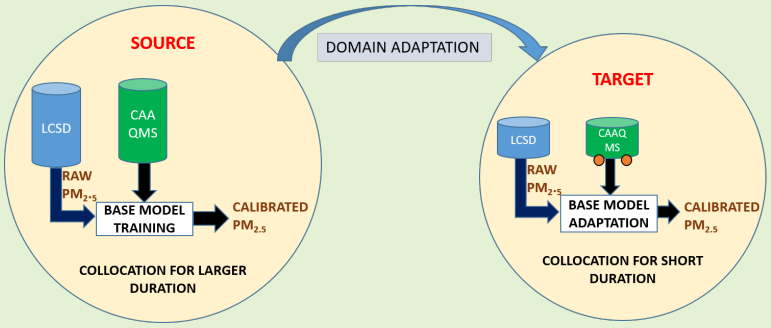
Domain adaptation based deep calibration of low-cost $PM_{2.5}$ sensors

Sonu Kumar Jha, Mohit Kumar, Vipul Arora, Sachchida Nand Tripathi, Vidyanand Motiram Motghare, A. A. Shingare, Karan Singh Rajput, and Sneha Kamble

***Abstract*— Air pollution is a severe problem growing over time. A dense air-quality monitoring network is needed to update the people regarding the air pollution status in cities. A low-cost sensor device (LCSD) based dense air-quality monitoring network is more viable than continuous ambient air quality monitoring stations (CAAQMS). An in-field calibration approach is needed to improve agreements of the LCSDs to CAAQMS. The present work aims to propose a calibration method for $PM_{2.5}$ using domain adaptation technique to reduce the collocation duration of LCSDs and CAAQMS. A novel calibration approach is proposed in this work for the measured $PM_{2.5}$ levels of LCSDs. The dataset used for the experimentation consists of $PM_{2.5}$ values and other parameters (PM_{10}, temperature, and humidity) at hourly duration over a period of three months data. We propose new features, by combining $PM_{2.5}$, PM_{10}, temperature, and humidity, that significantly improved the performance of calibration. Further, the calibration model is adapted to the target location for a new LCSD with a collocation time of two days. The proposed model shows high correlation coefficient values (R^2) and significantly low mean absolute percentage error (MAPE) than that of other baseline models. Thus, the proposed model helps in reducing the collocation time while maintaining high calibration performance.**



***Index Terms*— $PM_{2.5}$, Collocation, Calibration, Domain adaptation, Deep neural network**

I. Introduction

AIR pollution is a global problem that causes seven million deaths every year [1]. It has several adverse effects on human health. It also has a significant contribution to the mortality rate in India [2]. Hence, public awareness of air pollution is an essential requirement that demands regular monitoring of the pollution level. Air pollution from particulate matter (PM) ranks as one of the leading causes of death globally [3], [4]. The continuous ambient air quality monitoring stations (CAAQMS) can provide real-time $PM_{2.5}$ information to people. The establishment of a dense air pollution monitoring network using the CAAQMS is not a feasible option due to their high cost [5]. As a result, the spatial resolution of CAAQMS air quality measurements is insufficient for extensive spatiotemporal mapping [6].

Small and portable, low-cost sensor devices (LCSDs) may improve the capacity to characterize the $PM_{2.5}$ concentrations with high spatial and temporal resolution [7]. The LCSD is a potential technology to meet the PM monitoring network's requirement in densely populated cities [6], [8]. These sensors are capable of capturing spatial variability more effectively [9]. In recent years, various start-ups have emerged, providing compact and affordable wireless $PM_{2.5}$ sensors. However, the measured data using the low cost sensors is less reliable than that of the CAAQMS [10], [11]. Therefore, it is an essential requirement to calibrate the LCSDs against the CAAQMS.

A lot of research has been carried out on the calibration of LCSDs in recent years [12], [13]. Various studies have reported the satisfactory performance of the $PM_{2.5}$ low-cost sensors when compared against Federal Equivalent Methods (FEMs) or research-grade instruments [8], [14], [15]. It is observed that the LCSDs are helpful for the evaluation of short-term changes in the aerosol environment [14]. The authors [8] concluded that appropriate calibration models are an essential requirement for the LCSDs to achieve high accuracy and precision over a wide range in $PM_{2.5}$ concentration. In [16], LCSDs and reference monitors are collocated to monitor the gaseous pollutants and PM. They found that the adequately supported LCSDs with the data modeling tools have shown

Sonu Kumar Jha and Mohit Kumar are with the Centre for Environmental Science and Engineering, Indian Institute of Technology Kanpur, India (e-mail: ksonu@iitk.ac.in;mohitk@iitk.ac.in;)

Vipul Arora is with the Department of Electrical Engineering, (e-mail: vipular@iitk.ac.in)

Sachchida Nand Tripathi is with the Department of Civil Engineering and Centre for Environmental Science and Engineering, (e-mail: snt@iitk.ac.in)

Vidyanand Motiram Motghare, A. A. Shingare, Karan Singh Rajput, Sneha Kamble are with the Maharashtra Pollution Control Board, (e-mail: jdair@mpcb.gov.in; ms@mpcb.gov.in; srohq11@mpcb.gov.in; srohq10@mpcb.gov.in)

Corresponding author: Vipul Arora and Sachchida Nand Tripathi

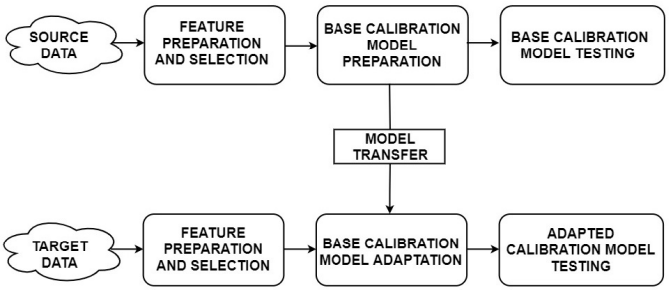


Fig. 1: The steps performed to obtain the proposed calibration model.

a considerable potential to measure the air quality. A system based on linear regression and Gaussian process regressor for the calibration of low-cost $PM_{2.5}$ sensors is proposed in [17]. This method is only effective for the high degree of urban homogeneity in $PM_{2.5}$ [17]. In [8], it is demonstrated that the performance of the quadratic calibration method is found better than that of the simple linear counterpart. The study [5] focused on the inter-comparison of low-cost PM sensors in polluted sites and observed the consistent performance of PM sensors. In [18], a calibration model is developed for various air pollutants such as $PM_{2.5}$ and CO_2 and tested across sites and across devices. This work does not use domain adaptation.

The methods discussed above to calibrate $PM_{2.5}$ values measured using LCSDs performed well at the same site at which they are trained. The deployment of these models at target locations may degrade the performance as the models are not location and device independent. Therefore, in the present work, a calibration method is proposed which is developed at one location (source location s) for a device d and can be adapted to deploy at a target site (s') and a new device (d'). First, the base calibration model M for a d, using machine learning algorithms, is developed at s using large training dataset of two months. After that, the M is adapted at s' for a d' using a lesser amount of training dataset. Hence, the collocation time of LCSDs with CAAQMS can be significantly reduced. To achieve the objective, the domain adaptation-based method is implemented in the present work. To the best of the authors' knowledge, domain adaptation-based calibration of $PM_{2.5}$ LCSDs has not been explored yet. This paper contributes the following facet.

1) The features derived from $PM_{2.5}$, PM_{10}, humidity (RH), temperature (Temp), time, and effective time-lag features are proposed. The use of the derived features with other existing features provides a more robust machine learning-based model for the calibration of $PM_{2.5}$ values measured using LCSDs.
2) The other contribution of the proposed work is that the developed model is adaptable at (s', d') with two days of collocation time.

The paper's remainder is as follows: Section II explains a brief overview of sensor deployment and data acquisition. Section III describes data preprocessing and features extraction, calibration models, domain adaptation, and performance evaluation criteria of the calibration model. Section IV describes the case study. Section V provides the conclusion and future work.

II. Sensors Deployment Sites and Data Collection

In this work, two start-up's A and B have installed the LCSDs to measure the $PM_{2.5}$. The start-up A has installed the LCSDs at Mumbai Airport, Borivali, Kalyan, Mahape, Nerul, Powai, Vileparle, and Worli in Mumbai Metropolitan Region (MMR) Maharashtra, India. These sensors are collocated with the CAAQMS sensors to measure the $PM_{2.5}$ levels. The PM sensors used by start-up A is Plantower PMS-7003, and the Bosch BME-280 sensors are used to measure Temp and RH. Start-up B has installed the LCSDs at Mumbai Airport, Mahape, Nerul, and Vileparle. Start-up B has used Telaire sensor for measuring the PM, and Sensirion sensor for the measurement of Temp, and RH. In this study, the data collected for November 1^{st}, 2020 to January 31^{st}, 2021 is utilized to perform the experiments. The data is available at an interval of 15, 30, and 60 minutes for CAAQMS sensors and 1, 15, 30, and 60 minutes for the LCSDs of start-up A. The start-up B has provided the data at an interval of 30 seconds and 1 minute. All the experiments are performed using the data available at an interval of 60 minutes. The data provided by start-up B is averaged out at 60 minutes.

III. Methodology

This work presents a methodology for the calibration of the $PM_{2.5}$ measured using LCSDs. The overall method is developed in two phases. In the first phase, the M for a d is developed using different machine learning techniques at s. Finally, the M is adapted for d' at s' using the domain adaptation methods with a shorter collocation time. The steps carried out for developing the calibration model are shown in Fig. 1, and described as follows:

A. Data pre-processing and feature preparation

The obtained dataset contains outliers and missing values. Therefore, it needs to be pre-processed before the experimental work. We have considered all outliers data points i.e. Temp $> 50\ ^{\circ}C$ or $< 1\ ^{\circ}C$ and RH $> 100\ \%$ or $< 1\ \%$ as missing values.

The dataset contains features that have different ranges. Hence, normalization of the input features to bring them on to a similar scale is essential. The normalization reduces the training's sensitivity to the inputs' scale and makes the features well-conditioned for optimization. The normalization is performed as follows:

$$N_f = \frac{z - \overline{z}}{\sigma_z} \tag{1}$$

Where N_f and z denote the normalized input features and actual input features, the $\overline{z}$ and σ_z represent the mean and standard deviation of the actual input features. Moreover, the output of the model has only one variable. Hence, the output vector is not normalized in this work.

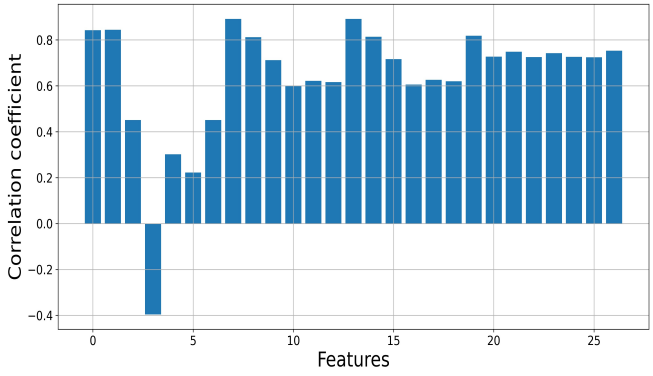


Fig. 2: Correlation of the features w.r.t CAAQMS' $PM_{2.5}$ at Airport site for start-up A.

The performance of the calibration model is improved by selecting useful features. There are many possible combinations of base features. We have empirically selected a few simpler combinations of the base features named as derived features. We get intuition from the study [8]. They have shown that $RH^2/(RH-1)$ is a useful feature to form the calibration model. So, we have considered $RH^2/(RH-1)$ and other derived features to train the calibration model.

In this paper, effective time-lags [19], weather, time, and non-linear features are used for developing the model. The effective values of lags for $PM_{2.5}$ and PM_{10} are found using the cross-correlation coefficients [20]. The other features are also selected using cross-correlation coefficients. The Pearson's correlation of the features w.r.t CAAQMS $PM_{2.5}$ is computed as follows:

$$r = \frac{\sum_{t=1}^{K}(x_t-\overline{x})(y_t-\overline{y})}{\sqrt{\sum_{t=1}^{K}(x_t-\overline{x})^2\sum_{t=1}^{K}(y_t-\overline{y})^2}} \quad (2)$$

Where x_t and y_t denote the value of t^{th} feature and CAAQMS' $PM_{2.5}$, respectively. The $\overline{x}$, and $\overline{y}$ represent the mean value of the features and CAAQMS' $PM_{2.5}$, respectively.

The 27 features that have shown high correlation with the reference $PM_{2.5}$ values are selected that are shown in Table I. Their correlation values are shown in Fig. 2. The indexes at 0, 5, 10, 15, 20, and 25 represent $PM_{2.5}$, $\text{Sin}(\frac{2\pi}{24}hour)$, $PM_{2.5}$ at lag t-23, PM_{10} at lag t-3, $PM_{2.5}$ × RH, and PM_{10} × Temp × RH, respectively. The two features showing the highest correlation coefficient in Fig. 2 at indices 7 and 13 are $PM_{2.5}$ and PM_{10} at lag 1, respectively. Hence, these features have the highest significance in the formation of the calibration model. The correlation coefficient of Temp is negative (index 3 in Fig. 2), this represents that Temp and CAAQMS' $PM_{2.5}$ have inverse relationship. Further details of the features are provided in Table I. The order of features in Table I is the same as in Fig 2.

Further, we have prepared input (features) and output pairs based on effective time-lags, base features, and combination of base features without removing missing values. Thereafter, we have removed all input-output matrix rows if one or more than one missing values are present in any specific row. Finally, the obtained features' matrix is used to train the calibration model.

TABLE I: Features used to train the calibration model.

Types of features		Input Parameters
Base Features	Pollution based features measured by LCDS	$PM_{2.5}$ in μgm^{-3}
		PM_{10} in μgm^{-3}
	Weather based features	RH in %
		Temp in $^{\circ}C$
Derived Features	Periodicity due to hour of the day	$\text{Cos}(\frac{2\pi}{24}hour)$
		$\text{Sin}(\frac{2\pi}{24}hour)$
	Derived RH feature [8]	RH^2/(RH-1)
	Effective time lags	$PM_{2.5}$ at t-1, t-2, t-3
		$PM_{2.5}$ at t-23, t-24, t-25
		PM_{10} at t-1, t-2, t-3
		PM_{10} at t-23, t-24, t-25
	Non-linear features	$PM_{2.5}$ × PM_{10}, $PM_{2.5}$ × RH
		$PM_{2.5}$ × Temp
		PM_{10} × RH, PM_{10} × Temp
		$PM_{2.5}$ × Temp × RH,
		PM_{10} × Temp × RH
		$PM_{2.5}$ × PM_{10} × Temp × RH

B. Techniques used to obtain the calibration model

The various technique to model the relationship between LCSDs and CAAQMS are summarised as follows:

1) Linear Regression (LR): It models the linear relationship between input and output. A multivariate linear regression model (MVR) is a commonly used method for the calibration of low-cost sensors [21], [22].

2) Ridge Regression (RR): It reduces the overfitting of the LR model using L_2 regularization [23].

3) Least Absolute Shrinkage and Selection Operator (LASSO): LASSO-based regression (LAR) reduces the overfitting of the LR model using L_1 regularization. [23].

4) Elastic Net Regression (ENR): It reduces the LR model's overfitting by incorporating both the L_1 and L_2 regularization [24].

5) Support Vector Regressor (SVR): SVR can also be used to train a model to map the low-cost sensors' data to the CAAQMS. It estimates the function using a support vector machine. The prediction using SVR depends upon the support vectors [25]. It is utilized to calibrate the low-cost sensors for measuring ozone concentrations [26].

6) Deep Neural Network (DNN): Artificial Neural Network is a widely used tool in various time series regression and forcasting problems [19], [27]. It approximates the non-linear relationship between inputs and output for developing the calibration model. It has an input layer, output layer, and a hidden layer between the input and output layers [28]. The DNN consists of more than one hidden layers.

C. Domain Adaptation

In the present work, a calibration model M is developed at (s,d) and adapted at (s',d') using a shorter duration of training data. This kind of approach is suitable at those sites where the CAAQMS monitors are available for a short duration. As the direct deployment of M at (s',d') may not show good performance due to domain shift [29]. Domain adaptation-based techniques are required to overcome these issues. The domain adaptation refers to adapt the knowledge from one domain to work to the new domain [29], [30]. This kind of approach is found useful for the calibration

of haptic sensors [31]. The steps of the domain adaptation method for start-up A and B are different. For start-up A, M is adapted at (s', d') in two steps as follows:

Algorithm 1: Domain adaptation method

Input: A labeled source domain dataset $\mathcal{D}_S = \{\mathcal{X}_S, \mathcal{Y}_S\}$, a small amount of labeled target domain dataset $\mathcal{D}_{T_1} = \{\mathcal{X}_{T_1}, \mathcal{Y}_{T_1}\}$, and large amount of unlabeled target domain dataset $\mathcal{D}_{T_2} = \{\mathcal{X}_{T_2}\}$.
Output: Labels $\mathcal{Y}_{T_2}$ of the unlabeled data $\mathcal{X}_{T_2}$ in the target domain
1. Normalize the source and target domain features using (1).
2. Learn a regression model f: $\mathcal{X}_S \rightarrow \mathcal{Y}_S^{cal}$ using (3) and (4).
3. Save the models learned at source domain as base model M.
4. Freeze the layer of M and add a new layer at the top of the M.
5. Use the $\mathcal{D}_{T_1}$ to train the newly added top layer with a higher learning rate using (5).
6. Fine-tune the entire model with a lower learning rate by using the $\mathcal{D}_{T_1}$ using (6).
7. Utilize the new learned regression models for predicting the labels of $\mathcal{D}_{T_2}$ as $\mathcal{Y}_{T_2}^{cal} = f_{new}(\mathcal{X}_{T_2})$.
8. Finally, evaluate the performance of the proposed model using the metrices such as, R^2, MAE, $MAPE$, and $SMAPE$ as shown in (7)-(11).

1. In step one, a new layer is added on the top of the layers of M learned at (s, d). The layers trained at (s, d) are frozen and only the newly added layer is trained with a much smaller dataset at (s', d') .

2. In this step, the entire model is finetuned with a much smaller dataset at (s', d').

For start-up B, M is adapted at (s', d') using the step-2 only. The proposed algorithm for the adaptation of M is summarised in Algorithm 1. The dataset of source $(\mathcal{D}_S)$ and target $(\mathcal{D}_T)$ domain are splitted in training and testing data. The size of training dataset at $\mathcal{D}_T$ is much smaller as compared to $\mathcal{D}_S$.

We consider M with parameters θ at (s, d), that maps LCSDs observations (x) to calibrated $PM_{2.5}$ $(\hat{y})$. We use a DNN model to implement M. The parameters of M at (s, d) are initialized randomly. The architecture of M consists of one input layer, seven hidden layers, and one output layer. The parameter θ of M is updated using the two months of data at (s, d) using gradient descent algorithm over $\mathcal{D}_S$ as follows,

$$\theta' \leftarrow \theta - \alpha \nabla \mathcal{L}_{\mathcal{D}_S}(\theta) \tag{3}$$

Here, $\alpha \in \mathbb{R}$ is the learning rate. The θ and θ' represent the randomly initialized weights and learned weights of M. The ∇ denotes the gradient. The used loss function is the mean absolute error, which is defined as

$$\mathcal{L}_{\mathcal{D}_S}(\theta) = \sum_{(x,y)\in\mathcal{D}_S} |f_\theta(x) - y| \tag{4}$$

Now, to adapt M at (s', d'), we discard the output layer of M and add a layer on the top of M. The weights of the layers trained at (s, d) are freezed and the weights (ϕ) of the added layer are trained using two days of data at (s', d'):

$$\phi' \leftarrow \phi - \beta \nabla_\phi \mathcal{L}_{\mathcal{D}_T}(\theta', \phi) \tag{5}$$

Here, $\beta \in \mathbb{R}$ is the learning rate which is kept higher than $\alpha \in \mathbb{R}$. The ϕ and ϕ' are the randomly initialize weights and the learned weights for the new layer added on top of M.

Finally, the parameters (θ', ϕ') of the entire model are fine-tuned using two days of data at (s', d') as follows,

$$\Theta \leftarrow (\theta', \phi') - \gamma \nabla \mathcal{L}_{\mathcal{D}_T}(\theta', \phi') \tag{6}$$

Here, $\gamma \in \mathbb{R}$ is the learning rate which is kept lower than $\alpha \in \mathbb{R}$. The Θ denotes the learned weight after the adaptation of the calibration model at (s', d'). The adapted model is tested using the rest of the data at (s', d') over $\mathcal{D}_T$.

TABLE II: The selected values of the parameters for developing the base calibration model.

Model	Parameters to be tuned		Used values
SVR	Kernel		Linear
ENR	Regularization parameters (alpha, beta)		1
RR	Regularization parameter (alpha)		1
LAR	Regularization parameter (Beta)		1
DNN Model	Units in input layer		27
	No. of hidden layers		7
	Units in hidden layer		512,256,128,64 32,16,8
	Units in output layer		1
	training epochs, activation function		Max-2000, RELU
	learning rate		0.001
	loss function		MSE
	early stopping (patience, monitor)		(100, val-loss)
Domain adaptation	Step-1	learning rate	0.1
		early stopping (patience, monitor)	(50, val-loss)
	Step-2	learning rate	0.0001
		early stopping (patience, monitor)	(50, val-loss)

D. Calibration model's performance evaluation criteria

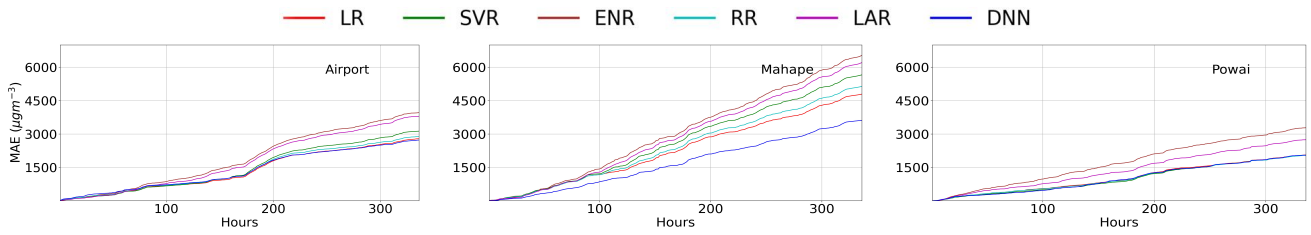
The coefficient of determination R^2, the mean absolute error (MAE), the mean absolute percentage error (MAPE), and symmetric mean absolute percentage error (SMAPE) are computed to evaluate the performance of the calibration model. These are expressed as follows:

$$\overline{z} = \frac{1}{K}\sum_{t=1}^{K} z_t \tag{7}$$

$$R^2 = 1 - \frac{\sum_{t=1}^{K} |z_t - \hat{z}_t(\mathbf{x})|^2}{\sum_{t=1}^{K} |z_t - \overline{z}|^2} \tag{8}$$

$$MAE = \frac{1}{K}\sum_{t=1}^{K} |z_t - \hat{z}_t(\mathbf{x})| \tag{9}$$

$$MAPE = \frac{1}{K}\sum_{t=1}^{K} \frac{|z_t - \hat{z}_t(\mathbf{x})|}{|z_t|} \times 100\% \tag{10}$$



(a) (b) (c)

Fig. 3: Performance evaluation in terms of cumulative MAE (μgm^{-3}), cumulated over deployment time of the different models for start-up A in Task I.

TABLE III: Average performance (over different locations) of different models for Task-I (start-up A).

Model	Performance of M with base features			Performance of M with base+derived features		
	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
UNC	-0.06	25.93	22.60	-0.06	25.93	22.60
LR	0.60	17.50	17.46	0.81	13.03	13.15
SVR	0.59	17.75	17.79	0.80	13.05	13.20
ENR	0.50	20.32	20.08	0.74	15.35	15.21
RR	0.60	17.50	17.46	0.80	13.33	13.50
LAR	0.58	18.25	18.22	0.76	14.10	14.21
DNN	**0.66**	**16.7**	**15.81**	**0.82**	**12.05**	**11.84**

TABLE IV: Average performance (over different locations) of different models for Task-I (start-up B).

Model	Performance of M with base features			Performance of M with base+derived features		
	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
UNC	0.50	18.00	18.31	0.50	18.00	18.31
LR	0.55	18.18	18.94	0.74	14.43	14.77
SVR	0.52	18.31	19.31	0.66	15.40	16.27
ENR	0.35	21.45	22.15	0.60	16.96	17.62
RR	0.55	18.17	18.93	0.70	14.48	14.85
LAR	0.48	19.13	20.23	0.61	16.52	17.70
DNN	**0.66**	**15.78**	**15.46**	**0.75**	**13.46**	**13.17**

$$SMAPE = \frac{1}{K}\sum_{t=1}^{K}\left(\frac{2|z_t - \hat{z}_t(\mathbf{x})|}{|z_t| + |\hat{z}_t(\mathbf{x})|}\right) \times 100\% \qquad (11)$$

where, K denotes total number of observations. The z_t and $\hat{z}_t(\mathbf{x})$ are the CAAQMS' and calibrated $PM_{2.5}$ values for observation t, respectively.

IV. Experimental Results and Discussion

We have experimented with two kinds of tasks. In the first task, we have trained M using the data at (s, d), and test its performance for the same (s, d). In the second task, the performance of M at (s', d') with and without domain adaptation are compared.

A. Task I: Calibration Model for a fixed site and device (s, d)

The training data consists of parallel data recorded simultaneously with the CAAQMS and d. The training and testing of M is performed using the data obtained by d, which is collocated with CAAQMS at s. For the experiments, out of the entire dataset, the last two weeks' data (336 hours) is used for testing, the data from two weeks previous to that is used for validation and the remaining data is used for training. We have experimented with two sets of features - base features set and base+derived features set. These feature sets have already been described in Table I. The average values of evaluation metrics, averaged over eight locations, using different models are summarised in Table III and IV for start-ups A and B, respectively. Here, UNC denotes the trivial model which outputs the uncalibrated $PM_{2.5}$ values. We have found that the DNN model performed best for start-up A and B. Also, with the addition of the derived features, an improvement in the performance of M, over that with just using the base features, can be observed with all the models for both the start-ups. We have found this improvement to be statistically significant as the p-value for the R^2 score is less than 0.05 for all the models applied to the uncalibrated data of start-up A.

We have also computed the MAE for each location's calibration model. The cumulated MAE for the calibrated $PM_{2.5}$ values at Airport, Mahape, and Powai are shown in Fig. 3 for

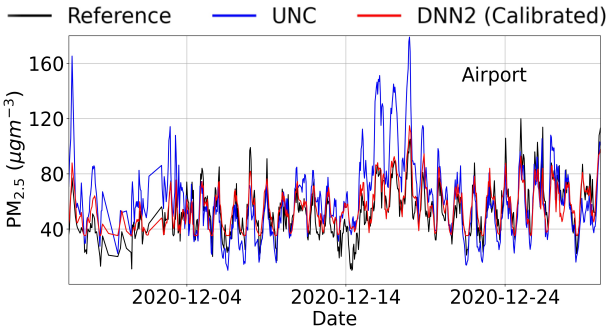
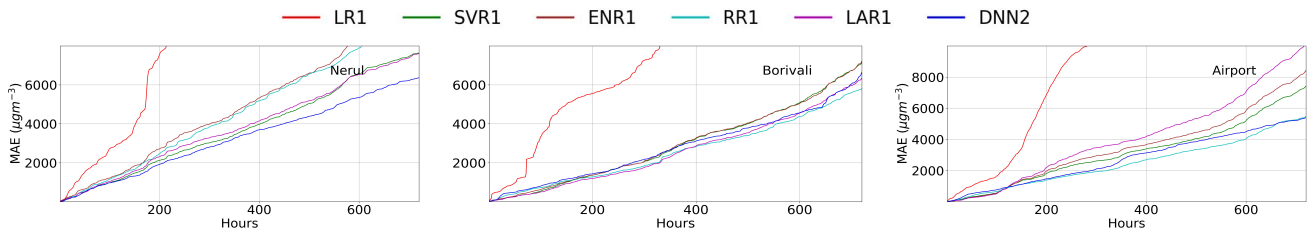


Fig. 4: The plot showing $PM_{2.5}$ levels measured by CAAQMS, uncalibrated $PM_{2.5}$ levels measured by LCSD, and calibrated $PM_{2.5}$ values using DNN2 for Task-II (Start-up B).

TABLE V: Average performance (over different locations) of different calibration models deployed at (s', d') without domain adaptation for Task-II (start-up A and B).

Model	**Testset-1**			**Testset-2**		
	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
UNC	0.31	22.81	20.70	0.36	21.45	19.77
LR	-1.53	37.76	39.31	-1.51	38.77	40.28
SVR	**-0.40**	35.01	38.40	-0.40	34.65	39.60
ENR	-0.55	36.02	38.32	-0.61	36.47	41.42
RR	-0.49	35.79	39.02	-0.39	34.47	39.07
LAR	-0.50	35.78	38.44	-0.57	36.22	41.47
DNN	-0.45	**34.38**	**37.62**	**-0.39**	**33.14**	**37.72**



(a) (b) (c)

Fig. 5: Performance evaluation of DNN2 and baseline models for Task-II in terms of cumulative MAE (μgm^{-3}), cumulated over deployment time for start-up A ((**a**), (**b**)) and start-up B (**c**) for Testset-1.

TABLE VI: Performance comparison of different calibration models for Task-II (start-up A).

	Testset-1			Testset-2		
Model	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
	Mahape					
UNC	0.27	24.05	20.80	0.65	15.85	15.30
LR1	-1.96	38.80	32.62	-5.40	44.72	33.66
SVR1	0.82	12.91	11.60	0.78	11.88	12.74
ENR1	0.77	15.72	13.31	0.66	14.64	15.68
RR1	0.60	16.54	14.91	0.69	14.98	14.51
LAR1	0.87	11.01	9.94	0.78	11.57	12.55
DNN1	0.71	15.08	13.96	0.66	15.59	16.68
DNN2	**0.88**	**9.49**	**9.32**	**0.84**	**10.93**	**11.83**
	Nerul					
UNC	0.63	16.81	15.96	0.65	14.60	14.78
LR1	-13.36	77.95	40.24	-3.19	45.75	42.84
SVR1	0.81	16.40	13.62	0.73	13.05	13.51
ENR1	0.68	21.01	16.91	0.48	17.54	18.55
RR1	0.69	17.06	15.86	0.48	17.05	18.96
LAR1	0.81	16.88	13.71	0.69	13.68	14.17
DNN1	-0.26	34.58	28.44	-0.84	31.09	35.40
DNN2	**0.87**	**11.45**	**10.96**	**0.84**	**10.73**	**10.62**
	Borivali					
UNC	0.80	17.38	16.19	0.61	28.46	24.17
LR1	-3.33	43.46	41.15	-7.31	69.18	53.11
SVR1	0.89	12.70	12.39	0.78	**20.89**	**18.21**
ENR1	0.88	13.41	12.47	**0.79**	22.27	19.16
RR1	0.92	10.89	10.18	0.69	26.15	22.16
LAR1	**0.91**	**11.24**	**10.71**	0.76	23.17	20.00
DNN1	0.66	20.79	18.94	0.15	38.82	30.81
DNN2	0.88	11.58	11.05	0.72	26.08	22.10

TABLE VII: Performance comparison of different calibration models for Task II (start-up B).

	Testset-1			Testset-2		
Model	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
	Mahape					
UNC	0.30	22.71	25.09	0.31	21.67	24.34
LR1	-13.79	72.90	90.80	-2.71	56.53	84.04
SVR1	0.49	19.41	18.63	0.49	19.76	21.76
ENR1	0.56	18.02	17.26	0.43	20.63	22.62
RR1	0.25	23.83	22.61	0.39	22.40	25.31
LAR1	0.60	15.94	17.06	0.09	29.16	34.90
DNN1	-4.09	63.27	95.72	-1.38	59.08	88.33
DNN2	**0.68**	**13.95**	**14.66**	**0.51**	**16.45**	**17.56**
	Vileparle					
UNC	0.09	22.39	19.79	0.21	21.76	19.08
LR1	-74.18	199.28	127.49	-41.74	175.17	114.12
SVR1	**0.70**	**15.51**	**14.09**	**0.79**	**13.87**	**12.83**
ENR1	0.59	18.10	16.02	0.51	18.59	19.02
RR1	0.54	18.33	16.36	0.55	21.01	17.88
LAR1	0.39	23.62	19.55	0.63	20.54	17.24
DNN1	-2.73	46.08	61.62	-1.31	49.17	64.77
DNN2	0.55	18.52	16.57	0.76	14.93	13.44
	Airport					
UNC	-0.52	35.53	30.04	-0.62	29.00	24.31
LR1	-4.54	74.73	48.83	-3.21	65.56	45.23
SVR1	0.46	20.08	20.65	0.42	18.31	20.57
ENR1	0.28	22.14	23.23	0.14	22.40	25.77
RR1	0.71	**15.74**	**15.33**	0.68	14.09	14.98
LAR1	0.03	25.46	28.74	-0.25	28.26	33.84
DNN1	-0.15	30.29	28.97	-0.83	34.84	41.02
DNN2	**0.73**	18.31	15.58	**0.80**	**13.28**	**12.16**

start-up A and found minimum for DNN model.

B. Task II: Model adaptation to different site and device: (s, d) → (s′, d′)

In Task II, M is deployed at (s', d') in two ways to show the effectiveness of domain adaptation. First, M is deployed at (s', d') without domain adaptation and the results are shown in Tables V. It can be noted that the performance of M is not satisfactory when deployed at (s', d') without domain adaptation. Similar observation is also found in [18]. Even if the performance of DNN is found to be slightly better as compared to the other models deployed at (s', d') without domain adaptation, the UNC is the best. Further, the M is adapted using the domain adaptation method at (s', d') with a shorter collocation period. Only two days of data (48 samples) is used to adapt M, which is validated using the next seven days' data for both start-ups A and B. The remaining data is divided into two parts named as Testset-1 and Testset-2 to examine the adapted model performance in different time duration. Testset-1 contains 720 samples, and number of samples in Testset-2 vary at different locations in the range of 322 samples to 1000 samples. This is due to the different number of missing samples corresponding to the different locations. It should be noted that Testset-1 and Testset-2, shown in Table V to VIII, are of same duration. For start-up A, the M is developed at Airport and adapted at seven target locations. For start-up B, the M is developed at Nerul and adapted at three target locations.

To show the usefulness of the domain adaptation method in reducing the collocation time, the baseline models are also developed at (s', d') from scratch. For the fair comparison of adapted model and baseline models the same split of training, validation, and the testing dataset are explored and the performance is compared in Tables VI and VII for start-up A and B, respectively. In these tables, the LR1, SVR1, ENR1, RR1, LAR1, and DNN1 are the baseline models trained at (s', d') from scratch, and DNN2 is the adapted calibration model. It can be observed that the performance of DNN1 is

TABLE VIII: Average performance of different calibration models across different sites for Task-II (start-up A and B).

Model	**Testset-1**			**Testset-2**		
	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
UNC	0.31	22.81	20.70	0.36	21.45	19.77
LR1	-8.85	64.87	47.15	-3.84	53.03	50.87
SVR1	0.68	16.61	15.20	0.72	15.21	15.03
ENR1	0.65	18.19	16.18	0.61	17.28	17.47
RR1	0.65	16.55	15.20	0.67	16.74	16.19
LAR1	0.67	16.64	15.36	0.61	17.39	17.79
DNN1	-0.48	30.57	33.16	-0.15	31.16	35.73
DNN2	**0.75**	**14.04**	**13.57**	**0.76**	**14.70**	**14.38**

found to be inferior as compared to LR1, SVR1, ENR1, RR1, and LAR1. This is due to the smaller size of the training dataset. The DNN1 requires a large amount of training data to improve the performance as compared to the other baseline models. However, with the use of the domain adaptation method, the limitation of the smaller training dataset can be overcome. This fact is reflected in the performance of the DNN2 which is formed using the domain adaptation method. For startup A, it can be observed that DNN2 has performed better at five out of seven target locations for Testset-1 and two target locations for Testset-2. For Mahape, Nerul, and Borivali, the performance parameters are shown in Table VI. A summary of the performance evaluation at the remaining locations for start-up A is provided in the supplementary material. From Table VII, it can be seen that for start-up B, the DNN2 have shown better performance at two out of three target locations. Even though we have observed that different baseline models have performed better than DNN2 at some locations and some test datasets, the performance of DNN2 is found to be overall consistent, as seen in Table VIII. It should also be noted that the performance of DNN2 is also found to be substantially better than that of M deployed at (s', d') without domain adaptation as can be seen in Table V and VIII.

These results demonstrate the usefulness of the proposed domain adaptation-based calibration model in reducing the collocation time of LCSDs with CAAQMS at target locations. The best-attained values of the evaluation parameters R^2, MAPE (%), and SMAPE (%) are shown in the bold font in Table VI and VII.

In Fig. 4, the time series of $PM_{2.5}$ for CAAQMS, UNC, and calibrated $PM_{2.5}$ using DNN2 are compared for start-up B. It can be observed from Fig. 4 and Table VII that the $PM_{2.5}$ time series obtained after calibration is showing better correlation with the CAAQMS' $PM_{2.5}$. We have also computed the MAE for DNN2 and baseline models at (s', d'). The cumulative sum of MAE for DNN2 is found to be lower or comparable to the baseline models at most of the sites. The cumulative MAE values for the calibrated $PM_{2.5}$ values are shown in Fig. 5. These results indicate the robustness of the proposed calibration methodology.

Finally, the performance of DNN and DNN2 are also compared for the same test dataset of 14 days at (s', d') and the results are shown in Table IX and X. It is expected that DNN will perform better than DNN2 because the former is trained with much more data at (s', d') as compared to the latter. However, it is interesting to note that the performance of the latter is not very far from that of the former. Moreover, at certain locations, the performance of DNN2 even surpasses that of DNN. These results show the potential of the proposed domain adaptation method in reducing the collocation time for low cost sensor calibration.

TABLE IX: Performance comparison of DNN and DNN2 (Start-up A). DNN: Trained using 2 months data at (s', d') from scratch. DNN2: Adapted at (s', d') using 2 days data.

	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
Model	Mahape			Borivali		
DNN	**0.89**	**9.37**	**9.92**	**0.7**	**18.52**	**16.23**
DNN2	0.84	12.57	13.75	0.38	30.24	24.85
	Nerul			Kalyan		
DNN	**0.92**	**8.15**	**7.87**	0.82	14.82	13.78
DNN2	0.85	11.40	10.93	**0.88**	**13.20**	**11.81**
	Vileparle			Worli		
DNN	**0.71**	**13.80**	**14.75**	**0.88**	**10.57**	**10.54**
DNN2	0.69	13.68	13.47	0.66	19.44	16.54
	Powai					
DNN	**0.92**	**8.99**	**8.66**			
DNN2	0.86	13.64	12.61			

TABLE X: Performance comparison of DNN and DNN2 (Start-up B).

Model	R^2	MAPE (%)	SMAPE (%)	R^2	MAPE (%)	SMAPE (%)
	Airport			Vileparle		
DNN	**0.84**	**10.72**	**10.02**	0.70	**14.63**	**14.69**
DNN2	0.82	13.21	12.10	**0.71**	16.32	14.57
	Mahape					
DNN	**0.60**	**17.80**	**17.14**			
DNN2	0.51	16.48	17.61			

We have used scikit-learn and TensorFlow libraries of Python programming language. To form the DNN model, the training time is 18 sec to 40 sec, and testing time is 0.005 sec with following hardware configuration (GPU: Quadro P2200, 64 GB RAM, intel i7-10700K processor, Python-3.7.10, and tensorflow-gpu-2.1.0). To adapt the model at (s', d') (DNN2), the training time is 45 sec to 65 sec and testing time is 0.4 sec with following hardware configuration (4GB RAM, intel i3 processor, Python-3.7.10, and tensorflow-cpu-2.1.0). The dataset and code used in this work are available at https://github.com/madhavlab/2021ksonuSensorAdaptation.

V. Conclusion and Future Work

In this work, a novel calibration method for measured $PM_{2.5}$ levels using LCSDs is proposed. This method makes use of deep learning for high performance and domain adaptation for reducing the time required to collocate the LCSDs with the CAAQMS at the target location. Also, new input features are derived that improve the performance of the calibration model. We compare the performance of the proposed domain adaptation based calibration method and the proposed features with several machine learning-based calibration methods and find improvements with the proposed method. With the proposed domain adaptation based calibration model, we find that two days of collocation with the CAAQMS is sufficient to calibrate a new LCSD at the target location. This method

is very useful in efficiently expanding the deployment of air quality monitoring if we have mobile CAAQMS available for calibration. Also, we find that the proposed method shows high calibration performance, even with a very short collocation, very close to the performance of the deep networks trained with a large collocation data. Our future work will focus on reducing this gap in performance. In future, we will also study the sensor drift problem from the domain adaptation perspective. We will also work towards anomaly detection in the sensor networks. In future, we will also work to adapt the proposed calibration model for seasonal variations. Moreover, we would like to extend the proposed method to the calibration of gaseous pollutants such as CO, SO_2, and NO_x.

ACKNOWLEDGMENT

This research work is supported by Maharashtra pollution control board (MPCB) and Bloomberg Philanthropies.

REFERENCES

[1] "How air pollution is destroying our health." 2019, accessed 2021-01-26. [Online]. Available: https://www.who.int/airpollution/news-and-events/how-airpollution-is-destroying-our-health

[2] S. Chowdhury, S. Dey, and K. Smith, "Ambient $PM_{2.5}$ exposure expected premature mortality to 2100 in India under climate change scenarios." *Nat. Commun.*, vol. 9, no. 1, p. 318, 2018.

[3] K. A. Koehler and T. M. Peters, "New methods for personal exposure monitoring for airborne particles." *Curr Envir Health Rpt*, vol. 2, pp. 399–411, 2015.

[4] S. S. Lim *et al.*, "A comparative risk assessment of burden of disease and injury attributable to 67 risk factors and risk factor clusters in 21 regions, 1990–2010: A systematic analysis for the global burden of disease study 2010."

[5] R. Sahu *et al.*, "Validation of low-cost sensors in measuring real-time PM_{10} concentrations at two sites in Delhi national capital region," *Sensors*, vol. 20, no. 5, p. 1347, 2020.

[6] S. Munir, M. Mayfield, D. Coca, and S. Jubb, "Analysing the performance of low-cost air quality sensors, their drivers, relative benefits and calibration in cities–A case study in Sheffield," *Environ Monit Assess*, vol. 191, p. 94, 2019.

[7] D. M. Holstius, A. Pillarisetti, K. R. Smith, and E. Seto, "Field calibrations of a low-cost aerosol sensor at a regulatory monitoring site in California," *Atmospheric Measurement Techniques*, vol. 7, no. 4, pp. 1121–1131, 2014.

[8] T. Zheng *et al.*, "Field evaluation of low-cost particulate matter sensors in high- and low-concentration environments," *Atmospheric Measurement Techniques*, vol. 11, no. 8, pp. 4823–4846, 2018.

[9] R. Piedrahita *et al.*, "The next generation of low-cost personal air quality sensors for quantitative exposure monitoring," *Atmospheric Measurement Techniques*, vol. 7, pp. 3325–3336, 2014.

[10] E. G. Snyder *et al.*, "The changing paradigm of air pollution monitoring," *Environmental Science & Technology*, vol. 47, no. 20, pp. 11 369–11 377, 2013.

[11] R. Sahu *et al.*, "Robust statistical calibration and characterization of portable low-cost air quality monitoring sensors to quantify real-time O_3 and NO_2 concentrations in diverse environments," *Atmospheric Measurement Techniques*, vol. 14, no. 1, pp. 37–52, 2021.

[12] F. Delaine, B. Lebental, and H. Rivano, "In situ calibration algorithms for environmental sensor networks: A review," *IEEE Sensors Journal*, vol. 19, no. 15, pp. 5968–5978, 2019.

[13] R. Wang *et al.*, "A category-based calibration approach with fault tolerance for air monitoring sensors," *IEEE Sensors Journal*, vol. 20, no. 18, pp. 10 756–10 765, 2020.

[14] A. Mukherjee, L. G. Stanton, A. R. Graham, and P. T. Roberts, "Assessing the utility of low-cost particulate matter sensors over a 12-week period in the Cuyama valley of California," *Sensors*, vol. 17, p. 1805, 2017.

[15] K. K. Johnson, M. H. Bergin, A. G. Russell, and G. S. Hagler, "Field test of several low-cost particulate matter sensors in high and low concentration urban environments," *Aerosol and Air Quality Research*, vol. 18, no. 3, pp. 565–578, 2018.

[16] C. Borrego *et al.*, "Assessment of air quality microsensors versus reference methods: The eunetair joint exercise," *Atmospheric Environment*, vol. 147, pp. 246–263, 2016.

[17] T. Zheng, M. H. Bergin, R. Sutaria, S. N. Tripathi, R. Caldow, and D. E. Carlson, "Gaussian process regression model for dynamically calibrating and surveilling a wireless low-cost particulate matter sensor network in Delhi," *Atmospheric Measurement Techniques*, vol. 12, no. 9, pp. 5161–5181, 2019.

[18] M. A. Zaidan *et al.*, "Intelligent calibration and virtual sensing for integrated low-cost air quality sensors," *IEEE Sensors Journal*, vol. 20, no. 22, pp. 13 638–13 652, 2020.

[19] S. K. Jha, C. L. Dewangan, and N. K. Verma, "Multi-step load demand forecasting using neural network," in *2019 20th International Conference on Intelligent System Application to Power Systems (ISAP)*, 2019, pp. 1–6.

[20] N. M. Pindoriya, S. N. Singh, and S. K. Singh, "One-step-ahead hourly load forecasting using artificial neural network," in *2009 International Conference on Power Systems*, 2009, pp. 1–6.

[21] W. Jiao *et al.*, "Community air sensor network (CAIRSENSE) project: evaluation of low-cost sensor performance in a suburban environment in the southeastern United States," *Atmospheric measurement techniques*, vol. 9, no. 11, pp. 5281–5292, 2016.

[22] J. M. Barcelo-Ordinas, J. Garcia-Vidal, M. Doudou, S. Rodrigo-Muñoz, and A. Cerezo-Llavero, "Calibrating low-cost air quality sensors using multiple arrays of sensors," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6.

[23] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[24] J. Li and W. Chen, "Forecasting macroeconomic time series: LASSO-based approaches and their forecast combinations with dynamic factor models," *International Journal of Forecasting*, vol. 30, no. 4, pp. 996–1015, 2014.

[25] A. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[26] J. M. Barcelo-Ordinas, M. Doudou, J. Garcia-Vidal, and N. Badache, "Self-calibration methods for uncontrolled environments in sensor networks: A reference survey," *Ad Hoc Networks*, vol. 88, pp. 142–159, 2019.

[27] C. L. Dewangan, S. N. Singh, and S. Chakrabarti, "Solar irradiance forecasting using wavelet neural network," in *2017 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, 2017, pp. 1–6.

[28] A. K. Alexandridis and A. D. Zapranis, *Wavelet neural networks: with applications in financial engineering, chaos, and classification*. John Wiley & Sons, 2014.

[29] X. Xu, X. Zhou, R. Venkatesan, G. Swaminathan, and O. Majumder, "d-SNE: Domain adaptation using stochastic neighborhood embedding," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2492–2501.

[30] Q. Sun, Y. Liu, T. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning," *CoRR*, vol. abs/1812.02391, 2018. [Online]. Available: http://arxiv.org/abs/1812.02391

[31] S. Wang and N. Xi, "Calibration of haptic sensors using transfer learning," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 2003–2012, 2021.