On the application of Graph Neural Networks for Indoor Positioning Systems

Facundo Lezama, Federico Larroca and Germán Capdehourat

Abstract Due to the inability of GPS (or other GNSS methods) to provide satisfactory precision for the indoor location scenario, indoor positioning systems resort to other signals already available on-site, typically Wi-Fi given its ubiquity. However, instead of relying on an error-prone propagation model as in ranging methods, the popular fingerprinting positioning technique considers a more direct data-driven approach to the problem. First of all, the area of interest is divided into zones, and then a machine learning algorithm is trained to map between, for instance, power measurements (RSSI) from APs to the localization zone, thus effectively turning the problem into a classification one. However, although the positioning problem is a geometrical one, virtually all methods proposed in the literature disregard the underlying structure of the data, using generic machine learning algorithms. In this chapter we consider instead a graph-based learning method, Graph Neural Networks, a paradigm that has emerged in the last few years and that constitutes the state-of-the-art for several problems. After presenting the pertinent theoretical background, we discuss two possibilities to construct the underlying graph for the positioning problem. We then perform a thorough evaluation of both possibilities, and compare it with some of the most popular machine learning alternatives. The main conclusion is that these graph-based methods obtain systematically better results, particularly with regards to practical aspects (e.g. gracefully tolerating faulty APs), which makes them a serious candidate to consider when deploying positioning systems.

Facundo Lezama
Facultad de Ingeniería, *Universidad de la República*, Montevideo, Uruguay, e-mail: facundo.lezama@fing.edu.uy

Federico Larroca
Facultad de Ingeniería, *Universidad de la República*, Montevideo, Uruguay, e-mail: flarroca@fing.edu.uy

Germán Capdehourat
Facultad de Ingeniería, *Universidad de la República*, Montevideo, Uruguay, e-mail: gcapde@fing.edu.uy

Key words: Graph classification; Graph signal interpolation; Localization

1 Introduction

The key enabler for location-based services is naturally an accurate positioning system. Although for outdoor scenarios GNSS systems (e.g. GPS) are generally enough, the signal is not strong enough to provide a sufficient precision for indoor applications. The resulting error is typically of tens of meters, barring its usage on museum self-guided tours, customer navigation in shopping malls, or to provide accessibility to visually impaired people, just to name a few examples of important location-based services (Küpper, 2005).

Several approaches have been proposed to address this problem, which basically consider other signals to infer the position of the mobile device. For instance, the received power or the Channel State Information from a set of Wi-Fi, Bluetooth, Ultra Wide-Band or radio-frequency identification tags (RFID) transmitters with a known and fixed location (thus generally known as anchor nodes) may be used to this end, constituting the so-called *ranging techniques* (Whitehouse et al., 2007). These are generally model-based, requiring in turn a precise model that relates the position of the mobile to the received power, a model which is generally unavailable.

Instead of collecting a large set of measurements to derive a channel model, the so-called *fingerprinting technique* takes a more data-driven approach to the problem at hand: directly learning to map, for instance, the received powers (from the anchor nodes) to the position of the mobile, transforming the problem into a regression one (Yiu et al., 2017). In fact, depending on the final application, the actual coordinates of both the anchor nodes and the mobile device may actually be unnecessary (or even unavailable). For instance, we may want to identify at what shop is a certain customer of a shopping mall, and not the precise coordinates. In this case, the problem turns into a classification one. That is to say, the area is divided into zones and the objective is, given the power measurements from the anchor nodes, inferring at which zone the mobile device is. Note that in both cases a measurement campaign is still necessary in order to train a learning algorithm to provide this mapping.

Interestingly, the vast majority of the existing fingerprinting techniques consider vector-based learning algorithms, ignoring the geometric nature of the problem (Zafari et al., 2019). Following our ongoing example, if the number of anchor nodes is n, then the input to the learning algorithm may be a vector in $\mathbb{R}^n$ (the power received from each anchor node), thus dropping the spatial information, which is to be inferred again by the learning algorithm. This will limit the generalization power of the resulting method. To illustrate the importance of considering the structure of the data, suffice to say that it is one of the main reasons behind the success of Convolutional Neural Networks (CNNs) for image and audio processing.

In what follows, we will assume we are using Wi-Fi Access Points (APs) as anchor nodes and the measured power (Received Strength Signal Indicator, RSSI) from all

APs as the input to the learning algorithm, which will map this input to a zone (i.e. a classification problem). Extending the proposed methodology to other technologies, inputs or to considering a regression problem is straightforward.

In this chapter, we study how to take into account the geometric information of the problem as an *a priori* on the learning algorithm. To this end, we will define a graph with APs as its nodes, where the existence of an edge and its weight is indicative of the distance between each pair of APs (e.g. the mean RSSI between each pair of APs). A given input (i.e. the RSSI of each AP as measured by the mobile node) is now actually a number associated to each node on the graph (or a two-dimensional vector, if the APs are dual-band), thus defining a signal on the graph. The proposed classifier is based on Graph Neural Networks, which basically extend the concept of CNNs to signals defined on graphs (Gama et al., 2019).

We will introduce two versions of the learning algorithm. In the first one the graph signal is mapped to a zone. In a nutshell, we will transform the positioning problem into a graph classification one. By means of two public datasets we will compare this method to some popular alternatives, namely K-Nearest Neighbors, a Fully Connected Neural Net, or the combination of methods used in the indoor positioning software FIND3[1]. Results indicate that the proposed method performs systematically better in terms of accuracy. For instance, in one of the datasets it performed above KNN using less than half of the training samples.

These virtues notwithstanding, this first method's main drawback is that it considers the geometry between APs only, ignoring the zones. We may include the zones on the graph, resulting in a so-called *heterogeneous* graph. In particular, although we may define the same kind of edges (e.g. mean RSSI from each AP to each zone), the signal on nodes corresponding to zones is clearly not of the same kind as the one on each AP (they may even be of different dimensions). In any case, the problem now turns into a graph signal interpolation one, where given the signal on the nodes corresponding to APs, we want to estimate a probability distribution over the nodes corresponding to zones. This method obtained similar results than the homogeneous case, with the addition to showing much better robustness in the scenario where one or several APs fail. It is important to note that neither of the methods require a map to construct the graphs, and we resort to the training dataset only.

The document is organized as follows. The next section presents the literature on indoor localization, focusing on the use of graphs. In section 3 we formally state the fingerprinting positioning technique, we present Graph Neural Networks in detail and discuss how to use it on the positioning problem. Finally, section 4 describes and analyzes the experiments carried out before concluding the chapter in section 5.

[1] https://github.com/schollz/find3

2 Related Work

The problem of indoor localization is a widely studied topic and different methods and technologies have been used to address it. In (Basri and El Khadimi, 2016) technologies such as the use of infrared sensors, ultrasound sensors, cameras and their subsequent processing, radio frequency technologies such as Radio Frequency Identification (RFID) and methods based on Wireless Local Area Network (WLAN) are presented. The latter has gained great interest in recent years due to the growth of Wi-Fi network deployments. The goal is to take advantage of the existing wireless connectivity infrastructure to solve the localization.

As we mentioned before, indoor positioning based on WLAN infrastructure is usually done by means of power measurements from the different network APs. Since it is not easy to fit a propagation model that accurately estimates the signal received at each point, fingerprint-based techniques have become the most popular approach to the problem (Zafari et al., 2019). In this case, the RF propagation model is not taken into account directly, but through RSSI measurements from mobile devices. This data-driven approach converts the problem into a machine learning one, turning the position estimation into a regression problem. It can be further simplified if, instead of the exact position, the goal is only to estimate the area where the mobile device is located, reducing the problem to a classification one.

One of the most used methods to address this problem is K-Nearest Neighbors (KNN), due to its simplicity and good results (Bahl and Padmanabhan, 2000; Varshavsky et al., 2007). Other machine learning techniques used for this problem include Deep Learning (Nowicki and Wietrzykowski, 2017), Random Forests, Support Vector Machine or Multi Layer Perceptron (MLP) (Bracco et al., 2020). However, all of them are vector-based learning algorithms, where the model simplification ignores the geometric nature of the position estimation problem, particularly the relationship between the RSSI measurements at a certain point from the different APs.

A novel machine learning approach which we will explore here and enables to take into account the geometric information as a model prior are Graph Neural Networks (GNNs). For this purpose, a graph is defined with the network APs as nodes, and the edges weights should reflect the distance between each pair of APs. As detailed later on, in this model the RSSI values measured by the mobile devices will be signals defined on the graph. Although there are still not many indoor localization works based on GNNs in the literature yet, some promising results are already beginning to show that it is a successful approach to address the problem (Yan et al., 2021; Lezama et al., 2021).

In (Yan et al., 2021) a GNN based method is presented, analyzing a simulated scenario where the distance between nodes is modeled with noisy values of the Euclidean distance between them. Results with a real dataset like (Torres-Sospedra et al., 2014) (which we present later in this chapter) can be found in (Sun et al., 2021), which uses a graph convolutional network (GCN) to address the problem. Two different ways to build the graph are discussed, an important issue that will be explained in Section 4. A real world implementation is presented in (Ding et al.,

2022), for on-demand delivery on multi-floor malls. Another advantage of the graph based approach discussed in (Chen and Chang, 2022) is the possibility of using transfer learning, adapting a previously trained network with data from other location, thus requiring less number of measurements to train the model.

Finally, it is worth highlighting other variants of the problem, which are different applications where the GNNs-based approach is still very useful. On the one hand, (Chiou et al., 2020) studies the case where the signals are not measurements of the power of a Wi-Fi network, but images from multiple cameras of the mobile device. It is clear that the graph model (named Graph Location Networks in this context) still suits to the problem, in this case modeling the relationship between images taken at different locations. Another problem extension is addressed by (Lin et al., 2021), which uses a GNN to estimate the device next location, taking advantage of the graph to model the trajectories given by the device locations sequence.

3 GNNs for Indoor Localization

3.1 Problem Statement

As we mentioned before, in the fingerprinting approach the localization problem is usually turned into a classification one: given the RSSI measurements of the n_{AP} APs as received by the device, the objective is to learn how to map these values to the corresponding zone. Let us denote by $\mathbf{X} \in \mathbb{R}^{n_{AP} \times F_{in}}$ one of these measurements, where for instance $F_{in} = 2$ when measurements for both the 2.4 and 5 GHz bands are available. We will use $\mathbf{x}_i \in \mathbb{R}^{F_{in}}$ to indicate the i-th row of $\mathbf{X}$, corresponding to the RSSI measurement from AP i (with a default value of, for instance, -100 dBm in case this particular AP's RSSI was below the sensitivity of the device). Given n_z possible zones, we want to estimate the parameters of a function $\mathbf{\Phi} : \mathbb{R}^{n_{AP} \times F_{in}} \rightarrow \{1, \ldots, n_z\}$ that minimizes a certain loss over the available training set.

We remark again that if other type of measurements are available (such as Channel State Information), they may easily be included on the framework by modifying the definition of $\mathbf{x}_i$ accordingly. Moreover, if we were interested in the actual coordinates of the mobile node, we would simply change the codomain of the function $\mathbf{\Phi}$ and consider a regression problem; i.e. we would have $\mathbf{\Phi} : \mathbb{R}^{n_{AP} \times F_{in}} \rightarrow \mathbb{R}^3$ and the cost function would for instance be the Mean Squared Error. In any case, the family of functions $\mathbf{\Phi}$ typically chosen (e.g. a Neural Network) does not consider at all the underlying structure of the problem, which is expected to be learned from the training set instead. Here we consider an alternative approach, where the geometric information is provided *a priori* by means of a graph.

3.2 Graph Neural Networks

Learning from data represented by graphs or networks has received considerable attention over the last few years. Applications range from the analysis of social networks (Wang et al., 2015) to predicting properties of chemical compounds (Coley et al., 2019). The most important challenge is that, differently to for instance audio or images, graph data is highly irregular and non-euclidean.

Some of the first works to propose Graph Neural Networks, which basically try to emulate the success obtained by CNNs onto the graph domain, represent node i (for $i = 1, \ldots, n$) by a vector $\mathbf{x}_i \in \mathbb{R}^d$, which is iteratively updated by combining it with its neighbors' vector representation (Scarselli et al., 2009). After a number of iterations, a final vector representation of each node is obtained, which is then used to, for instance, node classification.

An alternative approach to the problem was provided by taking a Graph Signal Processing perspective (Shuman et al., 2013). In particular, a spectral-based graph convolution was first considered (Estrach et al., 2014), an operation that is extremely costly and numerically unstable. To remedy this, a Chebyshev polynomial on the Laplacian matrix was first proposed to approximate the spectral convolution (Defferrard et al., 2016).

However, considering the analogous to discrete-time convolution, a first-order convolution layer for a GNN may be obtained as follows (Kipf and Welling, 2017):

$$\mathbf{x}'_i = \sigma\left(\mathbf{\Theta}^T \sum_{j \in \mathcal{N}_i \cup \{i\}} S_{j,i}\mathbf{x}_j\right), \tag{1}$$

where $\mathbf{x}'_i \in \mathbb{R}^{d'}$ is the output of the layer, $\sigma(\cdot)$ is a point-wise non-linearity (e.g. the ReLU function), $\mathbf{\Theta} \in \mathbb{R}^{d \times d'}$ is the learneable parameter of this layer, $\mathcal{N}_i$ is the set of neighbors of node i, and $S_{i,j}$ is the i, j entry of matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$, the so-called Graph Shift Operator (GSO). This is a matrix representation of the graph, which should respect its sparsity (i.e. $S_{i,j} \neq 0$ whenever there is an edge between nodes i and j). The adjacency matrix of the graph, its Laplacian or their normalized versions are all valid GSOs. Moreover, larger values of $S_{j,i}$ means that $\mathbf{x}_j$ will have more weight on Equation (1), and thus should be indicative that the signal on nodes i and j are more related to each other.

To grasp the rationale behind Equation (1) note that a discrete-time signal may be represented by a linear graph (successive samples are joined by an edge). If $d = d' = 1$, we would be linearly combining the previous and current samples, and then evaluating this with function $\sigma(\cdot)$. For a general graph we would be linearly combining the vector representation of the node's neighbors. As we concatenate K such layers, the final vector representation of node i will depend on its neighbors at most K hops away.

Let us now consider matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, which basically stacks all nodes' vectors $\mathbf{x}_i$, and is termed a graph signal. Computing the matrix product $\mathbf{SX} = \mathbf{Y}$ we end up with another graph signal that aggregates at each node the information of its neighbors,

corresponding to the first-order convolution we used in Equation (1) (albeit without parameter $\mathbf{\Theta}$, which we will include shortly). By writing $\mathbf{S}^K\mathbf{X} = \mathbf{S}(\mathbf{S}^{K-1}\mathbf{X})$ we may see that this way we aggregate the information K hops away. A general graph convolution is defined simply as a weighted sum of these K signals (i.e. $\sum_k \mathbf{S}^k\mathbf{X}h_k$, where scalars h_k are the taps of the filter). Notice that parameter $\mathbf{\Theta}$ in Equation (1) is now interpreted as a filter bank. Indeed, by considering a $d \times d'$ matrix $\mathbf{H}_k$ instead of the scalar taps, a single-layer GNN (or graph perceptron) results of applying a pointwise non-linear function $\sigma(\cdot)$ to this convolution (Gama et al., 2019; Isufi et al., 2021):

$$\mathbf{X}' = \sigma\left(\sum_{k=0}^{K-1} \mathbf{S}^k\mathbf{X}\mathbf{H}_k\right), \tag{2}$$

whereas a deep GNN is constructed by concatenating several perceptrons.

In addition to having empirically been shown to provide state-of-the-art performance in a number of important problems (Zhou et al., 2020), GNN's theoretical properties have been intensively studied during the last few years. Important to our problem at hand, are their permutation equivariance (i.e. the output signal on each node is independent of the nodes' ordering), stability (i.e. small perturbations on the graph's edges lead to small perturbations on the output graph signal) and transferability (i.e. one may actually train in a small graph, and as long the statistic is similar, the performance should remain the same on a larger one) (Gama et al., 2020; Ruiz et al., 2021).

3.3 Proposed Architectures

3.3.1 Homogeneous Graphs

Let us then go back to our localization problem. As we mentioned before, we will first consider the APs graph. Although the details on how the graph can be constructed for our particular case are included in Section 4.1, suffice to say at this point that each node is an AP, an edge exist between nodes i and j if the received power between them is above a certain threshold, and that the edge's weight will be the corresponding mean RSSI plus this threshold (so as to turn larger weights to more related signals on the nodes).

Over this graph, we will consider the signal we defined before $\mathbf{X} \in \mathbb{R}^{n_{AP} \times F_{in}}$, corresponding to the RSSI measurements for each AP (with $F_{in} = 2$ if measurements for both bands are available). The learning algorithm should then output on what zone (from the n_z possibilities) was that measurement taken. This thus corresponds basically to a graph classification problem.

Graph classification is typically achieved by taking the output of a GNN, passing it through a so-called readout layer (which transforms all nodes signals into a single vector that represents the whole graph) and then applying, for instance, an MLP followed by a softmax (Wu et al., 2021). Although typical readout layers (e.g. the

sum) enforce permutation equivariance (Navarin et al., 2019), it is clear that which AP had a certain representation is important to infer at which zone was the measurement taken. We have thus used as readout a stacking of the node's representation, resulting thus in a vector $\mathbf{y} \in \mathbb{R}^{n_{AP} F_{out}}$, where F_{out} is the last layer's dimension of the GNN. An illustrative example diagram is displayed in Figure 1.

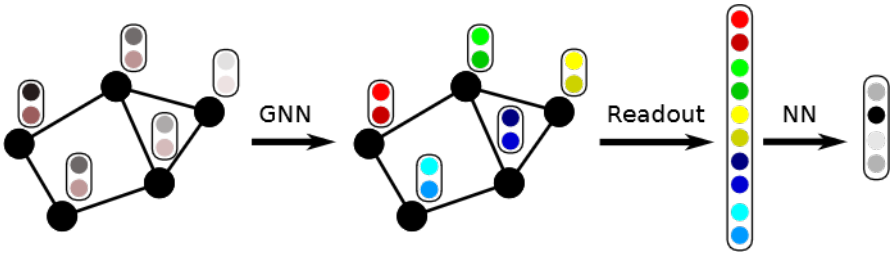


Fig. 1: Illustrative diagram of the graph classification problem for localization. The signal of a graph with $n_{AP} = 5$ dual band APs (and thus each $\mathbf{x}_i$ has $F_{in} = 2$ dimensions) is processed by a GNN which produces a signal with $F_{out} = 2$ dimensions too. The readout layer then stacks all vectors and passes it through a Neural Net which produces a probability distribution over the $n_z = 4$ zones. Note that training can be performed end-to-end.

It may seem that we have returned to a learning algorithm that drops the geometry of the data, as in traditional learning algorithms. Although this is actually true for the zones, a drawback we will address in the following section, just considering the geometry of the APs will have important performance advantages over traditional baselines, as will be further discussed on Section 4.

3.3.2 Heterogeneous Graphs

The question thus remains on how to include the zones' geometry as an *a priori*. Let us then consider that zones are now nodes on the graph. The first thing to note is that we have no input signal associated to a zone. We may for instance associate an arbitrary scalar to all zones or a one-hot encoded vector (Vignac et al., 2020). On the contrary, we have an output signal (e.g. a 1 if the measurement was taken at that zone, or 0 if not) which is not present on the APs' nodes. We will thus consider the output signal only at the zones' nodes to compute the loss.

Furthermore, we have at least two types of edges now: the edges between APs (which we have been considering so far) and between an AP and a zone. Although we may define the latter just as the one we used for the inter-APs edges (i.e. the mean RSSI at that zone from that AP plus a threshold), it is clear that in terms of propagating information they should be considered different. We may even include

edges between zones, considering for instance their distance or if it is possible to transit from one to the other.

In any case, we are in the context of *heterogeneous graphs* (or networks) and learning therein (Dong et al., 2017). The most typical application is in relational data or knowledge graphs, such as academic graphs (where nodes may be papers, authors, conferences, etc.) (Tang et al., 2011) or recommender systems (where nodes may be users and films, as well as actors, directors, country of origin, etc.) (Shi et al., 2019). In our particular case, we want to learn to estimate the zones' signal given the APs' one, thus turning the localization problem into a graph signal interpolation one.

Graph convolution is easily extended to the heterogeneous case. For instance, Equation (1) is now written as (Schlichtkrull et al., 2018):

$$\mathbf{x}'_i = \sigma\left(\mathbf{\Theta}_0^T \mathbf{x}_i + \sum_{r\in\mathcal{R}} \mathbf{\Theta}_r^T \sum_{j\in\mathcal{N}_i^r} S^r_{j,i}\mathbf{x}_j\right), \tag{3}$$

where $\mathcal{R}$ is the set of possible relations (e.g. between APs or between an AP and a zone), $\mathcal{N}_i^r$ are the set of neighbors of node i of relation type r, and $S^r_{j,i}$ is the j,i entry in the GSO if the corresponding relationship is of type r (and 0 else). Note that we now have an specific learning parameter $\mathbf{\Theta}_r$ for each type of relation. An illustrative example diagram of the proposed signal interpolation system is presented in Figure 2.

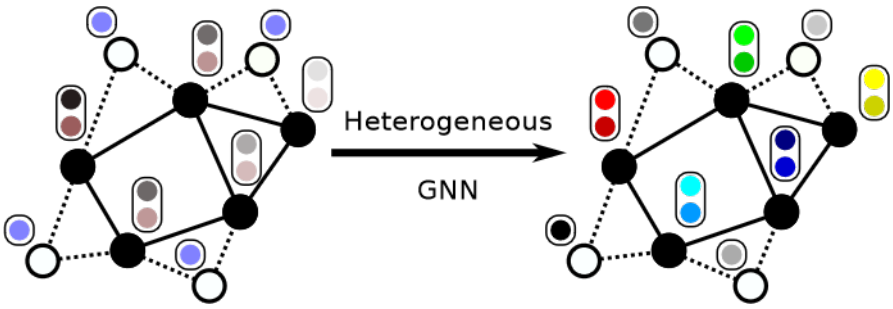


Fig. 2: Illustrative diagram of the positioning system as a graph signal interpolation problem in the heterogeneous context. Similarly to Figure 1, there are $n_{AP} = 5$ dual band APs and $n_z = 4$ zones. The graph now also includes nodes representing these zones (the white ones) and edges connecting them to the APs (the slashed ones), with a weight which may be the mean RSSI as measured in the corresponding zone. The input signal on the APs is the same as before, and we have arbitrarily set all input signals on the zones as a constant. This heterogeneous signal is processed by a (heterogeneous) GNN, which will produce a vector for each AP (which we will ignore in the training cost) in addition to a distribution over the zones' nodes. Note that information is propagated through all nodes and edges.

4 Implementation and Experimental Results

4.1 Graph Construction

Before presenting the results we obtained with the proposed methods, let us discuss how we built the graphs underlying them. Recall that in the graph nodes are the APs (and maybe zones) and the edges' weight should be related to the distance between them. If a map with the position of the APs and zones is available, then we may build the graph by considering the inverse (or some other decreasing function) of the distance between them. Note however that we are using the RSSI from all APs as the graph signal in order to infer the corresponding zone. It may well happen that two points are very near to each other but, for instance, separated by a thick wall, meaning that the RSSI as measured in each point may be significantly different.

It would then be more sensible to use an edge weight related to a "RF distance" between points. To this end, we have used precisely the RSSI between them. If we had access to the area where the positioning system is deployed, we may for instance measure the received RSSI at a given AP from all the rest. However, this was not the case here, as we had access to the datasets only.

We have thus proceeded as follows to construct the inter-AP graph. Given a certain AP indexed by i, all $\mathbf{X}$ in the training set which have measurements for this AP are considered and a certain (large) quantile of the corresponding set $\{\mathbf{x}_i\}$ is used as a threshold. We then consider only the subset of measurements $\mathbf{X}$ for which $\mathbf{x}_i$ is above this value, the rationale being that these should correspond to measurements that were taken near AP i. The edge's weight between APs i and j is simply the mean over this subset of the RSSI for AP j (i.e. the mean of the corresponding $\{\mathbf{x}_j\}$ on this subset).

Note that in the case when there are measurements for both bands (i.e. $F_{in} = 2$) we will obtain two weights per edge. Although this is easily accommodated for the methods we discussed before, results do not change significantly when using either of them or both, so we will focus on the results of using a single weight per edge (the one corresponding to the 2.4 GHz band). Furthermore, in order to work with positive weights, we have subtracted the minimum RSSI value to all measurements as a pre-processing step. Note that this way AP pairs may be disconnected on the graph (with a weight equal to zero), effectively reflecting they are far apart. Finally, note that the resulting graph is not necessarily symmetric.

In the case of the heterogeneous graph, edges between zones and APs can be constructed similarly. We have simply considered as an edge weight the mean RSSI as measured as that zone (minus the minimum RSSI). Furthermore, we have used 0 as the input signal on the nodes corresponding to zones (very small changes were obtained using other alternatives).

4.2 Datasets

In order to be able to compare the obtained results with those from other studies, two datasets were used: MNAV (Bracco et al., 2020) and UJIIndoorLoc (Torres-Sospedra et al., 2014). We now briefly describe them.

4.2.1 UJIIndoorLoc

This dataset can be found at *Kaggle*[2] and was used as the official dataset of the IPIN2015 competition (Torres-Sospedra et al., 2014). This dataset was designed to test WLAN fingerprinting techniques. The data was acquired in three buildings of the University of Jaume I, each with 4 floors or more and covering an area of over 110,000 m^2. In total, it has 19,937 training samples and 1,111 validation/test samples acquired 4 months after the training data. The dataset was collected by more than 20 users using 25 different models of devices. In addition to RSSI measurement from 520 APs, each measurement is labeled with the corresponding building and floor, along with its longitude and latitude (which were not taken into account here).

Note that in this case the positions of the APs are unavailable, highlighting the practical importance of the method to construct the graph we discussed in Section 4.1. Furthermore, it was observed that the columns associated with some APs had very few significant values, so they were not taken into account for the construction of the graph (resulting in 253 APs).

Different definitions of zones may be explored in this case. For instance, a simple and coarse case-scenario is to predict at which building is the device. We have considered the more interesting scenario where we want to predict both the building and the floor (resulting in 13 zones).

4.2.2 MNAV

The second dataset we considered here was created within the framework of (Bracco et al., 2020), which sought to provide an indoor localization system to the *Museo Nacional de Artes Visuales* (MNAV, National Museum of Visual Arts) in Uruguay, using fingerprinting techniques with Wi-Fi. The dataset is available at *Github*[3]. Furthermore, the article includes a map of the museum, including the position of the deployed APs and the 16 areas that were defined.

The dataset has 10,469 measurements from 188 AP addresses, each labeled with the corresponding zone. Inside the museum there are 15 APs, each one using both the 2.4GHz and 5GHz bands, thus defining 30 of the 188 addresses available in the dataset. The rest are APs outside the museum that the devices found while searching for Wi-Fi networks. In this work, only the features corresponding to the APs within

[2] https://www.kaggle.com/giantuji/UjiIndoorLoc

[3] https://github.com/ffedee7/posifi_mnav/tree/master/data_analysis

the museum were used, discarding the rest. We have used an 80%/20% split for training and test purposes.

4.3 Experimental results

Let us now present the experimental results of the proposed methods, along with some popular baselines. Regarding the latter, we have used KNN, a fully connected Neural Net and the combination of methods discussed in (Bracco et al., 2020) (which in turn is a small variation on the combination used in the popular FIND3 software[4]), implemented through Scikit-Learn (Pedregosa et al., 2011). All GNNs were implemented using Pytorch Geometric (now PyG) (Fey and Lenssen, 2019).

Regarding the homogeneous case, the structure of the model consists of two layers with an output dimension of 20, and a filter length of $K = 2$ for UJIIndoorLoc and $K = 3$ for MNAV. In particular, we used the implementation of the architecture proposed in (Du et al., 2017) (cf. Eq. 2). After the readout we used an MLP with the same size as the number of zones (cf. Figure 1). In the heterogeneous case we had to resort to the simpler architecture studied in (Morris et al., 2019), basically a single-tap filter (generalized to heterogeneous graphs in (Schlichtkrull et al., 2018) as presented in Eq. 3), as the more general architecture we used before did not support heterogeneous graphs in PyG. In any case, the final architecture has 4 layers, all with an output dimension of 20 (except, naturally, the last one, which has dimension equal to 1).

All hyper-parameters (including mini-batch sizes, learning rates and weight decay) were obtained through cross-validation. All code generated for the experiments is available at https://github.com/facundolezama19/indoor-localization-gnn.

4.3.1 On the size of the training set

Let us first discuss the overall test accuracy of the different methods on both datasets. Table 1 presents these results. There are three important conclusions that may be drawn from the resulting accuracies. Firstly, the performance of all methods is relatively high, with a minimum accuracy of 87.7% for KNN on the UJIIndoorLoc dataset. Secondly, that the homogeneous GNN systematically performs better than the rest of the methods, particularly in the UJIIndoorLoc dataset. Figure 3 displays the corresponding confusion matrix, where it can be seen that the bigger errors occur when classifying zone 4 as 5 (corresponding to two floors of the second building). Lastly, the heterogeneous GNN presents very competitive results, coming third on the UJIIndoorLoc case (only 0.1% below the FCNN), and second on the MNAV one.

Note that, as stated for example in (Bracco et al., 2020), the fingerprint gathering stage is time-consuming and represents a non-negligible part of the total cost of

[4] https://github.com/schollz/find3

Table 1: Classifier accuracy on both datasets.

Method	**Dataset**	
	UJIIndoorLoc	MNAV
KNN	87.7%	95.8%
FCNN	90.1%	95.9%
(Bracco et al., 2020)	87.9%	96.0%
GNN (homogeneous)	**93.7**%	**97.2**%
GNN (heterogeneous)	89.9%	96.8 %

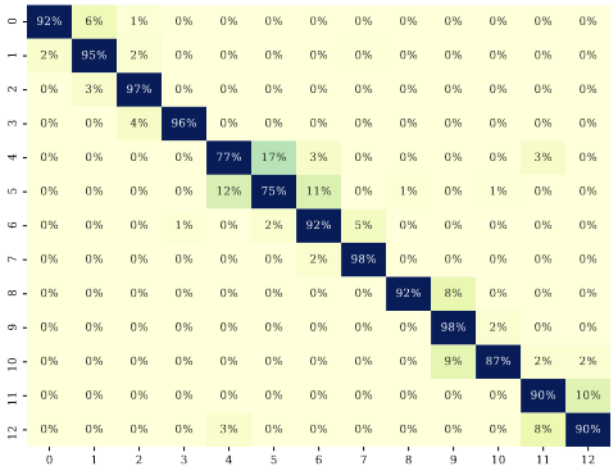
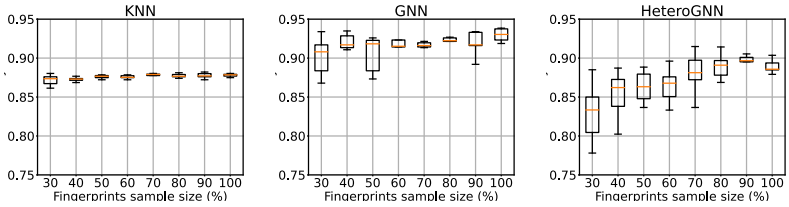


Fig. 3: Confusion matrix obtained by the homogeneous GNN on the UJIIndoorLoc dataset.

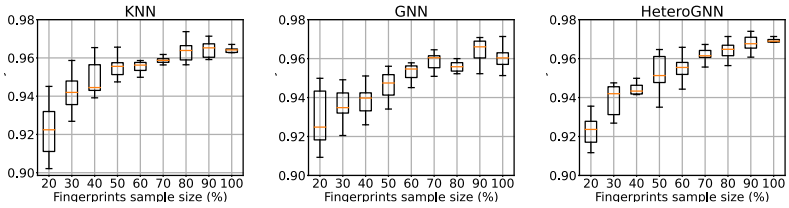
the system. It is then pertinent to evaluate the merits of the different methods in terms of how many training samples they require. To this end, we have considered sub-samples of the training set with varying sizes (uniformly chosen among zones, so that we do not incur in an imbalance) and measured the obtained accuracy on the testing set.

In particular, for each sub-sample's size (measured as a percentage of the original training set's size) we have constructed 10 random training sub-samples and report the test results in Figure 4 in the form of a boxplot for both datasets. For the sake of presentation clarity, as a baseline we are only showing KNN in this figure, since results are similar or worse for the other methods. For instance, this case-study was also carried out in (Bracco et al., 2020) for their method in the MNAV dataset,

resulting in an accuracy as low as 90% when using 30% of the dataset, and 95% when using 70%. For this dataset, all three methods in Fig. 4b obtain a similar performance but using approximately 50% of the samples, and fare well above an accuracy of 90% when using only 30% of the samples. The superiority of the GNN-based method is clearer in the UJIIndoorLoc dataset (Fig. 4a), where we may see that the homogeneous GNN's performance is above that of the KNN, even when using 30% of the samples. On the other hand, the heterogeneous architecture requires roughly 70% to surpass KNN.



(a) UJIIndoorLoc dataset



(b) MNAV dataset

Fig. 4: Accuracy using different amount of fingerprints for KNN (left), homogeneous GNN (middle) and heterogeneous GNN (right). GNN based methods maintain an excellent performance even with very few training samples.

4.3.2 Propagation environment changes

One of the main drawbacks with fingerprint-based localization methods is the need of updating the system in terms of the propagation environment. For instance, if a wall is built it will change the RSSI received at certain zones, thus potentially resulting in a lower performance of the localization method. This is remedied by periodically taking new measurements (or at least when such changes occur), which in turn further increases the cost of these deployments.

In this subsection, we will consider a variant of this scenario, where the RSSI received from a certain AP is lower (by a certain constant) than expected. This is achieved by simply decreasing the RSSI corresponding to this AP only on the test set. This scenario may occur if, after the system is trained and deployed, for instance

the AP configuration is changed and starts transmitting at a lower power, its antennas are rotated, or simply because it is lowered in height.

Figure 5 shows the results for the MNAV dataset when substracting 5 or 10 dBm to the received RSSI of a single AP. Each boxplot represents the resulting 15 accuracies (one for each AP in the dataset). Note how in this case KNN and the one studied in (Bracco et al., 2020) are the methods that obtain the best results, although they are closely followed by the homogeneous GNN. Furthermore, although the heterogeneous graph does perform competitively well in the -5 dBm example, its performance is significantly degraded when the offset is -10 dBm. Finally, it is interesting to highlight that FCNN is the method that performs worst in these examples. This shows the importance of considering the structure of the data in the form of the underlying inter-APs graph.

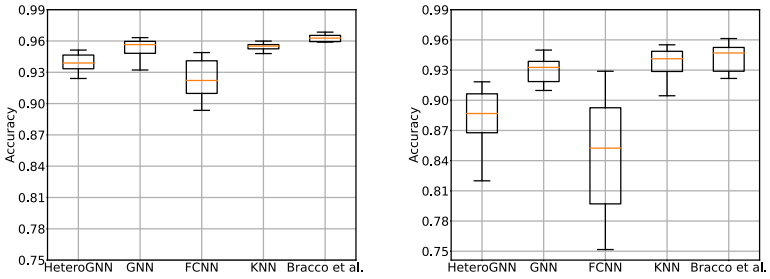


Fig. 5: Accuracy when an offset of -5 dBm (left) or -10 dBm (right) is applied to the received power of one AP during testing only. Although KNN and (Bracco et al., 2020) perform best, the homogeneous GNN method obtains competitive results.

Note that the way we constructed the graph (cf. Sec. 4.1) the tag of which zone correspond to each measurement is not necessary. This means that we may actually update the edges' weight as clients send the measured RSSI (assuming a centralized scheme, where clients send their measurements and the server answers with the estimated zone). A reasonable question is to what point do this updated GSO affects the resulting accuracy in this scenario. Our experiments show a somewhat marginal gain, of approximately 1%.

4.3.3 AP failures

Let us now consider a typical failure scenario, where one or several APs cease operation. This may happen due to a faulty AP or an electrical power problem on the building. In the latter case, it would affect a group of APs, which are not necessarily all of the ones used for localization. Although this type of problems are typically

transient and (should) last for a limited amount of time, it is important to verify if the system still works reasonably well during these problems.

Note that this AP failure case-scenario may be considered as a variation of the one we studied in the previous sub-section, where we are basically dropping to zero the transmitting power of a certain group of APs. Quiet interestingly, results show that differently from that case, KNN now performs much worse.

We will consider that a certain number of APs fails (1, 2 or 3), which is simulated by simply placing the minimum default RSSI on the entries corresponding to the faulty APs on the test dataset. All possible combinations of APs are considered, and results are reported in the form of a boxplot in Fig. 6.

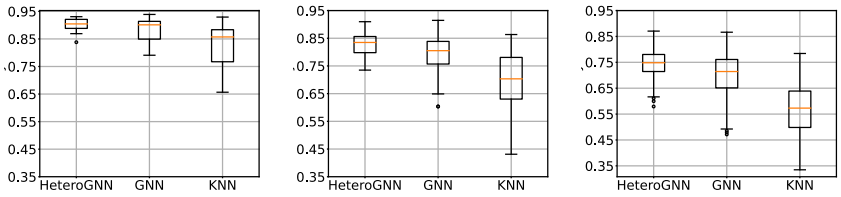


Fig. 6: Test accuracy on the MNAV dataset when a 1 (left), 2 (middle) or 3 (right) APs fail. All possible combinations of faulty APs are considered. GNN-based methods (particularly the heterogeneous one) perform much better than the baseline.

The main conclusion to draw from Figure 6 is that GNN-based methods tolerate much more gracefully AP failures. Whereas with a single AP failure these methods may still obtain over 90% accuracy (and this may even be the case in the two AP failure scenario, depending on which APs fail), this is not at all the case for baselines (only KNN is shown for the sake of clarity of the figure, but similar results were obtained with the other methods). Even in the case of a single failure, results are typically below 90%, whereas with two AP failures they obtain a performance similar to, or even worse than, the GNN-based methods when three APs failed.

5 Conclusions

In this chapter we have explored the possibility of applying graph-based learning methods, Graph Neural Networks (GNNs), to the indoor localization problem. Differently to traditional methods, the rationale was to consider the structure of the data *a priori*. We have presented the necessary theoretical background, and discussed two ways of constructing the underlying graph, neither of which require the map of the deployment. One of these methods results in a so-called heterogeneous graph, since both APs and zones are now included in the graph.

We have then conducted a thorough performance evaluation of the homogeneous and heterogeneous methods, including two datasets, and comparing them to popular methods. Results show that GNNs achieves systematically better results. We have furthermore evaluated the practical advantages of these methods, where for instance they gracefully tolerate faulty APs.

In any case, these results are promising and encourage further research on using GNNs to approach indoor localization problems. For instance, other network architectures may be explored (e.g. attention-based), or include temporal information to the model.

References

Bahl P, Padmanabhan VN (2000) Radar: An in-building rf-based user location and tracking system. In: Proceedings IEEE INFOCOM 2000. Conference on computer communications. Nineteenth annual joint conference of the IEEE computer and communications societies (Cat. No. 00CH37064), Ieee, vol 2, pp 775–784

Basri C, El Khadimi A (2016) Survey on indoor localization system and recent advances of wifi fingerprinting technique. In: 2016 5th International Conference on Multimedia Computing and Systems (ICMCS), IEEE, pp 253–259

Bracco A, Grunwald F, Navcevich A, Capdehourat G, Larroca F (2020) Museum accessibility through wi-fi indoor positioning. arXiv preprint arXiv:200811340

Chen B, Chang RY (2022) Few-shot transfer learning for device-free fingerprinting indoor localization. CoRR abs/2201.12656, URL https://arxiv.org/abs/2201.12656, 2201.12656

Chiou M, Liu Z, Yin Y, Liu A, Zimmermann R (2020) Zero-shot multi-view indoor localization via graph location networks. In: MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020, ACM, pp 3431–3440, DOI 10.1145/3394171.3413856, URL https://doi.org/10.1145/3394171.3413856

Coley CW, Jin W, Rogers L, Jamison TF, Jaakkola TS, Green WH, Barzilay R, Jensen KF (2019) A graph-convolutional neural network model for the prediction of chemical reactivity. Chemical science 10(2):370–377

Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems 29

Ding Y, Jiang D, Liu Y, Zhang D, He T (2022) Smartloc: Indoor localization with smartphone anchors for on-demand delivery. Proc ACM Interact Mob Wearable Ubiquitous Technol 5(4), DOI 10.1145/3494972, URL https://doi.org/10.1145/3494972

Dong Y, Chawla NV, Swami A (2017) Metapath2vec: Scalable representation learning for heterogeneous networks. Association for Computing Machinery, New York, NY, USA, KDD '17, DOI 10.1145/3097983.3098036, URL https://doi.org/10.1145/3097983.3098036

Du J, Zhang S, Wu G, Moura JMF, Kar S (2017) Topology adaptive graph convolutional networks. CoRR abs/1710.10370, URL http://arxiv.org/abs/1710.10370, 1710.10370

Estrach JB, Zaremba W, Szlam A, LeCun Y (2014) Spectral networks and deep locally connected networks on graphs. In: 2nd international conference on learning representations, ICLR, vol 2014

Fey M, Lenssen JE (2019) Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds

Gama F, Marques AG, Leus G, Ribeiro A (2019) Convolutional neural network architectures for signals supported on graphs. IEEE Transactions on Signal Processing 67(4):1034–1049, DOI 10.1109/TSP.2018.2887403

Gama F, Bruna J, Ribeiro A (2020) Stability properties of graph neural networks. IEEE Transactions on Signal Processing 68:5680–5695, DOI 10.1109/TSP.2020.3026980

Isufi E, Gama F, Ribeiro A (2021) Edgenets:edge varying graph neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence pp 1–1, DOI 10.1109/TPAMI.2021.3111054

Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. 5th International Conference on Learning Representations (ICLR-17)

Küpper A (2005) Location-based services: fundamentals and operation. John Wiley & Sons

Lezama F, González GG, Larroca F, Capdehourat G (2021) Indoor localization using graph neural networks. In: 2021 IEEE URUCON, pp 51–54, DOI 10.1109/URUCON53396.2021.9647082

Lin H, Liu G, Li F, Zuo Y (2021) Where to go? predicting next location in iot environment. Frontiers Comput Sci 15(1):151306, DOI 10.1007/s11704-019-9118-9, URL https://doi.org/10.1007/s11704-019-9118-9

Morris C, Ritzert M, Fey M, Hamilton WL, Lenssen JE, Rattan G, Grohe M (2019) Weisfeiler and leman go neural: Higher-order graph neural networks. Proceedings of the AAAI Conference on Artificial Intelligence 33(01):4602–4609, DOI 10.1609/aaai.v33i01.33014602, URL https://ojs.aaai.org/index.php/AAAI/article/view/4384

Navarin N, Tran DV, Sperduti A (2019) Universal readout for graph convolutional neural networks. In: 2019 International Joint Conference on Neural Networks (IJCNN), pp 1–7, DOI 10.1109/IJCNN.2019.8852103

Nowicki M, Wietrzykowski J (2017) Low-effort place recognition with wifi fingerprints using deep learning. In: International Conference Automation, Springer, pp 575–584

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12:2825–2830

Ruiz L, Gama F, Ribeiro A (2021) Graph neural networks: Architectures, stability, and transferability. Proceedings of the IEEE 109(5):660–682, DOI 10.1109/JPROC.2021.3055400

Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2009) The graph neural network model. IEEE Transactions on Neural Networks 20(1):61–80, DOI 10.1109/TNN.2008.2005605

Schlichtkrull M, Kipf TN, Bloem P, van den Berg R, Titov I, Welling M (2018) Modeling relational data with graph convolutional networks. In: Gangemi A, Navigli R, Vidal ME, Hitzler P, Troncy R, Hollink L, Tordai A, Alam M (eds) The Semantic Web, Springer International Publishing, Cham, pp 593–607

Shi C, Hu B, Zhao WX, Yu PS (2019) Heterogeneous information network embedding for recommendation. IEEE Transactions on Knowledge and Data Engineering 31(2):357–370, DOI 10.1109/TKDE.2018.2833443

Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P (2013) The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Processing Magazine 30(3):83–98, DOI 10.1109/MSP.2012.2235192

Sun Y, Xie Q, Pan G, Zhang S, Xu S (2021) A novel GCN based indoor localization system with multiple access points. In: 17th International Wireless Communications and Mobile Computing, IWCMC 2021, Harbin City, China, June 28 - July 2, 2021, IEEE, pp 9–14, DOI 10.1109/IWCMC51323.2021.9498616, URL https://doi.org/10.1109/IWCMC51323.2021.9498616

Tang J, Zhang J, Jin R, Yang Z, Cai K, Zhang L, Su Z (2011) Topic level expertise search over heterogeneous networks. Machine Learning 82(2):211–237

Torres-Sospedra J, Montoliu R, Martínez-Usó A, Avariento JP, Arnau TJ, Benedito-Bordonau M, Huerta J (2014) Ujiindoorloc: A new multi-building and multi-floor database for wlan fingerprint-based indoor localization problems. In: 2014 international conference on indoor positioning and indoor navigation (IPIN), IEEE, pp 261–270

Varshavsky A, LaMarca A, Hightower J, De Lara E (2007) The skyloc floor localization system. In: Fifth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'07), IEEE, pp 125–134

Vignac C, Loukas A, Frossard P (2020) Building powerful and equivariant graph neural networks with structural message-passing. In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H (eds) Advances in Neural Information Processing Systems, Curran Associates, Inc., vol 33, pp 14143–14155, URL https://proceedings.neurips.cc/paper/2020/file/a32d7eeaae19821fd9ce317f3ce952a7-Paper.pdf

Wang P, Xu B, Wu Y, Zhou X (2015) Link prediction in social networks: the state-of-the-art. Science China Information Sciences 58(1):1–38

Whitehouse K, Karlof C, Culler D (2007) A practical evaluation of radio signal strength for ranging-based localization. SIGMOBILE Mob Comput Commun Rev 11(1):41–52, DOI 10.1145/1234822.1234829, URL https://doi.org/10.1145/1234822.1234829

Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS (2021) A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems 32(1):4–24, DOI 10.1109/TNNLS.2020.2978386

Yan W, Jin D, Lin Z, Yin F (2021) Graph neural network for large-scale network localization. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021, IEEE, pp 5250–5254, DOI 10.1109/ICASSP39728.2021.9414520, URL https://doi.org/10.1109/ICASSP39728.2021.9414520

Yiu S, Dashti M, Claussen H, Perez-Cruz F (2017) Wireless rssi fingerprinting localization. Signal Processing 131:235–244, DOI https://doi.org/10.1016/j.sigpro.2016.07.005, URL https://www.sciencedirect.com/science/article/pii/S0165168416301566

Zafari F, Gkelias A, Leung KK (2019) A survey of indoor localization systems and technologies. IEEE Communications Surveys & Tutorials 21(3):2568–2599

Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, Wang L, Li C, Sun M (2020) Graph neural networks: A review of methods and applications. AI Open 1:57–81, DOI https://doi.org/10.1016/j.aiopen.2021.01.001, URL https://www.sciencedirect.com/science/article/pii/S2666651021000012