









# Autoencoder Extreme Learning Machine for Fingerprint-Based Positioning: A Good Weight Initialization is Decisive

Darwin P. Quezada Gaibor , Lucie Klus , *Student Member, IEEE*, Roman Klus , *Student Member, IEEE*, Elena Simona Lohan , *Senior Member, IEEE*, Jari Nurmi , Mikko Valkama , *Fellow, IEEE*, Joaquín Huerta , and Joaquín Torres-Sospedra 

**Abstract**—Indoor positioning based on machine-learning (ML) models has attracted widespread interest in the last few years, given its high performance and usability. Supervised, semisupervised, and unsupervised models have thus been widely used in this field, not only to estimate the user position, but also to compress, clean, and denoise fingerprinting datasets. Some scholars have focused on developing, improving, and optimizing ML models to provide accurate solutions to the end user. This article introduces a novel method to initialize the input weights in autoencoder extreme learning machine (AE-ELM), namely factorized input data (FID), which is based on the normalized form of the orthogonal component of the input data. AE-ELM with FID weight initialization is used to efficiently reduce the radio map. Once the dimensionality of the dataset is reduced, we use  $k$ -nearest neighbors to perform the position estimation. This research work includes a comparative analysis with several traditional ways to initialize the input weights in AE-ELM, showing that FID provide a significantly better reconstruction error. Finally, we perform an assessment with 13 indoor positioning datasets collected from different buildings and in different countries. We show that the dimensionality of the datasets can be reduced more than 11 times on average, while the positioning error suffers only a small increment of 15% (on average) in comparison to the baseline.

**Index Terms**—Autoencoder (AE), extreme learning machine (ELM), indoor positioning, singular value decomposition (SVD), weight initialization, Wi-Fi fingerprinting.

## I. INTRODUCTION

**G**IVEN the high demand for indoor positioning services, it is of high importance to provide reliable solutions for indoor positioning systems (IPSs) in mobile and Internet of Things (IoT) devices. Indoor positioning—and in particular,

Manuscript received 20 March 2023; revised 1 July 2023; accepted 17 July 2023. Date of publication 27 July 2023; date of current version 22 August 2023. This work was supported in part by the European Union's Horizon 2020 Research and Innovation programme under the Marie Skłodowska Curie under Grant 813278 (A-WEAR: A network for dynamic wearable applications with privacy constraints), and Grant 101023072 (ORIENTATE: Low-cost Reliable Indoor Positioning in Smart Factories), and in part by the Academy of Finland under Grant 319994, Grant 323244, Grant 328214, Grant 338224, Grant 345654, and Grant 357730. (Corresponding author: Joaquín Torres-Sospedra; Darwin P. Quezada Gaibor.)

Darwin P. Quezada Gaibor and Lucie Klus are with the Electrical Engineering Unit, Tampere University, 33720 Tampere, Finland, and also with the Institute of New Imaging Technologies, Universitat Jaume I, 12071 Castellón de la Plana, Spain (e-mail: quezada@uji.es; lucie.klus@tuni.fi).

Roman Klus, Elena Simona Lohan, Jari Nurmi, and Mikko Valkama are with the Electrical Engineering Unit, Tampere University, 33720 Tampere, Finland (e-mail: roman.klus@tuni.fi; elena-simona.lohan@tuni.fi; jari.nurmi@tuni.fi; mikko.valkama@tuni.fi).

Joaquín Huerta is with the Institute of New Imaging Technologies, Universitat Jaume I, 12071 Castellón de la Plana, Spain (e-mail: huerta@uji.es).

Joaquín Torres-Sospedra is with the Centro ALGORITMI, Universidade do Minho, 4800-058 Guimarães, Portugal (e-mail: jtorres@algoritmi.uminho.pt).

Digital Object Identifier 10.1109/JISPIN.2023.3299433

fingerprinting—is a relevant research area for mobile computing, as researchers are continuously looking for new methods not only to improve the positioning accuracy [1], [2], [3] but also to reduce the costs of calibration (site survey) and the maintenance and/or execution at the operational phase [4], [5], [6], [7]. In addition, despite the fact that the current devices and data centres have improved their storage resources significantly over the past few years, additional data optimization prior to the data storage is still required. Data optimization not only allows the efficient use of storage but also improves the speed of data extraction.

The exploitation of the crowdsourcing and crowdsensing schemes to enrich the indoor positioning data collection, as well as the development of new methods that allow updating the radio map are two of the main reasons why IPSs became highly popular in the recent years [8]. However, a higher amount of available IPSs data might increment the data processing load and the time-to-estimate the user position. This high amount of data occurs due to a significant amount of samples collected nowadays by mobile devices for positioning, sensing, and/or communication purposes. In addition, the device diversity increases the heterogeneity of the datasets [9]. Several alternatives have been proposed to reduce the complexity and dimensionality of radio maps [10], [11], [12] and computational load at the operational stage [13].

Machine learning (ML) is one of the alternatives that has been successfully used for indoor positioning, not only to estimate the user position but also to reduce and denoise the radio maps. For instance, Jang and Hong [14] proposed a convolutional neural

network (CNN) model to deal with the random effects due to wireless signal propagation in indoor scenarios. The CNN is mainly used to extract useful information from the data, given its strong filtering properties. As well as CNN, extreme learning machine (ELM), is becoming more and more popular to solve classification [15], regression [16], and data compression [17] problems. Unlike other algorithms, ELM provides lower training time, which makes it suitable for computationally restrained applications. In spite of the fact that deep learning and ELM are growing in popularity in the field of indoor positioning,  $k$ -nearest neighbors ( $k$ -NN) is still used as the core element of many IPS solutions to estimate the user location [10], [18], [19]. However, it is paramount to highlight that these models and algorithms can be, and have been, combined in more complex applications, e.g., Alitaleshi et al. [20] used autoencoder extreme learning machine (AE-ELM) with CNN to extract relevant information from the datasets and estimate the device position accurately.

Autoencoder (AE)-based ML models were successfully used to transform high-dimensional data into a low-dimensional representation [21]. Generally, AEs are composed of two main submodels: the encoder, which is responsible for dimensionality reduction or data compression, and the decoder, which attempts to reconstruct the input data from the compressed or encoded representation. For instance, in [22], the authors used a self-encoder to reduce the dimensionality of IEEE 802.11 Wireless LAN (Wi-Fi) radio maps as well as a stacked model with the random-forest algorithm in order to reduce the positioning error. Consequently, the authors were able to improve the floor hit rate by more than 3% in comparison with the baselines.

This work applies AE-ELM to reduce the radio-map dimensionality by extracting the most relevant features of the IPS dataset. Subsequently,  $k$ -nearest neighbors ( $k$ -NN) is applied on the reduced radio map to estimate the user or the device position. The core of this work is to provide a novel and efficient method to compute the input weights, in order to reduce the reconstruction error after decoding. Although, the orthogonal random initialization provides better performance than other random weight initialization methods, it can be further improved by using one of the orthogonal components, calculated from the input data using singular value decomposition (SVD). The main contributions of this article are the following.

- 1) Proposing factorized input data (FID), a novel analytical estimation of the input weights for AE-ELM.
- 2) Exhaustive assessment of FID against six traditional weight initialization methods from the literature.
- 3) Comprehensive comparison of various combinations of AE-ELM with different weight initialization methods, including the proposed method (FID), and algorithms to estimate the user/device position. Furthermore, we offer a performance analysis of the proposed combination (AE-ELM + FID +  $k$ -NN) against two ML models from the literature. We thus provide an overall view of the advantages and drawbacks of the different positioning solutions in terms of positioning accuracy.

The rest of this article is organized as follows. Section II provides an overview of current studies in the field of interest. Section III provides a general description of ELM, AE-ELM,

and weight initialization methods. Section IV introduces the proposed method, FID. Section V shows the experimental setup and the primary results. Section VI provides a discussion of the advantages and drawbacks of FID. Finally, Section VII concludes this article.

## II. RELATED WORK

This section reviews the related literature on Fingerprinting, radio map compression, and AE-ELM, highlighting how this article contributes to them.

### A. Wi-Fi Fingerprinting Positioning

Fingerprinting, as a method of choice for indoor positioning purposes, has significantly gained in popularity in the recent decade. One of the major advantages of this solution is its independence from any path-loss model, as indoor localization is performed in rich scattering environments with numerous multipath components for each signal modeling. Utilizing path-loss models in such scenarios is either highly ineffective or computationally expensive, depending on the model's parameters. In recent years, multiple indoor positioning surveys were published, such as [23], [24], [25], or [26], highlighting fingerprinting as one of the most relevant and reliable methods.

Traditionally, fingerprinting is realized by matching the measurement array we want to localize with a database of similar arrays, which were obtained at known location coordinates. The matching algorithm is usually based on  $k$ -NN and weighted variants. Recently, Subedi and Pyun [27] combined  $k$ -NN with a weighted centroid algorithm. The method was able to reduce the number of fingerprints by 40% with a similar localization error. Duan et al. [28] proposed multiple improvements to fingerprinting, including an access point (AP) switching strategy, by considering multiple AP power levels and introducing a new matching algorithm called dynamic nearest neighbors, which enabled passive user localization. The combination of the magnetic field strength and channel state information measurements for fingerprinting purposes was realized in [29] and showed that such a combination outperformed the plain  $k$ -NN algorithm while identifying line-of-sight measurements. Cellular signals can also be used to perform indoor positioning. For example, Pecoraro et al. [30] used long term evolution channel frequency response fingerprints for localization, outperforming state-of-the-art methods.

In this case, we combined two ML algorithms, namely AE-ELM + a new weight initialization method and  $k$ -NN, to compress/reduce the radio map and estimate the user/device position, respectively. Although AE-ELM significantly reduces the radio map size, the positioning error was not significantly affected, while in some cases, the positioning error was even reduced.

### B. Radio Map Compression

Given the high importance of data compression and dimensionality reduction in today's data management of Big Data databases, there are several emerging approaches. Many of them may be applied to radio-map data-size reduction [31]. Although

the basic mechanism of each method may be different, their most important property is their capability to reduce the amount of data in any dimension, while keeping the information value as high as possible, and consequently, not reducing the quality of the data itself [32]. Many methods greatly reduce the size of the database, but significantly reduce the capability of the positioning system to accurately determine the position as well. The tradeoff margin between the achievable data compression and the reduction of the dataset's positioning capabilities is a key mechanism, that establishes which method is suitable for a given task [33].

Although some of the methods are preserving or even improving their positioning capabilities, their achievable compression may not be sufficient, or the computational cost of the whole process may be too great to actually pay off. The higher the computational cost, the higher the financial cost required and/or the time needed to locate the user using that particular positioning system.

In recent literature, several ML-based approaches were utilized in order to overcome some of these shortcomings. The majority of the methods present mechanisms capable of significantly reducing the database size while keeping comparable or better positioning accuracy to the benchmarks.

Seong and Seo [34] proposed an algorithm based on the minimum description length principle and positioning based on the Chi-squares test. The authors achieved an improvement in positioning of up to 5% compared to the Euclidean distance method while decreasing the size of the original radio map by 38.56%.

Similarly, Klus et al. [35] proposed a system involving  $k$ -means clustering and  $k$ -NN-based positioning, capable of compressing the received signal strength (RSS) fingerprinting dataset with compression ratios of up to 15.97. The proposed method slightly decreased the positioning error (by 1% on average) of the system, compared to the performance without the proposed compression on the considered datasets.

Talvitie et al. [36] used a spectral compression in order to reduce the database used for fingerprinting maps. They achieved up to 80% of dataset size reduction, while the positioning accuracy remains comparable with the traditional approach. This approach is later broadened and deeply analyzed in [37].

In this study, we employed the AE-ELM technique to reduce the dimensionality of the radio map by over 1.5 times in each dataset. This not only decreases the training effort but also minimizes the compression error when compared to several benchmark methods. In addition, this approach facilitates effortless expansion of the training database. However, it is important to note that the average positioning error increased by 15% under a *simple configuration*.

### C. Autoencoder Extreme Learning Machine (AE-ELM)

AEs are artificial neural network (ANN) structures from the unsupervised learning branch of ML. The difference over the typical ANN model lies in utilizing the same data as both training samples and labels, allowing an efficient feature extraction. AEs map the input into the same output through a hidden layer,

also called encoder, which changes the passed information by increasing (e.g., sparse encoding) or decreasing (e.g., compression) the dimensionality of the data, or otherwise augmenting the data itself. The output layer, also called the decoder, transforms the encoded information back to its original form. The encoder's output, denoted encoded layer, serves as an input of the decoder. AEs are nowadays utilized in numerous fields, including data compression [38], feature extraction [39], [40], encryption [41], channel coding [42], or indoor localization [43].

An end-to-end IPS based on the AE was also proposed in [44], where the authors specified two operational phases of the system as offline, used for model training, and online, for prediction and localization. The AE was able to significantly reduce the localization error when using ANN for location prediction.

AE-ELM, proposed in [45], is a special type of the AE, which sets the weights of the encoder layer at random, and calculates the decoder weights using the least-squares method. The main advantage of such approach is to remove the necessity of an iterative training process. Ding et al. [46] showed the suitability of the AE-ELM in electroencephalogram classification. Another advantage of the AE-ELM is its ability to stack multiple models, resulting in a deep AE architecture with better compression capabilities. Nuha et al. [47] utilized a stacked AE-ELM to efficiently compress seismic data, achieving comparable performance with much lower training times and effort, compared to the benchmark. Lu et al. [48] utilized the AE-ELM structure for fingerprint-based indoor positioning, efficiently removing the noise from the measurements, increasing the positioning accuracy and reducing the worst-case error, compared to the other IPS. The suitability of the stacked AE-ELM was also evaluated in [21], where the proposed model outperformed the commonly utilized fingerprinting methods such as  $k$ -NN.

In contrast to the literature, we propose an AE-ELM with novel weight preinitialization in the encoder. The proposed solution can learn the optimum weights from the input data, providing a stabilized performance compared to the randomly initialized AE.

## III. BACKGROUND

In this section, we provide a general overview of ELM networks, AE-ELM, and weight initialization methods. This section also introduces the most common symbols used in this work (see Table I).

### A. Extreme Learning Machine (ELM)

An ELM is considered as a learning method for single hidden layer feedforward neural network (SLFN), which improves training, learning, and weight initialization process [49]. We consider an input with distinct  $N$  sample pairs ( $i = 1, \dots, N$ ), represented as  $(\mathbf{x}_i, \mathbf{t}_i)$ , where  $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in \mathbb{R}^n$  ( $n$  is the number of features),  $\mathbf{t}_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbb{R}^m$  ( $m$  is the number "classes"). Then, the SLFN model can be represented as follows:

$$f_L(x_j) = \sum_{i=1}^L \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) \triangleq t_j, j = 1, 2, \dots, N \quad (1)$$

TABLE I  
SYMBOLS AND NOTATIONS

Symbol	Dim.	Definition
$N$	$\mathbb{N}$	Number of samples
$L$	$\mathbb{N}$	Number of hidden nodes
$c$	$\mathbb{R}$	Regularization term
$\gamma$	$\mathbb{R}$	Learning rate
$\xi$	$\mathbb{R}$	Average Root Mean Square Error (RMSE) [-]
$b_i$	$\mathbb{R}$	Bias term
$\mathbf{x}_i$	$\mathbb{R}^n$	Input vector
$\mathbf{t}_i$	$\mathbb{R}^m$	Target vector
$\mathbf{w}_i$	$\mathbb{R}^n$	Random initial weights
$\tilde{\mathbf{f}}_i$	$\mathbb{R}^n$	FID weight initialization vector
$\mathcal{F}$	$\mathbb{R}^{L \times n}$	Initial weight matrix from the orthogonal component $\mathcal{V}$
$\beta$	$\mathbb{R}^{L \times m}$	Output weight matrix
$T$	$\mathbb{R}^{N \times m}$	Training data target matrix
$\mathbf{X}$	$\mathbb{R}^{N \times n}$	Input data
$\mathbf{H}$	$\mathbb{R}^{N \times L}$	Hidden layer output of the SFLN
$\varphi$	$\mathbb{R}^{N \times N}$	Left singular vectors of $\mathbf{X}$
$S$	$\mathbb{R}^{n \times n}$	Diagonal singular values of $\mathbf{X}$
$\mathcal{V}$	$\mathbb{R}^{L \times n}$	Right singular vectors of $\mathbf{X}$
$H^\dagger$	$\mathbb{R}^{N \times L}$	Moore–Penrose generalized inverse of $\mathbf{H}$
$g(\cdot)$		Activation function
$\Psi$		Represents the weights ( $\mathbf{w}_i$ , $\beta_i$ and $b_i$ )
CR		Compression ratio [-]
$\epsilon_{3D}$		Mean 3-D positioning error [m]
$\tilde{\epsilon}_{3D}$		Normalized 3-D positioning error. The benchmark is the result of the simple configuration with $k$ -NN [-]
$\zeta$		Floor hit rate [%]
$\sigma$		Standard deviation

where  $\beta_i = [\beta_{i1}, \beta_{i2}, \dots, \beta_{im}]^T$  is the weight vector that connects the  $i$ th hidden node with the output layer,  $g(\cdot)$  is the activation function (AF) of the hidden layer,  $\mathbf{x}_j$  represents the input vector,  $\mathbf{w}_i = [w_{i1}, w_{i2}, \dots, w_{in}]$  is the weight vector connecting the  $i$ th hidden nodes with the input layer,  $b_i$  (bias term) represents the threshold of the  $i$ th hidden node and  $L$  represents the number of hidden nodes in the hidden layer.

Equation (1) can be also represented as

$$\mathbf{H}\beta = \mathbf{T} \quad (2)$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1 \mathbf{x}_1 + b_1) & \dots & g(\mathbf{w}_L \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \mathbf{x}_N + b_1) & \dots & g(\mathbf{w}_L \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L} \quad (3)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \quad \text{and} \quad T = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m} \quad (4)$$

where  $\mathbf{H}$  represents the hidden layer output matrix of the SFLN [50] and  $\mathbf{T}$  is the training data target matrix. Huang et al. [49] demonstrated that the input weights ( $\mathbf{w}_i$ ) and the hidden bias ( $b_i$ ) in the SFLN do not have to be necessarily adjusted or tuned as is done in traditional neural networks. In this case, instead of training an SFLN, it is only required to find the least-squares solution of (2).

Thus, if the number of neurons in the hidden layer ( $L$ ) is less than the number of the inputs ( $N$ ) ( $L < N$ ), Huang et al. [49]

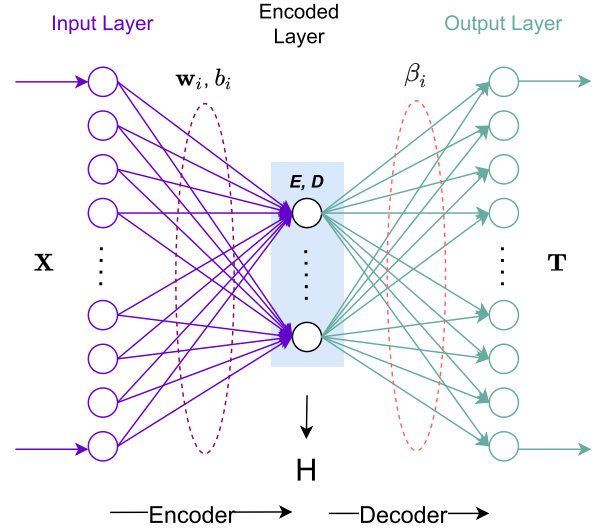


Fig. 1. AE-ELM general scheme.

use the *Moore–Penrose generalized inverse* ( $\mathbf{H}^\dagger$ ) of the matrix  $\mathbf{H}$  to solve the linear system (2), and therefore, the output weight matrix ( $\beta$ ) can be calculated by

$$\beta = \mathbf{H}^\dagger \mathbf{T}. \quad (5)$$

The aforementioned equation is the equation of the learning method for the SLFN, also called ELM.

There are several methods to compute the *Moore–Penrose generalized inverse*, for instance, the orthogonal projection method [49], when  $\mathbf{H}^T \mathbf{H}$  is nonsingular, as

$$\mathbf{H}^\dagger = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \quad (6)$$

or

$$\mathbf{H}^\dagger = \mathbf{H}^T (\mathbf{H} \mathbf{H}^T)^{-1} \quad (7)$$

when  $\mathbf{H} \mathbf{H}^T$  is also nonsingular. In addition Huang et al. [49] mention that multiple AFs can be used with ELM, including periodic functions and nonlinear functions such as sine or sigmoid function, among others.

## B. Autoencoder (AE)

In the case of the AE-ELM, the target output is identical to the input data, therefore, is using unsupervised learning. Thus, the AE-ELM compresses high-dimensional input data  $\mathbf{X}$  ( $\mathbf{X} = [\mathbf{x}_1^T, \mathbf{x}_2^T, \dots, \mathbf{x}_N^T]^T \in \mathbb{R}^{N \times n}$ ) into a low-dimensional representation  $\mathbf{H} \in \mathbb{R}^{N \times L}$ , preserving the meaningful information of the input data. In general, better performance can be achieved using the orthogonal random input weights and bias [51], instead of using uniformly distributed weights. In addition, in order to reduce overfitting issues and provide a robust solution, a regularization term can be added as follows [52]:

$$\beta = \left( \mathbf{H}^T \mathbf{H} + \frac{1}{c} \right)^{-1} \mathbf{H}^T \mathbf{T} \quad (8)$$

where  $c$  ( $c > 0$ ) is the regularization term.

Fig. 1 shows a graphical representation of the AE-ELM, which is composed of two parts, the *encoder* (left) and the *decoder* (right). The encoder consists of the input data  $\mathbf{X}$ , and a densely connected layer with input weights ( $\mathbf{w}_i$ ) and bias term ( $b_i$ ), whereas the decoder is composed of the densely connected layer with output weights ( $\beta_i$ ) and the training data target  $T$ . The encoded layer is shared between the encoder and the decoder as their output and input, respectively. Generally, the initial weights are generated using random uniformly distributed data in the range  $[-r, r] \{\forall r \in \mathbb{R} | r \neq 0\}$ . Similarly, the bias term is a random 1-D array.

### C. Weight Initialization

In the literature, many different methods to initialize the input weights in neural networks were analyzed and proposed, in order to reduce the convergence time and improve the model performance. The existing literature distinguishes two methods of weights initialization, namely the initialization of the input weights with zeros/ones and the initialization of the input weights based on a random distribution [53]. In zero/one initialization, all weights have values of either zero or one. In a random weight initialization, real numbers ( $\mathbb{R}$ ) will be set as initial weights, sampled from the given distribution based on predefined variance or range [54].

Some of the random weight initialization methods proposed in the literature are *random uniform initialization*, *orthogonal random initialization*, *identity*, or *variance scaling* [53]. They are widely used in popular ML libraries and frameworks, such as TensorFlow, Keras, and Pytorch. Their objective is to prevent exploding or vanishing gradient issues during the training stage in neural networks with multiple hidden layers, leading to a stabilized propagation of gradients.

Glorot et al. [55] and He et al. [56] developed two well-known weight initialization methods based on a random initialization. Glorot et al. [55] proposed a formula based on the number of input and output neurons using a heuristic  $U = [-1/\sqrt{n}, 1/\sqrt{n}]$ , where  $U[\cdot]$  is the uniform distribution between the given intervals and  $n$  is the size of the previous layer (weights). He et al. [56] extended the previous analysis on the basis of the nonlinearities of the rectified linear unit (ReLU) AF. They added a scale product  $\sqrt{2}$  to the input weights, obtaining the  $U = [-2/n, 2/n]$ .

In contrast with the previous methods, Romero [57] proposed ELM-Input based on the input data to determine the input weights in ELM networks. The authors selected  $n$  random features from the input data, and in the case of multilayer perceptron, the selected features were normalized with zero mean and unit variance. ELM-Input improved the classification accuracy by 2% to 10% with respect to the ELM network using the radial basis function.

Narkhede et al. [54] provide an overview of the available initialization methods for neural networks in general, highlighting the impact of the initialization in faster convergence, avoiding false optima, coping with network depth, or improving posttraining accuracy of the model. For initialization of the ELM, particle swarm or evolutionary algorithms [58], [59] offer strong performance with increased pretraining requirements.

Javed et al. [60] combine the wavelet theory with the ELM for *a priori* weight calculation while considering neuron-wise summation and a double AF. In comparison to the methods presented previously, the method proposed in this work does not require extensive and iterative pretraining for finding optimal weights of the encoder layer, while estimating the required network dimensionality in the process.

## IV. PROPOSED METHOD

In this section, we introduce a novel method to initialize the input weights in AE-ELM networks and describe the key components of the proposed indoor positioning solution.

### A. FID Weight Initialization

Unlike the previous methods (see Section III-C), this research work proposes a novel weight initialization method based on the orthogonal components of the input matrix, namely FID weight initialization. Aiming to reduce the reconstruction error of the decoder of the AE-ELM.

Since ELM is the method used to learn an SLFN, the weights, bias, and output weights in an SLFN can be found using (9), which is equivalent to minimizing the cost function. For instance, using a gradient-based learning algorithm [see (10)]. In this case, the weights are iteratively tuned or adjusted during the training phase, so that

$$\min_{\mathbf{w}_i, b_i, \beta} \|\mathbf{H}(\mathbf{w}_1, \dots, \mathbf{w}_L, \dots, b_1, \dots, b_L)\beta - \mathbf{T}\| \quad (9)$$

and

$$\Psi_l = \Psi_{l-1} - \gamma \frac{\partial \xi(\Psi)}{\partial(\Psi)} \quad (10)$$

where  $\Psi$  represents the weights ( $\mathbf{w}_i$ ,  $\beta_i$ , and  $b_i$ ),  $\gamma$  is the learning rate, and  $\xi$  is the error in prediction.

Huang et al. [49] stated that an SLFN can be solved using the minimum normal least-squares method, because the inputs weights ( $\mathbf{w}_i$ ) and biases ( $b_i$ ) do not need to be necessarily tuned. However, weight initialization is crucial in any neural network, which is why many researchers investigated the best way to initialize this parameter.

Given the importance of weight initialization, we propose the following method.

- 1) *Step 1: Input Data Factorization*: Based on the analysis provided by Saxe et al. [61] in 2014, where the authors demonstrated the advantages of using random orthogonal initialization to have a better propagation of gradients, we propose to determine the best *initial weights* by factorizing the input matrix  $\mathbf{X}$  using SVD [see (11)] to get the orthogonal components ( $\mathcal{V}$ ), as

$$\mathbf{X} = \varphi S \mathcal{V}^T \quad (11)$$

where  $\varphi$  is a square matrix ( $N \times N$ )  $\in \mathbb{R}$ ,  $\mathcal{V}$  is a ( $n \times n$ ) matrix  $\in \mathbb{R}$  and both are orthogonal.  $S$  is a rectangular diagonal matrix ( $N \times n$ )  $\in \mathbb{R}$  composed by the singular values of  $\mathbf{X}$  ordered in descending order of magnitude ( $s_1 \geq s_2 \geq \dots \geq s_p \geq 0$ , where  $p = \min(N, n)$ ).

In this case,  $\mathcal{V}$  is used to generate the input weights in the proposed method and the  $S$  component is used to determine the initial number of dimensions to be compressed the input data.

- 2) *Step 2: Components Selection:* To efficiently compress the dataset, it is necessary to first determine the best approximation of the optimal number of dimensions in order to prevent a significant increment in the positioning error. The following procedure is used to determine the number of compressed features, which equals to the number of hidden nodes  $L$ .

The variance explained can be used to determine the number of compressed features used to represent the dataset. The overall variance explained can be obtained as follows:

$$\eta = \frac{s_i^2}{\sum_{j=1}^p s_j^2} * 100 \quad (12)$$

where  $\eta$  represents the overall variance explained,  $s_i$  is the  $i$ th column of the component  $S$  and  $s_j$  are the individual singular values of  $S$  ( $j = 1, 2, \dots, p$ ).

The number of dimensions is determined by the percentage of selected variance ( $\eta$ ). For instance, a dataset of 1000 samples and 200 features can be reduced using the proposed method to the same number of samples and a significantly lower number of features, while keeping, e.g., 90% of the total variance. Therefore, the first  $L$  components that sum up to  $\geq 90\%$  of the total variance are selected, while the rest are discarded. The number of hidden neurons used in the ELM is equal to the number of the selected components.

Thus, the new initial weight matrix ( $\mathcal{F}$ ) is composed of the first  $L$  rows of the orthogonal component  $\mathcal{V}$  as follows:

$$\mathcal{F} = \begin{bmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{L1} & \dots & v_{Ln} \end{bmatrix}_{L \times n} \quad (13)$$

- 3) *Step 3: Input Weight Normalization:* Weight normalization aims to restrict the weights under certain statistical properties during the optimization (training) [62]. As a result, the training process is more stable than without normalization, and the difference in magnitude between the features is reduced or removed. Although the orthogonal components of the input data are already constraining the training process, an additional restriction is introduced to the initial weights matrix ( $\mathcal{F}$ ), which is the *unit norm* normalization [63], also called *vector-length* normalization, shown as follows:

$$\widehat{\mathcal{F}}_i = \frac{f_i}{\|f_i\|}, i = 1, 2, \dots, n \quad (14)$$

where  $\widehat{\mathcal{F}}_i$  represents the new normalized input weight vector (FID weight initialization vector).  $f_i$  is the  $i$ th column of initial weight matrix ( $\mathcal{F}$ ), and  $\|f_i\|$  is the Euclidean norm of  $f_i$ . The random initial weights ( $\mathbf{w}_i$ ) are substituted by the new initial weights  $\widehat{f}_i$  in (3) in order

---

### Algorithm 1: AE-ELM and FID.

---

weight initialization

**Require:** training dataset

$X \leftarrow$  training dataset

/\* Get the orthogonal component  $\mathcal{V}$  using SVD \*/

$X = \varphi S \mathcal{V}^T$

/\* Compute the number of components  $\eta$  \*/

$\eta = \frac{s_i^2}{\sum_{j=1}^p s_j^2} \times 100$

/\* Get the initial weight form  $\mathcal{V}$  \*/

$\mathcal{F} = v_{ij}, i = 1, 2, \dots, L$  and  $j = 1, 2, \dots, n$

/\* Input weight normalization - Unit norm \*/

$\widehat{\mathcal{F}}_i = \frac{f_i}{\|f_i\|}$

/\* Substitute  $\mathbf{w}_i$  by  $\widehat{f}_i$  into  $\mathbf{H}$  \*/

see (15)

/\* Compute the output weights \*/

$\beta = (\mathbf{H}^T \mathbf{H} + \frac{1}{c})^{-1} \mathbf{H}^T$

**Output:**  $\beta, \mathbf{H}$

---

to compute the output weights  $\beta$  [see (8)]. Thus, (8) and (15) are differentiated only by the initial weights.

$$\mathbf{H} = \begin{bmatrix} g(\widehat{f}_1 \mathbf{x}_1 + b_1) & \dots & g(\widehat{f}_L \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\widehat{f}_1 \mathbf{x}_N + b_1) & \dots & g(\widehat{f}_L \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L} \quad (15)$$

Algorithm 1 summarizes the steps used to determine the input weights, the number of components and how these parameters are used in AE-ELM network. This algorithm returns the output weights ( $\beta$ ) as well as the hidden layer output ( $\mathbf{H}$ ).

### B. Position Estimation

Wi-Fi fingerprinting technique consists of two phases, the offline and the online phases. The offline phase is devoted to collecting all the RSS values at known reference points to build the radio map. In this phase, data processing techniques such as data cleaning or dimensionality reduction are applied and ML models are trained.

In the online phase, the  $k$ -NN algorithm computes the distance between the incoming fingerprint and the fingerprints in the radio map. Then, the fingerprints with the lowest distances to the incoming fingerprint are selected to compute the position estimate.

Fig. 2 shows a general scheme of how the AE-ELM is combined with the well-known  $k$ -NN algorithm in order to reduce the dimensionality of the radio map and estimate the user or device position. As the fingerprints are transformed, we will use the term “encoded” to refer to the fingerprints and radio map after applying the AE-ELM over the original samples. As we can observe, the input and output weights determined in the offline phase ( $\widehat{f}_i$  and  $\beta$ , respectively) are used in the online phase to encode the incoming fingerprint. Once the incoming fingerprint is encoded,  $k$ -NN is used to estimate the position with the encoded radio map.

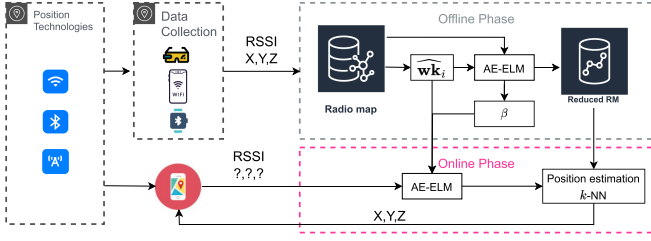


Fig. 2. Wi-Fi fingerprinting.

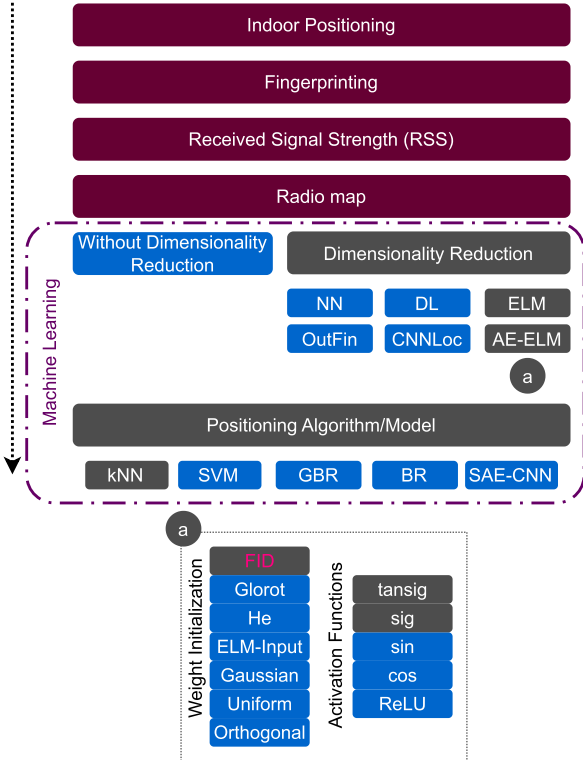


Fig. 3. General diagram of the proposed combination and the benchmarks used in the experiments. Dark red color represents the common components; blue is the benchmark; and dark gray is the suggested combination.

## V. EXPERIMENTS AND RESULTS

Fig. 3 shows a general diagram highlighting the proposed combination (dark gray) and the algorithms used as benchmarks (blue). The red color denotes the common components of both the benchmarks and the suggested combination (i.e., AE-ELM +  $k$ -NN). Overall, we compare the proposed AE-ELM with FID +  $k$ -NN with different algorithms, methods, or models (and their combinations) to provide a general view of the advantages and drawbacks of our proposal against other methods. For instance, the impact of FID weight initialization on AE-ELM networks is compared against AE-ELM combined with six different weight initialization methods (e.g., Glorot, He, Orthogonal, etc.). Additionally, we offer an analysis of how the AFs affect each combination (AE-ELM and weight initializations). This first step shows the performance of each combination to encode and decode fingerprinting datasets with minimal errors.

TABLE II  
DATASETS' PARAMETERS

Dataset	$ \mathcal{T}_{TR} $	$ \mathcal{T}_{TE} $	$ \mathcal{A} $	$ L $	Data rep.	#b	#f
DSI 1	1369	348	157	46	powed	1	1
DSI 2	576	348	157	43	positive	1	1
LIB 1	576	3120	174	31	positive	1	2
LIB 2	576	3120	197	45	positive	1	2
MAN 1	14300	460	28	15	exponential	1	1
MAN 2	1300	460	28	8	exponential	1	1
TUT 1	1476	490	309	50	positive	1	4
TUT 2	584	176	354	55	exponential	1	3
TUT 3	697	3951	992	113	positive	1	5
TUT 4	3951	697	992	182	positive	1	5
TUT 5	446	982	489	6	positive	1	3
TUT 6	3116	7269	652	53	positive	1	4
TUT 7	2787	6504	801	116	positive	1	3

Once the datasets are reduced, they are used with different regression algorithms to estimate the user position and determine which combination provides the lowest mean positioning error. The results obtained with the AE-ELM +  $k$ -NN are compared against two positioning models from the literature (i.e., CNNLoc and OutFin).

The following paragraph details the experiments carried out in this research work.

### A. Experimental Setup

The experiments were carried out using a computer with the following characteristics: Intel Core i7-8700 T at 2.40 GHz and 16 GB of RAM, the operating system is Windows 10, and the software used is MATLAB (AE-ELM and  $k$ -NN) and Python [CNNLoc, OutFin, support vector machine (SVM), gradient boosting regressor (GBR), and Bayesian ridge (BR)]. The developed algorithms, supporting software and data used are available in [64] for research reproducibility and replicability.

The experimental setup consisted of 13 indoor positioning datasets from different environments, which are available online for the public usage [65]. The datasets were collected by the following entities: Tampere University (TUT 1—TUT 7) [66], [67], [68], University of Minho (DSI 1 and DSI 2) [69], University of Mannheim (MAN 1 and MAN 2) [70], [71], and Universitat Jaume I (LIB 1, LIB 2) [72].

The advantage of using these datasets is that they were collected in different environments with varying numbers of devices in real-world deployments, and therefore, provide a complete analysis of the proposed methods in a multitude of realistic conditions. Moreover, these datasets were collected for different purposes, including the comprehensive assessment of limited crowdsourced radio maps over a large area in TUT 3. Assessment with multiple diverse datasets considered in this work is being consolidated in the indoor positioning community [73], [74], [75], [76], [77].

Table II shows the features of the datasets used in the experiments.  $|\mathcal{T}_{TR}|$  represents the number of samples in the training set,  $|\mathcal{T}_{TE}|$  the number of samples in the test set,  $|\mathcal{A}|$  is the number of AP (i.e., features), #b is the number of buildings, and #f represents the number floors. The table also includes the number of compressed features  $|L|$  (i.e., the number of hidden nodes in

the ELM network), and data representation (*data rep.*)—we use the same data representation as in [73] and described in [78], allowing us a better data abstraction by ML algorithms than using the raw data. Despite the fact that each dataset considers different scenario, scale of deployment, measurement methodology, sample density, etc., they were formatted into a consistent structure with an array of RSS measurements as features, and an array of local coordinates ( $x, y, z$ , coordinates, floor index, and building index) as labels.

Despite the traditional assessment in indoor positioning and ML holding a ratio  $\approx 80 : 20$  for training and testing, we use the data partition provided with the datasets. Applying  $k$ -fold cross validation is not recommended in fingerprinting as the samples taken in a short period may end in the training and testing sets, having overoptimistic results for them (i.e., data leakage). Therefore, we ensure that training and testing samples are independent, the evaluation can be reproduced as the same data partitions can be reused, and the results can be directly compared to previous and further works using these datasets.

Two baselines have been considered for the  $k$ -NN algorithms: *simple configuration* ( $k = 1$ , city-block distance metric and positive data representation) and *best configuration*, both defined in [73] and [78]. The analyzed methods were compared in terms of compression ratio (CR), positioning error ( $\epsilon_{3D}$ ), and floor hit rate ( $\zeta$ ). The CR is obtained by dividing the number of features in the original dataset by the number of features in the compressed dataset.

The proposed weight initialization is later compared with six well-known algorithms: Gaussian (zero mean and standard deviation equal to 0.01), random orthogonal (random values in the range of  $[-1, 1]$ ), random uniform (distribution bound 0.01), He [56], Glorot [55], and ELM-Input [57]. The AE-ELM was implemented using the aforementioned algorithms along with nonlinear and periodic AFs (sine, cosine, ReLU, sigmoid, and hyperbolic tangent sigmoid), and then, tested on each dataset. Given that the initial weights and/or the bias terms are initialized randomly, the AE-ELM was executed 20 times. We thus compute the root mean square error (RMSE) of the reconstruction error (i.e., the difference between the AE input and output), and its confidence bounds are obtained using the standard deviation. Likewise, the positioning model based on AE-ELM and  $k$ -NN was run 20 times to determine the oscillation of the positioning error.

The AE-ELM with FID weight initialization was also tested using three regression models (SVM, GBR, and BR) to comprehensively analyze our proposal. These three algorithms were implemented using *Sklearn* library (default hyperparameters). We thus reduced the dimensionality of the datasets using AE-ELM and each weight initialization method described previously to subsequently estimate the mean 3-D positioning error, employing the aforementioned regression algorithms.

In addition, the proposed AE-ELM using fixed weights +  $k$ -NN is compared with to CNNLoc [43] and OutFin [79], in order to compare their efficiency in terms of positioning accuracy. Unlike CNNLoc, OutFin is not a positioning or localization solution. Nevertheless, it consists of an AE as a part of software provided by the authors, and therefore, is a suitable baseline to

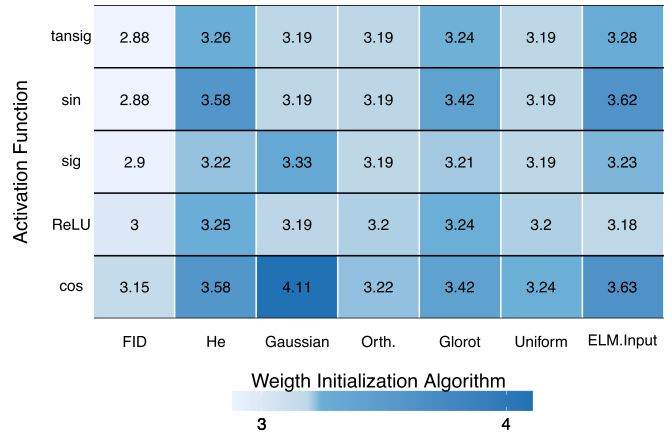


Fig. 4. Heatmap depicts the average RMSE (unitless) of the weight initialization algorithm versus AF. Figs. 5 and 6 show in detail the average RMSE of each combination (weight initialization model and AF) using a different number of hidden neurons.

compare our solution to. In each case, the mean positioning error was selected to be compared with our proposal. In both cases, the software developed in Python was obtained from the authors' repositories [80], [81]. We only adapted the inputs/outputs in those scripts to fit our data structure, keeping the data processing workflow, initialization mechanisms, and other parameters as originally defined.

As mentioned in the previous paragraph, we utilize a deep learning model called CNNLoc [43] to compare the positioning performance with the proposed solution. CNNLoc's architecture consists of an individually trained stacked AE, whose encoder's part serves as an input to the consecutive convolutional parts of the network. The model utilizes multiple 1-D convolutional layers as the core network. Three separate convolutional pipelines, each finalized with a single densely connected layer, predict latitude and longitude coordinates, building index and floor, respectively.

## B. Results

This section describes the results achieved by using the proposed weight initialization method in the AE-ELM network. These results include the reconstruction error, computational performance, as well as the positioning results using the methods described in the previous section.

1) *Weight Initialization*: This section provides the empirical results of using six random initializations and the proposed FID in AE-ELM.

Fig. 4 shows the average RMSE—reconstruction error—(scale 1:100) of combining seven weights initialization methods and five AFs across all datasets. Although the RMSE is comparable for each combination, the hyperbolic tangent sigmoid (tansig) AF provided the lowest reconstruction error regardless of the weight initialization method used in the network. We can also observe that the proposed FID algorithm outperforms the initialization algorithms commonly used in the literature.

The performance of the FID algorithm remains almost unchanged with the sigmoid, hyperbolic tangent sigmoid, and



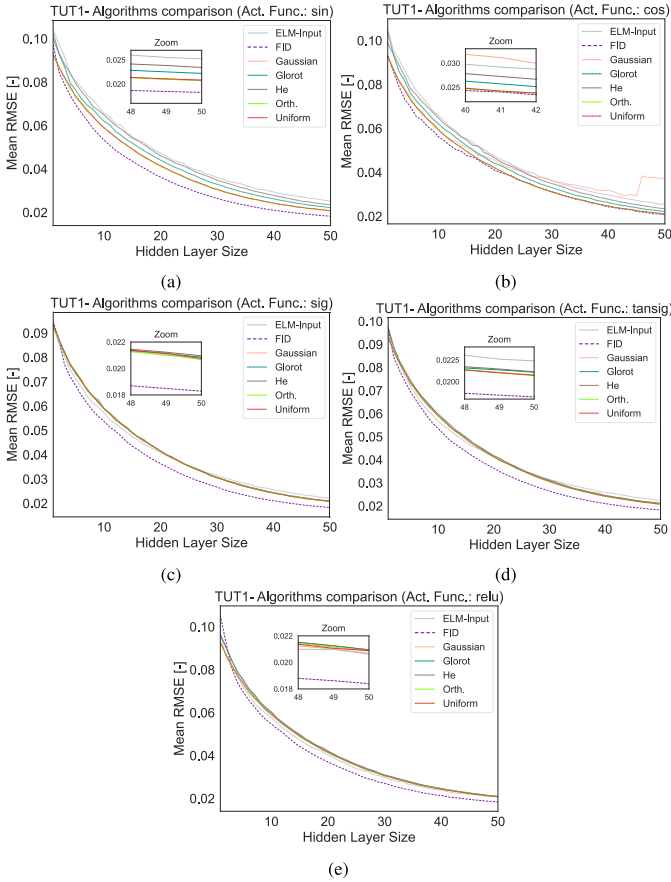


Fig. 5. Performance of weight initialization methods in TUT 1 for five AFs: (a) sine/sin, (b) cosine/cos, (c) sigmoid/sig, (d) hyperbolic tangent sigmoid/tansig, and (e) rectified linear unit (ReLU).

ReLU AFs, providing low reconstruction error. On the contrary, when the FID and periodic AFs are combined, the average RMSE increased by more than 6% in comparison with the tansig function. However, the computed error is still lower than the error of its counterparts at the same settings.

Fig. 5 shows the performance of the AE-ELM when different AFs and weight initialization algorithms are used in TUT 1 dataset. The  $x$ -axis represents the hidden layer size used in the network and the  $y$ -axis represents the mean of the RMSE per each hidden node after executing the AE-ELM 20 times.

As shown in Fig. 5(a), random uniform initial weights, orthogonal and Gaussian initialization, have similar performance when the sine AF is used in the network. Similarly, FID keeps its statistical properties, providing the lowest reconstruction error. In the case of using the cos AF, the performance of FID and the network is degraded with respect to sine.

When random Gaussian initialization and cos AF are used in the AE-ELM, the reconstruction error remains high despite increasing the hidden layer size. This behavior can be observed in TUT 1 [see Fig. 5(b)], MAN 1 and DSI 1 (see Fig. 6). For DSI 1 and MAN 1, the high RMSE is also produced by the type of data representation used in each dataset (powered and exponential, respectively). Gaussian weight initialization provides better performance when positive data representation is

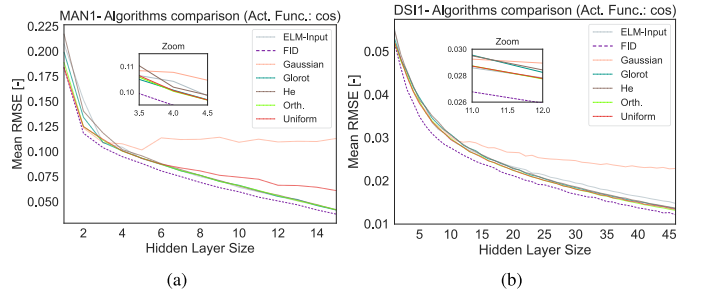


Fig. 6. Weight initialization algorithms and cosine AF. (a) MAN 1 dataset. (b) DSI 1 dataset.

TABLE III  
COMPARISON OF THE WEIGHT INITIALIZATION

Dataset	$\xi_{OR}$	$\tilde{\xi}_{OR}$	$\tilde{\xi}_{HE}$	$\tilde{\xi}_{GL}$	$\tilde{\xi}_{GA}$	$\tilde{\xi}_{UN}$	$\tilde{\xi}_{EI}$	$\tilde{\xi}_{FI}$
DSI1	0.023	1	1.015	1.010	1.002	1.001	1.024	<b>0.912</b>
DSI2	0.059	1	1.018	1.016	0.999	1.000	1.018	<b>0.914</b>
LIB1	0.025	1	1.015	1.011	0.999	0.999	1.006	<b>0.910</b>
LIB2	0.028	1	1.015	1.012	0.999	1.000	1.007	<b>0.914</b>
MAN1	0.083	1	1.021	1.015	1.004	0.995	0.992	<b>0.905</b>
MAN2	0.069	1	1.067	1.024	1.014	0.996	1.085	<b>0.923</b>
TUT1	0.041	1	1.016	1.011	1.001	1.000	1.006	<b>0.903</b>
TUT2	0.041	1	1.025	1.020	1.000	1.001	1.034	<b>0.895</b>
TUT3	0.028	1	1.027	1.023	1.000	1.000	1.039	<b>0.902</b>
TUT4	0.024	1	1.018	1.014	1.000	1.000	1.027	<b>0.896</b>
TUT5	0.039	1	1.025	1.022	1.002	1.001	1.057	<b>0.902</b>
TUT6	0.037	1	1.018	1.015	1.001	1.001	1.020	<b>0.911</b>
TUT7	0.021	1	1.021	1.017	1.000	1.001	1.064	<b>0.895</b>
Avg.	0.040	1	1.023	1.016	1.001	0.999	1.029	<b>0.906</b>

Best results are typeset in bold.  $\xi$ : Average RMSE [-],  $\tilde{\xi}$ : Normalized Average RMSE [-], OR: Random orthogonal, HE: He, GL: Glorot, GA: Random Gaussian, UN: Random uniform, EI: ELM-Input, and FI: FID weight initialization algorithm.

applied to the dataset as shown in Fig. 5(b). Moreover, the cos AF produces a similar performance using other weight initialization methods like uniform initialization in MAN 1.

Fig. 5(c) shows that the mean RMSE is similar in all weight initialization methods based on random values. Again, the reconstruction error obtained with FID is lower than other initialization methods. A similar behavior is obtained with tansig and ReLU AF [see Fig. 5(e)].

ELM-Input weight initialization provided a good performance using sigmoid, hyperbolic tangent sigmoid, and ReLU AFs [see Fig. 5(c)–(e)]. The best performance is obtained with the tansig AF, offering a reconstruction error lower than random-based methods.

Table III summarizes the results of the data reconstruction error ( $\xi$ , mean RMSE) using the tansig AF. The results are normalized to the random orthogonal initialization (OR), which serves as the baseline.

FID (FI) is the best-performing weight initialization, achieving 9% lower positioning error than the baseline (OR). Despite ELM-Input (EI) being also based on the input data, its average RMSE is 2.9% higher than the baseline (OR). The average RMSE of random uniform (UN) and Gaussian (GA) is very close to the baseline, and it is slightly higher for the models proposed by He (HE) and Glorot (GL).

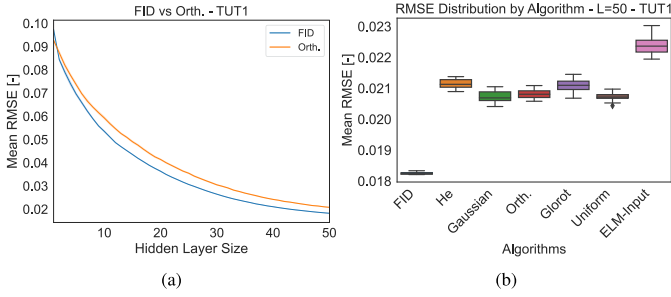


Fig. 7. Evaluation of TUT 1 dataset based on the (a) number of hidden layers, and (b) distribution of the RMSE.

Tapson et al. [82] mentioned that the random orthogonal initialization (OR) provided better performance and better generalization of ELM network than random uniform (UN), while we show here that their reconstruction errors are similar in Table III for the 13 fingerprint datasets. Anyway, the worst reconstruction errors are provided by the initialization based on ELM-Input (EI).

Given that weights ( $\mathbf{w}_i$ ) and/or bias ( $\mathbf{b}_i$ ) are randomly initialized each time that the ELM is executed (with the exception of  $\widehat{\mathbf{w}}\mathbf{k}_i$ , which is analytically determined), the model and its reconstruction error (RMSE) may significantly vary.

Fig. 7(a) shows the distribution of RMSE along the  $L$  hidden nodes and the confidence bound (standard deviation) of the RMSE, using FID and random orthogonal weight initialization on TUT 1. Despite the oscillation of the random values, the minimum error obtained with each hidden node is higher than the average RMSE obtained with the FID weight initialization.

Similarly, Fig. 7(b) shows the oscillation of the RMSE when the dimensionality of the dataset (TUT 1) is reduced to  $L$  dimensions. The  $x$ -axis represents the weight initialization algorithm and the  $y$ -axis represents the RMSE. The box plot distinguishes the median of the RMSE along with the first quartile, the third quartile, and the minimum and the maximum values of RMSE. The median error of using FID is lower than using He, Glorot, Gaussian, uniform random, and ELM-Input weight initialization by more than 13%. Furthermore, Fig. 7(b) also shows that the distribution of RMSE is significantly more stable (narrow) in FID than in any of its counterparts.

**2) Computational Performance:** This section is devoted to analyzing the computational performance of the proposed weight initialization. In the same vein as the previous section, FID initialization weight is compared (in terms of processing time) against six traditional methods from the literature: random orthogonal (OR), uniform (UN), He (HE), Glorot (GL), Gaussian (GA), and ELM-Input (EI) weight initializations.

Fig. 8 shows the time required by the analyzed methods to initialize the input weights, reduce the dimensionality of the datasets and the time used to estimate the position before and after the dimensionality reduction.

Fig. 8(a) shows the average time required to initialize the input weights using seven different methods, including FID. As expected, FID requires more time than random weight initialization methods, given that FID factorizes the input data, and then,

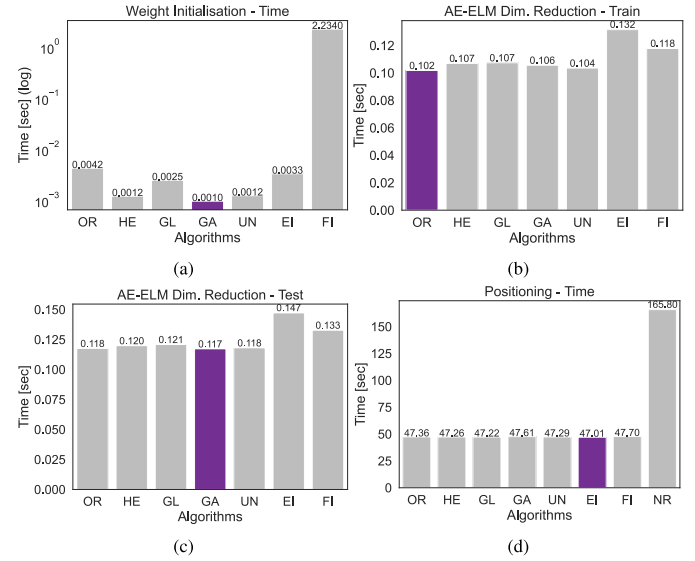


Fig. 8. Time analysis. (a) Average time to initialize the input weights. (b) Average time to reduce the dimension of the training dataset. (c) Average time to reduce the dimension of the test dataset. (d) Average time to estimate the position.

normalizes the selected dimensions. Once the input weights are initialized, the time employed to reduce the dimensionality of the datasets is similar in all the algorithms, except for ELM-Input, which is slightly higher than the others [see Fig. 8(b) and (c)]. Finally, Fig. 8(d) shows the time employed to estimate the position before the data encoding ( $NR =$  no dimensionality reduction) and after encoding the datasets using AE-ELM with the aforementioned initialization methods (gray bars).

It is important to emphasize that, the weight initialization is performed only once in the offline phase of fingerprinting, thus the computational load in the online phase will not be significantly affected by the type of weight initialization method used in the AE-ELM [see Fig. 8(c)].

**3) Positioning Performance:** In this section, we evaluate the positioning performance on the 13 datasets. The ELM with the proposed weight initialization (FID) and  $k$ -NN matching algorithm is compared with ELM using six weight initialization methods with  $k$ -NN. As such, we test the efficiency of the methods in real and heterogeneous indoor positioning deployments. In addition, given that sigmoid and tansig AFs facilitate a better learning process, both functions have been used to estimate the position and floor hit rate. The  $k$ -NN parameter setting corresponds to the *best configuration* baseline per each dataset, as found in [73].

Table IV provides results after running the AE-ELM with  $k$ -NN 20 times. We report the average and standard deviation of the mean 3-D positioning error ( $\epsilon_{3D}$ ) and the floor hit rate ( $\zeta$ ). The baseline is the combination of AE-ELM using random orthogonal weight initialization, sigmoid (S), and tansig (T) AFs with  $k$ -NN.

In general, the AE-ELM with FID weight initialization method has a good performance reducing the positioning error by more than 1 m in the best case (MAN 2 and TUT 2) in comparison with the baseline. Likewise, the floor hit rate is

TABLE IV  
PERFORMANCE COMPARISON OF AE-ELM AND  $k$ -NN WITH SIX WEIGHT INITIALIZATIONS AND TWO AFS

Dataset	AF	Orthogonal		He		Glorot		Gaussian		Uniform		ELM-Input		FID	
		$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\epsilon_{3D}$ [m]	$\zeta$ [%]
DSI 1	S	5.2±0.5	100.0	5.6±0.3	100.0	5.5±0.6	100.0	4.9±0.3	100.0	5.6±0.5	100.0	26.4±7.9	100.0	<b>4.6±0.0</b>	100.0
	T	5.4±0.4	100.0	6.1±0.6	100.0	6.0±0.7	100.0	5.6±0.8	100.0	5.9±0.7	100.0	26.6±7.9	100.0	<b>4.7±0.1</b>	100.0
DSI 2	S	5.0±0.3	100.0	5.5±0.5	100.0	5.9±0.8	100.0	5.7±1.0	100.0	5.4±0.5	100.0	31.0±7.3	100.0	<b>4.4±0.1</b>	100.0
	T	5.3±0.4	100.0	6.3±0.7	100.0	6.3±1.2	100.0	5.8±0.8	100.0	5.9±1.3	100.0	31.4±9.2	100.0	<b>4.8±0.2</b>	100.0
LIB 1	S	2.8±0.1	99.8±0.3	2.9±0.2	99.4±0.6	2.9±0.2	99.2±0.9	3.4±0.3	99.2±0.7	2.9±0.2	99.6±0.3	5.8±1.1	80.8±13.1	<b>2.5±0.0</b>	<b>99.9±0.0</b>
	T	3.0±0.2	99.6±0.4	2.9±0.3	99.5±0.7	2.9±0.2	99.2±0.9	2.9±0.2	99.2±0.8	3.1±0.2	99.6±0.3	5.4±1.3	79.8±15.3	<b>2.6±0.1</b>	<b>99.8±0.1</b>
LIB 2	S	5.4±0.6	85.0±11.0	6.4±0.8	72.0±15.3	5.9±1.0	79.6±17.3	5.9±0.9	80.6±16.0	5.6±0.6	83.6±14.4	6.6±1.2	66.6±12.9	<b>3.3±0.0</b>	<b>98.7±0.1</b>
	T	5.6±0.8	80.2±13.8	6.6±1.2	71.4±18.8	6.2±0.8	77.2±14.4	5.4±0.5	85.1±11.0	5.8±0.9	77.6±15.2	6.4±0.8	71.4±13.3	<b>3.7±0.2</b>	<b>99.2±0.3</b>
MAN 1	S	3.7±0.8	100.0	4.0±0.6	100.0	4.2±0.8	100.0	10.9±1.5	100.0	5.1±1.4	100.0	6.2±2.9	100.0	<b>2.6±0.1</b>	100.0
	T	4.7±1.8	100.0	4.5±1.1	100.0	4.2±0.9	100.0	6.5±2.0	100.0	6.0±1.9	100.0	9.3±3.6	100.0	<b>2.8±0.4</b>	100.0
MAN 2	S	3.8±0.8	100.0	6.0±3.1	100.0	5.0±1.7	100.0	6.0±1.5	100.0	5.2±1.9	100.0	9.7±5.4	100.0	<b>3.0±0.3</b>	100.0
	T	6.3±1.9	100.0	8.6±6.0	100.0	8.9±5.8	100.0	7.6±4.9	100.0	7.4±3.6	100.0	17.3±6.4	100.0	<b>3.6±1.3</b>	100.0
TUT 1	S	11.6±1.0	81.5±3.8	11.8±0.6	79.5±3.0	11.5±0.6	80.9±3.2	11.8±1.0	82.7±2.5	11.9±1.0	80.1±3.7	17.1±4.6	73.6±7.8	<b>11.2±0.3</b>	<b>84.7±0.4</b>
	T	12.1±1.0	79.3±3.9	12.0±0.9	78.9±2.9	11.5±0.8	79.5±4.4	12.0±1.0	79.8±3.3	12.0±0.9	80.8±2.9	16.3±2.0	72.9±5.4	<b>11.8±1.5</b>	<b>83.1±1.6</b>
TUT 2	S	20.3±2.5	66.2±7.0	27.0±3.5	57.4±4.3	24.2±2.7	58.6±5.8	19.3±1.7	67.2±6.8	19.4±2.5	67.8±7.0	61.5±10.5	30.8±9.4	<b>13.3±0.2</b>	<b>73.0±0.6</b>
	T	21.0±3.8	61.7±6.6	28.5±3.4	54.4±4.6	27.7±3.4	56.1±4.3	20.7±2.5	65.2±6.8	20.2±2.9	62.8±6.0	60.5±8.7	32.0±7.3	<b>17.3±1.2</b>	<b>66.6±1.1</b>
TUT 3	S	9.7±0.1	89.8±0.4	10.1±0.1	88.4±0.7	10.0±0.1	88.8±0.7	12.5±0.4	82.8±1.1	9.8±0.1	89.9±0.5	22.5±17.2	69.9±17.9	<b>9.0±0.0</b>	<b>90.5±0.1</b>
	T	9.7±0.2	89.6±0.5	10.5±0.2	87.6±0.8	10.4±0.2	87.7±0.8	9.8±0.1	89.4±0.7	9.8±0.2	89.7±0.6	20.6±12.0	69.5±13.6	<b>9.2±0.1</b>	<b>89.9±0.2</b>
TUT 4	S	6.3±0.1	94.8±0.5	6.9±0.2	94.3±0.6	6.8±0.2	94.3±0.4	<b>6.1±0.1</b>	<b>95.3±0.2</b>	6.4±0.1	94.7±0.4	22.5±17.5	69.6±21.3	6.5±0.0	95.2±0.1
	T	<b>6.4±0.1</b>	94.7±0.4	7.2±0.2	93.9±0.4	7.1±0.2	94.0±0.4	<b>6.4±0.1</b>	<b>94.9±0.4</b>	<b>6.4±0.1</b>	94.7±0.5	24.3±13.5	66.0±17.2	6.9±0.1	94.5±0.2
TUT 5	S	24.5±7.0	64.3±8.2	28.5±15.1	63.9±9.4	28.8±9.1	62.5±11.6	22.4±6.2	66.6±8.6	29.9±11.8	59.9±12.8	21.2±10.5	64.2±10.7	<b>12.3±0.3</b>	<b>71.8±2.1</b>
	T	30.1±13.6	59.9±9.1	<b>20.2±8.2</b>	<b>65.8±8.2</b>	27.2±9.3	63.0±11.8	28.6±9.5	60.0±9.9	29.5±12.6	60.1±9.9	23.4±11.7	59.4±11.9	30.8±10.8	55.8±10.0
TUT 6	S	2.8±0.1	99.9±0.0	3.0±0.1	99.9±0.1	2.9±0.0	99.9±0.0	11.5±0.7	92.9±2.0	2.8±0.1	99.9±0.0	4.6±2.9	98.7±3.5	<b>2.6±0.0</b>	<b>100.0±0.0</b>
	T	3.0±0.2	99.9±0.1	3.1±0.1	99.8±0.1	3.1±0.1	99.8±0.1	3.2±0.3	99.8±0.5	3.1±0.1	99.8±0.1	4.8±3.5	98.2±5.5	<b>2.8±0.1</b>	<b>100.0±0.0</b>
TUT 7	S	4.2±0.1	98.0±0.1	4.2±0.1	98.0±0.1	4.1±0.1	98.1±0.1	5.7±0.3	96.8±0.4	4.0±0.1	98.1±0.2	18.6±18.0	81.1±22.7	<b>3.7±0.0</b>	<b>98.6±0.0</b>
	T	4.0±0.1	98.2±0.2	4.3±0.1	97.9±0.1	4.2±0.1	97.9±0.1	4.1±0.1	98.1±0.1	4.1±0.1	98.1±0.1	14.1±13.3	85.3±13.4	<b>3.9±0.0</b>	<b>98.3±0.1</b>
Average	S	8.10	90.72	9.38	88.68	9.05	89.38	9.70	89.55	8.77	90.28	19.52	9.64	<b>6.08</b>	<b>93.26</b>
	T	8.97	89.47	9.29	88.40	9.67	88.80	9.12	90.12	9.17	89.48	20.03	79.58	<b>8.07</b>	<b>91.32</b>

Best values are in bold.

TABLE V  
MEAN 3-D POSITIONING ERROR—SVM, GBR, AND BR

Dataset	Orthogonal			He			Glorot			Gaussian			Uniform			ELM-Input			FID		
	SVM	GBR	BR	SVM	GBR	BR	SVM	GBR	BR	SVM	GBR	BR	SVM	GBR	BR	SVM	GBR	BR	SVM	GBR	BR
	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$	$\epsilon_{3D}$
DSI1	12.51	16.82	22.83	14.25	21.83	20.24	15.16	14.30	32.84	13.73	14.60	15.10	11.13	14.44	12.04	38.09	15.30	81.99	7.85	10.88	9.50
DSI2	10.42	10.77	11.04	13.48	15.61	21.90	14.50	23.20	17.55	11.54	11.24	16.04	11.24	9.59	16.16	25.59	22.55	29.32	8.60	13.06	11.15
LIB1	3.42	3.73	3.22	4.05	5.94	4.67	4.63	4.34	8.11	3.26	3.37	3.21	3.13	3.20	4.12	6.79	6.89	19.09	3.79	2.72	3.41
LIB2	5.94	5.03	6.32	5.63	5.79	5.71	5.87	7.01	7.95	5.12	5.61	7.96	6.19	7.59	6.91	5.28	6.88	17.98	5.97	4.83	13.87
MAN1	5.02	4.73	6.69	5.94	4.71	10.80	5.33	3.91	7.20	6.76	7.95	156.90	6.60	4.63	15.93	19.18	16.44	31.11	4.38	3.69	6.52
MAN2	6.56	7.33	21.04	9.94	4.53	24.11	5.34	7.86	41.66	6.43	13.83	145.23	5.05	4.42	12.40	15.96	18.84	108.39	4.36	4.12	7.41
TUT1	13.14	17.96	28.07	13.90	22.99	32.71	12.01	15.14	44.80	14.28	15.59	22.11	13.32	16.19	20.74	16.88	21.32	60.96	25.27	12.88	39.06
TUT2	26.31	29.78	40.29	27.02	29.52	46.73	29.27	32.91	48.03	25.16	28.80	36.02	28.02	31.18	35.58	55.93	47.42	116.43	23.21	20.02	25.74
TUT3	14.38	13.82	18.72	15.88	15.20	28.08	16.41	15.79	22.59	15.77	14.91	18.25	14.66	14.35	16.96	22.53	18.31	37.80	14.15	13.16	15.77
TUT4	14.93	15.03	21.35	15.93	16.61	36.00	15.94	15.71	44.75	14.12	14.55	19.57	16.38	15.03	20.07	34.52	23.90	89.03	16.05	11.03	16.14
TUT5	23.55	32.76	47.00	21.41	24.09	31.91	26.81	27.47	39.19	36.74	35.46	1252.61	24.31	41.77	108.53	27.54	29.19	69.80	29.86	26.98	34.91
TUT6	11.30	8.32	24.38	12.18	7.89	25.01	12.09	7.43	39.48	11.83	7.78	21.29	10.89	8.29	22.77	15.33	10.29	121.03	11.86	7.76	21.17
TUT7	13.14	10.35	21.43	13.03	9.48	37.38	12.71	11.09	46.39	12.56	12.19	23.43	13.60	10.59	23.36	30.17	17.23	147.74	12.84	10.08	17.04
Avg.	<b>12.36</b>	13.57	20.95	13.28	14.17	25.02	13.54	14.32	30.81	13.64	14.30	133.67	12.66	13.94	24.27	24.14	19.58	71.59	12.94	<b>10.86</b>	<b>17.05</b>

improved by more than 10% in many datasets when using FID weight initialization. FID also has the lowest variability for both metrics in most of the cases.

However, there are some cases where there is a marginal improvement compared to the baseline (random orthogonal initialization). In addition, in TUT 4, the lowest positioning error and floor hit rate are obtained using random orthogonal, uniform, or Gaussian initialization. Despite the individual occurrences, the proposed weight initialization FID provides a robust and reliable method over a large array of state-of-the-art initialization while reducing the variance of the positioning result caused by the training process of the neural network. The lowest positioning error was achieved in 12 out of 13 datasets when using FID initialization.

Table V shows the mean 3-D positioning error obtained using the reduced datasets and three different regression algorithms. As can be observed from this table, AE-ELM + FID outperform AE-ELM using other weight initialization methods when GBR and BR are used to estimate the user or device position. Nevertheless, when we use this combination with the SVM regressor algorithm, the mean positioning error is slightly worse than using orthogonal and/or uniform weight initialization methods (5% approximately). Although GBR algorithm and AE-ELM + FID provide good performance, the mean positioning error is still higher than combining AE-ELM + FID and  $k$ -NN.

Despite the fact that the FID weight initialization provides a better reconstruction of the encoded data (see Table III), the positioning error estimated is not always the best. This is the case

TABLE VI  
PERFORMANCE OF 1-NN, STATE-OF-THE ART IPS, AND AE-ELM IN TERMS OF POSITIONING ERROR, FLOOR HIT RATE, AND COMPRESSION RATE

Dataset	1-NN			AE-ELM + 1-NN			AE-ELM + $k$ -NN			CNNLoc – A			CNNLoc – R			OutFin + $k$ -NN		
	$\epsilon_{3D}$ [m]	$\zeta$ [%]	$\bar{\epsilon}_{3D}$ [-]	CR [m]	$\bar{\epsilon}_{3D}$ [-]	$\zeta$ [%]	CR [-]	$\bar{\epsilon}_{3D}$ [-]	$\zeta$ [%]	CR [-]	$\bar{\epsilon}_{3D}$ [-]	$\zeta$ [%]	CR [-]	$\bar{\epsilon}_{3D}$ [-]	$\zeta$ [%]	CR [-]	$\bar{\epsilon}_{3D}$ [-]	$\zeta$ [%]
DSI 1	4.97	100.00	1	3.41×	1.09	100.00	3.41×	0.92	100.00	–	1.70	100.00	–	1.84	100.00	3.41×	1.30	100.00
DSI 2	4.96	100.00	1	3.65×	1.13	100.00	3.65×	0.89	100.00	–	1.82	100.00	–	1.83	100.00	3.65×	1.63	100.00
LIB 1	3.01	99.84	1	5.61×	1.03	99.88	5.61×	0.84	99.92	–	1.38	98.97	–	1.40	99.55	5.61×	1.43	93.40
LIB 2	4.18	97.63	1	4.38×	0.92	98.27	4.38×	0.79	98.73	–	0.93	98.08	–	0.90	99.65	4.38×	2.38	51.19
MAN 1	2.84	100.00	1	1.87×	1.15	100.00	1.87×	0.92	100.00	–	1.87	100.00	–	1.02	100.00	1.87×	2.04	100.00
MAN 2	2.47	100.00	1	3.50×	1.35	100.00	3.50×	1.23	100.00	–	1.30	100.00	–	1.50	100.00	3.50×	3.56	100.00
TUT 1	9.55	90.00	1	6.18×	1.20	84.29	6.18×	1.18	84.71	–	1.25	89.59	–	1.30	86.74	6.18×	1.03	87.96
TUT 2	15.11	71.02	1	6.44×	0.83	78.98	6.44×	0.88	73.04	–	1.23	89.21	–	1.25	92.05	6.44×	1.29	60.80
TUT 3	9.58	91.42	1	8.78×	1.00	91.12	8.78×	0.94	90.54	–	1.77	91.65	–	1.90	90.89	8.78×	1.19	86.97
TUT 4	6.40	95.41	1	5.39×	1.06	93.97	5.39×	1.01	95.16	–	1.54	94.26	–	1.70	94.69	5.39×	1.03	94.12
TUT 5	6.92	88.39	1	81.50×	1.77	71.41	81.50×	1.78	71.80	–	1.81	98.57	–	2.77	98.98	81.50×	1.22	89.61
TUT 6	2.08	99.96	1	12.30×	1.24	100.00	12.30×	1.24	99.99	–	4.62	99.84	–	5.56	99.77	12.30×	1.19	99.85
TUT 7	2.62	99.12	1	6.91×	1.14	98.82	6.91×	1.42	98.57	–	3.64	98.03	–	6.46	96.80	6.91×	1.08	98.88
Average	5.75	94.83	1	11.53×	1.15	93.60	11.53×	1.08	93.27	–	1.91	96.79	–	2.26	96.85	11.53×	1.57	89.44

of TUT 1, TUT 2, and TUT 5, whose positioning error and floor hit rate are not outstanding (see Table IV). It is worth mentioning that the authors of those datasets applied data preprocessing so that the original fingerprints were averaged in cell grids. However, in the vast majority of cases, the proposed method not only reduces the positioning error but also provides a more stable output according to the standard deviation of the results.

#### 4) Comparison With State-of-The-Art Positioning Models:

The previous sections demonstrated that FID is the best-performing weight initialization method for fingerprinting based on AE-ELM and  $k$ -nearest neighbors ( $k$ -NN). Further experiments compare it to the plain  $k$ -NN and two state-of-the-art AE-based localization/positioning algorithms CNNLoc [43], [80], and OutFin [79]. The results of the comparison are provided in Table VI.

Although the number of features was significantly reduced in most datasets, the positioning error was not highly affected after applying the AE-ELM. The proposed AE-ELM seems to get rid off those APs that do not contribute to positioning. For instance, the positioning error increased by only 15% on average after reducing the dimensionality of the datasets, considering that the same  $k$ -NN configuration and data representation were used in both cases (*simple configuration*). In LIB 2 and TUT 2, the normalized positioning error was below 1, which means that reducing the dimensionality improved their performance.

When the AE-ELM is combined with the *best configuration* setting of  $k$ -NN, as described in [73], the positioning error was reduced by 7% compared with AE-ELM + 1-nearest neighbors (1-NN). The lowest compression ratio was 1.87 for dataset MAN 1, where the positioning error was reduced by approximately 7% compared with the plain  $k$ -NN and 23% compared with AE-ELM + 1-NN, improving the positioning performance while reducing the dataset size. That database was, along with MAN 2, already having the lowest number of APs (see Table II). The highest compression ratio was 81.5 for TUT 5 dataset, but its positioning error increased by more than 70% compared to the 1-NN. In general, the dataset dimensionality was reduced 11.5 times on average, at the expense of a small increment in the positioning error.

When considering the floor hit rate, the results show a slight decrease in the accuracy when the dimensionality of the datasets was reduced. The mean accuracy was reduced from 94.83% to

93.60% for *simple configuration* and to 93.27% when using the *best configuration*. In particular, there is one dataset (TUT 5) where the floor hit rate was strongly affected, reducing its floor hit rate accuracy by almost 20% from 88.39% to 71.41% when using *simple configuration* and to 71.80% when using the *best configuration*. In TUT 5, the authors averaged the fingerprints in areas of 25 m<sup>2</sup>. In LIB 1–2, TUT 2, and TUT 6, the floor hit rate slightly increased, whereas in the remained multifloor datasets, the floor hit rate slightly decreased.

The performance of CNNLoc is evaluated with two training optimizers, Adam (CNNLoc – A) and RMSProp (CNNLoc – R), as done in [43]. In general, CNNLoc provides an outstanding general floor hit rate with both optimizers, with special emphasis on TUT 5, a challenging dataset. However, the positioning error is significantly worse than the 1-NN baseline, except for just one dataset, LIB 2. It is worth mentioning that CNNLoc results are obtained without doing any additional dimensional reduction. Overall, the results show the excellent capability of the neural network to classify the data, while accurate regression is significantly harder to achieve. Comparing the CNNLoc–A with AE-ELM + FID weight initialization +  $k$ -NN (*simple configuration*), the CNNLoc is more accurate than AE-ELM + FID weight initialization +  $k$ -NN in the floor hit rate by more than 3%. However, our proposal outperforms CNNLoc–A by almost 40%. The AE provided in [79] was optimized for a dataset. Here, the bottleneck layer is modified to have the same compression ratio as the AE-ELM for each dataset, enabling the direct comparison between OutFin and AE-ELM. The average positioning error for OutFin +  $k$ -NN is around 57% higher than the baseline, while the floor hit rate is 5% worse than 1-NN. In general, OutFin is also worse than AE-ELM in terms of positioning error and floor hit rate, except for TUT 1 and TUT 7, where OutFin provides a lower mean positioning error and a higher floor hit rate.

## VI. DISCUSSION

The ELM has gained popularity since its inception in 2003, given its fast learning speed and its stable performance. Thus, the ELM was used in several applications where a fast response is a must. Generally, the AE-ELM network uses orthogonal random initialization in both the input weights and bias terms, offering a good generalization as was shown in the literature.

The initialization of the input weights and bias terms are highly relevant to provide a better performance of the network and in the case of AE-ELM to reconstruct the encoded data (decode) with low error. Ideally, the output of the decoder should be identical to the input data of the encoder. In Section III-C, we proposed a new method to initialize the input weights, which is based on orthogonal components of the input data. This method offers lower reconstruction error than *orthogonal random input*, *random uniform*, *HE*, *Glorot*, *Gaussian*, and *ELM-Input* initialization, as demonstrated in Section V-B.

The proposed FID performs better than random-based weight initialization methods in most cases. Nevertheless, if the number of hidden nodes is less than  $\approx 2\%$  of the original dataset, the error might not be lower than while using a random weight initialization. Still, the proposed initialization offers a more stable output (see Fig. 7). It is important to highlight that *unit norm* normalization allows the model to constrain the input weights, preserving its statistical characteristics.

In general, the FID weight initialization provides a better performance when hyperbolic tangent sigmoid and sigmoid AFs are used in the AE-ELM. On the contrary, periodic functions (sine and cosine) reduce the capability of the network to extract meaningful information from the dataset. This effect is more evident when a random Gaussian distribution is used to initialize input weights in AE-ELM networks.

In terms of positioning, the AE-ELM approach is providing an outstanding compression rate while the positioning error and floor hit rate are only slightly worse. In contrast, two state-of-the-art positioning methods (CNNLoc and OutFin) may not be appropriate for all datasets. In terms of computational load, the analytic training of the AE-ELM with FID is usually faster than training neural networks with Backpropagation-based algorithms (CNNLoc), while the inference time (operational/online phase) is usually slow in those models based on  $k$ -NN. The number of fingerprints in the radio map may be a critical feature while training a neural network with backpropagation-based algorithms and while inferring the position estimates with a matching model based on  $k$ -NN, where the computation of the distances may not scale. Recent works have shown that the efficiency of fingerprinting based on  $k$ -NN can be further improved by applying clustering models.

However, it seems that the proposed FID works better with unprocessed datasets. Additional data preprocessing, such as averaging fingerprints in a medium/large area, should not be applied in combination with data dimensionality reduction. This may be the cause of the poor results obtained for TUT 5 datasets with AE-ELM. Also, the improvement in the positioning and reconstruction errors provided by FID is at the cost of an increment in the execution time, as the time for initializing the input weights is significantly higher. Nevertheless, the time needed for initialization is negligible when considering the time required for training the model.

## VII. CONCLUSION

ML is widely spread in the positioning community for efficient and accurate indoor position estimation. In this article,

we analyze the relevance of input weight initialization in the AE-ELM for dimensionality reduction in fingerprint-based indoor positioning. Due to the complexity of fingerprinting radio maps, we also propose a new method to initialize the input weights, FID, that helps to learn the complex patterns of radio propagation. The AE-ELM is used in combination with  $k$ -NN to provide the final position estimates. Discussion is based on the results of 13 public Wi-Fi fingerprinting datasets for indoor positioning.

We have performed a comparison of different initialization models, including random orthogonal, Gaussian, Glorot, He, uniform, ELM-input, and the proposed FID. The results have shown that AE-ELM with FID provides a lighter representation in terms of dimensions and reduces the reconstruction error of the encoded data by up to 10% less than the traditional initialization models, i.e., a correct initialization is of utmost relevance when designing a light model based on AE-ELM.

Later, the positioning system based on AE-ELM with FID and  $k$ -NN was compared to a baseline and the state-of-the-art fingerprinting models (CNNLoc and OutFin). The results show that the proposed AE-ELM with FID significantly outperforms CNNLoc and OutFin in terms of the positioning error, being similar to the baseline. CNNLoc remains as the most accurate model for coarse floor-level localization. On average, AE-ELM with FID provides a dimensionality reduction of more than 11 times with respect to CNNLoc and the baseline while keeping the positioning accuracy at most 66% lower than the DNN benchmark.

Overall, FID method offers several advantages over some well-known weight initialization methods/algorithms and some of them are listed as follows.

- 1) FID can be used with different AFs without affecting the performance of the neural network.
- 2) It demonstrates robustness to different data representations, outperforming state-of-the-art methods.
- 3) FID ensures a consistent output by not relying on random numbers.
- 4) It enhances the performance of ELM networks.

Future work will focus on extending this weight initialization method to deep neural networks in order to analyze its advantages and disadvantages. In such a way, FID can be optimized for different neural networks and not limited to ELM. In addition, the optimized AE-ELM will be combined with other neural networks to provide fast learning models that can be used in real-time indoor positioning applications.

## REFERENCES

- [1] S. Mazuelas et al., "Robust indoor positioning provided by real-time RSSI values in unmodified WLAN networks," *IEEE J. Sel. Topics Signal Process.*, vol. 3, no. 5, pp. 821–831, Oct. 2009.
- [2] P. Pivato, L. Palopoli, and D. Petri, "Accuracy of RSS-based centroid localization algorithms in an indoor environment," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 10, pp. 3451–3460, Oct. 2011.
- [3] C. Pendão and A. Moreira, "Fastgraph enhanced: High accuracy automatic indoor navigation and mapping," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 1027–1045, Mar. 2021.
- [4] M. T. Hoang et al., "A soft range limited  $k$ -nearest neighbors algorithm for indoor localization enhancement," *IEEE Sensors J.*, vol. 18, no. 24, pp. 10208–10216, Dec. 2018.

- [5] J. Zuo, S. Liu, H. Xia, and Y. Qiao, "Multi-phase fingerprint map based on interpolation for indoor localization using iBeacons," *IEEE Sensors J.*, vol. 18, no. 8, pp. 3351–3359, Apr. 2018.
- [6] Q. Wan, X. Duan, Y. Yu, R. Chen, and L. Chen, "Self-calibrated multi-floor localization based on Wi-Fi ranging/crowdsourced fingerprinting and low-cost sensors," *Remote Sens.*, vol. 14, no. 21, 2022. [Online]. Available: <https://www.mdpi.com/2072-4292/14/21/5376>
- [7] I. Silva, C. Pendão, and A. Moreira, "Real-world deployment of low-cost indoor positioning systems for industrial applications," *IEEE Sensors J.*, vol. 22, no. 6, pp. 5386–5397, Mar. 2022.
- [8] N. Fu, J. Zhang, W. Yu, and C. Wang, "Crowdsourcing-based wifi fingerprint update for indoor localization," in *Proc. ACM Turing 50th Celebration Conf. - China*, New York, NY, USA: Association for Computing Machinery, May 2017, Art. no. 34. [Online]. Available: <https://doi.org/10.1145/3063955.3063989>
- [9] W. Kim, S. Yang, M. Gerla, and E.-K. Lee, "Crowdsourced based indoor localization by uncalibrated heterogeneous Wi-Fi devices," *Mobile Inf. Syst.*, vol. 2016, pp. 1–18, Jan. 2016.
- [10] A. Abusara, M. S. Hassan, and M. H. Ismail, "RSS fingerprints dimensionality reduction in WLAN-based indoor positioning," in *Proc. Wireless Telecommun. Symp.*, 2016, pp. 1–6.
- [11] A. Abed and I. Abdel-Qader, "RSS-fingerprint dimensionality reduction for multiple service set identifier-based indoor positioning systems," *Appl. Sci.*, vol. 9, no. 15, 2019. [Online]. Available: <https://www.mdpi.com/2076-3417/9/15/3137>
- [12] J. Wang, X. Wang, J. Peng, J. G. Hwang, and J. G. Park, "Indoor fingerprinting localization based on fine-grained CSI using principal component analysis," in *Proc. 12th Int. Conf. Ubiquitous Future Netw.*, 2021, pp. 322–327.
- [13] L. Ma, Y. Zhang, and D. Qin, "A novel indoor fingerprint localization system based on distance metric learning and AP selection," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–15, Nov. 2022.
- [14] J. Jang and S. Hong, "Indoor localization with WiFi fingerprinting using convolutional neural network," in *Proc. 10th Int. Conf. Ubiquitous Future Netw.*, 2018, pp. 753–758.
- [15] C. Shen, S. Zhang, J. Zhai, D. Luo, and J. Chen, "Imbalanced data classification based on extreme learning machine autoencoder," in *Proc. Int. Conf. Mach. Learn. Cybern.*, 2018, pp. 399–404.
- [16] A. Gupta, P. Manikumar, R. Reddy, and A. K. Bhattacharya, "Extreme learning machines with frequency based noise filtering for prediction of critical digressions in a noisy industrial process," in *Proc. IEEE 14th India Council Int. Conf.*, 2017, pp. 1–6.
- [17] L. L. C. Kasun, Y. Yang, G.-B. Huang, and Z. Zhang, "Dimension reduction with extreme learning machine," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3906–3918, Aug. 2016.
- [18] H. Gan, M. H. B. M. Khir, G. Witjaksono Bin Djaswadi, and N. Ramli, "A hybrid model based on constraint OSELM, adaptive weighted SRC and KNN for large-scale indoor localization," *IEEE Access*, vol. 7, pp. 6971–6989, 2019.
- [19] Y. Zhong, Q. Xie, S. Zhao, and X. Luo, "WiFi location method based on TSNE-KNN," in *Proc. Int. Conf. Image Video Process., Artif. Intell.*, 2019, Art. no. 113212X.
- [20] A. Alitalessi, H. Jazayeriy, and J. Kazemitabar, "EA-CNN: A smart indoor 3D positioning scheme based on Wi-Fi fingerprinting and deep learning," *Eng. Appl. Artif. Intell.*, vol. 117, 2023, Art. no. 105509. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0952197622004997>
- [21] Z. Ezzati, A. Hajihoseini, and A. Ghorashi, "A fingerprint method for indoor localization using autoencoder based deep extreme learning machine," *IEEE Sensors Lett.*, vol. 2, no. 1, Mar. 2018, Art. no. 6000204.
- [22] G. JunLin, Z. Xin, W. HuaDeng, and Y. Lan, "WiFi fingerprint positioning method based on fusion of autoencoder and stacking mode," in *Proc. Int. Conf. Culture-Oriented Sci. Technol.*, 2020, pp. 356–361.
- [23] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Commun. Surv. Tuts.*, vol. 21, no. 3, pp. 2568–2599, Thirdquarter 2019.
- [24] G. W. Samu and P. Kadam, "Survey on indoor localization: Evaluation performance of bluetooth low energy and fingerprinting based indoor localization system," *Int. J. Comput. Eng. Technol.*, vol. 8, no. 6, pp. 23–35, 2017.
- [25] L. Batistić and M. Tomic, "Overview of indoor positioning system technologies," in *Proc. 41st Int. Conv. Inf. Commun. Technol. Electron. Microelectron.*, 2018, pp. 0473–0478.
- [26] T. J. Chowdhury, C. Elkin, V. Devabhaktuni, D. B. Rawat, and J. Oluoch, "Advances on localization techniques for wireless sensor networks: A survey," *Comput. Netw.*, vol. 110, pp. 284–305, 2016.
- [27] S. Subedi and J.-Y. Pyun, "Practical fingerprinting localization for indoor positioning system by using beacons," *J. Sensors*, vol. 2017, 2017.
- [28] Y. Duan et al., "Data rate fingerprinting: A WLAN-based indoor positioning technique for passive localization," *IEEE Sensors J.*, vol. 19, no. 15, pp. 6517–6529, Aug. 2019.
- [29] X. Huang, S. Guo, Y. Wu, and Y. Yang, "A fine-grained indoor fingerprinting localization based on magnetic field strength and channel state information," *Pervasive Mobile Comput.*, vol. 41, pp. 150–165, 2017.
- [30] G. Pecoraro, S. Di Domenico, E. Cianca, and M. De Sanctis, "CSI-based fingerprinting for indoor localization using LTE signals," *EURASIP J. Adv. Signal Process.*, vol. 2018, no. 1, 2018.
- [31] L. Wirolo, L. Wirolo, and R. Piché, "Bandwidth and storage reduction of radio maps for offline WLAN positioning," in *Proc. Int. Conf. Indoor Positioning Indoor Navigation*, 2013, pp. 1–9.
- [32] S. Hamdan, A. Awaian, and S. Almajali, "Compression techniques used in IoT: A comparative study," in *Proc. 2nd Int. Conf. New Trends Comput. Sci.*, 2019, pp. 1–5.
- [33] A. Arya, P. Godlewski, and P. Mellé, "A hierarchical clustering technique for radio map compression in location fingerprinting systems," in *Proc. IEEE 71st Veh. Technol. Conf.*, 2010, pp. 1–5.
- [34] J.-H. Seong and D.-H. Seo, "Wi-Fi fingerprint using radio map model based on MDLP and Euclidean distance based on the chi squared test," *Wireless Netw.*, vol. 25, no. 6, pp. 3019–3027, 2019.
- [35] L. Klus, D. Quezada-Gaibor, J. Torres-Sospedra, E. S. Lohan, C. Granell, and J. Nurmi, "RSS fingerprinting dataset size reduction using feature-wise adaptive k-means clustering," in *Proc. 12th Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops*, 2020, pp. 195–200.
- [36] J. Talvitie, M. Renfors, and E. S. Lohan, "Novel indoor positioning mechanism via spectral compression," *IEEE Commun. Lett.*, vol. 20, no. 2, pp. 352–355, Feb. 2016.
- [37] J. Talvitie, M. Renfors, M. Valkama, and E. S. Lohan, "Method and analysis of spectrally compressed radio images for mobile-centric indoor localization," *IEEE Trans. Mobile Comput.*, vol. 17, no. 4, pp. 845–858, Apr. 2018.
- [38] F. Yang, L. Herranz, J. van de Weijer, J. A. I. Guitián, A. M. López, and M. G. Mozerov, "Variable rate deep image compression with modulated autoencoder," *IEEE Signal Process. Lett.*, vol. 27, pp. 331–335, Jan. 2020.
- [39] M. Roy, S. K. Bose, B. Kar, P. K. Gopalakrishnan, and A. Basu, "A stacked autoencoder neural network based automated feature extraction method for anomaly detection in on-line condition monitoring," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2018, pp. 1501–1507.
- [40] S. Ryu, H. Choi, H. Lee, and H. Kim, "Convolutional autoencoder based feature extraction and clustering for customer load analysis," *IEEE Trans. Power Syst.*, vol. 35, no. 2, pp. 1048–1060, Mar. 2019.
- [41] A. Pepe and H. Fu, "Three-level security communication of multimedia data via hybrid sparse autoencoder-parallel compressive sensing," in *Opto-Electron. Commun. Conf.*, 2020, pp. 1–3.
- [42] H. Ye, L. Liang, and G. Y. Li, "Circular convolutional auto-encoder for channel coding," in *Proc. IEEE 20th Int. Workshop Signal Process. Adv. Wireless Commun.*, 2019, pp. 1–5.
- [43] X. Song et al., "CNNLoc: Deep-learning based indoor localization with WiFi fingerprinting," in *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov.*, 2019, pp. 589–595.
- [44] J. Liu, N. Liu, Z. Pan, and X. You, "AutLoc: Deep autoencoder for indoor localization with RSS fingerprinting," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process.*, 2018, pp. 1–6.
- [45] S. Ding, H. Zhao, Y. Zhang, X. Xu, and N. Ru, "Extreme learning machine: Algorithm, theory and applications," *Artif. Intell. Rev.*, vol. 44, pp. 489–501, Jun. 2013.
- [46] S. Ding, N. Zhang, X. Xu, L. Guo, and J. Zhang, "Deep extreme learning machine and its application in eeg classification," *Math. Problems Eng.*, vol. 2015, pp. 1–11, May 2015.
- [47] H. Nuha, A. Balghonaim, B. Liu, M. Mohandes, M. Deriche, and F. Fekri, "Deep neural networks with extreme learning machine for seismic data compression," *Arabian J. Sci. Eng.*, vol. 45, pp. 1367–1377, May 2019.
- [48] X. Lu, H. Zou, H. Zhou, L. Xie, and G. Huang, "Robust extreme learning machine with its application to indoor positioning," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 194–205, Jan. 2016.
- [49] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S09525231206000385>

- [50] G.-B. Huang, "Learning capability and storage capacity of two-hidden-layer feedforward networks," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 274–281, Mar. 2003.
- [51] K. Sun, J. Zhang, C. Zhang, and J. Hu, "Generalized extreme learning machine autoencoder and a new deep neural network," *Neurocomputing*, vol. 230, pp. 374–381, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S092523121631503X>
- [52] H. Ge, W. Sun, M. Zhao, and Y. Yao, "Stacked denoising extreme learning machine autoencoder based on graph embedding for feature representation," *IEEE Access*, vol. 7, pp. 13433–13444, 2019.
- [53] H. Li et al., "A comparison of weight initializers in deep learning-based side-channel analysis," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur.*, 2020, pp. 126–143.
- [54] M. V. Narkhede, P. P. Bartakke, and M. S. Sutaone, "A review on weight initialization strategies for neural networks," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 291–322, 2022.
- [55] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *J. Mach. Learn. Res. - Proc. Track*, vol. 9, pp. 249–256, Jan. 2010.
- [56] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," *CoRR*, vol. abs/1502.01852, 2015. [Online]. Available: <http://arxiv.org/abs/1502.01852>
- [57] E. Romero, "Benchmarking the selection of the hidden-layer weights in extreme learning machines," in *Proc. Int. Joint Conf. Neural Netw.*, 2017, pp. 1607–1614.
- [58] J. F. L. de Oliveira and T. B. Ludermir, "An evolutionary extreme learning machine based on fuzzy fish swarms," in *Proc. Int. Conf. Artif. Intell.*, 2012, p. 1.
- [59] L. D. Pacifico and T. B. Ludermir, "Evolutionary extreme learning machine based on particle swarm optimization and clustering strategies," in *Proc. Int. Joint Conf. Neural Netw.*, 2013, pp. 1–6.
- [60] K. Javed, R. Gouriveau, and N. Zerhouni, "SW-ELM: A summation wavelet extreme learning machine algorithm with a priori parameter initialization," *Neurocomputing*, vol. 123, pp. 299–307, 2014.
- [61] A. M. Saxe, J. L. McClelland, and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," 2014.
- [62] L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, and L. Shao, "Normalization techniques in training DNNs: Methodology, analysis and application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 10173–10196, Feb. 2023, doi: [10.1109/TPAMI.2023.3250241](https://doi.org/10.1109/TPAMI.2023.3250241).
- [63] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 901–909.
- [64] D. Quezada-Gaibor et al., "Supplementary material 'auto-encoder extreme learning machine for fingerprint-based positioning: A good weight initialization is decisive,'" 2023. [Online]. Available: [https://github.com/darwinquezada/elm\\_fid\\_ips](https://github.com/darwinquezada/elm_fid_ips)
- [65] J. Torres-Sospedra, D. Quezada-Gaibor, G. Mendoza-Silva, J. Nurmi, Y. Koucheryavy, and J. Huerta, "Supplementary materials for 'new cluster selection and fine-grained search for k-means clustering and Wi-Fi fingerprinting,'" Zenodo, Jun. 2020, doi: [10.5281/zenodo.3751042](https://doi.org/10.5281/zenodo.3751042).
- [66] A. Razavi, M. Valkama, and E. Lohan, "K-means fingerprint clustering for low-complexity floor estimation in indoor mobile localization," in *Proc. IEEE Globecom Workshops*, 2015, pp. 1–7.
- [67] A. Cramariuc, H. Huttunen, and E. S. Lohan, "Clustering benefits in mobile-centric WiFi positioning in multi-floor buildings," in *Proc. Int. Conf. Localization GNSS*, 2016, pp. 1–6.
- [68] E.-S. Lohan, J. Torres-Sospedra, H. Leppäkoski, P. Richter, Z. Peng, and J. Huerta, "Wi-Fi crowdsourced fingerprinting dataset for indoor positioning," *Data*, vol. 2, no. 4, Oct. 2017, Art. no. 32. [Online]. Available: <https://www.mdpi.com/2306-5729/2/4/32>
- [69] A. Moreira, I. Silva, and J. Torres-Sospedra, "The DSI dataset for Wi-Fi fingerprinting using mobile devices," Zenodo, Apr. 2020, doi: [10.5281/zenodo.3778646](https://doi.org/10.5281/zenodo.3778646).
- [70] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg, "CRAWDAD dataset Mannheim/compass (v. 2008-04-11)," Apr. 2008. [Online]. Available: <https://crawdad.org/mannheim/compass/20080411>
- [71] T. King, T. Haenselmann, and W. Effelsberg, "On-demand fingerprint selection for 802.11-based positioning systems," in *Proc. Int. Symp. a World Wireless, Mobile Multimedia Netw.*, 2008, pp. 1–8.
- [72] G. M. Mendoza-Silva, P. Richter, J. Torres-Sospedra, E. S. Lohan, and J. Huerta, "Long-term WiFi fingerprinting dataset for research on robust indoor positioning," *Data*, vol. 3, no. 1, 2018.
- [73] J. Torres-Sospedra et al., "A comprehensive and reproducible comparison of clustering and optimization rules in Wi-Fi fingerprinting," *IEEE Trans. Mobile Comput.*, vol. 21, no. 3, pp. 769–782, Mar. 2022.
- [74] N. Saccomanno, A. Brunello, and A. Montanari, "What you sense is not where you are: On the relationships between fingerprints and spatial knowledge in indoor positioning," *IEEE Sensors J.*, vol. 22, no. 6, pp. 4951–4961, Mar. 2022.
- [75] J. Torres-Sospedra et al., "Towards ubiquitous indoor positioning: Comparing systems across heterogeneous datasets," in *Proc. Int. Conf. Indoor Positioning Indoor Navigation*, 2021, pp. 1–8.
- [76] G. G. Anagnostopoulos and A. Kalousis, "Can I trust this location estimate? Reproducibly benchmarking the methods of dynamic accuracy estimation of localization," *Sensors*, vol. 22, no. 3, 2022. [Online]. Available: <https://www.mdpi.com/1424-8220/22/3/1088>
- [77] A. Brunello, A. Montanari, and N. Saccomanno, "A genetic programming approach to WiFi fingerprint meta-distance learning," *Pervasive Mobile Comput.*, 2022, Art. no. 101681. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1574119222000980>
- [78] J. Torres-Sospedra, R. Montoliu, S. Trilles, O. Belmonte, and J. Huerta, "Comprehensive analysis of distance and similarity measures for Wi-Fi fingerprinting indoor positioning systems," *Expert Syst. Appl.*, vol. 42, no. 23, pp. 9263–9278, 2015, doi: [10.1016/j.eswa.2015.08.013](https://doi.org/10.1016/j.eswa.2015.08.013).
- [79] F. Alhomayani and M. H. Mahoor, "Outfin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach," *Sci. Data*, vol. 8, 2021.
- [80] X. Song et al., "CNNLoc: Deep-learning based indoor localization with WiFi fingerprinting," Jul. 2019. [Online]. Available: <https://github.com/XudongSong/CNNLoc-Access.git>
- [81] F. Alhomayani and M. H. Mahoor, "OutFin, a multi-device and multi-modal dataset for outdoor localization based on the fingerprinting approach," Oct. 2020. [Online]. Available: [https://figshare.com/articles/dataset/OutFin\\_a\\_multi-device\\_and\\_multi-modal\\_dataset\\_for\\_outdoor\\_localization\\_based\\_on\\_the\\_fingerprinting\\_approach/12069993](https://figshare.com/articles/dataset/OutFin_a_multi-device_and_multi-modal_dataset_for_outdoor_localization_based_on_the_fingerprinting_approach/12069993)
- [82] J. Tapson, P. de Chazal, and A. van Schaik, "Explicit computation of input weights in extreme learning machines," in *Proc. ELM-2014*, Cham, Switzerland: Springer International Publishing, vol. 1, Dec. 2014, pp. 41–49.



**Darwin P. Quezada Gaibor** received the bachelor's degree in mechatronic engineering from Universidad Tecnológica América, Quito, Ecuador, in 2013, and the master's degree in radioengineering—GNSS receivers: Hardware and software from Samara National Research University, Samara, Russia, in 2017. He is currently working toward the Ph.D. degree in indoor positioning and cloud computing with Universitat Jaume I, Castellón de la Plana, Spain, and Tampere University, Tampere, Finland.

His main interests include voice over Internet protocol, Cloud Computing, networking, servers, and open-source software.



**Lucie Klus** (Student Member, IEEE) received the master's degree in electronics and communication from the Faculty of Electrical Engineering and Communication, Brno University of Technology, Brno, Czechia, in 2019.

She has been an Early Stage Researcher of A-WEAR project in joint Doctoral degree programme with Tampere University, Tampere, Finland, and Jaume I University, Castellón de la Plana, Spain, since 2019. Her current research topic aims to improve the efficiency of data

transfer and storage of the wearable-originated data. Her research interests include modern approaches in wireless communications, data analytics, optimization, and machine learning techniques.



**Roman Klus** (Student Member, IEEE) received the Ing. degree (Czech equivalent of master) in electronics and communications from the Brno University of Technology, Brno, Czechia, in 2019.

He is a Doctorate Researcher with Tampere University, Tampere, Finland. His research interests include modern machine learning approaches in 5G and beyond networks, especially utilizing neural network structures in mobility management and positioning.



**Mikko Valkama** (Fellow, IEEE) received the M.Sc. (Tech.) degree in electrical engineering and D.Sc. (Tech.) degree in information technology, both with honors, from the Tampere University of Technology, Tampere, Finland, in 2000 and 2001, respectively.

In 2003, he was with the Communications Systems and Signal Processing Institute, San Diego State University, San Diego, CA, USA, as a Visiting Research Fellow. He is currently a Full Professor and the Head with the Unit of Electrical Engineering, Tampere University, Tampere. His research interests include radio communications, radio localization, and radio-based sensing, with particular emphasis on 5G and 6G mobile radio networks.



**Elena Simona Lohan** (Senior Member, IEEE) received the M.Sc. degree in electrical engineering from the Polytechnics University of Bucharest, Bucharest, Romania, in 1997, the DEA degree (French equivalent of master) in econometrics from Ecole Polytechnique, Paris, France, in 1998, and the Ph.D. degree in telecommunications from the Tampere University of Technology, Tampere, Finland, in 2003.

She is currently a Full Professor with the Electrical Engineering Unit, Tampere University, Tampere, and the Coordinator of the MSCA EU A-WEAR European Joint Doctorate network. Her current research interests include global navigation satellite system, low Earth orbit positioning, navigation, and timing, seamless wireless location techniques, wearable computing, and privacy-aware localization.



**Joaquín Huerta** received the B.Sc. and M.Sc. degrees in computer science from the Polytechnic University of Valencia, Valencia, Spain, and the Ph.D. degree in computer engineering from Universitat Jaume I, Castellón de la Plana, Spain, in 1999.

He is a Full Professor in computer science with the UJI, where he is also the Deputy Director with the Institute of New Imaging Technologies, the Director of the research group GEOTEC, and the co-Director of the Erasmus Mundus Master of Science in Geospatial Technologies. His research interests include geographic information systems, indoor location, digital twins, and augmented reality.



**Jari Nurmi** received the D.Sc. (Tech.) degree in information technology from Tampere University of Technology, Finland, in 1994.

From 1987 to 1994, he held various research, education, and management positions with Tampere University (formerly Tampere University of Technology, TUT), Tampere, Finland, and was the Vice President of the SME VLSI Solution Oy, from 1995 to 1998. He has been working as a Professor with Electrical Engineering Unit, Tampere University, since 1999. He

is working on embedded computing systems, system-on-chip, approximate computing, software-defined radio and networks, wireless localization, and positioning receiver implementations.

Prof. Nurmi is a Member of the Technical Committee on VLSI Systems and Applications, IEEE Circuits and Systems Society, an Associate Editor/handling Editor of three international journals, the Director of national DELTA doctoral training network of 200 Ph.D. students, a Coordinator of APROPOS European doctoral training network, and the Head of A-WEAR European joint Ph.D. program with Tampere University.



**Joaquín Torres-Sospedra** received the Ph.D. degree from Universitat Jaume I, Castellón de la Plana, Spain, in 2011.

He is currently an MSCA Postdoctoral Fellow with the University of Minho, Guimarães, Portugal, where he works on Indoor Positioning (Wi-Fi, BLE, VLC, and Machine Learning) for Industrial applications. He has authored more than 160 articles in journals and conferences, and has supervised 16 master's and two Ph.D. students. He is currently supervising six Ph.D.

students.

Dr. Torres-Sospedra is the Chair of the Indoor Positioning and Indoor Navigation (IPIN) International Standards Committee and IPIN Smartphone-Based Off-Site Competition.