Max Willian Soares Lima

Efficient Indoor Localization using Graphs

Brasil
2019

Max Willian Soares Lima

Efficient Indoor Localization using Graphs

Dissertação apresentada ao Instituto de Computação da Universidade Federal do Amazonas como requisito para obtenção do título de Mestre em Informática.

Universidade Federal do Amazonas – UFAM

Instituto de Computação – ICOMP

Programa de Pós-Graduação em Informática – PPGI

Supervisor: Prof. Dr. Edleno Silva de Moura

Co-supervisor: Prof. Dr. Horácio A. B. F. Oliveira

Brasil

2019

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Lima, Max Willian Soares
L732e Efficient Indoor Localization using Graphs / Max Willian Soares Lima. 2019
35 f.: il. color; 31 cm.

Orientador: Edleno Silva de Moura
Coorientador: Horácio A. B. F. Oliveira
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. indoor positioning systems. 2. nearest neighbors. 3. small world graphs. 4. machine learning. I. Moura, Edleno Silva de II. Universidade Federal do Amazonas III. Título

FOLHA DE APROVAÇÃO

"Efficient Indoor Localization using Graphs"

MAX WILLIAN SOARES LIMA

Dissertação de Mestrado defendida e aprovada pela banca examinadora constituída pelos Professores:

Prof. Edleno Silva de Moura - PRESIDENTE

Prof. Horácio Antonio B. Fernandes de Oliveira - MEMBRO INTERNO

Prof. Leandro Nelinho Balico - MEMBRO EXTERNO

Manaus, 05 de Agosto de 2019

Acknowledgements

Esse trabalho marca a conclusão de uma etapa. Mesmo sendo uma etapa repleta de dificuldades, com medos e angústias, essa conclusão é tida de forma satisfatória e cheia de agradecimentos a serem feitos.

Primeiramente ao meu pai e a minha mãe, sem eles os meus esforços não teriam sentido. Obrigado por serem sempre o meu maior apoio e motivação.

Agradeço também ao meu orientador Prof. Dr. Edleno Silva de Moura, pela dedicação, apoio e instruções no desenvolvimento deste trabalho. Assim como ao meu coorientador Prof. Dr. Horácio A. B. F. Oliveira pela atenção, envolvimento, cuidados e auxílios fundamentais para que tenhamos conseguido chegar aos resultados deste trabalho.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001. Além do apoio também da Positivo Tecnologia.

Agradecimentos especiais são direcionados aos meus colegas de pesquisa e amigos para a vida. Obrigado Mateus, Yuri e João. Meu sentimento é de satisfação e gratidão a todos, meu sincero obrigado!

Abstract

The main goal of an Indoor Positioning System (IPS) is to estimate the position of mobile devices in indoor environments. For this, the primary source of information is the signal strength of packets received by a set of routers. The fingerprint technique is one of the most used techniques for IPSs. By using supervised machine learning techniques, it trains a model with the received signal intensity information so it can be used to estimate the positions of the devices later in an online phase. Although the k-Nearest Neighbors (kNN) is one of the most widely used classification methods due to its accuracy, it has no scalability since a sample we need to classify must be compared to all other samples in the training database. In this work, we use a novel hierarchical navigable small world graph technique to build a search structure so the location of a sample can be efficiently found, allowing the IPSs to be used in large scale scenarios or run on devices with limited resources. To carry out our performance evaluation, we proposed a synthetic IPS dataset generator as well as implemented a complete real-world, high scale IPS testbed. We compared the performance of our graph-based solution with other known kNN variants, such as Kd-Tree and Ball-Tree. Our results clearly show the performance gains of the proposed solution at 96% when compared to the classic kNN and at least 77% when compared to the tree-based approaches.

Key-words: indoor positioning systems, nearest neighbors, small world graphs.

List of Figures

List of Tables

Contents

1 Introduction

Location Based Services (LBSs) have been extensively used in outdoor environments by several applications due to the ready availability of accurate positioning information (Liu et al., 2007; Farid et al., 2013; Boukerche et al., 2007). Currently, most mobile devices are equipped with *Global Positioning System* (GPS), allowing the development and growing usage of some exciting and useful applications such as Waze and Google Maps, to cite a few (Boukerche et al., 2008).

Unfortunately, even though several LBSs have been used in outdoor environments, the same cannot be observed in *indoor* settings, such as offices, shopping malls, and parking lots. The main reason for this is the inaccurate positions in these environments reported by GPS, since the latter requires a direct view of the satellites to work properly (Lima et al., 2018). Thus, a lot of interesting applications that could exist today still hasn't been developed for these indoor scenarios due to the lack of accurate positioning technology.

Currently, the leading available solution for this problem is the *Indoor Positioning Systems* (IPSs). In this case, a local infrastructure available or installed in the building is used to allow the precise location of mobile devices inside this building. Among the most used infrastructure for IPS is the WiFi, due to its abundant availability and no need for additional hardware, since most buildings and devices (e.g., smartphones, smart TVs, Internet of Things) already come equipped with this technology. In this case, the *Received Signal Strength Indicator* (RSSI) of the packets received by the WiFi access points are used to locate the mobile device that sent these packets.

However, using the available WiFi infrastructure for an indoor location of mobile devices is not something trivial due to the high variability of the RSSI. Unfortunately, the several obstacles and walls inside buildings make it difficult to translate a signal strength into distance information to be used in the positioning. To overcome this problem, currently, the most considered solution for an IPS is using the *fingerprint* technique.

As depicted in Figure 1, the fingerprint technique is divided into two phases: the training and the positioning, also known as offline and online phases, respectively. In the training phase, several points of the scenario are trained by physically going to these locations and start sending WiFi packets. The packets sent from each *training point* are received by several access points that can compute the RSSI of these packets. All of this information is then sent to a central server that stores them in a *training database*. In the positioning phase, which is when the IPS is online and working, when a mobile device sends a packet, the RSSIs of this packet perceived by several access points are sent to

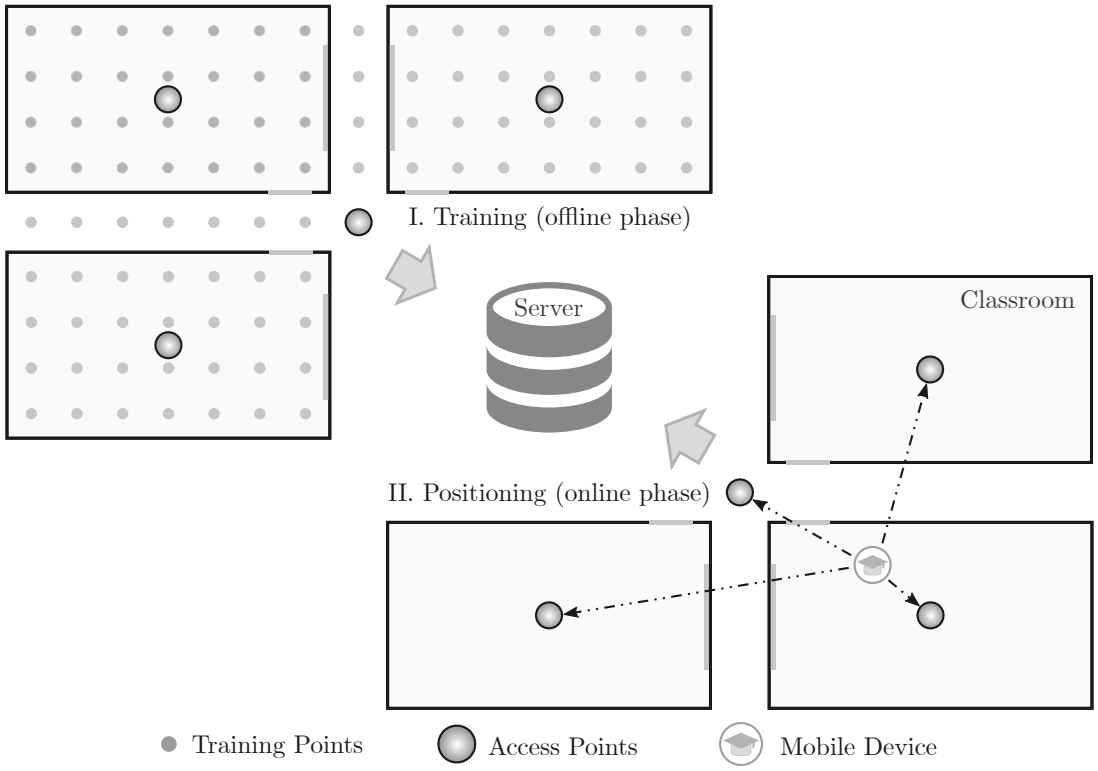

Figure 1: The two phases of a centralized fingerprint-based indoor positioning system implemented in a campus scenario: training and positioning.

the central server. The server then compares the received RSSIs to the ones stored in the training database to find the training point that mostly matches the received data. The returned point is the position of the mobile device.

To implement a fingerprint-based IPS, we typically use an architecture as depicted in Figure 2. Usually, *supervised machine learning techniques* are among the most viable solutions (Fang and Lin, 2012; Sayed et al., 2005; Yim, 2008). They use the previously recorded training database to train a classifier, such as the *k-Nearest Neighbors* (kNN) (Khodayari et al., 2010; Tao et al., 2008; Yim, 2008), so it can be used to estimate/classify the positions of the nodes later in the online phase.

The use of the fingerprint technique for this purpose has grown due to two crucial factors: (1) there is no need for additional hardware to gather location information since the WiFi infrastructure is usually enough; and (2) the ability of the system to remain consistent even with the unpredictability of the signal strength values of the devices. Recently, the fingerprint technique has been improved by several proposals, so that in some works it is already possible to locate devices in an indoor environment with errors of only one meter or so (Khodayari et al., 2010; Li et al., 2006; Fang and Lin, 2012).

kNN is one of the most used classification methods in fingerprint-based positioning systems due to its high precision results (Lin and Lin, 2005). However, it lacks scalability since each sample to be classified must be compared to all other samples in the training

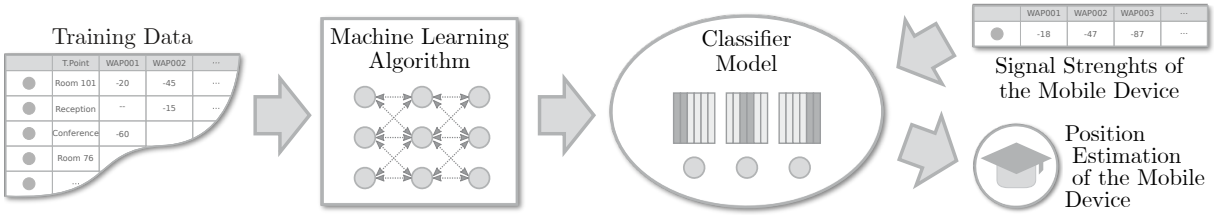

Figure 2: A typical architecture for a fingerprint-based indoor positioning system running on a centralized server.

database (Kuo and Tseng, 2011). Also, the training database tends to increase in size drastically as we increase the location scenario or when we need more precision, since the more data we have, the more precise it tends to be the estimated positions (Chen et al., 2006).

If we take an example of a medium sized room where there may be several training points (e.g., one for every two square meters), and knowing that for each of these points we need several packet samples (e.g., between 100 and 1000 packets), we can quickly have more than tens of thousands of samples in the training database collected from a single room. Moreover, each sample (row) in the database has several attributes (columns) according to the number of WiFi access points present in the whole scenario. Thus, it is easy to see that as we increase the area of the IPS, we also need to add more access points, increasing both the number of attributes and the dimension of the classification (Fang and Lin, 2012). Taking the example of a shopping mall and a university campus, which are relatively large in scale, the database used for training and positioning can easily reach billions of samples with hundreds or even thousands of attributes. Finally, taking into account that the goal is to have real-time location information for each of the thousands of mobile devices every second (something easily done by GPS), it is clear that a highly efficient classification method is required (Kuo and Tseng, 2011; Li et al., 2006; Youssef et al., 2003).

In this dissertation we studied the problem of the *Indoor Positioning Systems*, focusing on the fingerprint approach. We proposed using a novel technique called *Hierarchical Navigable Small World* (HNSW) (Malkov and Yashunin, 2016) to speed up the task of searching for nearest neighbors. In this technique, a graph-based data structure is carefully crafted to have the characteristics of a small-world network (Guidoni et al., 2012). This data structure is then used to find the nearest neighbors, as done by the kNN technique. Our hypothesis was that the use of a graph-based approach would yield performance gains over the approaches being used in the literature, without causing quality losses in the result.

To validate the hypothesis, experiments were performed in practical environments and in reference collections. We evaluated the performance of the solution using our proposed large-scale synthetic fingerprint dataset generator. We argue and show that this simulation-based synthetic dataset can evaluate the performance of any large scale IPS under several different scenarios, something not possible to do in a real-world testbed. Additionally, we also carried out high-scale real-world experimentations, implementing a complete IPS testbed in a school building. Finally, we compared our results to the UJI Indoor dataset, used by several known works to evaluate the performance of IPSs (Torres-Sospedra et al., 2014). In all of these cases, we compared the performance of the HNSW to two other known kNN optimizations: Kd-Tree and Ball-Tree. Our results clearly show the performance gains when using the graph-based solution by 96% when compared to classic kNN and at least 77% when compared to the tree-based approaches.

The remainder of this paper is organized as follows. In the next section, we present our related work. The graph-based technique is explained in section 3. We then show and discuss our performance evaluation in Section 4. Finally, in section 5 we present our conclusions and future work.

2 Related Work

Our related work is divided into three categories: (1) scenario analysis and fingerprint, in which we show known fingerprint-based techniques; (2) computational cost reduction, in which we show proposed techniques to improve the efficiency of the fingerprint, usually reducing the training data; and (3) efficiency without loss of information, in which we show some current work that is more closely related to our approach that improves efficiency without reducing the training data.

2.1 Scenario Analysis and Fingerprint

The work developed by Bahl and Padmanabhan (2000) was one of the first to use the RSSI of WiFi devices as information to estimate their locations. The proposed fingerprint-based solution, called RADAR, resulted in an accuracy of 2 to 3 meters. Subsequently, as seen in Bahl et al. (2000), the authors have made improvements based on the Viterbi algorithm, increasing the accuracy of location up to 33%.

Roos et al. (2002) presented a fingerprint-based approach to estimate the location of users. The authors considered the location as a machine learning problem, where the task would be to model how the RSSIs are distributed in different geographic areas based on samples collected in various known areas. From this, they presented a probabilistic structure responsible for estimating the location of the mobile devices. Such a structure uses Bayes' theorem to compute the likelihood between the collected signals given each possible location of the same. Using real-world samples to demonstrate their results, they have shown estimates with an average location error of about 2 meters.

In Haeberlen et al. (2004), it is also proposed a probabilistic approach. The authors collected the data received throughout a 3-story building, remaining about 2 minutes in each office or region with a mobile device emitting the signals, resulting in a location accuracy of 95% in all predictions. However, the proposed solution has a room-based resolution, not being able to pinpoint the position of the users inside the rooms. The results of this work also presented robustness concerning various unforeseen phenomena at the time of data collection, such as the presence or absence of people in the environment throughout the day. These and most current proposed solutions focus mainly on the accuracy and precision of the systems (Lin and Lin, 2005) while other aspects related to complexity and performance are ignored.

2.2 Computational Cost Reduction

To reduce the computational cost, Youssef et al. (2003) proposed a technique based on clustering, in which it groups the locations from a set of WiFi access points. Called Joint Clustering, the technique has two key points: (1) use of probability distributions to improve accuracy and (2) use of groups of mapped data to reduce the cost of online positioning. Later, in Youssef and Agrawala (2005), the authors presented the system entitled Horus, following the same probabilistic approach and optimization from clustering, but with advances in the preprocessing of signal power models to consider spatial and temporal variations characteristics of the wireless channel. The results of the Horus system presented an average precision of up to $39cm$ in one of the tests performed.

Chen et al. (2006) seek to reduce the computational cost by reducing the size of the database used to train the classification algorithms. They developed an algorithm called CaDet that selectively chooses some access points to be used in the estimation, allowing an intelligent subdivision of access points necessary for a precise location. In this way, by reducing the required number of access points for each estimate, there is an efficiency gain. Also, the solution allows for a lower energy cost, which is paramount to the approach, since the classification processing is done directly on mobile devices. CaDet uses a combination of information theory, cluster analysis, and decision tree to achieve good accuracy with performance gain.

Yim (2008) sought to optimize IPSs by proposing the discretization of the RSSI values in the database to train a decision tree-based model during the offline phase. The authors had the premise of reducing the time needed to both generate the training model and locate the mobile device. From that, they presented experiments that demonstrate gains concerning the complexity of the proposed approach. However, little was said about the results of accuracy, only making it clear that they are no worse than the methods compared.

Fang and Lin (2012) reduce the computational cost by combining related access points information and using Principal Component Analysis (PCA) to decrease the dimensionality of the data. An algorithm is proposed to intelligently transform incoming signals into main components, instead of selecting access points, so the information received can be used more efficiently, reducing the computation required in the online phase. Compared to other approaches, the authors have demonstrated that this solution can reduce the positioning error by 34% and has a 40% reduction in time complexity. Also, the transformation into main components results in some other benefits, such as the need for less training data and tolerance to anomalies in the received signals.

While all of these works result in the reduction of the computational cost, they are all based on the removal or combination of features from the training base, resulting in the loss of information. This information removal can lead to higher errors in specific scenarios since there is usually a loss of potentially relevant information, also making it difficult or even impossible to implement adaptive solutions and other forms of calibration, which can be used to reduce the location errors (Chen et al., 2005).

2.3 Efficiency Without Loss of Information

In our current proposal, we focus mainly on solutions that do not need to lose training information to achieve better performance. Some known algorithms of tree-based classification have been proposed with this goal, such as Ball-Tree (Omohundro, 1989) and KD-Tree (Bentley, 1975), as well as other approaches (Muja and Lowe, 2014; Houle and Nett, 2015; Chen et al., 2006; Yim, 2008) that take advantage of this type of data structure to quickly search and find, within the training base, a set of data that behaves similarly to the search entry. Recently, the use of graph-based methods to search for the k-nearest neighbors have shown competitive gains in performance when dealing with large datasets, as seen in Dong et al. (2011) and Malkov et al. (2014a).

Thus, in this work we seek to explore the application of hierarchical navigable small world graphs, recently proposed by Malkov and Yashunin (2016), to reduce the computational cost of IPS without loss of information while maintaining the accuracy of the results. The main idea of this algorithm is to use graph-based data structures and small world techniques to improve the search for areas that are similar to the input entry. The use of small world techniques allows large jumps in the graph, as explained in more detail in the next section. As far as we know, this is the first time that a graph-based machine learning algorithm is explored and applied to indoor positioning systems.

3 IPS using Hierarchical Navigable Small World Graphs

When it comes to optimizing the search for a particular area of the training database where the attributes are similar to the one we are searching (i.e., the nearest neighbors), most approaches tend to perform a reduction of the training data, causing loss of information (Houle and Nett, 2015; Chen et al., 2006; Yim, 2008), or are based on a tree structure to rearrange the training data (Omohundro, 1989; Bentley, 1975; Muja and Lowe, 2014).

Recently, some proposed solutions have shown significant advances in the use of graph-based methods to search for the nearest neighbors on high dimensional datasets. This efficiency in high dimensional datasets is an essential aspect for fingerprint-based high-scale indoor positioning systems since a training database can be composed of hundreds to millions of access points (i.e., attributes, dimensions).

One of the critical points for the use of a graph-based structure is the implementation of a small world model. In this case, the generated graph has two essential characteristics for the search process: (1) small path length values between nodes (samples in the case of IPSs), and (2) high cluster coefficient values (Guidoni et al., 2012). The first feature allows for a quick (logarithmic or less) searching for a similar sample in the training database. This fast search happens because a small percentage of distant nodes can have links between them, allowing the search to bypass most of the database in a single hop. The second feature guarantees the connectivity of the nodes (existence of paths) and facilitates the location of the k values closest to a given sample.

Based on that, a proximity graph for the nearest neighbor search called Navigable Small World was proposed in Ponomarenko et al. (2011) and Malkov et al. (2014b). These solutions use an alternative approach of the simple greedy search used on proximity graphs and also a probabilistic skip list structure (Pugh, 1990) that allows a logarithmic scaling for the search.

Proposed by Malkov and Yashunin (2016), the *Hierarchical Navigable Small World Graphs* (HNSW) implemented the idea of representing each sample of the dataset as a node in the graph. It separates their links according to their length scale, producing a multilayer graph that allows the evaluation of only a small subset of the connections for each element, thus reducing the computational cost of performing the search.

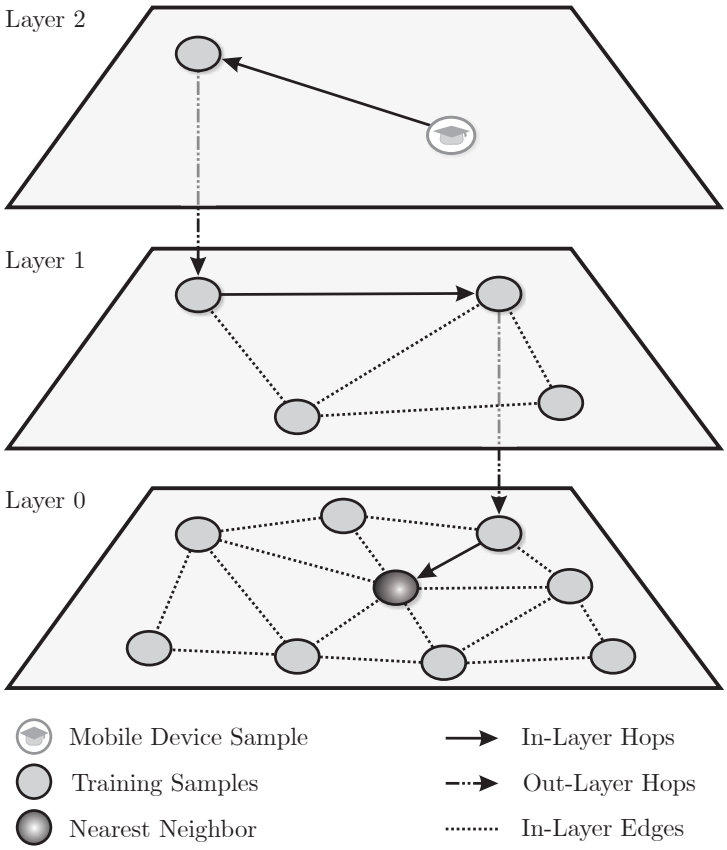

Figure 3: Indoor positioning system classification using hierarchical navigable small world graphs, based on Malkov and Yashunin (2016).

To build the multilayer graph, the links are set with different distance scales by artificially introducing *layers*, as depicted in Figure 3. Each layer is represented by an integer number, called *level number*, and every node on the graph receives a level number representing the maximum layer for which this node belongs. The top layers represent connections between distant nodes, which gives the graph its main small-world characteristics. On the bottom layers, the graph has the connections between closer nodes, connecting a node to its neighborhood. Each node is inserted on the graph at the layer represented by its level number, as well as at the descending layers up until the *ground layer* (Layer 0), being linked to its nearest nodes in each level. The level number is randomly selected, and an exponentially decaying probability distribution is used to concentrate the nodes at the lower layers and enable close connections, and letting only a few nodes at the top layers, using a parameter to normalize this distribution.

After the graph construction, as depicted in Figure 3, when a new search is required, it starts on the highest layer and greedily traverses the graph selecting elements only at that layer until it finds a local optimum. Once the local optimum is found, the search continues on the next layer below in the hierarchy. This process is repeated starting from the local minimum found in the previous layer, using its nodes as entry points, and so on until the ground layer is reached.

A variant of a greedy search algorithm based on Malkov et al. (2014b) is used to find the closest neighbors at each layer. A dynamic list is used to keep the *ef* closest nodes found at that layer, and during the search, this list is updated at each step by evaluating the neighborhood of the closest previously non-evaluated element in the list until the neighborhood of every element from the list is evaluated. Initially started as 1, to represent a simple greedy search, the *ef* parameter is incremented up to *efConstruction*, another parameter which is used to control the recall of the greedy search procedure. This value can be increased to improve the quality of the constructed graph, leading to higher accuracies but also increasing the indexing time. Another parameter, M, defines the maximum number of neighbors in the zero and above-zero layers. In most of our experiments we used the default value of 100 for both the *efConstruction* and *ef* parameters, and the value 15 for M, which resulted in the best accuracies.

The HNSW has two of the main characteristics we need for a fingerprint-based, high-scale IPS: (1) connectivity (clusterization) of near/similar samples, allowing the election of the best sample based on the k nearest neighbors (kNN), and (2) a fast search for the sample with most similar attributes. In this work, we explore the use of the HNSW in the task of estimating the location of mobile devices in a fingerprint-based IPS. Our performance evaluation methodology, as well as our obtained results, are discussed in the next sections.

4 Performance Evaluation

Our performance evaluation aimed at comparing the cost of building the search index (model fitting) and also the cost of searching for the nearest neighbors (sample classification) achieved by the studied methods. Also, we analyzed the error of the indoor positioning system when varying the algorithm adopted to find the nearest neighbors. Below we describe our evaluation setup and experimental results.

4.1 Methodology and Datasets

We compared the hierarchical navigable small world graphs (HNSW) to two tree-based methods: Kd-Tree and Ball-Tree. We also compared the results to the classic kNN search, commonly known as *brute force*. For the implementation of the HNSW search, we used the Non-Metric Space Library (NMSLIB) (Boytsov and Naidan, 2013), while for the other methods we used the Scikit-Learn (Pedregosa et al., 2011), both using Python language. All of the results were reached through the mean of a ten-fold cross-validation (Kohavi et al., 1995).

We evaluated the performance of the implemented solutions using three very different datasets:

1. *Synthetic Indoor Positioning Dataset*
2. *SmartCampus Indoor Positioning Dataset*
3. *UJI Indoor Localization Dataset*

These datasets are detailed in the next sections.

4.1.1 Synthetic Indoor Positioning Dataset

We built an indoor positioning simulator based on known propagation models for indoor environments to generate synthetic IPS training databases from different scenarios such as varying the number of routers, rooms, and training points. By using this simulator, we were able to evaluate the performance of the studied algorithms in hundreds of different indoor scenarios, something that would not be feasible to do in a real-world testbed.

Even though we implemented some of the most known propagation models such as the Free-Space (Friis, 1946) and the Log-Distance (Rappaport, 2001), they didn't result in a good performance for the positioning since they do not take into consideration some

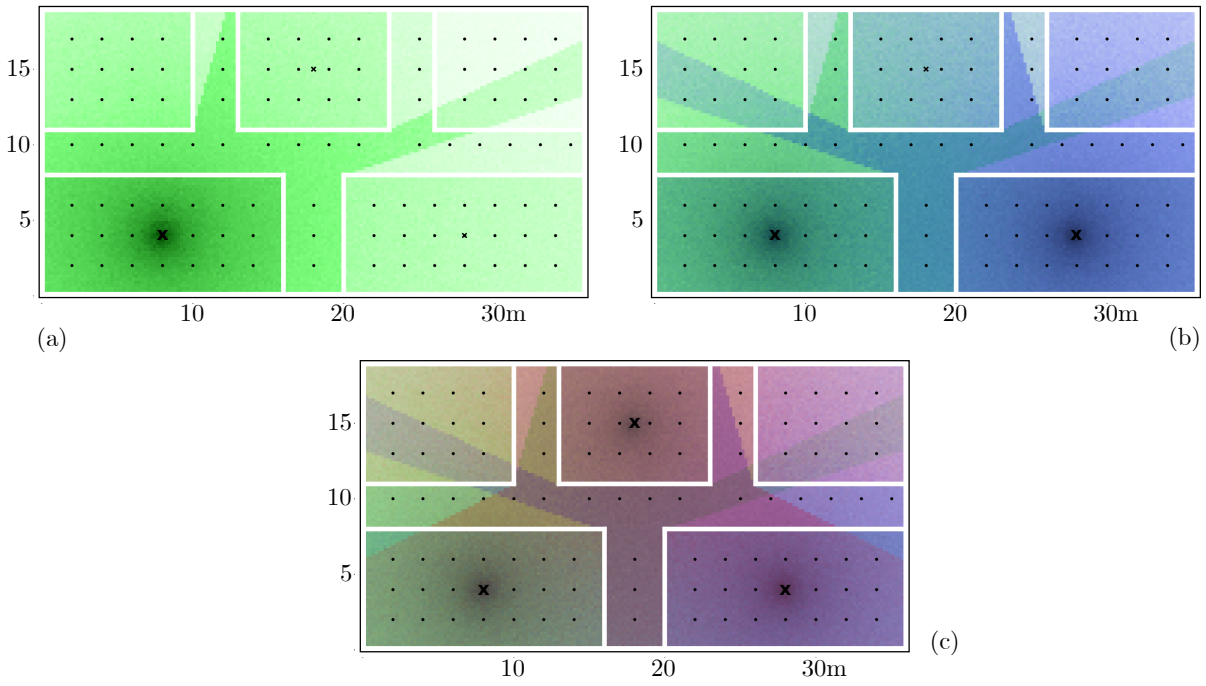

Figure 4: Example of a signal strength simulation: white lines are the walls of the rooms. X marks the position of the routers. In (a), we have only a single router in the most bottom-left room. While in (b) and (c) we have two and three routers, respectively.

scenario information such as walls, floors, and ceilings. Thus, we implemented our model based on the known Simple Indoor Signal Propagation Model (Plets et al., 2013), in which the authors proposed a heuristic indoor path loss prediction model that is both simple and quick (as the statistical models) but is also very accurate. They formulated an indoor path loss model for 2.4 GHz-band, performing simulations in four very different buildings to help on the path loss measurements and parameter determinations. The model takes into consideration both wall and floor attenuation factors, as suggested by Bahl and Padmanabhan (2000).

By feeding the simulator with the map of the location, as well as the values for the model variables, we were able to compute the RSSI perceived by the routers from any place in the scenario. Figure 4 shows an example of the generated signals strengths in a simple scenario composed of 5 rooms. In Figure 4(a), we have only a single router in the most bottom-left room. While in Figures 4(b) and 4(c), we have two and three routers, respectively. From this, to create our synthetic positioning training dataset, we got several samples of RSSIs from equally spaced training points inside the rooms.

The synthetic training dataset generated by us has 378 distinct rooms within one floor, being that 196 simulates rooms and 182 simulates halls. There are 21 points for each room and 7 in each hall, where we simulated 50 samples for each point, a total of about $270,000$ samples.

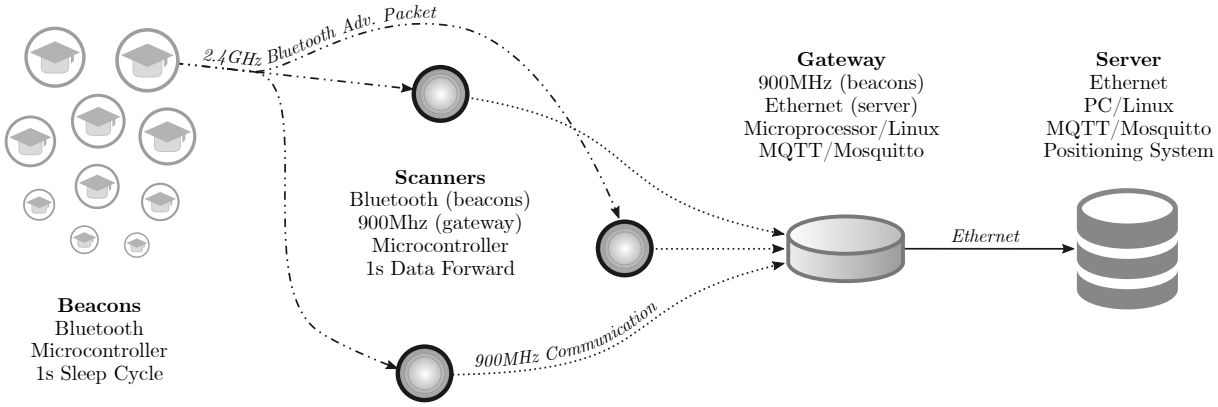

Figure 5: SmartCampus Architecture: a complete, large-scale, indoor positioning system testbed. All beacons send a Bluetooth advertisement packet every second. These packets are received by the scanners, that forward the data to the gateway using a 900MHz, long-range, communication. The gateway sends the data to the central server, which can localize all of the beacons using the proposed graph-based technique.

4.1.2 SmartCampus Indoor Positioning Dataset

The second dataset used on our performance evaluation was based on our real-world indoor positioning testbed. We implemented a complete IPS in a working high school composed of 42 custom-made routers (we called them *scanners*) and 60 rooms. For each room, we trained several training points every two meters. For each training point, we collected more than $1,000$ samples, resulting in an indoor positioning dataset composed of approximately $490,000$ samples.

This testbed is the beginning of a bigger project to create an indoor positioning system financed by Positivo Technologies, one of the biggest technology companies in Brazil. This project, called SmartCampus, aims at providing location information of students in high schools, allowing several interesting applications such as:

- automatic student attendance record keeping
- checking whether a student is in the classroom he is supposed to be
- entry and exit control of students from the school
- where the student sits inside the classroom
- interaction among students and friend groups

For this, we proposed and implemented the SmartCampus Architecture, depicted in Figure 5. In this architecture, *beacon nodes* (worn by the students) send Bluetooth advertisement packets. These packets are received by several Bluetooth-enabled devices, called *scanners* (similar to routers in WiFi IPSs). These scanners compute the RSSI of the received packets and send them directly to a central device, the *gateway*, using long-range,

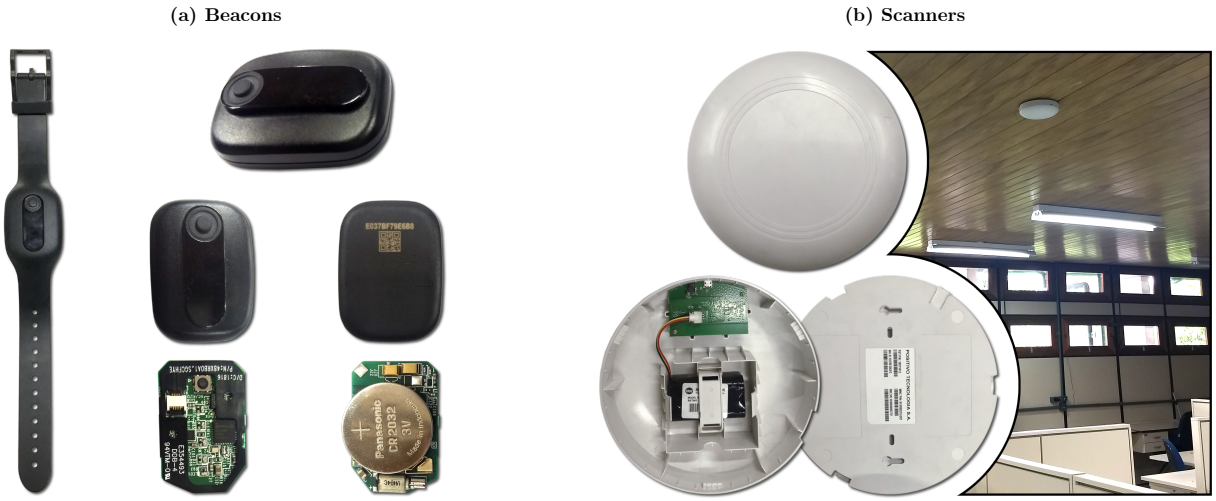

Figure 6: SmartCampus Hardware: (a) beacons with Bluetooth communication; one beacon is inside a bracelet, as used by the users; the other beacons are outside the bracelet (front and back), and opened to show hardware (front and back); and (b) scanners with Bluetooth and 900Mhz long-range communication (front, opened, back, and installed on the ceiling). The gateway has a similar hardware footprint as of the scanners, and the server is not shown.

900Mhz, communication. The gateway is connected to a central server, that can locate all of the received beacons using the proposed graph-based machine learning technique.

All of the hardware used in the SmartCampus architecture was developed and created by Positivo Technologies. The hardware is depicted in Figure 6. Finally, Figure 7 shows an overview of the floorplan of the school in which we implemented the testbed as well as the location of the gateways.

4.1.3 UJI Indoor Localization Dataset

For the third dataset used to evaluate the performance of the techniques, we used another real-world scenario: the popular UJI Indoor Localization dataset (Torres-Sospedra et al., 2014). This dataset has been used in the literature and allows the comparison of our results to other published work. Also, this scenario is quite different from our SmartCampus testbed. It covers a surface of $108,703m^2$ with 3 different buildings comprised of 4 to 5 floors. There is a total of 520 routers with a total of 21,049 samples. The data was collected by more than 20 users with 25 different mobile device models.

Compared to our SmartCampus dataset, it is easy to see that the UJI testbed scales on the number of routers since it has 520 routers while our testbed contains only 42 routers. However, the UJI dataset does not have several samples per training points, and these training points are not equally distributed. Thus, the UJI contains only 21,049 packet samples, while our SmartCampus testbed contains approximately 490,000 samples. Therefore, our SmartCampus testbed scales on the number of samples.

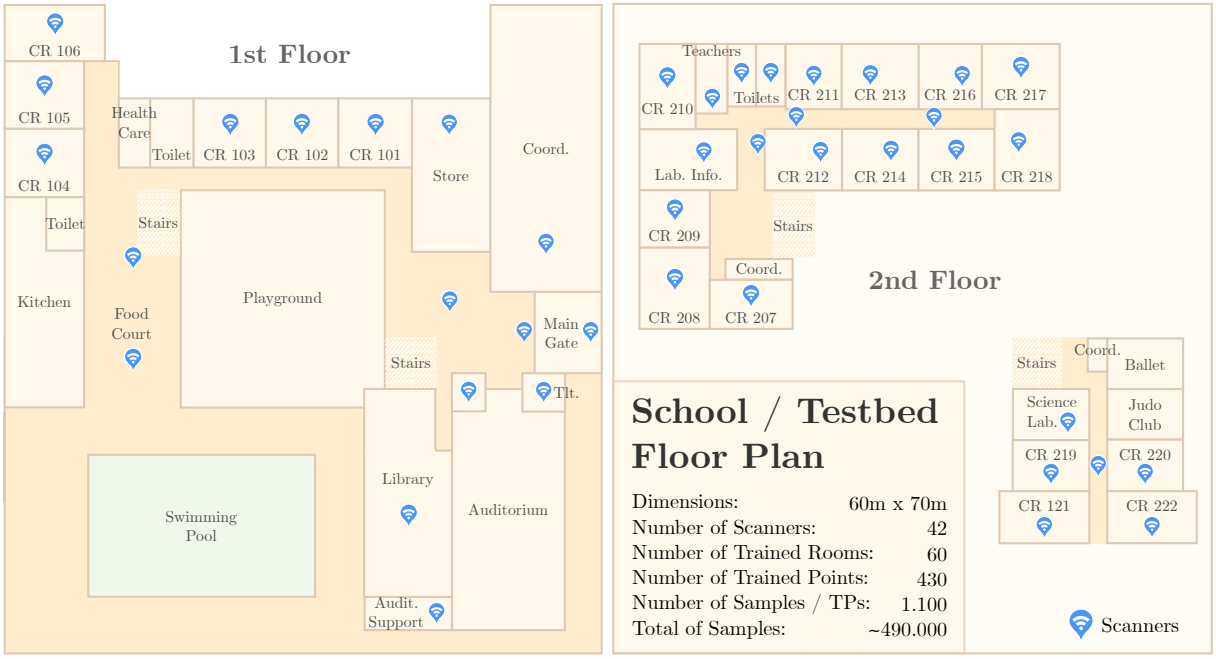

Figure 7: SmartCampus Testbed Floor Plan: an entire school composed of 60 rooms/halls were trained. 42 scanners were distributed throughout the school; 430 points were trained, about $2m$ distant from each other; for each training point, about 1100 samples were collected, to a total of about 490.000 samples.

Positioning Dataset / *Feature*	**Synthetic**	**SmartCampus**	**UJI**
Number of Routers	196	42	520
Number of Rooms	378	60	123
Number of Training Points	5,390	430	493
Number of Samples per TPs	50	~1,100	~10 (avg)
Total Number of Samples	~270,000	~490,000	~21,000

Table 1: Comparison among the three experimented IPS datasets.

Table 1 compares the main characteristics of the three datasets. Next, we show the results of our performance evaluation, starting with the experiments with the synthetic base, since this is the most controlled scenario.

4.2 Impact of the Number of Samples on the Classification

The use of a graph-based data structure by the HNSW is done to reduce the search time required at the time of classification and, also, to avoid an increase in the positioning error. With this in mind, in our first experiment, we varied the number of samples in the training database from 26 thousand to 269 thousand samples to analyze the scalability of the methods as the size of the training base increases. In this scenario, we had 196 access points (attributes in the dataset).

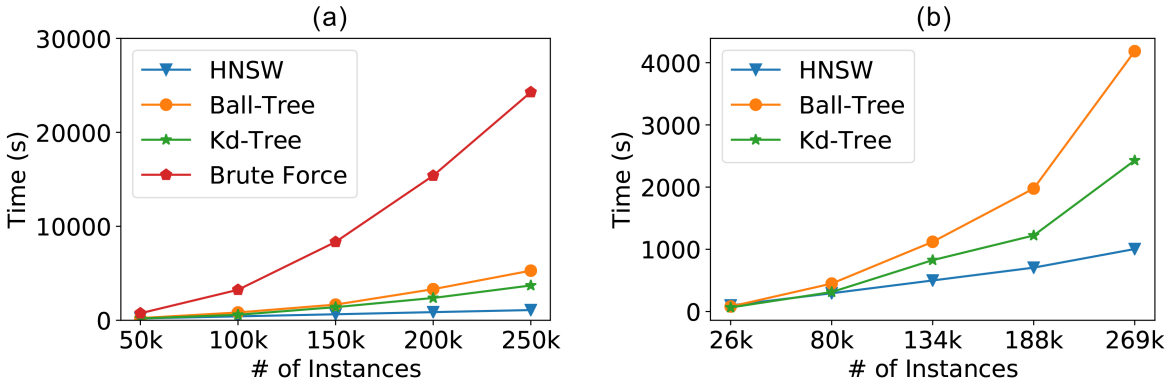

Figure 8: Scalability test results from a dataset containing 196 access points (attributes) and varying the number of samples. (a) classification time including the brute force and (b) without the brute force method.

For the synthetic database scenario, we can see in Figure 8(a) the results regarding the performance of the compared methods. In this case, we are comparing the time needed to classify 10% of the database (used as the test dataset), using the ten-fold cross-validation. It is easy to see that any of the optimized methods exhibit a considerable performance gain over the classic kNN using brute force. The graph-based solution is 96% faster than the classic kNN brute force, while the Kd-Tree and the Ball-Tree were 85% and 77% faster, respectively.

To facilitate the visualization, Figure 8(b) repeats the previous experiment without the brute force kNN, since the difference from the optimized methods is too high. Now, we can clearly see that the HNSW technique is also able to outperform the tree-based approaches. In these experiments, the graph-based technique was 59% faster than Kd-Tree and 76% faster than Ball-Tree. Also, we can easily see the difference in the curve behavior. For now on, we will refrain from showing the brute force kNN in order to better evaluate the optimized approaches.

In Figure 9(a) we compare the results of the methods concerning the positioning error as we increase the size of the training database. As expected, the more training data we have, the lower the positioning error. In our case, we can see that the position error decreased from $2.33m$ to $1.19m$, a reduction of 49%, as we increased the number of samples from 26 thousand to 269 thousand. These results help us realize the importance of additional samples in the training database for more accurate positioning. Although the brute force method is not on the chart, it is important to note that it has achieved roughly the same error results as Kd-Tree and Ball-Tree. In the chart we can also notice another significant result, the HNSW technique resulted in smaller positioning errors when compared to tree-based methods. Even though the difference is not much, it was still possible to reduce the positioning error by 9% at best.

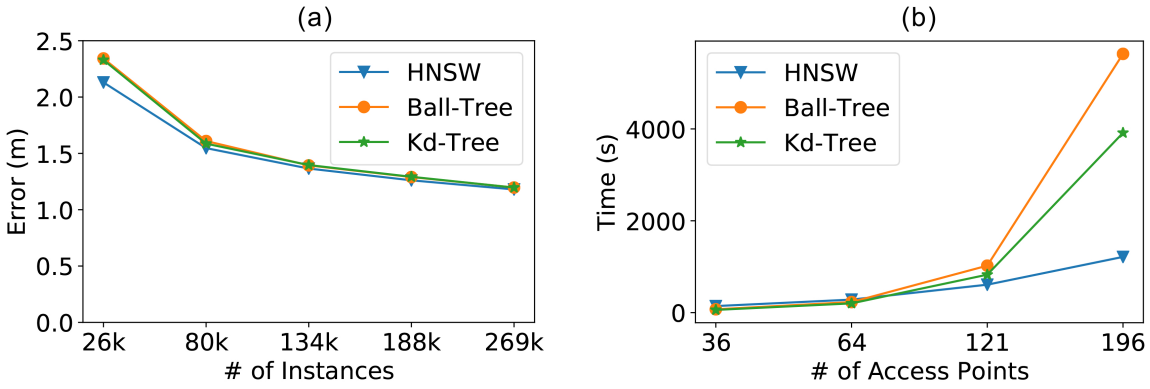

Figure 9: Scalability test results from a dataset containing 196 access points (attributes) and varying the number of samples. (a) Positioning error obtained by the experimented methods; and the (b) impact of the number of access points.

4.3 Impact of the Number of Access Points

The number of access points is also another factor that significantly affects the computational cost of an IPS since each additional access point is accountable for including a new column (attribute) for all samples in the training database. Besides, increasing the density of the access points is also a known technique used in the literature to reduce the positioning error, where better accuracy is sought by increasing the number of access points per square meter.

To assess this impact, we varied the number of access points in the experiments from 36 to 196, but we maintained the density of the access points constant, i.e., we increased both the number of access points and the size of the scenario (number of rooms and number of training points). This constant access point density is needed since we wanted to analyze the impact on the computational cost but without the positioning error being affected by poor distribution of the density of the access points.

As depicted in Figure 9(b), another interesting aspect of the HNSW technique is that its performance advantages also scale up as we increase the number of access points, being 77% faster than Ball-Tree for 196 access points.

4.4 Impact of the Number of Samples on the Model Fitting

One of the steps in an IPS architecture is the creation of the model, or the indexing of the data in its respective search structure, based on the available training data (the *fitting task*). This step was depicted earlier in Figure 2. Even though the fitting task execution time is not an essential factor in IPSs since it is executed only once, it might affect some known techniques that require a periodic model fitting, such as calibration-

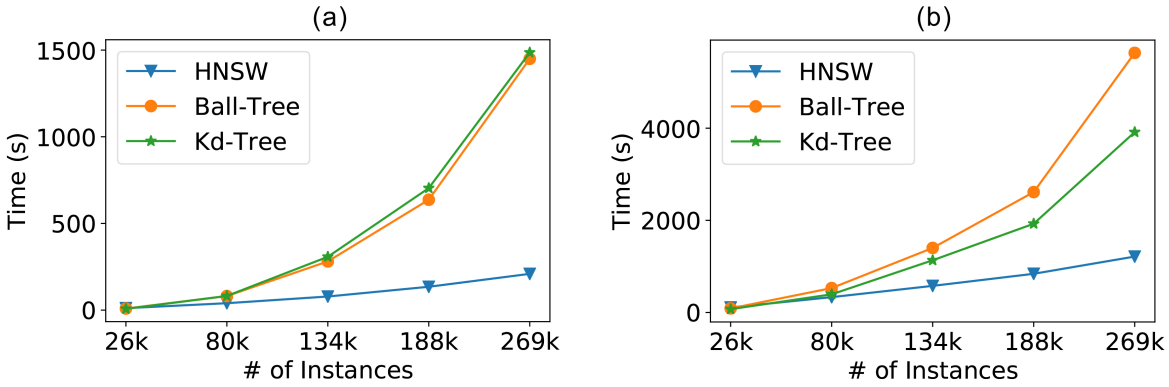

Figure 10: Scalability evaluation including the (a) model fitting time; and (b) the total time (fit+classification).

based and adaptive solutions (Chen et al., 2005) that are known to reduce even more the positioning error over time.

In this way, it is depicted in Figure 10(a) the fitting times as we increase the number of samples in the training database. For this task, the brute-force method (not included in the chart) has a constant time close to zero, since it does not execute any indexing of the data. As we can see, especially in the larger scenarios, the creation of the graph-based model is faster than the tree-based models, reaching 86% faster fitting times.

Finally, Figure 10(b) shows the complete total time (fitting+classification) of the experimented models. When including both model creation and sample classification, the HNSW technique is 79% faster then Ball-Tree and 69% faster than Kd-Tree.

4.5 SmartCampus Dataset Experiments

As mentioned before, we also performed real-world experiments of our solution by implementing a complete IPS in a school building. The resulted SmartCampus dataset represents a real-world scenario in a building with distinct floors and spaces, and it approaches the number of samples of the synthetic dataset, allowing the analysis of the scalability of our proposal in a real scenario. Despite this, the SmartCampus is the dataset with the least amount of access points among the three evaluated datasets, even though 42 is a reasonable amount of access points (higher than most experiments in the literature) and is also wholly consistent with many situations in the real world.

With the complete dataset, we analyzed the impact of the k variation on precision and time. In Figure 11(a) we can see that there was a slight advantage in the use of graphs, with a mean error difference of a few centimeters, when looking to the best results achieved for each method.

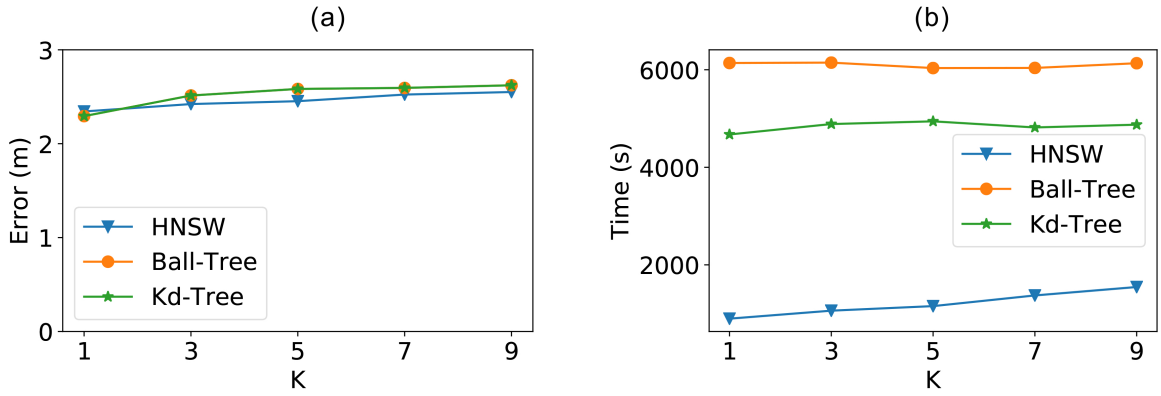

Figure 11: (a) Positioning error and (b) total time (fitting + classification) when using the real-world SmartCampus scenario dataset and varying the k value.

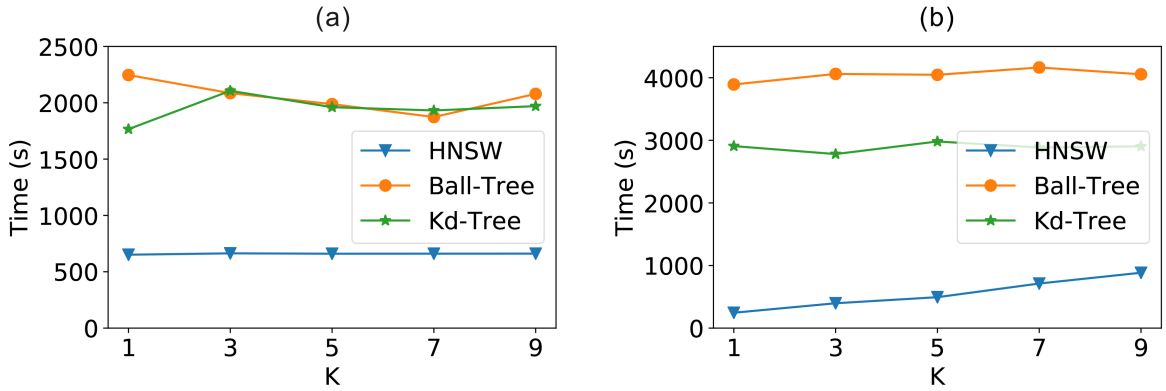

Figure 12: (a) Fitting time and (b) classification time evaluation results when using the real-world SmartCampus scenario dataset and varying the k value.

For this experiment, we changed the default values of HNSW parameter M, from 15 to 50, and the parameter *ef*, from 100 to 200. In other words, we allow greater connectivity in the graph and allow a search with greater recall, respectively. This adjustment was necessary to reach the accuracy comparable with the other methods, with up to a certain advantage, which meant an extra cost of performance, compared to the standard values of the parameters. However, the method allows this flexibility, and yet we maintained a remarkable superior performance.

While the accuracy advantage is small, the performance gain achieved by HNSW is impressive, as it can be seen in Figure 11(b). For instance, detailing each different task, with $K = 5$ the graph-based method was 3 times faster than both tree-based methods on the fitting time, as shown in Figure 12(a). Moreover, for the classification task, the HNSW was 6 times faster than Kd-Tree and more than 8 times faster than the Ball-Tree, as shown in Figure 12(b).

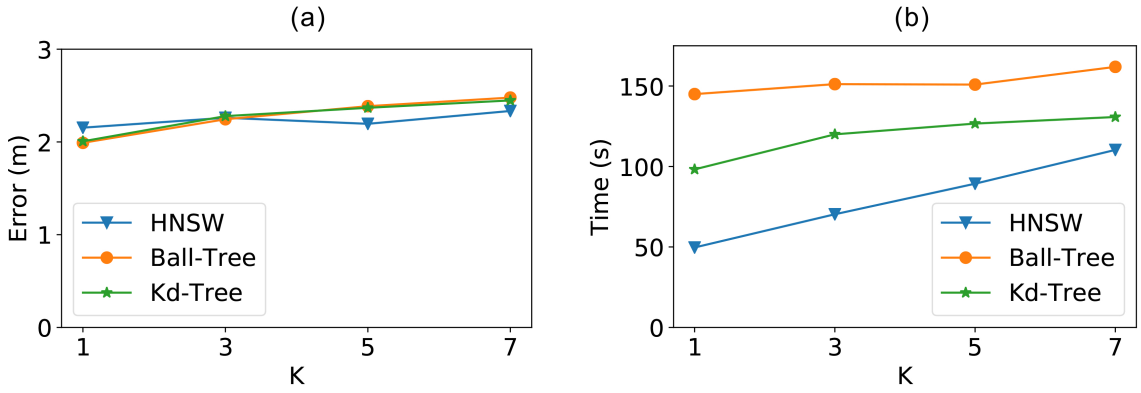

Figure 13: (a) Positioning error and (b) classification time when using the real-world UJI indoor dataset and varying the k value.

4.6 UJI Dataset Experiments

Finally, we also performed our experiments using the popular UJI Indoor Localization (Torres-Sospedra et al., 2014) dataset in order to evaluate the performance of the methods using real scenario data available in the literature. A point to take into account of this training dataset is that even though it has 529 access points (attributes), it only has approximately 20 thousand samples, since for each place determined for the collection of the samples there was little to no repetition, that is, only a few packets were sent by the mobile device from each training point.

Because of this, the training dataset of the UJI has only 10% the number of samples from our synthetic database. On the other hand, this dataset has 2.7 times the number of access points, which is an interesting scenario to analyze, being a very different case from our previous experiments. Since this is a relatively small dataset, and since we did not have much control over it, we kept all of the fixed scenario characteristics and changed the value of the k nearest neighbors to be found from 1 to 7 to perform the experiments.

Analyzing the results in Figure 13(b), the HNSW method was faster in all cases, being 66% faster than Ball-Tree and 50% faster than Kd-Tree for k equals to 1. In the figure, we can also see that the total time for HNSW increases at a higher rate than the other tree-based methods, showing the influence of the k value on the creation of the graph structure.

However, as depicted in Figure 13(a), the lowest positioning errors are obtained by the lowest k values, which can be expected since in this dataset there was almost no repetition of packets from the same location.

5 Conclusions

High efficiency is a key factor toward a precise, scalable, and real-time indoor positioning system. It can lead to a precise and robust positioning by allowing the increase in the number of samples per training point. It allows the scalability of the scenario by allowing the increase in the number of training points and also the increase in the number of access points. Finally, it allows real-time locations by being able to compute thousands or millions of positions every second.

Given the growing interest in applications based on IPSs, we have a growing area of research to look at. While these new applications and possibilities are starting to becoming real, new accurate and efficient positioning techniques will be required. Thus, we can clearly see the need for efficient and robust machine learning algorithms as we intend to scale these IPSs in large-scale, real-time scenarios.

Based on some recent work that has shown significant advances in the use of graph-based methods to look for the nearest neighbors in high-dimensional datasets, in this work we proposed the use of Hierarchical Navigable Small World graphs as an optimized data structure to be used in IPSs. We also created a simulation-based synthetic dataset as well as implemented a complete IPS testbed in a school building to evaluate the performance of the experimented solutions under different conditions.

Our results clearly show the performance gains of the proposed graph-based solution since it was able to estimate the position of the nodes 96% faster than the classic brute force kNN and at least 77% faster than the tree-based optimizations. All of these performance gains were obtained while maintaining almost the same positioning error in all of the experimented synthetic and real-world scenario datasets. Since our solution does not require the loss of information on the training database, most current techniques proposed in the literature to improve accuracy and performance can also be applied on top of our solution.

These results are very promising, but some advantages still need to be further exploited in future works. For instance, we intend to use the knowledge of the last known position of the mobile device, a data readily available in IPSs, to reach even faster the ground layer of the hierarchical navigable small world graph allowing for an almost instantaneous location of a node that has been located before.

Bibliography

Bahl, P. and V. N. Padmanabhan

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)*, volume 2, Pp. 775–784 vol.2. 14, 21

Bahl, P., V. N. Padmanabhan, and A. Balachandran

2000. Enhancements to the radar user location and tracking system. *Microsoft Research*, 2(MSR-TR-2000-12):775–784. 14

Bentley, J. L.

1975. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517. 16, 17

Boukerche, A., H. A. Oliveira, E. F. Nakamura, and A. A. Loureiro

2008. Vehicular ad hoc networks: A new challenge for localization-based systems. *Computer Communications*, 31(12):2838 – 2849. Mobility Protocols for ITS/VANET. 10

Boukerche, A., H. A. B. F. Oliveira, E. F. Nakamura, and A. A. F. Loureiro

2007. Localization systems for wireless sensor networks. *IEEE Wireless Communications*, 14(6):6–12. 10

Boytsov, L. and B. Naidan

2013. Engineering efficient and effective non-metric space library. In *Similarity Search and Applications - 6th International Conference, SISAP 2013, A Coruña, Spain, October 2-4, 2013, Proceedings*, Pp. 280–293. 20

Chen, Y., Q. Yang, J. Yin, and X. Chai

2006. Power-efficient access-point selection for indoor location estimation. *IEEE Transactions on Knowledge and Data Engineering*, 18(7):877–888. 12, 15, 16, 17

Chen, Y.-C., J.-R. Chiang, H.-h. Chu, P. Huang, and A. W. Tsui

2005. Sensor-assisted wi-fi indoor location system for adapting to environmental dynamics. In *Proceedings of the 8th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '05, Pp. 118–125, New York, NY, USA. ACM. 16, 27

Dong, W., C. Moses, and K. Li

2011. Efficient k-nearest neighbor graph construction for generic similarity measures.

In *Proceedings of the 20th international conference on World wide web*, Pp. 577–586. ACM. 16

Fang, S. H. and T. Lin

2012. Principal component localization in indoor wlan environments. *IEEE Transactions on Mobile Computing*, 11(1):100–110. 11, 12, 15

Farid, Z., R. Nordin, and M. Ismail

2013. Recent Advances in Wireless Indoor Localization Techniques and System. 00228. 10

Friis, H. T.

1946. A note on a simple transmission formula. *Proceedings of the IRE*, 34(5):254–256. 20

Guidoni, D. L., A. Boukerche, L. A. Villas, F. S. de Souza, H. A. Oliveira, and A. A. Loureiro

2012. A small world approach for scalable and resilient position estimation algorithms for wireless sensor networks. In *Proceedings of the 10th ACM International Symposium on Mobility Management and Wireless Access*, MobiWac '12, Pp. 71–78, New York, NY, USA. ACM. 12, 17

Haeberlen, A., E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki

2004. Practical robust localization over large-scale 802.11 wireless networks. In *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, MobiCom '04, Pp. 70–84, New York, NY, USA. ACM. 14

Houle, M. E. and M. Nett

2015. Rank-based similarity search: Reducing the dimensional dependence. *IEEE transactions on pattern analysis and machine intelligence*, 37(1):136–150. 16, 17

Khodayari, S., M. Maleki, and E. Hamedi

2010. A rss-based fingerprinting method for positioning based on historical data. In *Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2010 International Symposium on*, Pp. 306–310. 11

Kohavi, R. et al.

1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, Pp. 1137–1145. Montreal, Canada. 20

Kuo, S. P. and Y. C. Tseng

2011. Discriminant minimization search for large-scale rf-based localization systems. *IEEE Transactions on Mobile Computing*, 10(2):291–304. 12

Li, B., J. Salter, A. G. Dempster, and C. Rizos
2006. Indoor positioning techniques based on wireless LAN. In *Lan, First Ieee International Conference on Wireless Broadband and Ultra Wideband Communications*, Pp. 13–16. 00402. 11, 12

Lima, M. W. S., H. A. F. de Oliveira, E. M. dos Santos, E. S. de Moura, R. K. Costa, and M. Levorato
2018. Efficient and robust wifi indoor positioning using hierarchical navigable small world graphs. In *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, Pp. 1–5. IEEE. 10

Lin, T.-N. and P.-C. Lin
2005. Performance comparison of indoor positioning techniques based on location fingerprinting in wireless networks. In *2005 International Conference on Wireless Networks, Communications and Mobile Computing*, volume 2, Pp. 1569–1574 vol.2. 11, 14

Liu, H., H. Darabi, P. Banerjee, and J. Liu
2007. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1067–1080. 10

Malkov, Y., A. Ponomarenko, A. Logvinov, and V. Krylov
2014a. Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems*, 45:61–68. 16

Malkov, Y., A. Ponomarenko, A. Logvinov, and V. Krylov
2014b. Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems*, 45:61–68. 17, 19

Malkov, Y. A. and D. A. Yashunin
2016. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *ArXiv e-prints*. 6, 12, 16, 17, 18

Muja, M. and D. G. Lowe
2014. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (11):2227–2240. 16, 17

Omohundro, S. M.
1989. *Five balltree construction algorithms*, number ICSI Technical Report TR-89-063. International Computer Science Institute Berkeley. 16, 17

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau,

M. Brucher, M. Perrot, and E. Duchesnay
2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830. 20

Plets, D., W. Joseph, K. Vanhecke, E. Tanghe, and L. Martens
2013. Simple indoor path loss prediction algorithm and validation in living lab setting. *Wireless Personal Communications*, 68(3):535–552. 21

Ponomarenko, A., Y. Malkov, A. Logvinov, and V. Krylov
2011. Approximate nearest neighbor search small world approach. In *International Conference on Information and Communication Technologies & Applications*. 17

Pugh, W.
1990. Skip lists: a probabilistic alternative to balanced trees. *Communications of the ACM*, 33(6):668–676. 17

Rappaport, T.
2001. *Wireless Communications: Principles and Practice*, 2nd edition. Upper Saddle River, NJ, USA: Prentice Hall PTR. 20

Roos, T., P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen
2002. A probabilistic approach to wlan user location estimation. *International Journal of Wireless Information Networks*, 9(3):155–164. 14

Sayed, A. H., A. Tarighat, and N. Khajehnouri
2005. Network-based wireless location: challenges faced in developing techniques for accurate wireless location information. *IEEE Signal Processing Magazine*, 22(4):24–40. 11

Tao, X., X. Li, J. Ma, and J. Lu
2008. Cluster filtered knn: A wlan-based indoor positioning scheme. In *2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks(WOWMOM)*, volume 00, Pp. 1–8. 11

Torres-Sospedra, J., R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta
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)*, Pp. 261–270. 13, 23, 29

Yim, J.
2008. Introducing a decision tree-based indoor positioning technique. *Expert Systems with Applications*, 34(2):1296 – 1302. 11, 15, 16, 17

Youssef, M. and A. Agrawala

2005. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, Pp. 205–218. ACM. 15

Youssef, M. A., A. Agrawala, and A. U. Shankar

2003. Wlan location determination via clustering and probability distributions. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003).*, Pp. 143–150. 12, 15