UNIVERSITAT JAUME I

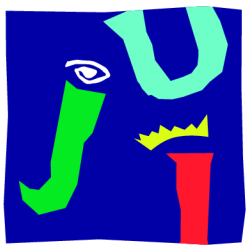
Doctoral School

BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

DOCTORAL THESIS

Castellón de la Plana, October 2023 Ing. TOMÁŠ BRAVENEC



UNIVERSITAT JAUME I

DOCTORAL SCHOOL

BRNO UNIVERSITY OF TECHNOLOGY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIO ELECTRONICS

EXPLOITING WIRELESS COMMUNICATIONS FOR LOCALIZATION: BEYOND FINGERPRINTING

DOCTORAL THESIS

AUTHOR — Ing. Tomáš Bravenec

ADVISORS — Dr. Michael Gould Carlson

Dr. Joaquín Torres-Sospedra

doc. Ing. Tomáš Frýza, Ph.D.

CASTELLÓN DE LA PLANA, October 2023

This dissertation is funded by the European Union's Horizon 2020 Research and Innovation programme under the Marie Skłodowska Curie grant agreement No. 813278 (A-WEAR: A network for dynamic WEarable Applications with pRivacy constraints, https://www.a-wear.eu/).

Abstract

The field of Location-based Services (LBS) has been steadily growing over the last decade. There are several factors contributing to this growth, including growing interest in tracking of fitness and other activities, robotics and eHealth. Each field has different requirements for technology, precision of positioning, computational resources, as well as for privacy and security. There are several available solutions in these fields however there are unresolved challenges and issues.

This dissertation evaluates currently employed privacy related measures in Indoor Positioning Systems (IPS). The main focus is on the most widespread wireless technology around us, Wi-Fi. Wi-Fi networks are ubiquitous and the majority of our devices possess the capability to use these networks. As such the question of privacy comes to mind: are the networks all around us safe, and is it possible to compromise our privacy without our knowledge?

We address non-cooperative user tracking by exploiting regular wireless communications. User Equipment (UE) uses management frames (specifically Probe Requests) of Wi-Fi for scanning the environment for Wi-Fi Access Points (AP) in the proximity. These management frames are not encrypted and contain enough information required for fingerprinting devices. That is despite the widely adopted practice of Media Access Control (MAC) address randomization.

Following presence detection, the next step is the introduction of an algorithm capable of estimating the occupancy of a room based on the passive 'sniffing' of Wi-Fi management frames. This is done by measuring the Received Signal Strength Indicator (RSSI) of the received packets from a known device close to the edges of the room. Furthermore, the measured RSSI can be used as a threshold for filtering devices that are out of range.

Previous research shows that management frames of Wi-Fi are a threat to location privacy. The threat ranges from just presence detection of users, to possibly compromising the exact location of users in case of the employment of more sniffers. Room occupancy can be determined without the need for user information, and as such, it can be useful in energy regulations of smart buildings.

The second point of this thesis is the exploration of methods to reduce computational requirements of machine learning (specifically neural networks), and positioning algorithms. This is split into two parts: reduction of memory requirements of neural networks and usage of interpolation techniques to improve fingerprinting methods in IPS.

The memory requirements of neural networks were reduced by changing the data type of weights from a single precision floating point format to just half precision. This simple change resulted in a reduction of memory requirements by 50 %. In this

case, the accuracy drop was negligible. However, reducing the data type further, without any retraining provided mixed results. Overall, the findings were that single precision is unnecessarily accurate for neural networks, and half precision is a good choice for weights, without any sacrifice in accuracy.

The second part of the optimization was focused on interpolations of RSSI based Radio Map (RM) for IPS. As means of evaluation, the time required for data collection, accuracy, and inference of position using k-Nearest Neighbors (kNN) were used. For interpolation, Gaussian Process Regression (GPR), linear interpolation, and a combination of both were used. From the accuracy point of view, the interpolation did not really increase accuracy. However, the takeaway is that the grid of 0.5 m is much too dense and actually reduces the accuracy while also increasing the inference time. However, it was clear that it is possible to drastically reduce the inference time by using RM with a single sample per Reference Position (RP), as it dramatically lowers the amount of reference samples for kNN and all of the samples are created from information contained in many samples.

Finally, during this thesis several datasets of probe requests were created, as well as data analysis scripts and packet sniffer firmware for ESP32 microcontroller. All of these are publicly available for future research, and to improve the reproducibility and replicability of the work.

The thesis provides insight into non-cooperative tracking of users, presence detection and occupancy estimation, as well as into optimizations of machine learning algorithms and indoor positioning algorithms employing RSSI fingerprinting.

Keywords: *machine learning, indoor positioning, privacy, Wi-Fi, 802.11, data analysis, presence detection, occupancy estimation, memory optimization, radio map interpolation.*

Resumen

El campo de los servicios basados en la localización (LBS por sus siglas en inglés) no ha dejado de crecer, de forma general, en la última década. Hay varios factores que han contribuido a este crecimiento. Uno de ellos es el creciente interés por el seguimiento de la actividad física (deportes), la robótica (vehículos autónomos) y el *eHealth* (la sanidad digital). Cada campo tiene necesidades diferentes en cuanto a la tecnología utilizada, la precisión del posicionamiento, los recursos informáticos y los requisitos de privacidad y seguridad. Aunque hay varias soluciones disponibles que cubren muchos de los factores anteriormente mencionados, todavía hay algunos retos y cuestiones que deben resolverse.

Esta tesis se centra en la evaluación de las medidas relacionadas con la privacidad empleadas actualmente en los Sistemas de Posicionamiento en Interiores (IPS por sus siglas en inglés). El foco principal se centra en la tecnología inalámbrica más extendida a nuestro alrededor, la red Wi-Fi. Las redes Wi-Fi están presentes a nuestro alrededor y la mayoría de nuestros dispositivos son capaces de utilizarlas. De ahí que surja la cuestión de la privacidad: ¿Son seguras las redes que nos rodean?, ¿Es posible comprometer nuestra privacidad sin que lo sepamos?

Esta tesis se centra principalmente en el rastreo no cooperativo de usuarios mediante la explotación de las comunicaciones inalámbricas habituales. Las tramas de gestión de Wi-Fi, las cuales se utilizan para escanear el entorno en busca de puntos de acceso Wi-Fi (AP por sus siglas en inglés), no están cifradas y proporcionan mucha información sobre los dispositivos presentes en el entorno. Por ello, estas tramas se pueden utilizar con la finalidad de rastrear a los usuarios en las proximidades del equipo de usuario (UE por sus siglas en inglés), a pesar de la aleatorización de las direcciones de control de acceso al medio (MAC por sus siglas en inglés), muy extendida en la actualidad.

Tras la detección de presencia, el siguiente paso consiste en proponer un algoritmo que sea capaz de estimar la ocupación de una habitación (o entorno), basado en la captura pasiva de las tramas de gestión Wi-Fi. Para ello, se mide el indicador de intensidad de la señal recibida (RSSI por sus siglas en inglés) de los paquetes recibidos de un dispositivo conocido, cerca de los limites de la habitación; además, el RSSI medido puede utilizarse como umbral para filtrar los dispositivos fuera de alcance.

De las investigaciones realizadas sobre este tema, se desprende que las tramas de gestión de Wi-Fi constituyen una clara amenaza para la privacidad. Ya sea la mera detección de presencia o, incluso, estimar la posición del usuario en caso de uso de más sensores. La ocupación de habitaciones puede realizarse sin necesidad de información del usuario y, como tal, puede ser útil en las regulaciones energéticas de

los edificios inteligentes.

El segundo punto de esta tesis es la exploración de formas de reducir los requisitos computacionales del aprendizaje automático (en concreto, las redes neuronales) y los algoritmos de posicionamiento. Esto se divide en dos partes, la reducción de los requisitos de memoria de las redes neuronales y el uso de técnicas de interpolación para mejorar los métodos basados en huellas Wi-Fi en IPS.

Los requisitos de memoria de las redes neuronales se redujeron cambiando el tipo de datos de los pesos de un formato de coma flotante de precisión única a sólo media precisión, este simple cambio dio lugar a una reducción de los requisitos de memoria del 50 %. En este caso, el descenso de la precisión fue insignificante. Sin embargo, reducir aún más el tipo de datos, sin ningún reentrenamiento, proporcionó resultados desiguales. Y para mejorarlos, sería necesario el reentrenamiento para actualizar los pesos para un nuevo tipo de datos. En general, las conclusiones fueron que la precisión simple es innecesaria para las redes neuronales y que la media precisión es una buena opción para los pesos, sin sacrificar la precisión.

La segunda parte de las optimizaciones se centró en las interpolaciones del Mapa de Radio (RM) basado en RSSI para IPS. Esto se evaluó con métricas basadas en: tiempo necesario para la recogida de datos, precisión e inferencia de la posición utilizando k-Nearest Neighbors (kNN). Para la interpolación se utilizó regresión de proceso Gaussiano (GPR por sus siglas en inglés) y interpolación lineal, así como la combinación de ambas. Desde el punto de vista de la precisión, la interpolación no aumentó realmente la precisión, pero la conclusión es que una cuadrícula de 0,5 metros es demasiado densa. Aunque reduce el error de posicionamiento, hace que aumente el tiempo de inferencia. Sin embargo, quedó claro que es posible reducir drásticamente el tiempo de inferencia utilizando RM con una sola muestra por posición de referencia (RP por sus siglas en inglés), ya que reduce drásticamente la cantidad de muestras de referencia para kNN y el re-muestreo se crea a partir de la información contenida en un conjunto con un número elevado de muestras.

Por último, pero no menos importante, durante esta tesis se crearon varios conjuntos de datos, así como scripts de análisis de datos y firmware para los sensores (*sniffers* de tramas de gestión) para el microcontrolador ESP32. Todo ello está disponible públicamente para que cualquiera pueda continuar la investigación por su cuenta, así como para garantizar la reproducibilidad y replicabilidad del trabajo.

La tesis profundiza en el seguimiento no cooperativo de usuarios, la detección de presencia y la estimación de ocupación, así como en la optimización de algoritmos de aprendizaje automático y algoritmos de posicionamiento en interiores que emplean RSSI *fingerprinting*.

Palabras clave: *aprendizaje automático, posicionamiento en interiores, privacidad, Wi-Fi, 802.11, análisis de datos, detección de presencia, estimación de ocupación, optimización de memoria, interpolación de mapas de radio.*

BRAVENEC, Tomáš. *Exploiting Wireless Communications for Localization: Beyond Fingerprinting*. Castellón de la Plana: Universitat Jaume I, Doctoral School, and Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Radio Electronics, September 2023, 156 p. Doctoral thesis. Advised by Dr. Michael Gould, Dr. Joaquín Torres-Sospedra and doc. Ing. Tomáš Frýza, Ph.D.

Author's Declaration

Author:	Ing. Tomáš Bravenec
Author's ID:	173619
Paper type:	Doctoral thesis
Academic year:	2022/23
Topic:	Exploiting Wireless Communications for Localization: Beyond Fingerprinting

I declare that I have written this thesis independently, under the guidance of the advisors and using exclusively the technical references and other sources of information cited in the paper and listed in the comprehensive bibliography at the end of the paper.

As the author, I furthermore declare that, with respect to the creation of this thesis, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll. of the Czech Republic, Section 2, Head VI, Part 4.

Castellón de la Plana ....

author's signature*

*The author signs only in the printed version.

ACKNOWLEDGEMENT

This dissertation thesis and the research carried out were done at Universitat Jaume I in Castellón de la Plana, Spain, and at Brno University of Technology, Czech Republic, between 2019 and 2023. It has been a life-changing experience during which I got to meet many excellent researchers, gain new friendships and gain a lot of new experiences. Here I would like to mention everyone that had an impact on me during the time of my doctoral studies.

Starting with my supervisors, I would like to sincerely thank *Michael Gould*, *Joaquín Torres-Sospedra (Ximo)* and *Tomáš Frýza* for their support, professional guidance, suggestions, and all the help I have received at every step in the creation of this thesis and the prior publications.

Next, I would like to thank everyone involved in the A-WEAR project, especially *Alex Ometov* and *Simona Lohan* for all the guidance provided throughout my time in A-WEAR. I would also like to mention everyone in GEOTEC, for all the support and assistance I have received during my time in Spain.

I also have to thank *my mom Vladimíra and dad Vlastimil*, brother *Ondřej*, *grandmom Anna with uncle Josef*, and *grandmom Marie with grandad Václav* as well as *everyone else in my family* for their support throughout the whole Ph.D. journey, especially for the unwavering support and patience during my sudden relocation to Spain. I know it wasn't easy to see me uproot my life and move to the other side of the continent.

There is also a special thank you necessary to *my girlfriend Nadia*, for motivating me and unconditionally supporting me in anything I do. Thank you for being in my life, I can no longer imagine mine without you!

Finally, I would like to thank *all of my friends* both in the Czech Republic and in Spain for all their help, support, and friendship.

Thank you all!

Contents

List of Figures

List of Tables

1 Introduction

This chapter introduces the background information and motivations for this dissertation and the A-WEAR *project. It also contains the definition of objectives and research questions, as well as the outline of the thesis.*

1.1 Motivation

This dissertation is one of the 15 doctoral theses within the four years long H2020 Marie Skłodowska-Curie *Innovative Training Network* (ITN)/*European Joint Doctorate* (EJD) called *A network for dynamic WEarable Applications with pRivacy constraints* (A-WEAR). The project connects five universities: *Tampere University* (TAU), Finland; *Brno University of Technology* (BUT), Czech Republic; *Universita Mediterranea di Reggio Calabria* (URC), Italy; *University "Politehnica" of Bucharest* (UPB), Romania; and *Universitat Jaume I de Castellon* (UJI), Spain, to provide post-graduate education, supervision, and training for the 15 early stage researchers. All topics of the project are closely related to wearable applications, be it regarding privacy, security, localization, communication, or health applications. Apart from working in academia, every *Early Stage Researcher* (ESR) has a mandatory industrial secondment in a company with a similar focus as their thesis and academic secondment in their secondary university.

Together, all ESRs worked on an extensive survey regarding wearable devices, including the history of wearables, current state-of-the-art, and challenges the wearable industry faces now and will in the future [1].

The topics of individual ESRs range significantly, as listed here. Waleed Bin Qaim focused on energy-efficient edge computing [2], [3].

Lucie Klus explored ways of crowd sourcing wearable data collection [4], data compression and processing [5]–[7] and use of *Machine Learning* (ML) for indoor positioning [8]–[10].

Another critical research topic related to wearable devices is wireless communication. In the last couple of years, the *Fifth-Generation* (5G) mmWave applications came into the spotlight of the research community and, as part of the A-WEAR project, were covered by Nadezda Chukhno. Her research focused mainly on modeling optimal and nearly optimal fast multicast transmission strategies in directional 5G mmWave networks [11]–[14]. She also explored *Device-to-Device* (D2D) technology to improve cellular positioning accuracy by utilizing cooperative positioning methods [15] and D2D-assisted multicast scheduling [16] to cope with the high attenuation at mmWave bands. In a similar way to Nadezda, the research of Asad Ali focused on communications in 5G mmWave networks [17]–[20].

Olga Chukhno researched the new advanced *Internet of Things* (IoT) application, where computing-, storage-, and battery-constrained IoT devices are augmented via *Digital Twins* (DT) hosted on edge servers [21], [22]. Moreover, she explored the effects of *eXtended Reality* (XR) user motion from the communication and computing perspectives [23], [24]. She also analyzed the directional properties of mmWave systems and their impact on the overall system performance [25], [26].

A prominent topic in the A-WEAR project is indoor positioning, which is also a focus of Darwin Quezada. Specifically, the cloud platform for positioning and localization [27], [28] and algorithms improving the accuracy of *Indoor Positioning System* (IPS) [29]–[32]. Similarly to Darwin, Pável Pascacio worked with IPS but focused on infrastructure-less collaborative approaches to indoor positioning [33]–[36].

To further expand the list of technologies used for research, the main topic of interest for Laura Flueratoru is the use of *Ultra-Wideband* (UWB) in wearable applications for localization [37]–[40].

On the other hand, Asma Channa used wearables to monitor health disorders like Parkinson's disease or neurocognitive disorder [41]–[44], COVID-19 [45]–[47]. Justyna Skibińska, like Asma, focused on health-related applications for wearables [48]–[51].

An important use of wearable devices is in industrial applications to ensure the safety of workers, and in the scope of A-WEAR, it is covered by Ekaterina Svertoka [52], [53]. She also explored the usefulness of LoRaWAN technology for localization purposes [54], [55]. A different look at industrial applications is provided by Salwa Saafi, whose research was aimed at low-latency communications in industrial applications [56]–[58]. Additionally her research also turned to 5G communications [59] and maritime networks [60].

To further change the focus, the research of Viktoriia Shubina was aimed at privacy-aware approaches for IoT and wearable devices [61], [62], as well as on contact tracing solutions for COVID-19 control [63]–[66]. Hand in hand with privacy goes security, which is the leading research interest of Raúl Casanova and his work on privacy-preserving and enhancing technologies in wearable applications [67]–[69].

This thesis follows a similar focus as research of other ESRs, with wearable devices at the center of attention. In this case, the central question is privacy in indoor positioning and localization. The problem of indoor positioning has grown rapidly over the last couple of years. The radio signal characteristics and the ways radio signal propagates throughout the environment is one of the main reasons wireless networks are popular in the indoor positioning and indoor navigation community [70]–[72].

Because of the nature of wireless communications, moving through the air, the data transfers can be exploited by a third party. That makes it easy for adversaries to capture the packets and this raises privacy concerns. As such this thesis explores the possibilities of passive capture of management frames of the Wi-Fi protocol to

find possibilities for user tracking without the knowledge of the users.

Furthermore, this thesis dives into the very important field of algorithm optimizations. Since memory is limited in mobile and IoT devices, the ML optimizations in this thesis focus on memory. Also, specific *Radio Map* (RM) improvements through interpolations are explored.

1.1.1 Wearable Technologies

Wearable technology has become increasingly popular over the last several years, and the market for wearable technology is growing rapidly. As the name suggests, wearable devices can be worn by humans. There are three main categories of how close the wearables are to the human body [1]: *near-body* (for example smartphones), *on-body* (smart watches, smart rings, fitness trackers, earbuds, etc.), and *in-body* (implants).

Apart from the classification based on placement, wearable functions can be split into several sections: *health monitoring* (measuring heart rate, electrocardiograms, blood pressure, oxygen saturation levels, etc.), *fitness tracking* (counting steps, traveled distance and in case of outdoor activities location information or burned calories), and *lifestyle tracking* (sleep, stress, water, and calories income).

In general, wearable electronics are designed to provide a wide range of functions from providing us with useful information, tracking health and fitness metrics all the way to vital signs monitoring, and in some cases, even providing remote health care services. As such, the sensors on wearable electronics have the front seat into the privacy of the user [73]. The data captured by all of the sensors and the transfer between devices are protected by encryption. However, radio signals are still a place of weakness. This is mainly due to the employment of wireless technologies, as these can be used for breaching the location privacy of users [74], [75].

1.1.2 Technologies for Indoor Positioning Systems

There are many wireless technologies with the possibility to be employed in IPS [76]. Two of the more common technologies, mainly due to their widespread adoption in smartphones, are Bluetooth [77]–[79] and Wi-Fi [80]–[82]. Less common are technologies like UWB [38], [83], which due to its higher cost [84], is yet to gain widespread adoption. Other technologies used in indoor positioning are ZigBee [85], [86], visible light [87], millimetre wave radar [88] and others using computer vision [89] or dead reckoning [83], [90]. Computer vision and dead reckoning are two technologies capable of working without infrastructure changes, as neither requires any beacons. However, both rely on the environment itself. Computer vision is used to analyze the environment using the images captured by the phone camera [91] and dead

reckoning [92] using incremental location estimation from a known *Reference Point* (RP). Each technology possesses its own advantages and disadvantages [93].

The majority of *User Equipment* (UE) manufactured throughout the last decade includes a Wi-Fi and Bluetooth interface. With the widespread adoption of Wi-Fi technology as the main protocol for *Wireless Local Area Network* (WLAN), Wi-Fi based IPS are very easy to create. Unlike Wi-Fi, employing *Bluetooth Low Energy* (BLE) beacons depends on the placement of new anchors. Even then, the majority of UE devices possess BLE interface, and as such, it is a very viable approach to indoor positioning.

1.1.3 Privacy in Indoor Positioning Systems

Since the majority of UE contains wireless interfaces like Wi-Fi and Bluetooth, the question arises on the possibilities of privacy breaches using common wireless networks. This is especially the case with management frames of Wi-Fi, which do not use any encryption. There are several ways this can be an issue.

- In case of probe request frames without anonymized *Media Access Control* (MAC) address, the address becomes a unique identifier of the device, which can then be tracked.
- The *Preferred Network List* (PNL) can be used for fingerprinting, and even to identify the places the user previously connected to Wi-Fi using a public database of *Access Points* (APs) [94]. This can reveal the work and residence locations of the user.
- Some probe request fields contain information that can directly identify the owner of the UE. *Wi-Fi Protected Setup* (WPS) field contains a field for the device name, which could be set to something like "*Julia's iPhone*" thus directly revealing the name of the user.

This breach of private user information is only one part of the puzzle. Regarding the location of the user, presence detection using Wi-Fi management frames is only the first step. Detecting the presence of users carrying a Wi-Fi enabled device is relatively simple because the UE keeps sending probe requests in order to stay connected to the known WLANs in proximity. More complicated than presence detection is actual localization. A single AP is enough for monitoring presence detection. By employing at least 3 cooperating APs monitoring the radio environment, a user's location can be compromised.

Unlike *Global Navigation Satellite Systems* (GNSS), which are covering the entire surface of the earth, the IPS needs to be adapted to individual buildings. This is due to the physical differences between buildings, be it because of a different floor plan, variances in the interior furnishing, or placement of IPS anchors. Regardless of

technology, the variety of indoor spaces requires analysis of the environment prior to deployment of an IPS.

1.2 Objectives

In this section, the thesis Research Objectives (ROx) are defined based on the research gaps found in Chapter 1.1.

- **RO1. Analyze the possibilities of passive presence detection and positioning in indoor environments.**
  The Wi-Fi communication protocol is the most widespread WLAN technology. People use it pretty much daily to connect to the internet at home, at work, or in public spaces, in part to save mobile data usage in the package provided by the cellular network carrier. Another factor is the use of Wi-Fi by our smartphones, not only for communication but also for coarse localization. This combined with the management frames opens the door to a possible breach of user privacy. The questions then are: *Do our devices leak data and if so, how? Are there ways adversaries could exploit this leak to track us without our knowledge?*
- **RO2. Balance the achieved accuracy and necessary compute requirements of IPS and ML algorithms.**
  Even though the computational capabilities of UEs grows every year, the complexity of algorithms increases too. However, it should not need to do the same. This would result in a reduction in the processing time. By optimizing the algorithms we can reduce the processing time and subsequently have more time available either for other tasks or for saving energy by entering a low-power mode during idle. This is especially a good thing in embedded and wearable applications, in which battery life is an important factor. The question is: *What are the ways to preserve the accuracy of machine learning algorithms, while reducing the hardware requirements?*

1.3 Contributions

The publications included in the creation of this thesis contain the contributions that can be split according to the chapters as follows:

- **Chapter 2:** Reproducibility components, Software and Datasets
  1. **proposed** a new k selection equation for grid-based IPS using *k-Nearest Neighbors* (kNN) depending on the density of a grid and amount of samples per RP;

2. **probe request sniffer** firmware for ESP32, with the capability to capture the entire packets, filter MAC addresses, collect *Channel State Information* (CSI), and store the data on connected SD card, which allows anyone to easily create new datasets, or work as an entry into Wi-Fi packet capture applications;
3. **new datasets** of 802.11 probe requests, captured over the whole duration of an international scientific conference, and month-long period in the office. These up-to-date datasets will be useful to the community in further research into privacy issues with Wi-Fi management frames;
4. **new dataset** for network-side indoor positioning algorithms based on Wi-Fi and *Received Signal Strength Indicator* (RSSI) measurements, that can be used by the community in continuous research into indoor positioning using Wi-Fi and optimizations of RMs.

- **Chapter 3:** Presence detection
  1. **simple** presence detection exploiting Wi-Fi communication based on ESP32 microcontroller, to make the packet sniffing and presence detection affordable and easy, while raising the awareness for the vulnerabilities of Wi-Fi;
  2. **presence tracking** algorithm based on passive collection of Wi-Fi management frames, to increase the awareness on the ease of tracking Wi-Fi devices;
  3. **low complexity** algorithm for room occupancy tracking exploiting Wi-Fi, in order to optimize smart building energy management.
- **Chapter 4:** Optimizations in ML and IPS
  1. **reducing** the high memory requirements on convolutional neural networks, to allow inference on hardware with fewer available resources;
  2. **improving** computational requirements of IPS based on fingerprinting of RSSI by approximating measured RMs.

1.4 Thesis Outline

The thesis comprises of privacy aspects of indoor localization. The structure of the thesis is split into five chapters briefly described as:

- **Chapter 1** contains the motivation for the research done in this thesis as well as the contributions to the scientific field. Additionally, this chapter also includes the list of publications, datasets, and research stays.
- **Chapter 2** describes the algorithms and methods used in following chapters. It also contains descriptions of software and datasets published in public repositories.

- **Chapter 3** explores the ability to exploit management frames of the Wi-Fi protocol for fingerprinting users and analyzing the presence detection of users.
- **Chapter 4** contains research focused on improving edge computing capabilities as well as the accuracy of indoor positioning algorithms by pre-processing RM with interpolation algorithms.
- **Chapter 5** has a summary of the results and achieved contributions. It also includes a discussion about the future research directions.

Finally, following all the main chapters is the list of references.

2 Methodology

This chapter introduces common components of this thesis.

First, in Section 2.1, the context and information are given regarding the problems, and how can they be solved.

In Section 2.2, the common wireless technologies used for data collection are described.

The common algorithms and techniques used throughout the thesis are described in Section 2.3.

Following the wireless technologies, Section 2.4 focuses on introduction of indoor positioning techniques.

Section 2.5 contains the created software used in several publications as well as the description of the collected datasets.

2.1 Background

As mentioned before, this thesis focuses on identifying vulnerabilities in the Wi-Fi protocol, with the main focus on the probe request frames, that lack encryption. The principle Wi-Fi characteristics are presented in the following Section 2.2, which is focused on the introduction and technical background on the wireless protocols used in the following chapters.

In some of the works described in the following chapters, deterministic algorithms were proposed, and, as such, these are described at the time of their introduction. However, more common algorithms and techniques are described later in Sections 2.3 and 2.4, respectively. Section 2.3 contains the description of ML algorithms, which have the ability to find hidden patterns and characteristics in data, that would otherwise not be possible to detect using traditional approaches. This makes ML very useful, and some of the described ML algorithms were used in the next chapters of this thesis. Section 2.4 describes the techniques that can be used for indoor positioning.

Finally, the software and data used throughout the following chapters are described in the last Section of this Chapter. Since these components, be it software or data, are all publicly available they are contributions to open science, reproducibility, and replicability of the work done in this thesis.

2.2 Wireless Technologies

There are 3 different network types used by wearable electronics. These are divided by the area they can cover, and subsequently, the technology that is suitable for such

networks. The largest scale, with region or nationwide coverage, has category *Wireless Wide Area Network* (WWAN). It is usually covered by cellular technologies like 2G, 3G, 4G (LTE), or 5G networks. On a much smaller scale are WLANs. WLAN usually provides coverage for building floors, houses, or rooms. The most common technology for such networks is an IEEE standard 802.11, known as Wi-Fi. The smallest scale, which covers near proximity of a person is *Personal Area Network* (PAN). The most common for such networks is the IEEE standard 802.15, more commonly known as Bluetooth Low Energy (BLE).

For the research into indoor positioning and privacy issues, the WLAN and PAN technologies are most interesting. As such, the technologies used in this thesis are briefly described in the following sections.

2.2.1 Wi-Fi

Over the years and with the spread of wireless communications, Wi-Fi [95] became the most used technology for local wireless networks. The main reasons for the spread of Wi-Fi are the simple setup, reasonable price, and adequate transfer speeds (with the latest iterations approaching that of physical wired connections). The range of up to 100 m [96], [97] is another reason for the high rate of adoption as it usually covers the entire living area. This depends in part on the frequency used, as the most common Wi-Fi variants 802.11b/g/n [98] work in the 2.4 GHz band. With more modern variants like 802.11ac [99] using the 5 GHz and the latest 802.11ax [100] and in progress 802.11be [101] using both 2.4 GHz and 5 GHz as well as 6 GHz frequency band. With more standards amendments being in the works [102]. Currently adopted and planned Wi-Fi standards, with their frequency bands and maximum theoretical bit rate are in Table 2.1. There is also high throughput 802.11ad working in 60 GHz frequency band but with a much lower range. Since higher frequency means shorter wavelength, using higher frequency bands means the signal is more prone to interference and has a lower range than a signal using lower frequency at the same power, however, the achievable transfer speeds are higher [97].

From the frequency point of view, each connection does not occupy the entire band, but only a small part of it. Each frequency band is split into several channels, to reduce interference between neighboring networks [103]. In the 2.4 GHz band, for 802.11 b/g/n are defined 14 channels (Channels 12 and 13 are avoided in the USA, and channel 14 is only used in Japan for 802.11b) with a bandwidth of 20 MHz. The center frequency offset between channels is 5 MHz. This results in the channels overlapping each other. The visualization of the 14 possible channels and highlighted possible use of channels without overlaps can be seen in Fig. 2.1. For 802.11n and newer, the bandwidth can be increased and in this band, it is possible to

Tab. 2.1: Comparison of Wi-Fi standards over the years.

IEEE Standard	Year	Frequency [GHz]	Maximum bit rate [Mbit/s]
802.11	1997	2.4	1-2
802.11b	1999	2.4	1-11
802.11a	1999	5	6-54
802.11g	2003	2.4	6-54
802.11n	2008	2.4/5	72-600
802.11ac	2014	5	433-6933
802.11ax	2019	2.4/5	574-9608
802.11be	*2024*	2.4/5/6	1376-46120

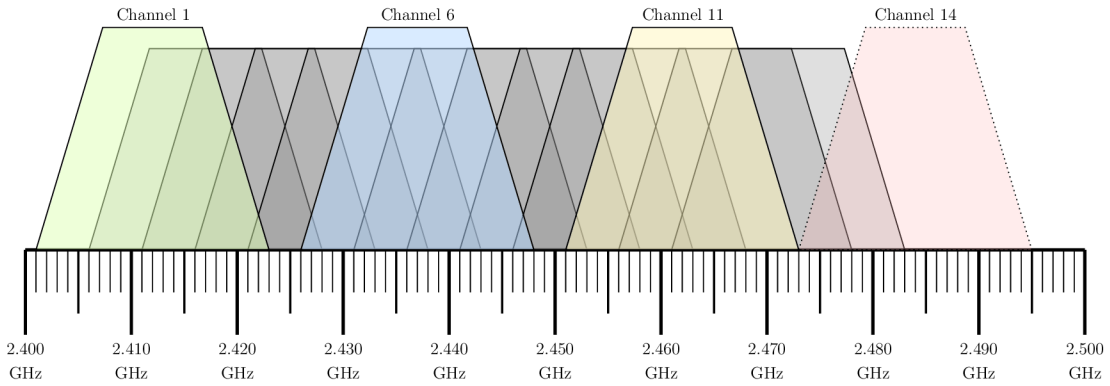


Fig. 2.1: Visualisation of the 14 defined 802.11g/n channels with bandwidth 20 MHz in 2.4 GHz band with highlighting of 3 not overlapping channels and Japan-specific channel.

use a bandwidth of 40 MHz as well. In 2.4 GHz band, this limits the number of non-overlapping channels to only 2.

The 5 GHz band used by the Wi-Fi standard 802.11ac or 802.11ax, is much wider and networks using this band can use the channel bandwidth from 20 MHz to 160 MHz. Unlike the lower frequency band of Wi-Fi, 5 GHz band has also other uses than just user electronics. Due to this, there are more limitations on this frequency band, most of the time affecting the outdoor use of these channels. An example of this are channels between 116 and 132, which are used by weather radars [104]. The distribution of the channels in the 5 GHz band along with the channel numbers is visualized in Fig. 2.2.

As a part of any communication network, Wi-Fi works with several frame types. There are 3 main categories: data frames for transfer data, management, and control

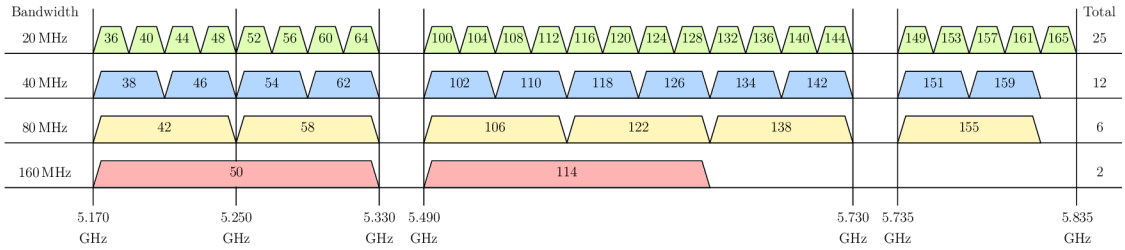


Fig. 2.2: Visualisation of the 802.11ac channel distribution for each of the possible bandwidths.

frames for managing the connection [95], [105]. Each of the frames has a role, but for the purposes of the thesis, only a few of the management frames are important. The management frames are:

- Probe Request,
- Probe Response,
- Beacon,
- Association Request,
- Association Response,
- Re-association Request,
- Re-association Response,
- Disassociation,
- Authentication,
- De-authentication.

The only three important frames for the research done in this thesis, are the Beacons, Probe Requests, and Probe Response frames. All of these frames, even if the function is different, share a similarity, that is they can be in some way used for positioning.

Beacon Frames

The Beacon frames are transmitted by the APs to inform the UE in the proximity of its existence. These frames are transmitted at regular intervals and contain the *Service Set Identifier* (SSID), supported data transfer speeds, and security settings.

Probe Request Frames

For the purposes of this thesis, one of the most important frame types of the Wi-Fi protocol is the Probe Request [95]. These frames are used for detection of nearby Wi-Fi APs. This detection is mainly used for the discovery of nearby wireless networks to which the user can connect. But the device transmits these frames periodically to discover a known network to which it could connect without any user

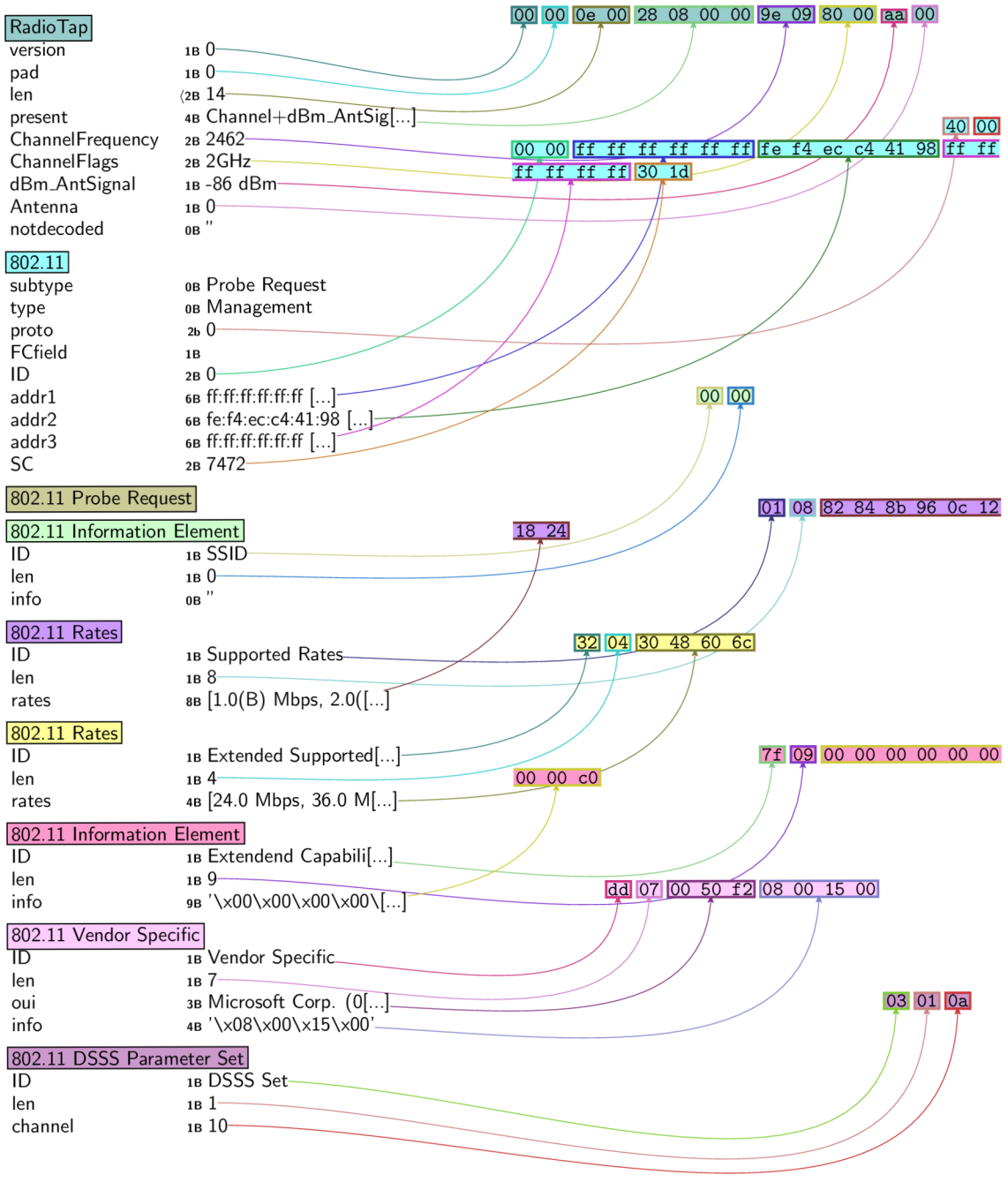


Fig. 2.3: Probe request structure including RadioTap information generated by Scapy.

input to preserve mobile data. The device also sends out the probe request frames in localization scenarios to discover nearby access points. The discovered access points can then be matched to an internal database of the system developer [106] or publicly available database (like WIGGLE [94]) and provide an approximate location, without using GNSS solutions. The structure of a simple probe request, generated by Scapy is in Fig. 2.3.

Probe Response Frames

Probe Responses [95] are, as the name suggests, response frames transmitted by the access points after receiving a probe request. The probe response frames are not broadcast for every device listening like probe requests but instead targeted at the device sending the request. The data contained in the probe response frame contain all the information necessary to establish a connection between the device and the access point. Some of the information commonly included in probe response frames are the network MAC address, SSID, transmission speeds the access point is capable of, and others.

All in all, probe responses do not pose a privacy issue, unless they are named after personal things since they do not change the location they are placed in very often. If the access point is also named correctly, it is also possible to avoid inclusion in the localization databases. To opt-out of participation in the Google database, the SSID has to be renamed with a suffix *_nomap* followed up by launching Google Maps for a quick update in accordance with [107].

For the purposes of positioning, probe responses can be used in the active scenario, where the UE gathers the signals from nearby APs, and calculates its own position. For the same purpose, beacon frames can be used too, but probe responses can provide the data required for positioning on demand of the UE.

2.2.2 Bluetooth

There were several iterations of Bluetooth over the years [108]. The majority of modern devices support Bluetooth version 4 and newer [109], and since the addition of Low Energy consumption support in version 4, these versions are commonly known as Bluetooth Low Energy (BLE) and older versions are grouped under the Bluetooth Classic name. Most often, BLE is used for networks of type PAN. That means connecting accessories in close proximity. Be it connections between wearables like mobile phones, smartwatches, and earphones or *Human Interface Device* (HID) like wireless mice and keyboards for control of computers, remote controls for televisions, and other short-range applications.

- **Bluetooth Classic** The communication in older versions of Bluetooth [110] worked only in *Point-to-Point* (PtP) mode. The main style of communication was Master-Slave configuration with one device initiating the connection and controlling the communications. It works in the same 2.4 GHz band like Wi-Fi, but instead of using *Direct Sequence Spread Spectrum* (DSSS) or *Orthogonal Frequency Division Multiplexing* (OFDM) modulations, Bluetooth employs *Frequency Hopping Spread Spectrum* (FHSS) and much lower transmitting power. Before the BLE versions, it used to employ 79 channels with a width

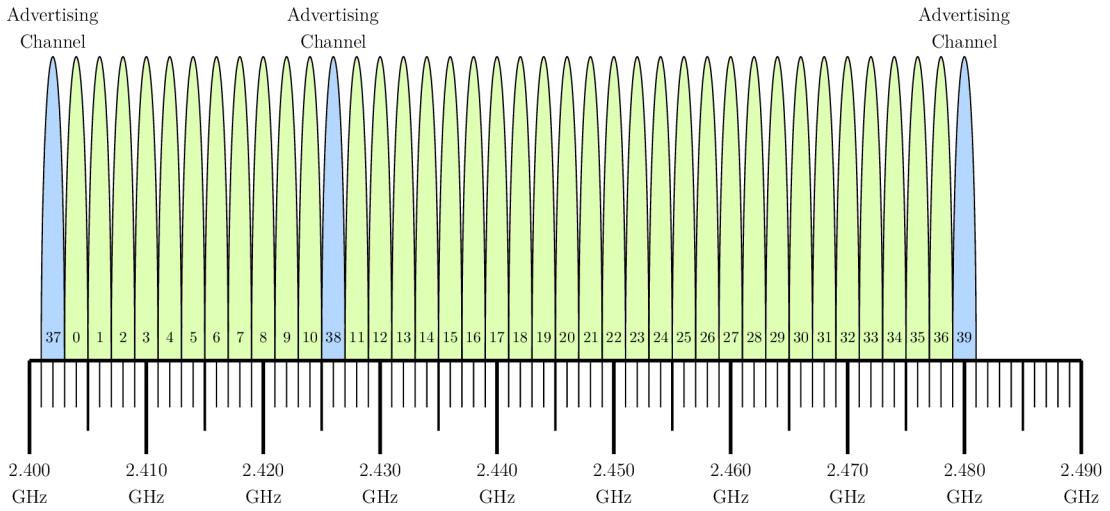


Fig. 2.4: Visualisation of the 40 BLE channels with highlighting of 3 advertising channels.

of 1 MHz. When the devices are connected, the use of FHSS means that the communication switches between channels based on a pre-determined pattern.

- **Bluetooth Low Energy** Modern versions of Bluetooth provide much higher capabilities [111] than the Bluetooth Classic. The connection between devices is now not limited to the PtP mode and supports also broadcasting through beacons and mesh networks [111], [112]. With the addition of *Angle of Departure* (AoD) and *Angle of Arrival* (AoA) [113], Bluetooth devices are very interesting technology for indoor positioning as well. Another change from Bluetooth Classic is the use of the radio environment. Instead of 79 channels, BLE uses only 40, however, the width of each channel doubled. Channels 0 to 36 are used for data transfers and the last 3 channels 37, 38, and 39 are for advertising frames. All communications begin in one of the advertising channels. The visualization of BLE channel distribution is in Fig. 2.4.

Similar to Wi-Fi, BLE relies on several frame types to control the data transfers and connection between devices. The main frames are:

- Advertising frames,
- Page frames,
- Connection request,
- Asynchronous Connection-Less data frames,
- Synchronous Connection Oriented frames,
- Control frames,
- Beacon frames.

Due to the FHSS, it is difficult to monitor all Bluetooth communications at once. For indoor positioning, the most important are the advertising frames.

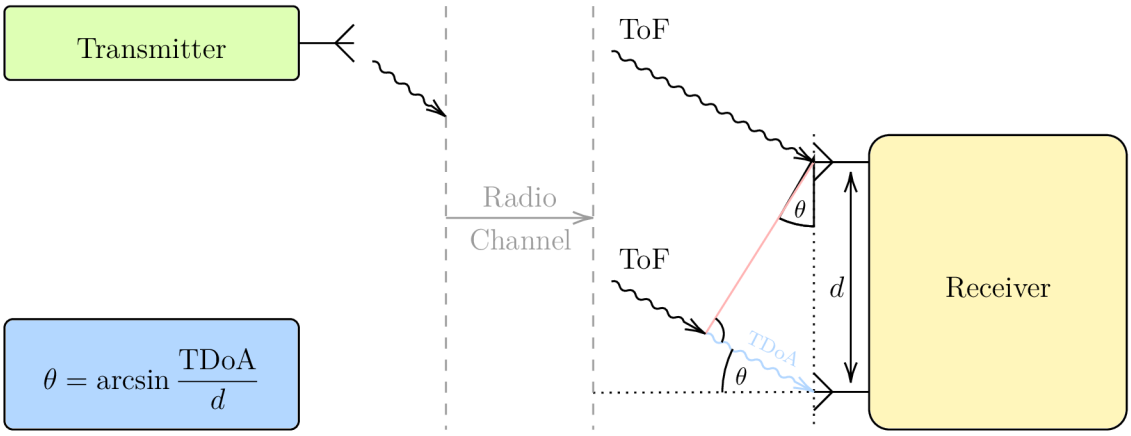


Fig. 2.5: Calculation of AoA from TDoA and distance between 2 receiving antennas.

Advertisement Frames

There are several variants of advertisement frames, connectable in case the device is open to creating connections and non-connectable, scannable in case the device receiving the advertisement packet wants more information and non-scannable, directed in case it is targeting a specific device or undirected to inform every Bluetooth device about its presence. Advertisement packets most often consist of the device name, transmit power and service *Universal Unique Identifier* (UUID), and additional information specific to the beacon. The advertising can be periodic, and the interval can be adjusted from 20 ms to 10.24 s, with a step of 625 µs.

In general, for indoor positioning applications, the connectable variant is not very helpful, as the user usually wants to move to a different location, which means the UE disconnects from the beacon device when out of range. Using scannable or non-scannable advertising packets depends on the type of positioning system. In case it operates only with RSSI fingerprinting, the non-scannable advertising frames are enough. However, in case there is a need for localization frames in order to get the AoD and AoA, the scannable frames are necessary to ask for such frames. The main difference in localization frames is in the way these are transmitted or received, depending on the approach. In the case of AoA, the transmitting side only uses a single antenna [114], while the receiver uses multiple antennas. The signal travels between the transmitter and the receiving antenna for a *Time of Flight* (ToF). There is a difference in the phase of the signal at each antenna of the receiver, depending on the direction from which the signal is transmitted. This phase difference can be used to get *Time Difference of Arrival* (TDoA). TDoA is then used along the distance between antennas of the receiver to calculate the AoA, as is visualized in Fig. 2.5. To get AoD, the opposite approach is used, with the transmitter having multiple antennas and the receiver using only one.

2.3 Algorithms

In the upcoming chapters, several common *Machine Learning* (ML) algorithms are used. These algorithms are described in the following subsections. In the process of writing this thesis, machine learning was used in addition to traditional algorithms. The reason for using machine learning is the ability of the created models to learn patterns and relationships between data, without being explicitly programmed to do so. This is especially useful as patterns or insights may be very difficult or downright impossible to reveal using traditional deterministic algorithms. In the context of indoor positioning and as such this thesis, this is a very valuable feature. For example in fingerprinting (explained in Section 2.4.3) ML algorithms can find patterns between several fluctuating RSSI values from different APs to infer user location. Due to these fluctuations, deterministic algorithms are unsuitable, as the underlying patterns might not be clearly distinguishable.

2.3.1 k-Nearest Neighbors

Algorithm kNN is a supervised machine learning algorithm, depending on labeled data. It can be used for both regression and classification problems. As the name suggests, the model searches for k reference samples closest to the value used for inference because kNN expects values with similar labels will be close together. The metric for setting the distance from each neighbor needs to be selected as well. The most common is the Minkowski metric defined for two points (X, Y) in n-dimensional space by the equation:

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}, \tag{2.1}$$

where the choice of the parameter $p = 1$ leads to Manhattan distance and $p = 2$ results in Euclidean distance. However, these are not the only possible metrics and there are many other options [115]. Since, in most cases, the indoor positioning systems contain missing values for APs out of range, Euclidean distance might not be the best to use [116].

For classification problems, the kNN searches for k closest samples from the reference samples and returns the most common label among them. Regression works in a similar way, but instead of returning the most common label, the returned value is the average between the closest samples. The example of kNN with data split into 2 classes is in Fig. 2.6.

One of the differences between kNN and *Artificial Neural Network* (ANN) (described further in Section 2.3.4) is the lack of training time. The only hyper-parameter needed to evaluate ahead of using kNN is the value of k. There are issues with

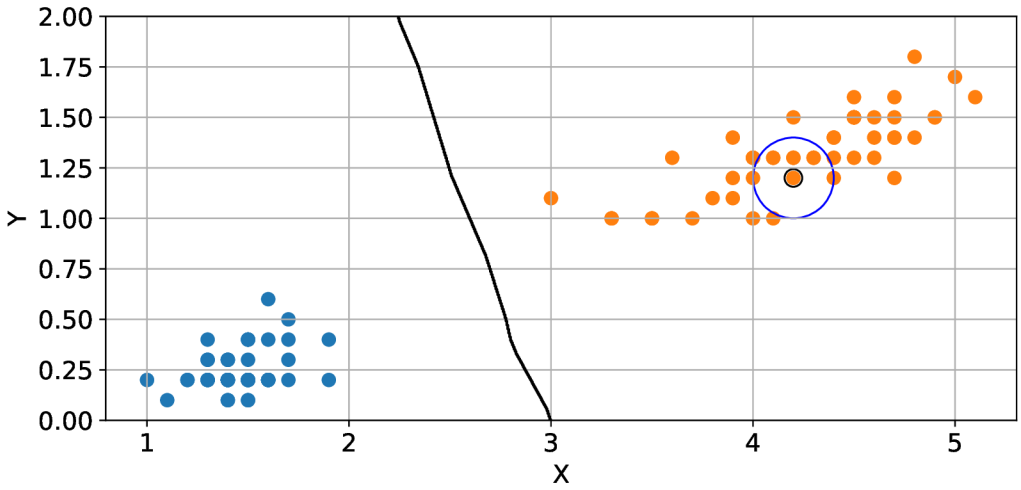


Fig. 2.6: Visualisation of kNN with $k = 5$.

k being both too small and too large. Optimally the k should reflect the data. Depending on the amount of data, different data sets and conditions will benefit from various values of k. Because of this, one way to impartially select k, is by using an equation [117]:

$$k = \left[\sqrt{n}\right], \tag{2.2}$$

where $[\cdot]$ represents the standard rounding function and n is the number of reference samples of kNN. Generally, a higher value of k provides more stable results unless too high k was selected, at which point the error starts to grow again. Smaller values, on the other hand, better represent variations in data, but using too small k results in the kNN being more affected by local changes. The special case is when $k = 1$, in which case the kNN remembers the entire dataset and searches for the closest reference sample to the evaluated value.

For indoor positioning applications, the equation 2.2 is not very viable, as the k is not based on the grid and the amount of data available per reference point. As such, depending on the size of the space or building, all RP in a room can contribute to one location. Because of this, an equation in [118] that solves this issue and selects k depending on the grid density and the number of samples available per RP was formulated. The equation is:

$$k = \left[\left(\frac{2g}{d} + 1\right)^2 \sqrt{2n}\right], \tag{2.3}$$

where $[\cdot]$ corresponds to the standard rounding function, g equals to the displacement around the central point, d presents the distance between neighboring RPs, and n is the number of samples per RP. The equation is also constrained by the fact that g must be a multiple of d.

The absence of training time also means that the kNN does not learn a generalized function out of the training dataset, but in a way has memorized the entire dataset, as the training dataset is the machine learning model. In the case of large datasets, the kNN model can have a large memory footprint. Memory is not the only issue while using large datasets, as the computational complexity increases with the number of reference samples.

Advantages: kNN is a simple algorithm to understand and implement. With well-chosen values of k, the algorithm is robust to noisy data, which is due to the majority voting of neighboring samples. kNN is also a lazy learning algorithm, which means it does not need training, and all of the training data are used in predictions.

Disadvantages: The main disadvantage of kNN is the exponentially increasing computational intensity with more reference data and model size dependence on the training set size. It can also be biased towards the class with the majority of samples. Using data with high dimensionality can also be difficult, due to the distance metric becoming less meaningful with the increasing number of dimensions.

2.3.2 Random Forests

The Random Forests algorithm is an ML algorithm belonging to the ensemble learning category and is often used for classification. Models under the ensemble learning umbrella train several models and combine them together into one. In the case of Random Forests, it is created out of n regression trees.

Each regression tree is a non-linear decision tree [119]. Decision trees are often much more understandable in comparison to more complex machine learning algorithms like *Support Vector Machines* (SVM) or ANNs, which can often look like a black box from the outside. On the contrary, decision trees split data according to simple and more understandable questions.

The questions for the decision trees can have many different forms depending on the data. They can be as simple as just matching an exact value, have a *binary* response to a yes/no problem ($value > threshold$), to more complex formulations like logical or linear combinations of other questions.

Each of the end nodes of a decision tree is called a *leaf*. The output of a decision tree is the last leaf reached by the input of the decision tree starting from the topmost node.

As mentioned at the beginning of this subsection, Random Forests are classified in the ensemble learning category. That is because Random Forests are created by using n decision trees and combining the results of all of them to provide the final prediction. The combination of predictions is most commonly done through

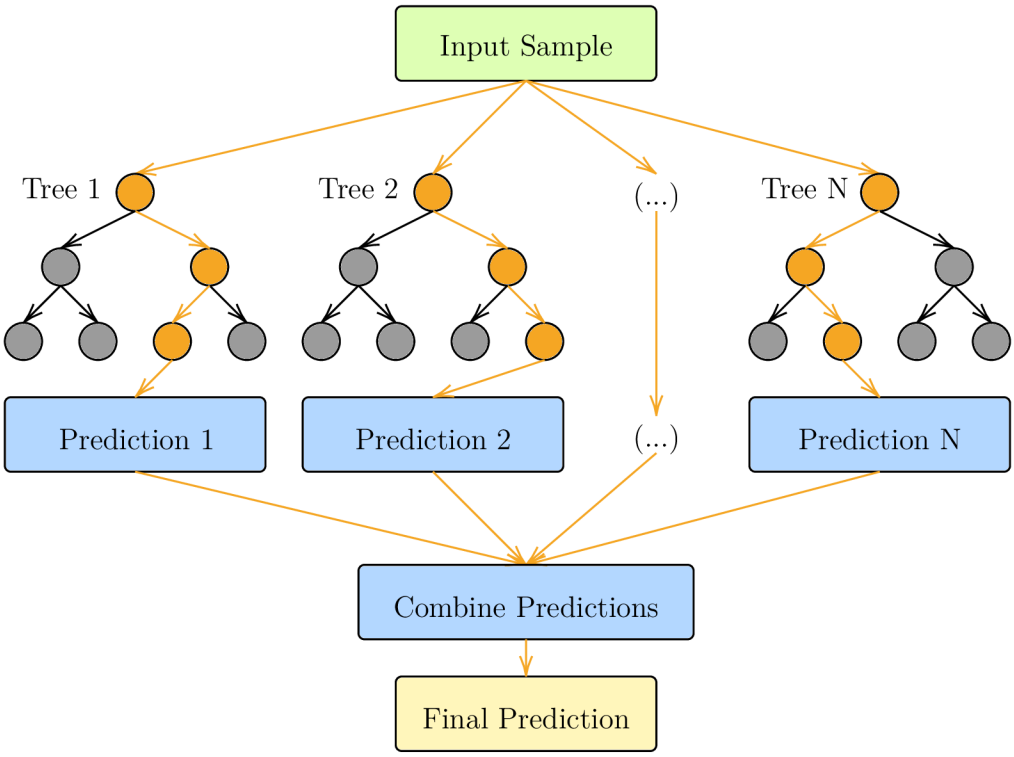


Fig. 2.7: Diagram of a Random Forests algorithm with N Decision trees.

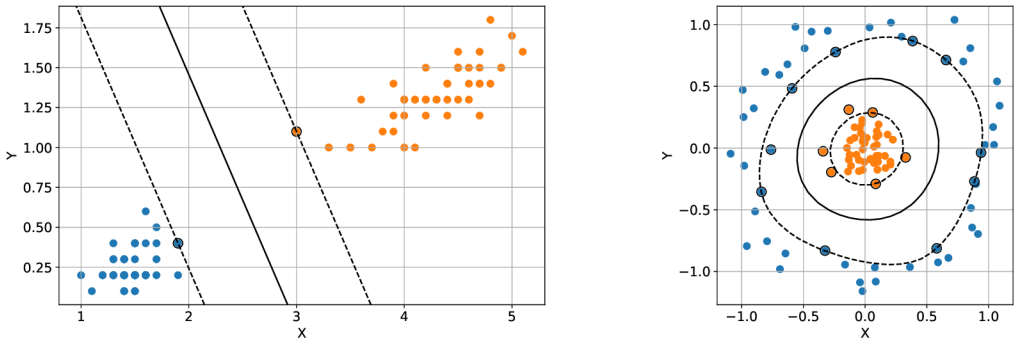
majority voting in classification applications and averaging for regression problems. A diagram of a simple Random Forests algorithm is presented in Fig. 2.7.

Advantages: The advantage of Random Forests is the robustness when it comes to noisy and missing data. They also work well with both data split into categories, as well as continuous data.

Disadvantages: Random forests use several hyper-parameters (number of trees, tree depth, etc.), which can make overfitting easy, in situations where more trees or bigger depth than necessary were used. Random forests are also not the best in cases outside of the scope of training data, as they are not very good at extrapolating. Furthermore, they can be biased to the most commonly occurring label.

2.3.3 Support Vector Machines

The SVM is one of the simple supervised machine learning algorithms. Its main goal is to find an optimal hyperplane for splitting the data between the possible output classes. The optimal hyperplane can be calculated with different kernel functions like linear, polynomial, Gaussian, *Squared Exponential* (SE), and others. With the kernel function, the SVM algorithm calculates in an iterative way the maximum margin from both classes (SVM in its most simple type supports a split into 2 possible data classes, for multi-class classification, the problem is split into multiple binary



(a) Linear SVM kernel.

(b) Non-linear SVM kernel.

Fig. 2.8: Visualisation of SVM decision hyperplanes and support vectors in linear and non-linear scenarios.

classification problems.) The name Support Vector Machines is taken from the way the SVM algorithm finds the optimal hyperplane, as the data points closest to the optimal hyperplane are called *Support Vectors*. Visualization of SVM in 2 simple scenarios, using linear and non-linear kernel (using SE as kernel function), are in Fig. 2.8.

Advantages: SVM is less prone to overfitting because it searches for hyperplane maximally separating data, which encourages generalization. It is suitable for solving both linear and non-linear problems. It also does not need a large training data set as it relies on finding support vectors.

Disadvantages: Just like other ML algorithms, SVM has its disadvantages. It can be computationally intensive as it requires solving complex optimization problems. SVM is also quite sensitive to the choice of a kernel function, and choosing a bad function will lead to poor results. As with kNN and Random Forests, SVM can also be biased in case of imbalanced training data.

2.3.4 Artificial Neural Networks

ANNs fall under the heading of Machine Learning, which is a sub field of Artificial Intelligence [120]. Artificial Intelligence as a whole encompasses the ability of machines to simulate the behavior of humans and mimic their decision-making. The machine learning sub-field focuses on the ability of machines to learn without being programmed to do so. And lastly, the field of deep learning focuses on the creation of ANNs, which are capable of making decisions based on previously learned patterns.

Neural networks were at first designed to imitate the functionality of neurons in the human brain. Neurons in the human brain fire depending on the neural connections leading to the neuron. A very similar principle is applied in the core

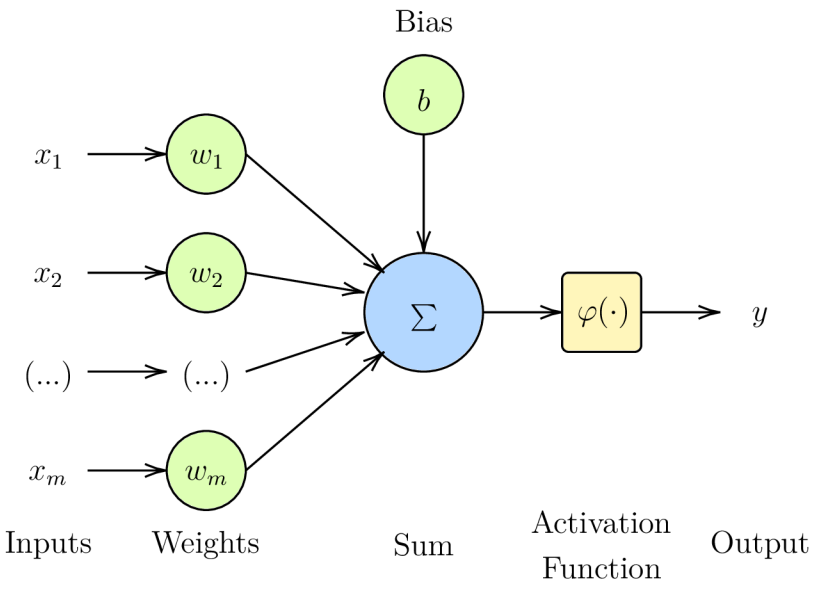


Fig. 2.9: Graphical representation of artificial neuron.

design of modern neural networks, called an artificial neuron.

Artificial Neuron

The perceptron, or artificial neuron, is the elementary building block used in the design of neural networks. Just like neurons in the human brain create a signal based on the input connections, the artificial neuron structure presented by [121] considers all of its input values and produces a result based on the input. The artificial neuron is a sum of weighted inputs and a bias, where weights and biases are the result of the training process. This is a linear process, to make it non-linear, the result of the sum is passed through a non-linear activation function. The function of an artificial neuron is described by [121] equation:

$$y = \varphi\left(\sum_{i=1}^{m} x_i w_i + b\right) \tag{2.4}$$

where m is the number of input values, x_i is the input value with index i, w_i is trained weight for the path, b represents the bias and φ corresponds to the non-linear activation function used. The visualization of Equation 2.4 is in Fig. 2.9.

The activation function is very important. Without the added non-linearity the neural networks could not learn the complex patterns in the training data, and the process would be simple and linear. There are many different activation functions [122], which can significantly differ from each other. The five most commonly used functions are described below.

- **Sigmoid:** is an S-shaped activation function [122]. The sigmoid function has always positive output bounded in the range of 0 to 1. This, and the shape of the function, results in a tendency of bringing the output values closer to one end of the curve (close to 0 or to 1). As the range of output values is bounded, sigmoid does not suffer from output values exploding unreasonably high. On the other hand, there are several disadvantages to using sigmoid as an activation function. The output value tends to change very little towards both ends of the curve. These small changes close to the ends of the curve imply a very small gradient. Due to this, the sigmoid is prone to the vanishing gradient problem, which leads to very slow or impossible learning past this point. The Sigmoid activation function is described by the following equation:

$$\varphi(x) = \frac{1}{1 + \exp(-x)} \tag{2.5}$$

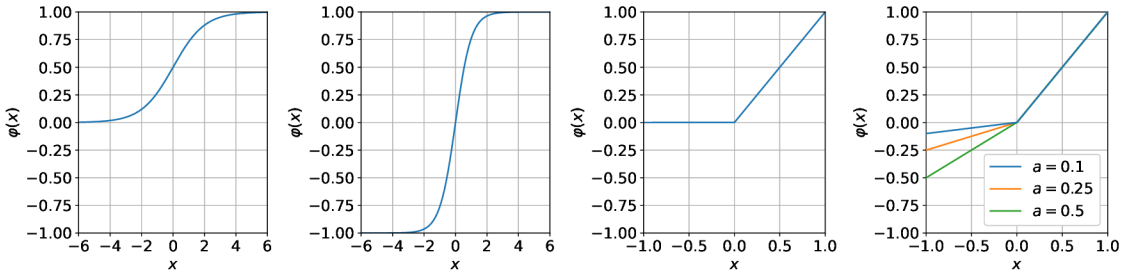
  where y is the output value and x is the input. The graphical representation of the sigmoid is in Fig. 2.10a.
- **Tanh:** is a very similar activation function to sigmoid, the difference being its range. Unlike sigmoid, which ranges from 0 to 1, the range of tanh is from -1 to 1. Just like sigmoid, it is prone to the vanishing gradient problem as the change in output is minimal closer to the ends of the curve. The tanh activation function can be described by an equation:

$$\varphi(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{2.6}$$

  where y is the output value and x is the input. The graphical representation of tanh is in Fig. 2.10b.
- ***Rectified Linear Unit* (ReLU):** was first introduced in 2010 as a way to improve restricted Boltzmann machines [123]. Since then ReLU became the most popular activation function in neural networks used in computer vision. The main reason might be its simplicity and low computational requirements. The ReLU can be described by an equation:

$$\varphi(x) = max(0, x) \tag{2.7}$$

  where x is the input value. The result is the same as the input if x is bigger than 0, otherwise, the result is 0. There are two issues with using ReLU. The fact that neurons can not have a negative output value may result in some nodes not learning anything. While not having any upper limit may cause the values to approach infinity. The graphical representation of ReLU is in Fig. 2.10c.



(a) Sigmoid. (b) Tanh. (c) ReLU. (d) Leaky ReLU.

Fig. 2.10: Most commonly used activation functions.

- **Leaky ReLU:** is a similar activation function to ReLU [124]. Its output is the same when it comes to positive input values as ReLU. But to solve the dying ReLU problem, when it comes to negative values, instead of discarding them they are multiplied by a constant in a range of 0 to 1. This can be described by an equation:

$$\varphi(x) = max(ax, x) \tag{2.8}$$

  where x is the input value and a is a constant, usually of a value of around 0.1. This can solve the dying of nodes, as even neurons outputting negative values are considered in subsequent computations. The graphical representation of Leaky ReLU is in Fig. 2.10d.
- **SoftMax:** This activation function is not used as a follow up to layers in the network to add non-linearity but at the end of classifier networks to present a probability of each class. The SoftMax activation function can be described with an equation:

$$y_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{2.9}$$

  where y_i corresponds to the output with index i, x_i represents input with index i and $\sum_j \exp(x_j)$ represents sum of all input exponentials.

There are many more functions that can be used as an activation [125]. The main requirement for the activation function is non-linearity. Some of the less common activation functions are:

- **Hard Sigmoid** Less computationally intensive Sigmoid function,
- **SiLU** Sigmoid-Weighted Linear Units function,
- **dSiLU** derivative of the Sigmoid-Weighted Linear Units function,
- **Hard Tanh** Less computationally intensive Hyperbolic Tangent function,
- **Softplus** Smooth version of Rectified Linear Unit function,
- **Softsign** Quadratic polynomial function,
- **PReLU** Parametric Rectified Linear Unit function,

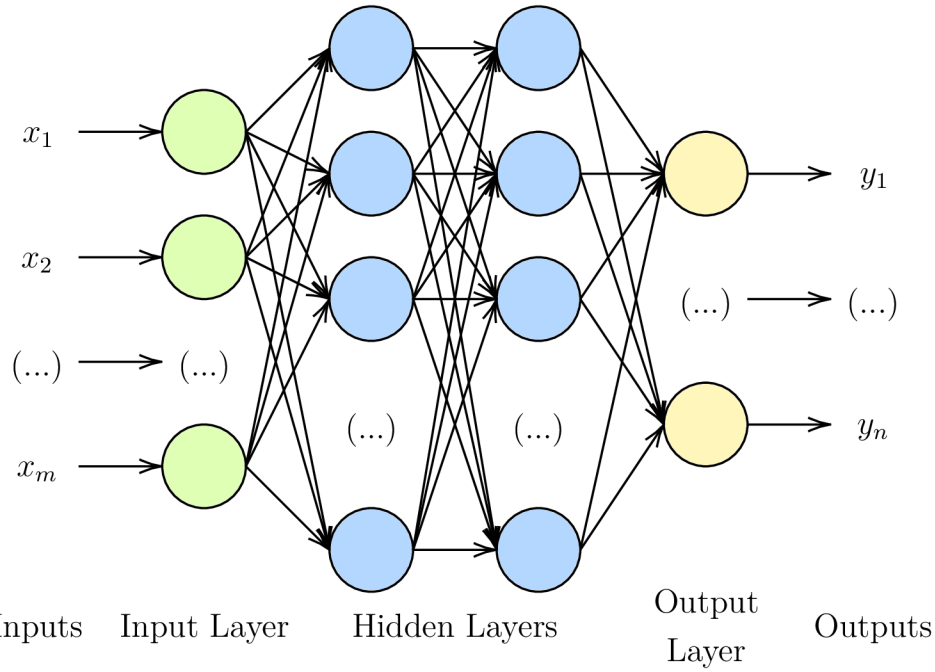


Fig. 2.11: Diagram of a Fully Connected Neural Network with 2 hidden layers.

- **RReLU** Randomized Leaky Rectified Linear Unit function,
- **SReLU** S-shaped Rectified Linear Unit function,
- **ELU** Exponential Linear Unit function,
- **PELU** Parametric Exponential Linear Unit function,
- **SELU** Scaled Exponential Linear Unit function,
- **Maxout** Function applying non-linearity as a product between weights and data,
- **Swish** Input multiplied by Sigmoid function,
- **ELiSH** Exponential Linear Squashing function,
- **Hard ELiSH** Hard Exponential Linear Squashing function.

Fully Connected Layer

Using several artificial neurons in parallel results in a basic type of layer for neural networks called **Fully Connected** or **Dense**. A simple neural network can be constructed by connecting several of these layers in series, as is visualized in Fig. 2.11, which represents a fully connected neural network with 1 input layer, 1 output layer, and 2 hidden layers. Every neuron in networks like this has as an input the outputs of all neurons from the preceding layer. Fully connected layers commonly have thousands of nodes, which results in a lot of parameters being trained and later stored. This makes neural networks using many fully connected layers containing thousands of parameters in each layer very heavy on memory requirements.

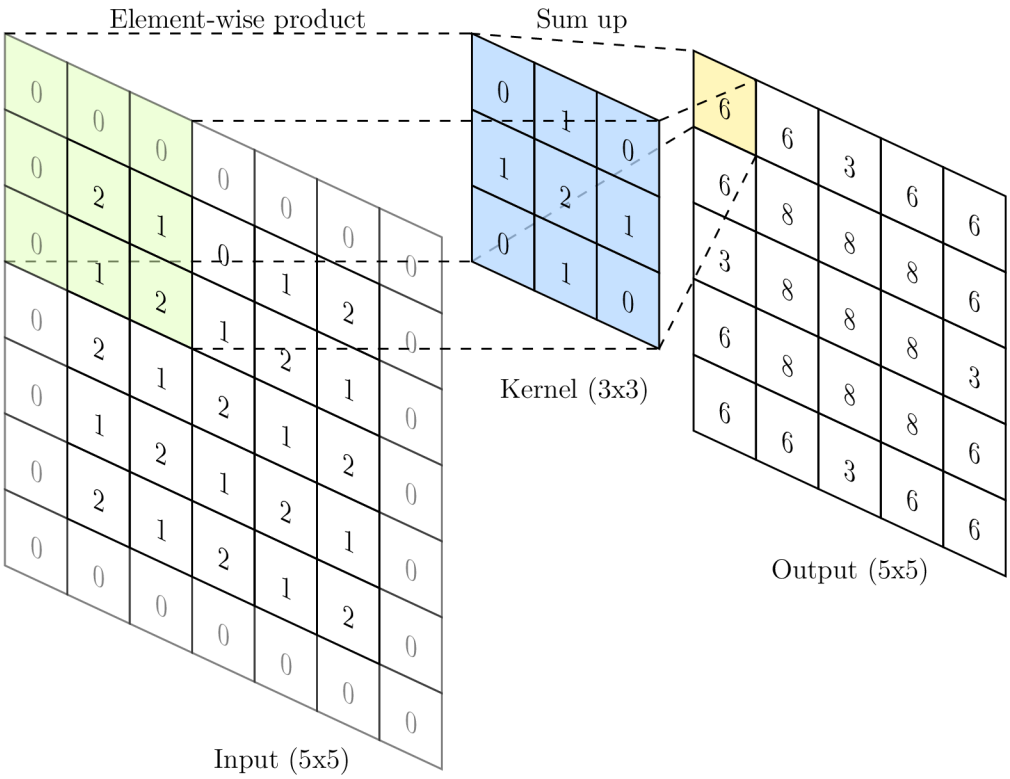


Fig. 2.12: Process of 2D convolution in convolutional layers of neural networks with zero-padded input tensor.

Convolutional Layer

Fully connected layers are not the only ones that are used in neural networks. Another very common type of layer is **Convolutional** [126]. As the name suggests, this layer performs an operation of convolution over the input data. This layer becomes very useful for identifying patterns in the data, regardless of position in the input tensor. Convolutional layers are very common in *Computer Vision* (CV) applications. In this case, the weights that are trained are the values of the convolutional kernel. The process of convolution in 2D space is visualized in Fig. 2.12.

Pooling Layer

Pooling layers play a very important role in modern neural networks. They are used to reduce the dimensions of feature maps inside the network. There are quite a lot of techniques as described by [127] that can be used to reduce the dimensions of feature maps. The two most common pooling techniques are max pooling and average pooling. Max pooling returns the maximum value of the considered range. A usual stride of pooling is 2, to reduce all dimensions of the feature map in question to half. The visualization of max pooling is in Figure 2.13. The second most used pooling

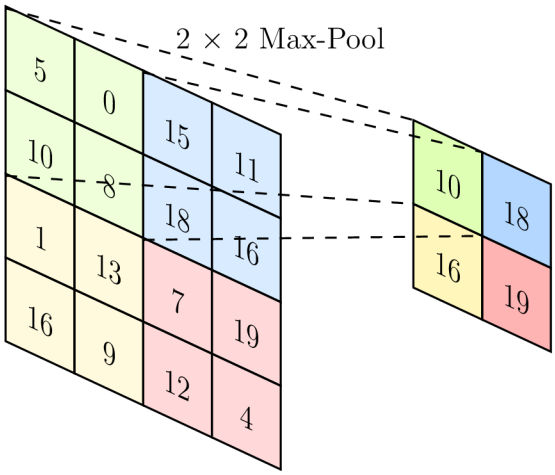


Fig. 2.13: Example of 2D Max-Pooling layer.

technique is average pooling, which, as the name suggests, returns the average value of the feature map area.

Other Layer Types

There are many more different types of layers usable in the design of neural networks. Some are more common than others, and each has its specific function. Some of the more common layers are:

- **Dropout Layer:** The main purpose of the dropout layer is to prevent co-adaptation of multiple feature detectors and is only used during the training phase. The co-adaptation is not desirable as it may lead to overfitting issues [128] and make the accuracy of the network suffer in the process. Dropout layers are one of the ways to fight this issue by disabling a set percentage of feature detectors randomly during the training process [129], [130]. Disabling a percentage of the nodes in the neural network will effectively create a simpler network, which is then trained. The disabled neurons are then restored, and the process of disabling random nodes and training is repeated again. By having some layers use different neurons in every training iteration, the neural network will become more robust and less prone to errors during evaluation.
- **Batch Normalization Layer:** Training very deep neural networks, with a lot of layers is a challenging task. One of the possible reasons for this difficulty is the distribution of the inputs deep into the network. This change in the distribution of the inputs inside the network is referred to as an internal covariant shift. To fight this issue, a batch normalization layer was proposed in [131]. This type of layer normalizes the input into the following layer, which results in a more stable network during training and better results during inference.

- **Deconvolutional Layer:** As the name suggests, this layer is used for up-scaling of feature maps, used, for example, by UNet [132] architecture.
- **Recurrent Layer:** Works as a memory, the main purpose is to analyze the combination of current data and previous input. Networks using this layer are called on *Recurrent Neural Network* (RNN).

Issues of Neural Networks

Even though neural networks are lately being used in almost every scientific field, there are some issues with using them. The issues are most often related to the training of the networks and the hardware required for inference.

- **Initial setup:** Just like SVM, Random Forests, and many other machine learning algorithms, neural networks require training. When comparing the required training time, most often the neural networks need much more time to train. This process adjusts the weights and biases inside of the network to reduce the output error. This process of adjusting weights based on the difference between output and ground truth is called back-propagation. Back-propagation is a computationally intensive task, during which the weights are adjusted based on their influence on the output.
- **Over-fitting:** One of biggest concerns when creating and training a new model of neural network is over-fitting [128]. Over-fitting a neural network means that the neural network learned the training data so well, it can not generalize well enough to make valid predictions on never seen data. In the real world, it means the neural network produces a very low error rate during the training process, but the error is much higher with never before seen input data. If the model is over-fitted, the underlying reason might be too small a training dataset or having a far too complex model for the task at hand.
- **Under-fitting:** The opposite of over-fitting is the issue of under-fitting [133]. Under-fitting occurs when trying to train too simple a model for an overly complex task. This can make the model inflexible in learning features from the training dataset, which results in low accuracy and bad results. The solution to under-fitting is to create a more complex model better suited for the task at hand, increase hyperparameters, reduce the number of features, and train longer.
- **Output expansion:** Due to the nature of neural networks, the output classes or regression outputs are fixed during training. This makes the expansion of the neural network outputs quite difficult, as the architecture itself requires modifications. Additionally, the network will require retraining to tweak the weights to account for another output. The training dataset also needs to

contain occurrences of the new class. All this makes output expansion of already trained existing neural networks difficult and inconvenient.

- **Hardware requirements:** Neural networks can become very hardware intensive. The networks achieving the highest accuracy have usually millions of parameters. This results in high usage of memory and a huge amount of operations to execute. From their nature, neural networks highly benefit from parallel processing. Massive parallelism available by using *Graphical Processing Units* (GPUs) for general purpose computing helps reduce inference times but results in a lot higher overall power consumption of the system. In the last couple of years, as a reaction to the rise of neural networks in popularity, regular *Central Processing units* (CPUs) also started introducing integrated machine learning accelerators [134], [135].

2.4 Techniques

Similar to the ML algorithms described in the previous Section, the description of data processing techniques and data analysis are also provided. This Section will focus on processing techniques in IPS used in general and throughout the rest of the thesis.

2.4.1 Inertial Positioning

Inertial positioning uses sensors available in the UE to track the movement of the user over time [92], [136]. Most commonly, the data from accelerometers and gyroscopes are used to keep track of the movement. The accelerometer measures linear velocity along 3 axes, and the gyroscope measures angular velocity along the same 3 axes.

By tracking these measurements, the UE can estimate its own position. However, the accuracy is highly dependent on the accuracy of the initial location as well as on the quality of the sensors. The quality of sensors is the main reason for the decrease in accuracy over time, due to the accumulation of errors over time. To fight this error, inertial systems are often combined with other positioning systems like GNSS to periodically calibrate their position.

Inertial positioning has many use cases, such as navigation of *unmanned Aerial Vehicle* (UAV), indoor robots, or XR systems. However, due to the error accumulation, it can be challenging to create high-accuracy indoor positioning and navigation system using inertial positioning.

2.4.2 Computer Vision based Positioning

The basic idea behind computer vision-based positioning is the use of cameras and identification of features or landmarks in the captured images to try to determine the user location [89], [91]. Upon successful identification of a feature, the system can employ algorithms like triangulation or simultaneous localization and mapping [137] to estimate the position and rotation of the feature.

The advantage of using computer vision in positioning applications is that they can work in situations where other approaches struggle. As an example, in an environment without access to the *Line of Sight* (LoS) necessary for GNSS and in places without ideal signal propagation properties for the use of Wi-Fi or BLE based IPS.

In addition to being able to work well in environments with bad signal reception, computer vision can achieve high accuracy in indoor environments, where the image processing algorithms can leverage the rich visual information from the environment. Computer vision positioning also has limitations which are mainly tied to processing complexity, which might be a deciding factor for many applications, and the susceptibility to changes in lighting conditions, occlusion of the environment, or changes to the furnishing of the space.

Despite the limitations, computer vision-based positioning shows promising results in ongoing research [138]. Currently, the accuracy, robustness, and scalability for real-world use cases are the main focuses of research [139].

2.4.3 RSSI Fingerprinting in Indoor Positioning Systems

Fingerprinting is one of the most common ways to approach indoor positioning [140], [141]. It is a relatively easy approach, whose main disadvantage is the need to adapt it to each building, and it is also susceptible to changes in the interior, as it depends on the signal propagation properties. The main requirement for fingerprinting is the availability of **anchors**. As anchors, in case the technology of choice is Wi-Fi, the existing infrastructure of Wi-Fi APs can often be reused. Otherwise, the anchors have to be placed around the space to create the positioning infrastructure from scratch. That is the case with other technologies used for indoor positioning, like BLE or UWB. Fingerprinting is mainly used in situations when the user carries a device. The 2 approaches are:

- **Active:** The UE is collecting signal propagation information from anchors placed around the environment to calculate its own location.
- **Passive:** Sometimes also called **Server-Side** because the device does not do any actual processing of data. In this approach, the anchors collect signals from the devices in their range and calculate the location of the device.

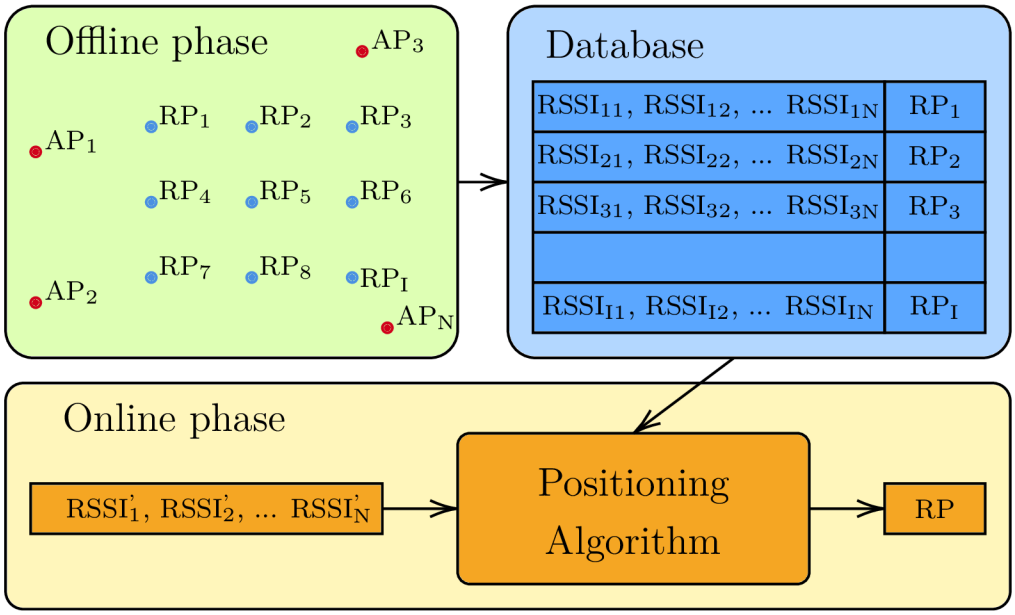


Fig. 2.14: Diagram of the fingerprinting method in indoor positioning using network *Access Points* (APs) and *Reference Points* (RPs) for data collection.

The process of fingerprinting in indoor positioning has 2 main stages:

1. **Offline:** During this stage, it is necessary to evaluate the area of interest. In case of lack of reusable infrastructure, availability of anchors. Check if the anchors are spread around and the UE is in range of signal from the anchors, or anchors in range of the signal transmitted by the UE depending on the approach. When this is ready and there is a coordinate system in place to collect coordinates as labels for the collected RSSI measurements, the database of fingerprints, also known as RM, can be created.
2. **Online:** During the online phase, the database of fingerprints is used, along with a positioning algorithm to get the location out of RSSI values collected either by the UE or the server.

The diagram describing the 2 phases of fingerprinting in indoor positioning applications is presented in Fig. 2.14. The diagram shows the connection of both phases to the fingerprint database.

2.5 Reproducible Research

Reproducibility in research is an important part of scientific work. As such, all software developed in the creation of this thesis is open-source and available to the public. The same goes for the collected datasets, which are available to download as supplementary materials for each published work from public repositories.

2.5.1 ESP32 Probe Request Sniffer

In several of the publications used in this thesis, the packet sniffer based on an ESP32 MCU is used for dataset collection. To simplify, the ESP32 firmware captures Wi-Fi management frames. Following the capture, using a filter it selects only probe requests and saves them in a standardized binary packet capture file compatible with several network traffic applications like Wireshark (network protocol analyzer) or Python library Scapy (library for packet manipulation) and others.

The first time the firmware was created for use in works described in Chapter 3, it did not include saving of RSSI into the file, and it was also missing MAC address filtering. These features were added later for work on RM interpolation in Chapter 4.

Probe requests do not contain a field with transmission time, and because of that, the ESP32 first connects to a Wi-Fi network specified in the *config.h* file to download current time from the *Network Time Protocol* (NTP) servers. After synchronizing the internal clock with the outside world, the ESP32 initializes the connection to the SD card and mounts the file system. Using the output file mask configured in the *config.h*, the firmware checks if any files match the mask; if so, it will find out the highest index and opens a new file without overwriting any of the existing ones. After that, the startup sequence is finished by changing the state of the wireless interface into monitoring mode and executing the sniffing and saving tasks.

While the sniffing task runs, all received packets are checked for their type. If the packet is a probe request (in case the variant of the firmware with MAC address filtering, the transmitter MAC address also needs to match the one in the filter), the packet is saved in a file on the SD card. All other received packets are discarded. The output file is saved after a time interval set in the *config.h*, this is to prevent the corruption of data caused by the loss of power.

The sniffing task continues to collect probe request frames until the user presses the *Stop* button, this action raises an *Interrupt Service Routine* (ISR), which stops the sniffing task, saves the current pcap file, and safely dismounts the SD card. The simplified working of the ESP32 firmware is in the Algorithm 1.

All variants of the firmware are publicly available for download and modifications under the Public Domain license. The version without MAC address filtering is available from its own GitLab repository [142]. Saving of RadioTap information is available on a separate branch of the repository. The modified version with MAC address filtering is also available on GitLab [143], in this case as a part of a repository containing all created software used for work described in Section 4.3.

Algorithm 1: Probe Request Sniffer.

```
1  Initialize MCU peripherals and GPIO
2  Connect to Wi-Fi and download the current time
3  Initialize and mount SD card
4  Check for existing files and open a new pcap file
5  Reinitialize Wi-Fi in monitoring mode
6  Start probe sniffing task - run callback Received Packet
7  Start saving task - periodically run callback Save PCAP

8  callback Received Packet:
9  |  if packet.type = ProbeRequest and packet.mac = FilterMac then
10 |  |  Write packet to file
11 |  end
12 end

13 callback Save PCAP:
14 |  Close current pcap file
15 |  Open a new pcap file
16 end

17 isr On Button Press:
18 |  Stop probe sniffing task
19 |  Close the pcap file and unmount the SD card
20 end
```

2.5.2 ESP32 Probe Request Sender

Another firmware for the ESP32 in this thesis is the Probe Request Sender used for the creation of RMs in Section 4.3. It is much easier in comparison to the Sniffer firmware, even though the initialization sequence might be very similar.

The ESP32 initializes all required peripherals and *General-Purpose Input/Output* (GPIO), connects to predefined Wi-Fi, and synchronizes the internal clock with the NTP servers. At last, the SD card is mounted, and the ESP32 becomes ready for transmissions of Probe Requests. The sequence of scans for nearby Wi-Fi networks is started by the press of a button. This raises an ISR, which runs a predefined number of scans with a delay in between each of the scans. Both number of scans and the delay can be configured in the *config.h* file. After all scans in the sequence are completed, the time of the first and last scan is saved into a text file for future reference. Simplified working of the firmware is described by Algorithm 2.

In a similar way to the ESP32-based Probe Request Sniffer described above, the

Algorithm 2: Probe Request Sender.

```
1  Initialize MCU peripherals and GPIO
2  Connect to Wi-Fi and download the current time
3  Initialize and mount SD card

4  isr On Button Press:
5  |  for Scans Per Cycle do
6  |  |  Wi-Fi Scan
7  |  |  Wait set delay between Scans
8  |  end
9  |  Write time of first and last scan to file
10 end
```

Probe Request Sender is also available to the public. It is a part of the repository containing the software for work described in Section 4.3, just like the Sniffer with MAC address filtering.

2.5.3 Datasets

Using the previously mentioned ESP32 MCUs, several datasets were collected. A couple of datasets of probe requests were collected in 2 different environments, one week in 2021 and one month in 2023 inside of the university office and during an international conference. These datasets, due to the nature of probe requests, had to be anonymized, which was done by application of the SHA512 algorithm on all fields containing data about the user. Some of these fields were the SSID from the preferred network list, WPS fields that can contain the device model, name, and many others. The output of the SHA512 is 64 bytes long, however, only small sections of the hashing output were used. There are 2 reasons for that:

- MAC address field of the transmitter, has a fixed length of 6 bytes, to preserve compatibility with network analysis tools, the MAC address length was preserved.
- reducing the memory requirements, in case every anonymized field had a length of 64 bytes, the analysis of longer time intervals would be more intensive on the memory requirements of the system.

The MAC address requires special treatment during the anonymization process. To keep the possibilities for analysis intact, the first 3 bytes of the original MAC address were left unchanged because those are the most important for analysis. For devices not randomizing their MAC address, this keeps the ability to identify the manufacturer of the UE as the first 3 bytes can be matched to the *Organizationally*

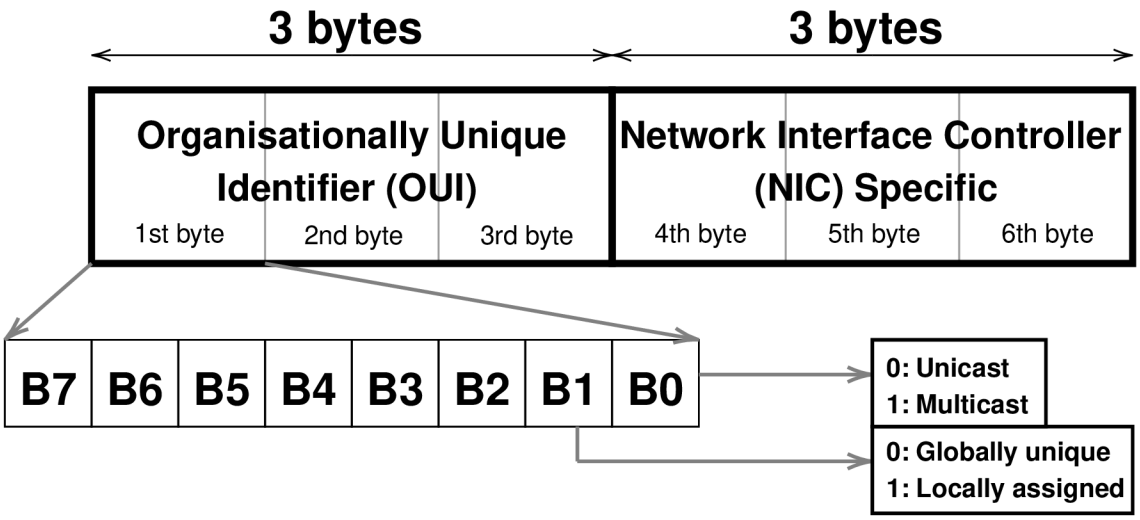


Fig. 2.15: Structure of MAC address with the functional bits.

Unique Identifier (OUI) registered by IEEE [144], hide the original MAC address and preserve the ability to recognize if the MAC address before anonymization was randomized or not. The structure of the MAC address is presented in Fig. 2.15.

2.5.4 Probe Request Dataset - IPIN

The probe request dataset collected at the 11th International Conference on Indoor Positioning and Indoor Navigation (IPIN 2021) contains 390810 captured probe requests. The monitoring of the Wi-Fi started 38 minutes before the conference program began with a tutorial session, on Monday 29$^{\text{th}}$ November, 2021 at 08:22. The final probe request was collected on Thursday 2$^{\text{nd}}$ December, 2021 at 13:02, just a moment after the closing ceremony concluded. It was not possible to keep the sniffer working past the official end of the conference because of the preparation of the conference space for the following event.

The dataset contains all captured probe requests. Most of them had additional fields with information about the capability of the UE. The majority of the fields contain information related to the supported Wi-Fi standards, however, a couple of fields, mainly *Vendor Specific element* and WPS carry information about the UE. The frequency of occurrence of each field is in Table 2.2.

The captured probe requests, in some cases, contained the user information. Some of the user information was real MAC addresses of their UE, SSIDs from the PNL the UE transmits in search of AP it connected in the past. In some cases, the UE transmitted even the name of the device and the user, which happened in the transmission of probe request with WPS field. To keep the computational requirements low, the data anonymization was not included directly during the

Tab. 2.2: Probe request fields and their frequency of occurrence in a dataset collected at IPIN 2021.

Information Element	Included in Probes	[%]
Supported rates	390 211	99.85
Extended Supported rates	385 606	98.67
HT Capabilities	359 391	91.96
VHT Capabilities	51 031	13.06
Extended Capabilities	312 181	79.88
Vendor Specific elements	228 970	58.59
1 Vendor Specific element	84 215	21.55
2 Vendor Specific elements	67 663	17.31
3 Vendor Specific elements	55 524	14.21
4 Vendor Specific elements	21 462	5.49
5+ Vendor Specific elements	106	0.03
WPS - UUID-E	3733	0.96
WEP Protected	599	0.15
Total Collected Probe Requests	390 810	

capture - the captured data are the same as with other network analysis tools like Wireshark. After the capture ended, the anonymity of the data was ensured by anonymizing all fields of the probe request that can contain user-related information with the SHA512 algorithm.

The in-person conference event took place during the COVID-19 pandemic and was quite isolated. In the conference space, there was minimal presence of people not participating in the event. The only people in the proximity of the sniffer were the attendees of the conference, conference organizers, and hotel staff.

The anonymized version of the probe request dataset collected at IPIN 2021 is publicly available through the Zenodo repository under the A-WEAR community [145]. The dataset was published as supplementary material to conference publication [146], which is described later in Section 3.3.

2.5.5 Probe Request Dataset - UJI 2021

This dataset was collected during one week starting on Thursday 9th December, 2021 and ending on Wednesday 15th December, 2021 in the GEOTEC office at UJI. This dataset was collected in a similar way to the probe request dataset captured at the IPIN 2021 conference using the ESP32-based probe request sniffer described

Tab. 2.3: Probe request fields used to create device fingerprint and frequency of occurrence in data collected in the lab.

Information Element	Included in Probes	[%]
Supported rates	340 360	100.00
Extended Supported rates	340 198	99.95
HT Capabilities	312 227	91.73
VHT Capabilities	20 252	5.95
Extended Capabilities	232 918	68.43
Vendor Specific elements	194 801	57.23
1 Vendor Specific element	29 681	8.72
2 Vendor Specific elements	47 375	13.92
3 Vendor Specific elements	8661	2.54
4 Vendor Specific elements	104 559	30.72
5+ Vendor Specific elements	4525	1.33
WPS - UUID-E	35 908	10.55
Total Collected Probe Requests	340 360	

in Section 2.5.1. The dataset contains 340360 probe requests with the frequency of occurrence of information element fields presented in Table 2.3.

Just like the probe request dataset collected at IPIN 2021, this dataset contained user information and it stays to say that the analysis was only approached from the implementation of Wi-Fi protocol, specifically to explore potential privacy issues in current implementations. The dataset was then anonymized in the same way as the dataset from IPIN 2021 by using SHA512 hashing algorithm over sensitive fields. Even though the originally collected data contains both real and randomized MAC addresses, it is not possible to match MAC addresses to specific individuals because the analysis was done over an anonymized version of the dataset, and additionally, data regarding the actual presence in the office or the building were not collected.

The anonymized version of the probe request dataset collected at the GEOTEC office in 2021 is publicly available through the Zenodo repository under the A-WEAR community [147]. The dataset was published as supplementary material to conference publication [148], which is described later in Section 3.2.

2.5.6 Probe Request Dataset - UJI Probes 2023

This dataset was also collected in the office of the GEOTEC department of University Jaume I, Spain and the data descriptor paper was accepted for publication [149].

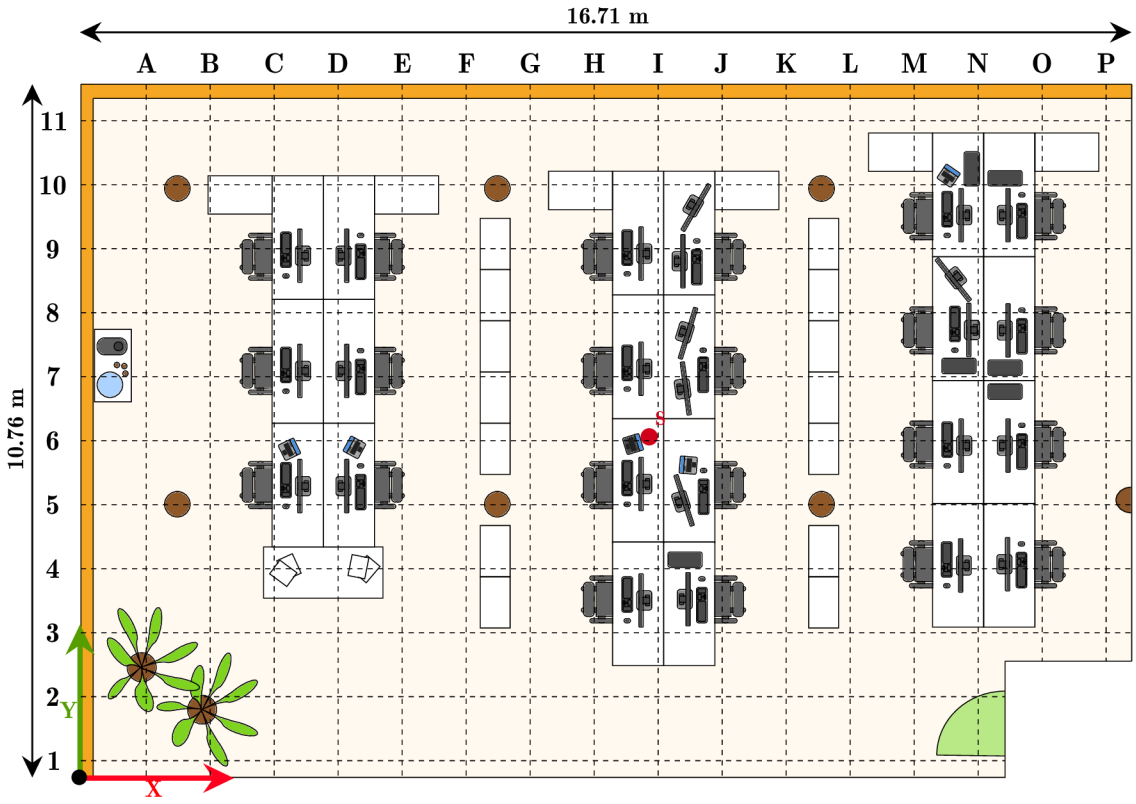


Fig. 2.16: Floor plan and location of sniffer in the office space of GEOTEC department at UJI, Spain.

The office has a rectangular shape with dimensions of 16.71 m by 10.76 m. The office is of open-space style with an average occupancy of approximately 14-20 people. The floor plan of the office is depicted in Fig. 2.16.

The probe requests were collected during the month of March, to capture apart from regular work weeks also the local holiday Magdalena 2023, during which the university was mostly closed. Magdalena 2023 started on Saturday 11$^{\text{th}}$ March, 2023 and ended a week later on Sunday 19$^{\text{th}}$ March, 2023. The reduced occupancy of the office is clearly visible from Fig. 2.17, where the holidays of Magdalena are visualized in light green color.

There are some noticeable trends present in the dataset that are visible from Fig. 2.17. One of these trends is the noticeable constant transmission of probe requests. During the entire collection time, probe requests were being sent at all times, be it during the day, or night, workday or weekends. The explanation for the probe requests captured at night can be all-in-one computers using Wi-Fi instead of a wired connection, phones, or IoT devices used for experiments in the office that can be transmitting probe requests at all times. The second noticeable thing is a peak in the transmitted probe requests happening every day around 07:00. This is due

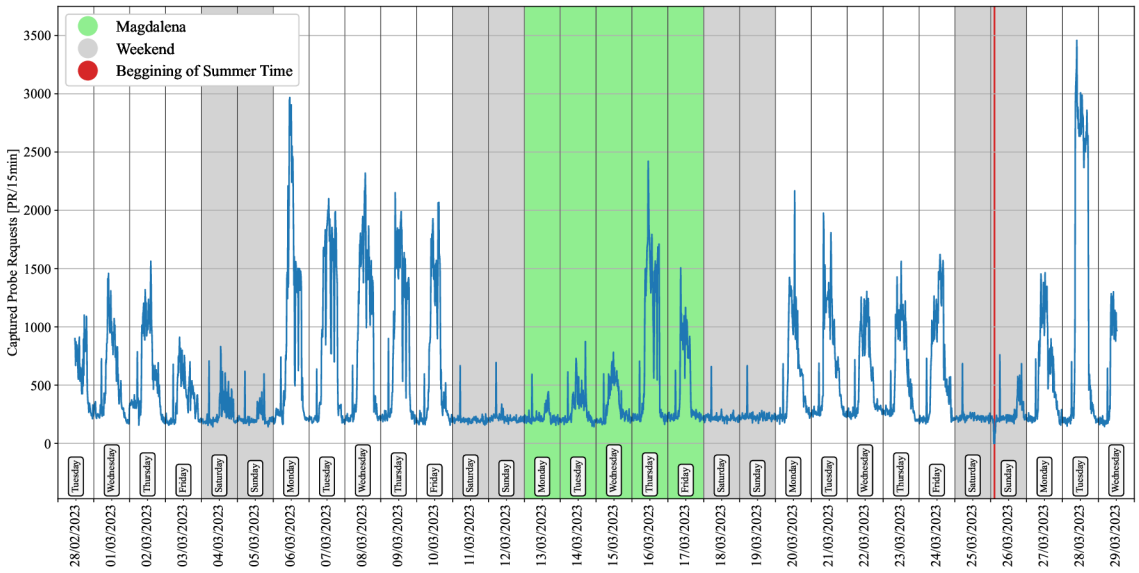


Fig. 2.17: Density of captured Probe Requests in the UJI Probes dataset (amount of probe requests grouped in 15-minute clusters).

to the scheduled reboot of a Wi-Fi access point present in the office and devices searching for a network to connect to after being disconnected from it. Another noticeable thing is a short time period at night of the Sunday 26$^{\text{th}}$ March, 2023 with no captured probe requests. That is due to the switch to the Summer Time when the time changed from 02:00 to 03:00.

It can be disclosed that some of the employees went to the office during some of the days of Magdalena and during the first weekend since the beginning of the monitoring. This explains the lower but still detectable increase in captured probe requests.

MAC addresses and SSIDs: Another important factor in the analysis of probe requests are MAC addresses of devices. In case of devices not using randomization of MAC addresses, such UE is very easy to track. Randomized MAC address is also very easily identifiable due to the 2nd least significant bit **B1** in the first byte of the MAC address. When this bit is set, the MAC address was randomized by the network controller of the UE. The least significant bit of the first byte **B0** distinguishes between individual devices and device groups. The structure of the MAC address is presented in Fig. 2.15. Considering this limitation of the 2 least significant bits, the 2nd digit of locally assigned MAC address in hexadecimal format has only four options: 2 (0010), 6 (0110), A (1010), or E (1110). From the captured probe requests, about 35 % used randomized MAC addresses, which is presented in Fig. 2.18.

Second very important parameter is the PNL, which is the list of network SSIDs

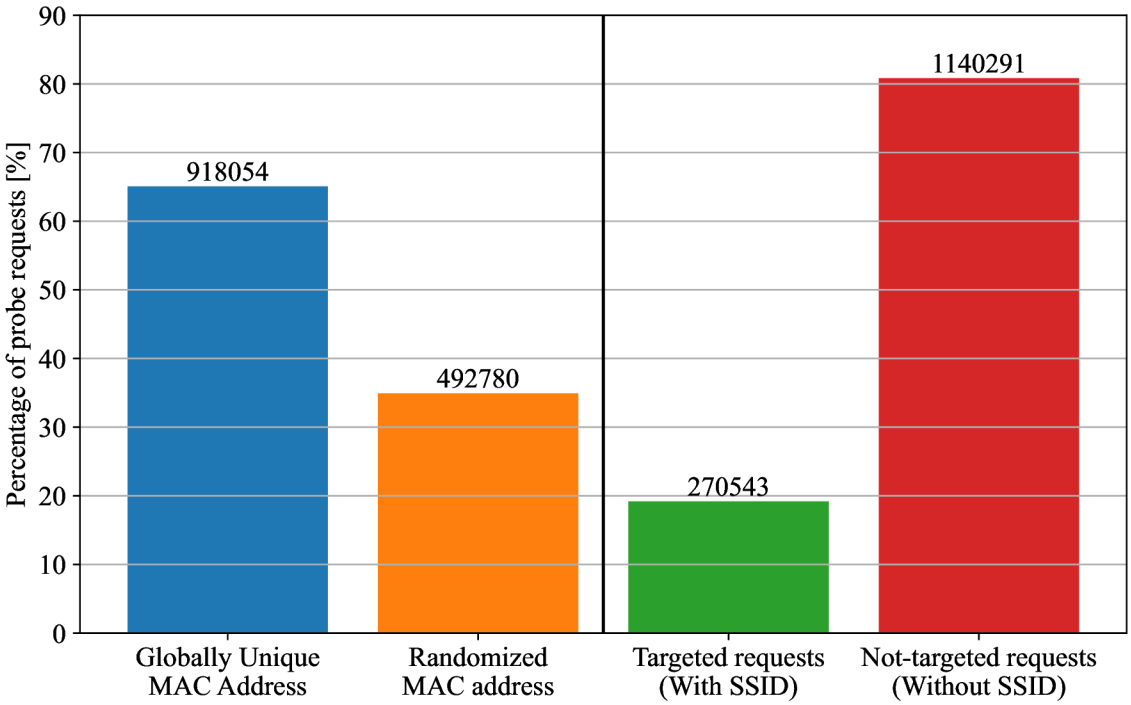


Fig. 2.18: Split between randomized and globally unique MAC addresses in the captured probe requests and split between targeted and not targeted probe requests.

the UE often connects to. This can be leaked by the device sending probe requests targeted for specific networks. In this dataset, 19 % of the captured probe requests contained an SSID, which is presented in Fig. 2.18. Even though this seems like a fairly low number, the dataset contains 2030 unique SSIDs.

Information Elements: This optional section of probe requests is very useful for fingerprinting, as it presents us with a set of capabilities and supported functions that are not changing over time for a single UE. The information ranges from supported data rates, capabilities related to certain Wi-Fi standards (High Throughput (HT) capabilities -802.11n, Very High Throughput (VHT) capabilities - 802.11ac, and High Efficiency (HE) capabilities - 802.11ax), vendor-specific elements (some devices had several of them) to WPS fields, which can contain many user identifying information ranging from the device manufacturer all the way to device name (and since many devices are named by users - *Julia's iPhone*, this can leak the name of the user). WPS fields also come with *Universally Unique IDentifier-Enrollee* (UUID-E). UUID-E is also a unique identifier that can compromise the anonymity while using randomized MAC addresses as it does not change over time. The occurrences of each field are shown in Table 2.4.

Radio Information: The ESP32 sniffer was configured to gather probe requests in all channel scans. Due to the lack of support of ESP32 for the 5 GHz band of

Tab. 2.4: Probe request fields and their frequency of occurrence in data in the UJI Probes dataset.

Information Element	Included in Probes	[%]
Supported rates	1 410 832	100.00
Extended Supported rates	1 405 288	99.61
HT Capabilities	1 093 575	77.51
HE Capabilities	394 347	27.95
VHT Capabilities	262 365	18.60
Extended Capabilities	1 182 014	83.78
Vendor Specific elements	315 815	22.39
1 Vendor Specific element	135 973	9.64
2 Vendor Specific elements	110 071	7.80
3 Vendor Specific elements	9607	0.68
4 Vendor Specific elements	257	0.02
5+ Vendor Specific elements	146	0.01
WPS - UUID-E	16 596	1.18
WEP Protected	2	0.00
Total Collected Probe Requests	1 410 834	

Wi-Fi, only probe requests transmitted in the 2.4 GHz band were captured.

The distribution of RSSI values are presented using a histogram in Fig. 2.19. From the histogram, it is clearly visible that most frames were captured with RSSI in the range of −100 dBm to −40 dBm (0.000 000 000 1 mW to 0.0001 mW). A little surprisingly, a lot of probe requests had RSSI higher than −30 dBm. All of these probes were transmitted by a single device that had the wireless interface configured to use higher transmit power than usual. Another point of interest is the high amount of probes captured with the RSSI around −90 dBm, which are most likely devices from a neighboring office. The radio signal propagation throughout the office was mapped for research done in Section 4.3. The RM of the office mapped using another ESP32 MCU is in Fig. 2.20. The capture RM can be used as a rough distance filter, as it can be used for the selection of RSSI threshold.

Usage Examples

There are several ways to use the dataset. In this section, a few examples of possible use cases are provided.

Wi-Fi Signal Stability Evaluation: By capturing the RSSI values for each captured

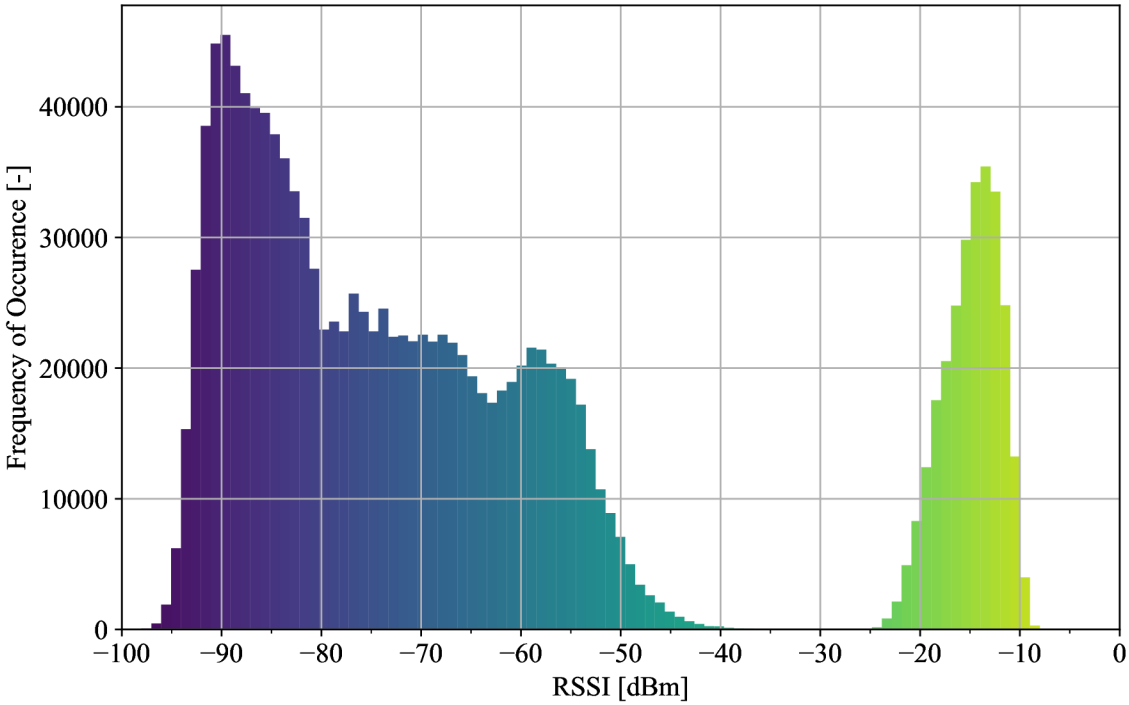


Fig. 2.19: Frequency of occurrence of RSSI in the captured probe requests.

probe request, it is possible to analyze the signal from the received signal strength point of view as well. The proposed example takes a look at the signal stability over time. As an example, the random MAC address of a single device that appears on several days was selected (MAC address of the selected device in the anonymized dataset is *f4:7b:09:1f:5b:72*), and visualized the RSSI of all probe requests over time. From this, the changes over time are visible. To highlight the change, a trend line was also included in Fig. 2.21 representing the decrease in RSSI throughout the capture time.

Presence Detection & Room Occupancy Estimation: The network traffic can be also used for presence detection. This can be easily seen in Fig. 2.17, from where it can be clearly seen when the activity in the proximity of the sniffer increased. The drops on weekends and holidays can also be clearly distinguished. By also using the RM from Fig. 2.20, it is possible to roughly estimate the distance the UE is from the sniffer. This information can be utilized for presence detection or even room occupancy detection. On a smaller scale, the estimation of room occupancy at the university campus from captured probe requests is in Section 3.4.

Unfortunately, capturing the ground truth of room occupancy in the GEOTEC office is nearly impossible. With up to 30 people present in the office, moving in and out at all times during work hours, the collection of ground truth was not possible with the current setup. However, the occupancy can be classified into categories like:

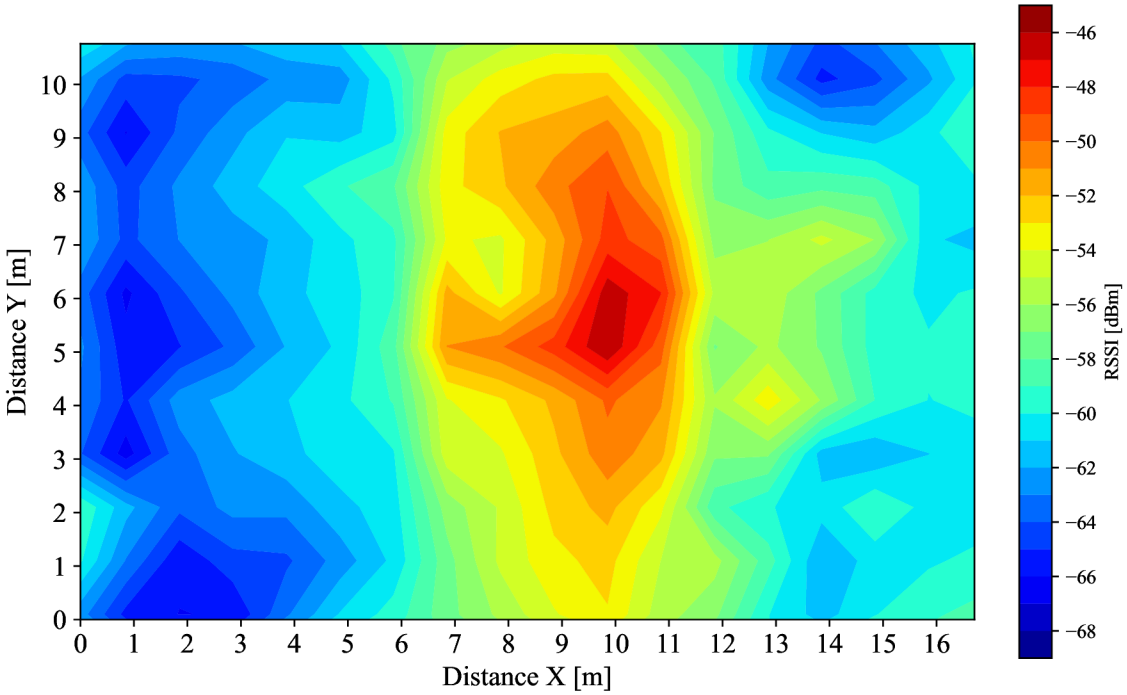


Fig. 2.20: Radio Environment Map of the RSSI in different locations of the office mapped with ESP32 MCUs.

empty, low, medium, high, or on a scale from *low* to *high*, which was used previously in occupation estimation work by Ciftler *et al.* [150].

User Privacy Exploitation: Since during the anonymization of the dataset no data were removed, but all fields use pseudonyms created by hashing, the dataset allows for privacy leaking behaviors in an up-to-date data. The *Time of Arrival* (ToA) of packets was not modified in any way either, which allows for studies into temporal analysis similar to the work done by Matte *et al.* [151].

It can also be used for the analysis of randomized MAC address recurrences or identification of users despite the randomization. The dataset may be used to reveal present vulnerabilities of the probe request mechanism and creation of a new type of analysis.

The probe request dataset is available to the public on Zenodo repository [152]. The accompanying data description paper was accepted to the 13th edition of the IPIN conference - IPIN 2023.

2.5.7 REM Interpolation Dataset

This dataset was created for the work focused on balancing the accuracy, compute requirements, and time required for the data collection of RMs for fingerprinting

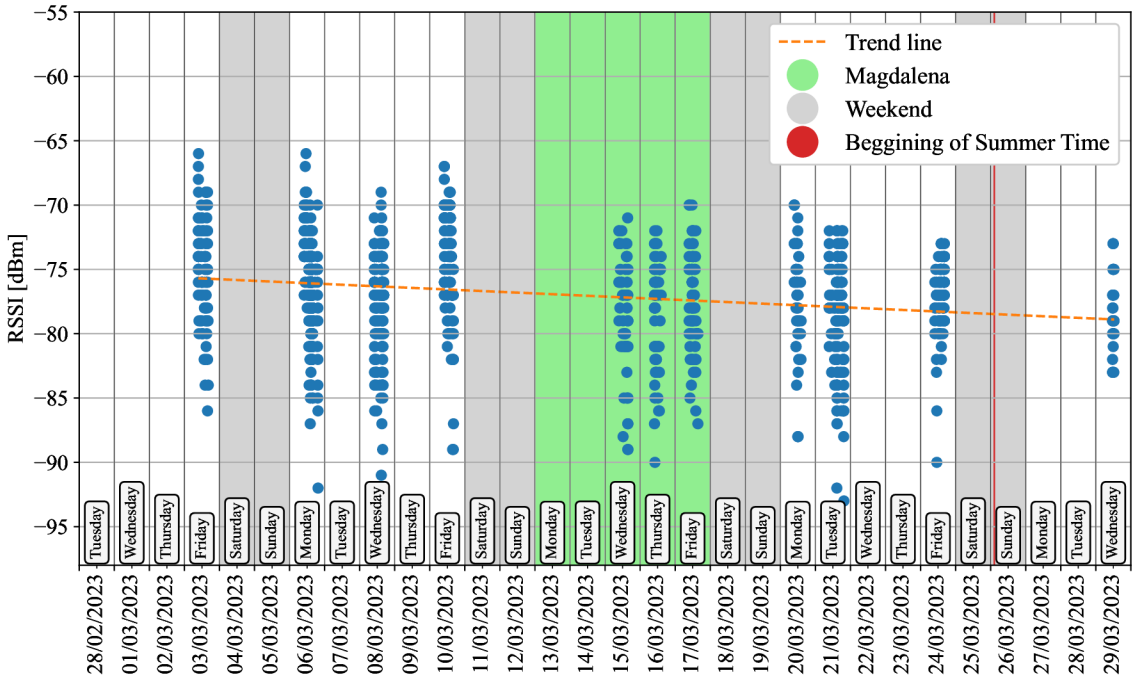


Fig. 2.21: Example of RSSI long-term stability evaluation for one MAC address from the dataset (*f4:7b:09:1f:5b:72*) including trend line.

approaches to indoor positioning. The dataset represents the radio environment map created in the office at UJI, Spain.

The data of the dataset are split into 4 folders, files with complete probe requests for both main radio map and extra evaluation data, processed data stored in *Comma Separated-Values* (CSV) files, 14 scenarios including train-test split, and documentation:

- **Data** - Raw packet capture data stored in pcap files split by sniffer station representing the complete RM. There are in total 142 pcap files matching all easy-to-access RPs in the office aligned in 1 m grid. The pcap files are named according to RP marks placed in the GEOTEC office, which are explained in Table 2.5.
- **Data_Eval** - Extra evaluation data in raw packet capture files split by sniffer station. These samples were collected to fairly evaluate the positioning accuracy of the kNN based IPS. The evaluation samples were collected in inaccessible spaces as well as between regular RP. Since these samples are collected in a denser grid, the coordinates are directly in the names of the files. The data were collected at an additional 31 RPs.
- **Data_Scenarios** - 14 data configurations, each split into 4 files, training RSSI, training labels, evaluation RSSI, and evaluation labels. The data in these scenarios reflect the interpolation approaches used and evaluated of this

Tab. 2.5: RP naming convention as marked in the office.

RP	**X** [m]	**Y** [m]	RP	**X** [m]	**Y** [m]		RP	**X** [m]	**Y** [m]
A1	0.85	0.1	**B1**	1.85	0.1	-	-	-	-
A2	0.85	1.1	**B2**	1.85	1.1	-	-	-	-
A3	0.85	2.1	**B3**	1.85	2.1	-	**P3**	15.85	2.1
-	-	-	-	-	-	-	-	-	-
A11	0.85	10.1	**B11**	1.85	10.1	-	**P11**	15.85	10.1

dataset. The code names of each scenario are described in Table 2.6.

- **Processed_Data** - Combined data from raw packet capture files in CSV format. The combined data contain X and Y coordinates and 5 RSSI measurements corresponding to each ESP32-based sniffer.
- **Documentation** - Contains documentation with visualization of the office space and description in README.md file.

For the collection of this dataset, 5 ESP32 MCUs with sniffer firmware described above were used, and placed around the office. The first 4 were placed in each corner and the last 1 in the center of the room at approximately equal distance to the sniffers placed in the corners of the office. RSSI by nature fluctuates, and using more sniffers helps reduce the influence of the fluctuations in RSSI by having more samples.

The dataset was collected as supplementary materials for journal publication [118] described in Section 4.3. Subsequently, to completely reproduce the analysis, get the results, and recreate the figures, all software components (data analysis scripts and ESP32 firmwares) are available at GitLab [143]. The dataset was published through the Zenodo repository under the A-WEAR community [153].

Tab. 2.6: Description of dataset code names.

Data Name	Scenario Description
GPR00	Only measured data, 50 samples per reference position
GPR01	Measured data with empty spots filled using Linear interpolation, 50 samples per reference position
GPR02	Gaussian Regression trained only on measured data - 1m output grid, 50 samples per reference position
GPR03	Gaussian Regression trained only on measured data - 0.5m output grid, 50 samples per reference position
GPR04	Gaussian Regression trained on linearly interpolated data - 1m output grid, 50 samples per reference position
GPR05	Gaussian Regression trained on linearly interpolated data - 0.5m output grid, 50 samples per reference position
GPR06	Gaussian Regression trained selection of linearly interpolated data - 1m output grid, 50 samples per reference position
GPR07	Gaussian Regression trained selection of linearly interpolated data - 0.5m output grid, 50 samples per reference position
GPR08	Gaussian Regression trained only on measured data - 1m output grid, 1 sample per reference position
GPR09	Gaussian Regression trained only on measured data - 0.5m output grid, 1 sample per reference position
GPR10	Gaussian Regression trained on linearly interpolated data - 1m output grid, 1 sample per reference position
GPR11	Gaussian Regression trained on linearly interpolated data - 0.5m output grid, 1 sample per reference position
GPR12	Gaussian Regression trained selection of linearly interpolated data - 1m output grid, 1 sample per reference position
GPR13	Gaussian Regression trained selection of linearly interpolated data - 0.5m output grid, 1 sample per reference position

3 Exploiting Wi-Fi Management Frames for Presence Detection

This chapter focuses on the detection of human presence in using passive monitoring of nearby Wi-Fi network traffic.

Section 3.1 summarizes related research into the presence detection and privacy exploits using Wi-Fi. It also includes a description of countermeasures that were implemented to protect the privacy of users.

In Section 3.2, small-scale presence detection in a university office over one week is explored. The collected Wi-Fi management frames were used for this task. As part of this evaluation, the temporal pattern analysis was also presented.

Section 3.3 is a case study conducted at international conference IPIN 2021. And it focuses on the occupancy analysis of the conference space in regard to the conference program. The tracking of user presence in proximity to the sniffer is also analyzed.

The main focus of Section 3.4 is different outlook on presence detection. Instead of focusing on individual users, this section analyses room occupancy from Wi-Fi management frames collected by the ESP32 MCU.

3.1 Background

The analysis of privacy weaknesses in management frames of the 802.11 protocol is not new and has been explored for user tracking in the past. Out of the management frames, probe requests are most vulnerable to tracking.

In 2012, the authors of [154] exploited the availability of unique identifiers (MAC address) in the probe request for urban mobility tracking. Before the introduction of privacy measures considering Wi-Fi probe requests, specifically MAC address randomization in 2014 by Apple in the operating system iOS 8 [155], the Sapienza Probe Request Dataset was published [156], [157]. Several researchers already proved the vulnerability of probe requests before the implementation of MAC address randomization in [158], [159].

Since the introduction of MAC address randomization, researchers focused on exploring ways to bypass the newly introduced privacy measures and revealing the globally unique MAC address of each device using locally assigned MAC address [160]. One year after the introduction of MAC address randomization, researchers worked on reverse-engineering the MAC address randomization Apple had used [161]. Researchers also captured probe requests of Wi-Fi users in Italy during two political events and focused on figuring out the origin of the participants [162]. Their results were very closely matching the official voting reports. Other forms of analysis used

temporal differences between subsequently transmitted probe requests to distinguish different devices [151]. The authors of [74] compiled a very comprehensive study of privacy-related measures in probe request frames of 802.11. This study did explore when exactly the MAC address randomization is not enough and when exactly it fails to protect user privacy. In 2020, the authors of [163] successfully explored ways of protecting user privacy by encrypting probe requests, but the encryption of probe request frames was not adopted by the industry yet. As a follow-up to [74], another deep study of MAC randomization was published [75], which provides a deep look into the progress of protecting user privacy over the years.

The industry introduced the MAC address randomization in 2014 [155], but its implementation is still not perfect as sensitive information (enabling user's tracking) is still leaked.

3.1.1 Current MAC Randomization Implementation

Implementation of randomizing MAC addresses varies depending on the manufacturers and software developers. This fragmentation in implementation exists due to the lack of a commonly followed standard for MAC randomization. Few years after the MAC randomization was introduced, the specification of a standard amendment 802.11aq-2018 [164] was specified by IEEE SA Standards Board in 2018, but the implementation itself differs between manufacturers.

Identification of Randomized MAC Address

Even with the fragmentation in implementations, all implementations follow the setup of the two least significant bits in the first byte of MAC address as shown in Fig. 2.15. In case the second least significant bit of the first byte **B1** is set, the MAC address was assigned locally by the network controller of the device. The least significant bit of the first byte **B0** distinguishes between individual devices and device groups. Since randomized MAC address will always have bit **B1** set to 1 and individual devices have bit **B0** cleared to 0, recognition of randomized MAC address is simple due to the fixed values of the least significant bits (2, 6, A, or E) as previously mentioned in Section 2.5.6.

Google Android

The MAC address randomization was supported by Android since version 6, although the implementation of randomized MAC addresses for probing was first included with Android 8 [165]. Android 9 first implemented an option to connect to a Wi-Fi network using a randomized address, even though it was disabled by default and only available

through developer options. Starting with Android 10, MAC randomization is enabled by default [165] for every new network and randomizes the MAC address for every SSID. The randomized address does not change as the generated address depends on the network profile (SSID, security, and so on), non-persistent randomization that might change the MAC address for every connection to a network can also be enabled in the developer options [166].

In the first versions of Android supporting MAC address randomization, the system randomized only the last 3 bytes of the address while using a fixed prefix for the first 3 bytes. Android devices using this older MAC randomization implementation had their randomized MAC address starting with DA:A1:19 prefix [74].

There is one catch to this: even though the latest version is Android 13 at the time of writing (March 2023), barely any device is actually running it. Even though the majority of devices are running Android 10 or higher, there are still a lot of devices running older versions of Android, which do not use MAC randomization or any other privacy measures when it comes to probe requests.

Microsoft Windows

Similar to Google's Android, the operating system developed by Microsoft supports MAC address randomization per SSID. The MAC randomization is present in Microsoft's operating system since version 10 [167] and is turned on by default. The default option uses the same randomized MAC address for each SSID, and the MAC address stays randomized for the actual connection to the access point. This approach is the same as in Android and helps keep MAC address-based authentication working. On the other hand, there is an option to change the MAC address daily, but this needs to be enabled manually as this approach could cause issues with connection to some networks using MAC address authentication. The implementation of MAC address randomization in the newest version of the system developed by Microsoft stayed the same as in Windows 10, with MAC addresses staying the same for one SSID.

Apple iOS

Apple first introduced MAC address randomization with iOS 8 in 2014 [155]. Then, later on, iOS 10 added a tag in the information element of the probe request, which allowed for simple identification of iOS devices. The last change to the implementation of MAC address randomization happened in iOS 14. Since iOS 14, the devices use randomized MAC address per SSID just like Microsoft Windows and Android devices running the latest versions of their own systems. The implementation of MAC address randomization on iOS 15 [168] changed a little. The modifications were

related to the changes in the locally assigned address of the device for one network. This happens on multiple occasions:

- on forgetting the network and reconnecting again,
- if the device did not connect to the network for 6 weeks,
- on-device content reset or network settings reset.

This all makes the implementation of MAC address randomization on Apple devices very robust while not breaking existing systems with MAC address authentication.

Other Devices

From other common operating systems, the implementation of MAC addresses again differs on the developer or the manufacturer of the device. Linux supports MAC address randomization since 2014, specifically kernel 3.18 [169]. For Apple devices, the support changes from device to device. The support for the non-iOS devices produced by Apple varies based on the generation of the product [170].

3.1.2 Motivation

Indoor positioning is not only determining the exact position of a user or a device. It can also be just detecting a presence of people in the *Area of Interest* (AoI) or in the proximity of a *Point of Interest* (PoI). There are many reasons why presence detection is useful. Be it for power-saving purposes (smart lighting, ventilation, heating, or air conditioning control), safety (knowing someone is in the building during emergency situations), advertising, and other purposes.

The approaches to presence detection can vary and depend on the used technology. Solutions to presence detection range from computer vision systems employing surveillance cameras [171], networks of motion sensors [172], *Passive Infrared* (PIR) and *Infrared* (IR) sensor arrays [173], [174], ultrasonic sensors [175], and by monitoring the communication of regular wireless devices. It is also possible to measure the energy consumption [176] in a room and evaluate the presence of humans by the increased power consumption.

Employing IR sensor arrays can be used to creation of human heat maps. This approach was explored in [173], the authors placed 8×8 IR arrays above doors. By counting the occurrences of increased temperature by a human presence, it is possible to count the number of humans occupying the room. In [174], the authors used the heat maps created by the IR sensors to roughly classify the room occupancy by density of people in the laboratory into the categories: zero, low, medium, and high occupancy.

Ultrasonic sensors can be used to calculate distance. By placing them on the desks [175] and measuring the distance to the nearest obstacle - a human sitting at the desk, rough room occupancy can be also evaluated. This approach can be enhanced by also putting accelerators on the chairs to identify user sitting in it.

Since most of the approaches require the creation of a new infrastructure, in the following sections, the focus is on presence detection employing Wi-Fi technology and passively sniffing management packets from the radio environment. Since Wi-Fi infrastructure is usually already in place, it is beneficial to use it to reduce the cost of presence detection systems.

In the following sections, the tracking of the presence of users through Wi-Fi management packets and the room occupancy estimations are presented.

3.2 Temporal Pattern Analysis Aided Tracking based on Wi-Fi

Having an internet connection on us as people move through the world has had a major impact on both our professional and personal lives. The practically constant connection to the internet, be it through cellular data or Wi-Fi, introduces a question of how much privacy, locational and otherwise, are people giving up. They are often giving up their privacy willingly for the use of services that make their daily lives easier. In other cases, they do not know who or what may be identifying and/or tracking them.

Devices using Wi-Fi for connecting to the internet are extremely common, with most people having at least one around them at all times, for example, mobile phones, smartwatches, laptops, and smart TVs. Since the majority of these devices are connected to the internet through wireless networks, the issue of privacy and device tracking on those networks should be something people are aware of. Our devices are communicating with the surrounding world using standardized protocols. For instance, a device in a IEEE 802.11 network is uniquely identified by the MAC address, which is used in all the messages involving the device. The device probe request is a type of wireless frame used to gather information about Wi-Fi access points in the proximity of a device. This is beneficial to the users as the device can identify and connect to a known access point without any user input to switch to another AP with better coverage in a large public Wi-Fi network, as well as help with increasing accuracy of geolocation navigation by checking nearby Wi-Fi devices and comparing the signal strength of detected access points with previously detected ones at the same location. These probe requests can be a major weak point of a Wi-Fi protocol since they allow for non-cooperative user tracking if the device does not use

enough privacy measures such as MAC address randomization.

Tracking using Wi-Fi protocols can vary as they can be used to determine the past whereabouts of users, current presence, or both. The past locations of devices can be determined if the devices are transmitting the PNL (list of the networks the device was connected to in the past), which can be matched to the location using access point databases [94]. The current presence tracking can be done using a fingerprinting approach or, in the case of devices without randomized MAC addresses, just by matching the globally unique MAC addresses of separate probe requests.

3.2.1 Analysis

A dataset to perform the analysis was collected to base the research work on up-to-date data. The dataset was described in Section 2.5.5. The collection of probe requests was done at the GEOTEC office for 6 days of December 2021. The office is in the corner of the 5^{th} floor and during peak times is occupied by about 15 researchers. The office space is visualized in Fig. 3.1. During that time, the sniffer collected 340 360 probe requests. The data was collected using an ESP32 MCU with a connected micro SD card for storage of the collected probe requests. The placement of the sniffer in the office is in Fig. 3.1, marked by a red point S. The firmware created for the ESP32 MCU to capture probe requests is described in Section 2.5.1.

About 10 % of captured probe requests contained WPS sections, which provide additional information about the device, starting with the device name, manufacturer, and model. Since many devices use the name of their owner, this may pose a privacy leak in devices transmitting this additional information, which is unnecessary for the correct functioning of probe request frames. Even sending a device manufacturer name can reveal the user identity if the device itself is less common than others (e.g., Susie is the only Motorola user here). The most important issue of this WPS section, though, is the presence of UUID-E data, which is unique for a device since it is acquired using the globally unique MAC address of the device and does not change. Devices transmitting probe requests containing UUID-E are then easily localized as their globally unique MAC address can be recovered using UUID-E reversal techniques - by looking up the globally unique MAC address from hash tables [160]. Therefore, the additional information provided in the WPS Section was hashed to ensure the privacy of users.

In the past, the tracking of mobile devices using only probe requests was not very difficult as there were several factors that made the identification of a single device fairly straightforward. These include non-randomized MAC addresses, consecutive Sequence Numbers, common time differences between 2 probe requests, or data in the Information Element like supported transfer rates and vendor information.

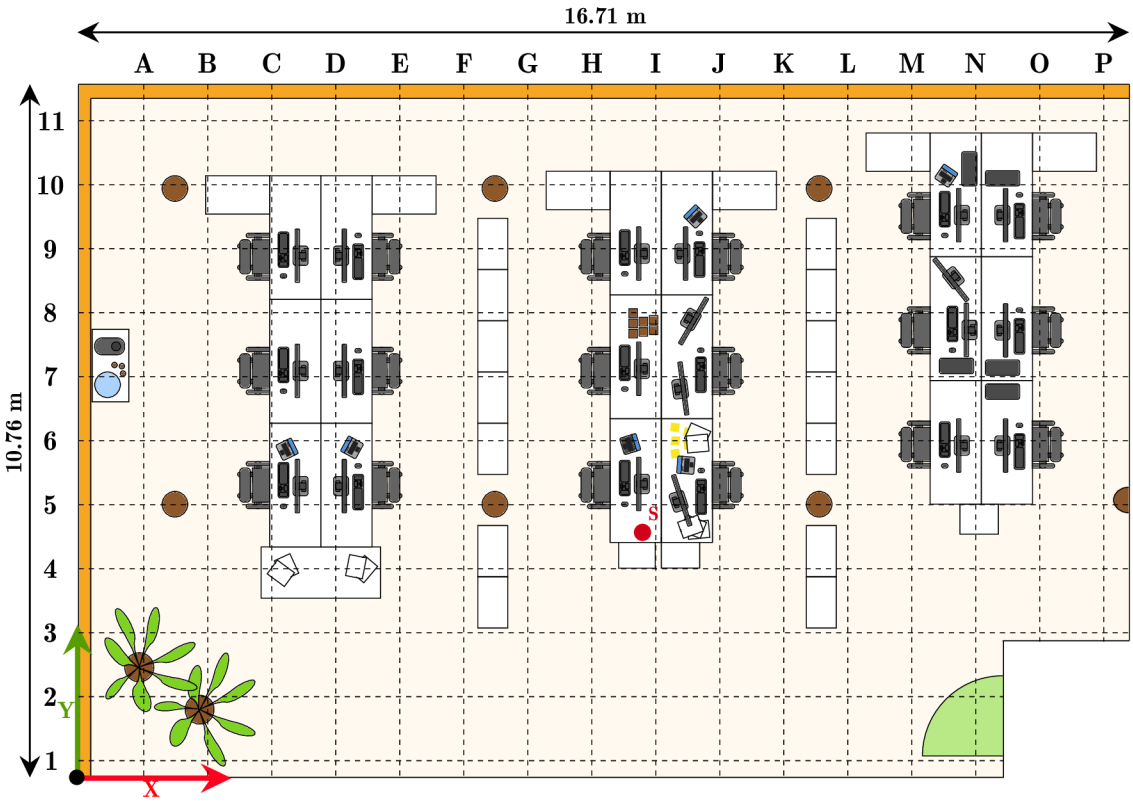


Fig. 3.1: Floor plan and location of sniffer in the office space of GEOTEC department at UJI, Spain.

MAC addresses

Even though MAC addresses cannot be used effectively to locate most modern devices, they can still be used to identify a single device during a single scan. From the analysis of the data collected at the GEOTEC office, the devices do not randomize MAC addresses after every probe request. This makes identification of a single scan instance from one device very easy since they keep the same address for the scanning sequence or multiple sequences. A solution to increase privacy would be to randomize the MAC address for every probe request, or at least more often than the devices do at the moment.

Sequence Numbers

Sequence numbers in probe request packets allow for another opportunity to easily identify packets coming from a single device during one scan instance, without the need to check the MAC address. The reason for this is the incremental nature of sequence numbers in probe requests coming from a single device. Every time the device sends a packet, the sequence number increments by 1. The sequence number can increase by more than 1, which happens if a device sends another packet or

frame between 2 subsequent probe requests. Addressing this issue would be fairly straightforward by using a random sequence number for each probe request. This, combined with randomization after every probe request, would make the identification of packets coming from a single device a lot more challenging as new techniques for identification through probes would be required.

Fingerprinting with Information Elements

To identify individual devices, the device-specific information fields available in probe requests are analyzed, out of which a single unique identifier is created [177]. The information element fields in probe requests can contain various additional data, starting with supported transfer speeds, and information about the vendor of the wireless chip inside of the device, including the connected peripherals and device name. As mentioned in the section describing the dataset, WPS information might also be present, which contains enough information to create a unique fingerprint of the device. The biggest issue there is the UUID-E field, which is unique per device and makes MAC address randomization pointless in devices that transmit WPS data since instead of MAC address the UUID-E can be used to identify a device. And that is without considering UUID-E reversal techniques, which can be used to determine the globally unique MAC address of the device [160].

For fingerprint creation, all of the fields that remain constant for one device between transmissions are used. Supported transmission speeds, vendor information, WPS field, and others are used to create a hash using the SHA512 algorithm. This ensures every combination of values has a unique fingerprint. All of the fields in the device fingerprint are presented in Table 2.3, with the frequency of occurrences in the data collected at the GEOTEC office. As can be seen, the supported data rates are presented in 100 % of collected probe requests, with an extended list of supported data rates being missing from just 0.05 % of the probes. The HT Capabilities (802.11n specific information regarding supported frequency bandwidth, etc.) were also present in a majority of probe requests, followed up by extended capabilities and at least 1 vendor-specific field, though the most common number of vendor-specific fields for one probe request in the data collected in the GEOTEC lab was 4, in about 30 % of all probe requests. Since devices from the same vendor will have the same vendor-specific fields, having 4 vendor-specific fields the same increases the probability that the devices are the same.

Fingerprinting SSID Lists

By using the previously mentioned techniques, it is easy to differentiate all probe requests sent by a single device in a single scan instance. Knowing that all probe

requests came from a single device then allowed for compilation of PNL with all the different SSIDs in captured probe requests. By using sets with each SSID represented only once, it is possible to use set similarity as described in Equation 3.1 to calculate a probability of two devices being in fact a single device by using the transmitted SSID list.

$$p = \frac{\text{set}(A) \textbf{ and } \text{set}(B)}{\text{set}(A) \textbf{ or } \text{set}(B)}. \tag{3.1}$$

There is also a possibility for the attacker to identify the users directly through the SSIDs from the PNL, as there is a possibility to match some of the networks directly to people (for example, SSID of the network at university in another country while the adversary knows there is the only one person around that used to study there).

Device Identification

Combining the use of non-randomized MAC addresses, device fingerprint elements, use of transmitted SSIDs to differentiate devices, and UUID-E available in the probe requests with WPS field results in enough information to identify a single Wi-Fi scan instance (Algorithm 3) as well as the reappearance of a device. Even with MAC randomization, the information elements in the probe requests allow the adversary to identify devices.

After identifying the Wi-Fi scan instances, device identification can begin. Firstly, there is a check if the MAC addresses of 2 separate instances are the same. If they are the same, they can be considered instances belonging to the same device. If the MAC addresses are randomized or different from each other, the presence of the WPS field is checked. In case of its inclusion, the UUID-E field can be used in place of MAC address as a unique identifier, with the UUID-E, it can be evaluated if the device is the same or not. In case the WPS field is not included and MAC addresses are not matching, it is necessary to determine the similarity using the information elements section of probe requests and calculate a similarity score between the two PNLs. If the similarity is higher than a set threshold, the 2 scan instances can be considered to be from the same device. Since the PNL revealed through probe requests in the majority of the cases is quite short and in many cases can be incomplete, the threshold was set to >0.5. Since with two transmitted SSIDs in each scan instance, one identical SSID will result in a similarity of 0.5. And since Wi-Fi networks have quite unique names, at least two matching SSIDs can be considered the same device. The process of identifying a single device is shown in Algorithm 4.

Algorithm 3: Scan Instance Identification.

```
variables
    probe.mac, MAC address of the instance
    probe.has_wps, Instance with WPS field
    probe.uuid-e, UUID-E of the instance
    probe.ie, Information Element of the instance
    probe.sn, Sequence number of the probe request
end variables

if probe_1.mac = probe_2.mac then
    if probe_1.has_wps = probe_2.has_wps then
        if probe_1.uuid-e = probe_2.uuid-e then
            return True                          ▷ True - Same instance
        else
            return False                         ▷ False - Different instance
        end
    end
    if probe_1.ie = probe_2.ie then
        if probe_1.sn < probe_2.sn < probe_1.sn + 5 then
            return True                          ▷ True - Same instance
        else
            return False                         ▷ False - Different instance
        end
    else
        return False                             ▷ False - Different instance
    end
else
    return False                                 ▷ False - Different instance
end
```

Temporal Pattern Analysis

One of the more difficult parameters to mask for a single device sending multiple probe requests is the time difference between 2 probe requests. From the analysis of the Sapienza Probe Request dataset [157], slightly more than 98 % of subsequent probe requests sent by a single device are transmitted less than 65 ms apart. These bursts of transmitted probe requests can be used for fingerprinting of the device. This is useful in conjunction with incrementing sequence numbers to distinguish two different devices and will be a potential threat to the users in the future since

Algorithm 4: Device Identification.

variables
  $instance.mac$, MAC address of the instance
  $instance.has\_wps$, Instance with WPS field
  $instance.uuid\text{-}e$, UUID-E of the instance
  $instance.ie$, Information Element of the instance
  $instance.SSIDs$, List of SSIDs from 1 instance
  $threshold$, Minimum similarity threshold
end variables

if $instance_1.mac = instance_2.mac$ **then**
  return *True* ▷ `True - Same device`
else if $instance_1.has\_wps$ **and** $instance_2.has\_wps$ **then**
  if $instance_1.uuid\text{-}e = instance_2.uuid\text{-}e$ **then**
    return *True* ▷ `True - Same device`
  else
    return *False* ▷ `False - Different device`
  end
else if $instance_1.ie = instance_2.ie$ **then**
  $p = \frac{\text{set}(instance_1.SSIDs) \textbf{ and } \text{set}(instance_2.SSIDs)}{\text{set}(instance_1.SSIDs) \textbf{ or } \text{set}(instance_2.SSIDs)}$
  if $p > threshold$ **then**
    return *True* ▷ `True - Same device`
  else
    return *False* ▷ `False - Different device`
  end
else
  return *False* ▷ `False - Different device`
end

the incrementing sequence number could reveal one device employing MAC address randomization after every probe request.

The proposed temporal analysis does not use the time difference between two probe requests [151]. Since devices do not change their MAC address during the scan instance, this is not necessary. Instead, the proposed approach to time analysis is different. It uses all of the similar device data from device identification, analysis of the recurrent appearances of each device, and possible similarity to others. This way, the underlying patterns can be discovered, which then allows us to identify cases where a single device looked like several devices. This can be achieved by considering

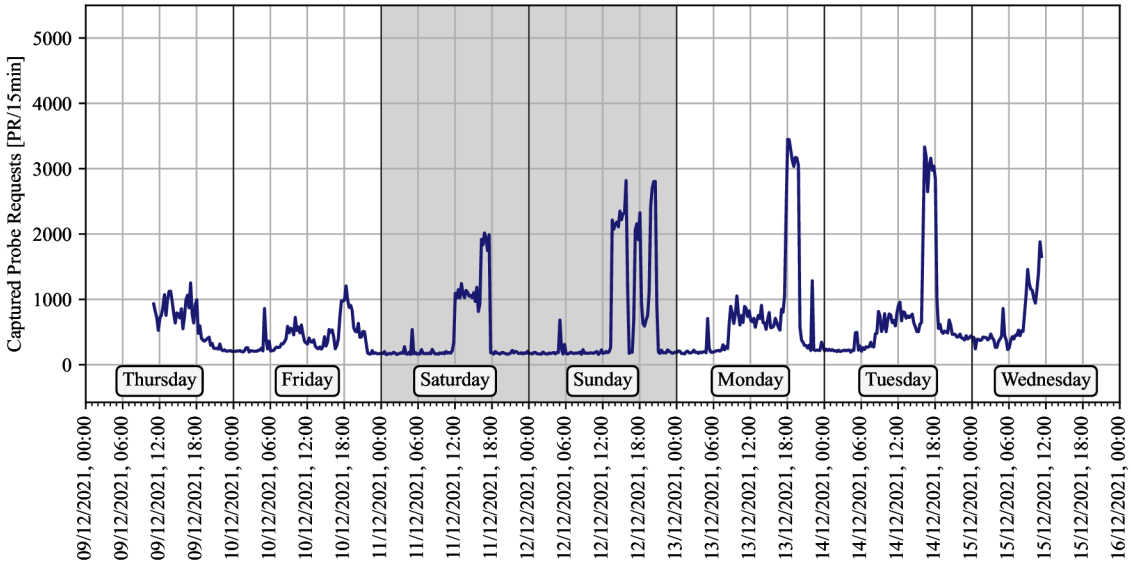


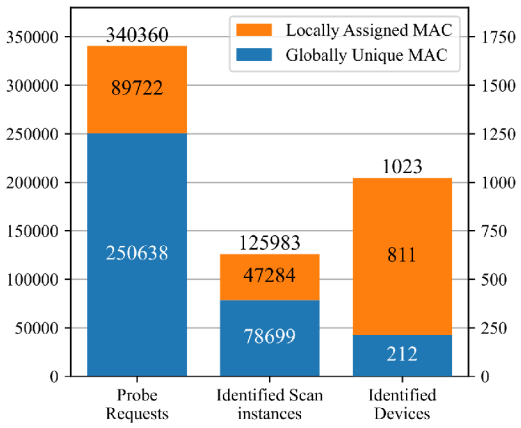
Fig. 3.2: Density of captured Probe Requests over time in the office space of the GEOTEC department (amount of probe requests grouped in 15-minute clusters).

scan instance appearances of one device and clustering them together based on time. Then, the number of clusters between devices was compared. In case two devices had the same amount of appearance clusters, the overlay similarity between clusters was compared. From this comparison was then decided if the devices were, in the end, a single device misidentified as many, or to skip it and move to the next device.

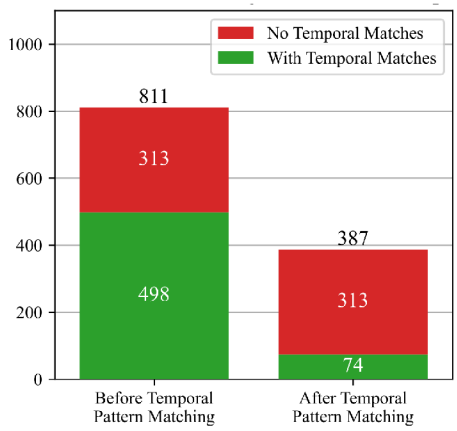
3.2.2 Results

Throughout the week, the sniffer captured 340 360 probe requests. This number is increased above normal due to other experiments with mobile devices (the UE used in the experiments was using BLE and was not connected to Wi-Fi, however, the Wi-Fi was not turned off) being run in the office on the weekend and in the late hours of Monday and Tuesday. This increased peaks in the distribution of probe requests, which is shown in Fig. 3.2. From the distribution is clearly visible that some devices in the office are running without interruptions, as the amount of counted probe requests never reaches zero during the 15-minute window. Additionally, the increases in captured probes at the beginning and drops at the end of the workdays are nicely visible.

From the 340 360 probe requests collected at the office, the Algorithm 3 identified in total 125 983 scan instances. As a follow up, the Algorithm 4 identified 1023 devices, as is represented in Fig. 3.3a. As a single instance was counted any single probe request or burst of probe requests according to Algorithm 3. These instances were then clustered based on their similarity following the Algorithm 4. This way, it



(a) Randomized MAC addresses in probe requests, identified scan instances and devices.



(b) Identified devices with locally assigned MAC address before and after temporal pattern matching.

Fig. 3.3: Dataset information and device identification in the office space of the GEOTEC department.

was possible to match at least two instances to a single device. If the tested instance showed no similarity to others, that instance was discarded as a single instance device that was no way to track or locate.

For devices that do not randomize their MAC addresses, the tracking is very effective and it is easily visible when the UE was inside of the GEOTEC office or in its proximity. The reason is that the unique identifier is the MAC address, which never changed. Due to this, Algorithm 4 was able to identify a significant number of devices that could be easily tracked and analyzed for presence patterns. Between those devices were also a few that never left the proximity of the probe request sniffer as well as some that showed up in monitored range only for a few minutes. As presented in Fig. 3.3a, 212 devices did not use MAC randomization, and the example of presence in time for 8 such devices is shown in Fig. 3.4.

The identification of devices randomizing MAC addresses is more complicated, but despite MAC randomization making the process more difficult, it was possible to identify many devices using the techniques mentioned before in the Analysis section. The results of the analysis can be seen on 8 devices using randomized MAC address in Fig. 3.5. Even with the more complicated approach to identification, from the resulting data, the analysis of user presence is still possible.

The Algorithm 4 provides the identified instances clustered as one device. The instance matching is not 100 % accurate and, in some cases, especially in those considering devices with randomized MAC addresses, can misidentify a single device as several devices. The Algorithm 4 matched the 125 983 instances to 1023 devices,

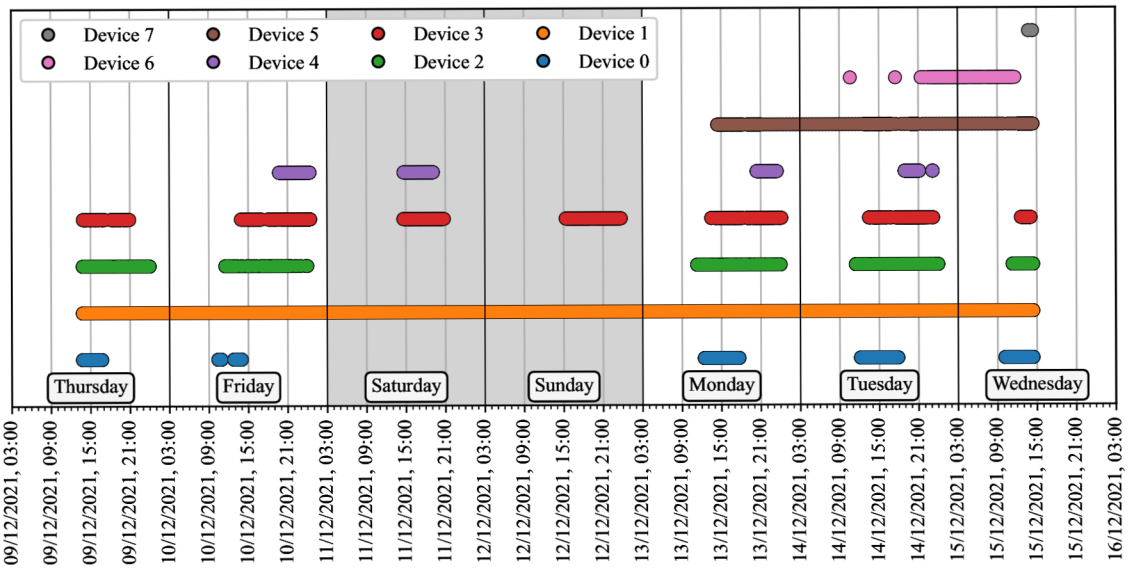


Fig. 3.4: Occurrence of several devices identified by the usage of globally unique MAC address in the office space of the GEOTEC department.

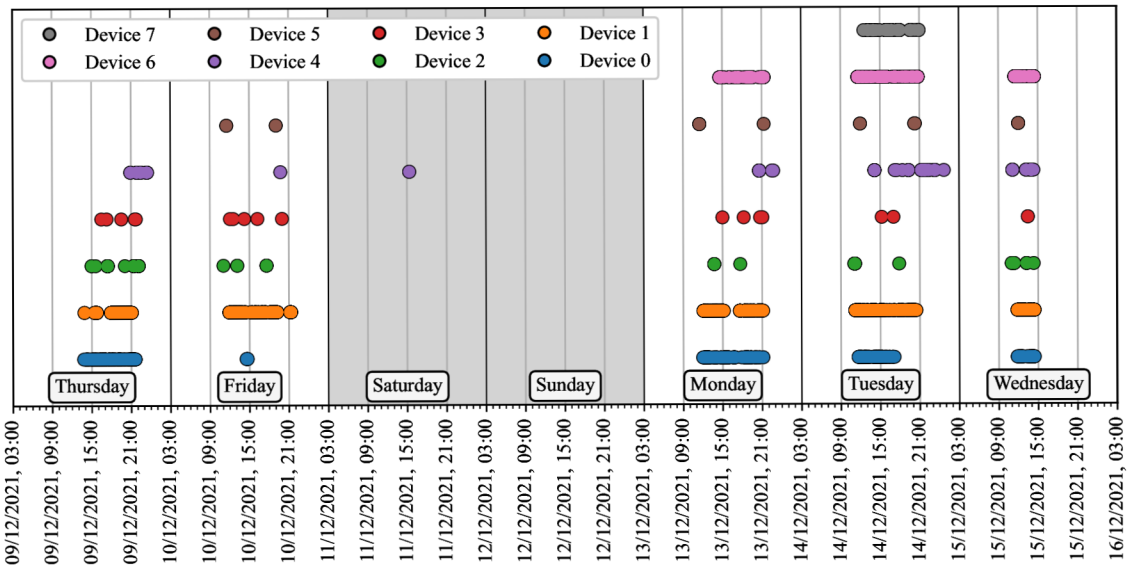


Fig. 3.5: Occurrence of several devices identified despite the use of MAC randomization in the office space of the GEOTEC department.

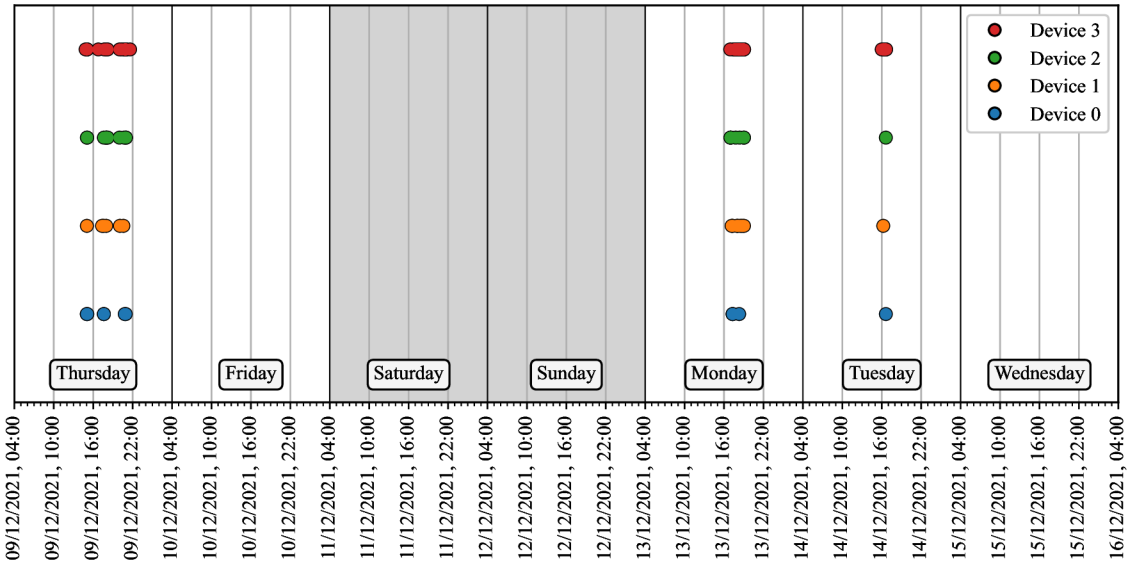


Fig. 3.6: Occurrence of single device misidentified as multiple devices, later identified as a single device through the similarity in temporal patterns in the office space of the GEOTEC department.

which can be seen in Fig. 3.3a. After instance matching, temporal pattern matching became possible. It managed to detect similarity among 498 misidentified devices and reduce this amount to only 74 devices using MAC randomization. 313 devices with locally assigned MAC addresses did not match temporal patterns of other devices. The number of devices with locally assigned MAC address before and after temporal pattern matching can be seen in Fig. 3.3b. The probe request transmission patterns were quite closely matching each other, as can be seen in Fig. 3.6, which led to identifying these appearances as a single device or single user carrying multiple devices.

3.3 Presence Analysis with Wi-Fi Probe Requests

The attendance at the international conference IPIN 2021 provided an idea for a case study. As the conference focuses on indoor positioning and indoor navigation, the case study focused on presence detection during the conference. The data and results of this study were then presented at the same conference as they were collected at, but one-year later [146], at IPIN 2022.

3.3.1 Analysis

Similarly to the work based in the office, which was described in Section 3.2, probe requests were collected using ESP32 based sniffer described in Section 2.5.1. The

collected dataset was presented first in Section 2.5.4 as part of the reproducible research.

The conference took place in Lloret de Mar, Spain in Evenia Olympic Congress Centre from 29 November to 2 December 2021. The only people present around the hotel lobby and near the session rooms from the beginning to the end of the conference were attendants of the conference, conference organizers, hotel employees, and cleaning staff.

The entire conference space was around the lobby, with hotel rooms and hotel restaurants being far enough to not pose interference and capture probe requests from unwanted sources. Similarly, the location of the conference was in a single-floor section of the hotel complex, which guaranteed that all of the collected probe requests were from the area of the actual conference. The probe request sniffer was placed under the stage in Session Room 2, as presented in Fig. 3.7, which was in the middle of the session rooms and close to the lobby. The entire conference space was in the radio range of the sniffer.

To analyze the gathered probe requests, the data were looked at from two perspectives. At first, the gathered probe requests were used for crowd presence detection. Secondly, they were used for the analysis of the impact of MAC randomization on the ability to track individual users, both with MAC randomization on and off.

3.3.2 Results

The results are split into several categories. At first, the presence of users in the proximity of the sniffer is evaluated. This follows by tracking the presence of individual users based on the probe request frames. For this, the fingerprinting of information elements introduced in Section 3.2 is used.

Presence Detection

To see how big of a crowd could be detected from the density of probe requests, the amount of captured probe requests was counted every couple of minutes. The chosen interval is 15 minutes as a compromise between readability and resolution of the final plot. From Fig. 3.8, it is visible that during every keynote, tutorial, or session, the presence of users was much higher compared to the breaks in between the sessions. This can be caused by people turning off computers for the duration of coffee breaks. Quite a lot of people also left the range of the sniffer to go into the hotel restaurants for lunch. During the nighttime, it is also noticeable if someone stayed in the lobby, be it for work or socializing. During the night from Monday to Tuesday, the lobby stayed empty with someone going through it but not staying long. The next two nights, the lobby was not empty.

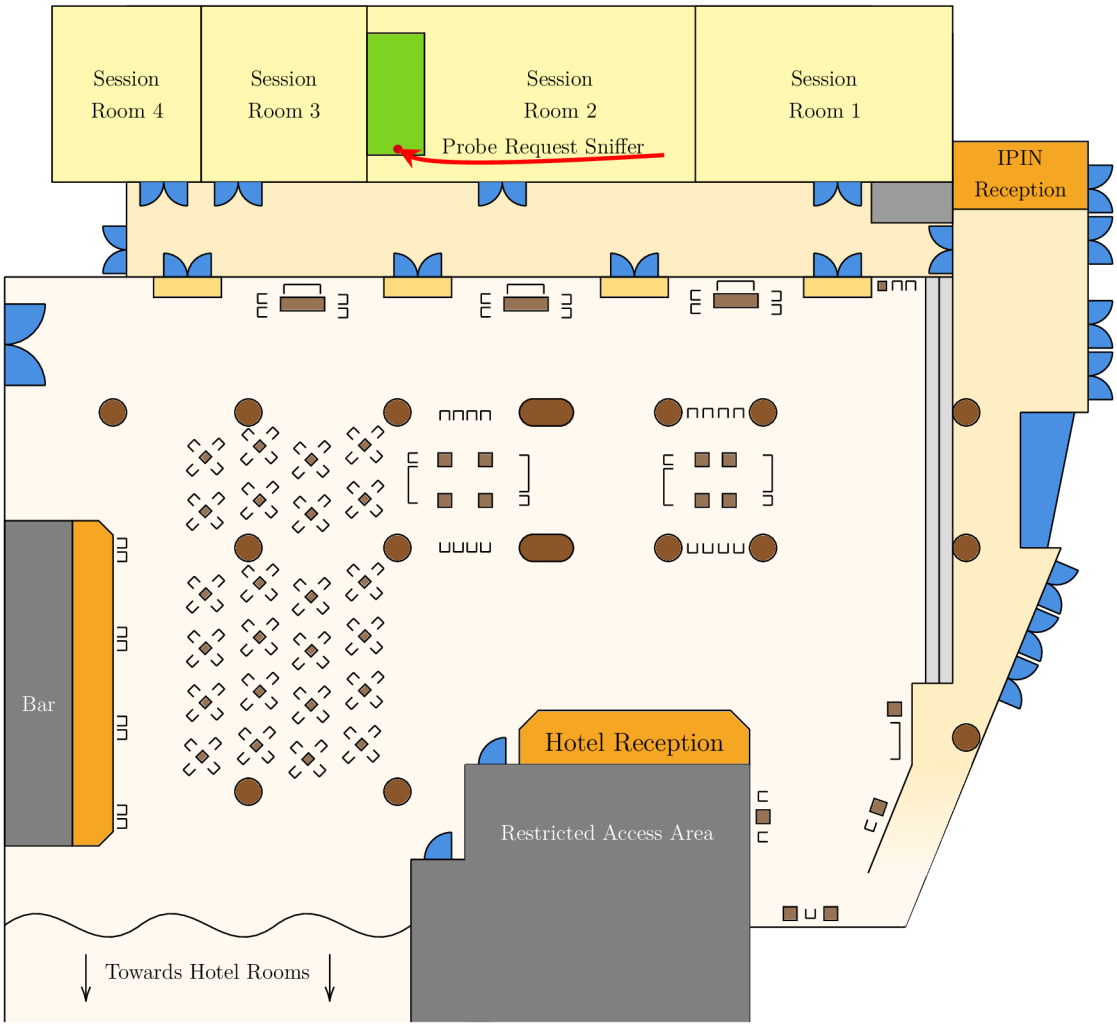


Fig. 3.7: Floorplan of the Evenia Olympic Congress Centre (Lloret de Mar, Spain).

From the data plotted in Fig. 3.8, it is also visible which keynote or session group (IPIN 2021 had 4 parallel session tracks) was more interesting to the participants of the conference. Unfortunately, due to the deployment of only one Probe Request Sniffer, it was not possible to implement an indoor localization method based on RSSI to determine which session room the participants of the conference occupied at any time.

The Tuesday social event (Networking in the Kitchens) took place mostly out of the range of the probe sniffer in one of the hotel's restaurants. After the event, some of the participants stayed for further socializing, which can be seen on the small local peak right after the event ended. Another drop in received probes happened on Wednesday during the gala dinner, which took place in a neighboring village, and the presence in the conference space was minimal.

One of the noticeable trends is also the drop in the amount of captured probe requests during coffee breaks. This indicates people leaving the area either to get

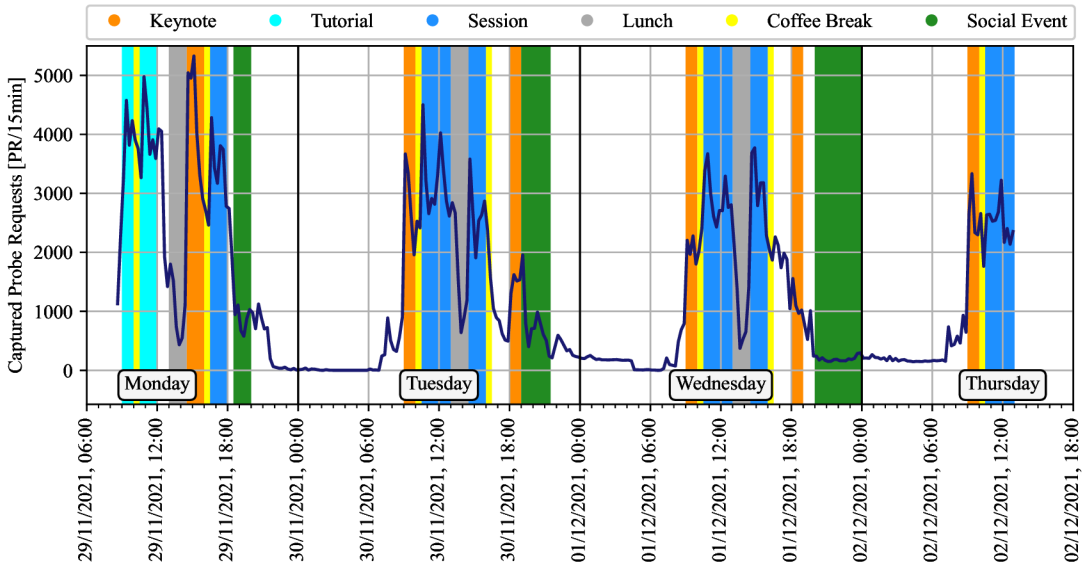


Fig. 3.8: Density of captured Probe Requests correlated with the program of the IPIN 2021 conference (amount of probe requests grouped in 15-minute clusters).

some fresh air outside of the hotel lobby, use the restroom, or go to their hotel rooms. Since the amount of probe requests increases after each coffee break again, it is safe to assume that the conference attendants were coming back after each of the coffee breaks was over.

Analysis of User Presence with Global MAC Address

It is no surprise that devices that transmit their real MAC address are very easy to track. Since it is possible to identify probe requests using their globally unique identifier, their identification is very simple. At the IPIN 2021 conference, 28.62 % of identified scan instances (58 393 of 204 038 scan instances) used their globally unique MAC address. Since these devices used their real MAC address, subsequently, the identified instances were clustered together and distinguished 229 individual devices without MAC address randomization through the duration of the conference. This data can be seen in Fig. 3.9. Afterward followed the exploration of the presence of these devices in the proximity to the probe request sniffer. This presence in time proved how easy it is to track devices that do not employ any privacy-related measures regarding the probe requests. The temporal presence of 10 devices using their real MAC address in the conference space is in Fig. 3.10 as an example of the simplicity of tracking these devices.

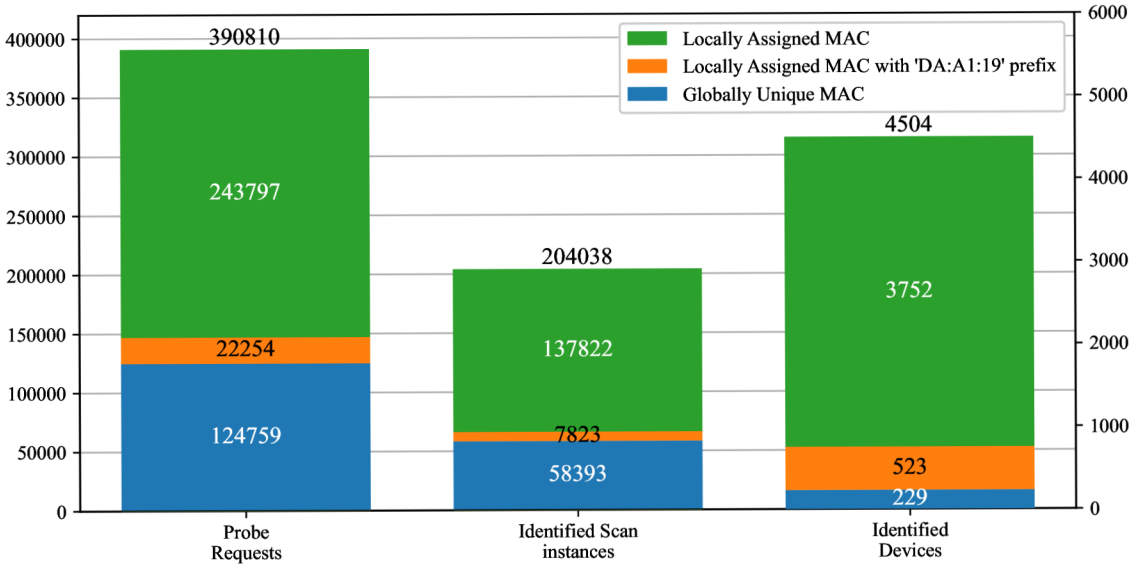


Fig. 3.9: Randomized MAC addresses in probe requests, identified scan instances, and distinguished devices at the IPIN 2021 conference.

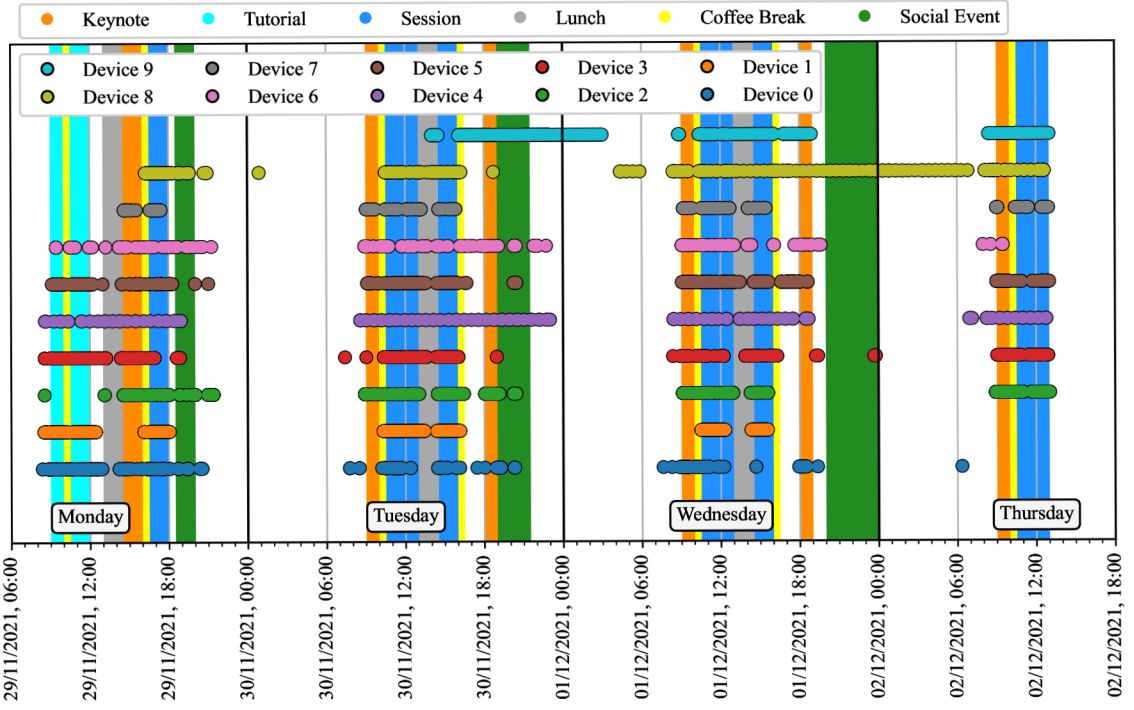


Fig. 3.10: Repeated occurrences of devices identified by the usage of globally unique MAC address at the IPIN 2021 conference.

Analysis of User Presence with Local MAC Address

On the contrary, tracking devices which do not transmit unique identifiers is much more challenging. Out of the captured probe requests, 68.08 % (266 051) were using locally assigned MAC addresses. Since the devices do not change the MAC address they transmit during the Wi-Fi search burst, using Algorithm 3 it was possible to identify 204 038 different scan instances using just the MAC addresses available from the captured probe requests.

Out of all the transmitted probe requests with randomized MAC address, 22 254 used *DA:1A:19* as the first 3 bytes of their MAC address. There were 7823 individual scan instances using this prefix. After matching these instances together using fingerprinting of information elements, the similarity between transmitted PNLs (Equation 3.1) and the recurrence of the same randomized MAC addresses, the analysis identified 523 devices using the *DA:1A:19* MAC address prefix. These data are presented in Fig. 3.9 with the comparison to the number of devices with a fully randomized MAC address and with a globally unique one. 50 of these devices then showed up more than 10 $\times$ (10 appearances proved to be a reasonable threshold during the analysis as devices with more than 10 appearances are easy to track over time).

After identifying individual scan instances, the Algorithm 4 was used to match together all other devices as well. This approach initially distinguished 4274 devices using locally assigned MAC address out of which 3752 randomized 46 out of the 48 bits in a MAC address. In this initial number of devices, 3544 appeared less than 10 times. On the other hand, 296 devices with fully randomized MAC addresses showed up more than 10 $\times$, which made them easily identifiable despite them using randomized MAC addresses, as can be seen from 10 example devices in Fig. 3.11.

Single Occurrence of Devices in Time Domain

From Fig. 3.9, it is visible that the number of identified devices is still really high for just 3 full days in a conference space. Especially since the event space was primarily occupied by the attendants of the conference and hotel staff. The sniffer was also in the range of the sidewalk next to the entrance of the hotel. It is quite possible many of the single occurrences were just from pedestrians walking in the proximity of the sniffer. Another reason for this is a good implementation of MAC address randomization, the transmission of reduced information elements in the probe requests, and omitting the transfer of SSIDs from the saved PNL. Representation of unmatched devices is shown in Fig. 3.12 with 10 examples.

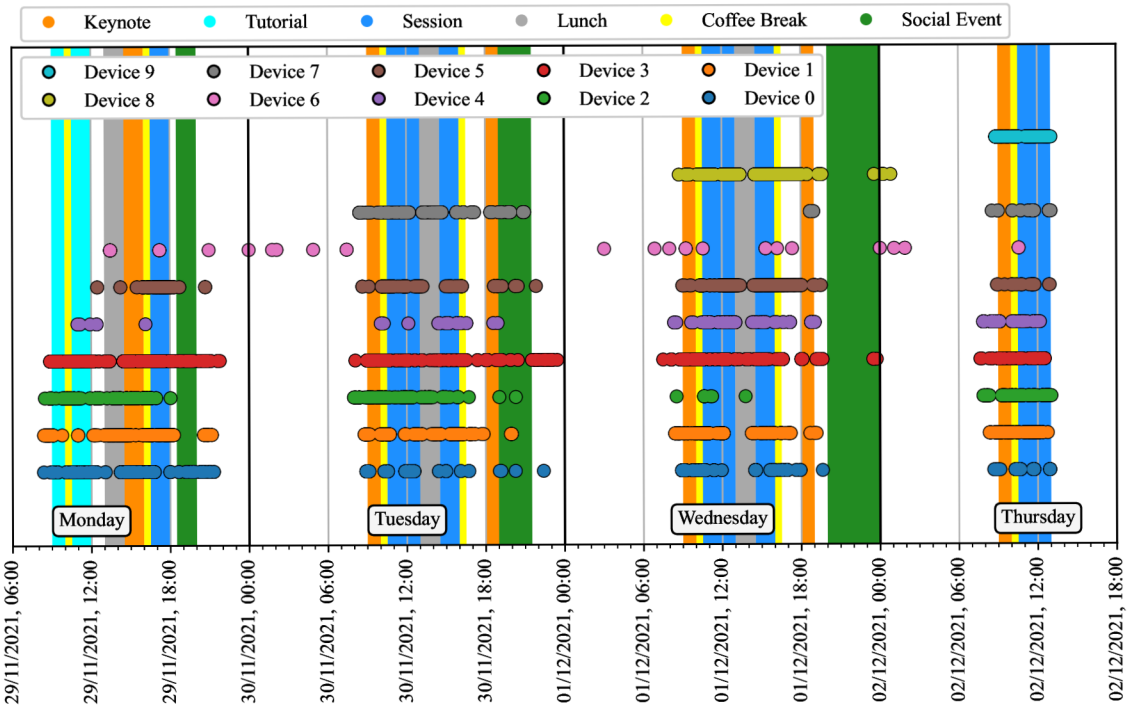


Fig. 3.11: Recurrent identification of the same devices despite using locally assigned MAC address at the IPIN 2021 conference.

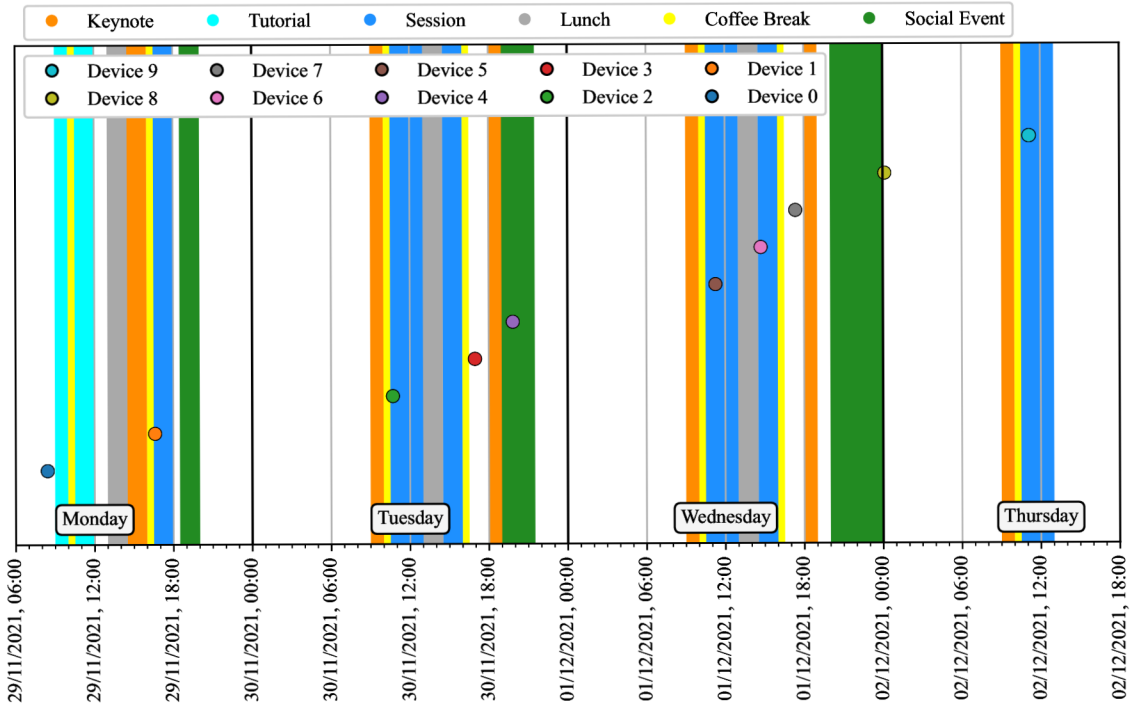


Fig. 3.12: Detected devices without identified recurrences in time at the IPIN 2021 conference.

3.4 Room Occupancy Detection Using Wi-Fi Probe Requests

Initially, the ESP32-based sniffer described in Section 2.5.1 did not implement the collection of radio information like RSSI or channel. After implementing this, the sniffer increases in capability drastically, as it can now roughly evaluate the distance between itself and Wi-Fi devices. This can be useful in several ways, as it can be used to roughly reduce the radius of AoI or in case of employing multiple sniffers using either trilateration or machine learning to infer the position of the UE. The work exploring room occupancy detection was presented [178] at the international conference MAREW 2023.

3.4.1 Analysis

The data were collected during five working days in three defined-sized rooms and scenarios: office, laboratory, and meeting room. At first, the signal strength distribution in all three locations was recorded to get the RSSI thresholds near the edges of the rooms. Therefore, it was possible to accurately remove non-interesting devices from the analyzed data. The values were measured 1 m above the floor and the ceiling height in all rooms was 2.9 m.

In relevant publications, machine learning algorithms are used to obtain occupancy rates. They are mainly kNN, SVM, and *Hidden Markov Models* (HMM) classifiers for distributing the number of people in rooms into several levels. So, unlike the research presented, they do not estimate the total number of people. The proposed algorithm for estimating room occupancy with low computational complexity is outlined in Algorithm 5.

Room occupancy is analyzed within a given time interval, the length of which can be set according to the intended application. At the beginning of the interval, the number of detected people in the room and also the list of detected mobile devices are reset. Each probe request is tested against its RSSI value and the device's MAC address is obtained from the probe request. The occurrence of each unique MAC address is accumulated within the time interval. After the selected interval, the frequency of MAC addresses is assessed. The targeted estimation of room occupancy is equal to the number of MAC addresses more frequent than the threshold.

3.4.2 Results

To support the algorithm's functionality for estimating room occupancy, three datasets of probe requests were created, collected during five days between October

Algorithm 5: Room occupancy estimation algorithm.

```
variables
    probe, Received probe request
    probe.rssi, Received Signal Strength Indicator
    probe.mac, MAC address of probe request device
    rssi_thr, RSSI threshold
    interval, Analyzed time interval
    interval.macs, Histogram of unique MACs
    occupancy, Occupancy estimation
    mac_thr, Threshold for one MAC occurrence
end variables

foreach interval do
    interval.macs = {empty: 0}          ▷ Initialize histogram of MACs
    interval.occupancy = 0                    ▷ Reset occupancy counter
    foreach probe do
        if probe.rssi > rssi_thr then
            interval.macs = {probe.mac: +1}
        end
    end
    occupancy = device_count(interval.macs(occurrence > mac_thr))
end
```

3rd and 7th, 2022 at the Department of Radio Electronics at the Brno University of Technology, Czechia. The data sets correspond to three scenarios: the first is from the office of one academic staff member (5.2 m × 3.5 m), the second is from a regular laboratory, and the third is a lecture room partly used as a spare meeting room (dimensions of both are 10.7 m × 6.8 m).

Within the entire measurement period, a total of 143 871, 265 797, and 281 661 probe requests were recorded in the office, laboratory, and lecture room, respectively. These requests came not only from the mobile devices of students and department employees but also from fixed wireless devices, as well as from maintenance employees or from devices situated on higher or lower floors. Based on the knowledge of the probe requests sniffer location and RSSI signal distribution, minimal limits of −56 dBm, −63 dBm, and −66 dBm were determined in test rooms, below which the probe requests were filtered. The number of processing requests was thus reduced to 4761, 29 827, and 12 262, respectively. All measurements were performed on an ESP32-based platform with an embedded camera module and custom firmware.

Tab. 3.1: Experimental results of occurrence estimation.

Room	Captured probes [-]	RSSI threshold [dBm]	Considered [-]	MACs [-]	RMSE [-]
Office	143 871	−56	4761	149	0.208
Lab	265 797	−63	29 827	4126	0.797
Meeting	281 661	−66	12 262	3765	3.389

Data processing and evaluation were done in the Python language.

The testing was carried out during normal university operations, such as laboratory exercises, lectures, and meetings, and was attended by more than 100 people. Due to the randomization of MAC addresses a total of 8040 unique addresses were captured during the recording of probe requests. For each scenario, MAC addresses were collected at time intervals in which the presence of a person in the room was decided. The best results were obtained when the number of accumulated addresses was set between three and five.

To assess the error rate of the proposed algorithm, the actual occupancy of the rooms was recorded. In the minimal-scale scenario, a questionnaire system was used, where the user wrote down the times when he entered and left the room. In the small and medium scenarios, a web camera was used to record the space for occupancy verification. For each time interval, the average number of people in the room was determined by video post-processing. The differences between the results of the proposed detection algorithm were compared to the actual occupancy using *Root Mean Squared Error* (RMSE). The summarized data, the selected parameters, and the resulting estimation error are shown in Table 3.1.

The results in the minimal, small, and medium scenarios, obtained by the proposed method (Estimated) and by real observations (Measured), are shown in Fig. 3.13. Note that the occupancy was analyzed in five-minute intervals. It can be seen that the number of mobile devices or persons in the office was most often equal to one. Occasionally, other people were also present in the office for short consultations. The resulting low RMSE error means a high degree of agreement between the proposed algorithm and reality. For small- and medium-scale scenarios, where larger rooms and numbers of people are considered, the difference between the estimated and actual number of people is greater. Specifically, it is 0.797 for the laboratory and 3.389 for the lecture & meeting room. However, the difference is not large and confirms the suitability of the used method for determining room occupancy and for implementing

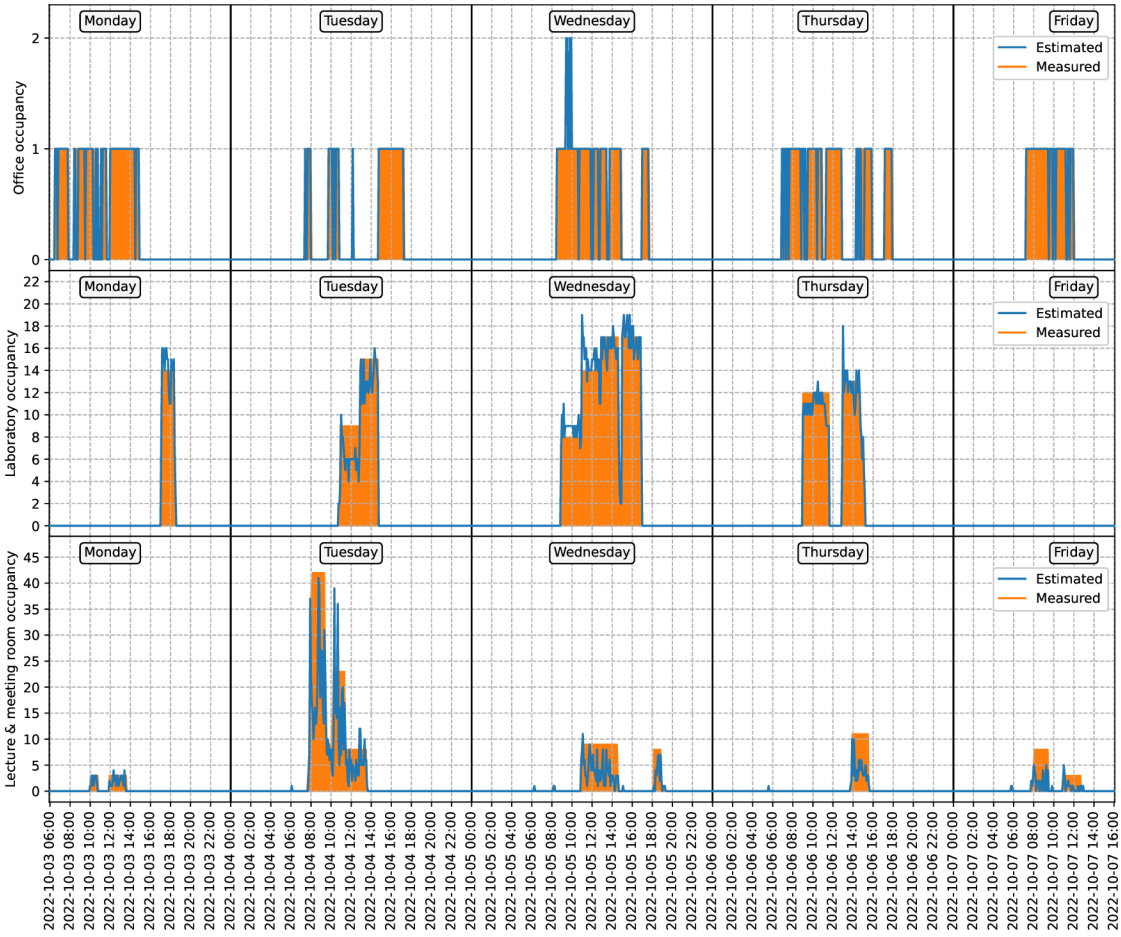


Fig. 3.13: Estimated and measured occupancy in minimal-scale office, small-scale laboratory, and medium-scale lecture & meeting room scenarios.

into smart building management systems and providing energy savings.

3.5 Discussion

In this chapter, the possibilities of passive presence detection and tracking in multiple scenarios as well as room occupancy estimation were explored. For this purpose, was used ESP32 MCU with custom firmware for sniffing Wi-Fi probe requests.

The analysis of the gathered probe requests was done in a few ways. At first, the probe requests and all the information included with them were used for fingerprinting and non-cooperative tracking. This was done in the university office and during the international conference IPIN 2021. During the analysis, the probe requests were used to match together devices that had the same MAC address, be it a globally unique one or a locally assigned one. Subsequently, the information elements were used for fingerprinting and similarity of PNLs to further identify different scan instances as

one device. Not surprisingly, the devices without a randomized MAC address were easily tracked. As expected, devices employing MAC address randomization were more difficult to track, but even those are, in many cases, possible to be tracked.

Even though the manufacturers are employing privacy-related measures like MAC address randomization since 2014, many devices are still easily tracked. That is not to say that MAC address randomization is not working, as there were many appearances of only 1 or 2 scan instances. For these devices, the implementation of locally assigning MAC address either works well or the sniffer also captured probe requests from outside of the AoI. It is expected, the situation will be getting better in time with older devices (with worse implementations of MAC address randomization) being replaced by newer devices after reaching their end of life.

The implication of this knowledge is the ability to easily track users without their knowledge. And the need for the industry to implement better privacy-preserving features in the future standard amendments of Wi-Fi.

Furthermore, this chapter focused on the determination of the room and building occupancy. The knowledge of occupied rooms in buildings can help managers of smart buildings in the efficient use of energy and in locating people in the event of an accident, natural disaster, or other catastrophe. As such, the study of occupancy detection in small and medium-sized rooms based on wireless communication of wearable devices can end up being used not only for energy saving but also for security reasons.

The presence of a person or device depends on the so-called probe requests, which are constantly transmitted by the device's Wi-Fi interface. The benefit of the proposed systems is their reliability, low price, and ease of installation in new as well as existing smart buildings. By connecting the designed detection system to the control of smart buildings, it will be possible to save valuable energy.

3.6 Summary

The presence detection and room occupancy analysis are summarized in the following paragraphs:

- At first, the presence detection and tracking of humans in the office was presented. This was achieved by sniffing Wi-Fi management packets from the radio environment, which was done in a completely passive way, that prevented any nearby device from detecting the sniffer. The collected frames were then used for temporal analysis and tracking by employing fingerprinting of data available in the unencrypted packets.

- Similar to the presence detection in the office, the case study evaluating the presence of conference participants near the session rooms was conducted. The conference IPIN 2021 was chosen as the place to conduct this study. From the gathered data, it was possible to again prove that non-cooperative tracking using Wi-Fi is possible and improvement of privacy-related measures in the near future is necessary.
- Finally, the room occupancy estimation was done using the same management frames of Wi-Fi. The ability to estimate the occupancy of rooms is going to be an important function of smart buildings, to reduce the energy requirements for air conditioning, lighting, and air circulation. The use of existing infrastructure is also a good step in the integration into smart building systems.

The research presented in this chapter was previously published in the following publications: [146], [148], [178].

4 Balancing Accuracy and Complexity

This chapter focuses on the optimizations in machine learning approaches to reduce computational complexity while preserving or increasing the accuracy of machine learning computing at the edge and in positioning algorithms.

Section 4.2 describes a possibility to reduce the memory requirements of Convolutional Neural Networks. This is necessary for employing neural networks in edge computing due to the lower memory capacity of such devices. As such, the approach presented is simple and does not require re-training already existing models by reducing the precision of the data type used for storing weights.

In Section 4.3, the possibilities of using interpolation techniques to find a balance between the size of Radio Map, the time required to create Radio Maps, and the computational complexity of the final Indoor Positioning System are presented. For interpolation of Radio Map, the linear interpolation and the Gaussian Process Regression algorithms are used. There is also an extensive overview of 14 different scenarios using varying approaches to adjusting the measured data.

4.1 Background

Ever since the beginnings of digital computing, researchers tried to implement more computationally intensive algorithms. In the last few decades, a rapid increase in available computational performance in our everyday lives brought such algorithms to the forefront. Be it on our desktop computers, where the performance of both CPUs and GPUs grew exponentially with each new generation [179], or our smartphones, which are at the touch of a finger capable of doing tasks possible only on a supercomputer just two decades ago.

4.1.1 Neural Networks

A great example of such growth in the complexity of algorithms came with much more easily available devices capable of capturing digital photos and videos as well as powerful computers, as neural networks started to be employed in almost every field using data analysis. The field of computer vision was the one with the biggest jumps in complexity thanks to the possibility of using *Convolutional Neural Networks* (CNNs). Following the success of AlexNet [180], these types of networks redefined the computer vision field, as the neural networks can learn convolutional filters capable of finding patterns in a way that could be impossible to do by hand. Following in the steps of computer vision, neural networks became more and more prominent in other fields as well.

At the time, when neural networks for image processing started to gain in popularity, emphasis was, and partly still is, on the maximum accuracy of the result. This approach, to achieve the highest possible accuracy of the results, was encouraged by the annual ImageNet challenge that started with the introduction of the ImageNet dataset [181]. Focusing on accuracy did not have a positive impact on the computational performance of the majority of the networks. The size and complexity of the neural networks are growing, and without sacrificing a little accuracy, the memory requirements and inference speed of such networks will suffer especially on constrained hardware.

Thanks to this trend, the major deep learning frameworks, like TensorFlow [182], are creating more lightweight solutions capable of running on less powerful hardware. In the case of TensorFlow, there is a TensorFlow Light, which is much more optimized when compared to classic TensorFlow.

These lightweight solutions are especially useful in applications that require the computation to be done close to the sensors collecting the data, either due to the latency of using cloud computing (automotive industry for lane detection, pedestrian and traffic sign recognition, and other systems used in autonomous vehicles) or in places the collected data can not leave the device for privacy and security reasons like, for example, in medical applications. The devices used in these applications are usually small and have relatively low power, and due to their positioning close to the data collection, sensors are called edge devices, and using them is called computing at the edge [183]. These edge devices are usually not equipped with the same amount of memory and are incapable of the same performance as desktop and server grade GPUs and the neural networks to be used on such devices have to be designed with low-performance devices in mind.

One way to still be able to use the complete functionality of a selected deep learning framework, or just to reduce the neural network's memory utilization, is to reduce data type precision of stored weights. This could potentially reduce the memory requirements of the network by half or to only a quarter of their original size when using a 16-bit floating point or 8-bit integer compared to the typical 32-bit floating point, respectively. Especially on NVIDIA-based graphics cards supporting Compute Capability 5.3 or higher [184], this weights quantization would result in a slight speed up of inference. In the last couple of generations of NVIDIA GPUs, the most important change for deep learning applications in the architecture of the GPUs was the inclusion of dedicated tensor cores. These cores were specifically designed to drastically lower the time needed for matrix operations with Half-Precision floating point [185]. This is especially useful due to the nature of operations used in neural networks. The tensor cores, as useful as they might be for performance increase, are not common in devices usable for edge computing, as they are available only in the

more expensive line of the NVIDIA Jetson lineup (Jetson Xavier NX and Jetson AGX Xavier) [186].

Another option is the quantization of the neural network's weights all the way to 8-bit integer, specifically, the data type called QINT8 (Quantized 8-bit Integer or also known as 8-bit fixed point data type). This can make the network's memory requirements even smaller than using weights in 16-bit floating point format, but that is not always the case as parts of the network are still using 32-bit floating point weights. Due to this, the weights can end up using more memory than completely converting the network to Half-Precision. The issue with using a Quantized Integer as a data type for weights is the required modification of the neural network architecture to add support for the quantized data type operations. Another disadvantage may become an advantage, depending on the device the network's inference should be done on. Usage of weights in Quantized Integer format is not yet supported for computations on GPUs, and the network inference can only be done using a CPU. That may become an advantage in systems without a dedicated GPU or with an embedded or external neural network accelerator.

Hardware Support for Lower Precision Arithmetic on a CPU

As implementation goes, some of the frameworks used for deep learning applications like TensorFlow [182] have support for mixed precision in both training and inference, which is only supported on GPUs and transition to the Half-Precision format usually follows after the training. Conversion of the input data is done before every forward pass through the network, unlike the weights of the neural network, which are converted only once. For inference on a regular CPU, this is not very useful as the only Half-Precision operation supported on x86-64 CPUs is the conversion of 32-bit floating point to 16-bit floating point and vice versa allowed through the F16C instruction set extension [187]. Unlike x86-64 CPUs, which do not support Half-Precision arithmetic at all, the ARM-based CPUs do not support 16-bit floating point arithmetic until cores based on Armv8.2-A and later architectures [188]. In the end, if a CPU were asked to do some Half-Precision arithmetic, the 16-bit variables would be automatically promoted to standard 32-bit floating point, and the arithmetic would continue in Single-Precision mode.

As the Half-Precision arithmetic is not supported on the majority of available processors, the usage of lower precision does not produce any visible gain in terms of inference performance at the moment. This could change in the future with Half-Precision rising in popularity and CPU manufacturers adding Half-Precision compute units into the design of their CPUs. On the other hand, the network could be taking up less space in the system memory.

While Half-Precision floating point is not supported on most CPUs on the market today, any modern CPU can use fixed point arithmetic. The fixed point or Quantized Integer data type does not posses the same range of values as the regular floating point. The smaller range of values is a downside made up by processing performance, as integer arithmetic is usually twice to four times faster than floating point arithmetic. The usage of 8-bit Quantized Integer would mean using just a quarter of the memory required for standard Single-Precision floating point.

4.1.2 Indoor Positioning

Currently, the use of existing wireless communication protocols for indoor positioning and navigation has been growing rapidly. The exploitation of radio interfaces and the signal propagation characteristics proved to be a valid approach for IPS. Since every building is different, be it due to the floor plan, furniture placement or just anchors of the IPS being placed in different locations, the indoor positioning must be adapted for each building. Regardless of technology, interior heterogeneity is the reason why a majority of IPSs rely on an analysis of the environment first. This analysis can be very time-consuming, affected by the size and complexity of the environment, required accuracy, and technology employed.

Technologies for Indoor Positioning Systems

Indoor localization can be done using several possible wireless technologies ranging from Bluetooth [77]–[79], Wi-Fi [81], UWB [38], [83], ZigBee [85], [86], visible light [87] to millimeter wave radar [88] and others employing computer vision [89] or dead reckoning [83], [90], which can work without depending on previously created radio infrastructure. Each of the technologies has its own advantages and disadvantages. Almost every type of UE made in the last decade contains a Wi-Fi interface. This, combined with the widespread availability of Wi-Fi infrastructure, makes Wi-Fi based IPS easy to deploy. Usage of BLE beacons requires the deployment of new infrastructure, but most UE devices are already equipped with BLE wireless interfaces. UWB technology achieves the highest positioning accuracy, but the hardware is not widespread at this moment, mainly because of its higher cost [84].

Approaches to Indoor Positioning & Localization

There are three main approaches to locating humans inside a building [189], [190] using radio networks. First, there is active positioning when the UE collects and processes the data to obtain its own location. The second and third approaches are passive, and neither device nor the user does anything to obtain their position

because the anchors analyze either the data packets transmitted by the UE or detect changes in the radio signal propagation. These are not the only options for indoor positioning because other approaches to indoor positioning do not rely on radio networks.

Active Positioning: The active approach to indoor localization works by comparing RSSI of the mobile UE devices to previously collected database of measurements known as RM. The comparison to RMs is called fingerprinting as described in Section 2.4.3, a very popular technology for IPS [70]–[72] because Wi-Fi networks are widely available and do not require any extra deployment costs. The fingerprints are created by the collection of measurements of RSSI at each RP, which is also known as the offline phase of fingerprinting. This approach can be extended by sensor fusion of RSSI and collection of sensor data available to the UE [191]. Another way of active positioning is not by using the signal strength but by employing range-based localization using radio information like ToA, TDoA and AoA [83], [192], which is currently employed by UWB [193] for centimeter-level positioning. The accessibility of this information is dependent on the available capabilities of individual UEs.

Passive Device-Dependent Localization: Very similar to the active approach is passive localization, which relies on users carrying devices. In this case, the positioning infrastructure collects the data transmitted by the UE that is required for obtaining the position. This approach works in almost the same way as the active approach, using a previously collected map of fingerprints.

Passive Device-Free Localization: A third approach uses passive localization of users without relying on the users carrying any device with them. This approach is based on observing abnormal changes in signal propagation created by the interference of the human body in the radio environment. In this case, there are observable fluctuations in RSSIs or variations in the CSI [194]. This approach is becoming more popular as CSI is more stable than RSSI: it is a frequency response to the environment unlike the mere single value of RSSI, and it can also benefit from multi-path signal propagation [195], [196].

Indoor positioning without Radio Networks: Other approaches such as usage of computer vision [89], or dead-reckoning [83], [90], [197], [198] allow alternative indoor positioning. Dead-reckoning requires the initial location of the user, and then by using sensor functions of the UE, it incrementally tracks the position of the user. By combining an accelerometer, gyroscope, and magnetometer, the direction, orientation, and speed of movement of the user can be obtained [199], [200]. Additionally, with a barometer, the movement in the vertical axis can be roughly determined [197], [201].

Section 4.3 focuses on the passive positioning determined by the network, assuming the users are carrying a Wi-Fi enabled device. This approach was chosen as a follow-

up to Sections 3.2 and 3.3, which focused on presence detection using Wi-Fi packet sniffers. This passive with-device method was described in previous paragraphs. In the following sections, the focus is only on localization using RSSI based wireless maps. The creation of RMs have several drawbacks:

- They are very susceptible to changes in the environment, be it the moving of furniture, AP, or anything else that changes the signal propagation in the area of interest. Any change to the environment may require recreating the RM from scratch.
- The creation of the RM can be very time-consuming. The usual approach to creating wireless maps is to collect data at each RP, and the time required for the data collection is very dependent on the size of the space and the density of the RPs.
- Usability of the map of the radio environment is quite limited since the wireless signal propagates differently in each environment. This makes it very difficult to reuse the same indoor positioning system in different locations.
- Not all locations in the environment can be easily accessed for RSSI collection. This means the positioning accuracy in these spots can suffer.

The ability to create RMs more quickly is very useful for the research community focused on indoor positioning. Spending less time on the measurements collection can simplify the adaptation of IPS for new environments. The difference may be in hours, or even days depending on the environment.

4.1.3 Motivation

To improve the situation regarding computational requirements in order to allow run time on low-power devices, reduce power consumption and prolong battery life or free up processor time, the research described in the following sections focuses on finding a balance between accuracy and performance.

At first, the look into the reduction of memory requirements is explored. For this, neural networks are taken into account, as models can have hundreds to thousands of MB. The reduction of memory by reducing the data type precision without retraining is the focus of Section 4.2. This can help adapt large models for memory-constrained devices, and on devices with support for lower precision computing even increasing performance.

Section 4.3, on the other hand, takes into account the data pre-processing to find a balance between accuracy and processing performance, while enhancing the original data. The data pre-processing can help in many ways, be it interpolation of radio signal propagation in spaces the RSSI was not measured, to a fusion of data and reducing the complexity of data sets.

4.2 Reducing Memory Requirements by Lowering Data Precision

The high computing requirements that come with some machine learning algorithms, as well as the need for large amounts of *Random Access Memory* (RAM) had been the main reason to explore the ways to balance the requirements and accuracy. This study, based on the evaluation of the data precision on the accuracy and memory requirements of neural networks, was published at MAREW 2021 [202].

4.2.1 Analysis

A couple of popular CNNs for image classification with very different architectures were selected for testing the influence of Half-Precision weights on their accuracy. The first of the selected networks is the well-known AlexNet [180], followed by a network called GoogLeNet [203], sometimes also called Inception V1 based on the inception blocks used in the network's architecture. The network Inception V3 [204], which is built out of the same blocks and stands as the third version of the inception-based networks was also tested. Another of the tested networks is the architecture called ShuffleNet V2 [205], which was designed to be the best compromise between efficiency and accuracy and is one of the networks with the smallest memory footprint, followed up with another efficient architecture designed for use in mobile devices called MobileNet V2 [206], which just like ShuffleNet V2 requires a small amount of memory when compared to other networks.

The deep learning framework used for testing was PyTorch [207] due to its implementation of Half-Precision data types available directly and simple reduction of weights precision. The possibility of using a Half-Precision data type is also implemented for data tensors, which means precision reduction is just as simple for input data.

The tested networks were also modified to use a quantized 8-bit integer. The networks have not been retrained as only post-training quantization was done, which compared to fully quantization-aware training might produce worse results. Post-training quantization was used because, unlike with the use of quantization-aware training, the use of post-training quantization, as the name suggests, is applied to the already trained network, which means the time-consuming retraining of the neural network is not necessary. Then, followed the comparison of the weights size and the impact of using 8-bit integer instead of 32-bit floating point operations on the accuracy with the unmodified networks. Some architectures may also be affected by using quantized 8-bit integer more than others, and some might not be affected at all, while others may produce much worse results.

Tab. 4.1: Comparison of Single-Precision, Half-Precision, and Quantized Integer Influence on the Size of Networks Weights.

	Single-Precision (32-bit) Size [MB]	Half-Precision (16-bit) Size [MB]	Quantized Integer (8-bit) Size [MB]
AlexNet	244.4	122.2	68.5
GoogLeNet	52.2	26.2	13.1
Inception V3	109.0	54.6	24.0
ShuffleNet V2	9.3	4.7	2.4
MobileNet V2	14.3	7.2	3.6

The neural network with reduced data type precisions was then tested on the single board computer NVIDIA Jetson Nano. The CPU in Jetson Nano is built using 4 Arm A57 cores running at 1.5 GHz while the GPU uses 128 CUDA cores based on the NVIDIA Maxwell architecture running at a frequency of 0.9 GHz, with 4 GB of DDR4 memory shared between the CPU and the GPU. The system as a whole uses at most 10 W. The NVIDIA Jetson Nano was selected for testing for a couple of reasons: its relatively low price, low power consumption compared to traditional GPUs, and due to the inclusion of NVIDIA GPU with complete NVIDIA CUDA support, which is used by most of the deep learning frameworks for inference acceleration.

Testing itself was done in two parts. First of all, the memory footprint of the neural network was compared before and after weights casting to Half-Precision. And second, the accuracy was tested on the classification of 1000 classes present in the ImageNet dataset [181], specifically, a validation subset containing 50 000 annotated images was used (1762 images were not used in the validation process since they are blacklisted due to too difficult classification since the ImageNet 2013 challenge, which makes the total amount of validation images 48 238).

4.2.2 Results

Before any testing, the weights had to be converted to the desired data types. From the results showing the size of weights before and after conversion to 16-bit floating point in Table 4.1, it can be seen that reducing the precision of the network's weights had reduced the size of all tested networks almost in half just as expected and using quantized 8-bit integer reduced the size of the networks almost to a quarter of the original size. This is clearly visible from Fig. 4.1.

For evaluating accuracy was used the Top-N error metric, specifically, the Top-1 error, which presents the classic view at accuracy representing the percentage of cases in which the predicted class with the highest probability did not match the ground truth class. The Top-5 error metric, which is commonly used when working

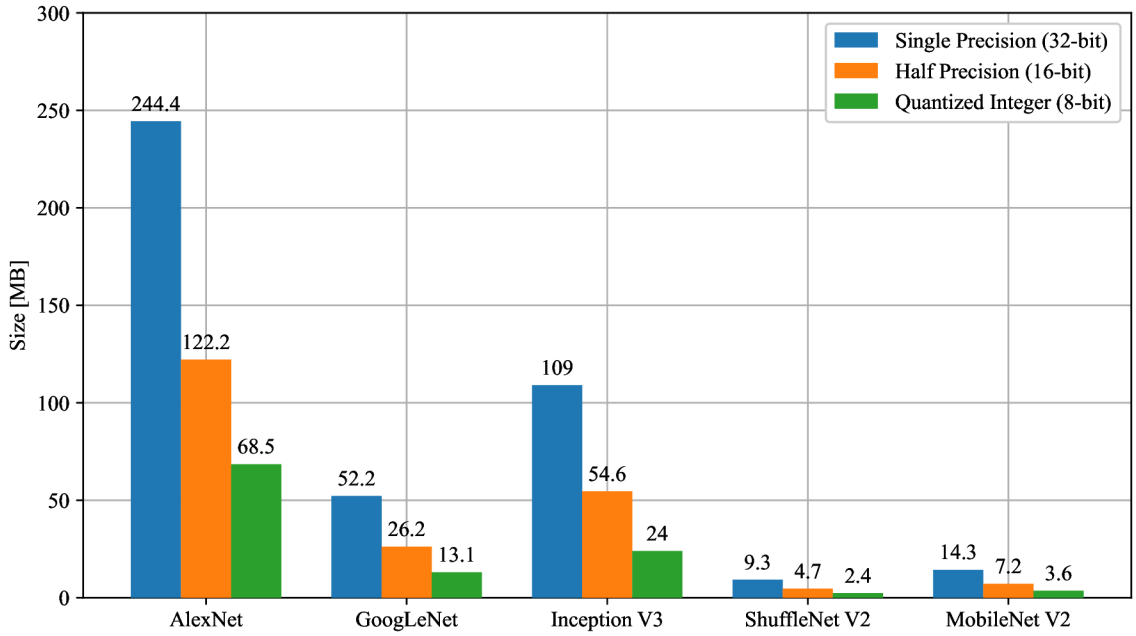


Fig. 4.1: Comparison of the influence of Half-Precision and Quantized Integer data types on the size of weights.

with the classification accuracy over the ImageNet dataset, was also used. The Top-5 error presents how often the ground truth class is not present in the top 5 predictions with the highest probability of being correct.

The resulting accuracy of the networks using Half-Precision weights provided quite interesting results, as the accuracy did not suffer very much or at all. The overall difference in the accuracy of the networks using Half-Precision weights did not get over 0.04 %, which means that out of the 48 238 validation images, the network with the biggest difference classified incorrectly less than 20 images more, than the same network with weights in the standard Single-Precision floating point data type. A little worse results were achieved while using a Quantized Integer. Some networks were not affected at all, while some suffered a big loss in accuracy. The achieved Top-1 and Top-5 errors for all networks using weights with all tested data types are present in Table 4.2 as well as in graphical representation in Figure 4.2.

When it comes to the performance of the networks while using different data types, the results of networks using Quantized Integer weights can not be directly compared to the networks using weights in either one of the floating point data types. That is due to the lack of support for Quantized Integer operations on the GPU in the PyTorch framework for networks quantized post-training. When compared to the computations with Single-Precision and Half-Precision on the GPU, the Half-Precision achieved approximately 1.01-1.17 times better frame rate than using Single-Precision weights. For comparison of all possible options, the inference

Tab. 4.2: Comparison of Single-Precision, Half-Precision, and Quantized Integer Data Type Influence on Top-1 and Top-5 Error.

	Single-Precision (32-bit)		Half-Precision (16-bit)		Quantized Integer (8-bit)	
	Top-1 Error [%]	Top-5 Error [%]	Top-1 Error [%]	Top-5 Error [%]	Top-1 Error [%]	Top-5 Error [%]
AlexNet	43.963	21.008	43.967	21.019	43.965	21.004
GoogLeNet	30.159	10.405	30.184	10.392	30.171	10.444
Inception V3	22.439	6.312	22.418	6.300	30.551	11.456
ShuffleNet V2	30.721	11.696	30.754	11.698	31.917	12.741
MobileNet V2	28.351	9.644	28.364	9.642	68.890	46.584

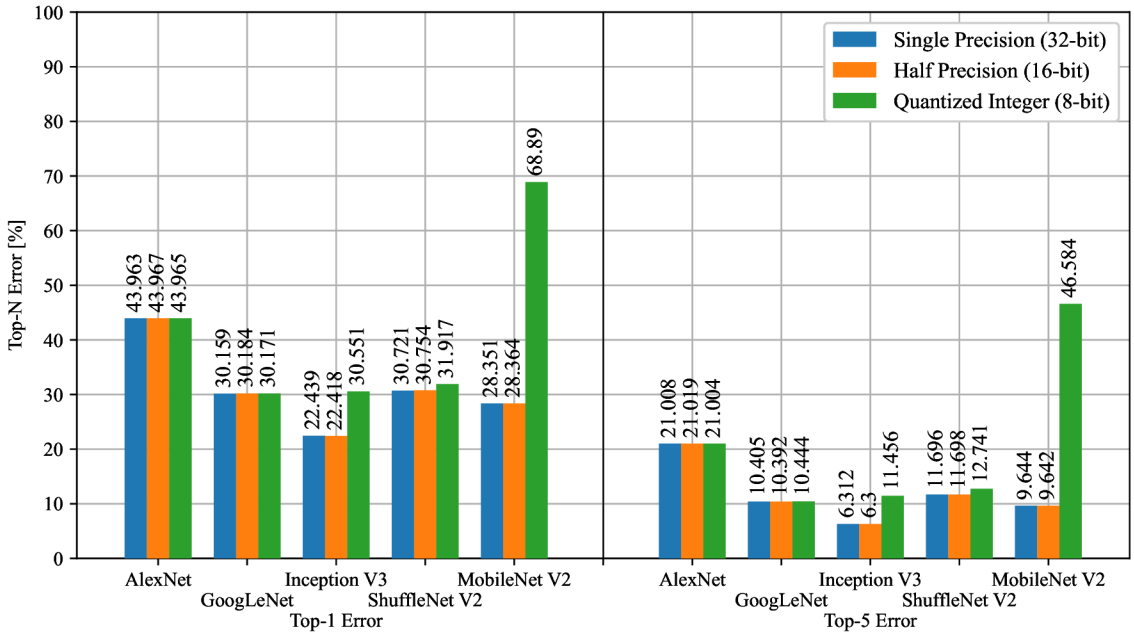
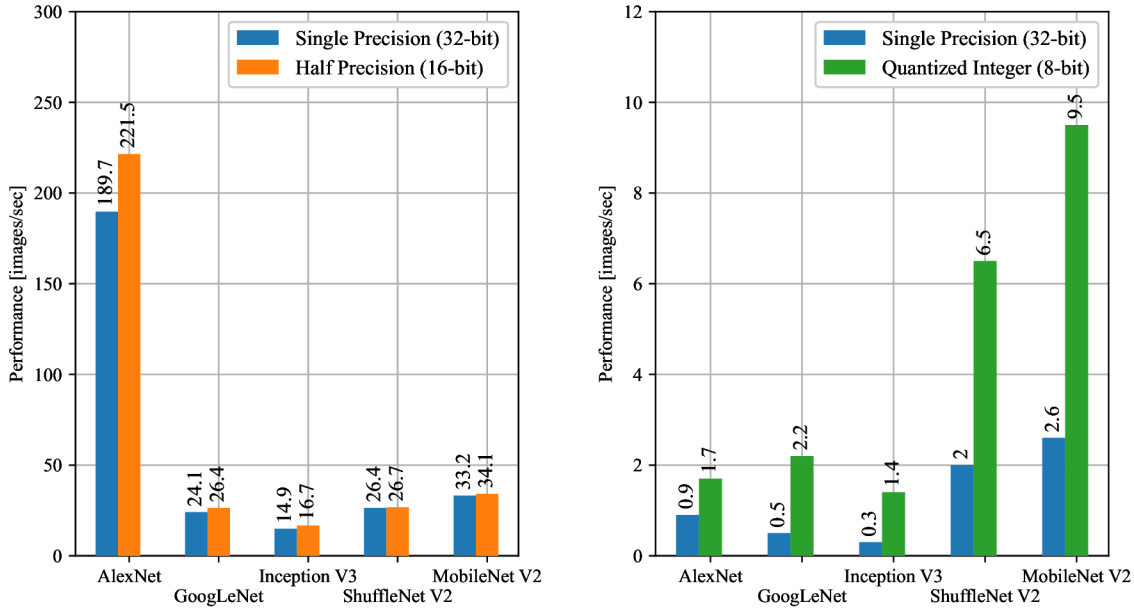


Fig. 4.2: Comparison of the influence of Half-Precision and Quantized Integer data types on the Top-N Error.

speed was also measured on the CPU while using both standard Single-Precision weights and weights in Quantized Integer format. The average frame rate achievable on the NVIDIA Jetson Nano using PyTorch for each network on both the CPU and GPU is presented in Figure 4.3 and in Table 4.3.

AlexNet: AlexNet is now quite an old architecture with the largest weight file and lowest accuracy of the tested networks. While using Half-Precision weights, the accuracy dropped, but not very much, only by 0.004 % for Top-1 error and 0.011 % for Top-5 error. That is, only 2 and 5 images were classified incorrectly for Top-1 and Top-5 errors, respectively.

Even with weights in quantized 8-bit integer data type, the network performed pretty much the same as only 1 more image was classified incorrectly compared



(a) Comparison of average frame rate on GPU

(b) Comparison of average frame rate on CPU

Fig. 4.3: Comparison of inference performance on GPU and CPU with weights in all supported data types.

Tab. 4.3: Comparison of Single-Precision, Half-Precision, and Quantized Integer Influence on the Average Frames per Second on NVIDIA Jetson Nano.

	GPU		CPU	
	Single Precision (32-bit) [images/sec]	Half Precision (16-bit) [images/sec]	Single Precision (32-bit) [images/sec]	Quantized Integer (8-bit) [images/sec]
AlexNet	189.7	221.5	0.9	1.7
GoogLeNet	24.1	26.4	0.5	2.2
Inception V3	14.9	16.7	0.3	1.4
ShuffleNet V2	26.4	26.7	2.0	6.5
MobileNet V2	33.2	34.1	2.6	9.5

to the network using Single-Precision weights and there were just 2 more images classified correctly in the top 5 predictions.

When it comes to performance, the simplicity of the AlexNet architecture proves to be the key for great performance using the integrated GPU on Jetson Nano, as the network managed to get great real-time performance with almost 190 processed frames per second using weights in Single-Precision format. While using Half-Precision, the memory bandwidth was halved and the performance increased by another 30 processed frames per second all the way to 221 FPS. The weights in fixed

point format can not be used on GPUs without retraining the whole network. Due to this, the performance on the CPU was tested instead and it was not great at just 1.7 frames per second, which is not even twice as fast when compared to using Single-Precision on the CPU.

GoogLeNet: Just like AlexNet, the GoogLeNet's Top-1 error dipped a little bit, the Top-5 error on the other hand provided better results when using Half-Precision weights, which is due to the fact that the guess that was the same as ground truth using Single-Precision weights did not have the highest probability when using Half-Precision weights, but was still in the top 5 classes with the highest probability of being correct, all this while the correct results not falling from the top 5 guesses. The difference was also minimal with just 12 more images classified incorrectly for the Top-1 error and only 6 more predictions in the top 5.

The same story as with Half-Precision weights follows with weights in quantized 8-bit integer, the accuracy of GoogLeNet did not suffer almost at all and the difference was lower than 20 incorrect classifications.

GoogLeNet gained on average about 2.3 frames per second while using weights and input data in 16-bit floating point format. That is an increase in performance of 10 % as seen with AlexNet. The computational performance, using weights in fixed point format, did not help with the performance though, as the network was not able to use the integrated GPU of the Jetson Nano to accelerate the highly parallel computations used in neural networks and managed to achieve on average only 2.2 frames per second. Even then the performance using fixed point data type was about 4 $\times$ faster when compared to using Single-Precision arithmetic on the CPU.

Inception V3: The most interesting result was delivered by the network Inception V3, which provided better results in Top-1 error by 0.021 % and in Top-5 error by 0.012 % while using Half-Precision weights. Even though the difference is not very big, by only 10 images in Top-1 and 6 images in Top-5 category, the difference in favor of lower precision weights is there and the Inception V3 network is slightly more accurate while using weights in 16-bit floating point instead of the standard 32-bit floating point data type.

Quite surprisingly, due to the better accuracy while using Half-Precision weights, Inception V3 suffered in terms of accuracy when using fixed-point arithmetic. The difference being about 8 % (3913 images) in Top-1 and 5 % (2481 images) in Top-5 error. This is a significantly lower accuracy compared to the unmodified network. The accuracy while using quantized weights could be increased with quantization-aware training, which would require the network to be retrained.

Just like the networks AlexNet and GoogLeNet, the performance of inference on the GPU while using Half-Precision weights increased about 10 %, which makes a difference of 1.5 frames per second, which is achieved due to the decrease in memory

bandwidth. While using fixed point arithmetic on the CPU, Inception V3 managed to process frames at a rate of 1.4 frames per second. The result is not very impressive but is still 4 × faster than using 32-bit floating point arithmetic on the CPU.

ShuffleNet V2: ShuffleNet V2 is the only tested network other than GoogLeNet, for which the number of incorrectly classified images with Half-Precision weights for Top-1 metric, exceeded 10, specifically to 16 and even then the percentage difference in Top-1 error is just 0.033 %. In the Top-5 error, there was a difference of only a single image classified incorrectly.

The quantized 8-bit integer weights have a small impact on the accuracy of the ShuffleNet V2, as both the Top-1 and Top-5 error increased by 1.2 %, which resulted in about 550 incorrectly classified images for each category.

As one of the smallest networks that were tested, the achieved frame rate on CPU with fixed point weights still can not compete with inference on the GPU, but achieved the best performance to accuracy ratio on the CPU on average of 6.5 frames per second, which is more than 3 × faster when compared to the unmodified network.

MobileNet V2: Even with the usage of 32-bit weights, MobileNet V2 is a very small and accurate network. The accuracy stays almost the same even while using Half-Precision weights further lowering the memory requirements of the network. The Top-1 metric was impacted by incorrect classification of just 6 images. The Top-5 error was reduced a little by 1 more correct classification in the top 5 predictions when compared to the usage of Single-Precision weights.

The most sensitive network to weights being quantized all the way down to the 8-bit fixed point format proved to be the MobileNet V2. The accuracy suffered the most, at 19 555 more incorrect classifications for Top-1 error, the network classifies correctly less than 35 % of all validation images. Similar results are for the Top-5 error, as the network with quantized weights makes 17 819 more incorrect predictions. Just like with Inception V3, the accuracy while using Quantized Integer weights could be improved, if the network were retrained with the quantization-aware training method as post-training quantization does not provide very good results for either of the networks.

The inference performance did not increase almost at all, achieving about 33-34 processed frames per second with both Single-Precision and Half-Precision weights. That is due to the network being quite small and the memory bandwidth, even though lowered by half, is still very small compared to bigger networks like AlexNet, GoogLeNet, or Inception V3. On the other hand, MobileNet V2 proved to be the most efficient for inference on the CPU, mainly due to its small size.

The conversion of weights into fixed point format did not prove to provide good results in terms of accuracy, but on the other hand, the computational requirements

of the network decreased and the network proved to be the most efficient when it comes to inference using the CPU. At almost 10 frames per second, it is the fastest result while not using GPU acceleration. If MobileNet V2 was retrained with quantization-aware training and the accuracy was similar to single and Half-Precision variants, the network would provide the best balance of performance and accuracy.

4.3 Interpolation of Radio Maps for Indoor Positioning

This section focuses on data pre-processing. Specifically on enhancements of RMs with interpolation and signal propagation modeling from RSSI fingerprints and the influence such enhancements have on the accuracy, processing complexity, and time required for data collection. This research was published in the IEEE Sensors Journal.

4.3.1 Dataset Collection

For the evaluation of the influence of RM interpolation on both the accuracy and computational intensity of subsequent localization algorithms, the baseline RM data were collected in the office, whose floor plan, shown in Fig. 4.4, has a rectangular shape with dimensions of 16.71 m by 10.76 m.

Data Acquisition & Format

The dataset used for the analysis of the influence of data interpolation was collected using ESP32 MCUs with firmware described in Section 2.5.1. The ESP32 firmware was expanded specifically for this work with the ability to collect RSSI measurements and MAC address filtering of incoming probe requests.

To have control over the transmission of data, a secondary firmware for probe request transmission was also created. This firmware was described in Section 2. The board employing this firmware was then moved around the 142 reference locations available on the floor of the GEOTEC office at UJI to create a Wi-Fi radio map of the office space. In addition to the 142 RPs, there are 31 locations in the grid in the office that were used, these are not easily accessible due to the furniture and equipment. Those locations are highlighted in Fig. 4.4.

Additionally, to evaluate the accuracy of the interpolated RMs, a small dataset with values collected in both the inaccessible spaces as well as in between RPs of the grid was collected. This allows for better evaluations of locations in inaccessible spaces as well as assessment of the influence the denser RP grid has on positioning accuracy with RSSI values collected in locations outside of the original grid. The dataset, with the raw binary data, processed data for easy manipulation, and scenario splits from subsequent sub-sections, is described in Section 2.5.7.

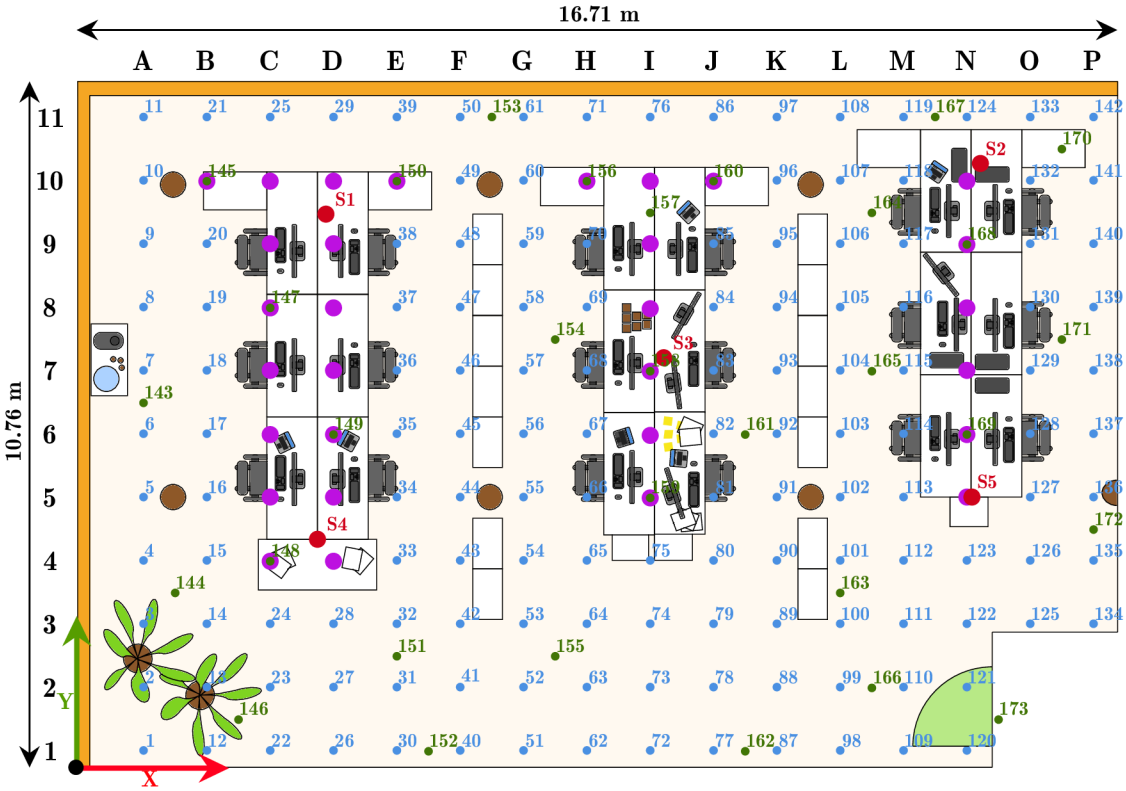


Fig. 4.4: Floor plan of the office space of GEOTEC department at UJI, Spain. (*S1–S5*: ESP32 Sniffer locations, 1–142: Reference Points, 143–173: Evaluation Points).

During the data acquisition, 5 ESP32 MCU boards for the collection of probe requests were used. 5 sniffers were chosen to reduce the influence of RSSI fluctuations on the signal stability by having more data. The reason for 5 sniffers, is coverage of each corner of the office and the last one in the center of the room, approximately equidistant from the ESP32s placed in the corners. During the pre-processing of the data, the probe requests were split according to time ranges recorded by the transmitter board. Then, the sequence numbers available in probe requests were used for measurement synchronization between all 5 boards. From the probe requests, only RSSI values were selected and stored in CSV file format, including the measurement coordinates.

Data Description

In the office, the ESP32 with Probe Request sender firmware emitted 50 probe requests at each RP, which were all collected by the 5 ESP32 MCU boards *S1-S5* used for data collection. The period between two consecutive emitted probe requests was 50 ms as it is a good compromise between a too short time period between 2 probe

requests and having to spend too much time at one RP. During the transmission, a person stood and rotated around the RP. The reason for this was to simulate a shadow of a person close to the UE in the radio environment, and by changing the direction the ESP32 sending probes faces, the influence of the Wi-Fi antenna directionality was reduced.

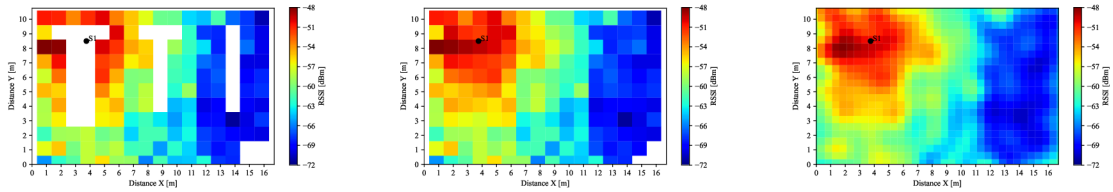
With 142 RPs easily accessible and 50 samples per RP, the dataset consists of 7100 samples. Furthermore, 10 % of the data (710 samples) were taken as an evaluation split for the main grid data. For splitting the data, the helper function of the Python package sklearn *train_test_split* was used. The split and data shuffle is reproducible using *random_state* parameter set to 0. To be able to evaluate the positioning accuracy in spaces where originally no data were measured in, additional 25 samples were collected at 31 locations (see the points 143–173 highlighted in green in Fig. 4.4) bound to a denser 0.5 m grid excluding all the RPs defined by the 1 m grid. Only 25 samples were collected in these locations to avoid creating a bias in the evaluation set by having significantly more samples in the extra locations. These 775 samples extend the evaluation split of the dataset to fairly evaluate positioning accuracy using interpolated RMs in the hard-to-access locations unused for the RM measurement.

4.3.2 Analysis

In the RM processing, several algorithms were used to achieve an approximation of data samples at hard-to-reach RPs and to acquire a higher density of RPs with approximated data in these locations. For the sake of simplicity, the visualizations in the following sections are only for the mean of all values of ESP32 sniffer *S1*. The algorithms that were used for data processing were linear interpolation and *Gaussian Process Regression* (GPR).

Linear Interpolation

Before the application of more complex methods, at first, all measurements in a 1 m grid were considered for an approximation of values in unreachable RPs. To achieve this, linear interpolation for grid data of the Python SciPy package [208] based on an implementation of the quickhull algorithm [209] was used. The Python package SciPy contains the implementation of several algorithms, of which the one used is the Delaunay triangulation to fill in the missing data for hard-to-access RPs. The RM with only the *Measured Data* (MD) by the ESP32 sniffer *S1* is in Fig. 4.5a while the resulting RM with approximated missing data using this interpolation technique can be seen in Fig. 4.5b. In Fig. 4.5a and 4.5b, the centers of the cells representing the RSSI are matching the RPs aligned in 1 m grid, while in the denser Fig. 4.5c,



(a) RM with measured only data.

(b) RM with interpolated data.

(c) GPR approximation of RM.

Fig. 4.5: Visualization of RSSI RM captured by *S1* (ESP32 placement) compared with RM with interpolation of missing values in unreachable RPs using the quickhull algorithm implementation of Python SciPy package.

Tab. 4.4: Comparison of difference in 95$^{\text{th}}$ percentile of positioning accuracy employing RMs created using different covariance functions. In bold is highlighted the best performing covariance function for each RM.

RM	$\frac{Samples}{\text{RP}}$ [-]	RM Grid [m]	SE [m]	SE Fixed [m]	Matern [m]	RQ [m]	SE+Matern [m]	SE+RQ [m]	Matern+RQ [m]
Measured RM	50	1.0	**4.73**	**4.73**	**4.73**	**4.73**	**4.73**	**4.73**	**4.73**
RM with LID	50	1.0	**4.68**	**4.68**	**4.68**	**4.68**	**4.68**	**4.68**	**4.68**
RM by GPR trained	50	1.0	4.81	**4.72**	4.85	4.85	4.85	4.79	4.79
on MD	50	0.5	**4.86**	4.98	5.08	5.26	5.08	5.24	5.26
RM by GPR trained	50	1.0	4.73	4.71	**4.66**	4.70	**4.66**	4.72	4.70
on LID	50	0.5	4.89	**4.85**	5.05	5.10	5.05	5.11	5.12
RM by GPR trained	50	1.0	**4.91**	4.92	4.94	5.02	4.94	5.02	5.05
on Selection of LID	50	0.5	5.24	**5.20**	5.28	5.21	5.28	5.36	5.47
RM by GPR trained	1	1.0	5.01	**4.87**	5.13	5.05	4.99	4.99	4.99
on MD	1	0.5	5.34	**5.27**	5.33	5.38	5.62	5.33	5.62
RM by GPR trained	1	1.0	4.95	4.95	4.93	4.95	4.92	**4.91**	**4.91**
on LID	1	0.5	5.34	5.31	5.35	5.30	**5.30**	5.43	5.37
RM by GPR trained	1	1.0	5.23	5.23	5.36	5.18	6.05	5.18	**5.15**
on Selection of LID	1	0.5	5.69	5.69	5.54	5.52	8.49	5.51	**5.26**

the grid is 0.5 m. The X and Y axes limits of the figures represent the boundaries of the office.

Gaussian Process Regression

Using Gaussian Process Regression [210], [211], a model representing the radio space is created, which is capable of getting an approximate RSSI value at any coordinates. This allows generating a RM of signal propagation that has as many RPs as necessary.

To use GPR, the selection of covariance function is required. The most commonly

used covariance function is the SE. In total, 3 covariance functions used previously in the literature [212] were selected, the SE function, Matern, and *Rational Quadratic* (RQ) functions. Compounds of these covariance functions were also used to evaluate the influence on the created RMs. In total, 7 functions have been tested. In addition to the 3 functions and their combinations, the SE function with fixed length scale parameter was also tested. As the deciding metric was chosen the 95^{th} percentile, as it presents the majority of prediction. From Table 4.4, it is visible that the differences between positioning accuracy achievable by RMs with different covariance functions are negligible. For proceedings in further evaluation, the covariance function SE with fixed length scale was chosen, as it provided the lowest 95^{th} percentile in most variations of RM enhancements and in the rest of the cases, the difference is marginal.

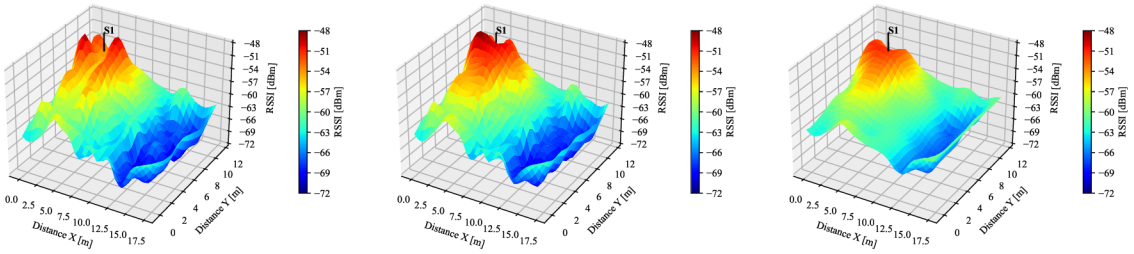
The collected dataset has 50 RSSI values per RP and board ($S1$–$S5$). In the first approach, 50 GPR models per board were created, which results in 50 RMs, each with varying level of RSSI in each RP, that is, the i-th RM is generated using the i-th sample of every RP. In this manner, the generated information better mimics the RSSI fluctuations in real setups.

In the second approach, only one GPR model was trained over all available samples, and this approach has 2 advantages:

- Every new RSSI sample is obtained considering information from 50 real measurements. This results in lower noise and reduced influence of outliers in the dataset.
- The total amount of samples is equal to the amount of RPs, subsequently, the compute requirements can be much lower compared to solutions using the same distribution but several RSSI values per RP.

Unlike the first approach, having 1 sample per position is not ideal for minimizing the influence of RSSI fluctuations of the Wi-Fi signal on the positioning accuracy. However, the lower amount of reference samples can result in significantly lower compute requirements for the final IPS.

The usage of IPS has issues in situations employing incomplete data, with lower accuracy around locations with missing data points. Unlike using just linear interpolation, GPR approximates data with a machine learning model, which means that in short distances to the edge of the measured RM it is possible to extrapolate the data by passing the model coordinates that are outside of the room boundaries. The extrapolation can be seen using the top view in Fig. 4.5c or visualized in 3D in Fig. 4.6. Three-dimensional visualizations of the mean output of GPR approximated RMs with 50 samples are presented in Fig. 4.6 with different input data. The visualization of GPR generated RMs using all input data to create one GPR model was omitted, as the surface plots were almost identical to the visualized mean in Fig. 4.6.



(a) RM approximation created by GPR with only MD, new RPs distributed in 0.5 m grid.

(b) RM approximation created by GPR with interpolated data at missing RPs with new RPs distributed in 0.5 m grid.

(c) GPR approximation of RM with interpolated data at missing RPs, using every 2^{nd} RP in the interpolated data.

Fig. 4.6: 3D visualization of RMs created using GPR out of only MD, interpolated data with approximated values in unreachable RPs using the quickhull algorithm implementation of Python SciPy package with 1 m and 2 m input grid.

Using only MD: In Fig. 4.6a, only the MD in accessible RP were used for the model training. In the location of the sniffer *S1*, where the device is closest to the network interface of the sniffer, the assumption is the RSSI of the tracked device would be the highest. Using only the MD however, provides different results showing a drop in RSSI value.

Using *Linearly Interpolated Data* (LID): The second visualization of using GPR to generate denser RM is in Fig. 4.6b. In this case, the linearly interpolated RSSI values in inaccessible places were included as input of the GPR. As can be seen, around the placement of the sniffer *S1*, there is no longer a drop in RSSI as is expected when the devices are in close proximity to each other. Looking at the radio propagation approximation further, it is noticeable that in places further from the sniffer the linear interpolation did not significantly change from Fig. 4.6a. The similarity to merely using linear interpolation is high, the main difference being the possibility to extrapolate past the edges of the room. The kNN algorithm was chosen for testing, which is known to have higher errors closer to the walls. The reason being the lack of RPs past the walls results in more neighbors being towards the center of the room. Extrapolating the RMs past the boundaries of the room can help battle lower accuracy at the edges of the office.

Using Selection of LID: To explore the ability of GPR to reconstruct RMs, again the RSSI values with LID in inaccessible RPs were used. In this case the input grid of 2 m was used, practically skipping every 2^{nd} value on both axes. The approximated RM is visualized in Fig. 4.6c and is without much of the details present in the RM using as the input all data in the 1 m grid. i.e., the approximation is smother.

Point A1 (see in Fig. 4.4) was selected as starting point for the lower-density grid for multiple reasons:

- When starting on the 2nd row of the grid, instead of at 5, 6 samples are lost in Y axis.
- Starting with line *B*, instead of *A* results in a loss of more points in proximity to the sniffer locations.
- Since *k*NN uses *k* neighboring values to determine the final result, having points around the edges of the room can increase accuracy 1 m away from the wall, which can be assumed to be more likely place for user to be, rather than staying right against the wall.

Even by starting at *A1* the peak in RSSI close to the placement of the sniffer *S1* is lower in Fig. 4.6c in comparison to RMs using denser input grid in Fig. 4.6a and Fig. 4.6b. Reducing the number of points in close proximity results in a loss of even more detail in the RM.

4.3.3 Results

The testing of the influence of RM interpolation was done in 4 ways. At first, the changes in accuracy are evaluated, followed by the processing speed of *k*NN based IPS. This is followed up by a look at the balance between the necessary time for RM gathering, processing time, and accuracy of the *k*NN based IPS. Finally, a t-test is used to determine if there is a statistical difference between the measured RM and all other RMs.

Although there are several enhanced *k*NN implementations, the plain *k*NN was chosen. It is simple, does not require additional hyperparameters to set, and is efficient when computing the position centroids. The reference locations were placed in regular grids of 1 m and 0.5 m, with the reference data equally distributed over the operational area except near the borders. In a larger operational area with an uneven distribution of reference locations and variable density of fingerprints, a dynamic approach would fit better [213], [214]. Nevertheless, other alternatives including *Weighted k-Nearest Neighbors* (w*k*NN), *Adaptive Weighted k-Nearest Neighbors* (aw*k*NN) [213], *Self-Adaptive Weighted k-Nearest Neighbors* (saw*k*NN) [214], *Signal Tendency Index – Weighted k-Nearest Neighbors* (sti-*k*NN) [215] and *Distance & Feature Weighted k-Nearest Neighbors* (dwfw*k*NN) [216] were evaluated on the measured RM, producing differences of the mean positioning error around 1 cm to 2 cm in the best cases.

The measured RM seems to not be sensitive to the positioning method given the density of samples, the number of devices sensing the environment, and the lack of missing (not detected) values in the dataset. The focus of this section is on the data itself rather than the method, making *k*NN a good choice for position estimation.

Tab. 4.5: Overview of accuracy and normalized performance achieved with final IPS based on kNN.

Final IPS Input RM	$\frac{Samples}{RP}$ [-]	RM Grid [m]	Selected k [-]	Reference Samples [-]	MAE [m]	Median Error [m]	75^{th} percentile [m]	95^{th} percentile [m]	RMSE [m]	Normalized Time [-]
Measured RM	50	1.0	90	6390	2.32	2.12	2.92	4.68	2.62	1.00
RM with LID	50	1.0	90	7940	2.30	2.10	2.93	4.68	2.60	1.25
RM by GPR trained	50	1.0	90	8800	2.37	2.17	3.07	4.74	2.68	1.38
on MD	50	0.5	250	40 800	2.39	2.16	3.12	4.94	2.73	6.52
RM by GPR trained	50	1.0	90	8800	2.30	2.11	2.95	4.72	2.61	1.39
on LID	50	0.5	250	40 800	2.34	2.13	3.00	4.82	2.66	6.39
RM by GPR trained	50	1.0	90	8800	2.37	2.16	3.02	4.95	2.72	1.39
on Selection of LID	50	0.5	250	40 800	2.38	2.13	3.08	5.11	2.78	6.35
RM by GPR trained	1	1.0	13	176	2.32	2.05	2.93	4.90	2.66	0.03
on MD	1	0.5	35	816	2.38	2.09	3.08	5.24	2.79	0.13
RM by GPR trained	1	1.0	13	176	2.29	2.01	2.89	5.01	2.65	0.03
on LID	1	0.5	35	816	2.39	2.07	3.10	5.26	2.81	0.13
RM by GPR trained	1	1.0	13	176	2.37	2.10	3.03	5.28	2.77	0.03
on Selection of LID	1	0.5	35	816	2.46	2.04	3.26	5.58	2.96	0.13

To make the accuracy comparison impartial to the selected k, Equation 2.3 described in Section 2.3.1 was used to select k values. Selected values of k for each RM are in Table 4.5. The validity of the selection can be checked by looking at Fig. 4.7. From Fig. 4.7, it is visible that the values selected by the equation match with the k with which each RM achieved the best mean accuracy.

Influence of Processed RMs on Positioning Accuracy

For positioning accuracy evaluation of the final IPS was used the mean error metric. To evaluate the behavior of kNN based IPS, specifically the influence of k on the accuracy, the k was tested starting with $k = 1$ up to $k = 250$, when the improvements in achieved accuracy diminished. The results are shown in Table 4.5. The dependency of the accuracy on the selected value of k for all tested RMs are in Fig. 4.7. Out of the 14 RMs, the trend is easily visible. The kNN based on any of the RMs generated using GPR with 0.5 m grid achieve worse accuracy than the kNN based only on the MD. The difference is not very significant as it differs in the worst cases by 25 cm from the MD baseline.

The best accuracy was achieved by GPR generated RMs with 1 sample per RP, trained on MD and LID. However, the accuracy started dropping with the value of k being higher than 16. That is due to the nature of kNN with low number of features; due to that, this sensitivity to the k is only present in kNN based on the

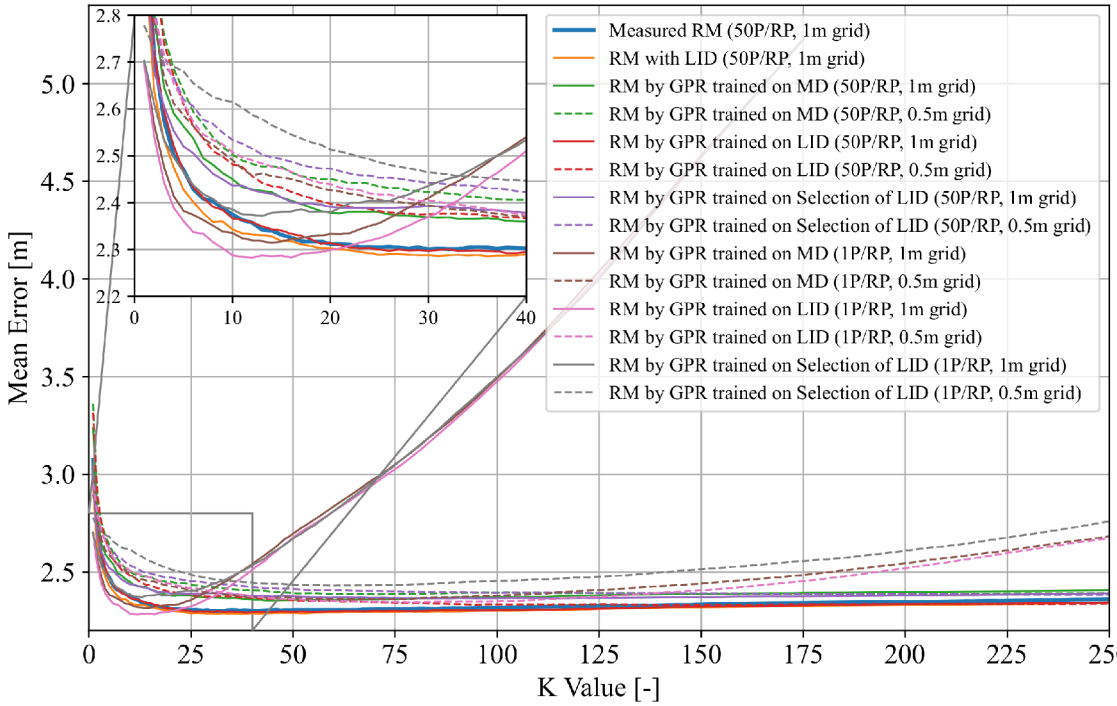


Fig. 4.7: Comparison of RM processing on the mean positioning accuracy of IPS depending on k of kNN algorithm.

smaller RMs.

The most consistent results were gained by using LID. The improvements in the accuracy were not as big as using GPR generated RMs with 1 sample per RP, but the accuracy was for every tested value of k better than using only the MD for training the kNN model.

As expected, the lowest accuracy was achieved by using samples collected at every 2$^{\text{nd}}$ RP. On the other hand, the difference from the baseline is consistently only about 10 cm worse than using all of the MD. That can be an acceptable decrease in accuracy as the time necessary for RM creation significantly decreases.

Additional error metrics to expand on the *Mean Absolute Error* (MAE) are also provided, like median error, 75$^{\text{th}}$ and 95$^{\text{th}}$ percentile as well as RMSE metrics [217]–[219]. All of these metrics are in Table 4.5. To visually show the distribution of the error, Fig. 4.8 presents the *Cumulative Distribution Function* (CDF) for each of the RMs. The figure also includes expanded views to the median and 95$^{\text{th}}$ percentile for better clarity. When looking at the median, it is visible that more than 50 % of evaluation values were predicted with higher accuracy using RMs using only a single sample per RP. This trend though does not continue, and when looking at the 95$^{\text{th}}$ percentile, the best results are achieved by the RM using linear interpolation, with only measured data and RM created using GPR on LID.

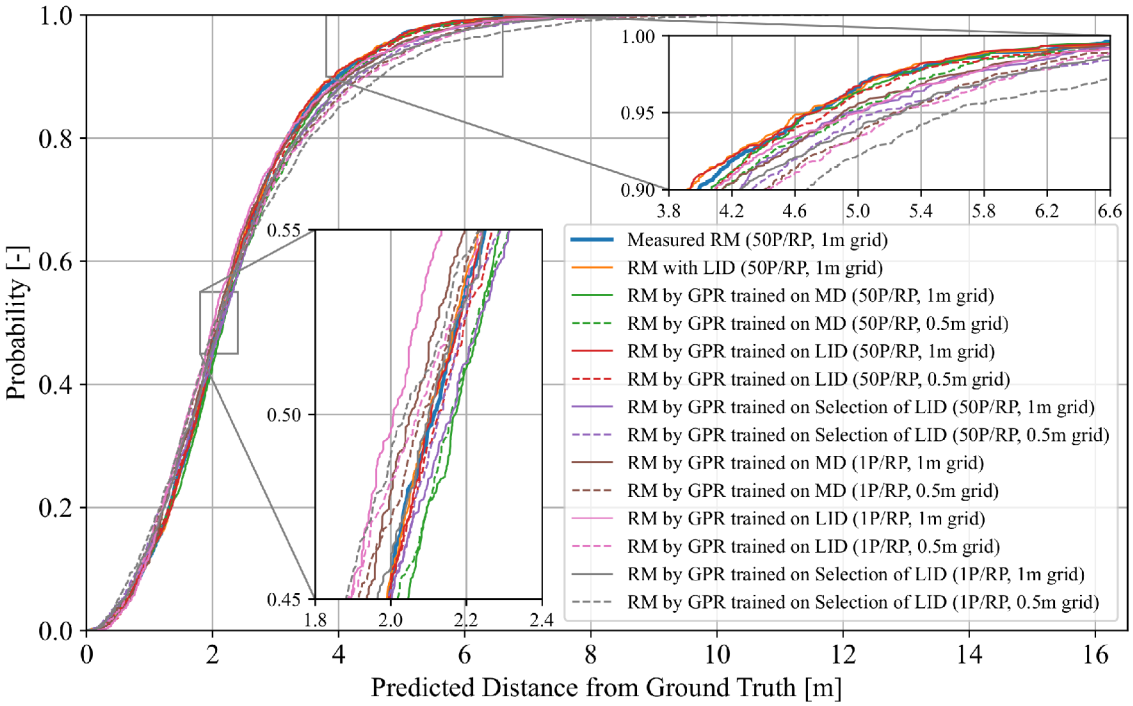
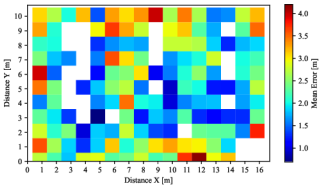


Fig. 4.8: Comparison of RM processing on the CDF of positioning error, highlighting median and 95$^{\text{th}}$ percentile.

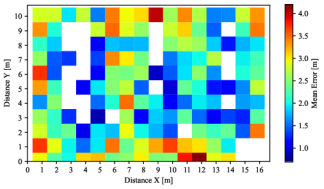
Another factor of accuracy that indoor positioning depends on is the dependency of positioning error on the distance from the edges of the room. Due to fewer neighboring RPs, more neighbors of kNN algorithm are further from the room edge resulting in higher error. This is presented in Fig. 4.9. In Fig. 4.9a, there is the baseline using only the measured data. From the figure, it is visible that the error increases mostly towards the edge of the office, while in the middle it stays relatively low. For comparison were selected RM approximations created using GPR aligned to a 1 m grid with 50 and 1 sample per RP due to their accuracy performance. These are presented in Fig. 4.9b and Fig. 4.9c, respectively. From the evaluation, the interpolation does not remove the issue of error at the room edges completely. However, in both cases of using interpolated RMs, the mean error at the edges of the office decreased. While using the RM with only one sample per RP, the error in the rest of the room increased.

Compute Requirements Based on RM Complexity

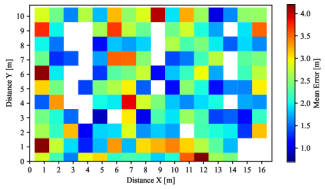
To evaluate the compute performance depending on the RM used for kNN based IPS, the evaluation run times were normalized to the baseline performance of RM created out of only MD. The normalized times were used in order to remove the computing hardware from the question and present relative performance increase or



(a) RM by MD.



(b) RM by GPR trained on LID, with 50 samples per RP in 1 m grid as output.



(c) RM by GPR trained on LID, with 1 sample per RP in 1 m grid as output.

Fig. 4.9: Visualization of mean error of kNN based IPS depending on the RP.

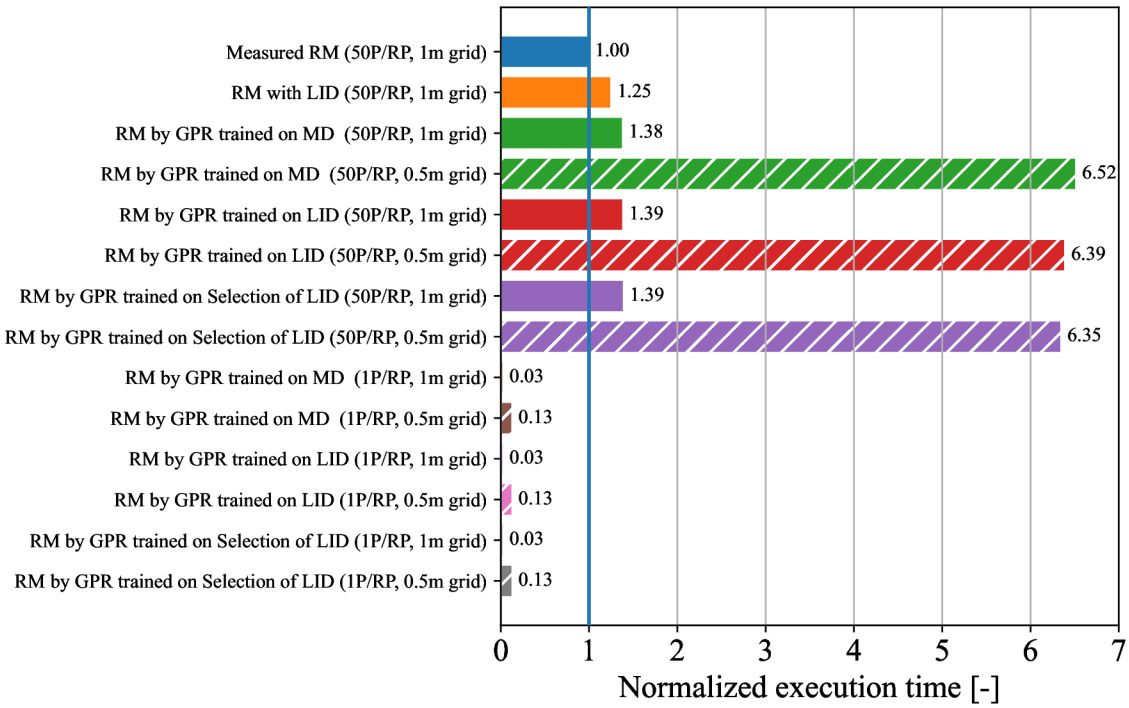


Fig. 4.10: Comparison of RM processing on relative compute requirements of IPS based on kNN algorithm.

decrease depending on the RM that was used. The normalized performance results are in Table 4.5.

The results, as expected, depend on the number of samples in a given RM. This means the RMs with RPs spread out in a 0.5 m grid, instead of 1 m grid used for collecting the measurements, contains approximately 500 % more samples. The performance difference in these cases is unsurprisingly much slower, as can be seen in Fig. 4.10. Similarly, the RM using LID is a bit slower than the baseline. RMs created using GPR with 1 m grid contain more samples than RM created with LID, which again results in slightly slower performance.

Tab. 4.6: Approximation of time required for RM collection in the office acquired using cross-multiplication.

RM type	Grid [m]	RPs [-]	Approximated Time [hh:mm]
Accesible RP	2.0	41	00:38
	1.0	**142**	**02:13**
	0.5	497	07:45
All RP	2.0	47	00:44
	1.0	173	02:42
	0.5	656	10:14

Following the same pattern, the RMs with just 1 sample per RP achieves the best run time performance. Due to the small size of such models, the time required for computation resulted in a fraction of the time required for any RM with 50 samples per RP. Slightly worse computing performance is achieved by the kNN models with just 1 sample per RP but with 0.5 m grid. These models do not suffer from the same issue of the accuracy getting worse with higher values of k, which is due to the RMs having approximately 500 % more samples, but their accuracy gets a big hit.

Reductions in RM Collection Time

The baseline MD with 142 RP took 2 hours and 13 minutes to collect, without considering the time required to mark the reference positions where to stand and broadcast the probe requests. For a relatively small space like the used office, this is a very time-consuming task, and this time increases according to the environment's size. The evaluation samples from an additional 31 RPs in about 29 minutes.

Using these collections times, the time required to collect data in a 0.5 m grid, or in 2 m grid can be estimated, depending on the application needs. The comparison of the time required to create baseline RM in the office, and approximations through cross-multiplication for RMs with different numbers of RPs are in Table 4.6. The table also contains the comparison of different grid size densities as well as collecting data in only accessible as well as in all RP.

When it comes to the time required for collecting the data of RM, the most time-efficient method according to the research is collecting data in a 2 m grid. The accuracy of such RM is lower than collecting RMs using the common and more precise 1 m grid, but the difference in positioning error is not big enough to make a huge difference in positioning of humans. Such RM can be constructed with 400 %

Tab. 4.7: t-test and Pearson correlation results comparing the positioning error of the measured RM with others. t-test tests the null hypothesis that the pairwise difference between the accuracy with the measured RM and each of the others has a mean equal to zero. The value of h indicates whether the t-test rejects ($h = 1$) or not ($h = 0$) the null hypothesis at 5% significance level.

Final IPS Input RM	$\frac{Samples}{RP}$	RM Grid	RM Samples	t-stat	α	Reject Null (h)	Correlation
	[-]	[m]	[-]	[-]	[-]	[-]	[-]
RM with LID	50	1.0	7940	2.63	0.01	1	0.98
RM by GPR trained	50	1.0	8800	−5.49	0.00	1	0.95
on MD	50	0.5	40 800	−5.32	0.00	1	0.91
RM by GPR trained	50	1.0	8800	1.65	0.10	0	0.98
on LID	50	0.5	40 800	−1.98	0.05	1	0.94
RM by GPR trained	50	1.0	8800	−3.15	0.00	1	0.89
on Selection of LID	50	0.5	40 800	−3.49	0.00	1	0.85
RM by GPR trained	1	1.0	176	−0.19	0.85	0	0.89
on MD	1	0.5	816	−3.17	0.00	1	0.83
RM by GPR trained	1	1.0	176	1.72	0.09	0	0.8
on LID	1	0.5	816	−3.51	0.00	1	0.83
RM by GPR trained	1	1.0	176	−2.90	0.00	1	0.85
on Selection of LID	1	0.5	816	−5.15	0.00	1	0.77

fewer RPs, which can save time. In case the application requires higher accuracy, reducing the RP grid is not a good option.

Statistical Analysis of Tested RMs

The two-side Paired-Sample t-test with a 5 % significance level has been used to compare the positioning errors obtained with the measured RM to the positioning errors obtained with the RMs employing LID and GPR. These statistical results are reported in Table 4.7.

From these results, it is visible that using linear interpolation to fill in samples at missing RPs significantly improves measured data. The other 2 cases were the measured data improved with GPR trained on the measured data expanded by linear interpolation with both 50 and 1 sample per RP aligned in 1 m grid. For both of these cases, there is no statistical significance as the pairwise difference between individual errors may have a mean equal to zero. In the case of the model obtained with the RM trained with GPR, 1 sample per RP aligned in 1 m grid, the differences in positioning error with respect to the model training with the measured map are not statistically conclusive, but the kNN matching time is considerably faster. In the remaining cases, the proposed solutions are worse in terms of positioning accuracy,

being the t-test rejecting the null hypothesis in most of cases.

4.4 Discussion

In previous sections, some of the ways to reduce the requirements of modern data processing were evaluated. Be it by reducing data type precision of weights for neural networks or pre-processing data.

At first, the focus of this chapter was on finding the influence of using post-training quantization to lower the precision of data types used for weights storage and of the input data on the accuracy of a few popular neural networks. The move from 32-bit floating point to 16-bit floating point resulted in almost no difference in accuracy as the neural networks provided almost the same level of accuracy, with a difference of less than 20 samples classified incorrectly, in comparison to the usage of single precision weights, which results in less than 0.04 % difference in accuracy of the classification. The network Inception V3 provided the most unexpected results, which unlike any other of the networks managed to classify correctly more images while using Half-Precision weights.

The performance on the GPU of the single-board computer NVIDIA Jetson Nano did not change much, as its architecture does not contain tensor cores that could be used for acceleration of Half-Precision operations. Even without the tensor cores the inference using Half-Precision weights was a little faster every single time. The difference was not very big, only in milliseconds, which is due to the lower memory bandwidth. The weights using Half-Precision data type are almost half the size of the originals, which means the data to be transferred between the assigned CPU and GPU memory sectors is halved. All things considered, there is no issue using the Half-Precision data type in deep learning applications, as the lowered precision does not dramatically reduce the accuracy of the neural networks.

The reduction of weights precision down to 8-bit Quantized Integer provided much more mixed results compared to Half-Precision. AlexNet and GoogLeNet were not affected at all, while MobileNet V2 became unusable. The issue with inconsistent results could be diminished by using quantization-aware training instead of post-training quantization. The increase in performance compared to standard 32-bit floating point was noticeable making it about 4 $\times$ faster while using Quantized Integers.

All things considered, the reduction of weights precision to 16-bit floating point is a valid possibility for reducing memory utilization of neural networks while not affecting accuracy almost at all and possibly gaining free performance increase on hardware that supports Half-Precision arithmetics or tensor processing acceleration.

Reducing the precision of the weight all the way to 8-bit Quantized Integer post-training is a much riskier way of reducing the memory requirements, as the impact on accuracy may be much higher.

When exploring the data pre-processing, Wi-Fi RM of the office was gathered using ESP32 MCUs. Subsequently, the influence of the approximated missing samples of the RM had on the accuracy of kNN based IPS was evaluated. The approximation of samples at missing RP was done using linear interpolation and GPR algorithms. Even though the focus was on server-side positioning, this approach for enhancing RMs can be used in active positioning as well. From the testing, the conclusion is that the benefits of using interpolated RMs depend on selected processing. Interpolation of RM can increase compute performance or reduce the time required for data collection while preserving accuracy. The accuracy dropped up to 38 cm, with the distance between RPs doubled and the collection time quartered.

To answer some raised questions, RM with missing samples in hard-to-access spaces can be completed by using interpolation algorithms. From the testing, the most stable performance in the positioning accuracy of the indoor positioning was using linear interpolation of data for RM creation. This approach provided consistently improved accuracy over the measured RM, regardless of the chosen value of k in kNN algorithm. The difference in accuracy though is not high and is in the cm range. Due to more samples, using this RM was slower by 25 %, compared to the baseline with just measured data. The reason for the slowdown is due to the interpolated RM containing more samples and for kNN the computational complexity grows with the number of reference samples.

Regarding the accuracy, for the RMs created by GPR, with 1 sample per RP aligned in 1 m grid, the positioning error with respect to the measured RM are not statistically conclusive, however, the compute requirements of kNN algorithm are significantly lower. The RM created by using linear interpolation significantly improved measured data and for the cost of slightly higher compute requirements performed consistently better than using just measured data.

4.5 Summary

The work done in the optimizations of machine learning algorithms and interpolations of RMs is summarized by the following paragraphs:

- The first look into the optimizations of ML was done by employing the reduction of data types used for the storage of neural network models. This proved to be an efficient way to reduce the memory requirements of neural networks to half and, in some cases, to a quarter of the original. The used approach

does not require retraining and by using a Half-Precision floating point data type the difference in accuracy is negligible. The use of fixed point data types proved to be more difficult, and for better results, it would need retraining to adapt weights with the data type constraints in mind. In any case, the memory requirements are and will continue to be an important aspect in computing, and the reduction of these requirements is an important way to keep the requirements lower while preserving the accuracy of ML models.

- Second look into optimizations, was more specifically targeted at indoor positioning using fingerprinting of RSSI to create RMs. The prospective of less time necessary to collect the RSSI samples for creation of the RM, by using interpolation techniques to use lower density grid results in saving of time. As a side effect using one approach to interpolation resulted in a negligible drop in accuracy, while the time required for predictions dropped to mere percents of the location predictions using the originally collected data. This approach drastically reduced both memory and computational requirements, which is and always will be important in order to have more processing time available for other tasks or save battery in battery-powered devices by needing less time and energy for computing.

The research presented in this chapter was previously published in the following publications: [118], [202].

5 Conclusions

In this Chapter, the conclusions are drawn in regards to the topics covered in the thesis and the research questions asked in Chapter 1, specifically in Section 1.2. There is also included the list of all publications, software, data, and stays that took place throughout the doctoral studies.

5.1 Research Outputs

The contributions can be split into three blocks with a similar topic. Contributions focused on reusable components were introduced in Chapter 2. Chapter 3 contained contributions with research done into the presence detection field. And Chapter 4 contained contributions regarding optimizations and balancing between accuracy and compute requirements.

5.1.1 Reusable Components, Data, and Research Reproducibility

An important part of research is reproducibility. Without reproducibility, there is no way to evaluate the results published in an article. Due to this, the majority of published works have the corresponding repository with datasets and code. These repositories were created as both supplementary materials, as well as software usable on their own.

The main part of this contribution block is the ESP32-based sniffer [142] of probe requests including all of the variants and options used for data collection in published articles [118], [146], [148], [149], [178]. Apart from these variants the repository also contains reduced data collection with only RSSI, MAC address, and a timestamp of the probe request arrival or expanded variant that collects apart from probe requests also CSI information.

The next part of this block is the datasets collected with the probe request sniffer or sniffers. These datasets were published as supplementary data for both journal and conference papers and just like the code, are available to the research community. All published datasets were based on passive capture of Wi-Fi probe requests. Be it just capture of probes for analysis of presence density or for RM construction, in order to evaluate passive tracking and RM optimizations.

These contributions to the open-source community and reproducible research are available from publicly available repositories [142], [145], [147], [152].

5.1.2 Presence Detection and Tracking Exploiting Wi-Fi

Presence detection is an important task and has many different use cases. It is very valuable for the automation of homes and industrial buildings, in the sense of power saving in case people are not present in the AoI. The tracking of user presence can also be used in crisis situations, to know the occupancy of a building. It can be used for tracking the interest the audience has for presentation sessions during conferences and more privacy-breaching applications for marketing or individual tracking purposes.

There are multiple approaches to presence tracking. Manual can be by users using *Radio Frequency Identification* (RFID) cards or tags, entering code into keypads or computers, or just writing a name down in a book. These approaches are prone to human error, which is not a question of **if** but **when** it happens. Other approaches can rely on sensors monitoring the AoI. In this thesis were presented approaches for with-device presence detection, by exploiting the Wi-Fi communication protocol and the unencrypted management frames for user tracking using fingerprinting as well as privacy-preserving room occupancy estimation.

The research done on this topic was published in several conference papers [146], [148], [178].

5.1.3 Optimizations of ML and Algorithms for IPS

The question of computational requirements is as long as the rise of computing itself. Since the adoption of digital computing, the complexity of algorithms grew, the same as the performance available with new generations of CPUs. The computational performance we have available differs according to the device we are using. While servers are usually at the top of available performance, followed up by workstations and desktop computers, mobile devices with battery and heat constraints have nowhere near as much available performance to them. Going even further and the resources at the disposal of wearable electronics and microcontrollers are in the majority of cases quite limited.

There are two main ways the algorithm can be difficult to run. One thing is the necessary time for finishing the run of an algorithm; the second is the memory footprint of the algorithm. Unlike our desktop and laptop computers, wearables and embedded devices have, in general, less of both computational capacity and available memory. The ML algorithms, in general, require both. Since the popularity of ML only grows and the trend from the last decade does not seem to be slowing down, the need to reduce the requirements of such algorithms to be able to use them in mobile devices is more and more apparent.

In the works used in this thesis, the reduction of memory requirements of CNN is explored, with a reduction to a quarter of the original size. The reduction of computational requirements, as well as the look into improving the accuracy of IPS, was published in a work regarding the interpolations of measured RMs.

The contributions in optimizations and balancing compute performance and accuracy were published in journals, conferences, and repositories [118], [143], [202].

5.2 Answering the Research Questions

The answers to the research questions, which were formulated in Section 1.2 are provided here:

RQ1. ***Do our devices leak data and if so, how? Are there ways adversaries could exploit this leak to track us without our knowledge?***

Throughout Chapter 3 user privacy in indoor positioning was analyzed. During the first scenario based in the office, the probe requests of the Wi-Fi protocol were used to evaluate the privacy-related measures implemented in Wi-Fi. During this study, temporal pattern matching was also introduced as a way to find devices hiding behind MAC address randomization by exploiting the appearances over time.

The second scenario, including the probe request capture was done at an International Conference on Indoor Positioning and Indoor Navigation 2021. The collected probe requests were then analyzed from different angles to find out the feasibility of non-cooperative tracking using Wi-Fi. Even with the widely spread out MAC address randomization since the first implementation, devices can still be tracked and by using fingerprinting of the additional information stored in the probes, in many cases, the randomization can be bypassed entirely. From the amount of captured probe requests, it was also very easy to see how the occupancy changed throughout the 4 conference days. This study clearly represents that the randomization of MAC addresses is not enough, and there is a need for better privacy-related measures.

To further build on top of sniffing the probe requests from the radio environment. Experiments to estimate the occupation of the rooms were conducted based on the amount of received probe requests. These experiments provided results very closely matching the actual recorded room occupancy. This means that an occupancy detection system based on Wi-Fi probe requests can be used to improve the energy efficiency of smart buildings as well as increase safety at the time of crisis.

In summary, the research into privacy and tracking using Wi-Fi results in the

necessity to improve user privacy in Wi-Fi networks beyond just MAC address randomization. And even though MAC randomization helps, it is far from being enough and it can give users a false sense of security.

RQ2. ***What are the ways to preserve the accuracy of machine learning algorithms, while reducing the hardware requirements?***

In Chapter 4, the possibilities of reducing computational requirements were explored. Specifically the techniques of reduction data type precision for weights of neural networks. Instead of using 32-bit floating points, the reduction to 16-bit floating point does not even require retraining. It is a straightforward way to reduce the memory required for storage of the model to just half of the original size without affecting the accuracy of the model. Reducing the type further to 8-bit fixed point data types without retraining provided mixed results and in such cases, the model would require further testing to ensure stability in accuracy performance. The use of lower precision data types can also result in inference performance by using computing devices supporting lower precision operations, like tensor processing units and other ML accelerators.

Apart from the memory requirements for ML algorithms, data augmentation specific to the indoor positioning field was also evaluated. Specifically the use of GPR for interpolation of RMs for fingerprinting approaches to indoor positioning. In this work, the most important results were achieved by creating a RM with a single sample per RP. In this case, the complexity of the dataset dropped drastically from 6390 samples to just 176. This means the single sample was created by a combination of the information from 50 samples belonging to the original RM. The computation speed dropped to just 3 % of the original RM, while the drop in accuracy was negligible in just a few cm. This provides a massive improvement in both memory requirements to store the RM, as well as in the speed of prediction of locations using this RM.

To summarize, 2 possible approaches to optimizations were introduced and proven to preserve the accuracy levels, while requiring much less storage space. The computational performance can also be increased, however, in some cases, it might need hardware with support for specific operations.

5.3 Impact of Publications and Supporting Material

All articles published throughout the doctoral studies are listed below. They are split into 2 parts, the ones included as part of the thesis and those that were not. There are in total 8 publications, out of which 6 are included in the thesis. Included in the thesis is 1 article published in an international journal and 5 conference papers.

Apart from published papers, during the period of the studies, several datasets and software components were published in public repositories.

Publications and Supporting Material Included in the Thesis

- **Journals:**
  1. **T. Bravenec**, M. Gould, T. Frýza and J. Torres-Sospedra, *Influence of Measured Radio Map Interpolation on Indoor Positioning Algorithms.* IEEE Sensors. 2023, doi: 10.1109/JSEN.2023.3296752
- **Conferences:**
  1. **T. Bravenec** and T. Frýza, *Reducing Memory Requirements of Convolutional Neural Networks for Inference at the Edge*, 2021 31st International Conference Radioelektronika (RADIOELEKTRONIKA), 2021, doi: 10.1109/RADIOELEKTRONIKA52220.2021.9420214.
  2. **T. Bravenec**, J. Torres-Sospedra, M. Gould and T. Frýza, *What Your Wearable Devices Revealed About You and Possibilities of Non-Cooperative 802.11 Presence Detection During Your Last IPIN Visit* 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2022, doi: 10.1109/IPIN54987.2022.9918134
  3. **T. Bravenec**, J. Torres-Sospedra, M. Gould and T. Frýza, *Exploration of User Privacy in 802.11 Probe Requests with MAC Address Randomization Using Temporal Pattern Analysis* 17th International Conference on Location Based Services (LBS), 2022, doi: 10.48550/arXiv.2206.10927
  4. T. Frýza, **T. Bravenec** and Z, Kohl, *Security and Reliability of Room Occupancy Detection Using Probe Requests in Smart Buildings*, 2023 33rd International Conference Radioelektronika (RADIOELEKTRONIKA), 2023, doi: 10.1109/RADIOELEKTRONIKA57919.2023.10109085.
  5. **T. Bravenec**, J. Torres-Sospedra, M. Gould and T. Frýza, *UJI Probes: Dataset of Wi-Fi Probe Requests* 13th International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2023, *Accepted for publication.*
- **Data sets:**
  1. **T. Bravenec**, J. Torres-Sospedra, M. Gould and T. Frýza, *Supplementary Materials for "What Your Wearable Devices Revealed About You and Possibilities of Non-Cooperative 802.11 Presence Detection During Your Last IPIN Visit"*, 2022 (1.0) [Data set]. Zenodo. doi: 10.5281/zenodo.6798302
  2. **T. Bravenec**, M. Gould, T. Frýza J. and Torres-Sospedra, *Supplementary Materials for "Influence of Measured Radio Environment Map Interpolation on Indoor Positioning Algorithms"*, 2022 (1.0) [Data set]. Zenodo. doi:

10.5281/zenodo.7193602

3. **T. Bravenec**, J. Torres-Sospedra, M. Gould and T. Frýza, *Supplementary Materials for "Exploration of User Privacy in 802.11 Probe Requests with MAC Address Randomization Using Temporal Pattern Analysis"*, 2023 (1.0) [Data set]. Zenodo. doi: 10.5281/zenodo.7858187

- **Software:**
  1. **T. Bravenec**, *ESP32 Probe Sniffer*, 2022. [Online]. Available: https://gitlab.com/tbravenec/esp32-probe-sniffer.
  2. **T. Bravenec**, Radio Environment Map Interpolations, 2022. [Online]. Available: https://gitlab.com/tbravenec/rem_interpolations
- **Stays:**
  1. Academic secondment held at the Department of Radio Electronics at Brno University of Technology (Brno, Czech Republic), under the supervision of doc. Ing. Tomáš Frýza Ph.D. from September 1, 2019 to August 31, 2021.
  2. Industrial secondment held at UBIK Geospatial Solutions S.L. (Castelló de la Plana, Spain) from September 1, 2022, to November 30, 2022.

Publications and Supporting Material Not Included in the Thesis

- **Journals:**
  1. L. Polak, S. Rozum, M. Slanina, **T. Bravenec**, T. Fryza and A. Pikrakis. *Received Signal Strength Fingerprinting-Based Indoor Location Estimation Employing Machine Learning.* Sensors. 2021; 21(13):4605. doi: 10.3390/s21134605
- **Conferences:**
  1. **T. Bravenec** and T. Frýza, *Multiplatform System for Hand Gesture Recognition*, 2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2019, doi: 10.1109/ISSPIT47144.2019.9001762
- **Software:**
  1. **T. Bravenec**, Computer Vision and Hand Gestures Detection and Fingers Tracking, 2019. [Online]. Available: https://gitlab.com/tbravenec/computer-vision-and-hand-gestures-detection-and-fingers-tracking

5.4 Future Work

During the creation of the dissertation, research into privacy in indoor positioning applications was conducted. Specifically in using Wi-Fi and the management frames to learn information about users without their knowledge. The next step in this is to find the actual coordinates of users without their cooperation. As such the work in the future is going to be focused on the non-cooperative positioning, and fusion of passive approaches with a device using RSSI and without it by using CSI. Even if the approaches are not new by themselves, using the fusion of both of these and possibly adding the active approach into the mix, the positioning accuracy could be improved.

Bibliography

[1] A. Ometov, V. Shubina, L. Klus, J. Skibińska, S. Saafi, P. Pascacio, L. Flueratoru, D. Q. Gaibor, N. Chukhno, O. Chukhno, *et al.*, "A survey on wearable technology: History, state-of-the-art and current challenges", *Computer Networks*, vol. 193, 2021.

[2] W. B. Qaim, A. Ometov, A. Molinaro, I. Lener, C. Campolo, E. S. Lohan, and J. Nurmi, "Towards energy efficiency in the internet of wearable things: A systematic review", *IEEE Access*, vol. 8, 2020.

[3] W. B. Qaim, A. Ometov, C. Campolo, A. Molinaro, E. S. Lohan, and J. Nurmi, "Understanding the Performance of Task Offloading for Wearables in a Two-Tier Edge Architecture", in *2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2021.

[4] L. Klus, D. Quezada-Gaibor, J. Torres-Sospedra, E. S. Lohan, C. Granell, and J. Nurmi, "Rss fingerprinting dataset size reduction using feature-wise adaptive k-means clustering", in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020.

[5] L. Klus, E.-S. Lohan, C. Granell, and J. Nurmi, "Lossy compression methods for performance-restricted wearable devices", in *WiP Proceedings of the International Conference on Localization and GNSS (ICL-GNSS 2020)*, CEUR-WS, 2020.

[6] L. Klus, R. Klus, E. S. Lohan, C. Granell, J. Talvitie, M. Valkama, and J. Nurmi, "Direct lightweight temporal compression for wearable sensor data", *IEEE Sensors Letters*, vol. 5, no. 2, 2021.

[7] L. Klus, D. Quezada-Gaibor, J. Torres-Sospedra, E. S. Lohan, C. Granell, and J. Nurmi, "Towards Accelerated Localization Performance Across Indoor Positioning Datasets", in *2022 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2022.

[8] R. Klus, L. Klus, D. Solomitckii, M. Valkama, and J. Talvitie, "Deep learning based localization and HO optimization in 5G NR networks", in *2020 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2020.

[9] R. Klus, L. Klus, D. Solomitckii, J. Talvitie, and M. Valkama, "Deep learning-based cell-level and beam-level mobility management system", *Sensors*, vol. 20, no. 24, 2020.

[10] R. Klus, L. Klus, J. Talvitie, J. Pihlajasalo, J. Torres-Sospedra, and M. Valkama, "Transfer learning for convolutional indoor positioning systems", in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2021.

[11] N. Chukhno, O. Chukhno, S. Pizzi, A. Molinaro, A. Iera, and G. Araniti, "Efficient Management of Multicast Traffic in Directional mmWave Networks", *IEEE Transactions on Broadcasting*, vol. 67, no. 3, 2021.

[12] N. Chukhno, O. Chukhno, D. Moltchanov, A. Molinaro, Y. Gaidamaka, K. Samouylov, Y. Koucheryavy, and G. Araniti, "Optimal Multicasting in Millimeter Wave 5G NR with Multi-beam Directional Antennas", *IEEE Transactions on Mobile Computing*, 2021.

[13] N. Chukhno, O. Chukhno, D. Moltchanov, A. Gaydamaka, A. Samuylov, A. Molinaro, Y. Koucheryavy, A. Iera, and G. Araniti, "The Use of Machine Learning Techniques for Optimal Multicasting in 5G NR Systems", *IEEE Transactions on Broadcasting*, 2022.

[14] N. Chukhno, O. Chukhno, S. Pizzi, A. Molinaro, A. Iera, and G. Araniti, "Approaching 6G Use Case Requirements with Multicasting", *IEEE Communications Magazine*, vol. 61, no. 5, 2023.

[15] N. Chukhno, S. Trilles, J. Torres-Sospedra, A. Iera, and G. Araniti, "D2D-based cooperative positioning paradigm for future wireless systems: A survey", *IEEE Sensors Journal*, vol. 22, no. 6, 2021.

[16] N. Chukhno, O. Chukhno, S. Pizzi, A. Molinaro, A. Iera, and G. Araniti, "Unsupervised Learning for D2D-Assisted Multicast Scheduling in mmWave Networks", in *2021 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, IEEE, 2021.

[17] A. Ali, "Mobility-Aware Analysis of Directional Deafness in mmWave Communications", in *Proceedings of XXXV Finnish URSI Convention on Radio Science*, URSI, 2019.

[18] A. Ali, O. Galinina, and S. Andreev, "Modeling system-level dynamics of direct XR sessions over mmWave links", in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, IEEE, 2020.

[19] A. Ali, O. Galinina, J. Hosek, and S. Andreev, "Performance Evaluation of Dynamic Computation Offloading Capability for Industrial Wearables", in *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, IEEE, 2021.

[20] A. Ali, O. Galinina, and S. Andreev, "System-level dynamics of highly directional distributed networks", *IEEE Wireless Communications Letters*, vol. 10, no. 7, 2021.

[21] O. Chukhno, N. Chukhno, G. Araniti, C. Campolo, A. Iera, and A. Molinaro, "Optimal placement of social digital twins in edge IoT networks", *Sensors*, vol. 20, no. 21, 2020.

[22] O. Chukhno, N. Chukhno, G. Araniti, C. Campolo, A. Iera, and A. Molinaro, "Placement of Social Digital Twins at the Edge for Beyond 5G IoT Networks", *IEEE Internet of Things Journal*, 2022.

[23] O. Chukhno, O. Galinina, S. Andreev, A. Molinaro, and A. Iera, "Interplay of User Behavior, Communication, and Computing in Immersive Reality 6G Applications", *IEEE Communications Magazine*, 2022.

[24] O. Chukhno, O. Galinina, S. Andreev, A. Molinaro, and A. Iera, "User and Content Dynamics of Edge-Aided Immersive Reality Services", *IEEE Networking Letters*, 2023.

[25] O. Chukhno, N. Chukhno, O. Galinina, Y. Gaidamaka, S. Andreev, and K. Samouylov, "Analysis of 3D deafness effects in highly directional mmWave communications", in *2019 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2019.

[26] O. Chukhno, N. Chukhno, O. Galinina, S. Andreev, Y. Gaidamaka, K. Samouylov, and G. Araniti, "A Holistic Assessment of Directional Deafness in mmWave-based Distributed 3D Networks", *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, 2022.

[27] D. Quezada-Gaibor, L. Klus, J. Torres-Sospedra, E. S. Lohan, J. Nurmi, and J. Huerta, "Improving dbscan for indoor positioning using wi-fi radio maps in wearable and iot devices", in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020.

[28] D. Quezada-Gaibor, J. Torres-Sospedra, J. Nurmi, Y. Koucheryavy, and J. Huerta, "Cloud platforms for context-adaptive positioning and localisation in GNSS-denied scenarios—A systematic review", *Sensors*, vol. 22, no. 1, 2021.

[29] D. Quezada-Gaibor, J. Torres-Sospedra, J. Nurmi, Y. Koucheryavy, and J. Huerta, "Lightweight Wi-Fi Fingerprinting with a Novel RSS Clustering Algorithm", in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE.

[30] D. Quezada-Gaibor, J. Torres-Sospedra, J. Nurmi, Y. Koucheryavy, and J. Huerta, "Lightweight Hybrid CNN-ELM Model for Multi-building and Multi-floor Classification", *arXiv e-prints*, 2022.

[31] D. Quezada-Gaibor, J. Torres-Sospedra, J. Nurmi, Y. Koucheryavy, and J. Huerta, "SURIMI: Supervised Radio Map Augmentation with Deep Learning and a Generative Adversarial Network for Fingerprint-based Indoor Positioning", in *2022 IEEE 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2022.

[32] D. Quezada-Gaibor, L. Klus, J. Torres-Sospedra, E. Simona Lohan, J. Nurmi, C. Granell, and J. Huerta, "Data Cleansing for Indoor Positioning Wi-Fi Fingerprinting Datasets", *arXiv e-prints*, 2022.

[33] P. Pascacio, S. Casteleyn, J. Torres-Sospedra, E. S. Lohan, and J. Nurmi, "Collaborative indoor positioning systems: A systematic review", *Sensors*, vol. 21, no. 3, 2021.

[34] P. Pascacio, J. Torres-Sospedra, and S. Casteleyn, "A Lateration Method based on Effective Combinatorial Beacon Selection for Bluetooth Low Energy Indoor Positioning", in *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, IEEE, 2021.

[35] P. Pascacio, S. Casteleyn, and J. Torres-Sospedra, "Smartphone distance estimation based on rssi-fuzzy classification approach", in *2021 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2021.

[36] P. Pascacio, J. Torres–Sospedra, S. Casteleyn, and E. S. Lohan, "A Collaborative Approach Using Neural Networks for BLE-RSS Lateration-Based Indoor Positioning", in *2022 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2022.

[37] L. Flueratoru, S. Wehrli, M. Magno, and D. Niculescu, "On the energy consumption and ranging accuracy of ultra-wideband physical interfaces", in *GLOBECOM 2020-2020 IEEE Global Communications Conference*, IEEE, 2020.

[38] L. Flueratoru, S. Wehrli, M. Magno, E. S. Lohan, and D. Niculescu, "High-Accuracy Ranging and Localization With Ultrawideband Communications for Energy-Constrained Devices", *IEEE Internet of Things Journal*, vol. 9, no. 10, 2021.

[39] L. Flueratoru, E. S. Lohan, and D. Niculescu, "Self-Learning Detection and Mitigation of Non-Line-of-Sight Measurements in Ultra-Wideband Localization", in *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2021.

[40] L. Flueratoru, E. S. Lohan, and D. Niculescu, "Challenges in platform-independent UWB ranging and localization systems", in *Proceedings of the 16th ACM Workshop on Wireless Network Testbeds, Experimental evaluation & CHaracterization*, 2022.

[41] A. Channa, N. Popescu, and V. Ciobanu, "Wearable solutions for patients with Parkinson's disease and neurocognitive disorder: a systematic review", *Sensors*, vol. 20, no. 9, 2020.

[42] A. Channa, R.-C. Ifrim, D. Popescu, and N. Popescu, "A-WEAR bracelet for detection of hand tremor and bradykinesia in Parkinson's patients", *Sensors*, vol. 21, no. 3, 2021.

[43] A. Channa, G. Ruggeri, N. Mammone, R.-C. Ifrim, A. Iera, and N. Popescu, "Parkinson's Disease Severity Estimation using Deep Learning and Cloud Technology", in *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, IEEE, 2022.

[44] A. Channa, N. Popescu, and M. Faisal, "Parkinson's Disease Gait Evaluation Leveraging Wearable Insoles and Deep Learning Approach", in *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, IEEE, vol. 1, 2022.

[45] A. Channa, N. Popescu, *et al.*, "Managing COVID-19 Global Pandemic with High-Tech Consumer Wearables: A Comprehensive Review", in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020.

[46] A. Channa, N. Popescu, *et al.*, "Robust technique to detect COVID-19 using chest X-ray images", in *2020 International Conference on e-Health and Bioengineering (EHB)*, IEEE, 2020.

[47] A. Channa, N. Popescu, J. Skibinska, and R. Burget, "The rise of wearable devices during the COVID-19 pandemic: A systematic review", *Sensors*, vol. 21, no. 17, 2021.

[48] J. Skibińska and R. Burget, "Parkinson's disease detection based on changes of emotions during speech", in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020.

[49] J. Skibinska, R. Burget, A. Channa, N. Popescu, and Y. Koucheryavy, "COVID-19 Diagnosis at Early Stage Based on Smartwatches and Machine Learning Techniques", *IEEE Access*, vol. 9, 2021.

[50] J. Skibinska and R. Burget, "The transferable methodologies of detection sleep disorders thanks to the actigraphy device for parkinson's disease detection", CEUR Workshop Proceedings, 2021.

[51] J. Skibinska and R. Burget, "Is It Possible to Distinguish COVID-19 Cases and Influenza with Wearable Devices? Analysis with Machine Learning", *Journal of Advances in Information Technology*, vol. 13, no. 3, 2022.

[52] E. Svertoka, A. Rusu-Casandra, and I. Marghescu, "State-of-the-art of industrial wearables: A systematic review", in *2020 13th International Conference on Communications (COMM)*, IEEE, 2020.

[53] E. Svertoka, S. Saafi, A. Rusu-Casandra, R. Burget, I. Marghescu, J. Hosek, and A. Ometov, "Wearables for industrial work safety: A survey", *Sensors*, vol. 21, no. 11, 2021.

[54] E. Svertoka, A. Rusu-Casandra, R. Burget, I. Marghescu, J. Hosek, and A. Ometov, "LoRaWAN: Lost for Localization?", *IEEE Sensors Journal*, 2022.

[55] E. Svertoka, I. Marghescu, A. Rusu-Casandra, R. Burget, J. Hosek, P. Masek, and A. Ometov, "Evaluation of Real-Life LoRaWAN Localization: Accuracy Dependencies Analysis Based on Outdoor Measurement Datasets", in *2022 14th International Conference on Communications (COMM)*, IEEE, 2022.

[56] S. Saafi, J. Hosek, and A. Kolackova, "Cellular-enabled wearables in public safety networks: State of the art and performance evaluation", in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020.

[57] S. Saafi, J. Hosek, and A. Kolackova, "Enabling next-generation public safety operations with mission-critical networks and wearable applications", *Sensors*, vol. 21, no. 17, 2021.

[58] S. Saafi, G. Fodor, J. Hosek, and S. Andreev, "Cellular connectivity and wearable technology enablers for industrial mid-end applications", *IEEE Communications Magazine*, vol. 59, no. 7, 2021.

[59] S. Saafi, O. Vikhrova, S. Andreev, and J. Hosek, "Enhancing Uplink Performance of NR RedCap in Industrial 5G/B5G Systems", in *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, IEEE, 2022.

[60] S. Saafi, O. Vikhrova, G. Fodor, J. Hosek, and S. Andreev, "AI-Aided Integrated Terrestrial and Non-Terrestrial 6G Solutions for Sustainable Maritime Networking", *arXiv preprint arXiv:2201.06947*, 2022.

[61] V. Shubina, A. Ometov, D. Niculescu, and E.-S. Lohan, "Challenges of Privacy-aware Localization on Wearable Devices", in *Proceedings of XXXV Finnish URSI Convention on Radio Science*, URSI, 2019.

[62] V. Shubina, A. Ometov, S. Andreev, D. Niculescu, and E. S. Lohan, "Privacy versus Location Accuracy in Opportunistic Wearable Networks", in *2020 International Conference on Localization and GNSS (ICL-GNSS)*, IEEE, 2020.

[63] V. Shubina, S. Holcer, M. Gould, and E. S. Lohan, "Survey of decentralized solutions with mobile devices for user location tracking, proximity detection, and contact tracing in the covid-19 era", *Data*, vol. 5, no. 4, 2020.

[64] V. Shubina, A. Ometov, and E. S. Lohan, "Technical perspectives of contact-tracing applications on wearables for COVID-19 control", in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, IEEE, 2020.

[65] V. Shubina, A. Ometov, A. Basiri, and E. S. Lohan, "Effectiveness modelling of digital contact-tracing solutions for tackling the COVID-19 pandemic", *The Journal of Navigation*, vol. 74, no. 4, 2021.

[66] E. S. Lohan, V. Shubina, and D. Niculescu, "Perturbed-Location Mechanism for Increased User-Location Privacy in Proximity Detection and Digital Contact-Tracing Applications", *Sensors*, vol. 22, no. 2, 2022.

[67] R. Casanova-Marqués, P. Pascacio, J. Hajny, and J. Torres-Sospedra, "Anonymous attribute-based credentials in collaborative indoor positioning systems", 2021.

[68] R. Casanova-Marqués, P. Dzurenda, and J. Hajny, "Implementation of Revocable Keyed-Verification Anonymous Credentials on Java Card", in *Proceedings of the 17th International Conference on Availability, Reliability and Security*, 2022.

[69] R. Casanova-Marqués and P. Dzurenda, "Readiness of Anonymous Credentials for Real Environment Deployment", in *Proceedings II of the 28th Conference STUDENT EEICT 2022 Selected Papers*, Brno University of Technology, Apr. 2022, ch. 177878.

[70] A. Kushki, K. N. Plataniotis, and A. N. Venetsanopoulos, "Kernel-based positioning in wireless local area networks", *IEEE transactions on mobile computing*, vol. 6, no. 6, 2007.

[71] S.-H. Fang, T.-N. Lin, and K.-C. Lee, "A novel algorithm for multipath fingerprinting in indoor WLAN environments", *IEEE transactions on wireless communications*, vol. 7, no. 9, 2008.

[72] C. Feng, W. S. A. Au, S. Valaee, and Z. Tan, "Received-signal-strength-based indoor positioning using compressive sensing", *IEEE Transactions on mobile computing*, vol. 11, no. 12, 2011.

[73] A. D. Thierer, "The Internet of Things and Wearable Technology: Addressing Privacy and Security Concerns without Derailing Innovation", *Adam Thierer, The Internet of Things and Wearable Technology: Addressing Privacy and Security Concerns without Derailing Innovation*, vol. 21, 2015.

[74] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown, "A Study of MAC Address Randomization in Mobile Devices and When it Fails", *arXiv preprint arXiv:1703.02874*, 2017.

[75] E. Fenske, D. Brown, J. Martin, T. Mayberry, P. Ryan, and E. Rye, "Three Years Later: A Study of MAC Address Randomization in Mobile Devices and When it Succeeds", *Proceedings on Privacy Enhancing Technologies*, vol. 2021, no. 3, 2021.

[76] S. Subedi and J.-Y. Pyun, "A Survey of Smartphone-Based Indoor Positioning System Using RF-Based Wireless Technologies ", *Sensors*, vol. 20, no. 24, 2020.

[77] A. Basiri, E. S. Lohan, T. Moore, A. Winstanley, P. Peltola, C. Hill, P. Amirian, and P. F. e Silva, "Indoor location based services challenges, requirements and usability of current solutions", *Computer Science Review*, vol. 24, 2017.

[78] A. J. Martín, I. M. Gordo, D. G. Gómez, S. G. de Villa, S. L. Plaza, and J. J. G. Domínguez, "BLE-Based Approach for Detecting Daily Routine Changes", in *2021 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, IEEE, 2021.

[79] L. Polak, S. Rozum, M. Slanina, T. Bravenec, T. Fryza, and A. Pikrakis, "Received signal strength fingerprinting-based indoor location estimation employing machine learning", *Sensors*, vol. 21, no. 13, 2021.

[80] "RADAR: An In-Building RF-Based User Location and Tracking System", in *Proceedings IEEE INFOCOM 2000. Conference on computer communications. Nineteenth annual joint conference of the IEEE computer and communications societies (Cat. No. 00CH37064)*, Ieee, vol. 2, 2000.

[81] M. N. Husen and S. Lee, "Indoor human localization with orientation using WiFi fingerprinting", in *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*, 2014.

[82] P. Roy and C. Chowdhury, "A Survey on Ubiquitous WiFi-based Indoor Localization System for Smartphone Users from Implementation Perspectives", *CCF Transactions on Pervasive Computing and Interaction*, vol. 4, no. 3, 2022.

[83] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrami, M. A. Al-Ammar, and H. S. Al-Khalifa, "Ultra wideband indoor positioning technologies: Analysis and recent advances", *Sensors*, vol. 16, no. 5, 2016.

[84] J. Kunhoth, A. Karkar, S. Al-Maadeed, and A. Al-Ali, "Indoor positioning and wayfinding systems: a survey", *Human-centric Computing and Information Sciences*, vol. 10, no. 1, 2020.

[85] Z. Y. Dong, W. M. Xu, and H. Zhuang, "Research on ZigBee Indoor Technology Positioning Based on RSSI", *Procedia Computer Science*, vol. 154, 2019.

[86] J. Zhen, B. Liu, Y. Wang, and Y. Liu, "An Improved Method for Indoor Positioning based on ZigBee Technique", *International Journal of Embedded Systems*, vol. 13, no. 3, 2020.

[87] Á. De-La-Llana-Calvo, J.-L. Lázaro-Galilea, A. Gardel-Vicente, D. Salido-Monzú, I. Bravo-Muñoz, A. Iamnitchi, and R. Gil-Vera, "Weak Calibration of a Visible Light Positioning System based on a Position-Sensitive Detector: Positioning Error Assessment", *Sensors*, vol. 21, no. 11, 2021.

[88] J. A. Paredes, F. J. Álvarez, M. Hansard, and K. Z. Rajab, "A Gaussian Process Model for UAV Localization Using millimetre Wave Radar", *Expert Systems with Applications*, vol. 185, 2021.

[89] S. Yang, L. Ma, S. Jia, and D. Qin, "An Improved Vision-based Indoor Positioning Method", *IEEE Access*, vol. 8, 2020.

[90] A. Riady and G. P. Kusuma, "Indoor Positioning System Using Hybrid Method of Fingerprinting and Pedestrian Dead Reckoning", *Journal of King Saud University-Computer and Information Sciences*, 2021.

[91] A. Morar, A. Moldoveanu, I. Mocanu, F. Moldoveanu, I. E. Radoi, V. Asavei, A. Gradinaru, and A. Butean, "A Comprehensive Survey of Indoor Localization Methods Based on Computer Vision ", *Sensors*, vol. 20, no. 9, 2020.

[92] Y. Wu, H.-B. Zhu, Q.-X. Du, and S.-M. Tang, "A Survey of the Research Status of Pedestrian Dead Reckoning Systems Based on Inertial Sensors", *International Journal of Automation and Computing*, vol. 16, 2019.

[93] R. F. Brena, J. P. García-Vázquez, C. E. Galván-Tejada, D. Muñoz-Rodriguez, C. Vargas-Rosales, and J. Fangmeyer, "Evolution of Indoor Positioning Technologies: A Survey", *Journal of Sensors*, vol. 2017, 2017.

[94] WiGLE, *WiGLE: Wireless Network Mapping*, 2022. [Online]. Available: https://wigle.net/.

[95] IEEE, "ISO/IEC/IEEE - International Standard - Telecommunications and Information Exchange Between Systems - Specific Requirements for Local and Metropolitan Area Networks - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE, Tech. Rep. IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016), 2022.

[96] J.-S. Lee, Y.-W. Su, and C.-C. Shen, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi", in *IECON 2007-33rd Annual Conference of the IEEE Industrial Electronics Society*, Ieee, 2007.

[97] W. Sun, O. Lee, Y. Shin, S. Kim, C. Yang, H. Kim, and S. Choi, "Wi-Fi Could be Much More", *IEEE Communications Magazine*, vol. 52, no. 11, 2014.

[98] M. Gast, *802.11n: A Survival Guide*. O'Reilly Media, Inc., 2012.

[99] M. S. Gast, *802.11ac: A Survival Guide: Wi-Fi at Gigabit and Beyond*. O'Reilly Media, Inc., 2013.

[100] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi, "A Tutorial on IEEE 802.11ax High Efficiency WLANs", *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, 2018.

[101] E. Khorov, I. Levitsky, and I. F. Akyildiz, "Current Status and Directions of IEEE 802.11be, the Future Wi-Fi 7", *IEEE access*, vol. 8, 2020.

[102] C. Deng, X. Fang, X. Han, X. Wang, L. Yan, R. He, Y. Long, and Y. Guo, "IEEE 802.11be Wi-Fi 7: New Challenges and Opportunities", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, 2020.

[103] E. G. Villegas, E. Lopez-Aguilera, R. Vidal, and J. Paradells, "Effect of Adjacent-Channel Interference in IEEE 802.11 WLANs", in *2007 2nd International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, IEEE, 2007.

[104] M. Mehrnoush and S. Roy, "Coexistence of WLAN Network with Radar: Detection and Interference Mitigation", *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, 2017.

[105] N. Darchis, "802.11 Frames: A Starter Guide to Learn Wireless Sniffer Traces", *Cisco Community*, 2010. [Online]. Available: https://community.cisco.com/t5/wireless-mobility-knowledge-base/802-11-frames-a-starter-guide-to-learn-wireless-sniffer-traces/ta-p/3110019.

[106] Google, *Google Developers: Geolocation API*, Google, 2022. [Online]. Available: https://developers.google.com/maps/documentation/geolocation/.

[107] Google, *Google Support: Control access point inclusion in Google's Location services*, 2022. [Online]. Available: https://support.google.com/maps/answer/1725632?hl=en.

[108] S. Zeadally, F. Siddiqui, and Z. Baig, "25 Years of Bluetooth Technology", *Future Internet*, vol. 11, no. 9, 2019.

[109] E. Au, "Bluetooth 5.0 and Beyond [standards]", *IEEE Vehicular Technology Magazine*, vol. 14, no. 2, 2019.

[110] I. Association *et al.*, "802.15. 1-2005-IEEE Standard for Information Technology - Local and Metropolitan Area Networks - Specific Requirements - part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN)", Tech. Rep., 2005.

[111] Bluetooth Special Interest Group, *Bluetooth Technology Overview*, 2021. [Online]. Available: https://www.bluetooth.com/learn-about-bluetooth/tech-overview/.

[112] Bluetooth Special Interest Group, *Bluetooth Core Specification Version 5.4 Technical Overview*, 2023. [Online]. Available: https://www.bluetooth.com/bluetooth-resources/bluetooth-core-specification-version-5-4-technical-overview/.

[113] F. A. Toasa, L. Tello-Oquendo, C. R. Peńafiel-Ojeda, and G. Cuzco, "Experimental Demonstration for Indoor Localization Based on AoA of Bluetooth 5.1 Using Software Defined Radio", in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, IEEE, 2021.

[114] Bluetooth SIG, "Enhancing Bluetooth Location Services with Direction Finding", *Bluetooth Resources*, 2019.

[115] S.-H. Cha, "Comprehensive Survey on Distance/Similarity Measures Between Probability Density Functions", *City*, vol. 1, no. 2, 2007.

[116] L. Calderoni, M. Ferrara, A. Franco, and D. Maio, "Indoor Localization in a Hospital Environment using Random Forest Classifiers", *Expert Systems with Applications*, vol. 42, no. 1, 2015.

[117] P. Nadkarni, "Core Technologies: Data Mining and "Big Data"", *Clinical Research Computing*, vol. 9, 2016.

[118] T. Bravenec, M. Gould, T. Frýza, and J. Torres-Sospedra, "Influence of Measured Radio Map Interpolation on Indoor Positioning Algorithms", *IEEE Sensors Journal*, 2023.

[119] C. Kingsford and S. L. Salzberg, "What are Decision Trees?", *Nature biotechnology*, vol. 26, no. 9, 2008.

[120] P. Sodhi, N. Awasthi, and V. Sharma, "Introduction to Machine Learning and Its Basic Application in Python", in *Proceedings of 10th International Conference on Digital Strategies for Organizational Success*, 2019. [Online]. Available: https://ssrn.com/abstract=3323796.

[121] A. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: a tutorial", *Computer*, vol. 29, no. 3, 1996.

[122] "Bio-inspired Neurocomputing", *Studies in Computational Intelligence*, 2021. [Online]. Available: http://dx.doi.org/10.1007/978-981-15-5495-7.

[123] V. Nair and G. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair", vol. 27, Jun. 2010.

[124] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models", in *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[125] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, *Activation Functions: Comparison of trends in Practice and Research for Deep Learning*, arXiv preprint arXiv:1811.03378, 2018. arXiv: 1811.03378 [cs.LG].

[126] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional Neural Networks: An Overview and Application in Radiology", *Insights Into Imaging*, vol. 9, 2018.

[127] H. Gholamalinezhad and H. Khosravi, *Pooling Methods in Deep Neural Networks, a Review*, arXiv preprint arXiv:2009.07485, 2020. arXiv: 2009.07485 [cs.CV].

[128] X. Ying, "An Overview of Overfitting and its Solutions", *Journal of Physics: Conference Series*, vol. 1168, Feb. 2019.

[129] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, *Improving neural networks by preventing co-adaptation of feature detectors*, arXiv preprint arXiv:1207.0580, 2012. arXiv: 1207.0580 [cs.NE].

[130] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting", *The journal of machine learning research*, vol. 15, no. 1, 2014.

[131] S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, arXiv preprint arXiv:1502.03167, 2015. arXiv: 1502.03167 [cs.LG].

[132] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*, arXiv preprint arXiv:1505.04597, 2015. arXiv: 1505.04597 [cs.CV].

[133] H. Zhang, L. Zhang, and Y. Jiang, "Overfitting and Underfitting Analysis for Deep Learning Based End-to-end Communication Systems", in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2019.

[134] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "Survey and Benchmarking of Machine Learning Accelerators", in *2019 IEEE High Performance Extreme Computing Conference (HPEC)*, IEEE, 2019.

[135] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "Survey of Machine Learning Accelerators", in *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, IEEE, 2020.

[136] Q. Wang, M. Fu, J. Wang, H. Luo, L. Sun, Z. Ma, W. Li, C. Zhang, R. Huang, X. Li, *et al.*, "Recent Advances in Pedestrian Inertial Navigation based on Smartphone: A Review", *IEEE Sensors Journal*, 2022.

[137] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual SLAM Algorithms: A Survey from 2010 to 2016", *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, 2017.

[138] S. Yan, Y. Su, A. Sun, Y. Ji, J. Xiao, and X. Chen, "Low-Cost and Lightweight Indoor Positioning based on Computer Vision", in *2022 4th Asia Pacific Information Technology Conference*, 2022.

[139] A. Basiri, T. Lines, and M. Fidel Pereira, "Scalable 3D Mapping of Cities Using Computer Vision and Signals of Opportunity", *International Journal of Geographical Information Science*, 2023.

[140] J. Torres-Sospedra, R. Montoliu, S. Trilles, Ó. Belmonte, and J. Huerta, "Comprehensive Analysis of Distance and Similarity Measures for Wi-Fi Fingerprinting Indoor Positioning Systems", *Expert Systems with Applications*, vol. 42, no. 23, 2015.

[141] A. Pérez-Navarro, J. Torres-Sospedra, R. Montoliu, J. Conesa, R. Berkvens, G. Caso, C. Costa, N. Dorigatti, N. Hernández, S. Knauth, *et al.*, "Challenges of Fingerprinting in Indoor Positioning and Navigation", in *Geographical and Fingerprinting Data to Create Systems for Indoor Positioning and Indoor/Outdoor Navigation*, Elsevier, 2019.

[142] T. Bravenec, *ESP32 Probe Sniffer*, 2022. [Online]. Available: https://gitlab.com/tbravenec/esp32-probe-sniffer.

[143] T. Bravenec, *Radio Environment Map Interpolations*, 2022. [Online]. Available: https://gitlab.com/tbravenec/rem_interpolations.

[144] IEEE, *IEEE Standards - OUI/MA-L*. [Online]. Available: https://standards-oui.ieee.org/.

[145] T. Bravenec, J. Torres-Sospedra, M. Gould, and T. Frýza, *Supplementary Materials for "What Your Wearable Devices Revealed About You and Possibilities of Non-Cooperative 802.11 Presence Detection During Your Last IPIN Visit"*, version 1.0, Zenodo, 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6798302.

[146] T. Bravenec, J. Torres-Sospedra, M. Gould, and T. Fryza, "What Your Wearable Devices Revealed About You and Possibilities of Non-Cooperative 802.11 Presence Detection During Your Last IPIN Visit", in *2022 IEEE 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2022.

[147] T. Bravenec, J. Torres-Sospedra, M. Gould, and T. Frýza, *Supplementary Materials for "Exploration of User Privacy in 802.11 Probe Requests with MAC Address Randomization Using Temporal Pattern Analysis"*, version 1.0, Zenodo, 2023. [Online]. Available: https://doi.org/10.5281/zenodo.7858187.

[148] T. Bravenec, J. Torres-Sospedra, M. Gould, and T. Fryza, “Exploration of User Privacy in 802.11 Probe Requests with MAC Address Randomization Using Temporal Pattern Analysis”, *arXiv e-prints*, 2022.

[149] T. Bravenec, J. Torres-Sospedra, M. Gould, and T. Fryza, “UJI Probes: Dataset of Wi-Fi Probe Requests”, in *2023 IEEE 13th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2023.

[150] B. S. Ciftler, S. Dikmese, I. Güvenç, K. Akkaya, and A. Kadri, “Occupancy Counting with Burst and Intermittent Signals in Smart Buildings”, *IEEE Internet of Things Journal*, vol. 5, no. 2, 2017.

[151] C. Matte, M. Cunche, F. Rousseau, and M. Vanhoef, “Defeating MAC Address Randomization Through Timing Attacks”, in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, 2016.

[152] T. Bravenec, J. Torres-Sospedra, M. Gould, and T. Fryza, *Supplementary Materials for "UJI Probes: Dataset of Wi-Fi Probe Requests"*, version 1.0, Zenodo, 2023. [Online]. Available: https://doi.org/10.5281/zenodo.7801798.

[153] T. Bravenec, M. Gould, T. Frýza, and J. Torres-Sospedra, *Supplementary Materials for "Influence of Measured Radio Environment Map Interpolation on Indoor Positioning Algorithms"*, version 1.0, Zenodo, 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7193602.

[154] A. Musa and J. Eriksson, “Tracking Unmodified Smartphones Using wi-fi Monitors”, in *Proceedings of the 10th ACM conference on embedded network sensor systems*, 2012.

[155] L. Hutchinson, *iOS 8 to Stymie Trackers and Marketers with MAC Address Randomization*, Jun. 2014. [Online]. Available: https://arstechnica.com/gadgets/2014/06/ios8-to-stymie-trackers-and-marketers-with-mac-address-randomization/.

[156] M. V. Barbera, A. Epasto, A. Mei, V. C. Perta, and J. Stefa, “Signals from the Crowd: Uncovering Social Relationships through Smartphone Probes”, in *Proceedings of the 2013 Conference on Internet Measurement Conference*, 2013.

[157] M. V. Barbera, A. Epasto, A. Mei, S. Kosta, V. C. Perta, and J. Stefa, *CRAWDAD dataset sapienza/probe-requests (v. 2013-09-10)*, 2013.

[158] M. Cunche, M.-A. Kaafar, and R. Boreli, “Linking wireless devices using information contained in Wi-Fi probe requests”, *Pervasive and Mobile Computing*, vol. 11, 2014.

[159] N. Cheng, X. O. Wang, W. Cheng, P. Mohapatra, and A. Seneviratne, “Characterizing privacy leakage of public wifi networks for users on travel”, in *2013 Proceedings IEEE INFOCOM*, 2013.

[160] J. Martin, E. Rye, and R. Beverly, “Decomposition of MAC Address Structure for Granular Device Inference”, in *Proceedings of the 32nd Annual Conference on Computer Security Applications*, 2016.

[161] J. Freudiger, “How Talkative is Your Mobile Device? An Experimental Study of Wi-Fi Probe Requests”, in *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, 2015.

[162] A. Di Luzio, A. Mei, and J. Stefa, “Mind your probes: De-anonymization of large crowds through smartphone WiFi probe requests”, in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, IEEE, 2016.

[163] X. Gu, W. Wu, X. Gu, Z. Ling, M. Yang, and A. Song, "Probe Request based Device Identification Attack and Defense", *Sensors*, vol. 20, no. 16, 2020.

[164] *IEEE 802.11aq-2018 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Preassociation Discovery*. [Online]. Available: https://standards.ieee.org/standard/802_11aq-2018.html.

[165] *Implementing MAC Randomization*. [Online]. Available: https://source.android.com/devices/tech/connect/wifi-mac-randomization.

[166] *MAC Randomization Behavior*. [Online]. Available: https://source.android.com/devices/tech/connect/wifi-mac-randomization-behavior.

[167] C. Huitema, *MAC address randomization in Windows 10*, Dec. 2015. [Online]. Available: https://huitema.wordpress.com/2015/12/31/mac-address-randomization-in-windows-10/.

[168] Apple, *Use private wi-fi addresses on iPhone, iPad, iPod Touch, and Apple Watch*, Nov. 2021. [Online]. Available: https://support.apple.com/en-us/HT211227.

[169] E. Grumbach, *iwlwifi: mvm: support random MAC address for scanning*, Commit effd05ac479b, Nov. 2014. [Online]. Available: https://github.com/torvalds/linux/commit/effd05a.

[170] Apple, *Apple Platform Security: Wi-Fi Privacy*. [Online]. Available: https://support.apple.com/en-gb/guide/security/secb9cb3140c/web.

[171] C. Zhang and Q.-S. Jia, "An Occupancy Distribution Estimation Method Using the Surveillance Cameras in Buildings", in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, IEEE, 2017.

[172] G. H. Khan and M. A. Rahman, "Room Occupancy Detection from Temperature, Light, Humidity, and Carbon Dioxide Measurements Using Deep Learning", in *2021 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2)*, IEEE, 2021.

[173] C. Perra, A. Kumar, M. Losito, P. Pirino, M. Moradpour, and G. Gatto, "Monitoring Indoor People Presence in Buildings Using Low-Cost Infrared Sensor Array in Doorways", *Sensors*, vol. 21, no. 12, 2021.

[174] Y. Yuan, X. Li, Z. Liu, and X. Guan, "Occupancy Estimation in Buildings Based on Infrared Array Sensors Detection", *IEEE Sensors Journal*, vol. 20, no. 2, 2019.

[175] M. Jin, R. Jia, and C. J. Spanos, "Virtual Occupancy Sensing: Using Smart Meters to Indicate Your Presence", *IEEE Transactions on Mobile Computing*, vol. 16, no. 11, 2017.

[176] A. R. Pratama, F. J. Blaauw, A. Lazovik, and M. Aiello, "Office Low-Intrusive Occupancy Detection Based on Power Consumption", *IEEE Access*, vol. 9, 2021.

[177] L. C. C. Desmond, C. C. Yuan, T. C. Pheng, and R. S. Lee, "Identifying Unique Devices Through Wireless Fingerprinting", in *Proceedings of the first ACM conference on Wireless network security*, 2008.

[178] T. Fryza, T. Bravenec, and Z. Kohl, "Security and Reliability of Room Occupancy Detection Using Probe Requests in Smart Buildings", in *2023 33rd International Conference Radioelektronika (RADIOELEKTRONIKA)*, IEEE, 2023.

[179] T. Baji, "Evolution of the GPU Device widely used in AI and Massive Parallel Processing", in *2018 IEEE 2nd Electron Devices Technology and Manufacturing Conference (EDTM)*, IEEE, 2018.

[180] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Communications of the ACM*, vol. 60, no. 6, 2017.

[181] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A Large-Scale Hierarchical Image Database", in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Ieee, 2009.

[182] Martín Abadi *et al.*, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, Software available from tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/.

[183] K. Cao, Y. Liu, G. Meng, and Q. Sun, "An Overview on Edge Computing Research", *IEEE access*, vol. 8, 2020.

[184] Wikipedia contributors, *CUDA — Wikipedia, The Free Encyclopedia*, https://en.wikipedia.org/w/index.php?title=CUDA&oldid=1000899800, [Online; accessed 2-January-2021], 2021.

[185] S. Markidis, S. W. Der Chien, E. Laure, I. B. Peng, and J. S. Vetter, "NVIDIA Tensor Core Programmability, Performance & Precision", in *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, IEEE, 2018.

[186] Embedded Linux Wiki, *Jetson*, https://elinux.org/index.php?title=Jetson&oldid=543051, [Online; accessed 29-January-2021], 2021.

[187] Wikipedia contributors, *Half-precision floating-point format — Wikipedia, The Free Encyclopedia*, https://en.wikipedia.org/w/index.php?title=Half-precision_floating-point_format&oldid=993484104, [Online; accessed 10-January-2021], 2020.

[188] Arm Limited, *Instruction Sets: Floating Point*, [Online; accessed 16-January-2021]. [Online]. Available: https://developer.arm.com/architectures/instruction-sets/floating-point.

[189] F. Zafari, A. Gkelias, and K. K. Leung, "A Survey of Indoor Localization Systems and Technologies", *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, 2019.

[190] F. Liu, J. Liu, Y. Yin, W. Wang, D. Hu, P. Chen, and Q. Niu, "Survey on WiFi-based indoor positioning techniques", *IET communications*, vol. 14, no. 9, 2020.

[191] A. Poulose, J. Kim, and D. S. Han, "A sensor fusion framework for indoor localization using smartphone sensors and Wi-Fi RSSI measurements", *Applied Sciences*, vol. 9, no. 20, 2019.

[192] Y. Wang and K. Ho, "Unified near-field and far-field localization for AOA and hybrid AOA-TDOA positionings", *IEEE Transactions on Wireless Communications*, vol. 17, no. 2, 2017.

[193] O. Abdul-Latif, P. Shepherd, and S. Pennock, "TDOA/AOA data fusion for enhancing positioning in an ultra wideband system", in *2007 IEEE International Conference on Signal Processing and Communications*, IEEE, 2007.

[194] E. Gönültaş, E. Lei, J. Langerman, H. Huang, and C. Studer, "CSI-based multi-antenna and multi-point indoor positioning using probability fusion", *IEEE Transactions on Wireless Communications*, vol. 21, no. 4, 2021.

[195] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni, "CSI-based indoor localization", *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, 2012.

[196] D. He, T. Bouras, X. Chen, W. Yu, Y. Zhang, and Y. Yang, "3-D spatial spectrum fusion indoor localization algorithm based on CSI-UCA smoothing technique", *IEEE Access*, vol. 6, 2018.

[197] J. Geng, L. Xia, J. Xia, Q. Li, H. Zhu, and Y. Cai, "Smartphone-based Pedestrian Dead Reckoning for 3D Indoor Positioning", *Sensors*, vol. 21, no. 24, 2021.

[198] S. Jeong, J. Min, and Y. Park, "Indoor Positioning Using Deep-Learning-Based Pedestrian Dead Reckoning and Optical Camera Communication", *IEEE Access*, vol. 9, 2021.

[199] A. Tsanousa, V.-R. Xefteris, G. Meditskos, S. Vrochidis, and I. Kompatsiaris, "Combining rssi and accelerometer features for room-level localization", *Sensors*, vol. 21, no. 8, 2021.

[200] H. Mehrabian and R. Ravanmehr, "Sensor fusion for indoor positioning system through improved RSSI and PDR methods", *Future Generation Computer Systems*, 2022.

[201] J. Li, Y. Wang, Z. Chen, L. Ma, and S. Yan, "Improved Height Estimation Using Extended Kalman Filter on UWB-Barometer 3D Indoor Positioning System", *Wireless Communications and Mobile Computing*, vol. 2021, 2021.

[202] T. Bravenec and T. Fryza, "Reducing Memory Requirements of Convolutional Neural Networks for Inference at the Edge", in *2021 31st International Conference Radioelektronika (RADIOELEKTRONIKA)*, IEEE, 2021.

[203] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[204] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[205] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, *ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design*, arXiv preprint arXiv:1807.11164, 2018. arXiv: 1807.11164 [cs.CV].

[206] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[207] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library", in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[208] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python", *Nature methods*, vol. 17, no. 3, 2020.

[209] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls", *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, 1996.

[210] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in Python", *the Journal of machine Learning research*, vol. 12, 2011.

[211] E. Schulz, M. Speekenbrink, and A. Krause, "A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions", *Journal of Mathematical Psychology*, vol. 85, 2018.

[212] A. Bekkali, T. Masuo, T. Tominaga, N. Nakamoto, and H. Ban, "Gaussian Processes for Learning-based Indoor Localization", in *2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, IEEE, 2011.

[213] S. Liu, R. De Lacerda, and J. Fiorina, "Performance Analysis of Adaptive K for Weighted K-Nearest Neighbor Based Indoor Positioning", in *2022 IEEE 95th Vehicular Technology Conference:(VTC2022-Spring)*, IEEE, 2022.

[214] J. Hu, D. Liu, Z. Yan, and H. Liu, "Experimental Analysis on Weight K-Nearest Neighbor Indoor Fingerprint Positioning", *IEEE Internet of Things Journal*, vol. 6, no. 1, 2018.

[215] H. Zou, M. Jin, H. Jiang, L. Xie, and C. J. Spanos, "WinIPS: WiFi-Based Non-Intrusive Indoor Positioning System With Online Radio Map Construction and Adaptation", *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, 2017.

[216] X. Liang, X. Gou, and Y. Liu, "Fingerprint-Based Location Positoning Using Improved KNN", in *2012 3rd IEEE international conference on network infrastructure and digital content*, IEEE, 2012.

[217] ISO, "Information Technology – Real Time Locating Systems – Test and Evaluation of Localization and Tracking Systems (ISO/IEC 18305:2016)", *International Organization for Standardization*, 2016.

[218] F. Potorti, A. Crivello, P. Barsocchi, and F. Palumbo, "Evaluation of Indoor Localisation Systems: Comments on the ISO/IEC 18305 Standard", in *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2018.

[219] F. Potorti, A. Crivello, and F. Palumbo, "The EvAAL Evaluation Framework and the IPIN Competitions", in *Geographical and Fingerprinting Data to Create Systems for Indoor Positioning and Indoor/Outdoor Navigation*, Elsevier, 2019.

Acronyms

5G Fifth-Generation

AoA Angle of Arrival

ANN Artificial Neural Network

AoD Angle of Departure

AoI Area of Interest

AP Access Point

A-WEAR A network for dynamic WEarable Applications with pRivacy constraints

aw$k$NN Adaptive Weighted k-Nearest Neighbors

BLE Bluetooth Low Energy

CDF Cumulative Distribution Function

CNN Convolutional Neural Network

CPU Central Processing unit

CSI Channel State Information

CSV Comma Separated-Values

CV Computer Vision

D2D Device-to-Device

DSSS Direct Sequence Spread Spectrum

DT Digital Twins

dwfw$k$NN Distance & Feature Weighted k-Nearest Neighbors

EJD European Joint Doctorate

ESR Early Stage Researcher

FHSS Frequency Hopping Spread Spectrum

GNSS Global Navigation Satellite Systems

GPIO General-Purpose Input/Output

GPR Gaussian Process Regression

GPU Graphical Processing Unit

HID Human Interface Device

HMM Hidden Markov Models

IPS Indoor Positioning System

IoT Internet of Things

IR Infrared

ISR Interrupt Service Routine

ITN Innovative Training Network

kNN k-Nearest Neighbors

LBS Location Based Services

LID Linearly Interpolated Data

LoS Line of Sight

MAE Mean Absolute Error

MCU Microcontroller Unit

MD Measured Data

ML Machine Learning

MAC Media Access Control

NTP Network Time Protocol

OFDM Orthogonal Frequency Division Multiplexing

OUI Organizationally Unique Identifier

PAN Personal Area Network

PIR Passive Infrared

PNL Preferred Network List

PoI Point of Interest

PtP Point-to-Point

RAM Random Access Memory

ReLU Rectified Linear Unit

RM Radio Map

RFID Radio Frequency Identification

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

RP Reference Point

RQ Rational Quadratic

RSSI Received Signal Strength Indicator

saw$k$NN Self-Adaptive Weighted k-Nearest Neighbors

SE Squared Exponential

SSID Service Set Identifier

sti-kNN Signal Tendency Index – Weighted k-Nearest Neighbors

SVM Support Vector Machines

ToA Time of Arrival

ToF Time of Flight

TDoA Time Difference of Arrival

UAV unmanned Aerial Vehicle

UE User Equipment

UUID Universal Unique Identifier

UUID-E Universally Unique IDentifier-Enrollee

UWB Ultra-Wideband

wkNN Weighted k-Nearest Neighbors

WLAN Wireless Local Area Network

WPS Wi-Fi Protected Setup

WWAN Wireless Wide Area Network

XR eXtended Reality