Abstract—The interest for RF-based indoor localization, and in particular for WiFi RSSI-based fingerprinting, is growing at a rapid pace. This is despite the existence of a trade-off between the accuracy of location estimation and the density of a laborious and time consuming survey for collecting training fingerprints. A generally accepted concept of increasing the density of a training dataset, without an increase in the amount of physical labor and time needed for surveying an environment for additional fingerprints, is to leverage a propagation model for the generation of virtual training fingerprints. This process, however, burdens the user with an overhead in terms of implementing a propagation model, defining locations of virtual training fingerprints, generating virtual fingerprints, and storing the generated fingerprints in a training database. To address this issue, we propose the Enriched Training Database (ETD), a web-service that enables storage and management of training fingerprints, with an additional “enriching” functionality. The user can leverage the enriching functionality to automatically generate virtual training fingerprints based on propagation modeling in the virtual training points. We further propose a novel method for defining locations of virtual training fingerprints based on modified Voronoi diagrams, which removes the burden of defining virtual training points manually and which automatically “covers” the regions without sufficient density of training fingerprints. The evaluation in our testbed shows that the use of automated generation of virtual training fingerprints in the ETD results in more than 25% increase in point accuracy and 15% in room-level accuracy of fingerprinting.

I. INTRODUCTION

In recent years, we are witnessing a rapid growth in the interest for indoor localization, with one indication being the large amount of works generally targeting this topic (e.g. [1]–[4]). Many of those efforts indicate that Radio Frequency (RF)-based localization, and particularly Wireless Fidelity (WiFi) Received Signal Strength Indicator (RSSI)-based fingerprinting, is one of the most promising candidates for an ubiquitous localization service indoors. The main reasons are the omnipresence of WiFi infrastructures, the possibility of “piggybacking” on already available WiFi signals, and the independence of fingerprinting approaches on the generally unreliable power-to-distance relationship [5].

For enabling a WiFi fingerprinting-based indoor localization service, the service provider is required to generate a training dataset, i.e. to survey an environment at predefined locations. The users’ generated fingerprints are then compared with fingerprints from the generated training dataset, and based on their similarities location estimates are reported. The required training survey is time and labor consuming, which is further accenteduated by the fact that the same survey, due to the changes in an environment or due to the collected measurements getting staled, has to be repeated after a certain period of time. The density of collected fingerprints in a training survey has a direct relation with the accuracy of fingerprinting [6], i.e. a higher density of training fingerprints yields a better accuracy of location estimation until a certain density threshold. A well-known concept of increasing the density of a training dataset, without an increase in the amount of time and labor required for surveying an environment for additional fingerprints, is to leverage a propagation model to create additional fingerprints at locations not surveyed, i.e. to generate virtual training fingerprints. This process, however, requires additional work in terms of defining locations of virtual training fingerprints, implementing a propagation model, generating virtual training fingerprints and storing them in a training database.

The work presented in this paper aims on simplifying the process of storage and management of training fingerprints, as well as the generation and storage of virtual training fingerprints in a training database. To this end, we propose the Enriched Training Database (ETD)\(^1\), a web-service that enables storage and management of WiFi RSSI-based training fingerprints. The adjective “enriched” indicates that the ETD provides a functionality that enriches its main purpose and which can be leveraged for generating virtual training fingerprints. Given an original set of training fingerprints of a certain density is stored in the ETD, the enriching functionality can firstly be leveraged for defining virtual training points. Secondly, based on a propagation model, the power levels from different WiFi Access Points (APs) at the defined virtual training points can be modeled and virtual training fingerprints can be generated and stored in the ETD. Using this functionality results in the improved accuracy of fingerprinting with the same density of physically collected training fingerprints, i.e. without increase in the efforts of performing a training survey.

The ETD features a modular design, where different types of fingerprinting algorithms can be supported. Furthermore, the modular design enables different methods to be implemented for both defining virtual training points and modeling of WiFi signal strengths at the defined virtual training points. In the current implementation, we support our claim of modularity by enabling two fingerprinting algorithms to work with the ETD. Moreover, the definition of virtual training points can

\(^1\)A prototype of the ETD is available at https://github.com/flemic/ETD.
be performed based on Voronoi diagrams or based on the user’s input. As for the propagation modeling, the current implementation allows the usage of Inverse Distance Weighted Interpolation (IDWI) and Multi-Wall (MW) propagation models, although, due to its modular design, additional models can be easily introduced.

The evaluation of two WiFi RSSI-based fingerprinting algorithms in our testbed environment shows that, for the best case scenario, leveraging the ETD functionality of generating virtual training fingerprints results in the accuracy increase of more than 25%, while for the room-level accuracy the increase is 15%. These improvements are archived without additional time and labor costs of collecting additional training fingerprints and with a practically negligible increase in the processing time of the evaluated algorithms.

The rest of this paper is structured as follows. In Section II, we provide an overview of the related work. Section III describes the ETD design and its envisioned usage, while Section IV provides details of the building blocks in different ETD modules. Section V overviews the implementation of the ETD. In Section VI and Section VII, we overview the experimental setup and the evaluation results, respectively. Finally, we conclude the work and outline directions for further improvements in Section VIII.

II. RELATED WORK

Various approaches exist for generating virtual training fingerprints based on propagation modeling, which is shown to be beneficial for improving the accuracy of fingerprinting. The authors in [6] investigate the influence of virtual training fingerprints generated using various propagation models on the accuracy of fingerprinting. The approach in [7] aims on generating virtual training fingerprints by modeling the WiFi signal power levels based on kriging, while the one in [8] bases the creation of virtual training fingerprints on the support vector regression. The authors in [9] propose a novel learning algorithm that reduces the calibration efforts of fingerprinting by creating virtual training fingerprints based on linear interpolation. Similarly, the approach in [10] aims on generating virtual training fingerprints, i.e. increasing the density of a training database, based on discontinuity preserving smoothing. Finally, in [11] the authors aim on generating virtual training fingerprints based on the higher-order Voronoi tessellation.

Contrary to the previously mentioned works, in which the focus is mainly on proposing novel methods for generating virtual training fingerprints, in this work we focus on the design of a training database with an additional enriching functionality. The goal of the ETD is to automate the application of different methods for generation of virtual fingerprints by providing a common platform. Similar tools have been proposed for different research domains, for example the ArcGIS Spatial Analyst [12] for the interpolation and modeling in spatial analysis. By using the ETD, without additional implementation burden, the user is able to generate and store additional virtual training fingerprints. While the other approaches focus the generation of virtual training fingerprints in locations explicitly provided by the user, we propose a novel method for automatically defining virtual training points. The method is based on Voronoi diagrams that are leveraged for defining virtual training points based on the locations of original training fingerprints in a way that “covers” the regions with small density of fingerprints, thus removing the burden of manually defining them.

III. SYSTEM OVERVIEW

The usual procedure of surveying an environment for WiFi RSSI fingerprinting-based indoor localization service is the following. The user firstly defines a set of training points, usually in a fairly regular grid fashion, although that is not a general requirement for WiFi fingerprinting. At each of the defined points, the user samples the WiFi environment and this sample, with a corresponding location coordinate, is stored in a training database. The procedure is repeated until WiFi RSSI measurements from all defined points are collected and stored in the training database. This usual procedure can be performed using the ETD, as depicted in Figure 1. Based on the requirement of algorithms to be used, the collected RSSI measurements that are stored in the ETD can be processed to generate training fingerprints. In the following step the generated training fingerprints can be stored in the ETD. In the ETD, the training fingerprints can be stored separately from the RSSI measurements for enabling an easy replacement of the fingerprinting algorithm by generating a new set of training fingerprints from the originally stored RSSI measurements.

In order to increase the density of training fingerprints without increasing the number of physical measurements, the enriching ETD functionality allows the generation of virtual training fingerprints. The envisioned procedure of generating virtual training fingerprints is further depicted in Figure 1. Based on the locations of the original training fingerprints stored in the ETD, a set of virtual training points is defined by leveraging a specific method for virtual training points definition selected by the user. In the next step, in the defined virtual training points the RSSI values from different WiFi APs are modeled based on a propagation model selected by the user. Finally, the generated virtual fingerprints are stored in the ETD together with the original set of training fingerprints.

IV. BUILDING BLOCKS

A modular design of the enriching ETD functionality allows easy replacement of the fingerprinting algorithm, as well as the usage of different methods for defining virtual training points and generating virtual training fingerprints, depending on the user’s preferences. Furthermore, additional methods can be implemented for both the definition of virtual training points and for the generation of virtual training fingerprints. In this section, we overview the currently supported finger printing algorithms and the methods implemented in modules for defining virtual training points and for generating virtual training fingerprints based on propagation modeling.
A. Defining Virtual Training Points

The current implementation of the ETD features two methods for defining virtual training points.

1) User’s Input: This method allows the user to define coordinates of intended virtual training points as an input to the following ETD processing block. Defining virtual training points based on user’s input is a traditional method adopted in most of the literature, which motivated us to provide it as a method in the current ETD implementation. However, the drawback of this method is that it puts a burden of defining virtual training points on the user. Also, by leveraging this method, it is possible to (by mistake) define virtual points that are close to one another or to some of the original training fingerprints, which only results in increased latency of providing location estimates, without benefiting the accuracy of fingerprinting.

2) Modified Voronoi Diagrams: This method is designed to allow defining virtual training points based on modified Voronoi diagrams [13]. For a given set of training points, the Voronoi diagrams create regions in the environment, where each region consists of all points that have the smallest distance from one of the original training points. An example is given in Figure 2. In Figure 2a), the locations of the original training points are depicted with red dots. Figure 2b), in addition to the original set of training points, depicts the Voronoi decomposition of the environment. As visible in the figure, the environment is partitioned in a set of Voronoi regions, and those regions intersect in points that are known as Voronoi vertices. Voronoi vertices are points equidistant to three or more of the original points, and here we use them as virtual training points. However, due to the irregularities in the original training grid, the Voronoi vertices can be relatively close to one another, as the ones inside of a blue circle depicted in Figure 2b). This would result in multiple virtual training points that are relatively close to one another, which ultimately results in increased latency of generating location estimates without benefiting the accuracy. Due to that, on top of the obtained Voronoi vertices we apply the following modification. We firstly calculate the minimum nearest neighbor distance of the original set of training points. We use the calculated distance as a metrics for detecting the Voronoi vertices that are relatively close to one another. If the distance between two or more virtual training points is less than half of the minimum nearest neighbor distance between the original training points, we consider these Voronoi vertices as relatively close to one another. In case two or more Voronoi vertices are relatively close to one another, we merge these Voronoi vertices based on the average value. Intuitively, one Voronoi vertex should be defined in the region between two or more adjacent original training points. In case more than one vertex is defined, the modification based on minimum nearest neighbor distance will detect that and merge the defined vertices into one virtual training point. Moreover, we limit the area of the Voronoi vertices to the minimum and maximum coordinates of the original training points, e.g. the Voronoi vertices inside the red circles in Figure 2b) are not included in the set of virtual training points. The final results of defining virtual training points based on the modified Voronoi diagrams is given in Figure 2c), where blue dots indicate locations of virtual training points. The benefits of this method are that it does not require users input and that it merges fingerprints that are close to one another according to the minimum nearest neighbor distance criteria.

B. Propagation Modeling

This section provides an overview of the two propagation models currently implemented as part of the ETD. More precisely, the first presented “model” is in fact an interpolation procedure, while the second one is a propagation model in its full meaning. This shows the capability of ETD in supporting both interpolation procedures and propagation models for generation of virtual training fingerprints. In the rest of the paper, we will refer to both of them as propagation models.

1) Inverse Distance Weighted Interpolation Model: The first propagation model is a simple and well-known IDWI
model [14], in which the modeled values are based on the previously collected ones. The benefit of this model lies in the fact that it only depends on the originally collected fingerprints, meaning that it does not require any additional input from the user. However, due to its simplicity, substantial errors are anticipated in the modeled power levels, which makes this model less beneficial in terms of improving the accuracy of fingerprinting. This shortcoming is specifically emphasized in case of a small number of original training fingerprints, since the inputs are in that case highly limited.

In this model, weights are given to the measurements according to the inverse of their distance to a point in which WiFi signals are to be interpolated. The equation of finding the interpolated value \( z \) at a point \( p \), based on measurements \( z_i = z(p_i) \) for \( i = 1, 2, ..., N \) (\( N \) being the number of points, \( d \) being the distance function), is given as follows:

\[
z(p) = \frac{\sum_{i=1}^{N} w_i(p) z_i}{\sum_{j=1}^{N} w_j(p)}, \text{ where } w_i(p) = \frac{1}{d(p,p_i)} \tag{1}
\]

2) Multi-Wall Model: The second propagation model is the COST 231 multi-wall and floor model for indoor radio propagation [15], with its applicability for generation of virtual training fingerprints being demonstrated in [16]. In comparison to the previous model, this model takes into account the type and number of walls, floors or obstacles in the environment, as well as the locations of transmitting WiFi devices, which increases the burden on the user that has to specify both. However, this model is anticipated to capture in a better way a propagation environment, which is more beneficial to the accuracy of fingerprinting, in comparison to the IDWI model.

The first attenuation contribution in the model is a generic and widely known one-slope term that relates the difference between transmitted and received power to the distance \( d \). Two parameters influence the attenuation in this term: the constant \( l_0 \) (the path-loss at 1 m distance and at the center frequency of 2.4 GHz) and the path-loss exponent \( \gamma \). The second attenuation contribution is the linear wall/floor/obstacle term. The number of obstacles in the direct path between transmitter and receiver is counted and for each type of obstacle an attenuation contribution is assumed. Given the model and the site-specific measurements collected in an environment, a simple least square fitting procedure can be leveraged, which allows minimization of the differences between powers \( P_m \), measured in each \( m \)-th \( (m = 1, 2, ..., M) \) training point from all used APs, and the model estimated received power \( EIRP - L(d_m) \), where \( EIRP \) denotes the effective isotropic radiated power at the transmitter. The equation is given by:

\[
\{l_c, \gamma, l_w\}_{opt} = \arg \min_{l_c, \gamma, l_w} \{ \sum_{m=0}^{M-1} (P_m - (EIRP - L(d_m)))^2 \} \tag{2}
\]

where \( L(d_m) \) contains both the attenuation from a power-distance relation and the attenuation of each wall/ floor/obstacle. Further, \( l_c \) is a constant used for optimizing the minimization of the cost in a multi-wall model, which includes the influence of the parameter \( l_0 \). Using the calculated parameters \( \{l_c, \gamma, l_w\}_{opt} \), a fingerprint of an environment, and locations of WiFi APs as inputs, the WiFi signal power levels at virtual training points can be modeled.

C. Fingerprinting Algorithms

The ETD currently enables the usage of two well-known WiFi RSSI-based fingerprinting algorithms presented in [17]. These algorithms leverage different types of fingerprints, which illustrates the capability of ETD to accommodate different types of fingerprinting algorithms.

1) Euclidean distance of averaged RSSI vectors: This fingerprinting algorithm uses the computed average value of RSSI measurements obtained from each AP used for localization. The fingerprint is an average value of the RSSI measurements obtained from each AP used for localization in both the training and the online step, where \( K \) is the length of the vector. Let \( \mu_{t,m} = [RSSI_{t,1}, ..., RSSI_{t,k}, ..., RSSI_{t,K}] \) be the vector of averaged RSSI values \( RSSI_{t,i} \) from each AP \( i \) obtained in training step at point \( m \in 1, ..., M_t \), i.e. training fingerprint. In the same manner, let \( \mu_{t,m} = [RSSI_{t,1}, ..., RSSI_{t,k}, ..., RSSI_{t,K}] \) be the vector of averaged RSSI values \( RSSI_{t,i} \) from each AP \( i \) obtained in training step at point \( m \in 1, ..., M_t \), i.e. training fingerprint.
\(\{\text{RSSI}_{r,1}, \ldots, \text{RSSI}_{r,k}, \ldots, \text{RSSI}_{r,K}\}\) be the vector of averaged RSSI values \(\text{RSSI}_{r,i}\) from each AP \(i\) obtained in the online step. The pattern matching procedure uses the Euclidean Distance (ED) between a training fingerprint at the cell \(m\) and the online fingerprint and it is given as:

\[
D_E(\mathbf{X}_{t,m}, \mathbf{X}_r) = |\mathbf{X}_{t,m} - \mathbf{X}_r|.
\]

\(\mathbf{X}_{t,m}\) and \(\mathbf{X}_r\) are fingerprint vectors in the training and the online steps, respectively. The training fingerprints with the smallest distance (also called smallest weight) are then used in the post-processing procedure. In the post-processing procedure we used the non-weighed k-Nearest Neighbors (kNN) method with the parameter \(k\) set to 3, since it is shown in [17] that this method achieves the best performance results in comparison to a large set of other evaluated post-processing methods.

2) Pompeiu-Hausdorff distance of RSSI quantiles: This fingerprinting procedure uses \(q\) quantiles of the RSSI values from each AP as fingerprints, which are calculated in two steps. First the Cumulative Distribution Function (CDF) of the RSSI measurements from each AP is computed. Second, the quantiles, i.e. RSSI values with probabilities \(k/(q-1)\), where \(k = 0, 1, \ldots, q - 1\), are calculated. The result of the quantile calculation in both training and online steps is a quantile matrix \(Q_{K,q}\), where \(K\) is the number of APs visible at the given location and \(q\) is a number of quantiles. The pattern matching procedure of this algorithm uses the Pompeiu-Hausdorff (PH) metric for capturing similarities between training fingerprints and an online one [18], as follows:

\[
D_{PH}(\mathbf{X}_{t,m}, \mathbf{X}_r) = \max_{x_{t,k} \in \mathbf{X}_{t,m}} \min_{x_{r,k} \in \mathbf{X}_r} d(x_{t,k}, x_{r,k})
\]

Here \(d(x_{t,k}, x_{r,k})\) is the Euclidean Distance measurement between elements of the online fingerprint \(\mathbf{X}_r\) and training fingerprint \(\mathbf{X}_{t,m}\) at point \(m\). The training point with the smallest Pompeiu-Hausdorff (PH) distance with the online fingerprint is reported as an estimated location. Same as for the previous algorithm, we use the 3NN method in the post-processing procedure.

V. ETD IMPLEMENTATION

In this section, we shortly overview the implementation of the ETD which enables the following features: extensibility, fast and reliable remote access, and language and platform independence. The ETD implementation is based on our previous work [19], where similar features have been selected to support data storage with the goal of experimental evaluation of RF-based indoor localization using pre-collected data traces, which removes the need of performing local experiments.

The ETD is a web service implemented in Python 2.7 using the Flask module, which provides a simple way of creating RESTful web services. The training fingerprints are stored in a MongoDB database, an open-source document database and the leading Not only SQL (NoSQL) database written in C++. A fingerprint is defined as a Protocol Buffer structure, a way of encoding structured data using an efficient and extensible binary format. The extensibility of the stored fingerprints is achieved using the Protocol Buffer for defining a fingerprint structure and MongoDB database for storing those fingerprints. This feature enables an easy storage of different types of training fingerprints, burdening the user only with the necessary modification of the Protocol Buffer message reflecting a new type of fingerprint. By using a NoSQL type of database, the ETD enables storage of any type of defined message, without a need of changing the schema and/or the database itself. The RESTful design and the implementation as a web-service enable remote access to the ETD using only HTTP requests. Protocol Buffers serialize messages into binary streams which support fast communication between the users and the ETD service. Furthermore, due to the fact that communication with the ETD service is done using HTTP requests, it is possible to manage data from different platforms, and also using different programming languages, since most of the modern languages provide libraries enabling HTTP requests.

VI. EXPERIMENTAL EVALUATION

The experimental evaluation in this work was performed by following the guidelines given in the EVARILOS Benchmarking Handbook (EBH) [20], which provides a methodology for experimental evaluation of RF-based indoor localization algorithms solution. The experimental evaluation in this work was performed by following the guidelines given in the EVARILOS Benchmarking Handbook (EBH) [20], which provides a methodology for experimental evaluation of RF-based indoor localization algorithms solution. The experimental evaluation in this work was performed by following the guidelines given in the EVARILOS Benchmarking Handbook (EBH) [20], which provides a methodology for experimental evaluation of RF-based indoor localization algorithms solution. The experimental evaluation in this work was performed by following the guidelines given in the EVARILOS Benchmarking Handbook (EBH) [20], which provides a methodology for experimental evaluation of RF-based indoor localization algorithms solution.
Two repetitions of measurements at the same locations provide additional insights in the temporal stability of the obtained results, which strengthens the reliability of our observations. While only two repetitions of the same experiment are not sufficient to provide statistical benefits, the comparability of results obtained in these repetitions excludes the possibility of a sudden change in the performance (e.g. due to interference, movements, changes in the environment), which could lead to errors in the conclusions. The collected measurements were stored in a web-based platform for streamlined experimental evaluation of RF-based indoor localization algorithms using previously collected raw data traces [19], [22]. This platform provides a simple way of reusing the same datasets for multiple evaluations, and by leveraging this functionality we were able to reuse the same set of measurements for generation of different types of fingerprints and for the evaluation of different algorithms.

In Figure 4, the accuracy of the used fingerprinting algorithms is depicted in case when the original training set is used and in case when additional virtual training fingerprints are generated based on the modified Voronoi diagrams and leveraging the two described propagation models. As presented in the figure in a regular box-plot fashion, for both repetition of the experiments and for both used algorithms, in case a simple IDWI model is used, there is almost no improvements in the accuracy of fingerprinting. However, in case a more complex MW model is used, the improvement in accuracy is visible and for both fingerprinting algorithms it increases with a number of virtual training fingerprints. In the best case scenario, the evaluation results show that the average localization error of the algorithm “Pompieu-Hausdorff distance of RSSI quantiles” decreases from roughly 2.5 m to less than 1.8 m, due to the generation of 285 virtual fingerprints, which is an improvement of roughly 28%. The improvement in the accuracy is less emphasized for the algorithm “Euclidean distance of averaged RSSI vectors”, which indicates that different gains in accuracy can be expected for different algorithms.

An increase in the number of training points generally increases the processing time of a fingerprinting algorithm, since the user’s generated fingerprint has to be compared with a larger number of training fingerprints. We evaluated the processing time of the used fingerprinting algorithms by requesting for each of the 20 evaluation points 100 times the location estimates. The time needed for providing each location estimate was measured and afterwards the statistical information about the processing time needed for providing one location estimate was calculated. The increase in the processing time of the used fingerprinting algorithms, due to the generation of the virtual training fingerprints, is given in Table II. As visible from the table, the increase in the accuracy comes at the cost of an increased processing time. For the aforementioned example, the increase of 28% in the accuracy of fingerprinting comes at the cost of roughly 45% increase in the processing time of the algorithm. However, this time is not a dominant factor in the latency of fingerprinting, since the sampling of the WiFi environment takes 2-3 s, depending on the hardware and device drivers.
TABLE I: Summarized statistics of the evaluation results

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>29 VTP</th>
<th>110 VTP</th>
<th>285 VTP</th>
<th>MW 29 VTP</th>
<th>MW 110 VTP</th>
<th>MW 285 VTP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Means</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Room acc. [%]</td>
<td>75.0</td>
<td>75.0</td>
<td>75.0</td>
<td>75.0</td>
<td>80.0</td>
<td>80.0</td>
<td>85.0</td>
</tr>
<tr>
<td>Room acc. [%]</td>
<td>65.0</td>
<td>65.0</td>
<td>65.0</td>
<td>65.0</td>
<td>70.0</td>
<td>70.0</td>
<td>70.0</td>
</tr>
<tr>
<td>Max error [m]</td>
<td>6.22</td>
<td>5.20</td>
<td>3.38</td>
<td>3.67</td>
<td>3.48</td>
<td>3.83</td>
<td>3.43</td>
</tr>
<tr>
<td>Min error [m]</td>
<td>1.05</td>
<td>0.23</td>
<td>0.77</td>
<td>1.09</td>
<td>1.07</td>
<td>0.92</td>
<td>0.52</td>
</tr>
<tr>
<td>Median error [m]</td>
<td>2.54</td>
<td>2.26</td>
<td>2.51</td>
<td>2.27</td>
<td>2.29</td>
<td>2.31</td>
<td>2.35</td>
</tr>
<tr>
<td>Mean time [s]</td>
<td>0.37</td>
<td>0.39</td>
<td>0.43</td>
<td>0.44</td>
<td>0.47</td>
<td>0.48</td>
<td>0.48</td>
</tr>
<tr>
<td>Min time [s]</td>
<td>0.23</td>
<td>0.27</td>
<td>0.30</td>
<td>0.28</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>Max time [s]</td>
<td>0.76</td>
<td>0.78</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
</tr>
<tr>
<td><strong>Pompeiu-Hausdorff distance of RSSI quantiles</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Room acc. [%]</td>
<td>75.0</td>
<td>75.0</td>
<td>75.0</td>
<td>75.0</td>
<td>80.0</td>
<td>80.0</td>
<td>85.0</td>
</tr>
<tr>
<td>Max error [m]</td>
<td>4.87</td>
<td>3.27</td>
<td>3.27</td>
<td>3.27</td>
<td>3.27</td>
<td>3.27</td>
<td>3.27</td>
</tr>
<tr>
<td>Min error [m]</td>
<td>0.14</td>
<td>0.06</td>
<td>0.10</td>
<td>0.06</td>
<td>0.10</td>
<td>0.06</td>
<td>0.06</td>
</tr>
<tr>
<td>Median error [m]</td>
<td>1.88</td>
<td>1.80</td>
<td>1.80</td>
<td>1.80</td>
<td>1.80</td>
<td>1.80</td>
<td>1.80</td>
</tr>
<tr>
<td>Mean time [s]</td>
<td>0.37</td>
<td>0.39</td>
<td>0.43</td>
<td>0.44</td>
<td>0.47</td>
<td>0.48</td>
<td>0.48</td>
</tr>
<tr>
<td>Min time [s]</td>
<td>0.23</td>
<td>0.27</td>
<td>0.30</td>
<td>0.28</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>Max time [s]</td>
<td>0.76</td>
<td>0.78</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
</tr>
</tbody>
</table>

VIII. CONCLUSION AND FUTURE WORK

In this paper, we presented the Enriched Training Database (ETD), a web-service that can be used for storing and managing WiFi RSSI measurements and training fingerprints for various types of fingerprinting algorithms. The enriching functionality of the ETD can be used to automatically generate virtual training fingerprints and store them together with the original training dataset. We have shown that leveraging the enriching functionality of the ETD results in increased accuracy of fingerprinting. The improvement is clearly dependent on the evaluation environment and the used algorithm, and it is expected that different propagation models will be optimal for different environments and algorithms, as indicated in [23].

A modular design of the ETD, apart from supporting an integration of different fingerprints algorithms, allows easy implementation of additional propagation models, and this feature can, in the longer run, serve as a basis for creating a framework for comparing the feasibility of different propagation models for generating virtual training fingerprints for different environments and algorithms. Finally, future work, in addition to the implementation of propagation models, will include integration of self-evaluating capability in the ETD. This capability is envisioned to allow the users to, in a simple way, identify a propagation model that should be leveraged for generating virtual training fingerprints for their environment in order to maximize the accuracy of a specific algorithm.

ACKNOWLEDGMENT

This work has been partially funded by the European Commission (FP7-ICT-FIRE) within the project EVARILOS (grant No. 317989). The author Filip Lemic was partially supported by DAAD (German Academic Exchange Service).

REFERENCES