2021

Deep Learning Methods for Fingerprint-Based Indoor and Outdoor Positioning

Fahad Alhomayani

DEEP LEARNING METHODS FOR FINGERPRINT-BASED INDOOR AND OUTDOOR POSITIONING

A Dissertation

Presented to

the Faculty of the Daniel Felix Ritchie School of Engineering and Computer Science

University of Denver

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

by

Fahad Alhomayani

August 2021

Advisor: Dr. Mohammad H. Mahoor

Author: Fahad Alhomayani
Title: DEEP LEARNING METHODS FOR FINGERPRINT-BASED INDOOR AND OUTDOOR POSITIONING
Advisor: Dr. Mohammad H. Mahoor
Degree Date: August 2021

ABSTRACT

Outdoor positioning systems based on the Global Navigation Satellite System have several shortcomings that have deemed their use for indoor positioning impractical. Location fingerprinting, which utilizes machine learning, has emerged as a viable method and solution for indoor positioning due to its simple concept and accurate performance. In the past, shallow learning algorithms were traditionally used in location fingerprinting. Recently, the research community started utilizing deep learning methods for fingerprinting after witnessing the great success and superiority these methods have over traditional/shallow machine learning algorithms. The contribution of this dissertation is fourfold:

First, a Convolutional Neural Network (CNN)-based method for localizing a smartwatch indoors using geomagnetic field measurements is presented. The proposed method was tested on real world data in an indoor environment composed of three corridors of different lengths and three rooms of different sizes. Experimental results show a promising location classification accuracy of 97.77 % with a mean localization error of 0.14 meter (m).

Second, a method that makes use of cellular signals emitting from a serving eNodeB to provide symbolic indoor positioning is presented. The proposed method utilizes Denoising Autoencoders (DAEs) to mitigate the effects of cellular signal loss. The proposed method was evaluated using real-world data collected from two different smartphones inside a representative apartment of eight symbolic spaces. Experimen-

tal results verify that the proposed method outperforms conventional symbolic indoor positioning techniques in various performance metrics.

Third, an investigation is conducted to determine whether Variational Autoencoders (VAEs) and Conditional Variational Autoencoders (CVAEs) are able to learn the distribution of the minority symbolic spaces, for a highly imbalanced fingerprinting dataset, so as to generate synthetic fingerprints that promote enhancements in a classifier's performance. Experimental results show that this is indeed the case. By using various performance evaluation metrics, the achieved results are compared to those obtained by two state-of-the-art oversampling methods known as Synthetic Minority Oversampling TEchnique (SMOTE) and ADAptive SYNthetic (ADASYN) sampling.

Fourth, a novel dataset of outdoor location fingerprints is presented. The proposed dataset, named OutFin, addresses the lack of publicly available datasets that researchers can use to develop, evaluate, and compare fingerprint-based positioning solutions which can constitute a high entry barrier for studies. OutFin is comprised of diverse data types such as WiFi, Bluetooth, and cellular signal strengths, in addition to measurements from various sensors including the magnetometer, accelerometer, gyroscope, barometer, and ambient light sensor. The collection area spanned four dispersed sites with a total of 122 Reference Points (RPs). Before OutFin was made available to the public, several experiments were conducted to validate its technical quality.

ACKNOWLEDGMENTS

To Allah, The Most Gracious and The Most Merciful, who has sustained me through the most challenging years of my life, granting me countless blessings and endless opportunities that allowed me to embark on and finally complete this journey.

To Dr. Mohammad Mahoor, my advisor, for always believing in me, even when I had doubts, and for his unwavering and invaluable academic, moral, and personal support throughout this process. I have become a better engineer and a better person thanks to Dr. Mahoor. I owe him a deep debt of gratitude.

To Dr. Haluk Ogmen, Dr. Zhihui Zhu, and Dr. Michael Keables, my committee members, for their wonderful guidance, questions, and suggestions to improve my dissertation.

To all my labmates, for their friendship and the kindness they showed me over the years. I wish them all the best in their lives going forward.

To the faculty of the Department of Electrical & Computer Engineering, for creating an inclusive, inviting, and stimulating learning environment.

To my parents, siblings, and friends in Saudi Arabia, for their endless love and encouragement.

To Mohammad and Faisal, my newborn twin boys. You have made me feel more fulfilled than I could have ever imagined.

Finally, to my loving and beautiful wife, Ahlam Alzaedi. There are no words to adequately express my gratitude to you. Thank you for your countless sacrifices in helping me get to this point, for always being supportive of my dream to finish, for keeping me sane over the past years, and for being my editor and proofreader. I never could have gotten through this endeavor without you. You have been my rock, and I am excited to be yours as you embark on your own doctoral journey.

TABLE OF CONTENTS

Acknowledgments iv

List of Tables viii

List of Figures ix

List of Acronyms xi

Chapter 1: Introduction 1
- 1.1 The Fingerprinting Approach to Indoor Positioning . . . 3
- 1.2 Problem Statement and Contributions . . . 5
  - 1.2.1 Why Deep Learning for Fingerprinting . . . 6

Chapter 2: Improved Indoor Geomagnetic Field Fingerprinting for Smartwatch Localization Using Deep Learning 9
- 2.1 Introduction . . . 9
- 2.2 Related Work . . . 11
- 2.3 Dataset Analysis . . . 13
  - 2.3.1 Dataset Description . . . 13
  - 2.3.2 Smartwatch Dataset . . . 14
  - 2.3.3 Smartwatch Dataset Preprocessing . . . 15
- 2.4 Proposed CNN Architecture . . . 17
  - 2.4.1 CNNs . . . 17
  - 2.4.2 Input and Output of the System . . . 17
  - 2.4.3 Performance Evaluation Metrics . . . 18
- 2.5 Experiments and Results . . . 22
  - 2.5.1 Performance on the Testing Dataset . . . 22
  - 2.5.2 Prediction Latency . . . 25
  - 2.5.3 Softmax Layer and User Tracking . . . 26

Chapter 3: Deep Learning-based Symbolic Indoor Positioning using the Serving eNodeB 27
- 3.1 Introduction . . . 27
- 3.2 Related Work . . . 28
  - 3.2.1 Machine Learning for Symbolic Indoor Positioning . . . 29

3.2.2 Machine Learning for Cellular-based Indoor Positioning 30
3.3 Dataset Description and Validation 32
3.3.1 Data Collection Platform 33
3.3.2 Data Collection Environment 34
3.3.3 Data Collection Procedure 34
3.3.4 Technical Validation 35
3.4 Background and Proposed Method 37
3.4.1 Autoencoders (AEs) 37
3.4.2 Proposed Method 37
3.5 Experiments and Results 40
3.5.1 Performance Evaluation 40
3.5.2 Effect of p_{loss} on Accuracy 41
3.5.3 Effect of Device Heterogeneity on Accuracy 42

Chapter 4: Oversampling Highly Imbalanced Indoor Positioning Data using Deep Generative Models 44
4.1 Introduction 44
4.2 Dataset Description 45
4.3 Experimental Setup 46
4.4 Discussion 49

Chapter 5: OutFin: A Multi-Device and Multi-Modal Dataset for Outdoor Localization Based on the Fingerprinting Approach 52
5.1 Introduction 52
5.2 Methods 55
5.2.1 Data Acquisition Platform 55
5.2.2 Data Collection Environment 56
5.2.3 Procedure 58
5.3 Data Records 61
5.4 Technical Validation 67
5.4.1 Measurement Reliability 67
5.4.2 Measurement Validity 69
5.4.3 Usage Examples 71
5.4.4 Code Availability 75

Chapter 6: Conclusion and Future Work 77

Publications 80

References 82

Appendices 101
Appendix A: Deep Learning Preliminaries 101

A.1 The Principals of Learning . . . 102
A.2 Learning Approaches . . . 105
A.3 Parameters vs. Hyperparameters . . . 105
A.4 General Guidelines for Training . . . 106
A.5 Deep Learning Architectures . . . 107
A.6 Deep Learning Software Frameworks . . . 112
Appendix B: Indoor Fingerprint Types . . . 117
B.1 Radio Frequency Fingerprints . . . 117
B.2 Magnetic Field and Inertial Measurement Unit (IMU) Fingerprints . . . 122
B.3 Image Fingerprints . . . 127
B.4 Hybrid Fingerprints . . . 130
B.5 Miscellaneous Fingerprints . . . 131
Appendix C: Indoor Positioning Datasets . . . 134
C.1 Radio Frequency Datasets . . . 135
C.2 Magnetic Field Datasets . . . 140
C.3 Image Datasets . . . 141
C.4 Hybrid Datasets . . . 142
Appendix D: Dedication . . . 145

LIST OF TABLES

2.1 Smartwatch dataset statistics after filtering 16
2.2 Maximal Information Coefficient (MIC) between R_x, R_y, R_z and (x, y) of the grid points 17
2.3 Architectural specifications and performance statistics 21
2.4 Performance comparison on the testing dataset 24
2.5 Performance on the testing set using different loss functions 25

3.1 Field labels of data samples and their description 36
3.2 Results of the correlation analysis (1) 36
3.3 Results of the correlation analysis (2) 37
3.4 Hyperparameter tuning 39
3.5 Results of the device heterogeneity analysis 43

4.1 Examples of imbalanced indoor positioning datasets 44
4.2 Downstream classifier results 50

5.1 Main aspects of publicly available fingerprinting datasets 54
5.2 Results of the correlation analysis (1) 68
5.3 Results of the correlation analysis (2) 71
5.4 Descriptive statistics of the OutFin dataset 72
5.5 Performance evaluation of commonly used algorithms for positioning 75

A.1 A comparison of the leading open-source deep learning frameworks . 114

B.1 WiFi Access Point (AP) vs. Bluetooth Low Energy (BLE) beacon . 120

C.1 A side-by-side comparison of the datasets (1) 135
C.2 A side-by-side comparison of the datasets (2) 136
C.3 The pros, cons, and download link for each dataset 137

LIST OF FIGURES

1.1 Number of published indoor positioning articles by year 3
1.2 An illustration of the fingerprinting approach to indoor positioning . 5

2.1 Indoor environment map and the data acquisition paths 14
2.2 325 equally separated/uniquely identifiable grid points 15
2.3 Training vs. validation loss for all models 22
2.4 Proposed system architecture 23
2.5 Empirical cumulative positioning error 24

3.1 The general scheme of the proposed method 28
3.2 A picture of Samsung Galaxy S10+ (Phone 1) and Google Pixel 4 (Phone 2) attached to the tripod 33
3.3 Layout of the apartment where data was collected 34
3.4 Plots of cellular data showing examples of outliers and data loss . . . 35
3.5 Symbolic space prediction results 40
3.6 The effect of p_{loss} on accuracy 43

4.1 A graphical representation of the collection environment 47
4.2 Examples of fingerprints transformed into recurrence plots 48
4.3 Scheme of the experimental setup 50

5.1 Pictures of the four sites where data was collected 56
5.2 An aerial map of the collection environment 58
5.3 Directory tree of the OutFin dataset 62
5.4 Visualization of the data collected by Phone 1 and Phone 2 70
5.5 Interpolation of the magnetic field magnitude 73
5.6 3D codes of WiFi Received Signal Strength (RSS) measurements . . . 74
5.7 Noisy vs. denoised features for positioning 76

A.1 The computational model of an artificial neuron 102
A.2 Equations and the corresponding plots of some activation functions . 103
A.3 An Fully Connected (FC) network with three hidden layers 103
A.4 The structure of a basic CNN 107
A.5 An Recurrent Neural Network (RNN) with a single core cell 109

A.6 The working principal of a Generative Adversarial Network (GAN) . 110
A.7 The architecture of an AE and a DAE 111
A.8 Several Restricted Boltzmann Machiness (RBMs) are stacked to form a Deep Belief Network (DBN) 112
A.9 The nine most popular deep learning frameworks 113
B.1 Channel State Information (CSI) vs. RSS 119
B.2 The macro-cell layout 122
B.3 Temporal stability of magnetic field measurements 123
B.4 The large and small grids. Figure reproduced from [27]. 124
B.5 The X, Y, and Z axes relative to a typical smartphone 125
B.6 The influence of elevators on the magnetic field 126
B.7 The influence of a mobile phone on the magnetic field 126
B.8 The two main approaches to image-based indoor positioning 128
B.9 Energy efficiency achieved by using hybrid fingerprints 129

LIST OF ACRONYMS

cm centimetre

dBm decibel-milliwatt

dB decibel

° degree

GHz gigahertz

hPa hectopascal

Hz hertz

lx lux

MHz megahertz

m meter

µlx microlux

µT microtesla

ns nanosecond

s second

m/s^2 meters per second squared

rad/s radian per second

AAL Ambient Assisted Living

ADASYN ADAptive SYNthetic

AE Autoencoder

AP Access Point

API Application Programming Interface

App 1 WiFi Analyzer Pro

App 2 Bluetooth Scanner Extreme Edition

App 3 NetMonitor Pro

App 4 Physics Toolbox Sensor Suite Pro

BLE Bluetooth Low Energy

BM Boltzmann Machines

BS Base Station

BSSID Basic Service Set IDentifier

CDF Cumulative Distribution Function

CGAN Conditional Generative Adversarial Network

CNN Convolutional Neural Network

CNTK Microsoft Cognitive Toolkit

CPU Central Processing Unit

CSI Channel State Information

CSV Comma-Separated Values

CV Cross Validation

CVAE Conditional Variational Autoencoder

DAE Denoising Autoencoder

DBN Deep Belief Network

DTW Dynamic Time Warping

E-UTRAN Evolved-UMTS Terrestrial Radio Access Network

EARFCN E-UTRAN Absolute Radio Frequency Channel Number

ECI E-UTRAN Cell Identifier

eNB E-UTRAN NodeB

FC Fully Connected

FCC Federal Communications Commission

GAN Generative Adversarial Network

GLONASS GLObal NAvigation Satellite System

GNSS Global Navigation Satellite System

GPS Global Positioning System

GPU Graphical Processing Unit

GRU Gated Recurrent Unit

HMM Hidden Markov Model

ILBS Indoor Location-Based Services

IMU Inertial Measurement Unit

IoT Internet of Things

IPIN The International Conference on Indoor Positioning and Indoor Navigation

kNN k-Nearest Neighbor

LBS Location-Based Services

LED Light Emitting Diode

LoS Line-of-Sight

LSTM Long Short-Term Memory

LTE Long-Term Evolution

M2M Machine-to-Machine

MAC Media Access Control

MAE Mean Absolute Error

MIC Maximal Information Coefficient

MIMO Multiple-Input Multiple-Output

NLoS Non-Line-of-Sight

OFDM Orthogonal Frequency-Division Multiplexing

PCA Principal Component Analysis

Phone 1 Samsung Galaxy S10+

Phone 2 Google Pixel 4

PLMN Public Land Mobile Network

RBF Radial Basis Function

RBM Restricted Boltzmann Machines

ReLU Rectified Linear Unit

RFID Radio Frequency IDentification

RGB Red-Green-Blue

RGB-D Red-Green-Blue-Depth

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

RP Reference Point

RSRP Reference Signal Received Power

RSRQ Reference Signal Received Quality

RSS Received Signal Strength

SIFT Scale-Invariant Feature Transform

SIG Special Interest Group

SINR Signal to Interference and Noise Ratio

SMOTE Synthetic Minority Oversampling TEchnique

SSID Service Set IDentifier

SURF Speeded Up Robust Features

SVM Support Vector Machine

TA Timing Advance

tanh hyperbolic tangent

TPU Tensor Processing Unit

UMTS Universal Mobile Telecommunications Service

UPINLBS The International Conference on Ubiquitous Positioning, Indoor Navigation and Location-Based Services

UUID Universally Unique Identifier

UWB Ultra-Wide Band

VAE Variational Autoencoder

VLC Visible Light Communication

WNIC Wireless Network Interface Card

WPNC The Workshop on Positioning, Navigation and Communication

CHAPTER 1
INTRODUCTION

Over the past two decades, the limitations satellite-based outdoor positioning systems (e.g., Global Positioning System (GPS), Galileo, GLObal NAvigation Satellite System (GLONASS)) have for indoor use [1] led researchers to propose a wide variety of indoor positioning systems. Indoor positioning or indoor localization is the process of determining one's indoor location with respect to a predefined frame of reference. Indoor navigation relies on positioning updates to reach a target location from the current location. All indoor positioning systems are designed to provide location information. Some go a step further to provide navigation capabilities. While the notion of location is broad, location information can generally be presented in one of four ways: physically, absolutely, relatively, and symbolically [2, 3]. Physical location is obtained with respect to a global reference frame (e.g., latitude and longitude in the geographic coordinate system). Absolute location is expressed with respect to a local reference frame and the resolution of the frame depends on grid size. Relative location expresses the user's proximity to known landmarks in the environment. Symbolic location expresses location in a natural-language way, thus, providing abstract information of where the user is (e.g., in the living room, in the kitchen, etc.).

A common theme in early indoor positioning systems is an infrastructure-based nature. In other words, early systems provide positioning by relying on special-

ized equipment that has to be deployed throughout the environment and carried by users. Such equipment include ultrasonic transmitters, infrared badges, and Radio Frequency IDentification (RFID) tags [2, 3]. In contrast, the most recent systems are either infrastructure-free or take advantage of the already deployed infrastructure (e.g., WiFi APs). These systems rely on the various sensors and modules found in users' smartphones to provide indoor positioning [4, 5]. Infrastructure-free positioning systems do not necessitate deployed hardware in the environment to operate. Examples of such systems include magnetic field-based systems and camera-based systems (if artificial markers are not required for positioning).

Designing an indoor positioning system has remained a challenging task since indoor environments are very complex and are often characterized by Non-Line-of-Sight (NLoS) settings, moving people and furniture, walls of different densities, and the presence of different indoor appliances that alter indoor signal propagation. Nevertheless, the demand for more complete solutions is higher than ever before. This demand is fueled by a multitude of potential applications and services enabled by indoor positioning. Indoor positioning is a key enabling technology for many domains including Indoor Location-Based Services (ILBS) [6], Internet of Things (IoT) [7], Ambient Assisted Living (AAL) [8], indoor emergency responders navigation [9], and occupancy detection for the energy-efficient control of buildings [10]. Attempting to satisfy the demand, researchers are forced to compromise between different design criteria (e.g., accuracy, precision, privacy, scalability, complexity, cost, etc.[3]). To date, no universally agreed upon solution has emerged to solve the indoor positioning problem. Because of this, indoor positioning research is vibrant. Researchers share their work in dedicated conferences such as, the International Conference on The International Conference on Indoor Positioning and Indoor Navigation (IPIN); The International Conference on Ubiquitous Positioning, Indoor Navigation and Location-

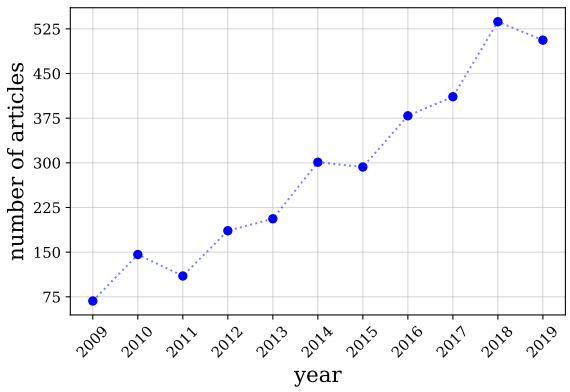

Figure 1.1: The number of published articles in IEEE Xplore by year (from 2009 to 2019) where authors used "indoor positioning", "indoor localization", or "indoor navigation" as a keyword.

Based Services (UPINLBS); and The Workshop on Positioning, Navigation and Communication (WPNC). As seen in Figure 1.1, the body of literature published in these conferences' proceedings, as well as at other venues and in other journals, continues to grow each year.

1.1 The Fingerprinting Approach to Indoor Positioning

Various approaches for indoor positioning have been proposed over the years. The main methods introduced include *angulation*, *lateration*, *proximity detection*, *pedestrian dead reckoning*, and *location fingerprinting*. Amongst these, the latter has recently received significant attention as a straightforward, inexpensive, and accurate approach for indoor positioning. Location fingerprinting, also referred to as *scene analysis*, or fingerprinting, employs low-power sensors that are integrated into smartphones and exploits existing infrastructure, such as WiFi APs, to achieve high positioning accuracy even in NLoS settings. The location of these APs is not a prerequisite for positioning, which eliminates the need to model complex indoor signal

propagation [11]. Moreover, fingerprinting systems are immune to accumulated positioning errors caused by IMU drifts [12].

The concept of fingerprinting is identifying indoor spatial locations based on location-dependent measurable features (location fingerprints). There are different types of fingerprints such as *radio frequency* fingerprints[13], *magnetic field* fingerprints [14], *image* fingerprints [15], and *hybrid* fingerprints [16]. Radio frequency fingerprints, particularly WiFi fingerprints, are, undoubtedly, the most used fingerprints.

From an implementation perspective, the fingerprinting approach to indoor positioning is a two-phase process that consists of an *offline phase* and an *online phase*. During the offline phase, *site surveying*, in which the fingerprints of the area of interest are sampled at predefined RPs, is performed. The fingerprints are sampled using smartphone sensors. For example, the WiFi module and the magnetometer are used to collect RSS and magnetic field fingerprints, respectively. The sampled fingerprints, along with their corresponding coordinates, are stored in a database. The data is then used to train a machine learning algorithm to learn a function that best maps the sampled fingerprints to their correct coordinates. The learned function is then used during the online phase to infer a user's coordinates given the measured fingerprints at the user's location. The process of fingerprinting is visually depicted in Figure 1.2.

The main source of error in fingerprinting systems is due to *location ambiguity*. Location ambiguity refers to the problem of different RPs exhibiting similar fingerprints [17]. Local ambiguity occurs when adjacent RPs have similar fingerprints, while global ambiguity occurs when distant RPs have similar fingerprints. As discussed later, different fingerprint types may suffer from one ambiguity more than the other. For example, WiFi fingerprints are generally immune to global ambiguity but prone to local ambiguity, while the contrary is true for magnetic field fingerprints.

Based on the number of samples needed for online positioning, a given system can be classified as either *one-shot* or *multi-shot* [18]. In a one-shot system, a location is estimated using only a single fingerprint sample; while in a multi-shot system, two or more samples (i.e., consecutive measurements) are required to refine the positioning estimate. Due to the time spent obtaining the additional samples and the pre/post-processing involved, multi-shot systems are generally slower but more accurate than one-shot systems.

1.2 Problem Statement and Contributions

Classical learning algorithms such as k-Nearest Neighbor (kNN), Naïve Bayes, and Decision Trees have traditionally been utilized for location fingerprinting [19, 20, 21, 22]. However, as compared to deep learning, such algorithms have several limitations that limit the applicability of indoor positioning (see Section 1.2.1 for details). Therefore, inspired by the success that deep learning methods have achieved in various research fields, this dissertation proposes the application of deep learning methods with the aim to:

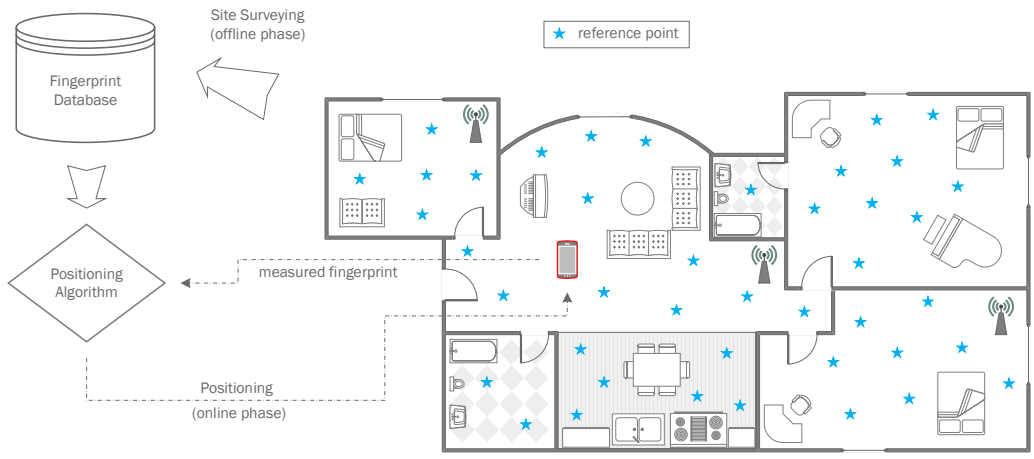

Figure 1.2: An illustration of the fingerprinting approach to indoor positioning.

1. Improve the accuracy and prediction latency of magnetic field-based positioning.

2. Mitigate the effects of cellular signal loss on symbolic positioning.

3. Improve prediction performance for imbalanced positioning datasets.

Additionally, motivated by the lack of publicly available datasets that researchers can use to develop, evaluate, and compare fingerprint-based positioning solutions, we propose OutFin, a publicly available, multi-device and multi-modal outdoor fingerprinting dataset.

1.2.1 Why Deep Learning for Fingerprinting

Listed below are some powerful deep learning algorithms properties and their positive implications on location fingerprinting:

1. Deep learning techniques often provide an end-to-end solution where the task of feature extraction is automatically performed and implicitly embedded in the architecture, avoiding the need for hand-engineered features, a time-consuming and knowledge-demanding process. This property is particularly crucial when dealing with high-dimensional and not-easily extractable features that are required for radio frequency and image fingerprinting.

2. Deep learning is well-known for effectively and efficiently processing massive amounts of raw data, a task otherwise difficult, if not impossible. In fact, the predictive performance of deep learning algorithms enhances with increased training samples. Consequently, there is no limit to the amount of fingerprint data used for training.

3. The parametric nature of deep learning, where computational complexity does not depend on dataset size and the ability to parallelize computation using

Graphical Processing Units (GPUs) results in infinitesimal inference latency (in the orders of milliseconds or less), makes deep learning algorithms ideal for real-time positioning applications. However, this often comes at the expense of a prolonged training phase.

4. Deep learning is the method of choice for classification/regression problems in which the nature of boundaries describing the features in input space is highly complex and nonlinear. This is the case in fingerprinting where the overarching goal is to distinguish between spatial locations that are, in many cases, separated by a few centimeters or less.

5. Deep learning is well-suited for transfer learning which involves transferring knowledge from pre-trained networks to minimize data collection and training efforts. Therefore, a fingerprinting system can be realized with minimal cost. In this regard, unsupervised and semi-supervised deep learning methods have also proven successful when the fingerprint data is scarce or unlabeled.

The dissertation's contributions are in line with its general organization:

- Chapter 2 proposes a CNN-based method for localizing a smartwatch indoors using geomagnetic field fingerprints, discusses the method's architecture, and compares the positioning accuracy and prediction latency achieved by the method to those achieved by two classical learning algorithms.

- Chapter 3 introduces an AE-based method to deal with incomplete measurements caused by unpredictable cellular signal loss in a symbolic positioning setting. Moreover, this chapter investigates the effects of varying p_{loss}, a parameter that controls the severity of signal loss, on positioning performance.

- Chapter 4 proposes a VAE-based method for oversampling highly imbalanced indoor positioning datasets. To quantify the gain in performance achieved by the proposed method, a baseline is constructed using a positioning model trained on imbalanced data. Furthermore, all results are compared against two oversampling techniques.

- Chapter 5 presents OutFin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach, conducts several experiments to validate OutFin's technical quality, and discusses some of the application domains that OutFin can be used for.

- Chapter 6 concludes the dissertation and suggests future research directions.

Additionally, for an overview of deep learning, including, among others, its architectures and software frameworks refer to Appendix A, for a review of various fingerprint types and a discussion of their advantages and disadvantages for indoor positioning refer to Appendix B, and for a review of indoor positioning datasets that are currently publicly available refer to Appendix C.

CHAPTER 2
IMPROVED INDOOR GEOMAGNETIC FIELD FINGERPRINTING FOR SMARTWATCH LOCALIZATION USING DEEP LEARNING

2.1 Introduction

The two main technologies that have been extensively used in conjunction with the fingerprinting approach are WiFi and Bluetooth. Advocates for using the RSS of these technologies as fingerprints have often overlooked several technical and practical flaws regarding the feasibility of real-world implementation as indoor positioning technologies. While each technology has its own shortcomings, for the sake of conciseness, we will only mention the shortcomings that both have in common. First, they require the deployment of a special infrastructure; meaning that without WiFi APs in the case of WiFi and without Bluetooth beacons in the case of Bluetooth, indoor positioning using these technologies is impossible. Second, the rapid hardware and software developments in both technologies often require the update or even the replacement of the exciting infrastructure, meaning that the laborious offline phase of constructing a radio map must be reperformed. Third, they use the already crowded 2.4 gigahertz (GHz) radio band so does other indoor appliances such as microwave ovens, cordless phones, and wireless baby monitors which directly translates into increased wireless signal interference. Fourth, due to the multipath effect, it is often observed that the measured RSS for the same indoor location is unstable and fluc-

tuates over time. Fifth, the RSS of all APs in the environment must be measured to accurately position a user; an energy consuming process not suitable for power-constrained devices such as smartphones. Furthermore, the swiftness of this process is not only bounded by the broadcasting rate of APs but bounded by the scan rate of the mobile device; making its applicability to real-time user tracking questionable.

On the other hand, using the anomalies of the geomagnetic field as fingerprints eliminates the shortcomings of the aforementioned technologies. The indoor geomagnetic field is very stable and does not need the deployment of a special infrastructure to be realized. Furthermore, the measuring of the geomagnetic field is instantaneous and requires only a magnetometer which modern day smart devices come equipped with. These appealing features of geomagnetic field fingerprinting have attracted researchers over the past years as a promising alternative for indoor localization [23, 24, 25, 26].

Here, we treat the indoor positioning problem as a multiclass classification problem. Each grid point in the environment has its own geomagnetic features and should be considered as a unique class. To distinguish one class from another i.e., one grid point from another, we propose a CNN-based approach for accurate and fast positioning.

The remainder of this chapter is organized as follows. Section 2.2 reviews some of the pioneering work in geomagnetic fingerprinting. Section 2.3, analyzes the dataset used in this study. Section 2.4, discusses the design and development of the proposed system. Section 2.5, reports on the evaluation experiments and analyzes the results.

2.2 Related Work

The complex distortions to the indoors geomagnetic field caused by steel structures and reinforced concrete has proven to be very stable over long periods of time [23, 24, 27]. Moreover, these distortions have also been proven to vary significantly across space; in the orders of a few centimeters or less [27, 28]. This property of spatial instability and temporal stability provides the basis of using geomagnetic field distortions as unique signatures of indoor locations.

Among the first to realize that the incorrect heading information of an electronic compass can be used as signatures for indoor localization are Suksakulchai *et al.* [14]. They mounted an electronic compass on top of a service robot "HelpMate" and collected the heading information as the robot traverses a corridor. The next time the robot traverses the corridor, it matches its measured heading information with the pre-collected information; if a match is found, the robot can determine its position. This concept was later extended by Chung *et al.* [23]. They used four magnetometers placed four feet above the ground. Data was collected from a corridor and an atrium in grid map fashion with cells 60 centimetre (cm) apart. A balanced dataset was constructed with each cell having 480 samples corresponding to the four sensors in various directions. A Nearest Neighbor (NN) algorithm was used for positioning with a ratio of about 99 to 1 for training and testing, respectively. In addition to the raw magnetic vector, they have also used the unit and the norm vectors as features. A positioning accuracy of less than 1 m 75.7 % of the time was reported, and after applying a search space constraint the accuracy rose up to less than 1 m 88 % of the time.

Gozick *et al.* [24] used the build-in magnetometer of a smartphone to build magnetic maps of corridors inside buildings. These maps were constructed with the

smartphone's y-axis parallel to the north and a prior knowledge of the corridor's solid steel and steel-reinforced concrete pillars locations. They have only used the magnitude of the magnetic vector as a feature to differentiate between pillars or a group of consecutive pillars (magnetic landmarks). They have shown that the magnetic signature collected by different smartphones with different sampling rates have the same pattern. They have later expanded on this idea by developing a smartphone application "LocateMe" and using a Dynamic Time Warping (DTW) classifier with a sliding window to localize users walking inside corridors [25]. Depending on the corridor's length, the minimum walking distance required for localization ranges between 2.1 m and 6.5 m with mean estimation errors ranging between 0.7 m and 4.0 m. The prediction latency of the application is also corridor length dependent, with prediction latencies reaching up to 10 second (s).

Most recently, Lee and Han [26] tried to improve on the work of [24] by extending the concept of magnetic landmarks to 2D spaces instead of only corridors. Unlike [24], the locations of the environment's steel structures needed not be known; instead they considered a location to be a magnetic landmark if the magnetic field intensity of that location is either lower or higher than the average intensity of the entire space. This approach however has the drawback of having wide spaces with no landmarks. For example, in some locations, a user had to walk for up to 6 m before encountering a landmark given that the testbed is only 12 m by 22 m. They used a CNN as a landmark classifier and used a sequence of magnetic data measurements as input features. Their approach is based on inferring the location of a user if a landmark was classified correctly. However, how close or far a user is from a landmark was not reported; instead they reported a classification accuracy of 80.8 %.

In this chapter, we hypothesize that by exploiting the powerful properties of CNNs, indoor geomagnetic positioning can still be improved in terms of both location classifi-

cation accuracy and prediction latency. Our preliminary experiments with a publicly available dataset [29] validates our hypothesis. In addition, the measuring device used to collect the fingerprints is a smartwatch. To the best of our knowledge, a smartwatch has not been used to build a geomagnetic positioning system before.

2.3 Dataset Analysis

The dataset used to develop and evaluate the proposed system is publicly available and was introduced by Barsocchi *et al.* [29]. We found this dataset particularly interesting because the data was collected in a representative indoor environment consisting of multiple corridors and rooms. More importantly, one of the measuring devices used to collect data is a smartwatch. We found this of great interest since one of our intentions was to investigate the use of a smartwatch for indoor geomagnetic positioning; something that has not been attempted before.

2.3.1 Dataset Description

The dataset is multisource and multivariate. It is multisource because two synchronized mobile devices were used to collect the data, a smartphone (Sony Xperia M2) and a smartwatch (LG G Watch R). It is multivariate because different information was collected by these devices, such as WiFi fingerprints, geomagnetic field fingerprints, and inertial sensor data. The data collection process involved two campaigns performed over a surface area of $185.12\,\mathrm{m}^2$ at a constant sampling rate of $10\,\mathrm{hertz}$ (Hz). The map of the environment and the paths taken to collect the data are depicted in Figure 2.1. As shown, the indoor environment is composed of three corridors of different lengths and three rooms of different sizes. The data was collected over the predefined grid points shown as red bullets in Figure 2.2. These grid points

are equally separated by 0.6 m in x and y directions. There is a total of 325 grid points; each one uniquely identifiable by a "PlaceID" and local coordinates (x, y).

2.3.2 Smartwatch Dataset

As mentioned earlier, the dataset is multisource and multivariate. However, since we intend to develop an indoor geomagnetic field positioning system for smartwatch localization, we are only interested in a subset of the dataset, namely, the data samples collected by the smartwatch. Furthermore, out of the different information collected by the smartwatch i.e., linear acceleration, angular acceleration, geomagnetic field strength, and absolute orientation, we are only interested in geomagnetic field and orientation information. A single geomagnetic field sample consists of a vector of three orthogonal components. Each component represents the geomagnetic field

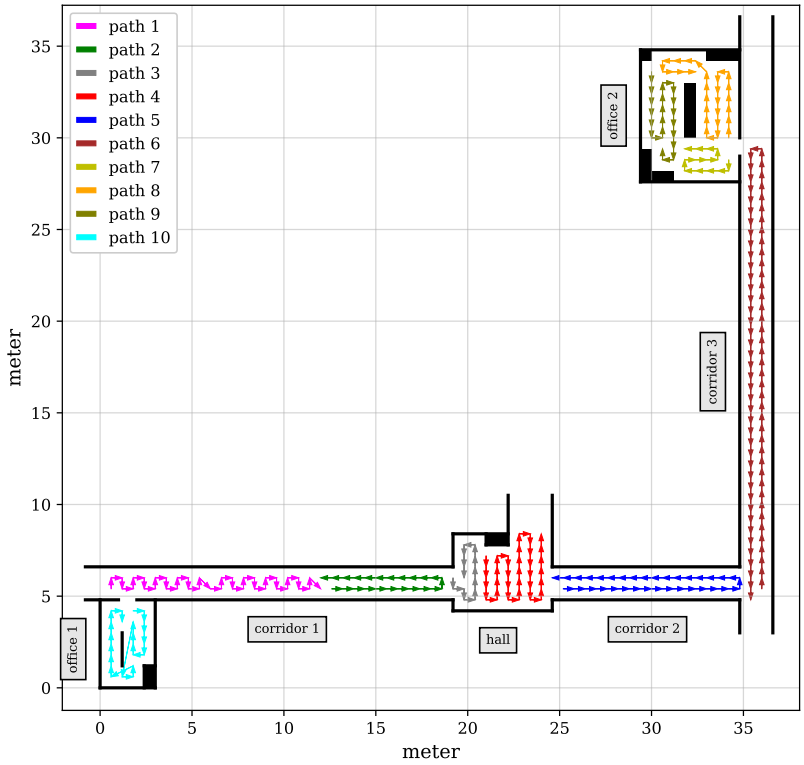

Figure 2.1: Indoor environment map and the data acquisition paths.

strength in microtesla (μT) with respect to the smartwatch's reference frame i.e., $\boldsymbol{B} = \left[B_x, B_y, B_z\right]$. A single orientation sample also consists of a vector of three orthogonal components. Each component represents the absolute orientation in degree (°) with respect to the smartwatch's reference frame i.e., $\boldsymbol{R} = \left[R_x, R_y, R_z\right]$.

2.3.3 Smartwatch Dataset Preprocessing

There are 58,374 continuous samples collected by the smartwatch during each of the first and second campaigns. After filtering these samples based on the arrival and departure timestamps at each grid point, only 11,354 and 10,667 samples are uniquely assignable to 317 grid points from the first and second campaigns, respectively. The arrival and departure timestamps of grid points (23, 33, 74, 103, 264, 273, 320, 325) were not reported in the dataset. Therefore, we could not assign any samples

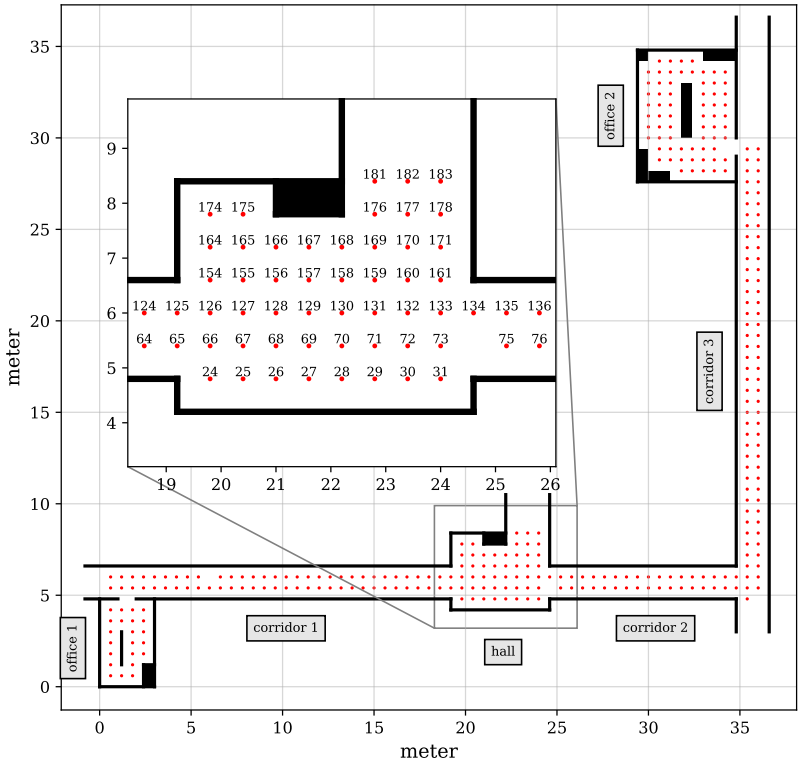

Figure 2.2: 325 grid points (equally separated and uniquely identifiable by a PlaceID).

to them and hence they were omitted. Table 2.1 shows some statistics about the smartwatch dataset after filtering. Clearly, the dataset is unbalanced but looking at the standard deviation of the combined dataset we consider this a nonissue. The remaining preprocessing steps are as follows:

1. To create a single dataset for training and testing the proposed system, we have combined both datasets.

2. The samples in the combined dataset were then randomly shuffled to ensure that the training and testing datasets are representative of the overall distribution of the combined dataset.

3. 80 % of the shuffled samples were allocated for training while the remaining 20 % were allocated for testing.

4. As the information of 8 grid points are missing, the grid points PlaceID were relabeled from 1 to 317.

5. Since the input features are measured in different units, their values were rescaled (normalized) between 0 and 1 using min-max normalization. This step is performed after the 80:20 split to avoid data contamination by leaking information about the testing dataset into the training dataset.

Table 2.1: Smartwatch dataset statistics after filtering (min: minimum number of samples per Place ID; max: maximum number of samples per Place ID; mean: mean number of samples per Place ID; std: standard deviation of samples per Place ID)

dataset	samples	min	max	mean	std
1st campaign	11,354	17	404	35.82	29.77
2nd campaign	10,667	20	190	33.65	13.92
combined	22,021	37	434	69.46	32.98

2.4 Proposed CNN Architecture

2.4.1 CNNs

For information about CNNs please refer to A.5.1.

2.4.2 Input and Output of the System

The goal is to build a system that takes (Equation 2.1) as input and produces (Equation 2.2) as output:

$$\boldsymbol{X} = \big[\boldsymbol{B}; \boldsymbol{R}\big] = \begin{bmatrix} B_x & B_y & B_z \\ R_x & R_y & R_z \end{bmatrix}; \boldsymbol{X} \in \mathbb{R}^{2\times 3} \tag{2.1}$$

$$\hat{\boldsymbol{y}} = \big[\Pr(PlaceID_1), \cdots, \Pr(PlaceID_{317})\big]; \hat{\boldsymbol{y}} \in \mathbb{R}^{317} \tag{2.2}$$

$PlaceID_c$ where $c \in \{1, 2, \cdots, 317\}$ with highest predicted probability is taken as the system's final prediction. Before we proceed any further, a feasibility study of using $B_x, B_y, B_z, R_x, R_y, R_z$ as fingerprints for localization is performed. Obviously, the use of geomagnetic field as fingerprints is already established; however, using the smartwatch's absolute orientation as fingerprints must be investigated. In other words, does a relationship exists between the smartwatch's absolute orientation and the grid points' location? We hypothesize that if a relationship exists, then R_x, R_y, R_z

Table 2.2: MIC between R_x, R_y, R_z and (x, y) of the grid points

(R_x, x)	(R_x, y)	(R_y, x)	(R_y, y)	(R_z, x)	(R_z, y)
0.84	0.55	0.37	0.34	0.47	0.44

can serve as auxiliary fingerprints providing high-level or abstract localization information. The basis of this hypothesis comes from observing human traffic patterns inside corridors and how they tend to follow a counterclockwise motion. To test this hypothesis, we applied the MIC which is a statistic that measures the relationship between two variables regardless of the relationship type (liner, non-linear, or even non-functional). MIC yields a continuous value in the range $[0, 1]$, where 0 indicates no relationship between the two variables, while 1 indicates a functional relationship. The MIC between R_x, R_y, R_z and the (x, y) coordinates of the grid points are shown in Table 2.2. As expected, a relationship exists. The strongest relationship is with respect to R_x. Considering the smartwatch's reference frame and the layout of the environment in Figure 2.1, this can be attributed to the fact that some values of R_x are more related to some grid points than others, especially in corridors. For examples, the grid points on the first half of path 6 have almost the same R_x value and differ from the R_x value of the grid points on the second half of the path by approximately $180\,^{\circ}$. Note that this analysis of R_x does not necessarily hold true for R_y and R_z as reflected by their MIC values. The greater MIC value obtained with respect to R_z as compared to R_y is a topic of future research. Nonetheless, determining the relative importance of each fingerprint in localization is ultimately left to the neural network.

2.4.3 Performance Evaluation Metrics

Two independent metrics are used for performance evaluation. These metrics are descried as follows:

Classification Accuracy

As stated earlier, we approach the indoor geomagnetic positioning problem from a multiclass classification perspective. Hence, one of the performance evaluation metrics

applied is the classification accuracy; which is defined as the ratio of correctly classified samples ($SAMPLE_{correct}$) to the total number of samples in a given validation or testing set ($SAMPLE_{total}$). It is often expressed as a percentage by multiplying this ratio by 100:

$$\text{Accuracy} = \frac{SAMPLE_{correct}}{SAMPLE_{total}} \times 100 \tag{2.3}$$

Euclidean Distance Error

Since all grid points are defined over a 2D Euclidean space, the second performance metric used is the Euclidean distance error; which is defined as the straight-line distance error (in m) between the $(\hat{x}, \hat{y})$ coordinates of the predicted PlaceID and the (x, y) coordinates of the ground truth PlaceID:

$$d_{error} = \sqrt{(x - \hat{x})^2 + (y - \hat{y})^2} \tag{2.4}$$

In deep learning, model selection is a fundamental process that involves choosing the best model from a set of competing models. More precisely, given a set of models, training data, and testing data, the model that is expected to outperform all other models on the testing data is selected. This is critical since the most reliable estimate of a model's generalization performance i.e., its performance on future data, is its performance on the testing data.

Two main problems affecting a model's generalization performance are overfitting and underfitting. Overfitting occurs when the model has learned to model the noise in the data instead of learning the underlying structure of the data. By contrast, underfitting occurs when the model has not "adequately" learned the underlying structure of the data. During the training/validation process, an overfit model is characterized

as having low training loss and high validation loss, while an underfit model is characterized as having high training loss and high validation loss. Overfitting is generally caused by an overly complex model, while underfitting is caused by an overly simple model. In deep learning, a model's complexity is reflected by its number of learnable parameters (more parameters mean increased complexity and vice versa).

Performing model selection is crucial in avoiding overfitting and underfitting. The most common approach to model selection is K-fold Cross Validation (CV). In K-fold CV, the training dataset is segmented into K disjoint partitions (folds) of equal size. During each iteration, one fold is used for validation, while the remaining $K-1$ folds are used for training. This process is repeated K times. The overall validation loss is calculated by averaging the validation losses from all K iterations. Finally, the model with the smallest averaged validation loss is selected.

Our approach is to start with a very simple model consisting of only a Softmax layer then gradually increase the complexity of subsequent models by adding a convolutional and/or an FC layer to the previous model. The training dataset (17,616 samples) is used to train and validate each model using 5-fold CV, i.e., in each iteration, 14,093 samples are used for training and 3,523 samples are used for validation. For each model, the averaged validation loss and the classification accuracy on the validation set (validation accuracy) is recorded. Finally, the model with smallest validation loss and highest validation accuracy is considered the best model.

A total of ten models were build using this approach. The architectural specifications along with the number of learnable parameters, averaged epoch latency (time per training iteration), training loss, validation loss, and validation accuracy of each model are shown in Table 2.3. All models were trained using Adam optimizer with a learning rate of 10^{-4}. Using the early stopping method, each model is trained until both its training and validation losses converge. TensorFlow, an open source deep

learning framework [30], is used for all software implementations. Note that determining the number of neurons in each FC layer and the number of filters in each convolutional layer is heuristic. We chose a fixed number of 317 neurons for FC layers and 16, 32, 32, and 64 filters for the first, second, third, and fourth convolutional layers, respectively. For all convolutional layers, a filter size of $(1, 1)$, a stride length of $(1, 1)$, and a zero-padding size of $(0, 0)$ are used.

From Table 2.3, it is noticed that training loss decreases as model complexity increases. This behavior occurs when the models start to memorize the training set instead of learning it. The degree of memorization is reflected by comparing a model's training loss to its validation loss. This is visually depicted in Figure 2.3 Note how models 1 to 5 have relatively high training and validation losses, while models 7 to 10 have relatively low training losses but high validation losses. From this observation, we conclude that models 1 to 5 are underfit, while models 7 to 10 are overfit. The model with the lowest validation loss and the highest validation accuracy is model 6. Therefore, model 6 which consists of two convolutional layers, two FC layers, and a Softmax layer is selected as the final model. Henceforth, we refer to model 6 as the proposed system. The architecture of the proposed system is illustrated in Figure 2.4.

Table 2.3: Architectural specifications of the ten models and their performance statistics (conv: convolutional layer + Rectified Linear Unit (ReLU); fc: FC layer + ReLU; SM: Softmax layer). Epoch latency is in s – based on an i5-5250U Central Processing Unit (CPU) @ 1.6 GHz

model	conv 1	conv 2	conv 3	conv 4	fc 1	fc 2	fc 3	fc 4	fc 5	num. of params.	epoch latency	train loss	validation loss	validation accuracy
1	-	-	-	-	SM	-	-	-	-	2,219	0.311	2.3980	2.5240	37.25
2	16	-	-	-	SM	-	-	-	-	30,781	0.458	0.5211	0.9435	79.08
3	16	32	-	-	SM	-	-	-	-	61,757	0.620	0.1755	0.6074	90.40
4	16	32	-	-	317	SM	-	-	-	162,563	0.973	0.0567	0.4480	94.51
5	16	32	32	-	317	SM	-	-	-	163,619	1.172	0.0400	0.4386	95.29
6	16	32	-	-	317	317	SM	-	-	263,369	1.420	0.0211	0.4186	96.08
7	16	32	32	-	317	317	SM	-	-	264,425	1.550	0.0220	0.4783	95.62
8	16	32	-	-	317	317	317	SM	-	364,175	1.709	0.0254	0.4976	95.39
9	16	32	32	-	317	317	317	SM	-	365,231	1.828	0.0216	0.5158	95.13
10	16	32	32	64	317	317	317	317	SM	529,013	2.503	0.0209	0.6429	94.69

2.5 Experiments and Results

The proposed system is retrained on the entire training dataset. This will expose the system to 3,523 more training samples than in the original CV process. As a result, the system's performance on the testing data is expected to improve. For retraining, we used the same settings as before i.e., using Adam optimizer with a learning rate of 10^{-4} and early stopping when the training loss has converged.

2.5.1 Performance on the Testing Dataset

Now that the system is retrained, it is ready for evaluation. To ensure an unbiased performance assessment of the system, the evaluation process is conducted using unseen samples from the testing dataset (a total of 4,405 samples). The metrics described in Equation 2.3 and Equation 2.4 are used for this purpose. The performance of the system is also compared against two existing machine learning classifiers; namely kNN and one-vs-all Support Vector Machine (SVM). For the sake of fair comparison, the hyperparameters of both classifiers were tuned using 5-fold CV. The evaluation results are summarized in Table 2.4. The results clearly show

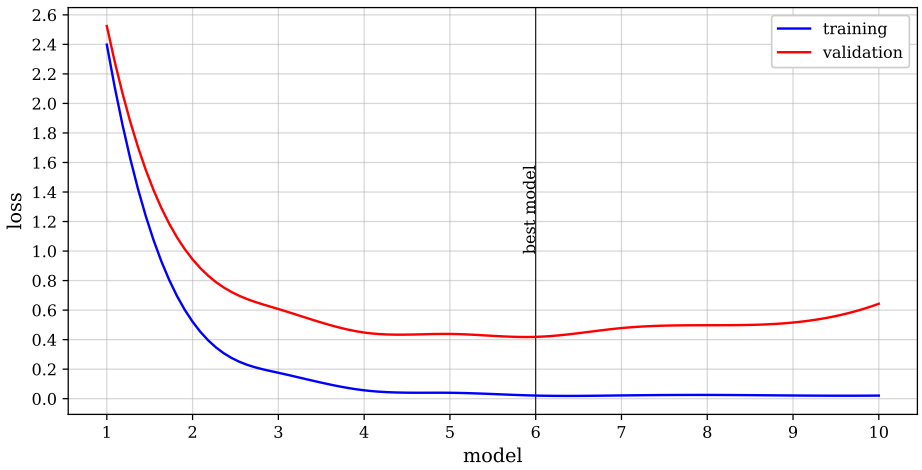

Figure 2.3: Training vs. validation loss for all models.

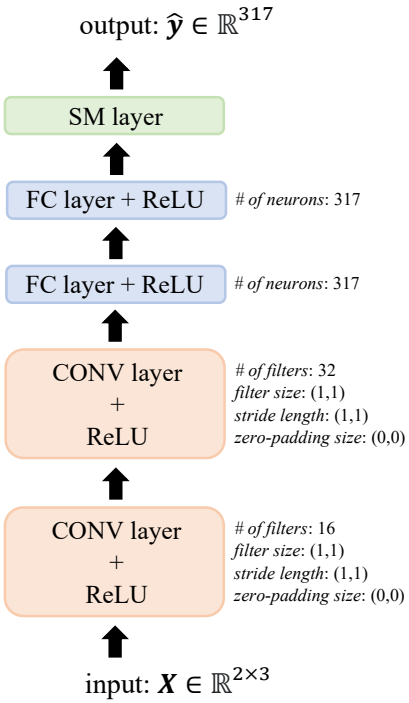

Figure 2.4: Proposed system architecture.

that the proposed system outperforms both methods in terms of accuracy and mean localization error. By observing the system's performance on the testing dataset, we expect the system to generalize well on future data. The empirical cumulative distribution function of d_{error} is plotted in Figure 2.5 for all methods. As shown, the system achieves a d_{error} of 0.0 m 97.80 % of the time, compared to a d_{error} of 0.6 m 97.45 % of the time and 96.91 % of the time for kNN and SVM, respectively.

Regression vs. Classification

From Table 2.4 it is clear that the proposed method archives high positioning accuracy. However, when a misclassification occurs, the max. positioning error can be high (40.76 m). To address this issue, we have casted the positioning problem as a multi-output regression problem. We have used the same architecture and hyper-

Table 2.4: Performance comparison on the testing dataset (min: minimum d_{error} in m; max: maximum d_{error} in m; mean: mean d_{error} in m; std: standard deviation of d_{error} in m)

method	min	max	mean	std	accuracy
proposed	0.0	40.76	0.136	1.70	97.77
kNN (with $k = 5$ and L1 distance)	0.0	40.01	0.231	2.14	93.51
SVM (with $C = 1$, $\gamma = 103$, and RBF kernel)	0.0	39.46	0.443	2.92	93.16

parameters as the proposed method but replaced the Softmax layer with an FC layer with two neurons. One neuron regresses the x coordinate and the other regresses the y coordinate. We experimented with two regression loss functions, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). Performance on the testing set is presented in Table 2.5. Two observations can be made. First, using RMSE loss yielded better result than using MAE loss. This is attributed to the fact that RMSE assigns more weight to large errors than it does for small errors. Second, comparing the results achieved by the proposed method with that achieved by the regression version with RMSE loss, we can see that the latter achieves comparable min. and

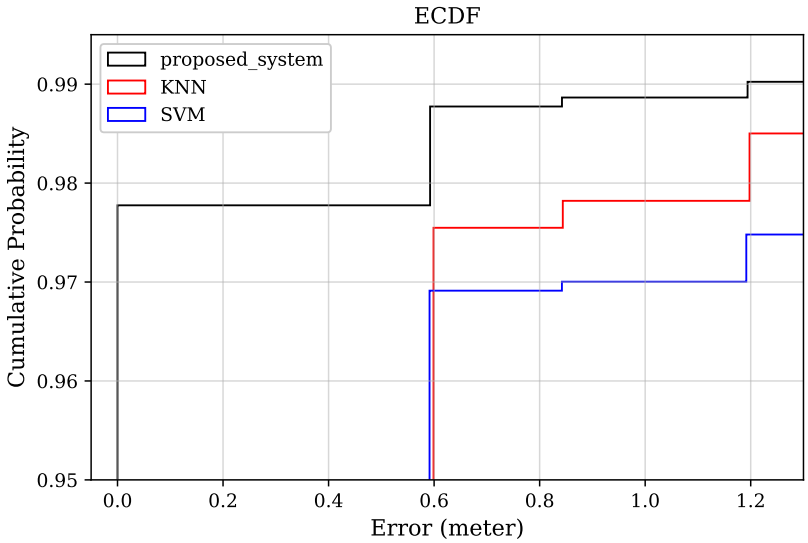

Figure 2.5: Empirical cumulative positioning error.

Table 2.5: Performance on the testing set using MAE loss and RMSE (min: minimum d_{error} in m; max: maximum d_{error} in m; mean: mean d_{error} in m; std: standard deviation of d_{error} in m)

loss function	min	max	mean	std
MAE	0.013	38.11	1.44	2.85
RMSE	0.014	33.81	1.32	2.03

std. errors to the former. Moreover, the max. error was reduced by 17.05 %. However, this comes at the expense of a higher mean error. This is expected because having a high classification accuracy reduces the mean error but does not guarantee a bounded max. error.

2.5.2 Prediction Latency

Prediction latency is defined as the time it takes a classifier to make a prediction on a single sample during the online phase. Thus, prediction latency is measured on the testing dataset. Note that this is different from epoch latency. On average, it takes the proposed system 0.0024 s to perform a single prediction, compared to 0.0617 s for kNN and 0.3529 s for SVM. Obviously, the system's prediction latency is significantly lower than both classifiers. This is attributed to the fact that, unlike kNN and SVM, the proposed system is parametric; meaning that once the parameters of the system have been learned, the training data can be discarded and the prediction of a new sample is accomplished through a sequence of matrices multiplication. This makes the system well-suited for real-time user tracking applications where high-speed performance is key.

2.5.3 Softmax Layer and User Tracking

We have seen how the Softmax layer turns the system into a probabilistic classifier by outputting a probability distribution over the set of grid points in the environment. This output is in a form of a vector $\hat{\boldsymbol{y}} \in \mathbb{R}^{317}$ where the grid point or $PlaceID_c$ with the highest probability is the system's first choice. In this regard, we analyzed the $\hat{\boldsymbol{y}}$ vector of all misclassified samples to determine if the system's second choice (i.e., $PlaceID_{i \neq c}$ with the second highest probability) would have been the correct prediction. We found that in 74 % of the misclassified samples, the system's second choice is indeed the correct prediction. In other words, should the system's first choice prove incorrect, there is a 0.74 probability that the second choice is correct. This finding is useful for user tracking applications that are based on sequence modeling algorithms. For instance, by implementing a simple outlier rejection mechanism that rejects the system's first choice if it is inconsistent with the current place and takes the second choice as output, the system's expected classification accuracy would increase by 1.66 % to become 99.40 % without even exploring the system third choice, fourth choice and so on. Note that this analysis does not hold for kNN and SVM since both classifiers are non-probabilistic.

CHAPTER 3
DEEP LEARNING-BASED SYMBOLIC INDOOR POSITIONING USING THE SERVING ENODEB

3.1 Introduction

The main drawback of fingerprinting is the laborious and time-consuming site surveying task in which fingerprints are collected at predefined RPs with known coordinates. Depending on the area to be covered by the system and the accuracy requirement, the number of required RPs can be significant. Symbolic positioning tries to relax this requirement by collecting fingerprints in zones rather than at points [31]. However, the concept of distance is lost since zones are independent and the user's location is now expressed symbolically (e.g., "in the kitchen") instead of physically (using a coordinate system).

In the proposed method, we treat the indoor positioning problem as a classification problem. Each symbolic space in the environment has different cellular signal propagation characteristics and, hence, should be considered as a unique class. To distinguish one class from another (i.e., one symbolic space from another), we leverage DAEs. The motivation behind employing DAEs, as opposed to other learning algorithms, is their ability to handle noisy data effectively and efficiently. Our experimental results, which are based on real signal measurements collected inside a residential apartment, verify that the proposed method outperforms conventional

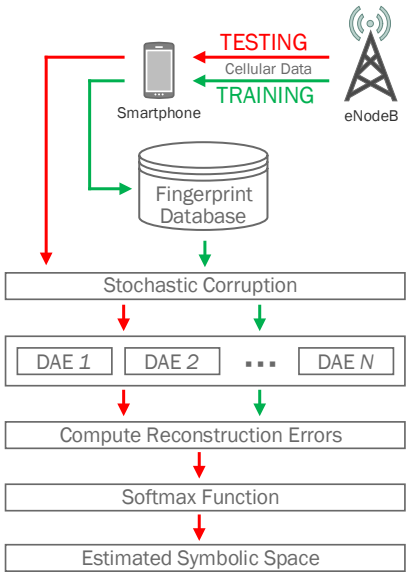

Figure 3.1: The general scheme of the proposed method representing the training and testing phases.

symbolic indoor positioning techniques on various performance metrics. The general scheme of the proposed method is depicted in Figure 3.1.

The remainder of this chapter is organized as follows. Section 3.2 reviews some of the recent work in deep learning-based indoor positioning. Section 3.3 describes and validates the dataset used in this study. Section 3.4 provides background on Autoencoders and discusses the design of the proposed method. Section 3.5 reports on the evaluation experiments and analyzes the results.

3.2 Related Work

In this section, a review of some recent research efforts that utilize machine learning for symbolic indoor positioning is provided, followed by a review of some recent research efforts that utilize machine learning for cellular-based indoor positioning.

3.2.1 Machine Learning for Symbolic Indoor Positioning

Werner *et al.*[32] utilized the CNN-based AlexNet [33] as a generic feature extractor to classify a query image to one of 16 rooms. No fine-tuning was performed on the pre-trained network; instead, the authors directly fed the features extracted by the first FC layer to a Naïve Bayes classifier. These features helped their model to generalize from local to global views (i.e., from small views in training to large views in testing) well. However, this did not hold when attempting to generalize from global to local views due to the spatial invariance of features introduced by the CNN. A room classification accuracy of 95 % was reported using global views for both training and testing.

Nowicki and Wietrzykowski [34] used an AE followed by an FC network for multi-building and multi-floor classification using WiFi fingerprints. The authors indicated that previous approaches based on hierarchical processing [35] have high complexity, requiring careful feature selection and a separate algorithm for each level of granularity (i.e., building then floor identification). The purpose of the AE is to perform dimensionality reduction. This is important because a WiFi fingerprint has entries for all APs detected in an entire environment, but only a subset of these APs is observed for different locations. This is especially true for large-scale environments. The FC network maps the compact representation into its corresponding class, where a class represents a flattened label of a building-floor combination (e.g., "`Building3-Floor5`"). The authors reported a 92 % classification accuracy on the UJIIndoorLoc dataset [36].

Most recently, Tamas and Toth [31] performed a performance analysis of five machine learning classifiers for symbolic indoor positioning. They used hybrid fingerprints (WiFi, Bluetooth, and magnetometer) to evaluate and compare the classifiers

being studied. The fingerprints were obtained from the Miskolc IIS dataset [37] which contains measurements from 21 zones of different sizes inside a three-story university building. Experimental results under controlled settings revealed classification accuracies of 96.77 % using an FC network, 92.26 % using kNN, 91.61 % using Naïve Bayes, 84.52 % using Decision Tree, and 80.65 % using Rule Induction.

Our proposed method has several advantages compared to the aforementioned works:

- It preserves privacy because it does not require the capturing of images for positioning.
- It is well-suited for small-scale indoor environments, namely residential apartments and homes, where people spend most of their time.
- It does not require on-premises infrastructures such as WiFi APs or Bluetooth beacons for operation. Instead, it relies on omnipresent cellular signals.
- It has little overhead because only a single fingerprint type is required for positioning which eliminates the complexity associated with fusing multiple fingerprint types.

3.2.2 Machine Learning for Cellular-based Indoor Positioning

Rizk *et al.* [38] used an FC network to perform cellular RSS fingerprinting. Data augmentation techniques were used to increase the training set by 8-fold. The testbed consisted of an 11 m by 12 m university building floor with 51 RPs spaced at an equal distance of 1 m. The authors achieved a positioning error of less than 3 m 90 % of the time. However, to achieve this accuracy, the RSS from 17 Second-Generation (2G) cellular Base Stations (BSs) had to be measured. Later, the authors used an RNN to

capture the temporal dependency between consecutive RSS measurements received from the serving BS [39]. The achieved positioning accuracy was comparable to that acquired by their previous approach, however, a measurement window of at least 3 s had to be fed to the RNN.

Arnold *et al.* [40] used a custom-built linear array of Multiple-Input Multiple-Output (MIMO) antennas installed in a 20 m by 7 m area for indoor positioning. They used an FC network to correlate the antennas' channel coefficients to a 3D position relative to the array's location. To avoid the burden of collecting a large dataset for training, a two-step training procedure was followed. First, the network was pre-trained on simulated Line-of-Sight (LoS) channel coefficients; then, it was fine-tuned with a small number of real LoS and NLoS measurements collected using a custom-built probe. Various sub-meter accuracies were reported based on the environment setting (LoS vs. NLoS), the number of samples for fine-tuning, and the samples' spatial locations.

Vieira *et al.* [41] used a CNN to learn the structure of massive MIMO channels for indoor positioning. A cellular channel model was used to generate unique channel fingerprints for each training/testing position. These fingerprints represent clusters of multipath components obtained from a BS equipped with a linear array of antennas. The fingerprints were transformed into an angular-delay domain to resemble sparse 2D images that were then used in training a CNN to regress the receiver's 2D coordinates. The authors reported distance in terms of wavelength (λ). RMSE, normalized by λ, was used as an accuracy metric, where the achieved RMSE was 0.6λ inside a $25\lambda \times 25\lambda$ confined area. Since all measurements were based on simulated data, real-world measuring impairments such as noise and channel fading were not considered.

Compared to the previous works, our proposed method combines several features that place it in a unique position:

- It employs DAEs to handle incomplete measurements caused by unpredictable cellular signal loss.
- It only utilizes the information measured from the serving BS, which is a Fourth-Generation (4G) cellular BS (also known as an eNodeB in Long-Term Evolution).
- It is well-suited for real-time positioning applications, given the parametric nature of DAEs, in addition to performing one-shot positioning (i.e., only a single fingerprint sample is required to estimate the user's location).
- It is based on real-world measurements emitting from a real eNodeB. No simulated, interpolated, or augmented data were used in this study.
- It is well-suited for smartphone-based positioning because all measurements were collected using smartphones as opposed to custom-built collection platforms.

3.3 Dataset Description and Validation

Nearly all indoor positioning solutions found in the literature were evaluated using private data. Thus, the results obtained are self-reported and cannot be reproduced. Additionally, the lack of publicly available datasets that can be used to develop, evaluate, and compare indoor positioning solutions constitutes a high entry barrier for studies. For these reasons, we made the dataset used in this study publicly available [42]. The following subsections describe the data collection platform, environment, procedure, and technical quality, respectively.

3.3.1 Data Collection Platform

We used two smartphones for data acquisition: Phone 1 and Phone 2. Both smartphones ran on Android 10. The motivation behind choosing Android-powered smartphones was twofold. First, Android provides Application Programming Interfaces (APIs) that allow for acquiring raw data at the hardware level. Second, Android-powered smartphones account for over 74 % of the market share worldwide [43]. We attached the two smartphones to a tripod using a dual mount (Figure 3.2). Both smartphones were in portrait mode and were kept at a fixed height of 130 cm. The tripod head was adjusted to tilt the smartphones at a $\sim 40^\circ$ angle to the vertical plane. We installed the same third-party app [44] used for the data collection on both smartphones. The app allowed for conveniently collecting and exporting cellular network data.

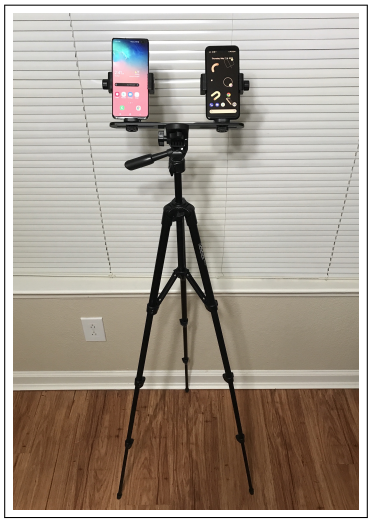

Figure 3.2: A picture of Phone 1 and Phone 2 attached to the tripod.

3.3.2 Data Collection Environment

We performed data collection inside a residential apartment of eight symbolic spaces. As seen from the apartment's layout (Figure 3.3), the symbolic spaces include a living room ($4.0\times3.0\,\mathrm{m}^2$), a sunroom ($2.6\times2.3\,\mathrm{m}^2$), a bedroom ($3.5\times3.2\,\mathrm{m}^2$), a hallway ($7.0\times0.8\,\mathrm{m}^2$), a dining room ($3.2\times2.0\,\mathrm{m}^2$), a kitchen ($2.8\times0.6\,\mathrm{m}^2$), a bathroom ($1.1\times1.1\,\mathrm{m}^2$), and a walk-in closet ($2.2\times1.6\,\mathrm{m}^2$). The floor plan delineating the apartment's dimensions is provided alongside the dataset in the form of a `.vsdx` file.

3.3.3 Data Collection Procedure

A smartphone's cellular modem constantly scans the cellular network for cell selection/reselection and handover purposes. Android provides APIs to extract data associated with scans such as Reference Signal Received Power (RSRP) and cell identity information [45]. For each of the symbolic spaces described above, we collected 25 minutes of cellular data (per phone) at a sampling rate of 1 Hz. During data collection, we systematically changed the position and orientation of the tripod to

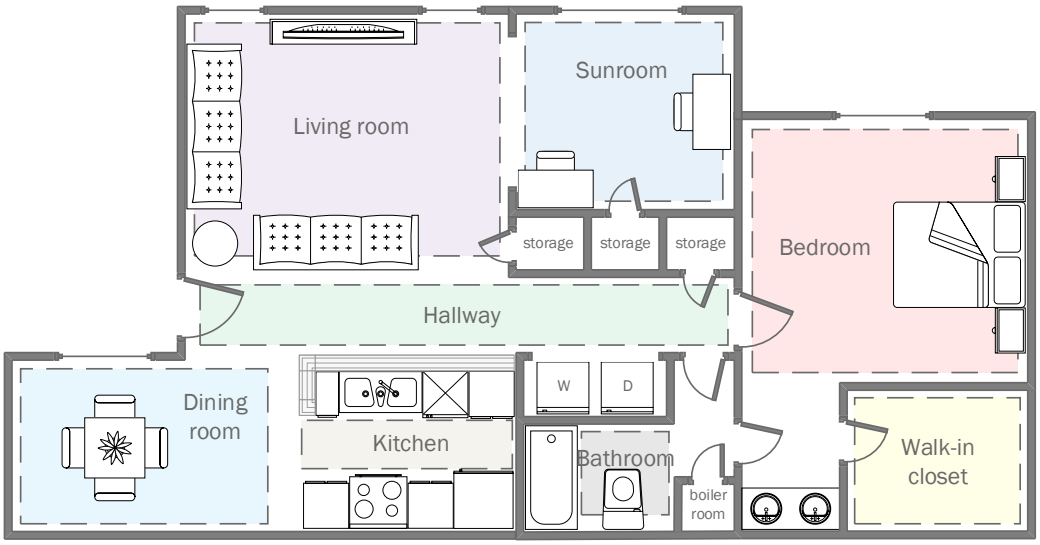

Figure 3.3: Layout of the apartment where data was collected.

uniformly cover space and direction. Sampling results were exported as a `.csv` file and named with the smartphone's and space's name (e.g., `Phone2_Bedroom.csv`). Table 3.1 lists all fields in each data sample and their descriptions. As an example, Figure 3.4 plots the data collected from the smartphones located inside the walk-in closet.

3.3.4 Technical Validation

The technical quality of the dataset was evaluated using experiments that consider its reliability and validity:

Measurement Reliability

Before the collection campaign, we captured cellular data over three different days at the same location. We used Spearman's and Kendall's correlation coefficients

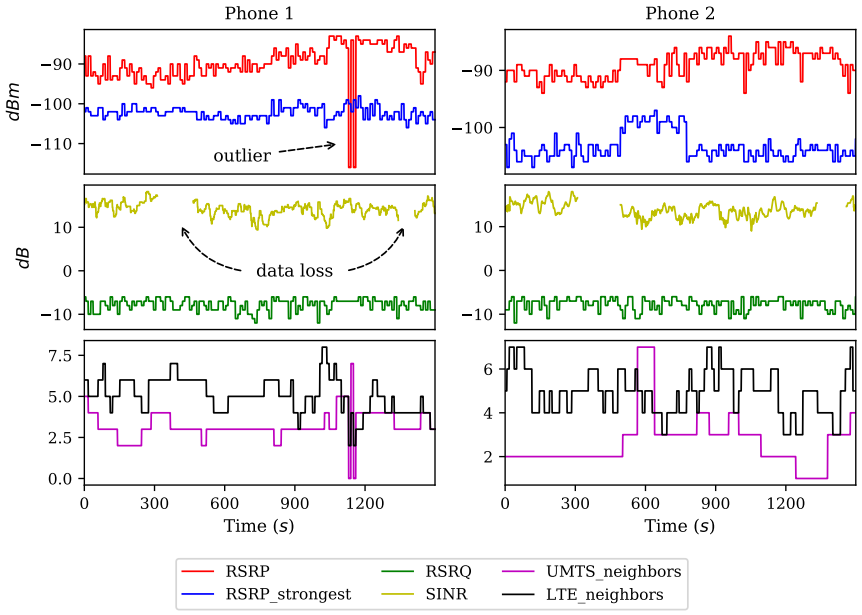

Figure 3.4: Plots of cellular data showing examples of outliers and data loss in the data collected by Phone 1 and Phone 2 inside the walk-in closet.

Table 3.1: Field labels of data samples and their description

#	Field label	Description
1	`Date_Time`	The date and time the sample was captured as `YYYYMMDDhhmmss`
2	`PLMN_ID`	The Public Land Mobile Network (PLMN) IDentifier
3	`eNodeB_ID`	The E-UTRAN NodeB (eNB) IDentifier that is used to uniquely identify an eNB within a PLMN
4	`Cell_ID`	The Cell IDentifier which is an internal descriptor for a cell. It can take any value between `0` and `255`
5	`ECI`	The E-UTRAN Cell Identifier (ECI) that is used to uniquely identify a cell within a PLMN. `ECI` $= 256 \times$ `eNodeB_ID` + `Cell_ID`.
6	`RSRP`	The RSRP in decibel-milliwatt (dBm)
7	`RSRQ`	The Reference Signal Received Quality (RSRQ) in decibel (dB)
8	`SINR`	The Signal to Interference and Noise Ratio (SINR) in dB
9	`UMTS_neighbors`	The number of neighboring Universal Mobile Telecommunications Service (UMTS) cells
10	`LTE_neighbors`	The number of neighboring Long-Term Evolution (LTE) cells
11	`RSRP_strongest`	The RSRP, in dBm, corresponding to the strongest neighboring LTE cell

to quantify the degree of consistency between temporal measurements for a given phone. Table 3.2 shows Spearman's and Kendall's correlation coefficients for the two smartphones for all possible pairs of days. Given that correlation results are high (i.e., close to the maximum value of 1.0), it can be concluded that the dataset possesses a high degree of reliability.

Measurement Validity

We assessed measurement validity by comparing the cellular data captured by the two phones and checking for consistency. Accordingly, for a given day, we used Spearman's and Kendall's correlation coefficients to quantify the degree of consistency

Table 3.2: Results of the correlation analysis between the measurements obtained on three different days for Phone 1 and Phone 2. The results were generated using synchronized readings of fields 6–11. The p-values of all results were less than 0.01.

	Phone 1			**Phone 2**		
	$\{Day1, Day2\}$	$\{Day2, Day3\}$	$\{Day1, Day3\}$	$\{Day1, Day2\}$	$\{Day2, Day3\}$	$\{Day1, Day3\}$
Spearman's ρ	0.992	0.988	0.981	0.990	0.976	0.980
Kendall's τ	0.983	0.973	0.956	0.977	0.945	0.953

between the measurements obtained by the phones. The correlation results for the foregoing three days are shown in Table 3.3. These results demonstrate high levels of consistency, which attests to the validity of the dataset.

3.4 Background and Proposed Method

3.4.1 AEs

For information about AEs please refer to A.5.4.

3.4.2 Proposed Method

The design of the proposed method is inspired by the successful application of AEs for anomaly detection [46]: An AE, when solely trained on normal data instances, fails to reconstruct abnormal data instances, hence, producing a large reconstruction error. The data instances that produce high residual errors are considered outliers.

The proposed method takes a normalized cellular data sample captured from the serving eNodeB as input (Equation 3.1) and produces an output (Equation 3.2) that is a probability distribution over the set of symbolic spaces in the environment:

$$X = \big[\texttt{RSRP}, \texttt{RSRQ}, \texttt{SINR}, \texttt{UMTS\_neighbors}, \texttt{LTE\_neighbors}, \texttt{RSRP\_strongest}\big]; \quad X \in \mathbb{R}^6 : \{x_i \in \mathbb{R} \mid 0 \leq x_i \leq 1\} \tag{3.1}$$

Table 3.3: Results of the correlation analysis between the measurements obtained from Phone 1 & Phone 2 for three different days. The results were generated using readings of fields 6–11. The p-values of all results were less than 0.02.

	Phone 1 & Phone 2		
	Day1	*Day2*	*Day3*
Spearman's ρ	0.991	0.995	0.968
Kendall's τ	0.979	0.989	0.927

$$
\begin{aligned}
Y &= \big[\Pr(space_1 \mid X), \cdots, \Pr(space_n \mid X)\big]; \\
Y &\in \mathbb{R}^n : \{y_i \in \mathbb{R} \mid y_1 + \cdots + y_n = 1\}
\end{aligned}
\tag{3.2}
$$

Given DAEs' data-driven learning ability, the proposed method does not make any assumptions about feature independence or the nature of the boundary separating the classes.

The input vector is corrupted to emulate a randomized loss of cellular data. This is accomplished using a Hadamard product of (Equation 3.1) and an all-ones vector ($\vec{1}$) whose elements are randomly set to 0 with a predefined probability p_{loss}. For example, if p_{loss} is set to 0.5, there is a 50 % chance that a given field entry will be set to zero. Our approach of representing cellular data loss (i.e., missing values) with zeros is inspired by the image inpainting problem in which a mask of zeros is often used to corrupt clean images before feeding them to an image inpainting algorithm that predicts the masked/missing values [47, 48].

During the training phase, a dedicated DAE is employed for each symbolic space. Each DAE is solely trained on the data collected at its corresponding symbolic space. By following this training strategy, we expect that, during the testing phase, all DAEs, except for one, will generate a relatively high reconstruction error when fed the same testing sample. Consequently, the symbolic space associated with the DAE that generated the lowest reconstruction error is considered as the estimated symbolic

Table 3.4: Hyperparameter tuning

Hyperparameter	**Value**
Batch size	100
Dropout rate	0.1
Optimizer	Adadelta ($\rho = 0.95$, $\epsilon = 1e{-}7$)
Learning rate	1.0
Activation function	ReLU
Epochs	1,200
Loss function	Binary cross-entropy
Weight initializer	Xavier uniform
Bias initializer	Zeros

space. To construct (Equation 3.2), we used a Softmax function (Equation 3.3) during the testing phase:

$$\Pr(space_i \mid X) = S(\mathcal{L}_i) = \frac{\exp(1/\mathcal{L}_i)}{\sum_{i=1}^{n} \exp(1/\mathcal{L}_i)} \tag{3.3}$$

where $\mathcal{L}_i$ is the reconstruction loss generated by the i^{th} DAE.

DAE Architecture

All DAEs have the same architecture which consists of an input layer of 256 neurons, a hidden layer (and its mirror layer) of 64 neurons, a bottleneck layer of 16 neurons, and an output layer of 256 neurons. We developed the DAEs using Keras [49] with the hyperparameters listed in Table 3.4. We selected these hyperparameters using grid search and cross-validation. We used early stopping and dropout to avoid overfitting.

Dataset Preprocessing

From each symbolic space, there were 1,500 samples collected by each smartphone. For the entire collection period, and throughout the collection environment, Phone 1 and Phone 2 camped on the same LTE cell (i.e., `ECI:98059528`). Thus, entries

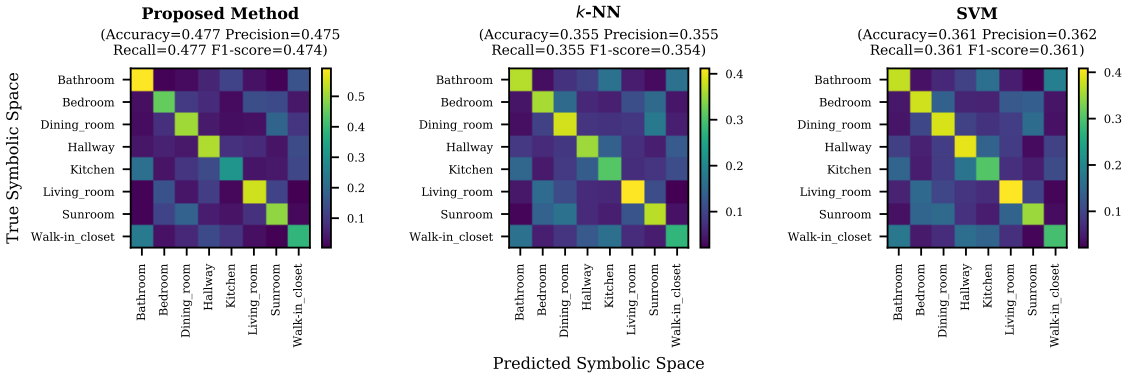

Figure 3.5: Accuracy, Precision, Recall, F1-score, and the confusion matrix of symbolic space prediction of the proposed method, kNN, and SVM.

for field labels 2–5 were identical for all samples. For a given symbolic space, we combined the samples collected by Phone 1 and Phone 2 to create a single dataset for training and testing the corresponding DAE. After the samples in the combined dataset were randomly shuffled, we allocated 80 % of them for training and validation, and the remaining 20 % for testing. Since input features are measured in different units, their values were normalized between 0 and 1. This was performed after the dataset was split to avoid data contamination. Figure 3.1 shows the general scheme of the proposed method.

3.5 Experiments and Results

This section evaluates the performance of the proposed method and investigates the impact of p_{loss} and device heterogeneity on positioning accuracy. Associated computing scripts are publicly available in our figshare repository [42].

3.5.1 Performance Evaluation

We trained the proposed method with a p_{loss} value of 0.5 applied to the training set. The metrics used for performance evaluation are Accuracy, Precision, Recall, and

F1-score as defined in [50]. We compared the performance of the proposed method against classifiers that are extensively used for indoor positioning, namely kNN and SVM. For the sake of fair comparison, we trained these classifiers on the same training set used for the proposed method and fine-tuned their parameters using grid search and cross-validation. The testing set used for comparison was contaminated with a p_{loss} value of 0.5. Figure 3.5 reports on the classification results and shows the confusion matrices of the three methods. The results clearly show that the proposed method outperforms both kNN and SVM on all metrics. As mentioned earlier, both smartphones connected to the same LTE cell throughout the environment. However, it is possible, depending on network parameters, that a connection alternates between multiple cells. Incorporating the information obtained by additional cells is expected to further enhance performance because location discernibility will increase with increased features.

Interesting observations can be made by examining the confusion matrices in Figure 3.5. For example, higher degrees of confusion tend to occur between symbolic spaces that are close to each other (e.g., Kitchen and Bathroom or Sunroom and Living room). Nevertheless, observations exist that prove contrary to this assumption. For instance, there is low confusion between Bedroom and Walk-in closet despite their proximity. In fact, it is more likely to confuse Bedroom for Dining room than it is to confuse Bedroom for Walk-in closet. Such observations could be the result of the complex changes that cellular signals undergo when propagating indoors. Confirming this conjecture is a topic of future research.

3.5.2 Effect of p_{loss} on Accuracy

To study the impact of p_{loss} on positioning accuracy, we evaluated the proposed method, kNN, and SVM using data contaminated with varying p_{loss} values. More

specifically, we generated 20 copies of the testing set and contaminated each copy with a different p_{loss} value that ranged from 0.0 to 0.95, using 0.05 increments. The methods' accuracy scores that corresponded to each p_{loss} value were recorded and plotted in Figure 3.6. One observation that can be made from Figure 3.6 is that, as loss increases, so does the performance gap between the proposed method and kNN/SVM. This primarily suggests that DAEs learned more robust features than kNN and SVM.

3.5.3 Effect of Device Heterogeneity on Accuracy

Smartphones obtain cellular data readings from their cellular chipsets. Since these chipsets are manufactured by different vendors, hardware and firmware specifications are not uniform across smartphones. This results in heterogeneous reception characteristics which, in turn, can degrade the accuracy of the positioning system [51].

In this experiment, we investigated device heterogeneity by training the proposed method, kNN, and SVM on the data obtained from one smartphone and testing on the data obtained from 1) the same smartphone and 2) the other smartphone to quantify the difference in performance. Table 3.5 reports on the experiment's results. As clearly seen from Table 3.5, device heterogeneity is a significant problem in all three methods. Average accuracy drops of 45.7 %, 40.8 %, and 42.9 % are observed in the proposed method, kNN, and SVM, respectively. In the field of indoor positioning, there is ongoing research regarding overcoming device heterogeneity and we intend to address this limitation in subsequent research.

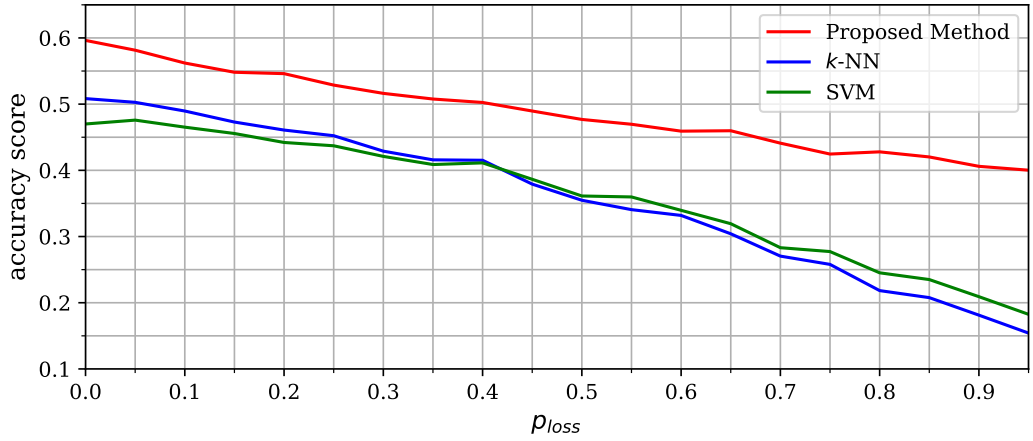

Figure 3.6: The effect of p_{loss} on accuracy for the proposed method, kNN, and SVM.

Table 3.5: Results of the device heterogeneity analysis. p_{loss} is set to 0.5 for both training and testing.

	Training Data		**Testing Data (accuracy)**	
	Phone 1	Phone 2	Phone 1	Phone 2
Proposed Method	✓		0.565	0.300
		✓	0.299	0.539
kNN	✓		0.395	0.235
		✓	0.230	0.391
SVM	✓		0.396	0.228
		✓	0.222	0.393

CHAPTER 4
OVERSAMPLING HIGHLY IMBALANCED INDOOR POSITIONING DATA USING DEEP GENERATIVE MODELS

4.1 Introduction

Fingerprinting typically utilizes supervised learning and is inherently dependent on labeled datasets. However, often real-world indoor positioning datasets are imbalanced, meaning that the class distribution of fingerprint samples is not uniform. For example, Table 4.1 illustrates discrepancies between the number of samples in the minority and majority classes of some publicly available indoor positioning datasets. Training on imbalanced data may result in a model biased toward the majority class(es). The techniques used to address this problem can be grouped into four main approaches: data sampling [52], algorithmic modification [53], cost-sensitive learning [54], and ensemble learning [55].

Table 4.1: Examples of imbalanced indoor positioning datasets

Dataset	**Type**	**Minority**	**Majority**	**Ratio**
Dataset described in [56]	WiFi	1	2	1:2
Dataset described in [57]	BLE	36	78	$\approx$ 1:2
Miskolc IIS [37]	Hybrid	18	208	$\approx$ 1:12
Dataset described in [58]	BLE	2	34	1:17
Dataset described in [29]	Magnetic	17	404	$\approx$ 1:24
UJIIndoorLoc [36]	WiFi	2	139	$\approx$ 1:70
Dataset described in [59]	LoRaWAN	1	398	1:398

This chapter deals with data sampling and, in particular, with oversampling data techniques. To the best of our knowledge, no study exists that investigates the problem of imbalanced data in the context of indoor positioning. The main contribution of this chapter is the application of a VAE [60] and a conditional variant, referred to as a CVAE [61], on a highly imbalanced indoor fingerprinting dataset. By using various performance evaluation metrics, the achieved results are compared to those obtained by two state-of-the-art oversampling methods known as SMOTE [52] and ADASYN sampling [62].

The remainder of this chapter is organized as follows: Section 4.2 describes the dataset used in this study, Section 4.3 outlines the experimental setup, and Section 4.4 discusses the results.

4.2 Dataset Description

Aranda *et al.* [63] introduced the dataset used in this study and made it publicly available. We chose this dataset because it is composed of BLE fingerprints. BLE is a recently introduced low-power communication protocol. It was designed with the IoT in mind, so it has received widespread adoption in indoor positioning applications [64]. The data we used was collected from a three-story Physics Department building. Each floor was comprised of two same-sized cubic structures joined by a hallway. Ten multi-slot BLE beacons were deployed per floor, and three different smartphones were used to collect fingerprints at various RPs.

This chapter is concerned with users' locations expressed symbolically instead of physically, also known as symbolic positioning [31]. Therefore, we treated each cubic structure on each side of a floor as an independent symbolic space. Since each symbolic space has different BLE signal propagation characteristics, it can be considered a

unique class, and the symbolic positioning problem can be cast as a classification problem. We preprocessed the dataset to exclude any samples collected outside of the cubic structures and create an initially balanced dataset. Additionally, to account for differences in beacon transmission powers resulting from multi-slot configuration, we transformed all fingerprints into recurrence plots according to (Equation 4.1):

$$\begin{aligned} \vec{x} = [x_1, x_2, \cdots, x_n]; R_{i,j} = |x_i - x_j|; \\ \vec{x} \in \mathbb{R}^n : \{x_i, x_j \in \mathbb{R} \mid 0 \leq x_i, x_j \leq 1\} \end{aligned} \tag{4.1}$$

where $\vec{x}$ is a fingerprint vector of dimension n; x_i, x_j are standardized RSS measurements corresponding to beacons i and j, respectively; and $R_{i,j}$ represents the distance between two RSS measurements. After preprocessing, the balanced dataset contained a total of 8,500 samples per symbolic space. We allocated 80 % of those for training and the remaining 20 % for testing. Figure 4.1 presents a 2D scheme depicting the collection environment, RPs, and beacon locations, while Figure 4.2 displays the recurrence plot of a randomly selected fingerprint from each symbolic space.

4.3 Experimental Setup

Our approach to employing VAEs and CVAEs for oversampling imbalanced indoor positioning datasets is inspired by applying deep generative models for data oversampling in other domains such as fraud detection [65] and image processing [66]. We assessed the performance of VAEs and CVAEs by creating imbalanced versions of the training set. We applied these models to generate synthetic fingerprints of the minority symbolic space(s) so that all symbolic spaces are equally represented (i.e., an artificially balanced training set is created). Since we are interested in highly imbalanced data [67], we set the imbalance ratio to 1:100 using random downsampling.

We used the artificially balanced training set to train a downstream classifier that acted as a positioning model that distinguished between different symbolic spaces. For this purpose, we chose a SVM since SVMs are extensively used in indoor positioning [68]. We used the `scikit-learn` implementation of SVM [69], with default parameters that were kept fixed for all experiments. We used the testing set, which is well-balanced, to quantify the performance of the classifier according to metrics (Equation 4.2), (Equation 4.3), and (Equation 4.4):

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}$$

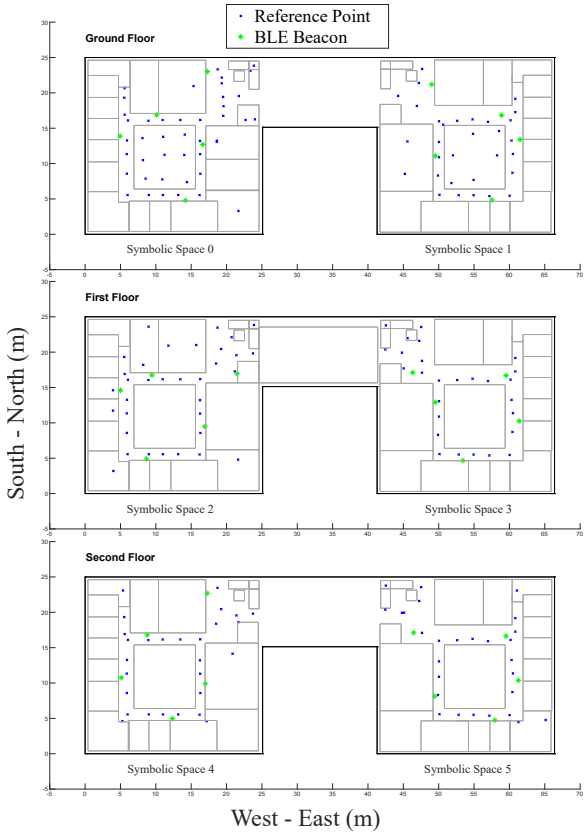

Figure 4.1: A graphical representation of the collection environment showing 2D floor plans, RPs, and beacon locations

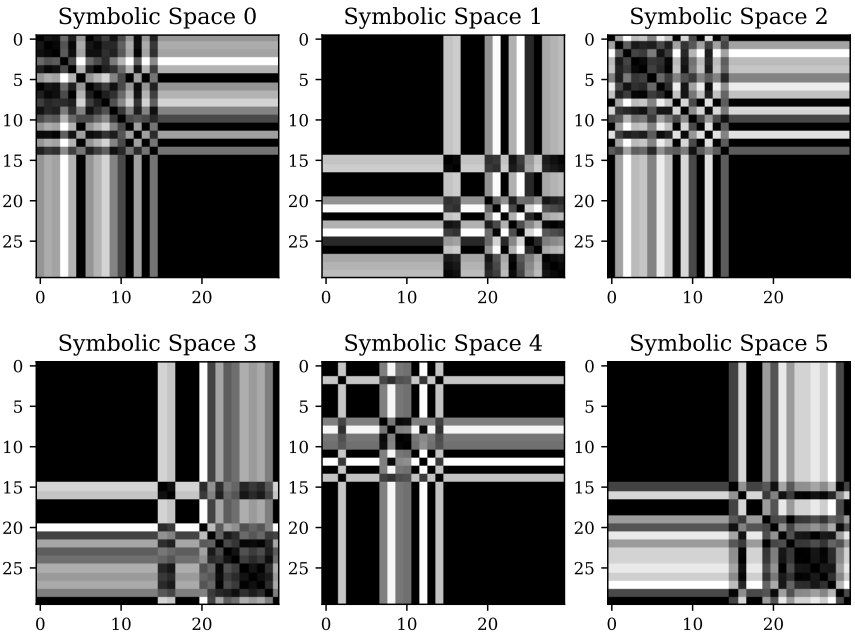

Figure 4.2: Examples of fingerprints transformed into recurrence plots

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.3}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

where TP are true positives, FP false positives, and FN false negatives. The aim is to determine whether VAEs and CVAEs can learn the distribution of the minority symbolic space(s) to generate synthetic fingerprints that promote enhancements in the classifier's performance. The performance of the classifier trained on the imbalanced version of the training set serves as the baseline. Performance results are expressed as a relative change compared to the baseline as calculated by (Equation 4.5):

$$\text{C}_\Phi = \frac{\Psi_\Phi - \Psi_{\text{IMBALANCED}}}{\Psi_{\text{IMBALANCED}}} \tag{4.5}$$

where C_Φ is the relative change for a performance metric Ψ obtained using an oversampling technique Φ.

Since there is a total of six symbolic spaces, we performed a total of five experiments. Each experiment corresponds to a different number of minority symbolic spaces ranging from 1 to 5. We conducted three trials for a given number of minority spaces (i.e., three imbalanced sets are constructed in which the spaces constituting a set are randomly chosen). For example, the experiment dealing with five minority spaces is composed of sets $\{0, 1, 2, 3, 5\}$, $\{0, 1, 3, 4, 5\}$, and $\{0, 1, 2, 3, 4\}$. The result is determined by averaging performance over all the trials. Table 4.2 presents the results of the experiments and compares them to those achieved by SMOTE and ADASYN as implemented in the `imbalanced-learn` library [70]. We used default parameters for SMOTE and ADASYN and we kept them fixed for all experiments. Similarly, VAE and CVAE architecture and hyperparameters implemented using `Keras` [49] were kept fixed for all the experiments. A general scheme of the experimental setup is presented in Figure 4.3.

4.4 Discussion

The results in Table 4.2 show that, in all experiments, using synthetic fingerprints generated by VAE, CVAE, SMOTE, and ADASYN all lead to an improved F1-score for the minority symbolic space(s) compared with classifiers trained on imbalanced datasets. Moreover, in all the experiments, every oversampling technique also resulted in a better F1-score for the majority symbolic space(s) and all spaces overall. This suggests that these oversampling techniques can enhance a classifier's overall learning ability, given that improvements are not isolated to the performance on the minority space(s). Finally, in general, SMOTE and ADASYN outperform VAE and CVAE.

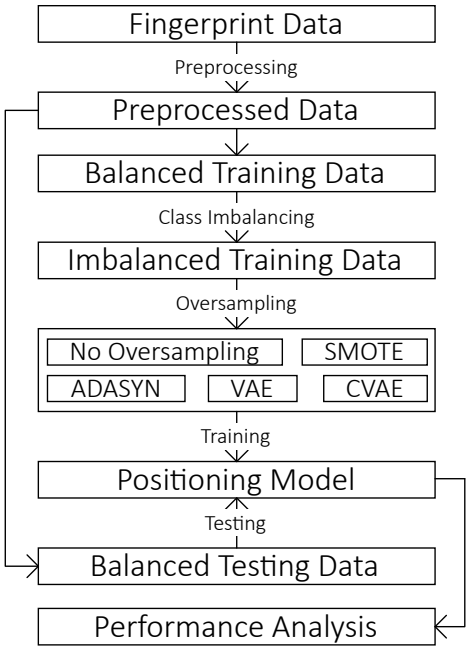

Figure 4.3: Scheme of the experimental setup

However, unlike VAE and CVAE, SMOTE and ADASYN are algorithms specifically designed to handle imbalanced data. Additionally, we expect that by fine-tuning VAE

Table 4.2: Downstream classifier results

		Minority			**Majority**			**Overall**		
Minority Classes	**Method**	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
1	SMOTE	-0.1597	11.0763	7.3103	0.048	-0.0153	**0.0255**	0.0049	0.0813	0.1511
	ADASYN	-0.1628	**11.3157**	**7.419**	**0.0486**	-0.0164	0.0254	0.0047	**0.0822**	**0.1529**
	VAE	**-0.0572**	9.6271	5.9637	0.0297	-0.0537	-0.0444	**0.0117**	0.0305	0.0592
	CVAE	-0.0775	2.6687	2.2359	0.0106	**0.0001**	0.0078	-0.0077	0.0234	0.0462
2	SMOTE	-0.1612	2.6073	1.7459	**0.0552**	-0.0731	0.0137	-0.0295	**0.1778**	**0.2649**
	ADASYN	-0.1619	**2.6083**	**1.7461**	0.0538	-0.0742	0.0129	-0.0306	0.1769	0.2643
	VAE	**-0.0363**	0.5386	0.5013	0.0128	**-0.0001**	0.0123	**-0.0052**	0.0504	0.0832
	CVAE	-0.0953	0.5552	0.4981	0.016	-0.0007	**0.0141**	-0.0268	0.0514	0.0843
3	SMOTE	-0.1863	3.9258	**2.4359**	**0.234**	-0.1229	0.101	-0.0323	**0.3697**	**0.6369**
	ADASYN	-0.1876	**3.9276**	2.4334	**0.234**	-0.0663	**0.1318**	-0.0332	0.3692	0.636
	VAE	**-0.0453**	1.533	1.2109	0.0703	-0.0039	0.0508	**-0.0029**	0.1637	0.305
	CVAE	-0.077	1.3086	1.0386	0.0672	**-0.0029**	0.0473	-0.0242	0.1401	0.2644
4	SMOTE	-0.0907	**2.1388**	**1.5017**	**0.5263**	-0.1097	**0.2738**	0.024	**0.5032**	**0.8718**
	ADASYN	-0.0932	2.1385	1.4979	0.5242	-0.1139	0.2703	0.0216	0.5001	0.8682
	VAE	0.0282	0.9697	0.8676	0.1912	-0.0064	0.1279	0.0584	0.2597	0.4881
	CVAE	**0.0618**	0.7553	0.6843	0.1363	**-0.0045**	0.0927	**0.0756**	0.2027	0.3808
5	SMOTE	0.012	**0.2202**	**0.2808**	**0.1315**	0.0601	**0.3246**	0.0283	**0.1845**	**0.2881**
	ADASYN	0.0084	0.2119	0.2724	0.1245	**0.0638**	0.3231	0.0242	0.1789	0.2809
	VAE	0.0705	0.1046	0.1461	0.0419	0.0477	0.1499	0.0666	0.0919	0.1468
	CVAE	**0.0782**	0.1008	0.1433	0.0555	0.0423	0.1457	**0.0751**	0.0877	0.1438

and CVAE architecture and hyperparameters, we can achieve comparable results to, if not better than, those obtained by SMOTE and ADASYN. Confirming this conjecture is a topic for future research. Computing scripts associated with this contribution are publicly available in our GitHub repository [71].

CHAPTER 5
OUTFIN: A MULTI-DEVICE AND MULTI-MODAL DATASET FOR OUTDOOR LOCALIZATION BASED ON THE FINGERPRINTING APPROACH

5.1 Introduction

Location-Based Services (LBS) has become a multibillion-dollar industry that is expected to continue to steadily grow over the upcoming years [72]. Some of these services include location-based marketing [73], authentication [74], gaming [75], and social networking [76], among others. A key enabling technology at the heart of such services is positioning [77]. However, the de facto standard for positioning, the Global Navigation Satellite System (GNSS), has two major issues that limit the use of LBS. First, the availability and accuracy of GNSS are severely degraded in urban areas due to shadowing and multipath effects [78]. Second, GNSS chipsets are notorious for being power-hungry, which is problematic for power-constrained devices such as smartphones and smartwatches [79]. A more energy-efficient approach for positioning is achieved using cellular networks. Yet, the offered accuracy, which is in the order of tens [80] to hundreds [81] of meters, fails to satisfy the accuracy requirements imposed by many services and applications.

Recently, in an attempt to devise positioning solutions that can yield better performance, researchers have turned their attention to fingerprinting, a positioning tech-

nique that has achieved great success in the indoor positioning domain, a domain where GNSS signals are generally unavailable [82]. Despite its low complexity and ability to produce accurate location estimates, the main drawback of fingerprinting is the laborious and time-consuming site surveying task. This drawback has led many studies to resort to either simulated [83] or crowdsourced data [84], where the former never fully reflects the real world and the latter may suffer from integrity and consistency problems. The proposal of OutFin aims at addressing these drawbacks by making real-world measurements and reliable ground truth coordinates publicly available.

Table 5.1 summarizes the main aspects of publicly available fingerprinting datasets published since 2014. Compared to these datasets, OutFin combines several features that place it in a unique position:

- To the best of our knowledge, OutFin is the first multi-modal, outdoor fingerprints dataset to be publicly available.
- The data was collected using two contemporary smartphones rather than outdated smartphones or custom-built platforms.
- The data was collected at highly granular RPs with 61 to 183 cm spacing.
- OutFin not only provides location fingerprints, but it also provides information about the devices that generated them (e.g., the service set identifier of an access point, the communication protocol of a Bluetooth device, and the number of neighboring cells of a serving cell).
- OutFin is accompanied by an interactive map that provides various information about the collection environment, such as RP coordinates (both ground truth and GPS estimates) and building ground elevations and heights.

Table 5.1: A comparison of the main aspects of publicly available fingerprinting datasets published since 2014. **Dataset**: the name of the dataset (if indicated) and a reference to its description. **Year**: the year the dataset was made available. **Category**: indicates whether the data was collected indoors or outdoors. **Environment**: a brief description of the collection environment. **Data type(s)**: the type(s) of data that was collected. **Device type(s)**: the type(s) of devices used to collect the data. **# of samples**: the highest place value of the number of samples in the dataset. **Granularity**: a descriptor indicating how close the RPs were to each other; High: indicates a spacing of fewer than 2 m, Medium: indicates a spacing between 2 and 8 m, and Low: indicates a spacing of greater than 8 m.

Dataset	**Year**	**Category**	**Environment**	**Data type(s)**	**Device type(s)**	**# of samples**	**Granularity**
UJIIndoorLoc [36]	2014	Indoor	Three university buildings	WiFi	Smartphone, Tablet	Tens of thousands	Medium
UJIIndoorLoc-Mag [85]	2015	Indoor	A research lab	sensor	Smartphone	Tens of thousands	Medium
Dataset described in [29]	2016	Indoor	A research facility	WiFi, sensor	Smartphone, Smartwatch	Tens of thousands	High
Dataset described in [37]	2016	Indoor	A university building	WiFi, Bluetooth, sensor	Smartphone	Thousands	High
PerfLoc [86]	2016	Indoor	An office building, two industrial warehouses, and a subterranean structure	WiFi, cellular, sensor	Smartphone	Millions	Medium
AmbiLoc [87]	2017	Indoor	An apartment and two university buildings	TV, FM, cellular	Dedicated data acquisition platform	Thousands	Medium
MagPIE [88]	2017	Indoor	Three university buildings	sensor	Smartphone	Hundreds of thousands	High
Dataset described in [89]	2018	Indoor	A university library	WiFi	Smartphone	Hundreds of thousands	High
Dataset described in [90]	2018	Indoor	Four residential homes	Bluetooth, sensor	Dedicated data acquisition platform	Hundreds of thousands	High
Dataset described in [58]	2018	Indoor	A university library	Bluetooth	Smartphone	Thousands	Medium
Dataset described in [91]	2018	Indoor	A research facility	Bluetooth	Smartphone, Dedicated data acquisition platform	Millions	High
Dataset described in [59]	2018	Outdoor	A large-scale urban area and a large-scale rural area	Sigfox, LoRaWAN	Dedicated data acquisition platform	Hundreds of thousands	Low
Dataset described in [57]	2019	Indoor	Two university buildings	Bluetooth	Smartphone	Thousands	High
Dataset described in [92]	2019	Indoor, Outdoor	Worldwide	Cellular	Smartphone	Millions	Low
OutFin [93]	**2020**	**Outdoor**	**A university campus**	**WiFi, Bluetooth, cellular, sensor**	**Smartphone**	**Hundreds of thousands**	**High**

In addition to facilitating the research and development of outdoor positioning solutions that are based on the fingerprinting approach, OutFin might spur innovation in other research realms, including but not limited to: machine learning [94], Bayesian optimization [95], simultaneous localization and mapping [96], and map-matching [97].

The remainder of this chapter is organized as follows. Section 5.2 describes the data collection platform, environment, and procedure, respectively. Section 5.3 provides an overview of the data files and their formats. Section 5.4 presents several experiments that validate the technical quality of the dataset.

5.2 Methods

5.2.1 Data Acquisition Platform

OutFin was created using two smartphones for data acquisition: Phone 1 and Phone 2. The former was released in the U.S. market on March 8, 2019, while the latter was released on October 24, 2019. Both smartphones ran on Android 10, released on September 3, 2019. The motivation behind choosing Android-powered smartphones was twofold. First, Android provides APIs that allow for acquiring raw data at the hardware level. Second, Android-powered smartphones account for over 74 % of the market share worldwide [43]. The two smartphones were attached to a tripod head using a dual mount that horizontally separated them by 10 cm (see Figure 5.1a). Both smartphones were in portrait mode. The tripod kept them at a fixed height of 132 cm. The tripod head was adjusted to tilt the smartphones at a $\sim 40^\circ$ angle to the vertical plane. The same set of third-party apps used for data collection were installed on both smartphones. These apps, which can be downloaded from the Google Play Store, included: WiFi Analyzer Pro (App 1) [98], Bluetooth

Scanner Extreme Edition (App 2) [99], NetMonitor Pro (App 3) [44], and Physics Toolbox Sensor Suite Pro (App 4) [100]. The apps allowed for conveniently collecting and exporting WiFi, Bluetooth, cellular, and sensor data, respectively.

5.2.2 Data Collection Environment

Data collection was performed at the University of Denver's campus where four separate sites were considered. The motivation behind collecting data at separate sites was to offer diversity. For instance, each site is different in terms of its reference points' number, arrangement, and spacing. Also, due to different ground elevations and heights of surrounding buildings, each site has different visibility to the GNSS. This is reflected by GPS errors produced at a given site. The mean GPS error was 12.1 m, 11.4 m, 4.3 m, and 12.7 m for the first, second, third, and fourth site, respectively. GPS estimates are provided in OutFin to help researches compare their system's performance to that obtained by GPS. A description of the data collection sites is provided below:

Site 1: Site 1 represents a portion of a covered sidewalk next to the east side of the 11.8 m high Boettcher Auditorium (see Figure 5.1a). Site 1 contained 31 RPs arranged in three north-to-south lines (see Figure 5.2). The spacing between

(a) Site 1 (b) Site 2 (c) Site 3 (d) Site 4

Figure 5.1: Pictures of the four sites where data was collected

RPs in each line was fixed at 152.5 cm and the distance between lines was fixed at 76.25 cm.

Site 2: Site 2 is ∼245 m north of Site 1 and represents a portion of a covered sidewalk next to the north side of the 11.5 m high Sie International Relations Complex (see Figure 5.1b). Site 2 contained 23 RPs arranged in a single east-to-west line (see Figure 5.2). The spacing between RPs was fixed at 101.5 cm.

Site 3: Site 3 is ∼40 m south of Site 2 and represents a portion of an open terrace next to the south side of the Sie International Relations Complex (see Figure 5.1c). Site 3 contains 35 RPs arranged in a seven-column and five-row grid (see Figure 5.2). The spacing between column RPs and row RPs were fixed at 61 cm.

Site 4: Site 4 is ∼288 m south of Site 3 and represents a portion of an open sidewalk by the south and west sides of the 13.4 m high Seeley Mudd Science Building (see Figure 5.1d). Site 4 contains 33 RPs arranged in a three-column and eleven-row grid (see Figure 5.2). The spacing between column RPs was fixed at 183 cm, while the spacing between row RPs was fixed at 146.5 cm.

Each RP is uniquely identified by an integer (an ID number) that symbolizes its order in the collection campaign. For example, data collection started with RP 1 on November 3, 2019, and ended with RP 122 on November 9, 2019. The ground truth locations of RPs belonging to a site are expressed with respect to a local frame of reference. Additionally, the easting and northing (X,Y) coordinates of all RPs were provided with respect to a global coordinate system (i.e., NAD83(2011)/Colorado Central). This was accomplished with help from the university's Department of Ge-

ography & the Environment and by using a geographic information system software [101].

5.2.3 Procedure

Data collection spanned six days (3–5/11/2019 and 7–9/11/2019) and involved four sites with a total of 122 RPs. The RPs surveyed each day are indicated in Figure 5.2. The sequence of steps performed during a day of data collection are described below:

Step 1: Before mounting the smartphones to the tripod, App 4 was launched to collect magnetic field measurements by rotating the smartphones around their X, Y, and Z axes multiple times (see Figure B.5). This process was performed for at least two minutes at a sampling rate of 1 Hz. The resultant data was exported

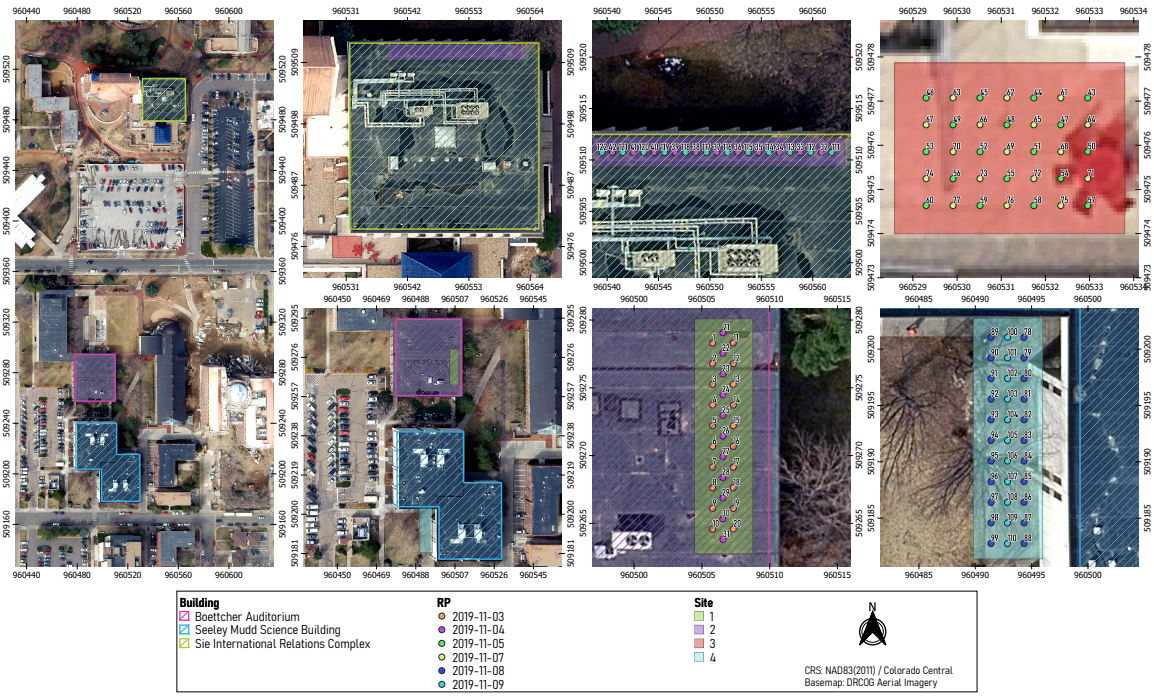

Figure 5.2: An aerial map of the collection environment showing the four collection sites and the 122 RPs. RPs are color-coded according to the date of collection.

as a Comma-Separated Values (CSV) file, named with the smartphone's name and date (e.g., `Phone1_051119.csv`). Such data can be used to offset the hard-iron distortion caused by placing the smartphones close to each other. After this process, the smartphones were mounted to the tripod and placed at the RP where data was to be collected.

Step 2: App 1 was launched to collect WiFi data, ensuring that at least two WiFi scans were performed along the four cardinal directions by routing the tripod head counterclockwise, $\sim 90^\circ$ at a time. A WiFi scan recorded the RSS from all APs in range in addition to information about the APs themselves. Android only supports passive scanning, and the duration of a scan varies depending on the smartphone's WiFi hardware and firmware. However, Google recently released a restriction that limits the frequency of scans that an app can perform to only four times in a 2-minute period [102]. This restriction applies to Android 9 and higher. The app reported scan results approximately every 30 s for Phone 1 and every 25 s for Phone 2. For Site 1 and 4's RPs, data collection started facing south and ended facing west. For Site 2 and 3's RPs, data collection started facing west and ended facing north. Collecting data along four directions mitigates the shadowing effect caused by the body of the data collector who is constantly facing the smartphone screens. Scan outcomes were exported as a CSV file, named with the smartphone's model as a prefix and the RP's ID as a suffix (e.g., `Phone2_WiFi_73.csv`).

Step 3: App 2 was launched to collect Bluetooth data. Android allows active Bluetooth scanning; thus, scans can be triggered by a user-level app. A Bluetooth scan involves an inquiry scan of approximately 12 s, followed by a page scan for each discovered device to retrieve its information and the RSS [103]. The

duration of a scan, for both smartphones, took anywhere between 15 and 30 s, primarily depending on the number of discoverable devices in the area. As in Step 2, the shadowing effect was accounted for by performing two scans along each cardinal direction. Scan results were exported as a CSV file with a naming convention like that described in Step 2 (e.g., `Phone1_Bluetooth_29.csv`).

Step 4: App 3 was launched to collect cellular data. A smartphone's cellular modem constantly scans the cellular network for cell selection/reselection and handover purposes. Android provides APIs to extract information associated with scans such as RSRP and cell identity information [45]. The sampling frequency can be set manually and was fixed to 1 Hz. As noted in Step 2, the shadowing effect was accounted for by collecting at least fifteen samples along each cardinal direction. Collected data was exported as a CSV file with a naming convention like that described previously (e.g., `Phone2_Cellular_14.csv`). Moreover, App 3 allowed for collecting GPS data as part of the data record. The GPS readings corresponding to RPs belonging to the same site were extracted and stored under a CSV file named with the site's name as a prefix and the smartphone's model and app name as a suffix (e.g., `Site1_GPS_Phone1_App3.csv`).

Step 5: App 4 was launched to collect sensor data. A smartphone's built-in sensors can be classified as either hardware-based, such as the magnetometer and gyroscope, or software-based, such as the gravity and linear acceleration sensors. Android provides APIs for accessing and acquiring raw sensor data at defined rates [104]. The sampling frequency was set to 1 Hz. Although sensor measurements are not subject to the shadowing effect, data was collected along the four cardinal directions to both conform with the

survey pattern established above and diversify the dataset since magnetic field strength can vary greatly even within a small area (in the orders of a few centimeters or less) [105]. At least fifteen samples were collected along each direction, following the same directions described in Step 2. Sensor data was exported as a CSV file with a naming convention like that described previously (e.g., `Phone1_Sensors_58.csv`). App 4 also allowed for collecting GPS data as part of the data record. As in Step 4, the GPS readings corresponding to RPs belonging to the same site were extracted and stored under a CSV file with a naming convention like that described in Step 4 (e.g., `Site3_GPS_Phone2_App4.csv`).

Step 6: The tripod was moved to the next RP and Steps 2–5 were repeated. This process continued until all RPs designated for a given day were surveyed.

5.3 Data Records

On April 2, 2020, the OutFin dataset was made publicly available on figshare [93]. Figure 5.3 shows the dataset's file structure and presents an overview of all CSV file types, their field labels, and a data record example. A description of the CSV file types and their field labels is provided below:

I. `<phone>_WiFi_<RP>.csv` contains WiFi data collected by a smartphone via App 1:

   1. `SSID`: The Service Set IDentifier (SSID) (i.e., the AP's network name).
   2. `BSSID`: The Basic Service Set IDentifier (BSSID) (i.e., the AP's Media Access Control (MAC) address) encoded as an integer.
   3. `Channel`: The channel number that the AP uses for communication.

4. `Width`: The bandwidth of the channel in megahertz (MHz); can be `20`, `40`, or `80` MHz.

5. `Center_Frequency_0`: The center frequency of the primary channel in MHz.

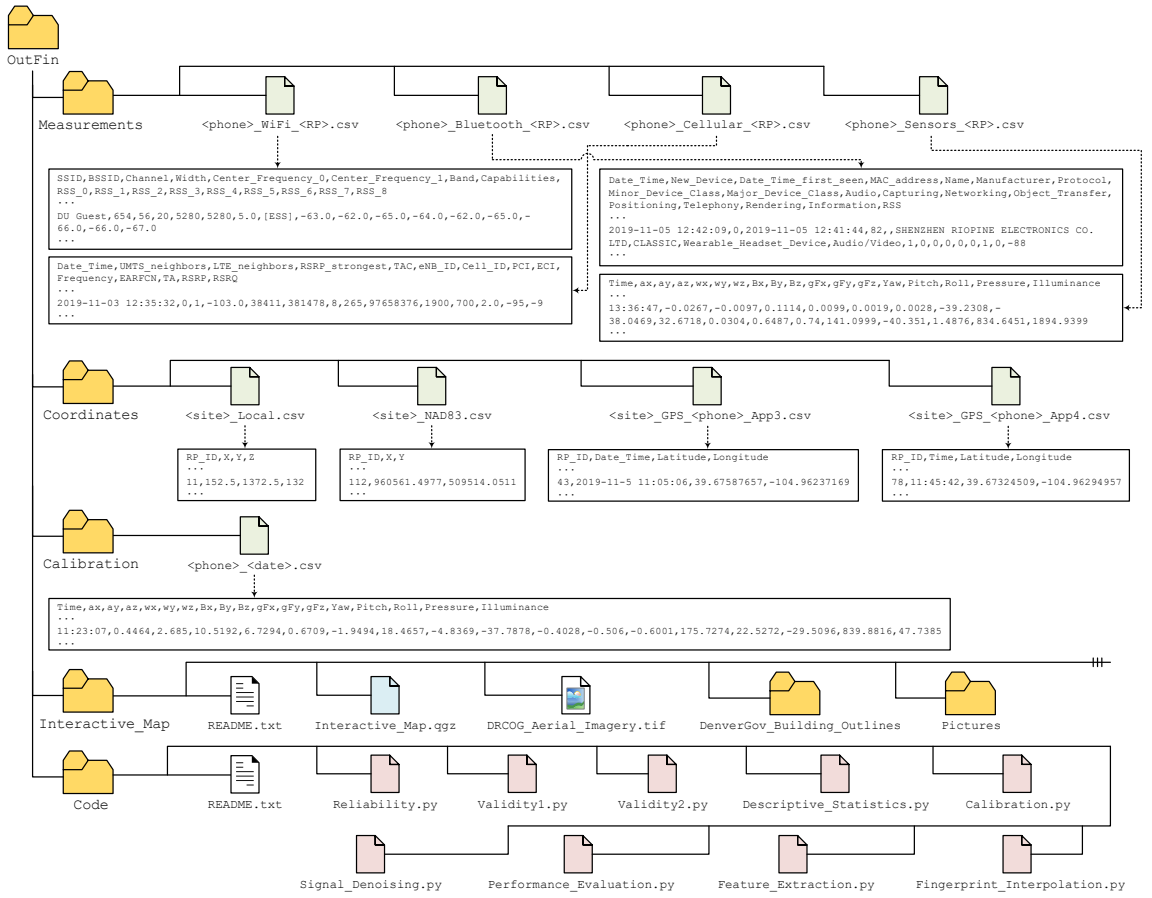

Figure 5.3: Directory tree of the OutFin dataset along with CSV file types and example data records. `<phone>` $\in$ {`Phone1`,`Phone2`}, `<RP>` $\in$ {`1`,`2`,`...`,`122`}, `<site>` $\in$ {`Site1`,`Site2`,`Site3`,`Site4`}, and `<date>` $\in$ {`031119`,`041119`,`051119`,`071119`,`081119`,`091119`}.

6. `Center_Frequency_1`: The center frequency of the 40 or 80 MHz-wide channel in MHz. If a 20-MHz channel is used, then `Center_Frequency_1` ≡ `Center_Frequency_0`.
7. `Band`: The AP's frequency band in GHz; can be either `2.4` or `5` GHz.
8. `Capabilities`: Describes the authentication, key management, and encryption schemes supported by the AP.

9–17. `RSS_0-RSS_8`: The RSSs in dBm, with respect to the back-to-back scans.

II. `<phone>_Bluetooth_<RP>.csv` contains Bluetooth data collected by a smartphone via App 2:

1. `Date_Time`: The date and time the scan was triggered as `YYYY-MM-DD` and `hh:mm:ss`. Denver, Colorado is in the Mountain Time Zone, which is seven hours behind Coordinated Universal Time (UTC-07:00).
2. `New_Device`: A binary flag that is set to 1 if the remote Bluetooth device is discovered for the first time at the current RP.
3. `Date_Time_first_seen`: The date and time the device was first discovered at the current RP. The date and time formats are as described above.
4. `MAC_address`: The device's MAC address encoded as an integer.
5. `Name`: The device's friendly name.
6. `Manufacturer`: The device's manufacturer name.
7. `Protocol`: The Bluetooth protocol that the device uses for communication; can be `CLASSIC` (Basic Rate/Enhanced Data Rate (BR/EDR)), `BLE`, or `DUAL` (BR/EDR + BLE).

8, 9. `Minor_Device_Class`, `Major_Device_Class`: Indicates the device's minor and major classes, respectively, as specified by the Bluetooth Special Interest Group (SIG)) [106].

10–17. `Audio`, `Capturing`, `Networking`, `Object_Transfer`, `Positioning`, `Telephony`, `Rendering`, `Information`: Binary flags that are set to 1 if the device is associated with any of the eight service classes specified by the Bluetooth SIG [106].

18. `RSS`: The RSS in dBm.

III. `<phone>_Cellular_<RP>.csv` contains cellular data collected by a smartphone via App 3. It should be noted that the entire collection environment was covered by LTE cells. The PLMN identifier is 310410:

1. `Date_Time`: The date and time the sample was captured. The date and time formats are as described above.
2. `UMTS_neighbors`: The number of neighboring UMTS cells.
3. `LTE_neighbors`: The number of neighboring LTE cells.
4. `RSRP_strongest`: The RSRP, in dBm, corresponding to the strongest neighboring cell, which employs the same technology as the serving cell.
5. `TAC`: The Tracking Area Code, which uniquely defines a group of cells within a PLMN.
6. `eNB_ID`: The eNB IDentifier that is used to uniquely identify an eNB (i.e., a base station in LTE) within a PLMN.
7. `Cell_ID`: The Cell IDentifier, which is an internal descriptor for a cell. It can take any value between `0` and `255`.

8. `PCI`: The Physical Cell Identifier that is used to indicate the physical layer identity of a cell. It can take any value between `0` and `503`.
9. `ECI`: The ECI that is used to uniquely identify a cell within a PLMN. `ECI` = 256 × `eNB_ID` + `Cell_ID`.
10. `Frequency`: The downlink frequency band in MHz.
11. `EARFCN`: The downlink Evolved-UMTS Terrestrial Radio Access Network (E-UTRAN) Absolute Radio Frequency Channel Number.
12. `TA`: The Timing Advance (TA) value which ranges from `0` to `1282`. A change of 1 in TA corresponds to a 156m round-trip distance [107]. For example, if TA = 7, then the eNB is located within a 546m radius from the smartphone.
13. `RSRP`: The RSRP in dBm.
14. `RSRQ`: The RSRQ in dB.

IV. `<phone>_Sensors_<RP>.csv` contains sensor data collected by a smartphone via App 4:

1. `Time`: The time the sample was captured. The time format is as described above.

2–4. `ax`, `ay`, `az`: The linear acceleration, in meters per second squared (m/s^2), along the smartphone's X, Y, and Z axes, respectively.

5–7. `wx`, `wy`, `wz`: The angular velocity, in radian per second (rad/s), around the smartphone's X, Y, and Z axes, respectively.

8–10. `Bx`, `By`, `Bz`: The magnetic field strength, in µT, along the smartphone's X, Y, and Z axes, respectively.

11–13. `gFx`, `gFy`, `gFz`: The g-force measured as the ratio of normal force to gravitational force (FN/Fg), along the smartphone's X, Y, and Z axes, respectively.

14–16. `Yaw`, `Pitch`, `Roll`: The angle of rotation, in °, around the smartphone's X, Y, and Z axes, respectively.

17. `Pressure`: The atmospheric pressure in hectopascal (hPa).

18. `Illuminance`: The illuminance in lux (lx).

V. `<site>_Local.csv` contains the local coordinates of RPs belonging to a site. Each site has its own frame of reference and the origins are at RPs 10, 122, 60, and 99 for Sites 1, 2, 3, and 4, respectively.

1. `RP_ID`: The RP IDentifier.

2–4. `X`, `Y`, `Z`: The X, Y, and Z coordinates of the RP in cm.

VI. `<site>_NAD83.csv` contains the global coordinates of RPs belonging to a site with respect to the NAD83(2011)/Colorado Central coordinate system.

1. `RP_ID`: The RP IDentifier.

2, 3. `X`, `Y`: The X and Y coordinates of the RP in m.

VII. `<site>_GPS_<phone>_App3.csv` contains the GPS coordinates of RPs belonging to a site as computed by the smartphone's GPS chipset and reported by App 3.

1. `RP_ID`: The Reference Point IDentifier.

2. `Date_Time`: The date and time the sample was captured. The date and time formats are as described above.

3, 4. `Latitude`, `Longitude`: The latitude and longitude coordinates of the RP.

VIII. `<site>_GPS_<phone>_App4.csv` contains the GPS coordinates of RPs belonging to a site as computed by the smartphone's GPS chipset and reported by App 4.

1. `RP_ID`: The RP IDentifier.
2. `Time`: The time the sample was captured. The time format is as described above.

3, 4. `Latitude`, `Longitude`: The latitude and longitude coordinates of the RP.

IX. `<phone>_<date>.csv` contains sensors data collected by a smartphone via App 3 before the smartphone is mounted to the tripod. Field labels are identical to that described in IV (`<phone>_Sensors_<RP>.csv`).

5.4 Technical Validation

The technical quality of the OutFin dataset was evaluated using experiments that consider two basic requirements that any high-quality dataset should satisfy, i.e., reliability and validity. Additionally, as a demonstration of the dataset's potential for positioning applications, a number of practical usage examples are presented.

5.4.1 Measurement Reliability

A data acquisition platform is said to be reliable if it provides consistent measurements at different points in time. To this end, before the collection campaign, WiFi, Bluetooth, cellular, and sensor data was captured over three different days at the same location. Spearman's and Kendall's correlation coefficients were then used to quantify the degree of consistency between temporal measurements for a given

phone. Table 5.2 shows Spearman's and Kendall's correlation coefficients for the two smartphones for all possible pairs of days. Given that correlation results are high (i.e., close to the maximum value of 1.0), it can be concluded that the dataset possesses a high degree of reliability.

	Phone 1			Phone 2		
	$\{day_1, day_2\}$	$\{day_2, day_3\}$	$\{day_1, day_3\}$	$\{day_1, day_2\}$	$\{day_2, day_3\}$	$\{day_1, day_3\}$
WiFi						
Spearman's ρ	0.960	0.949	0.946	0.952	0.968	0.936
Kendall's τ	0.837	0.826	0.815	0.828	0.877	0.796
Bluetooth						
Spearman's ρ	0.575	0.736	0.700	0.716	0.889	0.790
Kendall's τ	0.454	0.609	0.578	0.584	0.786	0.683
Cellular						
Spearman's ρ	0.964	0.964	1.0	0.964	0.964	1.0
Kendall's τ	0.904	0.904	1.0	0.904	0.904	1.0
Sensors						
Spearman's ρ	0.928	0.970	0.933	0.960	0.990	0.943
Kendall's τ	0.823	0.911	0.852	0.897	0.955	0.852

Table 5.2: Results of the correlation analysis between the measurements obtained on three different days for Phone 1 and Phone 2. Spearman's ρ varies between -1 and $+1$ with 0 implying no correlation, while values of -1 or $+1$ imply an exact monotonic relationship. Kendall's τ varies between -1 and $+1$. Values close to $+1$ indicate strong agreement, while values close to -1 indicate strong disagreement. For WiFi, the results were generated using averaged RSS readings of fifty randomly selected APs that were observed over the three days. For Bluetooth, the results were generated using averaged RSS readings of fifteen randomly selected devices that were observed over the three days. The relatively lower correlation results obtained for Bluetooth is attributed to the fact that Bluetooth signals are more vulnerable to channel gain and fast fading than WiFi signals, causing measurements to fluctuate severely over time [18]. For Cellular, the results were generated using averaged readings of UMTS neighbors, LTE neighbors, RSRP strongest, frequency, E-UTRAN Absolute Radio Frequency Channel Number (EARFCN), RSRP, and RSRQ from a cellular base station that a phone connected to over the three days. For Sensors, the results were generated using the averaged readings of linear acceleration, angular velocity, magnetic field strength, g-force, angle of rotation, atmospheric pressure, and illuminance. The p-value of all results ranged between 0.0 and 0.02.

5.4.2 Measurement Validity

A data acquisition platform is said to be valid if it accurately measures what it is intended to measure. In some cases, this requires the presence of theoretically-derived data to compare experimental data against. For example, WiFi RSS values can be computed using a path loss model. An input to the model is the distance between the transmitter and receiver. However, obtaining such inputs is not feasible since the exact location of all APs in the environment needs to be known. In the absence of theoretically-derived data, validity can be assessed by comparing data generated by different sources and checking for consistency. Accordingly, for a given day, Spearman's and Kendall's correlation coefficients were used to quantify the degree of consistency between the measurements obtained by the phones. The correlation results for the foregoing three days are shown in Table 5.3. These results demonstrate high levels of consistency, which attests to the validity of the dataset.

As graphical evidence of measurement validity, Figure 5.4 compares some of the data generated by the smartphones at randomly selected RPs side-by-side. Plots of the same data type exhibit the same profile despite corresponding to two different smartphones. Table 5.4 reports descriptive statistics of the data collected by each phone with respect to various variables. These statistics are compared against previously reported reference values, where applicable. The statistics displayed in Table 5.4 further support the validity of the dataset by ruling out the possibility that the dataset contains unrealistic, erratic, or random data.

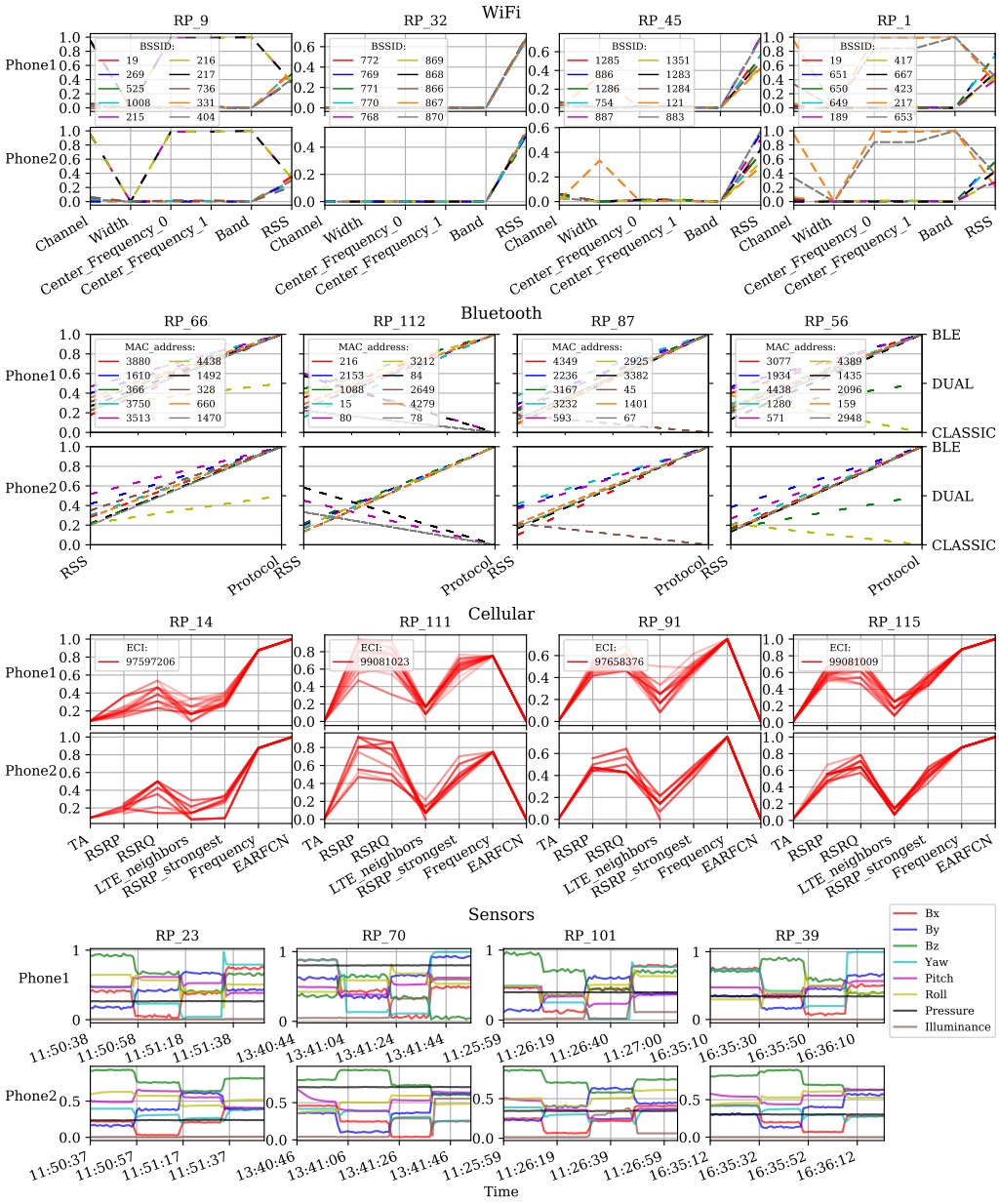

Figure 5.4: Visualization of the data collected by Phone 1 and Phone 2 over randomly selected RPs. WiFi, Bluetooth, and cellular data are represented using parallel coordinate plots of the most important features, while sensor data are represented using time plots of magnetic field strength, angle of rotation, atmospheric pressure, and illuminance. All features are normalized between 0 and 1.

5.4.3 Usage Examples

This subsection provides a brief demonstration of some of the application domains that OutFin can be used for. These include *fingerprint interpolation*, *feature extraction*, *performance evaluation*, and *signal denoising*.

	day_1	day_2	day_3
WiFi			
Spearman's ρ	0.920	0.925	0.893
Kendall's τ	0.773	0.796	0.728
Bluetooth			
Spearman's ρ	0.763	0.706	0.843
Kendall's τ	0.657	0.535	0.703
Cellular			
Spearman's ρ	1.0	1.0	1.0
Kendall's τ	1.0	1.0	1.0
Sensors			
Spearman's ρ	0.725	0.774	0.752
Kendall's τ	0.617	0.720	0.676

Table 5.3: Results of the correlation analysis between the measurements obtained from Phone 1 and Phone 2 for three different days. Spearman's ρ varies between -1 and $+1$ with 0 implying no correlation, while values of -1 or $+1$ imply an exact monotonic relationship. Kendall's τ varies between -1 and $+1$. Values close to $+1$ indicate strong agreement, while values close to -1 indicate strong disagreement. For WiFi, the results were generated using the averaged RSS readings of fifty randomly selected APs that were observed by both phones for a given day. For Bluetooth, the results were generated using the averaged RSS readings of fifteen randomly selected devices that were observed by both phones for a given day. For Cellular, the results were generated using averaged readings of UMTS neighbors, LTE neighbors, RSRP strongest, frequency, EARFCN, RSRP, and RSRQ of a cellular base station that both phones connected to for a given day. For Sensors, the results were generated using the averaged readings of linear acceleration, angular velocity, magnetic field strength, g-force, angle of rotation, atmospheric pressure, and illuminance for a given day. The p-value of all results ranged between 0.0 and 0.01.

	Phone 1				Phone 2				
	Min	Max	Mean	SD	Min	Max	Mean	SD	Reference values
WiFi									
Detected SSIDs	12	51	26.09	8.95	9	40	21.29	6.80	-
Detected BSSIDs	98	223	159.32	31.68	67	168	114.97	23.92	-
RSS (dBm)	-97	-53.33	-85.82	6.86	-99	-38	-84.20	6.88	$\approx [-102, -34]$ [108]
Bluetooth									
Detected MAC addresses	5	205	59.50	47.46	4	168	45.45	35.99	-
RSS (dBm)	-98	-53	-86.28	4.69	-113	-65	-99.40	5.35	$\approx [-110, -48]$ [109]
Cellular									
Detected ECI	1	5	1.45	0.91	1	4	1.35	0.73	-
LTE neighbors	0	12	2.36	1.53	0	14	2.45	1.79	-
RSRP strongest (dBm)	-128	-81	-103.32	6.90	-127	-82	-105.18	8.26	-
RSRP (dBm)	-118	-82	-99.86	6.28	-118	-82	-100.89	6.98	$\approx [-120, -70]$ [110]
RSRQ (dB)	-20	-7	-12.83	2.33	-20	-6	-12.87	2.48	$\approx [-24, -5]$ [110]
Sensors									
Magnitude of magnetic field (µT)	38.52	51.07	44.49	3.51	29.45	73.03	51.90	13.40	≈ 51 [111]
Atmospheric pressure (hPa)	833.14	845.02	837.93	3.13	831.67	843.52	836.37	3.12	$\approx (829.66, 843.21, 836.43)$ [112]
Illuminance (microlux (µlx))	1e−6	0.1508	0.0138	0.0271	2e−7	0.1243	0.0104	0.0207	$\approx (0.1, 0.01, 1e-6)$ [113]

Table 5.4: Descriptive statistics of the OutFin dataset. These include the minimum, maximum, mean, and standard deviation of the most important variables. Reference values are provided where applicable. Small variations in results between the phones are mainly attributed to *device heterogeneity* [114] (e.g., the sensitivity of the radio receiver or sensor). The reference value for the magnitude of the magnetic field represents the Earth's magnetic field around Denver, Colorado. The reference values for atmospheric pressure represent, respectively, the minimum, maximum, and mean recorded atmospheric pressure in Denver, Colorado, during the data collection period. The reference values for illuminance represent the light intensity for sunlight, daylight, and twilight, respectively. An hour-by-hour description of other weather conditions, such as temperature, humidity, and visibility at the time of data collection can be retrieved from [115].

Fingerprint Interpolation

Building a fingerprint map is usually required to provide positioning in a continuous fashion. The resolution of a map depends highly on the RP granularity (the higher the RP granularity, the better the map resolution). However, collecting fingerprints at highly granular RPs is time-consuming and labor intensive. Thus, interpolation methods are often employed to calculate the fingerprints between the locations of known fingerprints [116]. The choice of an interpolation technique is pivotal to the resulting map. For example, Figure 5.5 compares the magnetic field maps created for

Site 3 by two different interpolation techniques, namely linear and cubic interpolation. Clearly, the resulting maps are not identical, which suggests that a positioning algorithm would exhibit a difference in performance depending on the employed map.

Feature Extraction

A WiFi fingerprint has entries for all APs detected in an entire environment, but only a subset of these APs is observed at different locations. This is especially true for large-scale environments. For example, OutFin contains measurements from 1,379 unique APs; however, on average, only 10 % of these APs are observed at any given RP. Consequently, feature extraction techniques are often utilized to reduce the dimensionality of the fingerprint space in order to achieve efficient and robust positioning [117]. Figure 5.6 compares two dimensionality reduction methods, i.e., the AE and Principal Component Analysis (PCA). The reconstruction cost obtained by the AE is lower than that obtained by PCA. This suggests that the AE is better at compressing the fingerprint space into a lower dimensional representation that comprises the informative content of the fingerprint space.

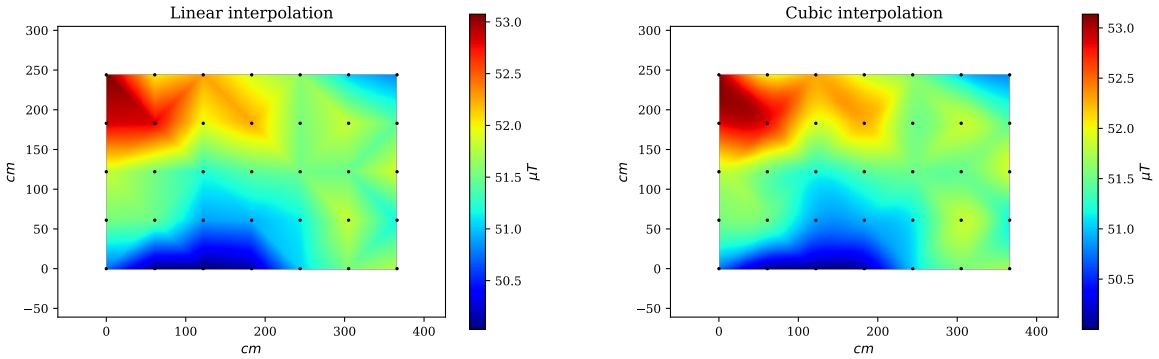

Figure 5.5: Interpolated magnetic field magnitude of Site 3 using linear interpolation (left) and cubic interpolation (right). The maps were generated using calibrated magnetic field measurements from Phone 1 and Phone 2.

Performance Evaluation

When proposing a new positioning method, the performance of the proposed method is often evaluated against the performance of previously proposed methods. It is often the case that at the heart of many of the methods benchmarked against is a machine learning algorithm, such as kNN, SVM, Decision Tree, or Naive Bayes [68]. Therefore, with the purpose of comparing the performance of such algorithms, the positioning problem was casted as a classification task where each RP is treated as a unique class. Various performance metrics were considered, including classification metrics, positioning error, and computational complexity. For the sake of fair comparison, the parameters of each algorithm were fine-tuned using grid search and cross-validation. Evaluation results, shown in Table 5.5, are reported on the Bluetooth measurements collected from Site 4. The results demonstrate that different algorithms can be ranked differently depending on the chosen performance metric.

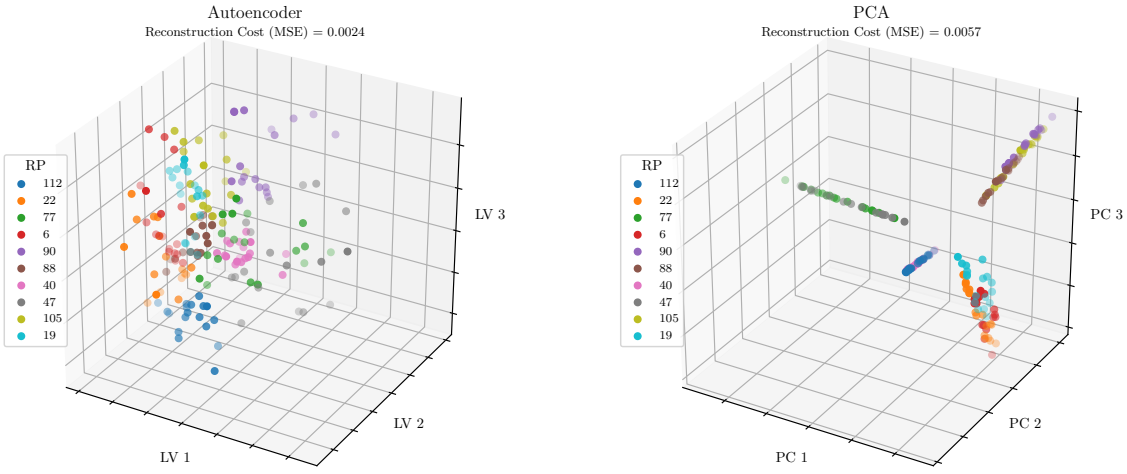

Figure 5.6: The 3D codes for 18 WiFi RSS measurements (9 measurements per phone) for 10 randomly selected RPs produced by the AE (left) and PCA (right). MSE: mean squared error; PC: principal component; LV: latent variable.

For example, the best classification accuracy was achieved by Radial Basis Function (RBF) SVM, while the lowest mean positioning error was achieved by kNN.

Signal Denoising

Signal loss can negatively impact the performance of a positioning system. Thus, denoising techniques are often integrated as a preprocessing step to enhance positioning [119]. As an example, a DAE was utilized as a denoising agent where the feature vector of a cellular fingerprint is corrupted to emulate randomized loss of data. The degree of corruption is controlled by a predefined probability (p_{loss}) where, for example, a p_{loss} of 0.03 indicates a 3 % chance of setting a feature to zero. Figure 5.7 demonstrates the differences in performance between using noisy cellular features and their denoised versions for positioning in Site 2. On average, the use of the denoising step resulted in a 1.43 % improvement in accuracy and a 13.25 cm reduction in positioning error.

5.4.4 Code Availability

Well-documented scripts, written in `Python 3.6.4` [120], are present alongside the dataset (also available on GitHub [121]). These include the scripts used to generate

	Classification Metric				Positioning Error (cm)				Computational Complexity [118]	
	Accuracy	Precision	Recall	F1	Min	Max	Mean	SD	Training	Prediction
Algorithm										
kNN	0.948	0.964	0.948	0.945	0.0	366.0	11.46	51.52	-	$\mathcal{O}(np)$
RBF kernel SVM	0.962	0.970	0.962	0.961	0.0	1098.0	18.81	121.46	$\mathcal{O}(n^2p + n^3)$	$\mathcal{O}(n_{sv}p)$
Decision Tree	0.957	0.967	0.957	0.956	0.0	732.0	15.19	83.19	$\mathcal{O}(n^2p)$	$\mathcal{O}(p)$
Naive Bayes	0.910	0.956	0.910	0.911	0.0	549.0	23.82	82.38	$\mathcal{O}(np)$	$\mathcal{O}(p)$

Table 5.5: Performance evaluation of commonly used algorithms for positioning with respect to various metrics. The results were generated using 530 Bluetooth samples (60 % training and 40 % testing) collected by both phones from Site 4. n: number of training samples; p: number of features; n_{sv}: number of support vectors.

the results described in the Technical Validation section as well as a script to calibrate magnetic field measurements against hard/soft-iron distortions. The data required to replicate the experiments reside in `OutFin/Code/temporal_data`. Depending on the script, some of the following libraries may be required: `os, pandas, scipy, random, sklearn, matplotlib, numpy, statistics, keras, math`. Additionally, a thorough description of the collection environment in the form of an interactive map (developed using `QGIS 3.10` [101]) is provided. The map is composed of several layers that display information such as RP coordinates (both ground truth and smartphone estimated), pictures of the collection sites, and building height and ground elevation (as provided by the City and County of Denver [122]). High-resolution aerial imagery (3-inch), provided by the Denver Regional Council of Governments [123], are used as the basemap.

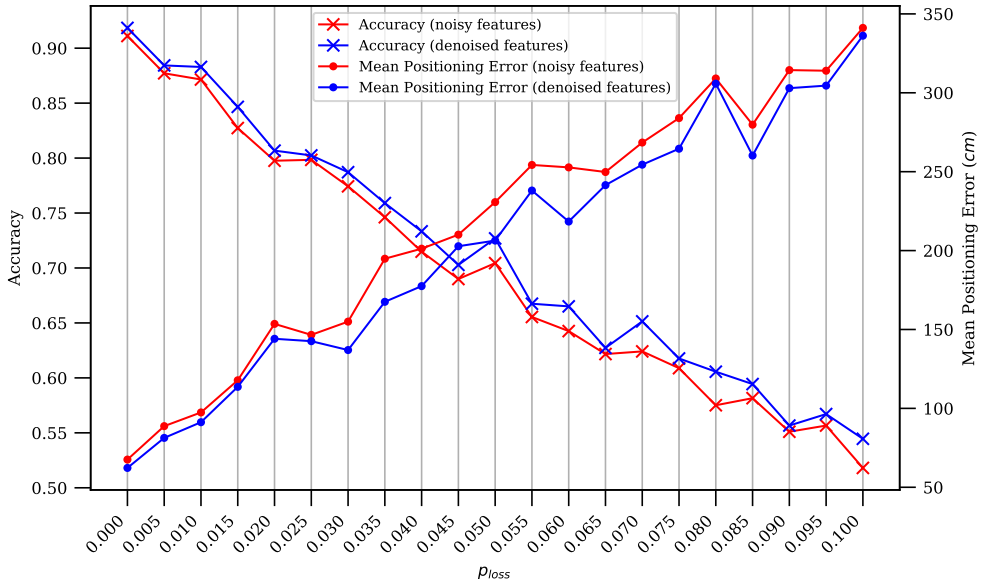

Figure 5.7: Noisy vs. denoised features for positioning. For a given p_{loss} value, the results were generated using 3,111 cellular samples collected by both phones from Site 2. A kNN algorithm is used for comparison where ∼60 % of the samples were used for training and the remaining ∼40 % for testing.

CHAPTER 6
CONCLUSION AND FUTURE WORK

To summarize, this dissertation put forward four main contributions: (i) a CNN-based smartwatch indoor positioning method using magnetic field fingerprints, (ii) a DAE-based symbolic indoor positioning method capable of handling missing cellular signal measurements using the serving eNodeB, (iii) a deep generative model-based oversampling method for highly imbalanced indoor positioning data, and (iv) a publicly available, multi-device and multi-modal outdoor fingerprinting dataset. These contributions were thoroughly discussed in chapters 2, 3, 4, and 5, respectively. For readers' convenience, in the following, we outline the concluding remarks and future research directions for each chapter separately.

Chapter 2 presented the design, development, and evaluation of a novel CNN-based indoor geomagnetic field fingerprinting system for smartwatch localization. The proposed system consists of two convolutional layers, two FC layers, and a Softmax layer. The system was built and tested using real world data collected from a representative indoor environment composed of multiple corridors and rooms. The system improved the mean localization error by 69.8 % and 225.7 % over kNN and SVM, respectively. Compared to kNN and SVM, the system possesses a significantly lower prediction latency making it well-suited for real-time user tracking applications. Chapter 2 also discussed how the Softmax layer of the system can be utilized to en-

hance the localization accuracy when used with sequence modeling algorithms. In this respect, we intend to extend this work by feeding the system's output to such an algorithm (e.g. Viterbi algorithm) for the purpose of further refining the system's output. This will allow the system to maintain temporal coherence when used in user tracking applications.

Chapter 3 presented the design and evaluation of a novel cellular-based symbolic indoor positioning method. At its core, the proposed method utilizes DAEs to alleviate the effects randomized signal loss has on positioning. Experimental results demonstrated that the proposed method outperforms two conventional methods with respect to several performance metrics. Moreover, Chapter 3 demonstrated that the performance gap becomes wider with the increased probability of signal loss. With regard to future work, we would like to investigate why in some cases there is low confusion between symbolic spaces that are close to each other and high confusion between symbolic spaces that are not close to each other. Also, we would like to study the problem of device heterogeneity in more detail with the aim of alleviating its effects on positioning performance.

Chapter 4 presented a VAE-based method that can effectively learn the distribution of the minority symbolic space(s) to generate synthetic fingerprints that promote enhancements in the performance of a positioning model. As part of future research, we intend to undertake a more in-depth analysis of the results to answer questions such as "Why does VAE generally produce better overall F1-scores than CVAE?" and "Why does VAE yield better minority space Precision and overall Precision when the minority spaces represent 50 % or less of the overall spaces, while CVAE performs better on these metrics when the minority spaces represent over 50 % of the overall spaces?". In addition, we would like to apply VAE and CVAE to other fingerprint types and investigate the effectiveness of other deep generative models such as GANs

and Conditional Generative Adversarial Networks (CGANs) for oversampling fingerprint data.

Chapter 5 presented OutFin, a publicly available, multi-device and multi-modal dataset for outdoor fingerprint-based positioning. The technical quality of the collected data was evaluated using experiments that consider both its reliability and validity. Additionally, Chapter 5 provided several usage examples to demonstrate the versatility of OutFin. For future work, we intend to augment OutFin with measurements collected from some of the emerging IoT wireless technologies such as SigFox and LoRa. Currently, very few publications regarding these technologies and fingerprinting can be found, and we mainly attribute this to the lack of fingerprinting datasets dedicated to these wireless technologies.

PUBLICATIONS

Journal Papers

1. **F. Alhomayani** and M.H. Mahoor. "OutFin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach." *Scientific Data*, 2021.

2. **F. Alhomayani** and M.H. Mahoor. "Deep learning methods for fingerprint-based indoor positioning: a review." *Location Based Services*, 2020.

Conference Papers

1. **F. Alhomayani** and M.H. Mahoor. "Oversampling highly imbalanced indoor positioning data using deep generative models." *IEEE SENSORS*, (under review).

2. **F. Alhomayani** and M.H. Mahoor. "Deep learning-based symbolic indoor positioning using the serving eNodeB." *IEEE International Conference on Machine Learning and Applications*, 2020.

3. **F. Alhomayani** and M.H. Mahoor. "Improved indoor geomagnetic field fingerprinting for smartwatch localization using deep learning." *International Conference on Indoor Positioning and Indoor Navigation*, 2018.

Datasets

1. **F. Alhomayani** and M.H. Mahoor. "Cellular network measurements for symbolic indoor positioning" *figshare*, 2020.

2. **F. Alhomayani** and M.H. Mahoor. "OutFin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach." *figshare*, 2020.

REFERENCES

[1] C. Laoudias, A. Moreira, S. Kim, S. Lee, L. Wirola, and C. Fischione, "A survey of enabling technologies for network localization, tracking, and navigation," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 3607–3644, Fourthquarter 2018.

[2] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *Computer*, vol. 34, no. 8, pp. 57–66, 2001.

[3] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1067–1080, 2007.

[4] K. Subbu, C. Zhang, J. Luo, and A. V. Vasilakos, "Analysis and status quo of smartphone-based indoor localization systems.," *IEEE Wireless Commun.*, vol. 21, no. 4, pp. 106–112, 2014.

[5] P. Davidson and R. Piché, "A survey of selected indoor positioning methods for smartphones," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1347–1370, 2017.

[6] H. Huang, G. Gartner, J. M. Krisp, M. Raubal, and N. V. de Weghe, "Location based services: Ongoing evolution and research agenda," *Journal of Location Based Services*, vol. 12, no. 2, pp. 63–93, 2018. eprint: https://doi.org/10.1080/17489725.2018.1508763. [Online]. Available: https://doi.org/10.1080/17489725.2018.1508763.

[7] D. Macagnano, G. Destino, and G. Abreu, "Indoor positioning: A key enabling technology for iot applications," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, IEEE, 2014, pp. 117–118.

[8] P. Rashidi and A. Mihailidis, "A survey on ambient-assisted living tools for older adults," *IEEE journal of biomedical and health informatics*, vol. 17, no. 3, pp. 579–590, 2013.

[9] A. F. G. Ferreira, D. M. A. Fernandes, A. P. Catarino, and J. L. Monteiro, "Localization and positioning systems for emergency responders: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2836–2870, 2017.

[10] Z. Chen, C. Jiang, and L. Xie, "Building occupancy estimation and detection: A review," *Energy and Buildings*, vol. 169, pp. 260–270, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378778818301506.

[11] E. Mok and G. Retscher, "Location determination using wifi fingerprinting versus wifi trilateration," *Journal of Location Based Services*, vol. 1, no. 2, pp. 145–159, 2007. eprint: https://doi.org/10.1080/17489720701781905. [Online]. Available: https://doi.org/10.1080/17489720701781905.

[12] R. Jackermeier and B. Ludwig, "Exploring the limits of pdr-based indoor localisation systems under realistic conditions," *Journal of Location Based Services*, vol. 12, no. 3-4, pp. 231–272, 2018. eprint: https://doi.org/10.1080/17489725.2018.1541330. [Online]. Available: https://doi.org/10.1080/17489725.2018.1541330.

[13] P. Bahl and V. N. Padmanabhan, "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)*, vol. 2, Mar. 2000, 775–784 vol.2.

[14] S. Suksakulchai, S. Thongchai, D. M. Wilkes, and K. Kawamura, "Mobile robot localization using an electronic compass for corridor environment," in *IEEE SMC 2000 conference proceedings*, vol. 5, Oct. 2000, 3354–3359 vol.5.

[15] T. Starner, B. Schiele, and A. Pentland, "Visual contextual awareness in wearable computing," in *Digest of Papers. Second International Symposium on Wearable Computers*, 1998, pp. 50–57.

[16] M. Azizyan, I. Constandache, and R. Roy Choudhury, "Surroundsense: Mobile phone localization via ambience fingerprinting," in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '09, Beijing, China: ACM, 2009, pp. 261–272, ISBN: 978-1-60558-702-8.

[17] J. Luo, L. Fan, and H. Li, "Indoor positioning systems based on visible light communication: State of the art," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2871–2893, 2017.

[18] R. Faragher and R. Harle, "Location fingerprinting with bluetooth low energy beacons," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, Nov. 2015.

[19] F. Pereira, C. Theis, A. Moreira, and M. Ricardo, "Performance and limits of knn-based positioning methods for gsm networks over leaky feeder in underground tunnels," *Journal of Location Based Services*, vol. 6, no. 2, pp. 117–133, 2012. eprint: https://doi.org/10.1080/17489725.2012.692619. [Online]. Available: https://doi.org/10.1080/17489725.2012.692619.

[20] P. Mirowski, P. Whiting, H. Steck, R. Palaniappan, M. MacDonald, D. Hartmann, and T. K. Ho, "Probability kernel regression for wifi localisation," *Journal of Location Based Services*, vol. 6, no. 2, pp. 81–100, 2012. eprint: https:

//doi.org/10.1080/17489725.2012.694723. [Online]. Available: https://doi.org/10.1080/17489725.2012.694723.

[21] J. Tamas and Z. Toth, "Classification-based symbolic indoor positioning over the miskolc iis data-set," *Journal of Location Based Services*, vol. 12, no. 1, pp. 2–18, 2018. eprint: https://doi.org/10.1080/17489725.2018.1455992. [Online]. Available: https://doi.org/10.1080/17489725.2018.1455992.

[22] C. Zhou and A. Wieser, "Modified jaccard index analysis and adaptive feature selection for location fingerprinting with limited computational complexity," *Journal of Location Based Services*, vol. 13, no. 2, pp. 128–157, 2019. eprint: https://doi.org/10.1080/17489725.2019.1577505. [Online]. Available: https://doi.org/10.1080/17489725.2019.1577505.

[23] J. Chung, M. Donahoe, C. Schmandt, I.-J. Kim, P. Razavai, and M. Wiseman, "Indoor location sensing using geo-magnetism," in *Proceedings of the 9th international conference on Mobile systems, applications, and services*, ACM, 2011, pp. 141–154.

[24] B. Gozick, K. P. Subbu, R. Dantu, and T. Maeshiro, "Magnetic maps for indoor navigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 12, pp. 3883–3891, 2011.

[25] K. P. Subbu, B. Gozick, and R. Dantu, "Locateme: Magnetic-fields-based indoor localization using smartphones," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, Oct. 2013. [Online]. Available: https://doi.org/10.1145/2508037.2508054.

[26] N. Lee and D. Han, "Magnetic indoor positioning system using deep neural network," in *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sep. 2017, pp. 1–8.

[27] B. Li, T. Gallagher, A. G. Dempster, and C. Rizos, "How feasible is the use of magnetic field alone for indoor positioning?" In *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2012, pp. 1–9.

[28] M. Angermann, M. Frassl, M. Doniec, B. J. Julian, and P. Robertson, "Characterization of the indoor magnetic field for applications in localization and mapping," in *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2012, pp. 1–9.

[29] P. Barsocchi, A. Crivello, D. La Rosa, and F. Palumbo, "A multisource and multivariate dataset for indoor localization methods based on wlan and geomagnetic field fingerprinting," in *Indoor Positioning and Indoor Navigation (IPIN), 2016 International Conference on*, IEEE, 2016, pp. 1–8.

[30] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, *et al.*, *Tensorflow: Large-scale machine learning on heterogeneous distributed systems*, 2015. [Online]. Available: http://download.tensorflow.org/paper/whitepaper 2015.pdf.

[31] J. Tamas and Z. Toth, "Classification-based symbolic indoor positioning over the miskolc iis data-set," *Journal of Location Based Services*, vol. 12, no. 1, pp. 2–18, 2018. eprint: https://doi.org/10.1080/17489725.2018.1455992.

[32] M. Werner, C. Hahn, and L. Schauer, "Deepmovips: Visual indoor positioning using transfer learning," in *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2016, pp. 1–7.

[33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[34] M. Nowicki and J. Wietrzykowski, "Low-effort place recognition with wifi fingerprints using deep learning," in *International Conference Automation*, Springer, 2017, pp. 575–584.

[35] A. Moreira, M. J. Nicolau, F. Meneses, and A. Costa, "Wi-fi fingerprinting in the real world - rtls@um at the evaal competition," in *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct. 2015, pp. 1–10.

[36] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, *et al.*, "Ujiindoorloc: A new multi-building and multi-floor database for wlan fingerprint-based indoor localization problems," in *Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference on*, IEEE, 2014, pp. 261–270.

[37] Z. Tóth and J. Tamás, "Miskolc iis hybrid ips: Dataset for hybrid indoor positioning," in *Radioelektronika (RADIOELEKTRONIKA), 2016 26th International Conference*, IEEE, 2016, pp. 408–412.

[38] H. Rizk, M. Torki, and M. Youssef, "Cellindeep: Robust and accurate cellular-based indoor localization via deep learning," *IEEE Sensors Journal*, vol. 19, no. 6, pp. 2305–2312, Mar. 2019.

[39] H. Rizk and M. Youssef, "Monodcell: A ubiquitous and low-overhead deep learning-based indoor localization with limited cellular information," in *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2019, pp. 109–118.

[40] M. Arnold, S. Dorner, S. Cammerer, and S. Ten Brink, "On deep learning-based massive mimo indoor user localization," in *2018 IEEE 19th Interna-*

tional Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Jun. 2018, pp. 1–5.

[41] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, "Deep convolutional neural networks for massive mimo fingerprint-based positioning," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct. 2017, pp. 1–6.

[42] F. Alhomayani and M. H. Mahoor, *Cellular network measurements for symbolic indoor positioning*, Sep. 2020. [Online]. Available: https://figshare.com/articles/dataset/Cellular_Network_Measurements_for_Symbolic_Indoor_Positioning/13010387/1.

[43] *Mobile operating system market share worldwide (dec 2018 - dec 2019)*, StatCounter. [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/worldwide.

[44] *Netmonitor pro*, Vitaly V. [Online]. Available: https://play.google.com/store/apps/details?id=ru.v_a_v.netmonitorpro&hl=en_US.

[45] *Android.telephony*, Android Developers. [Online]. Available: https://developer.android.com/reference/android/telephony/package-summary.

[46] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[47] C. Burlin, Y. Le Calonnec, and L. Duperier, *Deep image inpainting*, 2017.

[48] R. Karkare, R. Paffenroth, and G. Mahindre, "Blind image denoising and inpainting using robust hadamard autoencoders," *arXiv preprint arXiv:2101.10876*, 2021.

[49] *Keras: The Python Deep Learning library.* [Online]. Available: https://keras.io/.

[50] *Classification metrics.* [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics.

[51] A. Khalajmehrabadi, N. Gatsis, and D. Akopian, "Modern wlan fingerprinting indoor positioning methods and deployment challenges," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1974–2002, thirdquarter 2017.

[52] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[53] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD

'01, San Francisco, California: Association for Computing Machinery, 2001, pp. 204–213, ISBN: 158113391X. [Online]. Available: https://doi.org/10.1145/502512.502540.

[54] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 225–252, 2008.

[55] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2012.

[56] E. S. Lohan, J. Torres-Sospedra, H. Leppäkoski, P. Richter, Z. Peng, and J. Huerta, "Wi-fi crowdsourced fingerprinting dataset for indoor positioning," *Data*, vol. 2, no. 4, 2017. [Online]. Available: https://www.mdpi.com/2306-5729/2/4/32.

[57] G. M. Mendoza-Silva, M. Matey-Sanz, J. Torres-Sospedra, and J. Huerta, "Ble rss measurements dataset for research on accurate indoor positioning," *Data*, vol. 4, no. 1, p. 12, 2019.

[58] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J.-S. Oh, "Semisupervised deep reinforcement learning in support of iot and smart city services," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 624–635, 2018.

[59] M. Aernouts, R. Berkvens, K. Van Vlaenderen, and M. Weyn, "Sigfox and lorawan datasets for fingerprint localization in large urban and rural areas," *Data*, vol. 3, no. 2, p. 13, 2018.

[60] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[61] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," *Advances in neural information processing systems*, vol. 28, pp. 3483–3491, 2015.

[62] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008, pp. 1322–1328.

[63] F. J. Aranda, F. Parralejo, F. J. Álvarez, and J. Torres-Sospedra, "Multi-slot ble raw database for accurate positioning in mixed indoor/outdoor environments," *Data*, vol. 5, no. 3, 2020. [Online]. Available: https://www.mdpi.com/2306-5729/5/3/67.

[64] K. E. Jeon, J. She, P. Soonsawad, and P. C. Ng, "Ble beacons for internet of things applications: Survey, challenges, and opportunities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 811–828, Apr. 2018.

[65] V. A. Fajardo, D. Findlay, R. Houmanfar, C. Jaiswal, J. Liang, and H. Xie, "Vos: A method for variational oversampling of imbalanced data," *arXiv preprint arXiv:1809.02596*, 2018.

[66] V. A. Fajardo, D. Findlay, C. Jaiswal, X. Yin, R. Houmanfar, H. Xie, J. Liang, X. She, and D. Emerson, "On oversampling imbalanced data with deep conditional generative models," *Expert Systems with Applications*, vol. 169, p. 114 463, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420311155.

[67] E. Ramentol, Y. Caballero, R. Bello, and F. Herrera, "Smote-rs b*: A hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory," *Knowledge and information systems*, vol. 33, no. 2, pp. 245–265, 2012.

[68] F. Alhomayani and M. H. Mahoor, "Deep learning methods for fingerprint-based indoor positioning: A review," *Journal of Location Based Services*, vol. 14, no. 3, pp. 129–200, 2020. eprint: https://doi.org/10.1080/17489725.2020.1817582. [Online]. Available: https://doi.org/10.1080/17489725.2020.1817582.

[69] 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, pp. 2825–2830, 2011.

[70] G. Lemaitre, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 559–563, 2017.

[71] [Online]. Available: https://github.com/alhomayani/Oversampling_BLE_fingerprints.

[72] *Location-based services (lbs) market statistics and forecast-2026*, Allied Market Research, Oct. 2019. [Online]. Available: https://www.alliedmarketresearch.com/location-based-services-market.

[73] J. Hopkins and J. Turner, *Go mobile: location-based marketing, apps, mobile optimized ad campaigns, 2D codes and other mobile strategies to grow your business*. John Wiley & Sons, 2012.

[74] A. Hammad and P. Faith, *Location based authentication*, US Patent 9,721,250, Aug. 2017.

[75] D. Leorke, "Location-based gaming apps and the commercialization of locative media," *Mobility and Locative Media: Mobile Communication in Hybrid Spaces*, vol. 132, 2014.

[76] Y. Zheng, "Location-based social networks: Users," in *Computing with spatial trajectories*, Springer, 2011, pp. 243–276.

[77] H. Huang, G. Gartner, J. M. Krisp, M. Raubal, and N. V. de Weghe, "Location based services: Ongoing evolution and research agenda," *Journal of Location Based Services*, vol. 12, no. 2, pp. 63–93, 2018. eprint: https://doi.org/10.1080/17489725.2018.1508763. [Online]. Available: https://doi.org/10.1080/17489725.2018.1508763.

[78] P. Ranacher, R. Brunauer, W. Trutschnig, S. Van der Spek, and S. Reich, "Why gps makes distances bigger than they are," *International Journal of Geographical Information Science*, vol. 30, no. 2, pp. 316–333, 2016.

[79] A. Carroll, G. Heiser, *et al.*, "An analysis of power consumption in a smartphone.," in *USENIX annual technical conference*, Boston, MA, vol. 14, 2010, pp. 21–21.

[80] J. Paek, K.-H. Kim, J. P. Singh, and R. Govindan, "Energy-efficient positioning for smartphones using cell-id sequence matching," in *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '11, Bethesda, Maryland, USA: Association for Computing Machinery, 2011, pp. 293–306, ISBN: 9781450306430. [Online]. Available: https://doi.org/10.1145/1999995.2000024.

[81] P. A. Zandbergen, "Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning," *Transactions in GIS*, vol. 13, pp. 5–25, 2009.

[82] Q. D. Vo and P. De, "A survey of fingerprint-based outdoor localization," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 491–506, 2015.

[83] J. Luo and H. Gao, "Deep belief networks for fingerprinting indoor localization using ultrawideband technology," *International Journal of Distributed Sensor Networks*, vol. 12, no. 1, p. 5 840 916, 2016.

[84] B. Wang, Q. Chen, L. T. Yang, and H.-C. Chao, "Indoor smartphone localization via fingerprint crowdsourcing: Challenges and approaches," *IEEE Wireless Communications*, vol. 23, no. 3, pp. 82–89, 2016.

[85] J. Torres-Sospedra, D. Rambla, R. Montoliu, O. Belmonte, and J. Huerta, "Ujiindoorloc-mag: A new database for magnetic field-based localization problems," in *Indoor Positioning and Indoor Navigation (IPIN), 2015 International Conference on*, IEEE, 2015, pp. 1–10.

[86] N. Moayeri, M. O. Ergin, F. Lemic, V. Handziski, and A. Wolisz, "Perfloc (part 1): An extensive data repository for development of smartphone indoor localization apps," in *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*, IEEE, 2016, pp. 1–7.

[87] A. Popleteev, "Ambiloc: A year-long dataset of fm, tv and gsm fingerprints for ambient indoor localization," in *8th International Conference on Indoor Positioning and Indoor Navigation (IPIN-2017)*, 2017.

[88] D. Hanley, A. B. Faustino, S. D. Zelman, D. A. Degenhardt, and T. Bretl, "Magpie: A dataset for indoor positioning with magnetic anomalies," in *Indoor Positioning and Indoor Navigation (IPIN), 2017 International Conference on*, IEEE, 2017, pp. 1–8.

[89] G. M. Mendoza-Silva, P. Richter, J. Torres-Sospedra, E. S. Lohan, and J. Huerta, "Long-term wifi fingerprinting dataset for research on robust indoor positioning," *Data*, vol. 3, no. 1, 2018.

[90] D. Byrne, M. Kozlowski, R. Santos-Rodriguez, R. Piechocki, and I. Craddock, "Residential wearable rssi and accelerometer measurements with detailed location annotations," *Scientific data*, vol. 5, p. 180 168, 2018.

[91] P. Baronti, P. Barsocchi, S. Chessa, F. Mavilia, and F. Palumbo, "Indoor bluetooth low energy dataset for localization, tracking, occupancy, and social interaction," *Sensors*, vol. 18, no. 12, 2018.

[92] D. Gubiani, P. Gallo, A. Viel, A. Dalla Torre, and A. Montanari, "A cellular network database for fingerprint positioning systems," in *New Trends in Databases and Information Systems*, T. Welzer, J. Eder, V. Podgorelec, R. Wrembel, M. Ivanović, J. Gamper, M. Morzy, T. Tzouramanis, J. Darmont, and A. Kamišalić Latifić, Eds., Cham: Springer International Publishing, 2019, pp. 111–119, ISBN: 978-3-030-30278-8.

[93] F. Alhomayani and M. H. Mahoor, "OutFin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach," *figshare*, Apr. 2020. [Online]. Available: https://figshare.com/articles/OutFin_a_multi-device_and_multi-modal_dataset_for_outdoor_localization_based_on_the_fingerprinting_approach/12069993.

[94] P. Vepakomma, C. Tonde, and A. Elgammal, "Supervised dimensionality reduction via distance correlation maximization," *Electron. J. Statist.*, vol. 12, no. 1, pp. 960–984, 2018. [Online]. Available: https://doi.org/10.1214/18-EJS1403.

[95] K. Kandasamy, W. Neiswanger, J. Schneider, B. Poczos, and E. P. Xing, "Neural architecture search with bayesian optimisation and optimal transport," in

Advances in Neural Information Processing Systems 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., Curran Associates, Inc., 2018, pp. 2016–2025. [Online]. Available: http://papers.nips.cc/paper/7472-neural-architecture-search-with-bayesian-optimisation-and-optimal-transport.pdf.

[96] T. Kudo and J. Miura, "Utilizing wifi signals for improving slam and person localization," in *2017 IEEE/SICE International Symposium on System Integration (SII)*, Dec. 2017, pp. 487–493.

[97] A. Viel, D. Gubiani, P. Gallo, A. Montanari, A. D. Torre, F. Pittino, and C. Marshall, "Map matching with sparse cellular fingerprint observations," in *2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*, Mar. 2018, pp. 1–10.

[98] *Wifi analyzer pro*, Webprovider. [Online]. Available: https://play.google.com/store/apps/details?id=info.wifianalyzer.pro&hl=en_US.

[99] *Bluetooth scanner*, Zoltán Pallagi. [Online]. Available: https://play.google.com/store/apps/details?id=com.pzolee.bluetoothscanner&hl=en_US.

[100] *Physics toolbox sensor suite pro*, Vieyra Software. [Online]. Available: https://play.google.com/store/apps/details?id=net.vieyrasoftware.physicstoolboxsuitepro&hl=en_US.

[101] *Qgis 3.10 a coruña*. [Online]. Available: https://qgis.org/en/site/.

[102] *Wi-fi scanning overview*, Android Developers. [Online]. Available: https://developer.android.com/guide/topics/connectivity/wifi-scan#wifi-scan-throttling.

[103] *Bluetoothadapter*, Android Developers. [Online]. Available: https://developer.android.com/reference/android/bluetooth/BluetoothAdapter#ACTION_DISCOVERY_STARTED.

[104] *Sensors overview*, Android Developers. [Online]. Available: https://developer.android.com/guide/topics/sensors/sensors_overview.

[105] M. Angermann, M. Frassl, M. Doniec, B. J. Julian, and P. Robertson, "Characterization of the indoor magnetic field for applications in localization and mapping," in *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Nov. 2012, pp. 1–9.

[106] *Assigned numbers for baseband*, Bluetooth Special Interest Group. [Online]. Available: https://www.bluetooth.com/specifications/assigned-numbers/baseband/.

[107] *Specification #: 36.213*, The 3rd Generation Partnership Project. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2427.

[108] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta, "Ujiindoorloc: A new multi-building and multi-floor database for wlan fingerprint-based indoor localization problems," in *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2014.

[109] P. Baronti, P. Barsocchi, S. Chessa, F. Mavilia, and F. Palumbo, "Indoor bluetooth low energy dataset for localization, tracking, occupancy, and social interaction," *Sensors*, vol. 18, no. 12, p. 4462, 2018.

[110] V. Sevindik, J. Wang, O. Bayat, and J. Weitzen, "Performance evaluation of a real long term evolution (lte) network," in *37th Annual IEEE Conference on Local Computer Networks - Workshops*, Oct. 2012, pp. 679–685.

[111] *Magnetic field calculators*, NOAA National Centers for Environmental Information. [Online]. Available: https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml?useFullSite=true.

[112] *Denver, co weather history (november 2019)*, Weather Underground. [Online]. Available: https://www.wunderground.com/history/monthly/us/co/denver/KDEN/date/2019-11.

[113] *Recommended light levels (illuminance) for outdoor and indoor venues*, National Optical Astronomy Observatory. [Online]. Available: https://www.noao.edu/education/QLTkit/ACTIVITY_Documents/Safety/LightLevels_outdoor+indoor.pdf.

[114] L. Chen, E. H. Wu, M. Jin, and G. Chen, "Homogeneous features utilization to address the device heterogeneity problem in fingerprint localization," *IEEE Sensors Journal*, vol. 14, no. 4, pp. 998–1005, 2014.

[115] *Past weather in denver, colorado, usa*, Time and Date AS. [Online]. Available: https://www.timeanddate.com/weather/usa/denver/historic.

[116] A. Solin, M. Kok, N. Wahlström, T. B. Schön, and S. Särkkä, "Modeling and interpolation of the ambient magnetic field by gaussian processes," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1112–1127, 2018.

[117] M. Nowicki and J. Wietrzykowski, "Low-effort place recognition with wifi fingerprints using deep learning," in *Automation 2017*, R. Szewczyk, C. Zieliński, and M. Kaliczyńska, Eds., Cham: Springer International Publishing, 2017.

[118] *Computational complexity of machine learning algorithms*, Apr. 2018. [Online]. Available: https://www.thekerneltrip.com/machine/learning/computational-complexity-learning-algorithms/.

[119] F. Alhomayani and M. Mahoor, *Deep learning-based symbolic indoor positioning using the serving enodeb*, 2020. arXiv: 2009.13675 [eess.SP].

[120] *Python 3.6.4*, Release Date: Dec. 19, 2017. [Online]. Available: https://www.python.org/downloads/release/python-364/.

[121] *Outfin*, GitHub. [Online]. Available: https://github.com/alhomayani/OutFin.

[122] *City of denver open data catalog*, licensed under the Creative Commons Attribution 3.0 (http://creativecommons.org/licenses/by/3.0/). [Online]. Available: http://data.denvergov.org.

[123] *Drcog regional data catalog*, licensed under the Creative Commons Attribution 3.0 (http://creativecommons.org/licenses/by/3.0/). [Online]. Available: https://data.drcog.org.

[124] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[125] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[126] Y. M. Assael, B. Shillingford, S. Whiteson, and N. de Freitas, "Lipnet: End-to-end sentence-level lipreading," *arXiv preprint arXiv:1611.01599*, 2016.

[127] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[128] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[129] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, Springer, 2010, pp. 177–186.

[130] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[131] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[132] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on Challenges in Representation Learning, ICML*, vol. 3, 2013, p. 2.

[133] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., 2012, pp. 2951–2959.

[134] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231216315533.

[135] 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, pp. 1929–1958, 2014.

[136] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[137] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[138] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[139] C. C. Aggarwal, *Neural networks and deep learning*. Springer, 2018.

[140] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[141] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: An astounding baseline for recognition," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, ser. CVPRW '14, IEEE Computer Society, 2014, pp. 512–519, ISBN: 978-1-4799-4308-1.

[142] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[143] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[144] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[145] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[146] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[147] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[148] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.

[149] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled generative adversarial networks," *arXiv preprint arXiv:1611.02163*, 2016.

[150] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[151] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[152] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[153] *Companies using TensorFlow*. [Online]. Available: https://www.tensorflow.org/.

[154] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[155] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS workshop*, 2011.

[156] *Caffe2 and PyTorch join forces to create a Research + Production platform PyTorch 1.0*. [Online]. Available: https://caffe2.ai/blog/2018/05/02/Caffe2_PyTorch_1_0.html.

[157] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, ACM, 2014, pp. 675–678.

[158] *Model Zoo*. [Online]. Available: https://github.com/BVLC/caffe/wiki/Model-Zoo.

[159] *Caffe2: A New Lightweight, Modular, and Scalable Deep Learning Framework*. [Online]. Available: https://caffe2.ai/.

[160] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv preprint arXiv:1512.01274*, 2015.

[161] *MXNet-Deep Learning Framework of Choice at AWS*. [Online]. Available: https://www.allthingsdistributed.com/2016/11/mxnet-default-framework-deep-learning-aws.html.

[162] K. Kaemarungsi and P. Krishnamurthy, "Analysis of wlan's received signal strength indication for indoor location fingerprinting," *Pervasive and Mobile Computing*, vol. 8, no. 2, pp. 292–316, 2012, Special Issue: Wide-Scale Vehicular Sensor Networks and Mobile Sensing.

[163] Y. Chapre, P. Mohapatra, S. Jha, and A. Seneviratne, "Received signal strength indicator and its analysis in a typical wlan system (short paper)," in *38th Annual IEEE Conference on Local Computer Networks*, Oct. 2013, pp. 304–307.

[164] Y. B. Bai, S. Wu, G. Retscher, A. Kealy, L. Holden, M. Tomko, A. Borriak, B. Hu, H. R. Wu, and K. Zhang, "A new method for improving wi-fi-based indoor positioning accuracy," *Journal of Location Based Services*, vol. 8, no. 3, pp. 135–147, 2014. eprint: https://doi.org/10.1080/17489725.2014.977362. [Online]. Available: https://doi.org/10.1080/17489725.2014.977362.

[165] V. Bahl and V. Padmanabhan, "Enhancements to the radar user location and tracking system," Tech. Rep. MSR-TR-2000-12, Feb. 2000, p. 13. [Online]. Available: https://www.microsoft.com/en-us/research/publication/enhancements-to-the-radar-user-location-and-tracking-system/.

[166] S. Sen, R. R. Choudhury, B. Radunovic, and T. Minka, "Precise indoor localization using phy layer information," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, ser. HotNets-X, Cambridge, Massachusetts: ACM, 2011, 18:1–18:6, ISBN: 978-1-4503-1059-8.

[167] J. Xiao, K. Wu, Y. Yi, and L. M. Ni, "Fifs: Fine-grained indoor fingerprinting system," in *2012 21st International Conference on Computer Communications and Networks (ICCCN)*, Jul. 2012, pp. 1–7.

[168] X. Wang, L. Gao, S. Mao, and S. Pandey, "Csi-based fingerprinting for indoor localization: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 763–776, Jan. 2017.

[169] *Core specifications — bluetooth technology website*. [Online]. Available: https://www.bluetooth.com/specifications/bluetooth-core-specification.

[170] *Bluetooth market update 2020 — bluetooth technology website*. [Online]. Available: https://www.bluetooth.com/bluetooth-resources/2020-bmu/.

[171] J. A. del Peral-Rosado, R. Raulefs, J. A. López-Salcedo, and G. Seco-Granados, "Survey of cellular mobile radio localization methods: From 1g to 5g," *IEEE Communications Surveys Tutorials*, vol. 20, no. 2, pp. 1124–1148, Secondquarter 2018.

[172] *Wireless e911 location accuracy requirements*, federal communications commission (FCC), Feb. 2015. [Online]. Available: https://ecfsapi.fcc.gov/file/60001025925.pdf.

[173] *Wireless e911 location accuracy requirements*, federal communications commission (FCC), Mar. 2019. [Online]. Available: https://docs.fcc.gov/public/attachments/FCC-19-20A1.pdf.

[174] M. Ibrahim and M. Youssef, "Cellsense: An accurate energy-efficient gsm positioning system," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 1, pp. 286–296, Jan. 2012.

[175] V. Otsason, A. Varshavsky, A. LaMarca, and E. de Lara, "Accurate gsm indoor localization," in *UbiComp 2005: Ubiquitous Computing*, M. Beigl, S. Intille, J. Rekimoto, and H. Tokuda, Eds., Springer Berlin Heidelberg, 2005, pp. 141–158, ISBN: 978-3-540-31941-2.

[176] *Rootmetrics coverage map for sprint network by communication technology*, rootmetrics. [Online]. Available: http://webcoveragemap.rootmetrics.com/en-US.

[177] *Understanding reference frames and device attitude*, Apple developers. [Online]. Available: https://developer.apple.com/documentation/coremotion/getting_processed_device-motion_data/understanding_reference_frames_and_device_attitude#2875084.

[178] H. J. Jang, J. M. Shin, and L. Choi, "Geomagnetic field based indoor localization using recurrent neural networks," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec. 2017, pp. 1–6.

[179] Y. Du, T. Arslan, and Q. Shen, "A dynamic feature fusion strategy for magnetic field and wi-fi based indoor positioning," in *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2019, pp. 1–7.

[180] S. Wang, H. Wen, R. Clark, and N. Trigoni, "Keyframe based large-scale indoor localisation using geomagnetic field and motion pattern," in *2016 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), Oct. 2016, pp. 1910–1917.

[181] G. Schroth, R. Huitl, D. Chen, M. Abu-Alqumsan, A. Al-Nuaimi, and E. Steinbach, "Mobile visual location recognition," *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 77–89, Jul. 2011.

[182] N. Piasco, D. Sidibé, C. Demonceaux, and V. Gouet-Brunet, "A survey on visual-based localization: On the benefit of heterogeneous data," *Pattern Recognition*, vol. 74, pp. 90–109, 2018.

[183] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, Feb. 2016.

[184] H. Kawaji, K. Hatada, T. Yamasaki, and K. Aizawa, "Image-based indoor positioning system: Fast image matching using omnidirectional panoramic images," in *Proceedings of the 1st ACM International Workshop on Multimodal Pervasive Video Analysis*, ser. MPVA '10, Firenze, Italy: ACM, 2010, pp. 1–4, ISBN: 978-1-4503-0167-1.

[185] M. Werner, M. Kessel, and C. Marouane, "Indoor positioning using smartphone camera," in *2011 International Conference on Indoor Positioning and Indoor Navigation*, Sep. 2011, pp. 1–6.

[186] X. Li and J. Wang, "Image matching techniques for vision-based indoor navigation systems: A 3d map-based approach1," *Journal of Location Based Services*, vol. 8, no. 1, pp. 3–17, 2014. eprint: https://doi.org/10.1080/17489725.2013.837201. [Online]. Available: https://doi.org/10.1080/17489725.2013.837201.

[187] K. Guan, L. Ma, X. Tan, and S. Guo, "Vision-based indoor localization approach based on surf and landmark," in *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Sep. 2016, pp. 655–659.

[188] X. Qu, B. Soheilian, E. Habets, and N. Paparoditis, "Evaluation of sift and surf for vision based localization.," *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 41, 2016.

[189] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, pp. 2938–2946.

[190] M. Quigley, D. Stavens, A. Coates, and S. Thrun, "Sub-meter indoor localization in unmodified environments with inexpensive sensors," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 2039–2046.

[191] S. He and S.-H. G. Chan, "Wi-fi fingerprint-based indoor positioning: Recent advances and comparisons," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 466–490, 2016.

[192] S. L. Lau and K. David, "Movement recognition using the accelerometer in smartphones," in *2010 Future Network Mobile Summit*, Jun. 2010, pp. 1–9.

[193] E. Miluzzo, M. Papandrea, N. D. Lane, H. Lu, and A. T. Campbell, "Pocket, bag, hand, etc.-automatically detecting phone context through discovery," *Proc. PhoneSense 2010*, pp. 21–25, 2010.

[194] *Revision of part 15 of the commission's rules regarding ultra-wideband transmission systems*, federal communications commission (FCC), Apr. 2002. [Online]. Available: https://transition.fcc.gov/Bureaus/Engineering_Technology/Orders/2002/fcc02048.pdf.

[195] D. Geer, "Uwb standardization effort ends in controversy," *Computer*, vol. 39, no. 7, pp. 13–16, Jul. 2006.

[196] H. Jing, L. K. Bonenberg, J. Pinchin, C. Hill, and T. Moore, "Detection of uwb ranging measurement quality for collaborative indoor positioning," *Journal of Location Based Services*, vol. 9, no. 4, pp. 296–319, 2015. eprint: https://doi.org/10.1080/17489725.2015.1120359. [Online]. Available: https://doi.org/10.1080/17489725.2015.1120359.

[197] B. S. Çiftler, A. Kadri, and I. Güvenc, "Iot localization for bistatic passive uhf rfid systems with 3-d radiation pattern," *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 905–916, 2017.

[198] K. Liu, X. Liu, and X. Li, "Guoguo: Enabling fine-grained indoor localization via smartphone," in *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '13, Taipei, Taiwan: ACM, 2013, pp. 235–248, ISBN: 978-1-4503-1672-9.

[199] H.-P. Tan, R. Diamant, W. K. Seah, and M. Waldmeyer, "A survey of techniques and challenges in underwater localization," *Ocean Engineering*, vol. 38, no. 14, pp. 1663–1676, 2011.

[200] A. Cramariuc and E. Lohan, *Open-access wifi measurement data and python-based data analysis*, 2016. [Online]. Available: http://www.cs.tut.fi/tlt/pos/meas.htm.

[201] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization using lstms for structured feature correlation," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 627–637.

[202] A. Belmonte-Hernández, G. Hernández-Peñaloza, D. Martín Gutiérrez, and F. Álvarez, "Swiblux: Multi-sensor deep learning fingerprint for precise real-time indoor tracking," *IEEE Sensors Journal*, vol. 19, no. 9, pp. 3473–3486, May 2019.

[203] A. Zanella, "Best practice in rss measurements and ranging," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2662–2686, 2016.

[204] R. Roberto, J. P. Lima, T. Araújo, and V. Teichrieb, "Evaluation of motion tracking and depth sensing accuracy of the tango tablet," in *Mixed and Augmented Reality (ISMAR-Adjunct), 2016 IEEE International Symposium on*, IEEE, 2016, pp. 231–234.

[205] N. Moayeri, C. Li, and L. Shi, "Perfloc (part 2): Performance evaluation of the smartphone indoor localization apps," in *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2018, pp. 1–8.

[206] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2013.

[207] C. Löffler, S. Riechel, J. Fischer, and C. Mutschler, "Evaluation criteria for inside-out indoor positioning systems based on machine learning," in *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2018, pp. 1–8.

[208] *ISO/IEC 18305:2016, Information technology – Real time locating systems, Test and evaluation of localization and tracking systems*. [Online]. Available: https://www.iso.org/standard/62090.html.

APPENDIX A
DEEP LEARNING PRELIMINARIES

Deep learning, also known as deep neural networks, is a class of machine learning algorithms that seeks to imitate the learning process of humans by attempting to model the structure and functionality of the human brain. The success of deep learning can be attributed to its data-driven feature learning and ability to model extremely complex functions. This is what has empowered its models to not only outperform shallow learning algorithms but also surpass human ability in many tasks [124, 125, 126].

Deep learning models can automatically discover multiple levels of representations after being fed raw data [127]. This is accomplished through a multilayer stack of simple, but non-linear, processing units called *artificial neurons*. In mathematical terms, an artificial neuron performs a weighted sum over its inputs, adds a bias term, and feeds the result to an *activation function* before it is passed to the neurons in the next layer, as shown in Figure A.1. The bias term increases the neuron's flexibility by allowing it to shift the result horizontally, while the activation function introduces non-linearity to allow the neuron to model non-linear behavior. Commonly used activation functions are illustrated in Figure A.2.

Depending on layer types and how layers are organized within a network, various deep learning architectures can be formed. The most effective architectures include

Fully Connected Networks, Deep Belief Networks, Autoencoder Networks, Convolutional Neural Networks, Recurrent Neural Networks, and Generative Adversarial Networks. Before delving into the specifics of each architecture, the next few subsections will present some fundamental concepts that are common to all architectures. Afterward, the most popular software frameworks that are used to build these architectures are discussed.

A.1 The Principals of Learning

For the sake of better understanding, principals of learning are explained in the context of training a 5-layer FC network to predict the correct class label as shown in Figure A.3. An FC network is a *feedforward* neural network in which the output of one layer is the input for the next layer. Each neuron in an FC network is fully connected to the neurons in the next layer. In theory, one could force the network to output the correct class label if the right set of *parameters* (i.e., the real-valued weights and biases of the network) were obtained. In practice, this is accomplished through an iterative process called *training*, the goal of which is to minimize the distance between the network's output $\vec{\hat{y}}$ and the desired output $\vec{y}$. In other words, the purpose of training is to obtain a set of parameters that corresponds to the lowest error (or loss) possible, as given by the *loss function* $L(\vec{y}, \vec{\hat{y}})$.

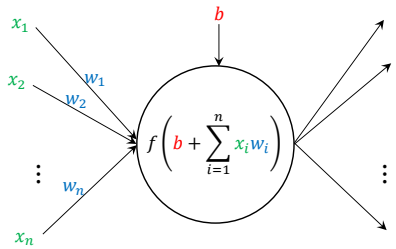

Figure A.1: The computational model of an artificial neuron. x_i, w_i, b, and $f(\cdot)$ are the inputs, weights, bias, and activation function, respectively.

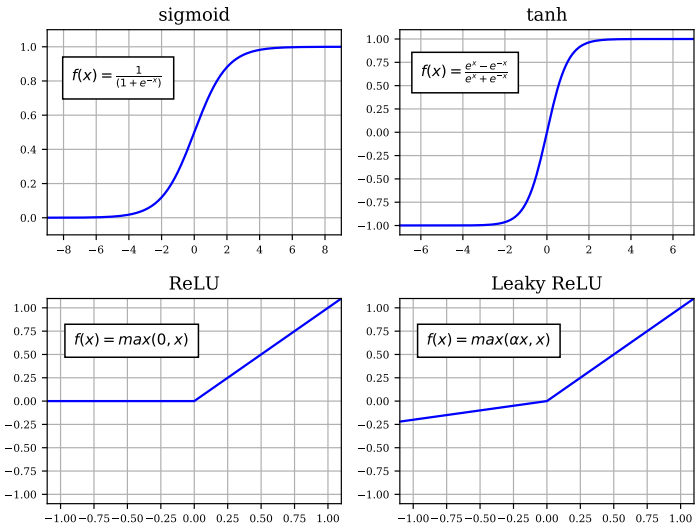

Figure A.2: Equations and the corresponding plots of the sigmoid, hyperbolic tangent (tanh), ReLU, and Leaky ReLU activation functions. α is a positive constant less than 1.

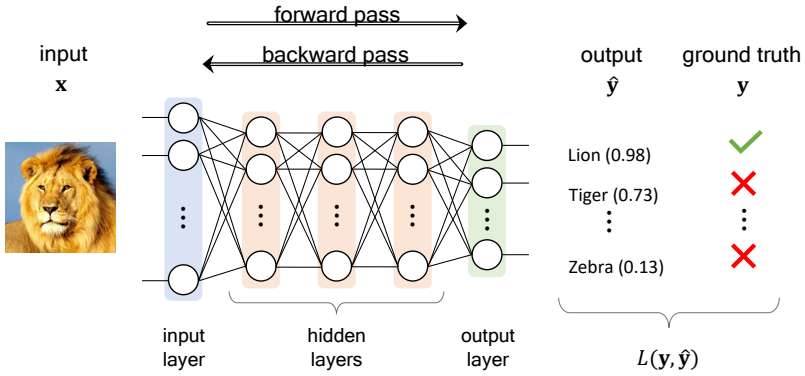

Figure A.3: An FC network with three hidden layers and the two passes of training. The bias terms have been omitted for simplicity.

Training involves two passes: a *forward pass* and a *backward pass*. During the forward pass, the network's parameters are initially set close to zero. Then, the network's *input layer* is fed a fixed-size input $\vec{x}$. The input layer's number of neurons is equivalent to the vector's dimension (e.g., the number of pixels in the input image). Each neuron in the first *hidden layer* performs the computation described in Figure A.1 and passes its activation to the neurons in the second hidden layer.

The process of computing activations and passing them to the next hidden layer continues until the activations are computed at the *output layer*. The output layer's number of neurons is equivalent to the number of classes and its activations represent *class scores*. Since the network's parameters were initialized in a random fashion, it is expected that the network's output is inconsistent with the desired output. This expectation is reflected by the error obtained from the loss function. The aim now becomes minimizing this error by tuning or adjusting the network's parameters. This is achieved during the backward pass by using a central deep learning algorithm called *backpropagation* [128].

Backpropagation is based on the idea that, by calculating the gradient vector of the loss function with respect to networks parameters and then moving the parameters in the opposite direction to the gradient, the loss is minimized. This requires that the error be backpropagated from the output layer up to the input layer by applying the chain rule of derivatives. Once the gradient is computed, it is fed to an *optimization algorithm* that performs *gradient descent* (i.e., moving the network's parameters in the direction that minimizes the error). The step size in that direction is controlled by the algorithm's *learning rate* η. Commonly used optimization algorithms include Stochastic Gradient Descent (SGD) [129], Momentum [130], and Adam [131].

The two training passes are then repeated over each remaining training instance in the *training set*. However, updating a network's parameters based on single training instances is computationally expensive; instead, parameters are updated based on training set *batches*, where the average error generated by the instances in a batch is backpropagated. If the network has seen all training batches, it is said to have completed a training *epoch*. It is often the case that training is repeated over several epochs before the error converges. After training is complete, network performance is evaluated using a separate set of instances called the *testing set*. The objective is

to obtain an estimate of the network's *generalization* ability by testing the network against instances that it has never seen before.

A.2 Learning Approaches

The learning approach described above falls under the category of *supervised learning*. Supervised learning (or teacher-based learning) is the most common form of learning in deep learning. In supervised learning, all training instances are labeled; hence, the output produced by the network can be judged as correct or incorrect. Supervised models are trained to either predict class labels (*classification*) or continuous quantities (*regression*). In contrast, *unsupervised learning* is used when training instances do not have any labels. The goal of unsupervised learning is to discover interesting patterns in the data such as clusters, associations, or anomalies. *Semi-supervised learning* falls between the two approaches where only a small subset of instances is labeled and the rest are not. This approach is typically used in *pseudo labeling* [132] where a network is trained on the labeled instances to produce labels for the unlabeled instances. Finally, *reinforcement learning* is used to enable a network to produce the right action (or a series of right actions) inside a dynamic environment by providing the network with feedback by using rewards and punishments.

A.3 Parameters vs. Hyperparameters

While a network's parameters are explicitly learned through backpropagation, there is another set of parameters, called *hyperparameters*, that cannot be learned. Unlike parameters, hyperparameters are design choices and configurations that remain fixed during training. For example, the number of hidden layers and the number of neurons in each layer, the activation and loss functions used, the batch size and

number of training epochs, and the learning rate are all considered hyperparameters. Properly setting these hyperparameters is crucial since they govern the training process. Yet, there is no efficient method that can optimize all the hyperparameters at once. Therefore, most practitioners often resort to *manual*, *random*, *grid search*, or *Bayesian* optimization [133].

A.4 General Guidelines for Training

- When training deep learning models, it is advised to use modern activation functions instead of the traditional sigmoid or tanh functions. Modern activation functions, such as ReLU, expedite learning and alleviate the *vanishing/exploding gradient* problems [33] where gradients either rapidly shrink or grow out of bounds as they are backpropagated through the network.

- Deep learning involves learning hundreds of thousands, or sometimes even millions, of parameters. Therefore, its models are often prone to *overfitting* (i.e., memorizing instead of learning), especially when the training set is not large and diversified enough [134]. One approach that can be used to combat overfitting is using *data augmentation* techniques that allow for the creation of additional training data by reasonably modifying the existing training data. Another approach is to use *dropout* [135]. Dropout is a technique that omits the activations of randomly selected neurons for an ongoing epoch, thereby tuning only a subset of parameters in each epoch rather than all parameters. Using a regularization hyperparameter in a loss function can also curb overfitting.

- In parallel computing terms, training deep learning models is considered an *embarrassingly parallel* problem (i.e., it takes little to no effort to parallelize the computations (matrix-vector multiplication) performed during training).

Therefore, using GPUs can significantly speed up the training process, sometimes by a factor of 50 or more [136].

- The speed, performance, and stability of deep learning models can be improved by employing *batch normalization*. Batch normalization is a technique that normalizes hidden layer inputs to combat the *internal covariate shift* problem, a result of the constantly changing distribution of each layer's inputs during training [137].

A.5 Deep Learning Architectures

This subsection provides a brief overview of the deep learning architectures that have been employed for fingerprint-based indoor positioning. These are Convolutional Neural Networks, Recurrent Neural Networks, Generative Adversarial Networks, Autoencoder Networks, and Deep Belief Networks. Readers looking for more details about these architectures may refer to [138] or [139].

A.5.1 Convolutional Neural Networks

A CNN is a feedforward neural network that was first designed to solve the problems of shift, scale, and distortion variance when classifying high dimensional patterns

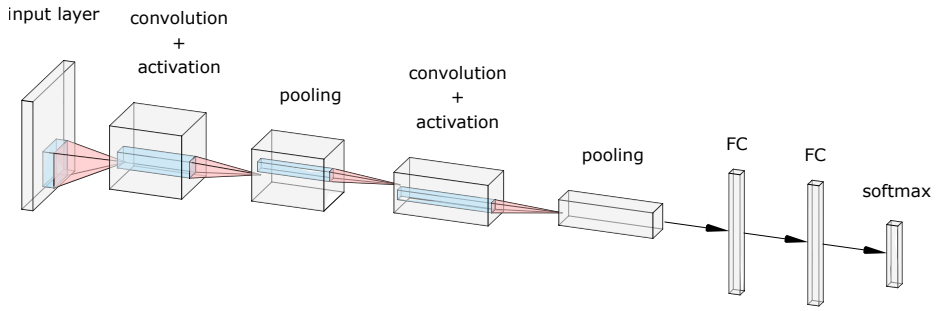

Figure A.4: The structure of a basic CNN.

such as handwritten characters [140]. The architecture is based on *local connectivity* and *weight sharing*, meaning that rich features are extracted in a hierarchical fashion and the number of parameters in CNNs is reduced significantly when compared to FC networks. Pre-trained CNNs are powerful tools for extracting the generic features of images [141] regardless of the application domain. A typical CNN (Figure A.4) consists of a combination of *convolutional* and *pooling* layers followed by FC layers and a *softmax* layer [33]. When a CNN receives an input (a 2D or 3D array), it performs a convolutional operation by sliding several fixed-size *kernels* with predefined horizontal and vertical *strides* over the input to produce *feature maps* (i.e., a number of 2D arrays that changes depending on the number of kernels). This convolution operation is a dot product between the kernel's weights and the input. The feature maps are then passed to a pooling layer that performs a sub-sampling operation to reduce the map's dimensionality and make the network more robust to scale, rotation, and position variance. Note that the number of kernels, their size, the stride lengths, and the type of pooling performed are hyperparameters that need to be set beforehand. The final feature maps (i.e., the results of performing multiple subsequent convolution and pooling operations) are then flattened into a 1D vector to be used by the FC layers for classification. A softmax layer, with neurons corresponding to the total number of classes, is typically added after the FC layers. This allows the network to output class probabilities instead of class scores.

A.5.2 Recurrent Neural Networks

A RNN is a popular deep learning architecture for dealing with sequential and time-series data. The output of the network is a function of the current input at time t in addition to the previous inputs at time $\tilde{t} < t$. RNNs are flexible and dynamic architectures that have been used to process various data types with variable-

length input/output sequences. A typical RNN consists of a core cell with hidden *internal states* as shown on the left in Figure A.5. Upon receiving a new input, the internal states are updated according to a *recurrence formula* and fed back to the cell. Unrolling the RNN in time yields its computational graph shown on the right in Figure A.5. The graph resembles a feedforward neural network over time, where its parameters are shared across all layers. Bidirectional RNNs [142] are extensions of RNNs that, in some scenarios, can produce better results. They are created by using two RNNs—one of a past-to-future time order and another of a future-to-past time order—where the outputs of the two networks are combining at each time step. In many cases, it is desirable to model long-term dependencies that require stacking multiple core cells at each time step or that deal with very long sequences (thousands of time steps). In such cases, a conventional RNN will fail to capture the dependencies due to the vanishing/exploding gradient problem [143]. To alleviate this problem, the RNN can be equipped with an explicit memory unit, such as the Long Short-Term Memory (LSTM) unit [144] or the Gated Recurrent Unit (GRU) [145]. These units implement a *gating mechanism* that allows for a better gradient flow and the learning of long-term dependencies.

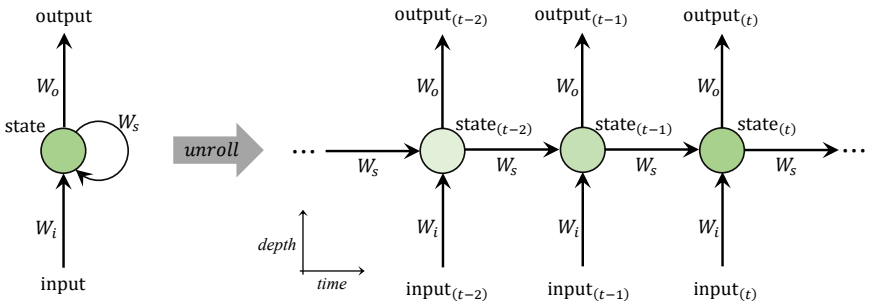

Figure A.5: An RNN with a single core cell (left) and its unrolled version (right).

A.5.3 Generative Adversarial Networks

A GAN is an emerging deep learning architecture that was first introduced by Goodfellow *et al.* in 2014 [146]. GANs are used to generate high-quality synthetic data from existing authentic data. As shown in Figure A.6, a GAN consists of two neural networks, namely a *generator* G and a *discriminator* D. During training, the idea is to have the two networks compete against each other with each network trying to maximize its own objective function. More specifically, the objective of D is to strengthen its ability to distinguish between authentic data (coming from the training set) and synthetic data (generated by G) while the objective of G is to strengthen its ability to produce synthetic samples that are able to mislead D into classifying them as authentic samples. Backpropagation is used to tune the parameters of both networks by training one network at a time until the generated samples become indistinguishable from the training set. However, optimizing two objective functions simultaneously makes training GANs a challenging task. For example, some of the issues that may arise during training are *non-convergence* and *mode collapse* (i.e., G produces samples that are limited to a subspace of the training set). Overcoming these problems is still an active area of research [147, 148, 149].

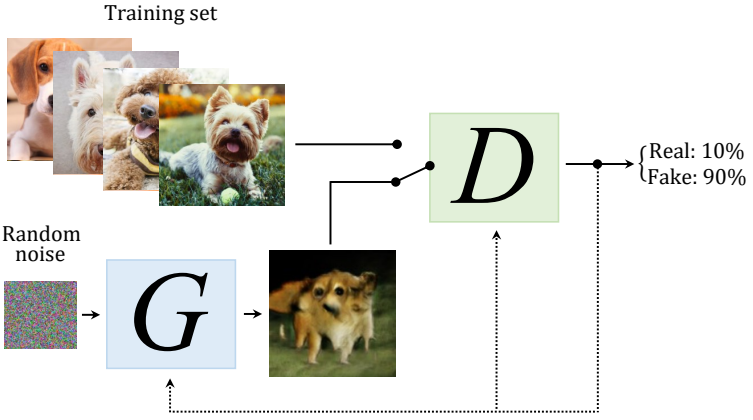

Figure A.6: The working principal of a GAN.

A.5.4 Autoencoder Networks

Autoencoder Networks, or AEs, are a family of feedforward neural networks that have been used in unsupervised learning tasks. AEs have the same number of neurons in the input layer as the output layer. A typical AE is trained to reconstruct an input without memorizing or directly copying it. Instead, an *encoder-decoder* approach is used as seen in Figure A.7. This hourglass-shaped architecture forces the network to encode (compress) the input into a *latent code* from which the input can be decoded (reconstructed). Backpropagation is used to learn the network's parameters by minimizing a *reconstruction loss* between the input and the reconstructed input. One common variant of AEs are DAEs [150]. DAEs are trained to reconstruct an input from a corrupted version of it (Fig. Figure A.7). Another common variant of AEs are VAEs [60]. VAEs not only learn useful representations of the data but also learn the distribution's statistical parameters (mean and variance). This allows for new data generation.

A.5.5 Deep Belief Networks

In 2006, the introduction of DBNs by Hinton *et al.* marked the beginning of the deep learning era [151]. DBNs are neural networks that use unsupervised learning to

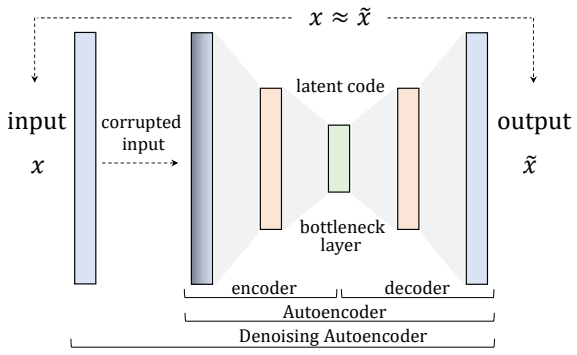

Figure A.7: The architecture of an AE and a DAE.

facilitate supervised learning. DBNs are formed by stacking several RBMs as shown in Figure A.8. Each RBM is a two-layer network that consists of both *visible* and *hidden* units [152]. RBMs are based on Boltzmann Machiness (BMs) that model the interaction between these units using *energy functions*. However, unlike BMs, the visible units in RBMs are only connected to the hidden units and vice versa. The training of DBNs involves two steps: unsupervised *pre-training* followed by supervised *fine-tuning*. The goal of pre-training is to initialize the network's parameters based on the underlying structure of the data. This makes the network less susceptible to overfitting, especially for small datasets [127]. Pre-training is accomplished using the *greedy algorithm* which trains the network one layer at a time, starting with the first visible layer and moving up to the last hidden layer [151]. After pre-training, the network's parameters are further optimized through fine-tuning. Fine-tuning is performed by training the network using labeled data with respect to a supervised training criterion.

A.6 Deep Learning Software Frameworks

Since the popularity of deep learning has surged in recent years, several open-source deep learning frameworks have been proposed by both academia and indus-

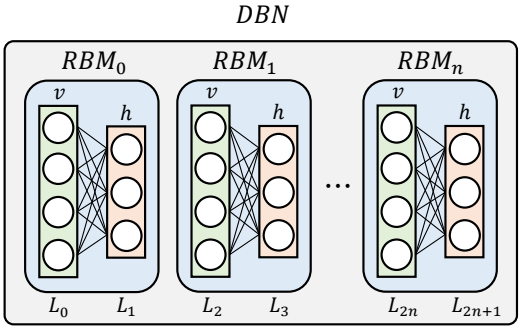

Figure A.8: Several RBMs are stacked to form a DBN.

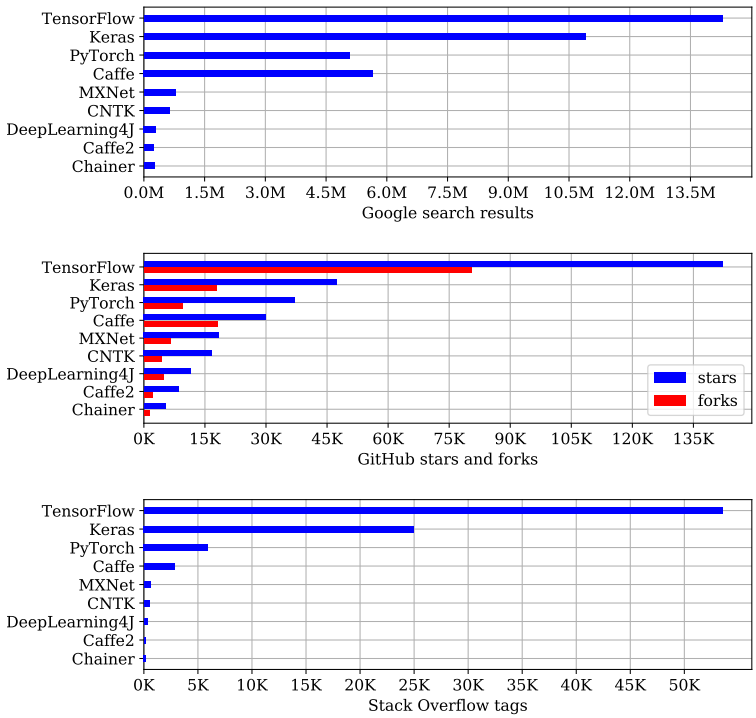

Figure A.9: The nine most popular deep learning frameworks based on the total number of: Google search results, GitHub stars and forks, and Stack Overflow tags. Data collected on Mar. 21, 2020.

try. These frameworks offer high-level programming interfaces that serve as building blocks for designing, training, validating, and deploying deep learning models. This subsection provides an overview of the five most popular frameworks as seen in Figure A.9. Table A.1 compares the different aspects of these frameworks.

A.6.1 TensorFlow

After its release in late 2015 [30], TensorFlow quickly became the most popular deep learning framework. Besides the various Google products that utilize it, TensorFlow has been adopted by other companies such as Intel, AIRBUS, Twitter, and Uber [153]. TensorFlow was initially developed by the Google Brain team and is based on data flow graphs in which the graph's nodes represent operations and the edges

Table A.1: A comparison of the leading open-source deep learning frameworks

Framework	Initial release date	Originally developed by	Backed by	Core language	Available APIs	Highlights
TensorFlow	November 2015	Google Brain Team	Google	C++	Python, JavaScript, C++, Java, Go, Swift (early release)	Provides visualization of the training process through TensorBoard; Supported by a large community of developers; TensorFlow Lite is the most complete solution for mobile and embedded systems to date.
Keras	March 2015	François Chollet	Google	Python	Python, R	Provides a simplistic and intuitive interface that makes implementing complex models straightforward; Easier to debug the code.
Caffe	March 2014	BAIR	UC Berkeley	C++	Python, MATLAB	Offers low training and inference latencies especially for CNNs; Smooth switching between platforms; Dozens of pre-trained models available online.
PyTorch	October 2016	FAIR	Facebook	Python	Python, C++ (unstable)	Allows for networks modification at runtime; Deeply integrated into Python which make coding as simple and flexible as in Python.
MXNet	December 2015	researchers from several universities	Amazon	C++	Python, C++, R, Java, Gluon, Perl, Scala, Clojure, Julia	Highly scalable which makes it suifor multi-GPU and cloud implementations; Provides interfaces for most mainstream programming languages.

connecting the nodes represent multi-dimensional data arrays (tensors). TensorFlow has APIs available in Python, JavaScript, C++, Java, and Go for constructing and executing these graphs. TensorFlow can be easily deployed across multiple CPUs, GPUs, and Tensor Processing Units (TPUs). In addition, the recent release of TensorFlow Lite enabled on-device inference for mobile and embedded systems. APIs for Android, iOS, and Raspberry Pi are currently available.

A.6.2 Keras

Keras [49] is a popular, high-level Python library for implementing deep learning models. It is not a framework on its own but rather a front-end API that integrates with many deep learning frameworks such as TensorFlow, MXNet, Microsoft Cognitive Toolkit (CNTK), and DeepLearning4J. Keras is well-documented and was developed with a focus on fast prototyping and ease-of-use. Keras minimizes users' required actions for common use cases and provides clear feedback about users' errors. Due to the simplified interface that allows for building deep learning models using just a few lines of code, TensorFlow has recently incorporated Keras as part of

its core API. Moreover, iOS provides official support for Keras through Apple's Core ML framework.

A.6.3 PyTorch

PyTorch [154], developed by Facebook's AI Research (FAIR), is a fast-growing deep learning framework for Python. PyTorch is based on Torch, a MATLAB-like scientific computing framework that uses the LuaJIT scripting language with an underlying C/CUDA implementation [155]. PyTorch is well-suited for research purposes given its high flexibility and usability. Unlike Torch, PyTorch does not implement deep learning models in containers which makes the development process more transparent to the user. Additionally, PyTorch uses reverse-mode auto-differentiation which allows for dynamic network changes on-the-fly. In May 2018, Facebook announced that it would merge Caffe2 into PyTorch to create a unified research-to-production platform named PyTorch 1.0 [156].

A.6.4 Caffe

Caffe (Convolutional Architecture for Fast Feature Embedding) was originally developed by the Berkeley Artificial Intelligence Research lab (BAIR) in 2014 [157]. It is written in C++ and uses NVIDIA's CUDA to provide support for GPU computations, with bindings available for Python and MATLAB. Caffe separates model representation from implementation (i.e., models are configured rather than hard-coded), which allows for seamless switching between heterogeneous platforms (e.g., CPU to GPU or the cloud). Caffe provides dozens of pre-trained models that can be downloaded through the Model Zoo platform [158]. In April 2017, Facebook released Caffe2 which is a more lightweight, modular, and scalable version of Caffe [159]. Caffe2 integrates with Visual Studio, Android Studio, and Xcode for mobile app development.

A.6.5 MXNet

MXNet is a promising deep learning framework that is currently incubated by the Apache Software Foundation. It was first created in collaboration with researchers from several universities [160]. MXNet combines the power of declarative programming with imperative programming to maximize efficiency and flexibility. It provides interfaces for a plethora of programming languages including Python, C++, R, Java, Gluon, Perl, Scala, Clojure, and Julia. Moreover, its high scalability, where speedups scale almost linearly with the number of added GPUs, led Amazon to select it as its deep learning framework-of-choice for Amazon Web Services [161].

APPENDIX B
INDOOR FINGERPRINT TYPES

This appendix provides an overview of different fingerprint types that are used for indoor positioning. For each fingerprint type, its advantages and disadvantages for indoor positioning are discussed first, followed by a brief account of the first documented time of using it for indoor positioning. The fingerprint types include Radio Frequency (WiFi, BLE, and Cellular), Magnetic Field and IMU, Image, Hybrid, and Miscellaneous Ultra-Wide Band (UWB), Visible Light, RFID, and Acoustic).

B.1 Radio Frequency Fingerprints

B.1.1 WiFi Fingerprints

The family of IEEE 802.11 Wireless Local Area Network (WLAN) standards, commonly known as WiFi, operate in two unlicensed bands: the 2.4 GHz and 5 GHz bands. WiFi was designed to provide high-speed wireless networking and Internet connectivity; thus, it is optimized for communication rather than localization. Nevertheless, using WiFi for localization is a natural choice because of its widespread adoption in user devices and the ubiquity of WiFi APs. Moreover, no additional infrastructure is required to realize localization, making WiFi fingerprinting a cost-effective solution.

WiFi fingerprints are formed by extracting RSS values from all visible APs in an environment. Thus, one drawback of WiFi fingerprinting is the time it takes to complete a scanning cycle. Depending on hardware/software limitations, this process can take several seconds [162]. This becomes problematic when the user is moving. Movement may lead to smearing the fingerprint across space [18]. Another drawback of using WiFi fingerprints is associated signal interference. Many indoor appliances such as microwave ovens, cordless phones, and wireless baby monitors operate in the same bands as WiFi. This often leads to high variability in RSS measurements, even when recorded at the same location [162, 163, 164].

In 2000, Microsoft Research proposed *RADAR* [13], a system widely known as the first WiFi fingerprinting system. The system collects RSS measurements at the AP side instead of the user side; thus, it is a tracking system. The kNN algorithm, with a Euclidean distance similarity metric, is used to compute a user's position. RADAR designers demonstrated that a user's orientation, the value of k, and the number of samples in the offline and online phases affect localization accuracy. The superiority of fingerprinting over lateration was also demonstrated. Fingerprinting achieved a median localization error of 2.94 m compared to 4.3 m achieved by lateration. Later, a Viterbi-like algorithm was proposed to enhance the system's tracking ability [165]. The median error was reduced to 2.37 m.

Currently, there is a trend in exploiting richer information enabled by Orthogonal Frequency-Division Multiplexing (OFDM) through CSI. CSI includes the amplitude and phase of each subcarrier from each antenna. CSI is a function of the combined effect of multipath, shadowing, power decay, and fading on a signal propagating from a transmitter to receiver. Since many subcarriers are available for each antenna, positioning using a single AP is feasible [166, 167]. Moreover, CSI values have proven to be more stable than RSS values as demonstrated in Figure B.1. However, the main

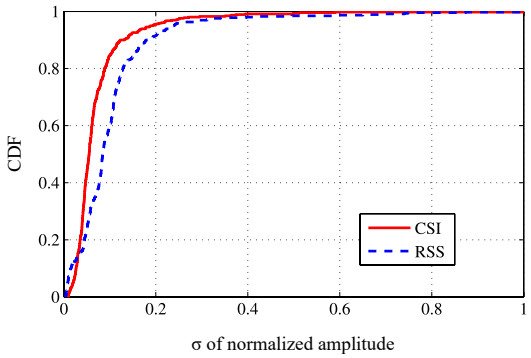

Figure B.1: Cumulative Distribution Function (CDF) of the standard deviations of CSI and RSS amplitudes for 150 locations using 50 measurements at each location. Figure reproduced from [168].

drawback of using CSI for fingerprinting is that most Wireless Network Interface Card (WNIC) do not provide means for conveniently extracting CSI values. Impractical solutions, such as hacking into device drivers, are commonly followed for data collection. At the time of writing, no implementation that uses a smartphone to collect CSI data exists.

B.1.2 BLE Fingerprints

BLE, also known as Bluetooth Smart or Bluetooth 4.0, is a popular wireless technology for low-power, Machine-to-Machine (M2M) communication. It has 40, 2 MHz wide channels that operate in the same 2.4 GHz radio band as WiFi [169]. Since the Bluetooth Special Interest Group introduced it in 2010, it has received widespread adoption with over 800 million BLE-enabled devices shipped in 2019 alone [170]. One of the main driving forces behind its popularity are BLE beacons. BLE beacons are small, inexpensive, and portable (battery-powered) transmitters that are used in a multitude of applications, including indoor positioning. Some beacons allow for the adjustment of transmission parameters such as transmission frequency, power, and bit rate. Beacons use three widely spaced channels to broadcast advertising messages

that contain the beacon's Universally Unique Identifier (UUID) and its transmission power in dBm. These messages are used by proximity-based positioning systems to provide positioning and navigation services. [64]. Two widely used industry protocols for BLE include Apple's iBeacon and Google's Eddystone.

Regarding fingerprinting, Faragher and Harle [18] investigated the feasibility of using BLE fingerprints for fine-grained indoor positioning. They conducted extensive experiments from which they reported several findings. First, the power draw on smartphones is much lower for BLE than WiFi. Second, BLE has a much higher scan rate than WiFi which makes BLE more suitable for user navigation and tracking applications. Third, if enough BLE beacons are strategically deployed in an environment, then the positioning accuracy could easily surpass that obtained by the existing WiFi infrastructure. However, BLE signals are more vulnerable to channel gain and fast fading than WiFi signals. As a result, BLE measurements fluctuate severely over time. The use of three channels (compared to one in WiFi) exacerbates this problem due to the wide spacing between these channels. Additionally, monitoring the battery level of the deployed BLE beacons to ensure uninterrupted services is still a major challenge [64]. Table B.1 compares some of the technical specifications of a typical WiFi AP and BLE beacon.

Table B.1: WiFi AP vs. BLE beacon

	WiFi AP[†]	BLE beacon[‡]
Battery powered	No	Yes
Max. power consumption (W)	12.7	0.01
Max. transmit power (dBm)	20	0
Max. range (m)	250	50
Weight (kg)	1.020	0.047
Cost ($)	≈ 100.00	≈ 30.00

[†]TP-Link EAP245 AP [‡]Aruba LS-BT20 beacon

B.1.3 Cellular Fingerprints

The use of cellular-based indoor positioning has primarily been motivated by the E-911 regulation imposed by the U.S. Federal Communications Commission (FCC) [171]. The most recent regulation mandates require cellular network operators to provide emergency call positioning within a 50 m horizontal accuracy [172] and 3 m vertical accuracy [173]. Due to the lack of access to proprietary cellular data, such as time and angle measurements, most academic solutions to cellular indoor positioning are either fingerprinting- or triangulation-based [1].

From a fingerprinting perspective, cellular-based fingerprinting has several advantages over WiFi/BLE fingerprinting. First, unlike WiFi and BLE, cellular signals operate in licensed bands which means they are less prone to interference. Second, not every cellphone necessarily supports WiFi/BLE; however, every cellphone, by definition, comes equipped with a cellular modem. Third, the typical coverage of cellular BSs ranges from hundreds of meters to tens of kilometers which is orders of magnitude greater than WiFi APs/BLE beacons. Fourth, there is no deployment cost associated with using cellular signals for fingerprinting since BSs are deployed and maintained outside the localization environment. Nonetheless, cellular fingerprinting has its drawbacks: First, cellular signals are not designed to penetrate deep inside buildings, often resulting in blind spots due to the shadowing effect. Second, BSs are often deployed on macro-cell layouts (Figure B.2) in which the overlap between the coverage area of neighboring BSs is kept to a minimum [171], resulting in few fingerprints for any given area. Third, standard-compliant modems can only report the RSS measurements from up to seven BSs [174], limiting the number of measured fingerprints to seven at any given time.

Historically, the first to exploit cellular RSS fingerprints for indoor positioning was Otsason *et al.* in 2005 [175]. They used a special modem that provided RSS measurements from up to 35 2G BSs. Experimental results conducted in three buildings demonstrated a median positioning error ranging from 2.48 m to 5.44 m using the kNN algorithm.

B.2 Magnetic Field and IMU Fingerprints

The complex distortions of Earth's magnetic field, caused by steel structures and reinforced concrete, form unique spatial signatures that can be used to construct magnetic maps of indoor environments. These signatures have been experimentally proven to be very stable over long periods [24]. They have also been proven to vary significantly across space (in the orders of a few centimeters or less) [27]. This property of temporal stability and spatial instability, as depicted in Figure B.3, provides the basis for using the distortions as location fingerprints. For example, Li *et al.* [27] investigated the changes of the geomagnetic field across a small area. They constructed two grids, a large one (8 by 8 RPs) and a small one (6 by 6 RPs). The spacing of the large grid was 30.5 cm while that of the small one was 5 cm. The small

Figure B.2: Macro-cell layout of a cellular network provider in the U.S. for a selected area inside the state of Colorado. Data obtained from [176].

grid is part of the large grid (see Figure B.4). Data were collected at each RP for 30 s. The changes of the geomagnetic field in an area of 4.6 m^2 are significant – the magnetic field intensity varies between 0.315 and 0.411, - 0.267 and -0.012, 0.808 and 1.108, in the X, Y and Z directions respectively. Even in an area of 0.09 m^2, the changes were noticeable. The intensity varies between 0.319 and 0.338, -0.130 and -0.116, 0.994 and 1.005, in the X, Y and Z directions respectively. This suggests that the geomagnetic field could be used for fine-grained positioning.

Magnetic field fingerprints are omnipresent and do not require the deployment of special infrastructure, such as APs in the case of RSS fingerprinting, to be realized. Moreover, a smartphone's magnetometer, which measures fingerprints in µT, consumes far less energy than its WiFi or Bluetooth modules [4]. As a result, magnetic field fingerprinting has attracted researchers since it appears to be a promising alternative for indoor positioning. However, most smart devices come equipped with triaxial magnetometers, meaning that the resultant fingerprints only have three fea-

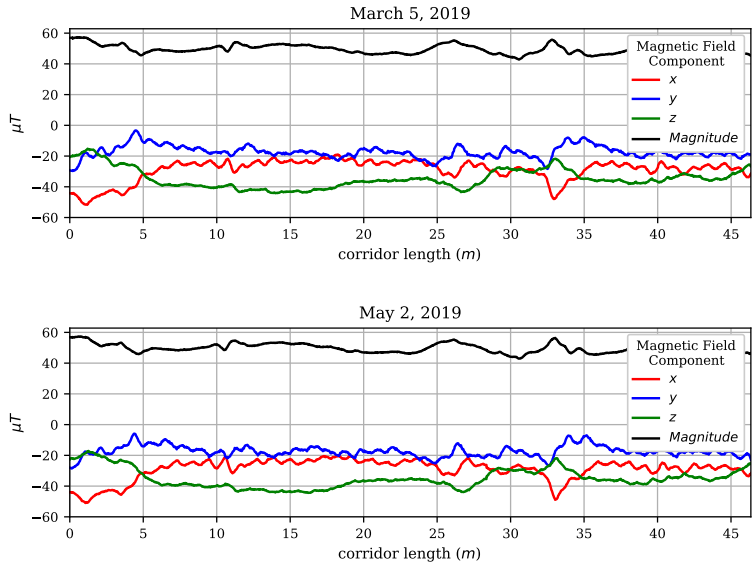

Figure B.3: Two measurements taken two months apart of the magnetic field strength along a 46 m long corridor.

tures. These features are orientation-dependent because they are measured with respect to the device's reference frame (Figure B.5). Consequently, the features are further reduced to two if no restrictions are posed on a smartphone's orientation during the online phase. An orientation independent measure is the magnitude of the magnetic field. However, the magnitude is a single component and using it as a fingerprint can lead to global ambiguity. Another drawback of a magnetic field fingerprint is the vulnerability to magnetic interference caused by live loads such as elevators and vending machines.

Li *et al.* [27] investigated the potential interference caused by some common live loads. A magnetometer was first placed about 30 cm away from two side-by-side elevators, then the distance was gradually increased to about 9 m. Data were collected at various distances from the elevators. The variation of the intensities measured at each test location are plotted in Figure B.6. The variation decreases very rapidly

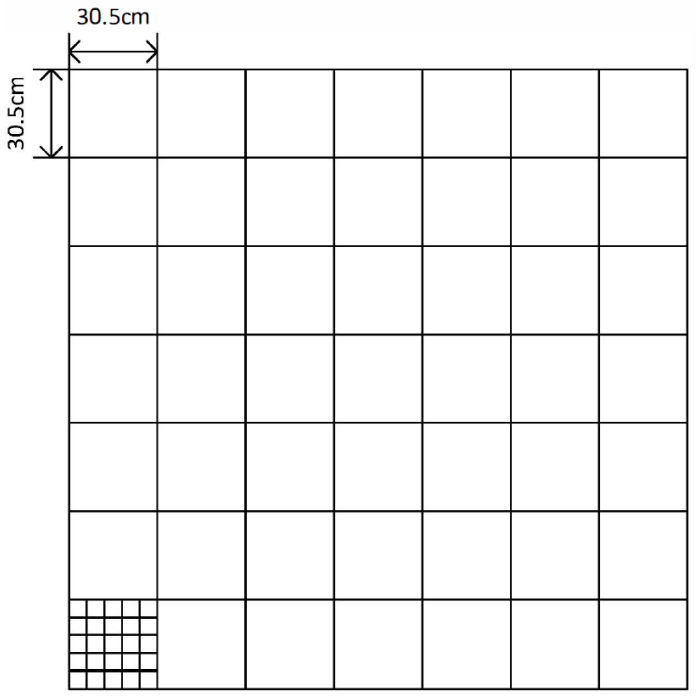

Figure B.4: The large and small grids. Figure reproduced from [27].

with the distance from the elevators, and at about a distance of 7 to 8 m the influence of the elevators is negligible. They also tested small objects such as a mobile phone, metal tin, and headphones. In these tests the magnetometer was fixed, and the small objects were placed very close to the magnetometer and then moved away slowly with constant speed. Figure B.7 plots the magnetic field intensities detected by the magnetometer when a mobile phone was tested. It shows the impact of the phone on the magnetic fields. The influence is significant if the object is very close to the magnetometer. However, as the distance between the phone and the magnetometer increases, the influence reduces quickly. When the distance is more than 15 cm the influence from the phone can be neglected. In the case of a headphone, a metal tin, and a laptop, the separation distance beyond which the object's influence is negligible varies (8 cm, 26 cm, and 32 cm respectively). The size of the object is an obvious factor - the larger the object the greater the separation distance. To address the problem of variations one could use multi-shot positioning [178]. By using continuous magnetic field measurements, the variations can be captured and considered when positioning.

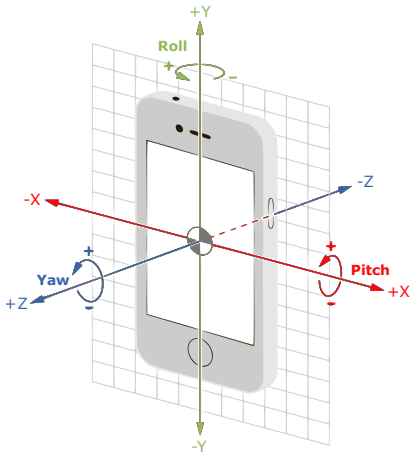

Figure B.5: Illustration of the X, Y, and Z axes relative to a typical smartphone. Figure reproduced from [177].

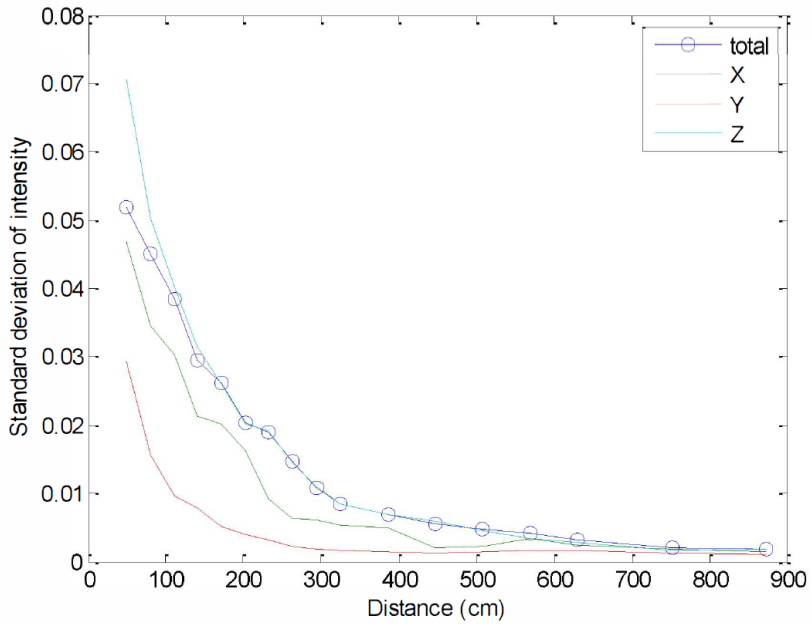

Figure B.6: The influence of elevators on the magnetic field. Figure reproduced from [27].

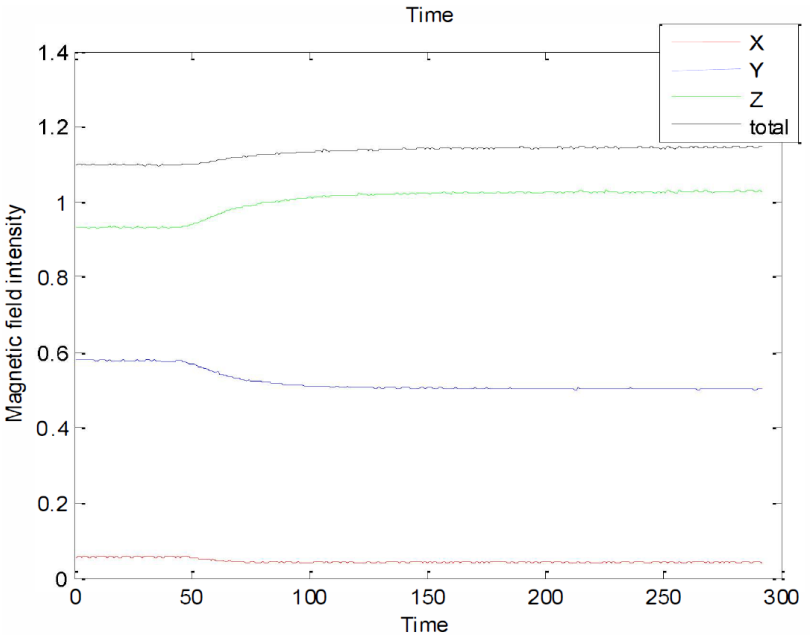

Figure B.7: The influence of a mobile phone on the magnetic field. Figure reproduced from [27].

In open spaces, magnetic fingerprints may not provide enough information for positioning in open spaces due to the lack of distortions. To address this problem, Du *et al.* [179] proposed a dynamic feature fusion strategy. The idea is to use hybrid fingerprints of WiFi and magnetic field. Their proposed method gives more weight to the WiFi fingerprints in areas with low magnetic field discernibility. They were able to achieve an improvement of 45 % in average error distance compared to positioning using magnetic field fingerprints alone.

Among the first to realize that an electronic compass' incorrect heading information can be used as a signature for indoor localization was Suksakulchai *et al.* in 2000 [14]. They mounted an electronic compass on top of a service robot "HelpMate" and collected the heading information as the robot traversed a corridor. The next time the robot traversed the corridor, it matched its measured heading information with the pre-collected information; if a match was found, the robot could determine its position. In 2011, Gozick *et al.* [24] used mobile phones' built-in magnetometers to build magnetic maps of corridors inside buildings. These maps were constructed with the phones' y-axes parallel to the north and prior knowledge of the corridors' steel pillars locations. The authors used the magnitude of the magnetic field as a feature to differentiate between the different pillars (magnetic landmarks). They showed that the magnetic signatures collected by different mobile phones with different sampling rates have the same pattern.

B.3 Image Fingerprints

Using images for indoor localization is viable because most smart devices are armed with cameras. Like magnetic- and cellular-based localization, image-based localization does not depend on infrastructure for operation. Nonetheless, in some

scenarios, cameras may not be allowed indoors due to privacy and security concerns [180]. Furthermore, image fingerprints are the largest in terms of memory footprint and number of features. For example, compare an image fingerprint captured by an iPhone 7, a fingerprint with 12 million features and a memory footprint of 6 MB (stored as a `.jpg` file), to a WiFi fingerprint with 127 features and a memory footprint of 4 KB (stored as a `.txt` file). Therefore, to reduce the number of features for training, image-based localization systems often re-size images to a lower resolution and use cropping to select only the region of interest. Additionally, image compression techniques should be considered when relying on a remote server for positioning or when the available bandwidth for transmission is limited [181].

As seen in Figure B.8, the methods used for image-based localization can be generally divided into *indirect* and *direct* methods [182]. Indirect methods cast the localization problem as an image retrieval task in which the query image is matched against previously collected images, thus, providing coarse pose information (i.e., position and orientation of the camera). Direct methods, on the other hand, treat the localization problem as a regression task where camera pose is directly estimated from a query image. The main source of positioning error is caused by *perceptual aliasing* [183], in which two images of two different places appear similar due to lighting condi-

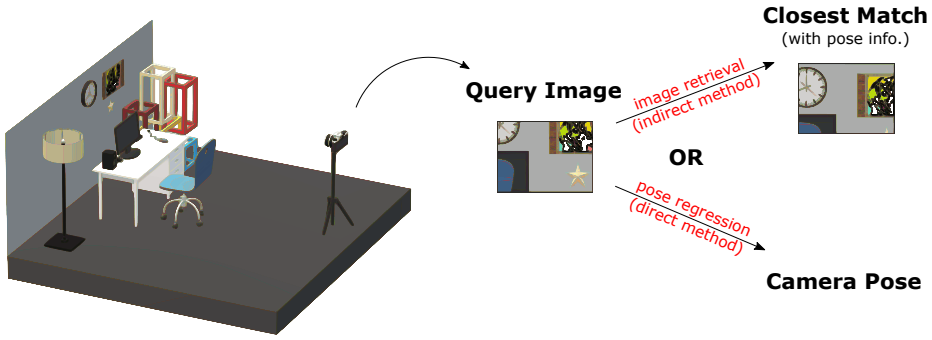

Figure B.8: The two main approaches to image-based indoor positioning (i.e., indirect and direct).

tions or repetitive structures and surfaces. To alleviate this issue, many solutions rely on classical feature-detection algorithms such as Scale-Invariant Feature Transform (SIFT), Affine-SIFT, and Speeded Up Robust Features (SURF) to extract robust, invariant features [184, 185, 186, 187]. While powerful, such algorithms are computationally expensive and require the additional step of feature-matching, instigating positioning latencies in the order of seconds if not minutes [184, 188, 189].

One of the earliest attempts of image-based indoor positioning was conducted by Starner *et al.* in 1998 [15]. The images captured by two hat-mounted cameras, one facing forward and the other downward, were used for positioning by employing a Hidden Markov Model (HMM) to model a user transitioning between adjacent rooms. Primitive features were used, composed of the mean value of the red, green, blue, and luminance pixels. A room classification accuracy of 82 % was achieved inside a 14-room testbed.

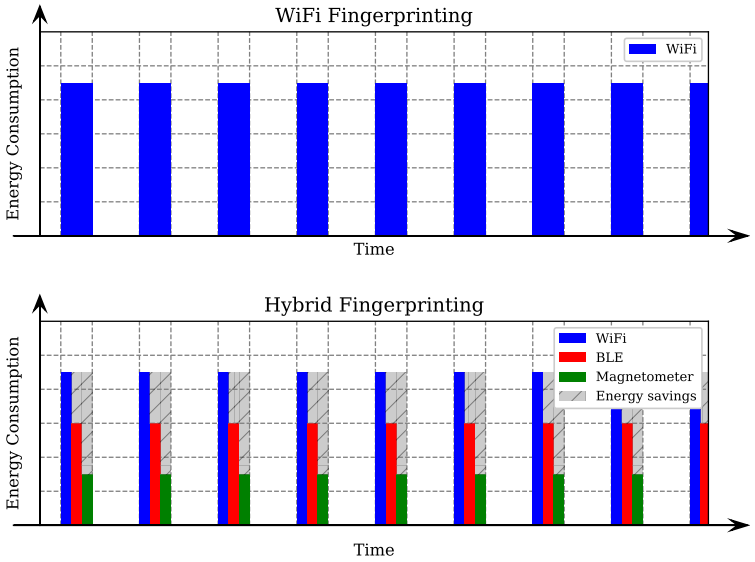

Figure B.9: An illustration of how hybrid fingerprints can reduce energy consumption. The upper plot represents a system that uses WiFi-only fingerprints, while the lower plot represents a system that uses a combination of WiFi, BLE, and magnetic field fingerprints. The scan rate/period is the same for both systems.

B.4 Hybrid Fingerprints

A hybrid fingerprinting system is a system that utilizes two or more fingerprint types for positioning. Hybrid fingerprinting systems aim to improve overall performance which can take the form of:

1. *Improved accuracy*: Combining different fingerprint types provides additional location-specific information. It increases feature dimensionality, resulting in a richer feature set that, in turn, enhances location discrimination. This is often demonstrated in literature by quantifying the gain in positioning accuracy obtained by using multimodal fingerprints instead of unimodal fingerprints [16]. Nonetheless, cautious handling of sensor synchronization and data fusion is essential to minimize the impact on response time [190].

2. *Improved energy efficiency*: Since different sensors vary in their power requirements, low-power sensors can be exploited to enhance the energy efficiency of an otherwise less-efficient system. This concept is visually illustrated in Figure B.9 however, this requires optimal *sensor scheduling* since degradation in positioning accuracy is expected if the time allocated for WiFi/BLE scanning isn't enough to detect all APs/beacons necessary for positioning [191]. Another way of enhancing energy efficiency is to activate sensors only when needed. To help decide when to activate/deactivate sensors, IMU and other sensor measurements can be analyzed to identify a user's state (stationary vs. walking) [192], as well as a phone's state (handheld vs. in-pocket) [193].

3. *Improved availability*: Hybrid fingerprints form the basis for *opportunistic localization* [51]. The idea of opportunistic localization is to maximize a system's availability through the exploitation of all available fingerprint types in a given

environment, without relying on specific infrastructure. It can be viewed as a fallback solution in case some fingerprint types cannot be obtained due to infrastructure maintenance/failure. The main drawback of opportunistic localization is its high implementation complexity.

SurroundSense, proposed by Azizyan *et al.* in 2009 [16], is recognized by many as the first hybrid fingerprinting system. The system combines multiple fingerprint types, such as sound, visible light, WiFi, and image fingerprints, to increase location discernibility. Evaluation results across 51 stores/shops demonstrated the system's ability to provide symbolic positioning with 87 % accuracy. This is an increase of 24 %, 17 %, and 13 % in positioning accuracy over WiFi, sound-and-WiFi, and sound-light-image fingerprints, respectively. However, the system's design is very complicated because it involves several filtering, formatting, matching, clustering, and audio/image processing modules.

B.5 Miscellaneous Fingerprints

B.5.1 UWB Fingerprints

UWB is a wireless technology designed for high-bandwidth, short-range (<10 m) communication. It works by transmitting ultra-short pulses (<1 nanosecond (ns)) across a wide spectrum of frequency bands (>500 MHz). Although the FCC permitted the operation of UWB in 2002 [194], slow progress in standardizing the technology has limited its adoption in consumer devices [195]. Concerning indoor positioning, UWB has proved superior to other wireless technologies, specifically for lateration-based approaches, due to its high time delay resolution and, hence, multipath resilience [196].

B.5.2 Visible Light Fingerprints

The emergence of Visible Light Communication (VLC) recently enabled Light Emitting Diode (LED)-based indoor positioning [17]. Due to the high directivity of visible light, LED-based positioning systems can provide sub-meter accuracy (based on lateration/angulation) [17]. Moreover, LEDs are low-cost, energy-efficient, provide stable performance, and have a long lifetime (∼50,000 hours). However, one drawback is the degradation of performance in NLoS conditions since VLC is inherently an LoS technology. Also, the coverage of such systems is low because visible light cannot penetrate opaque objects such as walls and panel partitions. Also, in green buildings, where, during the day, lighting is provided by sunlight, an LED-based positioning system may not be a practicable solution.

B.5.3 RFID Fingerprints

RFID is a wireless technology designed to retrieve data from transponders in proximity. Unlike WiFi or Bluetooth, RFID is not supported on mobile devices. Thus, RFID-based applications assume the deployment of dedicated infrastructure (RFID readers and tags). This makes RFID an unappealing and costly option for positioning. Nevertheless, due to their energy-efficient and durable operation, RFID has been widely used for asset management and access control [197].

B.5.4 Acoustic Fingerprints

The least popular indoor positioning systems are acoustic-based. This is due to the many challenges that arise when using acoustic signals for indoor positioning such as the strong attenuation of aerial acoustic signals, the limited bandwidth of microphones, the various interferences in the audible band, the short operation distance,

and the associated sound pollution [198]. Nevertheless, given how water, as a propagation medium, favors acoustic over radio frequency and light signals, acoustic signals are widely used for underwater positioning [199].

APPENDIX C
INDOOR POSITIONING DATASETS

This appendix provides a detailed review of datasets that are used to develop and benchmark fingerprinting systems. The datasets were selected based on various criteria, the most important of which was their suitability for training deep learning models from scratch. Deep learning is inherently a data-intensive endeavor. In other words, one of the major drawbacks of deep learning is its need for large datasets for training. Therefore, a dataset must at least contain thousands of location-tagged instances to qualify for review. Small-scale datasets, such as those described in [37, 87, 200, 201] were omitted from this review. However, small-scale datasets can be used to fine-tune pre-trained models as demonstrated in [201]. Other selection criteria included scientific quality, novelty, and potential application domains. Eleven datasets were identified and categorized into four categories according to the data types that they represent: radio frequency, magnetic field, image, and hybrid.

The first category, radio frequency, comprises four datasets of RSS fingerprints collected from either off-the-shelf smart devices or custom-built devices. The second category, magnetic field, contains two datasets of annotated magnetic field and IMU measurements captured using smartphones. The third category, image, contains two datasets of image fingerprints with accurate and precise position and pose information. The fourth category, hybrid, includes three labeled datasets of heterogeneous

Table C.1: A side-by-side comparison of the datasets with respect to the collection environment

Dataset (Year)	Type	Buildings	Floors	Rooms	Corridors	Area (m^2)	RPs	Spacing of RPs (m)
Radio Frequency								
UJIIndoorLoc (2014)	University buildings	3	13	254	-	108,703	933	-
[89] (2018)	University library	1	2	-	-	432	212	-
[90] (2018)	Residential homes	4	7	34	-	350	194	1
[91] (2018)	A research facility	1	1	8	1	237	277	0.6
Magnetic Field and IMU								
UJIIndoorLoc-Mag (2015)	A research lab	1	1	1	8	260	-	-
MagPIE (2017)	University buildings	3	3	-	-	960	-	-
Image								
7-Scenes (2013)	An office space	1	1	7	-	36.5	-	-
Warehouse (2018)	A warehouse	1	1	-	-	875	-	-
Hybrid								
[29] (2016)	A research facility	1	1	3	3	185	325	0.6
PerfLoc (2016)	Office; Industrial warehouses; Subterranean structure	4	7	-	-	30,000	900+	-
[202] (2019)	-	1	1	4	2	651	70	-

data simultaneously recorded using the same smart devices. The datasets within each group are described in ascending order by publication date. Table C.1 provides a side-by-side comparison of all discussed datasets with respect to the collection environment, while Table C.2 compares the datasets with respect to the sampling nature and collection platform. Table C.3 highlights some of the datasets' pros and cons and provides the download link for each dataset.

C.1 Radio Frequency Datasets

C.1.1 UJIIndoorLoc

The UJIIndoorLoc dataset [36], proposed in 2014, is well known for being the first publicly available RSS dataset. It was created to address the lack of a common dataset for comparing state-of-the-art WiFi fingerprinting systems. The data were collected from three adjacent multi-floor buildings (4-5 floors) of the Jaume I University campus. A single RP was placed at the center of each room and in front of the door(s) leading to the rooms. 25 smart devices carried by 20 participants were used

Table C.2: A side-by-side comparison of the datasets with respect to the sampling nature and the collection platform

	Samples					Platform				
Dataset (Year)	**Type**	**Rate (Hz)**	**Training**	**Testing**	**Features**	**Collection side**	**Devices**	**Type**	**OS**	**Orientation**
Radio Frequency										
UJIIndoorLoc (2014)	Discrete	-	19,938	1,111	520	User	25	Smartphone; Tablet	Android	Not provided
[89] (2018)	Discrete	-	∼15,500	∼88,000	620	User	1	Smartphone	Android	Provided for only two directions
[90] (2018)	Discrete; Continuous	5; 25	∼730,000	-	varies	Nodes	8 or 11	Raspberry Pi	-	Provided
[91] (2018)	Discrete; Continuous	10	∼2,820,000	-	varies	User; Nodes	1 to 11	Raspberry Pi; Smartphone	Android	Provided
Magnetic Field and IMU										
UJIIndoorLoc-Mag (2015)	Continuous	10	270	11	9	User	2	Smartphone	Android	Provided
MagPIE (2017)	Continuous	50; 200	591	132	9	User	2	Smartphone	Android	Provided
Image										
7-Scenes (2013)	Discrete; Continuous	-	26,000	17,000	307,200	User	1	Kinect Red-Green-Blue-Depth (RGB-D) camera	-	Provided
Warehouse (2018)	Discrete; Continuous	-	202,224	262,570	307,200	User	8	Web camera	-	Provided
Hybrid										
[29] (2016)	Discrete	10	36,795	-	varies	User	2	Smartphone; Smartwatch	Android	Provided
PerfLoc (2016)	Discrete; Continuous	from 0.3 to 100	varies	private	varies	User	4	Smartphone	Android	Provided
[202] (2019)	Discrete	-	1,010,640	-	16	Nodes	5	Raspberry Pi	-	Provided for only one angle

to collect over 20,000 discrete samples from 933 RPs. Each sample is comprised of 520 RSS measurements corresponding to the 520 APs scattered across the buildings along with ground truth information, such as building and floor numbers, latitude and longitude, a timestamp, and user and device labels. The RSS value of a detected AP ranged from 0dBm (very strong signal) to −104dBm (very weak signal). Undetected APs were given an artificial value of +100dBm. On average, 27 APs were detected per RP. 5 % of the collected samples were dedicated as a separate testing set. The authors provided a baseline of an 89.92 % hit rate and a 7.9 m mean error using the kNN classifier (with $k = 1$ and a Euclidean distance metric).

Table C.3: The pros, cons, and download link for each dataset

Dataset (Year)	Pros	Cons	Download Link
UJIIndoorLoc (2014)	Unique in terms of the area covered, the number of RPs surveyed, and the number of devices used in data collection.	No orientation information was provided which may lead to inconsistent measurements [203].	https://archive.ics.uci.edu/ml/datasets/ujiindoorloc
[89] (2018)	Samples were collected over 25 months which helps study temporal signal variations for the development of systems robust to these variations.	Samples were collected facing only two opposing direction for each RP. Didn't specify whether environment changes have occurred during the collection period.	https://doi.org/10.5281/zenodo.1309317
[90] (2018)	Since data were collected from private residential homes and from various activity zones, it is appealing for studying indoor tracking in support of AAL.	Not suited for studying smartphone-based indoor positioning.	https://doi.org/10.6084/m9.figshare.6051794.v5
[91] (2018)	Data was collected from both user and node sides. Various scenarios and transmission powers were explored.	The samples corresponding to a user/node sending signals to itself were not filtered out.	http://wnlab.isti.cnr.it/localization
UJIIndoorLoc-Mag (2015)	Data collection was repeated several times over the same path which makes it easier to detect noise and outliers in the measurements.	Provides very few calibration points since ground truth location information was only recorded at the beginning and end of each line segment.	http://archive.ics.uci.edu/ml/datasets/UJIIndoorLoc-mag
MagPIE (2017)	Data were collected with and without the placement of live loads. Orientation of the smartphone kept fixed throughout which is key for consistent magnetic field measurements.	Relied on Google Tango for ground truth measurements which has proven to be an unreliable source for accurate measurements [204].	http://bretl.csl.illinois.edu/magpie/
7-Scenes (2013)	Includes depth images which is compelling as smartphones equipped with depth cameras have recently started to appear in the market.	Each room has its own coordinate system which is contrary to real life scenarios in which an indoor environment composed of multiple rooms share the same coordinate system.	https://www.microsoft.com/en-us/research/project/rgb-d-dataset-7-scenes/
Warehouse (2018)	Various testing scenarios and highly accurate and precise ground truth measurements.	Requires more than 30 GB of memory space to store the entire dataset.	https://www.iis.fraunhofer.de/warehouse
[29] (2016)	Contains samples collected from a smartwatch. Additionally, magnetic field data was collected from rooms rather than corridors only.	The arrival and departure timestamps of some RPs are missing and the WiFi fingerprints were collected from the smartphone only.	http://wnet.isti.cnr.it/software/Ipin2016Dataset.html
PerfLoc (2016)	Most diversified in terms of the data types collected. Moreover, data were collected to comply with most of the testing and evaluation criteria as specified by the ISO/IEC 18305:2016 standard.	Non-uniform sampling rates across smartphones resulted in asynchronous data samples. Also, data is not directly accessible as there is a steep learning curve to decode the data before start using it [205].	https://perfloc.nist.gov/
[202] (2019)	Well-suited for studying indoor tracking using hybrid measurements. Moreover, the dataset contains Xbee measurements and has over 1 million samples.	Orientation is provided around a single axis only (i.e., yaw/heading angle). Not suited for studying smartphone-based indoor positioning.	http://www.gatv.ssr.upm.es/~abh/

C.1.2 Dataset described in [89]

The dataset described in [89] was collected over fifteen months. The primary goal of creating the dataset was to provide researchers with the data needed to study a system's robustness against short/long-term WiFi signal variations. Short-term variations are caused by multipath and shadowing while long-term variations are caused by environment and network changes. Data was collected using a smartphone on two identical floors ($3^{\text{rd}}$and 5^{th}) of a $12\times18\,\text{m}^2$ library wing with 106 RPs per floor. At each RP, consecutive samples facing the same directions were collected, multiple

times a month. During a month, 15 % of the samples collected were allocated for training while the remaining 85 % were allocated for testing, except for the samples collected during the first month (73 % training and 27 % testing). A total of 63,504 samples were collected by last month. Each sample consisted of a timestamp, ground truth floor number, RP coordinates, and the RSS values of all detected APs over the entire period (i.e., starting with 77 APs at month 1 and ending with 448 APs at month 15). Recently, the authors updated the dataset to include 40,080 new samples corresponding to an additional collection period of ten months with 172 newly detected APs. Supporting scripts in MATLAB, that allow for loading a desired set based on filtering criteria, are provided.

C.1.3 Dataset described in [90]

The dataset by Byrne *et al.* [90] contains approximately fourteen hours of annotated wearable measurements acquired from four single- and two-floor residential homes with four to eleven rooms. At each residence, a custom-built, wrist-worn transmitter sent accelerometer measurements, via BLE radio (in advertising mode), which were then received by several custom-built anchor nodes deployed throughout the residence. Upon reception, each node records the RSS of the advertised packet and timestamps it. Ground truth location labels were provided through fiducial floor tags that were placed 1 m apart throughout the home. A downward-facing camera, strapped to a participant's navel area, automatically captured the floor tags as the participant traversed them. At each floor tag, data were collected facing each of the four cardinal directions to account for the shadowing effect imposed by the participant's body. Additionally, the dataset incorporated samples generated from both scripted and unscripted scenarios. Scripted scenarios represented walking rapidly or slowly throughout the residence while unscripted scenarios represented participants

carrying out their normal daily living routine. The dataset also contains annotated data collected from "activity zones" (i.e., certain locations coincide with certain activities, such as cooking in the kitchen, eating at the dining table, or relaxing on the sofa). In total, the dataset contains around 730,000 samples. Python scripts for loading the the dataset form the repository are provided.

C.1.4 Dataset described in [91]

The dataset by Baronti *et al.* [91] was introduced as a general-purpose dataset that can be used for positioning, tracking, proximity/occupancy detection, and social interaction detection. Data collection was performed inside a $16.6 \times 14.3\,\mathrm{m}^2$ research facility consisting of eight rooms, a connecting corridor, and 277 RPs spaced 0.6 m apart. Each room contained a Raspberry Pi equipped with two BLE modules. One module continuously listened for signals while the other transmitted advertisements at 10 Hz. Similarly, mobile users carrying a smartphone (as a receiver) and a BLE tag (as a transmitter) were employed to enable data collection both ways (i.e., from user to anchor nodes and vice versa). Six scenarios were used for data collection: "survey", "localization", and four "social". In the survey scenario, the user stood over each RP and collected data along the $+x$, $+y$, $-x$, and $-y$ directions. The localization scenario represented a user walking a predefined path (i.e., continuous sampling). The social scenarios represented two/three users walking from their offices, attending meetings, and returning to their offices. For each scenario, three runs of data collection were performed, corresponding to three transmission powers (i.e., 3dBm, $-$6dBm, and $-$18dBm). Each sample consists of a timestamp, transmitter ID, receiver ID, and RSS value. Ground truth location information is provided through a separate file that maps timestamps to the coordinates of the RPs. Overall, the dataset has around 2,820,000 samples.

C.2 Magnetic Field Datasets

C.2.1 UJIIndoorLoc-Mag

The creators of the UJIIndoorLoc dataset introduced the UJIIndoorLoc-Mag dataset in 2015 [85]. The aim was to provide a common dataset for the evaluation of magnetic field fingerprinting systems as they became increasingly popular. Unlike UJIIndoorLoc, the data contained in UJIIndoorLoc-Mag was collected in a much smaller area (a single $15 \times 20\,\mathrm{m}^2$ office space). A smartphone was used to collect continuous samples along the office's eight corridors at a sampling rate of 10 Hz. Each continuous sample represents walking along a predefined path composed of multiple straight-line segments. The data collection process involved several predefined paths where sampling over each path was repeated multiple times yielding a total number of 281 continuous samples (or 40,159 discrete captures). Each discrete capture incorporated timestamped, raw measurements from the phone's magnetometer, accelerometer, and orientation sensor along its three axes (Figure B.5). Ground truth location information was recorded at the beginning and end of each continuous sample and turning points (i.e., the end of a segment and the beginning of another). The authors used a subset of the dataset to provide a baseline of a 7.23 m mean error using the kNN classifier (with $k = 1$ and a Euclidean distance metric).

C.2.2 MagPIE

The Magnetic Positioning Indoor Estimation (MagPIE) dataset [88] is, by far, the largest dataset for studying and comparing approaches to magnetic and inertial indoor positioning. The data were collected from three different university buildings. A smartphone, either handheld or mounted on a wheeled robot, was used to collect 723 continuous samples equaling 51 km of total distance traveled. The sampling rate

was 50 Hz for magnetometer data and 200 Hz for accelerometer and gyroscope data. To account for soft/hard iron biases, the dataset provides calibrated measurements as opposed to raw magnetic field measurements. A separate smartphone was used to provide ground truth location information by running Google Tango, an augmented reality platform for mobile devices (discontinued March 2018). Data were collected under two scenarios (i.e., with and without the placement of "live loads"). Live loads are certain objects, commonly found inside buildings, that may affect the magnetometer's measurements. However, the number of live loads placed, their description, and their ground truth location information were not provided.

C.3 Image Datasets

C.3.1 7-Scenes

The 7-Scenes dataset, introduced by Microsoft Research in 2013 [206], has been widely used for image-based localization. It is composed of Red-Green-Blue images and their corresponding depth images (collectively called RGB-D images) of seven small-scale indoor scenes. Each scene typically consists of a single room (e.g., office, kitchen). The spatial volume of these scenes ranges from $2\,\text{m}\times0.5\,\text{m}\times1\,\text{m}$ to $4\,\text{m}\times3\,\text{m}\times1.5\,\text{m}$. All images were captured using a handheld *Kinect* RGB-D camera at 640×480 resolution. Ground truth position and orientation information was provided by the SLAM-based *KinectFusion* system. The number of training images for each scene ranges from 1,000 to 7,000 while the number of testing images ranges from 1,000 to 5,000. Overall, the dataset contains 26,000 training images and 17,000 testing images. The dataset is considered challenging for positioning algorithms due to notable motion blur, variations in camera pose, and because scenes contain many ambiguous texture-less features.

C.3.2 Warehouse

Warehouse [207] is a dataset created for the development and benchmarking of image-based localization systems in industrial settings. For data collection, the authors utilized eight web cameras mounted on special platforms that placed them at 45° increments. Each camera captured Red-Green-Blue (RGB) images at 640×480 resolution inside a $25\times35\,\mathrm{m}^2$ industrial warehouse. Each image is labeled with a sub-millimeter position and sub-degree orientation information using a laser-based reference system. Two trajectories, intended to uniformly cover the area, were followed to obtain over 200,000 training images. The testing images were collected over carefully designed trajectories aimed at evaluating different aspects of the positioning system such as its ability to generalize and respond to environmental changes and scaling and its robustness to local and global ambiguity. The authors provided baselines of 1.08 m to 6.76 m mean errors (depending on the testing trajectory) using the CNN-based, pre-trained *PoseNet* [189].

C.4 Hybrid Datasets

C.4.1 Dataset described in [29]

Barsocchi *et al.* [29] collected WiFi, magnetometer, and IMU data from an indoor environment composed of three rooms of different sizes and three corridors of different lengths. Data collection was performed by concurrently wearing two synchronized smart devices: a smartphone and a smartwatch. A fixed sampling rate of 10 Hz was used for both devices. The smartphone was held at chest-level of the person collecting the data, with the screen facing up, while the smartwatch was wrist-worn. Data were collected over two campaigns from 325 uniformly distributed and regularly spaced RPs covering a surface area of $185\,\mathrm{m}^2$. The ground truth coordinates of these

points, along with arrival and departure timestamps at each point, are included in the dataset. In total, the dataset contains over 36,000 discrete instances.

C.4.2 PerfLoc

For PerfLoc [86], data were collected based on guidance from the *ISO/IEC 18305: 2016* international standard for testing and evaluating Localization and Tracking Systems (LTSs) [208]. The standard specifies that localization systems should be evaluated under different environmental and mobility settings. Hence, the data includes timestamped samples collected from four different buildings (including a subterranean structure) using different mobility modes such as walking, running, walking backward, crawling, and sidestepping. Four Android-based smartphones, strapped to the upper arms of the person collecting the data, were employed to collect data from the 900+ RPs placed throughout the buildings. Diverse data were collected including: WiFi, cellular, GPS, and all other available sensor data for a given smartphone (e.g., magnetic field, acceleration, temperature, pressure, humidity, light intensity, etc.). The sampling rate ranged from 0.3 Hz to 100 Hz, depending on the data type sampled and the smartphone's brand and model. The authors provide a private testing set through an online web portal where developers can upload their location estimates and get real-time feedback on their system's performance.

C.4.3 Dataset described in [202]

The dataset by Belmonte-Hernández *et al.* [202] contains Xbee, BLE, WiFi, and orientation measurements collected in a $31 \times 21\,\mathrm{m}^2$ area comprised of four rooms and two corridors. The data were collected using five Raspberry Pi receivers that were strategically placed in the environment. The entire environment was divided into seventy rectangular cells of different sizes, ranging from $1.5 \times 1.42\,\mathrm{m}^2$ to $2.56 \times 1.9\,\mathrm{m}^2$.

At least five minutes of measurement was recorded for each cell in all 360 °. A person wearing a Raspberry Pi transmitter attached to their hip would stand at the center of cells to complete data collection. These received measurements were then synchronized and labeled with the coordinates of the cells' centers. Overall, the dataset has about one million samples.

APPENDIX D
DEDICATION

To Fatma Fnais Alhomayani, my grandmother,
and Abdullah Fahad Alhomayani, my cousin,
who saw me embark on this journey,
but never got to see me complete it.
May your souls rest in peace.