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

Zeel Patel, Palak Purohit*, Harsh Patel*, Shivam Sahni*, Nipun Batra

Indian Institute of Technology Gandhinagar, Gujarat, India
{zeel.patel, palak.purohit, harsh.patel, shivam.sahni, nipun.batra}@iitgn.ac.in

Abstract

Air pollution is a global problem and has a severe impact on 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 as well as a state-of-the-art neural attention-based approach.

1 Introduction

Today, 91% of the global population lives under unsafe levels of air quality[1]. Long-term exposure to $PM_{2.5}$ increases cardiopulmonary mortality by 6–13% per 10 $\mu g/m^3$ 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 an AQ map rely on interpolation approaches such as Kriging, Trend, Spline, and KNN (K-nearest neighbors). Some recent approaches have taken an entirely 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.

Gaussian processes (GPs) are non-parametric Bayesian models often used in environment modeling (Sampson and Guttorp 1992). Kriging (Krige 1951) is a well-known variant of GPs in spatial modeling. 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 can 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) feature, 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 air quality, 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 with standard kernels because standard kernels are designed to encode distance-based smoothness in the resultant function. In contrast, categorical features do not have a continuous space for distance calculation. 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 have periodicity in the temporal dimension. 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 air quality. To encode this information, we utilize periodic kernels in our model in combination with other kernels.

We use the hourly air quality and meteorological data from Beijing (Cheng et al. 2018) over a month (March-2015) to evaluate our approaches. We compare our ap-

*These authors contributed equally.

[1]https://www.who.int/health-topics/air-pollution#tab=tab_2

proach against: i) conventional spatial interpolation baselines such as IDW (Inverse Distance Weighting), KNN (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 (Cheng et al. 2018). Our approach outperforms the baselines giving root mean square error (RMSE) of 24.52 compared to the best baseline (Random Forest), giving RMSE of 26.67 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 air quality inference accurate and help policy-makers make informed decisions to reduce air pollution.

Reproducibility Our work is fully reproducible, and the code, data, and experiments are available at https://github.com/patel-zeel/AAAI22.

2 Related Work

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

2.1 Dispersion models

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

2.2 Air quality forecasting

Air quality forecasting is a problem where a model predicts air quality t time-stamps in future, leveraging all available information till the current time. It is a well-explored problem in recent times 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 to achieve accurate estimates. Recent work (Yi et al. 2018) proposed a deep distributed fusion network-based method to forecast air quality on a large scale (300+ Chinese cities). Earlier work (Zheng et al. 2015) uses a combination of linear regression for temporal dimension and neural network-based spatial predictor. We have used the dataset for our work from (Zheng et al. 2015). We aim to solve a related but different problem to infer air quality at unmonitored locations at a given time-stamp instead of forecasting air quality in future at the monitored locations.

2.3 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) propose 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 the attention net. Attention net can predict the weights associated with stations dynamically. We use similar features as the authors except road networks and POIs due to the unavailability of data.

3 Problem statement

Given $\mathbf{S}$ air quality monitors, $\mathbf{T}$ time-stamps, $\mathbf{F}$ features (latitude, longitude, temperature, humidity, weather, wind speed and wind direction) and corresponding $PM_{2.5}$ observations, the aim is to predict $PM_{2.5}$ at a new set of locations $\mathbf{S}^*$ for the same $\mathbf{T}$ time-stamps using $\mathbf{F}$ features.

4 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.

In this paper, we propose Gaussian processes (GPs) based AQ models. GPs are Bayesian non-parametric methods. They can provide uncertainty implicitly and can also incorporate domain-specific information via an appropriate kernel design.

We first introduce the basics of GPs, then we discuss a limitation of standard GPs known as stationarity and how can we overcome it with non-stationary GPs (NSGPs). Furthermore, we discuss feature specific kernel design and finally, we elaborate on batch training of GPs for scalability.

4.1 Stationary GPs

GPs assume a prior distribution of functions over the observations.

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

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

where, $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector, $y_i \in \mathbb{R}$ is corresponding observation, and ϵ_i is i.i.d. 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 0 mean function without loss of generality. A well-known covariance function (kernel) is Radial Basis Function (RBF).

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

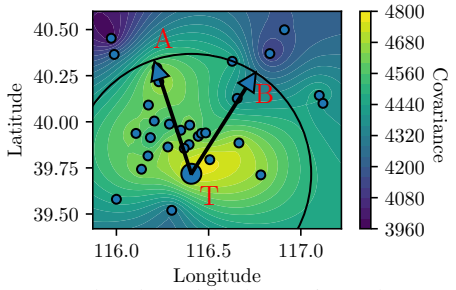

Figure 1: Interpolated empirical covariance between a target station (T) and all other stations. Two locations (A and B) with the same distance away from T have significantly different covariance. Thus, stationary kernels may fail to capture such a phenomenon. We may need a non-stationary kernel for such scenarios.

Where σ_f is kernel variance and ℓ is length scale. In the current setting, $\Theta = \{\sigma_f, \sigma, \ell\}$ are the model hyperparameters. Unlike most machine learning models, hyperparameters are learned during the training of a GP model. The negative log marginal likelihood $\mathcal{L}(\Theta)$ is minimized with respect to n training data points $(X_n, \mathbf{y}_n)$ to optimize the hyperparameters.

$$\mathcal{L}(\Theta) = \frac{1}{2}\left(\mathbf{y}_n^T \tilde{K}_{nn}^{-1}\mathbf{y}_n + \log|\tilde{K}_{nn}| + n\log(2\pi)\right) \quad (4)$$

Where $\tilde{K}_{nn} = K_{nn} + \sigma^2\mathbf{I}_n$ and $K_{nn} = k(X_n, X_n)$. Once the model is trained, predictive distribution of observations for t number of test points X_t is given as

$$f^* \sim \mathcal{N}(m^*, k^*) = \mathcal{N}(K_{tn}\tilde{K}_{nn}^{-1}\mathbf{y}_n, K_{tt} - K_{tn}K_{nn}^{-1}K_{tn}^T)$$

where m^* is predictive mean function and k^* is predictive covariance function and $K_{tn} = k(X_T, X_N)$.

4.2 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 air quality station (T) and all other stations in Figure 1 to investigate the need for non-stationarity in model. Empirical covariance was generated by computing covariance $k_{emp}(\mathbf{T}, s_i), i \in S$ from corresponding $PM_{2.5}$ values along the temporal dimension. We interpolated these values with kriging (spherical variogram) to generate a map. If we focus on the target station $\mathbf{T}$ and two locations in space $\mathbf{A}$ and $\mathbf{B}$, notice that $K(\mathbf{T}, \mathbf{A})$ is significantly different than $K(\mathbf{T}, \mathbf{B})$ despite both points $\mathbf{A}$ and $\mathbf{B}$ being equidistant from the target station $\mathbf{T}$.

Thus, the empirical covariance can not be modelled well using stationary kernels, which have the form $k_S(\mathbf{x}_i, \mathbf{x}_j) = f(||\mathbf{x}_i - \mathbf{x}_j||)$. Or, the covariance between two inputs depends only on the distance between the two inputs (in a Euclidean space) and not on the two inputs themselves. RBF kernel in Eq. 3 is a stationary kernel. To model complex phenomena, we may need a non-stationary kernel of form $k_{NS}(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i, \mathbf{x}_j)$ where the covariance depends on the actual inputs $\mathbf{x}_i$ and $\mathbf{x}_j$. We can also define non-stationarity in a similar way where the covariance evaluated at $\mathbf{x}_i$ and $\mathbf{x}_j$ given as: $k_{NS}(\mathbf{x}_i, \mathbf{x}_j)$ may be unequal to the covariance evaluated between $\mathbf{x}_i + \mathbf{x}$ and $\mathbf{x}_j + \mathbf{x}$ given as: $k_{NS}(\mathbf{x}_i + \mathbf{x}, \mathbf{x}_j + \mathbf{x})$.

There are multiple ways of inducing non-stationarity. 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).

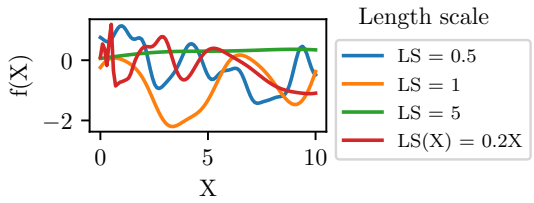

Figure 2: Effect of length scale on randomly drawn functions from a GP prior. A larger length scale allows more smoothness in function. We also show a function with a variable length scale uniformly varying from 0.01 to 2. Stationary GPs will underfit (show high bias) if the dataset has such a distribution.

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 $LS(X) = 0.2X$. In such cases, it might be better to model length scales as a function of input X. Now, we formally introduce the non-stationary covariance structure.

Any stationary kernel can be converted to a non-stationary kernel with the following kernel equation (Paciorek and Schervish 2003)

$$k_{NS}(x_i, x_j) = \sqrt{\frac{2\ell_i\ell_j}{\ell_i^2 + \ell_j^2}}\, k_S(x_i, x_j) \quad (5)$$

Where k_{NS} is the one-dimensional non-stationary kernel based on a one-dimensional stationary kernel k_S. Note that for a stationary kernel in Eq. 3, we had a single length scale parameter ℓ for the whole model. In k_{NS}, we have a separate ℓ_i associated with each data point x_i. While plugging a stationary kernel k_S in Eq. 5, we need to replace ℓ^2 with $(\ell_i^2 + \ell_j^2)/2$. Such a formulation is only possible if we can have a function $f_\ell(x_i) = \ell_i$ to compute ℓ_i for any arbitrary data point x_i. We assume the variation in length scales to be smooth. Thus, we can use another GP to model this smoothness. However, learning length scales at all the input points of size n would become computationally expensive.

Thus, (Plagemann, Kersting, and Burgard 2008) propose to learn ℓ_m for a set of m inducing points $X_m, m << n$. We use K-Means clustering with the number of clusters $\mathbf{K} = m$ on individual features and choose the cluster centers as

latent locations. The second level GP (length scale GP) is formulated as

$$f_\ell(x) \sim \mathcal{GP}_l(m_\ell(x), k_\ell(x, x')) \tag{6}$$

$$\ell_i = f(x_i) + \epsilon_{\ell_i}, \;\; \epsilon_{\ell_i} \sim \mathcal{N}(\mathbf{0}, \bar{\sigma}^2) \tag{7}$$

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 combined hyperparameters for both GPs can be tuned by maximizing log of a-posteriority probability $\log p(\boldsymbol{\ell}_n | X_n, \mathbf{y}_n)$ of latent length scales. In interest of space, as per (Plagemann, Kersting, and Burgard 2008), we provide the equations of objective function without going into further details

$$\log p(\boldsymbol{\ell}_n | X_n, \ell_m) = -\frac{1}{2}\left(\log |{K^*}_{nn}| + n \log 2\pi\right) \tag{8}$$

$$\log p(\mathbf{y}_n | X_n, \boldsymbol{\ell}_n) = -\frac{1}{2}\mathbf{y}_n t \tilde{K}_{nn}^{-1}\mathbf{y}_n - \frac{1}{2}\log |\tilde{K}_{nn}| - \frac{n}{2}\log 2\pi \tag{9}$$

$$\log p(\boldsymbol{\ell}_n | X_n, \mathbf{y}_n) = \log p(\mathbf{y}_n | X_n, \boldsymbol{\ell}_n) + \sum_{j \in d} \log p(\boldsymbol{\ell_j} | \mathbf{x}_j, \boldsymbol{\ell}_j, \mathbf{x}_m) \tag{10}$$

Where, $\tilde{K}_{nn}$ is non-stationary covariance matrix for full data $(X_n, \mathbf{y}_n)$. ${K^*}_{nn}$ is predictive covariance matrix of predicted length scales $\boldsymbol{\ell}_n$. Eq. 5 to Eq. 8 are defined for one-dimensional data points. For practical use in multi-dimensional data with d dimensions, we can model length scales for each feature dimension separately and optimize combined hyperparameters with Eq. 10.

4.3 ARD (Automatic Relevance Determination)

Air quality 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).

4.4 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.

$$k_{cat}(x_{cat,i}, x_{cat,j}) = \sigma_f^2 \exp\left(-\frac{\mathbb{I}_{x_{cat,i} \neq x_{cat,j}}}{2\ell^2}\right) \tag{11}$$

Where $x_{cat,i}$ denotes i^{th} category in a categorical feature. Note that kernel in Eq. 11 also avoids high dimensional feature space generated by one-hot-encoding, which may significantly affect the computational resources required during the training.

4.5 Local periodic kernel for temporal feature

Most environmental phenomena are periodic in the temporal dimension. Air pollution may also exhibit 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 present in practice, but it may vary smoothly across 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

$$k_{Periodic}(x_{t,i}, x_{t,j}) = \sigma_f^2 \exp\left(-\frac{2\sin^2(\pi|x_{t,i} - x_{t,i}|/p)}{\ell^2}\right) \tag{12}$$

$$k_{LocalPer}(\cdot, \cdot) = k_{Periodic}(\cdot, \cdot) \times k_{RBF}(\cdot, \cdot) \tag{13}$$

Where, p is period learned by the periodic kernel.

4.6 Final kernel

Finally, we use RBF or Matern kernels for continuous features (latitude, longitude, humidity, temperature and wind speed), Categorical kernel for categorical features (weather and wind speed) and Local Periodic kernel for the temporal feature. Final kernel is then of the following form:

$$k(\cdot, \cdot) = \sigma_f^2 k_{Matern/RBF}(\cdot, \cdot) k_{cat}(\cdot, \cdot) k_{LocalPer}(\cdot, \cdot) \tag{14}$$

4.7 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 scalability is necessary to successfully apply GPs to our dataset. Recently, stochastic gradient descent on GPs has been proven as an effective method for large scale regression (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 full dataset. We explore the following batching schemes in our approach:

- Uniform sampling: We sample the data points uniformly from the full data.
- Nearest Neighbors: 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.

- Time-split batching: We equally split our data along the temporal dimension. In our case, we split one month's data into 4 parts, considering a week's data as a batch.

Note that, assuming constant batch size in this setting, memory requirement stays constant and compute time increases linearly with an increase in the dataset.

5 Evaluation

5.1 Dataset

Dataset details In this work, 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[2]. 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.

Dataset preprocessing 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 have 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 time-stamp filling for all variables (Cheng et al. 2018). Thus, we believe that 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. To fill the missing data coming from categorical features (wind direction and weather), we choose the nearest timestamp value.

[2]https://bit.ly/38PcrbU

Data	Methods			Missing data
	Lin.	Quad.	Cub.	
$PM_{2.5}$	**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 our dataset have a few missing meteorological and $PM_{2.5}$ values. To choose the best method for filling this data with interpolation in the temporal dimension, we use five-fold cross-validation with non-missing data. Linear interpolation yields the least RMSE for all the variables, and thus we choose it to fill in the missing values. Lin. is Linear interpolation, Quad. is Quadratic spline, and Cub. is Cubic spline. observation filling method. Wind dir. is wind direction.

5.2 Baselines

We compare our work against the recently published state-of-the-art approach based on neural attention architecture (Cheng et al. 2018) and against baselines used in previous literature IDW (Inverse Distance Weighting), Random Forest (RF), XGBoost, KNN and Stationary GP) (Cheng et al. 2018; Zheng, Liu, and Hsieh 2013). We use RF, XGBoost, and KNN implementation from scikit-learn (Pedregosa et al. 2011) library and IDW from the Polire (Narayanan et al. 2020) library. We also note that we baselined against other regression methods like: support vector regression, linear regression, and decision tress, 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 different 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). The weight is inversely proportional to the distance between the point under consideration and the known point. Distance is computed by ℓ_p-norm where *exponent* p can be tuned as a hyperparameter.

XGBoost Extreme Gradient Boosting or XGBoost predicts by combining the results from weaker estimators. The algorithm makes use of 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 air quality at a a local station using the data from available stations. The model uses both time-invariant and time-series data in linear layers and recurrent neural network layers, respectively. 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. The code for the ADAIN paper is not publicly available, we have implemented it from scratch in Tensorflow.

5.3 Evaluation metric

We use the root mean squared error (RMSE) as the evaluation metric for all baselines and our approach. If we have a set of all test stations S and a set of time stamps T under consideration, we calculate the RMSE as:

$$RMSE = \sqrt{\frac{\sum_{s \in S} \sum_{t \in T} \left(y_{(t,s)} - \hat{y}_{(t,s)}\right)^2}{|S||T|}} \tag{15}$$

where, $y_{(t,s)}$ denotes ground truth observation and $\hat{y}_{(t,s)}$ denotes corresponding predicted values for s^{th} station at t^{th} timestamp.

5.4 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 time-stamp. We perform the 3-fold outer cross-validation on all models for a fair comparison, 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 5.5.

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 in a way 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[3] implementation of stationary GPs.

Our Model configuration We introduce the following configuration for our experiments and for the results shown in Table 2. $\mathbf{A}$ - 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).

Model	RMSE (Lower is better) Fold-1	Fold-2	Fold-3	Mean
Random Forest	25.04	29.46	25.53	26.67
IDW	49.11	50.00	45.18	48.09
XGBoost	34.07	34.23	33.25	33.85
KNN	38.09	38.85	37.02	37.98
ADAIN	30.70	32.97	32.19	31.95
$\bar{\mathbf{A}}\bar{\mathbf{N}}\bar{\mathbf{C}}\bar{\mathbf{L}}$	37.25	39.86	36.73	37.95
$\mathbf{A}\bar{\mathbf{N}}\bar{\mathbf{C}}\bar{\mathbf{L}}$	23.63	25.52	25.97	25.04
$\mathbf{A}\bar{\mathbf{N}}\mathbf{C}\bar{\mathbf{L}}$	*22.24*	*24.74*	*25.10*	*24.02*
$\mathbf{A}\bar{\mathbf{N}}\mathbf{C}\mathbf{L}$	**22.60**	**24.96**	**25.27**	**24.28**
$\mathbf{A}\mathbf{N}\mathbf{C}\bar{\mathbf{L}}$	24.09	27.83	26.30	26.07
$\mathbf{A}\mathbf{N}\mathbf{C}\mathbf{L}$	23.48	25.18	25.47	24.71

Table 2: Fold-wise RMSE comparison among all baselines and our approach (combinations of various settings). Our approach outperforms all the baselines, including a state-of-the-art neural attention method (ADAIN). $\mathbf{ANC\bar{L}}$ configuration is explained in Sectin 5.4. 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.

5.5 Results and Analysis

Our main result in Table 2 shows that our approach under the $\mathbf{A\bar{N}CL}$ configuration (shown in bold numbers) signifi-

[3] https://botorch.org/

cantly improves over all the other baselines. 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 $\mathbf{A\bar{N}C\bar{L}}$ whose RMSE is comparable to the best configuration chosen as per the (NLML). We also note that our approach with all extensions turned on ($\mathbf{A\bar{N}CL}$) is only slightly worse than the best performing configuration. We believe this reduction in performance might be due to not enough data for learning the highly sophisticated non-stationary kernel. We plan to study this phenomenon in greater depth in the future work.

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

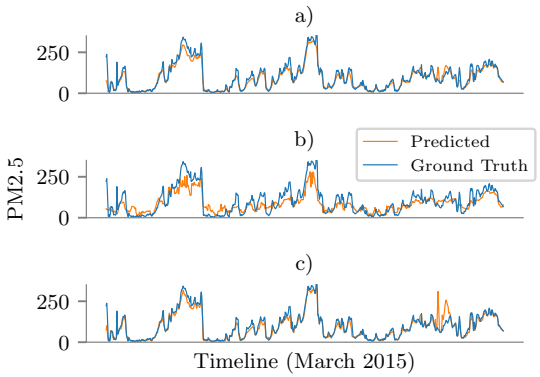

Figure 3: Predicted PM2.5 Concentration comparison between a) our approach ($\mathbf{A\bar{N}CL}$), b) XGBoost, and c) Random Forest model for a particular station data.

To interpret the model after enabling ARD, we plot optimal Non-ARD length scale with ARD length scales for each feature in Figure 4 with best method configuration: $\mathbf{A\bar{N}CL}$. Magnitude of 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 relationship did not exist, we would have learnt same lengthscale across all the features.

In the training of Non-stationary kernels, we attempted different batching techniques as discussed in Section 4.7. According to Table 3, Time-split batching yields the best train loss as well as 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 as well and results are comparable.

The following are the best hyperparameters for the models: a) SVR: kernel='rbf', C=1, epsilon=0.1; b) Random Forest: n_estimators=100, max_depth=None; c) IDW: exponent=3; d) XGBoost: n_estimators=100, learning_rate=0.1; d) KNN: n_neighbors=5; e) ADAIN: batch_size=64, epochs=15

6 Limitations and Future Work

We now discuss limitations and the future work:

Method	RMSE (Lower is better) 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 3: Effect of batching techniques on RMSE while training the non-stationary GPs. We observe that while each method performs well in different folds, the Time-split method has the overall best result (based on train loss). Nearest Neigh. is Nearest Neighbors sampling.

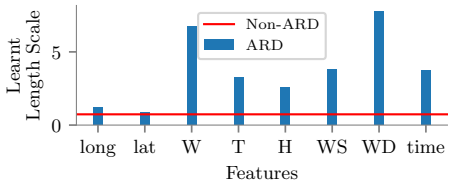

Figure 4: Effect of ARD (automatic relevance determination) on length scales learned for different features. This reveals key information about the relationship between a feature and $PM_{2.5}$. For example, $PM_{2.5}$ is smoothly varying with wind speed but varying quickly with latitude.

1. **Larger data:** In the current paper, we have only looked at a single month of data for a single pollutant. This was primarily 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). In the future, we plan to expand our dataset both in time and the estimated pollutants.
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).

7 Conclusions

Accurate air quality estimation at unmonitored locations is an essential step towards better policy and control of air pollution. Existing approaches for estimating air quality 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.

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.

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.

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.

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.

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.