GP posterior equations to get the posterior distribution of PM$_{2.5}$ values at X^*.

Scalable Batch Training

GP training involves evaluating marginal likelihood in each iteration which takes $O(tn^3)$ time for n data points and t iterations. Additionally, we need $O(n^2)$ memory to store the covariance matrix. We have nearly 14K data points from just a month's data, and thus we need scalability. Recently, stochastic gradient descent on GPs has been proven as an effective method for large datasets (Chen et al. 2020). Thus, we use batch-wise training to train non-stationary versions of our model on multiple batches of data. We were able to run the stationary version of the model without batched setting, and thus we used full data for it. There are multiple ways of sampling batches from the entire dataset. We explore the following batching schemes in our approach: 1) We sample the data points uniformly from the full data; 2) We sample a random data point and choose b points close to that point as a batch. We use Euclidean distance to determine the close points; 3) We equally split our data along the temporal dimension. In our case, we split one month's data into four parts, considering a week's data as a batch. Note that if we assume constant batch size in this setting, memory requirement stays constant, and compute time increases linearly with an increase in the dataset size.

Evaluation

Datasets

We have used openly available datasets for Beijing, China and London, UK in this work. We now describe the further details and preprocessing for each of the datasets.

Data	Methods			Missing
	Lin.	Quad.	Cub.	data
$PM_{2.5}$	**16.69**	18.29	18.66	3%
Temperature (T)	**1.24**	1.39	1.43	14%
Humidity (H)	**7.57**	9.21	9.49	13%
Wind speed (WS)	**3.32**	4.29	4.50	13%
Weather (W)	Nearest Time-stamp			15%
Wind dir. (WD)	Nearest Time-stamp			14%

Table 1: All stations in Beijing dataset have a few missing meteorological and $PM_{2.5}$ values. To choose the best method for data filling in the temporal dimension, we use five-fold cross-validation with non-missing data. We use the method yielding least RMSE (Linear interpolation) to fill in the missing values. We apply the same methodology for London dataset. Lin. is Linear interpolation, Quad. is Quadratic spline, and Cub. is Cubic spline. Wind dir. is wind direction.

Beijing dataset We use the hourly $PM_{2.5}$ data from 36 stations in Beijing and meteorological data (temperature, humidity, pressure, wind speed, wind direction and weather) from the stations in the same district (Cheng et al. 2018; Zheng et al. 2015). Among these features, wind direction and weather are categorical and others are continuous features. Wind direction contains 10 categories (including 4 cardinal, 4 ordinal directions along with unstable and no direction). Weather has 17 categories, including but not limited to rainy, foggy, sunny and dusty. The duration of the dataset is one year (2014-05-01 to 2015-04-30). The dataset is publicly available via the following website (Microsoft 2021). We note that the source papers of this dataset also use other features to model AQ such as POIs (points of interest) and road networks (total length of roads around a station), which are not publicly available and thus not used in our work. We observe a large amount of missing data in different stations at different time intervals. To enable the comparison with state-of-the-art neural baseline (Cheng et al. 2018), we chose a particular month (March 2015) having the minimum amount of missing data. We carry out a dataset preprocessing step to handle further issues with the dataset, such as missing values and anomalies.

To address the missing entries in the dataset, we remove the stations having a substantial amount of missing values. 50% of stations from the dataset has at least 60% missing values for the pressure feature. Thus, we drop pressure from our meteorological variables. Also, five stations (station IDs: 1009, 1013, 1015, 1020, 1021) each have merely 35% of the weather data, so we drop these five stations from our experiments. In the remaining data, we have at least 85% data available for all variables as shown in Table 1. To fill in the missing data for real-valued variables ($PM_{2.5}$, temp, humidity, and wind speed), we *interpolate in time*. We choose the best method among these with cross-validation on non-missing data. In previous literature, we either do not find missing data handling details or find trivial methods such as nearest timestamp filling for all variables (Cheng et al. 2018). Thus, our methodology provides a data-driven, systematic approach to data filling. Table 1 shows the least RMSE values on five-fold cross-validation for each of the interpolation methods applied on each feature. We choose the nearest timestamp value to fill the missing data coming from categorical features (wind direction and weather).

London dataset We use the London AQ dataset provided in KDD Cup 2018 challenge. The dataset includes $PM_{2.5}$ observations from 24 stations along with spatial locations of the stations and grid-wise (0.1×0.1) meteorological data. We take a distance-based weighted average of the nearest four grid points to merge meteorological data with AQ data. We choose a month (May 2017) with the least amount of missing entries in AQ data and drop 2 ('HR1', 'KF1') out of 24 stations having most of the missing entries. Our final dataset has 1.5% missing data. We follow similar steps as the Beijing dataset to fill in the missing values.

Baselines

We compare our work against the state-of-the-art neural attention architecture (Cheng et al. 2018) and baselines used in previous literature. Note that we baselined against other regression methods like: support vector regression, linear regression, and decision trees but do not include them here as the results were comparable or poorer than the mentioned baselines and due to space limits.

Random Forest Random Forest is widely used and known to perform efficiently on the non-linear regression tasks (Fawagreh, Gaber, and Elyan 2014). It uses an ensemble of multiple decision trees for regression such that the final output is the mean of the outputs from all trees.

Inverse Distance Weighting Inverse Data Weighting (IDW) (Lu, George Y., and David W. Wong 2008) is an interpolation technique that estimates the value of an unknown point by taking the weighted average of the known points. It is a commonly used method in spatial interpolation literature (Ikechukwu et al. 2017).

XGBoost XGBoost improves by iteratively combining the results from weak estimators. The algorithm uses gradient descent while adding new trees while training.

K-Nearest Neighbors (KNN) Regression KNN uses "feature similarity" to obtain the K nearest neighbors to a test point and averages their observations to estimate the value at the test point.

ADAIN We use the ADAIN model (Cheng et al. 2018) which is a neural network based approach to infer the AQ at a a local station using the data from available stations. The model uses both time-invariant and time-series data in linear and recurrent neural network layers. The importance of train stations in determining the AQ of a test location is dynamically computed by an attention layer. The final prediction is the weighted average of observations from the train stations where weights are attention weights. We would like to clarify that while the code for the ADAIN paper is not publicly available, we implemented ADAIN by discussing it with the original authors.

We use RF, XGBoost, and KNN implementation from the scikit-learn (Pedregosa et al. 2011) library and IDW from the Polire (Narayanan et al. 2020) library.

Evaluation Metrics

We use the root mean squared error (RMSE), mean abslute error (MAE) and R^2 as widely-used metrics in regression problems. We also use the following metrics specifically for the GP models: i) **CE:** Coverage Error (CE) is a useful metric for evaluating probabilistic models (Hatalis et al. 2017). CE is calculated as the absolute difference between x and the number of samples falling within the $x\%$ confidence interval. We report CE for 95% confidence interval; ii) **MSLL:** Mean Standardized Log Loss (MSLL) is an average of log pdf over all the test points considering the predictive distribution (Rasmussen and Williams 2005).

Experimental Setup

We now discuss the settings and data preparation done for each method.

Cross-validation Our experimental setup is similar to previous literature (Cheng et al. 2018). We consider an offline learning setting where we train a single model on the whole dataset leveraging the meteorological features and observations from the train stations. IDW and KNN are exceptions here as they can handle spatial features only, and thus we train and test separate models for each timestamp. We perform the 3-fold and 4-fold outer cross-validation for the Beijing dataset and London dataset, respectively, where each fold is split by a set of train and test AQ stations.

Hyperparameter tuning For hyperparameter tuning on non-GP based baselines, we carry out grid search for 5 inner folds on the training data. We use the 'GridSearchCV' routine from scikit-learn (Pedregosa et al. 2011) to perform the hyperparameter tuning. For Random Forest, we varied *n-estimators* in the set {100, 500, 1000} and the *max-depth* in {10, 50, 100, infinity}. The value of *exponent* in IDW was varied for values $\in \{0.5, 1, 2, 3, 4, 5, 6\}$ in order to get the best fit. For XGBoost, *n-estimators* was chosen among the set{100, 500, 1000} and the *learning-rate* from {0.01, 0.05, 0.1}. The grid of *n-neighbors* in KNN contained the values {2, 3, 5, 7}. The final values of hyperparameters after tuning are mentioned in Section .

Data preperation For distance-based models (IDW and KNN), each feature dimension is scaled between 0 to 1. For the ADAIN model, we follow the data preparation as described by (Cheng et al. 2018). The input data to ADAIN is scaled using robust scaling (dataset is scaled so that the inter-quartile range is scaled between 0 to 1 to reduce the effect of outliers on the scaling) on all continuous features.

We perform stationary GP regression over each dataset fold, experimenting with different kernel functions. We choose the kernel that yields the best training loss (in GP, best log marginal likelihood). Note that log marginal likelihood can be used as a model selection strategy here, as shown by (Fong and Holmes 2020). Due to numerical issues and non-convexity of log marginal likelihood, hyperparameter initialization plays a significant role in GP regression, as pointed out by (Basak et al. 2021). To combat this, we use 5 random restarts of initializing hyperparameters by the standard normal distribution, allowing the model to converge at the global optima potentially. We use BoTorch (Balandat et al. 2020) implementation of stationary GPs.

Model	RMS	MAE	R^2	CE (95%)	MSLL
RF	26.68	15.59	0.87	-	-
IDW	48.09	34.79	0.87	-	-
XGB	33.85	23.94	0.87	-	-
KNN	37.99	24.80	0.87	-	-
ADAIN	29.39	18.83	0.56	-	-
$\bar{\mathbf{A}}\bar{\mathbf{N}}\bar{\mathbf{C}}\bar{\mathbf{L}}$	37.95	26.66	0.74	0.73	49.04
$\mathbf{A}\bar{\mathbf{N}}\bar{\mathbf{C}}\bar{\mathbf{L}}$	25.04	15.81	0.89	0.58	29.40
$\mathbf{A}\bar{\mathbf{N}}\mathbf{C}\bar{\mathbf{L}}$	*24.02*	*15.03*	*0.90*	0.57	27.87
$\mathbf{A}\bar{\mathbf{N}}\mathbf{C}\mathbf{L}$	**24.28**	**15.23**	**0.89**	**0.55**	**24.68**
$\mathbf{A}\mathbf{N}\mathbf{C}\bar{\mathbf{L}}$	26.07	18.97	0.85	0.37	9.97
$\mathbf{A}\mathbf{N}\mathbf{C}\mathbf{L}$	24.71	15.77	0.89	*0.34*	*8.07*

Table 2: Comparison of metrics among all baselines and our approach (combinations of various settings) on Beijing dataset. Our approach outperforms all the baselines, including a state-of-the-art neural attention method (ADAIN). $\mathbf{A}, \mathbf{N}, \mathbf{C}, \mathbf{L}$ configuration is explained in Sectin . The bold row shows the best approach based on minimum negative log marginal likelihood. The italic row shows the approach performing best on the test dataset. We can observe that these two versions have comparable results on the test data. According to probabilistic metrics (CE and MSLL), $\mathbf{ANCL}$ is yielding the best results among all other variants.

Our Model configuration $\mathbf{A}$: ARD is enabled, $\mathbf{N}$: non-stationary kernel is used, $\mathbf{C}$: categorical kernel is used for categorical features and $\mathbf{L}$: Locally periodic kernel is used for time feature, $\bar{\mathbf{A}}$: Non-ARD version, $\bar{\mathbf{N}}$: Stationary kernel, $\bar{\mathbf{C}}$: one-hot-encoded categorical features without Hamming distance kernel, $\bar{\mathbf{L}}$: RBF/Matern kernel for time feature. We choose the best model with minimum negative log marginal likelihood (NLML).

Results and Analysis

Our main results in Table 2 and Table 3 show that our approach under the $\mathbf{A}\bar{\mathbf{N}}\mathbf{CL}$ configuration (shown in bold numbers) significantly improves over all the other baselines in terms of conventional metrics (RMSE and MAE). We selected the best configuration based on the lowest value of negative log marginal likelihood (NLML) across all the GP based methods. The best performing approach configuration on the test set (shown in italics) is the $\mathbf{A}\bar{\mathbf{N}}\mathbf{C}\bar{\mathbf{L}}$ whose RMSE is comparable to the best configuration chosen as per the (NLML). We note that as per the probabilistic metrics (CE and MSLL), our approach with all extensions turned on ($\mathbf{ANCL}$) is performing the best on both Beijing and London datasets. Best hyperparameters for baselines are provided in the appendix available in our repository.

Model	RMS	MAE	R^2	CE (95%)	MSLL
RF	4.69	3.06	0.87	-	-
IDW	8.01	5.50	0.87	-	-
XGB	4.90	3.44	0.87	-	-
KNN	4.75	3.20	0.87	-	-
ADAIN	4.78	3.36	0.56	-	-
$\mathbf{\bar{A}\bar{N}\bar{C}\bar{L}}$	4.86	3.33	0.66	0.06	3.15
$\mathbf{A\bar{N}\bar{C}\bar{L}}$	*4.64*	*3.19*	*0.69*	0.06	3.10
$\mathbf{A\bar{N}\bar{C}L}$	**4.65**	**3.20**	**0.68**	**0.07**	**3.12**
$\mathbf{ANC\bar{L}}$	5.51	3.85	0.56	*0.01*	3.07
ANCL	4.90	3.23	0.65	0.02	*2.95*

Table 3: Comparison of metrics among all baselines and our approach on London dataset. Our approach outperforms all the baselines. **ANCL** approach outperforms all other variants according to the probabilistic metrics (CE and MSLL)

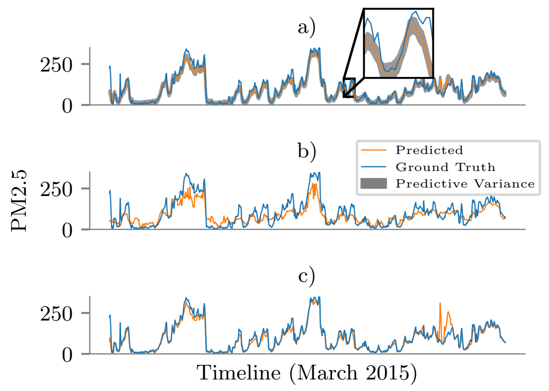

Figure 3: Predicted PM2.5 Concentration comparison between a) our approach ($\mathbf{A\bar{N}CL}$), b) XGBoost, and c) Random Forest model for a particular test station in Beijing data.

Figure 3 shows the comparison of predictions from our $\mathbf{A\bar{N}CL}$ with the XGBoost and the Random Forest baselines on the Beijing dataset. Our model can capture the nuances of the dataset better than the baselines.

To interpret the ARD enabled model, we plot Non-ARD length scale with ARD length scales for each feature in Figure 4 with best model: $\mathbf{A\bar{N}CL}$. The magnitude of the length scale gives insights into the smoothness of observations in a particular feature space. For example, $PM_{2.5}$ varies slowly with wind speed but varies rapidly with latitude. If these different relationships did not exist, we would have learned the same length scale across all the features.

In the training of Non-stationary GP, we explored multiple batching techniques on the Beijing dataset, as discussed in Section . As per Table 4, Time-split batching yields the best train loss and best test RMSE in our experiments. Note that we have used the Time-split method as default in the experiments presented in Table 2. Note that we have used other batching sampling, and the results are comparable.

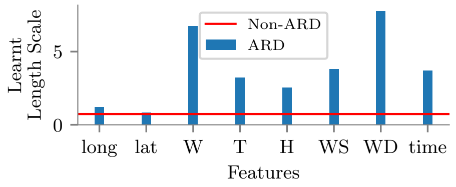

Figure 4: Effect of ARD (automatic relevance determination) on length scales learned for different features. This reveals key information about the relationship between a feature and $PM_{2.5}$. For example, $PM_{2.5}$ is smoothly varying with wind speed but varying quickly with latitude.

	RMSE (Lower is better)			
Method	Fold-1	Fold-2	Fold-3	Mean
Uniform	27.63	**26.97**	25.67	26.76
Time split	25.35	27.83	**25.47**	**26.22**
Nearest Neigh.	**24.48**	28.11	27.48	26.69

Table 4: Effect of batching techniques on RMSE while training the non-stationary GPs on Beijing dataset. We observe that while each method performs well in different folds, the Time-split method has the overall best result.

Limitations and Future Work

We now discuss limitations and future work:

1. **Larger data:** We have only looked at a single month of data for a single pollutant. This was done because some of the baselines require large contiguous data chunks (which are unavailable and inappropriate to fill like we did in the current work). We plan to expand our dataset both in time and the estimated pollutants in the future.
2. **Other non-stationary methods:** In the current paper, we looked only at a single method (length-scale based) for inducing non-stationary behaviour. In the future, we plan to look at other similar techniques (Heinonen et al. 2016; Wilson et al. 2016).
3. **GP Scalability:** In the current paper, we have looked at SGD based methods for scaling GPs. In the future, we propose to also study other sparse GP methods (Snelson and Ghahramani 2006; Titsias 2009).

Conclusions

Accurate AQ estimation at unmonitored locations is an essential step towards better policy and control of air pollution. Existing approaches for estimating AQ are either white-box and require extensive emission data or entirely data-driven, like neural networks. In this work, we present Gaussian processes based approach that can leverage domain insights such as: i) periodicity in time; ii) the relative importance of different features; iii) non-stationarity; and iv) encoding categorical features. Our approach is more accurate than state of the art. The uncertainty estimates in our approach make it more beneficial to the decision-makers.

Acknowledgments

We would like to thank Wessel Bruinsma, a PhD student from University of Cambridge for his suggestions during the rebuttal. We thank Deepak Narayanan and Apoorv Agnihotri, alumni B.Tech. students of IIT Gandhinagar for providing their insights from the initial experiments related to this work. This work is supported by SERB project SRG/2020/001127 "AI and Sensor Networks for Air-quality Monitoring" and STARS project 1/367.

References

Balakrishnan, K.; Dey, S.; Gupta, T.; Dhaliwal, R. S.; Brauer, M.; Cohen, A. J.; Stanaway, J. D.; Beig, G.; Joshi, T. K.; Aggarwal, A. N.; Sabde, Y.; Sadhu, H.; Frostad, J.; Causey, K.; Godwin, W.; Shukla, D. K.; Kumar, G. A.; Varghese, C. M.; Muraleedharan, P.; Agrawal, A.; Anjana, R. M.; Bhansali, A.; Bhardwaj, D.; Burkart, K.; Cercy, K.; Chakma, J. K.; Chowdhury, S.; Christopher, D. J.; Dutta, E.; Furtado, M.; Ghosh, S.; Ghoshal, A. G.; Glenn, S. D.; Guleria, R.; Gupta, R.; Jeemon, P.; Kant, R.; Kant, S.; Kaur, T.; Koul, P. A.; Krish, V.; Krishna, B.; Larson, S. L.; Madhipatla, K.; Mahesh, P. A.; Mohan, V.; Mukhopadhyay, S.; Mutreja, P.; Naik, N.; Nair, S.; Nguyen, G.; Odell, C. M.; Pandian, J. D.; Prabhakaran, D.; Prabhakaran, P.; Roy, A.; Salvi, S.; Sambandam, S.; Saraf, D.; Sharma, M.; Shrivastava, A.; Singh, V.; Tandon, N.; Thomas, N. J.; Torre, A.; Xavier, D.; Yadav, G.; Singh, S.; Shekhar, C.; Vos, T.; Dandona, R.; Reddy, K. S.; Lim, S. S.; Murray, C. J. L.; Venkatesh, S.; and Dandona, L. 2019. The impact of air pollution on deaths, disease burden, and life expectancy across the states of India: the Global Burden of Disease Study 2017. *The Lancet Planetary Health*, 3(1): e26–e39.

Balandat, M.; Karrer, B.; Jiang, D.; Daulton, S.; Letham, B.; Wilson, A. G.; and Bakshy, E. 2020. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. *Advances in neural information processing systems*, 33.

Basak, S.; Petit, S.; Bect, J.; and Vazquez, E. 2021. Numerical issues in maximum likelihood parameter estimation for Gaussian process interpolation. In Nicosia, G.; Pardalos, P.; and et al., eds., *7th International Conference on machine Learning, Optimization and Data science (LOD 2021)*. Grasmere, United Kingdom.

Chen, H.; Zheng, L.; Al Kontar, R.; and Raskutti, G. 2020. Stochastic Gradient Descent in Correlated Settings: A Study on Gaussian Processes. In *NeurIPS*.

Cheng, W.; Shen, Y.; Zhu, Y.; and Huang, L. 2018. A neural attention model for urban air quality inference: Learning the weights of monitoring stations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Duvenaud, D. 2014. *Automatic model construction with Gaussian processes*. Ph.D. thesis, University of Cambridge.

Fallah-Shorshani, M.; Shekarrizfard, M.; and Hatzopoulou, M. 2017. Integrating a street-canyon model with a regional Gaussian dispersion model for improved characterisation of near-road air pollution. *Atmospheric environment*, 153: 21–31.

Fawagreh, K.; Gaber, M. M.; and Elyan, E. 2014. Random forests: from early developments to recent advancements. *Systems Science & Control Engineering*, 2(1): 602–609.

Fong, E.; and Holmes, C. C. 2020. On the marginal likelihood and cross-validation. *Biometrika*, 107(2): 489–496.

Gardner, J. R.; Pleiss, G.; Bindel, D.; Weinberger, K. Q.; and Wilson, A. G. 2018. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *arXiv preprint arXiv:1809.11165*.

GPy. since 2012. GPy: A Gaussian process framework in python. http://github.com/SheffieldML/GPy.

Guizilini, V.; and Ramos, F. 2015. A Nonparametric Online Model for Air Quality Prediction. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, 651–657. AAAI Press. ISBN 0262511290.

Hatalis, K.; Lamadrid, A. J.; Scheinberg, K.; and Kishore, S. 2017. Smooth pinball neural network for probabilistic forecasting of wind power. *arXiv preprint arXiv:1710.01720*.

Heinonen, M.; Mannerström, H.; Rousu, J.; Kaski, S.; and Lähdesmäki, H. 2016. Non-stationary gaussian process regression with hamiltonian monte carlo. In *Artificial Intelligence and Statistics*, 732–740. PMLR.

Hutter, F.; Xu, L.; Hoos, H. H.; and Leyton-Brown, K. 2014. Algorithm runtime prediction: Methods & evaluation. *Artificial Intelligence*, 206: 79–111.

Ikechukwu, M. N.; Ebinne, E.; Idorenyin, U.; Raphael, N. I.; et al. 2017. Accuracy assessment and comparative analysis of IDW, spline and kriging in spatial interpolation of landform (topography): an experimental study. *Journal of Geographic Information System*, 9(03): 354.

Kloog, I.; Ridgway, B.; Koutrakis, P.; Coull, B. A.; and Schwartz, J. D. 2013. Long-and short-term exposure to PM2. 5 and mortality: using novel exposure models. *Epidemiology (Cambridge, Mass.)*, 24(4): 555.

Krige, D. G. 1951. *A statistical approach to some mine valuation and allied problems on the Witwatersrand: By DG Krige*. Ph.D. thesis, University of the Witwatersrand.

Lu, George Y., and David W. Wong. 2008. An adaptive inverse-distance weighting spatial interpolation technique. *Computers & geosciences - Elsevier*, 34.9(1044-1055).

Luo, Z.; Huang, J.; Hu, K.; Li, X.; and Zhang, P. 2019. AccuAir: Winning solution to air quality prediction for KDD Cup 2018. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1842–1850.

Microsoft. 2021. https://www.microsoft.com/en-us/research/publication/forecasting-fine-grained-air-quality-based-on-big-data/. Accessed: 12/15/2021.

Narayanan, S. D.; Patel, Z. B.; Agnihotri, A.; and Batra, N. 2020. A Toolkit for Spatial Interpolation and Sensor Placement: Poster Abstract. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, SenSys '20, 653–654. New York, NY, USA: Association for Computing Machinery. ISBN 9781450375900.

Paciorek, C. J.; and Schervish, M. J. 2003. Nonstationary Covariance Functions for Gaussian Process Regression. In *NIPS*, 273–280. Citeseer.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Plagemann, C.; Kersting, K.; and Burgard, W. 2008. Nonstationary Gaussian process regression using point estimates of local smoothness. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 204–219. Springer.

Rasmussen, C. E.; and Williams, C. K. I. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press. ISBN 026218253X.

Sampson, P. D.; and Guttorp, P. 1992. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 87(417): 108–119.

Snelson, E.; and Ghahramani, Z. 2006. Sparse Gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18: 1257.

Titsias, M. 2009. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, 567–574. PMLR.

WHO. 2021. https://www.who.int/health-topics/air-pollution#tab=tab_1. Accessed: 12/15/2021.

Wilson, A. G.; Hu, Z.; Salakhutdinov, R.; and Xing, E. P. 2016. Deep kernel learning. In *Artificial intelligence and statistics*, 370–378. PMLR.

Xu, Y.; and Zhu, Y. 2016. When remote sensing data meet ubiquitous urban data: Fine-grained air quality inference. In *2016 IEEE International Conference on Big Data (Big Data)*, 1252–1261. IEEE.

Yi, X.; Zhang, J.; Wang, Z.; Li, T.; and Zheng, Y. 2018. Deep distributed fusion network for air quality prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 965–973.

Zheng, Y.; Liu, F.; and Hsieh, H.-P. 2013. U-air: When urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1436–1444.

Zheng, Y.; Yi, X.; Li, M.; Li, R.; Shan, Z.; Chang, E.; and Li, T. 2015. Forecasting Fine-Grained Air Quality Based on Big Data. In *Proceedings of the 21th SIGKDD conference on Knowledge Discovery and Data Mining*. KDD 2015.