
Proceedings Paper:
Li, X., Li, H., Chan, H.K.-H. et al. (2023) Data imputation for sparse radio maps in indoor positioning. In: 2023 IEEE 39th International Conference on Data Engineering (ICDE). 2023 IEEE 39th International Conference on Data Engineering (ICDE), 03-07 Apr 2023, Anaheim, CA, USA. Institute of Electrical and Electronics Engineers (IEEE), pp. 2235-2248. ISBN: 9798350322286. ISSN: 1063-6382. EISSN: 2375-026X.

https://doi.org/10.1109/icde55515.2023.00173

Data Imputation for Sparse Radio Maps in Indoor Positioning (Extended Version)

Xiao Li[1] Huan Li[2] Harry Kai-Ho Chan[3] Hua Lu[1] Christian S. Jensen[4]
[1]Department of People and Technology, Roskilde University, Denmark
[2]College of Computer Science and Technology, Zhejiang University, China
[3]Information School, University of Sheffield, United Kingdom
[4]Department of Computer Science, Aalborg University, Denmark
[1]{xiaol, luhua}@ruc.dk [2]lihuan.cs@zju.edu.cn [3]h.k.chan@sheffield.ac.uk [4]csj@cs.aau.dk

***Abstract*—Indoor location-based services rely on the availability of sufficiently accurate positioning in indoor spaces. A popular approach to positioning relies on so-called radio maps that contain pairs of a vector of Wi-Fi signal strength indicator values (RSSIs), called a fingerprint, and a location label, called a reference point (RP), in which the fingerprint was observed. The positioning accuracy depends on the quality of the radio maps and their fingerprints. Radio maps are often sparse, with many pairs containing vectors missing many RSSIs as well as RPs. Aiming to improve positioning accuracy, we present a complete set of techniques to impute such missing values in radio maps. We differentiate two types of missing RSSIs: missing not at random (MNAR) and missing at random (MAR). Specifically, we design a framework encompassing a missing RSSI differentiator followed by a data imputer for missing values. The differentiator identifies MARs and MNARs via clustering-based fingerprint analysis. Missing RSSIs and RPs are then imputed jointly by means of a novel encoder-decoder architecture that leverages temporal dependencies in data collection as well as correlations among fingerprints and RPs. A time-lag mechanism is used to consider the aging of data, and a sparsity-friendly attention mechanism is used to focus attention score calculation on observed data. Extensive experiments with real data from two buildings show that our proposal outperforms the alternatives with significant advantages in terms of imputation accuracy and indoor positioning accuracy.**

I. Introduction

Indoor applications involving navigation, augmented reality, and moving robots require sufficiently accurate indoor positioning. According to Research and Markets, the global indoor positioning and navigation market will exceed $54 billion by 2026 [1]. While a variety of indoor positioning technologies exist, positioning based on Wi-Fi fingerprinting [26] is popular: the ubiquity of Wi-Fi enables positioning without the deployment of additional expensive infrastructure, and the technology is non-intrusive to users. However, the accuracy of Wi-Fi fingerprinting based positioning depends heavily on the quality of the radio map data used [38], [46], [52], [55].

Wi-Fi fingerprinting entails two phases, as shown in Fig. 1. The offline phase creates a so-called radio map that contains pairs of a vector of Wi-Fi *received signal strength indicator* values (RSSIs), called a fingerprint, and a location label, called a reference point (RP), in which the fingerprint was observed. An RSSI measures the signal strength of a Wi-Fi *access point* (AP) [31], and is an integer value in the range of $[-99, 0]$ dBm. An example radio map is shown in the top-left part of Fig. 1. The online phase localizes the users by utilizing a location estimation algorithm (e.g., *K*NN [57]) that compares the user device's fingerprint with the radio map.

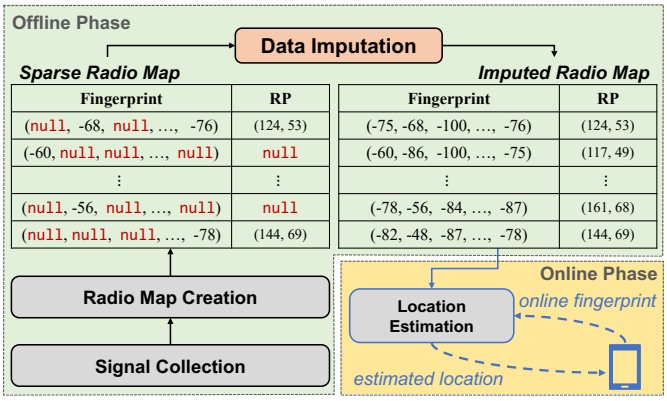


Fig. 1: Fingerprinting procedure and radio maps.

To build radio maps efficiently and economically for an indoor space, surveys are often performed by surveyors moving in the indoor space [12], [24], [31], [37], [54]. Surveyors collect RSSIs continuously while moving along predefined paths, as illustrated in Fig. 2. Due to fluctuation in the wireless environment and asynchrony between the collection and RPs (to be detailed in Section II-B), the results of walking surveys

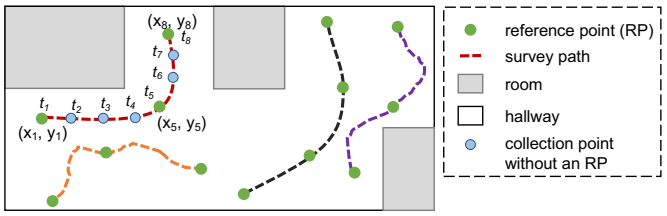


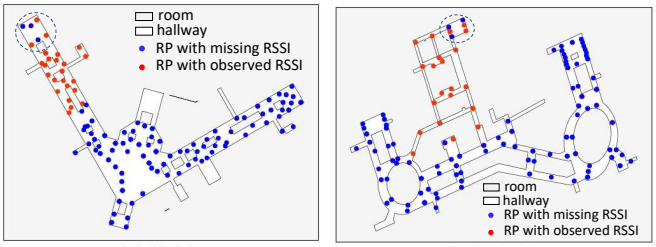
Fig. 2: Walking survey based data collection.

suffer from low data quality, having high rates of missing (i.e., percentages of `null`s) RSSIs and RPs in a radio map. For example, in the radio maps obtained from walking surveys in two real buildings called Kaide and Wanda (to be detailed in

Section V-A), the rates of missing RSSIs and RPs are between 85.6% and 93.7%. In other words, the radio maps are highly sparse, having many `nulls`. Such `nulls` must be replaced by real numbers in order for the radio map to be used by location estimation algorithms [32], [49]. Intuitively, imputing accurate real numbers for `nulls` in a radio map improves its usability in indoor positioning. However, existing studies employ straightforward strategies to fill-in RP `nulls` [18], [21], [23] and RSSI `nulls` [32], [37], [49], yielding subpar results (cf. Section V-C). Therefore, this study focuses on improving the quality and usability of sparse radio maps by accurately imputing missing RSSI and RP data. In doing so, the study contends with difficult challenges.

First, two types of missing RSSIs exist. **Missing Not At Random** (MNAR) RSSIs are caused by the unobservability of the signals of APs. This typically occurs when an AP is too far away and cannot be seen by a user's device. In contrast, **Missing At Random** (MAR) RSSIs[1] result from random events, e.g., the temporary presence of obstacles in transmission paths or occasional loss of contact with APs [18], [21].

An example of MAR RSSI and MNAR RSSI is shown in Fig. 3. An AP (access point) is selected in each venue and its deployment location is roughly within the dashed circle. For an RP (reference point), if all fingerprints collected at that RP have observed the selected AP, the RP is marked in red; otherwise, some of its fingerprints have missed the selected AP, and that RP is marked in blue. Clearly, most RPs far away from the selected AP are blue, indicating that the selected AP is unobservable at these RPs and the corresponding missing events are classified as Missing Not At Random (MNAR). On the other hand, most of the RPs near the dashed circle are red but there are several blue RPs that sometimes miss the selected AP's signals. The missing events in these RPs are incidental and should be treated as Missing At Random (MAR).



(a) Kaide (b) Wanda

Fig. 3: Observability of a selected AP's signals at different reference points (RPs).

As the two types of missing RSSIs have different causes and meanings, they should be differentiated before imputation [47]. However, neither traditional radio map completion methods [23], [37], [45] nor general data imputers [6], [10], [11], [13], [17], [25], [44], [56] differentiate the missing RSSI types. The former simply assume that all missing RSSIs are MNARs, while the latter treat them as MARs. We will offer empirical evidence of the benefits of differentiation.

To differentiate MNARs and MARs, we design a clustering based differentiator that clusters a radio map's fingerprints, and determines MARs and MNARs via intra-cluster analyses. To obtain appropriate clusters, we design two clustering algorithms that utilize a specialized accuracy metric and indoor topology, respectively.

The subsequent imputation of missing values is also challenging. Following existing radio map completion studies [23], [37], [45], we replace identified MNAR values by the value -100 dBm, the lowest RSSI value that thus reflects the unobservability of MNARs. However, imputing MARs and missing RPs is not straightforward. A MAR should be imputed with a value in $[-99, 0]$ dBm,[2] since its value would have been observed had the random event that caused it not occurred. Traditional radio map completion methods impute missing RPs using linear interpolation [37] or semi-supervised learning [49]. General data imputation methods employ matrix factorization [25] or chained equations [6]. However, all these methods fall short when data sparsity is high, and they do not consider the correlations and temporal dependencies between RSSIs and RPs collected along a path in walking-survey based radio map data collection. Next, time-series imputation methods [10], [11], [13], [17], [44], [56] target missing values in feature or source sequences (e.g., multivariate time series) with known labels. In contrast, we must contend with heterogeneous missing values—MARs in source (or fingerprint) sequences and missing RPs in target (or RP) sequences.

To impute missing MARs and RPs effectively for sparse radio maps, we design an encoder-decoder based data imputer that exploits both temporal dependencies in time series and correlations between source and target sequences to impute MARs in source sequences and missing RPs in target sequences jointly. A standard encoder-decoder is unsuitable in this setting, due to the many missing values in both source (fingerprint) and target (RP) sequences. The irregularity in the absence of RSSIs results in different durations between consecutive encoder units (see Table IV in Section IV-B), which a standard encoder-decoder cannot handle. Further, missing values in the input also degrades the ability of the model's attention mechanism at capturing the importance of each unit. To tackle these issues, we introduce a time-lag mechanism that considers the aging of the last observed value when modeling the relationships between consecutive encoder units, and we design an adapted attention mechanism to contend with the high sparsity of input features.

We make the following major contributions.

- We differentiate missing RSSIs as MARs and MNARs. To the best of our knowledge, we are the first to do so. Specifically, we provide a clustering-based approach to differentiate missing RSSIs (Section III).
- We devise a novel encoder-decoder that is capable of imputing missing RSSIs and RPs jointly by exploiting temporal

[1]The terms MNAR and MAR stem from literature [39], [47]. They are applied to missing RSSI values in this work.

[2]-99 dBm $\gg$ -100 dBm in terms of power as dBm is log-based [2].

dependencies in fingerprint and RP sequences as well as correlations among fingerprints and RPs. The data imputer considers both the aging of missing values and the sparsity of input sequences (Section IV).
- We report on extensive experiments on real data, finding that our proposals outperform the alternatives substantially in terms of data imputation accuracy and indoor positioning accuracy (Section V).

Section II presents preliminaries and problem settings, Section VI reviews related work, and Section VII concludes and discusses future work.

II. Preliminaries and Problem Settings

Table I lists notations used in the paper.

TABLE I: Notation

Symbol	Description
r_d	RSSI of the dth AP
$\mathbf{f} = (r_1, r_2, \ldots, r_D)$	a fingerprint of RSSIs from D APs
$\mathbf{l} = (x, y)$	a location or a reference point (RP)
$\{(\mathbf{f}_i, \mathbf{l}_i) : i = 1 \text{ to } N\}$	a radio map $\in \mathbb{R}^{N \times (D+2)}$
$\mathbf{M} \in \{-1, 0, 1\}^{N \times D}$	a radio map mask matrix
$\mathbf{b}_i$	a binary RSSI profile vector of $\mathbf{f}_i$
$\mathbf{x}_i = \mathbf{b}_i \oplus \mathbf{l}_i$	a concatenated radio map sample

A. Fingerprinting based Indoor Positioning

Given D Access Points (APs), a Wi-Fi **fingerprint** $\mathbf{f} = (r_1, r_2, \ldots, r_D)$ is a vector of one **received signal strength indicator value (RSSI)** per AP as measured at a **reference point (RP)**, so that r_d is the RSSI of the dth AP. The location $\mathbf{l} = (x, y)$ of an RP is usually preselected by a surveyor. A radio map consists of N pairs of the form, i.e., $(\mathbf{f}_i, \mathbf{l}_i)$, where $\mathbf{f}_i$ is the fingerprint obtained at location $\mathbf{l}_i$.

For simplicity, we consider a single floor. In a multi-floor setting, our proposal can be applied to each floor separately, as studies show that it is possible to perform floor identification with high accuracy (e.g., 99+% [53]).

As mentioned, fingerprinting based positioning has two phases. In the offline phase, surveyors collect fingerprints and use the collected data to create a radio map. We target the relatively efficient data collection approach based on walking surveys, to be detailed in Section II-B.

In the online location estimation phase, a user's current location is estimated by an algorithm that compares an online fingerprint $\mathbf{f}_o$ from the user's device with a pre-collected radio map. Typical location estimation algorithms are listed below.
- *K*NN [57] finds $\mathbf{f}_o$'s K nearest fingerprints in the radio map and uses the mean of their RPs as the estimated location.
- Unlike *K*NN, W*K*NN [19] uses a weighted mean. Weights are inversely proportional to the distances between $\mathbf{f}_o$ and the fingerprints in the radio map.
- Others [28] use a radio map (fingerprints as features and RPs as labels) to train a regression model (e.g., a Random Forest) that predicts $\mathbf{f}_o$'s location.

In all cases, the positioning accuracy relies heavily on the radio map data quality.

TABLE II: Walking Survey Record Table

Time	Type	Measurement	Time	Type	Measurement
$t_1 = 0$	RP	(x_1, y_1)	$t_5 = 9$	RP	(x_5, y_5)
$t_2 = 1$	RSSI	$\langle r_1 : -70, r_2 : -83, r_3 : -76 \rangle$	$t_6 = 12$	RSSI	$\langle r_1 : -74, r_5 : -80 \rangle$
$t_3 = 3$	RSSI	$\langle r_1 : -71, r_3 : -78 \rangle$	$t_7 = 13$	RSSI	$\langle r_2 : -77, r_5 : -82 \rangle$
$t_4 = 8$	RSSI	$\langle r_3 : -80, r_4 : -68 \rangle$	$t_8 = 16$	RP	(x_8, y_8)

TABLE III: Created Radio Map

No.	Radio Map Record	Time
1	$((-70, -83, -76, \texttt{null}, \texttt{null}), (x_1, y_1))$	t_2
2	$((-71, \texttt{null}, -78, \texttt{null}, \texttt{null}), \texttt{null})$	t_3
3	$((\texttt{null}, \texttt{null}, -80, -68, \texttt{null}), (x_5, y_5))$	t_4
4	$((-74, -77, \texttt{null}, \texttt{null}, -81), \texttt{null})$	t_6
5	$((\texttt{null}, \texttt{null}, \texttt{null}, \texttt{null}, \texttt{null}), (x_8, y_8))$	t_8

B. Walking Survey based Radio Map Creation

In a walking survey [12], [24], [31], [37], [54], a surveyor visits a sequence of preselected RPs with flexible movement in-between each two consecutive RPs, collecting RSSIs of APs along with corresponding collection times and then enters these into a *Walking Survey Record Table*.

Fig. 2 shows an example with four survey paths. The top-left one yields the record table in Table II. There are two types of records, namely RP and RSSI, sorted on timestamps. The surveyor started at RP (x_1, y_1) at time t_1, visited RP (x_5, y_5) at t_5, and reached RP (x_8, y_8) at t_8. The RSSI records capture additional RSSI data, e.g., at time t_2, RSSIs of the 1st, 2nd, and 3rd APs are -70 dBm, -83 dBm and -76 dBm, respectively.

As it is possible that the two types of records are collected asynchronously, a pre-processing method has been widely used [51] to create the radio map as follows.
- **Step 1** merges consecutive RSSI records if their time difference is below a threshold ϵ. The merged record uses the earlier time, and gets its RSSIs as follows. If an AP is in one record only, that RSSI is used. If an AP is in both records, the average RSSI is used. Otherwise, `null` is used.
- **Step 2** merges consecutive RSSI and RP records if their times differ by less than ϵ. The time and RSSIs are as produced in Step 1; the RP is copied from the RP record. Each remaining RSSI or RP record is converted into a record in which each missing value is set to `null`.

The threshold ϵ is specified by the surveyor. Setting $\epsilon = 1$ for Table II, we get the radio map records and times in Table III. Though a radio map does not contain timestamps, we show them in Table III because we use them for imputation later on. In Step 1, RSSI records at t_6 and t_7 are merged into $\langle r_1 : -74, r_2 : -77, r_3 : \texttt{null}, r_4 : \texttt{null}, r_5 : -81 \rangle$ at $t_6 = 12$. In Step 2, this new record is not merged with an RP record but is converted to a pair $((-74, -77, \texttt{null}, \texttt{null}, -81), \texttt{null})$. In contrast, the RP record at t_1 is merged with the RSSI record at t_2, resulting in the pair $((-70, -83, -76, \texttt{null}, \texttt{null}), (x_1, y_1))$. Likewise, records at t_4 and t_5 are merged into $((\texttt{null}, \texttt{null}, -80, -68, \texttt{null}), (x_5, y_5))$. Moreover, the RP record at t_8 is converted to $((\texttt{null}, \texttt{null}, \texttt{null}, \texttt{null}, \texttt{null}), (x_8, y_8))$.

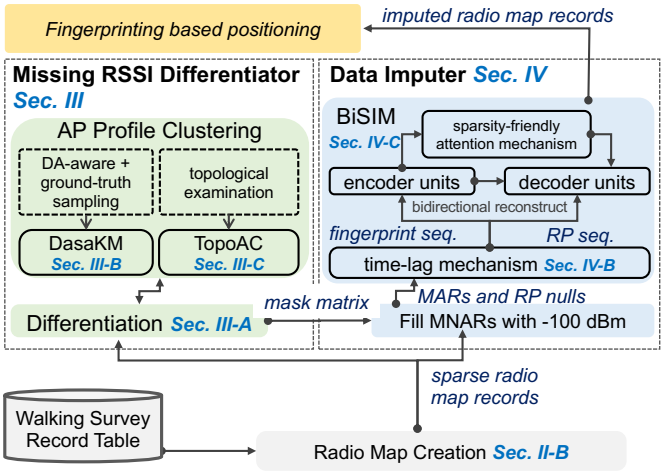


Fig. 4: Framework overview.

This method generates temporally dense radio map records that, however, may contain many RP and RSSI `nulls`.

C. Problem and Solution Overview

Problem (Radio Map Imputation)**.** *Given a radio map, we impute the RSSI and RP* `null` *values in the radio map, such that indoor positioning using this radio map yields lower positioning errors.*

As pointed out in Section I, missing RSSIs are random or non-random, yielding *Missing At Random* (MAR) and *Missing Not At Random* (MNAR) RSSIs. They should be differentiated before data imputation [47]. To solve this problem, we propose a framework (cf. Fig. 4) with two modules.

Missing RSSI Differentiator Module (Section III). Given a radio map, this module categorizes missing RSSIs as MNARs and MARs. Specifically, the differentiation process (Section III-A) regards MARs as random absences of AP signals in fingerprints and employs a clustering based approach to identify those random absences according to the locality of AP profiles (i.e., observability of APs). We design two algorithms (Sections III-B and III-C) for clustering AP profiles in different ways. The differentiation process returns the recognized MNARs and MARs as a mask matrix, where -1 means MNAR, 0 means MAR, and 1 means an observed RSSI.

Data Imputer Module (Section IV) imputes missing RSSI and RP values. Initially, all MNARs are assigned the value -100 dBm. Then, a bidirectional encoder-decoder based model called BiSIM (Section IV-A) imputes MAR and RP `nulls` jointly for a sequence of radio map records from a survey path. In particular, BiSIM considers the aging of records by applying a *time-lag mechanism* (Section IV-B) to sequential radio map records. BiSIM subsequently encodes fingerprint feature sequences and decodes the corresponding RP feature sequences to capture correlations in a radio map record and among sequential radio map records. BiSIM also employs a *sparsity-friendly attention mechanism* (Section IV-C) to perform weight calculation against missing values. The fingerprints and RPs predicted sequentially by the encoder/decoder units form the final imputed radio map records.

III. Missing RSSI Differentiator

A. Differentiation Approach

In a wireless setting, identifying MNARs is non-trivial due to the complexity of analyzing the signal transmission paths between RPs and APs [48]. To this end, we instead identify MARs as "unusual" RSSI missing events when comparing to observed RSSIs in the same or similar signal environments. We thus rely on the following **hypothesis**: *Within a certain small range of space, the observability of APs is similar due to the similar signal transmission surroundings.*

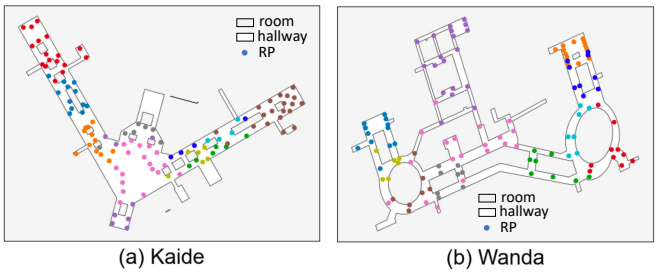
To verify this hypothesis, we did an exploratory analysis on two real-world shopping malls named Kaide and Wanda that we describe in detail in Section V-A.

First, we generate an **AP profile** for each observed RP by a process called Binarization. The process of Binarization is shown in Algorithm 1. We assume each RP corresponds to one fingerprint. In case multiple fingerprints are generated for an RP, the fingerprints are averaged into one. The process constructs a D-dimensional binary vector $\mathbf{b}_i$ for the RP $\mathbf{l}_i$: $\mathbf{b}_i[d] = 1$ if the dth AP is observed at $\mathbf{l}_i$, and $\mathbf{b}_i[d] = 0$, otherwise.

Algorithm 1 Binarization (an RP $\mathbf{l_i}$'s fingerprint $\mathbf{f}_i$)

```
binary vector b_i ← 1^D
for d = 1 to D do
    if f_i[d] is null then b_i[d] ← 0
return b_i
```

Next, we conducted a clustering of the binarized AP profiles. We use the widely-used K-means using Euclidean distance[3] and tune the hyperparameter K carefully. We color the resulting clusters and visualize the RPs in Fig. 5. We see that in most cases, the similar AP profiles (in the same cluster) are spatially close to each other. Although some exceptions occur due to noise (MARs) in fingerprints when generating the AP profiles, the hypothesis holds.



(a) Kaide (b) Wanda

Fig. 5: Preliminary clustering tests on real-world venues.

We thus identify MARs based on the clustering of AP profiles. The idea is that if a value r_d is missing in an AP profile $\mathbf{b}_i$ while an r_d value is present in a certain fraction of

[3]We also considered Manhattan distance, but it achieved inferior results. We thus employ Euclidean distance for K-means unless stated otherwise.

AP profiles similar to $\mathbf{b}_i$ in the same cluster, the missing r_d is likely to be a MAR in the fingerprint. To this end, a threshold η is used such that a fraction higher than η indicates MARs.

Algorithm 2 formalizes the differentiator with a predefined fraction threshold η as the input. It returns an $N \times D$ mask matrix $\mathbf{M}$ (initialized in line 1), where $\mathbf{M}[i,j]$ is 0 if the jth ($1 \leq j \leq D$) AP dimension of the ith fingerprint ($1 \leq i \leq N$) in the radio map is a MAR, -1 if it is an MNAR, and 1 if it is observed. Lines 2–5 construct the sample set X for clustering. We highlight two differences related to X in Algorithm 2 versus the exploratory analysis: First, each sample in X is a concatenation of the AP profile and the RP location. This enables us to utilize prior knowledge of RP locations to form clusters with spatially close RPs. Second, X covers all radio map records including those with `null` RPs. To this end, each `null` RP is interpolated linearly based on its previously and subsequently observed RPs in the radio map. Although imprecise, these interpolated RP positions capture spatial proximity, which improves the clustering effectiveness. Line 6 generates a set C of clusters by one of two clustering algorithms (to be detailed in Sections III-B and III-C).

Lines 7–12 identify MARs in each cluster c_k by considering all its AP profiles. In particular, each AP dimension r_j is checked (line 8) to determine whether the missing of r_j in c_k is unusual. If η_j, the fraction of observed r_j across all samples in c_k, exceeds the threshold η, r_j `nulls` are MARs and marked as 0 in $\mathbf{M}$. Otherwise, they are MNARs and marked as -1 (lines 9–12). Finally, $\mathbf{M}$ is returned.

Algorithm 2 DIFFERENTIATION (fraction threshold η)

1: mask matrix $\mathbf{M} \leftarrow \mathbf{1}^{N \times D}$
2: sample set $X \leftarrow \varnothing$
3: **for** each record $(\mathbf{f}_i, \hat{\mathbf{l}}_i)$ in the radio map **do**
4: $\quad \mathbf{x}_i \leftarrow \text{BINARIZATION}(\mathbf{f}_i) \oplus \hat{\mathbf{l}}_i$ ▷ $\hat{\mathbf{l}}_i$ is interpolated linearly
5: $\quad$ add $\mathbf{x}_i$ to X
6: $C \leftarrow \text{CLUSTERING}(X)$
7: **for** each cluster $c_k \in C$ **do**
8: $\quad$ **for** each AP dimension r_j **do**
9: $\quad\quad \eta_j \leftarrow$ the fraction of observed r_j for all samples in c_k
10: $\quad\quad$ **if** $\eta_j > \eta$ **then**
11: $\quad\quad\quad$ mark all r_j `nulls` within c_k as 0 in $\mathbf{M}$ ▷ MARs
12: $\quad\quad$ **else** mark all r_j `nulls` within c_k as -1 in $\mathbf{M}$ ▷ MNARs
13: **return** $\mathbf{M}$

Algorithm 2 works with different clustering algorithms. Section III-B presents DasaKM (Differentiation accuracy aware, sampling-based K-means) to replace the manually-tuned K-means used in the exploratory analysis. In Section III-C, we utilize indoor topology information and devise TopoAC (Topology-aware Agglomerative Clustering) that achieves even better performance without hyperparameters. In Section V-B, Algorithm 2 is evaluated experimentally with different clustering algorithms in terms of indoor positioning error. In general, TopoAC performs better as it takes the indoor topology into account, while DasaKM does not require any prior knowledge.

B. Algorithm DasaKM

A straightforward way of applying K-means is to use the elbow method [33] that employs a *within-cluster sum of square* metric to examine intra-cluster similarity. This method, however, leads to subpar performance at missing RSSI differentiation (see evaluations in Section V-B) as it disregards our ultimate goal of differentiation. To address this, we propose a more intuitive metric called differentiation accuracy (DA) that measures the differentiation ability of the clustering result. As the ground-truth MARs and MNARs are not known, we first propose a ground-truth sampling procedure.

Ground-truth Sampling Procedure. It is non-trivial to generate the ground-truth mask matrix $\mathbf{M}_g$ by manually differentiating MARs and MNARs. Thus, we modify the original sample set X to "create" ground-truth MARs and MNARs:

- *Sampling MARs*. We nullify some observations in a record and mark them as 0 in $\mathbf{M}_g$. They correspond to random RSSI missing events that are actually observable.
- *Sampling MNARs*. We search the indoor venue to sample a set of adjacent RPs that cover a sufficiently large area in the venue[4]. These RPs are likely to share a similar AP profile. If such RPs all missed an AP dimension in their records, then their corresponding missing values should be MNARs. The relevant masks in $\mathbf{M}_g$ are set to -1 accordingly.

Ideally, MARs and MNARs should be sampled according to their real distributions in the original dataset, which are, however, unknown. Hence, to mitigate potential ground-truth sampling biases, we propose to sample multiple ground-truth sets using different proportions of MARs and MNARs and measure the average accuracy on these ground-truth sets. Moreover, we design the differentiation accuracy as a balanced metric that is agnostic to the imbalanced proportion of the sampled ground-truth set.

Differentiation Accuracy Metric. The design of DA is based on the metric called *balanced accuracy*, which is shown to be effective for imbalanced positive and negative samples [5], [22]. Specifically, DA computes the true positive rate as the fraction of positive samples (MARs) identified correctly, and the true negative rate as the fraction of negative samples (MNARs) identified correctly. Then, DA simply takes the arithmetic average of the true positive rate and the true negative rate, thus disregarding the ratio of positive and negative ground-truth samples. The arithmetic average used by DA implies that either class is of equal importance for differentiation. In contrast, the conventional F-score measures only the performance of identifying positive samples—its *precision* and *recall* measure the fractions of correct positive samples in the result and positive samples being returned, respectively. Thus, the F-score is not used to implement DA.

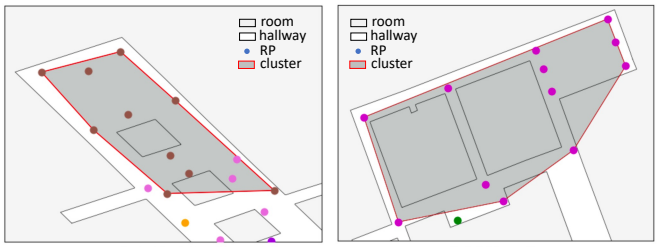
[4]In our implementation, we fix the RP size to 6. It forms a sufficiently large area and also avoids extra search cost caused by a larger size.

Algorithm. DasaKM (Algorithm 3) first generates iteratively a ground-truth set GS_γ in a particular input proportion of sampled MARs and MNARs and then removes it from the input dataset to form X_γ (lines 1–3). Next, it goes through a set of K values until reaching a predefined upper-bound U and selects the optimal $\hat{K}$ as the one achieving the highest DA (lines 4–10). For each K, DA is averaged over different ground-truth datasets (lines 6–9). Finally, the K-means clustering on the original data X using $\hat{K}$ is returned (line 11).

Algorithm 3 DASAKM (sample set X, proportion list Γ, upper-bound U)

1: **for** proportion $\gamma \in \Gamma$ **do**
2: sample a ground-truth set GS_γ from X such that $\gamma = \frac{\#(\text{MNARs})}{\#(\text{MARs})}$
3: $X_\gamma \leftarrow X \setminus GS_\gamma$
4: $maxDA \leftarrow 0$; $\hat{K} \leftarrow 0$
5: **for** $K = 1$ to U **do**
6: **for** $\gamma \in \Gamma$ **do** ▷ try different sampled datasets
7: $C_\gamma \leftarrow \text{KMEANS}(X_\gamma, K)$
8: $DA_\gamma \leftarrow$ calculate DA w.r.t C_γ and GS_γ
9: $\widehat{DA} \leftarrow \text{average}(\{DA_\gamma \mid \gamma \in \Gamma\})$
10: **if** $\widehat{DA} > maxDA$ **then** $\hat{K} \leftarrow K$
11: **return** $\text{KMEANS}(X, \hat{K})$

DasaKM finds close samples based on inter-vector distances in a transformed signal space, which may, however, batch samples having distinct signal transmission surroundings in the indoor space. We have found two abnormal cases, shown as the two resultant clusters in Fig. 6. Their RPs scatter around the rooms, and their AP profiles may differ largely due to the existence of the walls among them which constitute distinct signal transmission environments.



(a) Kaide (b) Wanda

Fig. 6: Result of DasaKM.

C. Algorithm TopoAC

To avoid abnormal cases and improve accuracy, we design the Topology-aware Agglomerative Clustering (TopoAC) that considers the topology of the indoor space.

Heuristic of Topology. If a set of RPs share similar AP profiles, there should not exist topological entities such as walls and obstacles that cause non-line-of-sight signal propagation within the closed region of these RPs. In other words, if the convex hull of a set of RPs contains topological entities, these RPs should not form a cluster. The basis for such a heuristic is formalized in Algorithm 4. It takes as input a cluster c and topological entities $\mathcal{T}$ in the form of a multipolygon. It returns True if any entities exist in the convex hull CH formed by the locations in cluster c. Otherwise, it returns False. For example, the case in Fig. 6(a) returns True because that cluster CH intersects polygons in $\mathcal{T}$.

Algorithm 4 ENTITYEXIST (cluster c, multipolygon $\mathcal{T}$)

1: location set $L \leftarrow \{l_i \mid x_i = (f_i, l_i) \wedge x_i \in c\}$
2: $CH \leftarrow$ convex hull covering L
3: **return** $(CH \setminus \mathcal{T} \neq \varnothing)$

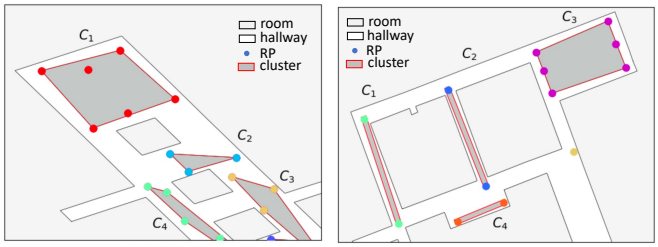
Integrating the Topology Heuristic into the Algorithm. The above topology heuristic can be integrated naturally into an agglomerative clustering process where two adjacent clusters are merged if the resulting cluster passes the examination of Algorithm 4. Note that the heuristic does not work with K-means, where it is too complex to assign samples to clusters while satisfying the heuristic.

The integrated clustering is detailed as TopoAC in Algorithm 5. Initially, each sample $\mathbf{x}_i$ forms a single cluster c_i. It then iteratively merges the pair of clusters with the minimum center-to-center Euclidean distance that passes the topological examination (lines 2–4). It terminates when no clusters can be merged. TopoAC does not require any hyperparameters.

Algorithm 5 TOPOAC (sample set X, multipolygon $\mathcal{T}$)

1: initialize $C \leftarrow \{c_i \mid \text{for each } \mathbf{x}_i \in X\}$
2: **while** $\exists$ cluster pair (c_i, c_j) *s.t.* !ENTITYEXIST$(c_i \cup c_j, \mathcal{T})$ **do**
3: pick (c'_i, c'_j) with the minimum distance *s.t.* !ENTITYEXIST$(c'_i \cup c'_j, \mathcal{T})$
4: merge c'_i and c'_j in C
5: **return** C

Results of TopoAC for the settings in Fig. 6 are visualized in Fig. 7. Each abnormal cluster in Fig. 6 is divided into smaller clusters, each spanning an open area.



(a) Kaide (b) Wanda

Fig. 7: Result of TopoAC.

IV. DATA IMPUTER

After the differentiation of missing RSSIs, the data imputer first replaces all identified MNARs with -100 dBm and changes their corresponding -1 in the mask matrix $\mathbf{M}$ to 1.

The amended matrix, denoted as $\mathbf{M}'$, contains 0s only for MARs and 1s for MNARs and observed RSSIs.

Subsequently, the data imputer imputes MARs and RP `null`s jointly using a sequential neural network. The intuition is that radio map records on the same survey path are temporally correlated and the fingerprint and RP in one record are also correlated. To capture the correlations among sequential records and in each radio map record, we propose a Bi-directional Sequence-to-Sequence Imputation Model (BiSIM).

A. *BiSIM Architecture*

The BiSIM architecture is shown in Fig. 8. The encoder-decoder [16] architecture enables BiSIM to handle heterogeneous input data such that fingerprint and RP data sequences can be fed into the encoder and decoder units, respectively. In Fig. 8, the tail of the encoders (the yellow part) is connected to the head of the decoders (the blue part) via a hidden vector $\mathbf{h}_T = \mathbf{s}_0$, meaning that the fingerprint sequence can decode the underlying RP sequence. Note that conventional RNN-based imputation models [11], [13], [17], [44], [56] can only handle homogeneous data sequences and thus fall short in our setting.

In general, BiSIM receives a sequence of T radio map records on a survey path as input and outputs a corresponding sequence of T imputed records. Its data flow is as follows.

First, the features of the ith ($1 \leq i \leq T$) fingerprint in the sequence is fed to an **encoder unit**. The input feature consists of three components $(\delta_i, \mathbf{f}_i, \mathbf{m}_i)$, to be detailed in Section IV-B. The ith encoder unit generates an imputed vector $\mathbf{f}_i^c$ as well as a latent vector $\mathbf{h}_i$ to be passed to the next encoder unit. The initial latent vector $\mathbf{h}_0$ is randomized.

Second, the features of the jth ($1 \leq j \leq T$) RP in the sequence is fed to a **decoder unit**. As also to be introduced in Section IV-B, its input consists of two components $(\mathbf{l}_j, \mathbf{k}_j)$. The jth decoder unit transforms the input features into an imputed RP vector $\mathbf{l}_j^c$ by utilizing the latent vector $\mathbf{s}_{j-1}$ from its preceding decoder unit, and it generates $\mathbf{s}_j$ that will be passed to the next decoder unit.

As the latent vector $\mathbf{s}_0$ ($\mathbf{h}_T$) is learned from the fingerprint sequence as a whole, we introduce a sparsity-friendly attention mechanism to make the decoder unit aware of on which parts of the fingerprint sequence to focus. The jth **attention unit** (in pink in Fig. 8) receives the latent vectors $\{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$ from all encoder units and the latent vector $\mathbf{s}_{j-1}$ from the $(j-1)$th decoder unit and then generates a context vector $\mathbf{c}_j$ that is passed to the jth decoder unit for generating $\mathbf{l}_j^c$.

The internals of BiSIM, including the encoder unit, decoder unit, and attention unit, are detailed in Section IV-C. Above, we covered the encoding-decoding process in the forward direction. Indeed, we also capture the backward dependencies of the feature sequences. As shown at the bottom of Fig. 8, we feed the feature sequences backwards to obtain another set of imputed vectors. Our loss function takes into account the imputed vectors obtained from both forward and backward inputs, to be covered in Section IV-D.

TABLE IV: Input Features for BiSIM

		r_1	r_2	r_3	r_4	r_5		x	y
mask vec.	$\mathbf{m}_1$	1	1	1	0	0	$\mathbf{k}_1$	1	1
	$\mathbf{m}_2$	1	0	1	0	0	$\mathbf{k}_2$	0	0
	$\mathbf{m}_3$	0	0	1	1	0	$\mathbf{k}_3$	1	1
	$\mathbf{m}_4$	1	1	0	0	1	$\mathbf{k}_4$	0	0
	$\mathbf{m}_5$	0	0	0	0	0	$\mathbf{k}_5$	1	1
time-lag vec.	δ_1	0	0	0	0	0			
	δ_2	3	3	3	3	3			
	δ_3	5	8	5	8	8			
	δ_4	9	12	4	4	12			
	δ_5	4	4	8	8	4			

B. *Input Feature Preparation*

Fingerprint Input Feature. Given a fingerprint $\mathbf{f}_i$, the corresponding row in the mask matrix $\mathbf{M}'$ is retrieved as $\mathbf{m}_i$. The mask vector $\mathbf{m}_i$ records which AP values of $\mathbf{f}_i$ are `null`s. In the encoding stack, each unit's encoding depends on the latent vector from the previous unit. Intuitively, a latent vector from a more distant time should exert less influence on the current unit. To reflect this time decay effect on encoding, we introduce a *time-lag vector* [11], [44] $\boldsymbol{\delta}_i = \langle \delta_i^1, \ldots, \delta_i^j, \ldots, \delta_i^D \rangle$ for each input fingerprint $\mathbf{f}_i$, where

$$\delta_i^j = \begin{cases} 0 & \text{if } i = 1 \\ t_i - t_{i-1} & \text{if } i > 1 \wedge \mathbf{m}[i-1, j] = 1 \\ \delta_{i-1}^j + (t_i - t_{i-1}) & \text{if } i > 1 \wedge \mathbf{m}[i-1, j] = 0 \end{cases} \quad (1)$$

In Eq. 1, each time-lag vector value δ_i^j for the first encoder unit is set to 0 by default. For other units, we differentiate two cases. If the previous observation is not `null` (i.e., $\mathbf{m}[i-1, j] = 1$), the value is simply the difference between the current time and the previous time, i.e., $t_i - t_{i-1}$. Otherwise, the value is the sum of $(t_i - t_{i-1})$ and δ_{i-1}^j (the value of the previous time). Note that only observed values from the previous time affect the current encoder unit. In this sense, δ_i^j in Eq. 1 keeps track of the difference between the current time and the last observation's time.

RP Input Feature. Given a RP $\mathbf{l}_j$, we generate a mask vector $\mathbf{k}_j \in \{0, 1\}^2$ as follows. If $\mathbf{l}_j$ is not `null` then $\mathbf{k}_j = \langle 1, 1 \rangle$; otherwise, $\mathbf{k}_j = \langle 0, 0 \rangle$. We have generated a similar time-lag vector for $\mathbf{l}_j$ as the decoder input. However, ablation studies in Section V-C show such extra decoder input brings about no gains. As time decay has been captured by encoder units, a more complex structure may degrade model generalizability.

Example 1. *Table IV shows mask vectors $\mathbf{m}_1$ to $\mathbf{m}_5$ and $\mathbf{k}_1$ to $\mathbf{k}_5$ for Table III. Fingerprints' time-lag vectors are generated as follows. According to Eq. 1, $\boldsymbol{\delta}_1$ is simply $\langle 0, 0, 0, 0, 0 \rangle$. For $\mathbf{f}_2$ of time $t_3 = 3$ in Table III, the values δ_2^1 to δ_2^3 all equal to $t_3 - t_1 = 3$; the value δ_2^4 equals to $\delta_1^4 + (t_3 - t_1) = 3$ as $\delta_1^4 = 0$, and $\delta_2^5 = 3$ follows a similar computation as δ_2^4. For $\mathbf{f}_2$ of time $t_4 = 8$ in Table III, $\delta_3^1 = t_4 - t_3 = 8 - 3 = 5$, whereas $\delta_3^2 = \delta_2^1 + (t_4 - t_3) = 3 + (8 - 3) = 8$. The subsequent computations are performed similarly.*

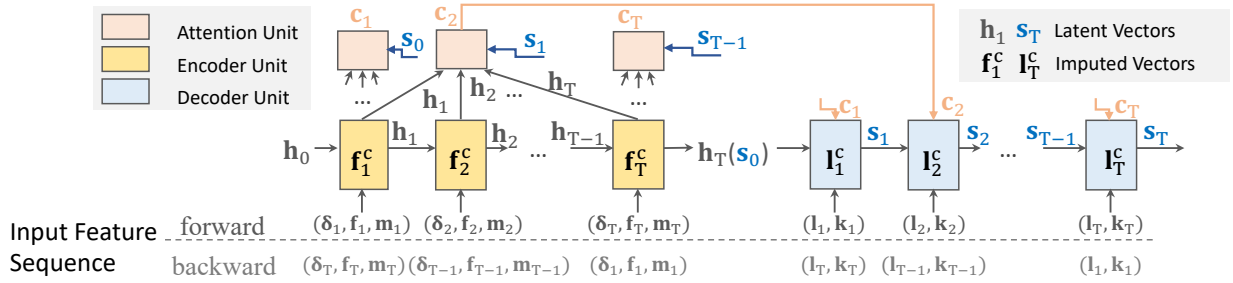


Fig. 8: The encoder-decoder architecture of BiSIM.

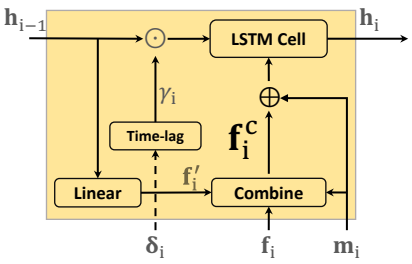


Fig. 9: The ith encoder unit.

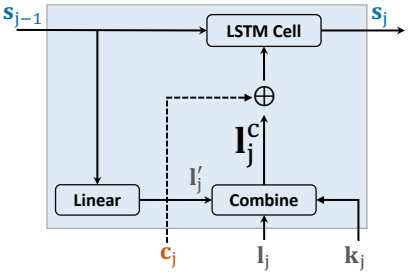


Fig. 10: The jth decoder unit.

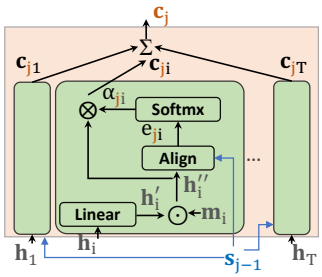


Fig. 11: The jth attention unit.

C. Internals of BiSIM

Unlike traditional encoder-decoder models, BiSIM must handle the `nulls` in the network units. This is achieved by including the mask vectors $\mathbf{m}_i$ and $\mathbf{k}_j$ in the computation. Taking the forward feature input as an example, we elaborate on each type of unit as follows.

Encoder Unit. Fig. 9 shows the ith encoder unit's internals. It takes $\mathbf{f}_i$, $\mathbf{m}_i$, δ_i, and the previous latent vector $\mathbf{h}_{i-1}$ as input, and it generates an intermediate imputed vector $\mathbf{f}_i^c$ and the current latent vector $\mathbf{h}_i$. The formulas are given below.

$$\mathbf{f}'_i = \mathbf{W}_f \mathbf{h}_{i-1} + \mathbf{b}_f \tag{2}$$

$$\mathbf{f}^c_i = \mathbf{m}_i \odot \mathbf{f}_i + (\mathbf{1} - \mathbf{m}_i) \odot \mathbf{f}'_i \tag{3}$$

$$\gamma_i = \exp\big(-\max(0, \mathbf{W}_\gamma \delta_i + \mathbf{b}_\gamma)\big) \tag{4}$$

$$\mathbf{h}_i = \sigma\big(\mathbf{W}_h(\mathbf{h}_{i-1} \odot \gamma_i) + \mathbf{U}_h(\mathbf{f}^c_i \oplus \mathbf{m}_i) + \mathbf{b}_h\big) \tag{5}$$

Above, matrices $\mathbf{W}_*$ and $\mathbf{U}_*$ and vectors $\mathbf{b}_*$ in the network units are learnable parameters. Eq. 2 is a linear operator that maps the previous latent vector $\mathbf{h}_{i-1}$ to an estimated fingerprint vector $\mathbf{f}'_i$. Eq. 3 is a combination operator that replaces the missing value in the fingerprint $\mathbf{f}_i$ with the corresponding values in the estimated fingerprint $\mathbf{f}'_i$. It performs an element-wise product (i.e., $\odot$) of the fingerprint vectors and the mask vector $\mathbf{m}_i$. The resulting complemented vector $\mathbf{f}^c_i$ forms the imputation result for $\mathbf{f}_i$ (see Eq. 13). Eq. 4 generates a scalar *temporal decay factor* γ_i based on the time-lag vector δ_i. Generally speaking, a larger δ_i leads to a smaller γ_i, capturing that the effect of a past observation is reduced if the observation is temporally distant. Finally, the temporal decay factor γ_i is applied to $\mathbf{h}_{i-1}$, and the result is passed to a standard LSTM cell along with the imputed fingerprint $\mathbf{f}^c_i$ concatenated with $\mathbf{m}_i$. The LSTM cell's computation is formalized in Eq. 5, where $\sigma(\cdot)$ is the sigmoid function and $\oplus$ is the concatenation operator.

Decoder Unit. Shown in Fig. 10, the internal of a decoder unit is similar to that of an encoder unit, except that no time-lag vector is used. The formulas are given below. In particular, the latent vector $\mathbf{s}_{j-1}$ is mapped to an estimated RP vector $\mathbf{l}'_j$ through a linear operator (Eq. 6). Then, $\mathbf{l}'_j$ is used to replace the `null` RP vector $\mathbf{l}_j$ in a combination operation (Eq. 7). Finally, the concatenation of the resulting imputed vector $\mathbf{l}^c_j$ and the context vector $\mathbf{c}_j$ is passed to an LSTM cell along with the latent vector $\mathbf{s}_{j-1}$. The LSTM cell in Eq. 8 generates the latent vector $\mathbf{s}_j$ for the next unit.

$$\mathbf{l}'_j = \mathbf{W}_l \mathbf{s}_{j-1} + \mathbf{b}_l \tag{6}$$

$$\mathbf{l}^c_j = \mathbf{k}_j \odot \mathbf{l}_j + (\mathbf{1} - \mathbf{k}_j) \odot \mathbf{l}'_j \tag{7}$$

$$\mathbf{s}_j = \sigma\big(\mathbf{W}_s \mathbf{s}_{j-1} + \mathbf{U}_s(\mathbf{l}^c_j \oplus \mathbf{c}_j) + \mathbf{b}_s\big) \tag{8}$$

Attention Unit. As shown in Fig. 11, the jth attention unit generates a context vector $\mathbf{c}_j$ to help the jth decoder selectively retrieve information from the fingerprint sequence in decoding the corresponding RP vector. We employ the Bahdanau attention mechanism [9], which can dynamically capture the relationship between the current decoding moment and each past encoding moment and then assign higher attention (i.e., weights) to the more related encoding moments. However, the original Bahdanau attention does not consider the incompleteness in the input of an encoder unit, which may involve noise in the resulting latent vector. To avoid this, we design a sparsity-friendly variant of the Bahdanau attention, by allowing only observed values' latent vectors to participate in the computation. Specifically in Eq. 9, we transform each latent vector $\mathbf{h}_i$ linearly to $\mathbf{h}'_i$ and retain only the observed part

of $\mathbf{h}'_i$ by performing an element-wise product of $\mathbf{h}'_i$ and $\mathbf{m}_i$.

$$\mathbf{h}'_i = \mathbf{W}_a \mathbf{h}_i + \mathbf{b}_a; \quad \mathbf{h}''_i = \mathbf{h}'_i \odot \mathbf{m}_i \tag{9}$$

$$e_{ji} = \text{MLP}(\mathbf{s}_{j-1}, \mathbf{h}''_i) \tag{10}$$

$$\alpha_{ji} = \exp(e_{ji}) / \sum_{k=1}^{T} \exp(e_{jk}) \tag{11}$$

$$\mathbf{c}_j = \sum_{i=1}^{T} \mathbf{c}_{ji}; \quad \mathbf{c}_{ji} = \alpha_{ji} \mathbf{h}''_i \tag{12}$$

Next, Eq. 10 – 12 use the original Bahdanau attention [9]. In particular, Eq. 10 implements an alignment function that aligns $\mathbf{s}_{j-1}$ and $\mathbf{h}''_i$ into an energy factor e_{ji} based on a Multilayer Perceptron (MLP). The energy factor reflects the importance of the encoder's latent vector $\mathbf{h}''_i$ with respect to the decoder's latent vector $\mathbf{s}_{j-1}$ in generating $\mathbf{s}_j$, the next decoder's latent vector. Afterwards, e_{ji} is normalized into a weight α_{ji} by a softmax function, in Eq. 11. With such weights, we calculate the context vector $\mathbf{c}_j$ as a weighted sum of all $\mathbf{h}''_i$s, in Eq. 12.

D. Output and Loss Function

Recall that we generate two pairs of imputed vectors, i.e., $\mathbf{f}^c_{i,\succ}$ and $\mathbf{l}^c_{i,\succ}$ for forward input features, and $\mathbf{f}^c_{i,\prec}$ and $\mathbf{l}^c_{i,\prec}$ for backward input features. We average the vectors from both directions to get the final output. Formally, we have:

$$\hat{\mathbf{l}}_i = (\mathbf{l}^c_{i,\succ} + \mathbf{l}^c_{i,\prec})/2; \quad \hat{\mathbf{f}}_i = (\mathbf{f}^c_{i,\succ} + \mathbf{f}^c_{i,\prec})/2 \tag{13}$$

As we lack ground-truth of the imputed results in model training, we base our loss function on the reconstruction errors between the observed values in the radio map and the corresponding values predicted by the model. Intuitively, if the model makes predictions close to the original observed values, the model is likely to impute missing values reliably [11]. The overall loss $\mathcal{L}^o$ of BiSIM is defined as follows.

$$\mathcal{L}^o = \mathcal{L}^{forward} + \mathcal{L}^{backward} + \mathcal{L}^{cross}, \text{where}$$

$$\mathcal{L}^{forward} = 1/T \cdot \sum_{i=1}^{T} \left(\mathcal{L}(\mathbf{f}'_{i,\succ}, \mathbf{f}_{i,\succ}, \mathbf{m}_i) + \mathcal{L}(\mathbf{l}'_{i,\succ}, \mathbf{l}_{i,\succ}, \mathbf{k}_i)\right)$$

$$\mathcal{L}^{backward} = 1/T \cdot \sum_{i=1}^{T} \left(\mathcal{L}(\mathbf{f}'_{i,\prec}, \mathbf{f}_{i,\prec}, \mathbf{m}_i) + \mathcal{L}(\mathbf{l}'_{i,\prec}, \mathbf{l}_{i,\prec}, \mathbf{k}_i)\right)$$

$$\mathcal{L}^{cross} = 1/T \cdot \sum_{i=1}^{T} \left(\mathcal{L}(\mathbf{f}'_{i,\succ}, \mathbf{f}'_{i,\prec}, \mathbf{m}_i) + \mathcal{L}(\mathbf{l}'_{i,\succ}, \mathbf{l}'_{i,\prec}, \mathbf{k}_i)\right)$$

$$\mathcal{L}(\mathbf{a}, \mathbf{a}', \mathbf{mask}) = \text{MSE}(\mathbf{mask} \odot \mathbf{a}, \mathbf{mask} \odot \mathbf{a}')$$

Above, $\mathbf{f}_{i,\succ}$ and $\mathbf{l}_{i,\succ}$ (resp. $\mathbf{f}_{i,\prec}$ and $\mathbf{l}_{i,\prec}$) are the forward (resp. backward) input features. The overall loss $\mathcal{L}^o$ consists of three terms. The forward loss $\mathcal{L}^{forward}$ captures the reconstruction error of the forward imputation results. The backward loss $\mathcal{L}^{backward}$ captures the reconstruction error of the backward imputation results. The cross loss $\mathcal{L}^{cross}$ captures the closeness between each pair of forward and backward imputation results. To measure reconstruction errors, we use the predicted vector (e.g., $\mathbf{f}'_{i,\succ}$ in Eq. 2) instead of the final imputation result (e.g., $\mathbf{f}^c_{i,\succ}$) because the observed part of the final imputation result comes directly from the input feature (e.g., $\mathbf{f}_{i,\succ}$). The function $\mathcal{L}(\mathbf{a}, \mathbf{a}', \mathbf{mask})$ measures the MSE (mean square error) between the *observed* parts of the input vectors $\mathbf{a}$ and $\mathbf{a}'$, where $\mathbf{mask}$ is a mask vector for retaining the original observed values in the input vectors. In particular, $\mathbf{mask}$ is $\mathbf{m}$ and $\mathbf{k}$ for fingerprints and RPs, respectively.

V. Experimental Studies

A. Experimental Settings

All algorithms are coded in Python 3.8 and run on a Linux server with 3.60 GHz Intel Core i9 CPU and NVIDIA RTX 3080 GPU with 12 GB memory. All neural network models are implemented using PyTorch 1.6 and trained on the GPU. The code, datasets, and tuning details are available online [3].

Datasets and Real Indoor Venues. We use real-world indoor positioning datasets [4] published by Microsoft Research, which encompass walking survey records, building topological information, and online testing data collected from shopping malls in China. For our studies, we randomly pick two malls: Kaide Mall and Wanda Square as Wi-Fi fingerprinting scenarios. In addition, to gain insights into the effectiveness of our proposals in other application scenarios and indoor venues, we conducted additional experiments using Bluetooth fingerprinting data from a different indoor venue named Longhu. For radio map creation, the parameter ϵ is set to 1 second for both venues. The characteristics of the venues and radio maps are given in Table V. Wanda features a larger radio map with a higher fingerprint dimensionality and more fingerprints, whereas Kaide features a higher RP density. Note that the APs in Longhu are Bluetooth-based instead of Wi-Fi based.

TABLE V: Statistics of Venues and Created Radio Maps

Venue	Kaide	Wanda	Longhu
Floor Area (m^2)	3225.7	4458.5	6504.1
RP density (per 100 m^2)	3.53	2.65	3.11
# of fingerprints	894	4104	4617
# of RPs	114	118	202
# of APs (i.e., # of fingerprint dimensions)	671	929	330

Evaluation Controls. To evaluate our overall solution framework with an MNAR/MAR differentiator A and a data imputer B, we employ an online location estimation algorithm C as follows. Given an original radio map, we select 10% of the records with observed RPs as testing data and use the RPs as ground-truth locations for evaluation. Modules A and B are combined to impute both testing data and the rest radio map records. After that, the remaining records form a radio map used by C to estimate the locations on the testing data[5].

Given different combinations of A, B, and C, we use the method of control variates in the evaluations. In Section V-B, we compare different differentiators (A), fixing B to BiSIM and C to W*K*NN. BiSIM and W*K*NN together perform best across different differentiators, to be shown in Section V-C, where we compare different data imputers (B) across different combinations of A and C.

[5]We also apply imputation to the (online) fingerprints in the test data. Usually, complete online fingerprints are obtained by using techniques unavailable or unaffordable for walking surveys [34].

B. Evaluation of Differentiators

1) Setting: **Methods**. Based on Algorithm 2, we evaluate three differentiators using different clustering methods[6], namely our DasaKM and TopoAC, and K-means based on the elbow method for K selection [33] (denoted as ElbowKM). For DasaKM and ElbowKM that decide K through iterations, we set K's upper-bound U to 200. To sample ground-truth sets in DasaKM, we fix the number of sampled MNARs (6960 for Kaide and 9612 for Wanda) and take the proportion $\gamma = \frac{\#(\text{MNARs})}{\#(\text{MARs})}$ from the list $\Gamma = (1, 2, \ldots, 20)$. The proportion starts from 1 as there should be more MNARs than MARs (i.e., random events) in practice. We also implement two baselines without differentiation: MAR-only treats all missing RSSIs as MARs, and MNAR-only treats all as MNARs.

Parameters. First, we examine how differentiators are affected by the sparsity of input radio maps. Specifically, we introduce a *removal ratio* $\alpha \in \{0, 5, 10, 15, 20\}\%$ such that a fraction α of RSSIs are randomly selected and nullified in an original radio map. As a result, the input radio map has $\{85.6, 86.3, 87.0, 87.7, 88.4\}\%$ missing RSSIs for Kaide, and $\{93.1, 93.4, 93.7, 94.0, 94.3\}\%$ missing RSSIs for Wanda. We test the performance of differentiators under such high missing rates of RSSIs in the input radio map.

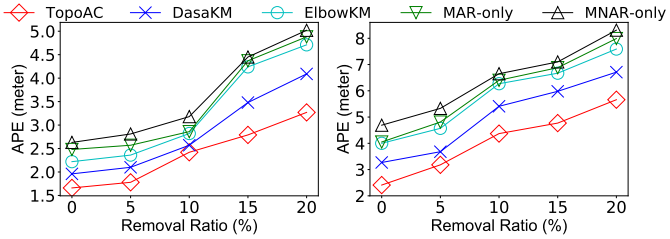
Further, we test the effect of the fraction threshold η in Algorithm 2 on the differentiators by varying it in $\{0, 0.1, 0.2, 0.3\}$. By default, we set $\alpha = 0$ and $\eta = 0.1$. In each test, we vary one parameter and set the others to default.

Metrics. We measure the **average positioning error** (**APE**) between all estimated locations and their ground-truth locations. Note that we do not evaluate the differentiators using the DA metric. As DA is utilized in DasaKM (and not in the other methods), this could lead to an unfair comparison.

2) Results: **Effect of Removal Ratio** α. The APE results for different removal ratios are reported in Fig. 12. All methods are affected negatively by a larger α since more observed values are removed, which reduces the final positioning accuracy. Also, the three differentiator methods consistently outperform MAR-only and MNAR-only, showing the significance of differentiation. By distinguishing MNARs and MARs and imputing them differently, these methods reduce bias that exists in MAR-only and MNAR-only methods which treat all the missing RSSIs as the same kind in the imputation. MAR-only always outperforms MNAR-only—MNAR-only naively fills in all missing RSSI with -100 dBm, while MAR-only employs the imputer to approach the true values of missing RSSIs. Regarding the differentiators, ElbowKM performs worse than DasaKM and TopoAC, and its performance degrades more rapidly. Due to its inferiority, ElbowKM is excluded from evaluations of data imputers in Section V-C.

Compared to ElbowKM, DasaKM improves the positioning accuracy by more than 0.3 m in Kaide and by more than 0.4 m in Wanda. In practice, a positioning error of 0.3 m is likely to localize a user mistakenly to another room behind a wall, thus impairing the quality of downstream services such as indoor navigation and contact tracing. Overall, the proposed differentiation accuracy (DA) is shown to be effective and necessary for the K-means based missing RSSI differentiation.

Compared to DasaKM, TopoAC requires no brute-force K search or DA measurement, while achieving better APE in all tests. This shows the effectiveness of utilizing indoor topology in clustering. However, in case topological information is unavailable, the proposed DasaKM provides a useful, alternative method for missing RSSI differentiation.

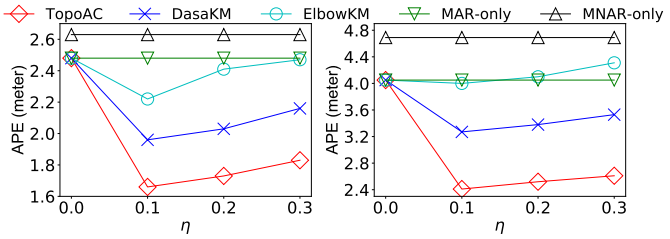


(a) Kaide (b) Wanda

Fig. 12: The removal ratio α vs. APE.

Effect of Fraction Threshold η. The APE results are reported in Fig. 13. Threshold η imposes requirements on the identification of MARs within a cluster. The setting $\eta = 0$ means that all differentiators consider missing RSSIs as MARs despite the clustering results; thus, they result in the same APE as MAR-only. As η increases, the requirement for a missing RSSI to be judged as a MAR becomes stricter, and thus previously incorrectly identified MARs are identified as MNARs, which initially improves the APE for differentiators (cf. $\eta = 0.1$). However, as η increases further, more MARs are mistakenly recognized as MNARs, which leads to worse APE results. This can be highlighted by ElbowKM (e.g., $\eta = 0.3$ in Wanda), where the APE is even higher than that of MAR-only. In contrast, DasaKM and TopoAC are more stable to the increasing η thanks to their effectiveness in clustering similar AP profiles against identification errors. If η goes up to 1, all three differentiators would have the same APE as MNAR-only, as all missing RSSIs are regarded as MNARs. Overall, TopoAC outperforms the others, and $\eta = 0.1$ is the best threshold for all differentiators.

Distribution of Differentiated Results. Based on TopoAC's differentiated results in the default setting, MARs account for 10.12% of all missing RSSIs in Kaide and 7.06% in Wanda. Note that this is only an estimated result—as mentioned earlier, the real distribution is unknown.



(a) Kaide (b) Wanda

Fig. 13: The threshold η vs. APE.

[6]We omit the inferior results of other clustering methods like DBSCAN.

C. Evaluation of Data Imputers

1) Setting: **Methods**. We implement two BiSIM variants: (1) D-BiSIM combines DasaKM and BiSIM and (2) T-BiSIM combines TopoAC and BiSIM. In addition, we include the following data imputers: (3) **Case Deletion** (CD) [32] removes all radio map records with `null` RPs and uses -100 dBm for each missing RSSI; (4) **Linear Interpolation** (LI) [37] differs from CD in that it interpolates the missing RPs linearly based on their previously and subsequently observed RPs along a path. (5) **Semi-supervised Learning** (SL) [49] replaces RP interpolation in LI by a semi-supervised model that utilizes records with observed RPs as samples for iterative inferencing of missing RPs; (6) **Multiple Imputation by Chained Equation** (MICE) [6] iteratively fills-in missing values of a column with other columns filled with their mean values by default; (7) **Matrix Factorization** (MF) [25] fills-in missing values in the radio map based on matrix completion; (8) **Bidirectional Recurrent Imputation for Time Series** (BRITS) [11] captures time-series data dependencies based on an RNN for imputing `null` RSSIs with known RPs and uses the LI[7] strategy to impute `null` RPs; (9) **Semi-Supervised Generative Adversarial Network** (SSGAN) [44] is the state-of-the-art GAN model for multivariate time series imputation.

Note that MICE, MF, and BRITS can also use MAR results obtained by either DasaKM or TopoAC. These imputation methods achieve better performance when using results from TopoAC. Only such results are reported due to the page limit. We divide all imputers into three categories: (1) CD, LI, and SL are *traditional imputers* used in fingerprinting based indoor positioning [32], [37], [49]; (2) MICE and MF are *autocorrelation based imputers* that exploit the autocorrelation of radio map records; and (3) BRITS, SSGAN, and *-BiSIM are *neural network imputers* that learn and utilize sequential data dependencies.

Implementation. For BRITS, SSGAN and *-BiSIM, we set the learning rate to 0.001, the batch size to 32, and the training epochs to 500. The latent vector lengths in encoder/decoder are set to 64. The length T of an input feature sequence is tuned optimally to 5. Longer sequences are sliced before encoding and assembled after decoding. The Adam optimizer is used; all neural networks are tuned to optimal for evaluations.

Parameters. We study how data imputers are affected by the sparsity of radio map. We introduce a *removal ratio* $\beta \in \{0, 10, 20, 30, 40, 50\}\%$—the fraction β of RSSIs (or RPs) are randomly removed in the original radio map. The removed values serve as the ground-truth for measuring the imputation errors (metrics to be given below). Here, β carries a different meaning from the one (i.e., α) used in Section V-B: the removal in this section is conducted after filling in all MNARs with -100 dBm. In addition, we scale the original RP density from 60% to 100% such that we only keep $\{60, \ldots, 100\}\%$ of RPs in the raw walking survey record table.

[7]BRITS cannot impute RSSIs and RPs jointly. The BRITS variants with CD and SL to missing RPs achieve similar performance. We omit them here.

Metrics. In addition to APE, we consider the errors of the imputed results with respect to their ground-truth. Specifically, we use the *Mean Absolute Error* (MAE) for the D-dimensional fingerprints and the *Euclidean Distance* for the 2-dimensional RPs. In the subsequent reporting, we highlight the **best** and second-best imputation errors in each group of experiments.

2) Results: **Accuracy Comparison**. We employ three location estimation algorithms: *K*NN [57], W*K*NN [19], and random forest (RF) [28]. Referring to Table VI, on both venues, *-BiSIM imputers always clearly outperform the competitors across different location estimation algorithms. This shows that the BiSIM data imputer contributes greatly to improving the indoor positioning accuracy.

In addition, T-BiSIM performs better than D-BiSIM, which shows the superiority of TopoAC. Both BRITS and SSGAN perform poorer than *-BiSIM as they fail to capture the dependencies between fingerprints and RPs, which are handled by the encoder-decoder in *-BiSIM. Overall, neural network imputers perform much better than traditional imputers and autocorrelation based imputers. The poor performance of autocorrelation based imputers is attributed to their inability to deal with heterogeneous radio map records.

Comparing the three location estimation algorithms, W*K*NN performs best in most cases. In subsequent experiments, we thus use W*K*NN for location estimation.

Imputation Time Cost Comparison. The total time costs to impute the radio map are given in Table VII. Traditional imputers, LI and SL, take much less time due to their simplicity. MICE and MF involve iterative processes on matrices and thus take more time. MF is the most time-consuming imputer as the high data sparsity of matrices makes it hard for MF to converge. Next, BRITS and *-BiSIM take time cost comparable to MICE and MF, but achieve much higher accuracy than all other models (cf. Table VI). SSGAN is the slowest among neural network-based imputers as its GAN model converges slowly [43]. The most accurate imputer, T-BiSIM, takes two minutes more than BRITS in imputation, while achieving an APE improvement of 1 m on both venues. Considering that imputation is an offline procedure, employing T-BiSIM is the most cost-effective.

Effect of Removal Ratio β. We consider the imputation of RSSIs and RPs, respectively. Referring to Fig. 14, when more RSSIs are removed from the radio map (due to a higher removal ratio), each method's MAE increases, as more missing values have to be imputed. Still, T-BiSIM and D-BiSIM perform the best and second best in all tests, respectively, and their performance is affected the least by an increasing β. The MAE of MICE and MF increase rapidly as their captured autocorrelation becomes less reliable when more RSSIs are removed. We disregard all traditional imputers from the RSSI imputation comparison as they fill in -100 dBm by default.

Referring to Fig. 15, for all imputers, the Euclidean distance error on RPs increases when more RPs are removed before imputation. Still, *-BiSIM is the best. When 50% of RPs are removed, T-BiSIM retains a distance of 2.59 (4.16) meters in

TABLE VI: Overall APE Comparison (unit: meter)

location estimation alg.	Kaide									Wanda								
	CD	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM	CD	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM
KNN	6.79	5.76	6.83	15.37	15.58	2.99	2.26	1.98	**1.78**	12.73	9.96	8.63	25.13	28.23	5.14	4.62	3.41	**2.43**
WKNN	6.64	5.76	7.10	15.37	15.65	3.07	2.23	1.96	**1.66**	12.52	9.95	8.45	27.91	28.35	4.78	3.47	3.27	**2.41**
RF	7.23	5.57	7.35	15.00	15.36	5.07	4.49	2.93	**2.70**	11.28	9.25	9.03	26.81	27.64	18.52	8.02	3.44	**3.10**

TABLE VII: Data Imputation Time Cost (unit: minute)

	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM
Kaide	1.35	2.41	12.06	29.89	13.84	21.41	12.87	15.10
Wanda	2.38	5.50	22.64	67.02	23.13	33.43	22.74	25.43

TABLE VIII: APE on Bluetooth Data (unit: meter)

	CD	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM
KNN	22.65	17.99	20.42	57.41	19.57	7.52	6.67	6.28	**5.95**
WKNN	22.76	16.14	18.7	57.27	19.68	7.33	6.74	6.24	**5.86**
RF	23.21	17.69	20.7	63.37	20.36	9.49	8.31	7.13	**6.29**

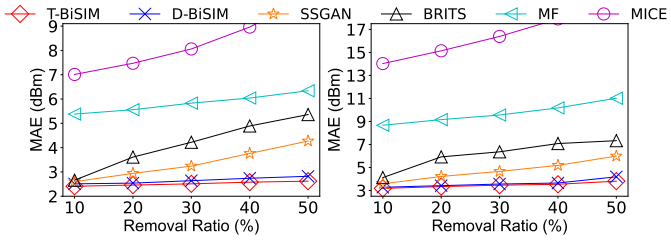
Kaide (Wanda), so it is robust to RP data sparsity. We omit CD, BRITS, and SSGAN for not involving RP imputation.

Effect of RP Density. Referring to Fig. 16, as fewer RPs are removed during walking surveying, APE improves for T-BiSIM as more RP information is available for the differentiator and the imputer. In particular, the differentiator TopoAC benefits from more RPs that results in better clustering of AP profiles, while the imputer BiSIM captures the temporal dependencies better if radio map records are denser. Also, we observe that Kaide constantly achieves better APE than Wanda. We believe this is because Kaide features denser RPs.

Ablation Study (Attention). We compare the T-BiSIM variants with (1) our adapted Bahdanau attention (Section IV-C), (2) traditional Bahdanau attention, and (3) no attention. Referring to Fig. 17, the variant without attention performs worst, showing the effectiveness of adding an attention unit in the encoder-decoder architecture. Moreover, our adapted Bahdanau attention outperforms the traditional Bahdanau attention on both venues. This is because the adapted attention design focuses on the observed part of the input features and generates more accurate weights for imputation.

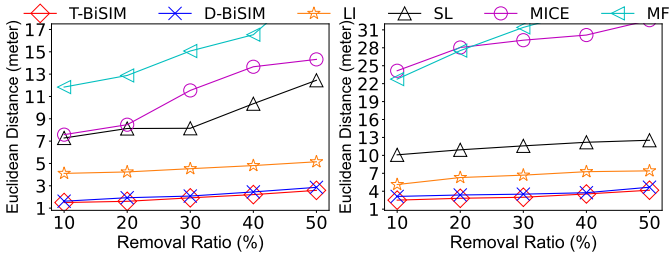
Ablation Study (Time-lag). Recall that Section IV-B introduces a time-lag mechanism into BiSIM. We compare the T-BiSIM variants with (1) time-lag employed in encoders (fingerprint part) only (our design), (2) time-lag employed in decoders (RP part) only, (3) time-lag employed in both encoders and decoders, and (4) no time-lag employed. Fig. 18 shows that our design with time-lag fingerprint vectors performs best and the variant without time-lag yields the highest APE. Interestingly, using time-lag vectors in both encoders and decoders degrades the performance. The possible reason is that the extra time-lag mechanism applied to decoders complicates the model and reduces its generalizability.

Generalizability. We conduct additional experiments with Bluetooth fingerprinting data in a third venue (i.e., Longhu) to study the generalizability of our proposals. The APE results for the Bluetooth dataset from Longhu are presented



(a) Kaide (b) Wanda

Fig. 14: The removal ratio β vs. MAE.



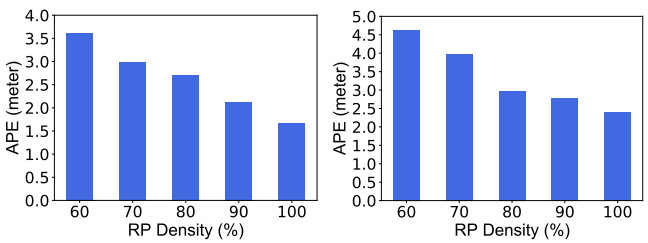
(a) Kaide (b) Wanda

Fig. 15: The removal ratio β vs. Euclidean distance.

in Table VIII. We see that *-BiSIM continues to outperform the other data imputation methods with a significant advantage, indicating that the proposed imputation framework is effective in Bluetooth fingerprinting scenarios [27] and has the potential for applications across diverse indoor positioning systems.

VI. Related Work

Radio Map Completion. Traditional positioning methods [18], [21], [23] simply replace `null` RSSIs in fingerprints with the minimum value of -100 dBm. However, this adds errors to a radio map as missing RSSIs may be caused by random events (e.g., temporarily blocked signal transmission) and their actual values are not `null` or -100 dBm. Next, existing studies handle missing RPs based on a simple deletion of corresponding pairs [32], linear interpolation with contextual RPs [37], or semi-supervised learning using records with observed RPs [49]. A major issue of linear interpolation and



(a) Kaide (b) Wanda

Fig. 16: The RP density vs. APE.

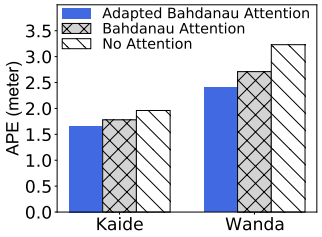


Fig. 17: Attention vs. APE.

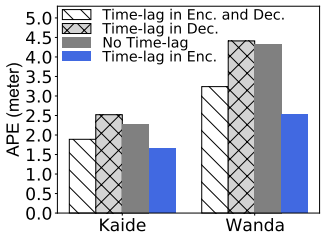


Fig. 18: Time-lag vs. APE.

TABLE IX: Neural Networks for Time-series Imputation

Models	Features	Labels	Structure	Time-lag	Attention
BiSIM	Imputed	Imputed	Seq2Seq	✓	✓
GRUD [13]	Imputed	-	RNN	✓	-
MRNN [56]	Imputed	-	RNN	✓	-
BRITS [11]	Imputed	-	RNN	✓	-
DeepMVI [10]	Imputed	-	Transformer	-	✓
GRIL [17]	Imputed	-	GRU-GNN	-	-
GRU-GAN [41]	Imputed	-	GRU-GAN	✓	-
NAOMI [40]	Imputed	-	GRU-GAN	-	-
E2GAN [42]	Imputed	-	GRU-GAN	✓	-
SSGAN [44]	Imputed	-	RNN-GAN	✓	-

semi-supervised learning is that a radio map itself is sparse. Differently, our solution differentiates MARs and MNARs and employs a sequential neural network to impute missing RSSIs and RPs jointly based on temporal dependencies of radio map records and correlations between fingerprints and RPs.

Missing Data Imputation. Straightforward zero and mean filling approaches usually yield low accuracy. Autocorrelation-based imputation methods such as MICE [6] and MF [25], [35] focus on homogeneous data records and do not fit in our setting where each record consists of a signal vector and a location. Moreover, these methods do not contend well with high data sparsity. In addition, there exist deep learning studies for imputating multi-variate time series data [10], [11], [13], [17], [36], [44], [56]. For instance, Che et al. [13] incorporate masking and time-lag mechanisms into a vanilla GRU and impute `nulls` based on a weighted combination of the last observation and a global mean. Relaxing smooth assumptions [13], Cao et al. [11] propose BRITS, a bidirectional RNN that regards missing values as trainable variables and imputes them directly by backpropagation of loss computed on observed data. Moreover, generative adversarial networks (GANs) [40]–[42], [44] have been used to learn the overall distribution of a time-series dataset to impute missing values. Table IX compares these works.

Existing neural network approaches do not apply to our problem setting directly. First, existing models impute missing values in feature sequences only, while our problem needs to handle missing values in both feature sequences (missing RSSIs) and label sequences (missing RPs). Second, existing models either disregard labels [10], [13], [17], [56] or assume a many-to-one setting where a time series corresponds to a single label [10], [11], [44], while our problem is a many-to-many setting, where each fingerprint is associated with one RP. Third, GAN-based models assume that all `nulls` are MARs, while our missing RSSIs form a mix of MARs and MNARs. All these key differences call for a way to differentiate types of missing RSSIs and means of handling missing values in both features and labels jointly.

Differentiating MARs and MNARs is beneficial to missing data imputation. Studies [39], [47], [50] point out that the design of differentiation methods requires domain knowledge, as data characteristics are tied closely to the specific application. Some studies focus on differentiation methods for specific domains, e.g., longitudinal clinical trials [30] and answer quality in surveys [15], but such studies are inapplicable in our data setting. To the best of our knowledge, we are the first to study the differentiation of missing RSSI values.

Indoor Positioning Data Cleansing. Some studies [7], [8], [14], [20], [29] use sensor deployment knowledge and time-series dependencies to repair missing readings caused by sensor failures. Missing values in these studies are identifiers of sensors such as RFID readers. In addition, Lin et al. [38] propose a semi-supervised scheme to detect and impute missing AP identifiers in raw Wi-Fi connectivity data. Our work differs from these works in that we aim to impute numerical values instead of AP or RFID reader identifiers. Also, our radio map imputation targets fingerprinting-based localization at a point level rather than at a regional level. Sun et al. [52] propose a sequential alignment-and-matching method to complete missing RSSI values and an AP distribution-based mapping method to amend missing and false location labels. That work assumes that all `nulls` are MNARs and location labels are at the room level. Therefore, it is not applicable to our problem.

VII. Conclusion

We impute missing *received signal strength indicator values* (RSSIs) and *reference points* (RPs) in radio maps by designing a framework encompassing a missing RSSI differentiator and a data imputer. The clustering-based differentiator determines missing at random (MAR) and missing not at random (MNAR), whereas the model-based imputer leverages temporal dependencies and correlations in data to impute MARs and missing RPs. Extensive experimental studies demonstrate that our proposed framework clearly outperforms existing alternatives in terms of positioning and imputation accuracy.

In future work, it is of interest to design more efficient methods that enable online imputation of fingerprints. Also, it is relevant to integrate our separate differentiator and imputer into a single model, thus enabling end-to-end support of imputation processes.

Acknowledgements

This work is an extended version of the paper entitled "Data Imputation for Sparse Radio Maps in Indoor Positioning" published at ICDE 2023. The work was funded by Independent Research Fund Denmark (No. 8022-00366B). Huan Li's work was supported by Aalborg University and EU MSCA programme (No. 882232). The work also benefited from discussions in the context of DIREC, a centre funded by the Innovation Fund Denmark.

REFERENCES

[1] https://www.researchandmarkets.com/reports/4765038/indoor-positioning-and-navigation-global-market.

[2] https://en.wikipedia.org/wiki/DBm.

[3] https://github.com/XLI-2020/BiSIM.

[4] https://www.kaggle.com/c/indoor-location-navigation.

[5] Douglas G Altman and J Martin Bland. Diagnostic tests. 1: Sensitivity and specificity. *BMJ*, 308(6943):1552, 1994.

[6] Melissa J Azur, Elizabeth A Stuart, Constantine Frangakis, and Philip J Leaf. Multiple imputation by chained equations: What is it and how does it work? *Int J Methods Psychiatr Res*, 20(1):40–49, 2011.

[7] Asif Iqbal Baba, Manfred Jaeger, Hua Lu, Torben Bach Pedersen, Wei-Shinn Ku, and Xike Xie. Learning-based cleansing for indoor RFID data. In *SIGMOD*, pages 925–936, 2016.

[8] Asif Iqbal Baba, Hua Lu, Xike Xie, and Torben Bach Pedersen. Spatiotemporal data cleansing for indoor RFID tracking data. In *MDM*, pages 187–196, 2013.

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[10] Parikshit Bansal, Prathamesh Deshpande, and Sunita Sarawagi. Missing value imputation on multidimensional time series. *Proc. VLDB Endow.*, 14(11):2533–2545, 2021.

[11] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Yitan Li, and Lei Li. BRITS: Bidirectional recurrent imputation for time series. In *NeurIPS*, pages 6776–6786, 2018.

[12] Kyungmin Chang and Dongsoo Han. Crowdsourcing-based radio map update automation for Wi-Fi positioning systems. In *Geocrowd*, pages 24–31, 2014.

[13] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Sci. Rep.*, 8(1):1–12, 2018.

[14] Haiquan Chen, Wei-Shinn Ku, Haixun Wang, and Min-Te Sun. Leveraging spatio-temporal redundancy for RFID data cleansing. In *SIGMOD*, pages 51–62, 2010.

[15] Pu-Shih Daniel Chen. Finding quality responses: The problem of low-quality survey responses and its impact on accountability measures. *Research in Higher Education*, 52(7):659–674, 2011.

[16] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *SSST-8*, pages 103–111, 2014.

[17] Andrea Cini, Ivan Marisca, and Cesare Alippi. Multivariate time series imputation by graph neural networks. *arXiv preprint arXiv:2108.00298*, 2021.

[18] Kai Dong, Zhen Ling, Xiangyu Xia, Haibo Ye, Wenjia Wu, and Ming Yang. Dealing with insufficient location fingerprints in Wi-Fi based indoor location fingerprinting. *Wirel. Commun. Mob. Comput.*, 2017.

[19] Shih-Hau Fang, Tsung-Nan Lin, and Po-Chiang Lin. Location fingerprinting in a decorrelated space. *IEEE Trans Knowl Data Eng.*, 20(5):685–691, 2008.

[20] Bettina Fazzinga, Sergio Flesca, Filippo Furfaro, and Francesco Parisi. Exploiting integrity constraints for cleaning trajectories of RFID-monitored objects. *ACM Trans. Database Syst.*, 41(4):1–52, 2016.

[21] Firdaus Firdaus, Noor Azurati Ahmad, and Shamsul Sahibuddin. Accurate indoor-positioning model based on people effect and ray-tracing propagation. *Sensors*, 19(24):5546, 2019.

[22] Vicente García, Ramon A Mollineda, and J Salvador Sánchez. Theoretical analysis of a performance measure for imbalanced data. In *ICPR*, pages 617–620, 2010.

[23] Rafał Górak and Marcin Luckner. Automatic detection of missing access points in indoor positioning system. *Sensors*, 18(11):3595, 2018.

[24] Dongsoo Han, Sangjae Lee, and Sunghoon Kim. KAILOS: KAIST indoor locating system. In *IPIN*, pages 615–619, 2014.

[25] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: Data mining, inference, and prediction*, volume 2. 2009.

[26] Suining He and S-H Gary Chan. Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons. *IEEE Commun. Surv. Tutor.*, 18(1):466–490, 2015.

[27] Héctor José Pérez Iglesias, Valentín Barral, and Carlos J Escudero. Indoor person localization system through rssi bluetooth fingerprinting. In *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 40–43. IEEE, 2012.

[28] Esrafil Jedari, Zheng Wu, Rashid Rashidzadeh, and Mehrdad Saif. Wi-Fi based indoor location positioning employing random forest classifier. In *IPIN*, pages 1–5, 2015.

[29] Shawn R Jeffery, Minos Garofalakis, and Michael J Franklin. Adaptive cleaning for RFID data streams. *Proc. VLDB Endow.*, 6:163–174, 2006.

[30] Man Jin. A hybrid return to baseline imputation method to incorporate mar and mnar dropout missingness. *Contemporary Clinical Trials*, 120:106859, 2022.

[31] Suk Hoon Jung, Byeong-Cheol Moon, and Dongsoo Han. Performance evaluation of radio map construction methods for Wi-Fi positioning systems. *IEEE Trans. Intell. Transp. Syst.*, 18(4):880–889, 2016.

[32] Jiří Kaiser. Dealing with missing values in data. *J. Syst. Integr.*, 5(1), 2014.

[33] Saeed Kargar, Heiner Litz, and Faisal Nawab. Predict and write: Using k-means clustering to extend the lifetime of NVM storage. In *ICDE*, pages 768–779, 2021.

[34] Ali Khalajmehrabadi, Nikolaos Gatsis, and David Akopian. Modern WLAN fingerprinting indoor positioning methods and deployment challenges. *IEEE Commun. Surv. Tutor.*, 19(3):1974–2002, 2017.

[35] Mourad Khayati, Michael Böhlen, and Johann Gamper. Memory-efficient centroid decomposition for long time series. In *ICDE*, pages 100–111, 2014.

[36] Mourad Khayati, Alberto Lerner, Zakhar Tymchenko, and Philippe Cudré-Mauroux. Mind the gap: An experimental evaluation of imputation of missing values techniques in time series. In *Proc. VLDB Endow.*, volume 13, pages 768–782, 2020.

[37] Chenhe Li, Qiang Xu, Zhe Gong, and Rong Zheng. TuRF: Fast data collection for fingerprint-based indoor localization. In *IPIN*, pages 1–8, 2017.

[38] Yiming Lin, Daokun Jiang, Roberto Yus, Georgios Bouloukakis, Andrew Chio, Sharad Mehrotra, and Nalini Venkatasubramanian. Locater: Cleaning WiFi connectivity datasets for semantic localization. *Proc. VLDB Endow.*, 14(3):329–341, 2020.

[39] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

[40] Yukai Liu, Rose Yu, Stephan Zheng, Eric Zhan, and Yisong Yue. NAOMI: Non-autoregressive multiresolution sequence imputation. *Adv Neural Inf Process Syst.*, 32:11238–11248, 2019.

[41] Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, et al. Multivariate time series imputation with generative adversarial networks. *Adv Neural Inf Process Syst.*, 31:1596–1607, 2018.

[42] Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E2GAN: End-to-end generative adversarial network for multivariate time series imputation. In *IJCAI*, pages 3094–3100, 2019.

[43] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

[44] Xiaoye Miao, Yangyang Wu, Jun Wang, Yunjun Gao, Xudong Mao, and Jianwei Yin. Generative semi-supervised learning for multivariate time series imputation. In *AAAI*, volume 35, pages 8983–8991, 2021.

[45] Teemu Pulkkinen, Teemu Roos, and Petri Myllymäki. Semi-supervised learning for WLAN positioning. In *ICANN*, pages 355–362, 2011.

[46] Darwin Quezada-Gaibor, Lucie Klus, Joaquín Torres-Sospedra, Elena Simona Lohan, Jari Nurmi, Carlos Granell, and Joaquín Huerta. Data cleansing for indoor positioning Wi-Fi fingerprinting datasets. In *MDM*, pages 367–371, 2022.

[47] Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[48] Sebastian Sadowski and Petros Spachos. RSSI-based indoor localization with the Internet of Things. *IEEE Access*, 6:30149–30161, 2018.

[49] Sameh Sorour, Yves Lostanlen, Shahrokh Valaee, and Khaqan Majeed. Joint indoor localization and radio map construction with limited deployment load. *IEEE Trans. Mobile Comput.*, 14(5):1031–1043, 2014.

[50] William R Sterner. What is missing in counseling research? reporting missing data. *Journal of Counseling & Development*, 89(1):56–62, 2011.

[51] Haotai Sun, Xiaodong Zhu, Yuanning Liu, and Wentao Liu. Wifi based fingerprinting positioning based on seq2seq model. *Sensors*, 20(13):3767, 2020.

[52] Jing Sun, Bin Wang, Xiaoxu Song, and Xiaochun Yang. Data cleaning for indoor crowdsourced RSSI sequences. In *APWeb-WAIM*, pages 267–275, 2021.

[53] Pengfei Wang and Yufeng Luo. Research on wifi indoor location algorithm based on rssi ranging. In *ICISCE*, pages 1694–1698, 2017.

[54] Chenshu Wu, Zheng Yang, and Yunhao Liu. Smartphones based crowdsourcing for indoor localization. *IEEE Trans. Mobile Comput.*, 14(2):444–457, 2014.

[55] Han Xu, Zheng Yang, Zimu Zhou, Longfei Shangguan, Ke Yi, and Yunhao Liu. Enhancing WiFi-based localization with visual clues. In *UbiComp*, pages 963–974, 2015.

[56] Jinsung Yoon, William R Zame, and Mihaela van der Schaar. Multi-directional recurrent neural networks: A novel method for estimating missing data. In *ICML Time Series Workshop*, 2017.

[57] Demetrios Zeinalipour-Yazti and Christos Laoudias. The anatomy of the anyplace indoor navigation service. *SIGSPATIAL Special*, 9(2):3–10, 2017.