Regular Paper

Classifying Passenger and Non-passenger Signals in Public Transportation by Analysing Mobile Device Wi-Fi Activity

Thongtat Oransirikul[1,a)] Ian Piumarta[2,b)] Hideyuki Takada[2,c)]

Received: April 12, 2018, Accepted: October 2, 2018

Abstract: Quality of service is one factor passengers consider when deciding to use the public transportation system. Increasing the quality of service can be done in several ways, such as improving on-time arrival, reducing waiting time, and providing seat availability information. We believe that if passengers have better information for making decisions, that can increase the quality of service. This paper proposes estimating the number of passengers by analyzing signals from their Wi-Fi devices, classifying them as originating from passenger or non-passenger devices using a real-time filtering mechanism. Experimental validation was performed aboard busses of different types taking different routes. Our experimental results show that filtered data can classify passenger device signals from environmental ones with an accuracy of 75 percent, which is a promising basis for providing real-time information to passengers that improves the quality of their service.

Keywords: real-time Wi-Fi signal analysis, passenger information, public transportation, congestion, estimation

1. Introduction

Travel by public transportation is a daily activity for many people. One important factor for passengers when deciding whether to use the bus system is the quality of service. Quality of service for passengers can be improved in several ways related to infrastructure, equipment, scheduling, and vehicle/network status information. Improvements in scheduling and passenger information rely in particular on real-time bus location and congestion information. Bus location information is trivially and cheaply available, thanks to GPS. Reliable, real-time congestion information is more difficult to obtain using cheap, scalable methods. Congestion information is nevertheless highly valuable and, combined with short-term congestion predictions, gives operators the possibility of dynamic scheduling to deal with a fluctuating load, and gives passengers an increased awareness of the current and impending state of the bus transportation system. The latter is particularly important, since the actual and perceived quality of travel can be significantly improved when passengers are guaranteed a seat. In some cases this is vital, as with physically-challenged or pregnant people, in other cases merely facilitative, as with people who work on laptop computers while commuting. A passenger who is forewarned that an approaching vehicle has no seating room, whereas the one following close behind will have plenty, can make an informed decision to wait for the second vehicle.

[1] Graduate School of Information Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525–8577, Japan
[2] College of Information Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525–8577, Japan
[a)] t_oran@cm.is.ritsumei.ac.jp
[b)] piumarta@cs.ritsumei.ac.jp
[c)] htakada@cs.ritsumei.ac.jp

Making a short-term estimation about the load on a vehicle depends on two sources of information. First is the current load of the vehicle, which can be measured aboard the vehicle in a number of ways. Second is the expected future load of the vehicle, which depends on the number of passengers that could board or alight at each stop. With these two sources of information we aim to provide real-time information to passengers about the potential availability of seats, to improve their quality of service.

Our previous work [1], [2] focused on estimating the number of passengers waiting at bus stops in real-time, from which an upper bound on the number of passengers able to board a bus can be derived. The new contributions described in this paper relate to a real-time method for estimating the number of passengers who are travelling on board a bus.

Many people carry smartphones these days, and all smartphones are capable of Wi-Fi communication. The number of smartphones on board a bus is therefore a potentially useful proxy for the number of passengers aboard the bus. Our approach to estimating the congestion on board a bus is to monitor Wi-Fi signals, classify them as originating from non-passenger or passenger devices, and use the result as a proxy to estimate the number of passengers on board the bus. The Wi-Fi activity we monitor is transmission of *probe requests* by smartphones searching for known networks, which they do regularly. This method is non-participatory (passengers do not need to install an app or visit a particular web site), non-invasive (passengers' normal use of their devices is not affected), and easy to implement.

The rest of this article is organized as follows. Section 2 describes related work. Section 3 describes our methodology for classifying signals. Section 4 presents our experimental method and results. Section 5 discusses our results, and Section 6 offers conclusions and thoughts about future work.

2. Related Work

Direct methods of counting passengers can be effective, but are usually invasive and/or participatory. Lim et al. [3] try to exploit availability and capability of passengers' mobile phones to share location information, requiring active consent and participation (via an app) from the passengers when they travel with the bus.

Optical and thermal technologies have been used to count passengers without their participation. Chen et al. capture real-time bus passenger flow statistics using light-sensitive wireless sensors installed at doors [4] or video processing [5] to detect passengers entering or leaving the bus, inferring an approximate running total of passengers aboard. Bernini et al. [6] also investigated counting the number of passengers aboard a bus using a camera (stereo couple) and image processing. While the accuracy of these approaches can be high, in difficult conditions they suffer from cumulative errors of as much as 33% that do not occur in our system. Khoeblal et al. [7] tried to catch fare dodgers by counting passengers using a DILAX (thermal) People Counting Unit and a RFID distance scanner.

Radio signal analysis, of both Bluetooth and Wi-Fi, have also been used. Nishide et al. [8] detected pedestrians from their devices' Bluetooth transmission in many places, but this technique is far less reliable recently because many people do not use Bluetooth and deactivate it to save battery power.

Invasive Wi-Fi monitoring has also been used. Nakano et al. [9] detects passengers on a train by installing a fake access point and counting the number of devices connected to it. Yamakawa et al. [10] also estimated passenger congestion by installing an access point in a train and capturing device signals using a Riverbed AirPcap-Nx, reporting an accuracy of 63.5%.

Mikkelsen et al. [11] use a mechanism very similar to ours but using only two parameters, RSSI and duration of device visibility, to obtain 50% accuracy. Compared to their mechanism, ours uses five filtering parameters in a two-tier approach that accounts for devices likely located inside or outside the bus and performs significantly better. Our mechanism also takes into account the elapsed time since a signal was last received from a particular device, allowing it to provide almost real-time data.

Our previously-reported work [1], [2] estimated the number of passengers waiting at bus stops using similar techniques, but either ignored RSSI or did not attempt to provide near real-time information.

Raspberry Pi has become a popular platform for building inexpensive Wi-Fi analyzers. Apetroaie-Cristea et al. [12], for example, used a Raspberry Pi to physically locate Wi-Fi devices, choosing channel 11 for collecting their data. Nalawade et al. [13] uses a Raspberry Pi and a GSM/GPRS module to track a bus by using telephone cell tower information.

3. Signal Classification Method

To estimate the number of passengers who are traveling on a bus we monitor and analyze the Wi-Fi probe request signals captured inside the bus. However there are many Wi-Fi signals coming from many directions around the bus, and we need to identify the signals sent from passengers' mobile devices. We therefore need a filtering mechanism to separate passenger device Wi-Fi signals from those coming from the environment outside the bus.

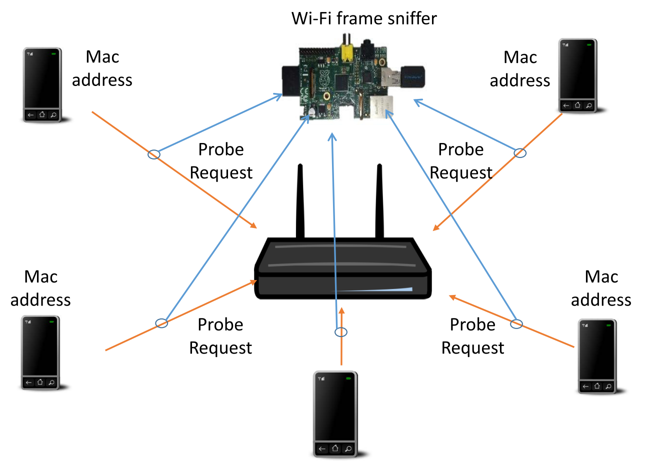

Fig. 1 System structure.

3.1 Collecting Wi-Fi Packets

Figure 1 illustrates the system we use to detect Wi-Fi activity. Wi-Fi signals consist of many types of packet. One type is a probe request packet, which is sent from mobile devices to search for known access points. Mobile devices send probe requests even when already connected to an access point, so we can easily eliminate packets sent from access points by discarding all but probe request packets.

The Wi-Fi sniffer captures packets transmitted by mobile devices. When seeking access points, mobile devices broadcast a probe request packet whose header includes the sender's Medium Access Controller (MAC) hardware address, a sequence number, and the packet type. Our sniffer adds further information about the signal strength. Intercepting probe requests is non-invasive and therefore preferable to advertising a fake open wireless network to attract mobile device connections, which could disrupt the passenger's normal network connectivity.

3.2 Receiver Hardware and Software

To implement the packet sniffer we found Wi-Fi hardware that supports 'monitor mode'. Monitor mode allows the network interface to be used for packet sniffing by making it receive all packets, even those not addressed to it.

An Arduino microcontroller [14] with a Wi-Fi interface was one device considered for monitoring, but the radio cannot be placed in monitor mode for packet sniffing. We eventually chose to use a Raspberry Pi [15] and a Powerlink PL-U2N USB Wi-Fi adapter, using the Ralink RT5370 chipset [16] which does support monitor mode (The Raspberry Pi is cheap and small, and has low energy consumption making it suitable to be installed on a bus with a USB Wi-Fi adapter).

We use dumpcap, part of the wireshark [17] suite for network analysis, to record captured Wi-Fi packets. It can control the channel used for capturing, using either all available channels (by channel hopping) or only a single channel. We set the channel for capturing packets to channel 6 in the 2.4 GHz Wi-Fi band (Capturing on only one channel is more efficient for detecting pack-

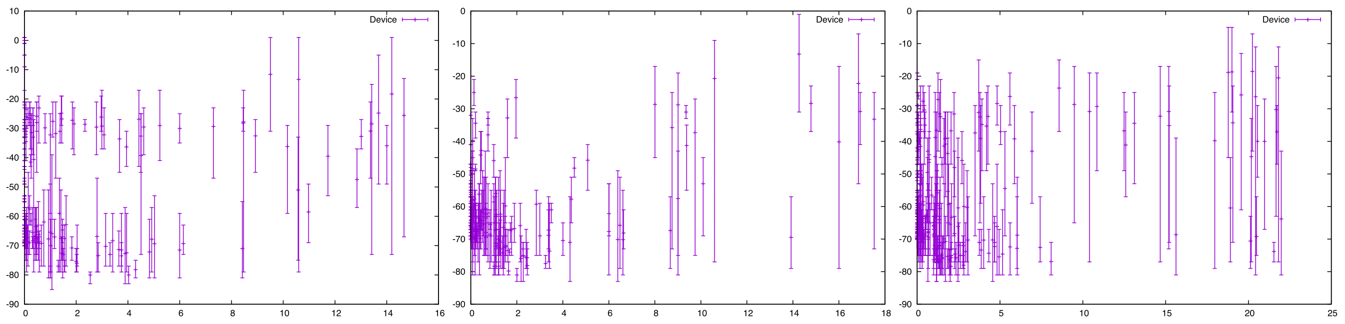

Fig. 2 Relationship of RSSI versus duration for three different bus trips.

ets, as it avoids channel hopping. From a short experiment we discovered that most devices use channel 6). For real-time packet filtering we set dumpcap to begin a new capture file every minute.

To separate mobile devices from base stations, link-layer headers for each received frame were captured to a SD card by dumpcap. We use tshark (also part of wireshark) to convert them to plain text, and awk (a standard Unix utility) to extract just the probe request packets for our filtering mechanism (Probe requests are broadcast only by mobile devices searching for base stations).

3.3 Filtering Parameters

We do not know if the Wi-Fi signal from a mobile device came from inside or outside the bus. To separate passenger signals from environmental signals (people passing by, etc.) we have to remove the outside signals from the collected Wi-Fi activity data.

Figure 2 shows the received signal strength indication (RSSI) against the total time duration for which each device was detected. Each vertical line shows the minimum, maximum, and average RSSI. We see varying RSSI values inside a bus, so we consider using RSSI as one of the parameters for filtering.

Another parameter we consider is time duration, which we can also use to identify a passenger device signal. If we detect a device only for a few seconds, that device is classified as a non-passenger signal.

For filtering data in real-time we need some parameter for eliminating a device after its passenger gets off the bus. We call this parameter *idle time*. This parameter is calculated as the current time minus the time when the last packet was received from that device. If a device is not detected for more than our idle time parameter, that device will be rejected because it probably belongs to a passenger who got off the bus at the last bus stop.

3.4 Validation of RSSI as a Filtering Parameter

To verify the intuitive relationship between RSSI and distance, a short experiment was conducted at a bus stop using three typical mobile devices (iOS and Android). The devices were placed at distances between 1 and 15 meters from a packet sniffer, at 2-meter intervals, and 100 packets were captured from each device at each location.

Figure 3 shows the results of these signal strength experiments at the bus stop. Signal strength falls steadily with increasing distance, from −31 dBm to −58 dBm. Beyond 13 meters the signal remains relatively steady.

To verify the existence of some useful relationship between

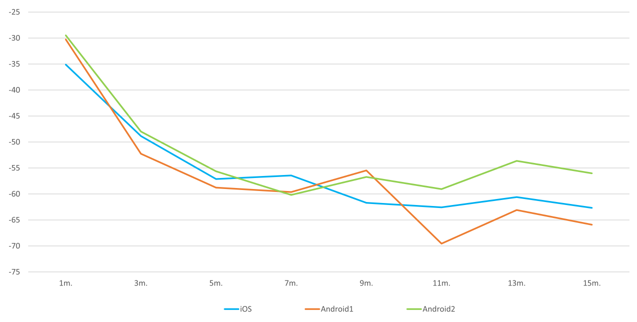

Fig. 3 Relationship of RSSI versus distance at the bus stop.

Table 1 RSSI inside bus.

	Front1	Rear1	Front2	Rear2
iOS	−44.76	−33.54	−38	−37.1
Android 1	−49.18	−34.74	−42.86	−36.94
Android 2	−39.1	−31.6	−39.98	−30.74
AVG.	−44.34	−33.29	−40.28	−34.92

Table 2 RSSI outside bus.

	Outside
iOS	−49.8
Android 1	−51.06
Android 2	−46.06
AVG.	−48.97

RSSI and location inside a bus, another short experiment was conducted on a bus using the same three typical mobile devices. RSSI measurements were made at both ends of the bus, front and rear, and we used the Raspberry Pi packet sniffer system running dumpcap, placed always at the same location in the middle of the bus. We performed the experiments twice, from both ends of the bus, capturing 100 packets from each device to calculate the average RSSI value.

Table 1 shows the results of the experiment inside the bus. The RSSI inside the bus varied from −33 dBm to −44 dBm. The Signal strength from the rear end is consistently stronger than from the front end. The rear end of the bus has more seats than the front, so the signal may be affected by reflection or obstruction.

To verify a useful relationship between RSSI and location outside the bus, we used the same packet sniffer (inside the bus) and the same three mobile devices placed 3 meters away from the bus. We again calculated the average RSSI value from 100 packets captured during the experiment.

Table 2 shows the results of the experiment from outside the bus. The average RSSI from outside the bus was −49 dBm. The Signal strength from devices outside the bus is therefore generally

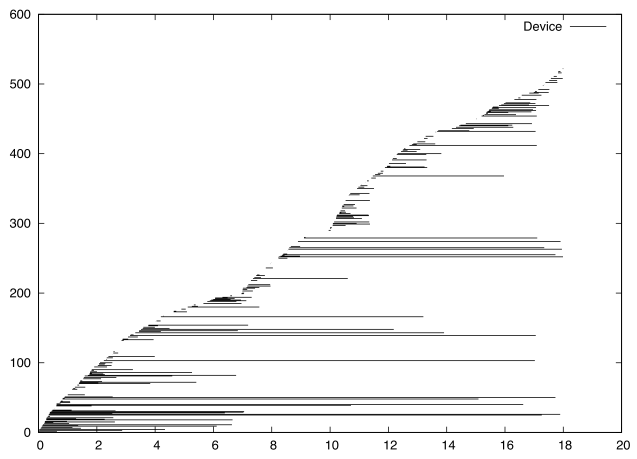

Fig. 4 Duration of each device detected during a journey.

weaker than from those inside the bus.

These results demonstrate a useful relationship between RSSI and distance from the packet sniffer, both outside and inside the bus. We can verify that the signal strength of devices inside the bus is generally stronger than that of devices outside the bus. This result is in qualitative agreement with other similar experiments investigating environmental effects on the signal strength, such as Luciani and Davis [18].

3.5 Defining Filtering Conditions Using Real Data

Regarding the interior of the bus, the front end has more standing space than the rear end of the bus but the rear end has more seats. The Signal strength can vary for many reasons such as the reflection and the absorption by passengers. In the middle and right graphs of Fig. 2 we see many devices that were detected for a long duration but which had average signal strengths of around −60 dBm, whereas in the left graph the long-duration devices have an average RSSI of around −40 dBm. We therefore use two different combinations of parameter values to detect probable passenger device signals: one is a strong signal strength even when duration is not yet known, and the other is a (potentially) weak signal strength but a longer observed duration.

Figure 4 shows the duration of each device's signal, from the time when the first packet was received until the time when the last packet was received, during a typical bus trip. We see many non-passenger signals (short lines) and we try to classify those as non-passenger signals using our filtering mechanism. We use a minimum detected duration of 90 seconds to classify a signal as belonging to a passenger device, because 90 seconds is the shortest time between two bus stops. Devices that are detected for fewer than 90 seconds probably represent external signals, from passers by, devices in other vehicles, passengers waiting at a bus stop, etc.

We also consider the number of packets received when classifying a device signal. If only a single packet is received then that signal will be classified as non-passenger. This applies particularly to situations where only a single packet with a very strong signal strength is received from a device, which we classify as non-passenger.

Real-time classification of signals can provide the most useful information to passengers. We use an idle time of 90 seconds to detect devices that are no longer on the bus. Since the shortest time between two bus stops is 90 seconds, we use 90 seconds as the idle time parameter in the classification algorithm.

```
if $avgRssi > -40$ and $npkt \geq 2$ and $idle \leq 90$ then
    $nEst+ = 1$
else
    if $duration > 90$ and $avgRssi > -60$ and $idle \leq 90$ then
        $nEst+ = 1$
    end if
end if
```

Algorithm 1 Filtering method parameter.

Our classification mechanism therefore uses a novel two-tier approach with five parameters. RSSI is used to roughly classify signals as probably inside (> −40 dBm), possibly inside, and probably outside (< −60 dBm) the bus; signals probably originating outside are classified as being from non-passenger devices. Minimum duration of 90 seconds is then used to qualify possible passenger signals, whereas 'probably inside' signals need only be received twice to qualify, regardless of the reception interval. In either case, most-recently received signals that are more than 90 seconds old are classified as non-passenger.

Algorithm 1 shows the classification algorithm using a combination of received signal strength indication, total duration for which the device's signal was detected (time of first packet to time of last packet), the number of packets that have been received from the device, and the idle time since the last packet was received from the device.

4. Experiments and Evaluation

In this section we explain our experimental design and methods, and present our results.

4.1 Experimental Design

Our bus experiments were designed to test the accuracy of estimating the number of passenger devices on the bus by monitoring their Wi-Fi signal activity. Two types of bus and six different bus routes were selected. Busses depart from the university to the nearest train station and then return to the university. There are several different routes between the university and the train station, some passing a large industrial factory and others passing schools or through villages.

Multiple experiments were performed on the following types of bus routes. A *local bus* is a normal bus route stopping at every stop for passengers to get on or off. The *shuttle bus* is a non-stop bus between the university and train station terminals with no stops in between, and therefore no passengers get on or off the bus during the journey. The shuttle bus follows the same route as one of the local busses.

4.2 Experimental Method

During the experiments we recorded the actual number of passengers on the bus at each minute. We also recorded the number of passengers getting on or off at each bus stop, the departure time, and the arrival time. **Table 3** shows an example of observations made during an experiment travelling with the bus. Each row represents one minute of the journey. The first column is the

Table 3 Bus trip observations.

Min	Number of Pax	Remark
1	9	
2	12	Departured
3	12	
4	12	
5	12	
6	13	get on 1
7	13	
8	13	
9	13	get on 1, get off 1
10	13	
11	13	Arrived

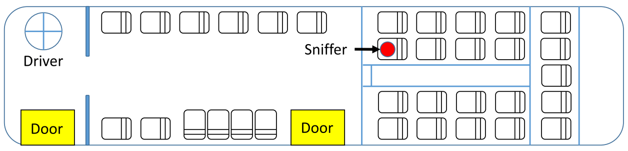

Fig. 5 Bus layout.

cumulative duration of the bus trip in minutes, the second column is the number of passengers aboard during each of those minute-long intervals, the third column lists significant events such as bus departure/arrival or passengers getting on/off the bus. We use this information to visualize passengers' behaviour on the bus to help validate a control data set for evaluation of our real-time classification mechanism.

Experiments were conducted between 2017/07/04 and 2017/08/17. Each experiment was performed inside a bus at various times and along various routes. **Figure 5** shows the layout of the bus and the Raspberry Pi packet sniffer's position which is indicated by a red circle. Although bus companies use several bus models, the length of a bus is always approximately 10 meters and the total number of seats inside the bus varies between 32 and 40 [19].

During the experiment the packet sniffer was active and recording Wi-Fi frame activity. It was placed at the same position every time in the middle of the bus (because the middle of the bus can better capture the Wi-Fi signals from all areas of the bus). We saved the data files generated by dumpcap on a SD card. For each packet received the system recorded the sender's MAC address, the time, the packet type, and the received signal strength indication.

4.3 Evaluation Method

One of the tools for measuring a classifier's accuracy is its F1 score. It considers both the precision and the recall. We use the formula shown below to calculate the precision and recall.

$$precision = \frac{Estimated\ Devices \bigcap Accurate\ Devices}{Estimated\ Devices}$$

$$recall = \frac{Estimated\ Devices \bigcap Accurate\ Devices}{Accurate\ Devices}$$

$$F = 2 \times \frac{precision \times recall}{precision + recall}$$

In order to calculate the precision and recall of our real-time algorithm, we first need to create a control data set of more accurately classified devices.

Table 4 F-Measure.

Date	Precision	Recall	F-Measure
July 4	0.79	0.62	0.67
July 6	0.7	0.73	0.7
July 7	0.82	0.58	0.67
July 10	0.76	0.8	0.77
July 31-1	0.74	0.68	0.67
July 31-2	0.56	0.75	0.62
July 31-3	0.81	0.75	0.62
July 31-4	0.8	0.72	0.73
July 31-5	0.49	0.53	0.5
August 17-1	0.83	0.63	0.71
August 17-2	0.77	0.75	0.74
August 17-3	0.86	0.72	0.76
August 17-4	0.87	0.64	0.72
AVG.	0.75	0.68	0.69

4.4 Obtaining a Control Data Set of Classification

We used complete knowledge of the future behavior of device signals to extract a control data set from the recorded data. Since the exact times of the first and last signals from a device are known, the idle time becomes irrelevant. We used two simple conditions to obtain a control data set containing signals with a very high probability of being from passenger devices.

(1) Max of RSSI > −40 dBm and duration > 60 seconds
(2) Average of RSSI > −60 dBm and duration > 240 seconds

The −40 dBm threshold comes from the RSSI experiment performed inside the bus. The weakest average signal strength is −44 dBm, but we want to be fairly certain that a signal was captured inside the bus so we choose a stronger value. As for the −60 dBm threshold, referring to Fig. 2, the signal strength has two separate groups and it has many long-duration devices with signal strengths between −60 dBm and −70 dBm. So we choose the strongest of these, −60 dBm. The 240 second threshold comes from the longest time between two bus stops, as a further justification that weak signals are from passenger devices.

4.5 Experimental Results

The binary data files recorded by dumpcap were converted to text files using tshark and prepared for the real-time classification mechanism. Our converter also selected only probe request packets from among all of the packets captured.

We used our real-time filtering algorithm to find the estimated number of passenger devices. The result was compared to the control data set using the F-measure (precision and recall) method. **Table 4** shows the F-measure result from all of the experiments. We performed the experiments on 13 bus trips representing several different bus types and routes. The F-measure score varies from 0.5 to 0.77. The average precision is 0.75, the average recall is 0.68, and the average F-measure is 0.69.

We also ran a real-time classification simulation to separate the passenger device signals from the environmental signals, and compared the resulting estimate of the number of devices to the observed number of passengers on the bus. The final column of **Table 5** shows the ratio of estimated devices to observed passengers.

Figure 6 shows example results from the experiments including the observed number of passengers, the control data set size, and the real-time classification results. The left-hand result is one of the best, with the estimated number of passenger device sig-

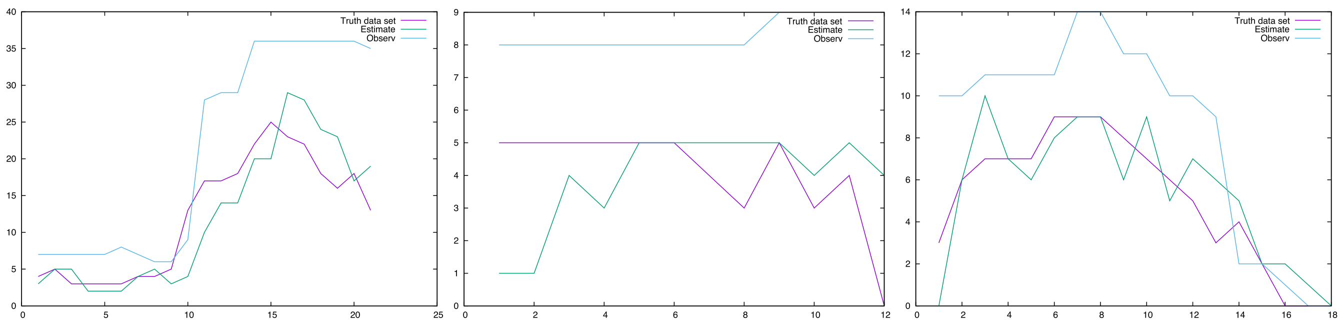

Fig. 6 Observed data, control data set, and real-time estimations for three different bus trips.

Table 5 Estimated number of passenger devices in real-time.

Min	Observed No. of Passengers	Estimated No. of Pax Devices	ratio
1	7	3	2.33
2	7	3	2.33
3	7	5	1.40
4	7	2	3.50
5	7	2	3.50
6	8	2	4.00
7	7	4	1.75
8	6	5	1.20
9	6	3	2.00
10	9	4	2.25
11	28	10	2.80
12	29	14	2.07
13	29	14	2.07
14	36	20	1.80
15	36	20	1.80
16	36	29	1.24

nals closely matching the control data set. The middle result is an average result, in which the control data set line is fairly stable for the first six minutes but the real-time estimate is stable at five people from minute 5 to minute 9, after which the real-time estimate begins to follow the control data set line. In the right-hand graph the control data set and the real-time estimate start at the same point and correlate well, but both show inconsistent agreement with the observed number of passengers.

5. Discussion

Our real-time classification method tries to separate signals of passenger devices travelling with the bus from non-passenger environmental device signals. For the real-time estimate we can classify a passenger device within 60 seconds of first detection, because dumpcap creates a new file every minute. When a passenger gets off the bus with an active device there is a 90-second delay before re-classifying that device corresponding to the idle time parameter. Increasing the idle time increases the number of device signals classified as passenger-related, but accuracy decreases and the additional delay is not good for real-time filtering.

The duration parameter of 90 seconds comes from the shortest time between two bus stops. We feel this duration justifies a signal as belonging to a passenger device since it appears to be travelling with the bus.

From our short experiments we know RSSI inside a bus is generally stronger than that coming from outside the bus. However, signal strengths from devices inside the bus can be lower than outside devices for reasons that we assume include weak batteries, signal absorption from passengers, etc. We used two different RSSI threshold values for our real-time classification method and considered strong signals with a short duration, or weak signals with a long duration, to be from passenger devices. Table 5 shows the number of estimated devices at each minute, and the ratio to the number of observed passengers travelling with the bus. The bus departed during minute 2. From the drop in the estimated number of devices in the next minute, maybe three signals were wrongly classified as passenger devices. Two people left the bus in minute 7 and another got on. One minute later a new signal was classified as being from a passenger device, but it was not until minute 9 that two passenger signals were demoted to non-passenger, possibly corresponding to the two passengers who left in minute 7.

In Table 4, the average of the recall and F-measure is around 70 percent, and the precision is 75 percent. However on some days, e.g., the July 31-5 trip, the accuracy is not so good. As shown in the right graph of Fig. 6, the observed data is stable at 11 passengers from minute 2 to minute 6, but our estimate decreases from 10 to 6 passengers and then increases to 8 passengers at minute 6. There can be many reasons for this such as passengers starting to use their smartphone for checking e-mail, weather, etc., after minute 4.

There are many reasons why we cannot detect all passenger devices. Firstly, if a passenger disables the Wi-Fi function we will not detect that device at all. Secondly, iOS devices have two states, 'active' and 'sleep', and in a sleep state the device sends a packet only every a few minutes [10]. Even if we can capture all packets from these devices, they will not meet our criteria for classification as signals from passenger devices.

Comparison with other methods is difficult because of differing goals, assumptions or technical limitations. Mikkelsen et al. [11] use a simpler filtering mechanism with only 2 parameters, RSSI and device visibility duration. They report an accuracy of around 50%, using a minimum duration of 6 minutes which is too large to provide real-time congestion information. Systems that count passengers boarding/alighting, such as Chen et al. [4], [5], suffer from cumulative errors that can be as high as 33%. Meaningful comparisons of the accuracy for an entire bus trip are therefore impossible to make.

6. Conclusion

We developed a method to estimate the number of signals originating from devices belonging to passengers travelling on a bus, by monitoring their Wi-Fi probe request activity.

We verified a useful relationship between the signal strength and the proximity inside the bus, and that the signals are stronger from devices inside busses than from those outside. A useful range of signal strengths for filtering method parameters was determined. We also observed and recorded the actual number of passengers travelling with busses serving our university and surrounding areas, including the number of passengers getting on or getting off at each bus stop during the trip. The accuracy of our classification mechanism was 75 percent.

Our method was simple, convenient, transparent (non-participatory and non-invasive), and easily scales using very cheap commodity components. We believe that the information we are trying to obtain about seating availability is useful and valuable to passengers, potentially increasing the quality of service they will receive, and that 75% accuracy is sufficient to provide a meaningful indicator. This system could also be potentially useful to operators, to help balance loads during busy periods.

Our ultimate goal is to inform passengers how many seats are available on the next one or two arriving bus, so they can make an informed decision whether to board, to wait, or to travel by another means. Our accuracy for detecting passenger device signals is good, but could probably be improved and further work is clearly needed to convert that data into useful information about passenger loads. Since not all passengers carry a mobile device, and some passengers may even carry several devices, the correspondence with the number of passengers aboard is only approximate. A simple coefficient of proportionality can be calculated, but is likely to change significantly with at least location and time. Also likely to change are the duration and idle parameters, which were based on the time taken for our specific busses to travel between bus stops. A drawback of the method is therefore that additional calibration steps are required before the data can be used to provide reliable information. This is a good candidate for further work.

After further refinements of our techniques to improve the accuracy and better estimate the actual number of passengers, we plan to share the information obtained with passengers via an application, an information board at the bus stop, or through a website.

References

[1] Oransirikul, T., Nishide, R., Piumarta, I. and Takada, H.: Measuring Bus Passenger Load by Monitoring Wi-Fi Transmissions from Mobile Devices, *Procedia Technology*, Vol.18, pp.120–125 (online), DOI: http://dx.doi.org/10.1016/j.protcy.2014.11.023 (2014).

[2] Oransirikul, T., Nishide, R., Piumarta, I. and Takada, H.: Feasibility of Analyzing Wi-Fi Activity to Estimate Transit Passenger Population, *2016 IEEE 30th International Conference on Advanced Information Networking and Applications* (*AINA*), pp.362–369 (online), DOI: 10.1109/AINA.2016.68 (2016).

[3] Lim, W.L.H., Lum, J.T.W., Yeo, I.J.W. and Keoh, S.L.: A crowd-assisted real-time public transport information service: No more endless wait, *2016 IEEE International Conference on Pervasive Computing and Communication Workshops* (*PerCom Workshops*), pp.1–6 (online), DOI: 10.1109/PERCOMW.2016.7457088 (2016).

[4] Chen, P., Chen, H., Zhang, C. and Tang, Z.: Real-time bus passenger flow statistics scheme based on light-sensitive wireless sensor network, *2016 35th Chinese Control Conference* (*CCC*), pp.8490–8494 (online), DOI: 10.1109/ChiCC.2016.7554712 (2016).

[5] Chen, C.H., Chang, Y.C., Chen, T.Y. and Wang, D.J.: People Counting System for Getting In/Out of a Bus Based on Video Processing, *2008 8th International Conference on Intelligent Systems Design and Applications*, Vol.3, pp.565–569 (online), DOI: 10.1109/ISDA.2008.335 (2008).

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

[7] Khoeblal, R., Laohapensaeng, T. and Chaisricharoen, R.: Passenger monitoring model for easily accessible public city trams/trains, *2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology* (*ECTI-CON*), pp.1–6 (online), DOI: 10.1109/ECTICon.2015.7207129 (2015).

[8] Nishide, R. and Takada, H.: Detecting Pedestrian Flows on a Mobile Ad Hoc Network and Issues with Trends and Feasible Applications, *International Journal on Advances in Networks and Services*, Vol.6, No.1&2 (2013).

[9] Nakano, R. and Numao, M.: Congestion level estimation in a train using scan request to wireless LAN access point, *DEIM Forum 2012*, A10-1 (2012).

[10] Yamakawa, T., Ogawa, M. and Miyagawa, Y.: Congestion Degree Estimation Method for Train Cars Using Wireless LAN Control Frames, *Journal of Signal Processing*, Vol.19, No.4, pp.127–130 (online), DOI: 10.2299/jsp.19.127 (2015).

[11] Mikkelsen, L., Buchakchiev, R., Madsen, T. and Schwefel, H.P.: Public transport occupancy estimation using WLAN probing, *2016 8th International Workshop on Resilient Networks Design and Modeling* (*RNDM*), pp.302–308 (online), DOI: 10.1109/RNDM.2016.7608302 (2016).

[12] Apetroaie-Cristea, M., Johnston, S.J., Scott, M. and Cox, S.J.: Indoor Localisation System Based on Low-cost Commodity Hardware, *Proc. 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, *UbiComp '16*, pp.13–16, ACM (online), DOI: 10.1145/2968219.2971441 (2016).

[13] Nalawade, S.R. and Akshay, S.D.: Bus tracking by computing cell tower information on Raspberry Pi, *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication* (*ICGTSPICC*), pp.87–90 (online), DOI: 10.1109/ICGTSPICC.2016.7955275 (2016).

[14] Arduino, available from ⟨https://www.arduino.cc/en/Main/Products⟩.

[15] Rasperry Pi Foundation: Rasperry Pi computer, available from ⟨http://www.raspberrypi.org⟩.

[16] Premiertek: Powerlink PL-H5DN-3070 WiFi adapter, available from ⟨http://www.premiertek.net/products/networking/PL-H5DN-3070.html⟩.

[17] wireshark, available from ⟨https://www.wireshark.org/docs/wsdg_html_chunked⟩.

[18] Luciani, D.P. and Davis, A.: RSSI based range analysis of near-ground nodes in Wi-Fi crowded environments, *2013 IEEE International Conference on Technologies for Homeland Security* (*HST*), pp.693–697 (online), DOI: 10.1109/THS.2013.6699088 (2013).

[19] Society of Automotive Engineers of Japan, Inc.: Nissan Diesel Heavy Duty Non-Step Bus, available from ⟨http://www.jsae.or.jp/autotech/data_e/2-12e.html⟩.

Thongtat Oransirikul received his M.E. degree in information Scsience from Ritsumeikan University in 2015. He is currently a doctoral stundent at Ritsumeikan University since 2015.

© 2019 Information Processing Society of Japan

Ian Piumarta received his Ph.D. in Computer Science from Manchester University, UK, in 1992. From 1993–2004 he worked in Paris, France, at IRCAM, and then for a joint project between INRIA and the Computer Science laboratory of the University of Paris. From 2004 he worked at Viewpoints Research Institute in California. In 2010 he was a visiting researcher at Kyoto University, and in 2012 a JSPS visiting scholar at Ritsumeikan University, Japan. Since 2014 he is an Associate Professor in the College of Information Science and Engineering at Ritsumeikan University. His research interests include collaborative and distributed systems, software engineering tools, programming languages, and the IoT.

Hideyuki Takada received his B.E., M.E. degrees in information science from Kyoto University in 1991 and 1993 respectively. He also received his Ph.D. degree in social informatics from Kyoto University in 2001 while he worked for Mitsubishi Electric Corporation from 1993 to 2003. After he worked as a researcher at Kyoto University from 2004 to 2006, he is currently a professor at College of Information Science and Engineering, Ritsumeikan University. His research interests include distributed systems, collaboration support systems and learning support systems. He is a member of IPSJ, IEICE, JSET, IEEE Computer Society and ACM.

© 2019 Information Processing Society of Japan