Let's Talk about k-NN for Indoor Positioning: Myths and Facts in RF-based Fingerprinting

Joaquín Torres-Sospedra*, Cristiano Pendão*, Ivo Silva*, Filipe Meneses*, Darwin Quezada-Gaibor†, Raúl Montoliu†, Antonino Crivello‡, Paolo Barsocchi‡, Antoni Pérez-Navarro§ and Adriano Moreira*
*Centro ALGORITMI, Universidade do Minho, Campus de Azurém, 4800-058 Guimarães, Portugal
†Institute of New Imaging Technologies, Universitat Jaume I, Castellón, Spain
‡Institute of Information Science and Technologies, Consiglio Nazionale delle Ricerche, Pisa, Italy
§Faculty of Computer Sciences, Multimedia and Telecommunication, Universitat Oberta de Catalunya, Barcelona, Spain

***Abstract*—Microsoft proposed RADAR in 2000, the first indoor positioning system based on Wi-Fi fingerprinting. Since then, the indoor research community has worked not only to improve the base estimator but also on finding an optimal RSS data representation. The long-term objective is to find a positioning system that minimises the mean positioning error. Despite the relevant advances in the last 23 years, a disruptive solution has not been reached yet. The evaluation with non-open datasets and comparisons with non-optimized baselines make the analysis of the current status of fingerprinting for indoor positioning difficult. In addition, the lack of implementation details or data used for evaluation in several works make results reproducibility impossible. This paper focuses on providing a comprehensive analysis of fingerprinting with k-NN and settling the basement for replicability and reproducibility in further works, targeting to bring relevant information about k-NN when it is used as a baseline comparison of advanced fingerprint-based methods.**

***Index Terms*—Wi-Fi Fingerprinting, Received Signal Strength, k–Nearest Neighbor, Reproducibility, Replicability**

I. Introduction

Since the first introduction of Wi-Fi fingerprinting positioning systems, the pattern matching algorithm k–Nearest Neighbor (k–NN) has been one of the most used techniques to provide a position estimate [1], [2]. k–NN is a non-parametric supervised learning method introduced in 1951 [3], which is used in numerous applications other than indoor positioning. k–NN may be implemented as a data classifier or regressor, where the output is a class membership or the average of the values of k nearest neighbours, respectively.

In positioning applications such as fingerprinting, k–NN regression provides a position estimate by selecting the k closest samples to the operational sample from within the training dataset and then averaging the position coordinates of the closest samples to obtain a position estimate. On the other hand, k–NN classification fits better for symbolic positioning applications based on reference locations, rooms or floors. To find the closest samples, generic distance functions (e.g., *City Block*) are used to compute the distance in the signal space between the operational sample and all training samples.

Corresponding authors: J. Torres-Sospedra (`info@jtorr.es`) and C. Pendão (`cpendao@dsi.uminho.pt`).

The authors gratefully acknowledge funding from projects ORIENTATE H2020-MSCA-IF GA.101023072; FCT UIDB/00319/2020; CYTED Network "GeoLibero"; PID2021-122642OB-C42; and PID2021-122642OB-C44.

Although several efforts have been made to improve the standard k–NN algorithm in terms of accuracy [4]–[6] and efficiency [7], [8], new improved versions of k–NN –including the weighted k–NN variant– have emerged with no disruptive alterations with respect to the original algorithm.

Recent applications of k–NN include but are not limited to, Wi-Fi/Magnetic matching, floor detection or smartphone carrying mode [9]. Furthermore, researchers use the k–NN algorithm as a core positioning system while evaluating other features of the positioning pipeline, such as data augmentation to enrich the radio map [10], [11] or to reconstruct areas with missing reference points [12]. Finally, it is the de-facto baseline for fingerprint-based positioning [13]–[17].

Achieving the best possible performance with k–NN requires optimizing its hyperparameters, i.e. the k, the distance function [18], [19] and the weighting strategy for centroid computation. Procedures involving data manipulation may also improve positioning accuracy. We consider that external hyperparameters, such as data representation [18], should also be considered when optimizing the k–NN hyperparameters as significant improvements can be obtained. However, k–NN is still used with *default* hyperparameters when it is the baseline for comparison with other proposed methods. Unoptimized baselines may not represent k–NN real performance.

In addition, some work about advanced fingerprinting lack details to reproduce/replicate it. For instance, how to deal with non-detected values, the threshold to consider a signal weak or other relevant hyperparameters. It is not easy to reproduce the results and replicate the method if core information is missing.

This paper aims to enhance the reproducibility and replicability of methods in fingerprint-based indoor positioning. The contributions of this paper include:

- Identification of the relevant steps and core hyperparameters in the k–NN execution pipeline required to reproduce an algorithm based on k–NN;
- Analysis of the main k–NN hyperparameters using 69 datasets and statistical tests to assess significance;
- Comparison with state-of-the-art improvements of k–NN.

We provide access to code and datasets that can be used by the research community to improve the comparative benchmarking of new methods. Other researchers are encouraged to add their own datasets and code [20].

II. Related Work

This section describes RADAR, the first Wi-Fi fingerprinting system –which was based on k–NN– and subsequent enhanced versions found in the literature.

Bahl and Padmanabhan [1] introduced RADAR in 2000. In their analysis, they included a radio map with at least 20 samples, considering 70 positions and 4 orthogonal orientations. The proposed empirical evaluation was conservative as they picked one of the locations and orientations tuples randomly, and then applied the k–NN method using all samples from the remaining 69 points and 4 orientations. The evaluation was done in the corridors of an area of 22.5 m by 43.5 m and considered the Euclidean distance to compute the dissimilarities between fingerprints. The results reported a median error of 2.94 m for k = 1 and 2.75 m for k = 5, with degraded performance for larger k.

Cha [21] performed a generic comprehensive analysis of distance (dissimilarity) and similarity functions because of their relevance for many pattern recognition problems. They were reviewed and categorized according to both syntactic and semantic relationships. This work served as inspiration for similar analysis focusing on Received Signal Strength (RSS) fingerprinting [18], [22] using a public dataset each.

Shin *et al.* [23] introduced a two-stage procedure to dynamically set the value of k for every operational fingerprint. First, the Euclidean distance to all reference samples is computed, and only the samples with a distance below a threshold are selected. Then, an additional rule based on differences is applied to further filter the selected candidates. Then, the workflow of weighted k–Nearest Neighbor (wk–NN) is conducted using the inverse of the distance to weigh the locations when computing the centroids. The evaluation was performed in a test space of 48 m to 22 m. Unfortunately, no clue about the threshold is provided.

Liang *et al.* [4] proposed a novel RSS-fingerprint distance that mimics the physical geometric distances and weights the base stations (bs) according to their strength. The execution pipeline afterwards follows the traditional wk–NN where the weights for computing the centroids correspond to the squared inverse of the new physical distance. The empirical evaluation was carried out outdoors over one single GSM network in an urban area, an ideal test-bed to compare the performance of traditional k–NN and improved k–NN algorithm.

Zou *et al.* [24] applied the wk–NN with a novel weighting scheme, where the computation was leveraged to the Signal Tendency Index (STI) instead of the raw Received Signal Strength (RSS). The distance among fingerprints was calculated with the Euclidean distance over STI features and the weights for centroid computation corresponded to the inverse of the distance of the selected neighbours. The empirical experiments were conducted in a 35.6 m to 16.6 m laboratory for 6 months. The results showed that leveraging on STI features compensated device heterogeneity and reduced the overall positioning error with respect to raw RSS, but extended analysis on other distance functions was omitted.

Recently, Liu *et al.* [25] proposed a rule to dynamically set the k value up to a maximum reestablished value using two threshold values and the cosine similarity values of the nearest neighbours ranked in the range of $[1 \ldots k_{max}]$. Although the proposed method improves the traditional wk–NN, it introduced two new hyperparameters to optimize.

In the last 23 years, none of the several k–NN enhancements emerged as a clear winner. The evaluation setups among works differ, baselines do not share the same configuration, or some key elements are underexplored (e.g., distance function among fingerprints). Some of the k–NN enhanced versions introduce strict restrictions, e.g., asking for reference locations in regular grids or having multiple fingerprints per position, while others introduce new hyperparameters to optimize. Furthermore, the empirical evaluation is limited to a few scenarios, limiting the generalization of results. In this work, we perform a comprehensive analysis of k–NN and variants.

III. The pipeline for positioning with k–NN

This section describes the pipeline for fingerprinting-based positioning in a reproducible way. The provided description is technology and measurement-agnostic.

A. *Off-line phase: Data collection*

Data collection is a time-consuming task as the radio map or reference set (denoted by $\mathcal{T}$ in this paper) is built with real measured data. Ideally, the operational area should correspond to a medium/large size scenario with a realistic deployment of anchors and a good distribution of reference locations.

The collection itself is straightforward, one or several fingerprints are collected at every reference location with a single or multiple devices. Reference locations should be equally distributed following a regular grid if possible. Data collection and/or selection of locations can be delegated to users by means of crowdsourcing. Depending on the radio map, cross-validation with some restrictions can be applied to generate independent training ($\mathcal{T}$) and validation sets ($\mathcal{V}$), enabling an objective procedure to select hyperparameters.

Nevertheless, a secondary radio map is needed, acting as the evaluation or test set ($\mathcal{E}$ in this paper). Although collecting all data in a single procedure is also viable, it is not recommended. Very similar fingerprints may end up in the training and testing sets, thus leading to over-optimistic results. For fair evaluation, training, validation and test sets should all be independent.

Depending on the device, raw data can be gathered as individual readings or as a vector with one reading per detected emitter. In the former case, samples should be aggregated within an appropriate time window to generate fingerprint vectors. In the vectors, non-detected values must be identified.

Hardware and software limitations must be cross-checked and documented. For instance, the process to sense a single Wi-Fi RSS fingerprint should take around 1.5 s to 6 s, depending on the number of scanned radio channels. However, *Android* buffers the RSS values and allows collecting multiple fingerprints per second, which results in having exactly the same RSS values (and fingerprints) in a short time window.

We do recommend sharing the data and providing it in its rawest form, with the anchor ID, the transmission frequency, the received intensity values and the timestamp. In the case of receiving individual readings, we also recommend providing data in vector format and the procedure to generate them.

The steps described above serve for static data collection. In the case of collecting data while moving, high-accurate real-time ground truth is needed to label reference positions, similar to what was introduced by Daniş *et al.* [26], where the authors track the position of a device with cameras and a Bluetooth Low Energy (BLE) beacon in an area with markers. They process the video streams to obtain precise pose estimations and reduce the positional error to less than 5 cm. They annotate BLE data with the position to create a radio signal dataset for evaluating a radio signal-based localization system.

B. Data processing at offline and online phases

An important, but commonly uncredited, step in fingerprinting is data pre-processing. It includes how to represent the non-detected values with a numeric value, as generic distance functions cannot handle missing data. In RSS-based fingerprinting, missing data are often replaced with −200 dBm, −150 dBm, −110 dBm or the minimum value in the dataset minus 1. Nevertheless, the proper values depend on the kind of measurement and positioning technology.

Another relevant pre-processing procedure to consider is filtering weak signals. i.e., assigning the non-detected value to, for instance, those signals weaker than a threshold or keeping only the n strongest signals in the fingerprint. If this optional step is applied, the full procedure should be documented.

Choosing the proper RSS data representation is a relevant procedure. Some machine learning models only work when input data are scaled to $[0,\dots,1]$ range. In k–NN, some distance functions do not work as expected if the input data are negative (e.g. Sorensen or Cosine [18], [21], [22]) and RSSs should be shifted to positive. Finally, the measurements in RSS-based fingerprinting are not linear, and some alternatives tried to mimic the Log Distance Path Loss model [4], [18].

Finally, the way different fingerprints collected at the same location are aggregated is also included in this pre-processing step. Some authors use all raw samples, as in RADAR, while others average fingerprints collected in the same location.

We do recommend sensible pre-processing in both, offline & online phases, such as choosing a proper data representation [18]. However, we do not recommend averaging fingerprints at the online phase as it requires being in the same location for a long period, which is not compatible with indoor navigation.

C. On-line phase: Data processing

The high-level description of k–NN-based positioning is provided in Algorithm 1, requiring 3 hyperparameters: 1) the function $dist$ to compute distances among fingerprints [21], 2) the value of k, and 3) the strategy to compute centroids. We assume infinite distance if j-th training $s_j^{\mathcal{T}}$ and i-th evaluation $s_i^{\mathcal{E}}$ fingerprints (ln.4) do not share any common emitter.

Algorithm 1 Pseudocode of k–NN for positioning

1: **input** $\mathcal{T}$, $\mathcal{E}$, k, $dist$, $centroid$
2: **for** i = 1 to $|\mathcal{E}|$ **do**
3:     **for** j = 1 **to** $|\mathcal{T}|$ **do**
4:         1 Compute RSS distances $d_j = dist(s_i^{\mathcal{E}}, s_j^{\mathcal{T}})$
5:     **end for**
6:     Sort distances in RSS space
7:     2 Select k' nearest training samples (lowest d): ns_i
8:     3 Compute centroid: $pos_i = centroid(pos^{\mathcal{T}}, ns_i)$
9: **end for**
10: Return: Estimated positions

Once distances are computed (ln.5), the nearest neighbours (ns_i) correspond to the samples with the lowest distance values d_j. If multiple neighbors share the k-th distance after sorting, we choose them all. We thus consider k' nearest neighbours (ln.7). Some authors use a similarity function in ln.4 (sim_j) instead of a distance one. In those cases, the nearest neighbours are those with higher similarities instead of lower distances.

The strategies for calculating the position pos_i (ln.8, eq.1) are: *unweighted centroid* and *weighted centroid* based either on inverse distance or on squared inverse distance. All weights are scaled so that their sum equals 1.

$$pos_i = centroid(pos^{\mathcal{T}}, ns_i) = \sum_{l=1}^{k'} \left(\frac{w_l \cdot pos^{\mathcal{T}}_{ns_i(l)}}{\sum_{m=1}^{k} w_m} \right) \quad (1)$$

$$\text{where: } w_l = \begin{cases} 1 & \text{unweighted centroid} \\ (d_l)^{-1} & \text{weighted centroid inverse distance} \\ (d_l)^{-2} & \text{weighted centroid sq. inverse dist.} \end{cases}$$

when computing w_l, if a similarity function (sim_j) is used in ln.4, distances d_j must be replaced with $(1 - sim_j)$ if $[0\dots1]$-scaled or $(sim_j)^{-1}$. Handling *div by zero* must be documented, we added small value $\epsilon = 10^{-8}$ to any distance.

IV. Empirical Analysis and Results

A. Experimental setup

To compare k–NN and improved variants we applied them over ***51 Wi-Fi datasets***: DSIn [27], LIBn [28], MANn [29], MINT1 [30], SAH1 & TIE1 [31], TUTn [32]–[37], UJIn [38], UTS [39], OFIN [40], GPRn [41], SODn [14], KIOSn [42], EEILn [43]; ***9 BLE datasets***: OFINBn [40], UEXBn [44], UJIBn [45]; and ***9 hybrid datasets***: HDBn [46]. The implemented methods and dataset descriptions are provided in [20] for research reproducibility and replicability [47].

We report the mean 3D positioning error ϵ_{3D} for each dataset. As aggregated metrics, we provide the average error considering all the 112239 individual evaluation samples (All) from the 69 datasets, and the average error considering the 69 mean values provided per dataset (DB). For aggregated results, we also report the normalized values (All n. and DB n.) with respect to baseline $\mathcal{C}_1$ (see Section IV-B). T-student, Anova1 and Wilcoxon test (tt, $a1$ and w in Table I) are used to check whether differences among results are significant ✓ or not ✗.

B. Analysis of hyperparameters in k–NN

First, we analyse the k–NN hyperparameters, comparing 4 configurations of k–NN: $\mathcal{C}_1$) default values; $\mathcal{C}_2$) optimal distance; $\mathcal{C}_3$) optimal distance & k tuple; and $\mathcal{C}_4$) optimal set of all hyperparameters. We use *City Block* distance, k=1 and unweighed centroid and positive data representation for RSS [18] as default configuration. Table I provides all results.

TABLE I
RESULTS FOR k–NN UNDER 4 HYPERPARAMETER CONFIGURATIONS

	ϵ_{3D}				$\mathcal{C}_2$ vs. $\mathcal{C}_1$				$\mathcal{C}_3$ vs. $\mathcal{C}_2$				$\mathcal{C}_4$ vs. $\mathcal{C}_3$				$\mathcal{C}_4$ vs. $\mathcal{C}_1$			
dataset	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$	$\mathcal{C}_4$	df	tt	al	w	df	tt	al	w	df	tt	al	w	df	tt	al	w
DSI1	4.95	4.45	3.64	**3.56**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
DSI2	4.95	4.45	3.72	**3.65**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
LIB1	3.02	3.02	2.44	**2.42**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
LIB2	4.19	2.86	2.35	**2.34**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
MAN1	2.82	2.82	2.14	**2.12**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
MAN2	2.47	2.14	**1.61**	**1.61**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
MINT1	2.67	2.67	**2.16**	**2.16**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
SAH1	9.07	8.55	5.96	**5.93**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
TIE1	6.55	4.12	**2.36**	**2.36**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✱	▲	✓	✓	✓
TUT1	9.57	5.27	**4.43**	**4.43**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
TUT2	14.37	10.19	8.75	**8.37**	▲	✓	✓	✓	▲	✓	✗	✓	▲	✗	✗	✗	▲	✓	✓	✓
TUT3	9.25	8.75	8.37	**7.76**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✓	✓	✓	▲	✓	✓	✓
TUT4	6.36	5.96	5.33	**5.20**	▲	✗	✗	✓	▲	✓	✗	✓	▲	✗	✗	✗	▲	✓	✓	✓
TUT5	6.92	5.65	**5.22**	**5.22**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
TUT6	1.87	**1.82**	**1.82**	**1.82**	▲	✗	✗	✗	=	✗	✗	✱	=	✗	✗	✱	▲	✗	✗	✗
TUT7	2.23	2.10	2.10	**2.04**	▲	✗	✗	✗	=	✗	✗	✱	▲	✗	✗	✓	▲	✓	✓	✓
UJI1	9.62	8.11	7.36	**7.33**	▲	✓	✓	✓	▲	✓	✗	✓	▲	✗	✗	✓	▲	✓	✓	✓
UJI2	8.04	7.59	**6.04**	**6.04**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
UTS1	8.44	7.85	6.51	**6.50**	▲	✗	✗	✓	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
OFIN1	2.30	2.08	1.57	**1.55**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
GPR00	2.95	2.95	**2.30**	**2.30**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
GPR01	2.86	2.86	**2.29**	**2.29**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
GPR02	3.29	3.24	**2.35**	**2.35**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR03	3.35	3.33	**2.39**	**2.39**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR04	3.06	3.00	**2.29**	**2.29**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR05	3.27	3.27	**2.35**	**2.35**	=	✗	✗	✱	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR06	2.98	2.96	**2.37**	**2.37**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR07	2.98	2.97	**2.41**	**2.41**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR08	2.73	2.70	**2.31**	**2.31**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
GPR09	2.97	2.89	**2.36**	**2.36**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR10	2.71	2.67	2.28	**2.27**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
GPR11	3.02	2.94	**2.35**	**2.35**	▲	✗	✗	✗	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
GPR12	2.85	2.69	2.37	**2.36**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
GPR13	2.96	2.75	**2.43**	**2.43**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
SOD01	3.26	2.78	**2.43**	**2.43**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
SOD02	1.99	1.95	**1.54**	**1.54**	▲	✗	✗	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
SOD03	2.02	1.99	**1.61**	**1.61**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
SOD04	3.30	2.90	2.01	**2.00**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
SOD05	3.43	3.34	2.88	**2.83**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
SOD06	3.89	3.89	**3.47**	**3.47**	=	✗	✗	✱	▲	✓	✗	✓	=	✗	✗	✱	▲	✓	✗	✓
SOD07	4.17	4.17	4.16	**3.25**	=	✗	✗	✱	▲	✗	✗	✗	▲	✓	✓	✓	▲	✓	✓	✓
SOD08	4.04	3.95	3.63	**3.62**	▲	✗	✗	✗	▲	✗	✗	✓	▲	✗	✗	✓	▲	✓	✗	✓
SOD09	4.55	4.48	3.55	**3.54**	▲	✗	✗	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
KIOS0	2.26	2.26	1.89	**1.87**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
KIOS1	2.35	2.13	1.82	**1.80**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
KIOS2	2.27	2.27	1.85	**1.83**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
KIOS3	1.78	1.78	1.58	**1.56**	=	✗	✗	✱	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
KIOS4	2.53	2.46	**1.97**	**1.97**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
KIOS5	2.09	2.03	1.76	**1.73**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
EEIL01	4.05	3.94	3.45	**3.41**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
EEIL02	4.51	4.34	3.84	**3.83**	▲	✗	✗	✗	▲	✓	✗	✓	▲	✗	✗	✗	▲	✓	✓	✓
OFINB1	3.55	2.30	**1.75**	**1.75**	▲	✓	✓	✓	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
OFINB2	4.28	2.73	**2.34**	**2.34**	▲	✓	✓	✓	▲	✗	✗	✗	=	✗	✗	✱	▲	✓	✓	✓
OFINB3	1.61	1.49	**1.20**	**1.20**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
OFINB4	3.35	2.53	**2.16**	**2.16**	▲	✓	✓	✓	▲	✓	✗	✓	=	✗	✗	✱	▲	✓	✓	✓
UEXB1	3.61	3.53	2.86	**2.81**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
UEXB2	4.80	4.28	4.01	**3.83**	▲	✗	✗	✗	▲	✗	✗	✗	▲	✗	✗	✓	▲	✓	✓	✓
UEXB3	7.24	6.99	6.18	**6.07**	▲	✗	✗	✱	▲	✗	✗	✓	▲	✗	✗	✗	▲	✓	✓	✓
UJIB1	3.09	2.86	**1.64**	**1.64**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
UJIB2	4.42	3.70	**2.04**	**2.04**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✗	▲	✓	✓	✓
HDB11	**1.34**	**1.34**	**1.34**	**1.34**	▲	✗	✗	✗	=	✗	✗	✱	=	✗	✗	✱	▲	✗	✗	✗
HDB12	2.35	2.30	2.03	**1.99**	▲	✗	✗	✗	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
HDB13	4.01	2.54	2.13	**2.08**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
HDB21	2.54	2.49	2.21	**2.19**	▲	✗	✗	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
HDB22	**1.77**	**1.77**	**1.77**	**1.77**	=	✗	✗	✱	=	✗	✗	✱	=	✗	✗	✱	=	✗	✗	✱
HDB23	2.85	2.40	2.09	**2.06**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓
HDB31	2.83	2.77	**2.23**	**2.23**	▲	✗	✗	✓	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
HDB32	2.75	2.64	**2.22**	**2.22**	▲	✓	✓	✓	▲	✓	✓	✓	=	✗	✗	✱	▲	✓	✓	✓
HDB33	**1.33**	**1.33**	**1.33**	**1.33**	▲	✗	✗	✗	=	✗	✗	✱	=	✗	✗	✱	▲	✗	✗	✗
All	3.37	3.12	2.69	**2.64**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✓	✓	✓	▲	✓	✓	✓
DB	3.94	3.51	2.92	**2.88**	▲	✗	✗	✓	▲	✓	✗	✓	▲	✗	✗	✓	▲	✓	✓	✓
All n.	1.00	0.94	0.82	**0.81**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✓	✓	✓	▲	✓	✓	✓
DB n.	1.00	0.92	0.77	**0.76**	▲	✓	✓	✓	▲	✓	✓	✓	▲	✗	✗	✓	▲	✓	✓	✓

✓ stands for performances statistically different; ✗ stands for equivalent performances; ✱ Wilconox could not be applied

When optimizing hyperparameters for a dataset, we consider k values in set $k \in [1, 2, \ldots, 19, 20] \cup [21, 23, \ldots, 49, 51]$. For distances, we considered *City Block*, *Euclidean*, *Minkowsky* 3, *Cosine*, *Sorensen*, *Neyman* (both sides), *LGD*, *PLGD* with penalty weights of $\alpha = 1/10$ and $\alpha = 1/40$. In all cases, we use positive RSS data representation. For centroid computation, we used the strategies after Eq. 1. Optimal hyperparameters refer to the combination with the lowest error.

When comparing $\mathcal{C}_X$ vs. $\mathcal{C}_Y$, the statistical results are summarized as follows. Symbol ▲ in columns *df* means that $\mathcal{C}_X$ reports a lower mean positioning error than $\mathcal{C}_Y$, while the cell color intensity indicates how many tests reported $\mathcal{C}_X$ being better than $\mathcal{C}_Y$. If both means agree, symbol = is provided in *df*. There is no case of $\mathcal{C}_Y$ reporting a lower error than $\mathcal{C}_X$.

Selecting only the optimal distance function, $\mathcal{C}_2$, has global reduction of error of around 6%–8% but it is statistically better with the 3 tests than $\mathcal{C}_1$ in 25 datasets. We used positive data rep. [18] as some distances only work with positive values.

Selecting the most appropriate distance function and k together, $\mathcal{C}_3$, highly improved the results. The global reduction of error is of around 18%–23% with respect to $\mathcal{C}_1$. The statistical results show that the differences of $\mathcal{C}_3$ with respect to $\mathcal{C}_2$ are statistically different in almost all datasets.

Selecting the most appropriate distance function, k, and weighting strategy, $\mathcal{C}_4$, is the least important, with a marginal improvement with respect to $\mathcal{C}_3$. In that case, the results provided by $\mathcal{C}_4$ are better –considering the 3 statistical tests– than $\mathcal{C}_3$ only for 2 datasets. The RSS measurements are noisy and, therefore, the RSS distances are not correlated to the geometric distances, i.e., the closest fingerprint in the RSS space may not match the closest fingerprint in the geometric (real-world) space, being a weighted centroid not optimal. $\mathcal{C}_4$ global improvement is around 19%–24% with respect to $\mathcal{C}_1$.

C. Comparison of k–NN implementations

Once the limits for k–NN are explored, we compare some improved versions. In addition to the normalized baseline (1-NN) in aggregated results, we include 3 k–NN baselines: $\mathcal{M}_1$) optimal k value and unweighted centroid; $\mathcal{M}_2$) optimal k value and weighted centroid (inv. dist.); $\mathcal{M}_3$) optimal k value and weighted centroid (sq. inv. dist.); In $\mathcal{M}_{1-3}$, we only consider *City Block* distance, as most of improved k–NN methods do not explore other distances. In this comparison, we also include $\mathcal{M}_{4-5}$) Adaptive wk–NN [25] with $k_{max} = 51$ and optimal k_{max} based on wk–NN, both with optimal thresholds values ($\tau_P \in [0.8, 0.85, 0.90, 0.95, 0.99]; \tau_S \in [0.20, 0.21, \ldots, 0.99, 1.00]$); $\mathcal{M}_{6-7}$) SAWKNN [48] with $k_{max} = 51$ and optimal k_{max} based on wk–NN, both with optimal threshold ($\gamma_{th} \in [0.01, 0.02, \ldots, 0.99, 1.00]$); $\mathcal{M}_8$) STIWKNN [24] with optimal k; $\mathcal{M}_{9-10}$) DWFWKNN [4] with the physical distance provided by authors and our implementation; and $\mathcal{M}_{11-13}$) ARWKNN [49] with optimal k for *City Block*, *Min-Max* and *Clark* distances. In $\mathcal{M}_{4-7}$, *optimal* refers to the value for a hyperparameter providing the lowest error. In $\mathcal{M}_{12}$, positive data rep. [18] was used with the *Min-Max* distance. The full results are reported in Table II.

TABLE II
RESULTS FOR k–NN, Wk–NN AND ENHANCED IMPLEMENTATIONS

	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_5$	$\mathcal{M}_6$	$\mathcal{M}_7$	$\mathcal{M}_8$	$\mathcal{M}_9$	$\mathcal{M}_{10}$	$\mathcal{M}_{11}$	$\mathcal{M}_{12}$	$\mathcal{M}_{13}$
DSI1	4.09	4.05	4.03	3.86	3.83	4.10	4.03	**3.82**	**27.63**	4.14	**4.78**	3.91	3.90
DSI2	4.22	4.17	4.11	3.85	3.88	4.11	4.05	**3.83**	**27.88**	4.11	**4.90**	3.95	3.93
LIB1	2.44	2.43	2.42	2.47	2.44	2.43	**2.41**	2.44	**4.11**	2.52	2.47	2.61	**2.90**
LIB2	3.70	3.70	3.69	**2.64**	**2.64**	3.66	3.68	2.68	**5.59**	3.35	**4.20**	2.69	3.25
MAN1	2.17	2.15	**2.14**	2.35	2.31	**2.14**	2.15	2.29	**5.11**	2.18	**2.47**	2.41	2.43
MAN2	1.69	1.70	1.71	**1.62**	**1.62**	1.70	1.69	1.65	**3.31**	1.67	2.15	**2.18**	1.90
MINT1	2.16	2.16	2.16	2.31	2.27	2.16	2.16	**2.40**	**21.06**	**2.12**	2.39	2.39	2.37
SAH1	6.47	6.45	6.44	**7.62**	7.25	6.45	6.40	6.65	**29.71**	**6.01**	7.55	7.31	6.39
TIE1	**5.38**	**5.38**	**5.38**	7.09	7.12	**5.38**	**5.38**	7.29	**15.27**	**8.09**	5.98	5.45	7.25
TUT1	8.28	8.20	8.06	6.21	**6.19**	7.67	8.20	6.28	**11.87**	7.70	**9.40**	6.93	6.95
TUT2	11.23	11.02	10.85	8.83	8.83	11.17	11.17	**8.78**	**16.22**	9.96	**12.40**	9.72	9.63
TUT3	9.06	8.65	8.36	8.33	8.37	8.35	8.37	**8.24**	**20.55**	8.57	**9.16**	8.34	8.39
TUT4	5.68	5.57	5.52	5.89	5.82	5.76	5.58	5.80	**20.27**	5.81	**6.25**	5.68	**5.49**
TUT5	6.29	6.23	6.19	**5.82**	5.92	6.14	6.19	5.93	**16.38**	6.39	**6.92**	6.11	5.84
TUT6	1.87	1.87	1.87	**2.12**	2.11	**1.76**	1.87	2.11	**14.32**	1.95	1.88	1.82	2.01
TUT7	2.23	2.23	2.15	**2.53**	2.51	**2.10**	2.23	2.51	**14.34**	2.23	2.24	2.11	2.23
UJI1	8.62	8.61	8.60	7.59	**7.57**	8.52	8.59	7.59	**29.51**	7.97	**9.41**	7.90	7.84
UJI2	6.35	6.33	6.31	6.66	6.62	6.41	6.31	6.60	**21.13**	**6.03**	**7.02**	6.72	6.53
UTS1	7.54	7.51	7.49	6.51	**6.46**	7.52	7.42	6.55	**16.28**	7.03	**7.88**	7.30	7.65
OFIN1	1.69	1.68	**1.67**	1.71	1.72	1.72	**1.67**	1.72	**6.38**	1.79	**1.90**	1.77	1.85
GPR00	**2.30**	**2.30**	2.32	2.31	2.31	2.42	2.42	2.39	**4.00**	**2.30**	**2.68**	2.63	2.56
GPR01	**2.29**	**2.29**	2.30	2.30	2.30	2.38	2.38	2.38	**3.99**	**2.29**	**2.63**	2.57	2.49
GPR02	**2.35**	2.36	2.37	2.37	2.37	2.44	2.44	2.42	**3.24**	**2.35**	**2.75**	2.74	2.62
GPR03	**2.40**	**2.40**	2.41	2.43	2.43	2.51	2.51	2.48	**3.74**	**2.40**	2.80	**2.81**	2.72
GPR04	**2.29**	**2.29**	2.30	2.32	2.32	2.38	2.38	2.39	**3.50**	**2.29**	**2.63**	2.60	2.55
GPR05	**2.36**	2.37	2.37	2.38	2.38	2.47	2.47	2.42	**4.43**	**2.36**	2.77	**2.80**	2.72
GPR06	2.39	2.39	2.39	2.38	2.38	2.45	2.45	2.43	**5.32**	**2.37**	2.69	**2.72**	2.51
GPR07	**2.42**	2.43	2.44	2.43	2.43	2.49	2.49	2.54	**5.46**	**2.42**	2.70	**2.72**	2.60
GPR08	2.38	2.37	2.36	2.37	2.35	2.41	2.38	2.38	**3.25**	**2.31**	**2.55**	2.53	2.47
GPR09	2.42	2.42	2.42	2.39	2.39	2.44	2.44	2.44	**3.57**	**2.36**	**2.74**	2.70	2.56
GPR10	2.34	2.33	2.33	2.35	2.32	2.37	2.34	2.39	**3.23**	**2.28**	**2.54**	2.53	2.40
GPR11	2.42	2.42	2.42	2.38	2.38	2.44	2.44	2.45	**3.73**	**2.36**	**2.82**	2.77	2.57
GPR12	2.46	2.45	2.44	2.39	2.40	2.48	2.45	2.42	**3.15**	**2.37**	**2.77**	2.74	2.42
GPR13	2.54	2.54	2.54	2.47	2.47	2.55	2.55	2.53	**4.90**	**2.45**	**2.87**	2.85	2.51
SOD01	2.65	2.65	2.64	2.98	2.84	2.67	2.64	2.83	**5.45**	2.66	**3.16**	2.98	**2.50**
SOD02	1.72	1.71	1.70	1.93	1.99	1.68	1.68	2.00	**4.38**	**2.48**	1.85	**1.65**	1.92
SOD03	1.80	1.80	1.79	1.97	1.98	**1.73**	1.79	2.07	**7.20**	**2.46**	1.90	1.79	1.82
SOD04	2.50	2.50	2.50	4.09	4.09	**2.48**	**2.48**	4.16	**5.33**	**4.66**	2.80	2.92	2.96
SOD05	2.93	2.92	2.91	**3.70**	3.14	**2.81**	2.86	3.17	**4.93**	3.50	3.18	3.09	2.86
SOD06	3.72	3.73	3.73	3.96	3.96	3.64	**3.60**	4.04	**8.14**	**4.70**	3.85	3.83	3.96
SOD07	**4.17**	**3.25**	3.26	3.46	3.43	3.29	3.27	3.46	**5.61**	4.11	3.58	3.62	3.77
SOD08	3.74	3.76	3.75	3.96	3.98	**3.64**	**3.64**	4.15	**9.07**	**4.42**	3.99	4.12	4.27
SOD09	**3.61**	3.63	3.65	3.98	3.86	3.73	3.62	4.08	**7.69**	**4.29**	3.76	3.77	4.09
KIOS0	1.90	1.89	1.88	1.93	1.94	1.90	1.89	1.94	**12.47**	**1.86**	**2.09**	2.08	1.95
KIOS1	1.85	1.83	1.84	1.88	1.88	**1.80**	**1.80**	1.92	**11.21**	1.83	**2.09**	2.07	2.03
KIOS2	1.85	1.84	**1.83**	1.99	1.99	1.89	**1.83**	1.99	**10.34**	1.86	**2.09**	2.06	2.01
KIOS3	1.58	1.57	1.57	1.71	1.71	1.57	**1.56**	**1.72**	**11.52**	**1.56**	1.71	1.69	1.62
KIOS4	2.06	2.06	2.05	2.19	2.18	2.09	2.06	2.17	**11.34**	**2.01**	**2.29**	2.22	2.04
KIOS5	1.83	1.81	1.80	1.83	1.86	1.82	1.80	1.88	**11.10**	**1.77**	**1.95**	1.94	1.83
EEIL01	3.45	3.43	3.41	3.73	3.60	3.45	**3.39**	3.63	**9.37**	**3.92**	3.78	3.52	3.69
EEIL02	3.90	3.89	**3.88**	4.05	4.01	3.95	3.89	4.03	**14.17**	**4.22**	4.08	3.92	4.07
OFINB1	2.54	2.62	2.69	2.34	2.34	2.92	2.70	**1.84**	**3.14**	2.24	**3.17**	2.18	2.30
OFINB2	3.76	3.78	3.79	**2.57**	2.62	3.80	3.70	2.72	**4.69**	2.85	**4.46**	3.26	2.72
OFINB3	1.20	1.20	1.20	**1.64**	**1.64**	1.20	**1.19**	1.30	1.31	1.25	1.29	**1.62**	1.50
OFINB4	2.77	2.80	2.82	2.65	2.32	3.25	2.84	**2.19**	**6.33**	2.80	**3.35**	2.70	2.84
UEXB1	3.23	3.15	3.10	**3.35**	**2.85**	3.08	3.07	2.91	**10.82**	3.19	3.29	3.27	3.19
UEXB2	4.40	4.29	4.20	4.28	3.97	4.07	4.07	**3.94**	**8.63**	4.17	4.55	**4.61**	4.28
UEXB3	6.45	6.32	6.24	**6.01**	6.19	**6.66**	6.51	6.08	**9.52**	6.16	6.25	6.35	6.36
UJIB1	1.81	1.79	1.78	1.71	1.69	1.85	1.80	1.65	**2.21**	**1.60**	2.14	2.05	**2.25**
UJIB2	2.75	2.71	2.67	2.58	**2.57**	2.68	2.67	2.62	**3.83**	2.59	**3.39**	3.33	3.02
HDB11	1.34	1.34	1.34	1.83	**1.90**	1.33	1.34	1.85	**2.27**	1.53	1.32	**1.31**	1.76
HDB12	2.03	2.02	2.01	**1.96**	**1.96**	1.97	1.97	2.02	**3.11**	2.11	**2.33**	2.26	2.15
HDB13	3.38	3.37	3.37	**2.08**	**2.08**	3.33	3.32	2.10	**6.99**	4.93	3.90	2.84	**5.69**
HDB21	2.26	2.26	2.26	**2.20**	**2.20**	2.26	2.26	**2.20**	**2.80**	2.27	2.33	2.35	**2.51**
HDB22	1.77	1.77	1.77	1.94	1.95	**1.59**	1.77	**2.00**	**4.35**	1.89	1.63	1.66	1.94
HDB23	2.47	2.46	2.46	**2.07**	**2.07**	2.46	2.46	2.08	**10.88**	3.65	2.83	2.46	**4.42**
HDB31	2.48	2.48	2.48	**2.38**	**2.38**	2.45	2.45	**2.38**	**2.79**	2.60	2.60	2.65	**2.86**
HDB32	2.39	2.39	2.39	**2.29**	**2.29**	2.35	2.35	2.31	**3.09**	2.53	2.52	2.54	**2.81**
HDB33	1.33	1.33	1.33	1.61	**1.63**	**1.26**	1.33	**1.63**	**4.43**	1.37	1.32	1.33	1.46
All	2.93	2.90	2.88	2.88	2.88	**2.87**	2.89	2.89	**8.63**	3.04	3.16	2.95	**3.17**
DB	3.31	3.27	3.26	3.25	**3.21**	3.28	3.27	3.22	**9.03**	3.40	**3.62**	3.35	3.42
All n.	0.88	0.87	0.87	0.90	0.89	**0.86**	0.87	0.90	**2.84**	0.93	0.94	0.90	**0.97**
DB n.	0.84	**0.83**	**0.83**	0.85	0.84	**0.83**	**0.83**	0.85	**2.44**	0.88	**0.92**	0.87	0.90

Background meaning: lowest error; highest error; 2nd highest error.

Overall, all methods have similar aggregated performance, except $\mathcal{M}_{9-13}$, and none of them fit in all the evaluated scenarios. In a few datasets, the improved versions have significantly better performance. As commonly done in the literature, the distance function is not optimized in the baselines, $\mathcal{M}_{1-3}$, so improvements may be over-optimistic.

Adaptive wk–NN ($\mathcal{M}_{4-5}$) and Self-Adaptive wk–NN ($\mathcal{M}_{6-7}$) are promising compared to the baselines. On the other hand, some proposed methods need further adjustments to operate. The physical distance in DWFWKNN is not working ($\mathcal{M}_9$), while our implementation is the best choice for some cases. ARWKNN was proposed for light-based positioning, providing weak results for RSS (e.g., $\mathcal{M}_{11}$).

Comparing the results in Table II, the improved versions of k–NN in the literature $\mathcal{M}_{4-13}$ provide lower positioning errors than the three baselines in 55 out of 69 datasets, being the aggregated results better in $\mathcal{M}_{5-7}$. Considering the results in Table I, k–NN with $\mathcal{C}_4$ provides the best overall results for 54 out of 69 datasets. For the remaining 15 datasets, $\mathcal{M}_{4-13}$ is few cm better than $\mathcal{C}_4$ (less than 5 cm in 11 datasets, ≈6 cm in 3 datasets), except for HDB22 where the difference is 18 cm.

V. DISCUSSION & CONCLUSIONS

The k–NN model is recurrently used in those fingerprint-based solutions, either as an improved method or as a baseline to compare the other advanced machine learning models. This work presents a deep analysis of this method.

Our work confirms that settling the optimal k value for each dataset is the most relevant hyperparameter. As already proved in the literature, k should be chosen according to the density of fingerprints available. Similarly, the distance function is also relevant and should be tuned for each dataset. Furthermore, we may take care of RSS data representation as some distance metrics do not work with negative values. The way to compute the position estimate (centroid) seems to not be relevant.

However, k–NN is still applied with a *default* configuration, such as low k and *City Block* distance when used as a baseline. We demonstrate that a wise selection of hyperparameters matters. As researchers, we need to compare our proposed methods to State-of-the-Art models and baselines with optimized parameters, avoiding comparisons to un-optimized baselines which led to over-optimistic error reductions.

The most important outcome is that a single hyperparameter configuration is not generalizable and does not fit well in all scenarios. As researchers, we do not only have to provide the method description, but also the fair procedure to set its hyperparameters. What is a trivial procedure for k–NN, can be complex in advanced neural networks where the weight initialization procedure, the number of hidden layers, the number of neurons and the transfer function –among others– are key hyperparameters. In those cases, replicating the same structure published in the literature in a completely different indoor scenario may end up in a suboptimal estimator.

Similarly, among the several proposed improved versions of k–NN, none of them emerges as a clear winner as the positioning performance depends on the case (dataset). When proposing a new method, we must ensure replicability by means of extensive evaluation in different and diverse scenarios. i.e., a method that fits very well in a database should not poorly perform in another scenario.

Having a diverse set of datasets is essential to ensure the generalization of a proposed method. However, the analysis presented in this work is based on test data; we also need datasets with independent training, validation and test subsets. We do recommend sharing datasets in its rawest form.

We also recommend sensible data pre-processing. In future works, we will explore other hyperparameters involved in the fingerprinting workflow, such as data representation, as well as include advanced machine learning models in the comparison.

References

[1] P. Bahl and V. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings IEEE INFOCOM 2000*, vol. 2, 2000, 775–784 vol.2.

[2] T.-N. Lin and P.-C. Lin, "Performance comparison of indoor positioning techniques based on location fingerprinting in wireless networks," in *2005 Int. Conf. on Wireless Networks, Communications and Mobile Computing*, vol. 2, 2005, pp. 1569–1574.

[3] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," *International Statistical Review*, vol. 57, no. 3, pp. 238–247, 1989.

[4] X. Liang *et al.*, "Fingerprint-based location positoning using improved knn," in *2012 3rd IEEE Int. Conf. on Network Infrastructure and Digital Content*, 2012, pp. 57–61.

[5] P. Dai *et al.*, "Combination of dnn and improved knn for indoor location fingerprinting," *Wireless Communications and Mobile Computing*, vol. 2019, p. 4 283 857, 2019.

[6] B. Wang *et al.*, "A novel weighted knn algorithm based on rss similarity and position distance for wi-fi fingerprint positioning," *IEEE Access*, vol. 8, pp. 30 591–30 602, 2020.

[7] G. Caso *et al.*, "A mixed approach to similarity metric selection in affinity propagation-based wifi fingerprinting indoor positioning," *Sensors*, vol. 15, no. 11, pp. 27 692–27 720, 2015.

[8] A. Kushki *et al.*, "Kernel-based positioning in wireless local area networks," *IEEE transactions on mobile computing*, vol. 6, no. 6, pp. 689–705, 2007.

[9] F. Potortì *et al.*, "Off-line evaluation of indoor positioning systems in different scenarios: The experiences from ipin 2020 competition," *IEEE Sensors Journal*, vol. 22, no. 6, 2022.

[10] G. G. Anagnostopoulos and A. Kalousis, "Proxyfaug: Proximity-based fingerprint augmentation," in *2021 Int. Conf. on Indoor Positioning and Indoor Navigation*, 2021.

[11] Y. Dong *et al.*, "A wifi fingerprint augmentation method for 3-d crowdsourced indoor positioning systems," in *2022 Int. Conf. on Indoor Positioning and Indoor Navigation*, 2022.

[12] F. Parralejo *et al.*, "Comparative study of different ble fingerprint reconstruction techniques," in *2021 Int. Conf. on Indoor Positioning and Indoor Navigation*, 2021.

[13] C. Kjellson *et al.*, "Accurate indoor positioning based on learned absolute and relative models," in *2021 Int. Conf. on Indoor Positioning and Indoor Navigation*, 2021.

[14] J. Bi *et al.*, "Supplementary open dataset for wifi indoor localization based on received signal strength," *Satellite Navigation*, vol. 3, no. 1, p. 25, Nov. 2022.

[15] R. Guan *et al.*, "Genloc: A new paradigm for signal fingerprinting with generative adversarial networks," in *2022 IEEE 12th Int. Conf. on Indoor Positioning and Indoor Navigation*, 2022.

[16] X. Luo and N. Meratnia, "A geometric deep learning framework for accurate indoor localization," in *2022 IEEE 12th Int. Conf. on Indoor Positioning and Indoor Navigation*, 2022.

[17] R. Klus *et al.*, "Machine Learning Based NLOS Radio Positioning in Beamforming Networks," in *IEEE Int. Workshop on Signal Processing Advances in Wireless Communication*, 2022.

[18] J. Torres-Sospedra *et al.*, "Comprehensive analysis of distance and similarity measures for wi-fi fingerprinting indoor positioning systems," *Expert Systems with Applications*, vol. 42, 2015.

[19] A. Brunello *et al.*, "A genetic programming approach to wifi fingerprint meta-distance learning," *Pervasive and Mobile Computing*, vol. 85, p. 101 681, 2022.

[20] J. Torres-Sospedra *et al.*, "Supplementary Materials for ≪Let's Talk about k-NN for Indoor Positioning: Myths and Facts in RF-based Fingerprinting≫," *Zenodo*, 2023, https://doi.org/10.5281/zenodo.7599736.

[21] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *Int. J. Math. Model. Meth. Appl. Sci.*, vol. 1, Jan. 2007.

[22] G. Minaev *et al.*, "Comprehensive survey of similarity measures for ranked based location fingerprinting algorithm," in *2017 Int. Conf. on Indoor Positioning and Indoor Navigation*, 2017.

[23] B. Shin *et al.*, "Enhanced weighted k-nearest neighbor algorithm for indoor wi-fi positioning systems," in *Int. Conf. on Computing Technology and Information Management*, 2012.

[24] H. Zou *et al.*, "Winips: Wifi-based non-intrusive indoor positioning system with online radio map construction and adaptation," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 8118–8130, 2017.

[25] S. Liu *et al.*, "Performance analysis of adaptive k for weighted k-nearest neighbor based indoor positioning," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, 2022.

[26] F. S. Daniş *et al.*, "An indoor localization dataset and data collection framework with high precision position annotation," *Pervasive and Mobile Computing*, vol. 81, p. 101 554, 2022.

[27] A. Moreira *et al.*, "The DSI dataset for Wi-Fi fingerprinting using mobile devices," version 1.0, *Zenodo*, Apr. 2020.

[28] G. M. Mendoza-Silva *et al.*, "Long-term wifi fingerprinting dataset for research on robust indoor positioning," *Data*, vol. 3, no. 1, 2018.

[29] T. King *et al.*, *CRAWDAD dataset mannheim/compass*, [online] https://crawdad.org/mannheim/compass/20080411, 2008.

[30] A. Moreira *et al.*, "Wi-Fi Fingerprinting dataset with multiple simultaneous interfaces," version 1.0, *Zenodo*, Sep. 2019.

[31] E. S. Lohan *et al.*, "WiFi RSS measurements in Tampere University multi-building campus, 2017," *Zenodo*, 2021.

[32] S. Shrestha *et al.*, "Deconvolution-based indoor localization with WLAN signals and unknown access point locations," [online] http://www.cs.tut.fi/tlt/pos/MEASUREMENTS_WLAN_FOR_WEB.zip, 2013.

[33] A. Razavi *et al.*, "K-means fingerprint clustering for low-complexity floor estimation in indoor mobile localization," in *IEEE Globecom Workshops (GC Wkshps)*, 2015.

[34] A. Cramariuc *et al.*, "Clustering benefits in mobile-centric WiFi positioning in multi-floor buildings," in *2016 Int. Conf. on Localization and GNSS*, 2016.

[35] E.-S. Lohan *et al.*, "Wi-fi crowdsourced fingerprinting dataset for indoor positioning," *MDPI Data*, vol. 2, no. 4, Oct. 2017.

[36] P. Richter *et al.*, "WLAN (WiFi) rss database for fingerprinting positioning," *Zenodo*,

[37] E.-S. Lohan, "Additional TAU datasets for Wi-Fi fingerprinting- based positioning," *Zenodo*, 2020.

[38] J. Torres-Sospedra *et al.*, "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," in *2014 Int. Conf. on Indoor Positioning and Indoor Navigation*, 2014, pp. 261–270.

[39] X. Song *et al.*, "A novel convolutional neural network based indoor localization framework with wifi fingerprinting," *IEEE Access*, vol. 7, pp. 110 698–110 709, 2019.

[40] F. Alhomayani and M. H. Mahoor, "Outfin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach," *Scientific Data*, vol. 8, no. 1, 2021.

[41] T. Bravenec *et al.*, "Supplementary Materials for "Influence of Measured Radio Environment Map Interpolation on Indoor Positioning Algorithms"," *Zenodo*, 2022.

[42] C. Laoudias *et al.*, *Kios wifi rss dataset*, https://www.researchgate.net/profile/Christos-Laoudias/publication/256482916_KIOS_WiFi_RSS_dataset, Sep. 2013.

[43] J. L. Salazar González *et al.*, "Energy-efficient indoor localization wifi-fingerprint system: An experimental study," *IEEE Access*, vol. 7, pp. 162 664–162 682, 2019.

[44] F. J. Aranda *et al.*, "Multi-slot ble raw database for accurate positioning in mixed indoor/outdoor environments," *Data*, vol. 5, no. 3, 2020.

[45] G. M. Mendoza-Silva *et al.*, "Ble rss measurements dataset for research on accurate indoor positioning," *Data*, vol. 4, no. 1, 2019.

[46] A. N. Nor Hisham *et al.*, "Hybrid wi-fi and ble fingerprinting dataset for multi-floor indoor environments with different layouts," *Data*, vol. 7, no. 11, 2022.

[47] G. G. Anagnostopoulos and A. Kalousis, "Towards reproducible indoor positioning research," in *IPIN 2021 conference*, I. 2. conference, Ed., Nov. 29, 2021, published.

[48] J. Hu *et al.*, "Experimental analysis on weight K -nearest neighbor indoor fingerprint positioning," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 891–897, 2019.

[49] S. Xu *et al.*, "Adaptive residual weighted k-nearest neighbor fingerprint positioning algorithm based on visible light communication," *Sensors*, vol. 20, no. 16, 2020.