Machine Learning Approach for Mobility Context Classification using Radio Beacons

Jana Koteich, Nathalie Mitton

Machine Learning Approach for Mobility Context Classification using Radio Beacons

Jana Koteich
Inria, France
jana.koteich@inria.fr

Nathalie Mitton
Inria, France
nathalie.mitton@inria.fr

***Abstract*—The study of human mobility becomes more and more crucial these days in transportation studies, urban planning, crowd mobility behaviors, and even more. In this paper, we propose a novel approach for studying human mobility by building a light machine learning (ML) model using observation of wireless networking information from WiFi and Bluetooth low energy (BLE) that are today naturally present in everyday devices such as mobile phones. Our goal is to build a mobility classification system using communicating devices of any kind with low processing complexity. However, we propose a new approach for mobility classification using a real dataset of WiFi and BLE beacons collected over one year for around 90 hours in different scenarios and conditions. The first model (B-model) aims to identify the status of a device if stationary or mobile. Then a complementary model (M-model) is applied to determine a more precise real-life situation of the device, which could be a Home, Office, Bus, Train, etc. The results show that decision-tree-based ensemble ML algorithms like LGBMClassifier and XGBClassifier gave the best results, in terms of accuracy and f1 score for both models with an accuracy of 99% and 94% respectively, confirming the capability of classifying mobility context from only WiFi and BLE data. We believe that such an approach could be leveraged for studying human mobility and an important step towards the large deployment of mobility-based applications by leveraging everyday mobile phones.**

***Index Terms*—datasets, machine learning, wireless network, IoT, Mobility Model**

I. Introduction

Understanding and modeling humans and device mobility have fundamental importance in mobile computing, with implications ranging from network design and location-aware technologies to urban infrastructure planning [1]. So, inferring mobility states such as being stationary, walking, or driving is critical for several applications. The fact that these days users carry several devices such as smartphones, laptops, and smartwatches equipped with radio communication technologies with each device offering a different set of services resulting in different usage and mobility, provides new approaches for studying human mobility.

Thus, several researchers have made the effort to study human mobility to determine people's fine-grained activities like using GPS positioning [2], but GPS-based mobility characterization raises many issues such as spotty coverage and battery consumption [3]. Other attempts and tools have been developed to predict global mobility [4] in general or only the next step [5]. Some studies [6] aimed to characterize people's mobility based on data traffic but only for a particular subset of people (students in higher education) and over very limited areas. Recently, some studies investigated the use of human mobility but mainly in the COVID-19 context to anticipate contamination [7]. Such approaches are different in the sense that they mainly aim to trace contacts between devices and not necessarily their mobility. Thus in our approach, we provide a novel technique to determine the real-life situation of a device. The novelty of such an approach is mainly characterized by its applicability, since nowadays almost all devices support WiFi and BLE, so no need for external hardware or module. Thus, by monitoring the behavior of WiFi and BLE in a device's range, we are able to infer a device's actual status. To the best of our knowledge, non of the previous approaches have modeled real-life situations of a device through WiFi and Bluetooth Low Energy (BLE) beacons only. We believe such a seamless approach could be an important lever for studying human mobility and deriving related services. In this paper, we investigate how wireless networks can be leveraged to infer a device's actual status by analyzing the behavior of wireless links within its range.

We examine various wireless technologies and data collection methods and discuss how this data can be analyzed to gain insights into crowd behavior. To this end, to reach the main goal which is a model that can determine the real-life situation of a device, we propose a joint ML-based method. The first model is called B-model, as its goal is to determine if a device is stationary or mobile, then the output of this model will be one of the input features for the second model which is called M-model to guess the real-life situation of a device. The models are trained using only WiFi and BLE beacons jointly. Mainly the contact duration of an access point (AP) with the scanner, the received signal strength indication (RSSI), and the MAC address are used for feature extraction for training the models. Training the models goes through two main steps. The first one is to specify the best period needed to ascertain the real-life situation of a device with the highest accuracy. Then justifying the need of having both technologies, WiFi, and BLE for inferring the output. The results showed that 1 minute is the minimum time to determine if a device is stationary or mobile, while two minutes for determining the real-life situation of a device, while using WiFi and BLE jointly gives better accuracy. The models are evaluated with a dataset of 80 hours gathered in different conditions and scenarios over one year. The methodology for collecting the

dataset is presented with a description of the collected data so far. Then we introduce the first model (*B-model*) that will determine whether a device is in a static or mobile scenario, then the second model (*M-model*) will determine the context of the device which could be in a car or bus or home or restaurant, etc. The results show that decision-tree-based ensemble ML algorithms like LGBMClassifier and XGBClassifier gave the best results, in terms of accuracy and f1 score for both models with an accuracy of 99% and 94% respectively, confirming the capability of determining mobility context from only WiFi and BLE data.

The rest of the paper is organised as follows: Section II introduces the motivation and model overview, then section III introduces the setup and methodology of data collection. Section IV presents the models for classifying the network context. Section V presents the results of the training. Section VI presents a general review of the state of the art. Finally, Section VII concludes the paper with future work.

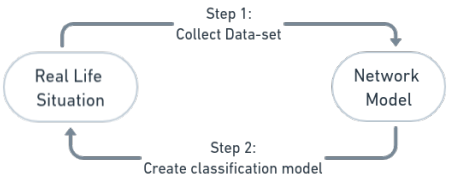


Fig. 1: Transition between real-life context and network model

II. Motivation

In this section, we motivate the concept behind studying mobility context through radio beacons, and how it is possible to leverage wireless links to determine a network context. A wireless connection between two devices can be established if and only if they are close enough to communicate. The maximum distance required between them to establish a contact depends in particular on the environment and the communication technology. So we believe the number of direct links a single device can establish at a given time and the rate at which these links break and appear are strong indicators of the context in which a device evolves (static vs mobile, urban vs rural environment, isolated vs social, transport publication vs individual locomotion, velocity, etc). Figure 2 illustrates two different perspectives in a single scenario. Connections are observed either from a device held by a pedestrian (Fig. 2a) or from a device traveling in a bus (Fig. 2b). When in a bus, several stable short range communications can be established with other bus passengers. They are completed by a set of longer range intermittent communications that can be sporadically established with devices exterior to the bus, such as pedestrians or cars. On the contrary, when held by a pedestrian, there are more or less stable short range communications and few brief longer range communications. Thus the networking view is defined as $(nb_i, r_i)_{0<i<n}$ which assesses the number of connections nb_i to be established with communication technology i and the stability r_i of them, where n is the maximum number of different communication technologies observed in the network. Thus, from a network perspective we can translate the mobility and the surrounding environments of a device into a network model, i.e. when a user walks in an urban area, on average, what and how many connections are they supposed to have and at what rate are they changing. Same question when a user is in a bus, in a cab, biking, etc and in diverse and various scenarios.

To achieve this, we need first to observe the variations of different wireless links in different scenarios. This requires collecting dataset from each real-life context. The idea is to translate the real-life situation into a network model, as later from the network model, we will be able to guess the real-life situation of a device (Fig. 1). To this end, in the next section the methodology and description of the dataset collected so far is illustrated.

III. Data Collection

Data collection is considered as the foundation of the Machine Learning model building. This section covers the set-up used to collect the dataset and illustrates the data collection process with a description of the data. Note that the storage of data and model training is performed off-line and does not need to be embedded on the devices themselves that will just receive the outcome they will run on new observations they will make.

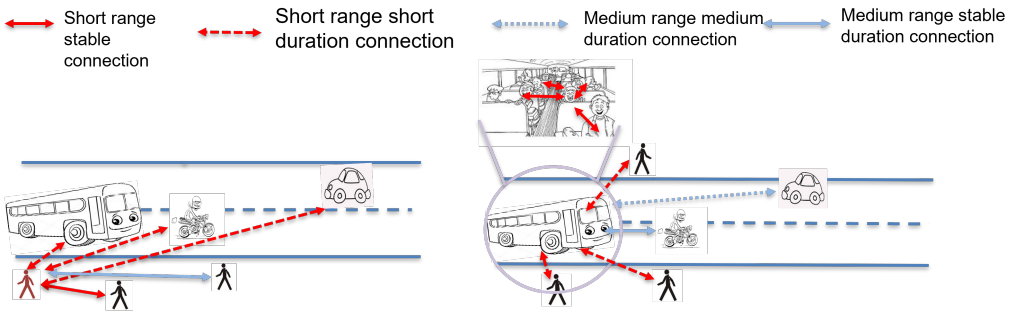
A. Methodology

The dataset is collected in different scenarios, with different variations. FiPy miro-controllers from pycom are used to scan for wireless beacons. In [8], the dataset description is illustrated in details, but mainly we are concerned with the following features:

- Node W (N_W): Scans for WiFi APs every two seconds.
- Node B (N_B): Scans for Bluetooth devices every second.

The goal is to observe and record the variations of the wireless technologies in different mobility contexts, which are mainly categorized into two: *Static* and *Mobile* scenarios. For *Static* we define the following cases: Home, Office, Restaurant, Bus station, University and Meetings. For *Mobile* we have the following scenarios: Pedestrian, Car, Bus, Metro, and Trains. The data is collected using diverse scenarios, including both rural and urban areas, to ensure that it was representative of a wide range of environments and populations, and was not biased towards any particular group or location. The data is collected and saved with a timestamp and a label from where it was collected as Comma-separated values (CSV) files to be ready for the pre-processing stage. The configuration of each wireless technology is as follows:

a) WiFi Node (N_W): The WiFi node is configured to start active scanning, as the device radio transmits a probe request and listens for a probe response from an AP or active devices such as phones or laptops. Upon detecting a probe request, scanners log several pieces of information related to that probe. Figure 4 shows a sample of the saved logs of the received WiFi packets.



(a) Network observation from a pedestrian perspective (b) Network observation from a bus perspective

Fig. 2: Illustration of different network observations

b) Bluetooth Node (N_B): The BLE node is configured for passive scanning to receive the advertising packets (PDUs) that are retrieved every second. Figure 3 shows a sample of the saved logs for BLE beacons.

B. Dataset Deep Insights

Table I represents a sample of the collected dataset in different scenarios, as we can see, the environment of the collected data is different for the same label, as example for data collected from a bus context, we have the data collected in rural and urban areas, and same for the other scenarios as illustrated in the table.

IV. Model Architecture

In this section, the model architecture is illustrated. As explained in Section II, the aim is to extract knowledge from natural crowd mobility through radio beacons, by translating each "real-life" situation into a network model. As 'real-life' situation refers to the context of a network, which is the mobility of the environment of a device. Thus first we simplified the use-case by classifying the context into two main categories: *Static* and *Mobile*.

Static scenario is referenced to any fixed scenario like home, office, restaurant, library, etc. For further explanation, device A that is considered as static, could be a fixed sensor located somewhere, or a device held by a person in an office, though the person could be moving in their office sometimes, but still will be considered as stationary since their status is still in the office, (as no frequent movement over time is happening). On the other hand, mobile scenarios are assigned to devices that are in a moving context, like bus, car, train, pedestrian, etc. As well if a device is in a bus it will be considered as mobile, since our reference is the global context of the scenario which is the bus, and not the individual reference with other devices in the bus. To this end, to achieve the main goal which is building a ML model that can determine the real-life situation of a device (eg. being in a bus, or train, or pedestrian, or at home, etc.), two models are defined. First, since the input data is labeled, then our models will undergo supervised learning approaches. We started with the binary model called *B-model* that can determine the general situation of the device which could be static or mobile. The output of this model helped to improve the performance of the second main model which is called *M-model*, that stands for multi-class classification model, which aims to determine Ten different status of a device that are: Train, Home, Office, Conference, Bus, Car, Metro, Pedestrian, Restaurant, and University. The output of B-model will be one of the input features for M-model.

A. Data Pre-processing and Feature Engineering

As illustrated in Section III, the datasets are saved as csv files. Since each scenario (label) has two separately scanned files (WiFi and BLE) that share same time-stamp, the two files are combined together to facilitate the analysis and observation of both datasets at the same time. Now, for each scenario we have one file that includes the date from WiFi and BLE at a certain time-slot. Each dataset is in the form of an n x 6 matrix, where $n > 0$ is a variable number that equals the number of received beacons/probe-responses from all detected APs over the scanning time t. But feeding data into a model must be a column matrix and not an n x m matrix. So, we need to transform the n x 3 matrix into a 1 x $(f + 1)$ where f is the number of selected features from both WiFi and BLE (to be defined in the next section) plus at the end the *label*. Thus, in this case each dataset will be represented by a single row. To transform the shape of the dataset from 2D to 1D, the dataset undergo through two main phases:

- ϕ_1: Get the main statistics for each AP, thus as a result we will get an n x f_1 matrix, where n is the number of unique AP that appeared during scanning, and f_1 is the number of extracted features (defined in section IV-C). So, the dataset still has the 2D form at this phase.
- ϕ_2: Transform each 2D dataset from ϕ_2 to a 1D vector (explained in section IV-D).

B. WiFi and BLE Selected Features

During scanning, the device frequently receives beacons from the APs that are in its communication range. If the scanner moves away from the AP, after some time the connection between the two devices is lost and the scanner stops receiving beacons from that AP, and vice versa if the AP is mobile and

TABLE I: Records from Mobile and Static Scenarios

Scenarios	*Label*	*From*	*To*	*Duration*	*Description*
Bus	B1	12:20:00	12:55:00	35min	Autocar between city and village - crowded
	B3	17:07:04	17:56:00	49min	Bus in a city
	B5	20:06:00	20:38:00	32min	Bus in a city - very crowded
Car	C1	17:26:00	18:03:00	37min	Auto-Route - rural area
	C2	13:21:00	14:04:00	43min	Auto-Route then between houses in villages
	C3	09:06:00	09:24:00	18min	Auto-Route then between houses in villages
Train	T3	20:17:00	21:02:00	45min	TER between two cities
	T6	10:15:00	13:12:00	2hr, 57min	TGV
	T15	19:14:00	19:56:00	42min	TER between two cities
Pedestrian	P1	09:23:00	09:34:00	11min	University Campus
	P2	18:58:00	19:07:00	9min	Crowded city
	P4	12:55:00	13:08:00	13min	Rural area - Countryside
Home	H1	01:14:00	02:36:00	1hr, 22min	Student residence
	H2	12:31:00	13:33:00	1hr, 2min	Studio in a crowded city
	H3	08:08:00	08:43:00	35min	Apartment in a building - village
	H8	10:50:00	12:13:00	1hr, 23min	Hotel in a village - rural area

2022-06-09 17:43:02: {'adv_flag': None, 'def_tx_pwr': 3, 'mac': b'26...8d', 'rssi': -75, 'name': None, 'scan_tx_pwr': 3, 'conn_tx_pwr': 64, 'tx_range': None, 'adv_tx_pwr': 3}
2022-06-09 17:43:03: {'adv_flag': None, 'def_tx_pwr': 3, 'mac': b'26...8d', 'rssi': -81, 'name': None, 'scan_tx_pwr': 3, 'conn_tx_pwr': 64, 'tx_range': None, 'adv_tx_pwr': 3}
2022-06-09 17:43:04: {'adv_flag': None, 'def_tx_pwr': 3, 'mac': b'26....8d', 'rssi': -85, 'name': None, 'scan_tx_pwr': 3, 'conn_tx_pwr': 64, 'tx_range': None, 'adv_tx_pwr': 3}
2022-06-09 17:43:05: {'adv_flag': None, 'def_tx_pwr': 3, 'mac': b'26...8d', 'rssi': -84, 'name': None, 'scan_tx_pwr': 3, 'conn_tx_pwr': 64, 'tx_range': None, 'adv_tx_pwr': 3}
2022-06-09 17:43:05: {'adv_flag': None, 'def_tx_pwr': 3, 'mac': b'267...8d', 'rssi': -72, 'name': None, 'scan_tx_pwr': 3, 'conn_tx_pwr': 64, 'tx_range': None, 'adv_tx_pwr': 3}
2022-06-09 17:43:06: {'adv_flag': None, 'def_tx_pwr': 3, 'mac': b'267...8d', 'rssi': -69, 'name': None, 'scan_tx_pwr': 3, 'conn_tx_pwr': 64, 'tx_range': None, 'adv_tx_pwr': 3}

Fig. 3: BLE log file

2022-06-09 17:43:02: {'ssid':'edm', 'bssid':b'<Q\...\xa0', 'sec':5, 'channel':1, 'rssi':-66}
2022-06-09 17:43:02: {'ssid':'guest', 'bssid':b'<Q\...\xa2', 'sec':0, 'channel':1, 'rssi':-66}
2022-06-09 17:43:04: {'ssid':'lA', 'bssid':b'<Q\...\x84#', 'sec':0, 'channel':6, 'rssi':-83}
2022-06-09 17:43:04: {'ssid':'lA-intr', 'bssid':b'<Q\...\x84$', 'sec':5, 'channel':6, 'rssi':-83}
2022-06-09 17:43:06: {'ssid':'lA-guest', 'bssid':b'<Q\...\xa2', 'sec':0, 'channel':1, 'rssi':-64}
2022-06-09 17:43:06: {'ssid':'edm', 'bssid':b'<Q\...\xa0', 'sec':5, 'channel':1, 'rssi':-65}

Fig. 4: WiFi log file

the scanner is fixed. This behaviour is therefore considered an important metric for identifying the mobility of a device.

To verify the assumptions, Figures 5a and 5b represent the received beacons from each access point over time, with the corresponding RSSI. In Figure 5a which displays the WiFi data collected from an office (static scenario), beacons are detected over the whole scanning time. Knowing that the scanner is fixed, would indicate that the access points detected by the scanner are also fixed. While in Figure 5b which is related to the data collected from a bus (mobile scenario) in an urban area, the beacons are appearing only for a very short duration. This is because the scanning device is losing connection with the access point because of the mobility of the bus. From these observations, we can see how each scenario has its unique pattern of received beacons and the importance of the contact duration between the scanning device and surrounding APs in differentiating between different scenarios. Thus as a result, the time-stamp, the RSSI and MAC address will be selected for feature engineering, since they are the main attributes to give insights for the collected datasets.

C. Feature Extraction: Phase 1

In this section, data processing and feature extraction is illustrated in details. As mentioned in IV-B, contact duration will be calculated for each AP, with the mean and standard deviation of the RSSI. As a result, we will end up with two dataframes that represent the statistics of the collected data for each wireless technology.

1. Contact Duration: Let M be the set of all unique MAC addresses appeared during Time t of scanning. The contact duration is calculated as follows:

$$duration(m) = T(m), \quad \forall m \in M \tag{1}$$

Where T(m) is:

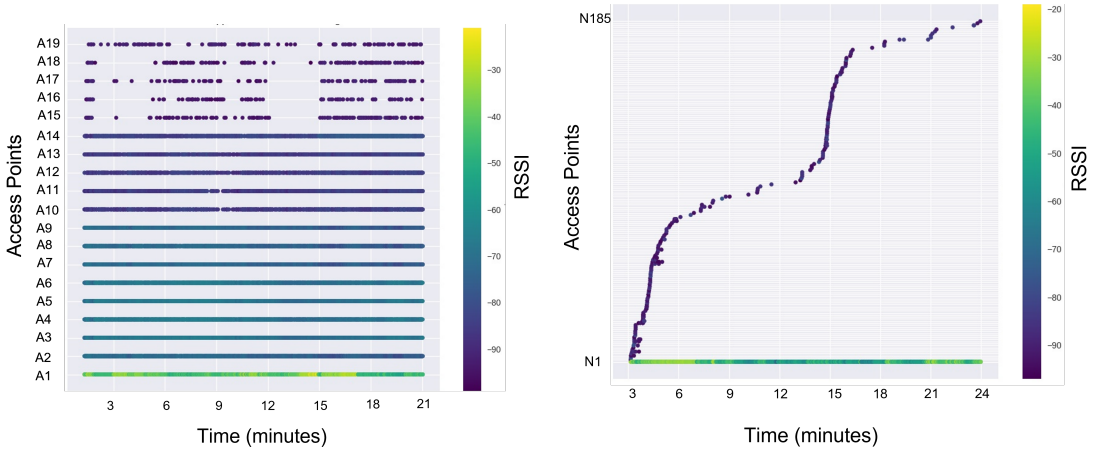
$$T(m) = t_i - t_0, \quad \forall m \in M \tag{2}$$

Where i is the last beacon appeared during the scanning.

2. Signal Strength Mean and Standard Deviation: The value of the RSSI tends to fluctuate even if the device is fixed due to external factors influencing radio waves (interference, diffraction, etc.), or when a Wi-Fi receiver is moving, the signal strengths it observes are noisier than when it is not moving. For this reason, for each AP, the average mean of the values recorded over time is calculated as follows:

$$\bar{\chi}_{x(m)} = \frac{1}{n}\sum_{i=1}^{n} x_i(m), \quad \forall m \in M \tag{3}$$

where $x(m)$ represents the RSSI of the MAC address *m*, x_i is the $i-th$ RSSI value in the sample, n is the total number of appearance of the beacon from the AP of the same mac



(a) WiFi beacons - Office monitoring

(b) WiFi beacons - Bus monitoring

Fig. 5: WiFi beacons in different scenarios

address during the scanning time, and x_i is the $i-th$ RSSI value in the sample. Then the standard deviation is calculated (eq. 4) to see how dispersed the data is in relation to the mean as this could indicate if the device is moving.

$$\delta_{x(m)} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{\chi}_{x(m)})^2}{n-1}} \tag{4}$$

D. Feature Extraction: Phase 2

Now as a result we have two data-frames that summarises the main information for each AP. Still we need to transform both 2D files to a 1D vector. Here is the second phase (ϕ_2) to get the important information from the dataframe for our model. Table II, shows the number of AP that appeared in different scenarios. We can see how the number of APs differs from one context to another as well as the average contact duration (Δ_t). Knowing that the contact duration is an important metric for differentiating between scenarios as mentioned in IV-B, then the dataframes will be transformed to a 1D vector by displaying the statistics of APs based on their contact duration (Δ_t). The features are mainly categorized into three main conditions: Long Δ_t, medium Δ_t, and short Δ_t based on the following criteria: First, get the percentage of contact duration as follows:

$$\%\Delta_t(m) = \frac{\Delta_t(m)}{t} \times 100 \quad \forall m \in M \tag{5}$$

1) L: Set of AP that has $\Delta_t > 70\%$ of the total time t.
2) M: Set of AP where $30\% < \Delta_t < 70\%$.
3) S: Set of AP that has a $\Delta_t < 30\%$ of the total time t.

Then for each Set (L, M, S), the mean and standard deviation of the RSSI of the access points that belongs to each set is calculated.

As a result we end up with a vector (V_i) that includes 24 features (12 features extracted from WiFi and 12 extracted from BLE), plus the label of the scanned scenario i. The same process is done for all the datasets that are collected in different scenarios, thus we end up with a dataframe of V_n input vectors, where n is the total number of datasets (see Figure 6).

TABLE II: Primary records from dataset analysis

	Wifi mac	**Wifi Δ_t**	**BLE mac**	**BLE Δ_t**
Office	19	9min, 16sec	27	4min, 27sec
Bus	185	17sec	43	2min, 32sec
Car	6	4min	6	3min, 56sec
Train	3	6min, 9sec	68	3min, 13sec

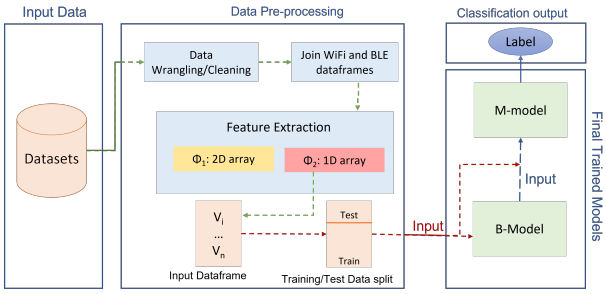


Fig. 6: Diagram that illustrates the ML process

V. Models Evaluation

As described in section III, the dataset is a labeled and collected over one year in different scenarios. The size of the dataset is 44,4 MB. The distribution of the datasets is unbalanced, to this end, with the available dataset for the moment, classical machine leaning algorithms will be suitable for our case since the dataset is small, and since it is a labeled dataset, supervised learning techniques will be applied to meet the final goal for constructing a classification model to guess the categorical label. In this section, several simulations are done to evaluate our model to select the one with the best performance. Knowing that each dataset is scanned with a

different time duration, first we want to divide all datasets to equal time spans. To achieve so, we will define a period, and all datasets will be divided by this period to finally have datasets with equal time duration. But to determine the best value of the period that will best perform on training the models, we will first define several period values and test the models on each of them to select the best period for each model. To this end, four periods are defined, P1: 5 min, P2: 3 min, P3: 2 min, P4: 1 min. Then we have selected 7 main classification algorithms that are: Boosting Decision Tree, KNN, Voting Classification, Gradient Boosting, Decision Tree, Neural Network, SVM, Naive Bayes, Random Forest Classifier, and Logistic Regression to be trained on the different selected periods, then compared the performance of each one.

A. Evaluation Metrics

In this section we will give a brief analysis of the models' performance to select the best one for the mentioned use case. Typically various metrics will be reported that assess how well the model is able to make predictions on new, unseen data. The choice of the metrics depends on the type of the problem, so for both models B-model and M-model the following metrics will be evaluated: Accuracy, Balanced accuracy and F1 score. Finally (AUC-ROC), an additional metric will be added for evaluating B-model, that stands for *area under the receiver operating characteristic curve*, it is a commonly used metric to evaluate the performance of binary classification models.

In general, F1 score, precision, and recall are metrics used in binary classification to evaluate the performance of a machine learning model. Where the *Precision* metric measures how often the model is correct when it predicts a positive sample. The formula for precision is:

$$Precision = \frac{TruePositives}{(TruePositives + FalsePositives)} \quad (6)$$

While the *Recall* metric measures how often the model correctly identifies a positive sample out of all the positive samples in the data. The formula for recall is:

$$Recall = \frac{TruePositives}{(TruePositives + FalseNegatives)} \quad (7)$$

But, the F1 score is the harmonic mean of precision and recall. It combines both metrics to provide a single score that represents the model's overall performance. The formula for F1 score is:

$$F1Score = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \quad (8)$$

Since Precision and Recall do not always provide a complete picture of a model's performance, and the F1 score balance the trade-off between precision and recall, as it provides a single score that summarizes both, then it will be used for the evaluation. The balanced accuracy in binary and multiclass classification problems is a metric to deal with imbalanced datasets. It is defined as the average of recall obtained on each class.

B. B-Model: Binary Classification

The aim of this model is to determine whether the device is in a static or mobile context as defined in Section IV. First we will train the seven selected models on the different defined periods. For each model the accuracy after cross validation is calculated. Here's a description of the distribution of the datasets for each specified period:

- (P1: 5 min): 921 input vectors
- (P2: 3 min): 1576 input vectors
- (P3: 2 min): 2375 input vectors
- (P4: 1 min): 4764 input vectors

The distribution of datasets is as follows: 33.9% train, 17% Home, 10% office, 9% conference, 7.5% bus, 7.3% pedestrian, 7.3% university, 3.5% restaurant, 1.6% metro, and 2% car. The percentage may slightly change from one period to another. So, each model is trained on the different defined periods. Figure 7 represents a clear comparison between different models' accuracy. We can see that a period of one minute gave a higher accuracy in almost all selected models, while five minutes has the lowest accuracy. Thus a period of one minute will be selected to divide the datasets into equal fragments for training B-model.

After determining the best period for training the models, now 27 classification models are selected for training. From the 27 models we have selected the most known models to compare, as table IV displays eleven models, each with its calculated evaluation metrics. We can see that LGBMClassifier and XGBClassifier gave an accuracy of 99%, and knowing that the trained dataset is not balanced, the *Balanced Accuracy* is calculated as it also gives a 99% accuracy, and same for F1 score. As a result, in our case, boosting Ensemble method (LGBMClassifier and XGBClassifier) improved model performance since we have unbalanced datasets, as they assign higher weights to misclassified examples in the minority class.

To justify the importance of using both information from WiFi and BLE jointly for such model, we repeated the training using only WiFi data, then only BLE, and compared the results with the models trained on both technologies jointly. Figure 8 displays the accuracy value (from cross validation) for each trained model for the three scenarios. The green curve represents the values from models that are trained with only BLE input data, and the red curve for WiFi input data only, thus we can see that WiFi information gives better accuracy than BLE input data in almost all trained models. The violet curve represents WiFi and BLE trained data jointly, the results shows a higher accuracy from such data, thus we can conclude that both WiFi and BLE jointly gives better accuracy for estimating the output.

C. M-model: Multi-class Classification

Now, the aim of this model is to determine a specified context of the network as mentioned in section IV. We have tested 26 different machine learning classifiers to train the multi-class model. Knowing that M-model has a different objective from B-model model, then we need to repeat the

same process for selecting the best period for classifying as in B-model. Table VI, shows the accuracy of 11 trained models on different period values. These models are the selected from the 26 trained models as the most known models and with the highest accuracy among the others. We can see that a period of two minutes and five minutes give a better accuracy in almost all tested algorithms. Thus (P3 = 2 min) will be selected as the period to train the models. We can see that the highest accuracy is equal to 93% from the LGBMclassifier and XGBclassifier and RandomForestClassifier as an example. To improve even more the accuracy, we added the results of B-model as an input feature to M-model to see how could the static/mobile information enhance the performance of the models. Table V shows the results of the same 11 models but with B-model input. We can see that the accuracy increased by 1% in almost all models, thus this indicates the importance of the information from B-model to give a better accuracy in determining the real-life situation of the device from M-model. After determining the best period for the classification which is two minutes, and improving the accuracy by the input values from B-model, the first three models that have higher accuracy that are: LGBMClassifier, XGBClassifier and RandomForestClassifier, are selected. We will test our chosen models again by getting the accuracy after cross validation and calculating the Time consumed for training and prediction.

TABLE III: Comparison between best three models

Model	*Accuracy*	*Training Time*	*Pediction Time*
XGBClassifier	93.62%	0.868	0.00436
LGBMClassifier	92.95%	0.907	0.00706
RandomForestClassifier	93.38%	0.433	0.01808

As shown in Table III, XGBClassifier has higher accuracy among the other selected models after cross validation, with the shortest prediction time, then it will be selected for hyper parameters tuning. After hyper parameters tuning, the accuracy remained the same, thus XGBoost classifier has the best accuracy with approximation to 94%.

TABLE IV: B-model models evaluation and comparison

Model	*Accuracy*	*Balanced*	*F1 Score*
LGBMClassifier	0.99	0.99	0.99
XGBClassifier	0.99	0.99	0.99
RandomForestClassifier	0.98	0.98	0.98
BaggingClassifier	0.98	0.98	0.98
SVC	0.97	0.97	0.97
AdaBoostClassifier	0.97	0.97	0.97
DecisionTreeClassifier	0.95	0.95	0.95
KNeighborsClassifier	0.94	0.94	0.94
LogisticRegression	0.93	0.93	0.93
RidgeClassifierCV	0.92	0.92	0.92
LinearDiscriminantAnalysis	0.92	0.92	0.92

VI. State of the Art

Analyzing human mobility is not new and is exploited since long to optimize infrastructure deployment especially for mobile telephony, mainly 5G [9] or edge computing. The literature has widely studied human mobility and investigated mobility models which is a step to measure their impact on network performances [10]. To this end, recognition of physical mobility states and activities has been studied because they provide useful information for investigating in human mobility [3]. Several studies in the literature used GPS positioning to infer physical activities. In [11] and [12], the authors uses GPS data with external knowledge about bus routes and bus stops to infer and predict a user's transportation mode such as walking, driving, or taking a bus by applying Bayes filters and Rao-Blackwellised particle. Although such algorithms perform well, they use computationally expensive models, thus it would be preferable to have simpler models that infer mobility states. Moreover, GPS data sampling is power-consuming. In [13], Krieg et al. proposed a transportation mode detection that helps in real-time parking, as through only 2 of sensor readings Accelerometer and Gyroscope, their system can decide whether a user is using one of the following transportation modes: walking, bicycle, bus, car, subway, motorbike, train, tram, airplane. Though this approach is applicable but it couldn't classify the real status of a device when it is static. Other approaches leveraged the information

TABLE V: M-model with B-model input feature

Model	*Accuracy*	*Balanced*	*F1 Score*
XGBClassifier	0.94	0.86	0.93
LGBMClassifier	0.93	0.86	0.93
RandomForestClassifier	0.94	0.85	0.93
BaggingClassifier	0.92	0.86	0.92
LogisticRegression	0.88	0.74	0.86
SVC	0.90	0.75	0.88
DecisionTreeClassifier	0.88	0.80	0.87
KNeighborsClassifier	0.85	0.69	0.84
RidgeClassifierCV	0.77	0.51	0.72
AdaBoostClassifier	0.51	0.20	0.36

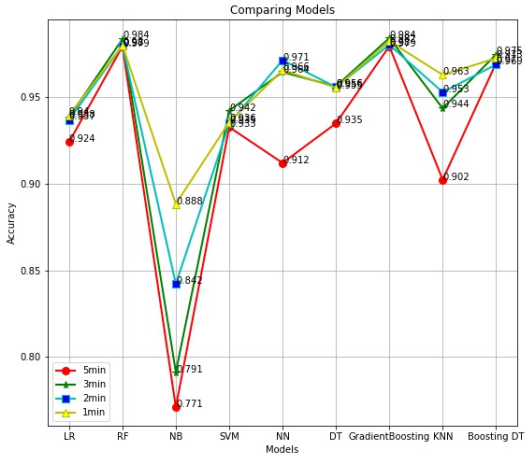


Fig. 7: Period comparison for B-model

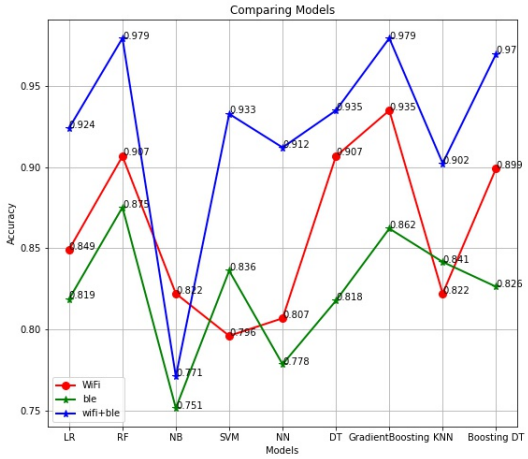


Fig. 8: Accuracy comparison for three different input datasets

from LoRa, BLE and WiFi wireless links but rather for indoor and outdoor localization systems like in [14], [15] and [16]. Thus our work is different in the sense that we adopt and exploit WiFi and BLE jointly to determine the status of the device in its real-life situation. To the best of our knowledge this is considered as a new approach for analyzing network context.

VII. Conclusion and Future Work

In this paper, we have proposed a novel approach for inferring human mobility by determining a device's status within its network context through wireless communication technologies, namely WiFi and BLE, working in conjunction. Firstly, we trained a model to ascertain whether a device is in a mobile or static network context. Subsequently, a complementary model was trained, providing a more precise classification of the device's real-life situation. These models were trained using real datasets collected over one year, for 90 hours, across diverse scenarios and conditions. In our initial approach, we achieved a 94% accuracy in classifying among 10 scenarios using a lightweight classical machine learning algorithm (XGBClassifier). As part of our future work, with a sufficiently large dataset, we plan to apply deep learning techniques to compare an online time series model with our offline classical ML model. Furthermore, we intend to incorporate additional scenarios to investigate how wireless links can be employed to determine a device's state in various contexts more comprehensively.

VIII. Acknowledgment

We would like to thank our colleagues Christiaan Geldenhuys and Hazem Chaabi for their valuable discussions, insightful feedback, and help during the early stages of this research work. Their insights and suggestions have significantly contributed to the development of our ideas.

TABLE VI: M-model binary classification period comparison

Model	*1 min*	*2 min*	*3 min*	*5 min*
LGBMClassifier	0.92	0.93	0.91	0.92
XGBClassifier	0.91	0.93	0.90	0.92
RandomForestClassifier	0.91	0.93	0.90	0.93
BaggingClassifier	0.89	0.91	0.88	0.90
SVC	0.89	0.90	0.88	0.86
AdaBoostClassifier	0.50	0.51	0.51	0.51
DecisionTreeClassifier	0.85	0.87	0.85	0.83
KNeighborsClassifier	0.84	0.83	0.83	0.80
LogisticRegression	0.85	0.87	0.87	0.84
RidgeClassifier	0.74	0.75	0.74	0.75
LinearDiscriminantAnalysis	0.19	0.12	0.17	0.25

References

[1] A. Trivedi, "Human mobility monitoring using wifi: Analysis, modeling, and applications," Ph.D. dissertation, University of Massachusetts Amherst, USA, 2021.

[2] G. M. Vazquez-Prokopec, D. Bisanzio, S. T. Stoddard, V. Paz-Soldan, A. C. Morrison, J. P. Elder, J. Ramirez-Paredes, E. S. Halsey, T. J. Kochel, T. W. Scott, and U. Kitron, "Using gps technology to quantify human mobility, dynamic contacts and infectious disease dynamics in a resource-poor urban environment," *PLOS ONE*, vol. 8.4, pp. 1–10, 04 2013.

[3] M. Y. Mun and Y. W. Seo, "Everyday mobility context classification using radio beacons," in *Proc. IEEE 10th Consumer Communications and Networking Conference (CCNC)*, 2013.

[4] D. Teixeira, A. C. Viana, J. M. Almeida, and M. S. Alvim, "Revealing challenges in human mobility predictability," *ACM Transactions on Spatial Algorithms and Systems*, 2021.

[5] D. Teixeira, M. Alvim, and J. Almeida, "On the predictability of a user's next check-in using data from different social networks," in *Proc. of the 2nd ACM SIGSPATIAL Workshop on Prediction of Human Mobility*, 2018.

[6] J. Heo, K.-M. Chung, S. Yoon, S. B. Yun, J. W. Ma, and S. Ju, "Spatial-data-driven student characterization in higher education," in *Proc. of the 1st ACM SIGSPATIAL Workshop on Prediction of Human Mobility (PredictGIS)*, 2017.

[7] E. Hernández-Orallo and A. Armero-Martínez, "How human mobility models can help to deal with covid-19," *Electronics*, vol. 10, no. 1, 2021.

[8] J. Koteich and N. Mitton, "PILOT Dataset: A Collection of Multi-Communication Technologies in Different Mobility Contexts," in *CoRes*, 2023.

[9] R. A. Paropkari, A. Thantharate, and C. C. Beard, "Deep-mobility: A deep learning approach for an efficient and reliable 5g handover," *CoRR*, vol. abs/2101.06558, 2021.

[10] T. Alam, "Fuzzy control based mobility framework for evaluating mobility models in manet of smart devices," *ARPN Journal of Engineering and Applied Sciences*, 2017.

[11] D. J. Patterson, L. Liao, D. Fox, and H. Kautz, "Inferring high-level behavior from low-level sensors," in *Proc. Ubiquitous Computing*, A. K. Dey, A. Schmidt, and J. F. McCarthy, Eds., 2003.

[12] L. Liao, D. J. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," *Artificial Intelligence*, vol. 171, no. 5, pp. 311–331, 2007.

[13] J.-G. Krieg, G. Jakllari, H. Toma, and A.-L. Beylot, "Unlocking the Smartphone's Sensors for Smart City Parking," *Pervasive and Mobile Computing*, vol. 43, pp. 78–95, 2018.

[14] K.-H. Lam, C.-C. Cheung, and W.-C. Lee, "Rssi-based lora localization systems for large-scale indoor and outdoor environments," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 11 778–11 791, 2019.

[15] P. S. Farahsari, A. Farahzadi, J. Rezazadeh, and A. Bagheri, "A survey on indoor positioning systems for iot-based applications," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7680–7699, 2022.

[16] P. Roy and C. Chowdhury, "A survey on ubiquitous wifi-based indoor localization system for smartphone users from implementation perspectives," *CCF Transactions on Pervasive Computing and Interaction*, vol. 4, no. 3, pp. 298–318, 2022.