Learning the micro-environment from rich trajectories in the context of mobile crowd sensing

Hafsa El Hafyani, Mohammad Abboud, Jingwei Zuo, Karine Zeitouni, Yehia Taher, Basile Chaix, Limin Wang

Noname manuscript No.
(will be inserted by the editor)

Learning the Micro-environment from Rich Trajectories in the context of Mobile Crowd Sensing

Application to Air Quality Monitoring

Hafsa El Hafyani · Mohammad Abboud · Jingwei Zuo · Karine Zeitouni · Yehia Taher · Basile Chaix · Limin Wang

Received: date / Accepted: date

Abstract With the rapid advancements of sensor technologies and mobile computing, Mobile Crowd Sensing (MCS) has emerged as a new paradigm to collect massive-scale rich trajectory data. Nomadic sensors empower people and objects with the capability of reporting and sharing observations on their state, their behavior and/or their surrounding environments. Processing and mining multi-source sensor data in MCS raise several challenges due to their multi-dimensional nature where the measured parameters (i.e., dimensions) may differ in terms of quality, variability, and time scale. We consider the context of air quality MCS and focus on the task of mining the micro-environment from the MCS data. Relating the measures to their micro-environment is crucial to interpret them and analyse the participant's exposure properly. In this paper, we focus on the problem of investigating the feasibility of recognizing the human's micro-environment in an environmental MCS scenario. We propose a novel approach for learning and predicting the micro-environment of users from their trajectories enriched with environmental data represented as multidimensional time series plus GPS tracks. We put forward a multi-view learning approach that we adapt to our context, and implement it along with other time series classification approaches. We extend the proposed approach to a hybrid method that employs trajectory segmentation to bring the best of both methods. We optimise the proposed approaches either by analysing the

Hafsa El Hafyani · Mohammad Abboud · Jingwei Zuo · Karine Zeitouni · Yehia Taher
DAVID Lab
UVSQ-Université Paris-Saclay
Versailles, France
E-mail: firstname.lastname@uvsq.fr

Basile Chaix · Limin Wang
emesis Research Team, INSERM
Institut Pierre Louis d'Épidémiologie et de Santé Publique
Paris, France
E-mail: firstname.lastname@iplesp.upmc.fr

exact geolocation (which is privacy invasive), or simply applying some a priori rules (which is privacy friendly). The experimental results, applied to real MCS data, not only confirm the power of MCS and air quality (AQ) data in characterizing the micro-environment, but also show a moderate impact of the integration of mobility data in this recognition. Furthermore, and during the training phase, multi-view learning shows similar performance as the reference deep learning algorithm, without requiring specific hardware. However, during the application of models on new data, the deep learning algorithm fails to outperform our proposed models.

Keywords Mobile Crowd Sensing · Trajectory Segmentation · Activity Recognition · Multivariate Time Series Classification · Multi-view Learning · Air Quality Monitoring

1 Introduction

Nowadays, the Internet of Things (IoT) basically relies on advanced sensor technologies to bridge the physical world and information systems. In particular, along with the widespread use of GPS, various mobile sensors bring rich information collected from both the surrounding environment and human activities, which are generally represented as geo-referenced time series (i.e. trajectories enriched with several measures). Mobile Crowd Sensing (MCS) [23] emerges as a new paradigm which empowers volunteers to contribute data acquired by their personal sensor-enhanced mobile devices. Polluscope[1], which is a French project deployed in Île-de-France (i.e., Paris region), is a typical use case study based on MCS. It aims at getting insights constantly on individual exposure to pollution everywhere (indoor and outdoor) while enriching the traditional monitoring system with the collected data by the crowd. The recruited participants, on a voluntary basis, collect air quality (AQ) measurements. Each participant is equipped with a sensor kit and a mobile device which allows the transmission of the collected measurements together with the GPS coordinates as a geo-referenced data stream containing (timestamp, longitude, and latitude). In addition, the participants are asked to annotate their environment type through a custom mobile application. This will allow participants to have personalized insights about their exposure to pollution everywhere, either in indoor or outdoor environments, and at a higher resolution along their trajectories; thereby, allowing to capture local variability and peaks of pollution, depending on participants' whereabouts, i.e., *micro-environments*.

It is worth mentioning that air quality strongly depends on the micro-environment, and so is the individual exposure to pollution. For this reason, there is a great interest in making exposure analysis micro-environment-aware. Beyond that, ignoring the micro-environment would make the data collection useless, precisely because of the influence of the micro-environment. However,

[1] http://polluscope.uvsq.fr

the micro-environment annotation is by far the most difficult information to collect in a real-life application setting since very few participants thoroughly annotate their micro-environment. Therefore, there is a great interest in unburdening the participants by automatically detecting the micro-environment.

The problem of automatically annotating MCS data can be seen as a problem of activity recognition from rich trajectory data collected by heterogeneous sensors. There is a broad variability of research studies on the subject of activity recognition. The survey by Yu Zheng [54] proposes a systematic review of the major research in trajectory mining. Whilst the author provides a variety of trajectory data mining methods, an overall approach that combines several sensor data besides GPS data is missing. In contrast, combining several sensory data suggests the usage of multivariate time series classification (MTSC) for activity recognition. Although this solution showed excellent performance in some application domains [37], its success is not guaranteed with heterogeneous sensors such as environmental data. First, the usage of heterogeneous data may induce some missing data problems when some sensors stop working. Therefore, there is a need for a model that characterises the micro-environments even with missing dimensions. Second, it is not known to what extent environmental data can determine micro-environments, which needs to be investigated.

As a matter of fact, when visually exploring the data, we noticed that micro-environments preserve a certain pattern. Moreover, we observe the existence of an inter-sensor correlation and with the micro-environment. Figure 1 shows the evolution of three dimensions (i.e. Black Carbon (BC), NO2 and Particulate Matters (PM)) with micro-environments identification. As shown in Figure 1, BC and NO2 preserve the same shapes and statistical characteristics in the micro-environment "car". Specifically, BC maintains the same fluctuations pattern in the micro-environment "car" and conserves approximately the same average value in these segments. Likewise, NO2 fluctuates promptly and preserves roughly the same average value in both segments of the micro-environment "car" as well as approximate maximum values. We also note that NO2 values keep roughly the same pattern in the micro-environment "indoor". Moreover, we can observe the existence of a correlation between the three dimensions during the whole timeline, meaning that when one of the dimensions fluctuates, the other two follow.

The idea we promote in this paper is to utilize a wisely chosen annotated dataset in order to train a model on the acquired rich trajectories (composed of both environmental and mobility dimensions) as predictors of the micro-environment. We hypothesize that the multivariate time series collected by the MCS campaigns not only depends on the micro-environment but could be a proxy of it.

The question that arises now is how to combine all these different heterogeneous aspects of the data (geo-location, sensors) to identify the user's micro-environment automatically, and how much a model can discriminate the observations in different micro-environments.

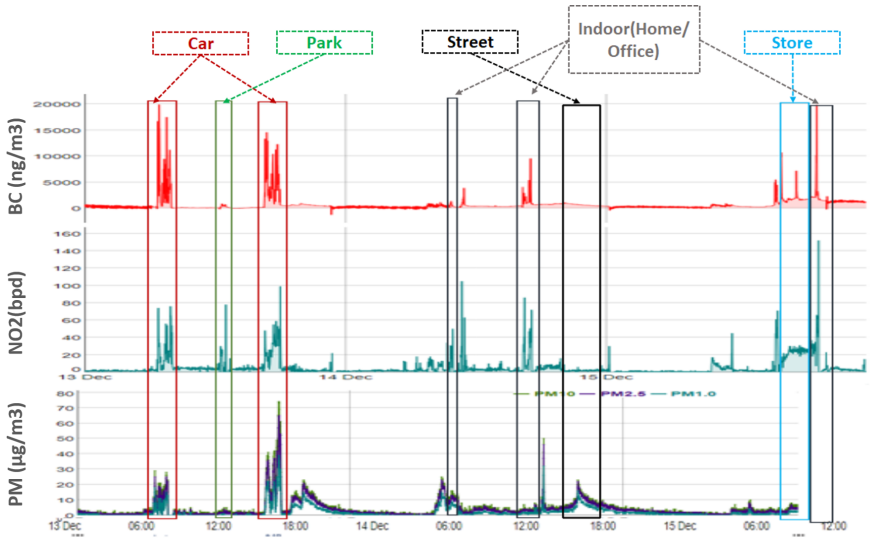


Fig. 1: Inter-sensor and micro-environment correlations.

In this paper, we evaluate different approaches and provide a framework dedicated to the preparation, the application, and the comparison of different machine learning algorithms. Precisely, we make the following contributions:

- first and foremost, we identify the problem of micro-environment recognition in the MCS context.
- we demonstrate that AQ determines the type of micro-environment.
- thereafter, we propose a ML approach based on multi-view learning for the recognition of micro-environment.
- afterwards, we extend the proposed micro-environment recognition approach to include another layer which consists of the detection of stay locations (i.e. stop detection) and transportation modes from GPS data; also known as trajectory segmentation. We refer to this extension as the hybrid approach.
- we optimise the proposed approaches either by analysing the exact geolocation, or simply applying some a priori rules. We emphasize that the first optimisation is privacy invasive whilst the other one is privacy friendly.
- we conduct extensive experiments in a real scenario setting and compare with baselines, which shows the effectiveness of our proposed approaches.

In the current work, Polluscope data is considered as a running application example based on MCS. However, the proposed approach can be generalised on other MCS applications besides the AQ scenarios. In fact, the same process of activity recognition of moving objects holds with other sensory data such as sound for noise sensing or temperature for heat comfort assessment. For instance, the collaborative AIRLESS project between Cambridge and Beijing, which is a typical use case study based on MCS, aims to understand the impact

of air pollution on human health in the world's largest country. One important element of AIRLESS is to automatically detect and classify major exposure-related micro-environments (home, work, other static, in-transit) using GPS coordinates, accelerometry, and noise. The classification of micro-environment can remarkably improve exposure metrics since pollutant inhalation rates vary significantly by location and micro-environment [7].

The rest of this paper is organized as follows. We introduce the related work in Section 2. Section 3 gives a thorough problem description. Section 4 describes the multi-view learning approach. The presentation of our micro-environment recognition model is discussed in Section 5 and 6. Section 7 presents the experimental results and evaluation of the micro-environment recognition model in the context of environmental crowd sensing. Section 6 gives an extensive discussion for the perspectives of this work. In section 7, we summarize our conclusions and provide directions for future work.

2 Related Work

Human activity recognition (HAR) has gained, in the recent years, a great interest from the research community. Its application domains encompasses all the activities performed within daily human activities [51][30][10] and human mobility [14][55] to cite a few. It represents a typical scenario of machine learning, and some public datasets are widely used in the benchmarks.

In this section, we review some activity recognition research designs that can be employed to infer the human activity from multivariate time series and GPS trajectories. We break the problem of activity recognition down into three categories, i.e. multivariate time series classification (MTSC), multi-view learning as another MTSC solution, and trajectory segmentation to infer the stop and move segments from GPS data.

2.1 Multivariate Time Series Classification

Human activity recognition falls in the problem of labelling data segments with the type of activity which leads to a multivariate time series classification (MTSC) problem based on data collected by multiple wearable sensors. There is a wide range of time series classification approaches that can be classified into four categories: distance-based methods [4], feature-based methods [35], ensemble methods [21], and deep learning models [19][9][46]. The one-nearest neighbor (1-NN) classifier with different distance measures, such as euclidean distance (ED) or dynamic time wrapping (DTW) [4], is always considered as the benchmark to give a preliminary evaluation in the MTSC problem.

Considering the real-life scenarios, where it is difficult or expensive to obtain a large amount of labeled data for training, some studies used both labeled and unlabeled data to learn the human activity, that is semi-supervised learning (SSL) [47] on MTSC. The pioneering work by [47] propose a semi-supervised technique for time series classification. The authors demonstrated

that semi-supervised learning requires less human effort and generally achieves higher accuracy than training on limited labels. The semi-supervised model [47] is based on the self-learning concept with the one-nearest-neighbor (1-NN) classifier. First, the labeled set denoted by P (as positively labeled) is applied to train the 1-NN classifier C. Then, the unlabeled samples U are given the pseudo labels progressively based on their distance to the samples in P. Thereafter, the enriched labeled set P allows iteratively repeating the previous step and improving the classifier. More recently, the deep learning-based models on MTSC show promising performance under weak supervision. For instance, Zhang et al. [53] propose a novel semi-supervised MTSC model named time series attentional prototype network (TapNet) to explore the valuable information in the unlabeled samples. TapNet projects the raw MTS data into a low-dimensional representation space. The unlabeled samples approach themselves to the class prototype in the representation space, where pseudo labels are generated by the distance-based probability allowing training the model progressively. Moreover, the hybrid convolutional neural network (CNN) and long short-term memory (LSTM) structure adopted in TapNet allows modeling, respectively, the variable interactions and the temporal features of MTS.

2.2 Multi-View Learning

Another line of studies propose multi-view learning to classify time series data originated from multiple sensors to recognize user activities. Garcia-Ceja et al. [21] propose a method based on multi-view learning and stacked generalization for fusing audio and accelerometer sensor data for human activity recognition using wearable devices. Each sensor's data is seen as a different "view", and they are combined using stacked generalization [48]. The approach trains a specific classification model over each view and an extra meta-learner using the view models as input. The general idea of the authors is to combine data from heterogeneous types of sensors to complement each other and thus, increase recognition accuracy.

Wang et al. [45] propose a framework based on deep learning to learn features from different aspects of the data based on features of sequence and visualization. In order to imitate the human brain, which can classify data based on visualization, the authors transform the time series into an area graph. Area graph here is used to model time series as images in order to apply Convolutional Neural Network (CNN) on top of it. They use well-trained Long short-term memory with an attention mechanism (LSTM-A) neural networks and CNN with attention (CNN-A) to extract the features of time series data. LSTM-A extracts sequence features, while CNN-A extracts visual features from the time series. Then, based on the fusion of features, the authors carry out the time series classification task. Although the approach gained promising results, further performance gain was achieved by recent deep learning methods such as InceptionTime [20].

Li et al. [28] propose a multi-view discriminative bilinear projections (MDBP) for multi-view MTSC. The proposed approach is a multi-view dimensionality reduction method for time series classification, which aims to extract discriminative features from multi-view MTS data. MDBP mainly projects multi-view data to a shared subspace through view-specific bilinear projections that preserve the temporal structure of MTS, and learns discriminative features by incorporating a novel supervised regularization.

2.3 Trajectory Segmentation

There has been a substantial increase in spatial and spatio-temporal trajectory data due to the advances in GPS-based tracking and mobile computing techniques. Plenty of research has been proposed to shape the field of trajectory data mining. In a survey paper, Zheng [54] provides a systematic review of the major research in the trajectory data mining field in order to thoroughly explore the existing techniques. The author proposed a methodical framework that comprises a list of trajectory data mining methods from the derivation of trajectory data to a variety of mining tasks (such as trajectory classification) and passing by trajectory data processing (such as trajectory segmentation) and trajectory data management.

Rehrl *et al.* [36] propose and evaluate a three-steps trajectory data mining approach based on machine learning techniques. The authors focus on detecting and classifying stop points in vehicle trajectories. The proposed approach describes three mining steps: stop detection, feature extraction, and stop segments classification. The authors first segment the trajectory into stay points clusters (referred to as trajectory segmentation). After extracting 14 characteristics of each stop, they classify the detected stops into traffic-relevant and non-traffic-relevant stops (referred to as trajectory classification). This work is most similar to ours, but it is based on spatial data and does not include temporal data.

As for the field of feature extraction, the research community has provided several extensive works based on *machine learning* techniques. Etemad *et al.* [18] provide a framework for the prediction of transportation mode based on GPS data only. The key contribution of the authors' work is to propose trajectory point features generation and trajectory segments feature extraction which comprise bearing rate, the change rate of the bearing rate, and the global and local trajectory features. The usage of these features has shown promising performances. Instead of using hand-crafted features with traditional machine learning algorithms, Dabiri et al. [11] propose a travel mode inference model based on convolutional neural network (CNN) schemes that can automatically drive high-level features from raw input, which attains the state-of-the-art accuracy on GPS data from GeoLife dataset [56].

Another line of work focuses on the detection of stops and moves within GPS tracks deriving more semantic trajectories [34]. Pappalardo *et al.* [33] propose a Python library for the analysis of mobility data. In addition to

addressing different problems such as spatio-temporal trajectories cleaning, trajectory visualization, and privacy risks, the authors provide an environment to reproduce existing research related to the detection of *Stay Points* or *Stops*. One of the common approaches is to apply spatial clustering algorithms on trajectory points based on their spatial proximity. Thus, the authors propose a `stops` function that finds the stay points visited by a moving object based on the spatial proximity and time spent by the object. Although the library has shown promising performance results, it uses GPS logs only as input and does not take into consideration the temporal data. More advanced approaches derive users' habits in terms of visiting patterns from trajectories, to cite but a few [40], [13], and some recent surveys [44].

We highlight that state-of-the-art that is related to micro-environment's recognition also focuses on data from sound and accelerometer [3], [46]. However, since this information is privacy-invasive and we do not use sound and accelerometer in our context, we will not focus on their related work. Furthermore, work proposals such as Sonawani *et al.* [42], and Sai *et al.* [39] whose main objective is monitoring AQ using machine learning methods, are out of the scope of our study. The main focus of such proposals is AQ forecasting and estimation, while our main objective is the detection of micro-environments from AQ data plus GPS. Therefore, we will not focalize on investigating state-of-the-art studies related to AQ estimation and forecasting.

To summarize, the study of related work of micro-environment recognition are either based on multivariate time series classification without any concern or attention to trajectory data to classify the label of the micro-environment, or related to stop and move detection in trajectory data where the label of the micro-environment is reduced to stop/move (where stops are indoor and moves are outdoor, in general). Transportation mode detection approaches can reveal the label of the move segments but not indoor segments. In contrast, to our best knowledge, no previous approaches that combine environmental time series and trajectory data in the micro-environment recognition problem have been proposed so far, which is the focus of the current work.

3 Problem formalization

Before detailing the methodology of the proposed approach, we start with a thorough description of the problem.

3.1 What are rich trajectories ?

In the context of MCS, the collected type of data are typically continuous sensors measures along with the participant's spatial location (e.g., GPS coordinates). They represent a specific type of trajectories that we call rich trajectories. We define hereafter this concept, starting from the definition of time series.

Definition 1 (Univariate Time Series). A univariate time series is a sequence $U = [(t_1, v_1), ..., (t_l, v_l)]$ where l is the length of U and for $i = 1...l$, $t_i \in T$ is a timestamp from a time domain T and $v_i \in D$ is a scalar value of a domain D.

Example 1 *Environmental sensor measurements such as temperature constitute a univariate time series.*

Definition 2 (Multivariate Time Series). A multivariate time series MV is defined as $MV = (U_1, U_2, ..., U_i, ..., U_n)$ where U_i is a univariate time series for dimension D_i, and $i = 1, ..., n$.

Example 2 *Environmental sensor measurements such as temperature, humidity and NO2 constitute a 3-Dimensional time series.*

Definition 3 (Trajectory). A trajectory T is defined as a multivariate time series with two or three dimensions for the spatial position.

Example 3 *A multivariate time series with latitude and longitude as dimensions represent a trajectory.*

Definition 4 (Rich Trajectory). A rich trajectory RT is defined as a multivariate time series where a subset of the dimensions D_i where $i \in [1, ..., n]$ constitutes a spatial position, plus additional non-spatial informational.

Example 4 *A GPS trajectory data of a moving object associated with environmental sensor measures such as temperature, humidity and NO2 is a typical example of a rich trajectory.*

3.2 What is micro-environment recognition ?

First, we define a trajectory segmentation, then we introduce the annotated version of rich trajectories before defining the target problem of micro-environment recognition learning.

Definition 5 (Rich Trajectory Segment). A rich trajectory segment RTS is defined as a sub-sequence of contiguous vectors of RT between j and k $(1 \leq j \leq k \leq l)$. So, $RTS = RT(j, k) = (U'_1, U'_2, ..., U'_i, ..., U'_n)$ where $U'_i = [(t_{ij}, v_{ij}), ..., (t_{lk}, v_{lk})]$, and $\forall 1 \leq i \leq n$.

Example 5 *A one hour trajectory constitutes a rich trajectory segment of a one week rich trajectory data.*

Definition 6 (Trajectory Segmentation). Given a trajectory or a rich trajectory as input, trajectory segmentation is a process that splits it into non overlapping trajectory segments.

Example 6 *Splitting trajectory data of a moving object into hourly segments represent a one form of trajectory segmentation.*

An annotated rich trajectory is defined as a sequence of trajectory segments along with annotations that belong to a predefined list of categories. Formally:

Definition 7 (Annotated Rich Trajectory). An annotated rich trajectory ART is defined as a sequence of couples $ART = [(RT(1, i_1), a_1), (RT(i_1, i_2), a_2), ..., (RT(i_j, i_{j+1}), a_{j+1}), ..., (RT(i_p, l), a_{p+1})]$, where $RT(i_j, i_{j+1})$ are rich trajectory segments RTS between j and $j + 1$, $a_k \in A$, and A is a discrete domain.

Example 7 *Rich trajectory segments enriched with contextual information such as the whereabouts of a moving object represent an annotated rich trajectory.*

In this work, annotations describe the micro-environment of the participant. In this work, micro-environments can either be an indoor space (e.g. home, office, restaurant, etc.), outdoor space (e.g. street, park, etc.) or a transportation mode (e.g. metro, bus, car, etc.). The micro-environment recognition question relates to the problem of segmenting data and assigning a label to each segment by combining every available data.

Definition 8 (Micro-environment Recognition). Given a rich trajectory RT as input, micro-environment recognition is a process that outputs the corresponding annotated rich trajectory ART.

Definition 9 (Micro-environment Recognition Learning). Given a set of annotated rich trajectories, train a model where the rich trajectory segments are the predictors, and the annotations constitute the class labels.

Using a trained model on a wisely chosen annotated dataset, we aim at predicting the annotation on a completely unseen data by the model.

Why is this information of the micro-environment important ? The annotation information of a moving object enables to understand rich trajectory on a semantic level. Its usefulness depends on the application scenario. For instance, in the case of peoples' trajectories, the semantic information can be used to identify the most visited places by the moving object and therefore to offer trip recommendations [57]. In the case of daily human activity recognition from wearable sensors, several application domains exist including, but not limited to, pervasive healthcare [52] [24] and following athletic activities [26]. In these applications, the annotation, described by daily activities such as walking, standing up, and raising hand, is fundamental to give feedback about the application scenario.

Specifically, and in our case, the information of annotation (which micro-environments here) allows us to give feedback about personal exposure to pollution, since it is directly correlated to people's habits and where they spend most of their time. For instance, if a person is highly exposed in their home during cooking time without much room ventilation, it would be time for them to revisit their habits and start ventilating the room when cooking. Therefore, the information of micro-environment is necessary to correctly interpret the

collected AQ data, get insight on the individual exposure, and for a participant, adapt her behavior to reduce her exposure.

While there are several works related to activity recognition from spatio-temporal trajectory (e.g. Sardianos *et al.* [40] and Toch *et al.* [44]), this work investigates the capability of environmental data in characterizing and inferring automatically the activity of the moving object. Therefore, we envision to combine every available information (i.e. AQ data, mobility data, declared annotations) to detect efficiently the micro-environment of the moving object.

3.3 How can micro-environments be recognised ?

Micro-environments can mainly be characterised by the temporal attributes (i.e. AQ measures) as well as the spatial one. There are several works for activity recognition that are either based on geographical or temporal information. However, an overall methodological approach for combining these different aspects on real-world complex trajectory data is missing. This combination may lead to a more robust detection model rather than the usage of a single attribute, and it needs to be investigated.

To employ every available facet of the rich trajectories, the design of the micro-environment recognition model needs to integrate two layers: a geographic layer and a multivariate time series layer. The geographic layer may infer the stop and move segments (aka *trajectory segmentation*) from GPS tracks only. This layer can go further and discover the location of home and work. The second layer of the multivariate time series may detect the exact label of segments (e.g. *home*, *office*, *store*, *metro*, *park*, etc.). Usually, this problematic leads to a multivariate time series (MTS) classification with AQ data as input and the detected micro-environments as output. However, MCS data is characterised by its heterogeneous property, designating that data is originated from different sensor readings. In fact, some sensors may be offline and do not transfer any data for hours, which may lead to missing data problems. Therefore, the usage of MTS classification is not straightforward. Naturally, it is necessary to design a model that combines data from heterogeneous sensors, and has the ability to classify it efficiently even if one (or more) dimension is missing.

Furthermore, in real-world settings, problems such as imbalanced data occur. For instance, we observe that the predominant labels are home and work since people spend most of their time there. Thereby, most segments are naturally mistaken by the model as home or work. Therefore, the proposed model should take into consideration all these aspects of the data and be efficient and robust enough to overcome these challenges. This model is explained further in Section 5.

Table 1: An example of the new generated dataset D'.

First-Level Learners						**Associated Prediction Probabilities**						**True Label**
l_1	l_2	...	l_i	...	l_n	p_1	p_2	...	p_i	...	p_n	y

4 Multi-view Learning Model

In this section, we present the approach of multi-view learning with stacked generalisation. We followed the proposal of Garcia-Ceja *et al.* [21], and adapted it to best fit for solving our problem.

It is not unusual to have applications in which heterogeneous types of sensors (e.g. accelerometers, gyroscopes) are involved for activity recognition. One way to deal with this problem is to extract features from each sensor and aggregate them to build the final classification model. However, this approach is not optimal since each sensor has its own statistical properties. Hence the idea of multi-view stacking to fuse data from heterogeneous sensors.

The multi-view paradigm consists of learning a model based on the different views of the data. The key idea is to consider each source of data independently and fuse them with *stacked generalization* (also called *stacking*), which is a type of ensemble method [58] for combining multiple learners.

The overall process is described as follows:

1. The first step consists of defining the set of first-level learner and *meta learner*.
2. Train the first-level learner on each view of the original data.
3. Predict the labels of each view using the first-level learner. Each view will produce a prediction vector with associated prediction probabilities.
4. Form a new matrix by column binding the prediction vectors and the true labels. This matrix forms the new training data D' for the meta-learner.
5. Train the meta-learner with D'.
6. Generate the final multi-view stacking model.

From an abstract view, assuming that Y_{it} is a dimension of the n-dimensional time series $Y_t = (Y_{1t}, Y_{2t}, ..., Y_{it}, ..., Y_{nt})$, each view V_i, where $V = (V_1, V_2, ..., V_i, ..., Vn)$ is the set of views, represents a dimension Y_{it} of the multivariate time series Y_t. Thus, we have as many views as dimensions.

The first-level learner takes as input the time series values coming from each view. Then, each view will generate its own predicted labels with associated prediction probabilities with the form $[l_i, p_1, p_2, ..., p_j, ..., p_k, y]$, where l_i is the predicted label of the first-level learner i, p_j is the associated prediction probability for each class j of the k possible classes, and y is the true label.

A new dataset D' is then created by column binding the output of each view and the true labels. We remind that these outputs consist of the predicted labels and the associated prediction probabilities for each of the k possible classes. Thus D' has the form shown in Table 1, where l_i is the predicted label

of the first-level learner i, p_i is the probability of this prediction, and y is the true label.

After generating a new dataset D', a second-level classifier, or *meta-learner*, is trained over D' through ensemble learning [58]. This approach allows to preserve the statistical properties of each view and learn the classes of the MTS instances with a significant improvement in the classification accuracy.

Many ensemble methods [58] have been proposed to further enhance the algorithm accuracy by combining learners rather than trying to find the best single learner. Due to their versatility and flexibility, ensemble methods attract many researchers and can be applied in different domains including, but not limited to, time series classification [21] and time series segmentation [16]. In a previous work [16], we used a multi-view approach for segmenting MCS data where we employed an unsupervised learning for change detection on each view.

5 Micro-environment recognition model

In this section, we provide an overview of our proposed framework for micro-environment recognition in the context of MCS [1].

Figure 2 provides a panorama of the steps followed to achieve the micro-environment recognition objective. It shows a roadmap from the derivation of air quality and trajectory data (i.e. step 1), to data preparation (i.e. step 2) which produce data ready to be consumed by a univariate time series classification model (e.g. kNN-DTW, LSTM, random forest, decision tree, etc.) (i.e. step 3). The outputs of the univariate time series classification constitute a new data set (i.e. step 4) which serves as an input for a meta-learner (i.e. step 5). The meta-learner produces the final classification results. In the following sections, we discuss each step separately. It is necessary to mention that the red dashed lines represent the hybrid approach, which we thoroughly discuss in Section 6.

5.1 Data Collection

The first step of the micro-environment recognition process is the data collection. During three campaigns, more than one hundred participants have been recruited to collect environmental measurements along with geo-location for one week, 24 hours a day, while performing their daily activities. Each participant carries a multi sensor box and a tablet empowered with a GPS chipset. The sensors collect time annotated measurements of Particulate Matter (PM1.0, PM10, PM2.5), nitrogen dioxide (NO_2), Black Carbon (BC), plus Temperature and Relative Humidity, and the tablet records participants' geo-locations and allows them to annotate their micro-environment by using a *self-reporting* mobile app. Therefore, they report every transition to a micro-environment (e.g., home, office, park, restaurant, etc.).

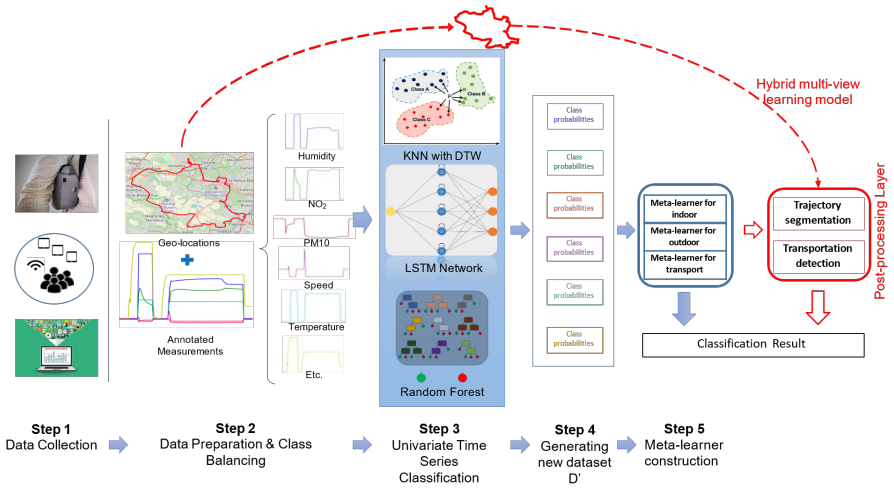


Fig. 2: Overview of the Micro-Environment Recognition Process.

5.2 Data Preparation

The second step is the data pre-processing which includes data de-noising, data imputation, data segmentation, and class balancing.

First, most sensor data are noisy, with irrelevant measurements from the actual condition. Even though the sensor data quality is a permanent preoccupation of the project, we observed the noisy data in both GPS (due to signal loss) and air quality data after careful evaluation before data selection and periodic qualification during the campaign [27]. The sensors for climatic parameters do not show such defects. Therefore, a de-noising process is applied on both GPS and air quality data. Precisely, we distinguish between peaks and artefacts by referring to the expert's judgment. The peaks that are judged to be real are then conserved. The same goes for GPS data which has been cleaned based on a maximum threshold of velocity (here 130km/h as it is the speed limit in France). Beyond this threshold, GPS points that are in charge of producing the velocity will be removed.

Second, the collected sensory data are usually incomplete due to device error or communication issues, with missing values at some time stamps. Therefore, we set a threshold of ten consecutive missing steps to conduct the imputation process. In other words, we perform data imputation on missing intervals that do not exceed 10 minutes (i.e., 10 steps). Precisely, new values are inferred with the linear interpolation approach on the non-missing temporal neighbors; that is, new values are interpolated by a linear function of the two temporal ends of the missing values. Globally, the highest quality sample of annotated data is selected as a baseline to validate the process of micro-environment recognition. The idea is to generalize the micro-environment recognition to all participants' data by using the model derived from a good-quality dataset.We

describe in Section 7.1 how the high quality sample of annotated data is selected.

Third, data is segmented into samples of fixed length (here 5 minutes). The choice of the fixed length value is discussed in Section 7.1. Each segment will be assigned a unique label. Essentially, the proposed model will take the observed measurements of the segment as input and produce a unique label by virtue of the multi-view learning.

Last but not least, micro-environment recognition is also subject to class imbalance problem. Usually, individuals spend most of their time indoors, either at home or at the office. A dataset is imbalanced if the classification categories are not equally represented, which is the case in our study.Therefore, because of this problem (home is the majority class, followed by office), the likelihood of having a good accuracy value of the classification is very high. The classifier will practically attribute the majority class to almost every data segment and fail to detect the minority classes, which leads to an overall good accuracy but does not necessarily reflect the actual performance of the classifier. Hence the solution of re-sampling and data augmentation, which are the commonly used techniques to solve this problem.

For data re-sampling, random oversampling of the minority classes and random under-sampling of majority classes are the most popular approaches. However, the random oversampling approach usually introduces duplicates to stabilize the training process, which does not thoroughly explore the valuable information from the data. Therefore, some work considers synthesizing new samples from the minority class. For instance, synthetic minority oversampling technique (SMOTE) [8] under-samples the majority class and over-samples the minority one based on the K-nearest neighbors. SMOTE selects samples that are close in the feature space, then generates a synthetic sample nearby. This procedure can be used to create as many synthetic examples for the minority class as required.

For data augmentation, Generative Adversarial Network (GAN) [22] has shown promising performance among various types of data, which uses existing data more effectively than re-sampling techniques. In the time series domain, the Time series Generative Adversarial Networks (TimeGAN or TGAN) [50] was proposed recently to generate realistic time series data considering the temporal dependency. However, in practice, it is generally hard to converge the adversarial training process with very limited samples [2], which is the case in our context.

Therefore, we combine both approaches of data re-sampling and data augmentation. First, we adopt SMOTE to under-sample the majority classes and over-sample slightly the minority classes. Then we apply the TimeGAN network to generate new samples over the minority classes. Figure 3 and 4 show the data distributions before and after class balancing respectively.

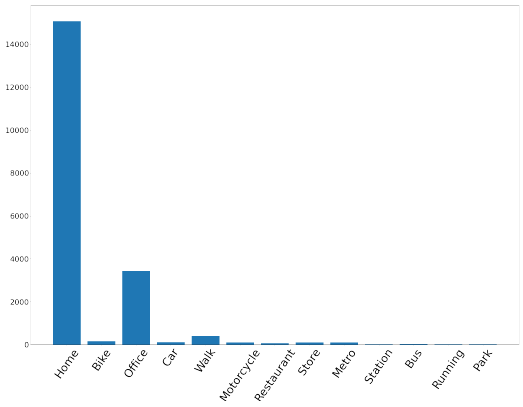


Fig. 3: Distribution of data over classes before class balancing.

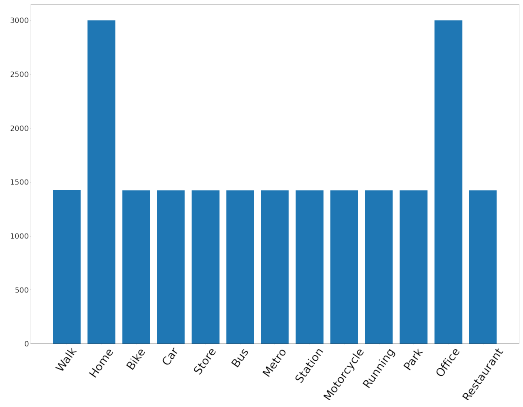


Fig. 4: Distribution of data over classes after class balancing.

5.3 Multi-View Learning Model Application

We propose to learn the micro-environment of participants from multivariate time series (MTS) through a two-stage model based on multi-view learning. The classification model consists of training a first-level learner on each view (i.e. step 3 in Figure 2), and then training a meta-learner (i.e. step 5 in Figure 2) to combine the output of each view and enhance the global accuracy of the classification. As stated before, we have as many views as dimensions. For instance, given a multivariate time series with four dimensions: temperature, humidity, speed, and NO2, each dimension will be considered as a view. Therefore, four different views will be considered in the multi-view learning model.

The spatial dimension as GPS tracks is not considered as a view because of the low spatial coverage. Also the target class is the type of micro-environment. The spatial dimension has less impact than the temporal pattern (2 locations could be spatially close but have different patterns in terms of exposure if one is indoor and the other outdoor).

In step 3, the first-level learner (e.g. *kNN*, LSTM, random forest, decision tree, etc.) takes as input the values of the time series data coming from each view, and outputs, for each view, a vector in the form $[l_i, p_1, p_2, ..., p_j, ..., p_k, y]$, where l_i is the predicted label of the first-level learner i, p_j is the associated prediction probability for each class j of the k possible classes, and y is the true label. Let us take the example above of the multivariate time series with four dimensions which are temperature, humidity, speed, and NO2, and examine the output of the first level learners. Let us say that our objective is to classify the MTS into three classes that are indoor, outdoor, and transport, while supposing that the true label is indoor. The temperature view will generate its own predicted label - let us say- indoor, and the associated predictions probabilities in this form $[l_{temperature} = indoor, p_{indoor} = 0.6, p_{outdoor} = 0.2, p_{transport} = 0.2, y = indoor]$. In the same way, the three remaining dimensions shall generate their own predicted labels with corresponding probabilities in this structure:
$[l_{humidity} = indoor, p_{indoor} = 0.7, p_{outdoor} = 0.1, p_{transport} = 0.2, y = indoor]$,
$[l_{speed} = outdoor, p_{indoor} = 0.4, p_{outdoor} = 0.5, p_{transport} = 0.1, y = indoor]$,
$[l_{NO2} = transport, p_{indoor} = 0.2, p_{outdoor} = 0.2, p_{transport} = 0.6, y = indoor]$.

In step 4, we aimed at giving a weight for each learner. Therefore, a new dataset D' is generated by column binding the output of first-level learner and the true label as shown in Table 1, where l_i is the predicted label of the first-level learner i, p_i is the probability of this prediction, and y is the true label. Continuing with the same example of the four dimensional MTS above, the feature structure of the generated dataset would be in the structure shown in Table 2.

Table 2: A concrete example of the new generated dataset D'.

First-Level Learners				**Associated Prediction Probabilities**				True Label
temperature	humidity	speed	NO2	temperature	humidity	speed	NO2	
indoor	indoor	outdoor	transport	0.6	0.7	0.5	0.6	indoor

In step 5, and after generating a new dataset D', a *meta-learner* is trained over D'. That said, by referring to the example above, the second level-learner (e.g. *Random Forest*) will take the generated features (i.e. every view's detected label plus its corresponding probability) as input and produce the final detected label. For instance and from D' shown in Table 2, the meta-learner takes as input the label produced by the view "temperature" (i.e. indoor) and its associated prediction probability (i.e. 0.6), plus the labels and their

corresponding probabilities from the other three views (i.e. humidity, speed and NO2). Therefore, the meta-learner's input has the following structure [indoor, indoor, outdoor, transport, 0.6, 0.7, 0.5, 0.6] and produces the final label (e.g. indoor) from the combination of labels and their associated prediction probabilities.

One of the advantages of multi-view learning is its versatility in first and second level learners' choices. One can flexibly substitute classifier choices as wished between kNN, LSTM, random forest decision tree, or any other classifier [15]. In this work, we opt for *Random Forest* classifier for the first as well as meta-learners since it has shown high performance when applied in the human activity recognition domain [21].

6 Hybrid Multi-view Learning Model

The multi-view learning model records some limitations, especially when it comes to discriminating between some indoor micro-environments that share similar characteristics such as "home" and "office", or between some transportation modes. Besides, the time of presence is often characteristic of some micro-environments (e.g. night hours is likely to indicate home, and working hours usually denotes the office). Identifying precisely some stay locations from GPS data is possible, and may improve discriminating the corresponding micro-environments. Thus, the need for an improvement in the model seems necessary.

We introduce new optimisations based on these observations. Specifically, the new optimisations include two approaches. The first one is *privacy invasive*. It includes the exact locations of home and office reported by participants in the post-processing layer. Certainly, it requires to have this private information ahead. The second approach is *privacy friendly*. The latter approach (i.e. privacy friendly) will be the subject of discussion in this section.

Figure 2 shows the new privacy friendly optimisations presented by the red dashed lines, which consist mainly of adding trajectory data as another layer while post-processing the results of the multi-view learning model, along with the disambiguation between home and office based on the location.

This post-processing layer consists of splitting trajectory data into stop and move segments (i.e. the trajectory segmentation box in Figure 2). We propose a stop detection algorithm based on grid density that we will present and discuss in the following section. We tag every stop with a unique and specific number to distinguish between them.Plus, we infer the location of home and office based on a priori rules according to the time of presence in the stop and the density of the stop. We further discuss these rules in Section 6.1. Furthermore, and after distinguishing between stop and move segments, the move segments are labeled by transportation means (e.g., metro, bus, car, etc.), which is represented by the transportation detection box in Figure 2. We take advantage of the work of Etemad *et al.* [18], which we have already discussed in the related work section, to detect the transportation mode and include the

results in the post-processing layer. In the following section, we present our proposed algorithm for Grid Density-Based Stop Detection (GDSD).

6.1 Stop Detection

In this section, we present a novel and robust algorithm for stop detection based on GPS data only, namely Grid Density-Based Stop Detection (GDSD). This approach can be used either as a separate view in the multi-view learning model to infer the stay places from GPS data (i.e. mobility view), or as a post-processing layer to correct the ambiguity between home, work, and other stop places. In the current work, the mobility data is used as a post-processing layer.

GDSD approach takes GPS points as input, and outputs segments of the same fixed length as the multi-view model's segments, to ensure their comparability. Each segment is labeled with the number of the stops or the label "home" or "office". Let us take an example of four segments s1, s2, s3, and s4. Each segment is 5 minutes long. The multi-view model results assign to segments s1 and s2 the home label, and to s3 and s4 the work label. However, according to the results of the stop detection model, all the four segments belong to the same cluster, which implies that the four segments are all together either at home, work, or any other indoor micro-environment. But, since these four segments (s1, s2, s3, and s4) are labeled "home" by the GDSD approach, the final class shall be "home". That is precisely the objective of this stop detection extension: to eliminate the ambiguity between stop micro-environments and improve the performance of the multi-view model.

Therefore, in this approach, the GPS tracks (i.e. latitude and longitude) are transformed into discrete values referencing a pixel of a rectangular grid with a spatial resolution (here of 50 m^2). Then, in order to organize the cells in a way that allows to maintain the locality of spatially close GPS points, we adopt spatial indexing using 2D Hilbert Space-Filling Curves (SFC), which provides a grouping feature per proximity [31]. In other words, neighboring cells are likely to be assigned to close Hilbert indices. Figure 5 shows the rasterization of the spatial dimension using Hilbert SFC. The spatial extent is defined to cover the study area (i.e. Paris region). It is worth mentioning that we only maintain cells corresponding to the locations with GPS data, and discard cells with no GPS data within.

The adopted rasterization approach allows us to derive the stay areas (often indoors) of participants. We can discover the places where a participant spends time the most based on cells densities.

When looking at the sample of GPS trajectory points in Figure 6, we can easily and unambiguously detect the existence of two stops plus some noise points that may or may not belong to any of the stops. The reason why the

[2] This value has been chosen in accordance with the resolution adopted by Airparif (the agency in charge of AQ monitoring in the Paris Region, also part of the Polluscope consortium) in their simulation models.

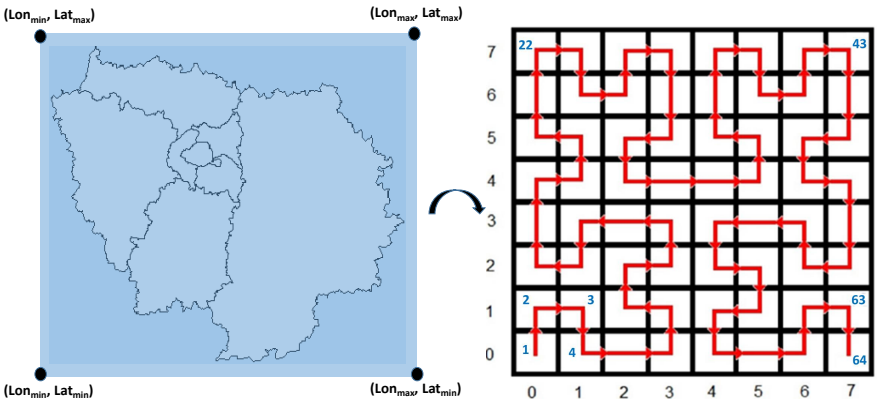


Fig. 5: Spatial Dimension Representation

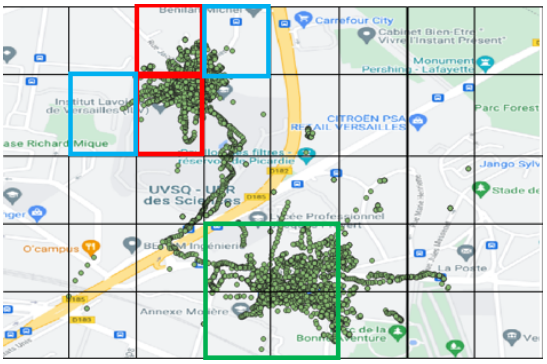


Fig. 6: Sample trajectory

human mind detects or recognizes the stops is because the density of points inside the stops is higher than outside them. We mimic the same human brain rational reasoning to detect the stops based on the cell density.

We formalize our intuitive notion of deriving stop places from a spatial dataset $\boldsymbol{D}$ in a 2D euclidean space. The key idea is to set a minimum density threshold **MinDens** in a cell in order to be detected as a stop cell. The adjustment of *MinDens* is utterly empirical and depends on the dataset size and GPS sampling frequency. For instance, the *MinDens* in a dataset collected over one week with a high sampling frequency (e.g. every one second) is naturally different from a *MinDens* chosen from a dataset collected over one day with the same sampling frequency. We explain further in Section 7 how we set the *MinDens* threshold.

Furthermore, because of the limitations of GPS readings, some stop clusters may be shared by more than one cell (e.g. green cells in Figure 6). For this reason, there is a need to include in the detected stop the neighbouring cells that may belong to the same stop. Yet, as Hilbert SFC are by definition hierarchical, one has only to divide the Hilbert index by 2^{2n}, where $n \in \mathbb{N}$ to access the neighbouring cells.

Thereafter, the stop detection algorithm based on grid density takes as input the minimum density threshold ***MinDens*** and $\boldsymbol{n}$ (as in $2^{2\mathbf{n}}$). The algorithm then joins the neighbouring cells to the detected stop even though these neighbouring cells do not verify the density condition, forming a new cluster composed of several cells and at least one cell that verifies the density condition.

Algorithm 1 presents a basic version of the stop detection algorithm without focusing on GPS data pre-processing and cleaning. Primarily, the algorithm rasterizes the data and creates a Hilbert index that is assigned to each cell. The algorithm then selects a set of cell indices whose density is higher than the desired density threshold ***MinDens***. All the selected cells are stop candidates. The algorithm moves systematically to a higher level of hierarchy by dividing the acquired Hilbert index by 2^{2n} (i.e., a grouping of 4 cells, 16 cells, 64 cells, etc.) and considers the whole grouping of cells around a stop candidate as a stop. The remaining cells will be considered as move segments. For instance, in Figure 6, let us suppose that only one of the four green cells verifies the density condition (the upper right green cell), then this cell forms a stop candidate. After dividing its Hilbert index by 2^{2*1}, we systematically go to a higher level of hierarchy and the whole grouping of the four green cells will be considered as a stop.

However, some outlier points may slip out of the grouping cells (e.g. blue cells in Figure 6). The algorithm has another step which consists of post-processing the trajectory segments based on a temporal threshold ***MinDur***. In other words, if a move (resp. stop) segment is jammed between two stops (resp. move) segments and the duration of this segment is less than ***MinDur***, then this segment is to be merged with the previous segments.

A comparison between Grid Density-based Stop Detection (GDSD) and state-of-the-art approaches is discussed further in Section 7.

Next comes the step of inferring the location of home and work based on some a priori rules. The straightforward way is to draw the location of home according to the time of presence in the stop. If the participant is static between 2am and 4am every day at the same location, it is very likely that the location in question is home. Another criterion for inferring the location of home and work is the density of the stops. Usually and based on common sense, people spend the most of their time in their home followed by their work. Thereby, the densest stop is likely to be home and the second densest stop is work, ceteris paribus. These assumptions are confirmed by the participants declared annotations of their micro-environments - if they exist.

Algorithm 1 Grid Density-Based Stop Detection (GDSD)

procedure GDSD($MinDens, n, MinDur$)
  $SetOfCells = \{\}$
  create HilbertIndex from (Lon, Lat) ▷ Create the Hilbert index from latitude and longitude.
  $SetOfIndex = \{HilbertIndex\}$ ▷ Create a set of all the possible Hilbert indices.
  $density \leftarrow GroupBy(HilbertIndex)$ and count
  $stops \leftarrow HilbertIndex\ where\ density >= MinDens$
  $k = 1$
  for Hindex in *stops* **do**
    $x \leftarrow Floor(Hindex/2^{2n})$
    $SetOfCells := SET(i\ i/2^{2n} = x)\ \forall i \in SetOfIndex\ label(SetOfCells) \leftarrow k$
    $k := k + 1$
  end for
  for $HindexinSetOfIndex \setminus stops$ **do**
    $label(Hindex) \leftarrow -1$
  end for
  $segments := Set(segments)$ ▷ segments is the set of all the stop segments
  $j := 1$
  while $j < segments.size$ **do**
    if $Duration(segments[j]) < MinDur$
      & $segments[j].label == segments[j+1].label$ **then**
      $segments[j-1] \leftarrow concat(segments[j-1], segments[j])$
      $del\ segments[j]$
    end if
  end while
  return *segments*
end procedure

7 Experiments and Results

The experiments are carried out in different environments. The multi-view learning model was implemented in Python 3.6 using scikit-learn 0.23.2 and tslearn [43]. The deep-learning models (MLSTM-FCN [25], TapNet [53]) were trained on a single Tesla V100 GPU of 32 Go memory with CUDA 10.2, using respectively Keras 2.2.4 and PyTorch 1.2.0.

7.1 Experimental Settings

We evaluate the proposed models in these experiments using real-life data collected in the scope of the Polluscope project. In Polluscope, three data collection campaigns have been conducted, covering the whole study area (i.e., Paris region). Each campaign was spread over 12 weeks, with a collection generally carried out every other week (in order to check and re-qualify the sensors). 103 volunteers participated in the data collection phase, which lasted one week for each participant. These participants are equipped with a kit that contains air pollution sensors and tablets empowered with GPS chipsets. The sensors collect, every one minute, time annotated concentrations of Particulate Matters (PM1.0, PM10, PM2.5), Nitrogen dioxide NO2, Black Carbon (BC), temper-

Table 3: General characteristics of the two campaigns VGP and RECORD.

Campaign	Number of participants	Measurement period	Sensor's wearing time
VGP	15	October 2019	7 days
	12	November 2019	
	09	November 2019	
	15	December 2019	
	12	December 2019	
RECORD	13	January - March 2020	7 days

ature, and relative humidity. The tablet serves to geolocate the participants and to fill in their time micro-environment via an Android app developed for this purpose. The speed dimension was derived from the geo-locational data.

In total, 13 activities (i.e., micro-environment to recognize) are considered in this study, which can be organized into three categories:

- Indoor environment: *home*, *office*, *restaurant*, *store*, *station*
- Outdoor environment: *park*, *walk*, *run*, *bike*
- Transport environment: *metro*, *car*, *bus*, *motorcycle*

Previously (i.e. in [1]), we related to the annotation of the given tool (i.e., an Android app installed in the tablet). In this work, data have been enriched both based on a tool (called TripBuilder Web) [6], and a thorough human control of participants' annotations within the third campaign called RECORD [5]. Therefore, this data is more reliable than our previously used data collected during the second campaign, called VGP. Table 3 presents the general characteristics of the two campaigns, i.e. VGP and RECORD. We select the data of 13 participants with the best annotation activities in the RECORD campaign. Overall, the dataset contains 8 dimensions, more than 1 million rows (per dimension), with an average of 82071 rows per participant. The collected data are split into two thirds for training and one third for testing, with care taken to keep the data of each participant grouped either in training or testing set. We use the cross-validation score with "repeated stratified k-fold" to re-split the training set into training and validation sets, while we evaluate the overall model performance on the testing set.

Considering the temporal feature of the data, we segment the collected data into samples of 5 minutes' length at maximum. Usually, people spend most of their time indoors. We should thus consider outdoor activities with a short period compared to indoor activities. For example, the average time spent in "station" is around 4 minutes as shown in Table 4 which depicts the average time spent per micro-environment. Participants tend to spend more time in some micro-environment (typically "home" and "office") than others (e.g. "walk", "metro", "store", etc.). Globally, as shown in Figure 3, the distribution of data samples is highly imbalanced over the different classes, leading to poor classification performance, especially for the minority classes. More precisely, the model tends to optimize the global loss error which is biased towards the majority classes while ignoring the minority ones. In consequence,

Table 4: Average time spent per micro-environment.

Micro-environment	**Stay duration (in minutes)**
Office	446
Bus	13
Home	899
Station	4
Store	24
Motorcycle	20
Metro	17
Park	76
Restaurant	46
Running	76
Car	29
Bike	50
Walk	12

the obtained accuracy is not reliable to evaluate the actual model performance. To cope with this problem, we re-balanced the classes via data re-sampling and data augmentation as mentioned in section 5.2 when pre-processing the data. Figure 3 and figure 4 show the data distributions before and after class balancing, respectively.

7.2 Experimental Design

First, we evaluate our basic multi-view learning model without integrating the post-processing layer. Considering the mobility information in our data, we carry out our experiments on the datasets with or without integrating the *speed* variable. Furthermore, to thoroughly evaluate the importance of the mobility information, we introduce and evaluate a two-step approach by first discriminating between *indoor*, *outdoor*, and *transport* micro-environments, followed by a refinement step to learn a more specific class.

Then, we evaluate our proposed algorithm for stop detection, which is a key component in our post-processing layer. We compare it with the state-of-the-art models implemented in Scikit-Mobility [33].

Finally, we conduct an extensive experiment considering various optimization techniques proposed in the post-processing step. We evaluate the effect of the post-processing layer not only on our multi-view learning model but also on other classic MTSC models. We optimize the proposed approaches by either analyzing the exact geolocation of the participant (privacy-invasive method) or using a priori rules (privacy-friendly method).

7.3 Model Performance without Post-processing

In this section, we detail the experimental results of the multi-view learning model without integrating the post-processing layer. First, we evaluate the

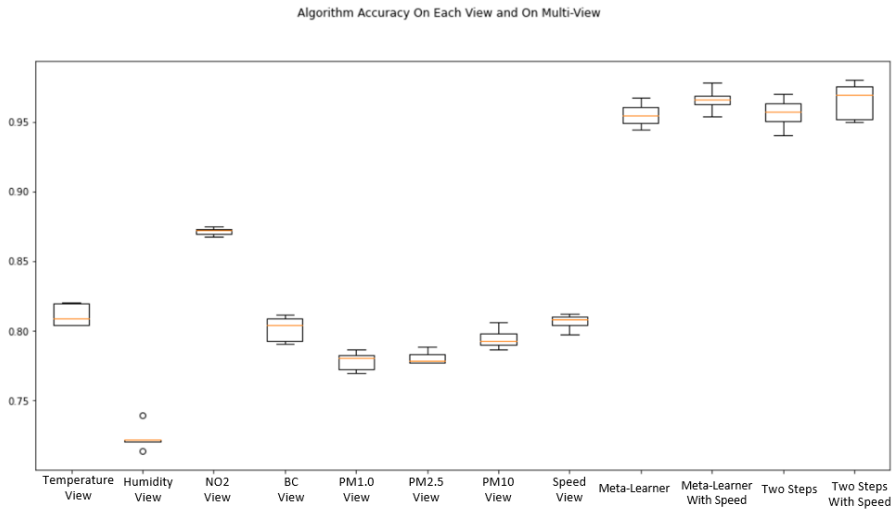


Fig. 7: Accuracy among different views.

first-level learners on each single view and the multi-view learner on the global view. We evaluate as well the multi-view learner when applying the two-step approach which learns firstly the coarse-grained classes (i.e., *indoor*, *outdoor* and *transport*) then refine them into more specific classes (e.g., *home*, *park*, *metro*, etc.). Then, we compare the multi-view learner with MLSTM-FCN [25], the state-of-the-art on Multivariate Time Series Classification.

As mentioned in Section 5.3, the micro-environment recognition can be formulated as a Multivariate Time Series Classification (MTSC) problem, and the multi-view learner combines the predictions of each independent view (i.e., dimension) from the first-level learners to get the final classification results. In Figure 7, we report the accuracy of the first-level learners over the different views, as well as the multi-view learner and the two-step approach with and without considering the mobility (i.e., speed) dimension. Globally, the results suggest that the multi-view learner shows comparable performance, with or without adopting the two-step approach. Integrating the *speed* dimension helps slightly improve the performance of the multi-view learner. We observe that the first-level learners usually have low accuracy performance, which is not surprising as the incomplete local information is not enough to train a reliable model. By combining the local information from different views, the multi-view learner can improve the model accuracy significantly.

To know how our multi-view learner performs compared to the state-of-the-art work, we select MLSTM-FCN [25], a powerful deep learning model for Multivariate Time Series Classification. We show as well the detailed evaluation results when applying the two-step approach. Since MLSTM-FCN requires enormous computational resources for parameter optimization, we train the model on GPU. In contrast, our multi-view-based approaches are trained on

Table 5: Performance of Multi-view Learner (with/out speed)

class	Without Speed			With Speed		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.96	0.86	**0.91**	0.95	0.87	**0.91**
Bus	0.99	0.96	**0.98**	0.98	0.97	**0.98**
Office	0.96	0.88	0.92	0.97	0.92	**0.95**
Restaurant	0.97	0.97	0.97	0.99	0.97	**0.98**
Home	0.87	0.97	0.92	0.90	0.99	**0.94**
Bike	0.92	0.97	0.94	0.96	0.99	**0.97**
Car	0.99	0.98	0.98	0.99	0.99	**0.99**
Store	0.94	0.93	0.94	0.96	0.96	**0.96**
Metro	0.96	0.93	0.94	0.98	0.95	**0.96**
Station	0.98	0.96	0.97	0.99	0.97	**0.98**
Motorcycle	0.99	0.99	**0.99**	0.99	0.99	**0.99**
Running	0.99	0.99	**0.99**	0.99	0.99	**0.99**
Park	0.99	0.98	0.98	0.99	0.98	**0.99**

Table 6: Performance of MLSTM-FCN (with/out speed)

class	Without Speed			With Speed		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.98	0.95	**0.96**	0.94	0.97	**0.96**
Bus	1.0	1.0	**1.0**	1.0	1.0	**1.0**
Office	0.97	0.95	**0.96**	0.96	0.94	0.95
Restaurant	1.0	1.0	**1.0**	1.0	1.0	**1.0**
Home	0.97	0.97	**0.97**	0.98	0.97	**0.97**
Bike	0.98	1.0	**0.99**	0.98	1.0	**0.99**
Car	0.99	1.0	**1.0**	0.98	1.0	0.99
Store	0.99	1.0	**0.99**	0.99	1.0	**0.99**
Metro	0.99	1.0	**0.99**	1.0	0.97	**0.99**
Station	0.99	1.0	**1.0**	1.0	1.0	**1.0**
Motorcycle	1.0	1.0	**1.0**	1.0	1.0	**1.0**
Running	1.0	1.0	**1.0**	0.99	1.0	0.99
Park	1.0	1.0	**1.0**	1.0	1.0	**1.0**

a normal CPU with less requirement on computational resources. For each of the models, we study the impact of using or not the mobility data and report the performance in terms of *precision*, *recall*, and *F1 score*.

The detailed results are grouped in Table 5, 6, and 7. Globally, the three models have comparable results before and after adding mobility. While MLSTM shows slightly better performance than the two-step model, the latter outperforms the multi-view model. Looking at the F1-score, the out-performance of MLSTM, compared to the two-step model, does not go beyond 3 point (e.g. 0.96 and 0.99 for the class bike) before adding mobility, and 2 points (e.g. 0.97 and 0.99 for the class store) after adding mobility, whereas the difference between the two-steps model and the multi-view model does not exceed 4 points (e.g. 0.91 and 0.95 for the class walk) before and after adding mobility.

As for our multi-view learner, when integrating the speed dimension for model training, we observe an improvement in the model's performance, par-

Table 7: Performance of Multi-view Learner (2-step approach with/out speed)

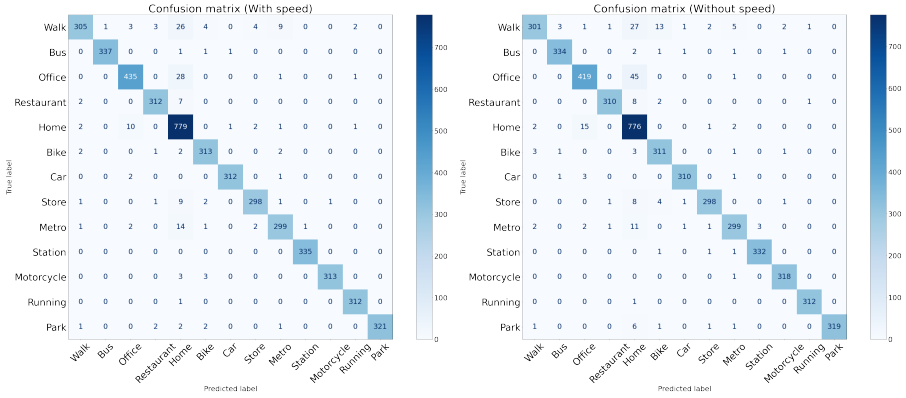
class	Without Speed			With Speed		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.93	0.97	**0.95**	0.95	0.96	**0.95**
Bus	0.99	0.99	**0.99**	0.99	0.99	**0.99**
Office	0.97	0.92	**0.94**	0.97	0.91	**0.94**
Restaurant	0.99	0.98	**0.98**	0.99	0.98	**0.98**
Home	0.93	0.97	**0.95**	0.93	0.98	**0.95**
Bike	0.97	0.96	0.96	0.97	0.97	**0.97**
Car	0.98	0.99	**0.99**	0.99	0.99	**0.99**
Store	0.98	0.97	**0.97**	0.98	0.96	**0.97**
Metro	0.98	0.97	**0.98**	0.98	0.98	**0.98**
Station	1.0	1.0	**1.0**	0.99	1.0	0.99
Motorcycle	0.99	0.98	0.98	0.99	0.99	**0.99**
Running	0.98	0.98	**0.98**	0.98	0.98	**0.98**
Park	0.99	0.96	0.97	0.99	0.97	**0.98**

ticularly the F1-score, while the performance of MLSTM-FCN does not improve or even deteriorates. Figure 8 shows the confusion matrix of multi-view approach. Figure 8a reports the confusion matrix with the presence of the mobility dimension (i.e. speed), while figure 8b corresponds to the confusion matrix of the model with the absence of mobility dimension. We notice that the model can easily discriminate between the "indoor", "outdoor" and "transport" activities, but it cannot perfectly distinguish between the micro-environments inside each category. For example, even though some of the samples in the "home" micro-environment are falsely predicted as "restaurant" or "office", the three micro-environments, "home", "office" and "restaurant" can be classified as indoor. Thereby, we introduced a grouping step before recognizing the micro-environment. In this step we classify the sample into either an "indoor", "outdoor", or "transport" environment. Based on the classification result, a model will be specialized for each indoor, outdoor or transport micro-environments. Table 7 shows the results of the added step.

7.4 Stop Detection Performance

In this section, we evaluate our proposed algorithm, Grid Density-based Stop Detection (GDSD). First, we study the parameter effects on GDSD's performance and select the best ones when applying the algorithm. Then, we compare GDSD with Scikit-Mobility [33], the state-of-the-art approach designed for stop detection. Here, we adopt only the GPS data of the 13 participants in the Polluscope RECORD campaign.

Before applying the proposed approach, the critical question is to set the minimum density per cell (i.e., density threshold) and the number of grouping cells within a stop. Generally, these parameters are set empirically. Without prior knowledge of the data, it is necessary to conduct various tests to discover the best parameters. To this end, we tune one parameter while blocking an-

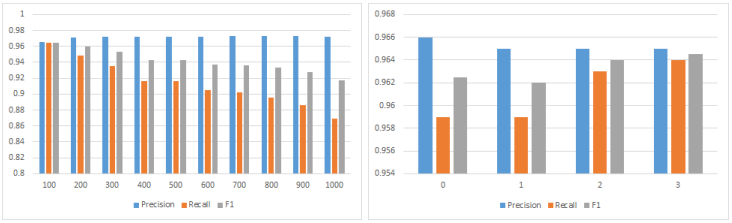


(a) With Speed (b) Without Speed

Fig. 8: Multi-view Approach Confusion Matrix

other one to observe how the model performance evolves. Precisely, we set the values between 100 and 1000 points, with a step of 100, for the density threshold. The number of grouping cells $\boldsymbol{n}$ varies between 0, 1, 2, and 3, meaning a grouping of, respectively, 1, 4, 16, and 64 cells. Firstly, we start by searching for the optimal value of the number of grouping cells. We iterate over the values of the density threshold (i.e. between 100 and 1000 with a step of 100). On each round, we compute the performance of the model while iterating the values of n. We report the results with the optimal iterations. Figure 9 shows the parameters' effects on the model's performance. From Figure 9a (with n=3), we observe that the precision improves slightly, whereas the recall drops with the increase of density threshold, indicating a trade-off between precision and recall when the density threshold equals 100. This observation is supported by the F1-score, whose value is optimum at 100. As for the parameter' effect of $\boldsymbol{n}$, Figure 9b (with density threshold equal to 100) shows that when $\boldsymbol{n}$ increases, the precision drops slightly, whereas the recall score increases a little, depicting an adequate trade-off between precision and recall when $\boldsymbol{n}$ equals 3 (i.e. a grouping of 64 cells). We observe as well an optimal F1-score under this setting. In the rest of the paper, we set the grid density threshold to 100 and set $\boldsymbol{n}$ to 3.

To validate the performance of our proposed stop detection algorithm GDSD, we compare it with the state-of-the-art models implemented in Scikit-Mobility [33]. We conduct the experiments of the stop detection for each campaign participant separately. Table 8 depicts the results of this comparison. On the one hand, when looking at the precision of the two models, the two approaches are comparable for some participants (e.g., rows ID 1, 2, 4, 5, and 9), our approach outperforms the baseline for others except for one participant (i.e., row ID 6). On the other hand, the recall score and the F1-score show that the grid-density-based approach outperforms the baseline on all partic-



(a) Performance of GDSD wrt to the density threshold per cell.

(b) Performance of GDSD wrt to parameter **n**.

Fig. 9: Parameters' effect on Grid-Density Stop Detection (GDSD)

Table 8: Comparison between Scikit-Mobility and grid density based model.

Row ID	Participant ID	Scikit-Mobility			Grid Density Based		
		Precision	Recall	F1	Precision	Recall	F1
1	988088403	0.988	0.891	0.937	0.985	0.993	**0.989**
2	988231648	0.988	0.961	0.974	0.981	0.995	**0.988**
3	982228564	0.936	0.849	0.89	**0.953**	0.945	**0.949**
4	986002161	1.0	0.82	0.901	1.0	0.969	**0.984**
5	986939872	0.813	0.78	0.796	0.826	0.895	**0.859**
6	988335737	**0.908**	0.299	0.45	0.858	0.658	**0.745**
7	986174566	0.96	0.854	0.904	**0.991**	0.971	**0.981**
8	986884172	0.92	0.66	0.769	**0.985**	0.956	**0.97**
9	986938604	0.995	0.684	0.811	0.995	0.99	**0.992**
10	985935431	0.812	0.325	0.464	**1.0**	0.6	**0.7**5
11	987014104	0.936	0.92	0.928	**0.993**	0.995	**0.994**
12	82119412	0.84	0.559	0.671	**0.953**	0.9	**0.926**
13	983602168	0.934	0.963	0.948	**0.966**	0.973	**0.969**

ipants. Overall, the proposed approach always has better performance than the baseline.

7.5 Experiment Extension

In this section, we extend the experiments by applying the proposed post-processing techniques which are designed to enhance the micro-environment detection model. We apply four basic MTSC models:

- MVB: our proposed Basic Multi-View learning model.
- MV-2steps: our proposed Basic Multi-View learning model with two-step classification as shown in Section 7.3.
- MLSTM-FCN [25]: a powerful deep learning model for Multivariate Time Series Classification.
- KNN-DTW [4]: the most popular benchmark for Time Series Classification which adopts the K-nearest neighbor (K-NN) classifier with dynamic time wrapping (DTW) distance.

As the models are trained on different hardware environments (e.g., MLSTM-FCN is trained on a GPU, which is ten times faster than running on CPU), it is unfair to compare them in terms of efficiency. However, according to the recent study [38], the deep learning-based models usually require more computational resources than classic data mining approaches; the lazy classifiers (e.g., KNN-DTW) are much slower than the tree-based classifier (e.g., Random Forest) due to the costly distance computations (e.g., DTW). As the first-level learner and meta-learner in our multi-view learning model are based on Random Forest, thus the model training and prediction are quite efficient compared to other models.

The model variants after applying the post-processing techniques are detailed in Table 9. We go through extensive experiments and test various model variants to select the best model combinations. We organize the model variants into two categories: **privacy-friendly** and **privacy-invasive** models. For privacy-friendly models, post-processing is performed using stop detection and transportation mode detection techniques, while for privacy-invasive models, additional private information is adopted such as *Location of Home (LH)* and *Location of the Office (LO)*.

7.5.1 Global accuracy comparison on the model variants

In this section, we used the trained model to predict the micro-environment of our real MCS data and we adopted the post-processing techniques on the results. Here, we show the global accuracy comparison between the models within each privacy category. Tables 10 and 11 report the accuracy of various privacy-friendly and privacy-invasive models, respectively. The *NaN* in the results of MLSTM and KNN models indicates that no complete data is collected, thus, the models are not applicable. More precisely, some variables are missing during the data collection process. However, the multi-view-based models succeed all to detect the micro-environment even some dimensions are missing.

For the privacy-friendly models, all the proposed multi-view-based models show higher accuracy compared to baselines (i.e., KNN-based and MLSTM-based models). On the one hand, there is a big performance difference between multi-view-based models and the baseline models, especially for the participants who did not collect the complete variable data on which the baseline models are not applicable (i.e., *NaN* value). On the other hand, the post-processing does show its generalizability which improves the performance of both multi-view-based and baseline models. Among all the privacy-friendly models, the MVP (Multi-view with Post-processing) model shows the best performance, which validates the reliability of our proposed model.

For the privacy-invasive models, we adopt the additional private information: Location of Home (LH) and Location of the Office (LO), to check their impact on the models. However, since the baselines showed poor performance in the privacy friendly models, they will not be considered in this comparison. It is already known that, even with the exact locations of home and office,

Table 9: The description of various model variants

Model	Description
MVB	Basic multi-view model
MV-2steps	Multi-view model having 2 steps, first discriminate between indoor/outdoor/transport and then classify the micro-environment.
MV-2stepsP	Multi-view model having 2 steps, first discriminate between indoor/outdoor/transport and then classify the micro-environment, with a post-processing step based on stop detection transportation mode detection models.
PMV	Multi-view model with a pre-processing step based on stop detection transportation mode detection models.
MVP	Multi-view model with a post-processing layer based on stop detection and transportation mode detection models.
MLSTMB	Basic MLSTM-FCN model.
MLSTMP	MLSTM-FCN with a post-processing layer based on stop detection and transportation mode detection models.
KNN-DTWB	Basic KNN-DTW model.
KNN-DTWP	KNN-DTW with a post-processing layer based on stop detection and transportation mode detection models.
MVB+LO	Basic multi-view model with a post-processing step based on the location of the office.
MVB+LH	Basic multi-view model with a post-processing step based on the location of home.
MVP+LO	Multi-view model with a post-processing layer based on stop detection and transportation mode detection models as well as the location of the office.
MVP+LH	Multi-view model with a post-processing layer based on stop detection and transportation mode detection models as well as the location of home.

Table 10: Performance comparison of various **privacy-friendly** models

Participant ID	MVB	MV-2steps	PMV	MVP	MV-2stepsP	MLSTMB	MLSTMP	KNN-DTWB	KNN-DTWP
988088403	88.2	89.2	89.6	**95.9**	95.8	63.5	66.6	85.8	86.5
988231648	90.9	90.7	90.9	93.5	**93.6**	NaN	NaN	NaN	NaN
982228564	94.3	93.8	91.7	94.8	**94.9**	23.6	24.0	74.8	76.7
986002161	**91.0**	89.5	89.5	**91.0**	89.7	NaN	NaN	NaN	NaN
986939872	**83.3**	82.2	80.0	88.1	86.4	34.4	34.4	72.0	75.2
988335737	60.9	59.2	59.2	**61.2**	59.5	NaN	NaN	NaN	NaN
986174566	92.0	92.5	92.1	92.2	**93.5**	10.8	11.3	76.7	78.3
986884172	85.9	85.7	86.5	**88.6**	88.3	NaN	NaN	NaN	NaN
986938604	98.6	98.4	98.3	**98.8**	98.7	37.6	37.3	91.4	91.9
985935431	90.7	90.8	**91.1**	90.9	**91.1**	NaN	NaN	NaN	NaN
987014104	98.1	97.6	97.6	**99.1**	98.6	38.5	39.6	89.8	92.2
82119412	96.8	95.9	96.0	**97.2**	96.2	10.0	9.8	66.0	65.5
983602168	89.8	89.9	89.1	92.4	**92.5**	NaN	NaN	NaN	NaN
Overall Accuracy	91.33	91.0	90.71	**93.43**	93.10	31.14	31.70	83.48	85.06

Table 11: Performance comparison of various **privacy-invasive** models

Participant ID	MVB +LO	MVB +LH	MVP +LO	MVP +LH
988088403	92.3	92.6	94.9	**96.0**
988231648	95.4	94.8	**96.0**	95.6
982228564	95.8	94.4	**95.9**	94.3
986002161	**95.7**	**95.7**	**95.7**	**95.7**
986939872	87.2	87.3	**89.2**	**89.2**
988335737	**67.2**	**67.2**	**67.2**	**67.2**
986174566	93.2	93.2	**93.3**	**93.3**
986884172	**94.4**	94.2	94.3	93.6
986938604	99.7	99.1	**99.8**	99.2
985935431	94.9	94.9	**95.0**	**95.0**
987014104	92.2	97.7	**99.4**	97.9
82119412	NaN	98.0	NaN	**98.3**
983602168	94.4	92.7	**94.5**	93.3
Overall Accuracy	94.70	94.20	**95.27**	94.87

they will fail with missing dimensions. By considering the office location correction, the MVP+LO model demonstrates the best performance among the privacy-invasive models. More importantly, MVP+LO shows the highest overall accuracy among both privacy-friendly and privacy-invasive models. The *NaN* in the results of MVB+LO and MVP+LO models indicates that the location of the office is unknown, and thereafter, the post-processing task is not applicable. Globally, the privacy-invasive models show better performance than the privacy-friendly models, indicating that the private information does help improve the models. However, in practice, private information is not always available. Therefore, a trade-off between model performance and user privacy should be considered in practice.

7.5.2 Post-processing effects on various basic MTSC models

In this section, we report the detailed results of various model variants applied to new data (not seen before by the model) to show the effects of the post-processing techniques. First, we show the performance of the privacy-friendly models before and after adopting the post-processing layer (i.e., stop-mode and transportation-mode detection). Then, for privacy-invasive models, we briefly compare the effects between various location-correction techniques (i.e., LO and LH) on our multi-view learner after post-processing (i.e., MVP).

For privacy-friendly models, we show the results on four basic MTSC models (i.e., multi-view learner, two-step multi-view learner, MLSTM-FCN, and KNN-DTW) with or without pre-processing. Tables 12, 13, 14 and 15 show the metric comparison (i.e., *precision, recall, F1-Score*) of the four models, respectively. Globally, the multi-view models (i.e., MVB and MVP) show better performance than both MLSTM-FCN and the two-step multi-view models. Furthermore, the post-processing allows improving all the basic models, especially for the F1-score, in which we can observe a noticeable improvement. To have a more detailed understanding of the results, we show in Figures 10, 11,

Table 12: Performance of Multi-view Learner on Participants' data (before/after) post-processing

class	MVB			MVP		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.82	0.79	0.80	0.85	0.83	**0.84**
Bus	0.85	0.64	0.73	0.96	0.69	**0.79**
Office	0.86	0.85	0.85	0.92	0.90	**0.90**
Restaurant	0.42	0.60	**0.50**	0.44	0.60	**0.50**
Home	0.95	0.95	0.95	0.96	0.96	**0.96**
Bike	0.57	0.61	0.59	0.61	0.61	**0.61**
Car	0.51	0.18	0.27	0.78	0.25	**0.38**
Store	0.61	0.61	0.61	0.64	0.68	**0.64**
Metro	0.62	0.70	0.66	0.71	0.70	**0.71**
Station	0.16	0.17	0.16	0.25	0.16	**0.20**
Motorcycle	0.33	0.08	0.12	0.65	0.30	**0.41**
Running	0.30	0.61	0.40	0.38	0.61	**0.48**
Park	0.32	0.86	0.47	0.36	0.85	**0.50**

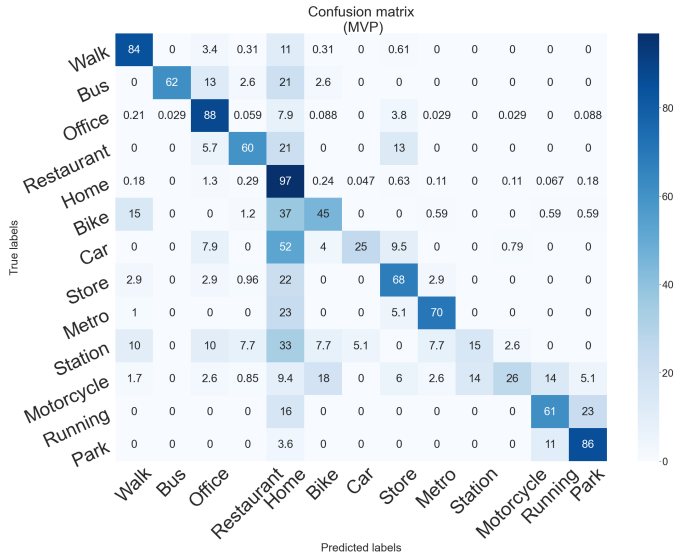


Fig. 10: MVP confusion matrix

and 12 the related confusion matrices of the models where we report the percentage of true predictions in each class. From the results, we observe that the post-processing improved largely the recognition of the **outdoor** (e.g., *walk*, *bike*, *park*) and **transport** (e.g., *car*, *bus*, *metro*) micro-environments, whereas the performance on **indoor** micro-environments (e.g. *station*, *restaurant*, and *store*) recognition is only slightly improved, which is mainly due to the limited sample numbers in the testing set.

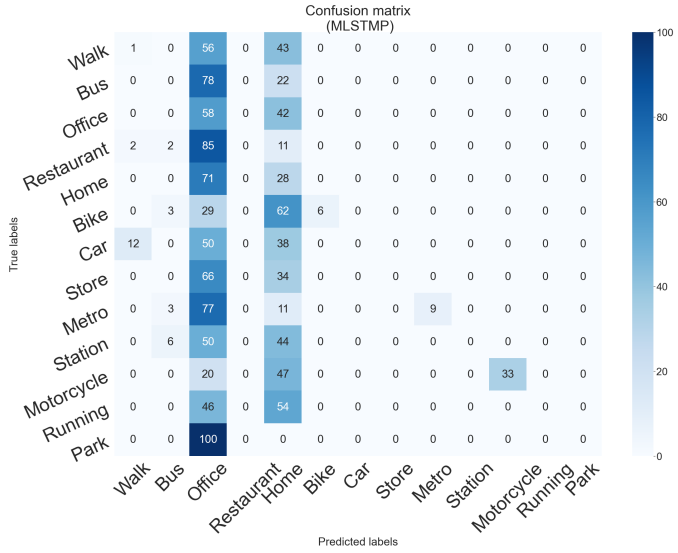


Fig. 11: MLSTMP confusion matrix

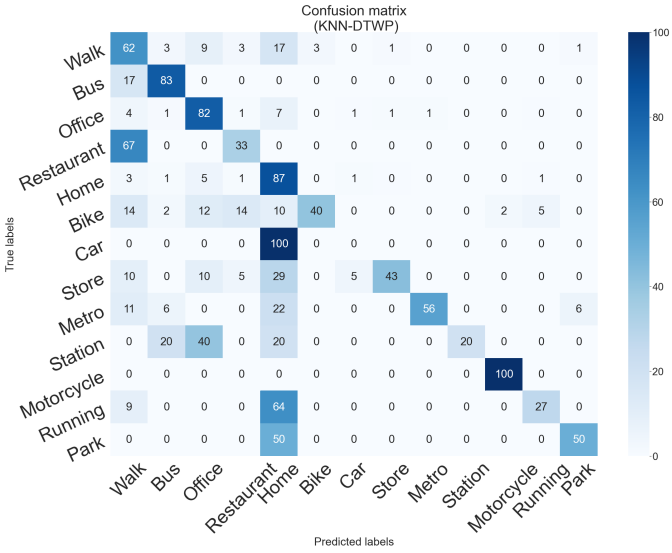


Fig. 12: KNN-DTWP confusion matrix

Table 13: Performance of Multi-view Learner (2 steps classification) on Participants' data (before/after) post-processing

class	MV-2steps			MV-2stepsP		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.75	0.79	0.77	0.78	0.83	**0.80**
Bus	0.73	0.64	0.68	0.81	0.69	**0.74**
Office	0.85	0.87	0.86	0.92	0.96	**0.94**
Restaurant	0.43	0.60	0.50	0.48	0.60	**0.54**
Home	0.95	0.95	0.95	0.97	0.97	**0.97**
Bike	0.43	0.38	0.41	0.52	0.41	**0.44**
Car	0.50	0.14	0.22	0.88	0.23	**0.36**
Store	0.45	0.62	0.53	0.60	0.68	**0.60**
Metro	0.57	0.50	**0.53**	0.76	0.45	**0.53**
Station	0.46	0.15	**0.23**	0.80	0.14	0.22
Motorcycle	0.27	0.04	0.07	0.76	0.28	**0.40**
Running	0.24	0.61	0.35	0.38	0.60	**0.46**
Park	0.41	0.93	**0.57**	0.38	0.96	0.54

Table 14: Performance of MLSTM-FCN on Participants' data (before/after) post-processing

class	MLSTMB			MLSTMP		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.13	0.02	0.03	0.13	0.09	0.01
Bus	0.0	0.0	0.0	0.0	0.0	0.0
Office	0.13	0.60	0.22	0.13	0.58	0.21
Restaurant	0.0	0.0	0.0	0.0	0.0	0.0
Home	0.73	0.27	0.40	0.74	0.28	0.41
Bike	0.0	0.0	0.0	0.67	0.06	0.11
Car	0.0	0.0	0.0	0.0	0.0	0.0
Store	0.0	0.0	0.0	0.0	0.0	0.0
Metro	0.0	0.0	0.0	1.0	0.08	0.16
Station	0.0	0.0	0.0	0.0	0.0	0.0
Motorcycle	0.0	0.0	0.0	0.55	0.33	0.42
Running	0.0	0.0	0.0	0.0	0.0	0.0
Park	0.0	0.0	0.0	0.0	0.0	0.0

However, as shown in Table 14 and 15, MLSTM-FCN-based and KNN-DTW-based models show bad performances even after post-processing. For instance, in Table 14, the MLSTM-FCN-based models fail to detect most of the micro-environments, even though they perform relatively better on detecting *home*, the performance is still much worse than multi-view-based models. Therefore, we draw a similar conclusion as mentioned in Section 7.5.1: the baseline models are not applicable on such complex scenarios where some dimensions are missing during data collection process; In other words, some sensors may be inoperative due to a technique issue for a long time, thus, some samples contain less dimensions than others. On this aspect, our multi-

Table 15: Performance of KNN-DTW on Participants' data (before/after) post-processing

class	KNN-DTWB			KNN-DTWP		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.13	0.52	0.20	0.26	0.62	0.36
Bus	0.06	0.55	0.11	0.1	0.83	0.17
Office	0.59	0.64	0.62	0.66	0.82	0.74
Restaurant	0.04	0.24	0.06	0.05	0.33	0.09
Home	0.92	0.74	0.82	0.97	0.87	0.92
Bike	0.22	0.33	0.26	0.55	0.40	0.46
Car	0.0	0.0	0.0	0.0	0.0	0.0
Store	0.11	0.41	0.18	0.0	0.0	0.0
Metro	0.09	0.40	0.16	0.37	0.55	0.44
Station	0.09	0.11	0.10	0.33	0.20	0.25
Motorcycle	0.05	0.08	0.06	0.44	1.0	0.62
Running	0.17	0.71	0.27	0.11	0.27	0.15
Park	0.04	0.50	0.08	0.33	0.50	0.40

Table 16: Performance of Multi-view Learner with Location Correction and Post-processing on Participants' data

class	MVP + LH			MVP + LO		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.89	0.78	0.83	0.88	0.80	0.84
Bus	1.0	0.53	0.69	0.95	0.56	0.71
Office	0.94	0.93	0.93	0.92	0.95	0.94
Restaurant	0.51	0.60	0.55	0.49	0.60	0.54
Home	0.96	0.98	0.97	0.97	0.98	0.98
Bike	0.67	0.54	0.59	0.68	0.62	0.65
Car	0.84	0.15	0.26	0.84	0.15	0.26
Store	0.78	0.54	0.64	0.71	0.41	0.52
Metro	0.78	0.49	0.60	0.83	0.64	0.72
Station	0.42	0.14	0.20	0.35	0.16	0.22
Motorcycle	0.74	0.35	0.47	0.67	0.31	0.42
Running	0.24	0.30	0.27	0.41	0.60	0.49
Park	0.32	0.96	0.48	0.31	0.96	0.47

view-based models show a key advantage compared to the baseline models, which can be explained by the fact that the meta-learner allows weighting the predictions of the fist-level learners, thus eliminating the effects of the missed dimensions. To this end, considering as well the high computation cost of the MLSTM-FCN and KNN-DTW models, we judge that the the baseline approaches are not qualified as appropriate models for predicting the micro-environments.

For privacy-invasive modes, we show in Table 16 and Figure 13 the performance of the MVP model when adopting the location-correction techniques (i.e., LO and LH). Compared to the MVP model's performance reported in

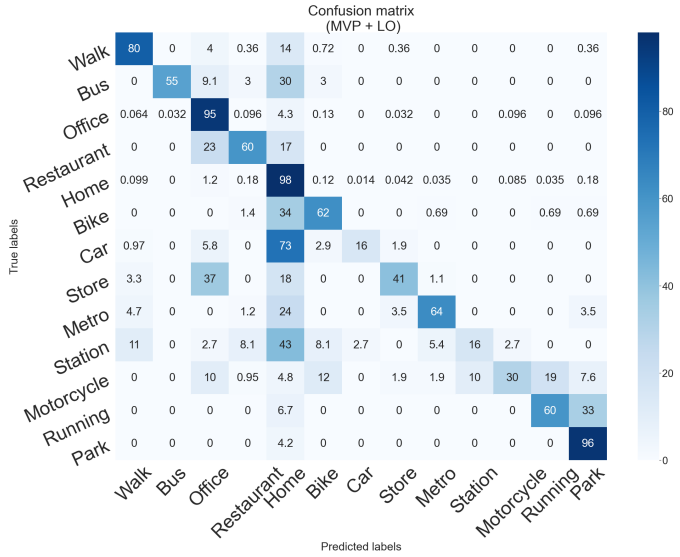


Fig. 13: MVP + Office location Correction confusion matrix

Table 12, we observe that adding the location corrections of home or office not only leads to better predictions on the target classes (i.e., home, office) but also improves the general model performance. Moreover, the location of the office (LO) helps the MVP model achieve better predictions in most classes than the location of home (LH), which is coherent with our conclusion in Section 7.5.1 where the MVP+LO model shows the best accuracy for most campaign participants. However, even though the location information allows to greatly improve the model's performance, the privacy stays as a crucial issue during both the data collection and data application process. In practice, a trade-off between the privacy and model performance should be considered.

In conclusion, MVP with location correction and MVP classifiers have comparable results. Although location corrections (i.e., LH and LO) can improve the model's performance, those location data are not always available due to privacy issues.

7.6 Model Generalization

In practice, we should consider the model generalization on unseen data, which allows evaluating the model in more complex scenarios. We have used the multi-view model (which have been trained over RECORD campaign data) to classify data that have never been seen by it before. We opt for the VGP campaign data, which was collected during a different time period from RECORD, to prove the generalization ability of the proposed model. For the VGP campaign, we don't have the ground truth for the data, so we have plotted the pre-

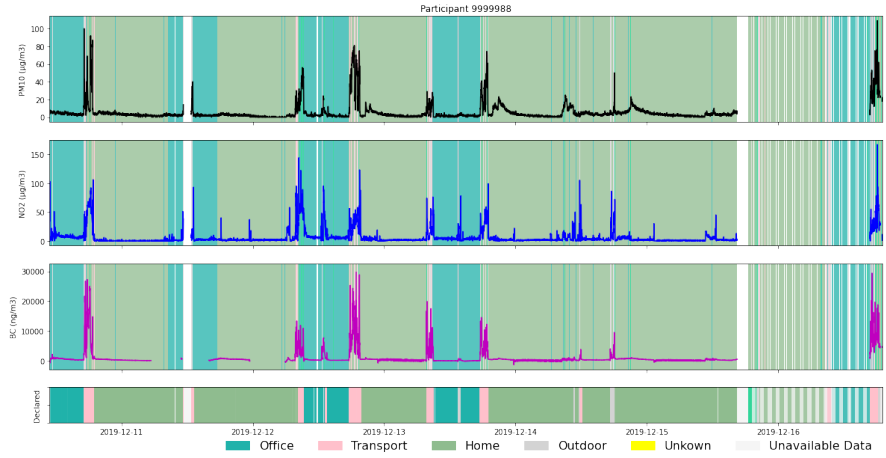


Fig. 14: Predictions of VGP campaign for participant 9999988.

dictions versus the declared activities (which is not guaranteed to be accurate). Figure 14 shows the plot of declared versus predicted micro-environments. For this participant (i.e. participant 9999988), we trust his/her annotations, so we can notice that the model has performed well. While for figure 15, as we don't have the real ground truth, we can see that the model's predictions are more reliable than the annotations. For instance, the participant in the plot has declared three times staying outdoors in the middle of the night (i.e. 24, 25 and 26th of October 2019), which is very unlikely to be true. Some other participants may completely forget to annotate the change of micro-environment, so the declared annotations are indeed imperfect.

8 Discussions & Perspectives

In this section, we discuss the perspectives for improving our multi-view learning model and the possibility for tackling the practical label issue in the context of Polluscope.

8.1 Multi-view Learner

The multi-view learner adopted in this paper is composed of the base learner (i.e., Random Forest) and the meta-learner (i.e., Random Forest), which has greatly improved the performance compared to the single kNN-DTW classifier. The objective of this paper is not to propose the best classifier for MTS classification, but to provide an insight that the multi-view learner is capable of coordinating effectively the information from different variables and achieving more reliable performance than a single base learner. Moreover, the results

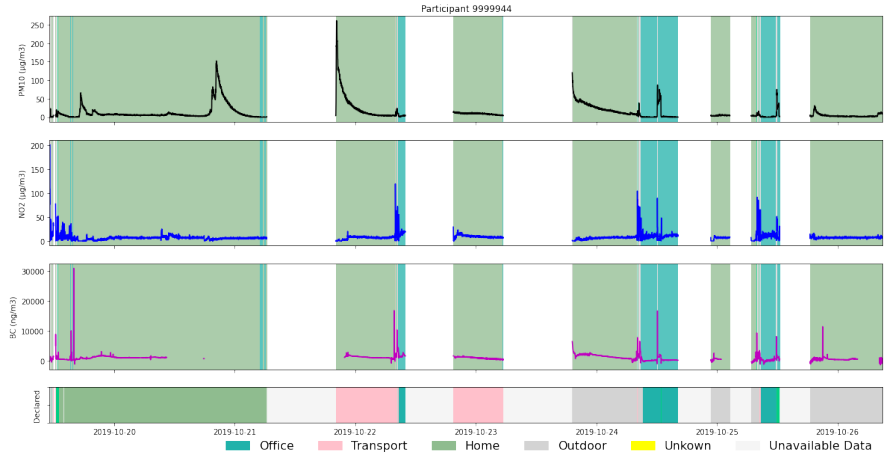


Fig. 15: Predictions of VGP campaign for participant 9999944.

of the grouping approach which is based on the multi-view approach confirms that there is a clear signature for each micro-environment, thus we can have an effective prediction with this approach. Moreover, the multi-view approach offers the reusability of the first-level learners, and allows using different classifiers and combinations for the first-level learners. Multi-view model doesn't require a special hardware such as GPU for training Neural Networks (i.e. MLSTM-FCN). In addition, it does not require a long execution time for classification as other classifiers such as KNN-DTW do. Besides, using the multi-view approach allows the prediction of micro-environments in the absence of some dimensions in the data. Another advantage of using the multi-view approach is that its meta-learner is trained on out-of-fold predictions thus the model will not over fit.

Nevertheless, the kNN-DTW is considered as the baseline for MTS classification and is widely outpaced by the advanced approaches such as Shapelets [49, 59, 60] or the frequent patterns [32]. Essentially, the kNN-DTW captures the global feature based on the distance measure between the entire sequences, while the local features (e.g., the frequent patterns [32], the interval features [12], Shapelets [49], etc.) are more appropriate when a specific pattern characterizes a class. More specifically, a combination of features extracted from different domains may dramatically improve the performance of the base learner [29]. Therefore, one of the perspectives consists of the **optimization** of the base learner and the exploration of the **explainability** of the multi-view learner on both the feature interpretation and the variable importance for building the classifier. For this reason, we have removed the NO2 and BC dimensions to show their importance for some classes. Table 17 shows the precision, recall, and F1 score for MVB while removing some dimensions (NO2 and BC) compared to the MVB model containing all dimensions. The compar-

Table 17: Performance of MVB without NO2 and BC VS. MVB

class	MVB without NO2 and BC			MVB		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Walk	0.74	0.76	0.75	0.82	0.79	**0.80**
Bus	0.58	0.54	0.56	0.85	0.64	**0.73**
Office	0.78	0.74	0.76	0.86	0.85	**0.85**
Restaurant	0.43	0.60	**0.50**	0.42	0.60	**0.50**
Home	0.92	0.93	0.93	0.95	0.95	**0.95**
Bike	0.49	0.47	0.48	0.57	0.61	**0.59**
Car	0.34	0.15	0.20	0.51	0.18	**0.27**
Store	0.54	0.57	0.55	0.61	0.61	**0.61**
Metro	0.52	0.60	0.55	0.62	0.70	**0.66**
Station	0.10	0.12	0.11	0.16	0.17	**0.16**
Motorcycle	0.25	0.07	0.11	0.33	0.08	**0.12**
Running	0.32	0.61	**0.42**	0.30	0.61	0.40
Park	0.26	0.89	0.41	0.32	0.86	**0.47**

ison shows that the F1-score of the MVB model for all classes is greater than that model without NO2 and BC. Except for *Running* class which is only one point difference. This comparison shows the importance and role of those dimensions (NO2 and BC) in micro-environment prediction. We have chosen to remove NO2 and BC because depending on figure 7, these 2 dimensions have the highest accuracy compared to other dimensions. The visual representation of Shapelets make them good candidates for such improvement.

8.2 Label Shortage Issue

The label shortage is a practical issue when building the learning model. In the context of Polluscope particularly, post-labelling for time series sensor data is much more costly than classic data (e.g., image, text, etc.) due to the low interpretability over the real-valued sequence. Therefore, the data need to be annotated during the data collection process. However, certain practical factors limit the availability of labels. For instance, the participants are not always conscious in annotating their micro-environment. Therefore, for certain time periods, no annotations were marked.

In order to give an insight about the consistency between the labeled and unlabeled data, and to see if the unlabeled data are valuable for improving the classifier's performance in our context, we conduct a preliminary test on the Polluscope data with the newly proposed semi-supervised MTSC model TapNet [53].

TapNet [53] is a deep learning based approach designed for multivariate time series classification. By adopting the prototypical network [41], TapNet allows learning a low-dimensional embeddings for the input MTS where the unlabelled samples help adjust the class prototype (i.e., class centroid), which leads to a better classifier than using only the labelled samples. Table 18 shows

the semi-supervised learning results on Polluscope data considering or not the *speed* variable. We evaluate the performance of TapNet under different supervision ratios in the training set. The results show that the unlabeled samples and the *speed* variable do improve the performance of the classifier. Besides, the accuracy didn't drop a lot when eliminating the annotations in the training set (from ratio=1 for fully labelled to 0.5, and even for 0.2 when only 20% data is labelled), indicating that the collected data within each class is not sparsely distributed. Thus learning under weak supervision is reliable with the aid of the unlabeled samples.

Table 18: The accuracy results of TapNet on Polluscope data under different supervision ratios

Condition	Sup_ratio=1	Sup_ratio=0.5	Sup_ratio=0.2
Speed	0.746	0.725	0.717
No speed	0.713	0.703	0.695

Giving the promising results on the data distribution consistency, another avenue worth exploring is to consider and integrate a semi-supervised model into our multi-view learner. Various semi-supervised frameworks are applicable to our model, such as applying self-learning [47] to produce the pseudo labels on the multi-view learner, or adopting the label propagation and manifold regularization techniques [17] on the base learner.

9 Conclusion

Activity recognition has gained the interest of many researchers nowadays, due to the widespread use of mobility sensors. Micro-environment recognition is essential in MCS projects such as Polluscope, in order to analyse the individual's exposure to air pollution and to relate it to his/her micro-environment. The major finding of our study is to show to some extent that the environmental observations can characterize the micro-environment. Moreover, the accuracy of the model is high enough to consider an automatic detection of the micro-environment without burdening the participants with self-reporting. By using the mobility feature as a time series, the accuracy improves slightly though the gain is moderate. Therefore, we can keep characterizing the micro-environment even in the absence of the speed dimension.

We employed different approaches and learners, and conducted a thorough experimental study, which shows the efficiency of the multi-view approach for time series classification, even though some dimensions are missing. We have also compared the results with the MLSTM-FCN and kNN-DTW classifiers which were considered as the baselines. During the training phase, MLSTM-FCN, on the one hand, showed promising results that could not be confirmed during the phase of applying the model on new data due to over-fitting. kNN-DTW, on the other hand, was not comparable, plus it is not suitable because

of its time consumption. Furthermore, training on a previous data set was biased by the quality of the annotation. But this limitation was overcome by using more reliable data that we did not have before.

Furthermore, we extend the proposed approach to include the detection of stay locations based on trajectory segmentation into stop and move segments. The move segments were labeled by the type of transportation mode. We combine time series plus trajectory data as pre-processing and post-processing layers to bring the best of them.

In addition, we present two optimisation methods which are either privacy friendly or privacy invasive. The later approach adds the private location of home and office in post-processing, whilst the first approach uses a priori rules. According to our experiments, we highly recommend using the privacy friendly models due to their ability to respect the private lives of the participants.

Acknowledgements This work has supported by the French National Research Agency (ANR) project Polluscope, funded under the grant agreement ANR-15-CE22-0018, by the H2020 EU GO GREEN ROUTES funded under the research and innovation programme H2020- EU.3.5.2 grant agreement No 869764, and by the DATAIA convergence institute project StreamOps, as part of the Programme d' Investissement d'Avenir, ANR-17-CONV-0003. Part of the equipment was funded by iDEX Paris-Saclay, in the framework of the IRS project ACE-ICSEN, and by the Communauté d'agglomération Versailles Grand Parc – VGP - (www.versaillesgrandparc.fr). We are thankful to VGP (Thomas Bonhoure) for facilitating the campaign. We would like to thank all the members of the Polluscope consortia who contributed in one way or another to this work: Salim Srairi and Jean-Marc Naude (CEREMA) who conducted the campaign; Boris Dessimond and Isabella Annesi-Maesano (Sorbonne University) for their contribution to the campaign; Valerie Gros and Nicolas Bonnaire (LSCE), and Anne Kauffman and Christophe Debert (Airparif) for their contribution in the periodic qualification of the sensors and their active involvement in the project. Finally, we would like to thank the participants for their great effort in carrying the sensors, without whom this work would not be possible.

References

1. Abboud, M., Hafyani, H.E., Zuo, J., Zeitouni, K., Taher, Y.: Micro-environment recognition in the context of environmental crowdsensing. Proceedings of the Workshops of the EDBT/ICDT 2021 Joint Conference **2841** (2021)
2. Antoniou, A., Storkey, A., Edwards, H.: Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340 (2017)
3. Asimina, S., Chapizanis, D., Karakitsios, S., Kontoroupis, P., Asimakopoulos, D., Maggos, T., Sarigiannis, D.: Assessing and enhancing the utility of low-cost activity and location sensors for exposure studies. Environmental monitoring and assessment **190**(3), 1–12 (2018)
4. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: KDD workshop, vol. 10, pp. 359–370. Seattle, WA, USA: (1994)
5. Chaix, B., Kestens, Y., Bean, K., Leal, C., Karusisi, N., Meghiref, K., Burban, J., Fon Sing, M., Perchoux, C., Thomas, F., et al.: Cohort profile: residential and non-residential environments, individual activity spaces and cardiovascular risk factors and diseases—the record cohort study. International journal of epidemiology **41**(5), 1283–1292 (2012)
6. Chaix, B., Kestens, Y., Perchoux, C., Karusisi, N., Merlo, J., Labadi, K.: An interactive mapping tool to assess individual mobility patterns in neighborhood studies. American journal of preventive medicine **43**(4), 440–450 (2012)

7. Chatzidiakou, L., Krause, A., Kellaway, M., Han, Y., Li, Y., Martin, E., Kelly, F.J., Zhu, T., Barratt, B., Jones, R.L.: Automated classification of time-activity-location patterns for improved estimation of personal exposure to air pollution (2022)
8. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: Smote: Synthetic minority oversampling technique. J. Artif. Intell. Res. (JAIR) **16**, 321–357 (2002). DOI 10.1613/jair.953
9. Chen, K., Zhang, D., Yao, L., Guo, B., Yu, Z., Liu, Y.: Deep Learning for Sensor-based Human Activity Recognition: Overview, Challenges and Opportunities. arXiv:2001.07416 [cs] (2020). URL http://arxiv.org/abs/2001.07416. ArXiv: 2001.07416
10. Cho, H., Yoon, S.M.: Divide and conquer-based 1d cnn human activity recognition using test data sharpening. Sensors **18**(4), 1055 (2018)
11. Dabiri, S., Heaslip, K.: Inferring transportation modes from gps trajectories using a convolutional neural network. Transportation research part C: emerging technologies **86**, 360–371 (2018)
12. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Information Sciences **239**, 142–153 (2013)
13. Do, T.M.T., Gatica-Perez, D.: The places of our lives: Visiting patterns and automatic labeling from longitudinal smartphone data. IEEE Transactions on Mobile Computing **13**(3), 638–648 (2013)
14. Do, T.M.T., Gatica-Perez, D.: The Places of Our Lives: Visiting Patterns and Automatic Labeling from Longitudinal Smartphone Data. IEEE Transactions on Mobile Computing **13**(3), 638–648 (2014). DOI 10.1109/TMC.2013.19
15. El Hafyani, H., Abboud, M., Zuo, J., Zeitouni, K., Taher, Y.: Tell me what air you breath, i tell you where you are. In: 17th International Symposium on Spatial and Temporal Databases, SSTD '21, p. 161–165. Association for Computing Machinery, New York, NY, USA (2021). DOI 10.1145/3469830.3470914. URL https://doi.org/10.1145/3469830.3470914
16. El Hafyani, H., Zeitouni, K., Taher, Y., Abboud, M.: Leveraging change point detection for activity transition mining in the context of environmental crowdsensing. The 9th SIGKDD International Workshop on Urban Computing (2020)
17. van Engelen, J.E., Hoos, H.: A survey on semi-supervised learning. Machine Learning **109**, 373–440 (2019)
18. Etemad, M., Soares Júnior, A., Matwin, S.: Predicting transportation modes of gps trajectories using feature engineering and noise removal. In: Advances in Artificial Intelligence: 31st Canadian Conference on Artificial Intelligence, Canadian AI 2018, Toronto, ON, Canada, May 8–11, 2018, Proceedings 31, pp. 259–264. Springer (2018)
19. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data Mining and Knowledge Discovery **33**(4), 917–963 (2019)
20. Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F.: Inceptiontime: Finding alexnet for time series classification. ArXiv **abs/1909.04939** (2020)
21. Garcia-Ceja, E., Galván-Tejada, C.E., Brena, R.: Multi-view stacking for activity recognition with sound and accelerometer data. Information Fusion **40**, 45–56 (2018). DOI 10.1016/j.inffus.2017.06.004. URL http://www.sciencedirect.com/science/article/pii/S1566253516301932
22. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
23. Guo, B., Wang, Z., Yu, Z., Wang, Y., Yen, N.Y., Huang, R., Zhou, X.: Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm. ACM computing surveys (CSUR) **48**(1), 1–31 (2015)
24. Jiang, W., Yin, Z.: Human activity recognition using wearable sensors by deep convolutional neural networks. In: Proceedings of the 23rd ACM international conference on Multimedia, pp. 1307–1310 (2015)
25. Karim, F., Majumdar, S., Darabi, H., Harford, S.: Multivariate lstm-fcns for time series classification. Neural Networks **116**, 237–245 (2019)

26. Kranz, M., Möller, A., Hammerla, N., Diewald, S., Plötz, T., Olivier, P., Roalter, L.: The mobile fitness coach: Towards individualized skill assessment using personalized mobile devices. Pervasive and Mobile Computing **9**(2), 203–215 (2013)
27. Languille, B., Gros, V., Bonnaire, N., Pommier, C., Honoré, C., Debert, C., Gauvin, L., Srairi, S., Annesi-Maesano, I., Chaix, B., et al.: A methodology for the characterization of portable sensors for air quality measure with the goal of deployment in citizen science. Science of the Total Environment **708**, 134698 (2020)
28. Li, S., Li, Y., Fu, Y.: Multi-view time series classification: A discriminative bilinear projection approach. Proceedings of the 25th ACM International on Conference on Information and Knowledge Management pp. 989–998 (2016)
29. Lines, J., Taylor, S., Bagnall, A.: HIVE-COTE: The Hierarchical Vote Collective of Transformation-based Ensembles for Time Series Classification. In: 2016 IEEE 16th international conference on data mining (ICDM), pp. 1041–1046 (2016)
30. Liu, L., Peng, Y., Wang, S., Liu, M., Huang, Z.: Complex activity recognition using time series pattern dictionary learned from ubiquitous sensors. Information Sciences **340-341**, 41–57 (2016). DOI 10.1016/j.ins.2016.01.020. URL http://www.sciencedirect.com/science/article/pii/S0020025516000311
31. Moon, B., Jagadish, H.V., Faloutsos, C., Saltz, J.H.: Analysis of the clustering properties of the hilbert space-filling curve. IEEE TKDE'01 **13**(1), 124–141 (2001)
32. Nayak, G., Mithal, V., Jia, X., Kumar, V.: Classifying multivariate time series by learning sequence-level discriminative patterns. In: Proceedings of the 2018 SIAM International Conference on Data Mining, pp. 252–260. SIAM (2018)
33. Pappalardo, L., Simini, F., Barlacchi, G., Pellungrini, R.: scikit-mobility: a python library for the analysis, generation and risk assessment of mobility data (2019)
34. Parent, C., Spaccapietra, S., Renso, C., Andrienko, G., Andrienko, N., Bogorny, V., Damiani, M.L., Gkoulalas-Divanis, A., Macedo, J., Pelekis, N., et al.: Semantic trajectories modeling and analysis. ACM Computing Surveys (CSUR) **45**(4), 1–32 (2013)
35. Pärkkä, J., Ermes, M., Korpipää, P., Mäntyjärvi, J., Peltola, J., Korhonen, I.: Activity classification using realistic data from wearable sensors. IEEE transactions on information technology in biomedicine: a publication of the IEEE Engineering in Medicine and Biology Society **10**(1), 119–128 (2006). DOI 10.1109/titb.2005.856863
36. Rehrl, K., Gröchenig, S., Kranzinger, S.: Why did a vehicle stop? a methodology for detection and classification of stops in vehicle trajectories. International Journal of Geographical Information Science **34**(10), 1953–1979 (2020)
37. Ruiz, A.P., Flynn, M., Bagnall, A.: Benchmarking Multivariate Time Series Classification Algorithms. arXiv:2007.13156 [cs, stat] (2020). URL http://arxiv.org/abs/2007.13156. ArXiv: 2007.13156
38. Ruiz, A.P., Flynn, M., Large, J., Middlehurst, M., Bagnall, A.: The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery **35**(2), 401–449 (2021)
39. Sai, K.B.K., Subbareddy, S.R., Luhach, A.K.: Iot based air quality monitoring system using mq135 and mq7 with machine learning analysis. Scalable Computing: Practice and Experience **20**(4), 599–606 (2019)
40. Sardianos, C., Varlamis, I., Bouras, G.: Extracting user habits from google maps history logs. In: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 690–697. IEEE (2018)
41. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems, vol. 30, pp. 4077–4087. Curran Associates, Inc. (2017)
42. Sonawani, S., Patil, K., Chumchu, P.: No2 pollutant concentration forecasting for air quality monitoring by using an optimised deep learning bidirectional gru model. International Journal of Computational Science and Engineering **24**(1), 64–73 (2021)
43. Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., Woods, E.: Tslearn, a machine learning toolkit for time series data. Journal of Machine Learning Research **21**(118), 1–6 (2020). URL http://jmlr.org/papers/v21/20-091.html
44. Toch, E., Lerner, B., Ben-Zion, E., Ben-Gal, I.: Analyzing large-scale human mobility data: a survey of machine learning methods and applications. Knowledge and Information Systems **58**(3), 501–523 (2019)

45. Wang, B., Jiang, T., Zhou, X., Ma, B., Zhao, F., Wang, Y.: Time-Series Classification Based on Fusion Features of Sequence and Visualization. Applied Sciences **10**(12), 4124 (2020). DOI 10.3390/app10124124. URL https://www.mdpi.com/2076-3417/10/12/4124
46. Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L.: Deep Learning for Sensor-based Activity Recognition: A Survey. Pattern Recognition Letters **119**, 3–11 (2019). DOI 10.1016/j.patrec.2018.02.010. URL http://arxiv.org/abs/1707.03502. ArXiv: 1707.03502
47. Wei, L., Keogh, E.: Semi-supervised time series classification. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '06, pp. 748–753. Association for Computing Machinery, New York, NY, USA (2006). DOI 10.1145/1150402.1150498. URL https://doi.org/10.1145/1150402.1150498
48. Wolpert, D.H.: Stacked generalization. Neural Networks **5**(2), 241 – 259 (1992). DOI https://doi.org/10.1016/S0893-6080(05)80023-1. URL http://www.sciencedirect.com/science/article/pii/S0893608005800231
49. Ye, L., Keogh, E.: Time series shapelets: A New Primitive for Data Mining. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09 p. 947–956 (2009)
50. Yoon, J., Jarrett, D., van der Schaar, M.: Time-series generative adversarial networks. In: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019). URL https://proceedings.neurips.cc/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf
51. Zhang, M., Sawchuk, A.A.: Motion primitive-based human activity recognition using a bag-of-features approach. In: Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium, IHI '12, pp. 631–640. Association for Computing Machinery, New York, NY, USA (2012). DOI 10.1145/2110363.2110433. URL https://doi.org/10.1145/2110363.2110433
52. Zhang, M., Sawchuk, A.A.: Motion primitive-based human activity recognition using a bag-of-features approach. In: Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium, pp. 631–640 (2012)
53. Zhang, X., Gao, Y., Lin, J., Lu, C.T.: TapNet: Multivariate Time Series Classification with Attentional Prototypical Network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 6845–6852 (2020)
54. Zheng, Y.: Trajectory data mining: An overview. ACM Trans. Intell. Syst. Technol. **6**(3) (2015). DOI 10.1145/2743025. URL https://doi.org/10.1145/2743025
55. Zheng, Y., Li, Q., Chen, Y., Xie, X., Ma, W.Y.: Understanding mobility based on GPS data. In: Proceedings of the 10th international conference on Ubiquitous computing, pp. 312–321. Association for Computing Machinery, New York, NY, USA (2008). URL https://doi.org/10.1145/1409635.1409677
56. Zheng, Y., Liu, L., Wang, L., Xie, X.: Learning transportation mode from raw gps data for geographic applications on the web. In: Proceedings of the 17th international conference on World Wide Web, pp. 247–256 (2008)
57. Zheng, Y., Zhang, L., Ma, Z., Xie, X., Ma, W.Y.: Recommending friends and locations based on individual location history. ACM Transactions on the Web (TWEB) **5**(1), 1–44 (2011)
58. Zhou, Z.H.: Ensemble Methods: Foundations and Algorithms. CRC press (2012)
59. Zuo, J., Zeitouni, K., Taher, Y.: Exploring interpretable features for large time series with se4tec. In: Proc. EDBT, pp. 606–609 (2019)
60. Zuo, J., Zeitouni, K., Taher, Y.: Incremental and adaptive feature exploration over time series stream. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 593–602 (2019)