Reducing the Complexity of Fingerprinting-Based Positioning using Locality-Sensitive Hashing

Larry Tang, Ramina Ghods, and Christoph Studer

School of Electrical and Computer Engineering, Cornell University, Ithaca, NY
email: {lwt29, rg548, studer}@cornell.edu; website: vip.ece.cornell.edu

***Abstract*—Localization of wireless transmitters based on fingerprinting finds growing use in indoor as well as outdoor scenarios. Fingerprinting localization first builds large databases containing channel state information (CSI) associated with measured location information. One then searches for the most similar CSI in this database to approximate the position of wireless transmitters. In this paper, we investigate the efficacy of locality-sensitive hashing (LSH) to reduce the complexity of the nearest neighbor-search (NNS) underlying most fingerprinting localization systems. We propose low-complexity and memory efficient LSH functions based on the sum-to-one (STOne) transform and use approximate hash matches, which enables low-complexity fingerprinting localization with the same accuracy as methods relying on exact NNS. We evaluate the accuracy and complexity (in terms of the number of searches and storage requirements) of our methods for line-of-sight (LoS) and non-LoS channels.**

I. Introduction

Localization of wireless transmitters in indoor and outdoor scenarios has received growing interest over the past decades. Conventional methods for outdoor localization are mainly based on triangulation or trilateration methods, which map time-of-flight (ToF), angle-of-arrival, or received signal strength (RSS) features to a specific location using geometric models [1]–[4]. All of these methods require line-of-sight (LoS) connectivity to multiple basestations, access points, or satellites. A popular instance of such methods are global navigation-satellite systems (GNSS) that rely on ToF measurements and geometric models. In situations that lack LoS connectivity (e.g., indoor scenarios) or systems in which GNSS is unavailable (e.g., ultra-low-power sensors), alternative positioning methods are required.

A. Localization using CSI-Fingerprinting

Location fingerprinting is a prominent method to enable wireless positioning in challenging propagation environments, such as non-LoS scenarios or channels with complex multi-path propagation; see, e.g., [2], [5]–[8] and the references therein. The principle of location fingerprinting is as follows. In a first phase, one tabulates a large number of measured channel-state information (CSI), which includes RSS [6], [7] or other measured channel features in the time or frequency domain [9]–[11] or at multiple receive antennas [12], with associated location information in a given area—the resulting CSI-fingerprint/location tuple is stored in a database. In a second phase, a new CSI-fingerprint is extracted from a wireless transmitter and the most similar CSI fingerprints in the database are retrieved. The stored locations associated with the nearest fingerprints are then used to generate an estimate of the transmitter's location. While a nearest neighbor search (NNS) in the CSI-fingerprint database can provide a simple (often accurate) estimate of the transmitter's location[1], the complexity of NNS in large fingerprinting databases can quickly become a computational complexity bottleneck.

More recently, localization approaches based on deep neural networks have been proposed in [15]–[17]. Such methods replace the NNS step and directly map measured CSI-fingerprints to location, which requires a large number of CSI measurements at fine resolution in space (often of the order of a few wavelengths) in order to train these neural networks. Furthermore, the large number of network parameters can easily be of the same order as the CSI-fingerprint/location database.

B. Contributions

In this paper, we reduce the complexity of the NNS step in traditional fingerprinting localization by using locality-sensitive hashing (LSH) [18]–[21]. We design a computationally efficient LSH function that builds upon the sum-to-one (STOne) transform [22] and use approximate hash matches that reduce the complexity of the NNS and the size of the hash tables. We evaluate the accuracy and complexity (in terms of the number of searches and storage requirements) of our method for LoS and non-LoS channels in a massive multiuser (MU) multiple-input multiple-output (MIMO) wireless system. Finally, we compare our approach with neural-network-based localization methods in [15]–[17].

II. Fingerprinting Localization via LSH

We now introduce the principles of fingerprinting localization and summarize the basics of LSH. We then detail our approach for low-complexity LSH with approximate hash lookups.

The work of LT was supported in part by the Cornell University Engineering Learning Initiatives (ELI). The work of RG and CS was supported in part by Xilinx Inc. and the US National Science Foundation under grants ECCS-1408006, CCF-1535897, CCF-1652065, CNS-1717559, and ECCS-1824379. The authors thank E. Lei and O. Castañeda for their help with CSI generation.

[1]More sophisticated methods, such as neural networks [13] or Bayesian methods [14] can be used to improve the nearest neighbor search estimate.

A. Basics of Fingerprinting Localization

Fingerprinting localization proceeds in two phases. In the first phase, CSI-fingerprints $\{\mathbf{f}_n\}_{n=1}^N = \mathcal{F}$ and associated positions $\{\mathbf{x}_n\}_{n=1}^N = \mathcal{X}$ are measured in a given area and stored in a database. Here, the vector $\mathbf{f}_n \in \mathbb{R}^D$ corresponds to a D-dimensional CSI-fingerprint at position $\mathbf{x}_n \in \mathbb{R}^d$, where D is typically high-dimensional and d is either two or three dimensions. Depending on the application, CSI-fingerprints can represent RSSs acquired at multiple receivers, power-delay profiles, angle-of-arrival, and many others; see [2] for a survey.

In the second phase, one wants to estimate the location $\mathbf{x}_{n'}$ of a new transmitting device with index n'. First, a CSI-fingerprint $\mathbf{f}_{n'}$ is extracted. Second, the indices associated with the K most similar fingerprints in the database $\{\mathbf{f}_n\}_{n=1}^N$ are identified

$$\mathcal{N}_K = \{n : \|\mathbf{f}_n - \mathbf{f}_{n'}\| < r, \mathbf{f}_n \in \mathcal{F}, n = 1, \dots, N\},$$

where $r > 0$ ensures that $|\mathcal{N}_K| = K$. One then approximates the location of transmitter n' from the set of similar locations $\mathcal{X}_K = \{\mathbf{x}_n\}_{n \in \mathcal{N}_K}$. If $K = 1$, then one can simply pick the location associated with the nearest channel feature; more sophisticated approaches are discussed in [13], [14].

B. Locality-Sensitive Hashing (LSH)

Finding the K nearest neighbors in a large dataset containing high-dimensional vectors suffers from the curse of dimensionality, which implies that carrying out an exhaustive search results in high complexity [19]. Locality-sensitive hashing (LSH) is a powerful method to perform an approximate nearest neighbor search in such large datasets [18]–[21]. In general, a hash function projects a value from a set with potentially infinite elements to a value from a set with fewer or a finite number of elements. The principle behind LSH is to construct locality-sensitive hash functions for which similar datapoints have matching hashes and dissimilar datapoints have mismatched hashes. More concretely, we are interested in LSH functions $h : \mathbb{R}^D \to \mathcal{S}$ for which the collision probability $\Pr[h(\mathbf{p}) = h(\mathbf{q})] = P_1$ of any two datapoints $\mathbf{p}, \mathbf{q} \in \mathbb{R}^D$ is large if $\|\mathbf{p} - \mathbf{q}\| \leq R$ and $\Pr[h(\mathbf{p}) = h(\mathbf{q})] = P_2$ is small if $\|\mathbf{p} - \mathbf{q}\| \geq cR$ with R being an application-dependent radius and $c > 1$, i.e., we are interested in the case $P_1 > P_2$. Here, the finite-cardinality set $\mathcal{S}$ contains so-called buckets (unique hash values).

Approximate NNS via LSH is carried out in two phases. In the first phase, one computes the hash values for all points in the dataset, i.e., $\{h(\mathbf{f}_n)\}_{n=1}^N$. In the second phase, for a given query point $\mathbf{f}_{n'}$, one computes $h(\mathbf{f}_{n'})$ and compares the resulting hash value to those in the dataset. One can then compare the true, high-dimensional CSI feature distance $\|\mathbf{f}_m - \mathbf{f}_{n'}\|$ associated to *only* those indices $m \in \{n : h(\mathbf{f}_n) = h(\mathbf{f}_{n'}), n = 1, \dots, N\}$ for which there was a collision. By using a set of T distinct and carefully crafted LSH functions (instead of just one function), we can guarantee to find at least one nearest neighbor while often significantly reducing the complexity of approximate NNS, even for very large datasets.

C. Fast LSH via the a Randomized STOne Transform

The literature describes a number of ways to construct LSH functions with the desired properties [19]. In what follows, we are interested in LSH functions that can be computed at low complexity and with low memory footprint with the goal of reducing the complexity of LSH-based location fingerprinting.

Our approach builds upon prominent LSH functions for normalized datapoints [23] and uses methods from [24] and [22] to lower the complexity of these hash functions. In [23], the authors introduce a family of hash functions called cross polytope LSH for applications in which data points are differentiated by angular distance. In cross polytope LSH, we hash a point $\mathbf{x} \in \mathbb{R}^D$ that lies on the unit sphere by computing $y = \mathbf{A}\mathbf{x}/||\mathbf{A}\mathbf{x}||$, where $\mathbf{A} \in \mathbb{R}^{D \times D}$ is a random matrix with i.i.d Gaussian entries. The closest standard basis vector of $\mathbb{R}^D$ to y is then used as the hash of $\mathbf{x}$. To make cross polytope hash functions more practical, [23] suggests using pseudo random rotations. Instead of multiplying an input vector $\mathbf{x}$ by a random Gaussian matrix, we can fake multiply it with $\mathbf{HD}$, where $\mathbf{H}$ is Hadamard and $\mathbf{D}$ is diagonal with ± 1 entries [24].

Concretely, we select a diagonal matrix $\mathbf{D} \in \mathbb{R}^{D \times D}$ in which the diagonal entries are pseudo-random ± 1 values. We also select a pseudo-random index sets $\Omega \subset \{1, 2, \dots, D\}$, where $|\Omega| = L$ is the length of the hash values. For $\mathbf{H}$, we use the sum-to-one (STOne) transform matrix $\mathbf{H} \in \mathbb{R}^{D \times D}$, which is a Hadamard matrix with multi-scale properties [22]. With these three ingredients we design the following LSH function: $h(\mathbf{f}) = [\mathrm{sign}(\mathbf{HDf})]_\Omega$. Here, the operator $[\cdot]_\Omega$ extracts the vector whose entries are associated to the indices in Ω; $\mathrm{sign}(\cdot)$ operates element-wise on vectors. For this LSH function, each hash value (bucket) is in the set $\mathcal{S} = \{-1, +1\}^L$. Since $\tilde{\mathbf{f}} = \mathbf{Df}$ can be computed in linear time and the STOne transform $\mathbf{H}\tilde{\mathbf{f}}$ in $D \log(D)$ time, computing hash values $h(\mathbf{f})$ is efficient and requires only little storage (only for diagonal entries of $\mathbf{D}$ and the subset Ω). To design multiple hash tables, we generate T pseudo-random subsets $\Omega_t \subset \{1, \dots, D\}$, $t = 1, \dots, T$. We then compute T hash functions as $h_t(\mathbf{f}) = [\mathrm{sign}(\mathbf{HDf})]_{\Omega_t}$, $t = 1, \dots, T$, which requires the computation of $\mathrm{sign}(\mathbf{HDf})$ only once.

The STOne transform matrix can be constructed using a 4 by 4 stencil matrix as shown:

$$S_4 = \frac{1}{2} \begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix}$$

We can then form a larger transform matrix by computing the Kronecker product of two stencil matrices, $S_{16} = S_4 \otimes S_4$, which still maintains its sum-to-one property. This STOne transform matrix can then be used on points of larger dimension.

D. LSH with Approximate Matches

Our final ingredient addresses the storage requirements of LSH-based location fingerprinting. Concretely, we declare a match whenever the Hamming distance between $h(\mathbf{p})$ and $h(\mathbf{q})$ is within a given threshold $\delta \geq 0$ (instead of declaring a match

only if the two hash values are equal). Note that for $\delta = 0$, we recover classical LSH. The key advantage of approximate matches (i.e., $\delta > 0$) are that fewer hash tables are sufficient to achieve the same performance (in terms of finding the K nearest neighbors) as conventional LSH. Furthermore, recent advances on in-memory processing enables one to identify approximate hash matches in a hardware efficient manner [25].

III. Results

We now show results of the proposed low-complexity LSH-based fingerprinting localization method. We provide a detailed accuracy/complexity trade-off analysis and a comparison to recent neural-network localization approaches [15]–[17]. Our main goals are as follows: (i) reduce the complexity of searching similar CSI fingerprints in large databases and (ii) reduce the complexity at minimal storage overhead.

A. Simulated Scenario

To evaluate the efficacy of our approach, we consider a massive MU-MIMO-OFDM localization scenario in LoS and non-LoS scenarios with a single basestation containing 32 antennas operating at 2.68 GHz with a bandwidth of 20 MHz and localizing 2000 transmitters distributed uniformly at random in an area of 40,000 m^2; the noisy channel vectors are generated using channel models from [26]. The CSI features are $D = 256$ dimensional (32 antennas and 8 maximally-spaced subcarriers) and correspond to the absolute value of beamspace/delay domain channel vectors as in [16]. We use the $K = 2$ nearest neighbors and average their location; we also use $\delta = 1$. Both of these algorithm choices yield consistently good results. Below, we provide simulation results that confirm this claim. To assess the complexity and storage requirements, we use the following metrics: (i) the fraction of high-dimensional vectors that have been compared relative to an exhaustive search; (ii) the memory area indicating the size of the precomputed LSH tables; and (iii) the total complexity, which is the fraction of comparisons times the memory area.

B. Results

Figure 1 shows results for a LoS and a non-LoS scenario. As a reference, we include the performance of an exhaustive search (denoted by "NN in feat. space") and that of finding the true nearest neighbors in real space (denoted by "NN in real space"). Each curve is parametrized by the number of hash functions T. We see in Fig. 1(a) that increasing the hash length L can reduce the number of comparisons by $10\times$, while achieving the same average distance error as an exhaustive search (indicated by the dashed "NN in feat. space" lines). As shown in Fig. 1(b), we see that increasing L increases the storage requirements for the hash values. This tradeoff between reduced complexity but increased memory area is a result of the increase in the number of hash tables needed but sparser hash buckets. The total complexity is defined as fraction of comparisons times memory area and is shown in Fig. 1(c), which reveals that the best trade-off in terms of fraction of comparisons and memory area is achieved by relatively small hash values, i.e., with $L = 12$ bits. Figures 1(d,e,f) show results for a non-LoS scenario. The trends are analogous to that of the LoS case, which confirms that the proposed method is able to reduce the NNS complexity of LSH-based fingerprinting localization under different propagation conditions.

Figure 2 shows results for the same LoS scenario in which we use different thresholds, δ, for finding approximate matches. We see that increasing δ from 0 to 1 achieves lower average distance error at the same total costs as $\delta = 0$. Increasing δ to 2 only provides a marginal improvement in average distance error, confirming that the $\delta = 1$ threshold is the optimal choice.

C. Comparison to Neural Network

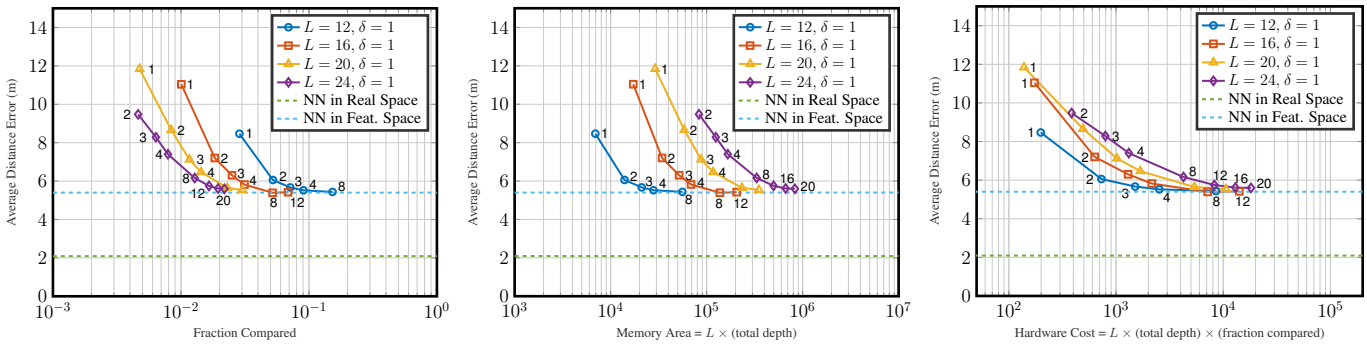
We compare the performance of our LSH-based approach to that of a neural network in terms of search complexity and memory area. We use fully connected neural networks (FCNN) with 6, 5, 4, 3, 2, and 1 layer to compare performance. The six layer FCNN has 512, 256, 128, 64, 32, and 2 activations per layer, and we then remove the layer with the most nodes to form the following smaller networks. Each layer except the final layer uses rectified linear unit (ReLU) activations with the last layer using a linear activation.

To evaluate the search complexity of each approach, we compute the total number of multiplication and addition operations needed to evaluate a single test point. In our LSH-based approach we compute the search complexity using the average number of comparisons needed for a given test point. The memory area for both the LSH-based approach and FCNN is computed in terms of how many numbers need to be stored for each approach.

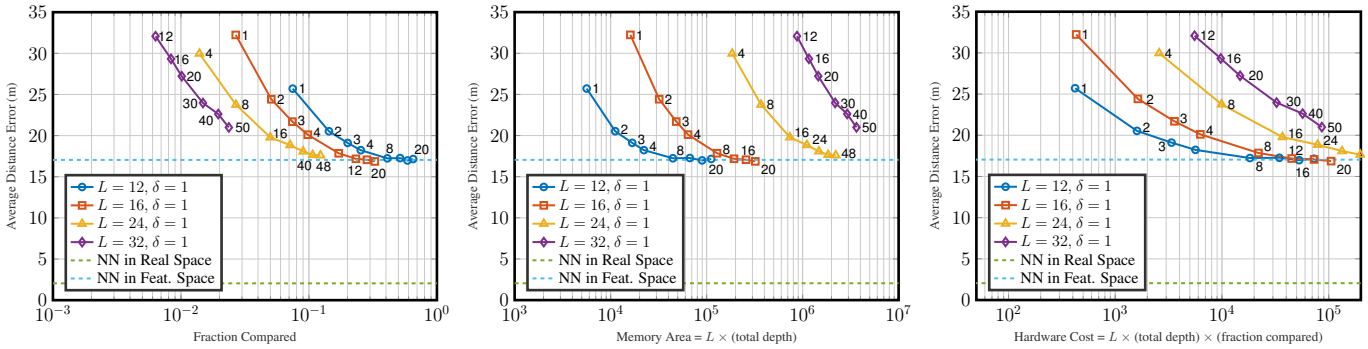
Figures 3(a,b,c,d) show results for a LoS and a non-LoS scenario. As a reference, we again show the performance of an exhaustive search and that of finding true nearest neighbors in real space. For the LoS scenario, Fig. 3(a,b) show that the LSH-based approach can achieve a lower average distance error with lower complexity than the neural network based approach. In terms of memory area, we observe that the FCNN consistently performs better since it will only need to store weight and bias values, whereas the LSH-based approach requires the entire dataset to be stored in memory so that comparisons can be made to points in real space. For the non-LoS scenario, Fig. 3(c,d) show that the LSH-based approach no longer outperforms the neural network in terms of search complexity. The neural network can produce slightly better average distance error with about the same complexity as the LSH-based approach. We see similar trends in memory area as we did in the LoS scenario.

IV. Conclusions and Outlook

We have shown that the complexity of fingerprinting-based localization can be reduced significantly by using locality-sensitive hashing (LSH). We have proposed computationally efficient LSH functions that build upon the sum-to-one (STOne) transform which requires a complexity of only $D\log(D)$, where D is the dimension of the channel-state information (CSI) fingerprint vector. Furthermore, we have shown that approximate hash matches enable one to reduce the storage



(a) Fraction compared; LoS (b) Memory area; LoS (c) Total complexity; LoS



(d) Fraction compared; non-LoS (e) Memory area; non-LoS (f) Total complexity; non-LoS

Fig. 1. Accuracy of LSH-based location fingerprinting for LoS and non-LoS massive MU-MIMO scenarios. "NN in feature space" are the nearest neighbors found through an exhaustive search and "NN in real space" are the true nearest neighbors in real space.

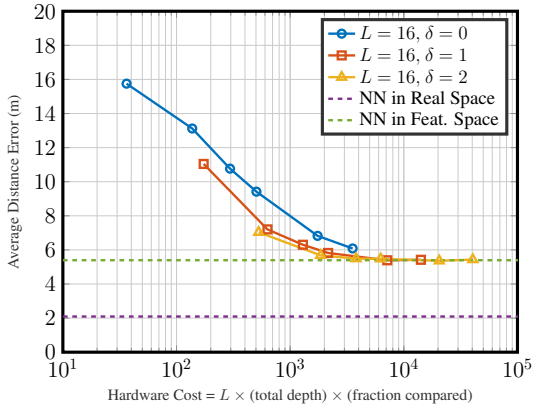


Fig. 2. Total complexity of LSH-based location fingerprinting with approximate matches for a LoS scenario.

requirements of conventional LSH. We have shown that the proposed method achieves the same accuracy as an exhaustive search over the CSI fingerprinting database for line-of-sight (LoS) and non-LoS scenarios—this is a first step towards demonstrating the robustness of our approach. Finally, we have demonstrated that our proposed LSH-based method can outperform neural network-based approaches for a LoS scenario, but does not show an advantage over neural networks for a non-LoS scenario. This implies that the approach with better performance will depend on the channel scenario.

There are many opportunities for future work. The development of CSI features that reduce a location in real space to a single hash bucket in an LSH table can potentially reduce the average distance error. In addition, further improvement of LSH hash functions can produce better performance.

REFERENCES

[1] F. Gustafsson and F. Gunnarsson, "Mobile positioning using wireless networks: Possibilities and fundamental limitations based on available wireless network measurements," *IEEE Signal Process. Mag.*, vol. 22, no. 4, pp. 41–53, Jul. 2005.

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Trans. on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1067–1080, Nov. 2007.

[3] S. Kumar, S. Gil, D. Katabi, and D. Rus, "Accurate indoor localization with zero start-up cost," in *Int. Conf. Mobile Comput.*, Sep. 2014, pp. 483–494.

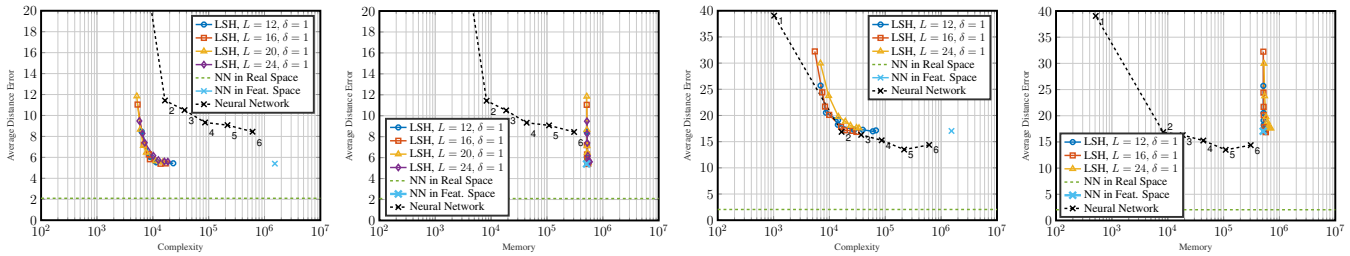
[4] N. Garcia, H. Wymeersch, E. G. Larsson, A. M. Haimovich, and M. Coulon, "Direct localization for massive MIMO," *IEEE Trans. on Signal Process.*, vol. 65, no. 10, pp. 2475–2487, May 2017.

[5] P. Bahl, V. N. Padmanabhan, V. Bahl, and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *IEEE Conf. on Computer Commun. (INFOCOM)*, Mar. 2000.

[6] K. Kaemarungsi and P. Krishnamurthy, "Modeling of indoor positioning systems based on location fingerprinting," in *IEEE Infocom*, vol. 2, Mar. 2004, pp. 1012–1022.

[7] ——, "Properties of indoor received signal strength for WLAN location fingerprinting," in *Intl. Conf. on Mobile and Ubiquitous Syst.: Networking and Services (MOBIQUITOUS)*, Aug. 2004, pp. 14–23.

[8] M. Bshara, U. Orguner, F. Gustafsson, and L. Van Biesen, "Fingerprinting localization in wireless networks based on received-signal-strength measurements: A case study on WiMAX networks," *IEEE Trans. on Veh. Tech.*, vol. 59, no. 1, pp. 283–294, Jan. 2010.



(a) Search complexity; LoS (b) Memory area; LoS (c) Search complexity; non-LoS (d) Memory area; non-LoS

Fig. 3. Comparison with Neural Network-based Positioning

[9] M. Ibrahim and M. Youssef, "CellSense: An accurate energy-efficient GSM positioning system," *IEEE Trans. on Veh. Tech.*, vol. 61, no. 1, pp. 286–296, Jan. 2012.

[10] X. Wang, L. Gao, S. Mao, and S. Pandey, "DeepFi: Deep learning for indoor fingerprinting using channel state information," in *IEEE Wireless Commun. and Networking Conf. (WCNC)*, Mar. 2015, pp. 1666–1671.

[11] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni, "CSI-based indoor localization," *IEEE Trans. on Parallel and Distributed Syst.*, vol. 24, no. 7, pp. 1300–1309, 2013.

[12] Y. Chapre, A. Ignjatovic, A. Seneviratne, and S. Jha, "CSI-MIMO: An efficient Wi-Fi fingerprinting using channel state information with MIMO," *Elsevier Pervasive and Mobile Computing*, vol. 23, pp. 89–103, Oct. 2015.

[13] A. Smailagic, J. Small, and D. P. Siewiorek, "Determining user location for context aware computing through the use of a wireless LAN infrastructure," *Institute for Complex Engineered Systems Carnegie Mellon University, Pittsburgh, PA*, vol. 15213, Dec. 2000.

[14] P. Castro, P. Chiu, T. Kremenek, and R. Muntz, "A probabilistic room location service for wireless networked environments," in *Int'l Conf. Ubiquitous Computing*. Springer, Dec. 2001, pp. 18–34.

[15] X. Wang, L. Gao, and S. Mao, "CSI phase fingerprinting for indoor localization with a deep learning approach," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1113–1123, Dec. 2016.

[16] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, "Deep convolutional neural networks for massive MIMO fingerprint-based positioning," in *IEEE Intl. Symp. Personal, Indoor, and Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–6.

[17] M. Arnold, S. Dorner, S. Cammerer, and S. Ten Brink, "On deep learning-based massive MIMO indoor user localization," in *IEEE Int'l Workshop on Signal Process. Advances in Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.

[18] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Very Large Data Base (VLDB)*, vol. 99, no. 6, Sep. 1999, pp. 518–529.

[19] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Communications of the ACM*, vol. 51, no. 1, p. 117, 2008.

[20] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proc. of the 34th Ann. ACM Symp. on Th. of Comp.*, May 2002, pp. 380–388.

[21] S. Har-Peled, P. Indyk, and R. Motwani, "Approximate nearest neighbor: Towards removing the curse of dimensionality," *Theory of computing*, vol. 8, no. 1, pp. 321–350, 2012.

[22] T. Goldstein, L. Xu, K. F. Kelly, and R. Baraniuk, "The STOne transform: Multi-resolution image enhancement and compressive video," *IEEE Trans. on Image Process.*, vol. 24, no. 12, pp. 5581–5593, Dec. 2015.

[23] A. Andoni, P. Indyk, T. Laarhoven, I. Razenshteyn, and L. Schmidt, "Practical and optimal LSH for angular distance," in *Advances in Neural Inf. Process. Syst. (NIPS)*, Dec. 2015, pp. 1225–1233.

[24] M. Kapralov, V. Potluru, and D. Woodruff, "How to fake multiply by a Gaussian matrix," in *Intl. Conf. on Machine Learning (ICML)*, June 2016, pp. 2101–2110.

[25] O. Castañeda, M. Bobbett, A. Gallyas-Sanhueza, and C. Studer, "PPAC: A versatile in-memory accelerator for matrix-vector product-like tasks," in *IEEE Intl. Conf. on Application-Specific Systems, Architectures, and Processors (ASAP)*, Jul. 2019.

[26] S. Jaeckel, L. Raschkowski, K. Börner, L. Thiele, F. Burkhardt, and E. Eberlein, "QuaDRiGa - Quasi Deterministic Radio Channel Generator User Manual and Documentation," Fraunhofer Heinrich Hertz Institute, Tech. Rep. v2.0.0, 2017.