Optimum NN Algorithms Parameters on the UJIIndoorLoc for Wi-Fi Fingerprinting Indoor Positioning Systems

Emad Ebaid
School of Computing & Communications
Lancaster University, Lancaster, UK
e.ebaid@lancaster.co.uk

Keivan Navaie
School of Computing & Communications
Lancaster University, Lancaster, UK
k.navaie@lancaster.ac.uk

***Abstract*—Wi-Fi fingerprinting techniques are commonly used in Indoor Positioning Systems (IPS) as Wi-Fi signal is available in most indoor settings. In such systems, the position is estimated based on a matching algorithm between the enquiry points and the recorded fingerprint data. In this paper, our objective is to investigate and provide quantitative insight into the performance of various Nearest Neighbour (NN) algorithms. The NN algorithms such as KNN are also often employed in IPS. We extensively study the performance of several NN algorithms on a publicly available dataset, UJIIndoorLoc. Furthermore, we propose an improved version of the Weighted KNN algorithm. The proposed model outperforms the existing works on the UJIIndoorLoc dataset and achieves better results for the success rate and the mean positioning error.**

Keywords— Indoor positioning, Wi-Fi fingerprinting, KNN algorithm, WKNN algorithm, data-driven KNN.

I. Introduction

Indoor Positioning System (IPS) determines an object's position inside a building [1]. Positioning technologies use different means such as radio signals [2], Optical [3], and Magnetic [4] technologies. The radio-based technology is favoured in IPS because it has a low cost and can easily cover a large area. In radio-based positioning systems, Wi-Fi technology is commonly used as it is widespread and does not require additional infrastructure to implement for indoor positioning, as many buildings have already been equipped with Wireless Access Points (WAPs) [5]. The most common method of Wi-Fi technology in IPS is utilizing Received Signal Strength Indicators (RSSI) measures through fingerprinting technique and creating a radio map of a place, i.e. collections of Reference Points (RPs). The user's position is then calculated by comparing and matching to pre-existing fingerprinting RSSI measures [6].

The Wi-Fi fingerprinting technique consists of two phases, the offline phase considering the collection RSSI and building the radio map (Fingerprinting) and the online phase calculating the position estimation. Fig.1 depicts an overall Wi-Fi IPS architecture. There are varieties of algorithms from simple ones such as Nearest Neighbour (NN) algorithms [7] to more complex such as Deep Neural Network (DNN) algorithms [8]. The NN algorithm is the most suitable classifier for indoor positioning as presented in [9] and [10], where different machine learning algorithms are compared for their suitability for indoor positioning, especially in pattern recognition and large data. In the fingerprinting approach, there are two methods: probabilistic methods and deterministic methods. The latter methods do not require prior knowledge of the Wi-Fi signal probability model, thus, easy to implement and widely used. In the deterministic methods, the Nearest Neighbour (NN) algorithms are commonly used; including Nearest Neighbour (NN), *K*-Nearest Neighbour (*K*NN), and *Weighted K*-Nearest Neighbour (*WK*NN) [11].

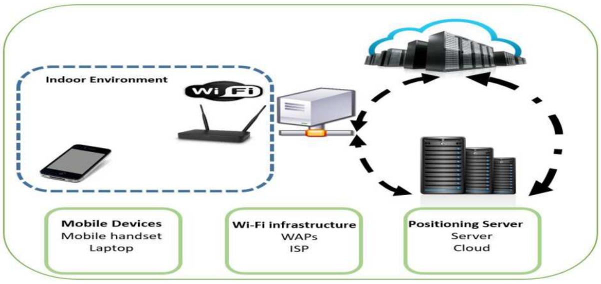

Fig. 1. A schematic of a generic Wi-Fi IPS.

In this paper, the NN algorithms, mainly *K*NN and *WK*NN are considered because of their simplicity and effectiveness in classification and regression problems. Although the computational cost and memory limitation are big drawbacks, the NN algorithms are highly efficient in pattern recognition [12, 13]. In multi-building multi-floor scenarios, there are two problems associated with positioning estimation. The correct location of the building and floor should be detected first followed by obtaining the in-floor position with the lowest estimation error. Therefore, our first objective is to obtain the correct location and in-floor position using only basic NN algorithms on the publically available UJIIndoorLoc database [14]. To mitigate the fingerprinting data linearity we introduced data representations as proposed in [15]. The accuracy issue was addressed by tuning the *k*-value and applying distance weight in combination with distance measures.

The second objective of this study is to identify the best parameters for both KNN and WKNN algorithms in the UJIINdoorLoc dataset as an alternative to simple Euclidean distance and k=1, as often presented in many studies. This provides insights into developing straightforward algorithms that are suitable for real-time operation applications such as IPS. In this work, we are motivated to provide the best NN algorithms parameters on the UJIIndoorLoc database as a benchmark for further development. Therefore, the significance of this paper is highlighting the parameter tuning of NN algorithms on the UJIIndoorLoc database, because it is challenging to deal with sparse datasets in indoor positioning and get acceptable positioning estimation.

In the following, Section 2 provides related work. Section 3 provides the methodology. Results of experiments and performance evaluation are presented in Section 4 followed by a discussion and brief conclusion in Section 5.

II. Related Work

The previous work can be divided into the two following categories:

A. Improving KNN and WKNN Algorithms

Many studies presented in the literature introduced either improving or optimizing the *K*NN algorithm in different ways. However, the *k*-value is significantly important to obtain the positioning estimation accuracy, as a fixed *k*-value is always not appropriate for all types of data. Hence, in [16] an adoptive *k*-value to the *K*NN was proposed by analyzing the correlation between the RSSI and *k*-value. This helped to boost the position accuracy above 30% compared with the fixed *k*-value. Furthermore, in [17] an algorithmic improvement to *K*NN by replacing the WAPs features with a Wi-Fi signal propagation model. Their exploratory results appear that the *K*NN optimization algorithm incorporates a certain degree of enforceability. Nevertheless, the results improved significantly in terms of positioning accuracy. Also [18] investigated applying quartile analysis to pre-process the RSSI data and tackle the signal interferences and variance of RSSI measurements to improve the positioning accuracy.

Taking a different approach, [19] adopted the Fuzzy *K*NN classifier by assigning membership in the probable label as a function of the *Euclidean* distance vector from the basic *K*NN algorithm. The experimental results show improving position accuracy. Nevertheless, this method requires exceptionally large memory. To address this issue, [20] proposed a compression method for the data before applying the *K*NN algorithm. This helped to reduce the memory required and maintain the accuracy of the algorithm. In [21] took an approach data-driven to compute the *k*-value based on sparse learning to overcome fixed *k*-value by learning the optimal *k*-value for each test sample.

For the *WK*NN algorithm, similar to the *K*NN, the *k*-value and the distance function are important parameters affecting the positioning performance. Therefore, researchers consider finding an optimal setting for the *k*-value by working on the cluster-filtered methods or adopting the environment changes to adjust the *k*-value adaptively [11]. In addition, adding weight to the calculated space distance, especially where k>1 in sparse data improves data classification performance. This dramatically improves the algorithm performance when it is combined with a dynamic *k* as in [22], in which they proposed a self-adaptive *WK*NN (SAWKNN) algorithm with a dynamic *k*. This SAWKNN is adjusting the value of *k* based on the RSS to obtain a better positioning accuracy than traditional *WK*NN.

In [23] an improved *WK*NN was introduced by selecting RP based on both distances of RSS the physical and space distance. A fusion of weighted distances algorithm was then applied to calculate the position estimation. This approach was also efficient to enhance the accuracy of experimental results. Similarly [11] considered fingerprinting clustering and signal-weighted *Euclidean* distance considering the position distribution of reference points (RPs). In [24] an entropy *WK*NN method was proposed to adapt to environmental changes based on location characteristics and indoor distribution. Obtaining the entropy weight requires often complex processing of large data by adaptively adjusting the weight index. An improved *WK*NN was also proposed in [25] considering positioning speed as an objective where a selection of WAP algorithms was combined with an asymmetric Gaussian filter algorithm.

B. ML Algorithms on UJIIndoorLoc Database

The related works that use UJIIndoorLoc have been summarized with their performance results in Table I. It is worth mentioning that not all works have the same database configuration, for example, some are tested on each building separately and others on a complete database. The first two methods in Table I have *K*NN as the main algorithm by Torres-Sospedra *et al.* [13] where they created the UJIIndoorLoc database. They also provided an explanation of the dataset and a baseline result based on *K*NN for k=1 and using *Euclidean* distance. They then respectively obtained 89.92% and 7.9 (m) for success rate and error. Later in [15], they introduced a comprehensive study on the distance metrics and shows that using k = 13 and *Sorensen* distance combined with power RSSI data representation can obtain 95.2 and 6.19 (m) for success and error, respectively. However, the results in positioning errors were obtained based on correctly identifying the building and floor, which is not always the case in practice.

The rest of the works [26-36] used different algorithms such as Decision Tree (DT), Deep Neural Network (DNN), Random Decision Forest (RDF), Artificial Neural Network (ANN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) with different methods and configurations.

III. Methodology

Our method to improve the *K*NN and *WK*NN algorithms in the position estimation context by investigating the impact of the *k*-value on positioning estimation using an UJIIndoorLoc dataset on the following procedure:

A. Tuning k-value for KNN & WKNN

In *K*NN and *WK*NN, the *k* represents the number of samples from the fingerprint dataset, setting a low value of *k*, such as 1, maybe not be adequate since only a single sample is considered to estimate the final position, whereas a high *k*-

TABLE I. Comparison of results on different configuration

Reference	Success (%)			Error (m)
	BLD	FLO	Mean	
Torres-Sospedra *et al.* (2014) [14]	-	-	89.92	7.90
Torres-Sospedra *et al.* (2015) [15]	-	-	95.2	6.19
RTLS@UM: Moreira *et al.*[a] (2015) [26]	100	93.74	-	6.20
Nowicki and Wietrzykowski (2017) [27]	-	-	92	-
Ibrahim *et al.* [b] [c] (2018) [28]	100	100		2.77
Hybloc: Akram *et al.* [d] (2018) [29]	-	-	85	6.29
Gan *et al.* (2019) [30]	100	95.41	-	6.40
CNNLoc: Song *et al.* (2019) [31]	100	96.03	-	11.78
Liu *et al.* (2021) [32]	99.64	91.18	-	8.39
CCpos: Qin *et al.* (2021) [33]	99.6	95.3	-	12.4
Cao *et al.* (2021) [34]	-	99.54	-	3.46
Abdalla *et al.* (2021) [35]	100	95.24	-	11.78
Tang *et al.*[e] (2022) [36]	100	94.20	-	8.42

a) The testing set was provided exclusively as part of the EVAAL competition.

b) Only the training dataset was used with spilt-out samples from the training dataset for testing.

c) Data was manipulated to obtain RSS time-series readings and then the new dataset was split.

d) A new attribute generated named Room ID consists of Building ID, Floor ID, and Space ID.

e) The validation dataset was split into new validation and test sets value could degrade the model. The previous studies used a given k-value for all the considered models. However, we have noticed that the k-value can be for each class or model rather than one *k*-value for the algorithm. Consequently, the best k-value performance is varying from model to model. Therefore, we test different *k*-value from 1 to 25 by generating a model on different values of *k* and checking their performance in each experiment configuration.

B. Database Size Configurations

UJIIndoorLoc database contains data for three buildings referred to as BLD0, BLD1, and BLD2. Each building has multi-floors, four in BLD0 and BLD1, while BLD2 has five floors. The UJIIndoorLoc database has a training dataset (19,938 samples) and a validation dataset (1,111 samples) as no testing dataset is available we used the validation dataset as a testing dataset. The datasets are represented as fixed-size vectors where each index corresponds to 520 WAPs available across the three buildings at Jaume I University in Spain. Those vectors contain the original RSS intensity values ranging from 0 (the highest signal) to -104 (the lowest signal) in (-dBm) and a default value of (100 dBm) denoted for those WAPs was not detected [14]. In dataset configuration, we have used a complete dataset then each building separately.

C. RSSI Data Representation

Using three data representations to the RSSI values by converting the raw RSSI measurement data to positive, exponential, and powered data. These data representations were introduced by Torres-Sospedra *et al.* 2015 [14]. This pre-processing action has shown evidence of improving the model performance to represent the RSSI measurements in a better way for efficient algorithm calculation. In addition, this pre-processing is a method that can reduce the training time and increase the maximum possible accuracy, here we have:

- Positive representation

$$Pos_i\,(x) = \begin{cases} (RSS_i - min) & if\ WAP_i \text{ is detected} \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

- Exponential representation

$$Exp_i(x) = \frac{exp(\frac{RSS_i - min}{\alpha})}{exp(\frac{-min}{\alpha})} \quad , \quad (2)$$

- Powed representation

$$Pow_i(x) = \frac{(RSS_i - min)^{\beta}}{(-min)^{\beta}}, \quad (3)$$

where RSS_i is a received signal strength measurement, min represents the minimum value of RSS_i in the datasets. Lastly, α and β are mathematical constants that have values of 24 and 2, respectively.

D. Distance Function

As suggested by [15], we further investigated the impact of distance metrics (distance function) on the NN algorithms using common distance matrices including *Cityblock*, *Euclidean*, *Minkowski*, *Cosine,* and *Correlation* with different *k*-values. We tested each distance function on a different dataset configuration and recorded their performance. These distance measures are defined in [15] and [37].

Cityblock, *Euclidean,* and *Minkowski* belong to the Minkowski family group, the general equation is defined as:

$$distance_p(P,Q) = \sqrt[p]{\sum_{i=1}^{d} |P_i - Q_i|^p} \ , \ \forall p \in N^+ \quad (4)$$

where *P* refers to Position Points and *Q* denotes the Quarry Points. The distance between these two vectors is being calculated and d refers to the length of the vector. In the above, p determines the distance measures, where p = 1 and 2 results in *Cityblock,* and *Euclidean* distance, respectively. We also tested p =3, 4, and 5 in the *Minkowski* group.

Cosine distance belongs to the Inner Product family. The cosine similarity computes the angle between two vectors from the same or different distributions. However, the two vectors should have the same number of features. Given two sample feature vectors P, Q $\in \mathbb{R}^n, p = \{p_1, p_2, \dots . p_n\}, q = \{q_1, q_2, \dots, q_n\}$. The cosine similarity subtracted from one is:

$$cos(P,Q) = 1 - \frac{\sum_{i=1}^{n} P_i Q_i}{\sqrt{\sum_{i=1}^{n} P^2}\sqrt{\sum_{i=1}^{n} Q^2}} \quad (5)$$

Correlation distance computes the correlation between two jointly distributed random variables. In which, n samples were sampled from a bivariate (P, Q) joint distribution. *Correlation* distance is a version of the *Pearson* distance, where the *Pearson* distance is scaled in the range between zero and one. Given n samples consisting of two features, $\{(p_1, q_1), (p_2, q_2), \dots, (p_n, q_n)\}$, the *correlation* distance is defined as:

$$Cor(P,Q) = \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{n}(P_i Q_i) - (n\bar{P}\bar{Q})}{\sqrt{\sum_{i=1}^{n} P_i^2 - n\bar{P}^2}\sqrt{\sum_{i=1}^{n} Q_i^2 - n\bar{Q}^2}}\right) \quad (6)$$

E. Distance Weight for KNN

In *K*NN, the distance weight (*w*) is equal, while in *WK*NN there are two distance weights namely *inverse* distance and *squared inverse* distance commonly used with the *K*NN algorithm to form *Weighted K*NN. We will test them with the best results obtained from previous steps. The *inverse* distance weight is given by:

$$w_{inverse}(P,Q) \ = \frac{1}{d} \ . \quad (7)$$

The *squared inverse* distance weight is given by:

$$w_{squared\ inverse}(P,Q) \ = \frac{1}{d^2}, \quad (8)$$

where *d* is the distance between two points in signal space.

IV. Performance Evaluation

There are two matrices, Success, and Error to evaluate the algorithm performance. Success is defined as the percentage of accuracy for a building or floor that is correctly predicted known as Hit Rate.

$$Hit\ Rate_{BLD\ or\ FLO} = \frac{(correct\ Predictions)}{All\ Predictions} * 100\% \quad (9)$$

We added building and floor accuracy to get the mean success rate as a metric accountable for both.

$$Mean\ Sucess_{BLD+FLO} = \frac{(Accuracy_{BLD} + Accuracy_{FLO})}{2} \quad (10)$$

The Error refers to the Position Error (PE), which is the *Euclidean* distance between the estimated position and the actual position given by:

$$PE = \sqrt{(x - \hat{x})^2 + (y - \hat{y})^2} \qquad (11)$$

where the $(x - \hat{x})$ represent Longitude error and $(y - \hat{y})$ represent Latitude error. The hat marks (^) refer to the estimated position and non-hat marks refer to the actual position. However, we are applying a vector of points, and as a result, we get the PE in a vector, therefore, we use the mean position error:

$$Mean\ Position\ Err = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(x - \hat{x})^2 + (y - \hat{y})^2} \qquad (12)$$

Where n is the number of errors.

A. Experiment Setup

Modelling and testing have been implemented using MATLAB R2021 (a), running and tested on a Lenovo Laptop, equipped with a processor Intel(R) Core(TM) i5-8265U CPU@ 1.60GHz, 1.80 GHz and 8 GB of RAM, and operated with Windows 10, 64 bits. The modelling procedure has three programming steps as follows:

a) *Import Training and Validation datasets*
b) *Pre-process Datasets*
c) *Apply KNN and WKNN algorithms on each model for each configuration.*
d) *Record and plot the performance results.*

UJIIndoorLoc dataset has four main labels (LONGITUDE, LATITUDE, BUILDING ID, and FLOOR) which are required to predict the location and estimated position. The Longitude and Latitude are regression problems, while the Building ID and Floor are classification problems. We followed the procedure in Section 3 to test the performance of *K*NN and *WK*NN algorithms.

B. Experiment Results

1) Obtaining the best *k*-value

i) The best *k*-value on the complete dataset

The performance results are shown in Table II and Fig 2. It shows that the highest success rate is 98.15% when using *correlation* distance with *exponential* representation and the lowest mean positioning error is 7.64 (m) using the *correlation* distance with *exponential* representation on a complete dataset. This is a significantly better set of results compared with those in Table I.

ii) The best k-value on each separated building

From obtained results in Tables III, IV, and V, we can see there is a variant of positioning errors from building to building. This leads us to look at the signal distribution on each building in Fig 3. We can interpret that the BLD1 has the lower RPs compared to the other two buildings, while the BLD2 has a higher number of RPs, yet we can see the error range in 9 m, unlike the BLD0 which has a lower error in the BLD2 has a higher number of RPs, yet we can see the error range in 9 m, unlike BLD0 which has a lower error in the range of 5 m. Thus, we looked at each building and found out that only two mobile devices surveyed the BLD0, while the other two buildings have many different devices. The heterogeneity of the devices might cause a large positioning error. A summary of the best results is presented in Table VI.

TABLE II. BEST PERFORMANCE OF *K* VALUE ON DIFFERENT DISTANCE FUNCTIONS AND DATA REPRESENTATIONS USING A COMPLETE DATASET

Distance Metrics	Data Rep.	Location Success (%)				Error (m)	
		k	BLD	FLO	Mean	*k*	Error
Cityblock	*Pos*	*1*	*98.88*	*89.28*	94.05	*16*	11.4158
	Exp	*1*	*99.00*	*90.81*	94.91	*1*	10.5256
	Pow	*1*	99.36	90.90	95.13	*1*	9.89261
Euclidean	*Pos*	*17,19*	99.36	91.26	95.31	*1*	9.19835
	Exp	*5*	99.55	92.79	96.17	*1*	8.58877
	Pow	*6*	99.73	93.15	96.44	*2*	8.76081
Minkowski P3	*Pos*	*8*	99.91	90.99	95.45	*5*	9.13964
	Exp	*6,8*	99.73	93.60	96.66	*2*	8.69334
	Pow	*5*	99.73	93.87	96.80	*1*	8.65620
Minkowski P4	*Pos*	*9,19*	99.82	91.08	95.45	*2*	8.92590
	Exp	*7,11*	100	94.05	97.02	*2*	8.39160
	Pow	*6*	99.82	94.05	96.93	*2*	8.42007
Minkowski P5	*Pos*	*8*	100	90.90	95.45	*2*	8.95207
	Exp	*5*	100	94.14	97.07	*2*	8.30021
	Pow	*11*	99.82	94.23	97.02	*2*	8.57079
Cosine	*Pos*	24	100	93.96	96.93	*13*	7.82302
	Exp	5,7	99.55	93.60	96.57	*1*	8.52173
	Pow	*21*	100	96.30	98.15	*23*	7.72135
Correlation	*Pos*	*22,24*	100	93.87	96.89	*24*	7.85285
	Exp	*22,23*	100	96.30	**98.15**	*22*	**7.64517**
	Pow	*20,22*	100	96.21	98.10	*23*	7.69166

TABLE III. PERFORMANCE RESULT ON BLD0

Distance Metrics	Data Rep.	FLO Success (%)		Positioning Error (m)	
		k	Hit Rate	*k*	Mean Error
Cityblock	*Pos*	9-11	97.20	10	5.47846
	Exp	3	97.01	4	5.29077
	Pow	4	97.38	4	5.42837
Euclidean	*Pos*	19	97.76	3	5.49774
	Exp	9	97.38	1	5.57302
	Pow	4,7	97.01	5	5.25164
Cosine	*Pos*	8, 9	97.76	8	5.74659
	Exp	1	97.57	4	**5.16713**
	Pow	1	97.57	6	5.39378
Correlation	*Pos*	7- 9	97.76	8	5.47846
	Exp	1	**97.94**	5	**5.17979**
	Pow	1	97.57	6	5.41620

TABLE IV. PERFORMANCE RESULT ON BLD1

Distance Metrics	Data Rep.	FLO Success (%)		Positioning Error (m)	
		k	Hit Rate	*k*	Mean Error
Cityblock	*Pos*	2	77.85	3	11.9846
	Exp	6	79.47	3	11.8343
	Pow	6,10	79.47	2	11.2378
Euclidean	*Pos*	18	78.50	23	11.1771
	Exp	18-24	81.75	14	10.3903
	Pow	4, 20	83.71	21	10.6452
Cosine	*Pos*	24	85.34	21	9.54734
	Exp	24	85.66	15	9.98617
	Pow	21	**93.81**	3	9.57655
Correlation	*Pos*	24	85.34	21	9.52438
	Exp	22, 23	93.48	23	**9.16244**
	Pow	21	**93.81**	24	9.74990

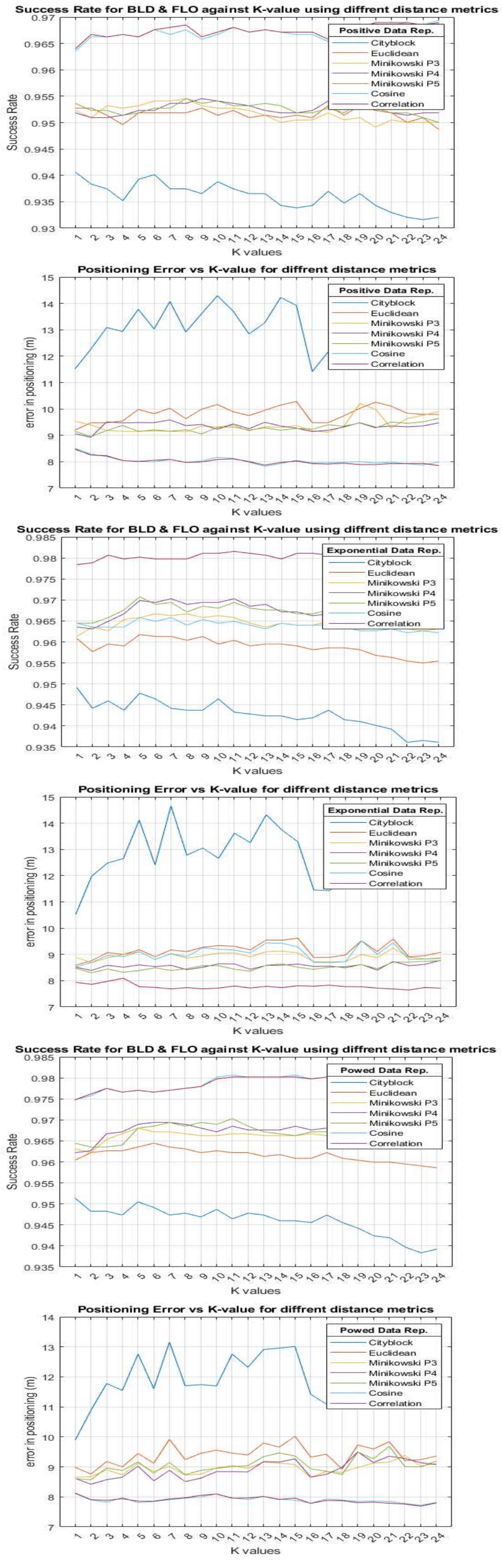

Fig. 2. Results of k-value on Success and Error for Different Configurations

TABLE V. PERFORMANCE RESULT ON BLD2

Distance Metrics	Data Rep.	FLO Success (%)		Positioning Error (m)	
		k	Hit Rate	*k*	Mean Error
Cityblock	*Pos*	1,2	90.29	9	12.9376
	Exp	1	94.02	1	11.3889
	Pow	1	94.77	1	11.0013
Euclidean	*Pos*	2	95.52	1	11.1046
	Exp	4	**97.38**	6	10.0375
	Pow	3-5	97.01	5	10.3945
Cosine	*Pos*	6,8	97.01	22	9.45524
	Exp	8-10	97.01	8	9.79197
	Pow	22, 24	**97.38**	9	9.72947
Correlation	*Pos*	7, 8	97.01	24	9.28354
	Exp	22-24	97.01	22	**9.24040**
	Pow	20, 22	97.01	5	9.86497

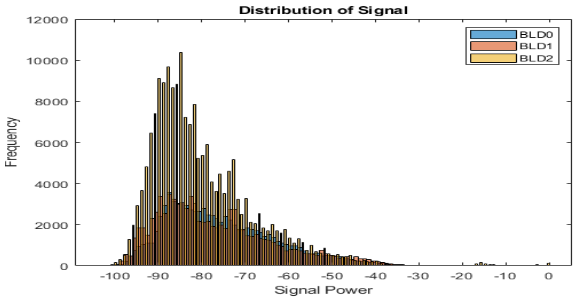

Fig. 3. Distribution of Signals on Each Building.

TABLE VI. BEST RESULTS ON EACH BUILDING

BLD No.	Data Rep.	Distance Metrics	FLO Success (%)		Positioning Error(m)	
			k	Hit rate	*k*	Error
BLD0	*Exp*	*Correlation*	1	97.94	5	**5.17979**
BLD1	*Exp*	*Correlation*	23	93.48	23	**9.16244**
BLD2	*Exp*	*Correlation*	22	97.01	22	**9.24040**
Average						**7.86087**

2) Obtaining the best distance weight (*w*)

The next step is to move from *K*NN to *WK*NN, by introducing distance weight (*w*) but this time using only the *Correlation* distance and *exponential* data representation as this combination showed the best performance in the previous experiments. This requires testing each distance weight (*w*), inverse, and squared inverse on a complete dataset and each building configuration. We must recalculate the *k*-value, since it was noticed that there is improvement following applying distance weight, hence, we further increased the *k*-value from 24 to 50 and recorded the best results when *k* >1 only in Table VII. The best result is 7.39 m, which is better than the *K*NN.

C. Comparison of Findings with Other Studies.

To make the comparison more reasonable, we evaluate our best-tuned algorithm (*WK*NN) with the same dataset configurations, here we have selected for comparison the studies that have similar settings of using both the training dataset and validation dataset regardless of the used algorithm or methods. In other words, comparing with the complete dataset and applying validation dataset for testing where possible. Table VIII shows our results compared to the other studies. It clearly shows that our proposed approach has yielded a significant improvement in the mean success rate and positioning error, which demonstrates an innovative design and contributes to *WK*NN's better performance.

TABLE VII. INTRODUCE THE DISTANCE WEIGHTS ON THE WHOLE DATASET FOR CORRELATION DISTANCE

Building No.	Data Rep.	weight	Location Success (%)				Positioning Error(m)	
			k	BLD *HR*	FLO *HR*	Mean	*k*	Error
012	*Exp*	***1/d***	***20,26***	**100**	**96.30**	**98.15**	***26***	**7.39643**
		$1/d^2$	*20,43*	100	96.21	98.10	*26*	7.44725
0	*Exp*	*1/d*	*2*	-	97.94	-	*2*	5.57302
		$1/d^2$	***2***	**-**	**97.94**	**-**	***24***	**5.52751**
1	*Exp*	***1/d***	***42,43***	**-**	**94.13**	**-**	***26***	**9.02968**
		$1/d^2$	*43,46*	-	93.48	-	*26*	9.25612
2	*Exp*	*1/d*	*2-5*	-	97.01	-	*29*	9.03861
		$1/d^2$	***4, 5***	**-**	**97.01**	**-**	***35***	**8.95107**

TABLE VIII. COMPARISON OF THE RESULTS OF DIFFERENT STUDIES

Reference	**Location Success (%)**			**Positioning Error (m)**
	BLD	**FLO**	**Mean**	
Torres-Sospedra *et al.* (2014) [14]	-	-	89.92	7.90
Torres-Sospedra *et al.* (2015) [15]	-	-	95.2	6.19
Gan *et al.* 2019 [30]	100	95.41	-	6.40
CNNLoc: Song *et al.* 2019 [31]	100	96.03	-	11.78
Liu *et al.* 2021 [32]	99.64	91.18	-	8.39
CCpos: Qin *et al.* 2021[33]	-	-	-	12.4
Abdalla *et al.* 2021 [35]	100	95.24	-	11.78
Tang *et al.* 2022 [36]	100	94.20	-	8.42
Our tuned WKNN	**100**	**96.30**	**98.15**	**7.39**

V. DISCUSSIONS AND CONCLUSION

In these experiments, we considered one *k*-value for both Longitude and Latitude, when calculating the mean positioning error. This makes a general *k*-value that works with widely used algorithms. In case there are multi-*k* values, we have chosen the higher two values. For example, in Table I, *Correlation* has different *k*-values with the same performance including 8, 20-22, and 24. We recorded the higher two *k*-values 22 & 24 as it is generally more suitable in big sparse data. We have noticed different best *k*-value performances on each model, except in the case of *correlation* and *cosine* where they intend to stay close to linearity for both classification and regression compared to other distances. This gives them the advantage to perform well for all *k*-value.

This study concludes that the *correlation* distance function is among the best algorithms in the UJIIndoorLoc dataset, especially when it combines with *exponential* data representation. In addition, introducing the distance weight has provided the lowest positioning error to the whole dataset (BLD012) at 7.39 (m) compared to *K*NN. However, the success rate does not improve and remains the same in both *K*NN and *WK*NN. In terms of distance weight, the *inverse* is the best for the whole dataset and it varies on the individual building between *inverse* and *squared inverse*. From Table VII, there is a slight improvement when applying distance weight in both *inverse* and *squared inverse* compared to previous results without distance weight.

From the results obtained on the complete dataset, the best tuning for *K*NN is *Correlation* distance in conjunction with *exponential* data representation and *k*=22. In *WK*NN, the best tuning is *Correlation* distance in conjunction with *exponential* data representation, *inverse* weight, and *k*=26. The contribution of this study is providing a real benchmark for the best basic *K*NN and *WK*NN at their highest performance on the UJIIndoorLoc database. This will help the research community to compare and design their system when applying machine-learning algorithms for Wi-Fi fingerprinting indoor positioning.

REFERENCES

[1] L. Qi, Q. Jiahui, and C. Yi, “Research and development of indoor positioning,” China Communications, vol. 13, no. Supplement2, pp. 67-79, 2016.

[2] P. Bahl, and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," Proceedings IEEE INFOCOM 2000 2000, pp. 775-784 vol.2.

[3] R. Mautz, and S. Tilch, "Survey of optical indoor positioning systems," International Conference on Indoor Positioning and Indoor Navigation, 2011, pp. 1-7.

[4] H. Huang, D. H. Lee, K. Chang, W. Li, and T. D. Acharya, “Development of mobile platform for indoor positioning reference map using geomagnetic field data,” Computers and Electrical Engineering, vol. 68, pp. 557-569, 2018.

[5] S. Wilson, O. Michael Adeyeye, and M. Nhlanhla Bw, “A State-of-the-Art Survey of Indoor Positioning and Navigation Systems and Technologies,” South African Computer Journal, vol. 29, no. 3, 2017.

[6] S. Xia, Y. Liu, G. Yuan, M. Zhu, and Z. Wang, “Indoor Fingerprint Positioning Based on Wi-Fi: An Overview,” ISPRS international journal of geo-information, vol. 6, no. 5, pp. 135, 2017.

[7] N. Bhatia, and Vandana, “Survey of Nearest Neighbor Techniques,” (IJCSIS) International Journal of Computer Science and Information Security, vol. 8, no. 2, 2010.

[8] F. Alhomayani, and M. H. Mahoor, “Deep learning methods for fingerprint-based indoor positioning: a review,” Journal of Location Based Services, vol. 14, no. 3, pp. 129-200, 2020-07-02, 2020.

[9] S. Bozkurt, G. Elibol, S. Gunal, and U. Yayan, "A comparative study on machine learning algorithms for indoor positioning." pp. 1-8.

[10] I. T. Haque, and C. Assi, “Profiling-Based Indoor Localization Schemes,” IEEE Systems Journal, vol. 9, no. 1, pp. 76-85, 2015-03-01, 2015.

[11] B. Wang, X. Liu, B. Yu, R. Jia, and X. Gan, “An Improved WiFi Positioning Method Based on Fingerprint Clustering and Signal Weighted Euclidean Distance,” Sensors, vol. 19, no. 10, pp. 2300, 2019-05-18, 2019.

[12] K. Taunk, S. De, S. Verma, and A. Swetapadma, "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification." 2019 International

Conference on Intelligent Computing and Control Systems (ICCS), 2019, pp. 1255-1260, doi: 10.1109/ICCS45141.2019.9065747.

[13] N. Bhatia, and Vandana, “Survey of Nearest Neighbor Techniques,” (IJCSIS) International Journal of Computer Science and Information Security, Vol. 8, No. 2, 2010.

[14] J. Torres-Sospedra, R. Montoliu, A. Martinez-Uso, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta, "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems." 2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2014, pp. 261-270, doi: 10.1109/IPIN.2014.7275492.

[15] J. Torres-Sospedra, R. Montoliu, S. Trilles, Ó. Belmonte, and J. Huerta, “Comprehensive analysis of distance and similarity measures for Wi-Fi fingerprinting indoor positioning systems,” Expert Systems With Applications, vol. 42, no. 23, pp. 9263-9278, 2015.

[16] J. Oh, and J. Kim, “Adaptive K-nearest neighbour algorithm for WiFi fingerprint positioning,” ICT Express, vol. 4, no. 2, pp. 91-94, 2018-06-01, 2018.

[17] X. Ge, and Z. Qu, "Optimization WIFI indoor positioning KNN algorithm location-based fingerprint." *2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2016, pp. 135-137, doi: 10.1109/ICSESS.2016.7883033.

[18] D. Ferreira, R. Souza, and C. Carvalho, “QA-kNN: Indoor Localization Based on Quartile Analysis and the kNN Classifier for Wireless Networks,” Sensors, vol. 20, no. 17, pp. 4714, 2020-08-21, 2020.

[19] P. Torteeka, and X. Chundi, "Indoor positioning based on Wi-Fi Fingerprint Technique using Fuzzy K-Nearest Neighbor." *Proceedings of 2014 11th International Bhurban Conference on Applied Sciences & Technology (IBCAST)* Islamabad, Pakistan, 14th - 18th January 2014, 2014, pp. 461-465, doi: 10.1109/IBCAST.2014.6778188.

[20] J. Salvador-Meneses, Z. Ruiz-Chavez, and J. Garcia-Rodriguez, “Compressed kNN: K-nearest neighbors with data compression,” Entropy (Basel, Switzerland), vol. 21, no. 3, pp. 234, 2019.

[21] S. Zhang, D. Cheng, Z. Deng, M. Zong, and X. Deng, “A novel k NN algorithm with data-driven k parameter computation,” Pattern Recognition Letters, vol. 109, pp. 44-54, 2018-07-01, 2018.

[22] J. Hu, D. Liu, Z. Yan, and H. Liu, “Experimental Analysis on Weight K- Nearest Neighbor Indoor Fingerprint Positioning,” IEEE Internet of Things Journal, vol. 6, no. 1, pp. 891-897, 2019-02-01, 2019.

[23] X. Peng, R. Chen, K. Yu, F. Ye, and W. Xue, “An Improved Weighted K-Nearest Neighbor Algorithm for Indoor Localization,” Electronics, vol. 9, no. 12, pp. 2117, 2020-12-11, 2020.

[24] H. Pen, and W. Xiang, "An Improved Weighted K-Nearest Neighbor Positioning Method in Buildings for Power Distribution Room." *2021 6th International Symposium on Computer and Information Processing Technology (ISCIPT)*, 2021, pp. 721-726, doi: 10.1109/ISCIPT53667.2021.00152.

[25] Z. Zhao, Z. Lou, R. Wang, Q. Li, and X. Xu, “I-WKNN: Fast-Speed and High-Accuracy WIFI Positioning for Intelligent Stadiums,” 2021, *Computers & Electrical Engineering*, Volume 98, 2022, https://doi.org/10.1016/j.compeleceng.2021.107619.

[26] A. Moreira, M. J. Nicolau, F. Meneses, and A. Costa, "Wi-Fi fingerprinting in the real world - RTLS@UM at the EvAAL competition."

[27] M. Nowicki, and J. Wietrzykowski, "Low-Effort Place Recognition with WiFi Fingerprints Using Deep Learning," Automation 2017, pp. 575-584: Springer International Publishing, 2017.

[28] M. Ibrahim, M. Torki, and M. Elnainay, "CNN based Indoor Localization using RSS Time-Series." *2018 IEEE Symposium on Computers and Communications (ISCC)*, 2018, pp. 01044-01049, doi: 10.1109/ISCC.2018.8538530.

[29] B. A. Akram, A. H. Akbar, and O. Shafiq, “HybLoc: Hybrid Indoor Wi-Fi Localization Using Soft Clustering-Based Random Decision Forest Ensembles,” IEEE Access, vol. 6, pp. 38251-38272, 2018-01-01, 2018.

[30] H. Gan, M. H. B. M. Khir, G. Witjaksono Bin Djaswadi, and N. Ramli, “A Hybrid Model Based on Constraint OSELM, Adaptive Weighted SRC and KNN for Large-Scale Indoor Localization,” IEEE Access, vol. 7, pp. 6971-6989, 2019-01-01, 2019.

[31] X. Song, X. Fan, C. Xiang, Q. Ye, L. Liu, Z. Wang, X. He, N. Yang, and G. Fang, “A Novel Convolutional Neural Network Based Indoor Localization Framework With WiFi Fingerprinting,” IEEE Access, vol. 7, pp. 110698-110709, 2019.

[32] S. Liu, R. De Lacerda, and J. Fiorina, "WKNN indoor Wi-Fi localization method using k-means clustering based radio mapping." *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1-5,doi:10.1109/VTC2021-pring51267.2021.9448961.

[33] F. Qin, T. Zuo, and X. Wang, “CCpos: WiFi Fingerprint Indoor Positioning System Based on CDAE-CNN,” Sensors, vol. 21, no. 4, pp. 1114, 2021-02-05, 2021.

[34] X. Cao, Y. Zhuang, X. Yang, X. Sun, and X. Wang, “A universal Wi-Fi fingerprint localization method based on machine learning and sample differences,” Satellite Navigation, vol. 2, no. 1, 2021-12-01, 2021.

[35] A. E. A. Elesawi, and K. S. Kim, “Hierarchical Multi-Building And Multi-Floor Indoor Localization Based On Recurrent Neural Networks,” 2021-12-23T11:56:31, 2021.

[36] Z. Tang, S. Li, K. S. Kim, and J. Smith, “Multi-Output Gaussian Process-Based Data Augmentation for Multi-Building and Multi-Floor Indoor Localization,” arXiv preprint arXiv:2202.01980, 2022.

[37] H. A. A. Alfeilat, *et al.*, “Effects of Distance Measure Choice on K-Nearest Neighbor Classifier Performance: A Review,” Big Data, vol. 7, no. 4, pp. 221, 2019.