ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Desarrollo de un modelo de aprendizaje profundo para la estimación de coordenadas GPS, utilizando metadatos del proceso de comunicación basado en el protocolo LoRaWAN.

Proyecto de Titulación

Previo a la obtención del Título de:

Magíster en ciencias de la computación

Presentado por:

Nelson Vicente Vera Méndez

Guayaquil - Ecuador

6 de mayo de 2025

Reconocimiento

Este camino ha sido un desafío lleno de aprendizajes, sacrificios y momentos inolvidables. No lo recorrí solo, y por ello, quiero expresar mi más profunda gratitud a quienes estuvieron a mi lado en cada paso.

A la Dra. Rebeca Estrada, por su orientación acertada y por ayudarme a convertir cada obstáculo en una oportunidad de aprendizaje. Su confianza en mí se reflejó en las responsabilidades y retos que me encomendó, los cuales hoy me definen y me permitieron descubrir mi potencial. Su orientación ha sido clave en este logro. Gracias por guiarme en el camino de la investigación y por brindarme oportunidades que marcaron mi crecimiento profesional.

Agradezco también a mis compañeros de investigación y colaboradores por su apoyo, dedicación y por las horas de esfuerzo compartido. Las discusiones que sostuvimos impulsaron nuevas ideas, y el espíritu de equipo convirtió cada desafío en una oportunidad de aprendizaje. Sin su compromiso y colaboración, este proyecto no habría sido posible.

Extiendo mi más sincero agradecimiento a la Escuela Superior Politécnica del Litoral (ESPOL) por brindarme la oportunidad de representar a esta prestigiosa institución y por proporcionarme los recursos necesarios para llevar a cabo la estancia de investigación de la cual surge este trabajo. Asimismo, expreso mi gratitud a la Universitat Politècnica de València (UPV) y al Departamento de Informática de Sistemas y Computadores (DISCA), en calidad de centro de investigación de acogida, por su colaboración y la cesión de espacios durante mi estancia.

A mi familia, mi pilar fundamental, gracias por su amor incondicional y apoyo —siempre presente— en cada etapa de mi vida. Sé que este logro es solo el comienzo de una nueva etapa y un escalón más para cumplir mis sueños. Este triunfo es tan suyo como mío; cada uno de mis éxitos existe gracias a ustedes y está dedicado a ustedes. Los llevo conmigo en todo lo que hago.

A esa persona especial que conocí durante esta etapa de mi vida, por su paciencia, comprensión y apoyo. Su compañía durante el desarrollo de este proyecto ha sido parte esencial de este logro.

A todos los que, de una u otra manera, dejaron su huella en este proyecto, les expreso mi más sincero agradecimiento. Este logro refleja el esfuerzo compartido y el apoyo inquebrantable que me acompañó en cada paso.

Evaluadores

Firmado electrónicamente por:
MIGUEL ANDRES REALPE ROBALINO
Validar únicamente con FirmaEC

Miguel Andrés Realpe Robalino

Tutor de proyecto

Firmado electrónicamente por:
REBECA LEONOR ESTRADA PICO
Validar únicamente con FirmaEC

Rebeca Leonor Estrada Pico

Revisor de proyecto

Firmado electrónicamente por:
DANIEL ERICK OCHOA DONOSO
Validar únicamente con FirmaEC

Daniel Erick Ochoa Donoso

Coordinador del programa

Declaración expresa

Yo Nelson Vicente Vera Méndez acuerdo y reconozco que:

La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor o autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por mí/nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que me/nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de mi/nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique al/los autor/es que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 05 de Mayo del 2025

Firmado electrónicamente por:
NELSON VICENTE VERA MENDEZ
Validar únicamente con FirmaEC

Nelson Vicente Vera Méndez

Resumen

Esta investigación aborda el desafío de la localización de dispositivos en espacios abiertos que se comunican de manera inalámbrica a través de redes LoRaWAN. Tradicionalmente, se emplea la tecnología GPS para la geolocalización; sin embargo, su uso implica la adición de un recurso adicional, lo cual puede elevar los costes. Por ello, se propone aprovechar la propia señal de comunicación de los dispositivos para fines de geolocalización, lo que reduce gastos y, además, ofrece versatilidad al ser una solución independiente de plataformas externas.

Se desarrolló un modelo de aprendizaje profundo basado en Redes Neuronales de Grafos (GNN) para estimar coordenadas GPS, utilizando metadatos generados durante la comunicación LoRaWAN —como RSSI e información proporcionada por los gateways—. La propuesta presenta una metodología que combina métodos basados en datos—como técnicas de aprendizaje automático—con métodos teóricos fundamentados en los principios de propagación de ondas. Además, integra un enfoque mixto que reúne ambos paradigmas, aprovechando los principios físicos y las técnicas de aprendizaje automático para capturar y modelar el ruido. Esto permite equilibrar y ampliar los conjuntos de datos de manera eficiente, seleccionando y adaptando el método más adecuado para cada gateway.

El estudio emplea algoritmos de clusterización como Clustering Difuso o Fuzzy C-Means (FCM), Clustering Sustractivo y Clustering Subjetivo para analizar y estructurar los datos. Los resultados destacan la efectividad del método híbrido, especialmente en gateways con mayor cantidad de datos disponibles para el desarrollo de la metodología propuesta, logrando un equilibrio entre precisión y eficiencia.

Además de ser una solución para la estimación de coordenadas GPS en entornos con cobertura LoRaWAN, esta investigación ofrece un marco para optimizar la caracterización de la propagación de señales en escenarios de exteriores. El enfoque planteado aborda de manera eficiente las limitaciones inherentes a datos escasos y desbalanceados. Además, ofrece soluciones escalables para aplicaciones futuras que demanden una caracterización avanzada de la propagación de señales, incluyendo la geolocalización inteligente como una de sus principales aplicaciones, así como la reducción de interferencias, la optimización de la calidad del servicio y el diseño eficiente de redes inalámbricas.

Tabla de contenido

Índice de figuras

Índice de Tablas

Lista de Algoritmos

1

Introducción

El crecimiento acelerado del internet de las cosas ("Internet of Things", IoT) ha impulsado el despliegue masivo de dispositivos en diversas áreas de la tecnología de la información, desde aplicaciones en entornos rurales, como la agricultura inteligente, hasta entornos urbanos, como las ciudades inteligentes [1].Sin embargo, estos dispositivos, enfrentan un desafío crucial: la necesidad de una solución de posicionamiento eficiente y económica, ya que, debido a sus limitadas capacidades de energía y cómputo, no suelen estar equipados con módulos de geolocalización como el "Global Positioning System" (GPS), lo que hace inviable o costoso su uso y compromete la obtención de su ubicación exacta. Esta carencia afecta la eficacia de aplicaciones que dependen de soluciones tecnológicas, como el monitoreo ambiental, la gestión de recursos en tiempo real, la cadena de suministro y la logística, donde la geolocalización es esencial para su correcto funcionamiento [2].

Para resolver este problema se realiza el desarrollo de una solución que utilice la metadata generada durante la comunicación de dispositivos inteligentes en redes de baja potencia y área amplia ("Low Power Wide Area Network", LPWAN), específicamente con la modulación de largo alcance ("Long Range", LoRa), para estimar la posición de estos dispositivos sin necesidad de un módulo de geolocalización físico [3, 4]. Este planteamiento permitirá dotar de capacidad de geolocalización mediante coordenadas de latitud y longitud estimadas a cualquier dispositivo inteligente que utilice estas tecnologías de largo alcance, reduciendo así los costos y el consumo de recursos.

Aprovechando la continua expansión de redes LPWAN que utilizan el protocolo "Long Range Wide Area Network" (LoRaWAN) y la creciente implementación de la modulación LoRa en dispositivos IoT para la comunicación a larga distancia [5], una de las tendencias más recientes en la industria es la geolocalización sin módulos GPS, conocida como 'Smart Geolocation for IoT'. Este enfoque permite estimar la posición utilizando modelos de aprendizaje profundo que emplean la metadata generada durante la comunicación en redes LoRaWAN, en particular los valores del indicador de potencia de señal recibida ("Received Signal Strength Indicator", RSSI), relación señal/ruido ("Signal-to-Noise Ratio", SNR) y la información de los gateways receptores [6].

Empresas como 1663 Solutions[1], Trackpac[2] y Semtech[3] ya están desarrollando soluciones basadas en estos principios, mientras que fabricantes como Semtech están produciendo dispositivos con capacidades avanzadas para el posicionamiento inteligente. Sin embargo, la mayoría de las soluciones de geolocalización inteligente no son de código abierto, a diferencia

[1] https://www.1663solutions.com
[2] https://www.trackpac.com
[3] https://www.semtech.com/

de LoRaWAN, cuyas especificaciones y plataformas como The Things Network (TTN) son de acceso público [7]. Aunque existen soluciones de código abierto [8], su precisión es limitada . La propuesta presentada en esta tesis aborda esa limitación mediante el desarrollo de un modelo de aprendizaje profundo, más específicamente una red neuronal de grafos ("Graph Neural Network", GNN) especializada para un entorno con cobertura de red LoRaWAN, que estima coordenadas de geolocalización de dispositivos en la red utilizando metadatos generados durante su comunicación.

Adicionalmente, esta tesis propone una metodología integral para incrementar y equilibrar conjuntos de datos generados a partir de la metadata del proceso de comunicación en redes LoRaWAN. 1.1

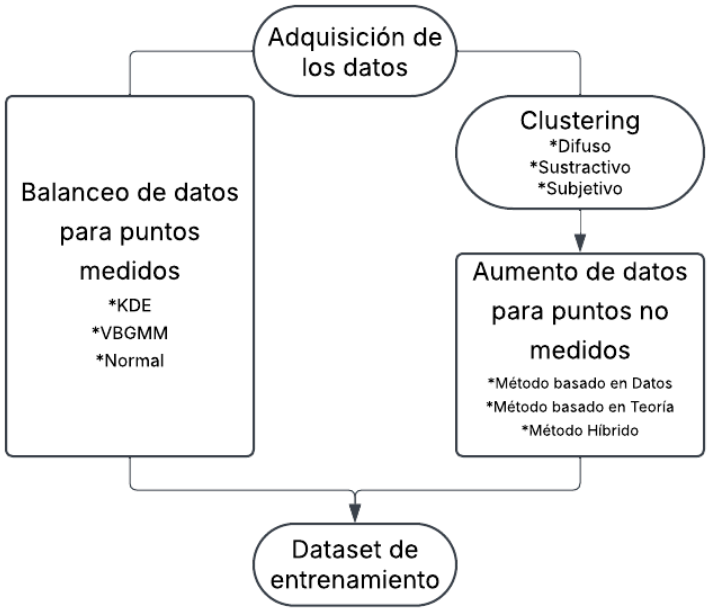


Figura 1.1: Arquitectura LoRaWAN en la UPV

Esta metodología incorpora los siguientes componentes:

1.- **Balanceo de data para puntos medidos empíricamente**: Se emplean modelos que se ajustan a la distribución subyacente de los datos recopilados empíricamente por punto. Estos incluyen:

- **Estimación de Densidad mediante Kernel (KDE)**: Un método para ajustar un modelo estadístico no paramétrico para estimar la densidad de probabilidad de los datos.
- **Modelos de Mezcla Gaussiana Bayesianos Variante (VBGMM)**: Modelos estadísticos que combinan varios componentes gaussianos para representar patrones complejos.
- **Distribución Normal Ajustada**: Ajusta una distribución normal para representar los datos de manera eficiente.

2.- **Aumento de data para puntos no medidos**: Se emplean modelos para la predicción de datos en nuevos puntos establecidos:

2.1.- **Método "Basado en Data"**: Modelos que utilizan técnicas de aprendizaje supervisado, como modelos de regresión, para aprender patrones subyacentes y generar muestras adicionales.

2.2.- **Método "Basado en Teoría"**:Implementa modelos basados en principios teóricos para generar datos, como el modelo de propagación de intercepto flotante, basado en la propagación de ondas.

2.3.- **Método "Mixto"**: Combina los anteriores enfoques, complementando el modelo de enfoque teórico con modelos "basados en data" para capturar errores no modelados.

Se emplearon algoritmos de clustering, incluyendo Fuzzy C-Means (FCM) y Subtractive Clustering, para analizar y estructurar el conjunto de datos previo a la implementación de los componentes mencionados anteriormente. Se utilizo la distancia de Wasserstein como métrica para evaluar los modelos propuestos.

En la comunicación inalámbrica, particularmente con la propagación de señales LoRa, la recolección de mediciones empíricas en cada ubicación es a menudo impráctica debido a las restricciones de tiempo y distancia. Caracterizar la propagación de señales LoRa es crucial para optimizar sistemas de comunicación inalámbrica, especialmente en entornos diversos como rurales, vegetados y suburbanos, donde la densidad y el tipo de obstáculos varían significativamente. Este desafío enfatiza la necesidad de metodologías robustas que permitan realizar un aumento de datos que aprovechen mediciones empíricas limitadas para generar datos sintéticos útiles mediante predicciones de modelos en áreas no medidas. Nuestra metodología facilita el desarrollo de soluciones prácticas, incluyendo la planificación del despliegue de redes, sistemas de sensores inteligentes optimizados, asignación inteligente de recursos y implementaciones más precisas de sistemas de posicionamiento y seguimiento.

En este estudio, se utilizó una arquitectura de LoRaWAN desplegada en la Universitat Politècnica de València (UPV) para habilitar un proceso de comunicación del cual se recopiló metadata destinada a caracterizar el entorno. La tecnología GPS se integró en el proceso de comunicación, generando mediciones que sirvieron como carga útil de las transmisiones y que nos servirán para poder entrenar nuestro modelo de posicionamiento. Los datos GPS se recopilaron en diversos puntos dentro de la UPV y, mediante integraciones disponibles en la plataforma de The Things Network, se extrajeron conjuntos de datos para desarrollar nuestra solución. No obstante, debido a las limitaciones inherentes a la recolección de datos GPS, como los retrasos en la fijación de señal, cada medición puede estar sujeta a latencias, lo que resulta en conjuntos de datos desequilibrados. Esto proporciona un escenario óptimo para probar enfoques dirigidos a mejorar el equilibrio de dichos conjuntos de datos en este trabajo.

2

Trabajos Relacionados

Esta sección presenta un breve resumen de las propuestas previas más relevantes relacionadas con la predicción de coordenadas GPS en redes LoRaWAN, el uso de Redes Neuronales basadas en Grafos (GNN), enfoques de aumento de datos, y la caracterización de la propagación de señales inalámbricas en entornos mixtos. Estos trabajos proporcionan el contexto necesario para las contribuciones de esta tesis, identificando avances existentes y áreas que requieren mayor investigación.

Diversos estudios han explorado metodologías para predecir coordenadas GPS en redes LoRaWAN. Por ejemplo, Moradbeikie et al. [9] propusieron un enfoque basado en trilateración utilizando la intensidad de la señal (RSSI) para estimar posiciones en entornos industriales. Aunque eficiente, este método mostró alta sensibilidad a la variabilidad del RSSI, especialmente en entornos urbanos, reportando errores promedio de hasta 50 metros. Liu et al. [10] utilizaron técnicas de aprendizaje profundo para mejorar la precisión del GPS en cañones urbanos, logrando un error promedio de 10 metros. Sin embargo, este enfoque no considera la estructura dinámica de las redes LoRaWAN, limitando su capacidad para manejar cambios complejos en la configuración de la red. A diferencia de estos trabajos, esta tesis propone el uso de modelos basados en GNN, que pueden capturar relaciones dinámicas y estructurales en redes LoRaWAN, reduciendo la sensibilidad a la variabilidad del RSSI y mejorando la precisión en escenarios mixtos.

Las redes neuronales basadas en grafos (GNN) ofrecen una solución natural para modelar relaciones estructurales en redes como LoRaWAN. Aunque las GNN se han aplicado ampliamente en áreas como redes sociales y biología computacional, su uso para la estimación de coordenadas GPS en LoRaWAN es escaso. Zhang et al. [11] demostraron que las redes neuronales gráficas (GNN) son capaces de modelar topologías dinámicas en sistemas de comunicación inalámbrica, destacando su eficacia en el manejo de datos no euclideanos. Inspirada por estas capacidades, esta tesis propone integrar las GNN para modelar las topologías dinámicas inherentes a las redes LoRaWAN, mejorando así la adaptabilidad y eficiencia de la solución en entornos altamente dinámicos.

El aumento de datos es una técnica ampliamente utilizada para mejorar el rendimiento de modelos de aprendizaje supervisado y no supervisado. Técnicas como SMOTE han sido empleadas para balancear clases minoritarias [12], mientras que las redes generativas adversariales (GANs) han permitido generar datos sintéticos que enriquecen los conjuntos de entrenamiento [13].En redes inalámbricas como LoRaWAN, las soluciones tradicionales se han centrado principalmente en el ámbito de la seguridad, mediante la detección de anomalías y de intrusos para proteger la red, logrando resultados prometedores. No obstante, estas técnicas no se han enfocado en la caracterización de las redes, como el modelamiento de la propagación de señales. En este trabajo, se propone implementar un modelo optimizado para predecir metadatos en redes LoRaWAN,

específicamente los valores de RSSI, considerados una métrica clave en la propagación de señales. La metodología se basa en un análisis comparativo entre modelos basados en datos, modelos teóricos y enfoques híbridos, con el objetivo de identificar la solución más eficaz y adecuada. Este enfoque mejora significativamente la calidad del conjunto de datos utilizado para entrenar modelos basados en Redes Neuronales de Grafos (GNN).

En términos de aprendizaje profundo, [10] implementa redes neuronales convolucionales para mejorar la precisión del GPS en entornos urbanos, mientras que otros estudios combinan enfoques supervisados y no supervisados para abordar la complejidad de los datos en IoT. Por ejemplo, Por ejemplo, Garcia et al. [14] aborda sistemas de aprendizaje móvil y ubicuo sensibles al contexto, destacando cómo estas técnicas mejoran la adaptabilidad en entornos dinámicos. De manera similar, López et al. [15] optimiza métodos de agrupamiento no supervisado aplicados al reconocimiento de patrones, lo cual es crucial para la clasificación y el análisis de datos heterogéneos en redes IoT. Además, Arco et al. [16] propone una metodología basada en características locales y globales para datos de medición inteligente, aplicando técnicas de agrupamiento no supervisado para capturar la variabilidad inherente de los datos generados en redes IoT. Estos enfoques destacan la importancia de integrar modelos híbridos para mejorar la adaptabilidad y precisión en aplicaciones complejas como el posicionamiento en redes LoRaWAN, sin embargo, no explotan la capacidad de las GNN para modelar relaciones estructurales en redes dinámicas.

En resumen, esta tesis propone un enfoque híbrido que combina modelos teóricos y basados en datos para mejorar la precisión de soluciones de aprendizaje profundo mediante el aumento de los datos de entrenamiento, específicamente en la inferencia de coordenadas GPS en entornos mixtos. Mediante el uso de GNNs, se aprovecha la topología de las redes LoRaWAN para modelar estructuras con diferentes números de gateways y conexiones, proporcionando mayor flexibilidad y precisión en las predicciones. Las aplicaciones de este trabajo se extienden a la localización de dispositivos IoT en escenarios desafiantes, como campus universitarios o áreas industriales, donde la combinación de edificios y espacios abiertos plantea retos significativos para los modelos tradicionales.

3

Metodología

Este estudio esta fundamentado en la recopilación, análisis/aumento y uso de una base de datos elaborada por el grupo de investigación Smart Network Technologies (TRI) de la Escuela Superior Politécnica del Litoral (ESPOL), en colaboración con el Departamento de Informática de Sistemas y Computadores (DISCA) de la Universidad Politécnica de Valencia (UPV). La base de datos fue generada a partir de la infraestructura LoRaWAN desplegada en el campus de la UPV, la cual está integrada con la plataforma The Things Network (TTN). Esta infraestructura incluye la configuración y despliegue de múltiples gateways en ubicaciones estratégicas, optimizados para proporcionar cobertura a lo largo del campus y gestionar la transmisión y recepción de mensajes provenientes de los nodos de la red. La infraestructura LoRaWAN consta de varios componentes clave: nodos sensores, gateways, un servidor de red y un servidor de aplicaciones. Los nodos sensores y los gateways implementan la tecnología LoRa (Long Range), un protocolo de comunicación inalámbrica diseñado para redes de baja potencia y largo alcance. LoRa utiliza una técnica de modulación de espectro ensanchado basada en "chirps", una señal cuya frecuencia cambia gradualmente con el tiempo. Este protocolo opera a bajas frecuencias para garantizar una alta tolerancia al ruido y codifica información mediante el factor de expansión ajustable (spreading factor). Esta configuración permite alcanzar rangos de transmisión de hasta 1 km en entornos urbanos y distancias significativamente mayores en áreas despejadas. Los datos recopilados comprenden mensajes transmitidos por los nodos junto con los metadatos asociados, generada durante los procesos de comunicación en redes LoRa con el protocolo LoRaWAN. Para garantizar consultas a estos datos, el protocolo MQTT fue configurado como una integración en la plataforma TTN, facilitando la extracción directa de los metadatos desde su interfaz de usuario.

3.1 Creación del Dataset

El desarrollo de esta investigación requirió la recopilación de un conjunto de datos diverso y representativo en el campus de la Universidad Politécnica de Valencia (UPV). El dataset, esta estructurado cuidadosamente para reflejar las condiciones ambientales heterogéneas del entorno de estudio. Además, los datos están segmentados por gateway receptor, identificando variaciones en la propagación de señales debido a la cobertura específica de cada dispositivo. Este conjunto de datos constituye la base para evaluar y optimizar los métodos propuestos, destacando la importancia de abordar desafíos como el desbalance y la falta de datos en ciertas ubicaciones.

3.1.1 Adquisición de datos

El nodo fue ubicado de una manera sistemática en una distribución tipo malla cubriendo 15 ubicaciones diferentes a lo largo del campus de la UPV. Cada punto de medición fue seleccionado estratégicamente para representar diversas condiciones ambientales y garantizar una cobertura completa del área de estudio.

Tecnologías y Plataformas Empleadas

El campus de la UPV cuenta con tres gateways, dos del tipo RAK7248 ("main-gtw-grc" y "rak7248-grc") 3.1(a), diseñados para uso en interiores, y uno del tipo RAK7289 ("itaca-upv-022") 3.1(b), adecuado para exteriores. Estos gateways, ubicados estratégicamente en los límites del campus, recibieron los mensajes transmitidos por los nodos Heltec LoRa WiFi V3 3.1(c), basados en el microcontrolador ESP32, que opera bajo el plan de frecuencia europeo de 868 MHz. Los nodos estaban equipados con un módulo GPS NEO6M configurado mediante el protocolo de comunicación UART, que capturaba coordenadas geográficas y las transmitía como carga útil mediante modulación LoRa 3.1(d).

(a) Gateway de interiores RAK7248 (b) Gateway de exteriores RAK7289 (c) Modulo Wifi LoRa v3 (d) Modulo GPS NEO6M

Figura 3.1: Componentes de la arquitectura para la recolección de datos LoRaWAN.

Las coordenadas capturadas originalmente estaban en formato NMEA (National Marine Electronics Association), un estándar para datos de GPS. Estas coordenadas, por ejemplo:

`$GPGGA,110617.00,41XX.XXXXX,N,00831.54761,W,1,05,2.68,129.0,M,50.1,M,,*42`

Contiene información codificada en un formato estándar para datos de GPS. Cada campo tiene un significado específico detallado a continuación:

- `$GPGGA`: Indica el tipo de mensaje NMEA. En este caso, GGA representa un mensaje de *Global Positioning System Fix Data*, que proporciona datos básicos de posición, como latitud, longitud, altura y calidad de la señal.
- `110617.00`: Es la hora UTC en formato HHMMSS.ss. Aquí, `11:06:17.00` indica que los datos fueron capturados a las 11 horas, 6 minutos y 17 segundos UTC.
- `41XX.XXXXX`: Representa la latitud en grados y minutos. El valor `41XX.XXXXX` es desglosado en 41 grados y una fracción de minutos (`XX.XXXXX`).
- `N`: Indica el hemisferio. Para este caso, `N` denota que la latitud se encuentra en el hemisferio norte.

- `00831.54761`: Representa la longitud en grados y minutos. El valor `00831.54761` es desglosado en 8 grados y 31.54761 minutos.
- `W`: Indica el hemisferio. Aquí, `W` denota que la longitud se encuentra en el hemisferio oeste.
- `1`: Es el indicador de calidad de la señal GPS. Un valor de `1` significa que se obtuvo una solución de posición válida (*fix*).
- `05`: Indica el número de satélites utilizados para calcular la posición. En el ejemplo 5 satélites son utilizados.
- `2.68`: Representa la dilución de precisión horizontal (*HDOP, Horizontal Dilution of Precision*). Este valor mide la calidad de la señal; valores más bajos indican mayor precisión.
- `129.0`: Es la altitud sobre el nivel del mar, expresada en metros. Aquí, `129.0` metros.
- `M`: Indica que la unidad de medida para la altitud es metros.
- `50.1`: Representa la altura del geoide (separación entre el geoide y el nivel del mar) en metros.
- `M`: Indica que la unidad de medida para la altura del geoide es metros.
- `,`: Este campo normalmente es dejado vacío, pero es utilizado para datos de corrección DGPS (*Differential GPS*), que no están presentes en este mensaje.
- `*42`: Es el valor de comprobación (*checksum*), que permite verificar la integridad del mensaje NMEA. Es calculado como un XOR de todos los caracteres entre $ y *.

El formato codificado contiene todos los datos esenciales para determinar la posición geográfica y otras características del *fix* obtenido, que luego son procesados y transmitidos a través de la red LoRaWAN.

Las coordenadas son procesadas mediante una librería en el dispositivo embebido para extraer las coordenadas de latitud y longitud del nodo. Posteriormente, las coordenadas son codificadas y transmitidas utilizando la red LoRaWAN. En el lado receptor, los gateways decodificaron los datos transmitidos en formato Base64 y los reenviaron al servidor de red, implementado a través de la plataforma gratuita The Things Network (TTN).

La plataforma TTN permite el acceso a la información recibida mediante varias integraciones, se optó por utilizar MQTT. Un cliente MQTT, implementado en Python, se suscribe al tópico correspondiente, y es configurado para recibir los datos captados por los gateways en formato JSON. Por ejemplo, el tópico puede tener el siguiente formato:

```
v3/rssi-measurements-/devices/eui-*********/up
```

En este contexto, un tópico es una dirección jerárquica que organiza y categoriza los mensajes, permitiendo al cliente recibir únicamente los datos relevantes. Estos datos incluyen las coordenadas GPS de los nodos transmisores como carga útil y los metadatos del envío, que contiene parámetros fundamentales como el indicador de potencia de señal recibida (RSSI), la Relación señal-ruido (SNR), la identificación de los gateways receptores y las marcas temporales de las transmisiones generadas con cada mensaje.

Escenario de estudio

En total, se obtuvieron 2,335 puntos de datos, distribuidos entre las 15 ubicaciones correspondiente a recepciones de los gateways presentes en el entorno como se muestra en la Fig.3.2.

Dado que múltiples gateways recibieron los mensajes transmitidos por la naturaleza de LoRaWAN, los datos recopilados

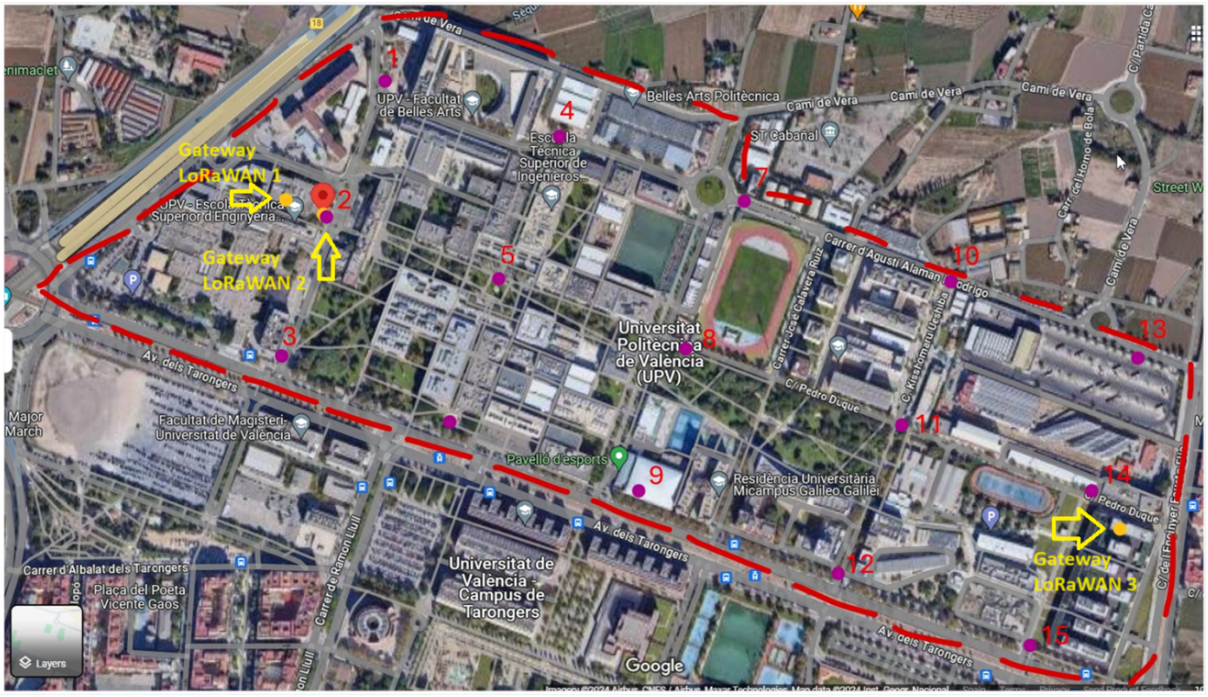
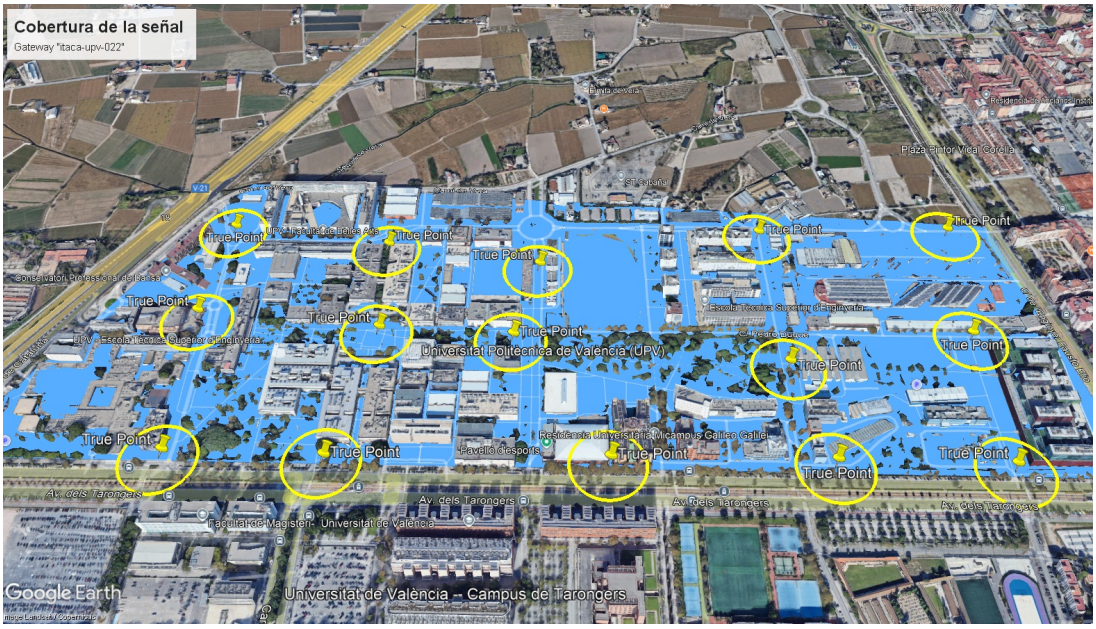


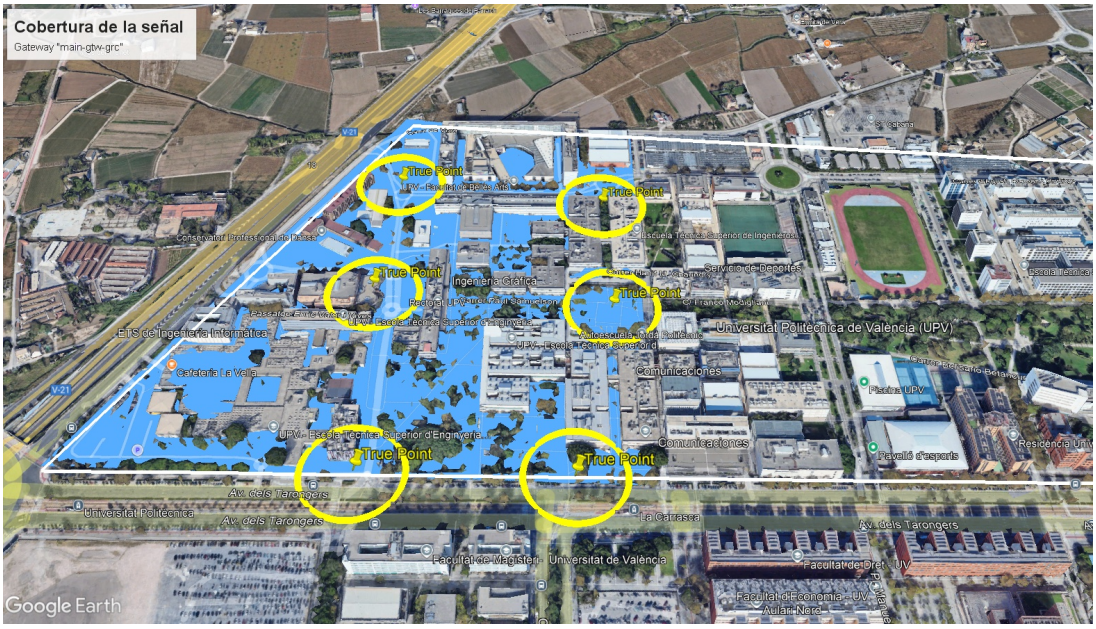
Figura 3.2: Arquitectura LoRaWAN en la UPV

fueron segmentados según el gateway receptor para garantizar coherencia en el análisis. Por ejemplo, los datos capturados por los gateways "itaca-upv-022" y "main-gtw-grc" fueron analizados de forma independiente, lo que permitió identificar variaciones en la propagación de la señal en función de la ubicación y cobertura específica de cada gateway. Adicionalmente, al segmentar los datos se observó que para el gateway "main-gtw-grc" no existía cobertura suficiente en algunos puntos de la red LoRaWAN, dejando sin datos ciertos de los 15 puntos donde se tomaron las mediciones. La cobertura de ambos gateways se puede apreciar en las imágenes 3.3(a)3.3(b). Esto destaca la importancia de considerar la cobertura variable de cada gateway. La estructura general del dataset y un ejemplo de los datos recopilados se presentan en la Tabla 3.1.

Adicionalmente, para los propósitos del estudio, se modificaron los conjuntos de datos por gateway seleccionando variables específicas, las cuales podrían generarse utilizando los métodos propuestos. Los conjuntos de datos organizados por gateway, estructurados ahora como se muestra en la Tabla 3.2, se encontraron desequilibrados, con un número variable de mediciones en ciertos puntos, lo que se presentará con mayor detalle en el siguiente capítulo. Esto conlleva una representación desigual de diferentes ubicaciones, destacando la necesidad de aplicar técnicas de aumento de datos.



(a) Visualización de la cobertura de la red LoRaWAN para el gateway 'itaca-upv-022'.



(b) Visualización de la cobertura de la red LoRaWAN para el gateway 'main-gtw-grc'.

Figura 3.3: Coberturas de los gateways LoRaWAN en la UPV.

Tabla 3.1: Formato de datos recopilada original

folder_name	coord_gps	gateway_id	timestamp	rssi	toa	latitude	longitude	altitude
Medición punto 1	39.48411560, -0.34499866	itaca-upv-022	3327095719	-100	0.045	39.48411560	-0.34499866	30.5
Medición punto 1	39.48415756, -0.34522617	main-gtw-grc	3464207824	-101	0.046	39.48415756	-0.34522617	30.5
Medición punto 2	39.48418808, -0.34534454	rak7248-grc-pm65	3574094099	-97	0.050	39.48418808	-0.34534454	30.5
Medición punto 3	39.48415375, -0.34535646	itaca-upv-022	3589100156	-101	0.048	39.48415375	-0.34535646	30.5

Tabla 3.2: Formato de datos recopilados por el gateway

folder_name	coord_gps	latitud	longitud	altitude	rssi
Medición punto 1	39.48411560,-0.34499866	39.48411560	-0.34499866	15	-100
Medición punto 1	39.48415756,-0.34522617	39.48415756	-0.34522617	15	-101
Medición punto 2	39.48418808,-0.34534454	39.48418808	-0.34534454	15	-97
Medición punto 3	39.48415375,-0.34535646	39.48415375	-0.34535646	15	-101

3.1.2 Clustering

Previo a la implementación los métodos propuestos, se graficaron los datos recopilados para analizar sus distribuciones. Se observaron ciertos puntos mostrando similitudes en sus gráficas, por lo que también se aplicaron técnicas de clustering sobre los datos. El agrupar los datos según las características de sus distribuciones permite mejorar la implementación de cada uno de los métodos propuestos, al trabajar con datos que comparten características. Esto beneficia especialmente a aquellos métodos que, debido a su simplicidad, no tienen inherentemente una gran capacidad de ajuste sobre los datos, permitiendo realizar una mejor predicción. Se utilizan tanto métodos de clustering supervisados (subjetivos) como no supervisados ("Fuzzy C-Means" y "Substractive Clustering").

Representación Estadística

Para aplicar las técnicas de clustering no supervisadas, se representaron los datos utilizando medidas estadísticas para reducir el impacto de valores atípicos y ruido, además de mantener la agrupación natural de los distintos datos, permitiendo así una clusterización que preserve las agrupaciones originales. Este enfoque asegura que el clustering se realice basado en el comportamiento de las características estadísticas representativas de los datos en diferentes puntos. Este paso es esencial para mejorar la robustez del proceso de clustering [17, 18]. Las medidas estadísticas clave utilizadas fueron:

- **Media:** Representa el valor promedio del conjunto de datos, proporcionando una medida de tendencia central [19].
- **Curtosis:** Evalúa la forma de las colas de la distribución, indicando si los datos tienen colas pesadas o ligeras en comparación con una distribución normal [20].
- **Desviación Estándar:** Indica la dispersión o variabilidad del conjunto de datos alrededor de la media [18].
- **Asimetría:** Mide la asimetría de la distribución, proporcionando información sobre la dirección de la concentración de los datos [20].

La Tabla 3.3 muestra qué métodos de clustering utilizan representaciones estadísticas para el clustering y cuáles no.

Tabla 3.3: Métodos utilizando representaciones estadísticas

Tipo de Clustering	Representación Estadística	Representación Individual
Clustering Subjetivo	X	✓
Clustering Difuso	✓	X
Clustering Sustractivo	✓	X

Clustering Difuso

Método de clustering no supervisado que permite que los puntos de datos pertenezcan a múltiples clústeres según un grado de membresía difusa, lo que lo hace ideal para capturar características ambientales superpuestas. FCM asigna grados de pertenencia a cada punto de datos, reflejando la incertidumbre en la asignación de clústeres. Este método es especialmente útil en entornos donde la propagación de la señal varía significativamente en distancias cortas [21] y es ampliamente empleado en escenarios que requieren flexibilidad para modelar datos con solapamientos [22].

Inicialmente, el algoritmo comienza con una estimación de los centros de clústeres, que representan la ubicación media de cada grupo. Luego, asigna a cada punto de datos un grado de pertenencia a cada clúster y, de manera iterativa, actualiza los centros de clústeres y los grados de pertenencia para optimizar la función objetivo 3.1, que minimiza la distancia euclidiana ponderada entre puntos y centros. Este proceso mueve gradualmente los centros de los clústeres hacia sus ubicaciones óptimas dentro del conjunto de datos.

La función objetivo minimizada por FCM es:

$$J_m = \sum_{i=1}^{C} \sum_{j=1}^{N} \mu_{ij}^m D_{ij}^2 \tag{3.1}$$

Donde:

- N: Número de puntos de datos.
- C: Número de clusters.
- m: Exponente de partición difusa, que controla el grado de difuminación en los clusters ($m > 1$).
- μ_{ij}: Grado de pertenencia del punto de datos j al cluster i.
- D_{ij}: Distancia entre el punto de datos j y el centro del cluster i.

Para aplicar esta técnica de clusterización a los datos, una vez calculadas las representaciones estadísticas por punto, se procede a normalizar los valores. Posteriormente, se utiliza la función **fcm** de MATLAB para realizar la clusterización, configurando el número de clústeres en `"auto"`. Esto implica que la función ejecuta 10 iteraciones, evaluando diferentes cantidades de clústeres desde 2 hasta 11, para determinar el número óptimo de centroides.

La configuración incluye un exponente para la matriz de partición difusa de 1, que es el valor mínimo posible con el parámetro `Exponent`. Esto limita la cantidad de intersección difusa durante el proceso de clustering, considerando que los valores de RSSI, por su naturaleza logarítmica, tienden a ser cercanos entre sí. Además, se establece un máximo de 100 iteraciones `MaxNumIteration` y una mejora mínima `MinImprovement` de 1e-5 como criterio de convergencia.

Una vez determinados los centroides, el algoritmo calcula para cada punto su grado de pertenencia a cada uno de los centroides. Finalmente, se asigna cada punto de datos al clúster con el mayor valor de pertenencia, completando así el proceso de clusterización.

Finalmente se itera sobre todos los datos y se va agrupando de acuerdo a los grupos formados sobre sus representaciones estadísticas.

Clustering Sustractivo

Método basado en densidad que identifica centros de clústeres en regiones con alta concentración de puntos de datos. Su principal ventaja es que no requiere especificar previamente el número de clústeres, lo que lo hace especialmente adecuado para el análisis exploratorio de datos, en particular para conjuntos de datos grandes [23]. Este algoritmo permite una identificación eficiente de los centros de clústeres al basarse en la densidad de los puntos circundantes. El clustering sustractivo asume que cada punto de datos es un potencial centro de clúster y basado en esa asunciónc el algoritmo 3.1 realiza los siguientes pasos:

```
1. Calcular la probabilidad de cada punto sea un centro de clúster basado
   en la densidad de puntos circundantes.
2. Seleccionar el punto con el mayor potencial como el primer centro de
   clúster.
3. Eliminar puntos cercanos basándonos en un rango de influencia
   especificado.
4. Seleccionar el siguiente punto restante con el mayor potencial de ser el
    siguiente centro.
5. Repetir hasta que todos los puntos estén dentro del rango de influencia
   de un centro de clúster.
```

Algoritmo 3.1: Clustering Sustractivo

Este método es ampliamente utilizado en sistemas de modelado y control difuso debido a su eficiencia en la identificación de estructuras de datos en espacios multidimensionales [24].

Para aplicar esta técnica de clusterización a los datos, una vez calculadas las representaciones estadísticas por punto, se procede a normalizar los valores y se utiliza la función sublcust de MATLAB para realizar la clusterización. Se configura el parámetro `clusterInfluenceRange` con un valor máximo de 1, considerando la cercanía inherente de los valores de RSSI debido a su escala logarítmica, lo que permite garantizar la mayor cantidad posible de clústeres.

Se emplea el valor predeterminado del factor `Squash` de 1.25, que ajusta el rango de influencia de los centros de clústeres. Este valor reduce la posibilidad de que puntos atípicos sean considerados parte de un clúster, generando así una mayor cantidad de clústeres. Se establecen un `radio de aceptación` de 0.5 y un `radio de rechazo` de 0.15, ambos valores relativos al primer centro de clúster identificado. Estos parámetros definen el umbral para determinar si un punto de datos es aceptado como un nuevo centro de clúster o rechazado como candidato.

Clustering Subjetivo

Este método se apoya en el juicio de expertos en lugar de métodos algorítmicos para definir los clústeres, permitiendo la clasificación de los datos en categorías predefinidas que se alinean con características ambientales específicas [25, 26].

El clustering subjetivo es particularmente útil cuando el conocimiento del dominio puede mejorar significativamente la interpretación de los resultados, como en aplicaciones de clasificación geográfica o análisis ambiental [25]. En este estudio, definimos dos clústeres principales:

- **Área Urbana**: Regiones con edificios y estructuras.
- **Área de Vegetación**: Áreas con follaje significativo y árboles.

Se selecciono un radio de 50 metros basado en estudios previos sobre modelos de propagación, donde el parámetro d_0 típicamente varía entre 1 y 100 metros, dependiendo del entorno.[27] Este rango es representativo de distancias que abarcan tanto efectos locales como transiciones entre entornos heterogéneos, como zonas de vegetación y urbanas. Optar por un radio intermedio de 50 metros garantiza un equilibrio entre capturar información local y reflejar variaciones espaciales relevantes en el área analizada.

Utilizando un código en Python, se generaron mapas estáticos con la librería `Folium` a partir de las coordenadas GPS (latitud y longitud) de los puntos de medición. En cada mapa, se dibuja un círculo de 50 metros alrededor del punto para delimitar el área de influencia3.4. Posteriormente, cada imagen se convierte a escala de grises, donde se genera un histograma que agrupa los niveles de intensidad por píxel. A partir del análisis del histograma, se define un rango dinámico de intensidades ajustado a la desviación estándar de los valores detectados, centrado en las intensidades correspondientes a los píxeles de color verde en escala de grises3.5. Este rango permite identificar los píxeles que representan vegetación, los cuales suelen acumularse en forma de un pico en el histograma cuando hay una alta presencia de áreas verdes.

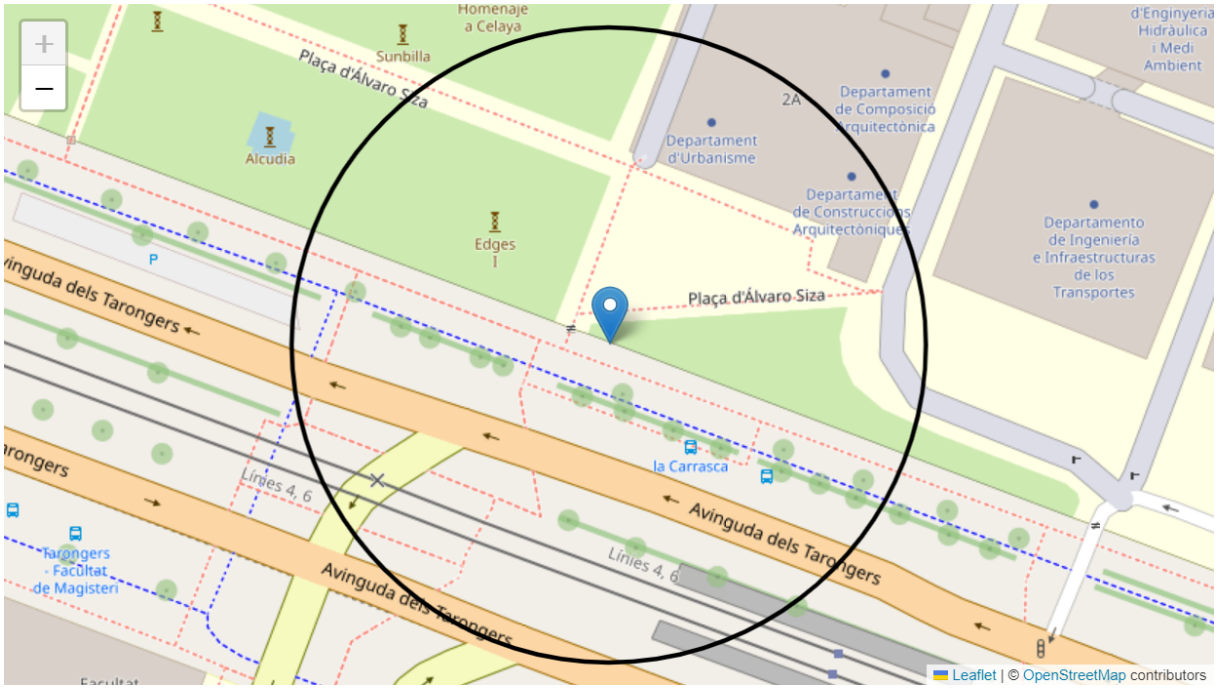


Figura 3.4: Visualización de punto graficado con Folium a partir de coordenadas.

Finalmente, se calculo el porcentaje de píxeles clasificados como vegetación, es decir, aquellos que caen dentro del rango definido, y se comparo con el total de píxeles dentro del círculo. Según este análisis, clasificamos el punto como *vegetación*

o *urbano* dependiendo de si el porcentaje de vegetación supera un umbral del 25%.

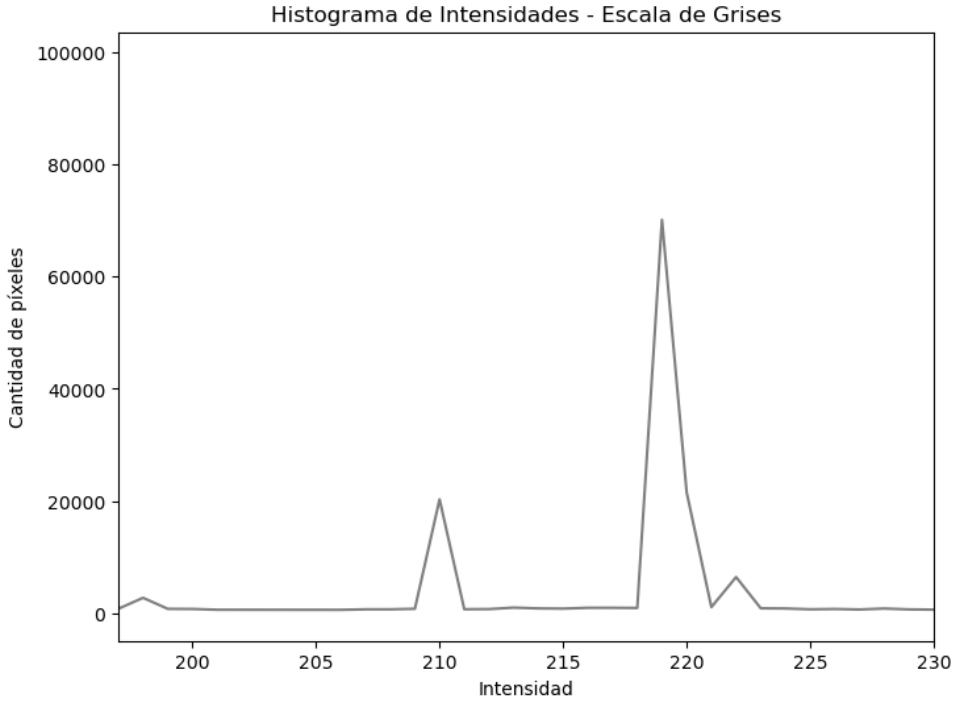


Figura 3.5: Histograma de intensidades para ROI a clasificar.

Adicionalmente, se reforzó la clasificación utilizando imágenes de `Google Earth`, ya que los mapas generados con `Folium` no permiten visualizar claramente la presencia de árboles. Esto es crucial, ya que en algunos casos los árboles pueden no ser suficientes en cantidad para clasificar un área como zona verde, aunque tengan un impacto significativo en la propagación de señales. Si en `Google Earth` se detecta la presencia de árboles con alturas superiores a 2 metros dentro del rango definido, se clasifica el punto como *vegetación*, dado que estos árboles afectan directamente la propagación de las señales.3.6

Figura 3.6: Verificación de obstáculos mediante visualización en Google earth.

3.1.3 Balanceo de datos para puntos medidos

Este estudio, se enfoca en los valores de RSSI medidos en cada punto. Debido a la naturaleza desequilibrada del conjunto de datos recopilado se propone usar tres modelos que se ajustan a la distribución de los datos subyacente por punto. Los modelos propuestos son:

- **Estimación de Densidad mediante Kernel (KDE)**: Modelo no paramétrico que permite estimar la densidad de probabilidad de los valores de una variable objetivo. Se ajusta una distribución sintética utilizando kernels predefinidos sobre la distribución de los datos reales, proporcionando una representación continua y suavizada de los datos originales. Se utilizó la regla de Silverman para calcular el ancho de banda óptimo, un parámetro necesario para equilibrar la precisión y la suavidad del ajuste. Esto evita que la estimación sea demasiado sensible al ruido o excesivamente suavizada, preservando así las características clave de los datos.1

- **Modelos de Mezcla Gaussiana Bayesianos Variante (VBGMM)**: Modelo estadístico que asume que los datos se generan a partir de una combinación de varias distribuciones gaussianas, cada una con sus propios parámetros de media y covarianza. Este método permite modelar distribuciones complejas al considerar que los datos pueden representarse como una mezcla de diferentes distribuciones gaussianas. La versión bayesiana de los GMM mejora el manejo de los parámetros del modelo al incorporar información previa sobre ellos, lo que facilita una estimación más robusta y flexible.1

- **Distribución Normal Ajustada**: Basada en una media y una desviación estándar calculadas a partir de los datos en un punto especifico, el proceso de ajuste de una distribución normal implica determinar cómo se distribuyen los

valores de los datos observados alrededor del promedio. La media representa el valor central de los datos, mientras que la desviación estándar mide la variabilidad o dispersión de los datos respecto a esa media. Al calcular estos dos parámetros, se puede trazar una curva de distribución normal que refleja la forma y el grado de dispersión de los datos reales. Este ajuste permite modelar los datos de manera más sencilla, facilitando predicciones basadas en la suposición de que los datos siguen una distribución normal.1

Después del ajuste de estos modelos, se emplearon dos métricas (la métrica de Wasserstein y la Divergencia de Kullback-Leibler) para evaluar las diferencias entre la distribución real de los datos por punto y las distribuciones predichas por los modelos. El modelo que mostró el mejor ajuste según la métrica de Wasserstein fue seleccionado para generar datos sintéticos.

La métrica de Wasserstein, también conocida como Distancia del Transportador de Tierra (Earth Mover's Distance), es una medida utilizada para cuantificar la diferencia entre dos distribuciones de probabilidad. La distancia de Wasserstein mide el mínimo "costo" necesario para transformar una distribución en la otra. Esta métrica es particularmente útil porque considera no solo las diferencias en las probabilidades individuales, sino también la "forma" y la "estructura" de las distribuciones. La Divergencia de Kullback-Leibler es una medida utilizada para cuantificar la diferencia entre dos distribuciones de probabilidad. Específicamente, mide cuánto se pierde al aproximar una distribución verdadera por una distribución aproximada.

La validación cruzada de tipo "K-Fold" fue empleada para evaluar la capacidad de generalización de los modelos. Este proceso dividió el conjunto de datos en múltiples particiones o "folds" y, en cada iteración, uno de los folds se utilizó como conjunto de prueba mientras los restantes sirvieron para el entrenamiento. Al finalizar las iteraciones del entrenamiento, se calculó el promedio de cada métrica (Divergencia KL, Distancia de Wasserstein) para cada modelo, realizando una comparación detallada del rendimiento en la representación y predicción de los valores RSSI para un punto en específico. Una semilla aleatoria fija es establecida para garantizar la reproducibilidad de los resultados, asegurando que las divisiones y otros procesos aleatorios sean consistentes entre ejecuciones.

Los 3 modelos mencionados se entrenaron por cada punto, siguiendo un método de validación cruzada como se muestra en el algoritmo 3.2

```
1. Definir función para cálculo del ancho de banda (Silverman):
   def silverman_bandwidth(data):
       return 0.9 * min(std(data), iqr(data) / 1.34) * len(data) ** (-1 /
          5)

2. Cargar y preprocesar datos:
   - Leer datos desde archivo CSV.
   - Filtrar valores donde distancia > 0.
   - Dividir datos en "X" (distancia) e "y" (RSSI).

3. Configurar validación cruzada:
   - Usar K-Fold con 5 divisiones.
   - Inicializar estructuras para almacenar métricas de cada modelo.

4. Proceso de validación cruzada:
```

```
   for fold in kfold:
       Dividir datos en entrenamiento y prueba (train/test).

       # Modelo 1: Distribución Normal
       Calcular media y desviación estándar de y_train.
       Generar datos simulados (distribución normal).
       Calcular métricas: KL, Wasserstein.

       # Modelo 2: KDE
       Configurar búsqueda en malla:
         kernels = ["gaussian", "tophat", "epanechnikov"]
         bandwidths = rango calculado por Silverman.
       Realizar GridSearch para ajustar mejor modelo KDE.
       Generar datos simulados.
       Calcular métricas: KL, Wasserstein.

       # Modelo 3: VBGMM
       Configurar búsqueda en malla:
         n_components = [1, 2, 3]
         covariance_types = ["full", "tied", "diag", "spherical"]
       Realizar GridSearch para ajustar mejor modelo VBGMM.
       Generar datos simulados.
       Calcular métricas: KL, Wasserstein.

5. Reportar resultados promedio:
   - Calcular promedio de métricas (KL, Wasserstein) para cada modelo.
   - Imprimir resultados para comparar desempeño.
```

Algoritmo 3.2: Entrenamiento de modelos para balancear los datos

Estos modelos permitieron generar puntos de datos sintéticos para los 15 puntos donde se realizaron las mediciones, reflejando con precisión la variabilidad inherente observada. La implementación se realizó con un script en Python 3.11.3[1], utilizando GridSearchCV de Scikit-learn 1.2.2[2] para una búsqueda en malla y empleando validación cruzada para evaluar el desempeño de cada propuesta. Para KDE, se optimizaron los hiperparámetros de ancho de banda, calculados mediante la regla de Silverman, y el tipo de kernel («gaussian», «tophat», «epanechnikov»), determinando así la mejor combinación para generar valores sintéticos representativos mediante este método. En el modelo VBGMM, se estableció automáticamente la configuración de un número máximo de componentes adecuado a la cantidad de valores únicos presentes en los datos que corresponden a la partición de entrenamiento durante el proceso de ajuste del método y también se probaron diferentes tipos de covarianzas («full», «tied», «diag», «spherical») a través de "Grid Search". Este proceso aseguró que los datos generados reflejaran las propiedades estadísticas originales y mejoraran la representación de áreas con datos insuficientes, equilibrando el conjunto para su posterior uso.

[1] https://www.python.org/downloads/release/python-3113/
[2] https://scikit-learn.org/1.2/install.html

3.1.4 Aumento de datos para puntos no medidos

Se comparan tres métodos diferentes para generar información correspondiente a puntos no conocidos: basado en datos, basado en teoría e híbrido. Se determina el mejor durante el entrenamiento y la inferencia.

Método Basado en Datos Este método recibe su nombre debido a los modelos que utiliza con capacidad de aprender, de manera supervisada, la información subyacente en los datos al ser entrenados con ellos. Su éxito radica en la complejidad y capacidad de ajuste de sus estructuras. Los modelos entrenados con el conjunto de datos seleccionado se utilizan para predecir valores de RSSI en función de las distancias al gateway. Este método permite generar predicciones basadas únicamente en el valor de la distancia, haciendo posible la estimación de valores de RSSI para otras distancias calculadas a partir de cualquier punto con coordenadas de latitud y longitud conocidas hasta las coordenadas del gateway.

La herramienta *Regression Learner* de MATLAB R2024a[3] se seleccionó por su facilidad de implementación y su capacidad para probar varios modelos de regresión. De entre los modelos entrenados, se utilizó y seleccionó aquel que obtuvo el mejor desempeño en el conjunto de datos empíricos. Para los entrenamientos realizados con la herramienta, se aplicó una técnica de validación cruzada con una configuración de 5 pliegues preconfigurada en MATLAB, reservando el 10 % de los datos para probar el modelo mejor ajustado.

Una vez que se entrenan los diversos modelos ofrecidos por la herramienta y se selecciona el mejor basado en la métrica de RMSE, se procede con la siguiente etapa. Dado que la herramienta no implementa validación cruzada por grupos, se exporta la función del mejor modelo, la cual puede ser utilizada posteriormente para realizar un reentrenamiento "manual" en MATLAB. Este reentrenamiento permite definir reglas específicas en un código personalizado, asegurando que se respeten las agrupaciones naturales de los datos. Esto significa que las mediciones pertenecientes a un mismo grupo no se mezclen durante los procesos de entrenamiento y prueba.

Este enfoque combina la facilidad de entrenamiento de múltiples modelos que ofrece MATLAB con un reentrenamiento personalizado, garantizando que se respete la estructura inherente de los datos. En los casos en los que no existan múltiples grupos en los datos de entrenamiento, se opta por la validación cruzada tradicional K-Fold para evaluar el rendimiento del modelo. A continuación, se describen los modelos de regresión utilizados brevemente:

Método Basado en Teoría (Intercepto Flotante) Se fundamenta en reglas inherentes de la física como el comportamiento de la propagación de ondas en la naturaleza. Los modelos que utiliza , tradicionalmente se ajustan sobre mediciones empíricas. El modelo escogido para la representación del método es **"Intercepto Flotante"**.

Este modelo es una variante del modelo de pérdida de trayectoria de log-distancia, se selecciono de un proceso de exploración del panorama del estado del arte de modelos de propagación que respaldan las redes LoRa, específicamente modelos de propagación empíricos comúnmente utilizados en sistemas de comunicación y cuyo desempeño se evaluó en diversos entornos como exteriores, interiores y en vegetación. Este modelo destaca la prevalencia del decaimiento logarítmico presente en la naturaleza de la propagación de señales, el cual también está presente en la mayoría de los modelos de propagación explorados.

[3] https://la.mathworks.com/products/new_products/release2024a.html

Tabla 3.4: Modelos de Regresión y sus Descripciones

Modelo de Regresión	Descripción
Support Vector Machines (SVM)	Encuentra un hiperplano que maximice el margen tolerado de error en los datos. Se utiliza por su capacidad para manejar datos de alta dimensionalidad y su flexibilidad al incorporar kernels [28].
Gaussian Process Regression Models	Modelos no paramétricos que definen una distribución sobre funciones y son útiles para la predicción con incertidumbre cuantificada. Se utilizan en problemas complejos debido a su flexibilidad [29].
Kernel Approximation Regression Models	Emplean técnicas como las características de Fourier aleatorias para aproximar funciones kernel complejas[30].
Ensemble of Trees	Combinan múltiples árboles de decisión para mejorar la precisión y reducir el riesgo de sobreajuste. Son robustos frente a datos ruidosos [31].
Regression Trees	Dividen iterativamente los datos en subconjuntos homogéneos según las características, lo que los hace intuitivos y efectivos para capturar relaciones no lineales [32].
Neural Networks	Modelos de aprendizaje profundo que pueden capturar relaciones complejas entre variables mediante arquitecturas jerárquicas. Son útiles para datos de gran escala y patrones altamente no lineales [33].

El modelo está conformado por un término de intercepto ajustable α, un término correspondiente a la pendiente β de la línea y un componente de decaimiento logarítmico, que representa el comportamiento de la fuerza de la señal promedio medida a medida que aumenta la distancia entre el receptor y el transmisor. Además, incorpora una distribución log-normal cerca de la media para capturar el ruido, lo que mejora su capacidad para ajustarse a datos empíricos, como se representa en la ecuación 3.2.

$$PL(d) = \alpha + 10\beta \log_{10}(d) + X_\sigma \tag{3.2}$$

Utilizando un proceso de validación cruzada por grupos, GroupKFold, que mantiene la estructura subyacente de agrupación de los diversos datos tomados en cada punto, se dividieron los datos en los pliegues correspondientes de entrenamiento y prueba, asegurando que las mediciones del mismo grupo no se mezclaran y evitando así la fuga de información. En los casos donde los datos de entrenamiento no presentan múltiples grupos o agrupaciones naturales, se emplea la validación cruzada tradicional K-Fold.

Posteriormente, se calculan los residuales basados en la diferencia entre la predicción del segmento de modelo ajustado y los datos reales del "fold", y sobre estos se calculó la desviación estándar. Luego, se introduce una distribución gaussiana ajustada con media cero y la desviación estándar de los residuales calculados; se procede a generar ruido aleatoriamente siguiendo dicha distribución y agregarlo a la predicción del segmento ajustado, completando así la predicción del modelo.

Al incorporar esta distribución de ruido modelada, el modelo busca simular condiciones del mundo real de manera más precisa, capturando aspectos que los términos y componentes logarítmicos no pudieron. Finalmente, se calcularon las métricas comparando con los datos de prueba de cada iteración, se promediaron las métricas a lo largo de todos los "folds" para evaluar la eficacia general del metodo, y se identificó y guardó los parámetros del modelo con la mejor distancia de Wasserstein para su futura implementación. El proceso se muestra en el Algoritmo 3.3.

```
1. Importar bibliotecas y definir funciones clave:
   - Definir PDF de distribución normal.
   - Definir modelo de RSSI como: RSSI = alpha + 10 * beta * log10(Distance).

2. Cargar y preprocesar datos:
   - Leer archivo CSV.
   - Filtrar filas con distancia > 0.
   - Dividir en "X" (distancia), "y" (RSSI) y "groups".

3. Configurar validación cruzada por grupos (GroupKFold):
   - Dividir en n pliegues respetando los grupos definidos.

4. Loop principal sobre cada pliegue:
   for cada fold in GroupKFold:
       a) Dividir datos en entrenamiento y prueba (train/test).
       b) Ajustar modelo logarítmico al conjunto de entrenamiento:
          - Estimar parámetros alpha y beta.
       c) Calcular residuos en entrenamiento y desviación estándar.
       d) Generar predicciones finales en prueba:
          - Modelo logarítmico + ruido basado en desviación estándar.
       e) Calcular distancia de Wasserstein:
          - Comparar distribuciones reales y generadas.
       f) Comparar la distancia de Wasserstein con la mejor registrada:
          - Si mejora, guardar parámetros alpha, beta y desviación estándar.

5. Almacenar resultados:
   - Guardar parámetros óptimos según distancia de Wasserstein.
   - Calcular promedio de distancia de Wasserstein sobre todos los pliegues.

### Configuraciones del modelo:
- Ajuste del modelo logarítmico: alpha, beta.
- Métrica utilizada:
  - Distancia de Wasserstein: Comparar distribuciones reales y generadas.
```

Algoritmo 3.3: Entrenamiento de modelos del metodo basado en teoría.

Método Híbrido (Intercepto Flotante + KDE/VBGMM) Combina las capacidades del método basado en teoría con el método basado en datos, al emplear modelos de ambos, como se muestra en el Algoritmo 3.4. Se combina el modelo de "Intercepto Flotante" junto con el modelo de **"Estimación de densidad de kernel (KDE)"** o **"Modelos de mezcla Gaussiana Bayesianos (VBGMM)"**. Este método aprovecha las fortalezas de ambos modelos, aprendiendo la distribución subyacente de los residuales con un modelo distinto al de una variable normal, característico del modelo tradicional "Intercepto Flotante".

Para probar este método se realiza un proceso de validación cruzada por grupos, GroupKFold. En las iteraciones de este proceso, se ajusta el segmento del modelo de intercepto flotante, pero con la diferencia en la forma de ajustar un modelo sobre los residuales, proponiéndose utilizar dos modelos: KDE y VBGMM. En los casos donde los datos de entrenamiento no presentan múltiples grupos o agrupaciones naturales, se emplea la validación cruzada tradicional K-Fold.

En el caso de KDE, se utiliza el ancho de banda de Silverman para obtener un valor óptimo y una configuración de búsqueda de hiperparámetros (ancho de banda y tipo de kernel) mediante una búsqueda en malla. Por otro lado, en el caso de VBGMM, se determina el número máximo de componentes basado en la cantidad de valores de residuales únicos y, de igual manera, mediante una búsqueda en malla, se optimizan los hiperparámetros (número de componentes y tipo de covarianza).

Finalmente, para ambos casos, se realizan las predicciones de los modelos y se suman a las predicciones del segmento del modelo logarítmico ajustado. Se calculan las métricas correspondientes para evaluar el rendimiento del modelo en cada "fold" y se identifica el "fold" con la mejor distancia de Wasserstein como métrica para guardar sus parámetros y exportar el modelo para su implementación. Los procesos se muestran en el Algoritmo 3.4.

```
1. Importar bibliotecas y definir funciones:
   - Función logarítmica: RSSI = alfa + 10 * beta * log10(Distance).
   - Función de ancho de banda Silverman para KDE (opcional).

2. Cargar y preprocesar datos:
   - Leer archivo CSV.
   - Filtrar distancias <= 0.
   - Dividir en "X" (distancia), "y" (RSSI) y "groups".

3. Configurar validación cruzada por grupos (GroupKFold) con 15 pliegues.

4. Loop principal sobre cada pliegue:
   for cada fold in GroupKFold:
       a) Dividir en conjuntos de entrenamiento (train) y prueba (test).
       b) Ajustar modelo logarítmico en conjunto de entrenamiento:
          - Optimizar parámetros alfa y beta con ajuste de curvas.
       c) Calcular residuos (ruido) del modelo logarítmico.

       d) Modelar residuos con KDE o VBGMM:
          - Opción 1: KDE
             - Usar GridSearchCV para optimizar:
               - Kernels: 'gaussian', 'tophat', 'epanechnikov'.
               - Bandwidth ajustado con Silverman.
```

```
            - Generar ruido basado en KDE ajustado.
         - Opción 2: VBGMM
            - Usar GridSearchCV para optimizar:
              - Número de componentes: 1 a máximo único en residuos.
              - Tipos de covarianza: 'full', 'tied', 'diag', 'spherical'.
            - Generar ruido basado en VBGMM ajustado.

      e) Predecir en conjunto de prueba:
         - Sumar ruido generado a predicciones logarítmicas para obtener
           predicciones finales.
      f) Evaluar métricas:
         - RMSE: precisión de predicciones.
         - Distancia de Wasserstein: similitud de distribuciones.
      g) Actualizar mejor modelo si distancia de Wasserstein mejora:
         - Guardar alfa, beta, modelo de residuos (KDE o VBGMM) y métricas
           asociadas.

5. Guardar resultados:
   - Parámetros promedio: Wasserstein.
   - Mejor modelo y configuraciones guardados para uso futuro.
```

Algoritmo 3.4: Entrenamiento de modelos del metodo híbrido.

Implementación de aumento de datos para puntos no medidos

Para implementar el aumento de datos en puntos no medidos mediante la predicción del modelo correspondiente al mejor de los métodos propuestos, se desarrolló un código en Python que genera puntos alrededor de coordenadas establecidas. Estas coordenadas se definen como el centro de un círculo en un plano bidimensional con un radio inicial de 50 metros. A partir de este centro, se calculan nuevas coordenadas mediante incrementos de 10 grados alrededor del círculo, completando una rotación completa de 360 grados. Posteriormente, el radio del círculo se reduce en 10 metros, manteniendo el mismo centro, y el proceso se repite, formando círculos concéntricos anidados. 3.7

Este método tiene como objetivo generar datos adicionales en áreas cercanas a las coordenadas establecidas, que serán utilizados para entrenar el modelo de aprendizaje profundo destinado a la predicción de coordenadas de latitud y longitud. Las nuevas coordenadas generadas se utilizan para calcular la distancia al gateway más cercano mediante la biblioteca geopy (función geodesic). Estas distancias sirven como entradas para aplicar el método que demostró el mejor desempeño durante las pruebas, permitiendo realizar una predicción para las nuevas ubicaciones de valores RSSI.

En el caso del gateway "itaca-upv-022", se selecciona como modelo óptimo para la predicción y aumento de datos el método mixto que combina el modelo de Intercepto Flotante con VBGMM (Variational Bayesian Gaussian Mixture Models) para modelar el ruido. Este modelo se entreno utilizando datos clusterizados mediante el método de clusterización sustractiva, debido a su consistencia al mejorar las métricas de desempeño en los tres métodos propuestos. Aunque este mismo método mixto también muestra un buen desempeño en las métricas para los datos clusterizados mediante "Fuzzy C-Means", se prioriza el

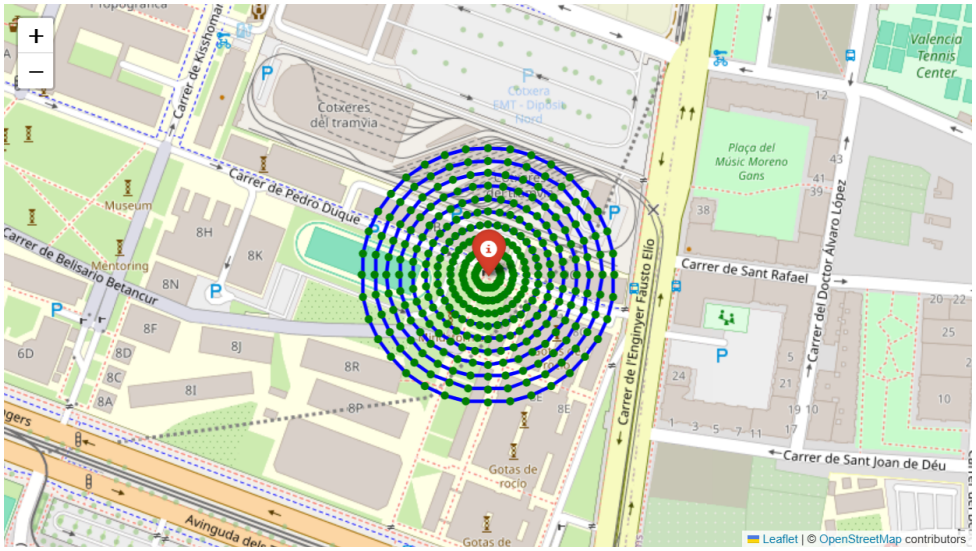


Figura 3.7: Coordenadas circundantes anidadas a un punto objetivo para implementación de la técnica de aumento de datos.

uso de la clusterización sustractiva por su superioridad consistente al equilibrar los datos en múltiples experimentos.

Por otro lado, para el gateway "main-gtw-grc", se opta por el modelo basado en datos, específicamente el modelo de regresión, debido a que su estructura le permite desempeñarse de manera adecuada con la cantidad limitada de datos disponibles. Al igual que en el caso del gateway anterior, el criterio de selección esta basado en la consistencia al mejorar las métricas cuando se utilizó el método de clusterización sustractiva. Este enfoque permite aprovechar de manera más eficiente los datos recolectados, asegurando resultados más fiables en entornos de datos escasos.

3.1.5 Dataset Balanceado y Aumentado

Una vez realizada la comparación de los tres métodos propuestos para la predicción de valores RSSI en puntos no conocidos, tanto para los datos sin clusterizar como para los clusterizados con los algoritmos mencionados anteriormente, así como la comparación entre el mejor de los tres métodos propuestos para balancear los datos en puntos conocidos, se crean conjuntos de datos para desarrollar la solución de predicción de coordenadas de latitud y longitud mediante un modelo de aprendizaje profundo basado en redes neuronales de grafos (GNN), y se analiza cómo los métodos propuestos afectan el entrenamiento del mismo.

Los conjuntos de datos se describen de la siguiente manera:

1. Datos originales recolectados en los 15 puntos, los cuales presentan un desbalance.
2. Datos balanceados en los 15 puntos, utilizando el método propuesto para el balanceo de datos en puntos medidos (ver sección 3.1.3).
3. Datos del conjunto 1, complementados con valores de RSSI generados mediante el método propuesto para el aumento de datos en puntos no medidos (ver sección 3.1.4), considerando **ambos *gateways***.
4. Datos del conjunto 1, complementados con valores de RSSI generados mediante el método propuesto para el aumento de datos en puntos no medidos (ver sección 3.1.4), considerando únicamente el ***gateway* con mayor cobertura** y, por

ende, mayor cantidad de datos (`main-gtw-grc`).

3.2 Arquitectura Basada en Grafos

Un grafo es una estructura matemática que se utiliza para modelar relaciones entre entidades. Está compuesto por un conjunto de nodos (también llamados vértices) y un conjunto de aristas (o conexiones) que describen cómo los nodos están relacionados entre sí. Los grafos son ampliamente utilizados en una variedad de dominios, como redes sociales, sistemas de comunicación y biología computacional [34].

Las Redes Neuronales Basadas en Grafos (*Graph Neural Networks*, GNN) son modelos de aprendizaje profundo diseñados para operar directamente sobre datos estructurados en forma de grafos. Estos modelos combinan la información incrustada de los nodos y sus aristas, propagándola en iteraciones denominadas capas sobre la red, mediante un mecanismo conocido como "message passing" 3.5. Este proceso permite que la información se intercambie entre los nodos de un grafo, facilitando el aprendizaje de representaciones enriquecidas que capturen tanto las propiedades individuales de los nodos como las relaciones entre ellos. Esto hace posible que las GNN aprendan representaciones útiles para tareas como la clasificación de nodos, la predicción de enlaces y la regresión [35, 36]. Las GNN son particularmente efectivas en problemas donde las relaciones entre las entidades son tan importantes como las características individuales de estas [37]. Gracias a la capacidad de generalización de las GNN a topologías variables de grafos durante el entrenamiento y la inferencia (siempre que las dimensiones de los atributos sean consistentes), estas se integran de manera eficiente con la arquitectura LoRaWAN y su naturaleza de recepción de mensajes variante.

```
1. Importar bibliotecas y definir funciones clave:
   - Definir la función "AGGREGATE":
     * Toma como entrada las representaciones de los nodos vecinos.
     * Retorna un mensaje agregado para el nodo objetivo.
   - Definir la función "UPDATE":
     * Toma como entrada la representación previa del nodo y el mensaje agregado.
     * Devuelve la nueva representación para el nodo.

2. Cargar e inicializar datos:
   - Definir el grafo G = (V, E) con nodos V y aristas E.
   - Asignar las representaciones iniciales: {h_v^(0) : v ∈ V}.
   - Especificar el número de iteraciones K.

3. Bucle principal de propagación de mensajes (Message Passing):
   for k in range(1, K+1):
       for v in V:
           m_v^(k) = AGGREGATE({ h_u^(k-1) : u in N(v) })
           h_v^(k) = UPDATE(h_v^(k-1), m_v^(k))

4. Almacenar y devolver resultados:
   - Guardar o retornar las representaciones finales: {h_v^(K) : v ∈ V}.
```

Algoritmo 3.5: "Message Passing" en una GNN.

A continuación, se describe el proceso de creación de una red neuronal basada en grafos (GNN) para predecir las coordenadas de latitud y longitud de un nodo transmisor en la red LoRaWAN. Esta metodología, inspirada en los conceptos de redes convolucionales en grafos [35] y aprendizaje representacional en grafos de gran escala [37], permite modelar las relaciones entre un nodo transmisor y dos gateways receptores.

Modelado de grafos

Los metadatos de un proceso de comunicación de un dispositivo en una red LoRaWAN fue reestructurada mediante un código en Python y los datos que serán de entrada para la red estarán organizados en archivos JSON. Cada archivo contiene información necesaria para representar un grafo, incluyendo nodos, aristas y etiquetas. A continuación, se describe cada elemento clave:

- **Nodos**: Representan los dispositivos transmisores (*devices*) y los gateways. Los gateways tienen atributos conocidos de latitud y longitud, mientras que los dispositivos tienen estos valores vacíos, ya que serán predichos por la red. Cada nodo incluye los siguientes campos:
  - **id**: Identificador único del nodo.
  - **type**: Tipo del nodo, que puede ser *gateway* o *device*.
  - **latitude** y **longitude**: Coordenadas geográficas (solo para *gateways*).
  - **latitude_label** y **longitude_label**: Coordenadas reales del dispositivo transmisor (solo para *devices*, usadas como etiquetas en el entrenamiento).

  [noitemsep, topsep=0pt]

- **Aristas**: Representan las conexiones entre dispositivos y gateways. Cada arista incluye:
  - **source** y **target**: Nodos conectados por la arista.
  - **rssi**: Indicador de fuerza de la señal recibida entre el dispositivo y el gateway.

Un ejemplo de un archivo JSON se muestra a continuación 3.6.

```
{
  "nodes": [
    {"id": "G1", "type": "gateway", "latitude": 39.4787, "longitude":
        -0.3338},
    {"id": "D1", "type": "device", "latitude_label": 39.4841, "
        longitude_label": -0.3455}
  ],
  "edges": [
    {"source": "D1", "target": "G1", "rssi": -122.0}
  ]
}
```

Algoritmo 3.6: Ejemplo de archivo JSON para la representación de un grafo.

En este contexto, cada *gateway* puede representarse como un nodo, mientras que un dispositivo (*device*) dentro del área de cobertura de la red que envía un mensaje modulado será recibido por varios *gateways* circundantes. Esto hace que la cantidad de nodos (*gateways* que receptaron el mensaje) sea variable, con cada uno aportando valores de RSSI, latitud y longitud. Esta flexibilidad se traduce en una estructura de grafo adaptable, donde el número de nodos *gateways* depende de cuántos escucharon el mensaje. Estos nodos variables pueden ingresar a la red GNN y participar en la predicción de la latitud y longitud de un dispositivo que ha transmitido un mensaje, mejorando la capacidad de inferencia de la arquitectura.

Un grafo $G = (V, E)$ se define formalmente como:

- V: Conjunto de nodos, donde cada nodo $v \in V$ representa un *device* o un *gateway*.

- E: Conjunto de aristas, donde cada arista $e_{ij} \in E$ conecta un nodo v_i con un nodo v_j. Cada arista está asociada a un atributo $rssi_{ij}$, que representa la fuerza de la señal entre el dispositivo y el gateway.

Esta definición permite incorporar información espacial y relacional entre dispositivos y gateways, lo cual es fundamental para la tarea de predicción de coordenadas geográficas.

Para transformar los datos provenientes de archivos JSON en grafos compatibles con *PyTorch Geometric*, se sigue un proceso organizado en tres etapas principales:

1. **Procesamiento de nodos**:

    - Los nodos de tipo *gateway* tienen como atributos iniciales sus coordenadas geográficas de latitud y longitud.

    - Los nodos de tipo *device* inician con atributos vacíos representados como $(0, 0)$, ya que sus coordenadas serán predichas por el modelo durante el entrenamiento.

2. **Procesamiento de aristas**:

    - Cada arista conecta un nodo de tipo *device* con un nodo *gateway*, utilizando como atributo el valor del Indicador de Fuerza de la Señal Recibida (RSSI, por sus siglas en inglés).

3. **Creación de etiquetas (labels)**:

    - Los nodos de tipo *device* tienen como etiquetas las coordenadas reales de latitud y longitud definidas en el archivo JSON, las cuales son utilizadas como el objetivo durante el entrenamiento.

Este proceso modela grafos con estructuras flexibles y adaptables a distintas configuraciones de datos, ya que varían en número de nodos y aristas. Un ejemplo se muestra en la Figura 3.8.

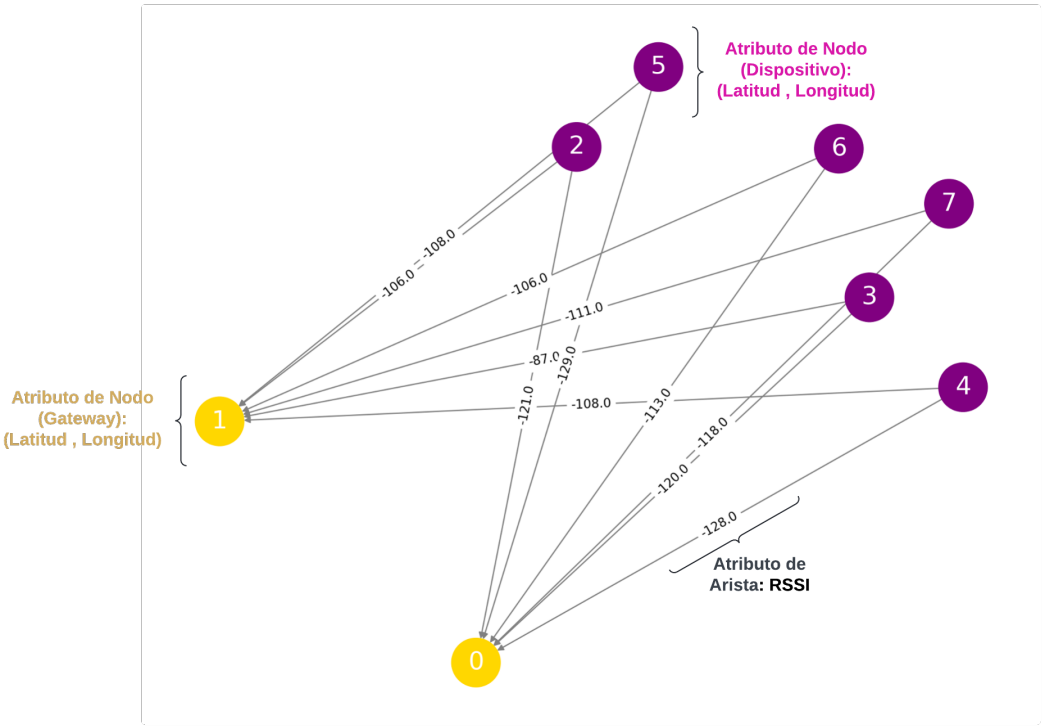


Figura 3.8: Grafo generado a partir de datos JSON. Los nodos morados representan dispositivos (*device*) con atributos vacíos, mientras que los nodos amarillos representan *gateways* con coordenadas conocidas. Las aristas indican conexiones con valores de RSSI como atributos.

3.2.1 Red GNN Propuesta

La solución propuesta se basa en una arquitectura de red que emplea dos módulos principales: Una capa del mecanismo de propagación de mensajes (*message-passing*) 3.5 y un modulo de predicción de coordenadas, juntos permiten la predicción de latitud y longitud a nivel de nodo. El modulo de propagación de mensajes describe cómo se intercambia la información entre los nodos a través de las aristas para actualizar sus representaciones de acuerdo al siguiente esquema:

$$m_{ij} = \text{ReLU}(W_m \cdot (x_i + e_{ij} + x_j)), \tag{3.3}$$

$$h'_i = \text{ReLU}(W_u \cdot \sum_{j \in \mathcal{N}(i)} m_{ij}), \tag{3.4}$$

donde:

- m_{ij}: Mensaje propagado desde el nodo j al nodo i.
- h'_i: Representación actualizada del nodo i.
- x_i, x_j: Representaciones iniciales de los nodos i y j.
- e_{ij}: Representación del atributo de la arista entre i y j.
- W_m y W_u: Matrices de pesos aprendibles.

Una vez que el módulo de propagación de mensajes obtiene la representación final del dispositivo, proceso llevado a cabo por las capas de la red GNN, la representación se alimenta a un perceptrón multicapa (MLP) que predice los valores :

$$(\hat{\text{lat}}, \hat{\text{lon}}) = \text{MLP}(h_{\text{device}}), \tag{3.5}$$

donde $\hat{\text{lat}}$, $\hat{\text{lon}}$ son las coordenadas estimadas. Durante el entrenamiento, a partir de esta representación final se calcula el error y se aplica el descenso de gradiente en el espacio del error específico. Posteriormente, mediante retropropagación, se actualizan los pesos correspondientes de las neuronas presentes en la arquitectura de la red.

3.2.2 Entrenamiento de la GNN

Los experimentos se llevaron a cabo en un sistema de computación de alto rendimiento (*High-Performance Computing, HPC*) equipado con cuatro GPUs NVIDIA TESLA A100, cada una con 40 GB de memoria VRAM. Los recursos computacionales fueron proporcionados por la *Corporación Ecuatoriana para el Desarrollo de la Investigación y la Academia (CEDIA)*. La implementación del modelo se realizó utilizando PyTorch 2.0.1[4] y Python 3.8[5].

División de Datos

La partición de los datos se realizó asignando el 80% de las muestras para el entrenamiento y el 20% para las pruebas. Adicionalmente, se empleó validación cruzada estratificada (*Stratified K-Fold Cross Validation*) con 5 grupos, garantizando que la distribución de las clases se mantuviera uniforme en cada pliegue [38].

Métricas de Evaluación

Para evaluar el desempeño del modelo, se utilizaron las siguientes métricas:

- **Error cuadrático medio (RMSE):** Esta métrica mide la desviación promedio de las predicciones respecto a los valores reales, penalizando errores grandes. Su fórmula es:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \tag{3.6}$$

  donde y_i representa el valor real, $\hat{y}_i$ la predicción y n el número total de muestras.

- **Distancia Euclidiana Promedio:** Esta métrica mide la distancia promedio entre las coordenadas reales y las predichas, calculada como:

$$d = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}, \tag{3.7}$$

  donde (x_i, y_i) son las coordenadas reales y $(\hat{x}_i, \hat{y}_i)$ son las predicciones del modelo.

[4] https://pytorch.org/get-started/pytorch-2.0/

[5] https://www.python.org/downloads/release/python-380/

Tunning de Hiperparámetros

Para el entrenamiento de la red GNN, se llevó a cabo un proceso iterativo de búsqueda de hiperparámetros, en el cual se modificó el rango de búsqueda para el número de neuronas en las capas ocultas y la tasa de aprendizaje (*learning rate*). Cada vez que se ejecutaba dicho proceso, se realizaba una búsqueda de hiperparámetros utilizando las bibliotecas `Optuna` [6] y `RayTune`[7]. En concreto, `Optuna` se encarga de llevar a cabo el muestreo bayesiano en el rango de hiperparámetros definido, mientras que `RayTune` implementa el entrenamiento de los diversos experimentos con las configuraciones correspondientes. Así, `Optuna` trabaja con el rango establecido en `RayTune` e implementa el muestreo bayesiano de dichos hiperparámetros. Por su parte, `RayTune` se encarga del entrenamiento del modelo en cada experimento, del cálculo y reporte de métricas, y de la finalización de aquellos experimentos que no resultan prometedores. Para ello, se configuró un experimento con 50 pruebas, cada una limitada a un máximo de 100 épocas, habilitando la detención temprana (*early stopping*).

Una vez finalizado este procedimiento y entrenadas las épocas necesarias, el proceso se repitió ajustando el centro del rango especificado en `Optuna` para refinar aún más la búsqueda de dichos hiperparámetros y, de este modo, explorar con mayor detalle el espacio de valores potenciales que pudieran mejorar el entrenamiento del modelo.

Al concluir los experimentos, se seleccionó la configuración con el mejor desempeño y se estableció como la configuración predeterminada para el aprendizaje de la red. A partir de ella, se incrementó la cantidad de épocas de entrenamiento con el fin de examinar si una mayor duración del proceso mejoraba los resultados.

Este procedimiento se aplicó a cuatro propuestas del conjunto de datos: Balanceado, no balanceado, aumentado para *ambos* gateways y aumentado *únicamente* para el gateway `itaca-upv-022`. Finalmente, tras identificar la mejor configuración, se fijaron los hiperparámetros óptimos y se aumentó el número de épocas de entrenamiento.

El optimizador elegido para el entrenamiento de los experimentos fue AdamW, debido a su amplia utilización en trabajos relacionados con el estado del arte. Este proceso permitió optimizar el modelo, mejorando su precisión en la predicción de las coordenadas objetivo.

[6] https://optuna.org/

[7] https://docs.ray.io/en/latest/tune/index.html

4

Resultados

Una vez implementados los diferentes procesos descritos en la metodología para realizar el balanceo de datos en puntos medidos y el aumento de datos en puntos no medidos, así como también el desarrollo de la red GNN y su entrenamiento, se presentan a continuación los resultados obtenidos, los cuales reflejan el impacto de la solución propuesta.

4.1 Clustering

A continuación, se presentan los resultados obtenidos al aplicar distintos métodos de clustering a los puntos de datos, con el fin de agruparlos según sus características y ubicación. Se muestra cómo cada método identifica patrones y distribuciones relevantes en el entorno analizado.

4.1.1 Clustering Fuzzy C-Means

El algoritmo Fuzzy C-Means (FCM) agrupa los datos asignando a cada punto un grado de pertenencia a múltiples clústeres, permitiendo una representación más flexible del entorno. Como se muestra en las Figuras 4.1 y 4.2, para el gateway "itaca-upv-022" se formaron dos clústeres: el primero incluye los puntos del 1 al 7, mientras que el segundo abarca del 8 al 15, posiblemente influenciados por la distancia al gateway. Por otro lado, para el gateway "main-gtw-grc", se formaron cuatro clústeres: uno para el punto 1, otro para el punto 2, otro para el punto 3, y un cuarto que incluye los puntos del 4 al 6. Estas divisiones reflejan cómo el algoritmo captura patrones espaciales y distribuciones de los datos en función de su proximidad a los gateways.

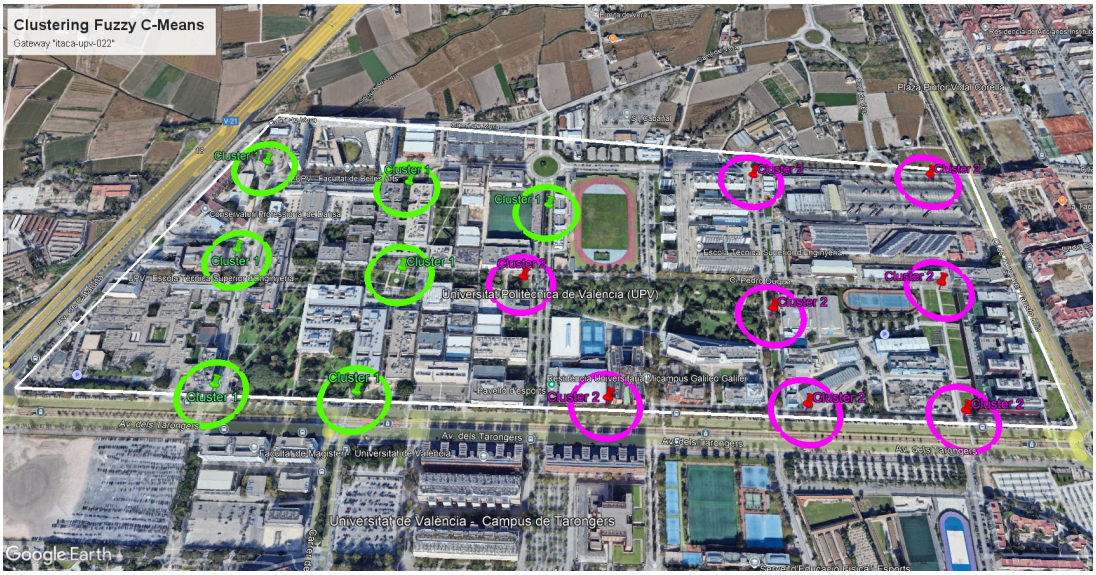


Figura 4.1: Puntos de Datos Agrupados por Fuzzy C-Means.

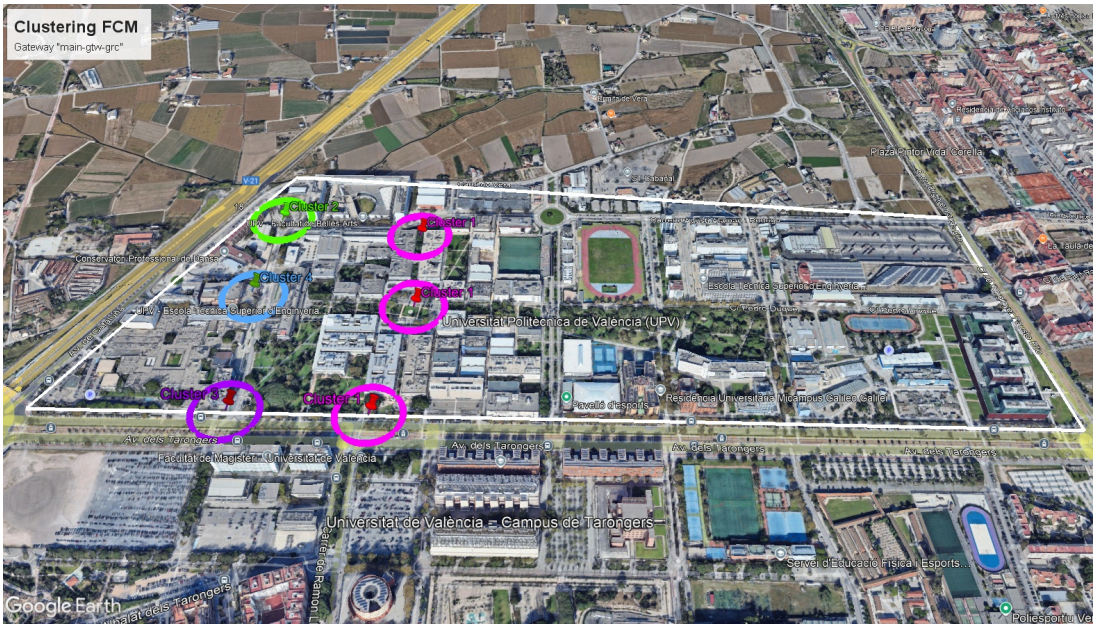


Figura 4.2: Puntos de Datos Agrupados por Fuzzy C-Means.

4.1.2 Clustering Substractivo

El clustering sustractivo identifica regiones de alta densidad en los datos y asigna los centros de los clústeres de manera eficiente, minimizando la necesidad de ajustes manuales y adaptándose de forma robusta a tamaños de clústeres variables. Como se observa en las Figuras 4.3 y 4.4, este método produce agrupaciones que reflejan la distribución espacial de los datos. Para el gateway "main-gtw-grc", se formaron cuatro clústeres: los puntos 4, 5 y 6 se agruparon en un clúster, mientras que los puntos 1, 2 y 3 formaron clústeres individuales. En el caso del gateway "itaca-upv-022", se generaron cuatro clústeres:

el primero incluyó los puntos 2, 5, 7, 8, 9 y 10; el segundo agrupó los puntos 1, 3 y 6; el tercero abarcó los puntos del 11 al 15, mientras que el punto 4 se asignó a un clúster independiente. Estas agrupaciones reflejan cómo este enfoque captura eficientemente las concentraciones de datos, adaptándose a la complejidad del entorno y la distribución de los puntos.

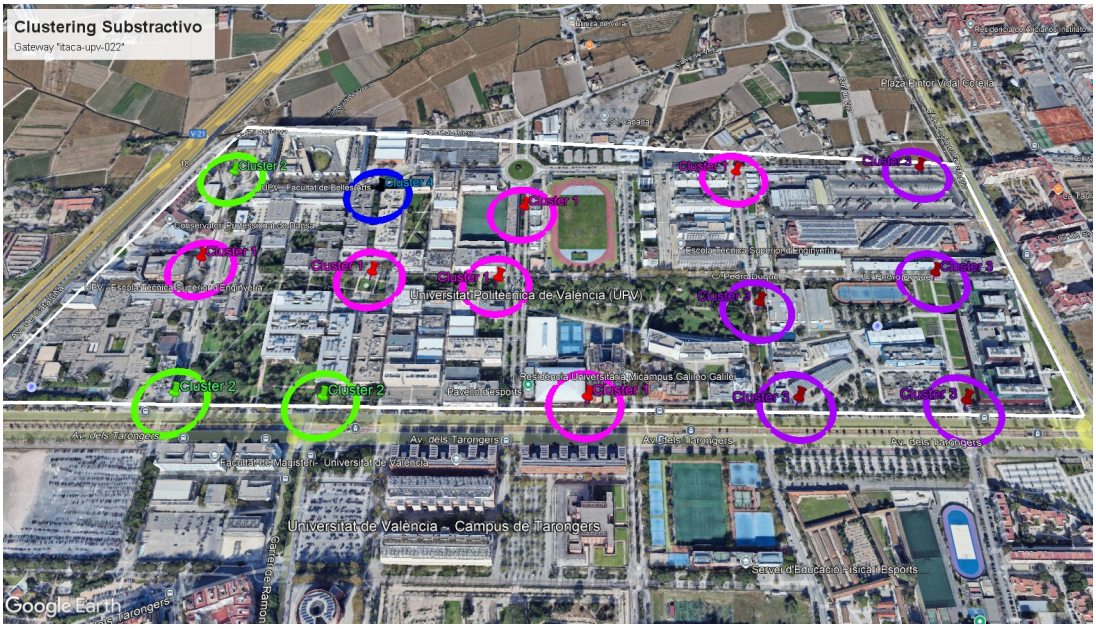


Figura 4.3: Puntos de Datos Agrupados por Clustering Substractivo.

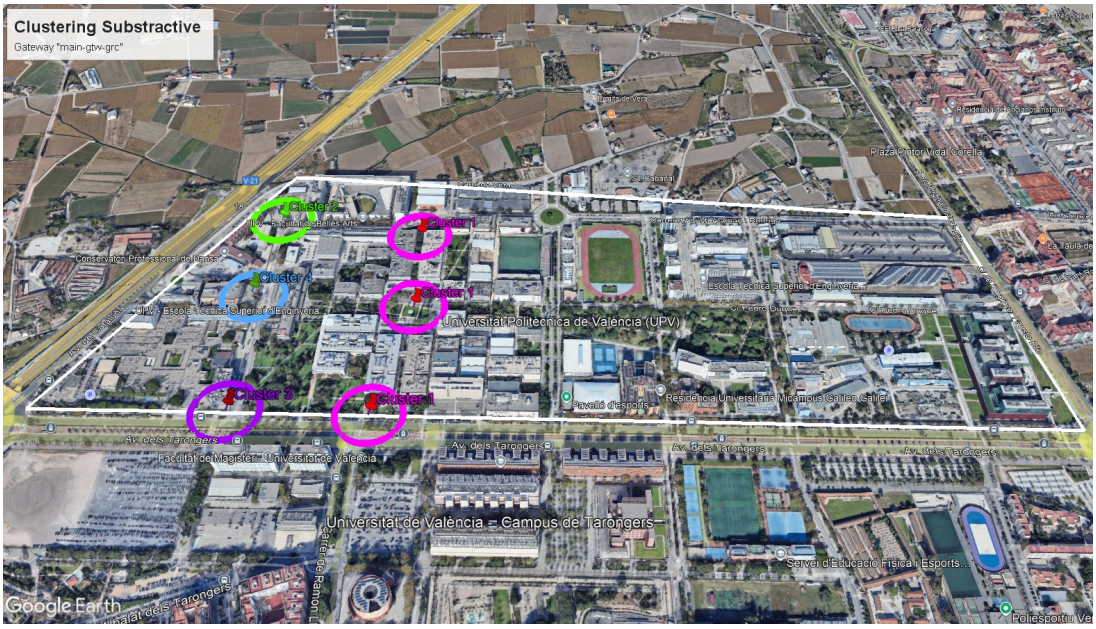


Figura 4.4: Puntos de Datos Agrupados por Clustering Substractivo.

4.1.3 Clustering Subjetivo

El clustering subjetivo propuesto agrupa los puntos de datos en clústeres predefinidos centrándose en una región alrededor del punto de interés y analizando las características de la imagen como el nivel de intensidad de los pixeles y presencia de obstáculos mediante una verificación visual. Como se ilustra en las Figuras 4.5 y 4.6, este enfoque categoriza los puntos en función de su ubicación relativa a entornos urbanos o de vegetación. Para el gateway "itaca-upv-022", los puntos 1 al 7, 9, 10, 12, 14 y 15 fueron clasificados como parte del clúster urbano, mientras que los puntos 6, 8 y 11 se agruparon en el clúster de vegetación. En el caso del gateway "main-gtw-grc", los puntos 1 al 6 fueron categorizados como urbanos, con el punto 6 también incluido en el clúster de vegetación.

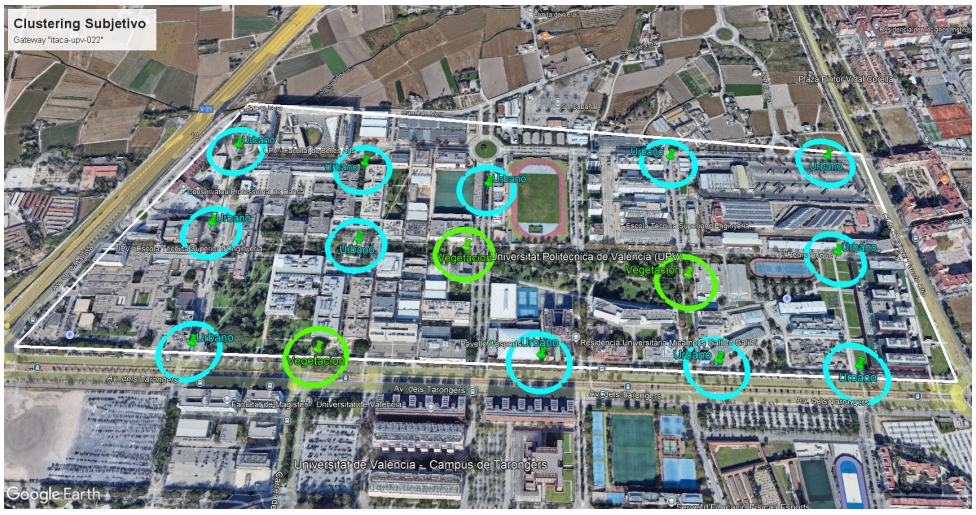


Figura 4.5: Puntos de Datos Agrupados Subjetivamente.

Figura 4.6: Puntos de Datos Agrupados Subjetivamente.

4.2 Balanceo de datos

En esta sección se presentan los resultados obtenidos al implementar la metodología de balanceo de datos para los gateways **"itaca-upv-022"** y **"main-gtw-grc"**.

En las Tablas 4.1 y 4.2, se presenta la distribución inicial de las mediciones por gateway antes de aplicar el proceso de balanceo, destacando en la columna "Cantidad" el desequilibrio existente. El balanceo se realizó utilizando el mejor modelo ajustado para cada punto de medición, según la distribución de los datos evaluada. Este proceso permitió generar datos adicionales que equilibran las distribuciones de datos por punto, optimizando así las condiciones del conjunto de datos para el posterior entrenamiento del modelo GNN. A continuación, se describen los detalles de las métricas de evaluación y el impacto del balanceo en las características de los datos.

La **Tabla 4.1** muestra los resultados previos al balanceo para el gateway **"itaca-upv-022"**, incluyendo la cantidad de datos disponibles por punto y las métricas de evaluación:

- **KL (Divergencia de Kullback-Leibler):** Cuantifica la diferencia entre la distribución de los datos en cada punto y una distribución objetivo, evaluando su similitud.

- **WS (Distancia de Wasserstein):** Mide la discrepancia entre las distribuciones de datos, proporcionando una métrica de proximidad.

- **Normal, KDE (Kernel Density Estimation) y VBGMM (Variational Bayesian Gaussian Mixture Model):** Representan los enfoques aplicados para modelar las distribuciones.

Cada fila de la tabla corresponde a un punto de medición, destacando en la columna **Enfoque** el modelo con mejor desempeño según las métricas. Las celdas resaltadas en verde indican los mejores resultados en KL, mientras que las celdas rojas corresponden a los mejores valores en WS.

De manera similar, la **Tabla 4.2** presenta los resultados del gateway **"main-gtc-grc"**. Se incluyen las métricas KL y WS junto con los enfoques aplicados, identificando el modelo óptimo para cada punto de medición. Estas tablas ofrecen una base cuantitativa para seleccionar las distribuciones más representativas y garantizar un balanceo de datos adecuado, fortaleciendo el conjunto de datos para el entrenamiento del modelo GNN.

Tabla 4.1: Cantidad de data y Evaluación de métricas previo al balanceo para Gateway “itaca-upv-022”. Abreviaciones: WS = Wasserstein, KL = Divergencia de Kullback-Leibler, KDE = Kernel Density Estimation, VBGMM = Variational Bayesian Gaussian Mixture Model.

Puntos	Cantidad	Normal		KDE		VBGMM		Enfoque
		KL	WS	KL	WS	KL	WS	
Punto 1	10	11.7203	0.85	8.9461	0.6	9.3831	0.6	KDE
Punto 2	43	6.1928	0.9279	6.3253	1.0168	6.1317	0.8473	VBGMM
Punto 3	58	5.8369	0.7828	5.9425	0.9054	5.8369	0.6816	VBGMM
Punto 4	43	10.203	1.6923	9.8592	2.4106	10.4464	1.4612	KDE
Punto 5	98	7.9875	1.5995	9.1098	1.2536	9.1587	1.784	Normal
Punto 6	63	9.2143	2.4035	8.1798	1.6493	9.2143	2.4035	VBGMM
Punto 7	59	8.5456	1.4075	5.288	0.8197	5.288	0.8197	KDE
Punto 8	99	3.2912	2.493	3.5718	0.9413	3.5718	0.73	VBGMM
Punto 9	79	3.9112	2.493	7.3653	1.6439	7.3653	1.6439	Normal
Punto 10	92	4.2724	1.0267	4.1758	0.95	4.2724	1.0267	KDE
Punto 11	100	4.601	0.975	4.94	0.925	2.7085	0.715	VBGMM
Punto 12	100	6.192	1.835	6.2038	1.275	6.2038	1.275	KDE
Punto 13	98	6.4534	1.3984	7.7107	1.5732	7.3515	1.5199	Normal
Punto 14	69	11.0038	2.5218	10.1009	2.1014	10.6431	2.1014	VBGMM
Punto 15	97	7.7756	1.557	6.3288	1.0784	6.4151	1.3441	KDE

Tabla 4.2: Cantidad de data y Evaluación de métricas previo al balanceo para Gateway “main-gtw-grc”. Abreviaciones: WS = Wasserstein, KL = Divergencia de Kullback-Leibler, KDE = Kernel Density Estimation, VBGMM = Variational Bayesian Gaussian Mixture Model.

Puntos	Cantidad	Normal		KDE		VBGMM		Enfoque
		KL	WS	KL	WS	KL	WS	
Punto 1	90	1.8146	0.5111	1.5895	0.6111	2.3274	0.4833	VBGMM
Punto 2	92	2.3959	0.4533	2.6035	0.5243	0.9711	0.5565	KDE
Punto 3	81	3.2408	1.1997	1.802	0.9425	2.9197	0.5273	VBGMM
Punto 4	54	2.3145	0.5538	2.8968	0.6719	2.4615	0.5631	Normal
Punto 5	53	1.6987	0.4305	1.6391	0.2892	1.5349	0.2892	VBGMM
Punto 6	84	1.4107	0.3586	0.9566	0.3683	1.0026	0.2416	VBGMM

4.3 Aumento de datos

En esta sección, se presentan los resultados obtenidos al aplicar los métodos propuestos para generar datos adicionales en puntos no medidos. Las métricas se analizan por separado para cada gateway, y los valores correspondientes se muestran

en las Tablas 4.3 y 4.4, permitiendo comparar el desempeño de los diferentes métodos y su efectividad en el proceso de aumento de datos.

4.3.1 Gateway "itaca-upv-022"

La Tabla 4.3 presenta una comparación de los métodos propuestos para el gateway "itaca-upv-022", utilizando la métrica de distancia de Wasserstein (Promedio Wasserstein) como indicador del desempeño. Las columnas presentan los métodos evaluados y las filas están organizadas según el método de clusterización empleado.

Los valores reportados en la tabla corresponden al desempeño durante el entrenamiento (utilizando datos del dispositivo Heltec LoRa V3 + GPS) y en pruebas independientes (utilizando datos de un dispositivo arbitrario Rak2270). Las celdas resaltadas en verde indican el mejor desempeño en cada fila, lo que permite identificar las configuraciones más efectivas para este gateway en particular. La tabla evidencia que los enfoques mixtos (especialmente con VBGMM) y el clustering sustractivo obtienen mejores resultados en términos de la métrica evaluada.

Tabla 4.3: Comparación de Métodos Utilizando Métricas Wasserstein [Promedio Wasserstein] para Gateway "itaca-upv-022". Las celdas resaltadas en verde indican el mejor desempeño para cada fila.

Enfoque	**Data Driven (Regresión)**	**Theory Driven (Interceptos Flotantes + Normal para Residuos)**	**Enfoque Mixto (Theory + Data Driven)**	
			KDE	**VBGMM**
Sin Clustering				
Entrenamiento (LoRa + GPS)	8.782303235	7.344776354	7.296920126	7.585732852
Prueba (Sticker)	5.483880645	5.451616129	6.774193516	6.064516129
Clustering Subjetivo				
Entrenamiento (LoRa + GPS)	10.25878258	7.827831588	8.250962906	8.042963989
Prueba (Sticker)	8.258074194	5.580641935	6.533405018	5.401430466
Clustering FCM				
Entrenamiento (LoRa + GPS)	8.324019043	7.288801715	7.39350352	7.629694982
Prueba (Sticker)	3.870977419	5.193548387	4.806441935	3.387106452
Clustering Substractivo				
Entrenamiento (LoRa + GPS)	5.480710605	5.086002076	5.6417963	5.885922202
Prueba (Sticker)	5.3225741946	3.483864516	3.483874194	3.451625806

4.3.2 Gateway "main-gtw-grc"

La Tabla 4.4 muestra el desempeño de los métodos propuestos para el gateway **"main-gtw-grc"** evaluados con la métrica de distancia de Wasserstein. La clusterización tuvo un impacto positivo en las métricas durante el entrenamiento, pero su efectividad en las pruebas depende de la cantidad y distribución de datos disponibles. El clustering sustractivo demostró ser el más consistente en el entrenamiento. En este gateway, la limitada cantidad de datos llevó a que algunos clústeres incluyeran solo un punto, lo que afectó negativamente a modelos rígidos como el de intercepto flotante. Esto resultó en

predicciones poco precisas para distancias desconocidas, especialmente en métodos como los basados en teoría y mixtos que hacen uso de este modelo.

Tabla 4.4: Comparación de Métodos Utilizando Métricas Wasserstein [Promedio Wasserstein] para Gateway "main-gtw-grc". Las celdas resaltadas en verde indican el mejor desempeño para cada fila.

Enfoque	**Data Driven (Regresión)**	**Theory Driven (Interceptos Flotantes + Normal para Residuos)**	**Enfoque Mixto (Theory + Data Driven)**	
			KDE	**VBGMM**
Sin Clustering				
Entrenamiento (LoRa + GPS)	6.384718261	2.757970652	2.895651087	2.654354783
Prueba (Sticker)	9.1	8.15	7.75	7.85
Clustering Subjetivo				
Entrenamiento (LoRa + GPS)	6.38261087	2.31958587	2.53287587	2.413343696
Prueba (Sticker)	7.7	127.2	127.85205	127.05
Clustering FCM				
Entrenamiento (LoRa + GPS)	1.213028043	0.925595	0.715049783	0.827107826
Prueba (Sticker)	1.3	460.00395	460.5	159.55
Clustering Substractivo				
Entrenamiento (LoRa + GPS)	1.213028043	0.767676957	0.689180843	0.827107826
Prueba (Sticker)	1.3	460	460.1155	159.55

4.4 Predicción de Latitud y Longitud

En esta sección se presentan los resultados obtenidos para la predicción de latitud y longitud utilizando una red GNN. La Tabla 4.6 resume los mejores hiperparámetros y las métricas de error asociadas a cada uno de los enfoques de datos utilizados: **No Balanceada**, **Balanceada**, **Aumentada**, y **Aumentada Gateway "itaca-upv-022"**. Estas métricas incluyen el **Promedio de Distancia** (distancia euclidiana promedio), **RMSE** (Root Mean Square Error) y **MAE** (Mean Absolute Error), que cuantifican la precisión de las predicciones realizadas.

Para evaluar cómo la calidad y cantidad de los datos influyen en el desempeño de la red GNN, el modelo fue entrenado utilizando diferentes versiones de los datos recopilados. Inicialmente, se emplearon los datos originales recolectados en los 15 puntos de medición. Estos datos incluyen información del gateway de interiores "main-gtw-grc", con cobertura en los puntos 1 al 6 debido a su alcance limitado, y del gateway de exteriores "itaca-upv-022", cuya cobertura abarca los puntos 1 al 15 y cubre la totalidad del área de interés. Posteriormente, se aplicó la metodología de balanceo de datos en los 15 puntos medidos, utilizando para cada punto el mejor modelo ajustado a la distribución específica de los datos según el gateway correspondiente, y se procedió con el entrenamiento de esta data.

En una tercera etapa, se combinaron los datos originales desbalanceados de los 15 puntos con los datos generados mediante la metodología de aumento de datos propuesta para puntos no medidos. Para este propósito, se empleó el mejor enfoque identificado para cada gateway: un modelo mixto (Intercepto Flotante + VBGMM) para "itaca-upv-022" y un modelo basado

en datos (regresión) para "main-gtw-grc". Finalmente, se realizó un experimento adicional en el que solo se incluyeron los datos aumentados para puntos no medidos correspondientes al gateway "itaca-upv-022", aprovechando su mayor cantidad de datos recolectados para el desarrollo de su método de aumento de datos. Esta decisión buscó evitar la introducción de ruido al incluir predicciones para "main-gtw-grc", que cuenta con una menor cantidad de datos disponibles en el desarrollo de su método.

Se puede observar que el enfoque Aumentada Gateway "itaca-upv-022", presenta el menor error en términos de las tres métricas evaluadas. Este modelo logró un Promedio de Distancia de 0.004379, un RMSE de 0.004810 y un MAE de 0.002692, lo que evidencia su mayor capacidad para predecir con precisión las coordenadas de latitud y longitud. Esto puede atribuirse a la configuración óptima de hiperparámetros, donde se utilizaron finalmente 20 épocas de entrenamiento y un learning rate fino (0.000202), lo que sugiere que este enfoque es menos propenso al sobreajuste. Adicionalmente, el modelo se entrenó con una dimensión oculta (hidden dim) de 413 neuronas.

Tabla 4.5: Mejores Hiperparámetros y Métricas por para Cada Aproximación de Data

Métrica	**No Balanceada**	**Balanceada**	**Aumentada**	**Aumentada Gateway**
Hidden Dim	483	355	288	413
Learning Rate	0.001878	0.009841	0.000262	0.000202
Epochs	100	100	100	20
Promedio Distancia	0.011062	0.004547	0.004830	0.004379
RMSE	0.011145	0.004908	0.005639	0.004810
MAE	0.006467	0.002798	0.002916	0.002692

Tabla 4.6: Mejores Hiperparámetros y Métricas por para Cada Aproximación de Data red 2 capas

Métrica	**No Balanceada**	**Balanceada**	**Aumentada**	**Aumentada Gateway**
Hidden Dim	403	427	445	386
Learning Rate	0.000540	0.000572	0.000199	0.000300
Epochs	20	20	20	100
Promedio Distancia	0.026063	0.031767	0.004354	0.004358
RMSE	0.026159	0.032047	0.004793	0.004763
MAE	0.018156	0.017260	0.002690	0.002691

Las Tablas 4.7 y 4.8 muestra una comparación detallada entre las coordenadas reales y las predicciones realizadas por cada modelo GNN con 1 y 2 capas respectivamente. Para el enfoque **Aumentada Gateway "itaca-upv-022"**, las predicciones se alinean mejor con los valores reales en comparación con los otros enfoques. Las predicciones correspondientes a los enfoques **No Balanceada** y **Balanceada** presentan desviaciones significativas de las coordenadas reales. Las tablas 4.9 y 4.10 muestra en metros la distancia del error entre las coordenadas reales y los puntos inferidos por cada uno de los modelos de GNN.

Tabla 4.7: Coordenadas Reales y Comparación de Predicciones para Cada Aproximación de Data,Red GNN con 1 capa

Reales		No Balanceada		Balanceada		Aumentada		Aumentada Gateway	
Latitud	Longitud	Latitud	Longitud	Latitud	Longitud	Latitud	Longitud	Latitud	Longitud
39.48266983	-0.34633100	39.48001862	-0.33449104	39.48103333	-0.34017596	39.48056030	-0.34051976	39.48065567	-0.34353796
39.48075104	-0.34716401	39.48001862	-0.33449104	39.48103333	-0.34017608	39.48056030	-0.34051973	39.48065567	-0.34353796
39.48308563	-0.34297001	39.48002243	-0.33449027	39.48102951	-0.34017608	39.48055649	-0.34051976	39.48065567	-0.34353796
39.48170090	-0.34356800	39.48001862	-0.33449057	39.48103333	-0.34017608	39.48056030	-0.34051988	39.48065567	-0.34353796
39.48006058	-0.34491000	39.48001862	-0.33449045	39.48103333	-0.34017596	39.48056030	-0.34051976	39.48065567	-0.34353796

Tabla 4.8: Coordenadas Reales y Comparación de Predicciones para Cada Aproximación de Data, Red GNN con 2 capas

Reales		No Balanceada		Balanceada		Aumentada		Aumentada Gateway	
Latitud	Longitud	Latitud	Longitud	Latitud	Longitud	Latitud	Longitud	Latitud	Longitud
39.48266983	-0.34633100	39.50023270	-0.32465118	39.48384476	-0.30975023	39.48056793	-0.34074286	39.48064423	-0.34017739
39.48075104	-0.34716401	39.50023270	-0.32465148	39.48384094	-0.30975023	39.48056793	-0.34074286	39.48064423	-0.34017742
39.48308563	-0.34297001	39.50023270	-0.32465184	39.48384094	-0.30975011	39.48056793	-0.34074286	39.48064423	-0.34017739
39.48170090	-0.34356800	39.50023270	-0.32465130	39.48384476	-0.30975047	39.48056412	-0.34074280	39.48064423	-0.34017715
39.48006058	-0.34491000	39.50023270	-0.32465178	39.48384476	-0.30975023	39.48056793	-0.34074286	39.48064423	-0.34017739

Tabla 4.9: Errores en metros entre puntos reales y predicciones para cada aproximación de datos en metros con Red GNN de 1 capa

Punto	No Balanceada	Balanceada	Aumentada	Aumentada Gateway
1	1058.00	558.77	551.10	328.05
2	1090.70	600.60	570.60	311.40
3	803.50	331.30	351.16	274.56
4	798.70	493.60	290.70	116.25
5	894.30	420.45	380.90	135.07

Tabla 4.10: Errores en metros entre puntos reales y predicciones para cada aproximación de datos en metros con Red GNN de 2 capas

Punto	No Balanceada	Balanceada	Aumentada	Aumentada Gateway
1	2687.15	3146.80	534.20	575.30
2	2720.90	3189.45	553.75	558.40
3	2515.60	2987.20	521.10	540.85
4	2598.30	3055.90	499.65	526.70
5	2642.45	3102.15	512.80	562.10

La Figura 4.7 muestra la inferencia de los diferentes modelos para un solo punto, donde puede observar que el enfoque **Aumentada Gateway "itaca-upv-022"** presenta una mejor aproximación respecto al punto real, mientras que los otros enfoques tienden a dispersarse más.

Figura 4.7: Inferencias de coordenadas de latitud y longitud de las diferentes redes de los diferentes métodos de Aproximación de Data.

Los resultados obtenidos en este estudio mostraron que el mejor desempeño en la predicción de coordenadas para puntos no vistos se obtuvo al utilizar el cuarto enfoque, que integra la metodología de aumento de datos para el gateway "itaca-upv-022" únicamente. Este resultado destaca el potencial de mejora del modelo a medida que se incrementan los datos utilizados para entrenar la metodología propuesta, evidenciando que una mayor cantidad de datos en el desarrollo de la metodología mejora las predicciones de tal manera que permite una adecuada generación y balanceo de datos para optimizar el desempeño del modelo GNN.

Ademas demuestran que una Red Neuronal Basada en Grafos (GNN) con una sola capa es más efectiva para la predicción de coordenadas geográficas a partir de metadatos de RSSI en redes LoRaWAN, en comparación con una arquitectura de mayor profundidad. Se observó que la GNN de 1 capa logró errores significativamente menores, con valores que oscilan entre 116.25 m en la aproximación Aumentada Gateway y 1090.70 m en la No Balanceada. En contraste, la GNN de 2 capas mostró un incremento considerable en los errores, alcanzando hasta 3189.45 m en la aproximación Balanceada, lo que sugiere una pérdida de capacidad de generalización.

El análisis sugiere que al aumentar la cantidad de capas en la arquitectura de la GNN, el modelo introduce ruido en lugar de mejorar la captura de relaciones espaciales relevantes. En particular, las aproximaciones No Balanceada y Balanceada experimentaron un incremento del error en un factor de 2.5 a 7.5 veces, indicando que la mayor profundidad de la red podría estar capturando patrones irrelevantes en los datos de entrenamiento, lo que perjudica la precisión de la inferencia. Esto es coherente con la naturaleza del problema, ya que la relación entre RSSI y coordenadas geográficas no requiere una representación espacial altamente compleja, por lo que una arquitectura más simple es suficiente para modelar la relación señal-posición sin incurrir en sobreajuste.

5

Conclusión

El enfoque basado en teoría y el enfoque mixto superaron a los modelos basados únicamente en datos, a pesar de que estos últimos cuentan con una mayor complejidad estructural. Si bien esta complejidad les permite adaptarse mejor a los datos y obtener buenos resultados durante el entrenamiento, su rendimiento disminuyó considerablemente al ser evaluados en puntos no conocidos. Por el contrario, los enfoques teóricos y mixtos, que integran en su estructura el conocimiento físico del decaimiento logarítmico de las ondas durante su propagación, demostraron un mejor desempeño en la predicción de valores en puntos no observados, aprovechando este conocimiento adicional para una generalización más precisa.

No obstante, el rendimiento del enfoque basado únicamente en datos en escenarios con una cantidad limitada de información puede ser aprovechado, ya que su estructura es más adecuada para situaciones donde la variable de interés tiene un carácter más determinista y menos aleatorio, como podría ser en contextos diferentes a la predicción de valores RSSI.

En este sentido, en escenarios donde los datos son escasos, los modelos basados en teoría tienden a ofrecer un desempeño más consistente, ya que dependen menos de las características estadísticas de los datos y más de principios matemáticos o físicos bien fundamentados. Por otro lado, los enfoques mixtos, como los basados en KDE y VBGMM, destacan al combinar los beneficios de los modelos teóricos con la capacidad de los modelos basados en datos para capturar patrones específicos. Estas mejoras son especialmente evidentes cuando los datos son agrupados mediante procesos de *clusterización*, ya que los modelos mixtos logran adaptarse mejor a escenarios con ruido o datos residuales, integrando información estructurada y variabilidad inherente de manera efectiva. Esto refuerza su utilidad en contextos donde las características de los datos son más complejas o menos deterministas, como en la predicción de valores RSSI.

La clusterización mostró una influencia positiva en la predicción final al considerar todos los enfoques propuestos para ambos gateways. Su implementación reveló un patrón de mejora en las métricas evaluadas tanto durante el entrenamiento como en las pruebas. Sin embargo, esta influencia puede tornarse negativa dependiendo de la cantidad y distribución de los datos utilizados, tanto en el entrenamiento como en la prueba en puntos no conocidos. Esto se debe a la limitada y desbalanceada cantidad de datos disponibles por punto al momento de realizar la agrupación, lo que lleva a que algunos algoritmos generen clústeres compuestos por datos de un único punto. Esta situación afecta negativamente a modelos de estructura no flexible, como el de intercepto flotante, que requieren información de múltiples puntos para un mejor desempeño, perjudicando enfoques como el basado en teoría y el mixto que dependen de este modelo.

En particular, en los modelos basados en teoría, la métrica de Wasserstein mostró mejoras en los tres tipos de clustering propuestos para el entrenamiento. Sin embargo, con los datos de prueba del gateway "main-gtw-grc", el desempeño no fue

satisfactorio debido a la escasa cantidad de información disponible tras la clusterización. En estos casos, el proceso de ajuste de curvas (curve fitting) del modelo de intercepto flotante se realiza sobre clústeres que a menudo contienen un único punto, equivalente a un solo valor de distancia. Esto provoca que el modelo aprenda a realizar el ajuste de curvas exclusivamente sobre ese único dato, lo que, al predecir valores no conocidos del gateway, como en los datos de prueba, genera predicciones para distancias desconocidas con errores muy altos en la métrica de Wasserstein.

El enfoque de clustering subjetivo propuesto, aunque incluye un procedimiento cuantificable para la categorización, como el conteo de píxeles en imágenes y la verificación de la presencia de elementos como árboles u otras estructuras que interfieren en la transmisión de señales, no logra un desempeño destacado. Aunque este método muestra algunas mejoras en las métricas de ciertos enfoques planteados, su efectividad es inferior a la de algoritmos no supervisados que trabajan directamente sobre los datos sin restricciones predefinidas de categorías. Estos algoritmos aprovechan características subyacentes más complejas en los datos, lo que permite una agrupación más precisa y enriquecida en comparación con clasificaciones simples como "urbano" o "vegetación".

En relación al desempeño de la red GNN, que es nuestra solución para la predicción de latitud y longitud, se demostró que su rendimiento mejora significativamente al implementar la metodología propuesta para el aumento de datos durante el entrenamiento. Un mayor volumen de datos iniciales recolectados refuerza los modelos desarrollados dentro de la metodología, permitiendo obtener inferencias más precisas. El mejor desempeño se alcanzó al aplicar la metodología de aumento de datos exclusivamente al gateway "itaca-upv-022", cuyo modelo inicial contaba con una mayor cantidad de datos, optimizando la generación y el balanceo de datos. En contraste, la incorporación de modelos asociados al gateway "main-gtw-grc", con datos limitados, introdujo ruido en el entrenamiento, lo que afectó negativamente el desempeño de la red. Estos hallazgos subrayan la importancia de una base de datos robusta y balanceada para maximizar la efectividad de la solución basada en GNN.

Los resultados muestran que una GNN de 1 capa es más efectiva que una de 2 capas para predecir coordenadas geográficas a partir de datos de RSSI en redes LoRaWAN. La GNN de 1 capa alcanzó errores entre 116.25 m (Aumentada Gateway) y 1090.70 m (No Balanceada), mientras que la de 2 capas mostró errores significativamente mayores, llegando hasta 3189.45 m (Balanceada), lo que sugiere una pérdida de generalización al aumentar la profundidad. Esto indica que la relación entre RSSI y posición no requiere una arquitectura más compleja de GNN, y que agregar más capas introduce ruido en lugar de mejorar el rendimiento. En este contexto, la GNN de 1 capa equilibra simplicidad y precisión, evitando el sobreajuste observado en redes más profundas.

Referencias

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[3] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: The rising stars in the iot and smart city scenarios," *IEEE Wireless Communications*, vol. 23, no. 5, pp. 60–67, 2016.

[4] B. Reynders, W. Meert, and S. Pollin, "An overview of lorawan," *LPWAN overview*, vol. 34, no. 6, pp. 1–7, 2018.

[5] A. Augustin, J. Yi, T. H. Clausen, and W. M. Townsley, "A study of lora: Long range & low power networks for the internet of things," *Sensors*, vol. 16, no. 9, p. 1466, 2016.

[6] J. Smith and R. Taylor, "Geolocation in lorawan networks using neural network approaches," *IoT Journal*, vol. 5, no. 3, pp. 30–45, 2022.

[7] "The things network: Open lorawan networks for iot," 2019. [Online]. Available: https://www.thethingsnetwork.org

[8] L. Cloud. (2025) Geolocation documentation. Accessed: 2025-01-28. [Online]. Available: https://www.loracloud.com/documentation/geolocation?url=v3.html

[9] A. Moradbeikie, A. Keshavarz, H. Rostami, S. Paiva, and S. I. Lopes, "Improvement of rssi-based lorawan localization using edge-ai," in *International Summit Smart City 360°*. Springer, 2021, pp. 140–154.

[10] Z. Liu, J. Liu, X. Xu, and K. Wu, "Deepgps: Deep learning enhanced gps positioning in urban canyons," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 376–392, 2024.

[11] L. e. a. Zhang, "Graph neural networks for wireless network topology optimization," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3951–3963, 2021.

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

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, vol. 27, 2014, pp. 2672–2680.

[14] D. S. Garcia-Chicangana, O. S. López-Erazo, C. González, and J. Muñoz, "Context-aware ubiquitous and mobile learning systems: research gaps and challenges," *International Journal of Technology Enhanced Learning*, vol. 13, no. 3, pp. 309–324, 2021.

[15] E. López-Rubio, E. J. Palomo, and F. Ortega-Zamorano, "Unsupervised learning by cluster quality optimization," *Information Sciences*, vol. 436, pp. 31–55, 2018.

[16] L. Arco, G. Casas, and A. Nowé, "Clustering methodology for smart metering data based on local and global features," in *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*. ACM, 2017, pp. 1–8.

[17] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers*. John Wiley & Sons, 2021.

[18] R. V. Hogg, J. McKean, and A. T. Craig, *Introduction to Mathematical Statistics*. Pearson, 2019.

[19] N. A. Weiss, *Introductory Statistics*. Pearson, 2012.

[20] D. Joanes and C. Gill, "Comparing measures of sample skewness and kurtosis," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 47, no. 1, pp. 183–189, 1998.

[21] J. C. Bezdek, "Fuzzy mathematical programming with applications to engineering and management science," *Computers & Geosciences*, vol. 10, no. 3-4, pp. 191–203, 1984.

[22] F. Höppner, F. Klawonn, R. Kruse, and T. Runkler, "Fuzzy cluster analysis: Methods for classification, data analysis and image recognition," *Fuzzy Sets and Systems*, vol. 112, no. 3, pp. 315–319, 1999.

[23] S. Chiu, "Clustering for approximation of nonlinear systems using fuzzy sets," *Proceedings of 1994 IEEE International Conference on Fuzzy Systems*, pp. 94–98, 1994.

[24] ——, "Extracting fuzzy rules from data for function approximation and pattern classification," *Fuzzy Information Engineering: A Guided Tour of Applications*, vol. 10, pp. 54–72, 1997.

[25] R. Xu and D. C. Wunsch, *Clustering*. Wiley-IEEE Press, 2008.

[26] V. Estivill-Castro and I. Yang, "Subjective and objective measures of cluster validity," *Pattern Recognition Letters*, vol. 16, no. 8, pp. 1145–1157, 1995.

[27] P. Avila-Campos, F. Astudillo-Salinas, A. Vazquez-Rodas, and A. Araujo, "Evaluation of lorawan transmission range for wireless sensor networks in riparian forests," in *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWIM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 199–206. [Online]. Available: https://doi.org/10.1145/3345768.3355934

[28] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, 1997.

[29] C. E. Rasmussen and C. K. Williams, *Gaussian Processes for Machine Learning*. MIT press, 2006.

[30] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in Neural Information Processing Systems*, 2008.

[31] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[32] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[34] M. Newman, *Networks*. Oxford University Press, 2018.

[35] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2017. [Online]. Available: https://arxiv.org/abs/1609.02907

[36] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, *Geometric deep learning: Grids, groups, graphs, geodesics, and gauges*. arXiv preprint arXiv:2104.13478, 2021. [Online]. Available: https://arxiv.org/abs/2104.13478

[37] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in Neural Information Processing Systems*, vol. 30, 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html

[38] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.