DeepBLE: Generalizing RSSI-based Localization Across Different Devices

Harsh Agarwal

CMU-RI-TR-20-10

April 15, 2020

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Prof. Kris Kitani, *chair*
Prof. Michael Kaess
Navyata Sanghvi

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Robotics.

Dedicated to my beloved family, Mom and Dad, teachers and mentors who have supported me throughout my life.

Abstract

Accurate smartphone localization ($<$ 1-meter error) for indoor navigation using only RSSI received from a set of BLE beacons remains a challenging problem, due to the inherent noise of RSSI measurements. To overcome the large variance in RSSI measurements, we propose a data-driven approach that uses a deep recurrent network, DeepBLE, to localize the smartphone using RSSI measured from multiple beacons in an environment. In particular, we focus on the ability of our approach to generalize across many smartphone brands (e.g., Apple, Samsung) and models (e.g., iPhone 8, S10). Towards this end, we collect a large-scale dataset of 15 hours of smartphone data, which consists of over 50,000 BLE beacon RSSI measurements collected from 47 beacons in a single building using 15 different popular smartphone models, along with precise 2D location annotations. Our experiments show that there is a very high variability of RSSI measurements across smartphone models (especially across brand), making it very difficult to apply supervised learning using only a subset smartphone models. To address this challenge, we propose a novel statistic similarity loss (SSL) which enables our model to generalize to unseen phones using a semi-supervised learning approach. For known phones, the iPhone XR achieves the best mean distance error of 0.84 meters. For unknown phones, the Huawei Mate20 Pro shows the greatest improvement, cutting error by over 38% from 2.62 meters to 1.63 meters error using our semi-supervised adaptation method.

Acknowledgments

I would first like to thank my advisor, Professor Kris Kitani, for giving me an opportunity to work on the NavCog project. Through all the triumphs and failures accompanying this Master program, I am glad to have his experience and guidance by my side.

I want to thank Navyata Sanghvi for helping in my journey, with her valuable feedback, suggestions, support along the course of the project and help me in improving upon the quality of work. A special thanks to Vivek Roy and Sandy Sun (Xi Sun) for helping us with the software development needed for the project. Thanks to Donghyun Yoo for getting me started on this two-year ride, and the insightful discussions which helped me overcome any doubts.

I would also like to thank Professor Michael Kaess for being a part of the committee, and his insights on the work performed. Thanks to all the members of KLab and Cognitive Assistance Lab for providing a thriving workplace and help me grow.

I cannot go without thanking my friends for providing both academic and emotional support in difficult times.

Funding

x

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Smartphone-based localization using Bluetooth Low Energy (BLE) beacons is utilized in various urban spaces, but its accuracy is severely limited by the large variance in Received Signal Strength Index (RSSI) measured across various smartphone models. Urban spaces such as hospitals, museums, shopping malls and airports utilize BLE beacons to offer location-specific services but their use has been largely limited to rough proximity sensing due to the instability of the RSSI measurement. Implementing more accurate forms of smartphone localization needed for applications such as wayfinding (e.g., assisting blind people to navigate indoor spaces) is challenging due to this lack of consistency between smartphone receiver characteristics. In this work, we work towards developing a BLE beacon-based localization technique that can generalize to any smartphone model.

The RSSI of BLE beacons measured by a smartphone can vary greatly for different brands and models due to a number of reasons. Hardware differences such as the form factor of the phone, receiver module, layout of the internal circuitry and antenna design can all affect how the RSSI is measured. Software factors such as built-in signal processing and data loss can also affect the RSSI measurements. Due to these differences in measurements, it is challenging to develop a single method that will work across multiple smartphones. Therefore, it is critical to develop localization methods that can adapt to changes in RSSI measurements across smartphones.

Due to the differences in RSSI measurements across smartphones, fingerprinting based localization methods are a favorable approach. Whereas trilateration methods
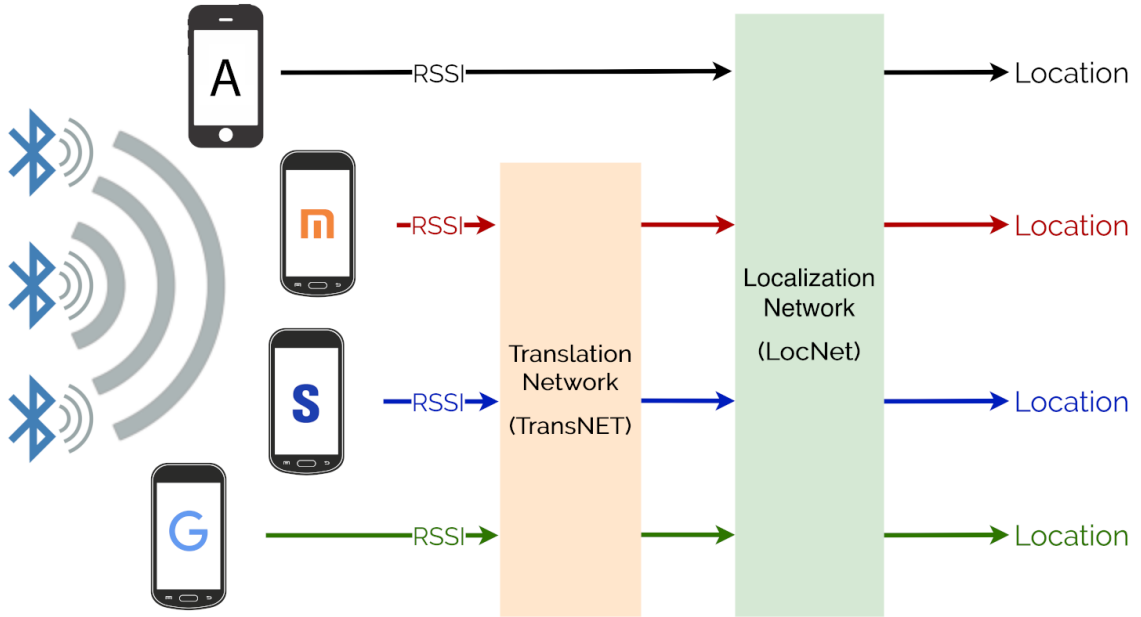
Figure 1.1: Proposed Approach: TransNet learns to transfer RSSI from any phone to RSSI from a reference phone we had during training. LocNet is the Localization engine to get positions given RSSI by BLE

often require calibration of measurements to fit a known sensor model to compute location (e.g., log power drop off), fingerprinting methods bypass the need for calibration by storing a large database of measurements to index location. Fingerprinting also do not require a known parametric model and can capture complex non-linear changes in RSSI measurements that may be challenging to define. The benefit of robustness does come at the cost of a requiring a labor intensive data collection process, where RSSI measurements must be recorded at many locations in the environment prior to use. However, using indoor mapping technology such as robotic mapping systems, the cost of data collection can be reduced.

In this work, we utilize a fingerprinting approach using a recurrent neural network which has the ability to generalize to any smartphone model. We first characterize the performance of a neural network-based localization system using a standard supervised learning approach and show how it fails to generalize to different smartphone models. We then propose a semi-supervise method for adapting the localization system by introducing a translation network which is able to take the RSSI measurements of an

unseen smartphone and convert it to the RSSI measurements of a known smartphone.

To evaluate our method we collect a large-scale dataset of BLE beacon data collected using 15 popular smartphone models from Apple, Samsung, Google, Huawei and Xiaomi. The data is taken in a large building instrumented with 47 BLE beacons. We collected over 15 hours of RSSI measurement using various smartphones. The dataset consists of over 50,000 beacon RSSI measurements and is 10 times larger than existing datasets. The contributions of this work are as follows:

1. We propose the concept of a translation function which maps the RSSI signals of unknown smartphones to the space of known smartphones

2. We provide an empirical characterization of RSSI localization on 15 popular smartphones and quantify the large variance in RSSI measurements;

3. We present the largest BLE beacon RSSI dataset to enable thorough evaluation of localization techniques across multiple smartphone models.

Chapter 2

Related Work

Various localization techniques based on RFID [33], UWB radios [13], and ultrasound [18] have been developed. Recently, localization using Channel State Information (CSI) has been investigated to achieve a high accuracy (about 1m) [3, 31, 35]. However, extracting the CSI in commodity smartphones is non-trivial. In contrast, RSSI information is available ubiquitously and is easy to retrieve. For this reason and relatively low infrastructure costs, using only RSSI from BLE or WiFi remain a popular choice for indoor localization [10, 11, 16, 29].

A clear choice between using RSSI from BLE or WiFi remains an active topic of research. The choice of WiFi often precedes BLE, as most buildings already have WiFi installed. A prime reason for WiFi-based methods to not perform at par with BLE is WiFi routers are not placed to support device localization but instead strategically placed for efficient data transfer. However, at the same time BLE suffers severe packet loss comparing with WiFi and other wireless techniques due to the following three reasons. Firstly, there are only three broadcast channels allocated for BLE beacons. When there are multiple beacon stations installed in the vicinity of the receiver, collisions may happen such that the receiver cannot recover the beacon signals correctly. Secondly, with a much lower transmission power of BLE beacons, the transmission range of BLE is much shorter than that of WiFi. It is thus more likely for receivers to be in the boundary areas of BLE beacon stations and suffer from beacon lost due to fluctuations of receiving signal power. Thirdly, BLE beacon stations are usually battery-powered, so there would be signal vacuums when the

batteries run out. However, in comparative studies [38] [15], the authors conclude that BLE is much more suitable for the application of indoor positioning and navigation.

RSSI-based localization can be categorized into two categories: fingerprinting and trilateration. Fingerprinting uses a database to store the RSSI strengths at various known reference points. At run time, this database is used to calculate the location. Both non-parametric method such as nearest neighbors, and parametric approaches such as neural networks can be used for this computation. However, as RSSI measurements must be collected for many locations in the environment, the process is often time-consuming and expensive. Moreover, as these methods are based on data of the RSSI distribution obtained from a set of receivers, naively using these methods is ought to result in failure when the received RSSI distribution changes. Trilateration on the other hand uses parametric models of RSSI signal attenuation to form geometric constraints of the receiver with respect to a set of transmitters (or vice versa). Known methods are able to estimate position within 2 meters [23, 26]. A variety of models have been proposed such as Gaussian models [11], Monte Carlo [6], Bayesian [20], Hidden Markov Models [17], and radio propagation models [4]. Ring overlapping approaches have been proposed in the past [21, 32], however, these methods tend to require additional calibration, both at the antenna and the phone level.This calibration process based on transmitters and receivers significantly inhibits their usage and generalization ability across different phones. Also modelling signal propagation methods for complex indoor environments is extremely difficult [22] and lead to wrong estimates.

Hence, often for large spaces fingerprinting based RSSI localization remains a popular choice. [9] Transmitters (some BLE, some WiFi) were deployed across a floor, and the ground truth locations were collected using an Active Bats system. Recently, [27] came up with adaptations to a classic particle filter (with an observation model as proposed in [11]) in order to deploy beacon-based localization in a large scale building and achieving a localization error in the range of 2-3 m. However, the method involves a lot of parameter calibration for different scenarios, drawing questions on the generalization of the method across phones.

With the advent of deep learning and machine learning, some methods have tried leveraging these parametric approximators for BLE localization. [2, 14, 34, 37] combine use of robust feature estimates from Deep networks, with classical machine

learning, algorithms like probabilistic KNNs and HMM's to come up with indoor positioning systems. . [5] proposed an end to end deep network performing RSSI based localization. Although the results have been published on WiFi RSSI signals, the end goal is very similar to what we are trying to achieve. Hence we use it as one of our baselines. Very recently, [30] tried solving the inverse problem of localizing beacon scanners, using RSSI strengths received. In the experiments section we compare a very similar network as proposed by [30] architecture to DeepBLE.

Although there is some work on understanding variations of BLE signals across different smart phone, it is not very well understood. [12] studies BLE technology on a variety smartphones models finding out key differences on per smart phone basis. However they do not test the BLE for Localization on different phones. Other works, like [7, 27] evaluate their methods on additional phone models/devices, a rigorous test of generalization is missing. As we see BLE Localization systems being deployed in real world scenarios, with almost 200+ smart phone models present this is inevitably a challenge that would inhibit the use case of RSSI based indoor localization methods in future.

In this work we take up this challenge, analysing BLE RSSI for multiple smartphones, creating a large scale dataset for 15 different smart phones and propose a localization engine that can achieve competitive results on all the phones, without explicitly training the neural network in a supervised fashion on each phone.

citefingerprintingVStrialteration [23]. To estimate the distance from each antenna, a variety of models can be employed: Gaussian models [11], Monte Carlo [6], Bayesian [20], Hidden Markov Models [17], and radio propagation models [4] have been presented over the years. Ring overlapping approaches have been proposed in the past [32] [21], however, these methods tend to require additional calibration, both at the antenna and the phone level. Lately, with Convolutional Neural Networks gaining popularity in Computer Vision, [24] encoding the information like RSSI, ToF and AoA as heat maps and passing through CNN's to get an estimate on of location.

Chapter 3

Problem Setup

Consider an indoor environment with B BLE beacons and P different smartphone models (*e.g.*, iPhone X, S10). Let the set of beacon indices be $\mathcal{B} = \{1..., B\}$, and the set of smartphone model indices be $\mathcal{P} = \{1, ..., P\}$.

At any time step t, smartphone $p \in \mathcal{P}$ is at location $\mathbf{x}_t^p \in \mathbb{R}^2$. The RSSI received by smartphone p from beacon $b \in \mathcal{B}$ at time t is $s_t^{p,b} \in \mathbb{R}_{\geq 0}$. The vector of RSSI received by smartphone p from all B beacons at time t is $\mathbf{s}_t^p = [s_t^{p,1} \ s_t^{p,2} \ ... \ s_t^{p,B}]^T \in \mathbb{R}_{\geq 0}^B$. For a history of H time steps, the matrix of RSSI received by smartphone p from all beacons at times $\{t - (H-1), t - (H-2), ..., t\}$ is $\mathbf{S}_t^p = [\mathbf{s}_{t-(H-1)}^p \ \mathbf{s}_{t-(H-2)}^p \ ... \ \mathbf{s}_t^p] \in \mathbb{R}_{\geq 0}^{B \times H}$

Consider the labeled dataset $\mathcal{D}_l = \{\langle \mathbf{S}_t^p, \mathbf{x}_t^p \rangle \mid p \in \mathcal{P}, \ t \in \mathcal{T}_l\}$ consisting of $P \times |\mathcal{T}_l|$ elements. Each element of $\mathcal{D}_l$ is a tuple, consisting of the matrix of RSSI received by a smartphone p, and its corresponding location at some time $t \in \mathcal{T}_l$.

Consider the unlabeled dataset $\mathcal{D}_u = \{\mathbf{S}_t^p \mid p \in \mathcal{P}, \ t \in \mathcal{T}_u\}$ consisting of $P \times |\mathcal{T}_u|$ elements. Each element of $\mathcal{D}_u$ is the matrix of RSSI received by a smartphone p at some time $t \in \mathcal{T}_u$.

Let there be an unknown function $f : \mathbb{R}_{\geq 0}^{B \times H} \rightarrow \mathbb{R}^2$ mapping the matrix of RSSI received by any smartphone to the location of the smartphone at that time, i.e. $f(\mathbf{S}) = \mathbf{x}$. In the problem of indoor localization, our goal is to approximate this function. In this paper, we consider the following scenarios:

1. We consider the ideal scenario where we have access to **labeled data** from all smartphones $\mathcal{D}_l$.

2. We consider the more realistic scenario where we only have access to **limited labeled data** from a subset of smartphone models. Let $\mathcal{P}'$ be a subset of smartphone models, i.e., $\mathcal{P}' \subset \mathcal{P}$. Then, the corresponding subset of labeled data is $\mathcal{D}'_l = \{\langle \mathbf{S}^p_t, \mathbf{x}^p_t \rangle \mid p \in \mathcal{P}', \ t \in \mathcal{T}_l\} \subset \mathcal{D}_l$. We consider the situation where we only have access to $\mathcal{D}'_l$.

3. Lastly, we consider the scenario where we have access to **limited labeled data** from a subset of smartphone models, as well as **limited unlabeled data** from a subset of unseen smartphone models. Let $\mathcal{Q}'$ be a subset of smartphones such that $\mathcal{Q}' \in \{\mathcal{P} \setminus \mathcal{P}'\}$. Then, the corresponding subset of unlabeled data is $\mathcal{D}'_u = \{\mathbf{S}^p_t \mid p \in \mathcal{Q}', \ t \in \mathcal{T}_u\} \subset \mathcal{D}_u$. We consider the situation where we only have access to $\mathcal{D}'_l$ and $\mathcal{D}'_u$.

In each scenario, our goal is to approximate the function f from the available data $\mathcal{D}_{\text{train}}$, such that our approximation $\hat{f}$ provides accurate localization of any smartphone in a known environment. Specifically,   scenario 1: $\mathcal{D}_{\text{train}} = \mathcal{D}_l$, scenario 2: $\mathcal{D}_{\text{train}} = \mathcal{D}'_l$,   scenario 3: $\mathcal{D}_{\text{train}} = \mathcal{D}'_l \cup \mathcal{D}'_u$.

Chapter 4

Proposed Approach

As discussed in Section 3, our goal is to construct an approximation $\hat{f}$ to the localization function $f : \mathbb{R}_{\geq 0}^{B \times H} \to \mathbb{R}^2$, which maps the matrix of RSSI received by any smartphone to the location of the smartphone at that time. In each of the three scenarios we consider, the localization function $\hat{f}$ is approximated with a deep neural network. For each scenario, we propose different paramterizations of $\hat{f}$, objectives, and training procedures based on the data available for training. We refer to our proposed methods collectively as **DeepBLE**.
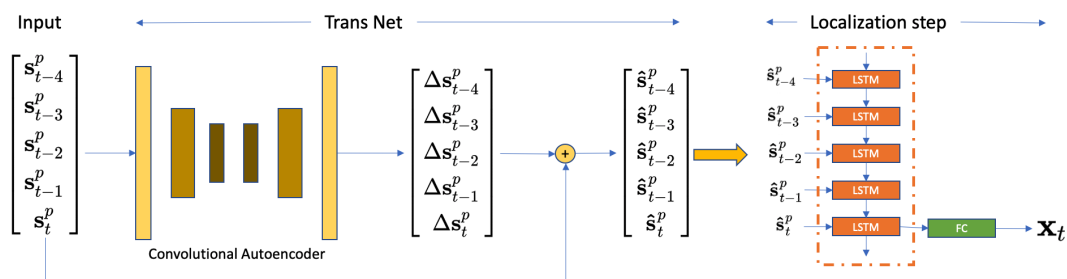
Figure 4.1: Proposed Architecture

4.1 Scenario 1: Labeled Data for All Smartphone Models

We develop our first DeepBLE method for learning the localization function $\hat{f}$ where we are given labeled data from all phone models for training, *i.e.*, $\mathcal{D}_{\text{train}} = \mathcal{D}_l$. While this scenario is ideal, it is often unrealistic as it requires that we obtain the ground truth location for every possible smartphone model for training.

4.1.1 Localization Network (`LocNet`):

The fully supervised localization network $\hat{f}$ is modeled as a 2 layer LSTM followed by two fully connected layers. We call the localization network, `LocNet`. At time t, it takes as input $\mathbf{S}_t^p$, i.e., the beacon readings received by smartphone p in the last $H = 5$ seconds, and outputs the location $\mathbf{x}_t^p$. During training, `LocNet` learns to correctly interpret temporal beacon information, e.g.: how fast signals are changing, patterns of oscillation in signals, etc., and maps it to a location $\mathbf{x}$.

4.1.2 Optimization:

We define the following objective function:

$$\min_{\hat{f}} \quad \mathbb{E}_{\langle \mathbf{S}, \mathbf{x} \rangle \sim \mathcal{D}_l} \left[\left\| \hat{f}(\mathbf{S}) - \mathbf{x} \right\|_2^2 \right] \tag{4.1}$$

This is the standard empirical risk minimization objective for supervised learning, where we learn the minimizer $\hat{f}$ over the training data for all phone model $\mathcal{P}$.

4.2 Scenario 2: Limited Labeled Data

We now develop our second DeepBLE method for a more realistic scenario, where we have access to limited labeled data from a subset of smartphone models $\mathcal{P}' \subset \mathcal{P}$, i.e., $\mathcal{D}_{\text{train}} = \mathcal{D}_l'$. Each model included in the subset $\mathcal{P}'$ belongs to a brand. For example, $\mathcal{P}'$ may include iPhone 7 and iPhone 8 from the Apple brand as well as Pixel 3 and

Pixel 4 from the Google brand. In this example, $\mathcal{P}'$ does not include any models from Samsung, Xiaomi or Huawei brands.

If we learn a localization function according to optimization 4.1 using only the limited labeled data $\mathcal{D}'_l$, we will overfit to the smartphone brands included in $\mathcal{P}'$ since the training data is not representative of all smartphone brands. In order to avoid overfitting, we propose the use of a BLE signal translation function which maps RSSI measured by any smartphone to that of a smartphone brand which is known, *i.e.*, one which was part of the labeled training dataset.

Consider a brand-specific localization function $h : \mathbb{R}^{B \times H}_{\geq 0} \to \mathbb{R}^2$, which maps the matrix of RSSI measured by smartphone models of a known brand to the smartphone's location. Let the set of smartphone models of that known brand be indicated by $\mathcal{J} \subset \mathcal{P}'$.

Now consider a BLE signal translation function $g : \mathbb{R}^{B \times H}_{\geq 0} \to \mathbb{R}^{B \times H}_{\geq 0}$, which takes as input the matrix of RSSI received by any smartphone model $p \in \mathcal{P}$ and transforms it to a corresponding matrix of RSSI measured by a model $p' \in \mathcal{J}$ from the known smartphone brand, at the same location. Then, $f(\mathbf{S}) = h(g(\mathbf{S}))$ is the true localization function, where $\mathbf{S}$ is the matrix of RSSI received by a smartphone model $p \in \mathcal{P}$. Our goal is to learn an estimate of the localization function $\hat{f} = \hat{h}(\hat{g})$.

For this scenario, we define a new loss function with the following components:

1. **Localization loss:** $\mathcal{L}_{\text{loc}} = \left|\left|\hat{h}(\hat{g}(\mathbf{S}_t)) - \mathbf{x}_t\right|\right|^2_2$. This quantifies deviation of estimates of our localization function from ground truth label $\mathbf{x}$ and is identical to the loss function in scenario 1.

2. **Position smoothness loss:** This quantifies the distance between consecutive location predictions, $\mathcal{L}_{\text{ps}} = \left|\left|\hat{h}(\hat{g}(\mathbf{S}_t)) - \hat{h}(\hat{g}(\mathbf{S}_{t-1}))\right|\right|^2_2$. This term acts as a regularization term to ensure that estimated motion is smooth.

3. **Statistic similarity loss (SSL):** Intuitively, we know that beacons that are near each other will often be 'seen' together by the smartphone Bluetooth receiver. This means that beacon RSSI measurements are correlated and that the outputs of the translation function g should also be correlated. Formally, consider the subset of models $\mathcal{J} \subset \mathcal{P}'$ of the known brand . Labeled data from models of this brand is given by $\mathcal{D}'_J = \{\langle \mathbf{s}^p_t, \mathbf{x}^p_t \rangle \mid p \in \mathcal{J}, \ t \in \mathcal{T}_l\}$. We use the brand-specific data $\mathcal{D}'_J$ to infer brand-specific correlation statistics.

Now, consider a matrix $\mathbf{M} \in \mathbb{R}_{\geq 0}^{B \times B}$ of statistics derived from data $\mathcal{D}'_J$. Each entry of $\mathbf{M}$ is described as:

$$\mathbf{M}_{ij} = \mathbb{E}_{\langle \mathbf{s}, \mathbf{x} \rangle \sim \mathcal{D}'_J}[\mathbf{s}_j \mid \mathbf{s}_i > 0],$$

where $\mathbf{M}_{ij}$ is the expected RSSI from the jth beacon when the RSSI from the ith beacon is detected.

We use the inferred statistics to quantify the deviation of the output of $\hat{g}$ from the expected statistics of smartphone models $\in \mathcal{J}$. Our proposed statistic similarity loss (SSL) is defined by:

$$\mathcal{L}_{\text{ssl}} = \sum_{t=1}^{H} \sum_{i=1}^{B} \sum_{j=1}^{B} w_{it} d_{ijt}$$

where:

$$w_{it} = \exp\left(\frac{(\mathbf{S}_{it} - \mathbf{M}_{ii})}{\tau} \right), \quad \text{and}$$

$$d_{ijt} = \begin{cases} \| \mathbf{M}_{ij} - \hat{g}(\mathbf{S})_{jt} \|_1 & \text{if} \quad \mathbf{M}_{ij} - \hat{g}(\mathbf{S})_{jt} < 0 \\ (\mathbf{M}_{ij} - \hat{g}(\mathbf{S})_{jt})^2 & \text{if} \quad \mathbf{M}_{ij} - \hat{g}(\mathbf{S})_{jt} \geq 0 \end{cases}$$

and τ is a hyperparameter. In our experiments, $\tau = 10$. Recall that the output of $\hat{g}$ is a matrix of translated RSSI and $\hat{g}(\mathbf{S})_{jt}$ denotes the entry corresponding to the jth beacon at time index t.

Our function $\hat{g}$ must map the matrix of RSSI received by any smartphone model to the corresponding RSSI received by a known smartphone model $\in \mathcal{J}$. Our proposed loss SSL quantifies the deviation of the output of $\hat{g}$ from the expected statistics of smartphone models $\in \mathcal{J}$. Specifically, $\mathcal{L}_{ssl}$ provides an exponential weighting (w_{it}) to any deviation (d_{ijt}) from the expected RSSI of the jth beacon when the ith beacon is detected, according to pairwise statistics $\mathbf{M}_{ij}$ of known smartphone models $\in \mathcal{J}$. When $\hat{g}$ is learned correctly the statistical deviation of the translated output signals should be low.

We softly encourage the corrected readings to have a higher mean, by giving an L1 penalty when readings are higher than the desired statistic. The insight

here is that having higher RSSI BLE generally helps in better localization performance, even for the known phones.

4. **Temporal smoothness loss:** One of the major problems we observed in lower-end smartphone models is lost RSSI information where beacons randomly measure zero RSSI. In order to address this problem, we introduce a temporal smoothness loss that ensures that the output of the translation function $\hat{g}$ does not randomly drop RSSI,

$$\mathcal{L}_{\text{ts}} = \sum_{t=1}^{H-1} \sum_{i=1}^{B} \|\ \hat{g}(\mathbf{S})_{i,t} - \hat{g}(\mathbf{S})_{i,t+1}\ \|_1.$$

$\hat{g}(\mathbf{S})_{i,t}$ is the entry corresponding to the i$^{\text{th}}$ beacon at time index t.

4.2.1   Optimization:

We propose a modified optimization with weights w_{loc}, w_{ps}, w_{ssl}, w_{ts} on each term respectively:

$$\min_{\hat{h},\ \hat{g}} \mathbb{E}_{\langle \mathbf{S}, \mathbf{x} \rangle \sim \mathcal{D}'_l} \left[w_{\text{loc}} \mathcal{L}_{\text{loc}} + w_{\text{ps}} \mathcal{L}_{\text{ps}} + w_{\text{ssl}} \mathcal{L}_{\text{ssl}} + w_{\text{ts}} \mathcal{L}_{\text{ts}} \right] \tag{4.2}$$

By training $\hat{h}$ and $\hat{g}$ to minimize this objective, we expect generalization of localization performance to unseen smartphones. The localization network $\hat{h}$ has the same architecture as `LocNet` detailed in the previous scenario. We now describe the signal translation network $\hat{g}$.

4.2.2   Signal Translation Network (`TransNet`):

Our signal translation function $\hat{g}$ transforms the matrix of RSSI measured by a phone $p \in \mathcal{P}$, to the matrix of RSSI measured by a model of a known brand in the same location. Intuitively, we expect the difference in RSSI measured by different phones at the same location to be bounded. Therefore, we constrain $\hat{g}$ to output a correction to its input. Formally, our signal translation function is a neural network of the form $\hat{g}(\mathbf{S}) = \text{ReLU}(\hat{r}(\mathbf{S}) + \mathbf{S})$. We call this network `TransNet`. The correction $\hat{r}(\mathbf{S})$ is a 6-layered 1-D convolutional autoencoder, composed of 3 encoder layers and 3 decoder

layers with ReLU activations.

4.3 Scenario 3: Limited Labeled and Unlabeled Data

We now describe our third DeepBLE approach for the scenario in which RSSI measurements for new smartphones (not contained in the labeled data) can be collected as people walk around the indoor environment. However, this data is not annotated with the smartphone's true location, *i.e.,* the data is unlabeled. In this scenario, we have access to limited labeled data $\mathcal{D}'_l$ from smartphone models $\mathcal{P}' \subset \mathcal{P}$, as well as limited unlabeled data $\mathcal{D}'_u$ from other smartphone models $\mathcal{Q}' \in \{\mathcal{P} \setminus \mathcal{P}'\}$.

4.3.1 Optimization:

Like the previous scenario, we consider the estimation of $\hat{h}$ and $\hat{g}$, so that the localization function $\hat{f} = \hat{h}(\hat{g})$. Intuitively, using the additional unlabeled data, we can further tune the translation function $\hat{g}$ to deal with changes in the input distribution. For example, some smartphones may exhibit more RSSI signal dropping. Training $\hat{g}$ and $\hat{h}$ to deal with such changes further equips our localization function $\hat{f}$ to generalize to unseen smartphones. While the previous two scenarios involved a supervised approach to learning $\hat{f}$, in this scenario, we describe a **semi-supervised** approach:

1. Using $\mathcal{D}'_l$, we first perform optimization (4.2) in a supervised manner, to obtain initial estimates $\hat{h}$ and $\hat{g}$.

2. Once we have initial estimates of $\hat{h}$ and $\hat{g}$, we use both labeled data $\mathcal{D}'_l$ and unlabeled data $\mathcal{D}'_u$ from new smartphones $\mathcal{Q}'$ to improve our estimates. Notice that, in optimization (4.2), only the localization loss $\mathcal{L}_{loc}$ depends on the label **x**. Thus, we define a modified objective function for the unlabeled data by adding loss functions that do not depend on the label. We use weights w_{loc}, w_{ps}, w_{ssl}, w_{ts} as before, as well as a weight w_u on the expected loss over the

unlabeled data:

$$\min_{\hat{h}, \hat{g}} \quad \mathbb{E}_{\langle \mathbf{S}, \mathbf{x} \rangle \sim \mathcal{D}'_l} \Big[w_{\mathrm{loc}} \mathcal{L}_{\mathrm{loc}} + w_{\mathrm{ps}} \mathcal{L}_{\mathrm{ps}} + w_{\mathrm{ssl}} \mathcal{L}_{\mathrm{ssl}} + w_{\mathrm{ts}} \mathcal{L}_{\mathrm{ts}} \Big] +$$

$$w_u \, \mathbb{E}_{\mathbf{x} \sim \mathcal{D}'_u} \Big[w_{\mathrm{ps}} \mathcal{L}_{\mathrm{ps}} + w_{\mathrm{ssl}} \mathcal{L}_{\mathrm{ssl}} + w_{\mathrm{ts}} \mathcal{L}_{\mathrm{ts}} \Big] \qquad (4.3)$$

By training $\hat{h}$ and $\hat{g}$ to minimize this objective, we expect better generalization of localization performance to unseen smartphones. The localization network $\hat{h}$ and signal translation network $\hat{g}$ have the same architecture as `LocNet` and `TransNet` detailed in the previous scenarios.

Chapter 5

Dataset

Model	Price	OS	Processor
Apple iPhone XR	$600	iOS 13	A12 Bionic
Apple iPhone 8	$450	iOS 13	A11 Bionic
Apple iPhone 7	$190	iOS 13	A10 Fusion
Samsung Galaxy S10	$700	Android 10	SM8150 SD-855*
Samsung Galaxy Note9	$600	Android 9	SDM845 SD-845*
Samsung Galaxy A50	$300	Android 9	Exynos9610
Google Pixel 4XL	$900	Android 10	SM8150 SD-855*
Google Pixel 3XL	$550	Android 10	SDM845 SD-845*
Google Pixel 3aXL	$400	Android 10	SDM670 SD-670*
Xiaomi Mi 9	$435	Android 9	SM8150 SD-855*
Xiaomi Mi 9T Pro	$360	Android 9	SM8150 SD-855*
Xiaomi Redmi Note 8	$180	Android 9	Mediatek Helio G90T
Huawei Mate 20 Pro	$550	Android 9	HiSilicon Kirin 980
Huawei Honor 20 Pro	$340	Android 9	HiSilicon Kirin 980
Huawei Honor View 20	$290	Android 9	HiSilicon Kirin 980

Table 5.1: Specs of Smartphone models used as BLE Receivers. *Qualcomm Snapdragon
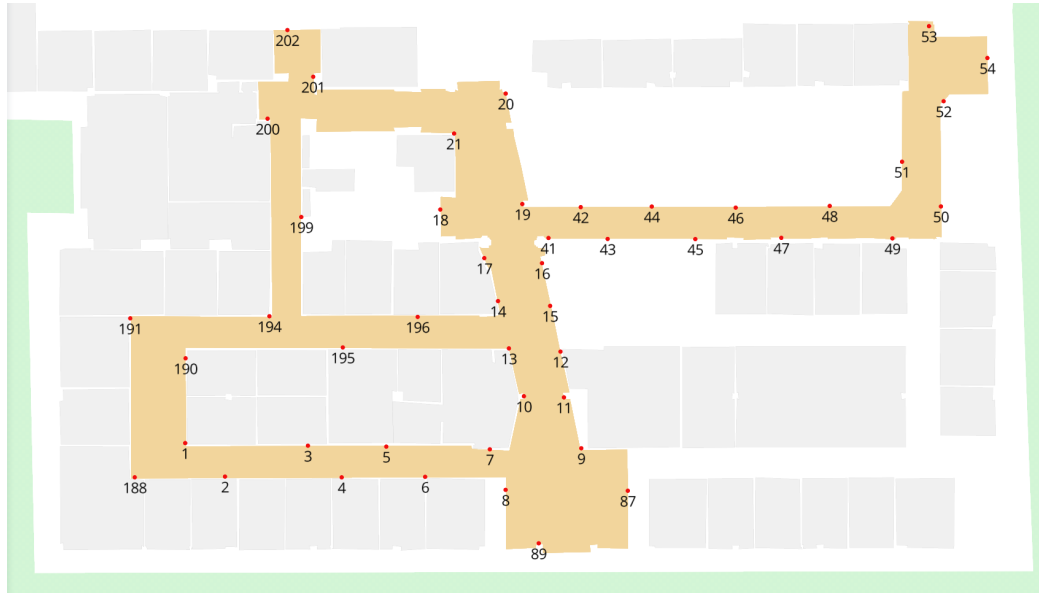
Figure 5.1: The colored region is the area we collect data in. The red dots show the beacons, along with the minor ids

Indoor localization using RSSI has been gaining momentum in the last few years, but the scarcity of large scale public datasets on BLE Localization has been a problem for researchers working toward this problem. To the best of our knowledge, [25] is the only publicly available BLE RSSI dataset. The dataset was collected using iPhone6S, receiving BLE RSSI from 13 different iBeacons deployed in the Waldo Library of Western Michigan University. The dataset contains two sub-datasets: a labeled dataset (1420 instances) and an unlabeled dataset (5191 instances). As we try to analyze how varying phone types, change RSSI signals, the data cannot be used for our analysis.

So towards this end, we collected a large scale BLE dataset with about 54K RSSI fingerprint samples along with fine-grained location information, for 15 smartphone models. The setup was deployed across a university floor spanning 2000 sq. m. in area. The dataset comprises of RSSI data ranging from Kontakt.io beacons for fifteen different phones from 5 major brands which include Apple, Samsung, Pixel, Huawei, and Xiaomi. In this section, we describe the data collection procedure in detail.

5.1   Overview

We designed a rig, as shown in 5.1, which can hold three phones and a Kaarta Stencil on top to get precise ground truth locations. Study in [38], suggests BLE data changes significantly with orientations, so having three facing directions helps us in augmenting the BLE data substantially at every single location. About 50 Kontakt.io beacons were installed all across the first floor of a university, as shown in 5.1. One of the factors that influence noise in RSSI systems is multipath fading, which often makes localization using BLE a lot more difficult. Therefore, the complex structure of the building makes it a perfect testing ground for our purpose.

Figure 5.2: The rig used for data collection

It might be enticing to clamp a large number of phones on a rig, to get data

for many different phones together. To do so, we designed a rig which can hold 15 phones together; however, the data received by different phones changes compared to RSSI received by a smartphone as a person is walking. Also, the number of beacons detected by each smartphone decreases significantly. For Huawei and Xiaomi, less than five beacons were recorded compared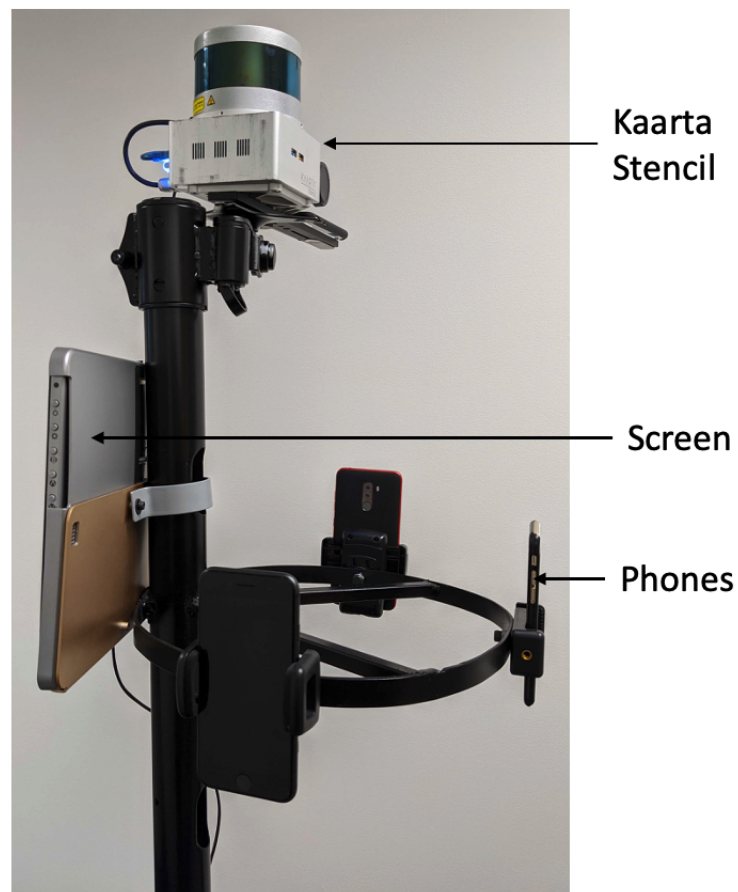 to 17-18 beacons when used separately. These limits in such extreme cases boil down to available bandwidth, transmission duty-cycle and physical proximity. If all the devices plan to talk to one 'hub', then the potential will be drastically reduced. Also, it is worth mentioning that although the theoretical limits and guarantees mentioned in the BLE protocol state a transmission rate of 1Mbps, the packets generally transmit in real-world scenarios at somewhere close to 115Kbps, increasing the packet loss to a high degree and limiting the use case when there are way too many devices using BLE.

5.2   Obtaining Ground truth Locations

Typically fingerprinting is performed standing at specific reference locations, collecting the RSSI and then using the BLE data at these reference locations to localize. However, as the data collection procedure is static, we cannot expect the methods to work very well as the robot/person is moving. However, collecting ground-truth as the robot moves requires high-end infrastructure in the building, making it extremely infeasible for scaling such methods. Stencil by Kaarta is a stand-alone, lightweight and low-cost system delivering the integrated power of mapping and real-time position estimation with a precision of ±30mm. The device is based on the scientific work [36] and depends on LIDAR, vision and IMU data for localization. The system uses Velodyne VLP-16 connected to a low-cost MEMS IMU and a processing computer running Robotic Operating System (ROS) for real-time six DoF mapping and localization. A 10-Hz scan frequency is used for the data capture VLP-16 has a 360° field of view with a 30° azimuthal opening using 16 scan lines. The stencil tilt angle is recommended to be within the ±15° envelope. The progress of the mapping can be monitored on-line via an external monitor attached via an HDMI cable.[19]. For robust estimates, we use the multi-modal localization engine, which gives us position estimates using data captured via both camera and LIDAR.

22

5.3 Transmitters - BLE Beacons

For our analysis and experiments, we used beacons manufactured by Kontakt.io. Some of the key factors and parameters of BLE beacons that can significantly influence localization performance are *Broadcasting Power*, *Advertising Interval*, and *Measured Power*. In our setup these values are -12dBm, 100ms and -77 dBm respectively. We refer readers to [1] for detailed analysis and impact of each of these terms.

5.4 Receivers - Commodity Smartphones

We used fifteen different smartphone models from 5 notable brands, with their price ranging between $250 to $ 900 (Table 5.1) as receivers. Broadly, choice in every brand comprises of one lower-end phone, one middle-end phone and one high-end smartphone. The detail and specs for every phone are consolidated in table 5.1. Note that every phone used has BLE version 5, except iPhone 7, which is using a version 4.2.

5.5 API for accessing BLE Beacon RSSI data

iOS (on iPhones) has an inbuilt API to log the beacon data being sensed. However, we note that these readings are processed, and the maximum frequency with which one can access the data is 1 Hz. For Android we tried using the Android Beacon Library for data collection. however we observed a huge amount of packet loss. We believe the application is sub-optimally designed for large scale Beacon data collection and hence we built our in-house Android Beacon logging library that logs beacon scans at a consistent rate of 1Hz with zero packet loss in processing. 5.3 shows the difference between the readings logged by Android Beacon library and our library.

5.6 Time Synchronization

We match the data recorded from the different phones with the ground truth from the stencil using timestamps recorded at both ends. In order to do this accurately,
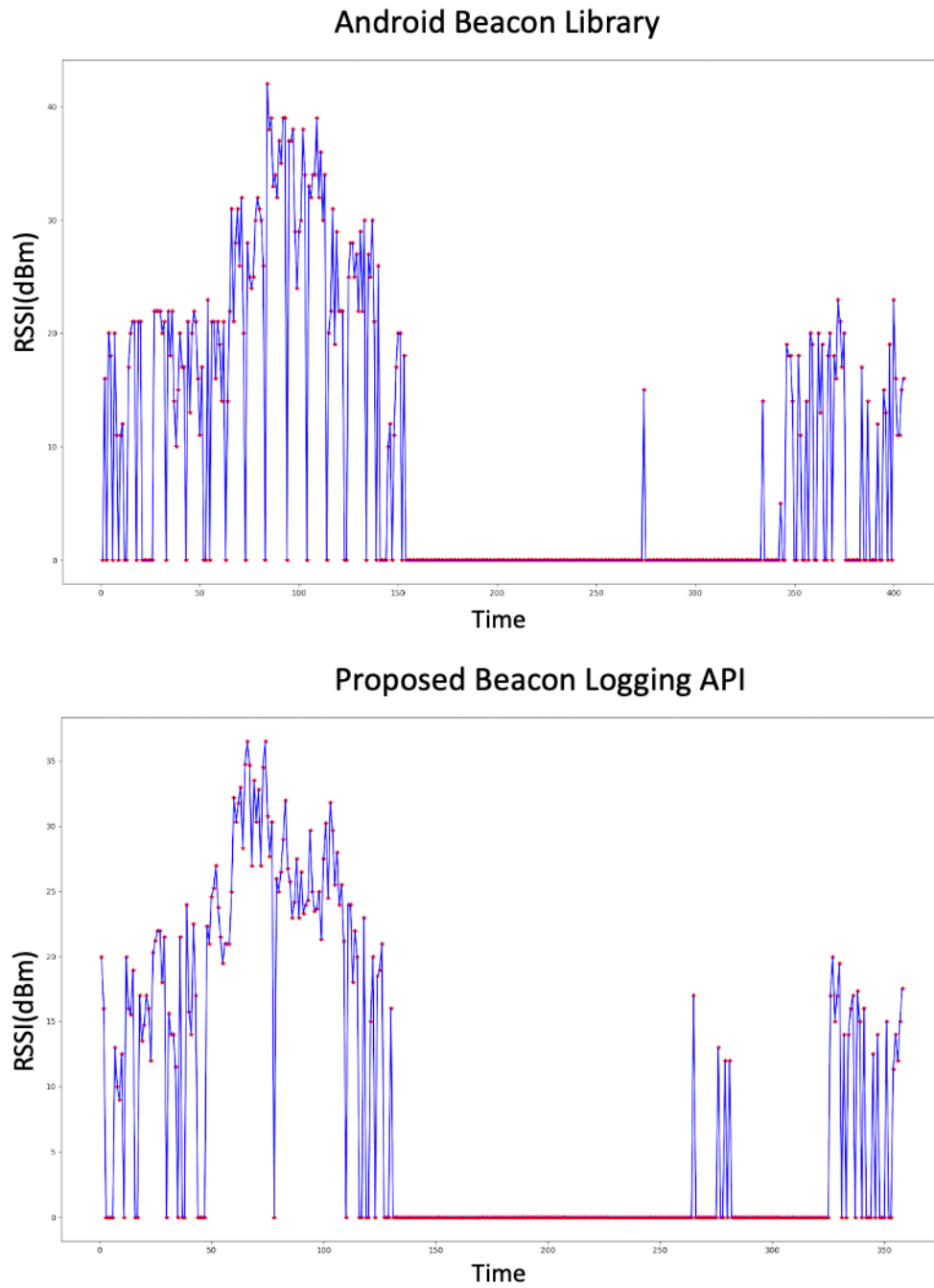
Figure 5.3: Comparison of RSSI BLE recorded by Android Beacon Library and our proposed API

the time on the devices need to be synced. We perform NTP time sync on both the stencil and the phones at the beginning of every run using the same NTP server [1] . We empirically observed the sync to be precise to a few microseconds.

5.7 Data Collection Procedure

We place Kontakt.io beacons across the floor of a university building, as shown in Figure 5.1(b). The rig is designed to hold three phones (Figure 5.1(a)). We group the phones by brand and collect around 60 mins of data for each brand. The colored region Figure 5.1(b) is the regions where we perform localization. To make sure the comparison between the phones is fair, and the beacon signals have not changed drastically between different brands, any given day we data is collected for all brands by the same person. This process is repeated on multiple days to collect the complete data set. The data has been collected, during the after-hours in the evening, so that noise due to external factors is subsidized, and we can focus on change in RSSI distributions introduced by varying smartphone models. For different runs, we ensure that the gait style and walking pattern does not vary a lot between brands. As we collect different runs of data, we observe that as the frequency of BLE data reception is low (typically 1 Hz), the BLE data received does not change drastically from person to person for normal walking speeds (1-1.2m/s). Expecting absolutely no change between different runs is not possible. However, the best efforts were made to collect the data to be as similar as possible to a person walking with a phone in hand, with almost all factors we can control remaining the same except the smartphone models.

5.8 Train Test Validation Split

We refer to the collected Dataset as $\mathcal{D}_c$ which we split into the train set ($\mathcal{D}_{c,\text{train}}$), validation set ($\mathcal{D}_{c,\text{val}}$), and test set ($\mathcal{D}_{\text{test}}$). Each of these sets are independent runs, with no overlapping samples. We note that through out our analysis, $\mathcal{D}_{\text{test}}$ remains untouched and is used only for final evaluation.

[1]NTP server used: time.google.com

Env. →	No Interference			W/ Interference		
Phone↓	Mean	Var	SnR	Mean	Var	SnR
iPhone 7	27.61	7.18	3.83	26.80	4.07	6.59
iPhone 8	30.20	1.73	17.49	27.52	8.25	3.34
iPhone XR	29.17	2.26	12.87	27.93	0.37	76.47
Galaxy S10	33.04	8.09	4.09	20.87	14.77	1.41
Galaxy Note9	30.30	2.86	10.58	22.35	40.92	0.55
Galaxy A50	30.83	17.93	1.72	26.80	19.03	1.41
Pixel 4XL	24.46	11.47	2.13	11.83	41.93	0.28
Pixel 3XL	30.35	0.49	62.22	16.41	42.72	0.38
Pixel 3aXL	28.98	3.20	9.07	20.61	50.37	0.41
Mi 9	27.39	45.90	0.60	19.11	41.77	0.46
Mi 9T Pro	30.39	55.79	0.54	19.12	62.34	0.31
Redmi Note 8	32.33	4.35	7.43	22.50	43.95	0.51
Mate 20 Pro	34.67	1.39	24.88	26.33	21.87	1.20
Honor 20 Pro	28.04	11.91	2.35	21.07	35.63	0.59
Honor View 20	26.09	3.99	6.53	25.17	7.06	3.57

Table 5.2: Per phone signal statistics of one beacon with (47 other beacons) and in absence of other beacons. RSSI distribution can change significantly across phones.

5.9 BLE RSSI Analysis

Here we perform simple empirical analysis to understand how RSSI data received differs between different smart phones.

5.9.1 RSSI without Interference

In an isolated environment, we consider one beacon and one smartphone at a time. The BLE beacon and smartphone are kept at a distance of 1m, and RSSI data is collected for 60 seconds for each phone sequentially. We calculate the mean and the variance of the RSSI signal for each phone using the signal strengths registered during the 1-minute interval. The numbers are reported in 5.2. Although the mean RSSI being registered across different phones is approximately the same, the variance changes drastically across phones. For particular phones, signal drops are observed. For instance, for Mi 9T Pro, we observe the beacon was not registered for 3 seconds continuously.
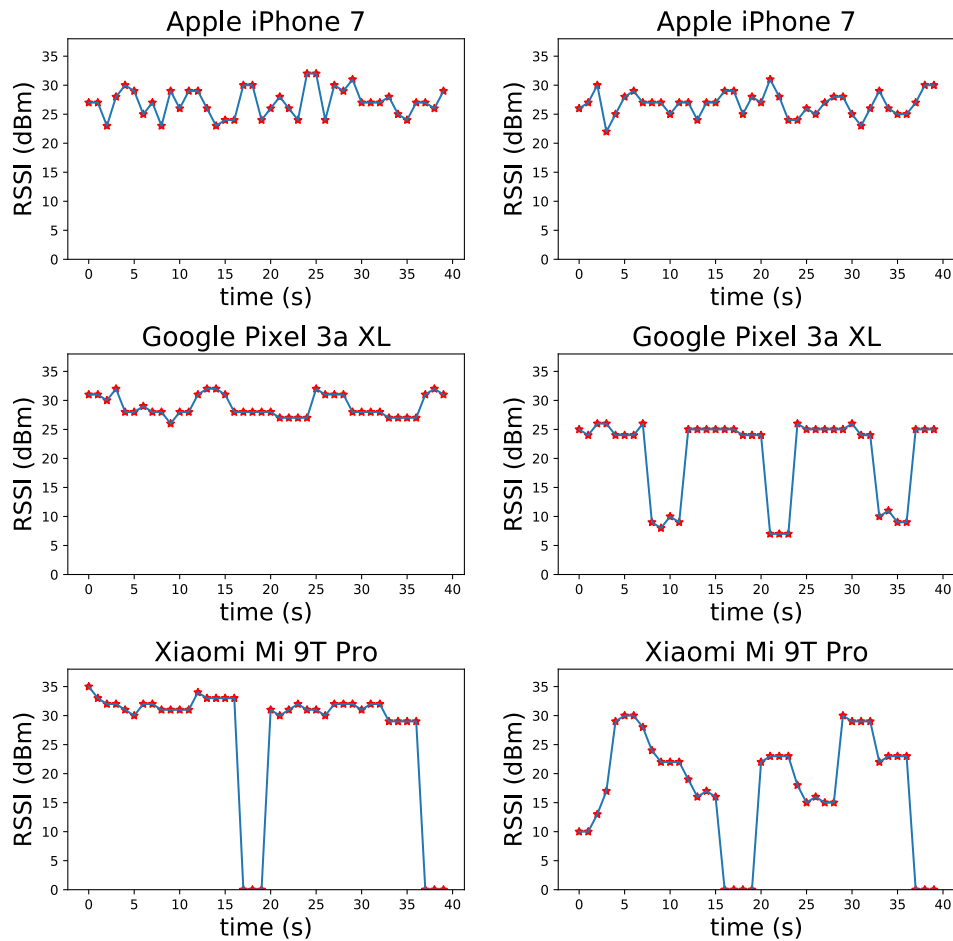
Figure 5.4: RSSI vs. Time plots for the two experimental setups show that having multiple beacons in the environment has little impact on the nature of the signal from iPhone, some impact on the Pixel phone, but major impact in case of the Xiaomi phone. We see significantly higher instability in Experiment 2

5.9.2   RSSI with Interference

We repeat a similar experimental setup, with the same beacon, but in the presence of multiple other beacons to act as sources of interference. According to [22], we know multi-path fading, and interference due to multiple beacons can lead to a significant change in RSSI behavior. The numbers are reported in 5.2, under the "W/Interference" heading. The experiments were performed in the building, where the dataset has been collected. We observe in addition to the variance, in this case, the mean RSSI changes significantly across phones, contrary to what we observed in the no interference setting. Also, we note that the variance (noise) is not directly related to what we observed in the absence of other sources of interference. For instance, Pixel3XL records a significant variance in this setup, and iPhone XR shows a remarkably stable reading here (5.4). The differences we observe can be due to multiple factors, which include multi-path fading and interference due to multiple beacons. Empirically we conclude, that the behaviour of RSSI distribution can change significantly across phones in the presence of other sources of interference.

5.9.3   RSSI Receiver Failure Statistics

A troubling phenomenon that we encountered were frequent occurrence of receiver failure where all RSSI measurements become zero at random time instances.

We compute the following metrics to quantify receiver failure:

1. Non Zero Mean RSSI - The average RSSI strength that a phone detects for RSSI measurements greater than -100 dBm

2. Receiver Failure - Percentage of samples in the train data when no beacons were detected

3. Dead time - Mean duration of such Receiver failure incidents.

These metrics obtained for each phone are reported in Table 5.3. Note that receiver failures happen about 15% of the time for Xiaomi models and can last roughly from 1 to 3 seconds. These statistics suggest that the translation network that we implement needs to be able to handle significant receiver failures where all beacons RSSI measurements are completely missing.

To provide a more qualitative visualization of the dead time issue, we had a

Phone Name	Mean RSSI(dBm)	Receiver Failure(%)	Dead time(s)
iPhone 7	12.09	0.00	0.00
iPhone 8	12.84.	0.00	0.00
iPhone XR	11.64	0.00	0.00
Huawei Mate 20 Pro	9.28	9.98	1.45
Huawei Honor 20 Pro	8.28	8.52	1.51
Huawei View 20	7.389	14.96	1.55
Xiaomi Mi9	8.27	15.18	3.20
Xiaomi Mi9TPro	10.03.	15.11	3.19
Xiaomi Redmi Note 8	. 10.03	7.88	1.00
Google Pixel 3aXL	9.68	0.00	0.00
Google Pixel 3XL	10.00	0.00	0.00
Google Pixel 4XL	8.63	0.00	0.00
Samsung S10	13.18	0.16	1.73
Samsung A50	10.25	0.17	1.31
Samsung Note9	9.25	0.16	1.32

Table 5.3: RSSI Receiver Failure Statistics

subject walk with two phones (one in each hand), across the building recording RSSI from 47 possible beacons on both phones. We plot the RSSI vs. Time plot for a particular beacon ($b = 11$) for the Apple iPhone 7 and Xiaomi Mi9T Pro shown in Figure 5.5. Notice that the Mi9T Pro has many instances of signal dropping (no RSSI measurements) over time. In contrast, the iPhone 7 has a strong RSSI signal throughout.

5.9.4 t-SNE Analysis

To visualize mathematically, how RSSI data from different phones differ, we train a simple classifier network with one phone from each brand and get the 3 dimensional t-SNE reduction to features of the last layer. 5.6 shows the plot so obtained. The plot suggests that each phone can be clustered in local regions and indeed the distribution for RSSI from each phone is different from one another. Inevitably, if we train our Machine Learning algorithm naively to fit one particular phone distribution, it won't be able to generalize across other phones.

Apple iPhone 7

Xiaomi Mi 9T Pro
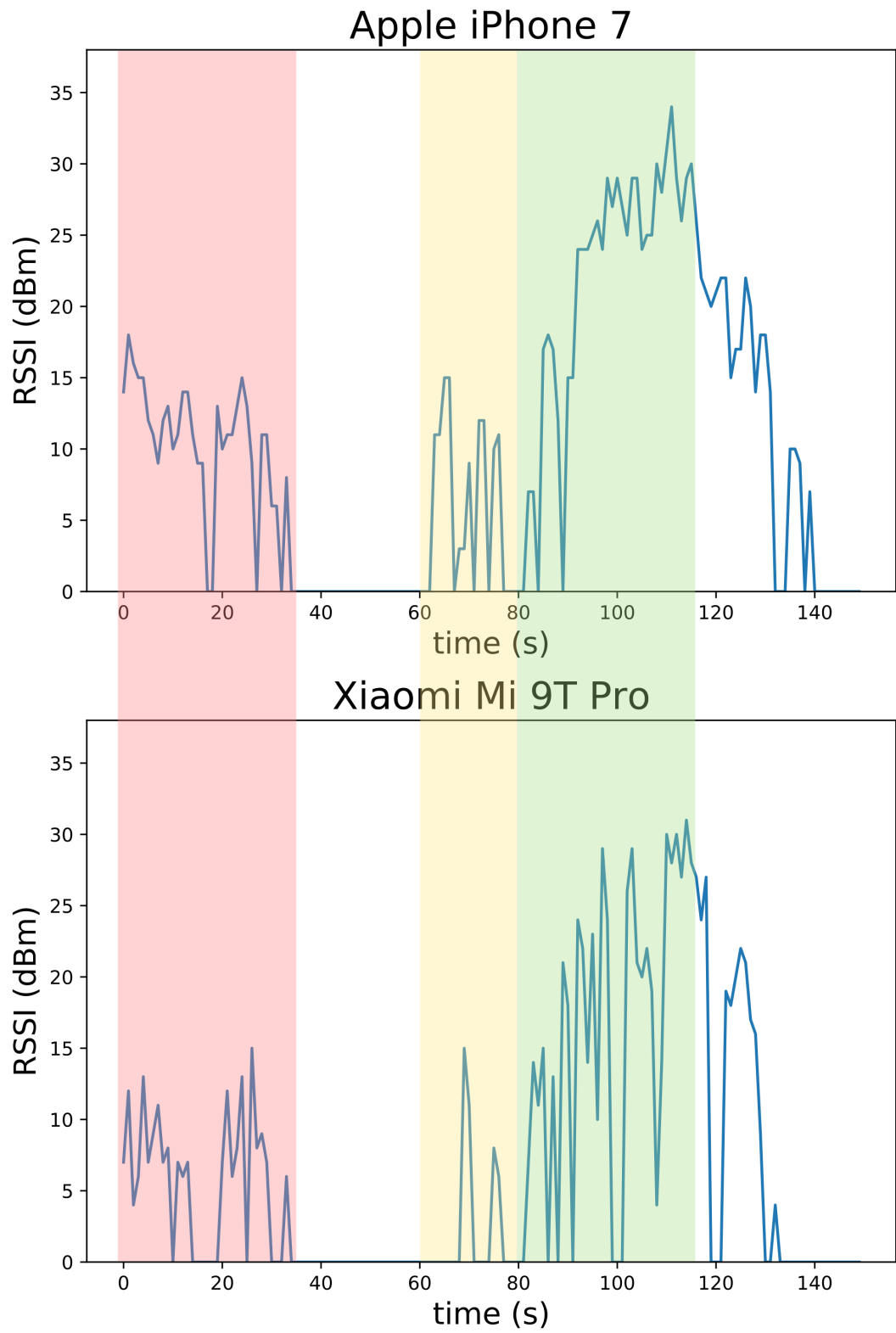
Figure 5.5: RSSI by iPhone 7 vs. RSSI by Xiaomi
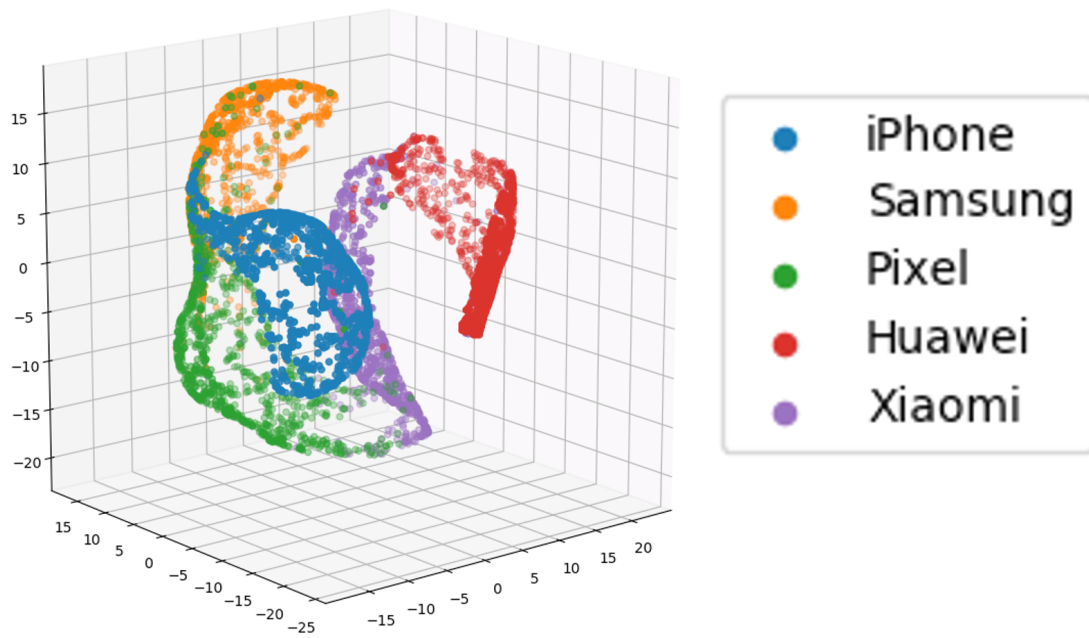
Figure 5.6: t-SNE plots for classification features from each phone

Chapter 6

Experiments

In this chapter we consider the experiments performed with our proposed approach and our comparisons with commonly used and state of the art methods for localization. But before we

6.1 Experiments: DeepBLE Methods

In this section, we present experiments performed using the DeepBLE methods developed in Section 4. We implement all methods using PyTorch [28]. Before detailing experiments for different scenarios, we describe our augmentations to the collected data $\mathcal{D}_{c,\text{train}}$ and the resulting labeled data set $\mathcal{D}_l$. We also provide details of the evaluation metrics used.

Data augmentation:

Based on the analysis of RSSI across different smartphone models (Section 5.9), we augment the collected train data $\mathcal{D}_{c,\text{train}}$ as follows:

- **Scaling the RSSI strengths**: Dependence of beacons on batteries often leads to a decrease in RSSI transmitted by a beacon with time. To replicate that effect, we scale down the BLE input $\mathbf{S}$ at multiple scales.
- **Imitating packet loss**: Given only three channels for transmission in BLE, Packet Loss is often observed when there are too many beacons in an environ-

Scenario	Scenario 1		Scenario 2					
	Proposed		Baseline 1		Baseline 2		Proposed	
Test Phone	Mean	Std	Mean	Std	Mean	Std	Mean	Std
iPhone 7	0.95	0.65	0.99	0.83	0.92	0.83	**0.91**	**0.75**
iPhone 8	1.09	0.73	1.13	0.95	1.16	0.99	**1.06**	**0.76**
iPhone XR	0.92	0.60	1.05	0.83	0.95	0.75	**0.84**	**0.60**
Mate 20 Pro	1.96	1.61	4.19	4.25	2.62	2.45	**1.76**	**1.61**
Honor 20 Pro	1.92	1.46	3.14	2.19	2.36	1.77	**1.95**	**1.42**
View 20	1.74	1.40	3.15	3.31	2.20	2.04	**1.64**	**1.59**
Mi 9	1.53	1.20	3.82	5.48	2.39	3.30	**1.69**	**1.57**
Mi 9T Pro	1.77	1.14	3.78	4.98	2.71	3.35	**1.78**	**1.41**
Redmi Note 8	1.50	1.17	2.25	2.01	1.76	1.57	**1.62**	**1.16**
Pixel 3a XL	1.05	0.83	1.51	1.30	1.25	1.12	**1.24**	**1.07**
Pixel 3XL	1.21	0.96	2.04	1.56	1.54	1.29	**1.22**	**0.95**
Pixel 4XL	1.21	0.83	2.13	1.64	1.53	1.28	**1.36**	**1.01**
S10	1.44	1.15	1.80	1.41	1.58	1.35	**1.31**	**1.05**
A50	1.49	0.98	2.11	1.68	1.62	1.34	**1.42**	**1.30**
Note9	1.14	0.87	1.67	1.23	1.27	0.98	**1.13**	**0.82**
Overall Average	1.37	1.11	2.23	2.76	1.67	1.83	**1.37**	**1.20**

Table 6.1: Mean and std. deviation of absolute localization error for all methods and scenarios. All numbers are in meters(m). Numbers in bold indicate best performance within each scenario. Numbers in color indicate best performance overall.

ment. We imitate such behaviour by adding random signal drops in the input signals in a constrained fashion.

- **Random Noise**: As the beacon data being received changes with scene dynamics, we add Gaussian white noise sampled from $\mathcal{N}(0, 5)$ to our BLE input.

We refer to the augmented data as $\mathcal{D}_A$. Thus, the labeled data from all smartphone brands is $\mathcal{D}_l = \mathcal{D}_{c,\text{train}} \cup \mathcal{D}_A$.

Evaluation:

In all scenarios, our methods are evaluated using $\mathcal{D}_{\text{test}}$. All reported statistics are with respect to absolute localization error (AE) of the learned localization function $\hat{f}$

Scenario	Scenario 3			
	Proposed (Huawei)		**Proposed** (Xiaomi)	
Test Phone	Mean	Std	Mean	Std
iPhone 7	0.95	0.88	**0.94**	**0.82**
iPhone 8	1.11	**0.83**	**1.07**	0.83
iPhone XR	0.89	0.80	**0.86**	**0.63**
Mate 20 Pro	**1.63**	1.32	**1.63**	**1.31**
Honor 20 Pro	1.99	**1.33**	**1.92**	1.34
View 20	1.66	1.32	**1.64**	**1.29**
Mi 9	1.65	1.31	**1.58**	**1.11**
Mi 9T Pro	1.84	1.45	**1.68**	**1.20**
Redmi Note 8	**1.61**	1.09	**1.61**	**1.06**
Pixel 3a XL	1.22	1.07	**1.21**	**1.06**
Pixel 3XL	**1.17**	0.96	1.18	**0.94**
Pixel 4XL	**1.39**	**1.07**	1.40	1.08
S10	1.30	1.02	**1.23**	**0.93**
A50	**1.36**	**1.22**	1.42	1.26
Note9	**1.15**	**0.81**	**1.15**	0.82
Overall Average	1.37	1.15	**1.34**	**1.09**

Table 6.2: Mean and std. deviation of absolute localization error for all methods and scenarios. All numbers are in meters(m). Numbers in bold indicate best performance within each scenario. Numbers in color indicate best performance overall.

on test sample $\langle \mathbf{S}, \mathbf{x} \rangle \sim \mathcal{D}_{\text{test}}$:

$$\text{AE}(\mathbf{S}, \mathbf{x}) \;=\; \left\| \hat{f}(\mathbf{S}) - \mathbf{x} \right\|_2$$

We now discuss our methods and results for each scenario in detail. For all methods, Table 6.1 represents mean and standard deviation of absolute localization error over test data from each smartphone model, as well as over all test data $\mathcal{D}_{\text{test}}$. For all methods, Table 6.3 represents mean, standard deviation, median, 90 percentile and maximum absolute localization error over all test data $\mathcal{D}_{\text{test}}$.

Stat. ↓	Scen. 1	Scen. 2			Scen. 3	
	Prop.	Base. 1	Base. 2	Prop.	Prop. (Hua.)	Prop. (Xia.)
Mean	1.37	2.23	1.67	1.37	1.37	**1.34**
Std. Dev.	1.11	2.76	1.83	1.20	1.15	**1.09**
Median	1.07	1.47	1.15	**1.02**	1.06	1.04
90 %ile	2.68	4.55	3.49	2.76	2.76	**2.66**
Max.	12.28	37.58	29.67	19.72	13.19	**11.82**

Table 6.3: Various statistics of absolute localization error over all test data for all methods and scenarios. All numbers are in meters (m). Numbers in bold indicate best performance over all methods.

6.1.1 Scenario 1: Labeled Data for All Smartphone Models

As discussed in Section 4, in our first scenario, we have labeled data from all smartphone models, *i.e.*, $D_{\text{train}} = \mathcal{D}_l$. We train the localization network `LocNet` for objective (4.1) in Section 4, using the Adam optimizer for 50 epochs, with a learning rate $= 10^{-4}$ and batch size = 256 samples.

Results:

We report mean and standard deviation of absolute localization error over test data in Table 6.1. Note, we do not have any numbers in bold here, since we do not compare our proposed method with any other method in this scenario. In this ideal scenario, since we have labeled data for all phones, performance on all phones is very good, with an average AE of **1.37 m** over all test data. Despite the high amounts of receiver failure for smartphone models from Huawei and Xiaomi brands (Section 5.9), AE for each of these models is **less than 2 m**. However, as noted before, it is unrealistic to assume that we can performing fingerprinting for all smartphone models.

6.1.2 Scenario 2: Limited Labelled data

In our second scenario, we have only limited labeled data $\mathcal{D}_{\text{train}} = \mathcal{D}'_l$ from a subset of smartphone models $\mathcal{P}' \subset \mathcal{P}$. In our experiments with this scenario, $\mathcal{P}'$ includes models

of the two most popular smartphone brands [1] - Apple and Samsung, *i.e.,* {iPhone 7, iPhone 8, iPhone XR, Samsung S10, Samsung A50, Samsung Note 9}. Our signal translation function $\hat{g}$ learns to transform RSSI measured by any smartphone model to that measured by the Apple brand, *i.e.,* $\mathcal{J} \subset \mathcal{P}'$ includes smartphone models from the Apple brand {iPhone 7, iPhone8, iPhone XR}.

Experiments were also performed considering $\mathcal{P}'$ to have { Apple, Huawei }, { Apple, Xiaomi} and {Apple, Pixel}. The performance for all the combinations eventually results in a similar performance, conditioned to independent hyperparameter search for each. Hence for brevity here we show experiments considering $\mathcal{P}'$ with {Apple, Samsung} only.

As detailed in Section 4, our localization function in this scenario is $\hat{f} = \hat{h}(\hat{g})$, where $\hat{h}$ is `LocNet` and $\hat{g}$ is `TransNet`. We train $\hat{f}$ for objective (4.2) in Section 4, using the Adam optimizer for 50 epochs, with a learning rate $= 10^{-4}$ and batch size $= 256$ samples. Appropriate weights for each loss term are obtained using validation data.

Baselines:

We compare our method against the following baselines:

1. Similar to the first scenario, we consider the situation where our localization function $\hat{f}$ is `LocNet` only. We limit the available labeled data to that from smartphone models of the Apple brand only, *i.e.,* $\mathcal{P}'$ includes {iPhone7, iPhone8, iPhone XR}. We train `LocNet` according to optimization (4.1) in Section 4, using labeled data from the Apple brand. We include this baseline to verify our hypothesis that data augmentation alone is not enough to generalize well to unseen phones. We expect our proposed method to outperform this method due to the inclusion of labeled data from an additional smartphone brand.

2. Again, we consider the situation where our localization function $\hat{f}$ is `LocNet` only. However, for a stronger baseline, we consider that we have access to the same data as our proposed method, i.e., $\mathcal{P}'$ includes models from the Apple and Samsung brands. We train `LocNet` according to optimization (4.1) in Section 4 using labeled data from Apple and Samsung brands. We include this baseline

[1]as reported by the IDC - Worldwide Quarterly Mobile Phone Tracker

to verify our hypothesis that simply training `LocNet` using limited labeled data will lead $\hat{f}$ to overfit to the known phone brands. We expect our proposed method to outperform this method due to the inclusion of a signal translation function $\hat{g}$. This is because $\hat{g}$ explicitly learns to translate RSSI measured by a (possibly unknown) smartphone to that measured by a known brand, using our novel statistic similarity loss (SSL). This enables $\hat{f} = \hat{h}(\hat{g})$ to generalize better to unseen phones.

Results:

From Table 6.1, we see that our proposed method significantly outperforms baselines in terms of both mean and standard deviation of AE over all test data, as well as over test data from individual smartphone models. Moreover, our proposed approach performs as well as the fully supervised approach from the first scenario, achieving an average AE of **1.37 m**, despite having access to only limited labeled data! Importantly, this demonstrates its ability to generalize well to unseen smartphone models.

While Baseline 2 performs significantly better than Baseline 1, it performs poorly on unseen smartphone brands Huawei and Xiaomi. By comparison, our proposed approach consistently gives mean AE of **less than 2 m** for models of these brands. Moreover, from Table 6.3, Baseline 2 shows a high maximum error of $\approx 30m$. This supports our hypothesis that training with limited data leads `LocNet` to overfit to known smartphones. Our proposed approach avoids this problem by training an explicit signal translation function $\hat{g}$ using our novel SSL loss.

Baseline 1 performs very well on iPhone models, with mean and standard deviation of AE within 1 m. However, its performance is significantly worse on other smartphone brands, especially Huawei and Xiaomi. Moreover, from Table 6.3, the maximum and 90 percentile errors are the highest in the table. Baseline 1 overfits to a larger degree than Baseline 2, supporting our hypothesis that data augmentation alone is not enough to generalize well to unseen phones.

6.1.3 Scenario 3: Limited Labeled and Unlabeled Data

In our third scenario, we have limited labeled data $\mathcal{D}'_l$ from a subset of smartphone models $\mathcal{P}' \subset \mathcal{P}$. Additionally, we have unlabeled data $\mathcal{D}'_u$ from other smartphone

models $\mathcal{Q}' \subset \{\mathcal{P} \setminus \mathcal{P}'\}$. Our training data is thus $\mathcal{D}_{\text{train}} = \mathcal{D}'_l \cup \mathcal{D}'_u$. Similar to the previous scenario, in our experiments with this scenario, $\mathcal{P}'$ includes models of Apple and Samsung brands. Our signal translation function $\hat{g}$ learns to transform RSSI measured by any smartphone model to that measured by the Apple brand. We consider $\mathcal{Q}'$ to include smartphone models of the Huawei brand or the Xiaomi brand. As detailed in Section 4, our localization function in this scenario is $\hat{f} = \hat{h}(\hat{g})$, where $\hat{h}$ is `LocNet` and $\hat{g}$ is `TransNet`. In this scenario,

1. We train $\hat{f}$ for objective (4.2) in Section 4, using the Adam optimizer for 50 epochs, with a learning rate $= 10^{-4}$ and batch size $= 256$ samples. Appropriate weights for each loss term are obtained using validation data.

2. We then train $\hat{f}$ for the **semi-supervised** objective (4.3) in Section 4, using the Adam optimizer for 20 epochs, with a learning rate $= 10^{-5}$ and batch size $= 256$ samples.

Results:

From Table 6.2, we see that our proposed method trained in a semi-supervised fashion using unlabeled data from the Xiaomi brand leads to better test performance for Xiaomi and Huawei smartphones, when compared to our proposed method in scenario 2. Moreover, our proposed method **outperforms the fully supervised method** in scenario 1, with a mean AE of **1.34 m** and standard deviation AE of **1.09 m** over all test data. Importantly, this result is achieved despite only having access to limited labeled data, demonstrating our proposed method's ability to generalize well to unseen phones. From Table 6.3, our proposed method leads to the best (or nearly the best) performance on all metrics over all the test data.

6.2 Experiments: Common Localization Methods

We verify our proposed approach and compare it with some of the most commonly used and state of the art methods in RSSI based Localization. Particularly, the performance of DeepBLE has been compared to the 1. K-Nearest Neighbours (KNN)[4] 2. Bayesian Estimation [8] 3. NavCog[27] and 4. DL-RNN [5]. For KNN, we perform a weighted KNN regression using k=10, and a weighing function inversely proportional to the
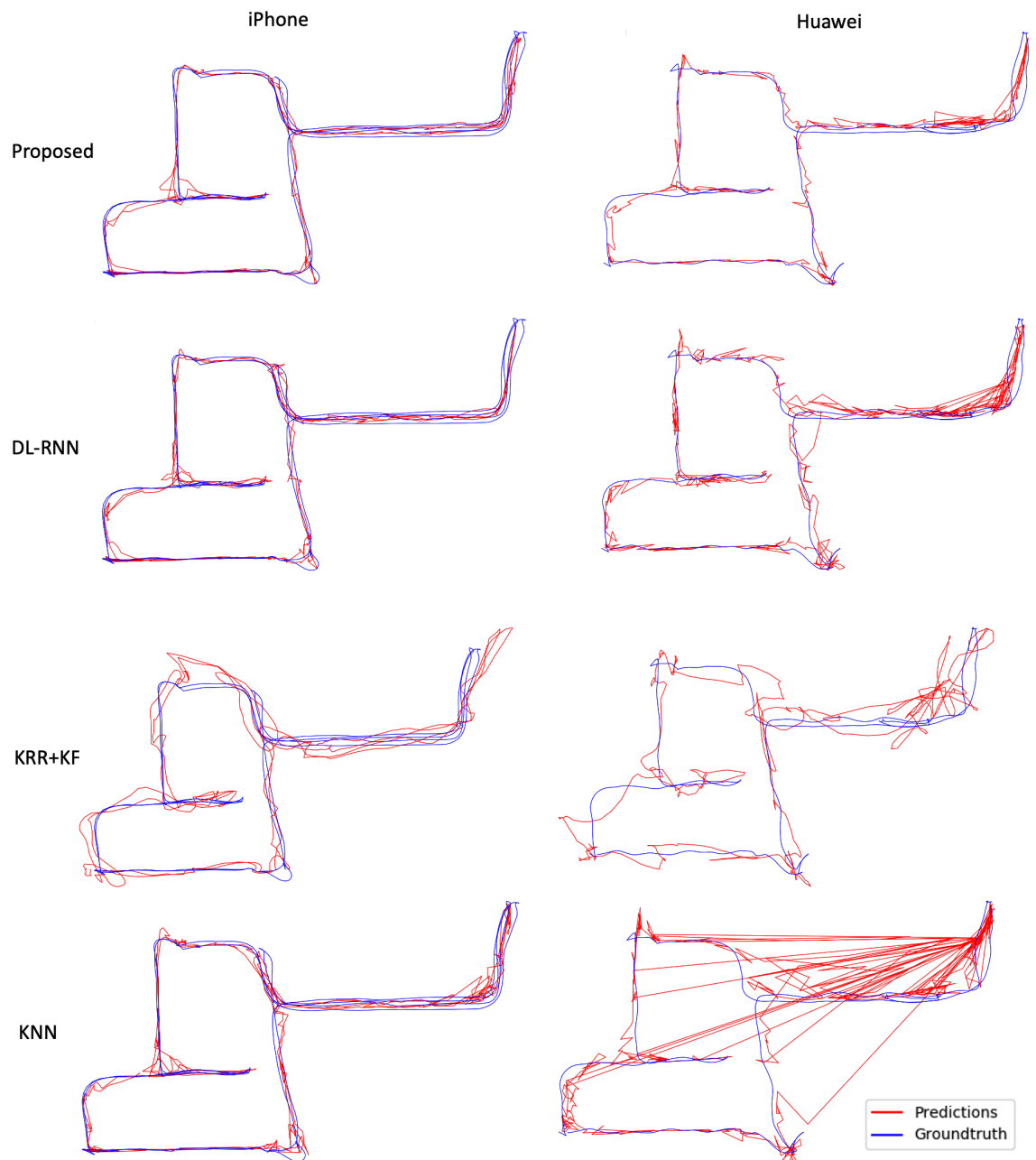
Figure 6.1: Localization Performance comparing different approaches on iPhone7 and Huawei Honor20Pro

Method →	KNN Reg		Bayesian		KRR+KF		DL-RNN	
Test on ↓	Mean	Std	Mean	Std	Mean	Std	Mean	Std
iPhone	1.1917	0.9298	1.5424	1.2267	1.7575	1.0770	1.0205	0.7712
Huawei	6.3579	9.595	4.8951	6.3216	3.0021	2.3075	2.2437	2.2014
Xiaomi	4.1781	7.2262	3.5959	4.2674	2.7569	1.9125	2.4360	3.0378
Pixel	1.9082	1.5561	2.1457	1.7167	1.8955	1.1816	1.4347	1.1638
Samsung	2.0139	2.5525	2.0843	2.3115	2.0767	1.2390	1.5282	1.1301
Average	3.0157	5.7190	2.7724	3.8090	2.2640	1.6644	1.6826	1.8650
Median	1.5230		1.7505		1.8884		1.2067	
90% ile	5.2514		5.5866		4.2660		3.2968	
Max	52.5572		48.3013		17.2977		31.9299	

Table 6.4: Evaluation results for different methods when trained using data from iPhone and Samsung phones

Method →	Proposed(Sup)		Proposed (Semi)	
Test on ↓	Mean	Std	Mean	Std
iPhone	**0.9409**	**0.7163**	0.9572	0.7741
Huawei	1.7900	1.5515	**1.7370**	**1.3260**
Xiaomi	1.7014	1.3918	**1.6284**	**1.1280**
Pixel	1.2769	**1.0183**	**1.2687**	1.0341
Samsung	1.2898	1.0808	**1.2732**	**1.0310**
Average	1.3700	1.2080	**1.3462**	**1.0979**
Median	**1.0258**		1.0436	
90% ile	2.7689		**2.6601**	
Max	19.7231		**11.8294**	

Table 6.5: Evaluation results for different methods when trained using data from iPhone and Samsung phones

distance from each considered neighbour. As [27] uses information from multiple sensors, we make a comparison to the observation model in the particle filter proposed to make a fair comparison to our proposed method as it's observation model depends on only RSSI based Localization. [5] although based on WiFi, solves the same problem of regressing locations directly using deep learning given only RSSI signals, and can be easily deployed for a BLE RSSI setting. Hence we use that as a baseline as well.

For all the methods we use a time history of 5 seconds of BLE beacon RSSI to make a fair comparison between them.

To evaluate each of the methods mentioned above, we calculate the mean and standard deviation of the absolute distance error, as used in 6.1. We calculate this metric on per brand basis as well as across all the phones. We report the median, 90%ile errors and the max for each of the above methods to evaluate the robustness of each method. The comparisons for all the methods have been made under Scenario 2: Access to limited labelled data, as none of these methods can use the unlabelled data in some way.

The numbers obtained are shown in Table 6.5. The proposed approach outperforms all the other methods in terms of the mean and standard deviation of distance errors. More importantly, the disparity between the localization performance shown by phones of different brands is significantly reduced. KNN, which is one of the most commonly used methods, the errors shoot up drastically as we switch phone brands indicating its susceptibility to variation in smartphone model. The Kernel Ridge regression can be a strong competitor, as it is fast and easy to deploy. However, KF is infamous for being very sensitive to the calibration parameters selected. Coming up with an effective way to do that for each phone, is tedious.

Furthermore, the comparison of the 90%ile error complemented with the Distance Error CDF comparison plot in 6.2 indicate that the gap between the proposed approach and others is significant. The DL-RNN is a strong competitor, and it performs very similar to what we see when we train LocNet using limited data.
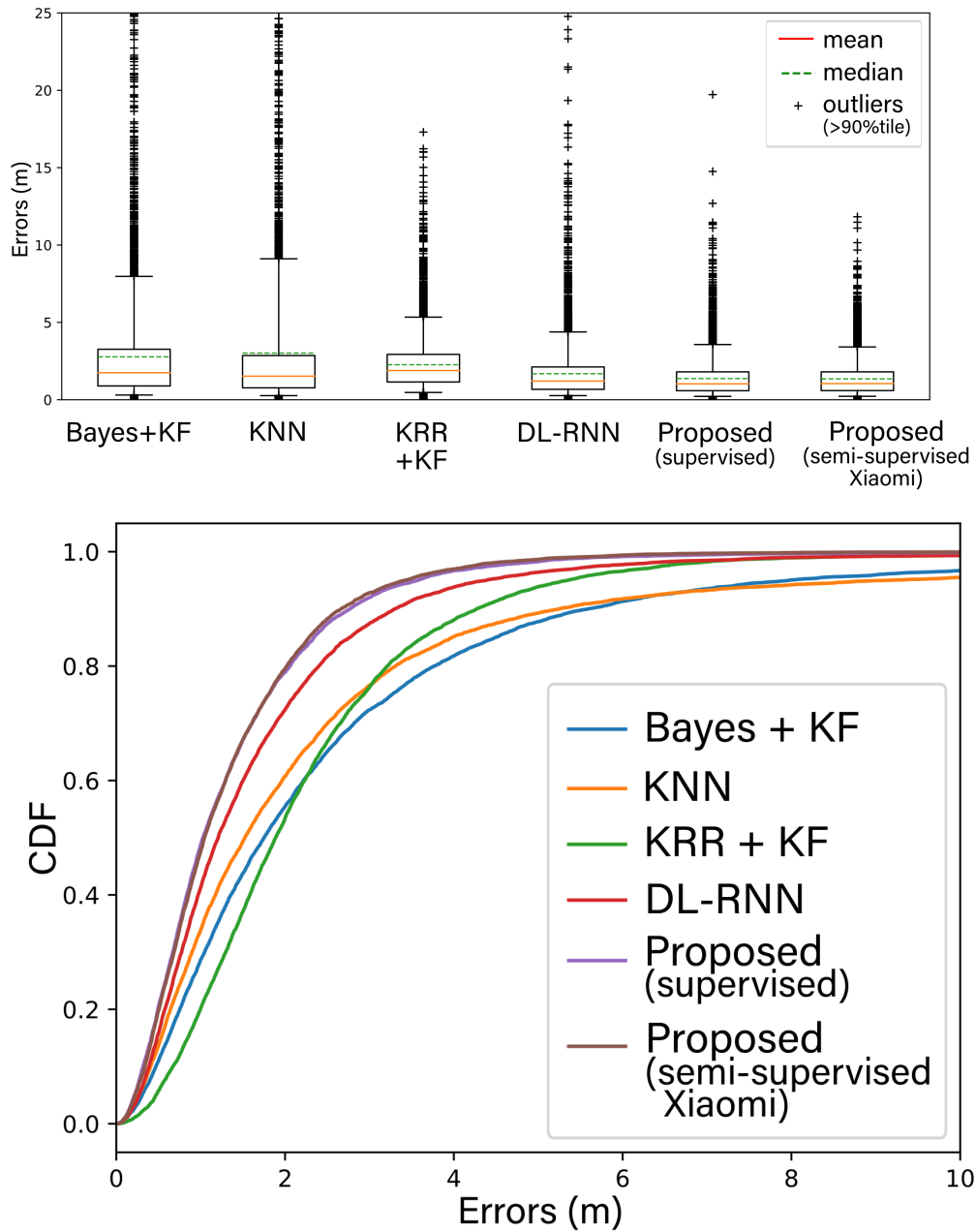
Figure 6.2: Evaluation: Box plot and Distance Error CDF comparing different methods

Chapter 7

Conclusions

We return back to the question we started with, as we deploy BLE based localization system in real world scenarios coming up with methods and algorithms that can work without performance disparity on multiple devices.

In this research, we try solving this for varying smartphone models, developing a localization engine that can work different smartphones. We define different scenarios in our problem statement, considering different levels of access to the training data. We collected a large scale BLE RSSI data set for 15 different smartphones. We perform empirical experiments concluding that smartphone signals can change significantly and unpredictably across phones in different situations. We observe that in particular phones, we observe receiver failures, detecting no observations for a couple of seconds. Such a situation inevitably would lead to failure. To fix the beacon signals in such scenarios, we propose the use of `TransNet` trained using our novel statistic similarity loss (SSL) that can learn to translate beacon signals from an unknown phone RSSI to a known phone RSSI. We provide solutions for the various all the possible scenarios listed with the proposed approach. The experiments and the evaluation indicate that the proposed approach can adapt itself towards a change of smartphone models, and outperforms other state of the art methods for RSSI based localization.

Chapter 8

Future Work

In this work, we have considered the problem of making RSSI based localization device agnostic to a certain degree. However, the change of phone models is not the only factor that affects the distribution of RSSI received. Dynamic factors like crowd density, changes in the phone software are inevitable and will significantly affect our localization performance. Development of a basis function that can treat the problem as a matrix factorization correcting for significant changes, instead of just learning minor corrections in the signal can be an exciting direction for work. At the same time, online adaptation model, replicating ideas from "few-shot" learning to adapt the network real-time, is exciting and would be very useful. With the recent advancements in BLE standards, we have information like AoA and ToA also available in some cases, use of which can be leveraged to further improve upon the accuracy. Finally, accounting to the low frequency of BLE RSSI received, making this technology multi-modal by merging the use of IMU can improve upon the localization performance to a high degree.

Bibliography

[1] Smart Card Alliance. Bluetooth low energy (ble) 101: A technology primer with example use cases. 2014. 5.3

[2] M. Altini, D. Brunelli, E. Farella, and L. Benini. Bluetooth indoor localization with multiple neural networks. In *IEEE 5th International Symposium on Wireless Pervasive Computing 2010*, pages 295–300, May 2010. doi: 10.1109/ISWPC. 2010.5483748. 2

[3] Roshan Ayyalasomayajula, Deepak Vasisht, and Dinesh Bharadia. Bloc: Csi-based accurate localization for ble tags. In *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*, pages 126–138, 2018. 2

[4] 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)*, volume 2, pages 775–784 vol.2, March 2000. doi: 10.1109/INFCOM.2000.832252. 2, 6.2

[5] S. Bai, M. Yan, Q. Wan, L. He, X. Wang, and J. Li. Dl-rnn: An accurate indoor localization method via double rnns. *IEEE Sensors Journal*, 20(1):286–295, Jan 2020. ISSN 2379-9153. doi: 10.1109/JSEN.2019.2936412. 2, 6.2

[6] Joydeep Biswas and Manuela Veloso. Wifi localization and navigation for autonomous indoor mobile robots. In *2010 IEEE international conference on robotics and automation*, pages 4379–4384. IEEE, 2010. 2

[7] Dongyao Chen, Kang G. Shin, Yurong Jiang, and Kyu-Han Kim. Locating and tracking ble beacons with smartphones. In *CoNEXT '17*, 2017. 2

[8] Liang Chen, Ling Pei, Heidi Kuusniemi, Yuwei Chen, Tuomo Kröger, and Ruizhi Chen. Bayesian fusion for indoor positioning using bluetooth fingerprints. *Wireless Personal Communications*, 70:1735–1745, 2013. 6.2

[9] R. Faragher and R. Harle. Location fingerprinting with bluetooth low energy beacons. *IEEE Journal on Selected Areas in Communications*, 33(11):2418–2428, Nov 2015. ISSN 1558-0008. doi: 10.1109/JSAC.2015.2430281. 2

Bibliography

[10] Ramsey Faragher and Robert Harle. Location fingerprinting with bluetooth low energy beacons. *IEEE journal on Selected Areas in Communications*, 33(11): 2418–2428, 2015. 2

[11] B. Ferris, D. Haehnel, and D. Fox. Gaussian processes for signal strength-based location estimation. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006. doi: 10.15607/RSS.2006.II.039. 2

[12] Jonathan Fürst, Kaifei Chen, Hyung-sin Kim, and Philippe Bonnet. Evaluating bluetooth low energy for iot. 04 2018. doi: 10.1109/CPSBench.2018.00007. 2

[13] S. Gezici, Zhi Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, and Z. Sahinoglu. Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. *IEEE Signal Processing Magazine*, 22(4): 70–84, July 2005. ISSN 1558-0792. doi: 10.1109/MSP.2005.1458289. 2

[14] Donald L Hall, Ram M Narayanan, and David M Jenkins. Sdr based indoor beacon localization using 3d probabilistic multipath exploitation and deep learning. *Electronics*, 8(11):1323, 2019. 2

[15] Suining He and S-H Gary Chan. Wi-fi fingerprint-based indoor positioning: Recent advances and comparisons. *IEEE Communications Surveys & Tutorials*, 18(1):466–490, 2015. 2

[16] Sebastian Hilsenbeck, Dmytro Bobkov, Georg Schroth, Robert Huitl, and Eckehard Steinbach. Graph-based data fusion of pedometer and wifi measurements for mobile indoor positioning. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, page 147–158, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450329682. doi: 10.1145/2632048.2636079. URL https://doi.org/10.1145/2632048.2636079. 2

[17] J. Krumm and E. Horvitz. Locadio: inferring motion and location from wi-fi signal strengths. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004.*, pages 4–13, Aug 2004. doi: 10.1109/MOBIQ.2004.1331705. 2

[18] Patrick Lazik, Niranjini Rajagopal, Oliver Shih, Bruno Sinopoli, and Anthony Rowe. Alps: A bluetooth and ultrasound platform for mapping and localization. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, page 73–84, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336314. doi: 10.1145/2809695.2809727. URL https://doi.org/10.1145/2809695.2809727. 2

[19] Ville V Lehtola, Harri Kaartinen, Andreas Nüchter, Risto Kaijaluoto, Antero Kukko, Paula Litkey, Eija Honkavaara, Tomi Rosnell, Matti T Vaaja, Juho-Pekka Virtanen, et al. Comparison of the selected state-of-the-art 3d indoor scanning

and point cloud generation methods. *Remote sensing*, 9(8):796, 2017. 5.2

[20] Julia Letchner, Dieter Fox, and Anthony LaMarca. Large-scale localization from wireless signal strength. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1*, AAAI'05, page 15–20. AAAI Press, 2005. ISBN 157735236x. 2

[21] Chong Liu, Tereus Scott, Kui Wu, and Daniel Hoffman. Range-free sensor localisation with ring overlapping based on comparison of received signal strength indicator. *IJSNet*, 2:399–413, 01 2007. doi: 10.1504/IJSNET.2007.014364. 2

[22] 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)*, 37(6):1067–1080, Nov 2007. ISSN 1558-2442. doi: 10.1109/TSMCC.2007.905750. 2, 5.9.2

[23] Hongbo Liu, Yu Gan, Jie Yang, Simon Sidhom, Yan Wang, Yingying Chen, and Fan Ye. Push the limit of wifi based localization for smartphones. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 305–316, 2012. 2

[24] Ayush Mittal, Saideep Tiku, and Sudeep Pasricha. Adapting convolutional neural networks for indoor localization with smart mobile devices. In *Proceedings of the 2018 on Great Lakes Symposium on VLSI*, pages 117–122, 2018. 2

[25] Mehdi Mohammadi, Ala I. Al-Fuqaha, Mohsen Guizani, and Jun-Seok Oh. Semi-supervised deep reinforcement learning in support of iot and smart city services. *CoRR*, abs/1810.04118, 2018. URL http://arxiv.org/abs/1810.04118. 5

[26] Esmond Mok and Günther Retscher. Location determination using wifi fingerprinting versus wifi trilateration. *Journal of Location Based Services*, 1(2): 145–159, 2007. 2

[27] Masayuki Murata, Dragan Ahmetovic, Daisuke Sato, Hironobu Takagi, Kris M Kitani, and Chieko Asakawa. Smartphone-based indoor localization for blind navigation across building complexes. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE, 2018. 2, 6.2

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 6.1

[29] Fazli Subhan, Halabi Hasbullah, Azat Rozyyev, and Sheikh Tahir Bakhsh. Indoor positioning in bluetooth networks using fingerprinting and lateration approach. *2011 International Conference on Information Science and Applications*, pages 1–9, 2011. 2

[30] K. Urano, K. Hiroi, T. Yonezawa, and N. Kawaguchi. An end-to-end ble indoor location estimation method using lstm. In *2019 Twelfth International Conference on Mobile Computing and Ubiquitous Network (ICMU)*, pages 1–7, Nov 2019. doi: 10.23919/ICMU48249.2019.9006638. 2

[31] Deepak Vasisht, Swarun Kumar, and Dina Katabi. Decimeter-level localization with a single wifi access point. In *Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation*, NSDI'16, page 165–178, USA, 2016. USENIX Association. ISBN 9781931971294. 2

[32] Vijayanth Vivekanandan and Vincent WS Wong. Concentric anchor-beacons (cab) localization for wireless sensor networks. In *2006 IEEE International Conference on Communications*, volume 9, pages 3972–3977. IEEE, 2006. 2

[33] JUE WANG and Dina KATABI. Dude, where's my card? rfid positioning that works with multipath and non-line of sight. *Computer communication review*, 43 (4):51–62, 2013. 2

[34] C. Xiao, D. Yang, Z. Chen, and G. Tan. 3-d ble indoor localization based on denoising autoencoder. *IEEE Access*, 5:12751–12760, 2017. ISSN 2169-3536. doi: 10.1109/ACCESS.2017.2720164. 2

[35] Zheng Yang, Zimu Zhou, and Yunhao Liu. From rssi to csi: Indoor localization via channel response. *ACM Comput. Surv.*, 46(2), December 2013. ISSN 0360-0300. doi: 10.1145/2543581.2543592. URL https://doi.org/10.1145/2543581.2543592. 2

[36] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. 5.2

[37] Wei Zhang, Kan Liu, Weidong Zhang, Youmei Zhang, and Jason Gu. Deep neural networks for wireless localization in indoor and outdoor environments. *Neurocomputing*, 194:279–287, 2016. 2

[38] X. Zhao, Z. Xiao, A. Markham, N. Trigoni, and Y. Ren. Does btle measure up against wifi? a comparison of indoor location performance. In *European Wireless 2014; 20th European Wireless Conference*, pages 1–6, May 2014. 2, 5.1