REAL-TIME PREDICTION FOR FINE-GRAINED AIR QUALITY MONITORING SYSTEM WITH ASYNCHRONOUS SENSING

Zixuan Bai, Zhiwen Hu, Kaigui Bian and Lingyang Song

School of Electronics Engineering and Computer Science, Peking University, Beijing, China

ABSTRACT

Due to the significant air pollution problem, monitoring and prediction for air quality have become increasingly necessary. To provide real-time fine-grained air quality monitoring and prediction in urban areas, we have established our own Internet-of-Things-based sensing system in Peking University. Due to the energy constraint of the sensors, it is preferred that the sensors wake up alternatively in an asynchronous pattern, which leads to a sparse sensing dataset. In this paper, we propose a novel approach to predict the real-time fine-grained air quality based on asynchronous sensing. The sparse dataset and the spatial-temporal-meteorological relations are modeled into the correlation graph, in which way the prediction procedures are carefully designed. The advantage of the proposed solution over existing ones is evaluated over the dataset collected by our air quality monitoring system.

***Index Terms*—** Air quality, fine-grained, real-time

1. INTRODUCTION

A recent report from the World Health Organization shows that one in eight of total global deaths are due to air pollution exposure [1]. Government agencies have defined the air quality index (AQI) to evaluate pollution degree where high concentration of fine particles is usually the main factor of a high AQI. Traditional AQI observation stations can only provide a coarse-grained and high-latency monitoring [2]. However, a recent study shows that the distribution of fine particles could vary within meters [3].

To make up for the above deficiency, it is recommended to deploy numerous low-cost tiny Internet-of-Things (IoT) sensing devices to monitor fine-grained air quality(mainly $\mathrm{PM}_{2.5}$ values) for the regions with complicated terrain [4]. Therefore, we have designed a wireless sensor network system. The massive commercial IoT sensors can monitor AQIs until the batteries are dead. In this way, the real-time fine-grained monitoring and prediction are achieved by the massive collected data and the techniques like machine learning [5, 6].

However, since the outdoor battery-powered sensors have limited energy [7, 8], it is preferred that the sensors wake up alternately in an asynchronous way to prolong the lifetime of sensing network [9]. Based on the collected asynchronous data, few existing methods can provide reliable fine-grained AQI prediction. For instance, Long Short Term Memory network (LSTM) [5] can only be implemented with a synchronous dataset rather than an asynchronous one. In addition, Multi-layer Perception (MLP) [6] and spatial-temporal distance weighting interpolation (IDW) [10] do not perform well based on asynchronous AQI data. This is because, MLP is unable to reveal the spatial-temporal-meteorological relations among the AQIs in different points-of-interest (POIs), and IDW is only an intuitive solution without learning from massive collected data and the influence of the weather conditions.

In this paper, we propose a novel scheme to conduct the fine-grained and real-time prediction of AQI based on asynchronous data collected by our monitoring system. By designing the correlation graph (CG), we present the asynchronous sensing data and the spatial-temporal-meteorological relations. Based on the CG model, the prediction procedures are carefully designed and an optimization problem arises. To obtain the real-time fine-grained prediction results, we aim to solve the optimization problem by an algorithm combining a closed-form derivation and genetic algorithm. The advantage of the proposed solution over existing ones is evaluated over the dataset collected by our monitoring system.

The main contributions of our work are listed as follows:

1) We present our air quality system in Peking University based on massive IoT sensors, where asynchronous sensing data are modeled into a spatial-temporal-meteorological graph.

2) We propose a novel prediction algorithm for real-time and fine-grained air quality based on sparse dataset.

3) We evaluate the performance gain of our approach over existing methods based on real measured data.

The rest of our paper is organized as follows. Section 2 introduces the CG model. Section 3 specifies the prediction procedures and applies appropriate methods to obtain real-time and fine-grained prediction. Section 4 shows the simulation results. Finally, conclusions are given in Section 5.

2. SYSTEM MODEL

In this section, we first introduce our air quality asynchronous sensing system in Section 2.1. Then, the correlation graph is constructed to model the asynchronous sensing data in Section 2.2. The spatial-temporal-meteorological relations are added into the CG as weighted edges in Section 2.3.

2.1. Asynchronous Sensing System

We establish a wireless sensor network system for fine-grained air quality monitoring. For power efficiency, the deployed IoT sensors are programmed to asynchronously monitor AQI (mainly $PM_{2.5}$ values) until the batteries are dead.

The IoT sensors have been deployed in the campus of Peking University since Feb. 2018. Each sensor uploads the AQI, the location and the time after monitoring [11]. In addition, weather conditions are collected from the website of China Meteorological Administration [12]. All features are preprocessed before being applied.

2.2. Correlation Graph for Asynchronous Data

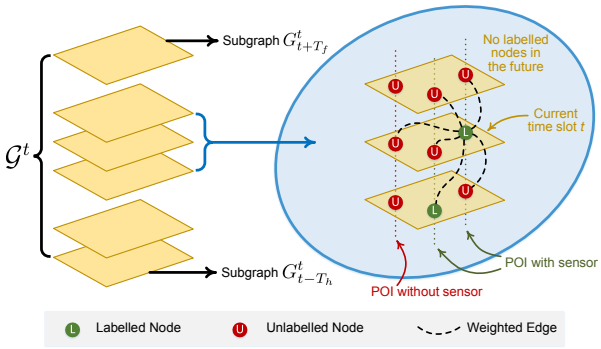

Fig. 1: An illustration of the Correlation Graph.

There are totally a number of M air quality sensors, denoted as $\{S_1, \cdots, S_m, \cdots, S_M\}$. Each sensor has a fixed 3D spatial coordinate (x_m, y_m, z_m). For fine-grained consideration, there are L POIs in concerned areas and we need to know the AQIs of the POIs, where $M < L$. Due to the limited number of the available sensors, the AQIs of the POIs without the sensors need to be estimated.

Since the sensors monitor data asynchronously, we divide the system into short equal-length time slots, where each data can be considered as collected in a specific time slot. The collected data therefore could be very sparse, because only a small proportion of the sensors collect and upload data at any certain time slot.

At time t, we construct correlation graph $\mathcal{G}^t$ to illustrate the state of the system. $\mathcal{G}^t$ has multiple subgraphs, denoted as $\{G^t_{t'}\}$. Each subgraph $G^t_{t'}$ contains L nodes, where each node implies the collected value or the estimated value of a specific POI at time t'. The node is *labeled* if the corresponding data is collected by the sensor, otherwise, the node is *unlabeled*. The sets of the *labeled* and *unlabeled* nodes are given by $\mathcal{L}^t$ and $\mathcal{U}^t$ respectively. Each *labeled* node v^t_n is combined with a measured value denoted as c^t_n. For each $\mathcal{G}^t$, we only consider limited number of subgraphs, given by $\{G^t_{t-T_h}, \cdots, G^t_{t+T_f}\}$, where T_h is the number of concerned historical subgraphs from current time t and T_f is the number of concerned future subgraphs from current time t. Therefore, the total number of the nodes in $\mathcal{G}^t$ is $N = (T_h + T_f + 1) \times L$. The set of all the nodes is denoted as $\mathcal{V}^t = \{v^t_1, \cdots, v^t_n \cdots, v^t_N\} = \mathcal{L}^t \bigcup \mathcal{U}^t$. We define the real-time fine-grained prediction as $\mathcal{F}^t = [F^t_1, \cdots, F^t_n, \ldots, F^t_N]^{\mathrm{T}}$, where F^t_n denotes the inferred value for node v^t_n at time t.

2.3. Feature Relations in Correlation Graph

Any of two nodes are connected by an edge. The weight of the edge depends on the spatial-temporal relation of the two nodes, as well as the weather conditions of the corresponding time slots.

The weighted feature vector: We first define the spatial-temporal-meteorological feature vector of node v^t_n as

$$\boldsymbol{q}^t_n = \left[q^t_{n,1}, \cdots, q^t_{n,k}, \cdots, q^t_{n,K}\right]^{\mathrm{T}}, \tag{1}$$

where K is the number of features and $q^t_{n,k}$ is the k-th feature of v^t_n. The feature vector $\boldsymbol{q}^t_n$ contains the spatial coordinates, the temporal coordinate of v^t_n, and the meteorological conditions, including weather types, wind speeds, wind directions, temperature and humidity at time slot t. Due to the fact that different elements in $\boldsymbol{q}^t_n$ may have different impacts on the relations between two nodes, we introduce a weight vector $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_k, \cdots, \beta_K]^{\mathrm{T}}$, where $\beta_k \in (0, 1), \forall k \in \{1, 2, \cdots, K\}$ and $||\boldsymbol{\beta}||_1 = \sum_{i=1}^{K} |\beta_k| = \sum_{i=1}^{K} \beta_k = 1$. Then the weighted feature vector $\boldsymbol{f}^t_n = [f^t_{n,1}, \cdots, f^t_{n,k}, \cdots, f^t_{n,K}]^{\mathrm{T}}$ of node v^t_n is expressed as

$$\boldsymbol{f}^t_n = \boldsymbol{q}^t_n \odot \boldsymbol{\beta} = \left[q^t_{n,1}\beta_1, \cdots, q^t_{n,k}\beta_k, \cdots, q^t_{n,K}\beta_K\right]^{\mathrm{T}}, \tag{2}$$

where $\odot$ denotes hadamard product.

Similarity of two nodes: We define the adjusted cosine similarity m^t_{ij} between v^t_i, v^t_j, which is given by

$$m^t_{ij} = \frac{\langle \boldsymbol{f}^t_i - \overline{\boldsymbol{f}}, \boldsymbol{f}^t_j - \overline{\boldsymbol{f}} \rangle}{\sqrt{\left\|\boldsymbol{f}^t_i - \overline{\boldsymbol{f}}\right\|_2^2} \sqrt{\left\|\boldsymbol{f}^t_j - \overline{\boldsymbol{f}}\right\|_2^2}}, \tag{3}$$

where $\overline{\boldsymbol{f}}$ denotes the mean values of the weighted features.

The weight on each edge: For each node, its k-nearest neighbors are defined as the nodes that have the top k highest similarity with this certain node. If node v^t_j is one of v^t_i's k-nearest neighbors **or** v^t_i is one of v^t_j's k-nearest neighbors, we define the weight of the edge between v^t_i, v^t_j as

$$w^t_{ij} = \frac{1}{2}\tanh\left(\alpha_1(m^t_{ij} - \alpha_2)\right) + \frac{1}{2}, \tag{4}$$

where the parameter $\alpha_1 (\alpha_1 > 0)$ magnifies the differences of similarities and the parameter $\alpha_2 \in (-1, 1)$ is a cutoff value. w^t_{ij} is in $(0, 1)$ since it is a monotone function and m^t_{ij} ranges from -1 to 1. If node v^t_i, v^t_j are not one of the k-nearest neighbors of each other, the weight w^t_{ij} between them is set to 0 to reduce the influence by dissimilar nodes and to improve computing efficiency. It is worth stressing that v^t_i is not a neighbor of itself and we simply set $w^t_{ii} = 0, \forall v^t_i \in \mathcal{V}^t$.

Further, a real symmetric weight matrix W^t can then be given by $W^t = [W^t]^{\mathrm{T}} = \left[w^t_{ij}\right]_{N \times N}$. In addition, the degree d^t_n of node v^t_n is given by $d^t_n = \sum_{i=1}^{N} w^t_{in} = \sum_{i=1}^{N} w^t_{ni}$.

3. PREDICTION PROCEDURES DESIGN

After receiving the monitoring data at time slot t, the prediction system immediately executes a round of iteration in order to obtain the real-time prediction $\mathcal{F}^t$, by using the history prediction $\mathcal{F}^{t-1}$ and the measured data $\{c_n^t\}$ for the *labeled* nodes. This includes two procedures, namely the preparation procedure and the estimation procedure, as shown in Fig. 2.

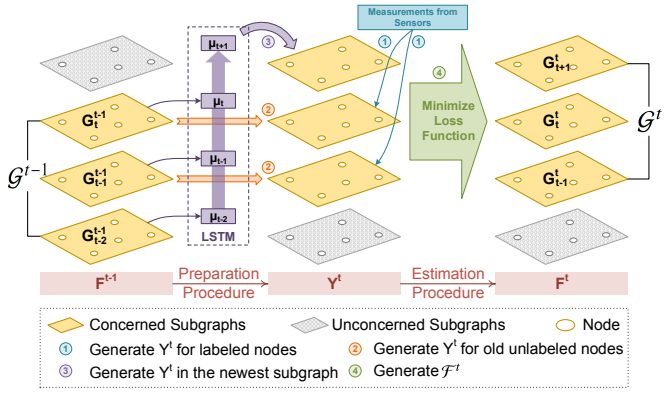

Fig. 2: An illustration of the prediction procedure, where ①, ② and ③ show the preparation procedure to obtain the pre-estimation vector $\mathcal{Y}^t$, and ④ shows the estimation procedure to obtain the final estimation result $\mathcal{F}^t$.

3.1. Preparation Procedure

Based on $\mathcal{G}^{t-1}$ and $\mathcal{F}^{t-1}$ for the last time slot $t-1$, and the measured values $\{c_n^t\}$ at the current time slot t, we first construct a pre-estimation vector for the current time, given by $\mathcal{Y}^t = [Y_1^t, \cdots, Y_n^t, \cdots, Y_N^t]^{\mathrm{T}}$. $\mathcal{Y}^t$ only shows the rough values of the nodes in $\mathcal{G}^t$ and needs to be refined in the subsequent estimation procedure.

For the *labeled* nodes: We set $Y_n^t = c_n^t$ for node v_n^t.

For the *unlabeled* nodes not in the newly-added subgraph: When it comes to the t-th time slot, $\mathcal{G}^t$ is generated by removing the oldest subgraph and adding a new subgraph for time $t+T_f$. The *unlabeled* nodes not in the newly-added subgraph have been estimated in the last-time prediction. Therefore, for the node v_n^t in these subgraphs, we set Y_n^t the same to the last-time estimated value $F_{n'}^{t-1}$ (Suppose the n-th node in $\mathcal{G}^t$ is the n'-th node in $\mathcal{G}^{t-1}$).

For the *unlabeled* nodes in the newly-added subgraph: An LSTM [13] network is used to determine their pre-estimated values. Specifically, the mean value μ_τ for each monitoring time slot τ is calculated first to represent the coarse-grained estimation, given by

$$\mu_\tau = \overline{F_\tau}, \quad \tau = 1, 2, \cdots, t-1+T_f, \tag{5}$$

where $\overline{F_\tau}$ denotes the average of the last-time estimated values for time slot τ. With the meteorological features and a series of the coarse-grained estimations as the training dataset, an LSTM network can be trained in advance and can be immediately used to predict the coarse-grained estimation μ_{t+T_f} in the newly-added subgraph. Finally, we set μ_{t+T_f} as the pre-estimations for all the nodes in $G_{t+T_f}^t$.

To sum up, the pre-estimation is given by

$$Y_n^t = \begin{cases} c_n^t, & v_n^t \in \mathcal{L}^t, \\ F_{n'}^{t-1}, & v_n^t \in \mathcal{U}^t \text{ and } v_n^t \notin G_{t+T_f}^t, \\ \mu_{t+T_f}, & v_n^t \in G_{t+T_f}^t. \end{cases} \tag{6}$$

The steps ①, ②, and ③ in Fig. 2 illustrate the three kinds of assignments in (6) respectively.

3.2. Estimation Procedure

Based on the pre-estimated values in $\mathcal{Y}^t$, we need to calculate more precise prediction values in $\mathcal{F}^t$.

3.2.1. *Minimizing the loss function*

Two aspects are concerned when calculating $\mathcal{F}^t$. The first one is smoothness, implying that two closely related nodes (with a high weight edge between them) need to have similar values. The second one is reliability, indicating that the final estimation and the pre-estimation should be as same as possible, given by $\mathcal{F}^t \approx \mathcal{Y}^t$.

Combining *smoothness* and *reliability*, we design the loss function with the help of [14], given by:

$$\mathbb{L}^t(\mathcal{F}^t, \boldsymbol{\beta}) = \frac{1}{2} \sum_{v_i^t, v_j^t \in \mathcal{V}^t} w_{ij}^t \left(\frac{F_i^t}{\sqrt{d_i^t}} - \frac{F_j^t}{\sqrt{d_j^t}} \right)^2 + \lambda \sum_{v_i^t \in \mathcal{V}^t} \left(F_i^t - Y_i^t \right)^2, \tag{7}$$

where λ is the balanced parameter. The first term on the right side in (7) is the *smoothness* term. Instead of using the differences of the function values on the two neighboring nodes directly, we regularize each function value F_i^t with the corresponding normalized coefficient $1/\sqrt{d_i^t}$. The second term on the right side in (7) is the *reliability* term.

It is worthy to note that $\boldsymbol{\beta}$ does not need to be updated frequently, since the weights of different features are supposed to be stable in a short time. Our objective is to minimize the average of the loss functions in $\mathcal{T}$, given by

$$\begin{aligned} &\arg\min_{\boldsymbol{\beta}, \{\mathcal{F}^t\}} \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \mathbb{L}^t(\boldsymbol{\beta}, \mathcal{F}^t), \\ &s.t. \quad ||\boldsymbol{\beta}||_1 = 1, \quad \beta_k > 0, \quad \forall \beta_k \in \{1, \cdots, K\}, \end{aligned} \tag{8}$$

where $\mathcal{T}$ represents a set of the history monitoring time slots. Note that the iterative utilization of the estimated *unlabeled* nodes for each next round forms a semi-supervised learning approach.

The minimization problem in (8) can be divided into two optimization sub-problems. Supposing that $\boldsymbol{\beta}$ is fixed, the first sub-problem is given by

$$\mathbb{L}_1^t(\boldsymbol{\beta}) = \min_{\mathcal{F}^t} \mathbb{L}^t(\mathcal{F}^t, \boldsymbol{\beta}). \tag{9}$$

Correspondingly, the second sub-problem is given by

$$\begin{aligned} &\arg\min_{\boldsymbol{\beta}} \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \mathbb{L}_1^t(\boldsymbol{\beta}), \\ &s.t. \quad ||\boldsymbol{\beta}||_1 = 1, \quad \beta_k > 0, \quad \forall \beta_k \in \{1, \cdots, K\}. \end{aligned} \tag{10}$$

3.2.2. *Solution to sub-problem (9)*

$\mathbb{L}^t(\mathcal{F}^t, \boldsymbol{\beta})$ in (7) is a convex function when $\boldsymbol{\beta}$ is given. We differentiate $\mathbb{L}^t$ with respect to $\mathcal{F}^t$ and set it to $\mathbf{0}$, given by

$$\frac{\partial \mathbb{L}^t}{\partial \mathcal{F}^t} = \left[\frac{\partial \mathbb{L}^t}{\partial F_1^t}, \frac{\partial \mathbb{L}^t}{\partial F_2^t}, \cdots, \frac{\partial \mathbb{L}^t}{\partial F_N^t} \right]^{\mathrm{T}} = \mathbf{0}^{\mathrm{T}}. \tag{11}$$

After derivation, the final solution of $\mathcal{F}^t$ is [1] given as

$$\mathcal{F}^t = \frac{\lambda}{1+\lambda}\left(I - \frac{1}{1+\lambda}(D^t)^{-\frac{1}{2}}W^t(D^t)^{-\frac{1}{2}}\right)^{-1}\mathcal{Y}^t, \quad (12)$$

where $(D^t)^{-\frac{1}{2}} = diag(1/\sqrt{d_1^t}, \cdots, 1/\sqrt{d_N^t})$.

It is noteworthy that once $\boldsymbol{\beta}$ is figured out, we can directly obtain the real-time fine-grained prediction according to (12).

3.2.3. Solution to sub-problem (10)

The value of $\boldsymbol{\beta}$ is obtained by the following genetic algorithm based on long-term weather conditions and the measured air quality data.

Each individual $\mathcal{P}(n)$ in the generation set $\mathcal{P}$ represents a trial solution to (10) and we encode it to a series of binary digits $\mathcal{P}(n) = (s_1^n, s_2^n, \cdots, s_{K\times R}^n)$ as its genotype, where R denotes the encoding length of each element in $\boldsymbol{\beta}$ and K is the number of features. Therefore, $K \times R$ denotes the total length of an individual's genotype. The genetic algorithm is specified as follows.

Crossover: For any two individuals in the old generation, crossover can be performed with a fixed probability p_c to produce new individuals. To be specific, for individuals $\mathcal{P}(i)$ and $\mathcal{P}(j)$, we randomly choose a position $p = 2, 3, \cdots, K \times R$ and then the genotypes for the new individuals are denoted as $\mathcal{P}(i') = (s_1^i, \cdots, s_{p-1}^i, s_p^j, \cdots, s_{K\times R}^j)$ and $\mathcal{P}(j') = (s_1^j, \cdots, s_{p-1}^j, s_p^i, \cdots, s_{K\times R}^i)$ after crossover.

Mutation: Each binary digit in the old generation after crossover remains the same in most of time and alters to the opposite with a fixed probability p_m.

Selection: The fitness value for each individual in the old generation is given by

$$z(\mathcal{P}(n)) = 1/\left(\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\mathbb{L}_1^t(\boldsymbol{\beta})\right), \ \forall \mathcal{P}(n) \in \mathcal{P}. \quad (13)$$

The new generation is chosen from the individuals after crossover and mutation according to their survival probabilities where the survival probability for each individual is in proportion to its fitness value.

Evolution Overview: Crossover and mutation are first performed to create new individuals in the old generation. To satisfy the constraint $||\boldsymbol{\beta}||_1 = 1$ for the new individuals, we normalize them and it is easy to prove that the fitness values won't change after normalization. The new generation is then produced by the selection procedure with constraining the size of the new generation equals to the size of the old one. The genetic algorithm is terminated when a better individual hasn't appeared in the latest E generations.

4. EVALUATION

4.1. Parameter Setup

In asynchronous sensing, the probability of each sensor detecting data at each time slot is simply set to $1/5$ to illustrate the most general case. The other parameters are listed in Table 1.

[1] Since $\left|I - \frac{1}{1+\lambda}(D^t)^{-\frac{1}{2}}W^t(D^t)^{-\frac{1}{2}}\right| \neq 0$, the matrix is invertible.

Table 1: Parameter Setup

Number of nodes in one subgraph L	60
Number of sensors in one subgraph M	between 5 to 30
Similarity parameters α_1 and α_2	20 and 0
Number of history subgraphs T_h	between 3 to 8
Number of future subgraphs T_f	between 1 to 7
Encoding length R	20
Parameter k in k-NN	200
Trade-off parameter λ	0.3
Crossover probability p_c	0.6
Mutation probability p_m	0.05
Termination conditions E	500
Length of each time slot	5 minutes

4.2. Simulation Results and Discussions

In Fig. 3(a), we set $M = 30$ and compare the performances when the numbers of the history/future subgraphs change, where the performances are indicated by average relative prediction errors. The results indicate that the appropriate numbers of the history/future subgraphs lead to a better performance. For the history subgraphs, $\mathcal{G}^t$ may lack history information when T_h is too small, and $\mathcal{G}^t$ will contain worthless information if T_h is too big. For the future subgraphs, $\mathcal{G}^t$ may have insufficient meteorological conditions in the near future when T_f is too small, and $\mathcal{G}^t$ will contain relatively few *labeled* nodes if T_f is too big.

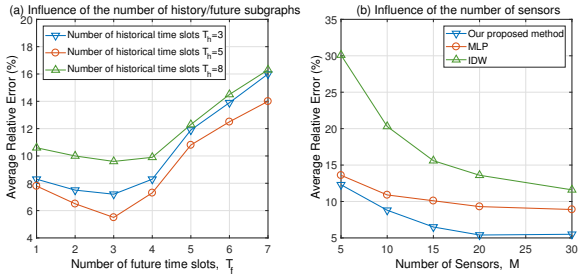

Fig. 3: The performance of the proposed solution with respect to different settings.

In Fig. 3(b), we compare the performances of different approaches, including Multi-layer perception (MLP), Inverse distance weighting interpolation [10], and the proposed CG Model (with $T_h = 5$, $T_f = 3$). A significant performance gain of our solution can be observed, and the average relative error can be reduced to 5% when there are enough sensors.

5. CONCLUSION

In this paper, we modeled the asynchronous air quality data and their spatial-temporal-meteorological relations by using the weighted CG. A novel real-time fine-grained prediction method was proposed based on the sparse dataset. Simulation results show that the size of the CG should not be too big or too small. The proposed scheme reduces the error to 5% (with $T_h = 5$, $T_f = 3$, $M = 30$) while the error of MLP is about 9% in asynchronous sensing situation.

6. ACKNOWLEDGEMENT

This work was supported by the CERNET Innovation Project under grant number NGII20161011.

7. REFERENCES

[1] World Health Organization et al., "7 Million Premature Deaths Annually Linked to Air Pollution," *World Health Organization, Geneva, Switzerland*, Mar. 2014.

[2] Beijing memc. (jul. 2018). *Beijing Municipal Environmental Monitoring center*. [Online]. Available: http://www.bjmemc.com.cn/ .

[3] T. N. Quang, C. He, L. Morawska, L. D. Knibbs, and M. Falk, "Vertical Particle Concentration Profiles Around Urban Office Buildings", *Atmospheric Chemistry and Physics*, vol. 12, no. 11, pp. 5017–5030, May 2012.

[4] Y. Hu, G. Dai, J. Fan, Y. Wu and H. Zhang, "BlueAer: a fine-grained urban PM2.5 3D monitoring system using mobile sensing," in *Proc. IEEE INFOCOM*, San Francisco, CA, Jul. 2016.

[5] C. J. Huang and P. H. Kuo, "A Deep CNN-LSTM Model for Particulate Matter (PM2.5) Forecasting in Smart Cities", *Sensors*, vol. 18, no. 7, pp. 2220, Jul. 2018.

[6] I. G. McKendry, "Evaluation of Artificial Neural Networks for Fine Particulate Pollution (PM10 and PM2.5 Forecasting", *Journal of the Air & Waste Management Association*, vol. 52, no. 9, pp. 1096–1101, 2002.

[7] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions", *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013.

[8] G. B. Fioccola, R. Sommese, I. Tufano, R. Canonico, and G. Ventre, "Polluino: An Efficient Cloud-based Management of IoT Devices for Air Quality Monitoring", in *Proc. IEEE Research and Technologies for Society and Industry Leveraging a better tomorrow*, pp. 1–6, Sep. 2016.

[9] Z. Hu, Z. Bai, K. Bian, T. Wang, and L. Song, "Real-time Fine-grained Air Quality Sensing Networks in Smart City: Design, Implementation and Optimization", *arXiv preprint arXiv:1810.08514*, Oct. 2018.

[10] L. Li, T. Losser, C. Yorke, and R. Piltner, "Fast Inverse Distance Weighting-based Spatiotemporal Interpolation: A Web-based Application of Interpolating Daily Fine Particulate Matter PM2.5 in the Contiguous U.S. Using Parallel Programming and k-d Tree", *International Journal of Environmental Research and Public Health*, vol. 11, no. 9, pp. 9101–9141, Sep. 2014.

[11] Dataset collected by our air quality monitoring system. [Online]. Available: https://github.com/pkubzx/pku-air .

[12] Meteorological conditions provided by China Meteorological Administration. [Online]. Available: http://data.cma.cn .

[13] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to Forget: Continual Prediction with LSTM," 1999.

[14] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with Local and Global Consistency," in *Proc. Advances in neural information processing systems*, Vancouver, British, Dec. 2004.