Sustainable Computing: Informatics and Systems 29 (2021) 100479

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

journal homepage: www.elsevier.com/locate/suscom

Phone-based ambient temperature sensing using opportunistic crowdsensing and machine learning

Amee Trivedi *, Phuthipong Bovornkeeratiroj, Joseph Breda, Prashant Shenoy, Jay Taneja, David Irwin

University of Massachusetts Amherst, USA

ARTICLE INFO

Keywords:
Smartphone
Temperature sensing
Temperature control
Crowdsensing
Indoor ambient temperature
Temperature prediction

ABSTRACT

Due to the ubiquitous nature of smartphones, opportunistic phone-based crowdsensing has emerged as an important sensing modality. Since fine-grain ambient temperature measurements are a pre-requisite for energy-efficient operation of heating and cooling (HVAC) systems in buildings, in this paper, we use mobile phone sensing in conjunction with a web-based crowdsensing system to obtain detailed ambient temperature estimates inside buildings. We present a machine learning approach based on a random forest ensemble learning model that uses the phone battery temperature sensor to infer the ambient air temperature. We also present a few-shot transfer learning method to quickly learn and deploy our model onto new phones with modest training overheads. Our crowdsensing web service enables predictions made by multiple phones to be aggregated in an opportunistic fashion, extending our approach from an individual level to a community level. We evaluate our ML-based model for a range of devices, operating scenarios, and ambient temperatures, and see mean errors of less than $\pm 0.5^{\circ}$F for our temperature predictions. More generally, our results show the feasibility of using an on-device ML model for ambient temperature predictions in mobile phones. This allows buildings – new and old, with and without sensing systems – to benefit from a new class of ubiquitous temperature sensors, enabling more sustainable operation.

1. Introduction

The ubiquitous nature of mobile phones along with their sensor-rich features have made mobile phone sensing an integral part of monitoring systems such as activity recognition [1,2], health monitoring [3,4], and environment monitoring [5], to name a few. Mobile phone sensing, when combined with opportunistic crowdsensing, has opened up new opportunities for sensing of our physical environment without the need to deploy a fixed sensing infrastructure for problems such as urban environment monitoring [6,7], traffic monitoring [8], public transportation tracking [9–11], and parking availability [12]. More recently, phone sensing has found use for energy sustainability, specifically for building energy efficiency.

Building energy efficiency is an important problem since buildings account for nearly 70% of the total electricity and 40% of the total energy consumption in the U.S. [13], with heating, ventilation, and cooling (HVAC) systems accounting for over 50% of the energy consumed. To increase the sustainability and usability of HVAC systems, there has been a substantial effort to enhance energy-efficient control of these systems while simultaneously providing improved user comfort [14]. Today's buildings have sophisticated HVAC systems with independently controlled zones that are driven by multiple thermostats and temperature sensors. However, research has shown that such instrumentation is often misconfigured, leading to energy wastage or discomfort for occupants. Older buildings tend to have older instrumentation that is more prone to such configuration issues. To address this challenge in this paper, we present a technique to crowdsense ambient temperature measurements from mobile phones, which can then serve as an additional source of temperature data to HVAC systems for its energy optimizations.

Our approach focuses on using mobile phones to sense indoor ambient air temperature, in effect using each phone as a mobile digital ambient thermometer. The primary challenge, however, is that phones lack a sensor to directly measure ambient temperature and must infer this information using other on-board sensors. The feasibility of using battery temperature sensors inside phones for ambient temperature

* Corresponding author.
E-mail address: amee@cs.umass.edu (A. Trivedi).

https://doi.org/10.1016/j.suscom.2020.100479

Available online 1 November 2020

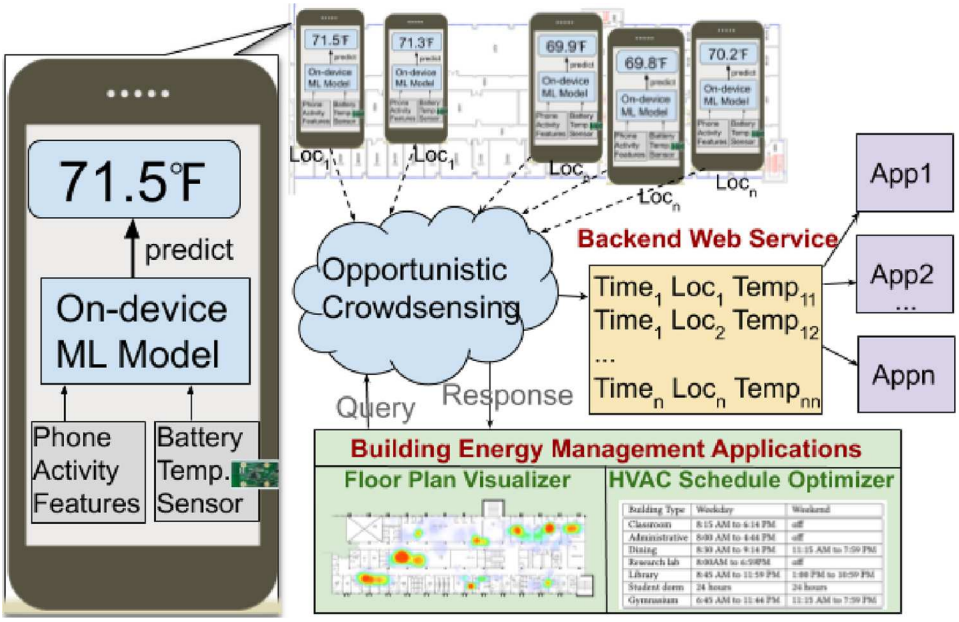

Fig. 1. Use of mobile phones to infer ambient air temperatures and a web service to crowdsense these predictions for energy management.

sensors was shown in [15] using a Physics-based model. Our approach, depicted in Fig. 1, is based on machine learning and significantly improves upon our prior work [15] in three important ways. First, we use a machine learning (ML) model based on a Random Forest Bagging Regressor to infer ambient air temperature and show that our approach yields a much more accurate temperature estimate (3× lower error) than the state-of-the-art Physics-based approach. Second, to ease the deployment of our ML model on a broad range of phones with modest effort, we use a few-shot based transfer learning method, where we use only a few labeled samples on a new phone to adapt a model trained on another phone. Third, we develop an end-to-end web-based system to incorporate phone sensing into a web-based opportunistic crowdsensing platform for ambient indoor temperature sensing at a community scale.

In designing and implementing our approach, we make the following contributions:

- We propose an approach to infer ambient temperature using on-device machine learning model that leverages mobile phone battery temperature. We use an opportunistic crowdsensing based approach that aggregates the readings across phones at a location and computes an ambient temperature reading per building zone or room.
- We design a data-driven, on-device, Random Forest regressor-based model for ambient temperature prediction and achieve better performance than the current state-of-the-art approach.
- We present a few-shot based transfer learning approach for the ambient temperature prediction model to be deployed on a never-before-seen phone.
- We develop a full prototype of our system implemented as a RESTful web service to crowdsense indoor temperature measurements in conjunction with an indoor positioning system (IPS). We develop policies to distinguish high-quality crowdsourced measurements from lower quality ones.
- We evaluate our approach using multiple devices and scenarios. Our results show that our ML approach yields a prediction error of 0.5 °F or less and MAE of 0.3 or less for our few-shot based transfer learning approach.

2. Background and related work

In this section, we present background, review state-of-the-art techniques, and discuss challenges to opportunistic crowdsensing-based ambient temperature sensing using ML techniques.

Energy sustainability. Energy sustainability and increasing resource-use efficiency are key United Nations Sustainable Development Goals (SDG). Buildings account for over 40% of the energy and 75% of the electricity usage in the many developed countries. Heating, cooling, and ventilation (HVAC) is the largest component of a building's energy usage and accounts for 50% of the energy usage in typical buildings [16]. Thus, reducing energy waste of HVAC systems and improving energy efficiency is important from a sustainability standpoint and one that can have a societal impact. Misconfigured and miscalibrated thermostats can result in over-use of the HVAC system (which wastes energy) or under-heat or under-cool areas (which causes user discomfort). We propose mobile sensing and crowdsensing based approach that allows anyone with a smartphone to collect ambient temperature anywhere in the building. We believe such an approach is easy to deploy in current buildings and can be used by HVAC systems as additional source of temperature data.

Building temperature sensing. Modern smart buildings incorporate fine-grained and ubiquitous instrumentation to provide energy-efficient and comfortable environments using smart thermostats. Researchers have explored the use of zonal temperature control [17,18], people-location scheduling based on preference [19], and personalized control system scheduling [20]. All of these techniques require a fine-grained temperature sensing infrastructure throughout the building. Older buildings have HVAC systems with controllable zones but often lack such a sensing infrastructure due to sparsely-deployed thermostats and instrumentation. As noted earlier, regardless of whether a building is old or new, misconfiguration of instrumentation is a common problem in practice, resulting in energy waste from non-optimal HVAC control. Our work explores the use of mobile phones as ambient temperature sensors to enable older buildings to provide features similar to newer smart buildings. Also, any building where sensors may be mis-calibrated can also benefit from our approach by leveraging it as an additional source of ambient temperature measurements and using it for HVAC control.

Mobile crowdsensing. Mobile crowdsensing combines mobile

sensing and web services to opportunistically crowdsource data from phones via the web with the common intention to collectively share the acquired data towards a common interest or for common good. The Mobile crowdsensing paradigm has been used across multiple contexts such as environmental quality monitoring, noise pollution assessment, and traffic monitoring. Here, we employ mobile crowdsensing to crowdsense ambient air temperature measurements inside buildings.

Mobile crowdsensing has been used to crowd-source measurements, using mobile phones, for applications such as environmental quality monitoring [7,21], noise pollution assessment [6,22,23], and traffic monitoring [8,24–26]. In [10] researchers have designed systems for Intelligent Transportation Systems (ITS) without explicit user input using participatory sensing. Here, we employ mobile sensing to crowdsense ambient air temperature measurements inside buildings.

Phone-based sensing. While a mobile phone is a sensor-rich platform, it lacks sensors to directly measure ambient air temperatures. To date, two phone models had such sensors (Motorola Moto X and Samsung Galaxy S3), but these sensors were removed in later models, seemingly for cost and accuracy reasons. Although phones lack an ambient temperature sensor, all phones are equipped with a battery temperature sensor to continuously monitor the battery and prevent it from overheating. Two prior studies [7,21] have shown the feasibility of crowdsensing *outdoor* air temperature at the city scales using battery temperature. However, the approach used coarse-grained measurements that are *not location-specific* and also did not consider the impact of heat generated by phone activities. In our prior work [15] we showed the feasibility of this approach for *indoor* temperature sensing. We demonstrated a strong correlation between ambient temperature and battery temperature across various levels of phone activities. We used a Physics-based approach to model the thermodynamic effects of phone activities on battery temperature and used Newton's law of cooling to estimate ambient air temperature. However, a key drawback of [15] is that it incorporates seven distinct models based on a different set of phone activities and requires a sophisticated detection algorithm to choose which model can best estimate air temperature. Also, accurate ambient temperature sensing directly from a smartphone is infeasible due to the internal heat generated by the phone that affects the temperature sensor reading [27]. Additionally, different manufactures and models use different thermal insulation material that hinders the usage of one fixed rule-based approach to extract ambient temperature from temperature sensor readings.

In this work, we argue for a purely data-driven approach to infer ambient air temperature based on machine learning. Our approach uses a single ML model based on ensemble learning instead of a set of distinct models to make its predictions. We hypothesize that the use of a single ML model, rather than multiple state-specific models, can simplify the task of ambient temperature prediction across various phone states and yield better accuracy than the prior state-of-the-art [15]. Additionally, we demonstrate that it is easier to train new phones by using few shot based transfer learning and opportunistic crowdsensing data from more than one phone improves the accuracy of the computed ambient temperature by at least 2.5 × , which itself is a significant contribution.

3. Ambient temperature sensing using mobile-based machine learning

In this section, we describe the behavior of phones under different conditions, present our machine learning model and a few-shot based transfer learning approach to quickly deploy our models on new phones with limited training.

3.1. Modeling phone temperature behavior

The battery temperature of the phone depends on activities that the phone is performing and the heat dissipation due to those activities. The four main activities that impact phone battery temperature are CPU utilization, network utilization, phone charging, and screen usage. Although CPU and network utilization are continuous values, we model them as binary variables with values high or low; the utilization is high when it exceeds 80% and is low otherwise. For simplicity, we also use binary states (on/off) for screen usage and battery charging (lithium battery charge control typically has two or three states). Different combinations of these activities (e.g., screen usage with high CPU load or screen use while charging) have different impacts on battery temperature due to different amounts of power draw and heat dissipation. Regardless of which activities are performed by the phone, its battery temperature shows the generic behavior depicted in Fig. 2.

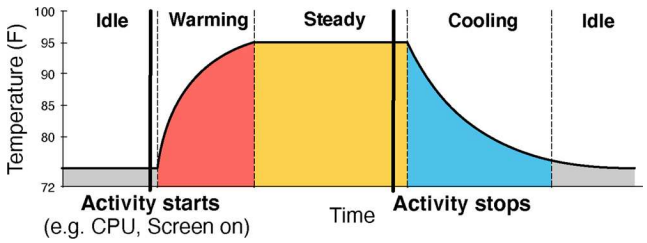

Fig. 2. Phone battery temperature behavior.

Table 1
Phone state transitions.

Current state	Next state
Idle	Idle/warming
Warming	Warming/steady
Steady	Steady/cooling
Cooling	Cooling/idle

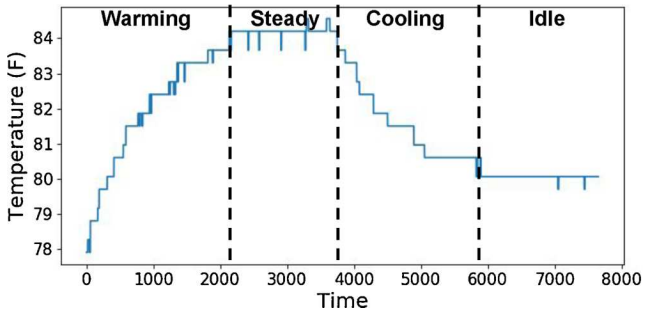

Fig. 3. Real phone battery temperature behavior.

The phone is initially idle where its temperature roughly equals ambient temperature. When the phone starts performing activities, its battery temperature starts rising as depicted by the warming phase in Fig. 2. The phone battery then reaches a steady state equilibrium temperature where heat generated equals heat loss. When the activity stops, the battery temperature starts falling, as depicted by the cooling phase.

This battery temperature behavior is similar with any combination of screen, CPU, network, and charging is considered; however, the rate of warming/cooling and time duration of each sub-state for different combinations are different in each case. Further, given Fig. 2, the phone may make a consistent transition between its states (idle, warming, steady, cooling) as shown in Table 1. For example, an idle state is always followed by a transition to warming. Fig. 3 shows an actual battery temperature on a phone, it follows the 4 sub-states we described earlier.

3.2. Phone context identification

Based on the empirical analysis and methods of our prior work [15], we have found that there can be a different impact on battery temperature and heat dissipation depending on how and where a phone is used

Table 2
Ambient temperature model features.

Features	Values
CPU	High/low
Network	High/low
Screen	On/off
Charging	On/off
Battery Current	Continuous value
Battery Voltage	Continuous value
Battery Level	Integer value
Battery temperature (current value and 5 previous values)	Continuous value

by the user. Our phone context identification method seeks to distinguish whether the phone is: (i) indoors or outdoors, (ii) in motion or stationary, and (iii) exposed to ambient air (e.g., in hand, on a table) or not (e.g., in a bag or a pocket). We use a simple classifier that uses a combination of WiFi and GPS signal strength to classify the environment as indoors or outdoors. For detecting, if the phone is in-motion or stationary, we use data from the accelerometer as well as the phone location determined by an Indoor Positioning System (IPS) explained in detail later in Section 4.2. Finally, the activity recognition model from [28,29] is used to determine if the phone is exposed to ambient air. Our prediction model is triggered only when the phone is indoors, exposed to ambient air, and stays at a given location for a settling period.

3.3. Machine learning model

For estimating ambient temperatures, we use an ensemble learning-based bagging Random Forest (RF) regressor. In the ensemble-based learning paradigm, rather than using a single highly-accurate model for predicting ambient air temperature, we learn a large number of weak models and combine their predictions. The number of decision trees in the forest, say B, is a hyper-parameter that can be tuned by the user. In the bagging-based RF regressor, we bootstrap samples from the dataset and create B samples. Bootstrapping to create B samples from the original dataset reduces the variance resulting in low overfitting. Next, we train B decision trees on each of the B samples.

Our model is trained on 13 features as input (see Table 2) and the true ambient air temperature as the output. While learning at each split of the decision tree, a random subset of features from the set of 13 features are selected. The selection of a random subset of features helps avoid correlation between trees in the forest. To make predictions, our ensemble model considers the average of all the predictions made by the trees ("weaker models"). Ensemble learning has multiple advantages such as low training overhead and low overfitting. Low correlation between trees aids in improving the overall accuracy of the regressor since good models will likely agree on the accurate/good predictions while bad models will likely disagree on other predictions due to the avoidance of correlation of the trees.

For our task of predicting ambient temperature, we use 13 features listed in Table 2 that are indicators of phone activity and state of the phone. As described in Section 3.1, the combination of the CPU, network, screen, and charging features captures ongoing phone activities. CPU and network load take the value high/low, with high indicating 80% or more usage and low otherwise. Screen usage can be on or off, and the charging feature is again on/off with on indicating that the phone is getting charged otherwise off. Any combination of high CPU load, the screen on, high network load, or charging phone state can lead to an increase in battery temperature. The state of the battery can be captured using four features: current, voltage, battery level, and temperature. Battery current provides an indicator of state change. A change in battery current coupled with trends shown by the temperature sample enables the model to detect a new state transition. Battery charge level and voltage level are also used as key input features.

Together, these features enable our bagging RF models to infer the ambient temperature under different activities and states. Additionally, battery current, battery voltage, and battery level can help identify the charging stage of the phone (e.g. fast charging) which also affects the temperature of the battery. We observe that the current battery temperature is also a function of heat dissipated by phone from the previous state. Also, to learn the current state (idle, warming, steady, cooling) we need to know the gradient of the curve. To address these problems, we need to add a series of previous samples of battery temperature to the features. From an empirical analysis, we found that combining the current temperature with the previous 5 temperature values (lag values) is adequate for the model to determine the state based on the curve gradient and learn the impact of battery temperature from the previous state.

3.4. Few-shot based transfer learning

With the advancement of mobile phone technology and the increasing market demand for mobile phones, an increasing number of phone models gets introduced in the market every year. Deploying and training our model on a new phone requires time, calibration, and ground truth, which is an expensive task. Hence, our system uses a transfer learning approach that enables accelerated bootstrapping of our pre-trained models on a new phone. To reduce the data gathering effort and computation cost of training a model from scratch on every new phone model we use a few-shot based transfer learning approach on a pre-trained model. Our on-device ML model is based on Random Forest, which is a collection of decision trees.

A decision tree is a non-linear model that has a hierarchical structure incorporating non-linear decision rules for making predictions. The usage of a few-shot transfer learning approach is motivated by a key observation that the features and feature dependencies used to predict the ambient temperature are the same across mobile phone models and hence, the decision trees across mobile phones exhibit a structural similarity. The task of ambient temperature prediction across phones from different vendors will need the same set of features but the threshold values of those features that result in the prediction might be different while the tree structure would be the same. We use this key observation to use a pre-trained model as a *seed* and use a few observations/data points collected on the new device to tune the seed parameters to perform inductive transfer learning. The seed model uses the prior knowledge to augment the supervised experience gained from the few new observations on the new mobile phone. In our case, we collect a

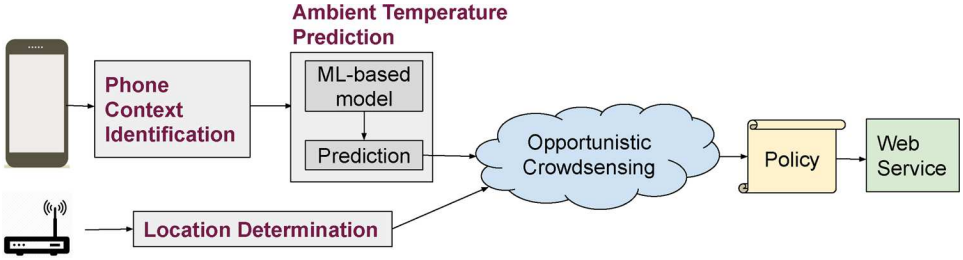

Fig. 4. Overview of our system.

few samples from the target phone (new mobile phone) on which we wish to transfer the pre-trained seed model and label those samples as target samples. Now, to adapt a pre-trained decision tree trained on the source sample to target samples, we fine-tune the threshold values used for features at each split of the decision tree based on the few target samples using gradient descent.

4. Crowdsensing temperature measurements

In this section, we describe our crowdsensing system to collect temperature predictions from the mobile phones of building occupants. The architecture and key components of our system are depicted in Fig. 4. We assume that each phone has an on-device ML model and makes temperature predictions every τ minutes so long as the phone is indoors and not in motion; our system currently uses $\tau = 5$min.

4.1. Handling settling times

User mobility is inherently nomadic, where the user walks to a location, stays there for some time, and then walks to another location. When a user walks to a new location, the phone needs some time to acclimatize to the new location before its temperature predictions are accurate. Later in Section 5, we have empirically determined that this acclimatization period ("settling time") can vary from 3 minutes to 18 minutes depending on the phone vendor, model, and temperature difference. This implies that the longer a phone stays at a location, the longer its temperature sensor has to acclimatize and the more accurate are its predictions. In practice, the crowdsensed predictions made in the first 15 minutes that phone is at a new location will have lower accuracy. Hence, crowdsensing yields data with varying quality of predictions. To mitigate this effect, we have designed a policy to weigh crowdsensed predictions by their likely quality.

Our *duration-based policy* assigns confidence to predictions crowdsensed from a phone based on the duration that the phone has been at a location. The longer the phone has been at a location, the higher the confidence in its predictions. The assignment of confidence enables a weighted average computation of the ambient temperature estimate at a location from multiple crowdsensed measurements. Since some phone models may be inherently more accurate than others (e.g., due to the quality of its hardware sensors), we also use a behavior-based policy that assigns a *device confidence* to each device based on the accuracy of historic predictions and its prediction accuracy when compared to other co-located high-confidence phones. The sum of confidence values assigned using both policies results in a weighted average computation of the ambient temperature prediction.

Additionally, to avoid the overhead of continuous sensing that can interfere with battery life and foreground user activities, we judiciously set the sensing and prediction frequency to once every 5 minutes and turn off predictions when the battery level drops below a low threshold to conserve battery life for other phone activities.

4.2. Location determination

Each crowdsensed measurement needs to be accompanied by the location of the device. Since GPS does not work indoors, we need to employ an Indoor Positioning System (IPS) technique to handle indoor localization of phones. We assume that a WiFi-based indoor positioning system is available for indoor localization—a reasonable assumption due to the widespread deployment of WiFi in office buildings. Many WiFi-based IPS approaches have been proposed in the literature, based on landmark-based localization or RTT-based multilateration, and time-of-flight approaches [30–32]. Any of these approaches can be used in our crowdsensing system; our experiments in our office building have shown the accuracy of around 10m for traditional landmark-based methods, which improves to less than 2 m when using the newest 802.11mc (aka WiFi 6) access points in our experimental deployment of CompuLab WiLD APs [33]. We use Anyplace [34], which is a free and open Indoor Navigation Service and it can embed buildings, floors, rooms into the indoor coordinate system. The phone derives its location using IPS, translates it to the indoor coordinate system from any place system and uses WiFi to upload the location with the measurement.

Table 3
Crowdsensing APIs.

Service	Description
POST /phone/register	Register a phone
POST /phone/localize	Localize the phone
GET /phone/{phone_id}/temp	Return phone temp.
PUT /phone/{phone_id}/temp	Upload temperature
GET /room/{room_id}/temp	Return room temp.
GET /floor/{floor_id}/temp	Return all room temp.
GET /building/all	Return all buildings
GET /floor/all?building={id}	Return all floor
GET /room/all?floor={id}	Return all rooms

Table 4
Characteristics of phones used in our evaluation.

Make	Model	OS	Screen size (in.)
Google/HTC	Pixel	Android 7.1	5.0
LG	Stylo3	Android 7	5.7
Alcatel	5044R	Android 7	5.0
Samsung	sm-g550t	Android 6	5.0
Motorola	Moto X	Android 5	4.7

4.3. Web service

We have implemented a prototype of the system using a RESTful web service in python to crowdsense temperature measurements from participating phones. Our web service exposes two types of interfaces: those to upload new measurements and those to querying prior measurement (see Table 3). The first set of API calls provides various services for the mobile phone such as authentication, localization, and allows the phone to share its measurement to the server for data processing and collection. The second set of interfaces enables other applications to query uploaded measurements and can be used by energy management applications to visualize temperature or for HVAC control to optimize heat in the building.

In this work we do not consider memory related parameters in our list of 13 parameters stated in Table 2 but we plan on exploring them in our follow-up work. Additionally, we are considering the use of statistical machine learning approaches such as fractional partial differential equations (PDE) [35], which is used to estimate diffusion equation models and predict the evolution of stochastic processes. To generalize the PDE based model across various manufacturers, makes, models, and OS we would use transfer learning approach.

5. Experimental results

In this section, we discuss our experimental methodology, followed by our results.

5.1. Dataset and parameter setting

Dataset: For the evaluation of our model, we use the phone dataset collected from [15]. The dataset, which is discussed in detail in [15], comprises data collected across 5 different phones that cover a wide spectrum of vendors, screen size, battery capacity, and OS as shown in Table 4. We use Google/HTC Pixel phone as the default phone. Our model is general and has been validated for a number of phones with different form factors and battery sizes. We believe the approach should work for iOS devices as well but we are unable to validate it since access

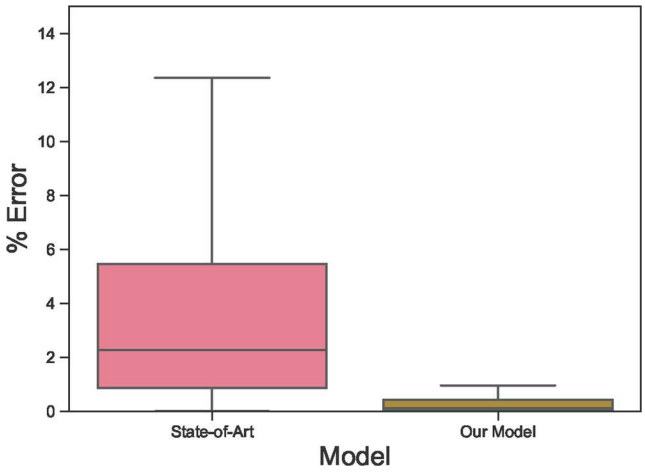

Fig. 5. Error(%) in ambient temp. prediction of state-of-art model and our model when run on same dataset.

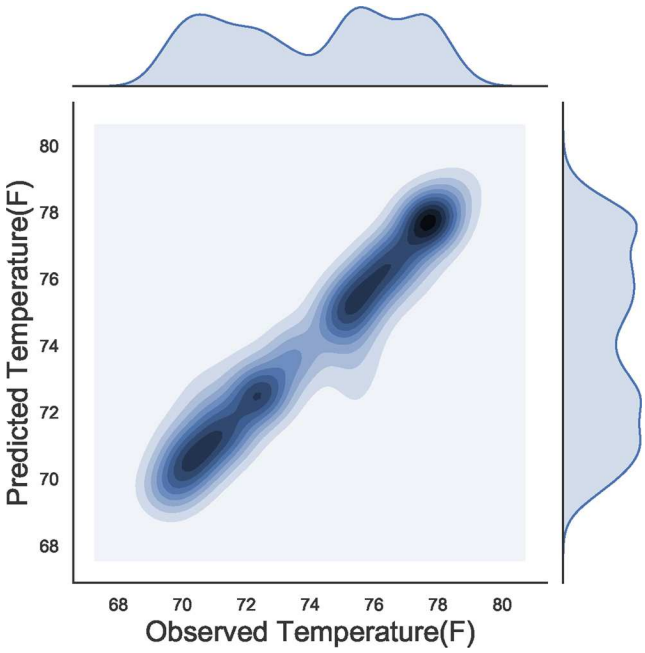

Fig. 6. KDE plot of observed and actual observations, with our model trained on 20% of data across all states and sub-states. Model accuracy 99.52%.

to battery temperature is restricted to the OS and not accessible to the application. For all the evaluations, the ground truth data were collected from a HOBO temperature logger. For each phone, we collect labeled datapoints at a sampling rate of 1 min/sample across all states (as defined in Table 1. We collect data for all combinations of CPU, Network, Screen and Charging.

Parameter setting: To evaluate the robustness of our proposed model we use a train-validation-test split of 20-20-60 with 20% data used as training data, 20% as validation data, and rest 60% as testing data (reason explained below). To ensure that the train, validation and test dataset is a reflection of data points across all stages of the battery temperature curve we select 20% data points from each state – idle, warming, steady, and cooling sections of the collected dataset as train dataset, 20% as validation dataset, and rest 60% as test dataset. This ensures the presence of data points across all sections of the battery temperature curve in training and testing as well as the stratified distribution of the data points across the training and testing datasets. This

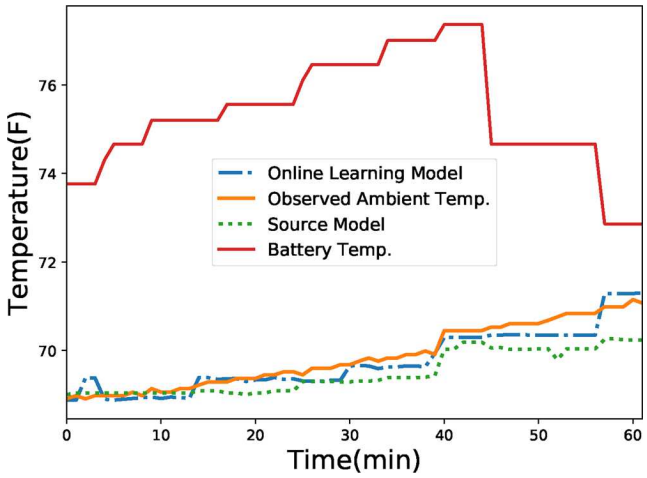

Fig. 7. Ambient temp. Predictions of source model trained on Alcatel Phone with mini-batch based transfer learning deployed on LG phone v/s pre-trained model on Alcatel Phone deployed on LG. Observed Ambient and Battery temp. are shown for reference.

stratified selection is important because the observed 4 sections of the curve may not have an equal number of datapoints hence, the distribution of data points in the training and testing datasets should be representative of the actual distribution. Additionally, we use a sampling rate of 1 datapoint/min during data collection because of a sampling rate of 1 datapoint/sec results in a lot of duplicate data points due to similar feature values. Reducing the sampling rate and a 20-20-60 split ensures no data leakage across the training and testing datasets. For the selection of model hyper-parameters, we use a grid search over the parameter space. We use MAPE as our evaluation metrics for comparison with state-of-the-art and other baseline regressors.

5.2. Baseline comparison

To evaluate the effectiveness of our model, we compare our proposed model with the state-of-the-art model [15], which is a Physics-based model based on Newton's law of cooling to compute the indoor ambient air temperature. To measure against the physics based model, we derive 7 models one for each of the most frequent combinations of the phone features (CPU, Network, Screen, and Charging) as described in [15] and predict the ambient air temperature on the test dataset. We run our trained model on the same dataset and we compute the percentage error for both the models across all depicted scenarios. Fig. 5 depicts the percentage error, shown as box plots for our model and the state-of-art model. Our model outperforms the state-of-art 6 times, with an average error of 0.48% while the state-of-art has an error of 3.32% a reduction of 2.84% points. Fig. 6 is the KDE of observed and actual observations of our model on the same dataset. Our model gives a prediction accuracy of 99.52%. As shown in Fig. 6 there is a very strong correlation between the observed and predicted temperatures.

5.3. Few shot based transfer learning

To evaluate the efficacy of our few shot based transfer learning technique, we train a model on data collected from Alcatel phone running Android 7 as the source model and deploy the model on an LG Stylo3 phone running Android 7. We collect first 10 samples on the LG Phone at the beginning of the warming state at the rate of 1 sample/min to create a mini-batch and the first 10 samples at the beginning of cooling state at the rate of 1 sample/min to create a second mini-batch for online training. We optimize the thresholds of the Decision Trees while maintaining the same tree structure as learned from the Alcatel phone using gradient descent with a very small learning rate of 0.001. As

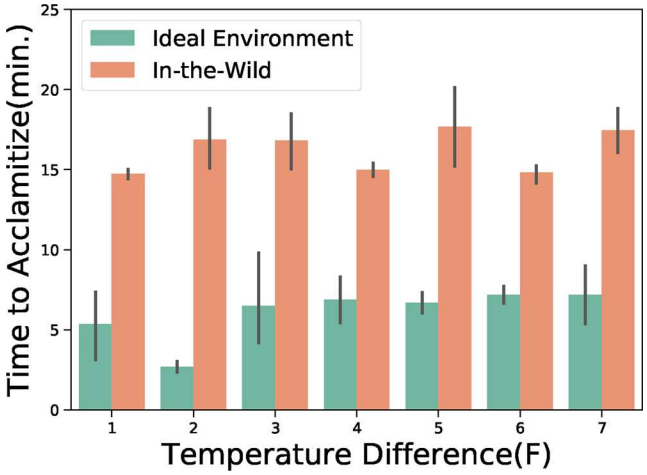

Fig. 8. Box plot showing acclimatization time of a phone after getting relocated to a new location with a different ambient temperature.

shown in Fig. 7 the few shot based transfer learning model has a lower error (MAPE 0.29) as compared to the error (MAPE 0.71) in prediction by the model trained on Alcatel phone and deployed on LG phone without any updates to the pre-trained model. The figure shows that the few shot based transfer learning model works well for an all the 4 sub-states of the curve using previously learned model trained on another phone model and trained with 10 samples from each of warming and cooling sub-state without having to collect full data for all the sub-states from a new phone.

5.4. Acclimatization time

Crowdsensing predicted ambient temperatures may result in varying qualities of ambient temperature predictions. Thus, determining the quality of the predicted ambient temperature is critical to our system. To compute a highly accurate temperature from the varying qualities of crowdsensed temperature values we formulate policies as presented in Section 4.1. To justify these policies, we conduct two experiments to empirically measure the acclimatization time taken by phones. We set up the first experiment in a controlled environment using a temperature chamber (Model: TestEquity 115A). The temperature chamber is a small enclosed space that provides an ideal environment where the temperature is very stable and uniform. We vary the temperature in a controlled fashion by increasing the temperatures in the range of 1–7 °F and measure the time taken by the predicted phone temperature to reflect the ambient temperature, referred to as the settling time of the phone. In the second experiment, we simulate user movements in the real-world environment by moving the phone to multiple indoor locations and measure the settling time. Though, the second experiment seems more realistic it has many uncontrollable factors. We conduct all experiments using the same Pixel phone with screen off, low CPU and network load and charging off.

As shown in Fig. 8, in an ideal environment the phone takes 2–8 min to converge to the true temperature and takes 14–20 min in the wild. The controlled chamber is an ideal setting and settling time is lower. In the real world, many uncontrollable factors impact settling time. Based on the need for phones to acclimatize we justify the need for the policy to select high and low quality measurements.

5.5. Crowdsourcing experiments

In this section, we evaluate the system end-to-end with 2 simulations. First, we crowdsense measurements from multiple phones at one location, with each phone having a different arrival time, and demonstrate the efficacy of our system to handle varying quality of

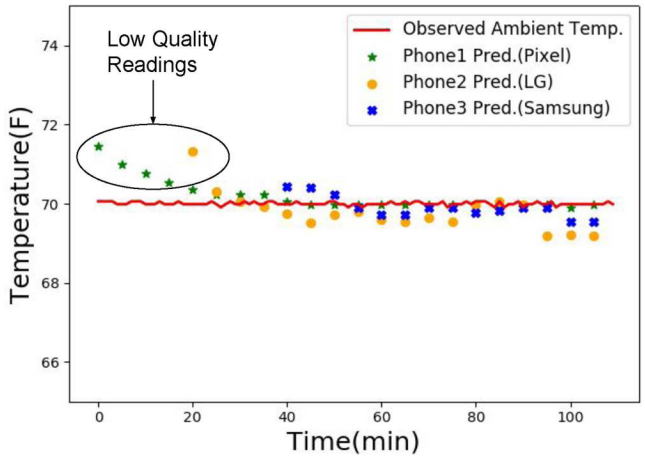

Fig. 9. Crowdsourced temperatures.

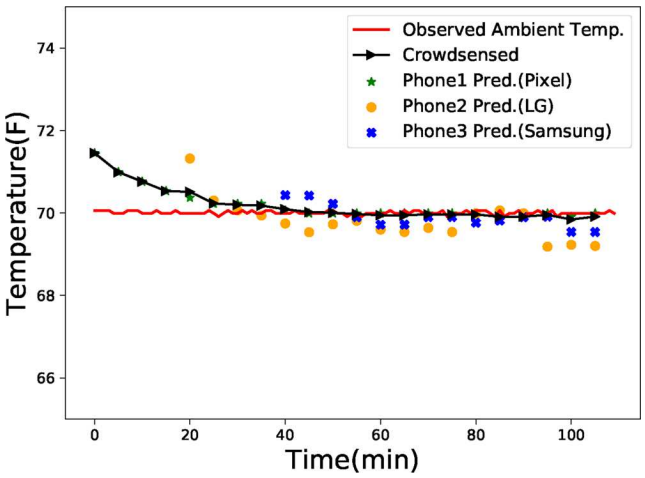

Fig. 10. Crowdsourced temperature prediction after policy evaluation.

crowdsensed data. Second, we crowdsense measurements from a single mobile phone at multiple locations (15 locations) and demonstrate the accuracy of predictions of the model under a high user mobility scenario.

5.6. Crowdsourced measurements of multiple phones at one location

In this experiment, we collect data from 3 mobile phones Google Pixel (Android 7.1), LG (Android 7), and Samsung (Android 7) at one location with different arrival times of the 3 phones at the location. We simulate a scenario where users arrive at a location (Loc1) at different times from different locations (source) and the temperature difference between source and Loc1 may be different. Pixel1 arrives first at Loc1. After 20 min the LG phone is placed at Loc1, and after another 20 min, Samsung phone is also placed at Loc1. Each phone makes a prediction every 5 min and the data is crowdsensed for the entire duration they are at Loc1.

Each of the three phones arrives at Loc1 from a different source location with a different temperature than Loc1. When the phones arrive at Loc1 and are in the settling period getting acclimatized the ambient temperature predictions may not be of high quality. Hence, crowdsensing predictions from all occupants at location Loc1 would result in a varying range and quality of predictions. Fig 9 shows the ground truth ambient temperature recorded at Loc1 using a HOBO (temperature logger) along with the predictions made by the phones at varying times the phone owners enter the room. As shown in the figure the initial

Fig. 11. Heatmap shows the prediction error at 4 locations. The average error over 4 locations is 1.25%.

predictions of ambient temperature made by the phones are low quality readings and as the phones get acclimatized to Loc1 their predictions converge to ground truth ambient temperature recorded at Loc1.

Next, to mitigate the impact of settling time and compute a single value of the ambient temperature at a location from the crowdsensed data, we apply the duration-based and device confidence policies. The duration-based policy weeds out the lower quality readings while the device confidence policy takes into account the historic temperature measurement accuracy of a phone. The device confidence of the three phones used for the crowdsourcing experiment are Pixel device confidence 95%, LG device confidence 65%, and Samsung device confidence 80%. We normalize the device confidence values based on the number of phones participating in crowdsensing data collection.

Fig. 10 shows the individual phone readings, observed ambient temperature, and crowdsensed ambient temperature derived after applying the policies. On applying the two policies to the crowdsensed data we find that for the initial 20 minutes when only Pixel phone is present the predicted crowdsensed policy temperature is the same as the predicted pixel phone temperature. Also, as the pixel phone acclimatizes the predicted ambient temperature gets closer to the actual observed ambient temperature. At 20 mins timeline LG phone – with lower prediction confidence – is introduced and though the LG phone shows low-quality readings initially the readings from pixel phones are given a higher weight based on the duration-policy. After 8 min when LG phone acclimatizes and now we have two ambient temperature predictions from Pixel and LG phones. We observe that duration and device confidence policies aid in computing highly accurate ambient temperature as seen in Fig. 10.

5.7. *Crowdsourced measurements of a single phone at multiple locations*

In this experiment, we place one phone at 4 different locations along with a temperature sensor, which measures the ground truth data. The phone predicts the ambient temperature and uploads the predictions to the server every 5 min for two hours. At each of the locations the phone screen was off, CPU and network load were low and charging was off. The result shows that our system can predict the temperature accurately with an average error of 1.25% across all locations as shown in Fig. 11. The higher errors were seen near the staircase with a door while the rest zones such as labs, discussion rooms, offices, and classrooms labs consistently showed an error of less than 0.72%.

6. Conclusions

In this paper, we presented an ambient temperature prediction approach using an on-device ML-based model and Opportunistic Crowdsensing. We presented a machine learning approach based on a random forest-based ensemble learning model that uses the phone battery temperature sensor to infer the ambient air temperature. We also propose a few-shot based transfer learning approach to deploy a pre-trained model on new phones. This approach is highly effective in reducing data collection and training overhead. We also presented a full prototype of our system implemented as a RESTful web service to crowdsense in-door temperature measurements in conjunction with an indoor positioning system (IPS) and designed policies to distinguish high-quality crowdsourced measurements from lower quality ones. Finally, we evaluated our approach using multiple devices under different scenarios and demonstrated the efficacy of our approach.

Acknowledgements

This research is supported by NSF grants 1763834, 1645952, and ARL W911NF-17-2-0196.

References

[1] J.R. Kwapisz, G.M. Weiss, S.A. Moore, Activity recognition using cell phone accelerometers, ACM SigKDD Expl. Newslett. 12 (2) (2011) 74–82.
[2] B. Zhou, J. Yang, Q. Li, Smartphone-based activity recognition for indoor localization using a convolutional neural network, Sensors 19 (3) (2019) 621.
[3] Y.-J. Hong, I.-J. Kim, S.C. Ahn, H.-G. Kim, Mobile health monitoring system based on activity recognition using accelerometer, Simul. Model. Pract. Theory 18 (4) (2010) 446–455.
[4] D.D. Luxton, R.A. McCann, N.E. Bush, M.C. Mishkind, G.M. Reger, mHealth for mental health: integrating smartphone technology in behavioral healthcare, Prof. Psychol.: Res. Pract. 42 (6) (2011) 505.
[5] E. Kanjo, Noisespy: a real-time mobile phone platform for urban noise monitoring and mapping, Mob. Netw. Appl. 15 (4) (2010) 562–574.
[6] N. Maisonneuve, M. Stevens, M.E. Niessen, P. Hanappe, L. Steels, Citizen noise pollution monitoring, Proceedings of the 10th Annual International Conference on Digital Government Research: Social Networks: Making Connections Between Citizens, Data and Government (2009) 96–103.
[7] A. Overeem, J. Robinson, H. Leijnse, G.-J. Steeneveld, B. Horn, R. Uijlenhoet, Crowdsourcing urban air temperatures from smartphone battery temperatures, Geophys. Res. Lett. (2013).
[8] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, J. Eriksson, VTrack: accurate, energy-aware road traffic delay estimation using mobile phones, Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (2009) 85–98.
[9] J. Biagioni, T. Gerlich, T. Merrifield, J. Eriksson, Easytracker: automatic transit tracking, mapping, and arrival time prediction using smartphones, 9th International Conference on EmbeddedNetworked Sensor Systems (2011) 446–455.
[10] P. Zhou, Y. Zheng, M. Li, How long to wait?: predicting bus arrival time with mobile phone based participatory sensing, Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (2012).
[11] K. Ali, D. Al-Yaseen, A. Ejaz, T. Javed, H.S. Hassanein, Crowdits: crowdsourcing in intelligent transportation systems, IEEE Wireless Communications and Networking Conference (WCNC) (2012) 3307–3311.
[12] T. Yan, B. Hoh, D. Ganesan, K. Tracton, T. Iwuchukwu, J.-S. Lee, Crowdpark: A Crowdsourcing-Based Parking Reservation System for Mobile Phones, 2011. University of Massachusetts at Amherst Tech. Report.
[13] Energy Information Association FAQ. https://www.eia.gov/tools/faqs/faq.php?id=86; 2019.
[14] A. Trivedi, J. Gummeson, D. Irwin, D. Ganesan, P. Shenoy, iSchedule: campus-scale HVAC scheduling via mobile WiFi monitoring, Proceedings of the Eighth International Conference on Future Energy Systems (2017) 132–142.
[15] J. Breda, A. Trivedi, C. Wijesundara, P. Bovornkeeratiroj, D. Irwin, P. Shenoy, J. Taneja, Hot or not: leveraging mobile devices for ubiquitous temperature sensing, Proceeding of ACM Buildsys, New York, NY (2019).
[16] J.D. Kelso, Buildings Energy Data Book, Department of Energy, 2012.
[17] K. Saurav, M. jain, S. Bandhyopahyay, Reducing energy consumption for space heating by changing zone temperature: pilot trial in Lulea, Sweden, 9th ACM International Conference on Future Energy Systems (e-Energy '18) (2018).
[18] Comfy Inc, Comfy App, 2019. www.comfyapp.com.
[19] S. Nagarathinam, A. Vasan, V. Sarangan, R. Jayaprakash, A. Sivasubramaniam, Good set-points make good neighbors – user seating and temperature control in uberized workspaces, 5th ACM Int'l Conference on Systems for Energy-Efficient Built Environments (BuildSys '18) (2018).
[20] D. Pisharoty, R. Yang, M. Newman, K. Whitehouse, ThermoCoach: reducing home energy consumption with personalized thermostat recommendations, 2nd ACM Int'l Conference on Systems for Energy-Efficient Built Environments (BuildSys '15) (2015).

[21] A. Grush, AndroidAuthority WeatherSignal Turns Your Phone into a Personal Weather Station, 2013. https://www.androidauthority.com/weathersignal-app-206232/.
[22] I.G. Martí, L.E. Rodríguez, M. Benedito, S. Trilles, A. Beltrán, L. Díaz, J. Huerta, Mobile application for noise pollution monitoring through gamification techniques, in: International Conference on Entertainment Computing, Springer, 2012, pp. 562–571.
[23] M. Stevens, E. D'Hondt, Crowdsourcing of pollution data using smartphones, Workshop on Ubiquitous Crowdsourcing (2010) 1–4.
[24] P. Mohan, V.N. Padmanabhan, R. Ramjee, Nericell: rich monitoring of road and traffic conditions using mobile smartphones, Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (2008) 323–336.
[25] A. Artikis, M. Weidlich, F. Schnitzler, I. Boutsis, T. Liebig, N. Piatkowski, C. Bockermann, K. Morik, V. Kalogeraki, J. Marecek, et al., Heterogeneous stream processing and crowdsourcing for urban traffic management, EDBT, vol. 14 (2014) 712–723.
[26] D. Vij, N. Aggarwal, Smartphone based traffic state detection using acoustic analysis and crowdsourcing, Appl. Acoust. 138 (2018) 80–91.
[27] L. He, Y. Lee, K.G. Shin, Mobile device batteries as thermometers, Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 4 (2020) 1–21.
[28] J. Wiese, S. Saponas, A. Brush, Phoneprioception: enabling mobile phones to infer where they are kept, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2013).
[29] J. Yang, E. Munguia-Tapia, S. Gibbs, Efficient in-pocket detection with mobile phones, UbiComp'13 (2013).
[30] Y. Chen, J.-A. Francisco, W. Trappe, R.P. Martin, A practical approach to landmark deployment for indoor localization, In 2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks, vol. 1 (2006) 365–373.
[31] C. Yang, H.-R. Shao, WiFi-based indoor positioning, IEEE Commun Mag 53 (3) (2015) 150–157.
[32] L. Schauer, F. Dorfmeister, M. Maier, Potentials and limitations of WiFi-positioning using time-of-flight, International Conference on Indoor Positioning and Indoor Navigation (2013) 1–9.
[33] Compulab, WILD, 2019. https://fit-iot.com/web/products/wild/.
[34] K. Georgiou, T. Constambeys, C. Laoudias, L. Petrou, G. Chatzimilioudis, D. Zeinalipour-Yazti, Anyplace: a crowdsourced indoor information service, in: 2015 16th IEEE International Conference on Mobile Data Management, vol. 1, IEEE, 2015, pp. 291–294.
[35] M.R. Znaidi, G. Gupta, K. Asgari, P. Bogdan, Identifying arguments of space-time fractional diffusion: data-driven approach, Front. Appl. Math. Stat. 6 (2020) 14, https://doi.org/10.3389/fams.2020.00014.