Comparing Machine Learning Algorithms for RSS-Based Localization in LPWAN

Chapter · January 2020

DOI: 10.1007/978-3-030-33509-0_68

3 authors:

Thomas Janssen
University of Antwerp
14 PUBLICATIONS **240** CITATIONS
SEE PROFILE

Raf Berkvens
University of Antwerp - imec
93 PUBLICATIONS **1,935** CITATIONS
SEE PROFILE

Maarten Weyn
University of Antwerp / iMinds
142 PUBLICATIONS **2,331** CITATIONS
SEE PROFILE

Comparing Machine Learning Algorithms for RSS-based localization in LPWAN

Thomas Janssen, Rafael Berkvens and Maarten Weyn

Abstract In smart cities, a myriad of devices is connected via Low Power Wide Area Networks (LPWAN) such as LoRaWAN. There is a growing need for location information about these devices, especially to facilitate managing and retrieving them. Since most devices are battery-powered, we investigate energy-efficient solutions such a Received Signal Strength (RSS)-based fingerprinting localization. For this research, we use a publicly available dataset of 130 426 LoRaWAN fingerprint messages. We evaluate ten different Machine Learning algorithms in terms of location accuracy, R^2 score, and evaluation time. By changing the representation of the RSS data in the most optimal way, we achieve a mean location estimation error of 340 m when using the Random Forest regression method. Although the k Nearest Neighbor (kNN) method leads to a similar location accuracy, the computational performance decreases compared to the Random Forest regressor.

1 Introduction

Locating a device in smart cities is becoming an increasingly challenging problem. The amount of Internet of Things (IoT) devices connected to Low Power Wide Area Networks (LPWAN) is forcing network operators to improve the scalability of their networks. Furthermore, these mobile devices are typically powered by a small battery that needs to last for several years. Sensors reporting air quality and smart water level meters are just a few examples of the growing need to locate devices throughout the city.

LPWANs are being used as an alternative to ordinary Global Navigation Satellite System (GNSS) receivers, which consume a significant amount of power. Besides, satellite-based solutions are not always desired, given their limitations in indoor

Thomas Janssen, Rafael Berkvens and Maarten Weyn
University of Antwerp - imec, The Beacon, Sint-Pietersvliet 7, 2000 Antwerp, Belgium, e-mail: {thomas.janssen,rafael.berkvens,maarten.weyn}@uantwerpen.be

environments, i.e. signals not penetrating well through walls. Sigfox, LoRaWAN and NB-IoT are the most commonly used LPWAN technologies [10]. While the latter operates in the licensed spectrum with low latency, Sigfox and LoRaWAN benefit from a longer range and battery life [7].

Several approaches exist to locate a transmitting device in an LPWAN. In every approach, a trade-off must be made between location accuracy and energy consumption. However, when comparing different studies of the same approach, several other parameters need to be considered. For example, the cost and effort to train a model or install equipment needs to be taken into account. Moreover, the indoor or outdoor environment and the amount of receiving gateways also plays a significant role in the resulting localization accuracy [8]. For Time Difference of Arrival (TDoA) and Angle of Arrival (AoA) approaches, the gateways and antennas need to be synchronized, respectively. Several state-of-the-art TDoA algorithms are compared in [6]. TDoA-based positioning and tracking with LoRaWAN are topics discussed in [9]. In this paper, we focus on Received Signal Strength-based (RSS) fingerprinting.

Outdoor RSS-based fingerprinting localization can be challenging, given the time and effort needed to create a training database and the dynamic environment of a city. However, Aernouts et al. managed to collect a large amount of RSS samples, together with GPS coordinates as ground truth data, in the city of Antwerp, Belgium [1]. Both Sigfox and LoRaWAN messages were collected. In previous research, we performed outdoor fingerprinting using Sigfox with a basic k Nearest Neighbors (kNN) algorithm [5]. The mean location estimation error was 340 m. Meanwhile, the LoRaWAN dataset size has grown to 13 426 samples. In this research, we want to explore and compare more advanced Machine Learning algorithms using this dataset. In the state-of-the-art, Support Vector Machines (SVM) are being used to classify an RSS fingerprint into a correct GPS node class in Wireless Sensor Networks (WSN) [11], in indoor environments [4] and in simulation environments [14]. Furthermore, several Machine Learning algorithms are evaluated in terms of location accuracy and computation time in an indoor environment [3]. In this research, we evaluate ten different Machine Learning algorithms using real measurement data collected in a city-scale outdoor environment.

The remainder of this chapter is organized in the following way. In section 2, we explain into more detail how the fingerprinting database is created and what preprocessing steps were taken to input the data into the algorithms. Next, the benefits and limitations of each regression algorithm are briefly discussed and we justify how parameter values are chosen. In section 3, we evaluate each algorithm in terms of evaluation time, location estimation error and R^2 score. Section 4 summarizes our main findings and lists the future work.

2 Methodology

In this section, we will describe the steps taken to evaluate every Machine Learning algorithm. First, some preprocessing steps need to be taken in order to represent our

data in the most optimal way. Second, we will explain into detail how each Machine Learning algorithm works and what parameters are optimized. For each algorithm, the benefits and limitations are listed. Third, we define the metrics used to evaluate each algorithm.

2.1 Dataset

The data that is being used to evaluate each algorithm is collected and updated by Aernouts et. al [1]. Version 1.2 of the publicly available dataset consists of a table with 130 426 rows, each row representing a LoRaWAN message sent in the city of Antwerp, Belgium. These messages are plotted in Fig. 1. For each message, the RSS values to all 72 LoRaWAN gateways are represented in the columns, appended by the ground truth GPS coordinate and some meta data. If a message is not received by a gateway, the RSS value in that gateway column is set to -200 dBm. Given the spatial spread of the LoRaWAN gateways, a lot of RSS values are set to -200 dBm. This emphasizes the challenge to represent the data in a proper way and extract the relevant data above the noise floor.

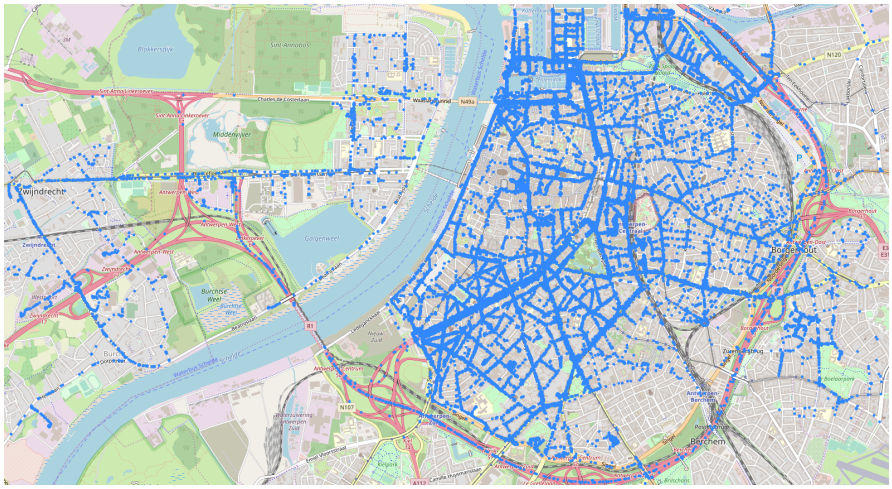

Fig. 1: The dataset consists of 130 426 LoRaWAN messages collected in Antwerp, Belgium. © *2019 OpenStreetMap contributors*

To put this data into a more Machine Learning perspective, each message is related to a single sample and each receiving gateway relates to a feature. Therefore, the dataset consists of 130 426 samples and 72 features in total.

2.2 Preprocessing steps

In Machine Learning, data preprocessing is required to prepare raw, sometimes incomplete data for further processing. Hence, before feeding the Machine Learning algorithms with the dataset, some preprocessing steps are taken.

A first step is to **transform the RSS** data into another format. In the raw data, the RSS values are represented in decibels relative to a milliwatt (dBm). Torres-Sospedra et al. suggest four different RSS representations in order to increase the localization accuracy [13]. The first two representations, positive and normalized, can be seen as lineal transformations of the dataset. The positive representation maps the minimum RSS value to a value of zero, resulting in all positive RSS values. In this way, all -200 dBm values are mapped to a value of zero. In the normalized representation, the positive values are mapped to the range $[0...1]$. Since signal strengths are represented in a logarithmic way, it is better to map the RSS values in a logarithmic way as well. Therefore, the 'exponential' and 'powed' representations are introduced in [13]. In our experiments, we evaluate the localization accuracy when using the normalized, exponential and powed RSS representation.

Although some **scaling** is introduced in the RSS representations of the previous paragraphs, it is recommended to use the `StandardScaler` of the `sklearn` package in Python, which is able to feed the data in the most optimal way to the Machine Learning algorithms in the `sklearn` package.

Finally, a **Principal Component Analysis (PCA)** is performed on the dataset. With PCA, we can extract the most relevant features out of the dataset. This is highly desired, since not all gateways receive each message, thus reducing the amount of features and noise. Moreover, extracting the principal components of the dataset is often used to decrease the evaluation time of each Machine Learning algorithm. We performed PCA on the dataset with 95% of the variance retained, resulting in a reduction from 72 to 40 components.

2.3 Regression algorithms

Since we are predicting continuous-valued output, regression-based algorithms are most suitable for our supervised Machine Learning problem. In total, ten different regression algorithms are evaluated on our dataset. These algorithms can be classified into four different categories. For every category of algorithms, we will explain how each algorithm works and how the parameters are tuned. Finally, the benefits and limitations of each algorithm are discussed. In our experiments, each algorithm is benchmarked in terms of location estimation error as the Vincenty distance between the GPS coordinate and the estimated coordinate; the R^2 score of `sklearn`; and evaluation time as the elapsed time between the fitting of a model and the estimation of an output coordinate, using a Virtual Machine with 32 GB RAM memory and 10 CPU cores, of which 6 are used in parallel in every algorithm.

2.3.1 Linear regression algorithms

Linear regression algorithms attempt to fit a linear function from the provided training data and estimate the numeric output values, given new input values. By fitting the function, a linear Machine Learning model is created. This model can be represented in the form of Eq. (1), where x is the feature vector of length n and w and b are the parameters that need to be learned by training the model.

$$y_{pred} = w[0] * x[0] + w[1] * x[1] + ... + w[n] * x[n] + b \tag{1}$$

Several variations of linear regression algorithms exist. In this research we evaluate the following linear algorithms: Ordinary Least Squares, Ridge, Lasso, Elastic Net, Stochastic Gradient Descent and Polynomial regression.

As the name suggests, Ordinary Least Squares (OLS) is the most basic linear regression algorithm. In this algorithm, the Mean Squared Error (MSE) between the predicted and real output values is minimized. If the features of the data are correlated, the number of random errors in the target values increases. This phenomenon is called multicollinearity. Therefore, the independence of the features is very important in the OLS algorithm.

Ridge regression is similar to OLS, with the difference that in Ridge regression the magnitude of the coefficients w is reduced by a factor α, as can be seen in Eq. (2). This constraint is known as ℓ_2 regularization. We find the optimal value of α by evaluating Ridge regression with cross-validation. During the cross-validation, we iterate over values ranging from 5×10^{-2} to 5×10^{9}. In this way, the optimal value of α is found to be 3290.

$$\min_{w} \|Xw - y\|_2^2 + \alpha \|w\|_2^2 \tag{2}$$

In the Lasso regression algorithm, some coefficients in Eq. (1) are set to zero. Consequently, some features are ignored by the model. This is called ℓ_1 regularization. Similar to Ridge, the optimal value of α equal to 2.52×10^{-5} is found by cross-validation. With only a few non-zero weights, the advantage of the Lasso regression is the reduced amount of time needed to train the model.

The Elastic-Net linear regression model performs both ℓ_1 and ℓ_2-norm regularization of the coefficients. In fact, it is a combination of the Ridge and Lasso algorithms, in the sense that there are fewer non-zero weights and the regularization properties of Ridge are maintained.

Stochastic Gradient Descent (SGD) is a Machine Learning algorithm that can be used for classification and regression problems. It is often used for training artificial neural networks. However, it can also be used for training linear regression models. Gradient Descent is a method to find the values of a function that minimizes the cost function. To find the optimal values, the initial parameters are constantly updated. Thus, Gradient Descent is an iterative method, resulting in slower training times. In the stochastic variant of this algorithm, one iterates over a few randomly selected samples, reducing the computational complexity in large datasets. Furthermore, different loss functions and corresponding parameters have been evaluated

on our dataset. The Huber loss function with ε equal to 1×10^{-2} and a stopping criterion equal to 1×10^{-3} lead to the most accurate results (i.e. smallest location estimation error). The Huber loss function uses a squared loss function and past a distance of ε it uses a linear loss function. Therefore, it becomes less sensitive to outliers in the dataset.

The last linear regression algorithm we discuss is the polynomial regression, which can be seen as a particular case of multiple linear regression. In this algorithm, a linear function is fitted within a higher-dimensional space. Thus, the degree of a polynomial function is lowered. In our case, the localization accuracy was maximized when reducing the degree to 1. This approach allows to fit a much wider range of data and benefits from the relatively fast performance of linear regression algorithms.

2.3.2 Support Vector Regression

Support Vector Machines (SVM) can be used in classification and regression problems, both in linear and multi-dimensional cases. In simple regression algorithms, we attempt to minimize the error rate, while in Support Vector Regression (SVR), the goal is to maximize the margin between the hyperplane and the support vectors (i.e. the data points closest to that hyperplane) In other words, we need to find a function that has at most ε deviation from the actually obtained targets in the training data [2]. Hence, a loss function with a margin of tolerance ε is defined because of the real numbered target values. Furthermore, SVR is characterized by a kernel function that maps lower dimensional data to higher dimensional data. We implemented an SVR with a third-degree polynomial kernel function and free parameters $\varepsilon = 0.01$ and $C = 1000$, determined experimentally. The latter controls the penalty imposed on observations that lie outside the ε margin and helps to prevent overfitting. The main advantage of SVR is that the computational complexity does not depend on the dimensionality of the input space. However, when the number of samples in the dataset exceeds a few tens of thousands, the algorithm can be computationally demanding.

2.3.3 *k* Nearest Neighbors

The *k* Nearest Neighbors algorithm is an intuitive yet effective Machine Learning approach which is often used in indoor and outdoor localization applications [5, 12]. During the offline training phase of the algorithm, the feature vectors and target values are only stored. In the online evaluation phase, we want to estimate the target values (i.e. the GPS coordinates) of a test feature vector. This is done by calculating the distance or similarity between each training vector and the test vector. This can be done by using various distances [13, 5]. As a final step, the centroid of the target values of the *k* smallest distances is used as the estimate for the target values, where *k* is user-defined. In our experiments, we iterate over *k* ranging from 1 to 20, thus

optimizing the amount of nearest neighbors. In the weighted variant of the *k*NN algorithm, neighbors that are closer to the test sample have a greater influence than neighbors which are farther away. The benefits of *kNN* are the fact that there is no explicit training phase, the simplicity of the algorithm and the variety of distances to choose from. On the contrary, the user-defined value of k, the computational complexity and the high outlier sensitivity are the limitations of the algorithm.

2.3.4 Random Forest

Random Forest (RF) is an ensemble technique, i.e. multiple Machine Learning algorithms are combined to solve classification or regression problems. The general idea is to construct multiple decision trees during the training phase and output the mean of all individual predictions as an estimated target value. The technique randomly samples the training observations when building the trees. Random Forest has been proven to outperform other Machine Learning algorithms in terms of indoor fingerprinting localization accuracy [15]. Despite the computational complexity of each individual decision tree, the overall training and matching speeds are very fast, even for high-dimensional input data. Finally, since multiple algorithms are combined, overfitting is reduced significantly and the stability of the technique increases. In our implementation, we chose 100 estimators that are combined using the bootstrap aggregation (bagging) technique in `sklearn`. From this amount of trees on, the benefit in prediction performance from learning from more trees did not make up anymore for the computational cost to learn these additional trees.

3 Results

The ten algorithms discussed in the previous section are now evaluated in terms of location estimation error, R^2 score and the time needed for the Virtual Machine with 6 CPU cores to compute the results. In Table 1, these metrics are summarized for every algorithm and for the lineal, exponential and powed representations of the fingerprinting dataset. A box plot of localization errors for every algorithm using the powed RSS representation is shown in Fig. 2.

As one can observe, the optimal RSS representation depends on the algorithm being evaluated. For example, the exponential RSS representation yields the smallest location estimation errors when evaluating the linear regression algorithms. By using this representation, the accuracy of all linear algorithms is around 785 m. Given their simplicity, the linear regression algorithms yield higher errors but are computed in the least amount of time. The R^2 score, which gives an indication of how close the actual target values are to the fitted regression line, increases from 0.70 using positive RSS to 0.73 using exponential RSS.

Out of all evaluated algorithms, the Support Vector Regression seems to be the least performing, both in terms of R^2 score and evaluation time. The increased eval-

Table 1: Mean location estimation error (in [m]), R^2, and evaluation time (in [s]) score for every Machine Learning algorithm using the lineal, exponential and powed RSS representation of the LoRaWAN dataset.

Algorithm	Lineal RSS			Exponential RSS			Powed RSS		
	Error	R^2	Time	Error	R^2	Time	Error	R^2	Time
kNN	354	0.90	131	345	0.90	517	349	0.90	131
kNN weighted	348	0.90	146	344	0.90	484	343	0.90	147
SVR	1148	0.57	1168	1206	0.49	1616	1155	0.55	1162
Linear OLS	801	0.70	1.19	785	0.73	1.15	786	0.72	1.08
Linear Ridge	800	0.70	0.82	785	0.73	0.84	785	0.72	0.83
Linear Lasso	801	0.70	0.88	785	0.73	0.90	786	0.72	0.98
Linear Elastic Net	801	0.70	0.94	785	0.73	0.92	786	0.72	0.98
Linear SGD	799	0.70	11	784	0.73	11	784	0.72	12
Linear Polynomial	801	0.70	1.20	785	0.73	1.20	786	0.72	1.29
Random Forest	351	0.91	56	609	0.84	56	340	0.91	53

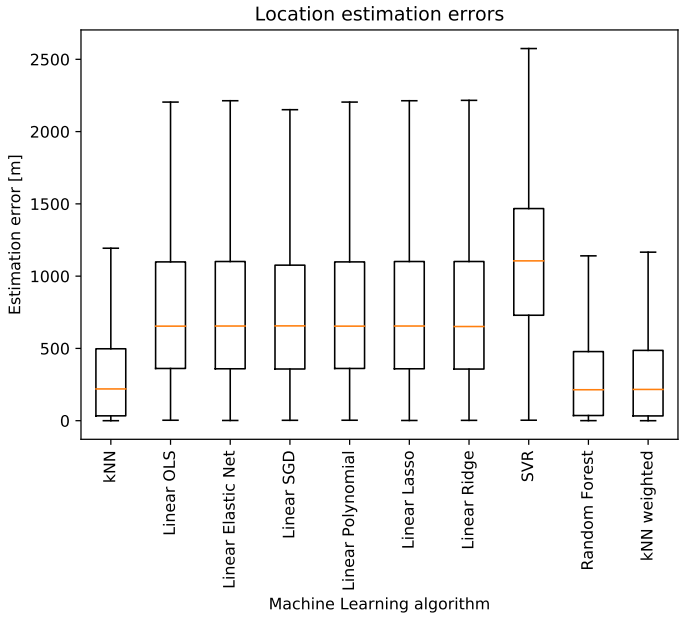

Fig. 2: Location estimation errors for every regression algorithm using the powed RSS representation of the LoRaWAN dataset

uation time is caused by the use of kernels, which is very time consuming. Additionally, the training complexity in SVR is highly dependent on the number of samples in the dataset. Hence, taking into account the mean localization error of over 1 km, SVR is not a good choice to evaluate with our large dataset.

In contrast, the (weighted) kNN and Random Forest algorithms yield the best results regarding localization accuracy and R^2 score. The weighted variant of the kNN algorithm is slightly more accurate than the basic version, leading to a mean location estimation error of 343 m using powed RSS and $k = 15$. Similarly, the Random Forest ensemble technique results in an accuracy of 340 m, using powed RSS as well. While it takes 147 seconds to compute the results in the kNN algorithm, Random Forest takes advantage of its bagging technique, reducing the computation time to 53 s.

4 Conclusion & Future work

In this work, we evaluated ten different Machine Learning algorithms on a dataset with 130000 samples, consisting of RSS values to 72 LoRaWAN base stations. Given the GPS coordinates as ground truth data, the objective was to compare different Machine Learning algorithms that locate the transmitting device based on this RSS data. To optimize the input of every algorithm, we changed the representation of the RSS data, performed scaling and PCA analysis on the dataset as preprocessing steps.

We evaluated every algorithm based on location estimation error, total evaluation time and R^2 score. Since the fit time complexity of SVR is more than quadratic with the amount of samples, the algorithm is unable to scale to datasets with more than a couple of 10000 samples, resulting in significantly higher evaluation times. Despite having different benefits and limitations, all six linear regression algorithms yield similar results. Fitting a linear model is fast but inaccurate, resulting in a location errors around 785 meters. Weighted kNN and Random Forest achieve the highest localization accuracy. Mean location estimation errors of around 340 meters are achieved when the RSS values are transformed to the powed representation. The Random Forest ensemble technique successfully avoids overfitting by averaging the predictions of multiple decision trees, while still being able to compute the results faster than the kNN algorithm, due to the bagging technique.

Our future work consists of further improving the dataset representation by preprocessing the dataset using a Deep Learning approach. Afterwards, the algorithms used in this work as well as more complicated Neural Network algorithms can be evaluated. Finally, we will create a coverage map of our test environment in Antwerp, in order to further improve the localization accuracy.

Acknowledgements Thomas Janssen is funded by the Fund For Scientific Research (FWO) Flanders under grant number 1S03819N.

References

1. Aernouts, M., Berkvens, R., Van Vlaenderen, K., Weyn, M.: Sigfox and LoRaWAN Datasets for Fingerprint Localization in Large Urban and Rural Areas. Data **3**(2) (2018). URL http://www.mdpi.com/2306-5729/3/2/13
2. Awad, M., Khanna, R.: Support Vector Regression. In: Efficient Learning Machines, pp. 67–80. Apress, Berkeley, CA (2015). DOI 10.1007/978-1-4302-5990-9{_}4. URL http://link.springer.com/10.1007/978-1-4302-5990-9_4
3. Bozkurt, S., Elibol, G., Gunal, S., Yayan, U.: A comparative study on machine learning algorithms for indoor positioning. In: 2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA), pp. 1–8. IEEE (2015). DOI 10.1109/INISTA.2015.7276725. URL http://ieeexplore.ieee.org/document/7276725/
4. Farjow, W., Chehri, A., Hussein, M., Fernando, X.: Support Vector Machines for indoor sensor localization. In: 2011 IEEE Wireless Communications and Networking Conference, pp. 779–783. IEEE (2011). DOI 10.1109/WCNC.2011.5779231. URL http://ieeexplore.ieee.org/document/5779231/
5. Janssen, T., Aernouts, M., Berkvens, R., Weyn, M.: Outdoor Fingerprinting Localization Using Sigfox. In: 2018 International Conference on Indoor Positioning and Indoor Navigation (Accepted). Nantes, France (2018)
6. Jin, B., Xu, X., Zhang, T., Jin, B., Xu, X., Zhang, T.: Robust Time-Difference-of-Arrival (TDOA) Localization Using Weighted Least Squares with Cone Tangent Plane Constraint. Sensors **18**(3), 778 (2018). DOI 10.3390/s18030778. URL http://www.mdpi.com/1424-8220/18/3/778
7. Mekki, K., Bajic, E., Chaxel, F., Meyer, F.: A comparative study of LPWAN technologies for large-scale IoT deployment. ICT Express **5**(1), 1–7 (2019). DOI 10.1016/j.icte.2017.12.005. URL https://doi.org/10.1016/j.icte.2017.12.005
8. Plets, D., Podevijn, N., Trogh, J., Martens, L., Joseph, W.: Experimental Performance Evaluation of Outdoor TDoA and RSS Positioning in a Public LoRa Network. IPIN 2018 - 9th International Conference on Indoor Positioning and Indoor Navigation (September), 24–27 (2018). DOI 10.1109/IPIN.2018.8533761
9. Podevijn, N., Plets, D., Trogh, J., Martens, L., Suanet, P., Hendrikse, K., Joseph, W.: TDoA-Based Outdoor Positioning with Tracking Algorithm in a Public LoRa Network. Wireless Communications and Mobile Computing **2018**, 1–9 (2018). DOI 10.1155/2018/1864209. URL https://www.hindawi.com/journals/wcmc/2018/1864209/
10. Raza, U., Kulkarni, P., Sooriyabandara, M.: Low Power Wide Area Networks: An Overview. IEEE Communications Surveys and Tutorials **19**(2), 855–873 (2017). DOI 10.1109/COMST.2017.2652320
11. Sallouha, H., Chiumento, A., Pollin, S.: Localization in long-range ultra narrow band IoT networks using RSSI. In: 2017 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2017). DOI 10.1109/ICC.2017.7997195
12. Song, Q., Guo, S., Liu, X., Yang, Y.: CSI Amplitude Fingerprinting-Based NB-IoT Indoor Localization. IEEE Internet of Things Journal **5**(3), 1494–1504 (2018). DOI 10.1109/JIOT.2017.2782479. URL https://ieeexplore.ieee.org/document/8187642/
13. Torres-Sospedra, J., Montoliu, R., Trilles, S., Belmonte, ., Huerta, J.: Comprehensive analysis of distance and similarity measures for Wi-Fi fingerprinting indoor positioning systems. Expert Systems with Applications **42**(23), 9263–9278 (2015). DOI 10.1016/J.ESWA.2015.08.013. URL https://www.sciencedirect.com/science/article/pii/S0957417415005527
14. Tran, D., Nguyen, T.: Localization In Wireless Sensor Networks Based on Support Vector Machines. IEEE Transactions on Parallel and Distributed Systems **19**(7), 981–994 (2008). DOI 10.1109/TPDS.2007.70800. URL http://ieeexplore.ieee.org/document/4384476/
15. Wang, Y., Xiu, C., Zhang, X., Yang, D.: WiFi Indoor Localization with CSI Fingerprinting-Based Random Forest. Sensors 2018, Vol. 18, Page 2869 **18**(9), 2869 (2018). DOI 10.3390/S18092869. URL http://www.mdpi.com/1424-8220/18/9/2869