Public Transport
https://doi.org/10.1007/s12469-023-00349-0

ORIGINAL RESEARCH

A low-cost automatic people-counting system at bus stops using Wi-Fi probe requests and deep learning

Cristina Pronello[1] · **Deepan Anbarasan**[1] · **Felipe Spoturno**[1] · **Giulia Terzolo**[1]

Accepted: 17 December 2023

Abstract

Counting people is an important part of people-centric applications, and the increase in the number of IoT devices has allowed the collection of huge amounts of data to facilitate people counting. The present study seeks to provide a novel, low-cost, automatic people-counting system for the use at bus stops, featuring a sniffing device that can capture Wi-Fi probe requests, and overcoming the problem of Media Access Control (MAC) randomization using deep learning. To make manual data collection considerably easier, a "People Counter" app was designed to collect ground truth data in order to train the model with higher accuracy. A user-friendly, operating system-independent dashboard was created to display the most relevant metrics. A two-step methodological approach was followed comprising device choice and data collection; data analysis and algorithm development. For the data analysis, three different approaches were tested, and among these a deep-learning approach using Convolutional Recurrent Neural Network (CRNN) with Long Short-term Memory (LSTM) architecture produced the best results. The optimal deep learning model predicted the number of people at the stop with a mean absolute error of ~ 1.2 persons, which can be considered a good preliminary result, considering that the experiment was done in a very complex open environment. People-counting systems at bus stops can support better bus scheduling, improve the boarding and alighting time of passengers, and aid the planning of integrated multi-modal transport system networks.

Keywords Deep learning · Automated people counting · Bus stops · Internet of Things · Wi-Fi probe requests · Neural networks

✉ Cristina Pronello
cristina.pronello@polito.it

Deepan Anbarasan
deepan.anbarasan@polito.it

[1] Interuniversity Department of Regional and Urban Studies and Planning, Politecnico di Torino, Turin, Italy

Published online: 24 February 2024

1 Introduction

Transport companies can benefit from reliable real-time people-counting systems to estimate travel demand, improve customer satisfaction, and optimize routes (Mehmood et al. 2019; Reichl et al. 2018). In recent years, thanks to the progress of Information and Communication Technologies (ICT), the number of Internet of Things (IoT) devices has soared. The increase in the number of Wi-Fi access points and client devices has enabled the collection of vast amounts of data, making people counting easier, since many of these devices are consumer devices (Cisco 2020).

Public Transport (PT), despite its sustainability, is still used predominantly by students and senior citizens (Kos-Łabędowicz 2019), but an improvement in service quality could make it more attractive to the working population; that is to say, if it had a higher frequency, increased comfort, and better accessibility to stops. The perception of the quality of public transportation in terms of location, waiting times, and equipment (such as benches) is influenced significantly by the characteristics of bus stops. To determine whether bus stops are optimally located, properly equipped, and have a frequency of buses that match demand, it is essential to monitor the flow of people at the stop. However, transport studies involving people counting have largely been limited to automatic onboard passenger counting (Fihn and Finndahl 2011; Mccarthy et al. 2021; Mehmood et al. 2019; Nitti et al. 2020), and little attention has been given to the importance of bus stops in improving transport infrastructure.

Traditionally, counting people at bus stops was done manually using hand-held counters, or via manual customer surveys once or twice a year. Such approaches are generally expensive and time-consuming, and only provide limited snapshots on the days that the counts/surveys are completed (Reichl et al. 2018). But with time and the advance of machine learning and artificial intelligence techniques, new methods have been developed for automatic data collection using treadle mats, cameras, Wi-Fi, Bluetooth, and other sensors (Charansonney 2018; Fihn and Finndahl 2011; Oberli et al. 2010; Rakebrandt 2015; Yang et al. 2010).

A number of Wi-Fi-based passenger counting systems have been studied in the last few years (Choi et al. 2022; Hidayat et al. 2018; Mccarthy et al. 2021; Nitti et al. 2020), and have been shown to have varying levels of accuracy. To the best of our knowledge, most validation of these Wi-Fi-based counting systems has been done either in controlled environments or only over short periods (Nitti et al. 2020; Oransirikul et al. 2014). Some studies have been done on people-counting systems based on Wi-Fi probe requests, but very few of these studies have focused on counting in the vicinity of local transport infrastructure in an open environment, like a bus stop. Also, while there have been attempts to use machine learning and deep learning (DL) methods for automatic people-counting, using source data based, for example, on link-blockage time in a Wi-Fi network (Ibrahim et al. 2019), on Channel State Information (CSI) from Wi-Fi signals (Choi et al. 2022; Liu et al. 2019), and on images from video cameras (Baumann et al. 2022), no studies, to the best of the authors' knowledge, have used Wi-Fi probe request packets as the source data for a CRNN-based DL method.

In this research we were seeking to estimate the number of people waiting at bus stops by detecting wireless devices in the vicinity and to process the collected information in order to determine whether DL could provide good results for this context. The structure of this paper is as follows: Sect. 2 provides an overview of various methods employed for automated people counting, specifically focusing on Wi-Fi sniffing and data processing approaches. The methodology used for device detection is presented in Sect. 3, followed by the results in Sect. 4. Finally, Sect. 5 concludes the paper by examining the significance of the proposed model, discussing the study's limitations, and suggesting future research directions.

2 Literature review

Traditionally, mat sensors have been used to count passengers by measuring their weight when boarding the bus (Basalamah 2016). However, these systems are expensive and require periodic maintenance, and they also have low accuracy (Baumann et al. 2022). Infrared sensors or cameras have been proposed as alternatives, but these have a number of limitations, such as blind spots, weak performance in poor visibility (Bernini et al. 2014; Liu et al. 2019; Saponara et al. 2016; Yahiaoui et al. 2010), and a high cost (Liu et al. 2019). Bluetooth signals have also been explored, but they are becoming less useful as a consequence of mobile devices automatically disabling Bluetooth functionality (Nishide and Takada 2013), their short transmission range, and their ability to operate in "hidden" mode (Schauer et al. 2014). Oransirikul et al. (2014) tested Bluetooth device detection as a way of counting passengers and were able to show that this approach is less reliable than Wi-Fi sniffing. Collaborative applications have been proposed as another option, but they require access to mobile device functions like GPS or microphones (Carrel et al. 2015; Gao et al. 2017).

Wi-Fi based detection has become popular owing to its lower energy usage, shorter discovery time, and a higher likelihood of devices having an active Wi-Fi interface (Kurkcu and Ozbay 2017; Lee et al. 2012; Schauer et al. 2014; Singh et al. 2021).

The ubiquity of smartphones, tablets, and wearables has led to the widespread use of Wi-Fi (Cisco 2020), which allows these devices to search for wireless access points in their immediate vicinity. This search is performed using probe requests that contain device-specific information, including the unique Media Access Control[1] (MAC) address. Even when Wi-Fi is switched off, devices continue to send out these requests to calculate device location based on the location of known Wi-Fi access points nearby (KODY 2018). By analysing these probe requests, it is possible to count the number of devices in the vicinity and estimate the number of people (Mikkelsen et al. 2016; Myrvoll et al. 2017; Reichl et al. 2018). Transport for London has reported[2] that analysing data from free Wi-Fi connections can offer valuable insights into customer movement at underground stations.

[1] https://standards.ieee.org/products-programs/regauth/.

[2] https://tfl.gov.uk/corporate/privacy-and-cookies/wi-fi-data-collection.

Oransirikul et al. (2014) attempted to use Wi-Fi signals to count the number of devices at a bus stop. However, the quantity of data collected was limited, as the experiment took place only over a short period and did not account for MAC address randomization. Because of privacy concerns, smartphone manufacturers have implemented a mechanism for randomizing the MAC address between probe requests, making it difficult to uniquely identify a device (Fenske et al. 2021; Matte 2017; Purvis and Dementyev 2020; Vanhoef et al. 2016). As a result, automatic people-counting systems based on unique MAC addresses in Wi-Fi probe requests are becoming obsolete.

Some authors have, nevertheless, proposed ways of overcoming the problem of address randomization. These include the iABACUS system proposed by Nitti et al. (2020), which uses a de-randomization algorithm. Other solutions involve machine learning models, such as those proposed by Guillen-Perez and Cano (2019) and Uras et al. (2020), but these have not been tested in scenarios with a large number of people.

Baumann et al. (2022) judged that the progress in DL over the past few years has unlocked fresh opportunities for businesses, thanks to the rise in storage capacity, wider availability of data, and increased computing speed. Specifically, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have shown impressive advances in performing various tasks (Liu et al. 2019). CNN algorithms apply multiple filters over the input data to detect patterns, more information about how CNN works can be found in Lecun et al. (1998). RNN models make use of a loop-structure to connect different layers to process sequential data. For more information on RNNs see Rumelhart et al. (1986). RNN models are particularly suitable for applications that involve sequence data, such as speech recognition (Graves et al. 2013) and handwriting alignment (Raue et al. 2015). However, traditional RNNs suffer from the vanishing gradient problem when dealing with long sequences. To overcome this, Long Short-Term Memory (LSTM) architecture is commonly used in conjunction with RNN (DiPietro and Hager 2020; Hochreiter and Schmidhuber 1997). A few studies have combined the benefits of both CNN and RNN, extracting information from features and then connecting the extracted features to produce estimations (Liang et al. 2018; Wang et al. 2019).

Various DL approaches have been tried in recent years for counting people. Ibrahim et al. (2019) proposed an RNN-based approach called "CrossCount" that maps a sequence of temporal link-blockage patterns to a human count, using a single Wi-Fi link and achieving an accuracy of about 55%. The accuracy of their model decreased as the number of people increased, which could be due to limitations in data collection and to the method of superimposition that they used to simulate multi-person data from single-person data.

Liu et al. (2019) proposed a crowd counting method called "DeepCount" that counts the number of people using the Channel State Information (CSI) from Wi-Fi signals, applying CNN. Its accuracy was 86.4% in a confined environment, but the authors did not report the performance and accuracy of their model when the number of people is greater than 5. Also, their model was designed and tested in an indoor environment, and it is doubtful that the data they collected would be relevant to all scenarios in an outdoor environment.

Similarly, Choi et al. (2022) constructed machine learning and Deep Neural Network (DNN) models for crowd counting using Wi-Fi Channel State Information (CSI) in a closed environment. Although the DNN models showed a slightly better performance, the authors ultimately preferred machine learning methods because of the high cost of model training required to achieve the system's best performance with DNN. Singh et al. (2020) proposed the use of DL for time-series forecasting of crowds over a short time horizon, based on time-stamped crowd data, but DL was not used in their study to address MAC address randomization. Liu et al. (2022) proposed a spatial analysis model to understand passenger flow on a Bus rapid transit (BRT) system. The authors used an approach of counting the number of unique devices based on the probe request packets, but the paper did not account for the phenomenon of MAC address randomization.

Although DL models have been used by a number of researchers for people counting, these models have been limited to smaller training datasets or synthetic data for the training phase. The information sources used in these studies were Wi-Fi CSI and link-blockage time, rather than Wi-Fi probe requests. Moreover, the methods were assessed in a controlled environment and their performance was reported for small crowds.

There have been a few other studies relating to people counting that address topics that we explore, although the focus of these studies has been different. The research by Myrvoll et al. (2017), while having some relevance to our work, did not deal with MAC address randomization. The study by Nitti et al. (2020) centers on de-randomization without integrating deep learning, and its contextual scope is limited to a simulated bus environment, which differs in significant respects from the real-world context of a bus stop. In the studies by Uras et al. (2020) and Vanhoef et al. (2016) emphasis is placed on de-randomization and uniquely identifying devices via fingerprinting, complemented by clustering algorithms like DBSCAN and OPTICS (we should mention that Vanhoef et al. (2016) relates neither to people counting nor to public transport). Although both of the studies just mentioned, like ours, use probe request data to overcome MAC address randomization, our study also seeks to utilize the patterns and inherent variability of different features in estimating the number of passengers, and it does not attempt to identify and individuate unique devices. This has the additional benefit of protecting user privacy. It is a change in approach that led us to opt for Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) rather than clustering algorithms.

Tu et al. (2021) also take a different path, encompassing network events beyond probe request packets and involving different data collection methods. In fact, the authors place a greater reliance on network events linked to user interactions via a router than on probe request data. As part of this observable shift in emphasis away from probe requests, the present study has an important contribution to make, addressing gaps mentioned above through the application of deep learning techniques to real-world Wi-Fi probe request data within the dynamic context of a bus stop.

In the context of bus stops, where it is essential to take the capacities of stops, together with temporal patterns, into account, as underscored by Voß et al. (2020), there have been very few studies that have used either Wi-Fi probe request packets

or DL methods to count passengers. Designing bus stops that balance functionality and safety hinges on a nuanced grasp of their utilization dynamics. Furthermore, studying bus stop crowding can provide insights into the dynamics of bus stop usage, have useful spin-offs relating to bus stop design, and potentially even help combat congestion by reducing dwell times of buses at stops (Tirachini 2014). A recent study by Jee et al. (2023) has also demonstrated that Wi-Fi sensors can be used at bus stops for estimating passenger queue lengths and waiting times, offering a cost-effective method to assess transit demand.

The study addresses gaps in people counting research by applying deep learning, more specifically CRNN, to real-world Wi-Fi probe request data within a bus stop environment, overcoming limitations of previous studies in relation to data size, sources, and methodologies. The research also seeks to use these methods to estimate passenger numbers, while protecting user privacy. It thus offers a singular contribution in the field of transportation-oriented passenger counting.

In summary, the paper makes valuable contributions as outlined below:

- The paper contributes to the existing literature by addressing limitations in people counting research through the use of deep learning on real-world Wi-Fi probe request data, avoiding the use of limited or synthetic data sources.
- In contrast to previous research centered on device de-randomization and identification, the study adopts a distinct approach: utilizing inherent patterns within diverse features to estimate passenger numbers while preserving user privacy.
- Within the context of bus stop dynamics and utilization, where Wi-Fi probe requests and DL methods have not been extensively harnessed, the paper offers substantial insights through the application of deep learning in a real-world setting, effectively bringing theory and practice together.

3 Methodology

The design of our proposed system using wireless devices for estimating the number of people waiting at bus stops is illustrated in Fig. 1. A device is installed at a bus stop to detect waiting passengers, and the data is saved to a cloud database. An algorithm is then applied to this data to estimate the number of people waiting for a bus, and the output is stored in the same database.

The methodological approach comprises two steps:

- Choice of device and data collection;
- Data analysis and development of algorithms.

The proposed system also includes a dashboard for displaying the people-count information in an easily understandable format. The dashboard was developed using Flutter,[3] an open-source user interface software development kit. The code is written

[3] https://flutter.dev/.

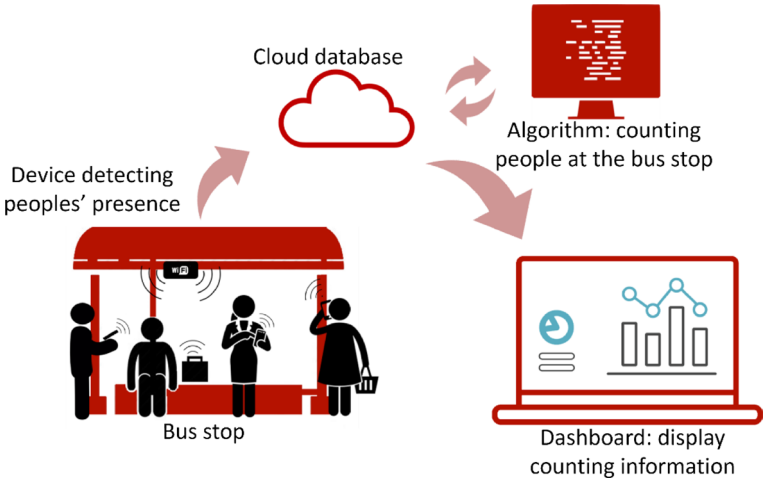

Fig. 1 Overview of the system

in the Dart programming language. This framework was chosen since it is highly customizable, powerful, and cross-platform, meaning that it can be installed on a large selection of devices and operating systems. It can be stored on a server and is accessible from the web via standard browsers.

3.1 Device choice and data collection

The hardware board selected for Wi-Fi probe request sniffing at bus stops must have dual Wi-Fi capabilities, be low-cost and energy-efficient, have small dimensions for ease of installation, and possess reasonable memory capacity to collect and send data. These requirements ensure that the device will be able to sniff and upload data, be cost-effective for large-scale installation, occupy minimal space, and have sufficient memory to store data temporarily. We identified the best options available on the market, and these are presented in Table 1.

The cloud system for Wi-Fi probe request sniffing at bus stops must have the capability to store sniffed packets in a database, process Hypertext Transfer Protocol[4] (HTTP) requests for data transfer, maintain a database for the entire system, and allow for high scalability. The device connects to the internet using Wi-Fi provided over a mobile hotspot that uses LTE. The system design means that it can easily be extended to new stops, and transport operators can integrate data smoothly into their systems or create a custom dashboard. The system uses a REpresentational State Transfer[5] (REST) Application Programming Interface[6] (API), and a multi-platform dashboard has been developed for easy access to collected and processed data.

[4] https://www.w3schools.com/tags/ref_httpmethods.asp.
[5] https://restfulapi.net/.
[6] https://dl.acm.org/doi/10.1145/800297.811532

Table 1 Suitable marketed hardware solutions

Hardware name	Capabilities	Cost in Euros
Cypress pioneer KIT PSoC 6 (Infineon Technologies AG 2022)	Dual Wi-Fi capability, 4 MB FRAM, and 288 KB SRAM	€ 96.53[a]
HANI IoT Board (RELOC 2022)	No dual Wi-Fi capability, 320 KB of SRAM	€ 79.83[b]
Arduino UNO 33 IoT + Arduino nano 33 IoT (Arduino 2022a, b)	No dual Wi-Fi capability, 520 KB of SRAM Memory	€ 44.80
PyTrack 2.0 X + WiPy 3.0 (Pycom 2022a, b)	Dual Wi-Fi capability, 4 MB of RAM	€ 62.60

FRAM Ferroelectric RAM, *SRAM* Static RAM

[a]Infineon-Cypress/CY8CKIT-062-WIFI-BT (2022)

[b]Arrow Development Tools (2023)

Scalability is critical for the quick and cost-effective installation of sniffing devices at a large number of bus stops in a city, and to this end a plug-and-play approach is used based on a configuration file.

Our collection of data at bus stops implied a number of choices:

(a) Choice of bus stops. These were selected using two criteria:
- The bus line needed to include bus stops at different types of locations, that is to say crowded central urban areas in addition to less populated suburban areas (line 55 operated by Gruppo Trasporti Torinese[7] – GTT – is an example of a line meeting this requirement);
- Stops had to be sufficiently distant from traffic intersections to avoid an excessive level of "noise" in the form of Wi-Fi probe request packets from devices in nearby stationary vehicles;

(b) Choice of the period for data collection (for automated and manual counting). Data collection took place for 9 days from March 1st to April 10th 2022, including weekends and public holidays, in three time slots: morning peak hours (7:30–10:30), midday peak hours (12:00 to 15:00), and evening peak hours (17:00–19:00);

(c) Choice of the characteristics of the Wi-Fi probe request packets to be used for automated counting. Some of the features used as input for the sequential (Sect. 3.2.2) and neural network approach (Sect. 3.2.3) for the Wi-Fi probe request data collection are reported in Table 2;

(d) Choice of the manual counting method. The manual counting was carried out using the "People Counter" mobile app, created for this purpose, that allows the user to increment, decrement, and reset the number of people recorded as waiting at the bus stop. The number of people, as soon as it changes, is immediately sent to Firebase Firestore[8] for storage. The back-end system integrates the "ground truth" with the data taken from the sniffing device. Some screenshots from the app are shown in Fig. 2. People waiting were counted manually in this way when they came within 11 m of the device (although at some stops people were waiting for a bus beyond this 11-m radius).

3.2 Data analysis and algorithm development

To analyze the data coming from the IoT sensors and estimate the number of people waiting at the bus stop, different approaches were tested. In what follows, an "event" will be considered as a single packet sniffed by the IoT device and posted to the cloud database. Sniffed packets coming from Wi-Fi devices are asynchronous events occurring on a continuous-time basis. To process the information collected by the IoT device through synchronous discrete-time algorithms, there must be a

[7] https://www.gtt.to.it/cms/.

[8] https://firebase.google.com/docs/firestore.

Table 2 Features of Wi-Fi request packets

Feature	Data type	Characteristics
Timestamp	Integer	A digital record of the time of occurrence of events
MAC address	A hexadecimal vector as a string	Media Access Control (MAC) address is a unique identifier assigned to a Network Interface Controller (NIC) for use as a network address in communications within a network segment
Frame aggregation	Integer	A feature that can mitigate the performance hit, by sending multiple data packets in a single 802.11 transmission
AMPDU Length exponent	Integer	Aggregated MAC Protocol Data Unit (AMPDU) improves data transmission by aggregating or grouping several MPDU blocks. This field can take the values between 0–7 and is used to communicate the size of the AMPDU that may be transmitted (Gast 2013)
Channel	Integer	Wi-Fi channels are the medium through which a wireless internet sends and receives data
FEC coding	Integer	Forward Error Correction (FEC) is a method to control errors in data transmission
MCS	Integer	The Modulation Coding Scheme (MCS) index is a metric based on several parameters of a Wi-Fi connection between two stations. Namely, for 802.11ac, it depends on the modulation type, the coding rate, the number of spatial streams, the channel width, and the guard interval
Noise floor	Signed Integer	The noise floor is the measure of the signal created from the sum of all the noise sources and unwanted signals within a measurement system, where noise is defined as any signal other than the one being monitored
RSSI	Signed Integer	The Received Signal Strength Indicator (RSSI) is a measurement of the power present in a received radio signal
SGI	Integer	Short Guard Interval (SGI) is intended to avoid signal loss from the multipath effect
STBC	Integer	Space–Time Block Coding (STBC) is an optional feature of the 802.11ac standard, which allows a transmitter to transfer multiple copies of data streams using an array of antennas
SSID	Alphanumeric string	The Service Set Identifier (SSID) is the name of the wireless network, also known as the Network Identifier or ID
Sequence number	Integer	The sequence number is a counter used to keep track of every byte sent outward by a host
Supported rates	Integer	Amount of information that can be exchanged per second expressed in Megabits per second (Mbps)

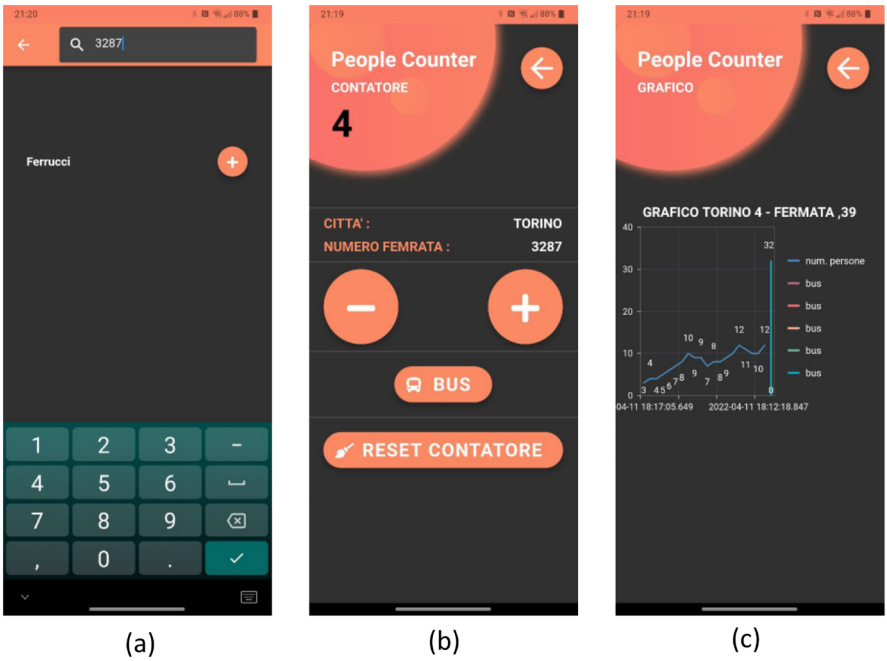

Fig. 2 People Counter app screenshots: **a** Search screen, **b** Count screen, **c** Plot screen

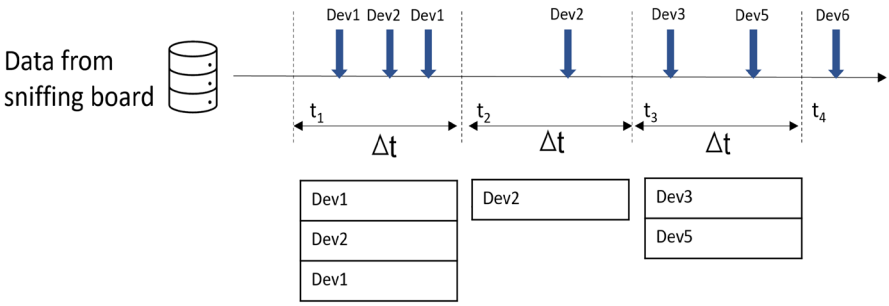

Fig. 3 Data preparation for the processing algorithms

discretization of time in the processing pipeline. The discretization that was implemented groups together events occurring inside different time windows t_i of length Δt, and the algorithm takes a fixed number of these time windows $N_{\Delta t}$ as input. As an illustration, consider the depiction in Fig. 3. Here we have four discrete time windows—t_1, t_2, t_3 and t_4—each of length Δt. The first three events, from two distinct devices, are assigned to time window t_1. Subsequently, the fourth event is assigned to time window t_2, and so on, depending on the time of arrival of the new event. So, for each time window, the input data will be a matrix with the number of columns corresponding to the number of features from Table 2 and the number of rows

corresponding to the number of probe request packets received in that time window. When events are used to estimate the number of people at the bus stop, multiple time windows are grouped together to form a "visibility period" and the data is provided in the form of multiple matrices, where the number of matrices is equal to the number of time windows $N_{\Delta t.}$ The average stop occupancy is estimated for the visibility period and compared with the ground truth. In order to facilitate alignment with the ground truth data, the ground truth information is segmented into time windows of equivalent duration.

At a given bus stop, each time window t_i of length Δt has a corresponding value P_i, that is the number of people at the stop at the end of the time window. The ground truth values are calculated as shown in Eq. (1) for a given *visibility period*, which can be described as the time period during which the actual average stop occupancy is compared with the people count calculated as described below.

$$\hat{R} = \frac{1}{N_{\Delta t}} \sum_{t_i=1}^{N_{\Delta t}} P_i \tag{1}$$

where:

- t_i: Time window of length Δt;
- P_i: number of people at the bus stop in each time window t_i;
- $N_{\Delta t}$: Number of time windows of length Δt considered for obtaining ground truth for a visibility period, which can be denoted in seconds as $\Delta t * \mathrm{N}_{\Delta t}$;
- $\hat{R}$: Ground truth people count for the visibility period ($\Delta t * \mathrm{N}_{\Delta t}$).

The number of people may be estimated from the detected packets using either of two approaches: a *packet-content independent approach*, or a *packet-content dependent approach.* As regards the three different methods proposed in this study, the first method, which uses a packet-content independent approach, is presented in Sect. 3.2.1, while the other two methods, using a packet-content dependent approach, are presented in Sects. 3.2.2 and 3.2.3.

3.2.1 Matched digital filter

In a packet-content *independent approach*, the content of the detected packets is ignored. Packets are treated simply as "events" to be counted, and then filtered as in a classical signal-processing task. The input of these algorithms is the number of events (sniffed packets) at each time.

The hypothesis is that the number of events observed in a given time window (t_i) is determined by the sum of two factors: the number of people (P_i) at the bus stop and a random variable (η) that encompasses various external factors such as the interference of nearby devices, fixed Wi-Fi antennas, and other perturbations. This relationship is expressed by Eq. (2):

$$\mu = P_i + \eta \tag{2}$$

where:

- μ corresponds to the number of events (probe requests) captured in time window t_i;
- P_i corresponds to the number of people at the bus stop at the end of the time window t_i;
- η corresponds to a random variable representing noise.

P_i and η are not zero-centered signals. The first pre-processing stage serves to center the signal μ to the value $\mathrm{E}(P_i)$ that can be determined from the training data, producing a pre-processed output y_e. The idea is to take the centered signal of the random variable as a "noise" signal to be filtered through a filter h, to produce the estimation of the number of persons as in Eq. (3).

$$\widehat{P} = y_e * h \tag{3}$$

where:

- h – filter to be applied to pre-processed input signal;
- y_e – input signal for the matched filter (output of pre-processing);
- $\widehat{P}$ – estimated people count at a stop during time window t_i.

The choice of testing a matched filter is in fact to find a baseline performance that can be obtained with a simple *packet-content independent approach*, and then to check whether a *packet-content dependent approach* can outperform this baseline.

3.2.2 Sequential processing algorithm

The packet-content dependent approach analyses the sequence of logged packets to estimate the number of people at a bus stop. The input to the algorithm is the sequence of packets sniffed at each time. Before analyzing the relevant features of the probe request packets, the input data are filtered based on the Received Signal Strength Indicator (RSSI) contained in the sniffed packets. Only packets with an RSSI greater than – 78 dB are considered in order to eliminate possible extraneous sources of noise.

The sequential processing algorithm is used to analyze the most relevant features of the probe request packets across $N_{\Delta t}$ time windows, each of length Δt. To estimate the number of people at the bus stop, the algorithm counts the number of unique MAC addresses in each time window. To address this issue of MAC address randomization (APPLE Inc. 2021; Fenske et al. 2021), a MAC address de-randomization approach is used. This approach considers the supported rates and RSSI (see Table 2) of each device within a time window. If the supported rates are unchanged

and the RSSI is in a range of ±4 from a previously detected MAC address within the same time window, then the device is considered to be the same as the previously detected device, as shown in Eq. (4).

$$MAC_j = \begin{cases} MAC_i^a, & if\ SR_i^a = SR_j\ and\ RSSI_i^a - 4 \le RSSI_j \le RSSI_i^a + 4\ and\ 1 \le a \le \mu \\ MAC_j, & otherwise \end{cases} \tag{4}$$

where:

- MAC_i^a: the MAC address of any detected device a in time window t_i;
- t_i, t_j: successive time windows where j = i + 1;
- μ: number of probe request events captured in time window t_i;
- $RSSI_i^a$ andSR_i^a—RSSI value and supported rates of the corresponding detected device (see Table 2) in time window t_i;
- MAC_j: the MAC address of a device being currently checked in time window t_j;
- SR_j is $RSSI_j$: RSSI value and supported data rates at the time of detection for the device with MAC address MAC_j.

The people count is then computed as the number of unique devices in each time window t_i, as shown in Eq. (5).

$$\hat{P} = \sum_{a=1}^{\mu} \begin{cases} 1, & if\ MAC^a\ has\ not\ been\ so\ far\ in\ time\ window \\ 0, & if\ MAC^a\ has\ been\ seen\ before\ in\ time\ window \end{cases} \tag{5}$$

where:

- $\hat{P}$: Estimated count of people in time window t_i;
- μ: number of events captured in time window t_i;
- MAC^a: MAC address of each detected event after applying Eq. (4).

3.2.3 Convolutional and Recurrent Neural Networks (CRNN)

The input data are in the form of probe request packets with features (or columns) as listed in Table 2. These data are further partitioned into distinct time windows, a process illustrated in Fig. 3. The size of the input data is not fixed, but variable, as a result of fluctuations in the number of probe requests received in different time windows. These fluctuations are contingent upon several elements, including the number of nearby devices, and factors dependent on device manufacturers and models. The current operational state of the device—be it idle mode, screen-on status, or Wi-Fi connectivity—likewise contributes to these fluctuations (Pintor and Atzori 2021).

In order to fix the size of the input data, transmit it as sequential data to the adjacent layers, and then process the sequential data, a combination of CNN and RNN is used. CNN is good for feature extraction due to its ability to identify spatial patterns, while RNN is adept at handling sequential data such as time series, making this combination well-suited for capturing both spatial and temporal aspects inherent

Table 3 Convolutional layer operation algorithm

Algorithm 1: Convolutional layer operation

1. For each time window Δt:
   (a) For each sniffed packet:
      - Apply each one of the F_c filters in the convolutional network
   (b) Sum the outputs of each filter for all the sensors in the time window producing a $1 \times F_c$ dimensional vector
2. Concatenate the output of the $N_{\Delta t}$ time windows producing a fixed-size $N_{\Delta t} \times F_c$ matrix

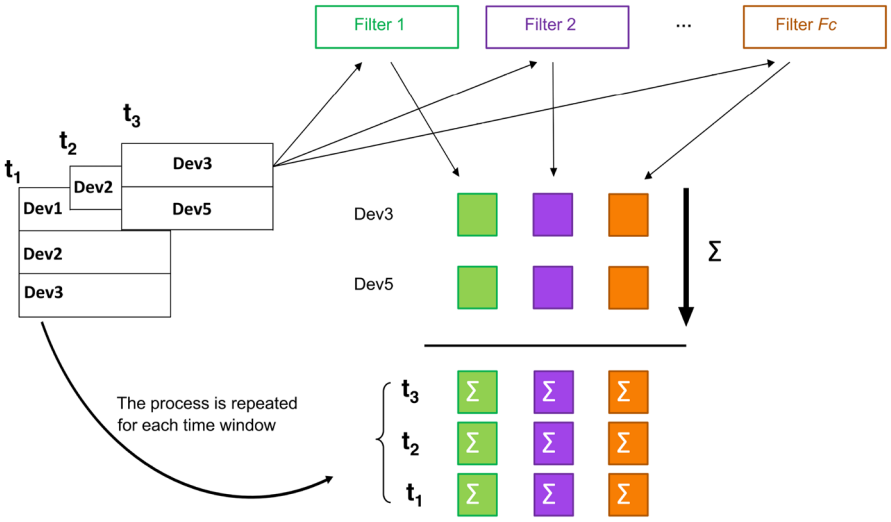

Fig. 4 Convolutional input layer

in predicting people counts over time (Hasan et al. 2019). LSTM cells are used for the RNN layer because of their capabilities in terms of long-term memory and their better handling of the vanishing gradient problem (DiPietro and Hager 2020; Yu et al. 2019). A CRNN algorithm (Liang et al. 2018; Wiest 2017) is used with a first convolutional layer that fixes the input data size for the following layers by applying a set of filters to each collected packet, as described in the algorithm reported in Table 3. The number of filters is denoted by F_c and is one of the parameters that is optimized during the model training phase. This first convolutional operation is shown in Fig. 4.

The convolutional filters at the CNN layer typically aid in capturing patterns within the features found in each probe request packet. Any of these filters may be of use in capturing conditions or patterns relating to RSSI values, SSID patterns, patterns with MAC address values and so on. These conditions help in identifying how many packets come from unique client devices belonging to passengers at the bus stop. Then by summing the outputs, the final vector for each window is prepared and

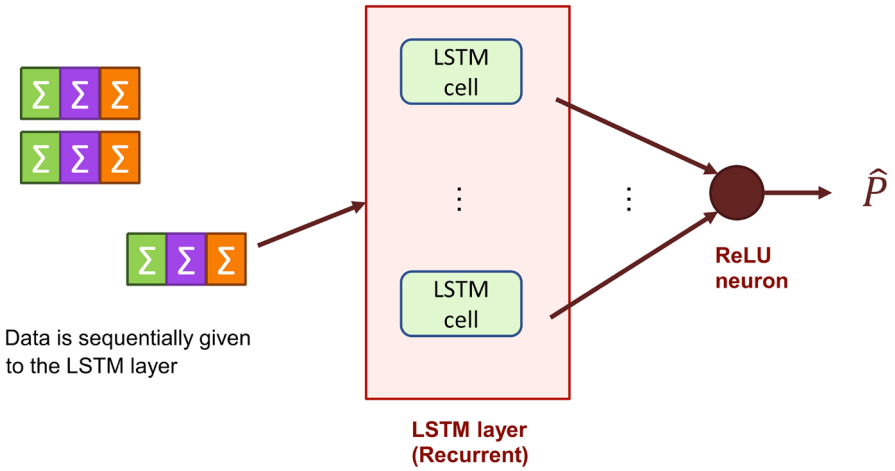

Fig. 5 Long Short-Term Memory (LSTM) and Rectified Linear Unit (ReLU) layer

concatenated to produce the final output matrix of fixed size $N_{\Delta t} \times F_c$, with the rows containing data from the different time windows, and the columns representing the filters that were applied to the data from the different time windows.

Each vector produced by the convolutional layer for each time window is then sequentially provided as input to a recurrent layer (composed of LSTM cells) that will learn the dynamics of how the patterns corresponding to the convolutional filters change over time, to produce a final estimation of the average stop occupancy via a final Rectified Linear Unit[9] (ReLU) neuron, as shown in Fig. 5. The final output contains the model's estimation of the number of people at the bus stop during the *visibility period*.

The only parameters for this architecture are the number of filters F_c and the number of LSTM cells L_n, that are optimized during the "Training campaign" using stochastic gradient descent optimization (Hardt et al. 2016).

4 Results

Among the suitable devices identified (Table 1), the one selected was the WiPy 3.0 module in combination with the PyTrack 2.0 X electronic board equipped with a microcontroller. WiPy 3.0 was used to capture Wi-Fi probe request broadcast packets. Probe requests are control frames sent by devices equipped with a Wi-Fi interface. The device was programmed to emit light signals as follows:

- Blue, when in operation;
- Green, when sending an information packet to MongoDB;

[9] https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html.

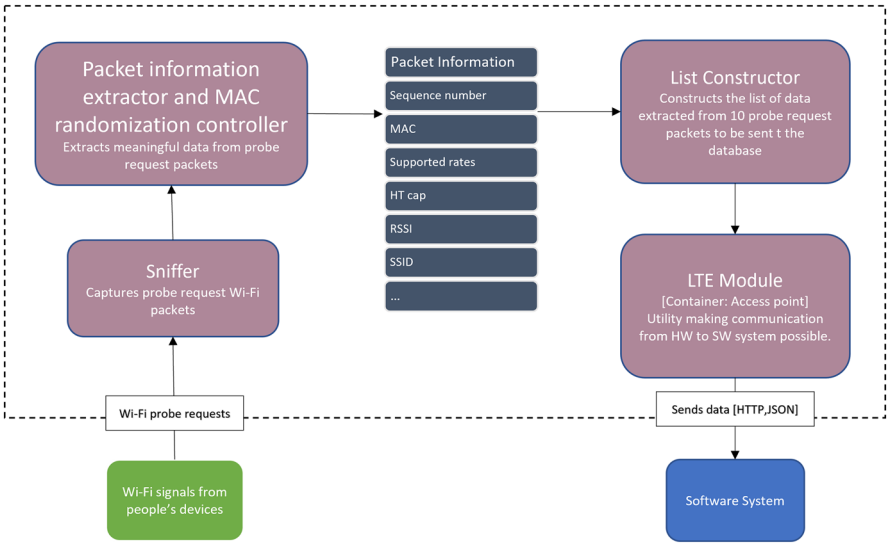

Fig. 6 Operations performed on device

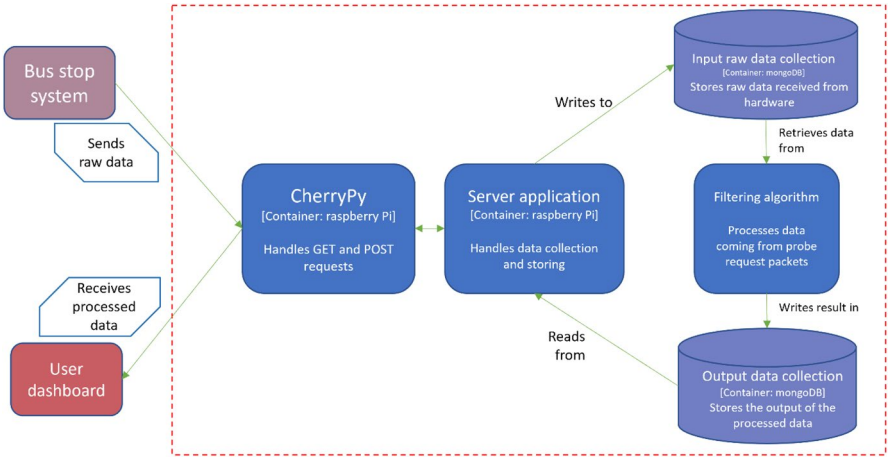

Fig. 7 Cloud system overview

- Red, when sending a packet to MongoDB fails. In this case, the data collected up to that point are saved in memory and the device is forced to restart. Once restarted, before starting to capture other probe request packets, the device sends everything it has previously saved in the memory.

The operations performed on the device are shown in Fig. 6. For every ten packets received, a JavaScript Object Notation (JSON) file is produced, which is sent to the MongoDB database. This is done to reduce the number of database connection requests the device makes.

Fig. 8 Ferrucci bus stop. Source: Screen capture from google maps

Table 4 Schedule of the manual data collection

Start time	End time	Bus stop number	Day of the week
2022-03-05 13:00	2022-03-05 13:00	(Artificially generated data)	Saturday
2022-03-12 10:25	2022-03-12 11:24	624	Saturday
2022-03-23 07:53	2022-03-23 10:31	3287	Wednesday
2022-03-27 07:46	2022-03-27 10:34	3287	Sunday
2022-03-28 06:52	2022-03-28 09:37	3287	Monday
2022-03-29 07:39	2022-03-29 09:43	3287	Tuesday
2022-04-11 15:17	2022-04-11 16:15	39	Monday
2022-04-20 10:31	2022-04-20 12:15	3287	Wednesday
2022-04-21 15:24	2022-04-21 17:01	3287	Thursday

The device used to store and compute all the cloud functions is a Raspberry Pi 3 Model B. It was chosen since resources available for this research are limited, and the use of faster, more expensive hardware would have resulted in over-dimensioning. Moreover, all the relevant software was designed to compile and work on any other Linux-based machine, which would, if necessary, allow migration to a more powerful machine with little effort. CherryPy is used to handle Representational state transfer (REST) requests for all the components of the architecture. This makes the system highly scalable since all transfers of data are handled using standard JSON files and GET/POST[10] requests (GET and POST are the two most common HTTP transfer requests).

To store the data, MongoDB is used, which is a NoSQL[11] document database. It is open-source and used mainly for high-volume data storage (Győrödi et al. 2015). It also provides scalability and flexibility. An overview of the cloud system is shown in Fig. 7.

[10] https://restfulapi.net/http-methods/.
[11] https://www.mongodb.com/nosql-explained.

Table 5 Distribution of collected ground truth values

Date	Stop number	Mean stop occupancy	Min passenger count	Max passenger count
05 March 2022	Test location	4	1	7
12 March 2022	624	4.83	0	12
23 March 2022	3287	2.97	0	9
27 March 2022	3287	3.48	0	11
28 March 2022	3287	2.55	0	7
29 March 2022	3287	3.08	0	9
11 April 2022	39	11.95	2	32
20 April 2022	3287	2.20	0	7
21 April 2022	3287	2.38	0	6

Table 6 Pre-processed features

Feature name	Processing
MAC Address	Split the string into a vector of six hexadecimal numbers. Conversion from hexadecimal to integer for each element
SSID	Conversion from string to integer through a hashing function

Table 7 Performance evaluation of the tested algorithms during the preliminary analysis

Algorithm	Mean Absolute Error [Number of persons]
Matched filter	1.78
Sequential algorithm	1.82
CRNN	1.25

For data storage in MongoDB, three types of collection were set up: (1) *input collection*, storing all the raw data from the sniffing boards with the addition of a timestamp; (2) *output collection*, storing the results evaluated by the counting algorithm together with a timestamp; and (3) *device collection*, including information about all the active sniffing devices.

Data were collected, on the one hand via the device installed on the bus stop bench, and on the other hand manually, at the Ferrucci bus stop 3287 on Corso Vittorio Emanuele II in Turin (Fig. 8), according to the schedule given in Table 4. On March 5th, artificially generated data were collected, in order to provide at least one dataset in a controlled environment. The data were collected in a room where all Wi-Fi capable devices apart from the test devices were removed. The data for March 12th, 2022 and April 22nd, 2022 were collected, respectively, at the bus stops 624 and 39 to check that the model works in different contexts, so as to avoid model overfitting (Gupta and Sharma 2022) to the selected stop.

Table 8 Best parameter values for the neural network training

Δt [s]	$N_{\Delta t}$	Number of convolutional filters (F_c)	Number of LSTM Neurons	Mean absolute error (MAE)
30	*10*	*60*	*5*	***1.14***
60	10	60	5	1.26
60	5	60	5	1.35
60	5	10	1	1.68
30	10	30	5	1.54
30	5	60	5	1.52

Italic highlights the parameters for the best model and the BoldItalic highlights the lowest error achieved by a model

More than 22,700 records of sniffed packets, and more than 1700 ground truth labels were obtained from the collected data. The collected ground truth data with the mean, minimum and maximum stop occupancy at the different bus stops is listed in Table 5. Stop number 39 is seen to exhibit a crowding pattern that is notably different from the other stops in the dataset. This variance in crowding patterns provided diversity in the training and testing of the model.

4.1 Data analysis and modelling

The conversion of the input data into a processable format allows feature selection and preparation for the analysis. Data processing algorithms need number-type data as input, and therefore sniffed packets that may contain information in string formats, such as the SSID name and the MAC address, must be converted in order to be processed. In addition, the literature shows that for some devices, MAC addresses are not randomized across all their fields, but only in a subset of them (Purvis and Dementyev 2020), and so in order to let the neural network learn this process, the MAC address was split into a vector of six numbers, as described in Table 6.

Using the collected data, the algorithms described in the methodology were implemented and tested with different parameters iteratively, to select the parameters that produced the best results. The results obtained in terms of mean absolute error are shown in Table 7. Based on the literature review, some studies have suggested the use of the Mean Absolute Error (MAE) the Root Mean Square Error (RMSE) (Brassington 2017; Willmott and Matsuura 2005) especially for datasets with errors of varying magnitude and potential outliers. Therefore, to maintain consistency among all tested methods, the MAE was chosen to be used. It can be seen that the best-performing algorithm is the CRNN, which has the lowest MAE among the compared algorithms. The CRNN algorithm was therefore selected for further fine-tuning with a larger training set, varying its parameters to find the best result.

After selecting CRNN as the best algorithm for our use case, we used different numbers of neurons for each layer, and different values for F_c, Δt and $N_{\Delta t}$ to train different models in order to determine which performed best. Two parameters of the model play an important role in addressing the issue of devices that come close to

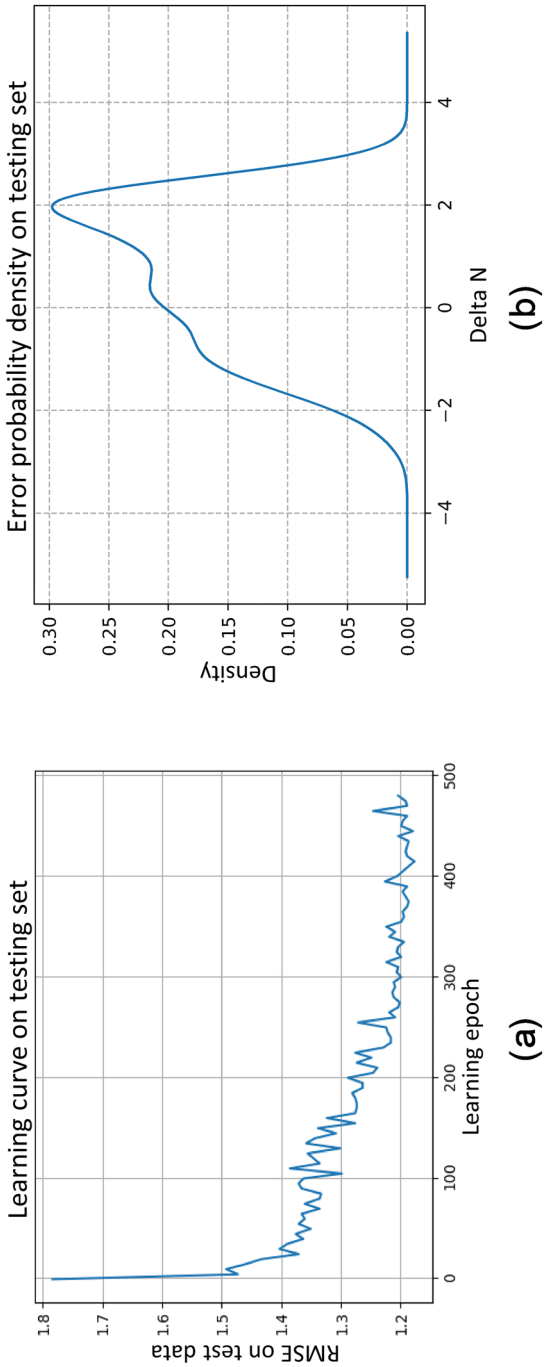

Fig. 9 **a** Learning curve and **b** final error density distribution on the testing set

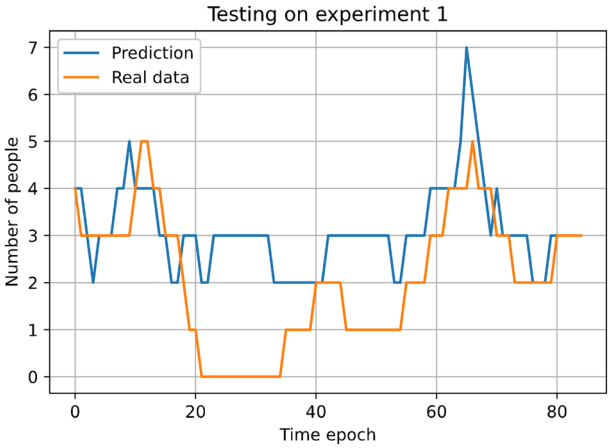

Fig. 10 Output in time for one of the testing datasets

the bus stop but do not remain there: these are the length of the time window Δt and the number of time windows $N_{\Delta t}$ that form the visibility period. If the time window duration is set too low, devices at the stop may be missed, but if it is too long there is a greater likelihood of counting devices that are simply passing by the stop, and there are also risks associated with MAC address randomization. The reference time window duration values were established based on prior research (Matte 2017) and in-house experiments in between arrival times of probe requests from client devices. Variations from the reference value, which ranged from 30 to 60 s, were explored in relation to fluctuations caused by device usage and model differences during the hyper parameter optimization phase of model training. The existence of multiple time windows helps smooth out any minor fluctuations resulting from passing devices.

The data at hand enabled us to train more than 40 models, each using distinct parameter values. Through careful analysis, we identified the best model by selecting the parameter values that yielded the most favorable outcomes on the testing dataset during the present study, as presented in Table 8. Even though more than 40 different configurations were used, Table 8 is restricted to only a few selected model configurations, along with the configuration that produced the best result.

The learning curve on the testing set for the best model with mean absolute error of 1.14 (Fig. 9a) shows that the algorithm improves its performance during the training by avoiding overfitting. The final distribution for the error (computed as error = prediction − real) is shown in Fig. 9b.

The final performance of the algorithm has a MAE of ~ 1.2 persons on the testing set. The visibility period of the final algorithm (the period taken into account in generating the people-count) is of length $\Delta t^{*}N_{\Delta t}=5$ min, based on $\Delta t=30$ s and $N_{\Delta t}=10$, as reported in Table 8. As can be seen from the density plot for the error (Fig. 9b), the algorithm has a positive bias, so it tends to overestimate the number of people at the bus stop. This can also be seen in Fig. 10, where the output in time for one of the experiments is shown. The tendency to overestimate might be due to a number of factors, such as the model requiring a larger training dataset to

Fig. 11 Search interface (dropdown menu)

Fig. 12 Global search screen

learn underlying patterns, or practical considerations, such as passengers carrying multiple devices, which can impact the estimation to some extent. The problem of overestimation resulting from passengers carrying multiple devices is a more difficult problem to overcome within the scope of this study, especially if the types of devices being carried are different—smartphones, tablets, and laptops—and if more than one device is used around the same time.

4.2 Data visualisation through the dashboard

To facilitate the monitoring of crowding at bus stops, the dashboard has two main screens. The first is a "search interface" (Fig. 11) featuring a dropdown menu to inform the user of all the cities and bus stops actively in the system. Alternatively, the user may search for and filter bus stops via city names, bus stop names, or bus numbers. As shown in Fig. 12, the query is global, meaning that a number or a few letters can be entered into a single text box, and the system automatically searches in all the fields.

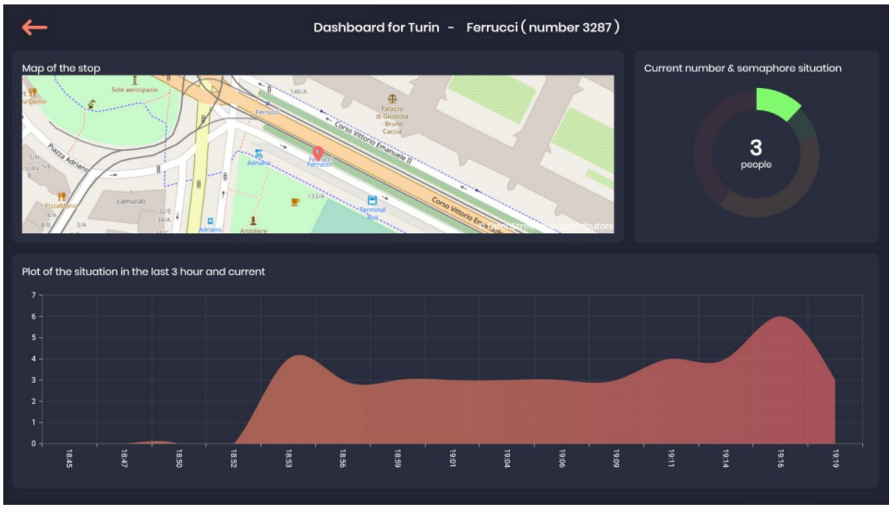

Fig. 13 Dashboard overview

Results are shown in real time and the user can select the entry of interest, which then loads on the next screen. The next screen shows the people count with an indicator of how crowded the stop is. The indicator will be one of three different colours: green, yellow, or red. Since stops are differently dimensioned, the transport operator can specify for each individual stop how numbers of people are to be mapped to colours. On the left there is a dynamic map with a location pin, marking the position of the bus stop. And finally, there is a plot showing the situation over the preceding three hours. The overview screen is shown in Fig. 13. The use of Flutter allows the dashboard to be adapted easily to different screen characteristics and operating systems.

5 Discussion and conclusions

Automated people counting at bus stops has several practical benefits for public transport management. It provides information about the number of people waiting for buses, which can help to ensure efficient resource allocation based on demand. Focusing on those waiting at stops, rather than on those who have actually boarded buses, can help to better understand specific stop-related demand. It can also offer valuable insights into passenger behaviours relating, for example, to waiting times, seasonal fluctuations, and boarding tendencies. This kind of information is crucial for optimizing service planning and resource distribution. An accurate count of passengers can guide informed decisions about resource allocation, even within budget constraints. It assists in selecting appropriate locations for resource installation, considering costs and maintenance needs.

For instance, with limited budgets, decisions about installing digital displays for real-time information or providing shelter for waiting passengers can have

significant implications for installation and maintenance costs. These decisions can be informed due to the knowledge of people counting systems and in understanding crowding patterns at stops. Decisions about bus stop design can in their turn impact bus dwell times and contribute to reducing congestion (Tirachini 2014). Overall, automated passenger counting at bus stops contributes to more effective public transport management and informed decision making.

To this end, the goal of this study was to estimate the number of people waiting at a bus stop, with Wi-Fi probe request packets emitted by modern ICT devices as source data, and using deep learning (DL) methodologies. The system that was developed has a number of advantages, employing a robust, low-cost device that is easy to install, requires little maintenance, and makes additional bulky devices unnecessary. The optimal DL model within the current study, using a Convolutional Recurrent Neural Network (CRNN) with Long Short-Term Memory (LSTM) architecture for the Recurrent Neural Network (RNN), yielded the best results, predicting the number of people waiting at the stop with a mean absolute error of 1.2 persons. This is a promising preliminary outcome, given the complexity of the experiment.

However, there are factors that can adversely affect the performance of the algorithm: not everyone owns a Wi-Fi capable device; the Wi-Fi interface may not be enabled; and there may be multiple Wi-Fi devices linked to the same user and undetected transmissions due to mobile devices passing through areas quickly, as also highlighted by Li et al. (2020). Additionally, environmental factors, such as noise and uncontrolled surroundings, can negatively impact the system's performance. In order to reduce the impact of the randomization of the MAC addresses and background noise, and to obtain an initial model capable of operating in a more controlled environment, the main bus stop in this study was placed at some distance from a road intersection to provide a reference case for future applications in more complex environments. However, other bus stops with a more complex environment were also used in the course of this study, including one in front of the city's central railway station, further enriching the breadth of the considered scenarios. However, it is important to recognize that most of the data collection was carried out at a single location, which may limit the generalizability of the results.

A big challenge in using the DL approach is obtaining a ground truth dataset that is sufficiently large to train the model. The "People Counter" app was designed to make manual counting more efficient and less resource-intensive, and to remove the need to translate manually collected data into digital format. The reliability and efficiency of this app allowed us to devote fewer resources to counting, and to increase the quantity and the accuracy of the collected data. Previous studies on automatic people counting evaluated their systems with limited numbers of people, such as up to 10 people (Choi et al. 2022; Ibrahim et al. 2019), or up to 5 people in an indoor environment (Liu et al. 2019). In contrast, the system evaluated in the present study was tested in an open environment with up to 30 people.

Another step forward as regards previous studies, to not compromise the reliability of DL models, was the extended duration of the data collection if compared, for example, to Oransirikul et al. (2014), who collected data for only

70 min, with 60 min for counting people at the bus stop. The reason is that a limited data collection can create an imbalanced data distribution for different count classes, which limits the availability of large amounts of training data for deep learning models (Ibrahim et al. 2019). However, making valid comparisons of the accuracy of our proposed people-counting system against that of existing crowd counting systems is far from straightforward, given the different testing approaches involved. Real data versus synthetic data is also an advantage to balance the training phase during model building, allowing an increase in accuracy, precision, recall, and F1 as well as a lower rate of false positive and false negative predictions (Rankin et al. 2020). The quality of synthetic data also depends on the source data, which can introduce bias and miss random behaviors of real-world data. Our study was able to confirm the system's accuracy using over 14 h of manual counting data as ground truth, with no use of synthetic data.

Given the various limitations mentioned above that are inherent in Wi-Fi sniffing, a significant factor in improving our proposed system will be the amount of data collected for training the neural network. The data must be acquired over a long period and at different times of the day and at various types of stops. Territorial variables such as the classification of stops based on the number of people present and the type of territory may potentially be used as inputs to enhance the performance of the network. Potential future research avenues include testing this system in a variety of scenarios, including counting passengers at different types of stops, and extending it to systems of transport other than buses, that is to say to trams, to metros, and at train stations. Future research could explore utilizing real-time passenger counts at bus stops to refine existing models that predict passenger arrival times, queue lengths (Jee et al. 2023) and patterns. This could facilitate long-term planning, route scheduling, and the optimization of headways and timetables (Olivo et al. 2019). Finally, in a future work we could consider associating the measurements of our system with the real-time bus passing times to monitor additional metrics.

Finally, we may assume that wireless technologies will be subject to future developments, in the same way that MAC address randomization was introduced a few years ago, and it is therefore important to keep abreast of new technologies and trends with a view to proposing automated counting systems that are alternatives to the well-established but more expensive camera-based systems.

Authors contributions Conceptualization: CP; Methodology: CP and DA; Formal analysis and investigation- Mobile app and Visualisation: GT; Formal analysis and investigation- Data Analysis: FS and DA; Writing—original draft preparation: DA; Writing—review and editing: CP; Supervision: CP.

Funding Open access funding provided by Politecnico di Torino within the CRUI-CARE Agreement.

Data availability The datasets generated during and analyzed during the current study are not publicly available due to sensitive business information that is not intended for public dissemination but are available from the corresponding author on reasonable request.

Declarations

Conflict of interest No potential conflict of interest was reported by the authors.

References

APPLE inc. (2021) Wi-Fi privacy [WWW Document]. Apple Support. URL https://support.apple.com/guide/security/wi-fi-privacy-secb9cb3140c/web. Accessed 19 Aug 2022

Arduino (2022a) UNO R3 | Arduino Documentation [WWW Document]. URL https://docs.arduino.cc/hardware/uno-rev3

Arduino (2022b) Nano 33 IoT | Arduino Documentation [WWW Document]. URL https://docs.arduino.cc/hardware/nano-33-iot

Arrow Development Tools (2023) HANI-IOT per Arrow Development Tools | Sistema integrato, schede e kit di sviluppo [WWW Document]. Arrow.com. URL https://www.arrow.com/it-it/products/hani-iot/arrow-development-tools. Accessed 27 Mar 2023

Basalamah A (2016) Sensing the crowds using Bluetooth low energy tags. IEEE Access 4:4225–4233. https://doi.org/10.1109/ACCESS.2016.2594210

Baumann D, Sommer M, Schrempp Y, Sax E (2022) Use of deep learning methods for people counting in public transport. In: 2022 International Conference on Connected Vehicle and Expo (ICCVE), pp 1–6. https://doi.org/10.1109/ICCVE52871.2022.9742924

Bernini N, Bombini L, Buzzoni M, Cerri P, Grisleri P (2014) An embedded system for counting passengers in public transportation vehicles. In: 2014 IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA), pp 1–6. https://doi.org/10.1109/MESA.2014.6935562

Brassington G (2017) Mean absolute error and root mean square error: which is the better metric for assessing model performance? Presented at the EGU General Assembly Conference, p 3574

Carrel A, Lau PS, Mishalani RG, Sengupta R, Walker JL (2015) Quantifying transit travel experiences from the users' perspective with high-resolution smartphone and vehicle location data. Transp Res Part C: Emerg Techn 58:224–239. https://doi.org/10.1016/j.trc.2015.03.021

Charansonney L (2018) New ways to collect traffic data, new challenges for road network authorities (PhD thesis). Université Paris-Est

Choi H, Fujimoto M, Matsui T, Misaki S, Yasumoto K (2022) Wi-CaL: WiFi sensing and machine learning based device-free crowd counting and localization. IEEE Access 10:24395–24410. https://doi.org/10.1109/ACCESS.2022.3155812

Cisco (2020) Cisco Annual Internet Report - Cisco Annual Internet Report (2018–2023) White Paper

DiPietro R, Hager GD (2020) Deep learning: RNNs and LSTM. In: Zhou SK, Rueckert D, Fichtinger G (eds) Handbook of medical image computing and computer assisted intervention, The Elsevier and MICCAI Society Book Series. Academic Press, pp 503–519. https://doi.org/10.1016/B978-0-12-816176-0.00026-0

Fenske E, Brown D, Martin J, Mayberry T, Ryan P, Rye E (2021) Three years later: a study of MAC address randomization in mobile devices and when it succeeds. Proc Priv Enhancing Technol 2021:164–181. https://doi.org/10.2478/popets-2021-0042

Fihn J, Finndahl J (2011) A framework for how to make use of an automatic passenger counting system. Uppsala University. Accessed from: http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-158139

Gao R, Zhao M, Ye T, Ye F, Wang Y, Luo G (2017) Smartphone-based real time vehicle tracking in indoor parking structures. IEEE Trans Mob Comput 16:2023–2036. https://doi.org/10.1109/TMC.2017.2684167

Gast MS (2013) 3. The MAC - 802.11ac: A Survival Guide [Book] [WWW Document]. URL https://www.oreilly.com/library/view/80211ac-a-survival/9781449357702/ch03.html

Graves A, Mohamed A, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE. pp 6645–6649. https://doi.org/10.1109/ICASSP.2013.6638947

Guillen-Perez A, Cano M-D (2019) Counting and locating people in outdoor environments: a comparative experimental study using WiFi-based passive methods. ITM Web Conf 24:01010. https://doi.org/10.1051/itmconf/20192401010

Gupta GK, Sharma DK (2022) A review of overfitting solutions in smart depression detection models. In: 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom), pp 145–151. https://doi.org/10.23919/INDIACom54597.2022.9763147

Győrödi C, Győrödi R, Pecherle G, Olah A (2015) A comparative study: MongoDB vs. MySQL. In: 2015 13th International Conference on Engineering of Modern Electric Systems (EMES), pp 1–6. https://doi.org/10.1109/EMES.2015.7158433

Hardt M, Recht B, Singer Y (2016) Train faster, generalize better: stability of stochastic gradient descent. In: Proceedings of the 33rd International Conference on Machine Learning, pp 1225–1234

Hasan MN, Toma RN, Nahid A-A, Islam MMM, Kim J-M (2019) Electricity theft detection in smart grid systems: a CNN-LSTM based approach. Energies 12:3310. https://doi.org/10.3390/en12173310

Hidayat A, Terabe S, Yaginuma H (2018) WiFi scanner technologies for obtaining travel data about circulator bus passengers: case study in obuse, Nagano Prefecture, Japan. Transp Res Rec 2672(45):45–54. https://doi.org/10.1177/0361198118776153

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Ibrahim OT, Gomaa W, Youssef M (2019) CrossCount: a deep learning system for device-free human counting using WiFi. IEEE Sens J 19:9921–9928. https://doi.org/10.1109/JSEN.2019.2928502

Infineon Technologies AG (2022) CY8CKIT-062-WIFI-BT - Infineon Technologies [WWW Document]. URL https://www.infineon.com/cms/en/product/evaluation-boards/cy8ckit-062-wifi-bt/

Jee H, Sun W, Schmöcker J-D, Nakamura T (2023) Demonstrating the feasibility of using Wi-Fi sensors for dynamic bus-stop queue length estimation. Public Transp. https://doi.org/10.1007/s12469-023-00336-5

KODY (2018) How to track Wi-Fi devices & connect to them using probequest [WWW Document]. WonderHowTo. URL https://null-byte.wonderhowto.com/how-to/track-wi-fi-devices-connect-them-using-probequest-0186137/

Kos-Łabędowicz J (2019) Is there an "intergenerational gap" in transport usage? Comparing transport preferences of seniors and students. Pr Nauk Uniw Ekon We Wrocławiu 63:58–70

Kurkcu A, Ozbay K (2017) Estimating pedestrian densities, wait times, and flows with Wi-Fi and Bluetooth Sensors. Transp Res Rec 2644:72–82. https://doi.org/10.3141/2644-09

Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324. https://doi.org/10.1109/5.726791

Lee S, Kim M, Kang S, Lee K, Jung I (2012) Smart scanning for mobile devices in WLANs. In: 2012 IEEE International Conference on Communications (ICC). IEEE, pp 4960–4964

Li Y, Barthelemy J, Sun S, Perez P, Moran B (2020) A case study of WiFi sniffing performance evaluation. IEEE Access 8:129224–129235. https://doi.org/10.1109/ACCESS.2020.3008533

Liang K, Qin N, Huang D, Fu Y (2018) Convolutional recurrent neural network for fault diagnosis of high-speed train bogie. Complexity 2018:e4501952. https://doi.org/10.1155/2018/4501952

Liu S, Zhao Y, Xue F, Chen B, Chen X (2019) DeepCount: crowd counting with WiFi via deep learning. https://doi.org/10.48550/arXiv.1903.05316

Liu D, Zhao C, Dong H, Huang Z (2022) Spatial analysis of bus rapid transit actual operating conditions: the case of Hangzhou City, China. Public Transp 14:503–519. https://doi.org/10.1007/s12469-022-00299-z

Matte C (2017) Wi-Fi tracking: fingerprinting attacks and counter-measures. Insa Lyon

Mccarthy C, Moser I, Jayaraman PP, Ghaderi H, Tan AM, Yavari A, Mehmood U, Simmons M, Weizman Y, Georgakopoulos D, Fuss FK, Dia H (2021) A field study of internet of things-based solutions for automatic passenger counting. IEEE Open J Intell Transp Syst 2:384–401. https://doi.org/10.1109/OJITS.2021.3111052

Mehmood U, Moser I, Jayaraman PP, Banerjee A (2019) Occupancy estimation using wifi: a case study for counting passengers on busses. In: IEEE 5th World Forum Internet Things WF-IoT 2019 - Conf. Proc. pp 165–170. https://doi.org/10.1109/WF-IoT.2019.8767350

Mikkelsen L, Buchakchiev R, Madsen T, Schwefel HP (2016) Public transport occupancy estimation using WLAN probing. In: Proceedings of 2016 8th International Workshop on Resilient Networks Design and Modeling, RNDM 2016. Institute of Electrical and Electronics Engineers Inc., pp 302–308. https://doi.org/10.1109/RNDM.2016.7608302

Myrvoll TA, Håkegård JE, Matsui T, Septier F (2017) Counting public transport passenger using WiFi signatures of mobile devices. In: 2017 IEEE 20th Int. Conf. Intell. Transp. Syst. ITSC. https://doi.org/10.1109/ITSC.2017.8317687

Nishide R, Takada H (2013) Detecting pedestrian flows on a mobile ad hoc network and issues with trends and feasible applications. Int J Adv Netw Serv 6(1 & 2):108–117

Nitti M, Pinna F, Pintor L, Pilloni V, Barabino B (2020) Iabacus: A Wi-Fi-based automatic bus passenger counting system. Energies 13:1–21. https://doi.org/10.3390/en13061446

Oberli C, Torres-Torriti M, Landau D (2010) Performance evaluation of UHF RFID technologies for real-time passenger recognition in intelligent public transportation systems. IEEE Trans Intell Transp Syst 11:748–753. https://doi.org/10.1109/TITS.2010.2048429

Olivo A, Maternini G, Barabino B (2019) Empirical study on the accuracy and precision of automatic passenger counting in European bus services. Open Transp J 13:250–260. https://doi.org/10.2174/1874447801913010250

Oransirikul T, Nishide R, Piumarta I, Takada H (2014) Measuring bus passenger load by monitoring Wi-Fi transmissions from mobile devices. Procedia Technol 18:120–125. https://doi.org/10.1016/j.protcy.2014.11.023

Pintor L, Atzori L (2021) A dataset of labelled device Wi-Fi probe requests for MAC address de-randomization. Comput Netw. Mendeley data. https://doi.org/10.17632/j64btzdsdy.1

Product detail - Infineon-Cypress/CY8CKIT-062-WIFI-BT (2022) URL https://www.mouser.it/ProductDetail/Infineon-Cypress/CY8CKIT-062-WIFI-BT?qs=lYGu3FyN48c6Xy1GM0bmkg%3D%3D. Accessed 20 June 2022

Purvis W, Dementyev S (2020) Get to know MAC Address Randomization in 2020. Mist. URL https://www.mist.com/get-to-know-mac-address-randomization-in-2020/

Pycom (2022a) Pytrack 2.0 X [WWW Document]. Pycom. URL https://pycom.io/product/pytrack-2-0-x/. Accessed 25 Sept 2022. See also https://docs.pycom.io/datasheets/expansionboards/pytrack2/. Accessed 07 Feb 2024

Pycom (2022b) WiPy 3.0 [WWW Document]. Pycom. URL https://pycom.io/product/wipy-3-0/. Accessed 25 Sept 2022. See also https://docs.pycom.io/datasheets/development/wipy3/. Accessed 07 Feb 2024

Rakebrandt A (2015) Defining APC Accuracy Standards in North America [WWW Document]. Mass Transit. URL https://www.masstransitmag.com/technology/article/12128610/defining-apc-accuracy-standards-in-north-america. Accessed 2 July 2022

Rankin D, Black M, Bond R, Wallace J, Mulvenna M, Epelde G (2020) Reliability of supervised machine learning using synthetic data in health care: model to preserve privacy for data sharing. JMIR Med Inf 8:18910. https://doi.org/10.2196/18910

Raue F, Byeon W, Breuel TM, Liwicki M (2015) Parallel sequence classification using recurrent neural networks and alignment. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR. IEEE, pp 581–585. https://doi.org/10.1109/ICDAR.2015.7333828

Reichl P, Oh B, Ravitharan R, Stafford M (2018) Using Wifi Technologies to Count Passengers in Real-time around Rail Infrastructure. In: 2018 International Conference on Intelligent Rail Transportation (ICIRT), pp 1–5. https://doi.org/10.1109/ICIRT.2018.8641595

RELOC (2022) HANI IoT Board. RELOC Join Internet Things Revolut. Today. URL https://www.reloc.it/products/hani-iot-board/. See also https://www.reloc.it/news-hani-iot-arm-mbed-2/. Accessed 07 Feb 2024

Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323:533–536. https://doi.org/10.1038/323533a0

Saponara S, Pilato L, Fanucci L (2016) Exploiting CCTV camera system for advanced passenger services on-board trains. In: 2016 IEEE International Smart Cities Conference (ISC2), pp 1–6. https://doi.org/10.1109/ISC2.2016.7580748

Schauer L, Werner M, Marcus P (2014) Estimating crowd densities and pedestrian flows using Wi-Fi and Bluetooth. In: MobiQuitous 2014 - 11th International Conference on Mobile and Ubiquitous

Systems: Computing, Networking and Services. pp 171–177. https://doi.org/10.4108/icst.mobiquitous.2014.257870

Singh U, Determe J-F, Doncker P, Horlin F (2020) Crowd forecasting based on WiFi sensors and LSTM neural networks. IEEE Trans Instrum Meas 69(9):6121–6131. https://doi.org/10.1109/TIM.2020.2969588

Singh U, Determe J-F, Horlin F, De Doncker P (2021) Crowd monitoring: state-of-the-art and future directions. IETE Tech Rev 38:578–594. https://doi.org/10.1080/02564602.2020.1803152

Tirachini A (2014) The economics and engineering of bus stops: spacing, design and congestion. Transp Res Part A Policy Pract 59:37–57. https://doi.org/10.1016/j.tra.2013.10.010

Tu L, Wang S, Zhang D, Zhang F, He T (2021) ViFi-MobiScanner: observe human mobility via vehicular internet service. IEEE Trans Intell Transp Syst 22:280–292. https://doi.org/10.1109/TITS.2019.2956744

Uras M, Cossu R, Ferrara E, Bagdasar O, Liotta A, Atzori L (2020) WiFi probes sniffing: an artificial intelligence based approach for MAC addresses de-randomization. In: IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp 1–6. https://doi.org/10.1109/CAMAD50429.2020.9209257

Vanhoef M, Matte C, Cunche M, Cardoso LS, Piessens F (2016) Why MAC address randomization is not enough: an analysis of Wi-Fi network discovery mechanisms. In: ASIA CCS 2016 - Proc. 11th ACM Asia Conf Comput Commun Secur. pp 413–424. https://doi.org/10.1145/2897845.2897883

Voß S, Mejia G, Voß A (2020) Mystery shopping in public transport: the case of bus station design. In: Stephanidis C, Marcus A, Rosenzweig E, Rau P-LP, Moallem A, Rauterberg M (eds) HCI international 2020 - late breaking papers: user experience design and case studies. Lecture notes in computer science. Springer, Cham, pp 527–542. https://doi.org/10.1007/978-3-030-60114-0_36

Wang R, Li Z, Cao J, Chen T, Wang L (2019) Convolutional recurrent neural networks for text classification. In: 2019 International Joint Conference on Neural Networks (IJCNN), pp 1–6. https://doi.org/10.1109/IJCNN.2019.8852406

Wiest L (2017) Recurrent Neural Networks - Combination of RNN and CNN. BayernCollab. URL https://wiki.tum.de/display/lfdv/Recurrent+Neural+Networks+-+Combination+of+RNN+and+CNN. Accessed 1 Nov 2022

Willmott CJ, Matsuura K (2005) Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Clim Res 30:79–82. https://doi.org/10.3354/cr030079

Yahiaoui T, Khoudour L, Meurie C (2010) Real-time passenger counting in buses using dense stereovision. J Electron Imaging 19(3):031202. https://doi.org/10.1117/1.3455989

Yang H, Ozbay K, Bartin B (2010) Investigating the performance of automatic counting sensors for pedestrian traffic data collection. In: Proceedings of the 12th World Conference on Transport Research. Lisbon, Portugal, pp 1–11

Yu Y, Si X, Hu C, Zhang J (2019) A review of recurrent neural networks: LSTM cells and network architectures. Neural Comput 31:1235–1270. https://doi.org/10.1162/neco_a_01199