DEMOCRATIC AND POPULAR REPUBLIC OF ALGERIA
UNIVERSITY OF SAAD DAHLEB BLIDA 1
Faculty of Science
Department of Computer Science

Thesis

Presented for obtaining the Master's degree
In computer science
Option: Computer Systems and Networks

Realised by:

Feddamadji Baya & Amaouche Lina

Proprietary location system based on cell phone tower signals and located devices.

Graduated on Tuesday 02/07/2024, In front of the jury composed of:

Dr. Rezzoug	USDB	**President**
Dr. Gabghoub	USDB	**Examiner**
Dr. Mohamed Benyahya	USDB	**Supervisor**
Dr. Allal Tiberkak	UYFM	**Supervisor**

2023/2024

Acknowledgements

We begin by thanking Allah, the merciful and compassionate, for blessing us with the patience to complete this work.

We express our profound gratitude to our supervisors, Dr. Allal Tiberkak and Dr. Benyahia Mohamed, for their invaluable guidance and continuous support. Their patience and assistance have been essential in developing our ideas and organizing our thesis, and their constructive advice has greatly enhanced the writing process.

We would like to thank the members of the jury for their attentive reading of our thesis and for their valuable remarks that will help us improve our work.

We also appreciate the support of the Department of Computer Science at Saad Dahleb University, Blida 1, throughout this program.

Lastly, we wish to thank our families our parents, sisters, brothers, and all our relatives and friend for their unwavering support and encouragement during this journey.

Dedication

I dedicate this work to my beloved parents, whose unconditional love, sacrifices, and unwavering belief in me have been my greatest source of strength and inspiration. To my dear sister Meriem and my brothers Othmane, Islem, and Ali, thank you for your encouragement that has been my source of strength in every difficulty.

A special dedication goes to my wonderful friends Bayane, Narimane, Sirine, Ilhem, Asma Bersali, Yasmina, and Abir. Your friendship and moral support have been essential in overcoming the challenges I faced.

I extend my heartfelt gratitude to all my other friends and loved ones who have supported me throughout this journey. Your help, encouragement, and presence have been precious to me. Thank you all for making this achievement possible.

Baya Feddamadji

Abstract

In today's world, the demand for reliable and efficient localization systems has become more important. Traditional systems like GPS often fall short in complex environments such as buildings or conflict zones, where reliable positioning can be challenging. This limitation highlights the need for precise, cost-effective localization solutions that can function seamlessly in various scenarios. To address this gap, we propose a proprietary indoor positioning system based on cell phone tower signals and located devices, offering an alternative that is both reliable and accessible.

Our system uses machine learning and deep learning techniques to process data collected from various mobile devices. By analyzing signals from cellular towers, it can accurately predict the location of users. This method enhances location accuracy in different environments while providing an alternative to traditional GPS systems. Our work addresses the growing demand for reliable location solutions.

Keywords:
localization systems, indoor positioning, machine learning, deep learning, cellular.

Résumé

Dans le monde d'aujourd'hui, la demande pour des systèmes de localisation fiables et efficaces est devenue de plus en plus importante. Les systèmes traditionnels comme le GPS montrent souvent leurs limites dans des environnements complexes tels que les bâtiments ou les zones de conflit, où il peut être difficile d'obtenir une position fiable. Cette limitation met en évidence la nécessité de solutions de localisation précises et économiques, capables de fonctionner de manière fluide dans divers scénarios. Pour répondre à ce besoin, nous proposons un système de positionnement en intérieur propriétaire basé sur les signaux des tours de téléphonie mobile et des dispositifs localisés, offrant une alternative à la fois fiable et accessible.

Notre système utilise des techniques d'apprentissage automatique et d'apprentissage profond pour traiter les données collectées à partir de divers appareils mobiles. En analysant les signaux des tours cellulaires, il peut prédire avec précision la position des utilisateurs. Cette méthode améliore la précision de la localisation dans différents environnements tout en offrant une alternative aux systèmes GPS traditionnels. Notre travail répond à la demande croissante de solutions de localisation fiables.

Mots clés:
systèmes de localisation, positionnement en intérieur, apprentissage automatique, apprentissage profond, cellulaire.

الملخص

في عالم اليوم، أصبحت الحاجة إلى أنظمة تحديد المواقع الموثوقة والفعالة أكثر أهمية. غالبًا ما تظهر الأنظمة التقليدية مثل نظام تحديد المواقع العالمي (GPS) حدودها في البيئات المعقدة مثل المباني أو مناطق النزاع، حيث قد يكون من الصعب الحصول على موقع موثوق. تبرز هذه المحدودية الحاجة إلى حلول تحديد مواقع دقيقة واقتصادية قادرة على العمل بسلاسة في سيناريوهات متعددة. استجابةً لهذه الحاجة، نقترح نظام تحديد المواقع الداخلي القائم على إشارات أبراج الهواتف المحمولة والأجهزة المحددة، مما يوفر بديلاً موثوقًا وقابلًا للوصول.

يستخدم نظامنا تقنيات التعلم الآلي والتعلم العميق لمعالجة البيانات التي يتم جمعها من مختلف الأجهزة المحمولة. من خلال تحليل إشارات الأبراج الخلوية، يمكنه التنبؤ بدقة بموقع المستخدمين. تعمل هذه الطريقة على تحسين دقة تحديد المواقع في بيئات مختلفة بينما تقدم بديلاً للأنظمة التقليدية مثل GPS. يساهم عملنا في تلبية الطلب المتزايد على حلول تحديد المواقع الموثوقة.

الكلمات المفتاحية:
أنظمة تحديد المواقع، تحديد المواقع الداخلي، التعلم الآلي، التعلم العميق، إشارات الأبراج الخلوية.

Contents

List of Figures

List of Tables

List of Acronyms

Acronym	Meaning
LBS	Location-Based Services
GPS	Global Positioning System
IoT	Internet of Things
KNN	K-Nearest Neighbors
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
RF	Radio Frequency
ML	Machine Learning
DL	Deep Learning
RSS	Received Signal Strength
AP	Access Point
UWB	Ultra-Wideband
RSRP	Reference Signal Received Power
RSRQ	Reference Signal Received Quality
SINR	Signal-to-Interference-plus-Noise Ratio
LTE	Long-Term Evolution
WCDMA	Wideband Code Division Multiple Access
CDMA	Code Division Multiple Access
MRSE	Mean Root Squared Error
MSE	Mean Squared Error
MAE	Mean Absolute Error
TOF	Time of Flight
LOS	Line of Sight
IR	Infrared
VLC	Visible Light Communications
AOA	Angle of Arrival
LIDAR	Light Detection and Ranging
RTT	Round-Trip Time
TDOA	Time Difference of Arrival
ADOA	Angle Difference of Arrival
PDOA	Phase Difference of Arrival
POA	Phase of Arrival
CSI	Channel State Information
GSM	Global System for Mobile Communications
UMTS	Universal Mobile Telecommunications System
WLAN	Wireless Local Area Network
RSSI	Received Signal Strength Indicator
PIR	Passive Infrared Sensor
BLE	Bluetooth Low Energy
SLAM	Simultaneous Localization and Mapping
FM	Frequency Modulation
RFID	Radio Frequency Identification
WSN	Wireless Sensor Network
DR	Dead Reckoning
MM	Map Matching

General Introduction

In today's world, location-based services (LBS) have become an integral part of daily life, with applications ranging from navigation and transportation to social media and emergency response. The dominant technology used for most of these services is the Global Positioning System (GPS), which, while effective, presents several significant limitations. GPS relies heavily on satellite signals, which can be obstructed by environmental factors such as tall buildings or inclement weather, leading to degraded accuracy, signal loss, or delays. Moreover, GPS systems consume a lot of energy, which presents challenges for devices that require prolonged usage, such as smartphones or IoT (Internet of Things) devices.

These challenges highlight the need for alternative localization technologies that can provide more reliable, accurate, and energy-efficient solutions. One promising alternative is the use of cellular networks, which leverage cell phone towers to estimate the location of a device. Unlike GPS, cell-based location systems can function in various environments, including indoors or in urban areas, where GPS typically struggles. Furthermore, cellular systems offer lower energy consumption, making them more suitable for applications requiring continuous location tracking over extended periods.

The proposed system aims to offer a new approach to location determination by relying on cellular network data and connected devices, without using GPS. This method could provide solutions better suited to localization needs in various environments, such as urban areas or indoor spaces. The goal is to create a reliable system that is capable of functioning effectively in different situations, particularly where existing technologies like GPS face challenges. This system could thus open new opportunities in many fields, including navigation, transportation management, and emergency services.

Problematic

The limitations of the Global Positioning System (GPS) in urban and challenging environments underscore the necessity for alternative localization solutions. While GPS has revolutionized location-based services (LBS), its susceptibility to signal obstruction and

high energy consumption presents significant challenges, particularly for mobile devices requiring prolonged use. This raises a critical question:

What alternative methods can be developed to enhance location accuracy and reliability in situations where GPS falls short?

This inquiry leads us to explore the potential of a proprietary system based on cellular networks and localized devices. By utilizing the signals from existing cell towers, can we create a solution that provides accurate and efficient location tracking without the limitations of GPS? Addressing these questions will be vital for advancing location-based technologies and ensuring dependable services in diverse environments.

Overall Objective

The overall objective of this research is to create a reliable location determination system that uses cellular networks. This system aims to provide accurate and efficient location tracking as an alternative to GPS, making location-based services more effective in various applications.

Thesis Organization

To achieve our stated objectives, we have organized our thesis as follows:

- **Chapter 1: State of the Art about Location Services**

  In this chapter, we present the fundamental concepts related to location services. We begin by discussing localization systems technologies and localization techniques. Specifically, we detail the workings of cellular towers-based indoor and outdoor localization systems. Additionally, we conduct a comparative analysis of these systems, highlighting their advantages and disadvantages.

- **Chapter 2: Machine Learning and Deep Learning**

  This chapter presents the principles of artificial intelligence, machine learning, and deep learning, along with the algorithms used in our work. We explain the functioning of these algorithms and their relevance to the development of our system.

- **Chapter 3: Proposed Proprietary Location System**

  Here, we present the architecture of our proposed proprietary location system. We outline the design choices made during development and provide detailed descriptions of the components involved. This chapter aims to illustrate how our system integrates the previously discussed technologies and techniques to achieve effective localization.

- **Chapter 4: Implementation and Results**

  In this chapter, we present the hardware and software resources used for the implementation of our proposed system, as well as the results obtained.

- **Conclusion**

  In the conclusion, we summarize the entire process and share our reflections on the results obtained, as well as the perspectives related to the continuation and improvement of our work.

Chapter 1

State of the art about location Services

1.1 Introduction

Location services play a crucial role in various fields such as navigation, shipment tracking, and emergency response. However, GPS, despite its widespread use, suffers from accuracy and reliability issues in urban environments or under adverse weather conditions. This chapter explores localization technologies and techniques, focusing on cellular network-based systems for precise indoor and outdoor localization without relying on GPS. We examined several cellular localization systems, detailing their architecture and operation. Finally, we conducted a thorough comparison of these systems to evaluate their performance and determine the most effective solutions for different contexts.

1.2 Localization systems technologies:

Localization system technologies are essential for accurately determining the position of objects and individuals in various environments. These systems encompass a wide range of technologies, including satellite-based navigation, inertial measurement units, magnetic field navigation, and sound-based technologies. Additionally, other innovative technologies play a significant role in localization. This section delves into the key technologies used in localization systems, examining their principles, strengths, and applications to provide a comprehensive understanding of how these systems operate.

1.2.1 Satellite Based Navigation:

The global positioning system (GPS) is the most popular system for outdoor localization. However, it requires a line-of-sight (LOS) between the satellites and the handset, making it inefficient for indoor location-based services due to building external walls. GPS

can be utilized by employing a steerable, high-gain directional antenna as the front-end of a GPS receiver. In areas where GPS signals cannot reach, pseudolites (pseudo satellites) are used as independent localization systems. These systems consist of pseudolites, transmission and receiver antennas, target receivers, and reference points. The central concept is to receive the GPS signal and retransmit it through indoor transmitters [1].

1.2.2 Inertial Navigation System:

The Inertial Navigation System (INS) uses inertial measuring units (IMU), like accelerometers and gyroscopes, to determine an object's position and movement with accuracy and energy efficiency. However, INS can be prone to errors, requiring sophisticated filtering methods like the Kalman filter, and has high deployment costs. To improve localization, hybrid solutions have been proposed. One method combines WiFi routers and iBeacons, as seen in the iBILL system, which integrates iBeacon localization and particle filter localization (PFL) to reduce errors and computational load. iBILL outperforms other systems like Magicol and dead reckoning (DR). Other hybrid approaches include combining inertial sensor-based dead reckoning with acoustic localization, fused with a Kalman filter, and Wi-Fi fingerprinting with inertial sensors. These hybrid methods demonstrate better performance than individual techniques [1].

1.2.3 Magnetic Based Navigation:

Magnetic-based technology is used for low-frequency localization by employing at least three reference magnetic stations that emit magnetic fields, which are received by a sensor and used to estimate location through trilateration. This method is accurate but sensitive to conductive and ferromagnetic materials. Typically, these systems rely on disturbances in the Earth's magnetic field caused by indoor ferromagnetic structures. By recording the magnetic field at known locations, magnetic maps are created to infer unknown locations, a technique known as magnetic fingerprinting. However, magnetic interference and variability in measurements from different devices can lead to localization errors. Hybrid techniques, combining magnetic sensors with inertial sensors or integrating cameras and magnetic fields with neural networks, have shown improved accuracy, achieving precisions of 1-2.8 meters and over 91% accuracy at 1.34 meters [1].

1.2.4 Sound Based Technologies:

Sound waves travel at a lower speed than electromagnetic waves, which simplifies time synchronization, a crucial aspect. Humans can hear sounds in a frequency range of 20 Hz to 20 kHz. Sound-based localization is divided into systems using ultrasonic waves and acoustic navigation systems [1].

1.2.4.1 Ultrasonic Based Navigation Systems:

The acoustic method is considered one of the earliest localization approaches, dating back to the Stone Age. Today, ultrasound is used to enhance localization with mobile devices. Ultrasonic localization is effective at short ranges due to its low power, minimal penetration losses through walls, affordable components, and compatibility with handheld devices. However, it faces challenges such as localization errors caused by surface reflections and synchronization issues between nodes. Ultrasonic systems also require complex signal processing algorithms. Despite this, they handle low time synchronization better than Ultra-Wideband (UWB) systems. A proposed method using flight time with a single transceiver achieved a median localization error of 15 cm, although the positions of ultrasonic nodes must be pre-determined for accuracy [1].

1.2.4.2 Acoustic Based Navigation Systems:

The technology uses built-in microphones in smartphones to capture sound from a source and identify its location relative to a reference. This is cost-effective as all smartphones have microphones and places like malls, airports, and hospitals have speakers and microphones. The transmitted modulated acoustic signal contains information such as a timestamp, which is used to estimate the time of flight (TOF) and determine the target's location using multilateration. To avoid disturbing people, the transmitted power should be low, requiring sophisticated signal processing to detect the low-power signal [1].

1.2.5 Optical Based Navigation:

Although optical signals are part of the electromagnetic spectrum, the techniques and challenges associated with them are distinct enough to warrant special attention. Two technologies are presented: Infrared technology and Visible light technology [1].

1.2.5.1 Infrared Technology:

Infrared (IR) systems are used for Line of Sight (LOS) localization with sensors like photodiodes, known for their simplicity and resistance to RF interference. However, they are susceptible to light interference and are expensive. These systems consist of IR emitters and sensors, identifying a target's location via a unique IR signal. The Active Badge is a commercial example. PIR sensors can detect heat changes emitted by living beings with high accuracy but are sensitive to environmental changes [1].

1.2.5.2 Visible Light Communications:

Visible Light Communications (VLC) are effective in places sensitive to radio frequencies, like hospitals, and offer higher accuracy than Wi-Fi. VLC uses LEDs, which

are durable, resistant to humidity, energy-efficient, and cost-effective. LEDs can quickly modulate light signals, transmitting coded light that sensors detect and compare with a database to determine location. VLC localization can use photodiodes or image sensors, with image sensors being more suitable for mobile devices. For precise localization, LEDs and sensors must be in line of sight. Techniques like Angle of Arrival (AOA) and LIDAR with inertial sensors further improve accuracy by providing detailed location information and detecting obstacles [1].

1.2.6 Radio Frequency Based Navigation:

Radio Frequency (RF) based systems are the most commonly used for localization as they cover a wide area with low-cost hardware and can penetrate materials like walls and human bodies, providing better results than infrared or ultrasonic systems. However, they should be avoided in hospitals and planes to prevent interference. Wireless technologies for indoor localization are categorized by the radio frequency used, affecting coverage, wall penetration, and obstacle resistance. Wireless Sensor Networks (WSN) are essential for applications such as indoor fire control, smart homes, and rescue missions, and localization within a WSN is crucial for measurement relevance. Localization systems can be centralized or distributed. RF-based navigation systems include WiFi, Bluetooth, Zigbee, Ultra-Wideband (UWB), and Radio Frequency Identification (RFID) [1].

1.2.6.1 Frequency Modulation Technology:

Frequency Modulation (FM) broadcasting, operating at 88–108 MHz, can be used for localization in both outdoor and indoor environments. FM signals, due to their lower frequency, are less affected by weather and terrain, penetrate walls easily, and consume less power compared to cellular networks and Wi-Fi. For indoor localization, Received Signal Strength (RSS) is used with the fingerprinting technique. Among various machine learning methods, k-nearest neighbor (kNN) showed the best performance. FM is most effective in small spaces like rooms, while Wi-Fi performs better in larger areas like floors [1].

1.2.6.2 Cellular Based Technology:

Cellular networks, operating on various frequency bands (0.9 GHz, 1.8 GHz, 2.8 GHz), offer better coverage than Wi-Fi and require no additional infrastructure. Localization methods include proximity within cell coverage, RSS fingerprinting, and TOA trilateration. RSS fingerprinting in cellular networks can achieve accuracy within 2.5–5.4 meters. Studies using GSM and UMTS cells for fingerprinting showed errors less than 5 meters, comparable to WLAN accuracy. LTE-based localization achieved errors below 8 meters in 50% of cases, and combining LTE with inertial measurement units reduced RMSE

to 3.52 meters. Synthetic aperture navigation (SAN) improved LTE accuracy, reducing RMSE to 3.9 meters. Combining LTE and WLAN fingerprinting significantly enhances performance compared to LTE-only methods. Cellular systems can also support other RF localization systems like RFID and Wi-Fi [1].

1.2.6.3 WiFi Technology:

Wi-Fi is a widely used wireless networking technology operating in the 2.5 GHz and 5 GHz RF bands. It is commonly found in large indoor environments like universities and office buildings, providing extensive coverage through hotspots. Wi-Fi-enabled devices include personal computers, smartphones, and tablets. The costs for Wi-Fi infrastructure and devices are relatively low, with a reception range now reaching up to 1 km. Wi-Fi localization uses RSS fingerprinting and can be combined with other RF techniques like RFID. It covers larger areas and offers higher throughput compared to Bluetooth, making it practical for many applications. Examples of commercial Wi-Fi-based positioning systems include RADAR, HORUS, COMPASS, HERECAST, and PlaceLab [1].

1.2.6.4 ZigBee:

ZigBee, based on the IEEE 802.15.4 standard, operates in different frequency bands globally and is used for long-distance transmission in wireless mesh networks. It is cost-effective, with a low data transfer rate and short latency compared to WiFi. ZigBee uses the RSS method to estimate distances between sensor devices. To reduce power consumption from WiFi Access Point (AP) scanning, an energy-efficient indoor localization system called ZIL was developed, using ZigBee to collect WiFi signals. Additionally, a ZigBee localization algorithm based on proximity learning was introduced, offering reduced computational time while maintaining accurate positioning compared to traditional triangulation methods [1].

1.2.6.5 Bluetooth:

Bluetooth (IEEE 802.15.1) enables short-range wireless communication using radio waves between 2.402 GHz and 2.480 GHz, similar to Wi-Fi. It is cost-effective, has low transmission power, long battery life, secure and efficient communications. Bluetooth Low Energy (BLE) extends the range to 70-100 meters and provides 24 Mbps with higher power efficiency, but is not suitable for large-area localization. BLE-based localization is used in smartphones through iBeacons (Apple) and Eddystone (Google) for precise indoor localization in places like airports, train stations, markets, malls, and restaurants by using area maps sent to the smartphone. Neural networks trained with RSS values and coordinates can detect user location based on online RSS measurements [1].

1.2.6.6 Ultra Wide Band:

According to the U.S. FCC, UWB signals have a bandwidth greater than 500 MHz and carrier frequency over 2.5 GHz. UWB offers large bandwidth, high-speed communication, high time resolution, and strong resistance to multipath interference. Its ability to pass through obstacles and immunity to interference make it ideal for indoor positioning. UWB achieves centimeter-level accuracy using TOA and TDOA algorithms. A hybrid localization system combining Wi-Fi and UWB can reduce costs and achieve a localization error of 20 cm. UWB can also localize an unlimited number of tags, with fingerprint estimation showing the best performance for position estimation [1]. The figure 1.1 presents the applications of UWB technology across various sectors, such as security, smart homes, health, and smart cities, highlighting its benefits: precise location, indoor navigation, safety, and process efficiency.

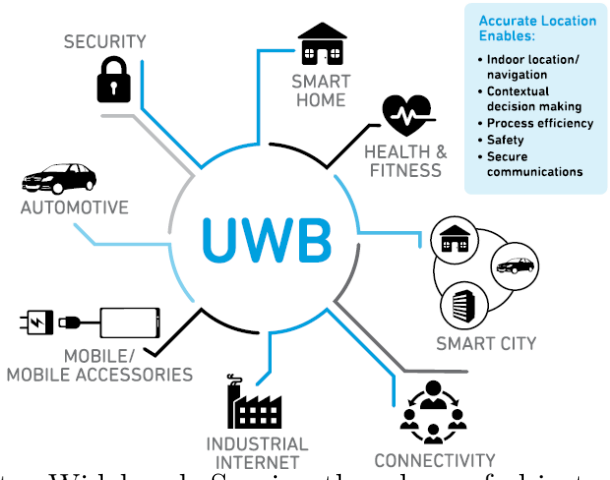


Figure 1.1: Ultra-Wideband: Sensing the where of objects and people [2].

1.2.6.7 Radio Frequency Identifcation:

RFID systems use tags and readers for signal processing, with tags being active, passive, or semi-active. Active RFID, with a detection range up to 100 m, is ideal for long-range tracking but not sub-meter accuracy. Passive RFID, cheaper and smaller, is used for sub-meter detection within 10 m. RFID is popular due to its low cost and deploys reference tags to estimate positions based on RSSI. Studies show RFID offers better accuracy and stability compared to other sensors [1].The figure 1.2 illustrates a basic radio frequency identification (RFID) system, detailing its components and functionality, including the reader, tags, and the communication process between them.

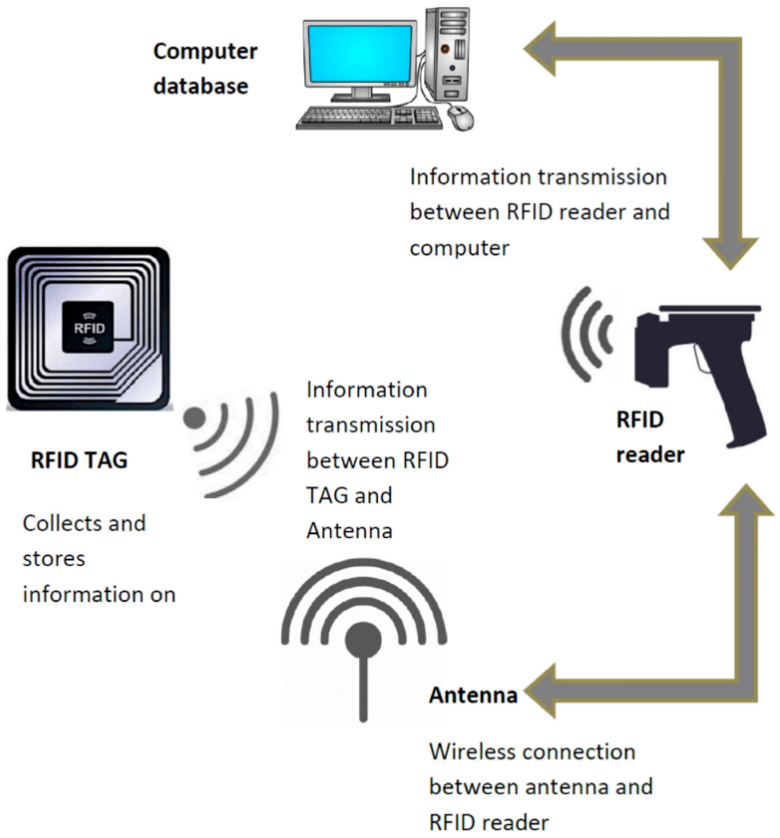


Figure 1.2: Basic radio frequency identification (RFID) system [3].

In conclusion, localization system technologies are essential for precise positioning in diverse environments, each offering unique advantages and evolving to enhance accuracy and reliability. Understanding these systems is crucial for effective navigation solutions.

1.3 Localization techniques:

Localization techniques are available that provide accurate localization for movable or fixed objects in both outdoor and indoor environments. These techniques often use a combination of various technologies to enhance accuracy, each with its own advantages and disadvantages impacting the system. The taxonomy of these localization techniques is shown in figure 1.3. Common methods include RSSI, TOA, and AOA/DOA for wireless signal measurements, while indoor localization also utilizes TDOA, RTT, ADOA, PDOA, POA, CSI, RSRP, and RSRQ [4].

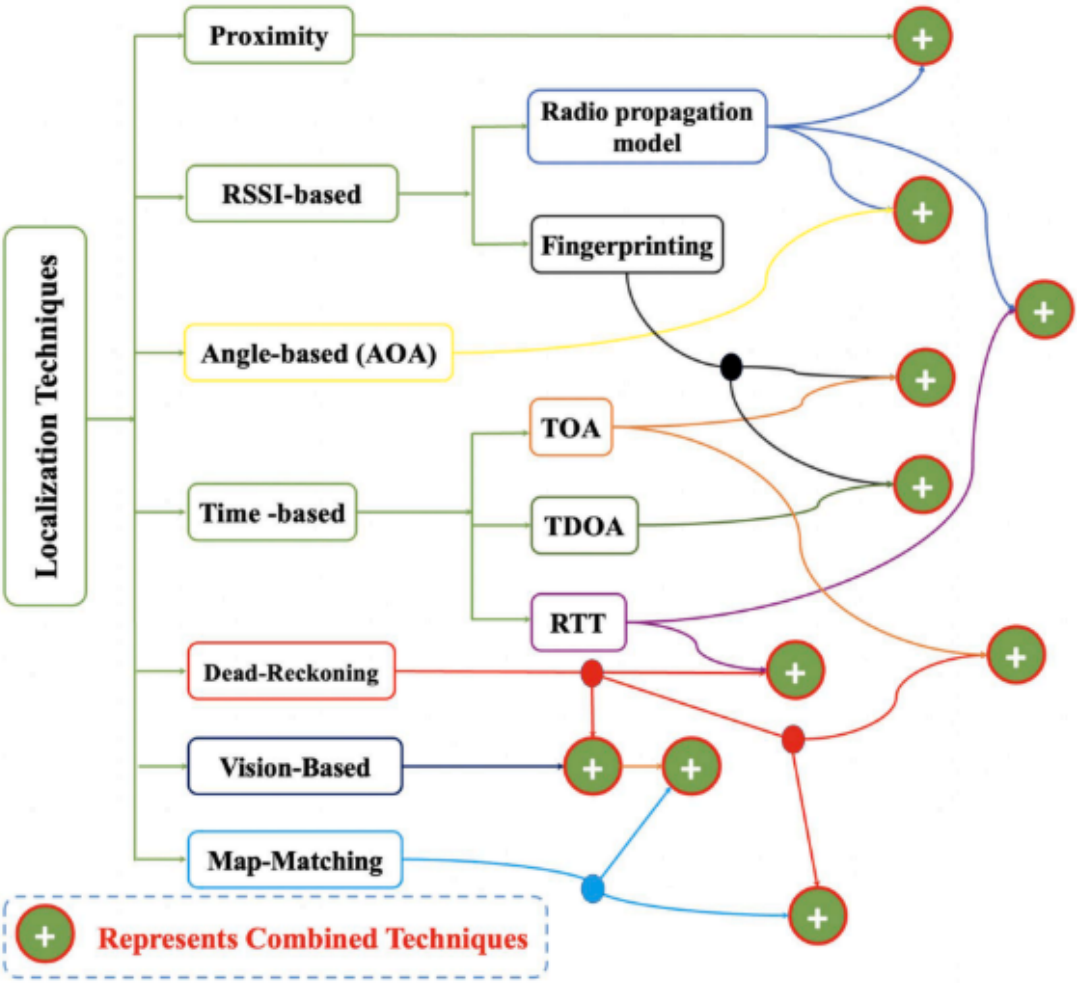


Figure 1.3: Taxonomy of Localization Techniques [4].

1.3.1 Proximity detection techniques

Proximity detection techniques locate devices by evaluating their position relative to a predefined area, requiring detectors at fixed locations. When a detector identifies a target device, it is considered to be within the detector's proximity zone. Figure 1.4 illustrates this method: in image a), devices A1 and A2 are within the detector's coverage area, while A3 is not. Image b) shows the coverage area divided into three sectors for more precise localization. This technique, commonly used in GSM systems, has an average accuracy of 76 to 216 meters but can be limited by radio coverage [4].

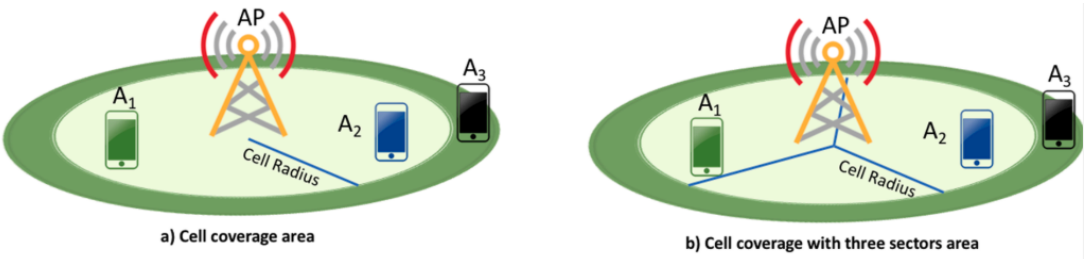


Figure 1.4: Proximity Localization Technique Measurement Configuration [4].

1.3.2 Received signal strength indicator-based techniques

Two approaches are used to determine the position of a mobile target based on RSSI measurements, including Fingerprinting-based and RSS-propagation models [4].

1.3.3 Fingerprinting technique

Fingerprinting is a technique that uses scene features from videos, virtual images, or electromagnetic signals to estimate target device position. There are two stages to it Like the figure 1.5 :online testing and offline training RSSI. Fingerprint-based location estimation algorithms are popular due to their efficiency and low cost. However, the distance ratio is not always linear, especially in indoor spaces. In order to anticipate real-time interior locations, machine learning algorithms must overcome a number of obstacles, including computational power, storage capacity, complicated environments, and time-consuming matching procedures [4].

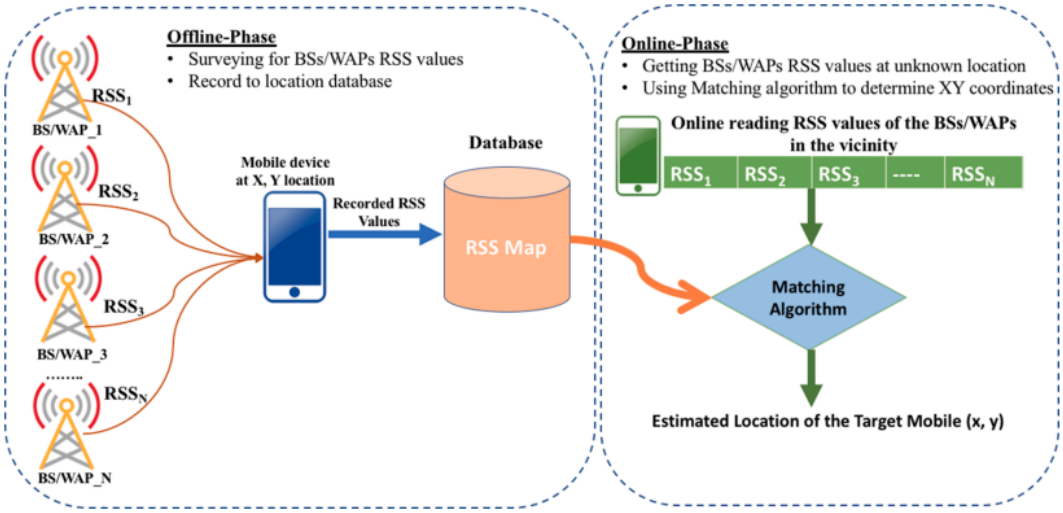


Figure 1.5: Fingerprinting technique [4].

1.3.4 RSSI Radio Propagation Technique

The RSSI-based localization system measures the signal strength between a target mobile device and multiple BS/WAP to calculate the distance and determine the position using trilateration. Although inexpensive and easy to implement, it has limited accuracy (2 to 4 meters) due to RSSI fluctuations. Figure 1.6 illustrates trilateration using RSSI distances between the target device and three BS/WAP. The distance d_i between the target device and the BS/WAP can be determined by the equation [4]:

$$d_i = d_0 \times 10^{\left(\frac{RSSi_0 - RSSi}{10 \times \eta_i}\right)} \tag{1.1}$$

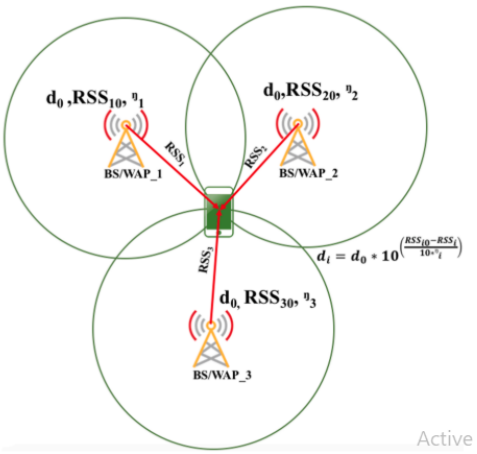


Figure 1.6: RSSI Radio Propagation Technique [4].

1.3.5 Angle based method (angulation)

The angulation technique, using Angle of Arrival (AOA), calculates a target device's position by measuring the angles from multiple fixed stations, as shown in figure 1.7. This method requires at least two fixed stations for 2D positioning, with higher accuracy achieved by using more stations. Despite its efficiency, AOA faces challenges such as high implementation costs, multipath issues, and signal reflection problems in indoor environments [4].

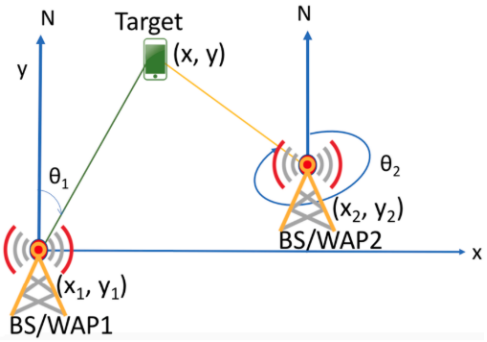


Figure 1.7: Angulation Location Measurement Configuration [4].

1.3.6 Time-based technique

Trilateration uses the known distances between a target device and multiple fixed stations (three in 2D or four in 3D) to calculate the device's position. This technique finds the intersection of circles centered on the fixed stations with radii equal to the measured distances. Time-based techniques, such as TOA and TDOA, measure the signal propagation time to estimate these distances, requiring at least three stations for accurate localization [4].

1.3.7 Vision-based technique

Vision-based localization uses cameras to extract scene features and estimate the position of a target device. It employs image processing methods and deep learning algorithms to enhance accuracy. These systems provide real-time assistance, particularly for visually impaired individuals, by detecting and navigating around dynamic obstacles [4].

1.3.8 Dead reckoning technique

The dead-reckoning (DR) technique estimates the current location of a movable item using a previously calculated location by integrating estimated speed, heading direction, and course over time. It employs sensors such as gyroscopes, accelerometers, speed pulse sensors, and magnetometers to provide high-accuracy navigation even when GPS/GNSS is unreliable. However, it has limitations including the need for a reference position and cumulative errors [4].

1.3.9 Map-matching technique

Map Matching (MM) connects the estimated position of a mobile device with geographical information from a digital map, improving navigation accuracy by using machine learning algorithms. Simultaneous Localization and Mapping (SLAM) enables intelligent robots to map their environment and predict their position using sensors like LiDAR and cameras. Digital maps can correct errors from inertial sensors, though they require detailed databases and may have their own inaccuracies. This method is crucial for applications such as autonomous robot navigation and drone delivery, providing real-time assistance and improving safety [4].

In conclusion, localization techniques are crucial for accurately determining positions in both indoor and outdoor environments. In the next section, we will describe the real-world systems invented to achieve precise localization. These systems will be discussed in detail to understand their methodologies and applications.

1.4 Cellular-towers-based indoor localisation systems:

Indoor localization systems based on cellular towers meet a growing need for accurate and ubiquitous localization services. GPS, while effective outdoors, has limitations indoors, and not all phones have Wi-Fi capabilities, especially in developing countries. These systems rely on RSS (Received Signal Strength) measurements from multiple cellular towers and deep learning algorithms to accurately estimate device positions. Since

all phones support cellular technology, this approach can provide a universal localization service compatible with virtually any cellphone, without additional energy consumption beyond standard phone operation.

In the following, we will present several indoor localization systems based on cellular towers, each with its own architecture and methodology to provide a comprehensive understanding of their capabilities and applications.

1.4.1 MonoDCell:

Figure 1.8 illustrates MonoDCell's system architecture, which operates in two stages: offline training and online tracking. In the offline stage, the Fingerprint Collector gathers cell tower IDs and RSS at sparse seed points using a phone application. The Spatial Interpolator generates synthetic measurements to extend these points, reducing fingerprinting overhead and aiding model generalization. The Trace Generator creates traces to simulate user movement, while the History Builder constructs a measurement history in temporal sequences. The Model Constructor builds and trains a deep model for localization. In the online stage, real-time tracking captures cell tower information, augmented by the History Builder. The Location Estimator then uses the deep model to determine the user's location.

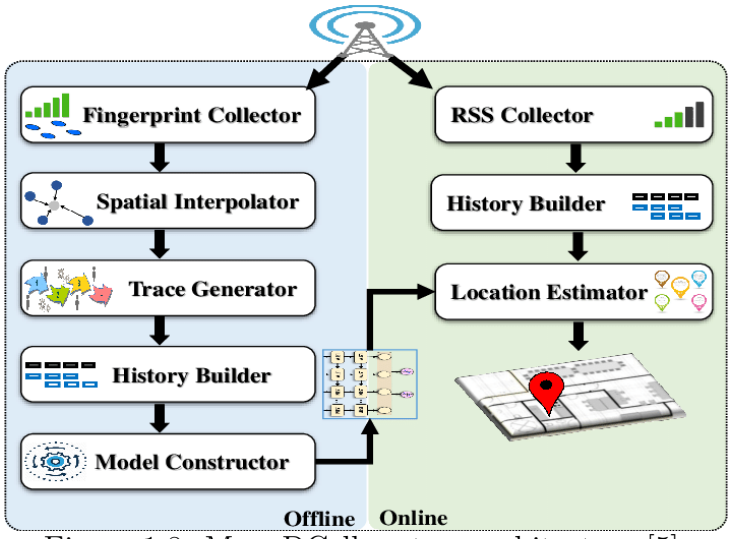


Figure 1.8: MonoDCell system architecture [5].

We present the details of the different modules of MonoDCell:

- **Fingerprint Collector:** During the offline phase, MonoDCell uses a Fingerprint Collector application to gather cellular data at sparse seed points, recording cell tower ID (CID) and received signal strength (RSS) at each point. This reduces data collection efforts while balancing accuracy and overhead.
- **Spatial Interpolator :** To improve localization accuracy without increasing data collection efforts, MonoDCell uses a Spatial Interpolator to generate synthetic cell

measurements. This module employs a KNN-Regressor, which predicts RSS measurements by calculating the inverse distance-weighted mean of the RSS values from the nearest seed points, giving more weight to closer points.

- **Trace Generator:** The Trace Generator module creates spatial traces from discrete fingerprint points provided by the Spatial Interpolator to train the deep network, avoiding the high overhead of continuous motion data collection. It uses a first-order Markov model to simulate human movement, selecting from eight adjacent points or staying stationary based on equal probabilities. The transition matrix reflects the environment's floorplan. Traces start from a uniformly distributed initial point and subsequent points are chosen based on transition probabilities until the trace is complete. RSS measurements from these traces are sent to the History Builder module, serving as a data augmentation process for neural network training.

- **History Builder:** The History Builder module prepares sequence data for the recurrent neural network during both offline and online phases. It segments traces into overlapping fixed-length sequences. Each sequence consists of preceding observations and the current one, transforming observations into one-hot encoded feature vectors for cell towers. RSS values are rescaled to [0, 1] for LSTM sensitivity. In the offline phase, sequences and user locations train the LSTM in the Model Constructor. In the online phase, sequences are used by the Location Estimator to determine the user's location.

- **Model Constructor:** The Model Constructor uses a recurrent neural network with stacked LSTM layers for effective handling of long and short-range dependencies in historical RSS data from cell towers. The input layer processes fixed-length sequences of RSS vectors, mapping them to 2D spatial coordinates as output. To prevent overfitting, the model employs dropout regularization and early stopping during training, using the Adam optimizer and mean square error as the loss function.

- **Location Estimator:** During the online phase, a user at an unknown location receives signals from the associated cell tower. The History Builder constructs a sequence with this current data and past observations, which the deep model uses to estimate the user's current location.

- **Evaluation:** MonoDCell was tested on various Android devices in two testbeds, achieving median location errors of 0.95m and 1.42m. It outperforms state-of-the-art systems by at least 202%, demonstrating its accuracy and potential as a superior localization system [5].

1.4.2 OmniCells :

OmniCells is a deep learning system for indoor localization uses cellular measurements from training devices to ensure consistent performance across unknown tracking phones .The OmniCells architecture operates in two stages as the figure 1.9: offline training and online tracking. In the offline stage, the Fingerprint Collector records cell tower data, which the Pre-processor converts into RSS vectors. The Feature Extractor then generates device-independent features and trains an encoder model. The Localization Model Constructor uses these features to train a localization model. During the online phase, the user's cell tower data is processed by the Pre-processor and Feature Extractor before the Location Predictor estimates the user's location based on the trained models.

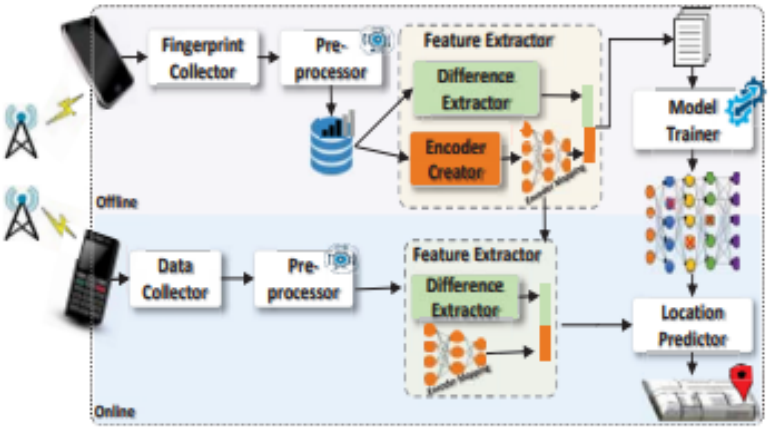


Figure 1.9: OmniCells system architecture [6].

We presents the details of the different modules of OmniCells including the Pre-processor,the Features Extractor and Localization Model Constructor modules as well as the processing done in online phase.

- **Pre-processor Module:** The Pre-processor module operates during both calibration and tracking phases, producing a vector of RSS values from up to seven detectable cell towers, assigning non-detected towers an RSS of 0. It normalizes RSS values between [0,1] and filters out weak signals below a threshold to ensure stable localization. To reduce data collection overhead and prevent overfitting, OmniCells employs a data augmentation framework that generates synthetic data reflecting typical RSS variations from short-term samples.
- **Features Extractor Module:** The Feature Extractor module in OmniCells consists of the Difference Extractor and Encoder Creator sub-modules.
  - *Difference Extractor:* The Difference Extractor calculates RSS differences between cell tower pairs in each scan to mitigate device heterogeneity, eliminating fixed offset values caused by varying receiver sensitivities and gains. This method uses relative RSS values to ensure consistency across different devices.

- *Encoder Creator:* The Encoder Creator module in OmniCells encodes pre-processed RSS vectors for consistent signal representation across different phones using an autoencoder neural network. The autoencoder has an encoder that maps RSS input to a lower-dimensional latent space and a decoder that reconstructs the input from this latent representation. Instead of reconstructing the same phone's data, OmniCells trains the autoencoder to map data from one phone to data from a different phone at the same location. This process creates a phone-invariant latent representation, enabling device-transparent localization.

- **Localization Model Constructor Module:** The Localization Model Constructor trains a deep localization model using aggregated features to estimate user location. The model is a deep fully-connected neural network with hidden layers using the tanh activation function for non-linearity and strong gradients. The input layer combines latent features and inter-difference values of cellular signal strengths from cell towers. The output layer, with neurons corresponding to fingerprint points, uses a softmax activation function to produce a probability distribution over reference locations, operating as a multinomial classifier.

- **Online Phase:** During the Online Phase, OmniCells locates the user in real-time by processing cell signals from nearby towers to extract a feature vector. This vector is fed into the trained localization model to estimate the user's location as one of the predefined fingerprint points. To ensure continuous tracking and improve user experience, OmniCells calculates the center of mass of all reference points using a spatial weighted average, with weights corresponding to the likelihoods from the classifier network.

- **Evaluation:** Evaluation of OmniCells in two realistic testbeds using different Android phones with different form factors and cellular radio hardware shows that OmniCells can achieve a consistent median localization accuracy when tested on different phones. This is better than the state-of-the-art indoor cellular-based systems by at least 101% [6].

1.4.3 Cellindeep:

Figure 1.10 illustrates CellinDeep's system architecture, which operates in two stages: offline training and online tracking. In the offline stage, the Data Collector gathers timestamped cell tower signal measurements at different reference points using a phone application. The Pre-processor handles noise in the input data and extracts and normalizes the required features. The Data Augmenter generates synthetic training samples to reduce training overhead and avoid over-fitting. The Model Creator builds and trains a

deep neural network (DNN) for localization, employing dropout regularization and early stopping to prevent over-fitting. In the online stage, real-time tracking captures cell tower information, filtered by the Pre-processor. The Online Predictor then uses the deep model to estimate the user's location, and the Fine Localizer refines the estimation to reduce outliers and provide a smoother location.

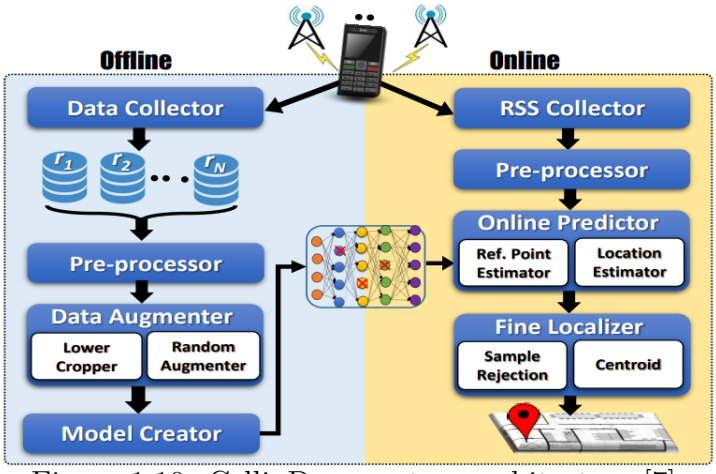


Figure 1.10: CellinDeep system architecture [7].

We present the details of the different modules of CellinDeep:

- **Data Collector module:** running on a cell phone, stores the timestamped cell towers signal measurements at different selected reference points within the area of interest. These collected training measurements are then sent to a CellinDeep running service in the cloud for further processing.
- **Pre-processor module:** is used to handle the noise in the input data and to extract and normalise the required features.
- **Data Augmenter module:** is employed to augment the collected training data by generating synthetic training samples. This not only reduces the training overhead but also mitigates the noise effect and avoids over-fitting . It has two novel data augmenters: Random Augmenter and Lower-bound Cropper.
  - *Random Augmenter technique:* uses the 6% probability of hearing the maximum number of cell towers to increase training data size. The idea is to randomly generate a binary mask that can be multiplied by the RSS vector to selectively drop certain cell towers in a scan (Figure 1.11), ensuring the device is always present to capture reality better.

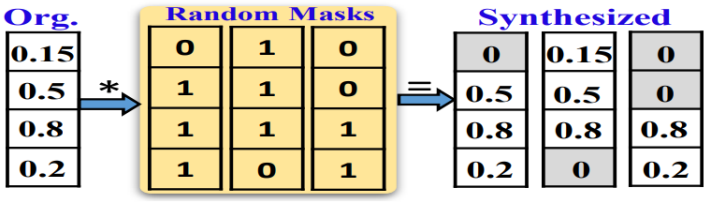


Figure 1.11: Example of the Random Augmenter operation on normalized RSS values [7].

- *L*ower-bound Cropper augmentation technique: uses weak received signals from cell towers to increase training data size. It removes entries with a value less than a certain threshold, mimicking that the cell tower has not been heard in the generated synthetic sample (show Figure 1.12).

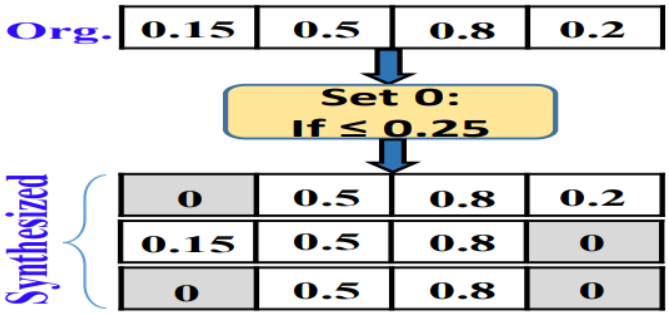


Figure 1.12: Example of the Lower Cropper on normalized RSS values [7].

- **Model Creator module:** The Model Creator module constructs and trains a deep neural network (DNN) to represent input data hierarchically, handling correlated and non-linear cellular signals for user locations. It uses a fully connected structure with four hidden layers and ReLUs to avoid the vanishing gradient problem. The output layer corresponds to reference points in the area of interest. To prevent overfitting, CellinDeep employs dropout regularization and early stopping. The trained model is then saved for use by the Online Predictor module.

- **Online phase:** users are tracked in real time by sending RSS information from cellular towers at the current unknown location to the CellinDeep server. This data is first filtered by the pre-processor module. The online predictor module then inputs the data into the deep model built during the offline phase to estimate the likelihood of the user being at various trained reference points. The system combines these likelihoods to determine the user's location in continuous space. Finally, the Fine Localizer module further refines the location estimate by reducing outliers for a smoother result..

- **Evalution:** The implementation of CellinDeep on various Android phones shows a median localization accuracy of 0.78 meters, outperforming current indoor cellular-

based systems by at least 350%. Additionally, CellinDeep saves over 93.45% in power compared to WiFi-based techniques [7].

1.5 System outdoor:

Outdoor localization systems based on cellular networks utilize mobile telephony infrastructure to offer a precise and reliable alternative. A major issue with GPS in outdoor environments is its high energy consumption, which quickly depletes the battery life of mobile devices. By using cellular signals to determine positions, these systems provide extensive coverage and better accuracy while being more energy-efficient. Below, we will describe these outdoor localization systems based on cellular networks, detailing their architectures and specific characteristics.

1.5.1 Crescendo:

Crescendo basic idea: As the figure 1.13 shows, the cellular network consists of multiple towers, each covering an area divided into sectors. Each sector, a slice of the circle around the tower, extends finitely due to signal attenuation. Multiple cells operating at different frequencies cover each sector. A mobile unit (MU) can detect up to seven cells at once but is associated with only one. These cells can be from the same or different towers and sectors. Information about the RSS of visible cells is available both to the provider and the device.

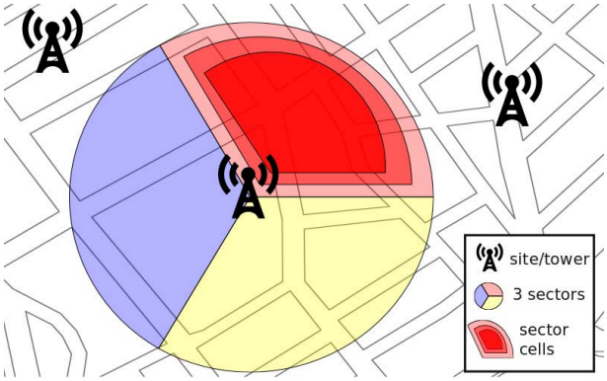


Figure 1.13: Basic cellular network architecture [8].

- Figure 1.14 shows the system architecture. The system works in two phases: an offline phase and an online tracking phase.

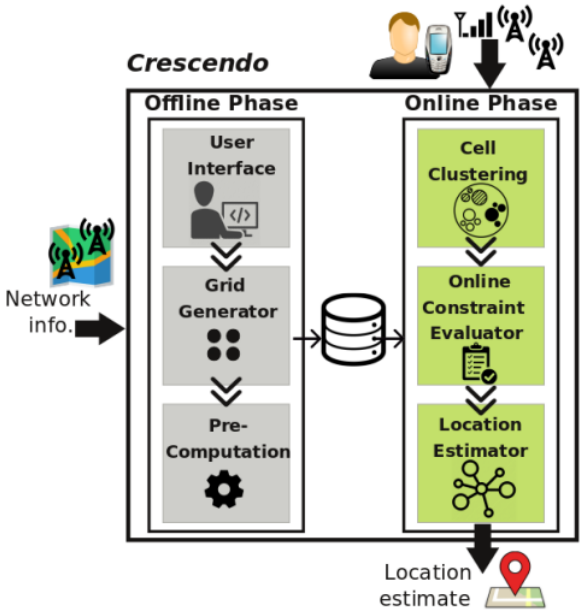


Figure 1.14: Crescendo system architecture [8].

- *Offline phase:* During the offline phase, the system administrator uses the User Interface module to import or enter necessary network information, such as site IDs and locations, and cell and sector information. A virtual grid is then generated and superimposed over the area of interest using the Grid Generator module to expedite calculations and manage RF propagation noise. The Pre-Computation module calculates the "discrete" Voronoi diagram of the area to determine the initial user ambiguity area and pre-calculates associated parameters for each grid point, like Pairwise Site Constraints and Containing Cell Set. This pre-calculated information is utilized during the online phase by the Online Constraint Evaluator module to reduce running time and manage noise.
- *Online Tracking Phase:* During the Online Tracking Phase, Crescendo generates location estimates based on visible network cells. The Cell Clustering module clusters visible cells by tower locations, using the strongest cell from each tower as a representative. The Online Constraint Evaluator module uses these representative RSS values to create online Pairwise Site Constraints and includes all visible cells in the online Containing Cell Set. It assigns scores to grid points based on their likelihood of being the MU's location. Finally, the Location Estimator module uses these scores to estimate the MU's final location.

- **Evaluation:** Evaluation of Crescendo in both an urban and a rural area using real data shows median accuracies of 152m and 224m, respectively. This is an improvement over classical techniques by at least 18% and 15%, respectively [8].

1.5.2 Active GSM Cell-ID Tracking:

The prototype **SS7Tracker** platform, a GSM Cell-ID-based solution for location-based services. Practical tests in a live GSM network demonstrate the platform's usability and performance limits, showcasing its potential to enhance location-based services and increase mobile operators' revenue. The SS7Tracker platform is proving to be an effective solution for location-based services and signal coverage diagnostics in GSM networks. Thanks to its robust architecture and active monitoring features, it provides valuable information for improving the quality of mobile services, identifying areas with poor coverage, and optimizing network performance. Additionally, its use in research on mobility and human activity highlights its potential for various applications, from urban planning to analyzing tourist behavior. Future research should focus on optimizing tracking intervals, improving positioning accuracy, and reducing the network's energy consumption and signaling load.

- **SS7Tracker platform architecture:** The SS7Tracker platform architecture utilizes active tracking to periodically request and store the location of tracked subscribers. It is based on a client-server architecture consisting of several key components. The Dialogic R SPCI4 SS7 signaling board facilitates communication with the live GSM network. The signaling network ensures the transmission of messages between different network elements. The main modules include :
  - **QueryCellId:** retrieves location information from the GSM network using MAP protocol messages, with response speed influenced by SMS delivery time, and Tracker, which manages tracking tasks based on a job file.
  - **Tracker requests the location of all mobile:** subscriber numbers (MSISDNs) and evaluates the next tracking interval based on the subscriber's last location, movement, and other criteria. The TrackerGUI application, a Java interface for SS7Tracker, allows users to create tasks, define tracking rule chains, summarize and visualize results, and export data to external mapping systems.
  - **Microsoft SQL Server 2000 database:** stores the location data and tracking results. The SS7 message flow sends 'SendRoutingInfo' messages to the Home Location Register (HLR) of the subscriber to update their International Mobile Subscriber Identity (IMSI) and Visitor Location Register (VLR). Signal coverage diagnostics: involve preventing inroamers from disappearing from mobile networks and determining where they switch to rival networks. A seven-hour experiment tracked 247 foreign inroamers in the Czech Republic in May 2008, with the most valuable information obtained just before a subscriber switched to a rival network. The tracking interval, whether 2 minutes or 6

minutes, significantly impacts data interpretation, necessitating precise tuning of tracking rules.

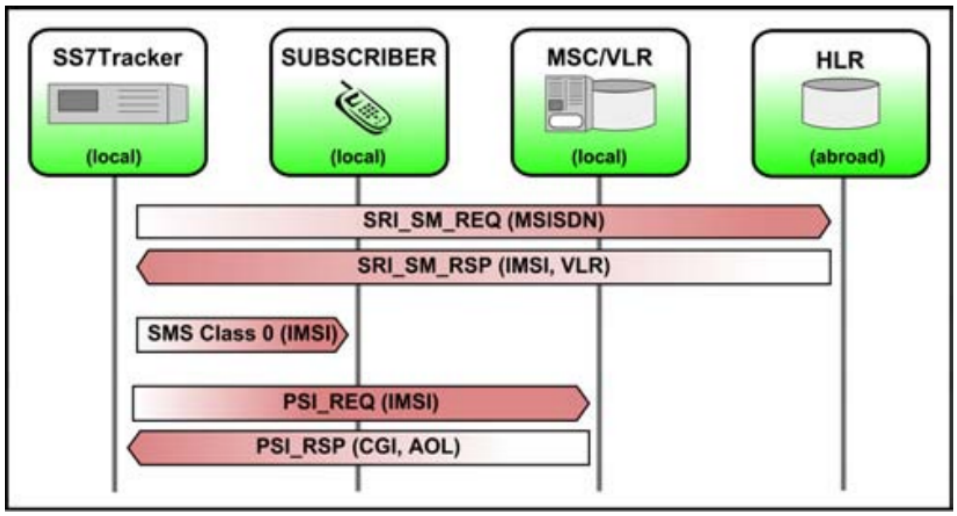


Figure 1.15: SS7 message flow to obtain subscriber location [9].

- **Visualization and analysis:** through third-part, navigation software allow for the visualization of inroamer movements using colored points and sectors to represent cell sites and antenna orientations. This helps identify overloaded or underused network segments. Graphs provide insights into inroamer behavior, network changes, and returns.
- **For research on mobility and human activity :** SS7Tracker offers improved spatial accuracy and continuity of personal location data. Aggregate statistics identify the most visited places and compare visit rates of different tourist attractions, while individual track analyses examine daily routines, action spaces, and mobility behavior. The data utilization enables studying daily mobility between key personal activity nodes like home, work, and leisure places [9].

This section covered several systems that utilize cellular networks. In the following section, we will conduct a comparison to assess their advantages and disadvantages.

1.6 system comparison

In the previous section of this report, we provided detailed descriptions of several cellular-based localization systems. Now, the table 1.1 offers a comparative view of these systems. This comparison helps provide a clearer understanding of which systems may be better suited for different use cases.

System	Methodology	Accuracy	Energy Consumption	Sensitivity to Signal Changes Due to Environment	Advantages	Disadvantages
MonoDCell [5]	Deep LSTM model based on single cell tower RSS.	0.95 m (small environment), 1.42 m (large environment).	No extra energy consumption (uses existing cellular network signals).	Very sensitive to obstacles and environmental changes.	Simple deployment, adapts to varied environments.	Requires good training data.
OmniCells [6]	Deep learning with autoencoders for cross-device localization.	1.67 m (small environment), 2.05 m (large environment)	No additional consumption, uses existing cellular signals.	Low to moderate; handles minor variations but sensitive to major changes.	Stable accuracy across different types of phones.	Requires complex autoencoders for normalization between devices.
CellinDeep [7]	Deep neural network leveraging multiple cell tower signals.	0.78 m median error.	93.45% energy savings compared to WiFi	Moderately sensitive to changes.	High accuracy, low energy consumption, handles noisy data.	Requires large training datasets and model regularization to prevent overfitting.
Crescendo [8]	Outdoor localization using Voronoi diagrams	152 m (urban area), 224 m (rural area)	No additional consumption.	Moderate; influenced by tower configurations and obstacles.	Suitable for large areas, no calibration needed.	Less accurate than WiFi systems, depends on cellular tower density.
Active GSM Cell-ID Tracking [9]	Active network-based tracking using the SS7 protocol.	Depends on cell size (large areas, lower accuracy).	High energy due to continuous queries.	Highly sensitive; static cell-ID without filtering makes it prone to noise.	Real-time tracking, no need for user cooperation.	High network overhead, privacy issues.

Table 1.1: Comparison of Cellular-based Localization Systems.

1.7 Conclusion

In conclusion to this chapter, we explained localization system technologies and localization techniques. We examined in detail different cellular-tower-based localization systems, both indoor and outdoor, highlighting their architectures and characteristics. By making a comparison between these systems, we were able to draw valuable insights into their effectiveness and performance.However, these systems also have limitations. Indoor localization systems, for example, may suffer from reduced accuracy due to signal interference and variable tower density. Similarly, outdoor localization systems can be affected by environmental factors, such as weather conditions, which can compromise the reliability of location data. These challenges underscore the importance of making improvements and innovations to existing localization systems.

In the next chapter, we will cover the basics of machine learning and deep learning, linking them to the location service concepts studied here. This transition will enable us to build our own system by integrating these advanced techniques, thereby enriching our overall understanding.

Chapter 2

Machine learning and Deep learning

2.1 Introduction

Location systems based on cellular networks leverage machine learning and deep learning techniques to accurately predict user positions. These systems analyze vast amounts of data from cellular signals using advanced algorithms to deduce precise locations. In this chapter, we will present the basic concepts of machine learning and deep learning, with a focus on solving regression problems. We will discuss the various algorithms used in this domain and how they contribute to improving the accuracy of location predictions.

2.2 Artificial intelligence:

Artificial intelligence (AI) is a branch of computer science focused on creating systems capable of performing tasks that typically require human intelligence. These tasks include recognizing speech, making decisions, and identifying patterns. On its own or combined with other technologies (e.g., sensors, geolocation, robotics), AI encompasses various technologies, such as machine learning and deep learning, as shown in Figure 2.1, which enable computers to learn from and adapt to new data without human intervention [10].

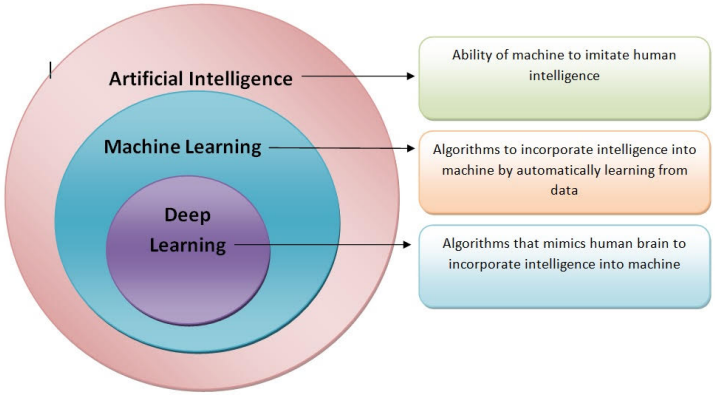


Figure 2.1: The relationship between Artificial intelligence,Machine learning and Deep learning [11].

2.2.1 Machine learning :

Machine learning (ML) is a branch of artificial intelligence that focuses on building algorithms that can learn from and make predictions or decisions based on data. Unlike traditional programming, where specific instructions are coded, ML systems improve their performance by identifying patterns in large datasets. ML is widely used in various fields such as natural language processing, computer vision, and predictive analytics, enabling automation and enhanced decision-making processes [12].

2.2.2 Deep learning :

In 1943, Warren McCulloch and Walter Pitts, two mathematicians and neuroscientists, developed the first models of neural networks. In their paper titled 'A Logical Calculus of the Ideas Immanent in Nervous Activity', they proposed a mathematical representation of the functioning of a biological neuron. Although the McCulloch-Pitts neuron model is limited in its capabilities and lacks a learning mechanism, it laid the foundation for artificial neural networks and deep learning [13] (Show Figure 2.2).

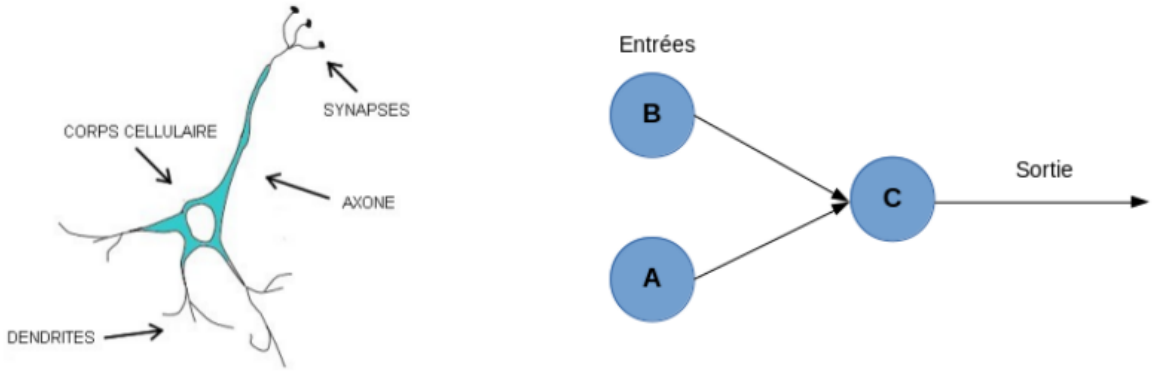


Figure 2.2: On the left, the diagram of a biological neuron, and on the right, the diagram of the formal neuron from 1943 [14].

Deep learning (DL) is a specialized subset of machine learning that utilizes neural networks with many layers (hence the term "deep") to model complex patterns in data. These neural networks, inspired by the structure of the human brain, consist of interconnected layers of nodes that process input data to generate outputs. The layered architecture allows deep learning models to automatically learn hierarchical representations of data, leading to high accuracy and performance in various applications [15].

2.3 Algorihm for regression :

Regression is a statistical approach used to analyze the relationship between a dependent variable (target variable) and one or more independent variables (predictor variables). The objective is to determine the most suitable function that characterizes the connection between these variables [16].

For regression tasks, we employ both machine and deep learning algorithms. The machine learning algorithms include those based on distance methods, while the deep learning models utilize neural networks. These algorithms are essential for making accurate predictions. A schema (Figure 2.3) below will illustrate the algorithms used in our study for both machine learning and deep learning.

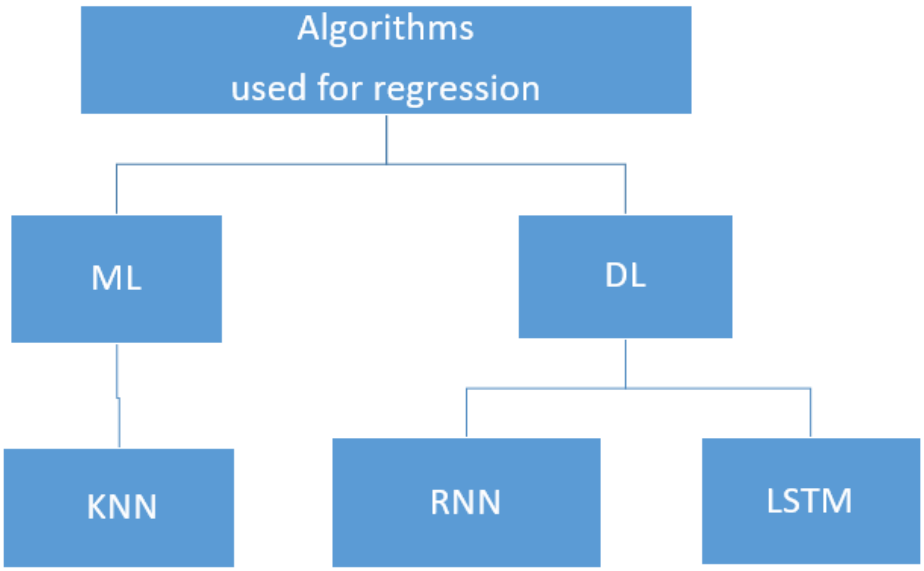


Figure 2.3: Regression algorithms.

2.4 Machine Learning Algorithms:

Machine learning algorithms are techniques that allow computers to learn from and make predictions based on data. These algorithms are used in a variety of applications,

such as classification, regression, clustering, and more. They can identify patterns and relationships within data to provide actionable insights. In our project, we utilized only the K-Nearest Neighbors (KNN) algorithm to address the regression problem we are working on.

2.4.1 K-Nearest Neighbors:

The K-Nearest Neighbors (KNN) algorithm is a non-parametric machine learning technique used for both classification and regression tasks. In KNN Regression, the algorithm predicts the value of a target variable by averaging the values of the k nearest data points. Here's a concise summary of how it works:

1. Data Collection: Start with a dataset containing input features and continuous target values.

2. Choosing K: Select the number of nearest neighbors (k) to use for predictions. A smaller k can lead to noisy results, while a larger k can smooth out predictions.

3. Distance Metric: Use a distance metric, like Euclidean distance, to measure similarity between data points.

4. Prediction: For a new input, calculate the distance to all points in the dataset, select the k closest points, and average their target values to make the prediction.

KNN Regression is intuitive and simple to implement but can be computationally intensive for large datasets. Proper tuning of k and the distance metric is essential for accurate predictions, often achieved through cross-validation and hyperparameter tuning [17].

2.5 Deep Learning Algorithms:

The primary deep learning algorithms include Recurrent Neural Networks (RNNs) for sequential data, and Long Short-Term Memory (LSTM) networks for tasks requiring long-term dependencies. In our study, we will explain RNN and LSTM algorithms.

2.5.1 Recurrent neural network:

RNNs are a specialized type of artificial neural network designed for handling time-series or sequential data, where data points are interdependent. Unlike feedforward neural networks, which are used for independent data points, RNNs incorporate dependencies between sequential data points. This is achieved through a concept of memory, allowing RNNs to store states or information from previous inputs to generate the next sequence of outputs.

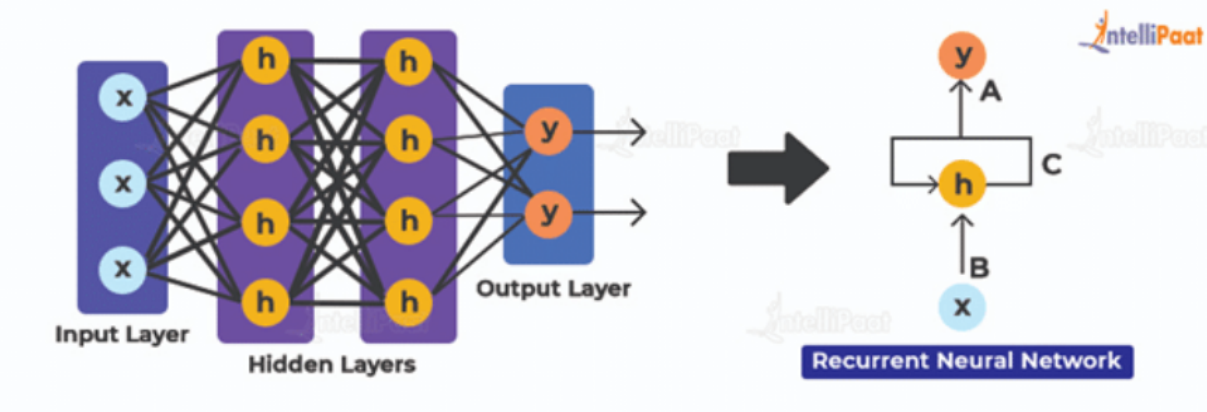


Figure 2.4: Recurrent Neural Network architecture [18].

It saves the output of a particular layer and feeds this back to the input to predict the output of the layer.as illustrated in Figure 2.4, which shows the architecture of Recurrent Neural Networks [18].

2.5.2 Long short-term memory:

This is the most common type of recurrent neural network .LSTMs are useful for remembering information for a long time because they store their internal memory in the same way a computer does; they read, write, and delete information as needed using gates. These gates help the network decide which information to keep and which to delete from memory (whether to open the gate or not) based on the importance attributed to each bit of information. This is very advantageous as it not only allows for the storage of more information (in the form of long-term memory) but also helps to eliminate unnecessary information that could alter the result of a prediction, such as article in a sentence [19]. The figure 2.5 shows an example of the LSTM architecture:

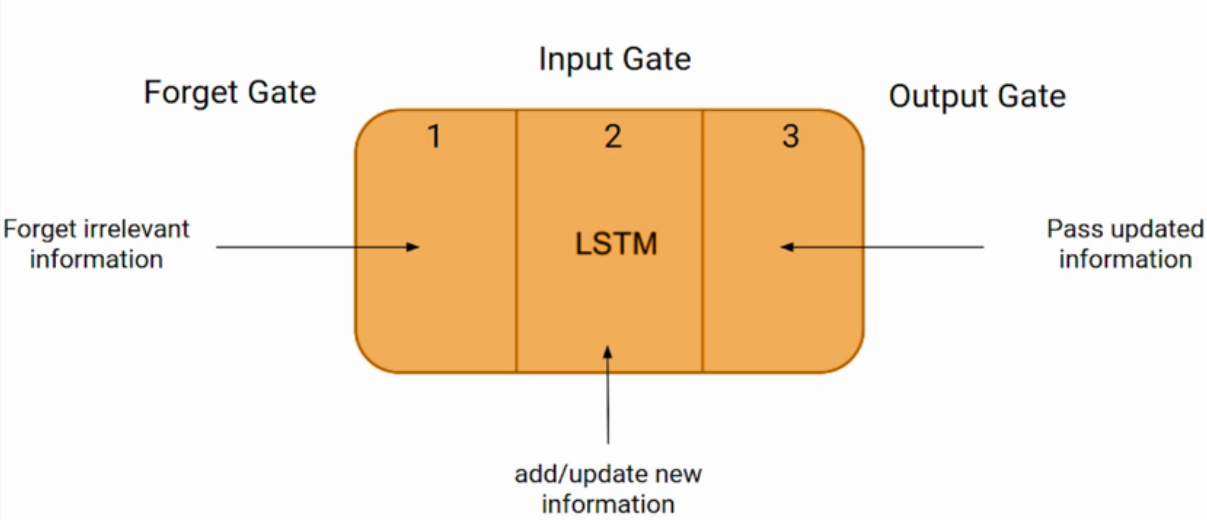


Figure 2.5: Long Short Term Memory Recurrent Neural Network Based Workload Forecasting Model For Cloud Datacenter [20].

2.6 Conclusin :

In conclusion, this chapter provided an essential overview of machine learning and deep learning, highlighting their roles in developing predictive and analytical models. We explored various algorithms and their practical applications, demonstrating the importance of these technologies in technological innovation and the enhancement of intelligent systems. In the next chapter, we will discuss the architecture and design of our system, detailing the different components and their integration to meet the defined objectives.

Chapter 3

Proposed Proprietary location system

3.1 Introduction

In this chapter, we explore the objectives, architecture, and design of our proposed system using UML modeling language. We begin by detailing the system's specific goals. Following this, we examine the system architecture, outlining the key components and their interactions. Finally, we discuss the system design, employing UML diagrams to illustrate the structure and functionality. This concise overview provides a comprehensive understanding of the system's purpose and implementation.

3.2 Objectives

The advancement in mobile technology and the need for accurate location services highlight the limitations of GPS due to environmental factors and battery consumption, especially indoors. To address these challenges, we propose a location system using cell tower signals and fingerprinting techniques, combined with machine learning and deep learning models. This system aims to provide a precise and reliable solution for determining user locations in various scenarios, enhancing location-based services.Our solution attempts to:

- **Improvement of Localization Accuracy:** Use digital fingerprints of cellular signals to provide more accurate location estimates than traditional GPS-based methods, especially in indoor or dense urban environments.
- **Resilience to Environmental Conditions:** Leverage the capabilities of machine learning and deep learning models to manage variations and interferences in signals, ensuring reliable localization even in complex and dynamic environments.

- **Reduction of Deployment Costs:** Implement a localization system that utilizes existing cellular signal, thereby reducing the need for additional equipment and the associated installation and maintenance costs.
- **Collection and Analysis of Rich Data:** Enable the collection of detailed cellular signal data and its analysis, providing valuable insights for the continuous improvement of localization systems and for other research.
- **Compatibility and Easy Integration:** Develop a system compatible with various mobile devices and easily integrable into existing applications, ensuring rapid adoption and widespread use.

In summary, our objectives aim to enhance localization accuracy, reduce costs, and improve user experience. In the next section, we will discuss our system's architecture.

3.3 Overview of system

Our system uses the fingerprinting technique, specifically RSSI fingerprinting, as detailed in the first chapter. During the offline phase, we store coordinates and signal strength readings in a database. In the online phase, current signal measurements are matched with this data to determine the device's location. Despite challenges like environmental changes and signal interference, machine learning algorithms enhance real-time location accuracy [4].
Our system operates in two phases: offline and online. In the offline phase, data is collected using a fingerprint collector applications and stored in a database. The data is then preprocessed and used to train models, which are evaluated for accuracy. In the online phase, the location predictor module determines the user's real-time position. A figure 3.1 illustrating our system accompanies this description for better understanding.

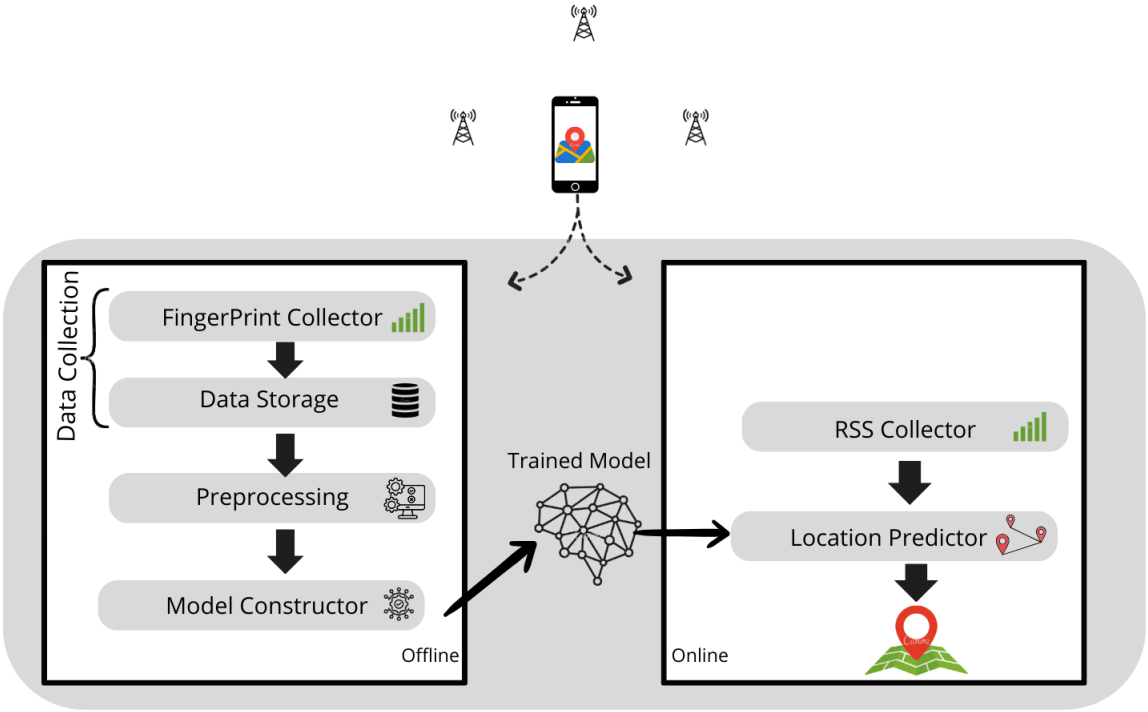


Figure 3.1: Architecture of the Fingerprinting-Based Localization System.

In summary, we have provided an overview of our system architecture. In the next section, we will detail the creation of our dataset and describe the system's modules, including data collection, preprocessing, model construction, and location predictor.

3.4 Building the dataset

In the modern era of ubiquitous mobile connectivity, accurately determining the position of a device based on cellular signals is essential for numerous applications. A comprehensive dataset of cellular signal strengths from different towers can significantly enhance location-based services. Our project aims to collect and analyze cellular signal data from various mobile devices to develop a robust and reliable dataset, enabling the training of machine learning and deep learning models for accurate indoor localization. In this section, we will discuss the major issues that led us to create our own dataset.

3.4.1 Problems:

Here is a summary of the problems that led us to create our own dataset:

1. **Incomplete or Inaccurate Data**: Existing datasets may not cover all situations and environments necessary for our precise localization models, leaving critical gaps in the data.

2. **Variability of Data Sources**: Datasets from different sources may have inconsistencies in terms of format, accuracy, and collection methods, making it difficult to integrate them into a single model.

3. **Scalability**: Existing datasets may not be large or diverse enough to train robust machine learning and deep learning models, limiting their effectiveness in varied real-world scenarios.

4. **Restricted Data Access**: Many existing databases are protected by access restrictions, copyrights, or privacy constraints, making their use difficult or impossible for our project. Additionally, we did not find a dataset that met our specific requirements.

5. **Data Obsolescence**: Existing data may be outdated and not reflect current cellular network conditions and environments, necessitating the collection of new, more relevant data.

These problems led us to build our own dataset, specifically tailored to our indoor localization needs, ensuring the accuracy of our models.

3.4.2 Data collection:

In the offline phase, the data collection module consists of two components: **fingerprint collector** and **Data Storage**. We collected information about cells and their locations in a 200 m^2 area, as depicted in the screenshot of the house from the map (Figure 3.2). The components of this module and the tools used for data collection are explained below.

Figure 3.2: Screenshot of the house from the map.

- **Fingerprinting Collector:** In our system, we utilize various mobile phones and one fixed phone, each equipped with specialized applications tailored to their specific functionalities. These phones connect to a centralized server, which in our case is our PC (Personal Computer). The server efficiently manages the connections and distinguishes between mobile and fixed devices, facilitating smooth communication and data transfer via sockets. The main objective of this architecture is to capture signal information from cell towers and their locations using different types of phones (mobile phones and fixed) and store this data in the next module.

  In the following , we will explain the roles of the elements that contributed to capturing this information. These elements include:

  - **Mobile phone:** The mobile phone used by the collector moves to different locations to collect cells and location informations. The user interacts with a map by clicking on their position. When the user clicks, the mobile phone sends a request to the fixed phone through an intermediate server. The fixed phone responds with the cell informations it detects, also via the server. The mobile phone receives both its own cell and location informations as well as the fixed phone's cell informations. All this data is then stored in a database on the mobile phone.
  - **Fixed phone:** remains static and is used to provide cell information on request. When it receives a request from the mobile phone via the intermediate server, it automatically responds by sending the cell informations it detects. This informations is crucial for understanding cell coverage and signal variations in fixed location. The fixed phone serves as a stable reference point for signal measurements.
  - **Note:** When collecting information about cells at different locations, we can use multiple mobile phones and a single stationary fixed phone.
  - **Server:** The intermediate TCP server, developed in an Eclipse IDE, manages connections and communication between multiple mobile phones and a single stationary fixed phone using sockets to ensure smooth and secure communication. It receives requests from the mobile phones, transmits them to the fixed phone, and then sends the responses back to the mobile phones. It also synchronizes the data to ensure correct information transfer. The server's role is essential for maintaining effective and reliable communication between the devices. The communication between these devices and the server will be illustrated in Figure 3.3.

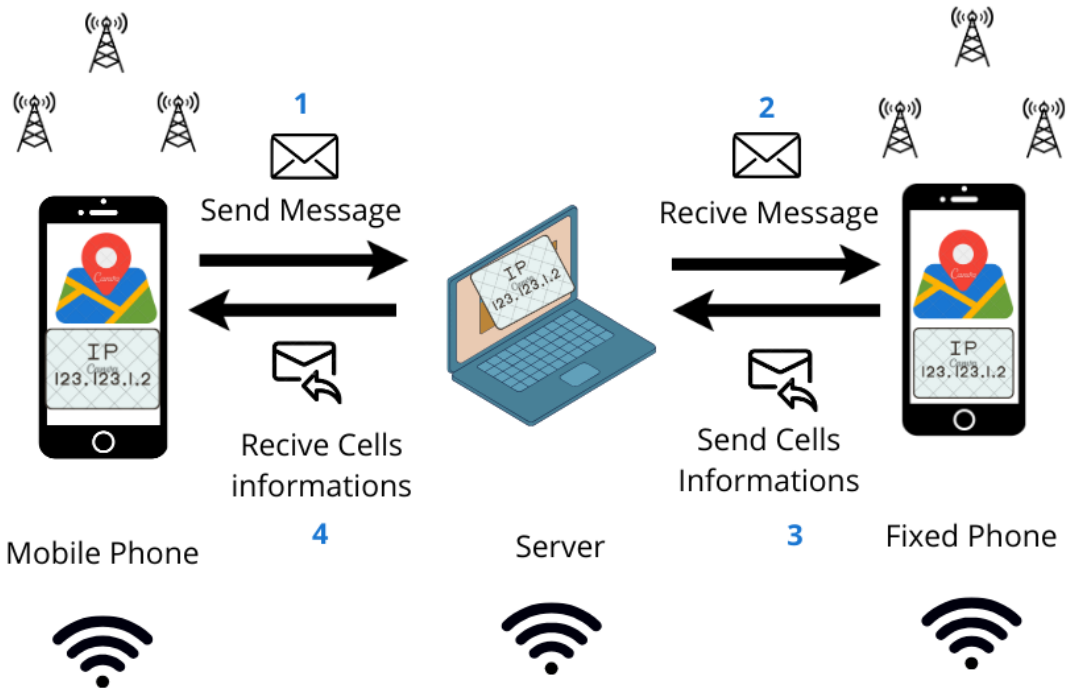


Figure 3.3: FingerPrint Collector.

– **Data Storage :** The data storage module records the cellular information collected by the fingerprint collector at various locations. The details of the cells and their locations from both types of phones (mobile and fixed) are stored in an SQLite database to ensure complete traceability and efficient use. This database allows for quick management and retrieval of information. The stored data is then used in the creation of our dataset. An illustration of this process is presented below (Figure 3.4) to show how we organize and store this essential data.

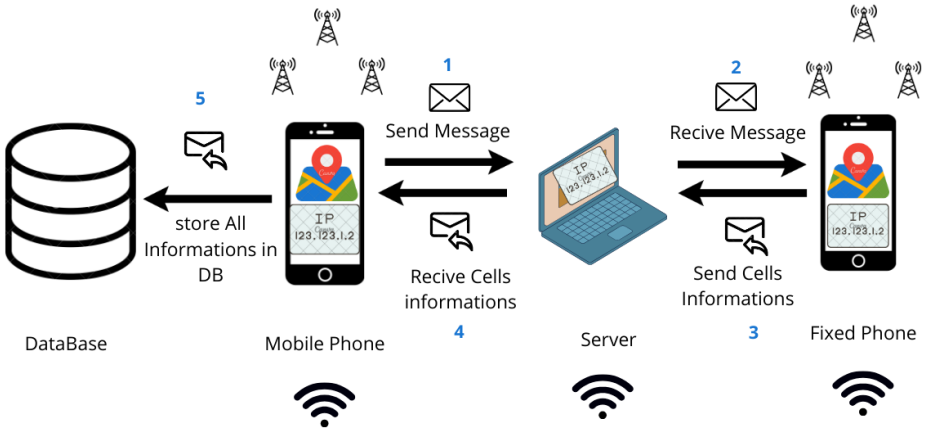


Figure 3.4: Data Storage.

Dataset Description

Creating a complete dataset is essential for training machine learning and deep learning models. A well-structured and well-explained dataset ensures data reliability, facilitates analysis, and allows for the reproduction of the results.

In previous studies on cellular-based indoor localization, various datasets were developed using mobile devices to capture Received Signal Strength (RSS) from cell towers, accompanied by coordinates that denote reference points in indoor environments. For instance, **the MonoDCell** dataset was collected from Android phones in two distinct indoor environments, measuring 132 m^2 and 629 m^2. This dataset provided data on RSS including the Cell Tower ID (CID) and the coordinates of reference points within the testbeds [5]. Similarly, **the CellinDeep** project utilized multiple Android devices to gather RSS, CID, and reference point coordinates within indoor spaces, applying data augmentation methods to enrich the dataset [7]. Additionally, **the OmniCells** collect data from various locations, such as an apartment and a campus building, using multiple phones to capture RSS, CID, and reference point coordinates in the indoor testbeds [6].

In the context of this research, our dataset was constructed based on cell information collected from various known locations using the data collection module. The data collection process involved capturing signal characteristics, specifically Received Signal Strength (RSS) and Cell Tower ID (CID), along with geographical coordinates (longitude and latitude) and additional signal metrics from both mobile phones and a stationary smartphone. We introduced a unique feature by incorporating a stationary smartphone, enabling us to analyze the effects of data collection with both moving and stationary devices. This distinction allows us to gain a better understanding of how signal strength changes in different scenarios, depending on whether the phone is mobile or stationary. In our case, we can capture four types of cells (LTE, GSM, WCDMA, and CDMA), but in our dataset, we only captured LTE and GSM. The collected data was initially stored in a database and later converted to an Excel file for ease of use.

Our dataset contains 15,191 rows, each representing a unique observation. The data includes detailed information on cells collected at various known locations and at different times, providing a comprehensive overview of signal characteristics in different environments. The organized structure of this dataset enables optimal model development, ensuring accurate predictions of user positions.

Below is a detailed description of the dataset's columns, presented in Table 3.1 :

Column Name	Data Type	Description
_id	Integer	Unique identifier of Localisation
longitude	Float	Geographic longitude
latitude	Float	Geographic latitude
address	String	Address
time	Date	Date and time
_idC	Integer	Unique identifier of Cells
_idCell	Integer	Cell identifier (CID)
signalStrengthLevel	Integer	Signal strength level
signalStrengthDbm	Integer	Signal strength in dBm (RSS)
signalType	Integer	Signal type [LTE,GSM,WCDMA,CDMA]
describeContents	Integer	Content description
asuLevel	Integer	ASU (Arbitrary Strength Unit) level
ecNo	Integer	Energy per chip over the noise spectral density) as dB
cqi	Integer	Get channel quality indicator
cqiTableIndex	Integer	Get table index for channel quality indicator
rsrp	Integer	Get reference signal received power in dBm
rsrq	Integer	Get reference signal received quality
rsri	Integer	Get Received Signal Strength Indication (RSSI) in dBm
rssnr	Integer	Get reference signal signal-to-noise ratio in dB
timingAdvance	Integer	Timing advance
cdmaDbm	Integer	Get the CDMA RSSI value in dBm
cdmaEcio	Integer	Get the CDMA Ec/Io value in dB*10
cdmaLevel	Integer	Get cdma as level 0..4
evdoDbm	Integer	Get the EVDO RSSI value in dBm
evdoEcio	Integer	Get the EVDO Ec/Io value in dB*10
evdoLevel	Integer	Get Evdo as level 0..4
evdoSnr	Integer	Get the signal to noise ratio.
bitErrorRate	Integer	Bit error rate
typeUser	String	User type[MOBILE Or FIXED]

Table 3.1: Description of dataset attributes.

In summary, successfully creating our dataset showcases our ability to effectively gather and organize data. This accomplishment is a significant step forward in our project, providing a solid groundwork for further development and achieving our goals. We have named our dataset "CellSignalTracker," reflecting its purpose and the comprehensive nature of the data it contains. The "CellSignalTracker" dataset will serve as a critical resource for training and validating our machine and deep learning models, enabling precise and reliable predictions of user locations.

3.4.3 Analysis Scenarios:

After collecting the data and constructing our dataset, we defined two scenarios for analysis.

1. **First Scenario: Mobile Phone Data Only**

   In this scenario, we use only the data collected from mobile phones. This data will be processed in the following modules to develop a model based exclusively on mobile devices information. The objective is to determine the model's accuracy and efficiency using only this type of data.

2. **Second Scenario: Mobile and Fixed Phone Data**

   In this scenario, we combine data from both mobile and fixed phones. This data will also be processed in the following modules, allowing us to develop a model enriched by the diversity of information sources. The objective is to evaluate whether integrating fixed phone data improves the model's accuracy and efficiency compared to using only mobile phone data.

Each scenario will go through the following modules for processing and analysis.

3.5 Preprocessing Module:

Data preprocessing involves evaluating, filtering, manipulating, and encoding data to ensure it is comprehensible and usable for analytical purposes. The primary goal of data preprocessing is to address issues such as missing values, enhance data quality, and make the data useful for further analysis [21]. The data preprocessing module is essential in our case for organizing and structuring the data for future use. We followed specific steps to prepare our data effectively. These steps will be explained in detail below and illustrated in Figure 3.5, which represents the preprocessing steps.

Figure 3.5: Steps of Data Preprocessing.

3.5.1 Data Cleaning Steps

As part of data preprocessing, we began with data cleaning to ensure its quality and reliability. This crucial step helps eliminate anomalies and prepare the data for further analysis. Here are the main steps we followed:

- **Handling Missing Values:** Missing values can occur for various reasons, including issues during data collection, missing information, or loading failures. They pose problems not only for data analysis but also for the accuracy of results, as they complicate data visualization and interpretation.
- **Removing Duplicates Values:** Detecting and removing duplicate data is often necessary in the data cleaning process. Duplicates can introduce bias and distort results, thus negatively affecting the performance of analytical models.

After collecting the data and constructing our dataset, we performed data cleaning, ensuring that there were no duplicate or missing values present. This essential step is crucial for preparing the data for the subsequent stages of preprocessing, allowing for more accurate and reliable analyses in later processes.

3.5.2 Feature Extraction:

Feature extraction transforms raw data into a set of meaningful features, highlighting relevant aspects and reducing complexity. This step includes attribute selection, creating new features, and encoding them.

- **Feature Selection:** identifies the most relevant attributes, simplifying the data and improving model accuracy and generalization.
- **Encoding Selected Features:** Our dataset contains two string-type attributes that needed to be transformed into binary values to be used as input for our models. This transformation is necessary because models require input in the form of numerical vectors.

3.5.3 Normalization

In our case, all attributes have been normalized to the range [0,1] using the min-max scaler method. This technique scales the data so that it falls between [0,1], while preserving the relative distances between values. The process involves subtracting the minimum value min(x) and dividing by the range (max(x) - min(x)), according to the following formula:

$$X_{\text{new}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} [22] \tag{3.1}$$

This normalization ensures a uniform scale, thus facilitating the efficient use of the data in our KNN and LSTM models.

3.6 Model Constuctor:

This module aims to predict user locations using various advanced machine and deep learning algorithms. Our dataset contains cellular data records from various locations, collected from mobile and fixed phones using an architecture detailed in the previous section. Using this collected information, we will train our models to accurately predict users' geographic positions. We split our dataset into 70% for training and 30% for testing. We used different algorithms, including KNN for machine learning and RNN ,LSTM for deep learning, chosen for their effectiveness with complex and temporal data. Figure 3.6 illustrates the architecture of our model, showcasing the different layers to produce accurate predictions at the output.

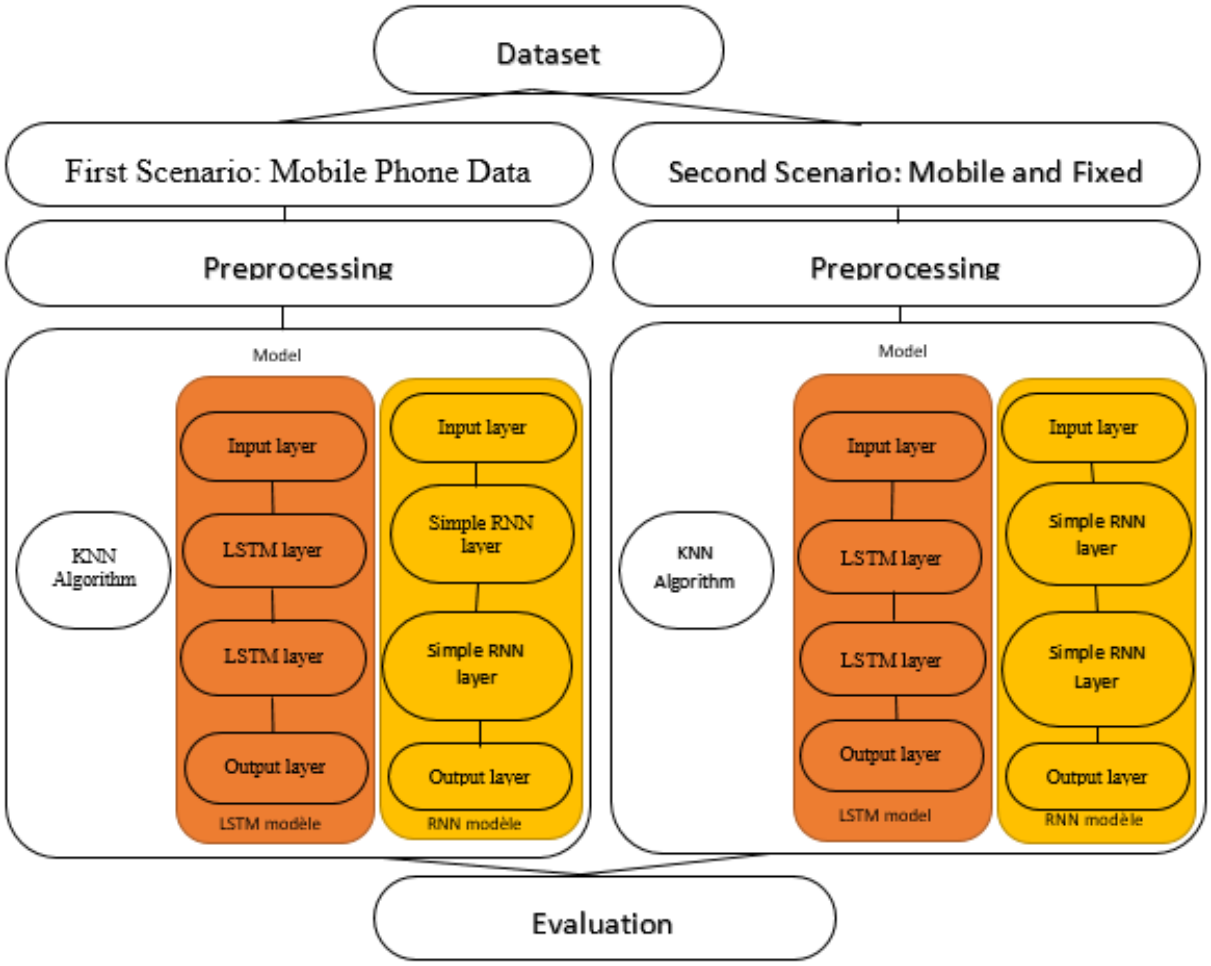


Figure 3.6: Model architecture.

- **Input Layer**: Defines the shape and structure of the input data that the model will process. It is responsible for specifying the dimensionality of time sequences

and the number of features per sequence, allowing the model to understand the configuration of the data it will receive. For example, for time series data, the input layer can define the length of the sequence (the number of time steps) and the number of features at each time step.

- **LSTM Layer**: This is a recurrent layer used for time sequences. It is capable of capturing long-term dependencies in sequential data.
- **Simple RNN Layer**: A SimpleRNN layer is a sequence processing layer that uses past information to predict future information in temporal data.
- **Output Layer**: In a location prediction model, the output layer could produce x and y coordinates. The main function of the output layer is to take the internal and complex representations of the data, generated by the LSTM and RNN layers, and transform them into interpretable and usable results.

Below, we will show the tables of the structure and parameters of our models.

- The Figure 3.7 illustrates the parameters and structure of an LSTM model. This model consists of two LSTM layers with 3,360 and 6,384 parameters, respectively, followed by a Dense layer with 58 parameters. The output shapes of the LSTM layers are (None, 28, 28) for the first layer and (None, 28) for the second layer, while the Dense layer has an output shape of (None, 2). The total number of parameters is 9,802, all of which are trainable.

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 28, 28)	3,360
lstm_1 (LSTM)	(None, 28)	6,384
dense (Dense)	(None, 2)	58

Total params: 9,802 (38.29 KB)
Trainable params: 9,802 (38.29 KB)
Non-trainable params: 0 (0.00 B)

Figure 3.7: Model LSTM Architecture.

- The second Figure 3.8 presents the parameters and structure of a SimpleRNN model. This model consists of two SimpleRNN layers with 840 and 1,596 parameters, respectively, followed by a Dense layer with 58 parameters. The output shapes of the SimpleRNN layers are (None, 28, 28) for the first layer and (None, 28) for the second layer, with the final Dense layer having an output shape of (None, 2). The total number of parameters is 2,494, all of which are trainable.

Model: "sequential_5"

Layer (type)	Output Shape	Param #
simple_rnn (SimpleRNN)	(None, 28, 28)	840
simple_rnn_1 (SimpleRNN)	(None, 28)	1,596
dense_5 (Dense)	(None, 2)	58

Total params: 2,494 (9.74 KB)
Trainable params: 2,494 (9.74 KB)
Non-trainable params: 0 (0.00 B)

Figure 3.8: Model RNN Architecture.

We performed our RNN and LSTM model with the following training parameters:

- Optimizer: Adam optimizer
- Batch Size: 28
- Number of Epochs: 30

3.7 Test And Evaluation:

Our evaluation module measures the performance and accuracy of our regression models using various metrics. These metrics help optimize the models, evaluate their final performance, and compare them. Here, we present the regression metrics we used for our model evaluation.

1. **MSE** : Mean Squared Error, is the average of the squared errors defined by the formula:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.2}$$

2. **RMSE**: Root Mean Squared Error is the square root of the MSE. Mathematically, it is defined by:

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{3.3}$$

3. **MAE**: Mean Absolute Error is the average of the absolute values of the errors, defined by the formula:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.4}$$

The MAE is in the same unit as the predicted variable, making it easy to interpret [23].

3.8 Location Estimator:

During the online phase, the user is at an unknown location and receives signal information from nearby cell towers. This module analyzes the received cell tower data to predict the user's location. After evaluating multiple models, the best-performing model is selected for this task. By comparing the current signal data with the stored dataset, the module accurately determines the user's position. This real-time prediction is crucial for providing timely and precise location services.

In conclusion, our architecture demonstrates a comprehensive and efficient approach to our project goals. In the next section, we will present our system design using UML diagrams.

3.9 System design

In the previous section, we presented the architecture of our system. This section now focuses on its design. For enhanced presentation, we utilized a modeling language to clearly illustrate our design.We have selected UML (Unified Modeling Language) as our modeling language to illustrate many scenarios that will be used during the implementation phase. This language enables us to accurately portray and analyze different system behaviors and interactions, ensuring a seamless transition from concept to implementation. In our project, we integrated UML diagrams such as use case diagram,the class diagram and sequence diagram. These modeling tools enabled us to describe the system architecture in detail, illustrate interaction flows between components, and capture functional requirements and usage scenarios.

3.9.1 Use case diagram:

The first step in UML analysis begins with the use of use case diagrams, which aim to depict how a system operates and capture its requirements. In our system, two main actors are identified: the collector, responsible for gathering data, and the stationary actor, representing a fixed entity involved in the process. We define the overall use case diagram of our system, as represented in the figure 3.9.

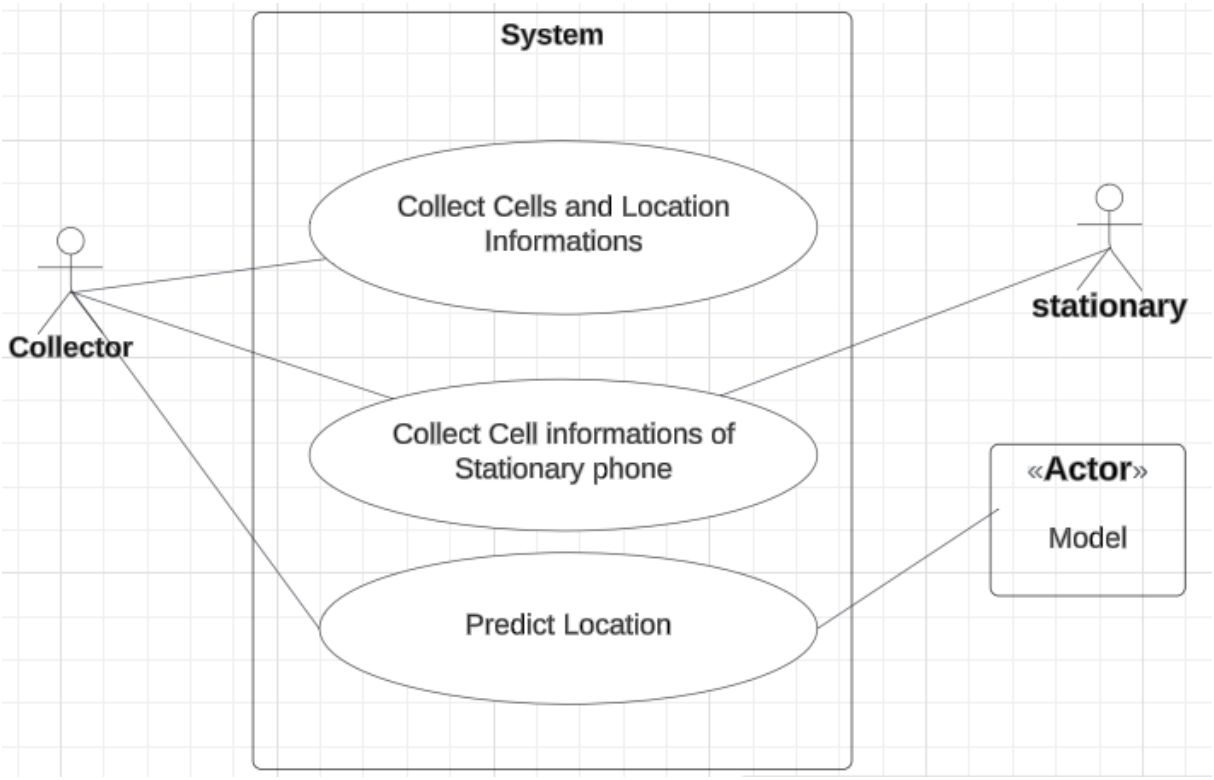


Figure 3.9: global use case diagram.

3.9.2 Sequence diagram:

In our project, after establishing the use case diagram , we developed the sequence diagrams. These were created based on usage scenarios, detailing interactions and message exchanges among different system objects. They played a crucial role in dynamically illustrating how components interact to perform specific actions defined in the use cases. In this section, we will analyse the most complex functional requirements of our system. We'll start by analysing Get Location and Cells informations , followed by Get Cells informations of Stationary phone and then Prediction localisation .We will present these diagrams below:

- **Get Location and Cells informations:**

  The sequence diagram GetLocation and Cells Information (Figure 3.10) illustrates the process by which the system collects location data and cell tower information from a moving user via a mobile phone, capturing cell information from various locations and storing it in a database.

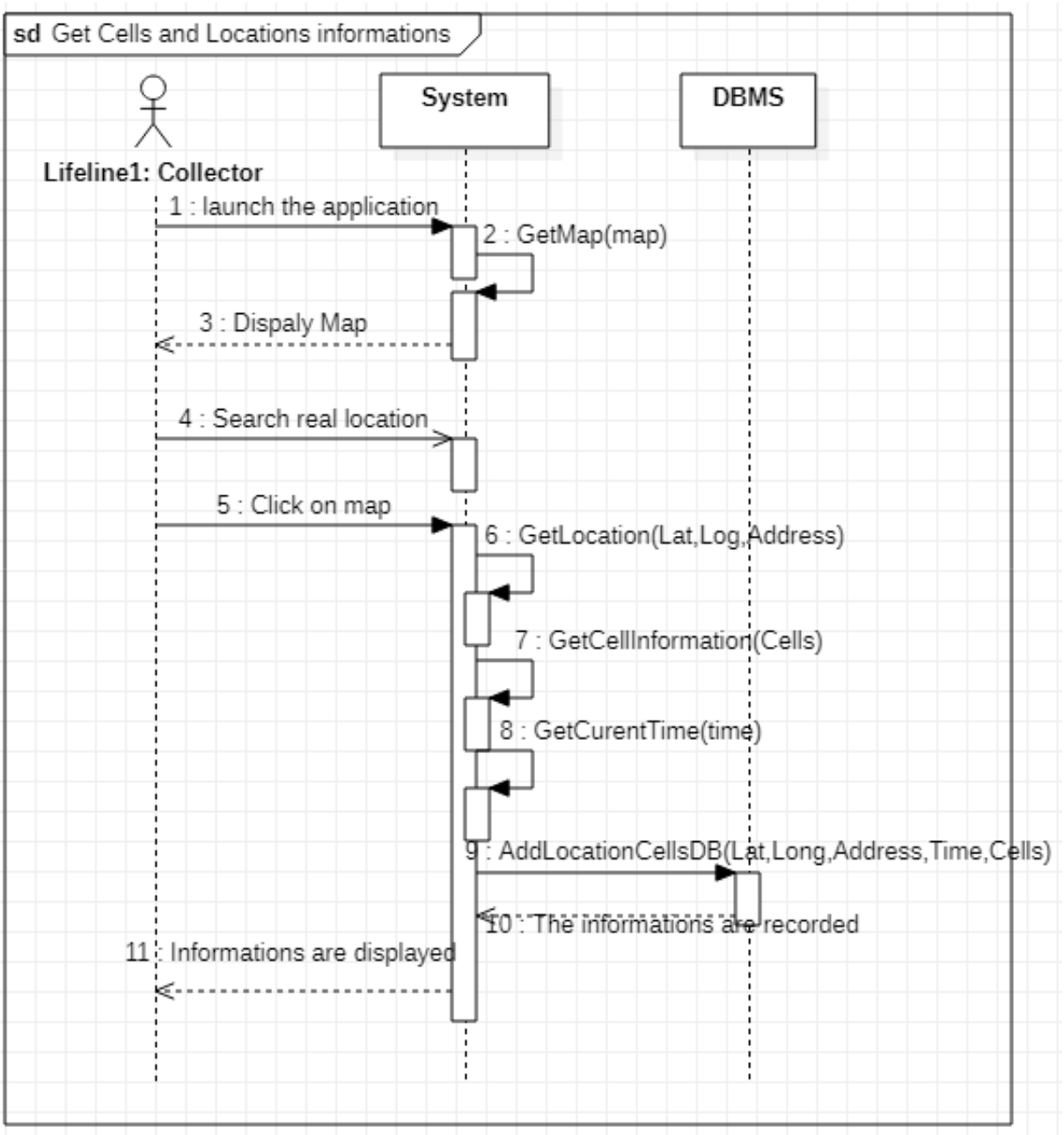


Figure 3.10: Get location and Cells informations.

- **Get Cells informations of Stationary phone:**

  The sequence diagram Get Cells Information of Stationary Phone (Figure 3.11) represents the process where a mobile collector interacts with a server to request and retrieve cell tower information from a stationary (fixed) phone. The collector first connects to the server, which then communicates with the stationary phone to gather the necessary cell data. Both the mobile phone, acting as a mobile station, and the fixed phone, acting as a fixed station, will be connected. Once the cell information from both the mobile and stationary phones is collected, it is displayed to the collector and stored in the database for further use.

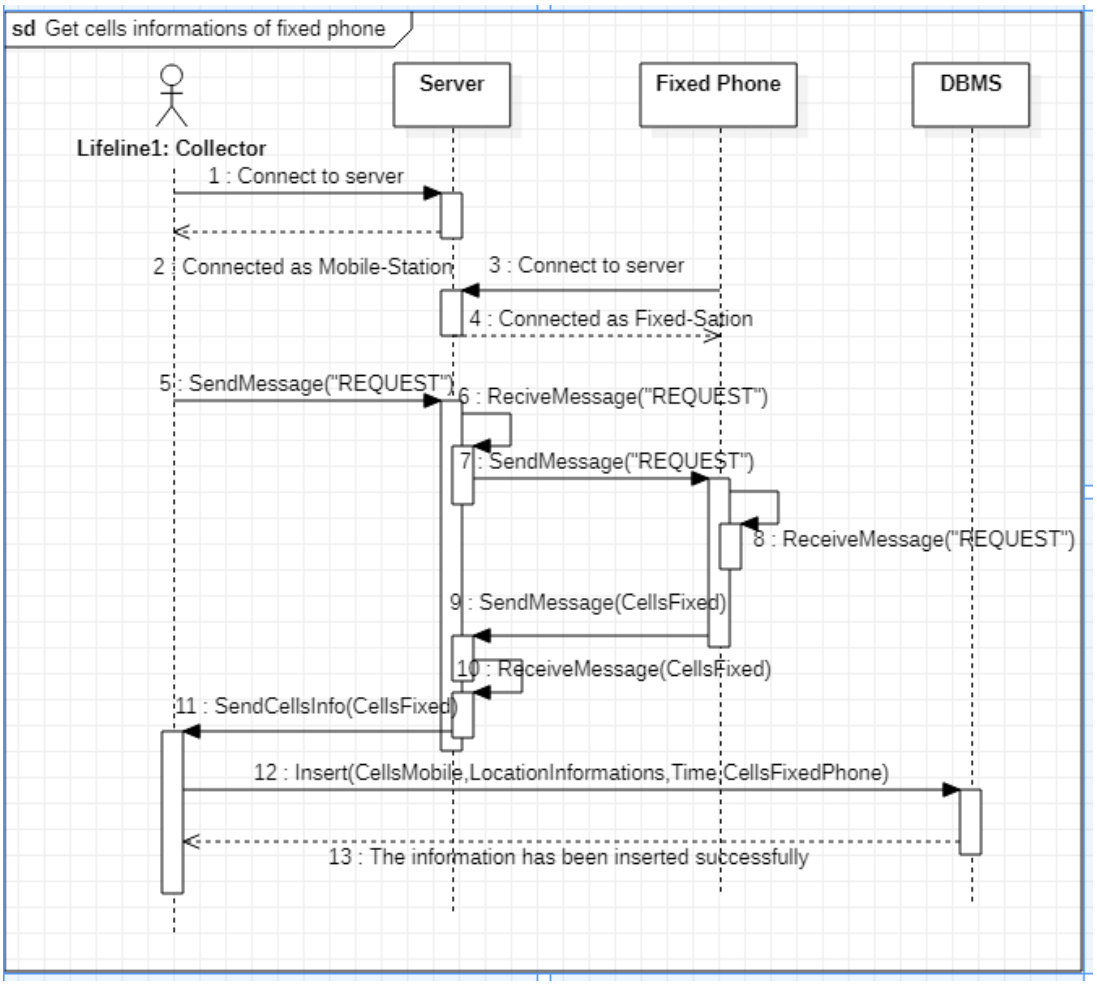


Figure 3.11: Get Cells informations of Stationary phone.

- **Prediction of location:**

  The figure 3.12 presents the sequence diagram concerning the location prediction scenario.

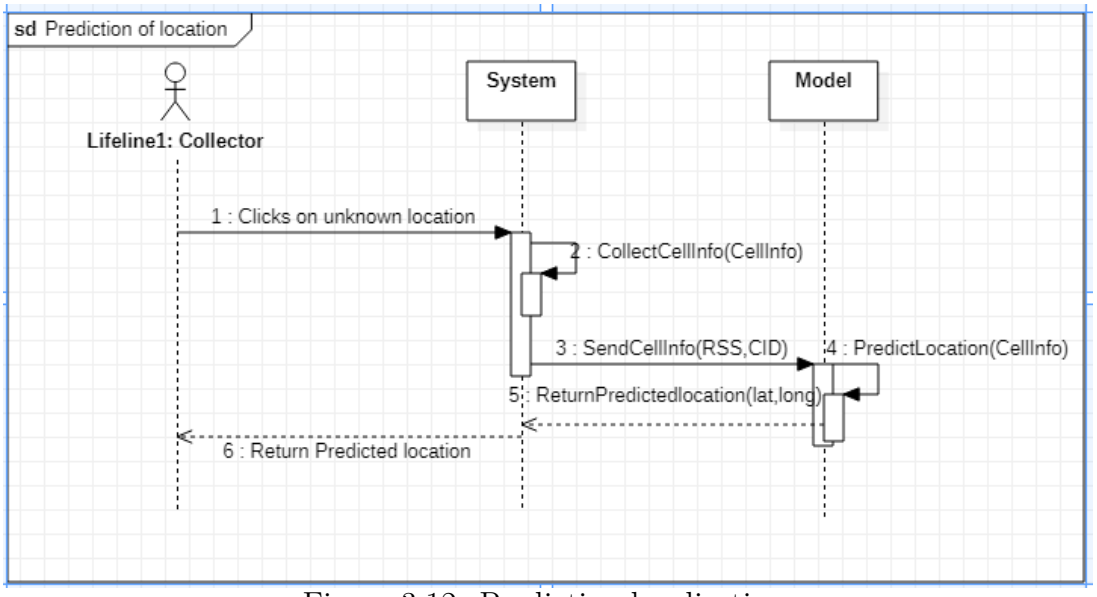


Figure 3.12: Prediction localisation.

3.9.3 Class diagram:

The class diagram represents the structure of a system designed to manage users, their locations, and cellular data. It illustrates the relationships between these entities, specifically how each user can be linked to multiple locations where cellular information is collected. The cells that the system can capture include four types: WCDMA, GSM, CDMA, and LTE. The diagram (Figure 3.13) provides an overview of these interactions.

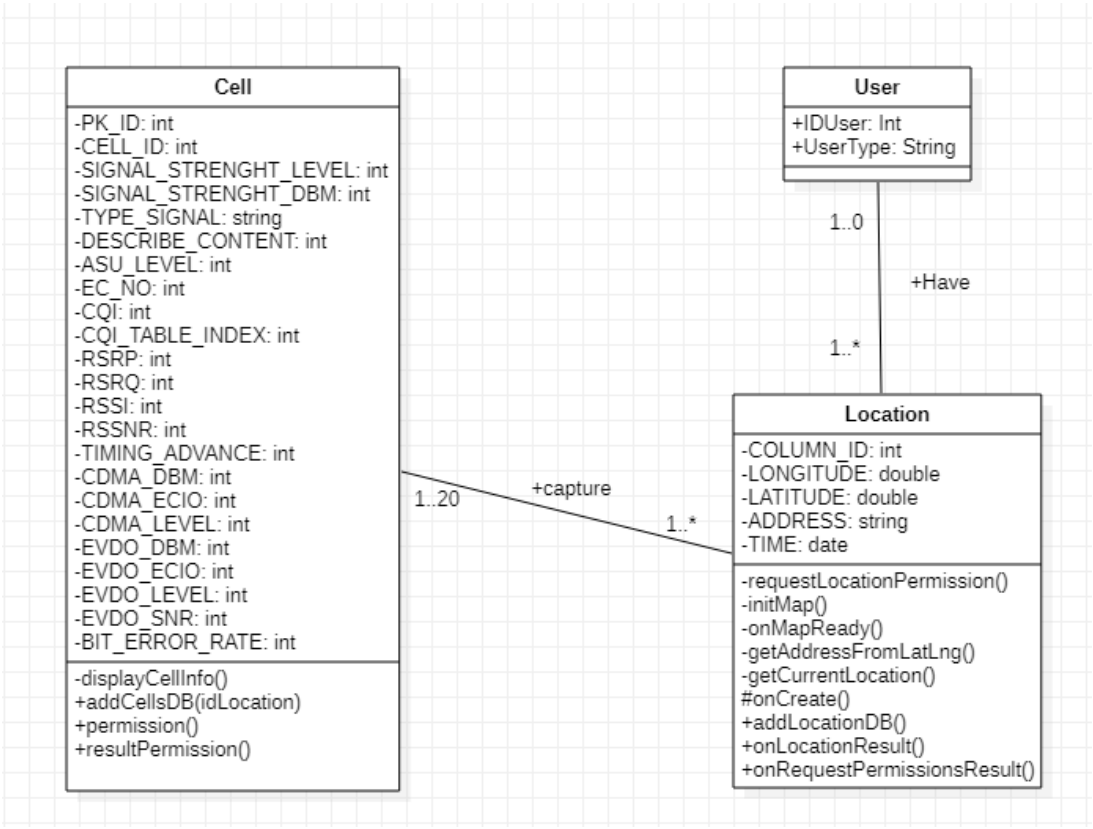


Figure 3.13: Class diagram.

3.10 Conclusion

In conclusion, our architecture demonstrates a comprehensive and efficient approach to achieving our project goals. In this chapter, we thoroughly explored the overall architecture of our proposed system, detailing its various components and how they interact to achieve our objectives. We discussed the collection and preprocessing of data, as well as the construction of machine learning and deep learning models for accurate location prediction. Additionally, we outlined the creation and significance of our custom dataset, emphasizing its role in enhancing the system's performance. We also presented the design of our system through UML diagrams. In the next chapter, we will delve into the implementation details of our system. We will also present the results of extensive testing and validation, demonstrating the effectiveness and reliability of our approach.

Chapter 4

implementation and results

4.1 Introduction

Our primary objective is to propose a proprietary location system based on cell tower signals and located devices. In this chapter, we focus on the implementation of our system. We begin by presenting the programming language, development environment, and tools used. Next, we provide a detailed overview of the various stages of our work, from data collection to the evaluation of the models used. We conclude this section by determining the most effective model.

4.2 Environment and work tools

In this section, we present the development environment and the tools used for the implementation of our system. We describe the choices of programming languages and the libraries employed, as well as the platforms and software that facilitated the development and testing of our solution.

4.2.1 Environment Hardware

The hardware used for testing is summarized in the following tables:

- **PC**:

Machine	**specifications**
Processor	Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz 2.30 GHz
Memory (RAM)	12.00 GB
Operating System	64-bit operating system, x64-based processor

Table 4.1: PC specifications.

- **Smartphone 1**:

Machine	**specifications**
Name	OPPO Reno 7
CPU	Qualcomm Snapdragon 680 Octa-core
RAM	8.00 GB
Storage	128 Go
Android version	14

Table 4.2: Phone 1 specifications.

- **Smartphone 2**:

Machine	**specifications**
Name	OPPO A77
CPU	Helio G35 Octa Core
RAM	4.00 GB
Storage	128 GB
Android version	14

Table 4.3: Phone 2 specifications.

- **Kaggle**: A subsidiary of Google, it is an online community of data scientists and machine learning engineers. Kaggle allows users to find datasets they want to use in building AI models, publish datasets, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges [24].

4.2.2 Programming language and software:

- **Java** :Java is a multi-platform, object-oriented, and network-centric language that can be used as a platform in itself. It is a fast, secure, reliable programming language for coding everything from mobile apps and enterprise software to big data applications and server-side technologies [25].

- **Python** :Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes

readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed [26]. In our case, we used Python to convert data stored in the database into a Excel file and to create machine learning and deep learning models.

- **Android studio** :Android Studio is the official Integrated Development Environment (IDE) for Android app development. Based on the powerful code editor and developer tools from IntelliJ IDEA , Android Studio offers even more features that enhance your productivity when building Android apps[27].

- **SQLite** :is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. SQLite is the most used database engine in the world. SQLite is built into all mobile phones and most computers and comes bundled inside countless other applications that people use every day [28].

- **Eclipse IDE** :In the computing world, Eclipse is an integrated development program for developing various computer applications using especially Java language as well as others, including C/C++, Python, PERL, Ruby, and many more. Being free and open source, Eclipse IDE is one of the most popular JAVA IDE in the computing market.Eclipse IDE can run on the most popular Operating Systems, including Windows, Mac OS, and Linux [29].

4.2.3 Libraries

the libraries we have imported into our project are defined in the following :

- **NumPy** :NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more [30].

- **Pandas** :Pandas is a Python library used for data analysis and manipulation. It provides two primary data structures: Series, which is a one-dimensional labeled array, and DataFrames, which is a two-dimensional labeled data structure similar to a table. Pandas enables users to perform operations such as data cleaning, transformation, and analysis efficiently, making it an essential tool for data science and machine learning tasks [31].

- **Matplotlib** :Matplotlib is a powerful plotting library in Python used for creating static, animated, and interactive visualizations. Matplotlib's primary purpose is to provide users with the tools and functionality to represent data graphically, making it easier to analyze and understand [32].

- **Keras** : keras is a high-level neural networks API, written in Python, and capable of running on top of TensorFlow. It allows for easy and fast prototyping of deep learning models and is widely used for creating and experimenting with neural networks [33].

- **Sklearn** : Scikit-learn, also known as sklearn, is an open-source, machine learning and data modeling library for Python. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python libraries, NumPy and SciPy [34].

- **Tensorflow** : tensorflow is an open-source deep learning library developed by Google. It is used for machine learning and neural network development, providing both high-level APIs for easy model building and low-level operations for flexibility [35].

4.3 Practical Deployment of the Dataset:

In this section, we discuss how the dataset was collected using a well-structured architecture, emphasizing both the mobile and fixed applications. This process ensured the comprehensive collection of cellular and location data, managed server connections, and facilitated efficient data storage and retrieval.

4.3.1 Mobile and Fixed Applications

The mobile and fixed applications share a similar user interface to facilitate the collection of cellular and location data. The mobile application is deployed on various mobile phones to collect data while moving, whereas the fixed application is installed on a single stationary Smartphone, which remains immobile, to provide stationary cellular information.

Figure 4.1 presents the user interfaces of both mobile and fixed applications designed for cellular and location data collection. The mobile application interface (left) is optimized for use on various mobile devices, allowing for data collection while on the move. In contrast, the fixed application interface (right) is tailored for stationary use, providing essential information from a single, immobile device.

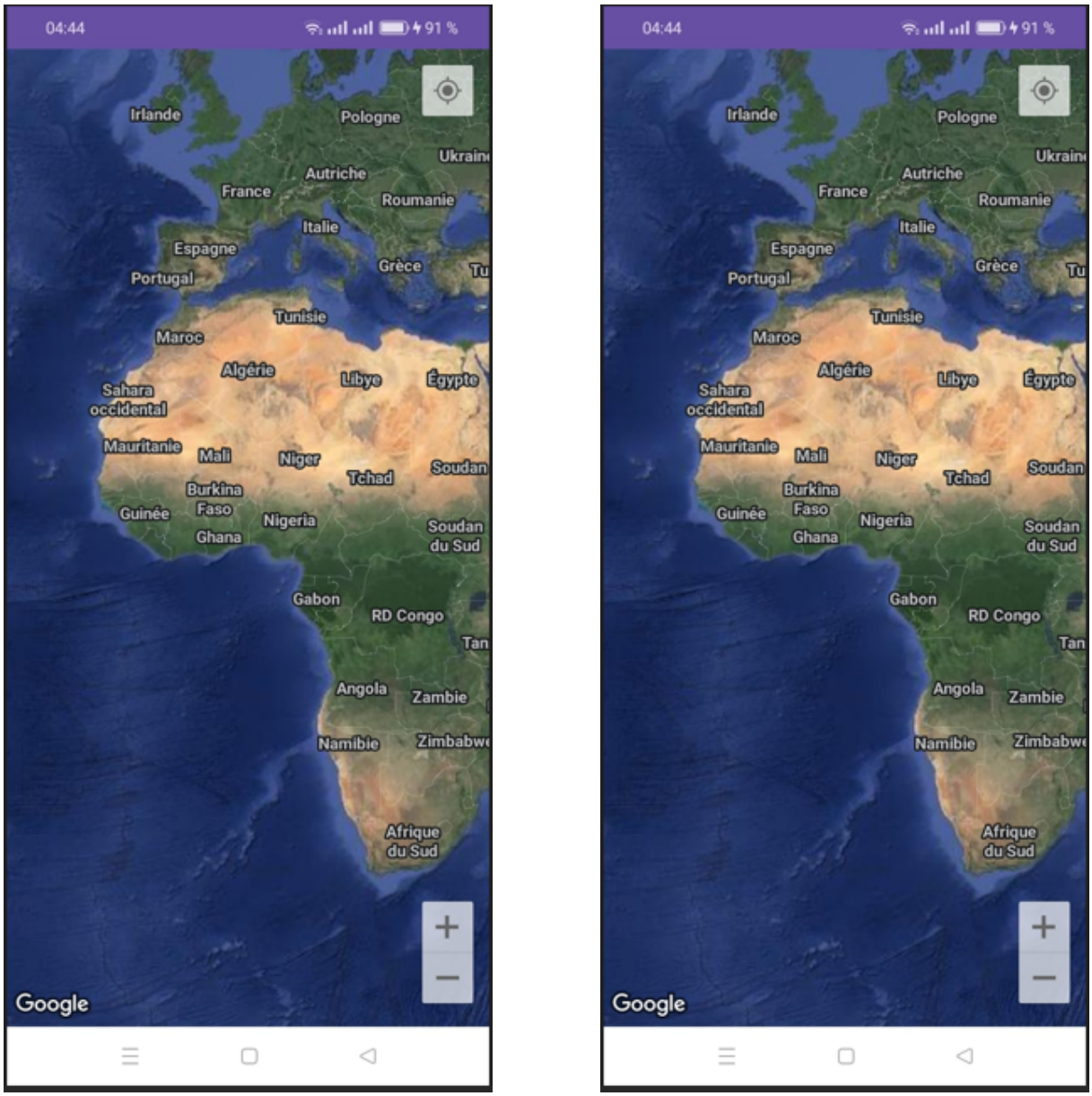


Figure 4.1: Monbile and Fixed phone interfaces applications.

4.3.2 Server Connection Management:

The server acts as an intermediary for managing connections between mobile and fixed applications. Developed in the Eclipse IDE, it facilitates secure and efficient communication using sockets. The server receives requests from mobile devices, processes them, and relays them to fixed phone. After obtaining responses, it sends them back to the mobile devices, ensuring real-time interaction. This architecture highlights the server's critical role in data collection for cellular and location-based services, improving the reliability and accuracy of the data exchange process. Figure 4.2 provides an overview of this architecture, showcasing the interactions between the server and connected devices.

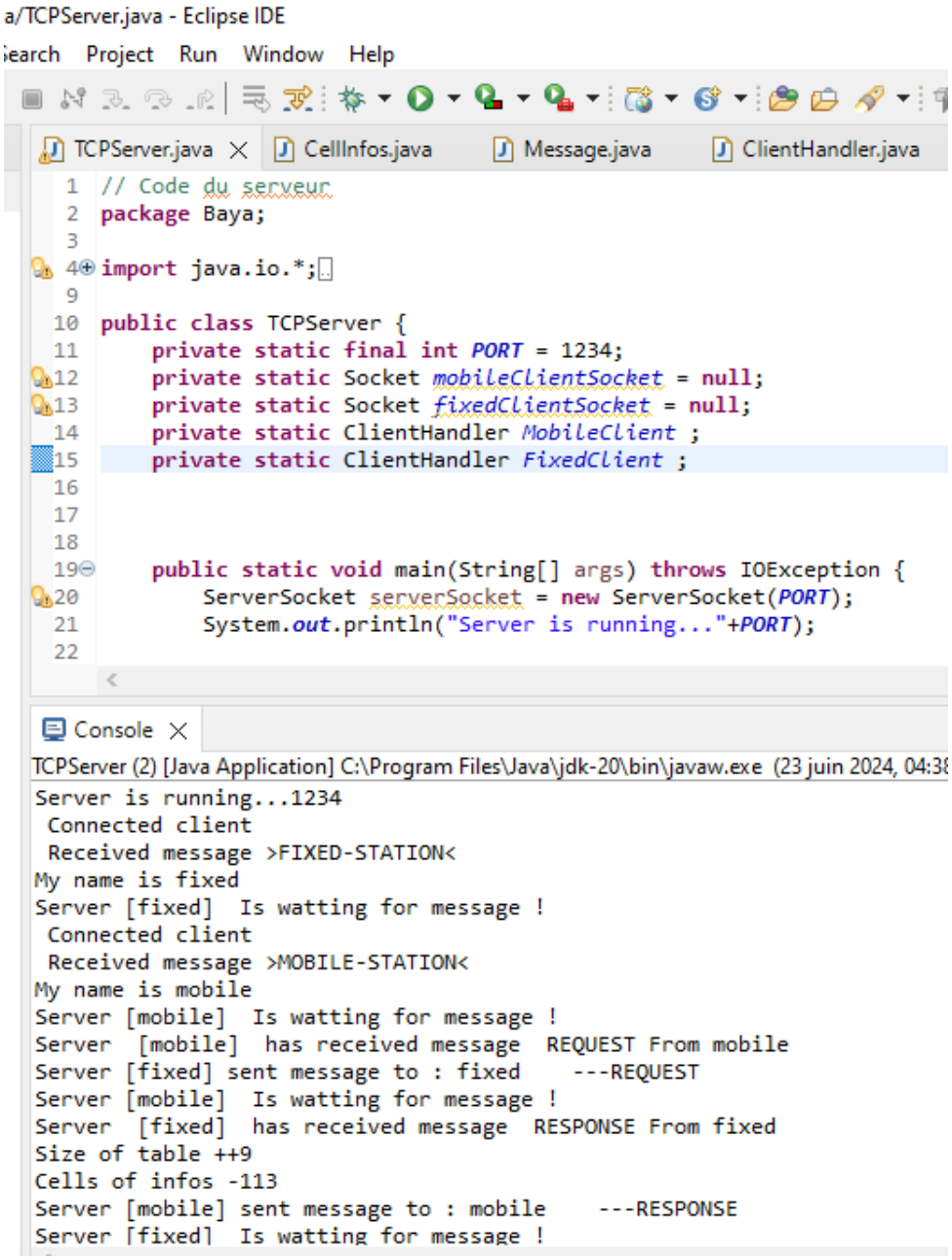


Figure 4.2: Server Connection Management.

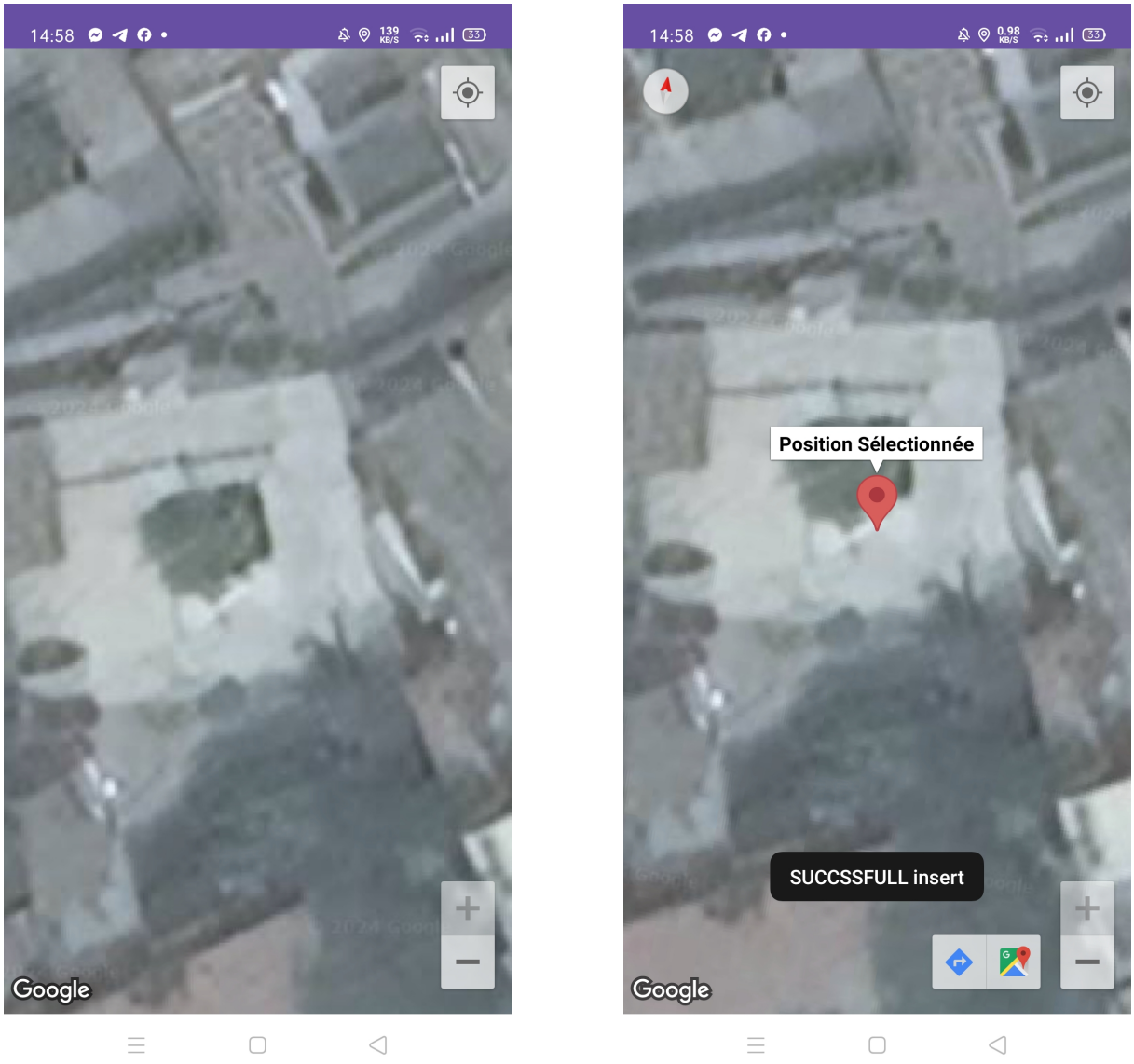
4.3.3 Communication Process:

In the communication process, we used socket which is a software endpoint that allows two programs to communicate within the same machine or over different machines connected to a network. Sockets employ a combination of IP address and port number to identify the particular application involved in communication. The combination ensures proper transmission of data at the correct destination in a networked environment. Sockets are usually used in a client-server architecture, where the server listens for client connection requests [36]. The communication scenario is as follows:

- User Location Search: After launching the mobile application, the data collector

initiates a search to determine the precise location on the map. This process allows for accurate localization of the current position, as illustrated on the left in Figure 4.3.

- Manual Location Selection: Once the location is identified, the user manually selects their precise location on the map, as shown on the right in Figure 4.3. This ensures increased accuracy in data capture.



(a) Location Search Interface. (b) Location Selection Interface.

Figure 4.3: Location Search and Selection Interface.

- Sending a Request from the Mobile Application to the Server: When a user clicks on their exact position on the map within the mobile application, a ”REQUEST” message is generated. This message is sent to the server, which then forwards the request to the fixed phone, signaling the need for cellular data capture from the fixed phone.

- Transmitting the Request to the Fixed Phone: Upon receiving the request, the server automatically forwards the message to the fixed phone, which is positioned to effectively capture cellular signals.
- Receiving Data by the Fixed Phone: When the fixed phone receives the request message (as shown in Figure 4.4), it sends all captured cells along with their respective information to the server. This step ensures smooth and efficient communication between the fixed phone and the server.

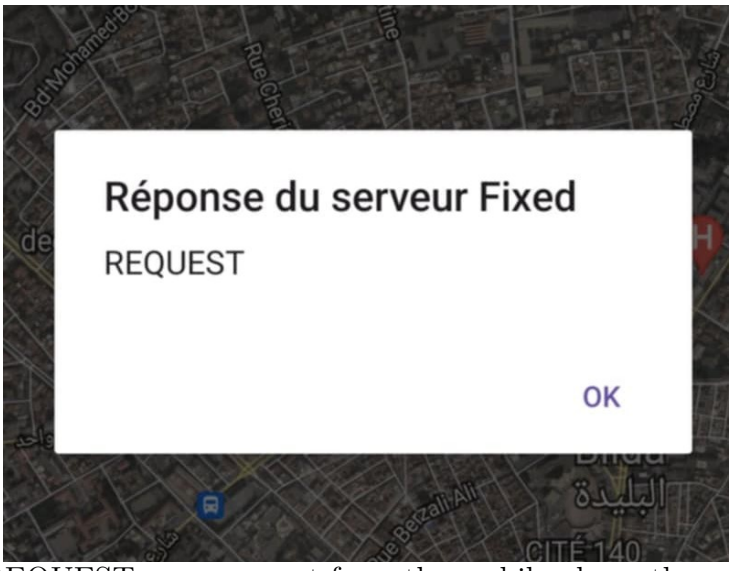


Figure 4.4: REQUEST message sent from the mobile phone through the server.

- Data Transfer to the Mobile Phone :The server then transmits this data to the mobile phone. The user receives detailed information about the captured cells in the specified area.
- Displaying Information: The location and captured cell information collected by the mobile phone are displayed on the screen, as shown in Figure 4.5 (on the left). Additionally, the data received from the fixed phone is presented to the user, as shown in Figure 4.5 (on the right).

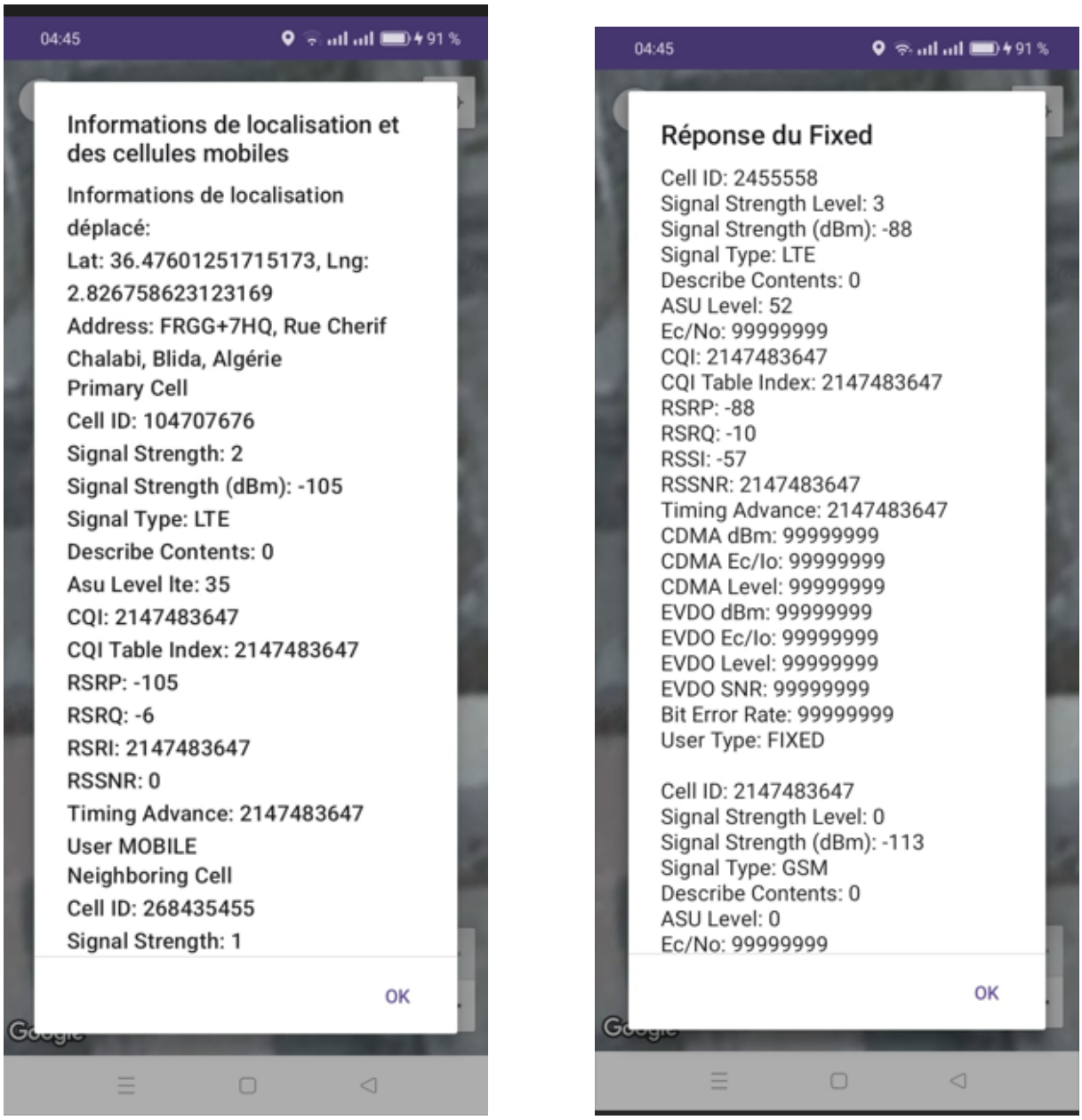


Figure 4.5: Information displayed from both the mobile and fixed phones.

- Storing Data for Future Use: Finally, all collected information, both from the mobile phone and the fixed phone, is stored in a database for future use, allowing for in-depth data analysis.

4.3.4 Data Storage:

All collected information is recorded in a SQLite database on the mobile phones. This database is structured to facilitate quick management and efficient retrieval of information.

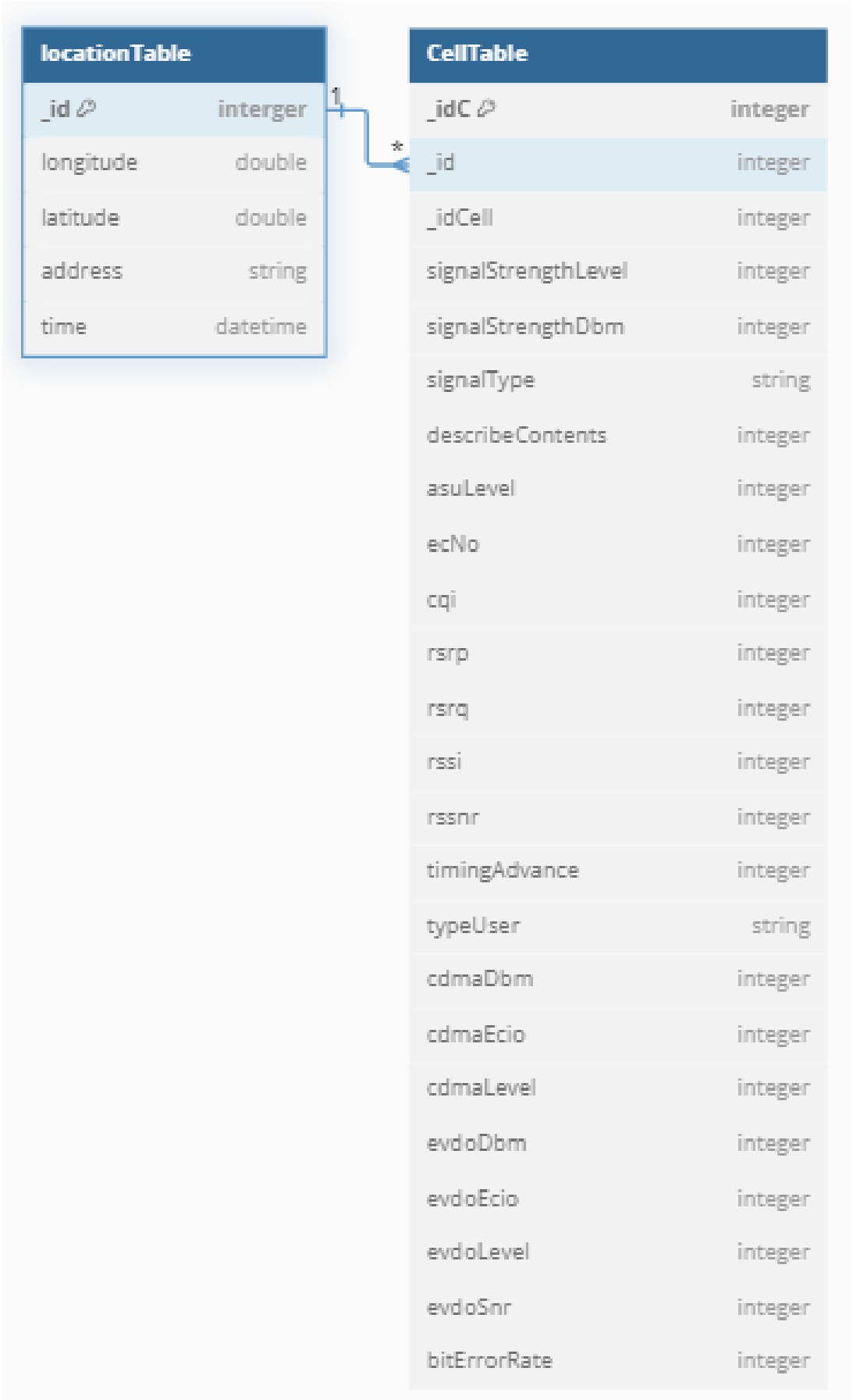


Figure 4.6: Database tables.

4.3.5 Conversion of the Database to Excel File

The process of converting the collected data from a SQLite database to an Excel file format simplifies data manipulation and analysis. As illustrated in Figure 4.7, a provided code snippet efficiently extracts data from the SQLite database and exports it to an Excel file, ensuring all relevant information is accurately preserved and easily accessible for further use.

```
import sqlite3
import pandas as pd
from google.colab import files

uploaded = files.upload()
conn = sqlite3.connect('LocationOfCel.db')
query = '''
SELECT locationTable.*, CellTable.*
FROM LocationTable
INNER JOIN CellTable ON locationTable._id = CellTable._id
'''
df = pd.read_sql_query(query, conn)
df.to_excel('donnees_combinees.xlsx', index=False)
conn.close()
files.download('donnees_combinees.xlsx')
```

Figure 4.7: Convert DataBase to Excel.

Here is our dataset in the form of an Excel file after converting from the database to a dataset, as illustrated in Figure 4.8.

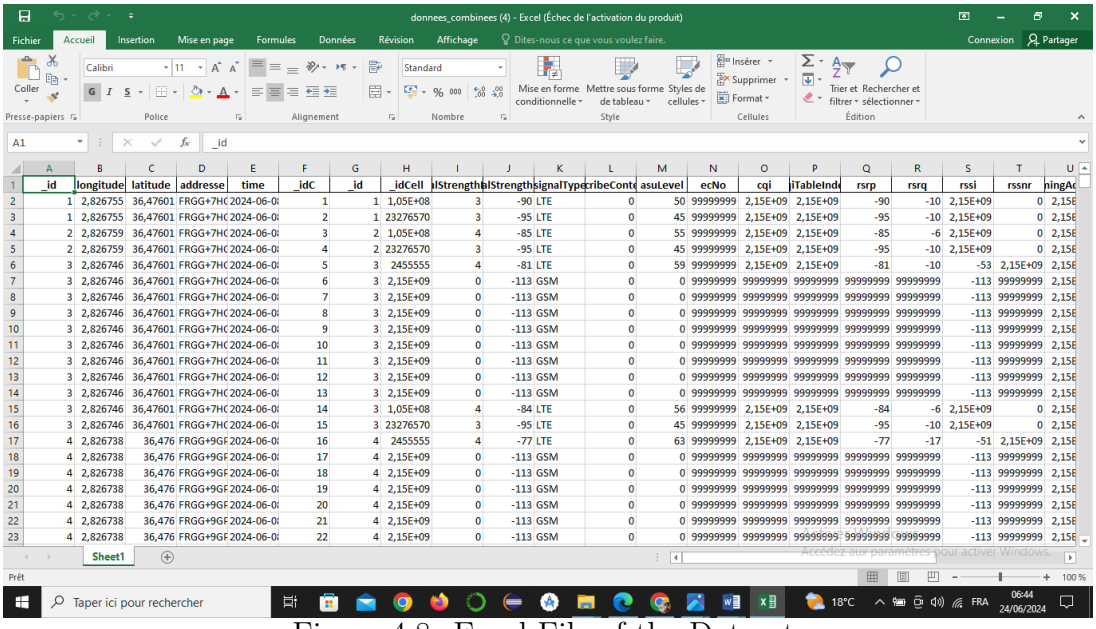


Figure 4.8: Excel File of the Dataset.

4.4 Results and Discussion:

After the data preprocessing and model training mentioned in the previous chapter, we will now discuss the results obtained from the different models. This section will

provide an analysis of the performance metrics, compare the effectiveness of each model, and highlight key observations. The goal is to understand the strengths and weaknesses of each model based on the results and to draw conclusions about their suitability for the given task. This analysis will help in identifying the best-performing model.

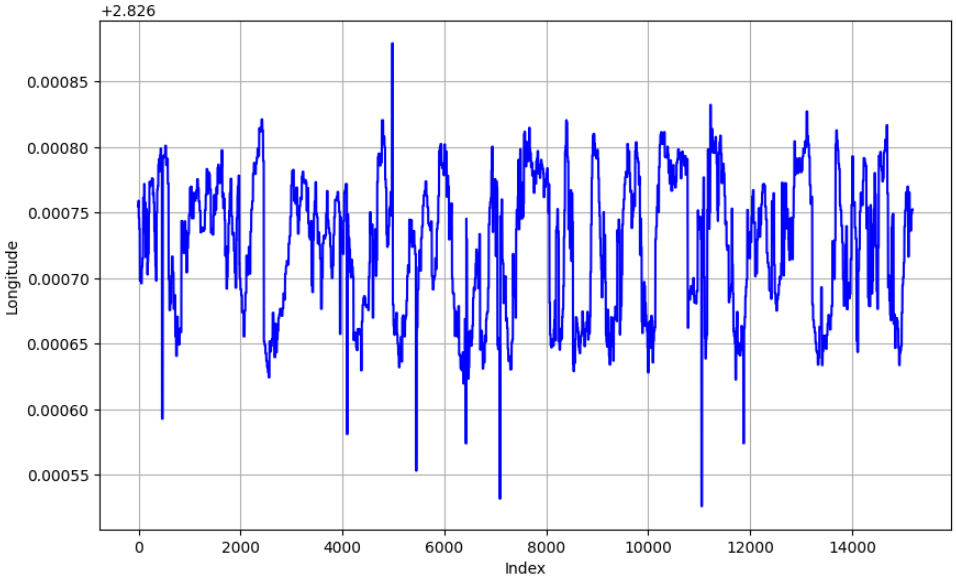


Figure 4.9: Longitude variation in our dataset.

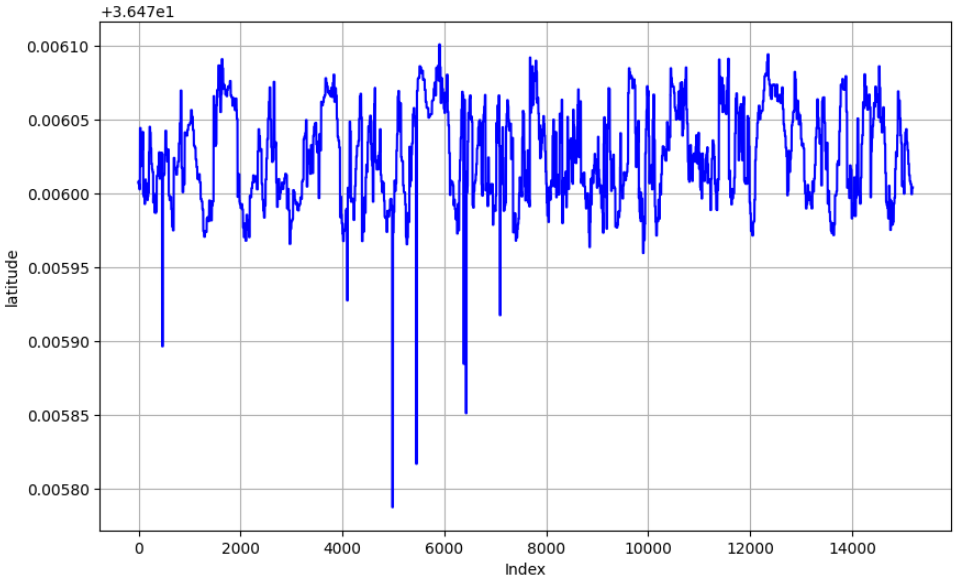


Figure 4.10: Latitude variation in our dataset.

The graphs below(Figure4.9 and Figure 4.10) show the variation of longitude and latitude over time in our dataset. We will use them to visualize geographic changes and identify anomalies or specific points of interest in the location data.

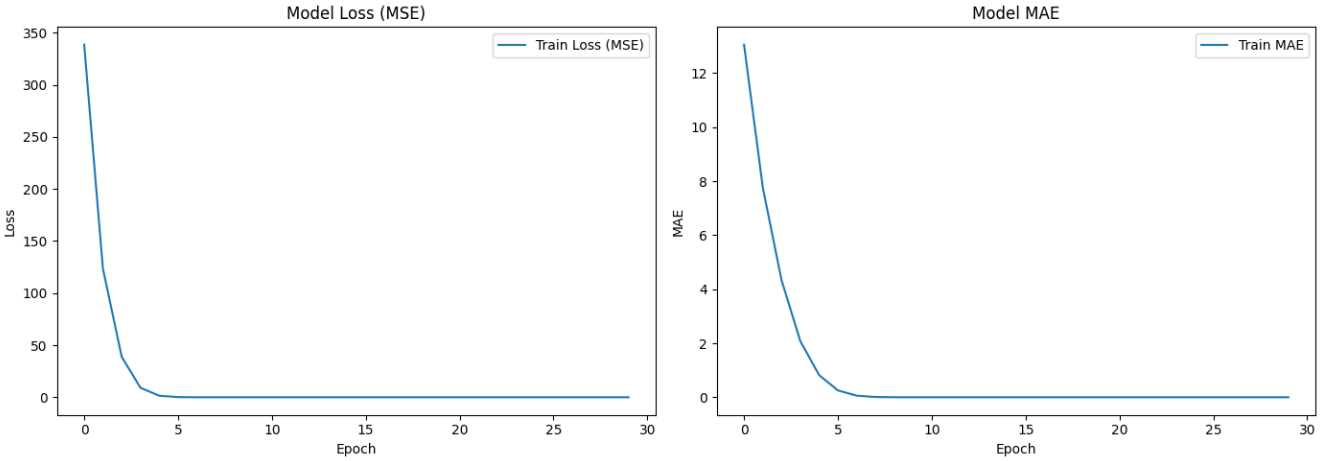


Figure 4.11: Loss and MAE in training.

- **Interpretation of the Loss and MAE Graphs:** The graphs above show the evolution of loss (Loss) and mean absolute error (MAE) over the epochs during the deep learning model training (as shown in Figure 4.11).
  - **Loss Graph (left):** The loss, measured in MSE (Mean Squared Error), decreases rapidly during the first few epochs, indicating that the model learns efficiently at the beginning of the training. After about 4 epochs, the loss stabilizes at a value close to zero, suggesting a successful convergence of the model.
  - **MAE Graph (right):** The mean absolute error (MAE) follows a similar trend to the loss, with a rapid decrease at the start of the training. The MAE reaches a very low value after about 5 epochs and stabilizes, indicating a continuous improvement in the model's prediction accuracy.

  The two graphs demonstrate that the model is training effectively, with rapid reductions in both loss and mean absolute error at the beginning of the training. The stabilization of the curves after about 5 epochs indicates that the model has converged and is well-fitted. This suggests that the model is ready for further testing or practical use, with no evident signs of overfitting.

	KNN Model		LSTM Model		RNN Model	
	Scenario1	Scenario2	Scenario1	Scenario2	Scenario1	Scenario2
MAE Error	$1.95*10^{-5}$	$7.78*10^{-6}$	$1.09*10^{-4}$	$3.76*10^{-5}$	$7.56*10^{-5}$	$5.66*10^{-5}$
MSE Error	$9.1*10^{-10}$	$3.64*10^{-10}$	$1.73*10^{-8}$	$2.21*10^{-9}$	$7.63*10^{-9}$	$4.36*10^{-9}$
RMSE	$3.02*10^{-5}$	$1.9*10^{-5}$	$1.31*10^{-4}$	$4.7*10^{-5}$	8.7310^{-5}	$6.6 * 10^{-5}$

Figure 4.12: Model comparison table.

The table (Figure 4.12) compares the performance of three models (KNN, LSTM, and RNN) under two different scenarios (mobile case and mobile+fixed case) using three

error metrics: MAE, MSE, and RMSE. The KNN model shows the lowest error values in both scenarios, indicating better overall performance compared to the LSTM and RNN models. The KNN model with K=3 proves to be the best-performing model. Therefore, we conclude that KNN model could be the most appropriate choice for our system. Here are the results found for predicting location information from cellular data in our dataset, as illustrated in Figures 4.13 and 4.14:

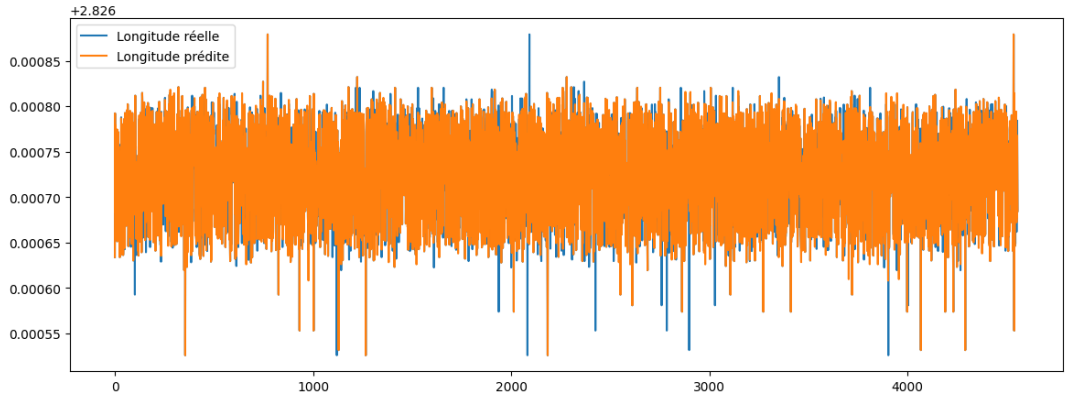


Figure 4.13: Model for prediction longitude.

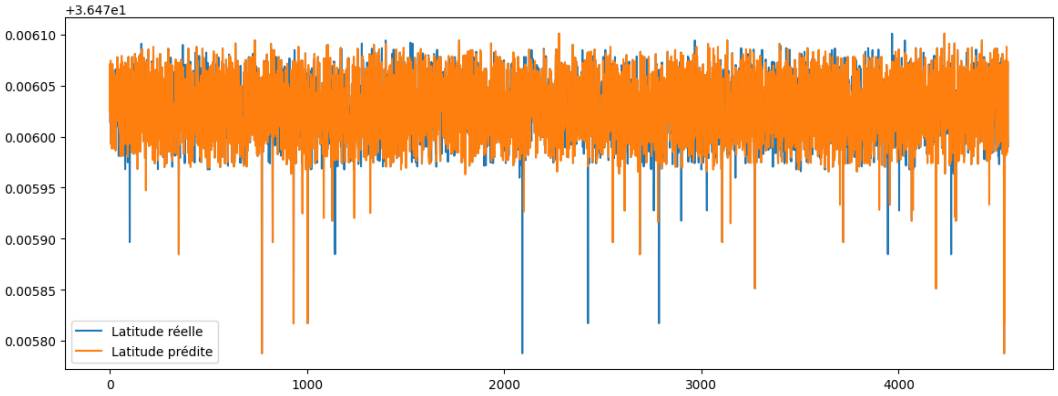


Figure 4.14: Model for prediction latitude.

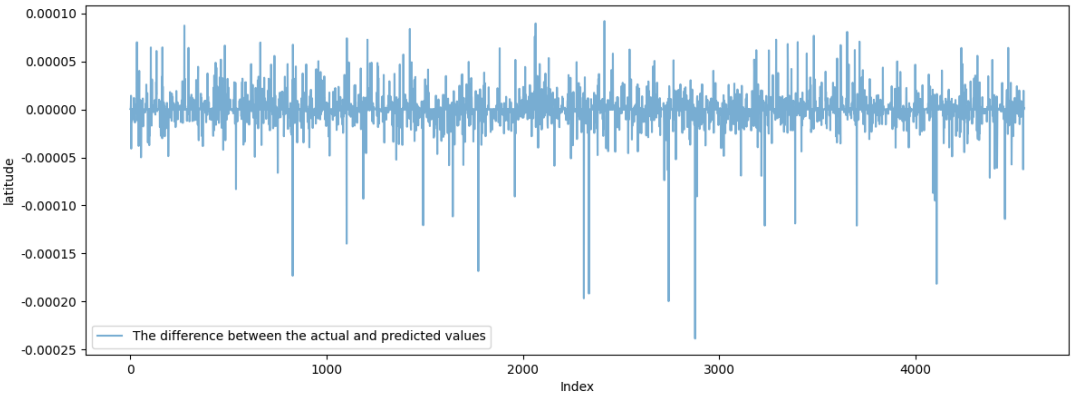


Figure 4.15: Difference between real and predicted latitude.

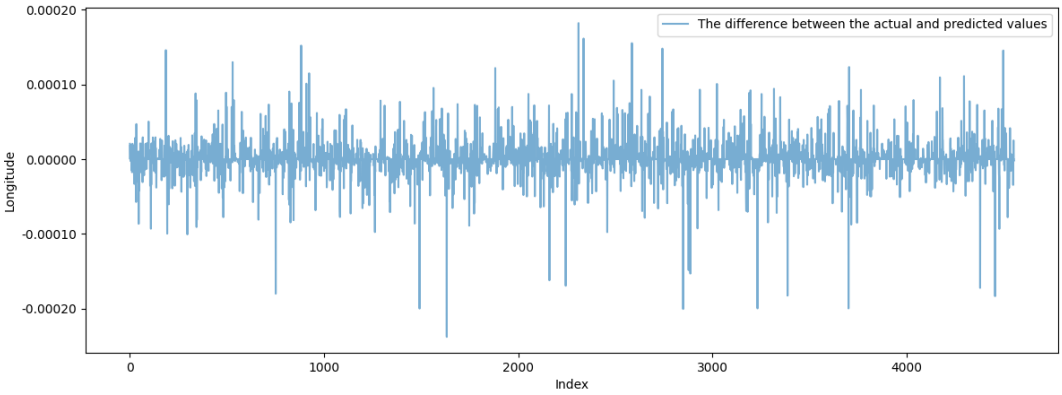


Figure 4.16: Difference between for prediction longitude.

According to the graphs (Figure 4.15 ,Figure 4.16), we notice that the difference between the actual and predicted values of longitude and latitude ranges from 0.00014 to -0.00024 degrees. This indicates that the error in location estimation is between 10 meters and 22 meters.

4.5 Conclusion :

This chapter detailed the construction of our dataset and the implementation of various location prediction models. We explained the choice of tools for data collection and preparation. Then, we compared several models in two scenarios: using only mobile phone data and combining mobile and fixed phone data. This analysis allowed us to evaluate the performance of the models and determine best model for our proprietary location system.

General Conclusion

Location-based services (LBS) are essential in many areas, such as navigation, logistics, social networking, and emergency response. While GPS is widely used, its limitations, such as high energy consumption and vulnerability in urban or indoor environments, highlight the need for alternative solutions that are more efficient, accurate, and energy-saving.

This project presents a proprietary localization system that uses cell phone towers to provide an accurate and efficient solution. We analyzed various cell-based localization systems, comparing their advantages and disadvantages in terms of accuracy and adaptability to environmental changes. Our system specifically addresses these challenges by utilizing cell tower information for indoor localization, offering a more reliable alternative to GPS in complex environments.

Additionally, our approach incorporates advanced machine learning and deep learning techniques, such as K-Nearest Neighbors (KNN) for regression tasks, and Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks to refine location prediction accuracy. Through extensive testing and comparisons, we were able to determine the optimal model, ensuring high accuracy in predicting device locations.

In conclusion, our study demonstrates that cell-based localization systems offer a promising alternative to GPS, especially in complex environments. By combining cellular signals with advanced predictive models, our system provides a reliable and energy-efficient approach to location tracking. This work illustrates how the evolution of LBS can lead to more sustainable and flexible solutions to meet the growing needs of modern applications. Furthermore, the experiences and challenges encountered during the implementation of this system have enriched our understanding and strengthened its robustness, paving the way for future advancements in the field of location-based services.

Perspectives

In order to improve the accuracy and efficiency of the system, various development perspectives can be explored, such as:

- Develop a system capable of estimating positions in real time with high accuracy.

- Add several fixed phones that remain stationary: By integrating more fixed phones into our architecture, we could collect additional data from stationary reference points. This would enrich our dataset, improve the accuracy of location predictions and reduce errors in challenging environments.
- Ensure that the system covers larger geographical areas and remains resistant to failures.

Bibliography

[1] Obeidat, H., Shuaieb, W., Obeidat, O., Abd-Alhameed, R. (2021). A review of indoor localization techniques and wireless technologies. Wireless Personal Communications, 119, 289-327.

[2] When You Ask "Where?" Ultra-Wideband Answers https://www.qorvo.com/design-hub/blog/when-you-ask-where-ultra-wideband-answers

[3] Basic radio frequency identification (RFID) system. https://www.researchgate.net/figure/Basic-radio-frequency-identification-RFID-system-Source-own-elaboration fig2_350478741

[4] Asaad, S. M., Maghdid, H. S. (2022). A comprehensive review of indoor/outdoor localization solutions in IoT era: Research challenges and future perspectives. Computer Networks, 212, 109041.

[5] Rizk, H., Youssef, M. (2019, November). Monodcell: A ubiquitous and low-overhead deep learning-based indoor localization with limited cellular information. In Proceedings of the 27th ACM SIGSPATIAL international conference on advances in geographic information systems (pp. 109-118).

[6] Rizk, H., Abbas, M., Youssef, M. (2020, March). Omnicells: Cross-device cellular-based indoor location tracking using deep neural networks. In 2020 IEEE International Conference on Pervasive Computing and Communications (PerCom) (pp. 1-10). IEEE.

[7] Rizk, H., Torki, M., Youssef, M. (2018). CellinDeep: Robust and accurate cellular-based indoor localization via deep learning. IEEE Sensors Journal, 19(6), 2305-2312.

[8] Elbakly, R., Youssef, M. (2019, April). Crescendo: An infrastructure-free ubiquitous cellular network-based localization system. In 2019 IEEE Wireless Communications and Networking Conference (WCNC) (pp. 1-6). IEEE.

[9] Dufková, K., Ficek, M., Kencl, L., Novák, J., Kouba, J., Gregor, I., Danihelka, J. (2008, September). Active GSM cell-id tracking: " Where Did You Disappear?". In Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments (pp. 7-12).

[10] artificial intelligence https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/artificial-intelligence-applications

[11] machine and deep learning https://www.analyticsvidhya.com/blog/2021/06/machine-learning-vs-artificial-intelligence-vs-deep-learning/

[12] machine learning https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained

[13] McCulloch, W. S., Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5, 115-133.

[14] neuron https://www.natural-solutions.eu/blog/histoire-du-deep-learning

[15] Deep learning https://www.ibm.com/topics/deep-learning

[16] Regression www.geeksforgeeks.org/regression-in-machine-learning/

[17] KNN https://medium.com/@nandiniverma78988/understanding-k-nearest-neighbors-knn-

[18] RNN https://intellipaat.com/blog/tutorial/artificial-intelligence-tutorial/recurrent-neural-network/

[19] Kumar, J., Goomer, R., Singh, A. K. (2018). Long short term memory recurrent neural network (LSTM-RNN) based workload forecasting model for cloud datacenters. Procedia computer science, 125, 676-682.

[20] LSTM architecture www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/

[21] Preprocessing https://lakefs.io/blog/data-preprocessing-in-machine-learning/

[22] MinMaxScaling https://rasbt.github.io/mlxtend/user_guide/preprocessing/minmax_scaling/

[23] Regression Metrics https://kobia.fr/regression-metrics-quelle-metrique-choisir/

[24] kaggle https://www.kaggle.com/discussions/general/328265

[25] Java https://aws.amazon.com/what-is/java/#

[26] Python https://www.python.org/doc/essays/blurb/

[27] Android Studio https://developer.android.com/studio/intro

[28] SQLite https://sqlite.org/

[29] Eclipse https://www.cogentinfo.com/resources/what-is-eclipse-ide-java-101#

[30] Numpy https://numpy.org/doc/stable/user/whatisnumpy.html

[31] Pandas https://mode.com/pythontutorial/libraries/pandas

[32] Matplotlib https://www.geeksforgeeks.org/python-introduction-matplotlib/

[33] keras https://keras.io/

[34] Sklearn https://domino.ai/data-science-dictionary/sklearn

[35] Tensorflow https://www.tensorflow.org/?hl=fr

[36] Sockets https://www.scaler.com/topics/computer-network/socket-programming