D. DOĞAN

RADIO FREQUENCY FINGERPRINTING-BASED EMITTER LOCALIZATION IN IOT-ENABLED SMART CITIES

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

DEREN DOĞAN

A MASTER OF SCIENCE THESIS
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

JULY 2023

ATILIM UNIVERSITY 2023

RADIO FREQUENCY FINGERPRINTING-BASED EMITTER LOCALIZATION IN IOT-ENABLED SMART CITIES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

DEREN DOĞAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JULY 2023

Approval of the Graduate School of Natural and Applied Sciences, Atılım University.

_____

Prof. Dr. Ender KESKİNKILIÇ
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Electrical and Electronics Engineering Department, Atılım University**.

_____

Prof. Dr. Reşat Özgür DORUK
Head of Department

This is to certify that we have read the thesis RADIO FREQUENCY FINGERPRINTING-BASED EMITTER LOCALIZATION IN IOT-ENABLED SMART CITIES submitted by DEREN DOĞAN and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Assoc. Prof. Dr. Yaser DALVEREN
Supervisor

Examining Committee Members:

Prof. Dr. Elif AYDIN
Electrical and Electronics Engineering
Atılım University _____

Assoc. Prof. Dr. Yaser DALVEREN
Electrical and Electronics Engineering
Atılım University _____

Prof. Dr. Ali KARA
Electrical and Electronics Engineering
Gazi University _____

Date: July 7, 2023

I declare and guarantee that all data, knowledge and information in this document has been obtained, processed and presented in accordance with academic rules and ethical conduct. Based on these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Deren DOĞAN

Signature :

ABSTRACT

RADIO FREQUENCY FINGERPRINTING-BASED EMITTER LOCALIZATION IN IOT-ENABLED SMART CITIES

Doğan, Deren

M.S., Department of Electrical and Electronics Engineering

Supervisor : Assoc. Prof. Dr. Yaser DALVEREN

July 2023, 57 pages

The rapid advancement of wireless technology has grown the significance of the Internet of Things (IoT). IoT applications are being used to decrease costs and improve performance across various industries. In smart cities, such applications are also utilized to offer localization-based services. Several localization procedures have been used for long years due to the demand for localization in geographic regions. Radio frequency fingerprinting (RFF) localization has become one of the most effective methods when considering the advantages provided by recent advancements in machine learning (ML) methods. Implementing a reasonable-priced and high-performance IoT wireless technology is a challenging issue in localization. In this regard, IQRF technology presents novel opportunities. Thus, in a system comprising IQRF sensor nodes operating in the 868 MHz band, this thesis proposes a received signal strength indicator (RSSI) fingerprint-based localization method implementing supervised classification methods in ML. To this end, measurements for Line-of-Sight (LoS) links were conducted in a local outdoor environment. The achieved results show the exceptionally strong prediction accuracy of the "Bagged Trees", "Weighted k-NN", and "Medium Gaussian SVM" methods. The results of the thesis have the potential to assist in the advancement of localization systems based on RFF in smart cities.

Keywords: Internet of things, radio frequency fingerprinting localization, IQRF technology, received signal strength indicator, classification

ÖZ

IOT ERİŞİMLİ AKILLI ŞEHİRLERDE RADYO FREKANSI PARMAK İZİ TABANLI YAYICI KONUMLANDIRMA

Doğan, Deren

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği

Tez Yöneticisi : Doç. Dr. Yaser DALVEREN

Temmuz 2023, 57 sayfa

Kablosuz teknolojinin hızlı gelişimi, Nesnelerin İnterneti'nin (IoT) önemini artırdı. IoT uygulamaları, çeşitli sektörlerde maliyetleri azaltmak ve performansı yükseltmek için kullanılıyor. Akıllı şehirlerde bu tür uygulamalardan yararlanılarak konumlandırma tabanlı hizmetler de sunulmaktadır. Coğrafi bölgelerde konumlandırma talebi nedeniyle uzun yıllardır çeşitli konumlandırma prosedürleri kullanılmaktadır. Radyo frekansı parmak izi (RFF) konumlandırması, makine öğrenimi (ML) yöntemlerindeki son gelişmelerin sağladığı avantajlar dikkate alındığında en etkili yöntemlerden biri haline geldi. Makul fiyatlı ve yüksek performanslı bir IoT kablosuz teknolojisini uygulamak, konumlandırmada zorlu bir konudur. Bu bağlamda, IQRF teknolojisi yeni fırsatlar sunmaktadır. Bu nedenle, 868 MHz bandında çalışan IQRF sensör düğümlerini içeren bir sistemde bu tez, makine öğreniminde denetimli sınıflandırma yöntemlerini uygulayan bir alınan sinyal gücü göstergesi (RSSI) parmak izi tabanlı konumlandırma yöntemi önerir. Bu amaçla, Görüş Hattı (LoS) bağlantıları için yerel bir dış ortamda ölçümler yürütüldü. Elde edilen sonuçlar, "Torbalı Ağaçlar", "Ağırlıklı k-NN" ve "Orta Gaussian SVM" yöntemlerinin son derece güçlü tahmin doğruluğunu gösterir. Tezin sonuçları, akıllı şehirlerde radyo frekansı parmak izine dayalı konumlandırma sistemlerinin ilerlemesine destek olma potansiyeline sahiptir.

Anahtar Kelimeler: Nesnelerin interneti, radyo frekansı parmak izi konumlandırması, IQRF teknolojisi, alınan sinyal gücü göstergesi, sınıflandırma

To my family...

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to Assoc. Prof. Dr. Yaser Dalveren for his priceless and in-depth guidance throughout the thesis period. Working with him was a great honour and opportunity. Over every step of my academic research, I have been inspired by his immense knowledge and wealth of experience.

I am deeply indebted to Prof. Dr. Ali Kara for his valuable comments, concern, and patience. Regular discussions and meetings have served as the foundation for the completeness of this thesis.

I would like to extend my sincere thanks to Prof. Dr. Elif Aydın for participating on my thesis committee and offering insightful feedback and recommendations.

Lastly, I would be remiss in not mentioning my family, especially Deniz Doğan, and my friends. This endeavour would not have been possible without their encouragement and emotional support.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

$\sigma(z)$	:	Sigmoid Function
w_i	:	Regression Coefficient
b	:	Constant Bias of Linear SVM
k	:	Number of Neighbors
σ_y	:	Standard Deviation of Gaussian Probability Density Function
μ_y	:	Mean Value of Gaussian Probability Density Function
I	:	Inducer of Ensemble Classifier
W_i	:	Sample Weight of Weighted k-NN Classifier

CHAPTER 1

INTRODUCTION

1.1 Overview

Localization has emerged as a primary concern mainly caused by the substantial development of wireless sensor networks (WSNs). WSNs are crucial for connecting different devices that are placed in distant positions during the networking and communication. Typically, a network is shared by many of wireless sensors in order that they can exchange relevant information. Localization, positioning as well as geolocation are all terms used to describe the process of determining the physical location of sensor nodes in a wireless network. Localization procedure is critical for data transfer over the network since the primary purpose of localization is to enable it simpler on objects to share information with one another by accurately determining the location of each object. Moreover, the localization associated with the target nodes is significant in reducing network latency by utilizing the most optimal path for data packet transfer across the network. It is a remarkable point since signal attenuation, refraction, multipath, and frequent interruptions can affect localization performance in urban environments, which are locations adjacent to buildings and other obstacles. Node information on locations in a wireless network can be utilized for a variety of purposes, including tracking mobile nodes, identifying coverage area or routing improvement. Localization network of the WSNs consists of anchor nodes, displaced nodes, and a central server. To communicate amongst the sensors, radio frequency (RF) signals are used. Anchor nodes are nodes that are aware of their location in the network and tag nodes are nodes that have an 8-bit microprocessor and an RF transceiver [1]. The data used as input is used to localize the sensor nodes. The main input data might be the location of the nodes that anchor, with various inputs dictated

by the computation methods employed in the localization procedure [2].

The extension of Information and Communication Technology (ICT) and the rise of the Internet of Things (IoT) have changed how devices communicate with one another. IoT is a remarkable and innovative communication paradigm that enables an intelligent network to connect to numerous electronic devices or objects over the Internet in order to provide reliable and real-time connectivity and gather data from sensors [3]. Since the building of sensor networks that can receive and transmit data to determine the location of the target, location-based services (LBSs) have become a highly valuable approach for IoT applications in smart cities and localization in the IoT environment enables tracking and monitoring targets through the sensor nodes [4]. According to [5], LBSs are services that use a mobile device's location with other data to offer more valuable information to the user. To this end, the intention is to employ IoT implementation to improve localization systems, provide ambient intelligence, and enable easy processing in dynamic scenarios for consumers. Location identification of the target or unknown nodes is a challenging task in IoT-based smart city applications. Such a sort of information can be accessed by implementing a localization approach such as radio frequency fingerprinting (RFF). RFF localization is a procedure that addresses the features of signal transmission to determine the device from which the signal originates. Every signal originator has a unique "fingerprint" depending on the location as well as the configuration of the signals it transmits. Solutions based on fingerprinting are not model-dependent. They are procedures that are data-driven based on the presumption that specific RF features are capable of reliably and uniquely identifying a location. RFF approaches are categorized primarily according to the type of information obtained, consequently RFF localization extracts useful features such as received signal strength indicator (RSSI), angle of arrival (AoA), time of arrival (ToA), time difference of arrival (TDoA), wavelet feature images, or channel impulse responses (CIRs) to use as location features. Therefore, the core concept behind the location fingerprinting is to link physically quantifiable features with discrete locations within an operational region. The primary goal underlying feature extraction is to generate a unique RF fingerprint profile that differentiates one source from others [6]. The analytic characterization of the relationship among the signal feature and the observer's position is not necessary, which makes fingerprinting appealing. This is an

intimidating task since it requires complicated characterization of propagation as well as solving the wave equation for non-homogeneous scenarios. Indeed, fingerprinting methods offer this functional relationship that exists between the signal feature and where it is located using a table of recorded samples in the training dataset. As a result, the dataset used for training is nothing more than samples of a function that connects a point in space to an observable feature at the same location.

1.2 Performance Metrics

A number of performance metrics are employed to assess the localization techniques. The metrics that are often employed for performing an accurate benchmark include: accuracy, precision, robustness, complexity, scalability, reliability and cost. The accuracy metric, commonly referred to as location error, serves as essential to evaluate localization systems. The mean distance error is often stated as the average Euclidean distance that exists between predicted and true locations. The better the system, the higher the accuracy; however, accuracy and other attributes are sometimes traded off. Location precision reveals the fluctuation in a localization technique's performance over a large number of trials, which serves as a measure of the robustness of the technique. The precision is assessed using the Cumulative Distributed Function (CDF) of the error of distance. The system with the CDF graph that achieves high probability values the quickest is favoured if the localization systems' accuracies are equal since its distance errors are typically fewer. The robustness metric of a localization technique is determined by how well it predicts the location of a target in the presence of interference from other signals or signal loss caused by damaged transmitters, or attenuation due to environmental conditions. The robustness of a technique is also determined by its application in varied conditions while preserving about the same needed accuracy. A localization technique's complexity can be related to software, hardware, and operational issues. Because most mobile devices lack powerful computational units and extended battery life, location algorithms with relatively small complexity are ideal. Since calculating the analytic complexity equations of various localization techniques is challenging, the computing time is used instead. The rate of location is a key indicator of complexity. The location lag is the delay among the

moment when a target shifts to a new position and the moment when that target's new location is detected by the system, which is the dual of the location rate. A localization system's scalability is determined by its performance when the positional scope changes. A location-based system may require two axes of scaling: geography and density. The geographical scale is proportional to the region covered. The amount of measurement units utilized in a specified area over a given time period is referred to as density. The level of confidence in the predicted location is reflected in reliability. It is generated from the variance in localization accuracy over time as well as the accuracy of each measurement. Many factors can influence the cost associated with a localization system. The number of reference nodes is regarded as a space cost. Energy is a significant cost factor in a system. Some mobile units, such as those with rechargeable batteries, have a battery life of several days without needing to be recharged [7].

1.3 Problem Statement

With advancing of innovations in wireless communication technologies, the necessity for solutions to find location for both individuals and objects has become increasingly urgent in IoT-enabled smart cities. Despite the fact that technologies such as 6LoWPAN, Bluetooth Low Energy (BLE), as well as ZigBee, have been extensively adopted for IoT solutions in smart cities, their restricted coverage may be seen as the biggest drawback. For better transmission efficiency, Low-Power Wide Area Networks (LPWANs) technologies which include SigFox, Ingenu, and LoRa have been proposed [8]. Although they have worthwhile advantages, specifically in terms of communication range, there are some significant shortcomings that must be considered. One addresses the issue of power consumption of data transmission, while the other is concerned with end-to-end connection latency. Therefore, adopting a reasonably priced wireless technology with low-power consumption represents a challenging task in smart cities. IQRF Technology, on the other hand, is an attractive invention that might be an alternative in IoT-based smart cities [9].

Despite increased interest in outdoor localization with the accessibility of a variety of applications, present solutions are unable to fully meet customer demands. Most

approaches fall short of the requisite accuracy. While some methods are capable of meters level accuracy, they necessitate additional hardware and a significant level of calculation complexity. Over and above, although various alternative localization techniques for urban scenarios have been constructed, each is limited to specific working circumstances. Integrating different techniques may result in a major localization improvement. The accuracy associated with these localization systems, nevertheless, are greatly dependent on how they are incorporated. For the systems that achieve accurate localization under changing circumstances additional flexibility is required. Accordingly, existing outdoor localization solutions lacks the capability to provide high accuracy location and robustness as the environment changes while being cost effective.

1.4 Literature Review

Several research on fingerprinting localization in IoT-enabled smart cities has been conducted in the last decade [10]-[15]. Based on the complexities in an industrial environment and the movable nature of the subjects, the study presented in [10] designed a localization system. In this method, inertial sensors present in smartphones utilize AoA-based Wi-Fi fingerprints to aid with localization. A stepped-length map has been created to pinpoint the user's location. Its training procedure comprises detecting step limitations and estimating orientation angles. In [11], a real-time locating system called iLocate is suggested, which leverages active radio frequency identification (RFID) tags for asset management and adds virtual reference tags to enhance localization accuracy as well as remove RFID RSSI noise. The proposed method creates RSSI sequences for virtual reference tags using interpolation. The iLocate utilized ZigBee for guaranteeing large-scale RFID networks. It has been implemented in all hardware by employing 2.45 GHz RFID chips to allow each active tag to interact with readers up to 1000 m distant in free space. The work in [12] investigated the basic principles of RSSI-based localization over long-range IoT networks that include SigFox. First of all, RSSI measurements of Global Positioning System (GPS) anchor nodes were utilized as markers to categorize other nodes into one of the GPS node classes. Whenever the classes of the nodes are isolated, measurements revealed that a

location classification of 100% accuracy is achieved. When classes are close together, measurements demonstrated that they can still reach an accuracy of 85%. In [13], an approach to localization was given that uses neighbor relative received signal strength for generating the fingerprint database and a Markov-chain prediction model to aid localization. In summary, the procedure is known as the novel localization method (LNM). The suggested LNM system analyzed the historical data of pedestrian placements to reduce unexpected signal fluctuations in an intelligent building setting while also providing calibration-free localization throughout various devices. The accuracy of an outdoor fingerprinting approach employing an extensive outdoor SigFox dataset that is freely available is described in [14]. The fingerprinting database was processed using the k-Nearest Neighbor (k-NN) technique. There were 31 distinct distance functions and 4 RSS formats tested. According to the results of the investigation, a mean prediction error of 340 m was revealed. A LoRa signal-based locating approach based on a fingerprint algorithm was proposed in [15]. The approach was tested experimentally by applying 3 distinct algorithms on interpolated fingerprint RSSI maps. For the purpose of testing, it has been built a private LoRa network and created LoRa gateways, end devices, and firmware as well. The most accurate mean error of 24.1 m using 46 validating points was found in the experiments.

1.5 Contributions

The contributions in this thesis cover the RFF localization of objects for local outdoor environments in IoT-enabled smart cities. Based on the most recent advances in supervised machine learning algorithms in the field of existing localization systems, a localization approach has been constructed that takes into account two key concerns: accuracy and cost efficiency. In this context, all classification algorithms available in machine learning were tested in order to obtain appreciable and acceptable accuracy values after transforming the localization problem into the classification problem. Measurements were carried out in order to evaluate each classification algorithm experimentally. Another contribution at this point is the comparison of each observed classifier, taking into account the validation and test accuracies, as well as the prediction speed and training time properties. Thus, the best performing classifier

for the relevant local area can be determined. As observed in Section 1.4, there is a lack of fingerprinting localization study that aims to implement all classifiers and a comparative analysis of their different aspects, and current IoT solutions lack a low-cost technology. As for the other concern which is the cost efficiency, this thesis contributes to the application of IQRF Technology to the field of fingerprinting localization since other technologies such as LPWANs technologies are more costly than IQRF. In fact, this thesis performs an RFF localization in a local outdoor area for IoT-enabled smart cities with IQRF Technology and tests and compares all available supervised classification algorithms in Matlab based on machine learning.

1.6 Thesis Layout

This thesis has the following basic layout. In Chapter 1, the background of the localization and radio frequency fingerprinting (RFF) localization including the primary performance metrics is addressed. Then, the deficiencies in the field are explained by giving the problem statement and the relevant literature review is provided. Finally, the contributions of the thesis work to the field are discussed. In Chapter 2, firstly, the main range-based algorithms used in localization approaches and their comparisons are presented. Secondly, the primary range-free algorithms are adressed, and then RFF-based localization is explained. In Chapter 3, the proposed fingerprinting method is explained from a mathematical point of view, and then the implementation phase is described. Moreover, the classification algorithms used in the tests are described. In Chapter 4, the execution of classification algorithms and their results are evaluated. Out of 20 classifiers, 3 classification algorithms with efficient performance are observed and the best among them is determined. In Chapter 5, the thesis work and related results are summarized and future work is addressed.

CHAPTER 2

LOCALIZATION ALGORITHMS

In [1], it is discussed how measurements in a wirelessly localization system are related to sensor positions. Equation 2.1 provides the general localization formula.

$$Y = h(X) + e \tag{2.1}$$

where X is the actual sensor coordinate vectors that distance is supposed to be estimated, e is the measurement error vector, and Y is the vector that includes all measurements.

Localization techniques are mainly classified into two fundamental categories based on the approach used to determine the location of the sensors: range-based localization and range-free localization. This categorization is based on whether the network requires to quantify exact distances or angles with regard to nodes in the network and whether accurate range is needed. Range-based techniques outperform range-free techniques in terms of accuracy. When the network's connectivity is poor, range-free techniques cannot function properly [16], [17]. These techniques are detailed in the following sections.

2.1 Range-Based Localization Algorithms

Range-based localization techniques use inter-device angles or device-to-device distances to determine distances among anchor and sensor nodes and estimate location using various geometric treatments. The Received Signal Strength Indicator (RSSI), Angle of Arrival (AoA), Time of Arrival (ToA), and Time Difference of Arrival (TDoA) are crucial techniques of range-based localization.

2.1.1 Received Signal Strength Indicator (RSSI)

Employing theoretical and empirical models, the receiver of RSSI-based localization techniques calculates the propagation loss of the signal according to the signal strength received from the transmitter. Then derives the location simply by transforming the signal propagation loss into the distance. RSSI analyzes the following Equation 2.2, which is the received power might be calculated:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2} \tag{2.2}$$

where P_r is the power at the receiving antenna, P_t is the transmitted power, G_t is the transmitter antenna gain, G_r is the receiver antenna gain, λ represents the wavelength of the transmitter signal in meters and d is the length of the path (distance between the antennas). This is known as Friis' Transmission Equation in free space [18].

Equation 2.2 might be modified and made more understandable for comparison by stating it as an inverse of the numeric path loss, or path gain, P_G. The path loss is the signal's attenuation as it transmits between the transmitter and receiver and it is caused by signal reflection, diffraction as well as scattering. In order to show the effect more clearly on the received signal strength, it might be used the path gain instead of the path loss as shown in Equation 2.3:

$$P_G = \frac{P_r}{P_t G_t G_r} = \left(\frac{\lambda}{4\pi d}\right)^2 \tag{2.3}$$

Working with logarithmic expressions, characterized by path gain in decibels (dB), is typically more convenient to use. Relevant formula is provided in Equation 2.4.

$$P_G\,(dB) = 20 \log\left(\frac{\lambda}{4\pi d}\right) \tag{2.4}$$

If transmitted power and antenna gains are known in free space, the distance might be calculated with high accuracy by using Equation 2.4 [19]. Extraction of the precise relation, nevertheless, appears to be unfeasible in real-life environments due to the unknown channel model. A significant portion of the available research depends on basic models to map RSSI to distance. Losses in the transmission path between the transmitter and receiver nodes are predicted using propagation models. The one-slop, log-distance propagation model which is given in Equation 2.5 and Equation 2.6

respectively is the most extensively utilized.

$$P_d\,[dB] = P_0\,[dB] - 10\alpha \log\left(\frac{d}{d_0}\right) + X_\sigma\,[dB] + b\,[dB] \tag{2.5}$$

$$d = \|X - S\| \tag{2.6}$$

where P_0 is the power at the transmitter's reference distance of d_0, which is typically 1 meter, P_d is the power received at distance d from the transmitter, X_σ is the shadowing effect which is frequently considered as Gaussian, α is the rate at which power decreases over distance expressed by the path loss exponent (PLE), and b is the bias error.

On one hand, RSSI is a cost-effective and energy-effective approach that requires no extra hardware other than a radio transceiver, which makes it suitable for sensor networks. Moreover, in comparison to its counterparts, it is easier to implement. On the other hand, the environments in which signals are transmitted corrupt signals to some extent. Thus, the RSSI-to-distance relationship is not always straightforward.

2.1.2 Angle of Arrival (AoA)

An antenna array or static antennas are used in the AoA technique to estimate the direction of arrival, and at least two sub-arrays of receivers are needed for locating sources. These antennas are commonly referred to as smart antennas since they rely on signal processing units for measuring AoA. It includes identifying the angular separation between a single beacon additionally a fixed axis or between two beacons. By computing the angle of receipt of several sensor nodes, triangulation is utilized to estimate location. At the target position, the direction lines of two terminals' directional antennas cross. Geometric relations may be used to find the intersection of two bearing lines from known reference points. The known fixed terminal coordinates and the antenna beam angles in regard to a common reference direction are used to determine the target coordinates. Equation 2.7 and Equation 2.8 offer the coordinates of the target position.

$$y = \frac{y_2 tan\,(\theta_2) - x_2}{tan\,(\theta_2) - tan\,(\theta_1)} \tag{2.7}$$

$$x = y.\ tan\,(\theta_1) \tag{2.8}$$

where (0, 0) and (x_2, y_2) are the coordinate values of the fixed stations F_1 and F_2 respectively. θ_1 and θ_2 are the AoAs corresponding to the received signals, and (x, y) are the coordinate values of the target location [19].

First and foremost, determining the AoA necessitates the use of real or virtual antenna arrays. To increase the system's accuracy, three or more fixed stations and highly directional antennas are required to measure angles, which quickly increases the cost of implementation. This is frequently impractical due to concerns such as cost and size. Therefore, when cost and size are the primary restrictions, the AoA-based algorithms are typically less desirable. Additionally, these arrays must satisfy specific requirements regarding antenna spacing and calibration.

2.1.3 Time of Arrival (ToA)

The ToA technique estimates distance directly from the time of transit between transmitter and receiver, whereas the time difference of arrival (TDoA) technique determines location using the variations in arrival times observed on pairs of transmission lines among the target and fixed terminals. They are both based upon the Time of Flight (ToF) distance measurement principle in which the time interval is transformed to distance by multiplying by the propagation speed. Lateration, also referred to as trilateration, is a commonly used technique in these time-based algorithms. It uses the known distance between at least three fixed stations in a two-dimensional environment to determine the location of a target object. The ToA estimates location by finding the points of intersection of spheres whose centres are at fixed stations and whose radii are estimated distances to the target. Trilateration serves to find the intersection of the circles when the center of each circle represents the location of a fixed station, and the radius of each circle represents the distance between the station and the target device. The trilateration method accurately predicts a unique location for given perfect information. The precise location will be where the circles intersect.

The emitting time and the propagation time are needed for location calculation in ToA algorithm, with the propagation time equivalent to the "arrival time – emitting time". The transmitter-receptor distance "d" is defined by the propagation time, and the receptor location is determined by the ToA measurement and geometric relation-

ship between them, particularly through circle intersections whose radius equals the distance between each of them and the target. After F_1 sends the signal to the target at t_0, the target, when it receives the signal at t_1, it is calculated the time of flight "T" by using the Equation 2.9 and Equation 2.10.

$$T = t_0 - t_1 \tag{2.9}$$

$$d = T.\, c \tag{2.10}$$

where c is the speed of light which equals $2.997 \times 10^8\ m/s \sim 3 \times 10^8\ m/s$.

Although the computation of the ToA appears straightforward in theory, there are several issues that result from imperfections in the hardware as well as the properties of the wireless propagation channel. Initially of all, it requires specific clock synchronization between the transmitter and receiver which might be challenging to accomplish in reality. For the sake of the precision of the arrival times, the sources and receivers must be time-synchronized. Regardless of perfect time synchronization, noise-related errors continue to arise.

2.1.4 Time Difference of Arrival (TDoA)

TDoA is a ToA variation and it is slightly more flexible than ToA. Basically, TDoA is the difference between ToAs from the fixed stations to a target. The TDoA finds the target at the intersections of hyperbolas created by foci at each fixed station in a pair. Three or four fixed stations at known locations are often required to determine the accurate location of the target device. When fixed stations send sync signals to the mobile node, these timing measurements are stored as separate time marks. These measurements are also analyzed by a locator, which calculates the difference in distance between a pair of fixed stations and the target device. After the signal is received at the fixed stations, the ToA difference between the target device and the two or more fixed stations is used to calculate the distance between the target device and the two fixed stations. This difference is calculated utilizing the following Equation 2.11:

$$\Delta d = c.\, \Delta t \tag{2.11}$$

where Δd is the difference in the arrival times at each fixed station and might be calculated by Equation 2.12.

$$\Delta d = \sqrt{(x_2 - x)^2 - (y_2 - y)^2} - \sqrt{(x_1 - x)^2 - (y_1 - y)^2} \tag{2.12}$$

With the help of Equation 2.12, the difference in range (d) between the fixed stations F_1 and F_2 is determined, and the hyperbola of possible locations is drawn since the equation may be transformed into a hyperbola utilizing nonlinear regression [20]. This procedure is repeated for the remaining fixed station pairings. The target device is placed at the intersection of three hyperbolas where the known locations of the fixed stations, F_1, F_2 and F_3 are (x_1, y_1), (x_2, y_2), and (x_3, y_3) respectively. The disadvantages of the TDoA algorithm are similar to the ToA. Environmental factors have a direct effect on the TDoA algorithm as well.

2.2 Comparison of Range-Based Localization Algorithms

The AoA, ToA, and TDoA techniques need either appropriate time synchronization or an array of antenna, which can increase the overall system cost. An RSSI-based localization system, on the other hand, employs characteristics associated with wireless signal intensity and does not need time synchronization or angle measurement. Furthermore, RSSI measurement is relatively simple and may use present wireless technologies with minimum or no hardware modifications, which avoids the extra cost and consumption of energy. All that must be undertaken is being able to access RSSI outputs, which all receivers offer, and analyse the information using location estimate software. A brief comparison of the RSSI, AoA, ToA and TDOA approaches based on their primary characteristics is given in Table 2.1.

Table 2.1 Comparison of Range-Based Localization Algorithms

Measurement	**Localization Accuracy**	**Sensitivity to Multipath**	**Cost**	**Hardware Size**
RSSI	High in short-range	Yes	Low	Small
AoA	Medium	Yes	High	Large
ToA	High	Yes	Medium	Large
TDoA	High	Yes	Medium	Large

The majority of RSSI-based localization research uses fingerprinting methods by reason of its outstanding degree of accuracy. Considering the advantages, using RSSI features RFF-based localization approaches is a reasonable option.

2.3 Range-Free Localization Algorithms

Range-free localization techniques estimate a node's location based on neighborhood information, hop counts from known anchor points or information learned from the area in which a node is thought to reside. Distances are not necessary to be calculated directly. The DV-Hop, APIT, and Centroid localization techniques are instances of traditional range-free localization. These algorithms are usually referred to as connectivity-based algorithms.

2.3.1 Distance Vector Hop (DV-Hop)

The DV-Hop technique uses a collection of anchor nodes which know their GPS location to estimate the unknown location of the nodes in the sensor network. Nonetheless, the predicted distance between anchor nodes and unidentified nodes is prone to being incorrect due to errors in anchor node distance hop estimation and a variety of other factors. As a result, the DV-Hop algorithm has poor localization accuracy.

The DV-Hop procedure can be divided into 3 steps. In the first step, each anchor in the network emits its location and a hop count value of 1 through a beacon packet. Following the receipt of beacon packets, every unknown node stores a table including the received information of each anchor node (x_i, y_i, h_i), wherein (x_i, y_i) is the coordinate values of anchor i, and h_i is the smallest number of hops from the anchor. To determine the smallest hop count value from the unknown nodes to the all network anchors, h_i of the table is exchanged with the hop count value of another received packet with a lower hop count value to a specific anchor node. The received packet is routed within the network with the hop count increased by 1 at each intermediary hop. The destination node discards this packet if it does not include a smaller hop count value. All unidentified nodes in the network acquire the smallest hop count value from every anchor nodes in this manner. The next step is for each anchor to calculate

its average hop distance (one-hop-size) from another anchor within the network by Equation 2.13.

$$Hop\ Size_i = \frac{\sum_{i \neq j}^{n} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{\sum_{i \neq j}^{n} h_{ij}} \tag{2.13}$$

where (x_i, x_j) is i's anchor coordinate, (y_i, y_j) is j's anchor coordinate, h_{ij} is the smallest hop counts among anchors i and j, and n is the total number of anchors. In the third step, each unknown node trilaterates its position using the least square (LS) technique. It is worth noting that, in the basic trilateration, unknown sensors predict their positions using only 3 distances from the anchors.

In the DV-Hop technique, the path of hops among nodes is considered to be a straight line. However, in practice, this presumption is not always applicable. Since the average distance of a hop is utilized to predict the distance among anchors and unidentified nodes, this presumption produces an error in the DV-Hop algorithm's localization findings. Furthermore, in the DV-Hop technique, the range of communication of each of the nodes in the network's structure is assumed to be a standard circle, which is not valid in the real world due to noise. These shortcomings cause errors in the DV-Hop algorithm's localization findigs [21].

2.3.2 Approximate Point in Triangle (APIT)

Regarding location estimation, APIT needs a heterogeneous network for sensing devices. The APIT algorithm uses an area-based method to conduct position estimate through field segmentation and requires a small percentage of anchors. Additionally, such nodes can be configured with a powerful radio transmitter. The primary principle behind APIT for node localization is the consideration of overlapping triangles. These triangles' vertices serve as anchors. Instead of a single node's coverage area, bounding triangles are generated using any set of 3 reference nodes. The sensor nodes in the APIT algorithm initially collect location information from adjacent anchors. Secondly, the Point in Triangulation (PIT) test determines if a sensor node is located within a virtual triangle constructed by linking the 3 anchors where the signals are received. Following the completion of the PIT test, the APIT algorithm combines the findings using a grid SCAN approach. To determine the node's location, the APIT

algorithm computes the Centre of Gravity (COG) based on the intersections of all the overlapped triangles within which it resides [22].

2.3.3 Centroid

The centroid algorithm assumes that all nodes know their position, hence the target node's position should be determined based on the centroid of the nodes' positions in their communication range. The technique is simple in theory, with simple implementation and low communication costs [23]. The implemented algorithm has 3 steps [24], [25]. In the first step, all anchors regularly send beacon packets containing anchor node id and localization information to all sensor nodes in their transmission range. All unidentified nodes gather beacon packet received from various reference points or the anchor nodes. In the second step, all unidentified nodes determine their position by taking the centroid from all n positions of the anchor (X_i, Y_i). Following the receiving these beacon packets, a node determines its location in the third step using the centroid calculation formula described in Equation 2.14.

$$(x_{est}, y_{est}) = \frac{(x_1 + x_2 + ... + x_k}{k}, \frac{y_1 + y_2 + ... + y_k)}{k} \tag{2.14}$$

where $(x_1, y_1)...(x_k, y_k)$ is the unidentified node that has access to the coordinate information for the anchor node [26]. Despite its simplicity, this approach has a large localization error. Centroid is an estimate used to localize practical nodes; the accuracy is highly associated with the uniformity and evenness of distribution of the nodes.

2.4 Radio Frequency Fingerprinting (RFF) Localization

Although there exist numerous RFF localization methodologies [27]-[35], each one involves key components such as correlation database (CDB), a location server, and pattern matching. First of all, the acquisition of data is required for RFF localization. Building and maintaining a trustworthy fingerprint database represents one of the RFF's main challenges. Field experiments are carried out to record RF fingerprints, which are then stored in the CDB and made accessible by the location server. In general, there are two phases including offline (populating) and online (matching).

The CDB is formed during the offline phase. Each RF fingerprint within a CDB corresponds to a specific location. The procedure's prediction accuracy is affected by the number of offline records in the database. As a consequence, a larger database yields better accuracy in predictions. The signal fingerprint of each reference point within the area of interest is investigated to produce a signal/radio map. Receiving location queries, verifying the CDB, and estimating the location are handled by the network element. To estimate the location, the location server is obliged to match the detected RF fingerprint to a subset of the RF fingerprints recorded in the CDB. Generally, this is referred to as pattern matching during the online phase [36], [37]. In a nutshell, during the node localization process, the recorded data within the database is continuously compared to the actual node location. Identifying how closely the fingerprints of trained locations and test locations match one another is the aim of pattern matching.

The major phases of RFF localization utilizing features extracted from RSSI data are illustrated in Figure 2.1 [6]. Fingerprint features are gathered during the offline phase, and a database of fingerprints is produced. The target location is then estimated during the online phase using localization algorithms.

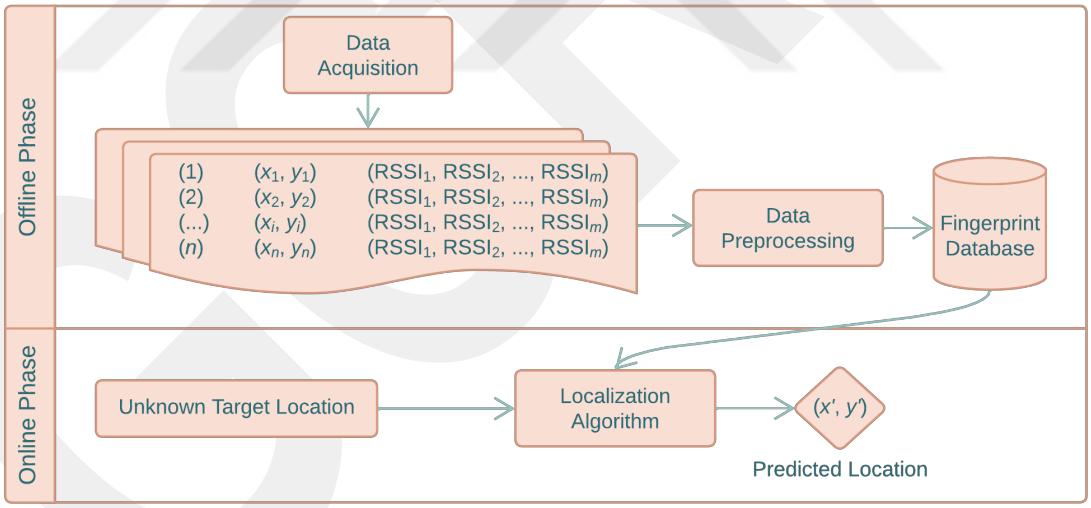

Figure 2.1 RSSI Fingerprint Feature-Based Localization Workflow Diagram

Recently, both academia and industry have paid a lot of attention to RFF-based methods, which compare location-specific fingerprints gathered previously with the fingerprint acquired from the target. In IoT-enabled smart cities, RFF has been used for numerous location-based applications since it is further streamlined and makes use of the IoT infrastructure to increase the localization accuracy. Therefore, the localiza-

tion method implemented in this thesis is chosen as RFF-based emitter localization considering its higher accuracy aspect. As the fingerprint feature, range-based RSSI algorithm is preferred, considering its low cost, small hardware size and high accuracy properties at short-ranges which is suitable for local areas. In the following chapter, the details of the method followed within the scope of the thesis, implementation process and the classification algorithms used are given.

CHAPTER 3

METHODOLOGY

3.1 Proposed Method

Associating a fingerprint with a location is the basis of fingerprinting algorithms, which is subsequently utilized to identify that location. The following procedure is implemented in the fingerprinting-based localization to extract the fingerprint database from the signal features. S stands for the signal feature which belongs to the space of signal feature $\mathbf{S}$, and m represents successive observation where $S = (S_1, ..., S_m) \in \mathbf{S}^m$. The signal is not always coming from a single source and can be interpreted as a composite of several real signals produced by various sources, such as beacons from multiple access points (APs). In general, there can be significant dependence between successive observations of signal features. Then, based on observations made in various locations fingerprints are created. A fingerprint-creation function f is a mapping $\mathbf{S}^m \rightarrow X^n$ that assigns an element X termed the fingerprint to observations at location u.

Following the selection of a signal feature and a fingerprint-creation function, the fingerprinting algorithm is designed. The initial step is building a database of location pairings and their fingerprints. Only a limited number of places in the localization space may be chosen during direct measurement in order to generate the database. Thorough measurements, simulation-based generation, or a combination of both methods can be used to build the training database. Lattices or irregular grids are two possibilities for the algebraic structure one can use for grinding. By quantifying the signal feature on the training locations within the training grid, the training database develops. Multiple measurements are taken at each location, and finger-

prints are created properly. The next step is to determine the position of a target node. Depending on the observed signal features, the fingerprinting algorithm gives the predicted location of the desired node. A fingerprint X_u is produced by a target node located on the location u measuring the chosen feature. Generally, there may be a difference in the number of measurements taken throughout the training and localization phases. After obtaining the fingerprint, the pattern matching function g is used to derive the location of the target node using fingerprints from the training database and the newly acquired fingerprint X_u. The mapping from X^n to the space of localization R is represented by the function g.

It is conceivable to speak of the pattern matching operation as a composite of different functions. As an example, the similarity Kernel function is able to determine the fingerprints within the database that are the most similar. Then, rather than announcing a single training location, one can also use k-nearest neighbor (k-NN) approaches to average among the k closest locations of training. The maximum of a function ψ, which compares two fingerprints, is an instance of a similarity Kernel. In the present example, the Kernel first determines the score of similarity between the target node's fingerprint and the fingerprints stored in the database, and then it selects the estimated location as the one that corresponds with the best score. The result for the following Equation 3.1 is the location prediction [38]:

$$\hat{u} = \arg\max_{v \in \wedge} \psi\left(X_u, X\right) \tag{3.1}$$

where $\hat{u}$ is the predicted location, v is the training point, and the training grid or group of training locations is represented by $\wedge$. The localization space R is divided into regions by the training locations in $\wedge$. A variety of measurements are performed at each point v, and corresponding fingerprints are produced. Equation 2.1 is depending on $(v, X) \in D$, whereas D represents the training database. D is composed up of pairs such as (v, X) and it is a subset of $\wedge \times X^n$. The similarity Kernel is reduced down into individual comparisons of the targeted node's fingerprint with the training fingerprints, then followed by the identification of fingerprints that have the greatest similarity to the fingerprint of the node in question. The final location is provided by averaged between the fingerprints obtained by the postprocessing methods like k-NN that have the highest k similarity scores. Therefore, for a localization space R, a fingerprinting algorithm includes the notations provided by Table 3.1 while Figure

3.1 depicts the operational block of a fingerprinting algorithm, from training database building through the localization of a random location u.

Table 3.1 Symbols Table of Fingerprinting Algorithm

Symbol	Definition
R	Localization Space
S	Signal Feature
$\mathbf{S}$	Signal Feature Space
f	Fingerprint-creation Function
u	Measurement Location
X	Fingerprint
g	Pattern Matching Function
v	Training Point
Λ	Set of Training Locations
D	Training Database

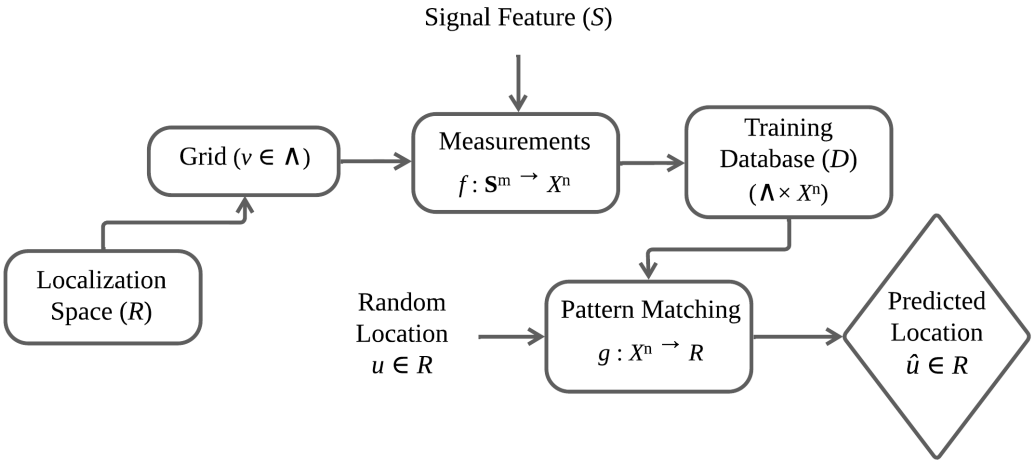

Figure 3.1 Operational Block of Fingerprinting Algorithm

In this thesis, it is proposed an RFF-based emitter localization that implemented by a classification procedure. A classification problem can be modelled to represent a localization problem. The localization problem needs to be mapped into a classification problem to use this model. Substantially, the localization area of interest is divided into a number of grids, and each grid represents one class in particular. Consequently,

the localization problem is turned into a classification problem. Classification-based localization can be expressed in two stages. The first stage is the class definition which is building a set of classes labelled $(C_1, C_2, ..., C_n)$ where each class C_i represents a particular region of the sensor network. The second stage is the training of the data so that the membership of each beacon node in each class C_i is known since the beacon locations are determined. Each beacon node's feature vector is represented by its signal strength vector. The training data of the classification method on class C_i is the feature vector. This model is used to estimate, for any sensor node and any class C_i, the membership of the sensor node in class C_i. This allows us to determine the area where the sensor node is located. To this end, RSSI fingerprint features which are acquired based on the IQRF Technology in a local outdoor environment are first divided into classes according to different transmitter locations. By assessing RSSI on the training locations $v \in \wedge$, a fingerprint database is build. A targeted node placed at the position $u \in R$ is localized using the measurements. The RSSI is taken from anchor i at every training location v, and the process of measuring is carried out a certain number of times. A large portion of these fingerprint features and their related locations are trained and validated by using various classifiers based on supervised learning in ML. To predict the unknown transmitter locations, a test dataset is tested over the trained and then validated dataset. Based on the results, classifier performances are compared in different aspects.

3.2 Implementation

In an RSSI-based RFF localization system that relies on classification algorithms, there are two crucial aspects. The first consideration is to select the most suitable IoT technology to be utilized in order to obtain RSSI features that will be stored in the database as fingerprints. After selecting the appropriate technology, a significant amount of features must be obtained to qualify the classification algorithms. This section proceeds with information on IQRF Technology, which serves as an IoT technology to build an extensive database of fingerprints. Then, the measurement scenario and data acquisition process are defined respectively, and finally, dataset construction is clarified.

3.2.1 IQRF Technology

IQRF is a sophisticated platform for packet-oriented communication that enables low-power, low-speed and simple-to-use wireless networking with peer-to-peer (P2P) or in complex networks. IQRF is a fully integrated ecosystem that includes hardware, software, development support, and services. It offers wireless connectivity in the sub-GHz ISM (Industrial, Scientific, and Medical) bands, i.e., 433 MHz, 868 MHz, and 916 MHz. The range of coverage is between tens and hundreds of meters. However, under specific circumstances, the range can be extended [9], [39]. IQRF Technology has significantly advanced in the previous years to compete with other technologies within the market. Its technical specifications are given in Table 3.2.

Table 3.2 Specifications of IQRF Technology

Specifications	
Data Rate	20 kb/s
Modulation	GFSK
Max. Power Consumption	15 μA
Max. Payload	64-byte
Latency	400 ms
Range in Free Space	1 km
Battery Lifetime	Min. 5 years

3.2.1.1 TR-72DAT Module

The IQRF platform's fundamental communication component is a transceiver module (TR). The TR is a very small (31.8×14.9×3.3 mm) and intelligent electronic board that has all of the circuitry required to operate wireless RF communication. The TR modules contain an 8-bit microcontroller, among other circuits, that runs an operating system (OS) in charge of mesh networking and communications. Meander Line Antenna (MLA) is built onto the printed circuit board (PCB) of the module, eliminating the need for additional components. A low-dropout regulator (LDO) and serial electrically erasable programmable read-only memory (EEPROM) are included in this

ready-to-use device. It consumes minimum power in battery-powered applications. Figure 3.2 shows the IQRF transceiver module that was used in this thesis work.

Figure 3.2 TR-72DAT Module

3.2.1.2 CK-USB-04A Programmer Kit

CK-USB-04A serves the purpose of IQRF programming and debugging. It is one of two primary IQRF development tools supported by the IQRF IDE (Integrated Development Environment). Uploading codes and programming specific TR modules plugged into the socket is feasible if the CK-USB-04A kit has been connected through USB (Universal Serial Bus). It functions as a TR host and is used to test IQRF applications. Figure 3.3 represents the CK-USB-04A device required to program the TR-72DAT modules.

Figure 3.3 CK-USB-04A Programmer Kit

3.2.1.3 DK-EVAL-04A Node Kit

IQRF-TRs that include application functionality codes that have been earlier uploaded through the programmer kit can be hosted in portable DK-EVAL-04A node kits. The

USB connector for this kit is strictly for charging and is not used for USB connectivity. In other words, IQRF-TRs to be plugged into the socket of this node kit must first be plugged into the socket of the CK-USB-04A programmer kit and programmed via IQRF IDE. The programmed TRs are then may be inserted into the socket of the DK-EVAL-04A kit. The DK-EVAL-04A device is shown in Figure 3.4.

Figure 3.4 DK-EVAL-04A Node Kit

3.2.1.4 IQRF Network

IQMESH is the abbreviation of the Mesh network topology that underlies IQRF Technology. IQMESH is based on directed flooding that is optimized. Its primary benefits are higher throughput and much greater robustness appearing in applications wherein reliability is required. A self-healing mechanism enables the network to seek a new path for the message to follow if a specific router fails. The IQMESH network is provided in Figure 3.5.

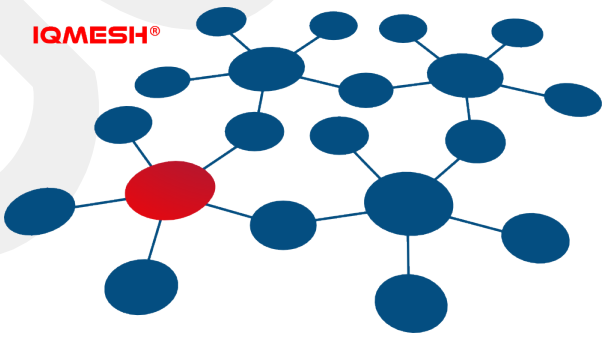

Figure 3.5 IQMESH Network

The device peripheral access (DPA) Framework (layer) implements IQMESH over the IQRF OS. A single coordinator node manages the Mesh network, which may include

up to 239 nodes in a single network. The data packets within such a network are available to the nodes which are targeted. IQMESH provides independent transmitting and receiving ability with resilient networking.

The networking functions described are not supported in non-networking configurations. The non-networking mode supports more than one peer-to-peer (P2P) device and is incompatible with the IQMESH network. In this case, data packets are available to all devices within the maximum attainable range. This mode of operation relies upon a user application layer that runs under OS and leverages predefined OS functionalities [9].

3.2.2 Measurement Scenario

In order to measure the necessary RSSI fingerprint features and then build a fingerprint database with these features in the offline phase of RFF-based localization, a local outdoor area has been selected on the Atılım University campus. The size of this area is 80×30 m, that is, 2400 m^2. The reason for choosing an area of this size is that IQRF Technology is an efficient solution for the short-range IoT applications mostly. Before the measurements used IQRF hardware equipment, the area of interest was divided into grids. The aim of dividing the area into grids is to transform the localization problem into a classification problem. Choosing the grid size is an important factor which significantly affects localization accuracy. In fact, the smaller grid size results in poor performance. Therefore, depending on the size of the experimental area, it is necessary to decide the grid size reasonably. In this context, the grid size of 5 m was deemed appropriate and the area was divided into square grids with 5 m on each side. Additionally, the selected area is free of obstacles such as buildings and trees to provide a line-of-sight (LoS) path between the embedded antennas of IQRF-TRs which are configured as multiple receivers and a transmitter. The aim is to perform RSSI measurements with the performance of IQRF Technology in the LoS link and then continue with the online phase of the localization process.

After dividing the area of interest into 96 grids with a grid size of 5 m, CK-USB-04A programmer kits with TR-72DAT modules plugged into their sockets were placed at each corner of the rectangular area. These 4 kits were used to measure RSSI values

since the modules were configured as receivers via IQRF IDE. 1 TR-72DAT module was configured as a transmitter and placed in the socket of the DK-EVAL-04A node kit. The transmitter is placed at a height of 1.6 m from the ground and the receivers at a height of 1.8 m from the ground. Experimental area is provided in Figure 3.6.

Figure 3.6 Experimental Area

Unlike Figure 3.6, throughout the measuring phase, there were no automobiles in the measurement area. RSSI fingerprint feature measurements were carried out at 868 MHz through the IQRF-TRs. Moreover, the measurement phase is based on peer-to-peer (P2P) communication, therefore, there is no client-server-based networking. Each device has equal responsibility and capability, each one is at once a server and a client without the need for a central server. P2P communication is common on small local area networks (LANs). Since it is aimed to perform transmitter localization in a local outdoor environment, P2P communication was adopted.

3.2.3 Data Acquisition

Each of the grids is identical and serves as the place where the transmitter stands to transmit data packets. Therefore, it was created as a series of classes labelled

(T_{x1}, T_{x2}, ..., T_{x96}). Initially, on the grid labelled as T_{x1}, the transmitter was positioned. It sent data packets in this grid, which were received by each receiver placed at the corners. For each data packet that is received, the last RSSI value was acquired. The same procedure was repeated until T_{x96}. The packet length is set to 1 byte. The transmitter is configured to send one packet every 20 ms during the entire measurement process, resulting in a sampling frequency of 50 packets/sec, the maximum achievable packet-sending rate. It is specified the number of packets to be sent as 3000 in each grid, as a result, sending 3000 packets from each grid takes around 1 minute. Since there are 96 grids and 3000 packets were sent from each grid to one receiver, a total of 288000 packets were transmitted per receiver. While this is the case in theory, in practice there is a concept called packet delivery rate, also known as packet delivery ratio, (PDR). It specifies the ratio of packets received by the destination to data packets produced by the source. Equation 3.2 indicates the PDR.

$$PDR = \frac{\sum number\ of\ packet\ receive}{\sum number\ of\ packet\ send} \tag{3.2}$$

Since not all packets sent by the transmitter can always be received completely by the receiver, the number of measured RSSI values per receiver needs to be optimized to create a fingerprint database. The number of packets sent is optimized as 2000 before using the Classification Learner Toolbox in Matlab. In other words, the number of packets received by each receiver from each grid was considered as 2000. As a result, there are a total of 8000 packets received by 4 receivers per grid, and thus 8000 RSSI values. Considering 96 grids, a total of 768000 packets were received and related RSSI values were measured. A dataset has been constructed of the measured RSSI values on the side of the receivers and from which grid the transmitter sends data packets related to the RSSI values. A brief summary of the dataset construction is given in Table 2.3. In this way, a dataset consisting of 192000 rows in total was obtained. As can be seen, this dataset has sufficient data to allow classification algorithms in ML to be applied to estimate the transmitter's location. Hence, this dataset was used in algorithms for location estimation. The labelled names of the receivers and the transmitter given in Table 3.3 are defined to be used in the online localization procedure. R_{x1}, R_{x2}, R_{x3} and R_{x4} are predictors, and T_{xi} is the response value in classification.

Table 3.3 Dataset Construction

Number of Row	R_{x1}	R_{x2}	R_{x3}	R_{x4}	T_{xi}
1	-55 dBm	-78 dBm	-57 dBm	-96 dBm	T_{x1}
2	-55 dBm	-78 dBm	-57 dBm	-97 dBm	T_{x1}
...	...	...	...	...	...
2000	-56 dBm	-80 dBm	-59 dBm	-97 dBm	T_{x1}
2001	-59 dBm	-85 dBm	-68 dBm	-79 dBm	T_{x2}
2002	-59 dBm	-85 dBm	-68 dBm	-79 dBm	T_{x2}
...	...	...	...	...	...
...	...	...	...	...	...
191999	-69 dBm	-69 dBm	-72 dBm	-49 dBm	T_{x96}
192000	-69 dBm	-69 dBm	-72 dBm	-49 dBm	T_{x96}

3.3 Classification Algorithms

A classifier with powerful learning abilities may be constructed based on supervised learning in ML and localization with a high prediction accuracy is obtained in this manner. Each data point in the training dataset is assigned a class label. The classifier is supposed to differentiate the test dataset based on these class labels during the process of testing. It is possible to perform common supervised learning tasks in Matlab by utilizing the Classification Learner Toolbox. It may be used to train, validate and test models including the following classifiers: decision trees, discriminant analysis, logistic regression classifiers, support vector machines (SVMs), *k*-nearest neighbor classifiers (*k*-NN), naïve Bayes classifiers, ensemble classifiers, neural network classifiers. The performance of these classifiers depends on the complexity, processing time, as well as features utilized. In light of the manner in which classifiers work, their performance becomes improved as the complexity increases. An essential consideration in the development of classifiers is achieving high accuracy while minimizing complexity and processing time. In this thesis, the aforementioned classifiers are all utilized and compared with each other to search for the best-performed methods. Thus, in this section, the classifiers mentioned above are discussed in detail.

There are several classification forms in the literature. Splitting a set's elements into two classes and estimating one of two classes is referred to as binary classification whereas estimating one of more than two classes is referred to as multi-class classification. Multi-class classification is also known as multinomial classification. In multi-label classification, each example is classified into one or more classes. It is essential to distinguish multi-class classification from multi-label classification, which requires several labels to be estimated as output for each instance. Lastly, an unbalanced classification problem is one in which the distribution of instances across classes is not equal. In this thesis, multi-label classification is performed by Classification Learner Toolbox.

3.3.1 Decision Trees

Modelling and information extraction from the given data depends heavily on the science underlying the evaluation of huge and complicated datasets to identify meaningful patterns. By creating decision rules for the huge amount of accessible information, decision trees are utilized to extract knowledge. Decision trees are non-parametric algorithms and implementation of decision trees is one of the potential approaches to multistage decision-making. The aim is to build a model that uses several input variables to estimate the value of a target variable [40]. A decision tree is a tree form that resembles a flowchart. The root node represents the node at the top of a tree, it sums up the entire argument or decision. Each internal/decision node represents a test on an attribute, each branch is an outcome of a test, and each terminal/leaf node is a class label indicating the best-suited target value. To acquire the proper output for the input pattern, these tests are filtered down through the tree.

According to the Hunt's Algorithm, the following is the process of building a decision tree from a set S of objects, every single object which belongs to one of the classes $(C_1, C_2, ..., C_n)$, if all the objects in S are members of the same class, namely C_i, the decision tree for S includes a leaf labelled using this class. In any other case, consider T to be a test with potential outcomes $(O_1, O_2, ..., O_k)$. The test divides S into subsets $(S_1, S_2, ..., S_k)$ wherein each object in S has one outcome for T. T represents the decision tree's root, and for each outcome of O, it is constructed a subordinate de-

cision tree by repeating the same steps on the set S_i. Whenever there are two objects within the training set that have the same value for every attribute but also belong to distinct classes, the attributes are sufficient for the classification. It is always feasible to create a test at the second step that leads to a non-trivial partition of S, ensuring that the method terminates if the attributes are sufficient [41]. A simple decision tree is demonstrated in Figure 3.7.

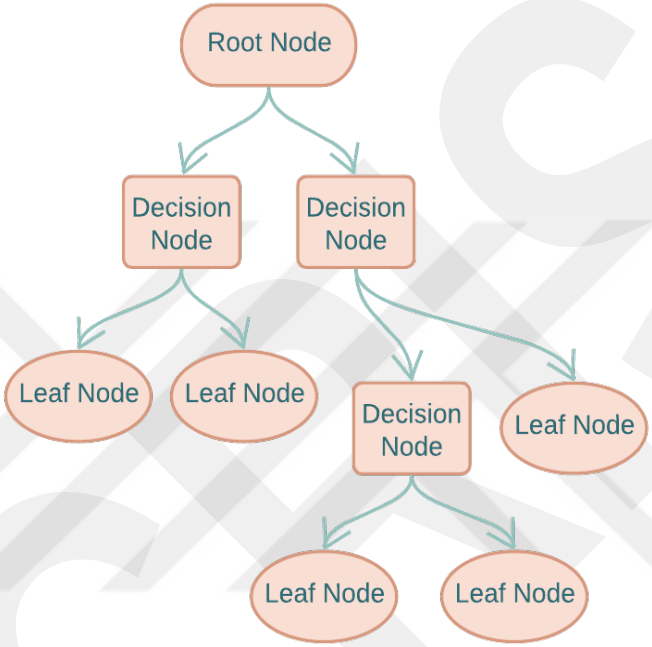

Figure 3.7 A Simple Manually Designed Decision Tree

With ensemble methods, decision tree methods function better. The strategies used in the present work include bagging, boosting, and random forest, together known as ensemble methods. These methods use more than one decision tree for prediction. To get superior prediction performance, it incorporates many decision trees.

3.3.2 Discriminant Analysis

A discriminant analysis's intention is to categorize objects into one of two or more mutually exclusive categories that are exhaustive based on a set of independent variables. In a nutshell, by creating an aggregate of the independent variables, the discriminant analysis concentrates on the association between a number of independent

variables and a dependent variable that is categorical.

Linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA) are the most often used discriminant analysis classifier. Their appeal stems from the fact that they rely on probabilistic underpinnings, making them adequate under the assumptions upon which they have been established. LDA is a classification approach that applies the Bayes theorem. LDA employs a set of features to generate a linear combination of independent variables that best divides the classes and, eventually, classifies data. For classifying observations with a better success rate, LDA frequently employs several features. The assumption is that the feature set might be defined by a multivariate normal distribution, with each feature being normally distributed. A common covariance matrix among all classes and a mean vector particular to each class serve as the cornerstones of the multivariate normal distribution for LDA. It seeks to maintain the location of the observations while establishing a boundary for decision-making to divide the various classes into the data [42], [43]. QDA is a traditional generative probabilistic classification approach. In general, QDA conditional distributions are multivariate Gaussian; posterior distributions are calculated using the Bayes theorem and utilized for classifying data into distinct classes. Originally, QDA parameters are determined by maximizing the joint likelihood for observations and their corresponding class labels. This generative approach, while computationally efficient, does not strive to minimize classification error and is not robust to model misspecification [44].

3.3.3 Logistic Regression Classifiers

One of the most prevalent classifiers used for binary (two-class) classification is logistic regression. Training a classifier that is capable of making a binary decision regarding the class of an unknown input observation is the aim of binary logistic regression. The primary concept behind logistic regression is to employ the Sigmoid function to nonlinearity multivariate linear regression to increase the algorithm's generalization abilities. It refers to a mathematical function with the property of being able to take any real value and transfer it to a number between 0 and 1. The logistic function is another name for the Sigmoid function. The following Equation 3.3

provides the Sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.3}$$

If the value of z goes to positive infinity, the estimated value becomes 1, and if it goes to negative infinity, the estimated value becomes 0. Moreover, a threshold value of 0.5 is chosen generally. This value is referred to as the decision boundary. If the outcome of the Sigmoid function is more than 0.5, it is classified that label as class 1, and if the Sigmoid function is less than 0.5, it is classified as class 0. Thus, the decision becomes "positive sentiment" or "negative sentiment", $P(y = 1 \mid x)$ states the probability of positive sentiment, and $P(y = 0 \mid x)$ states the probability of negative sentiment, wherein input observation x represents by features vector $(x_0, x_1, x_2, ..., x_n)$, and y is the classifier output. The logistic regression classification procedure is as follows. A regression coefficient $(w_0, w_1, w_2, ..., w_n)$ is multiplied by each feature as shown in Equation 3.4:

$$z = w_0x_0 + w_1x_1 + w_2x_2 + ... + w_nx_n \tag{3.4}$$

Equation 3.4 can be simplified as Equation 3.5.

$$z = w^T x \tag{3.5}$$

where w is the row vector is a term for the regression coefficient, and x is the column vector which is the classifier's input data. The curve of the Sigmoid function is provided in Figure 3.8. The major issue in traditional logistic regression is determining the best parameters of the loss function to reduce error. The gradient descent approach is commonly used to address this problem. However, the gradient descent approach is only applicable to functions with convexity, making it ineffective for solving non-convex functions. Logistic regression is a regression approach that describes the association between explanatory variables with discrete response variables. The key difference between linear regression and logistic regression modelling is that the response variable in linear regression is continuous while the response variable in logistic regression is discrete. The distinction is reflected in the parameters and assumptions applied [45].

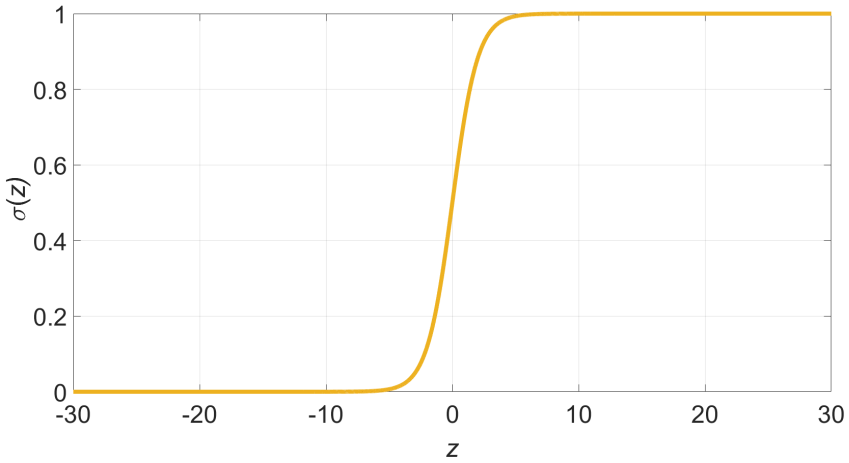

Figure 3.8 Curve of the Sigmoid Function

3.3.4 Support Vector Machines (SVMs)

An approach for classifying both linear and nonlinear data is the support vector machine (SVM). It represents a non-probabilistic and non-parametric discriminative predictor that directly models the mapping from input to the class label in order to derive the classification function. When the data can be separated linearly, the linear SVM finds the maximum separating hyperplane, that is, the one with a maximum distance between the nearest training tuples. This maximum separating hyperplane is often known as the maximum marginal hyperplane (MMH). Figure 3.9 represents the linear SVM in a two-dimensional space of feature.

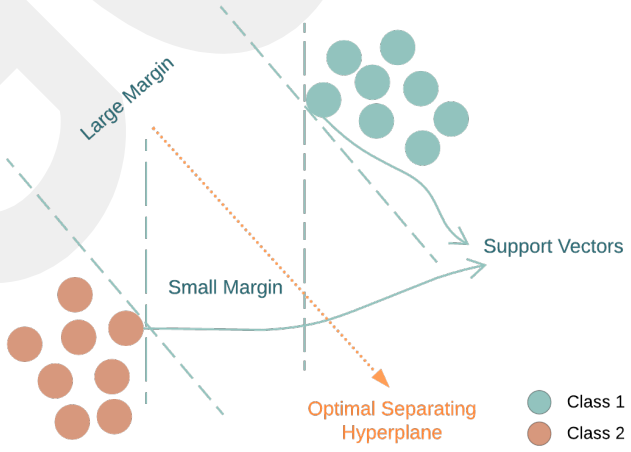

Figure 3.9 Linear SVM Demonstration

The distance of the hyperplane across the data of the closest classes must be the largest. This produces the optimal separating hyperplane. As the large margin widens, so does robustness. A support vector is made up of data from two separate classes that are closest to one other. The margins of the hyperplanes are defined by support vectors. Equation 3.6 states the general formulation for SVM.

$$f(x) = \sum_{i=1}^{N} \alpha i K(x, xi) + b \tag{3.6}$$

where $K(x, x_i)$ indicates the Kernel function of the feature modelling (x, x_i), x_i is the i^{th} training example's input vector, x is the test example's input vector, N states the number of support vectors, α_i is the i^{th} support vectors for $\alpha_i \neq 0$, and b stands for constant bias. Therefore, the classification is determined by analyzing the sign of the formula.

When the data are linearly inseparable, there is no straight line that may be observed to split the classes. In such a scenario, the linear SVM approach may be expanded to build nonlinear SVMs allowing the classification of linearly inseparable data. Such SVMs may detect nonlinear decision boundaries (also known as nonlinear hypersurfaces) within the input space utilizing the linear SVM Kernel function. This is known as the Kernel trick. The SVM is identified as linear, quadratic, or cubic by the Kernel function which uses the given Equation 3.7.

$$K(x, x_i) = (x x_i + c)^n \tag{3.7}$$

where c is a constant equating to 0 for the homogeneous polynomials and 1 for the inhomogeneous polynomials. The value of n equals to 1 for linear Kernel, 2 for quadratic Kernel and 3 for cubic Kernel.

Considering the linearly inseparable data, there are two major steps required. The original input data is converted into a higher dimensional space using nonlinear mapping in the first step, a variety of standard nonlinear mappings may be applied. The next step looks for a linear separation hyperplane in the newly created space once the data has been converted into the new higher space. It yields a quadratic optimization issue that can be addressed using the linear SVM formulation once more. In the original space, the largest marginal hyperplane corresponds to a nonlinear separating hypersurface [46].

3.3.5 *k*-Nearest Neighbor Classifiers (*k*-NN)

A *k*-nearest neighbor (*k*-NN) classifier searches the pattern space for the *k*-training tuples which are closest to the unidentified tuple when given an unidentified tuple. A distance metric such as Euclidean distance is used to determine closeness. The Euclidean distance between two points or tuples, for $X_1 = (x_{11}, x_{12}, ..., x_{1n})$ and $X_2 = (x_{21}, x_{22}, ..., x_{2n})$, is given by Equation 3.8.

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n}(x_{1i} - x_{2i})^2} \tag{3.8}$$

The difference between the relevant values of the attribute in tuple X_1 and tuple X_2 is taken, squared, and accumulated for each numeric attribute. The square root of the entire accumulated distance count is calculated. The unknown tuple is placed in the most prevalent class among its *k*-nearest neighbors for *k*-NN classification. Whenever $k = 1$, the class of the training tuple closest to the unknown tuple in pattern space is assigned to it. A basic process for nominal attributes (non-numeric) is to compare the value of the attribute in tuple X_1 with that in tuple X_2. When the two are the same, the difference is assumed to be 0. If the two are distinct, the difference is considered to be 1. Determining the *k* value in the *k*-NN approach is critical since the *k* value affects the prediction accuracy of the model. It may be identified by experimentation. Initially, with $k = 1$, it is utilized a test set for predicting the classifier's error rate as shown in Figure 3.10.

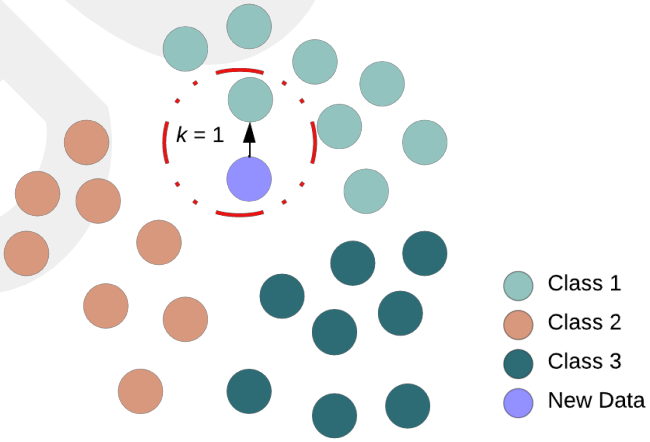

Figure 3.10 *k*-NN Classifier Demonstration

The above method can be performed as many times as k is increased to allow for one additional neighbor. The k value with the lowest error rate may be chosen. Generally, the value of k will increase as the number of training tuples increases. To avoid ties, an odd number is typically chosen. Determining the value of k is not always an experimental procedure. It is suggested with Equation 3.9.

$$k = \sqrt{n} \tag{3.9}$$

where n represents the number of training samples and is considered as $n > 100$. Equation 3.9 specified the optimal k value for all test samples.

Although Euclidean distance is generally used as a distance metric in the literature, there are also different distance metrics such as Cityblock distance, Chebyshev distance and Cosine distance. The one-norm of the distance between two vectors corresponds to the Cityblock distance. For a plane with data point p_1 at (x_1, y_1) and its closest neighbor p_2 at (x_2, y_2), the Cityblock distance is given in Equation 3.10.

$$dist(p_1, p_2) = |x_1 - x_2| + |y_1 - y_2| \tag{3.10}$$

The summation of the absolute differences in each point's coordinates is referred to as the Cityblock distance between two points. Therefore, the sum of the absolute differences between the opposing vector values that form this distance is presented. This formulation may be extended for more than two data points. The Cityblock distance metric is chosen over the Euclidean distance metric when the dimensions of the data points are higher.

Chebyshev distance analyzes the distance between data points as the largest difference between their coordinates. Suppose x and y represent two vectors, with $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$, wherein n is the number of variables. The following Equation 3.11 gives the Chebyshev distance between the two vectors, x and y:

$$dist(x, y) = \max_i |x_i - y_i| \tag{3.11}$$

The angle between two vectors is measured by the Cosine distance metric. It is typically employed when the vector's direction, rather than its magnitude, is of concern. When comparing the vectors, the cosine angle values are taken into consideration.

When the cosine angle value is 1, it indicates that data points are all oriented in the same direction. The data points are oriented in opposing directions if the cosine value is -1. There may be a similarity between the data points while the cosine value is 0. "1 − cosine similarity" gives the Cosine distance for two vectors A and B [47]. Equation 3.12 provides the cosine similarity.

$$\frac{A.B}{||A||\,||B||} \tag{3.12}$$

3.3.6 Naïve Bayes Classifiers

The naïve Bayes classifier, or Bayesian classifier, is a type of probabilistic classification system built on Bayes' Theorem. In naïve Bayes classifier, the feature matrix, and the response/target vector are the two components of the dataset. The dataset's vectors/rows, each of which is made up of the value of dependent features, are all contained in the feature matrix X, and $X = (x_1, x_2, ..., x_d)$ where d is the number of features. The value of the class variable for each row of the feature matrix is contained in the response vector y. The naïve Bayes model is stated in Equation 3.13.

$$P(y \mid X) = \frac{P(X \mid y)\,P(y)}{P(X)} \tag{3.13}$$

where the probability of identifying the class y based on X is $P(y \mid X)$ [48]. If it is assumed that each feature in X is independent of any other feature, then depending on the class y, Equation 3.14 is obtained.

$$P(y \mid x_1, x_2, ..., x_d) = \frac{P(y)\prod_{i=1}^{d} P(x_i \mid y)}{P(x_1)\,P(x_2)\,...P(x_d)} \tag{3.14}$$

The output with the maximum probability must be identified to determine the probability of a given sample for each possible value of the class variable y, as shown in Equation 3.15:

$$\hat{y} = \arg\max_{y} P(y) \prod_{i=1}^{d} P(x_i \mid y) \tag{3.15}$$

The Gaussian naïve Bayes, Multinomial naïve Bayes and Bernoulli naïve Bayes are examples of typical naïve Bayes classifiers. When classifying Gaussian distributed data, Gaussian naïve Bayes assumes that every feature's distribution is Gaussian. For

each class, the standard deviation and mean of the input values can be identified. The Gaussian Probability Density Function which is given in Equation 3.16 is used to compute the probability for new input values [49].

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma y^2}} exp\left(-\frac{(xi - \mu_y)^2}{2\sigma_y^2}\right) \tag{3.16}$$

where $P(x_i \mid y)$ indicates the Gaussian Probability Density Function, σ_y is the standard deviation, x_i represents the unknown input value, and μ_y is the mean value.

Multinomial naïve Bayes is frequently used for dividing classes using discrete features, and the features follow a multinomial distribution. It is suggested to use Bernoulli naïve Bayes to handle binary-valued features, whereby the target variable can only have two possible values, generally 0 or 1. The approach is based on the assumption that each feature is independent of all others and follows a Bernoulli distribution. Bernoulli naïve Bayes formula is provided in Equation 3.17.

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i) \tag{3.17}$$

3.3.7 Ensemble Classifiers

The ensemble classifiers' fundamental idea is to combine and weigh several different classifiers to create one that performs superior to all of them. It is used for increased generalizability [50]. When given a new tuple to be classified, the base classifiers vote by offering a class prediction. Using the votes of the basis classifiers, the ensemble produces a class prediction. When the models are highly diverse, ensembles produce superior outcomes. That is, there should be a low correlation among classifiers.

A common ensemble method for performing classification tasks works as follows. A represents the set of n input attributes, $A = (a_1, a_2, ..., a_n)$, and y represents the variable of the class. The base inducer serves as an induction technique that gets a training set and constructs a classifier that highlights the generalized relation between the input and target attributes. If I denotes an inducer, the expression $M = I(S)$ is used to describe a classifier M that is induced by inducer I over a training set S. The diversity generator component serves to generate diverse classifiers, while the combiner works to combine the classifications of the different classifiers.

For the purpose of building ensembles, it is necessary to differentiate between dependent and independent frameworks. The output of one classifier is utilized to build the next classifier in a dependent framework. As a result, it is feasible to use the information generated in the earlier iterations for guiding learning in the later iterations [51].

Bagging, boosting, and random forest are ensemble classifier methods. One of the widely used ensemble methods, bagging is derived from the acronym for Bootstrap Aggregation. Bootstrap and aggregate are two fundamental ideas of bagging. Bagging is a method to build several base learners or models relying on the bootstrap distribution. By integrating a random sample with a replacement strategy, the bagging method creates bootstrap sampling from training data. Bagging produces the data subsets needed for base learners' training using bootstrap sampling. The bagging method employs random selection with replacement to generate many copies of the original dataset. Then, each set of data is utilized to create a new model, and this group of models is used to build an ensemble. Bagging uses voting for classification, and it is a method used to decrease the variance of the prediction. Bagging may deal with binary classification as well as multi-class classification.

Boosting is a general method for increasing the algorithm's performance. The boosting method is used in a certain order. One base learner's output is fed into another. The method integrates the outputs of many classifiers to form an efficient committee. AdaBoost, Gradient Descent Boosting, and XGboost are the boosting method extensions. Random forest is an advanced algorithm that employs the bagging method. The overfitting issue in decision trees may be addressed by the random forest algorithm. The random forest is an ensemble of decision trees, and it constructs a forest of numerous random decision trees. The ensemble is produced in bagging by taking several distinct samples from the same training dataset and fitting a decision tree for each one. Each decision tree is unique since each sample in the training dataset is unique, which results in slightly varied predictions and prediction errors. Combining the predictions from each decision tree built reduces error compared to adapting a single tree [52].

3.3.8 Neural Network Classifiers

A neural network is roughly defined as a collection of connected input/output units, each with its own weight. The network learns throughout the learning phase by modifying the weights such that it may predict the proper class label. Neural networks are data-driven and capable of self-adapt approaches in that they may adjust themselves to the data with no specific indication of the core model's functional or distributional form. Since neural networks are nonlinear models, they are adaptable to model real-world complicated relations, and they may calculate posterior probabilities, which serve as the basis to construct classification rules [53].

On the other hand, as neural networks require a long training time, they are better suited to scenarios in which this is feasible. Moreover, the interpretability of neural networks has been challenged. It is challenging to grasp the symbolic significance of learning weights and "hidden units" within the network.

Backpropagation is the most widely used neural network algorithm. The neural network on which the backpropagation algorithm operates is a multilayer feed-forward network. An input layer, one or more hidden layers, and an output layer compose a multilayer feed-forward neural network. Figure 3.11 demonstrates the multilayer feed-forward neural network.

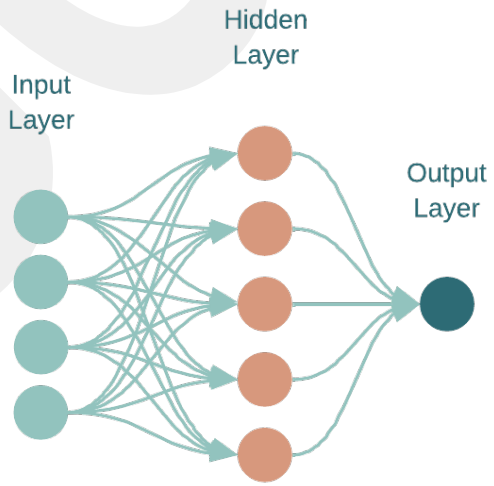

Figure 3.11 Multilayer Feed-Forward Neural Network Demonstration

It repeatedly learns a set of weights for predicting the class label of tuples. The network's inputs are attributes that are measured for each training tuple. The inputs are fed synchronously to the units that comprise the input layer. These inputs are routed through the input layer before being weighted and fed synchronously to the units referred to as the hidden layer. The hidden layer units' outputs may be fed into another hidden layer. Even though only one is typically utilized in practice, the total number of hidden layers is variable. The last hidden layer's weighted outputs pass into the output layer, which shows the network's prediction of given tuples. The weighted sum of the outputs from units within the preceding layer is the input for each output unit. The class prediction may be modelled as a nonlinear combination of inputs utilizing multilayer feed-forward neural networks.

The results and conclusion are discussed in the remainder of the thesis, after the mathematical representation of proposed fingerprinting-based localization, the dataset construction by the experimental process implemented with the help of related IQRF wireless technology, and additionally the classification algorithms.

CHAPTER 4

RESULTS

In this chapter, the classification process using the Classification Learner Toolbox of Matlab is addressed. Accordingly, the procedure of splitting the overall dataset as training, validation and test datasets to evaluate classification algorithms and the performance results of the different classifiers using these datasets are given. This chapter can be considered the online phase of the proposed transmitter localization approach.

4.1 Execution of the Classification Algorithms

After the offline fingerprint database construction, the overall dataset is divided into train, validation and test datasets, respectively, in order to proceed to the online prediction phase. Dataset splitting is a strategy that is widely regarded as essential for eliminating or reducing bias in ML models. This is always done to avoid producing overfitting that performs badly on actual test data [54]. A training dataset is for improving the setting parameters of the specified classifier and a test dataset is for validating the final classifier's generalization performance. The validation set is an additional preference: however, it is not compulsory.

In this thesis, it is divided 70% of the 192000 rows of data are divided into a training dataset and 30% into a test dataset. Thus, the process was taken into account with the training dataset of 134400 rows and the test dataset of 57600 rows. Moreover, 5-fold cross-validation was used to eliminate the problem of over-fitting. Cross-validation is a validation technique for making near-optimal use of available data by repeating training and as well as testing classifiers based on different subsets of data, often using

a large training set and a small validation set in every iteration. Therefore, the dataset was split into 5 subsets which are utilized for validation.

All the classifiers in the Classification Learner Toolbox were trained with the specified training dataset, and then test accuracy analysis was performed with the test dataset. At the same time, validation accuracy values were obtained with the applied 5-fold cross-validation. Meanwhile, the prediction speed and training time of these classifiers were investigated.

4.2 Results of the Classification Algorithms

After the execution of the classification algorithms is done, "Bagged Trees", "Weighted *k*-NN", and "Medium Gaussian SVM" methods performed best in terms of validation and test accuracy. They performed with over 95% accuracy for both validation and testing. The validation and test accuracy results of the 20 models are given in Figure 4.1.

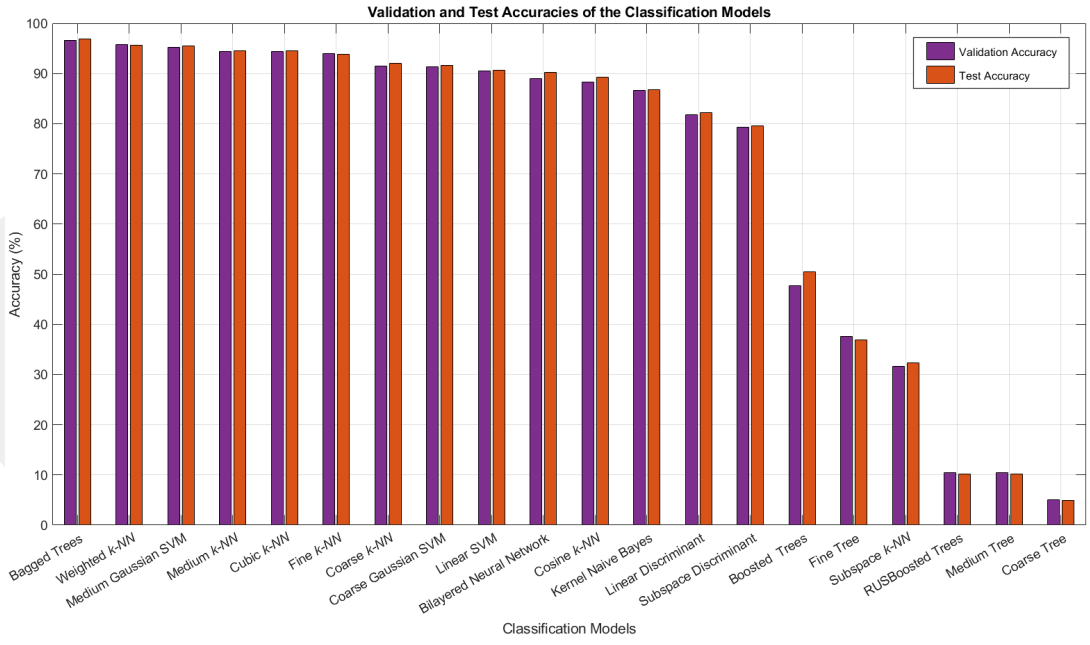

Figure 4.1 Validation and Test Accuracies of the Classification Models

Besides validation and test accuracies, prediction speed and training time characteristics were also observed to compare the performance and usability of the classification

algorithms. Observations with all characteristics are given in Table 4.1.

Table 4.1 Results of Classifiers

Preset	**Validation Accuracy**	**Test Accuracy**	**Prediction Speed**	**Training Time**
Bagged Trees	96.63%	96.87%	1919.9 obs/sec	570.85 sec
Weighted *k*-NN	95.71%	95.69%	37615 obs/sec	28.043 sec
Medium Gaussian SVM	95.17%	95.44%	196.01 obs/sec	8287.2 sec
Medium *k*-NN	94.42%	94.59%	79995 obs/sec	16.108 sec
Cubic *k*-NN	94.40%	94.57%	20831 obs/sec	37.661 sec
Fine *k*-NN	93.94%	93.79%	95562 obs/sec	17.365 sec
Coarse *k*-NN	91.42%	92.03%	13642 obs/sec	62.999 sec
Coarse Gaussian SVM	91.33%	91.64%	165.11 obs/sec	8472.1 sec
Linear SVM	90.58%	90.62%	196.28 obs/sec	7870 sec
Bilayered Neural Network	88.98%	90.21%	1.2702e+05 obs/sec	8409.8 sec
Cosine *k*-NN	88.29%	89.24%	3282.8 obs/sec	164.81 sec
Kernel Naïve Bayes	86.58%	86.72%	128.39 obs/sec	4944.1 sec
Linear Discriminant	81.82%	82.16%	64810 obs/sec	19.836 sec
Subspace Discriminant	79.32%	79.60%	1513.7 obs/sec	521.71 sec
Boosted Trees	47.72%	50.52%	1618.8 obs/sec	891.23 sec
Fine Tree	37.61%	36.91%	1.5049e+05 obs/sec	14.963 sec
Subspace *k*-NN	31.69%	32.31%	1680.3 obs/sec	536.98 sec
RUSBoosted Trees	10.39%	10.23%	2414.1 obs/sec	763.82 sec
Medium Tree	10.40%	10.21%	1.2766e+05 obs/sec	19.191 sec
Coarse Tree	5.04%	4.98%	1.1716e+05 obs/sec	16.663 sec

The prediction speed is calculated by dividing the total number of predictions generated by a method by the time it takes to make the predictions. It is expressed in terms of observations per second (obs/sec). The time it takes each method to construct and train a model for classification is referred to as training time [54].

Although "Bagged Trees" has higher validation and test accuracies, the "Weighted k-NN" method is superior among the top 3 methods, considering the higher prediction speed and lower training time than others. An improved version of the "k-NN" classification is the "Weighted k-NN" classification. It may avoid the drawback whereby the recognition accuracy of "k-NN" is highly influenced by the neighborhood size of k. With "Weighted k-NN" classification, the test sample's classification result is closer to the result of the training sample, since various weights are assigned to the neighbor samples depending on how similar each neighbor sample is to the test sample. The sample weight is determined using the inverse distance square weighting method, while the sample distance between samples is measured using the Euclidean distance. The votes of the nearest neighbor are weighted depending on the calculated distance; since the weight is inversely related to the distance squared, smaller distances lead to larger weights. The calculation formula of the weight of "k-NN" is given by Equation 4.1 and Equation 4.2, respectively.

$$h = \sqrt{(x - x_j)^2 + (y - y_j)^2} \tag{4.1}$$

$$W_i = \frac{1}{n}\left(\frac{1}{h_1^2} + \frac{1}{h_2^2} + ... + \frac{1}{h_n^2}\right) \tag{4.2}$$

where W_i is the sample weight, n represents the number of samples of i^{th} type within the k neighbor samples, and h is the distance from the unidentified sample to the training sample [55]. In such a manner, the accuracy of recognition is enhanced while the sensitivity to the value of k decreases, accordingly, improving the accuracy of the recognition results.

Figure 4.2 indicates the receiver operating characteristic (ROC) curve of the "Weighted k-NN" method belonging to the first 20 classes out of 96 classes. True positive rate (TPR) and false positive rate (FPR) are typical components of ROC curves. This suggests that the "ideal" point is in the plot's upper left corner, with an FPR of 0 and a TPR of 1. The ideal TPR is 1, indicating that there is a particular threshold at which all positives are labeled as positives. The optimal FPR is 0, indicating that there is a specified threshold above which no negatives are identified as positives. As a result, (0,1) is the most optimum point. The ability of a classifier to differentiate between classes is measured by the Area Under the Curve (AUC), which serves for a summary

of the ROC curve. The model performs better at differentiating across classes the higher the AUC.

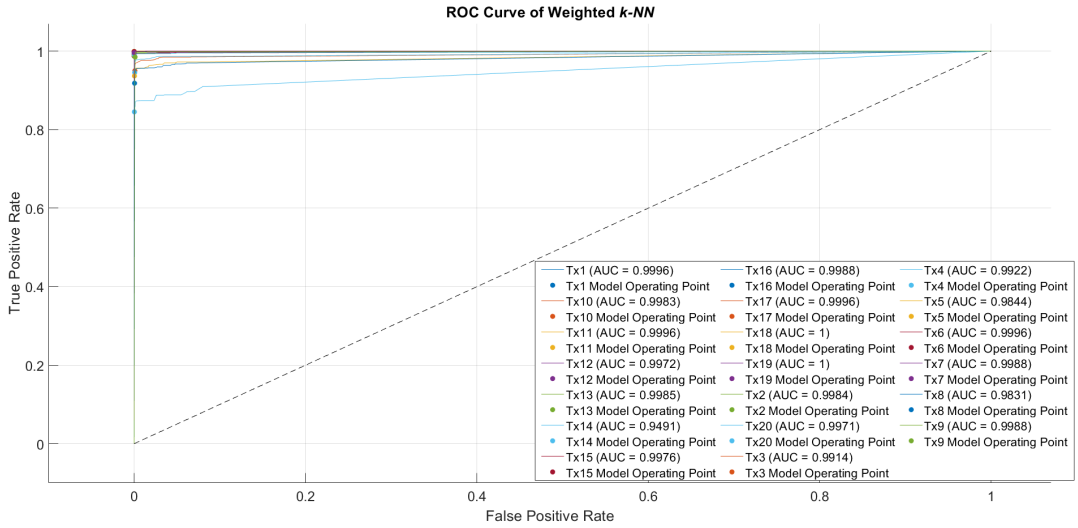

Figure 4.2 ROC Curve of the Weighted k-NN Method

True positive (TP) is an outcome in which the model accurately estimates the positive class. FP stands for false positive, which the model estimates the positive class inaccurately. TPR can also be referred to as sensitivity or recall while FPR is also known as fall-out. TPR and FPR formulas are given in Equation 4.3 and Equation 4.4, respectively.

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{4.3}$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{4.4}$$

FN stands for false negative, which is an outcome in which the model estimates the negative class inaccurately. True negative (TN) is an outcome in which the model accurately estimates the negative class. ROC curves are frequently applied in binary classification, where it is apparent what the TPR and FPR are. A notion of TPR or FPR can be obtained in the context of multiclass classification following binarizing the output.

Following the "Weighted k-NN" method, the method with the most efficient performance in terms of prediction speed and training time is the "Bagged Trees" method.

"Bagged Trees" is an ensemble classifier approach in which many versions of a classifier, in the present case decision trees, are constructed, each with its own prediction. The final outcome is the average of all of these unique predictions. Individualized decision trees have a tendency to overfit. Bagging mitigates the impacts of overfitting while improving generalization. The 3^{rd} method that gives over 95% validation and test accuracies is the "Medium Gaussian SVM". Although this method gives high accuracies, it is more disadvantageous due to slower estimation speed and longer training time than the previously mentioned "Weighted k-NN" and "Bagged Trees" methods. The "Medium Gaussian SVM" method is one of the methods that use the Kernel trick for data that cannot be separated linearly. Some specific RSSI values of the overall dataset obtained at the end of the experimental measurements cannot be separated linearly since they are very close to each other especially in the near grids.

The model hyperparameters of the addressed three methods which give the best validation and test accuracies are presented in Table 4.2. The hyperparameters given in Table 4.2 are the default settings implemented by the Classification Learner Toolbox. No changes have been made to these hyperparameters.

Table 4.2 Hyperparameters of Models

Model	**Hyperparameters**
Bagged Trees	Maximum number of splits: 134399 Number of learners: 30
Weighted k-NN	Number of neighbors (k): 10 Distance metric: Euclidean Distance weight: Squared Inverse
Medium Gaussian SVM	Kernel function: Gaussian Kernel scale: 2

For the "Bagged Trees" model, the number of learners represents the total number of decision trees. The bigger the value, the more subsets of data will be separated into to train each decision tree. Maximum number of splits is a splitting variable for each tree depth that will be used to separate the data. It essentially defines the decision tree's depth. For the "Weighted k-NN" model, the number of neighbors k is assigned as 10, Euclidean is applied as distance metric and additionally squared inverse is ap-

plied as distance weight. Therefore, it gives more weight to the points which are nearby and less weight to the points which are farther away. For the "Medium Gaussian SVM" model, Gaussian is implemented as a Kernel function. The radial basis function (RBF) Kernel, also known as the Gaussian Kernel, is arguably the most extensively used Kernel. The Gaussian RBF Kernel function is given by Equation 4.5.

$$K(x, x_i) = exp\left(-\frac{\| x - x_i \|^2}{2L^2}\right) \tag{4.5}$$

where $K(x, x_i)$ indicates the Kernel function of the feature modelling (x, x_i), x_i is the i^{th} training example's input vector as explained in Equation 3.6. In Equation 4.5, $\| x - x_i \|^2$ is the squared distance calculated by Euclid among the two feature vectors and L the Kernel length scale. The Kernel scale is a scaling parameter of the input data which is before being supplied to the Kernel function.

CHAPTER 5

CONCLUSION

In this thesis, it is aimed to perform RFF-based emitter localization by gathering RSSI fingerprint measurements in 868 MHz utilizing IQRF Technology as an alternative that can be used in the applications of short-range IoT-based smart cities. These fingerprint values were selected as valuable features to be used later in the online localization process.

Firstly, the local outdoor environment where the experimental measurements will be performed was determined. The size of the area was selected as 2400 m^2. This area is divided into square grids of 5 m on each side in order to transform the existing localization problem into a classification problem. A total of 96 grids were obtained. Each of these grids was treated as one class and thus, the present localization problem was transformed into a multi-class classification problem. Meanwhile, 4 of the IQRF-TR modules with embedded antennas were configured as receivers and 1 as a transmitter to complete the offline localization phase. Receivers were placed at each corner of the determined 2400 m^2 area and used to receive data packets and measure their respective RSSI values. The receiver, on the other hand, was moved manually from the 1^{st} grid to the 96^{th} grid respectively and sent 3000 data packets from each grid. The RSSI fingerprint features collected and recorded at this stage built the fingerprint database, this process is specifically referred to as the offline data acquisition phase. Secondly, data packets sent from each grid were optimized as 2000 instead of 3000 before proceeding to the steps in the online localization phase. Thus, a fingerprint dataset of 192000 rows was created to use the Classification Learner Toolbox in Matlab. This dataset is randomly divided into 70% training and 30% test datasets. In addition, 5-fold cross-validation was applied to avoid overfitting problems. There-

fore, it was aimed to obtain effective accuracies from the models used. Then, the classification process was performed with all the classifier models in Matlab, and the validation accuracy, test accuracy, prediction speed and training time characteristics of these classifiers were examined to compare with each other. The purpose was to identify the models that give the highest performance results according to these characteristics, in classification-based localization implementation.

According to the observations, 20 of more than 45 different classifiers provided by Matlab resulted in the above-mentioned characteristics during the training process, and the rest failed. Of these 20 models, models that yielded results with validation and test accuracy of over 95% were determined. Accordingly, from higher to lower accuracy rates; The "Bagged Trees", "Weighted k-NN" and "Medium Gaussian SVM" models emerged as the best performing models. Therefore, it was observed that these models gave remarkable results in multi-class classification of data that cannot be separated linearly. Among them, the "Weighted k-NN" model was chosen as the most efficient model, considering that the prediction speed was the highest and the training time was the lowest. The "Medium Gaussian SVM" model, on the other hand, has time complexity due to its slower estimation speed and longer train speed, although it provides high accuracy rates.

At the end of the thesis work, it was found that if the signal features are to be employed for reliable location identification, they need to satisfy certain criteria. First and foremost, the signal feature must be robust within the space of localization. This implies that similar fingerprints should correlate to neighboring locations; otherwise, a minor error in matching fingerprints may result to a big error in localization. Second, the fingerprint must be stable in time, that is, it should not vary much from the time the database is built to the time the localization executes; otherwise, the fingerprint of the unidentified position can differ significantly from the recorded fingerprint, resulting in large inaccuracies.

This thesis mainly focuses on RFF-based emitter localization in a local environment. In this context, the applicability of supervised learning algorithms based on classification was tested in the localization process. This thesis secondly investigates the potential use of IQRF Technology as an option for IoT solutions. It has been re-

vealed that IQRF Technology can be used in short-range IoT applications in smart cities. Thus, the IQRF is an applicable approach in local area networks (LANs). It has emerged within the scope of this thesis that this technology can be adopted in localization systems in the local area and this localization problem can be solved by machine learning (ML) techniques. It is observed that the embedded antennas of IQRF-TRs provide sufficient coverage in the interested local area. In future work, external antennas may be used to extend the range even further. Thus, by obtaining RSSI measurements in an expanded outdoor region, the localization process may be performed. Furthermore, IQRF technology or another wireless communication technology operating at 868 MHz may be operated in an outdoor environment that has non-line-of-sight (NLoS) links in order to perform localization, and the results can be compared with the LoS link.

REFERENCES

[1] G. Mao, *Localization Algorithms and Strategies for Wireless Sensor Networks Monitoring and Surveillance Techniques for Target Tracking*. Hershey PA: Information Science Reference, 2009.

[2] S. P. Singh and S. C. Sharma, "Range free localization techniques in wireless sensor networks: A Review," *Procedia Computer Science*, vol. 57, pp. 7–16, 2015.

[3] I. Rafiq, A. Mahmood, S. Razzaq, S. H. Jafri, and I. Aziz, "IOT applications and challenges in smart cities and services," *The Journal of Engineering*, vol. 2023, no. 4, 2023.

[4] S. N. Ghorpade, M. Zennaro, and B. S. Chaudhari, "Localization approaches for internet of things," *Optimal Localization of Internet of Things Nodes*, pp. 17–50, 2021.

[5] A. Voisard and J. H. Schiller, *Location-Based Services*. San Francisco, CA: Morgan Kaufmann Publishers, 2004.

[6] D. Dogan, Y. Dalveren and A. Kara, "A Mini-Review on Radio Frequency Fingerprinting Localization in Outdoor Environments: Recent Advances and Challenges," *2022 14th International Conference on Communications (COMM)*, Bucharest, Romania, 2022, pp. 1-5.

[7] M. Alfakih, "Indoor and Outdoor Localization", Doctoral Thesis, University of Science and Technology Mohamed Boudiaf, Algeria, 2019.

[8] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran and S. Guizani, "Internet-of-Things-based smart cities: Recent advances and challenges", *IEEE Communication Magazine*, vol. 55, no. 9, pp. 16-24, Sep. 2017.

[9] M. Bouzidi, Y. Dalveren, F. A. Cheikh, and M. Derawi, "Use of the IQRF technology in internet-of-things-based Smart Cities," *IEEE Access*, vol. 8, pp. 56615–56629, 2020.

[10] K. Lin, W. Wang, Y. Bi, M. Qiu, and M. M. Hassan, "Human localization based on inertial sensors and fingerprints in the industrial internet of things," *Computer Networks*, vol. 101, pp. 113–126, 2016.

[11] D. Zhang, L. Yang, M. Chen, S. Zhao, M. Guo and Y. Zhang, "Real-time locating systems using active rfid for internet of things", *Systems Journal IEEE*, no. 99, pp. 1-10, 2014.

[12] H. Sallouha, A. Chiumento and S. Pollin, "Localization in long-range ultra narrow band IoT networks using RSSI," *2017 IEEE International Conference on Communications (ICC)*, Paris, France, pp. 1-6, 2017.

[13] K. Lin, M. Chen, J. Deng, M. M. Hassan and G. Fortino, "Enhanced Fingerprinting and Trajectory Prediction for IoT Localization in Smart Buildings," in *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1294-1307, 2016.

[14] T. Janssen, M. Aernouts, R. Berkvens and M. Weyn, "Outdoor Fingerprinting Localization Using Sigfox," *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Nantes, France, 2018.

[15] W. Choi, Y.-S. Chang, Y. Jung, and J. Song, "Low-power LORA signal-based outdoor positioning using fingerprint algorithm," *ISPRS International Journal of Geo-information*, vol. 7, no. 11, p. 440, Nov. 2018.

[16] P. Singh, N. Mittal, and R. Salgotra, "Comparison of range-based versus range-free WSNs localization using adaptive SSA algorithm," *Wireless Networks*, vol. 28, no. 4, pp. 1625–1647, Mar. 2022.

[17] Y. Zhang, Z. Fang, R. Li and W. Hu, "The Design and Implementation of a RSSI-Based Localization System," *2009 5th International Conference on Wireless Communications, Networking and Mobile Computing*, Beijing, China, 2009.

[18] D. C. Hogg, "Fun with the Friis free-space transmission formula," in *IEEE Antennas and Propagation Magazine*, vol. 35, no. 4, pp. 33-35, Aug. 1993.

[19] A. Bensky, *Wireless Positioning Technologies and Applications*. Boston: Artech House, 2016.

[20] S. M. Asaad and H. S. Maghdid, "A Comprehensive Review of Indoor/Outdoor Localization Solutions in IoT era: Research Challenges and Future Perspectives," *Computer Networks*, vol. 212, p. 109041, Jul. 2022.

[21] S. Messous and H. Liouane, "Online Sequential DV-HOP Localization Algorithm for wireless sensor networks," *Mobile Information Systems*, vol. 2020, pp. 1–14, Oct. 2020.

[22] S. A. Hosseinirad, M. Niazi, J. Pourdeilami, S. K. Basu, and A. A. Pouyan, "On improving APIT algorithm for better localization in WSN," *DOAJ (DOAJ: Directory of Open Access Journals)*, vol. 2, no. 2, pp. 97–104, Jul. 2014.

[23] H. Chen, P. Huang, M. Martins, H. C. So, and K. Sezaki, "Novel centroid localization algorithm for three-dimensional wireless sensor networks," *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, 2008.

[24] W. Fubao, S. Long, and R. Fengyuan, "Self-Localization systems and algorithms for wireless sensor networks," *Journal of Software*, vol. 16, no. 5, p. 857, Jan. 2005.

[25] C. Liu, K. Wu, "Performance evaluation of range-free localization methods for wireless sensor networks," *PCCC 2005. 24th IEEE International Performance, Computing, and Communications Conference*, 2005.

[26] X. Yang, X. Wang and W. Wang, "An Improved Centroid Localization Algorithm for WSN," *2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, Chongqing, China, 2018, pp. 1120-1123.

[27] D. Li and Z. Niu, "A Wireless Fingerprint Positioning Method Based on Wavelet Transform and Deep Learning," *ISPRS Int. J. Geo-Inf.*, vol. 10, no. 7, p. 442, 2021.

[28] L. Wu, C.-H. Chen, and Q. Zhang, "A mobile positioning method based on deep learning techniques," *Electronics*, vol. 8, no. 1, p. 59, 2019.

[29] M. N. de Sousa, R. Sant'Ana, R. P. Fernandes, J. C. Duarte, J. A. Apolinário, and R. S. Thomä, "Improving the performance of a radio-frequency localization system in adverse outdoor applications," *EURASIP J. Wirel. Commun. Netw.*, vol. 2021, no. 1, pp. 1–26, 2021.

[30] R. D. Timoteo and D. C. Cunha, "A scalable fingerprint-based angle-of-arrival machine learning approach for cellular mobile radio localization," *Comput. Commun.*, vol. 157, pp. 92–101, 2020.

[31] M. N de Sousa and R. S Thomä, "Enhancement of localization systems in NLOS urban scenario with multipath ray tracing fingerprints and machine learning," *Sensors*, vol. 18, no. 11, p. 4073, 2018.

[32] D. Li and Y. Lei, "Deep learning for fingerprint-based outdoor positioning via LTE networks," *Sensors*, vol. 19, no. 23, p. 5180, 2019.

[33] Y. Lei, D. Li, H. Zhang, and X. Li, "Wavelet feature outdoor fingerprint localization based on resnet and deep convolution gan," *Symmetry*, vol. 12, no. 9, p. 1565, 2020.

[34] Y. Han, H. Ma, and L. Zhang, "An efficient RF fingerprint positioning algorithm based on uneven grid layout," In *Proc. ICWMMN, Beijing*, pp. 250–254, 2015.

[35] K. K. Naik and M. N. G. Prasad, "A system for locating users of WLAN using statistical mapping in indoor and outdoor environment-LOIDS," *2008 Fourth International Conference on Wireless Communication and Sensor Networks*, Indore, India, 2008.

[36] Q. B. Vo and P. De, "A Survey of Fingerprint-Based Outdoor Localization," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 491–506, Jan. 2016.

[37] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar, "A review of radio frequency fingerprinting techniques," *IEEE J. Radio Freq. Identif.*, vol. 4, no. 3, pp. 222–233, 2020.

[38] A. Behboodi, "A mathematical model for fingerprinting-based localization algorithms," *ArXiv*, Oct. 2016.

[39] I. Calvo, J. R. Gil-Garcia, I. Recio, A. López, and J. Quesada, "Building IoT Applications with Raspberry Pi and Low Power IQRF Communication Modules," *Electronics*, vol. 5, no. 4, p. 54, Sep. 2016.

[40] G. Kesavaraj and S. Sukumaran, "A study on classification techniques in data mining," *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, India, 2013.

[41] J. R. Quinlan, “Decision trees and decision-making,” in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, no. 2, pp. 339-346, 1990.

[42] J. Wen et al., “Robust Sparse Linear Discriminant Analysis,” in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 390-403, Feb. 2019.

[43] S. Guo and H. Tracey, “Discriminant Analysis for Radar Signal Classification,” in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 4, pp. 3134-3148, Aug. 2020.

[44] W. Cao and R. M. Haralick, “Quadratic Discriminant Revisited,” *2014 22nd International Conference on Pattern Recognition*, Stockholm, Sweden, 2014.

[45] X. Zou, Y. Hu, Z. Tian and K. Shen, “Logistic Regression Model Optimization and Case Analysis,” *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, Dalian, China, 2019.

[46] S. Ghosh, A. Dasgupta and A. Swetapadma, “A Study on Support Vector Machine based Linear and Non-Linear Pattern Classification,” *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, Palladam, India, 2019.

[47] S. Nayak, M. Bhat, N. S. Reddy, and B. A. Rao, “Study of distance metrics on k- nearest neighbor algorithm for star categorization,” *Journal of Physics*, vol. 2161, no. 1, p. 012004, Jan. 2022.

[48] P. Kaviani and S. Dhotre, “Short Survey on Naive Bayes Algorithm,” *ResearchGate*, Nov. 2017.

[49] C. Bemando, E. Miranda and M. Aryuni, “Machine-Learning-Based Prediction Models of Coronary Heart Disease Using Naïve Bayes and Random Forest Algorithms,” *2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)*, Pekan, Malaysia, 2021.

[50] M. S. Sefat, M. Shahjahan, M. Rahman and D. Valles, “Ensemble Training with Classifiers Selection Mechanism,” *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, New York, NY, USA, 2021.

[51] L. Rokach, “Ensemble-based classifiers,” *Artificial Intelligence Review*, vol. 33, no. 1–2, pp. 1–39, Nov. 2009.

[52] L. Halawi, A. S. Clarke, and K. W. George, “Decision Trees and Ensemble,” in *Harnessing the Power of Analytics*, 2022.

[53] H. Gish, “A probabilistic approach to the understanding and training of neural network classifiers,” *International Conference on Acoustics, Speech, and Signal Processing*, 2002.

[54] A. Y. Sangodoyin, M. O. Akinsolu, P. Pillai, and V. Grout, “Detection and Classification of DDoS Flooding Attacks on Software-Defined Networks: A Case Study for the Application of Machine Learning,” *IEEE Access*, vol. 9, pp. 122495–122508, Jan. 2021.

[55] Q.-F. Wang, S. Wang, B. Wei, W. Chen, and Y. Zhang, “Weighted K-NN Classification Method of Bearings Fault Diagnosis With Multi-Dimensional Sensitive Features,” *IEEE Access*, vol. 9, pp. 45428–45440, Jan. 2021.