Received September 5, 2019, accepted October 1, 2019, date of current version October 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2945929

SDR-Fi: Deep-Learning-Based Indoor Positioning via Software-Defined Radio

ERICK SCHMIDT, (Student Member, IEEE),
DEVASENA INUPAKUTIKA, (Student Member, IEEE),
RAHUL MUNDLAMURI, (Student Member, IEEE),
AND DAVID AKOPIAN, (Senior Member, IEEE)
Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, TX 78249 USA

Corresponding author: Erick Schmidt (erickschmidtt@gmail.com)

This work was supported in part by the National Science Foundation under Grant ECCS-1719043.

ABSTRACT Wi-Fi fingerprinting-based indoor localization has received increased attention due to its proven accuracy and global availability. The common received-signal-strength-based (RSS) fingerprinting presents performance degradation due to well-known signal fluctuations, but more recently, the more stable channel state information (CSI) has gained popularity. In this paper, we present *SDR-Fi*, the first reported Wi-Fi software-defined radio (SDR) receiver for indoor positioning using CSI measurements as features for deep learning (DL) classification. The CSI measurements are obtained from a fast-prototyping LabVIEW-based 802.11n SDR receiver platform. *SDR-Fi* measures CSI data passively from pilot beacon frames from a single access point (AP) at almost 10 Hz rate. A feed-forward neural network and a 1D convolutional neural network are examined to estimate location accuracy in representative testing scenarios for an indoor cluttered laboratory area, and an adjacent, covered outdoor area. The proposed DL classification methods leverage CSI-based fingerprinting for low AP scenarios, as opposed to traditional RSS-based systems, which require many APs for reliable positioning. Demonstration results are threefold: (a) A fast-prototyping SDR platform that passively extracts CSI measurements from Wi-Fi beacon frames, providing a genuine possibility for vendor network cards to provide such measurements, (b) two state-of-the-art DL classification methods outperforming traditional RSS-based methods for low AP scenarios, (c) a testing methodology for performance evaluation of the proposed indoor positioning system.

INDEX TERMS Channel state information, deep learning, fingerprinting, indoor positioning, neural networks, software-defined radio.

I. INTRODUCTION

Location-based services and navigation have become ubiquitous in the mobile computing era. An example of a popular navigation system is the Global Positioning System (GPS) [1], which uses a trilateration technique but has the limitation of working only outdoors. Because of this limitation, the necessity for an indoor positioning system (IPS) is anticipated for many applications such as navigating in airports and shopping malls, location-based information retrieval for marketing purposes, or locating people in need of medical assistance [2]. Due to its wide deployment and availability, wireless local area network (WLAN), commonly known as Wi-Fi, has recently been adopted for indoor positioning. There are many WLAN-based IPS methods proposed in literature that use analytical methods similar to GPS trilateration such as time-of-arrival [3], and time-difference-of-arrival [4], and triangulation such as angle-of-arrival [5] to estimate an indoor location. Other analytical solutions utilize complex tailored propagation models that typically require line-of-sight [6]. Further WLAN-based methods use fusion algorithms with additional hardware such as sensors and antennas [7], [8]. Empirical methods such as fingerprinting (FP) [9], have gained popularity in recent years because of their simplicity and proven accuracy (2-3 m).

The FP-based method uses two stages: offline and online. In the offline stage, characteristic radio-frequency signals are collected for a discrete grid of locations in an indoor setting to build a database, the radio-map. For each discrete location, a unique signature, a fingerprint, is generated from

The associate editor coordinating the review of this manuscript and approving it for publication was Byung-Seo Kim.

the collected radio-frequency signals. This radio-map is then used in the online phase where the user navigates in the indoor area by matching collected radio-frequency signals to the fingerprint radio-map via classification algorithms. These classification algorithms explore deterministic approaches, such as K-nearest-neighbor (KNN) [9], probabilistic methods such as maximum likelihood estimation (MLE) [10], and more recently, machine learning approaches such as support-vector machines (SVM) [11], [12], and deep learning (DL) [13]–[15].

Some FP-based IPS rely on sensor data such as magnetic and light [16]. Although, most FP-based IPS have adopted received signal strength (RSS) available in mainstream WLAN signal measurements for radio-map construction and online navigation [9]. However, it suffers from performance degradations in complex environments due to the well-known multipath fading phenomena, which eventually cause severe fluctuations in the measurements [13], [14]. These phenomena translate to an accuracy degradation in radio-map construction and classification due to unpredictable oscillations and measurement miss-rate [17]. In addition, reported RSS-based methods are feasible only in environments with dense deployments of Wi-Fi access points (AP) to ensure signature uniqueness and address measurement oscillations.

Recent literature has proposed a different fingerprint from WLAN signals, which potentially causes fewer fluctuations: the channel estimate. The channel estimate describes the indoor environment due to multipath fading and other phenomena such as reflections and refractions. These estimates characterize a specific indoor location with fine-grained information from the fading phenomena [12], [18]. Therefore, the channel estimate was recently adapted as a fingerprint, as opposed to less-stable RSS measurements. Due to said limitations, reported accuracies for RSS-based IPS are around 2-3 m [9], whereas channel-state-information-based (CSI) IPS have achieved more stable near-one-meter accuracies [18]. This accuracy is desired for precise indoor navigation in narrow hallways and commercial buildings.

The channel estimate can be measured from either time-based or frequency-based signals by approximating predefined training sequences or ''preambles'' contained in the received signal. The time-based channel estimate is called the channel impulse response (CIR), and the frequency-based channel estimate is the channel frequency response (CFR). While some solutions propose CIR exploration obtained from specialized equipment such as vector analyzers [19], [20], and ultra-wideband systems [21], recent FP-based solutions propose the CFR or so-called CSI from orthogonal frequency-division multiplexing-based (OFDM) systems. As per classification with CSI, probabilistic methods such as MLE [20], [22]–[25], and DL are most commonly used [13]–[15], [18], [26]–[30].

Most state-of-the-art CSI-based FP solutions obtain CSI from hacked hardware on a network interface card (NIC), e.g., Intel 5300 [31], or Atheros AR9580 [32]. As an alternative to obtaining the CSI, this work proposes *SDR-Fi*, a fast-prototyping WLAN OFDM-based software-defined radio (SDR) receiver for CSI measurement extraction. To the best of our knowledge, we are the first to propose an SDR-based solution that is able to extract the subcarrier (SC) channels from an OFDM symbol in passive mode from visible WLAN broadcast frames. Such broadcast frames are standardized frames transmitted by APs to identify Wi-Fi networks, and obtainable by conventional OFDM-capable NICs. Furthermore, both hacked solutions require dedicated connections and firmware modifications to the NIC and AP, making the setup cumbrous. Authors in [28] proposed an OFDM-based SDR transceiver but required synchronization of the transmitter and receiver via a hardware clock. Our SDR collects measurements asynchronously and in a real-time manner from genuine WLAN APs based on the 802.11 protocol [33]. The described SDR receiver achieves real-time packet collection at almost 10 Hz by use of acceleration features from a fast-prototyping software platform [17], [34]. Thus, we exploit SDR fast-prototyping features to assess a real-life application to an IPS where no modification to the WLAN infrastructure is required. This offers a genuine possibility for vendor network cards to provide such measurements.

In terms of classification, we explore state-of-the-art DL methods on variants of CSI measurements obtained from *SDR-Fi*. Specifically, two neural network algorithms are designed and explored: a feed-forward neural network (FFNN) and 1D convolutional neural network (CNN). The motivations for using the selected networks are as follows. In previous work [12], machine learning methods for SVM and DL were compared where the latter showed higher classification accuracy. Thus, FFNN has been selected as a superior representative model. Additionally, recent work in [15] has compared 1D and 2D CNNs for CSI-based FP; the former has shown higher accuracy for CSI data along with the benefits of a simpler network model. Therefore, in this work, we adopt state-of-the-art FFNN and 1D CNN models, and we tune them further for improved classification. Another popular category used recently in CSI-based DL classification is the long short-term memory (LSTM) network. LSTM networks use sequential time-series measurements of CSI and RSSI as inputs, thus, computing trajectory-based positioning due to a correlation between consecutive samples in time [35], [36]. Similarly, LSTM networks have also been reported for human activity recognition [37]. In this paper, our study focuses on the advantageous aspects of SDR platform's use for FP, and it is demonstrated for traditional snapshot-like FP. The platform can be also used for LSTM-like processing, which can be addressed in future work. Additionally, the proposed neural networks are assessed on variable configurations, as well as compared to two existing representative methods: (a) RSS-based *Horus* [10]; (b) and CSI-based *DeepFi* [13]. Finally, *SDR-Fi* is examined in two representative testing scenarios for a single AP: (a) an indoor cluttered laboratory area and (b) an adjacent covered outdoor (less cluttered) area.

TABLE 1. Comprehensive list of acronyms for reading easiness.

Acronym	Description
CDF	Cumulative distribution function
CFO	Carrier frequency offset
CFR	Channel frequency response
CIR	Channel impulse response
CNN	Convolutional neural network
CSI	Channel state information
DL	Deep learning
DLL	Dynamic link library
DSSS	Direct sequence spread spectrum
FFNN	Feed-forward neural network
FFT	Fast Fourier transform
FIFO	First-input-first-output
FP	Fingerprinting
HT	High-throughput
IFFT	Inverse fast Fourier transform
IPS	Indoor positioning system
KNN	K-nearest neighbor
L-LTF	Legacy long training field
L-SIG	Legacy signal
L-STF	Legacy short training field
LSTM	Long short-term memory
MAC	Medium access control
MIMO	Multiple-input multiple-output
MLE	Maximum likelihood estimation
MSE	Mean squared error
NIC	Network interface card
OFDM	Orthogonal frequency-division multiplexing
PHY	Physical layer
PPDU	Physical layer packet data unit
RBM	Restricted Boltzmann machine
RP	Reference point
RSS	Received signal strength
SC	Subcarrier
SCG	Scaled conjugate gradient
SDR	Software-defined radio
SGDM	Stochastic gradient descent with momentum
SSID	Service set identifier
SVM	Support-vector machine
TP	Test point
USRP	Universal software radio peripheral

The contributions of this paper are summarized as follows: (a) the first to propose a real-time WLAN OFDM-based SDR receiver that obtains CSI measurements in passive mode, i.e., beacon frame acquisition without dedicated connection, and without modifications to the WLAN infrastructure; (b) exploration of two state-of-the-art CSI-based DL methods with said SDR passive measurements for a single AP scenario; and (c) a *testing methodology* for performance evaluation of the proposed IPS.

Additionally, Table 1 presents a list of acronyms for reading easiness that can be used in the following discussion.

The remainder of this paper is organized as follows. Section II presents a WLAN overview and state-of-the-art IPS. Section III discusses the proposed *SDR-Fi* receiver architecture and WLAN data collection structure. In Section IV, two proposed neural network classification methods are described. An experimental methodology along with testing configurations, performance metrics, and performance analyses is presented in Section V. Section VI discusses related

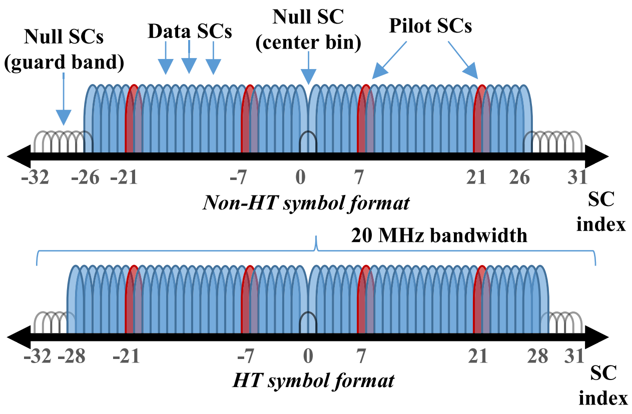


FIGURE 1. OFDM SCs for 20 MHz *non-HT* frame (top), and *HT* frame (bottom).

work. Finally, Section VII presents conclusive remarks and future work.

II. WLAN OVERVIEW AND STATE-OF-THE-ART

A. OFDM TECHNOLOGY OVERVIEW

Two main technologies are found in all releases of the 802.11 standard: DSSS and OFDM. One of the first releases was 802.11b, which is DSSS-based, deployed in the 2.4 GHz band, and is considered the *legacy standard*. As for OFDM, its first releases were 802.11a and 802.11g in the 5 GHz and 2.4 GHz bands, respectively, and have 20 MHz channel bandwidth; both, capable of up to 54 Mbps speeds. Later came 802.11n and 802.11ac releases, which extend to wider bandwidths, e.g., 40 MHz and 80 MHz, MIMO capabilities, up to 600 Mbps speeds, and use 2.4 and 5 GHz bands concurrently.

As for the PHY layer, OFDM-based WLAN releases (802.11a/g/n/ac) utilize a specific OFDM symbol format. Its total channel bandwidth of 20 MHz is comprised of 64 orthogonally overlapping SCs, each 312.5 MHz wide. The SCs, which communicate modulated bits, use frequency bin indices from -32 to 31 [31]. The center SC (zero bin) is not used; similarly, it has side null SCs as guard bands. Of the usable subcarriers, four are used as pilot SCs and the rest as data SCs. The initial OFDM release, i.e., 802.11a/g, uses 48 data SCs and four pilot SCs, from indices -26 to $+26$ in its symbol. Later release 802.11n, internally termed *HT*, extended to 52 data SCs and four pilot SCs ranging from frequency indices -28 to $+28$. Fig. 1 shows the OFDM symbol arrangement for *non-HT* and *HT* formats.

The first proposed WLAN-based IPS have been based on passive WLAN measurement collection [7], [9], [10]. This translates to passive beacon frame capturing from visible broadcasting APs. The beacon frame is a broadcast PPDU containing the necessary information to identify the AP. They are of the management frame category and contain a PHY preamble for synchronization as well as information fields relevant to the AP, such as the MAC address and the SSID [17]. These beacon frames are typically collected by commercial NICs to compute RSS and CSI data internally from a given AP, and eventually report visible Wi-Fi networks (CSI is typically not made available by commercial NICs). A deployed WLAN network using the most recent 802.11 release equipment, unless manually configured through the AP, employs the *non-HT* symbol format for broadcasting beacon frames by default [33].

As opposed to RSS measurements, which are reported on non-modified commercial off-the-shelf WLAN NICs, the CSI is not commonly accessible directly. Recently, modified firmware on the Intel 5300 NIC called *CSI Tool*, has been made available to access CSI measurements [31]. Similarly, *Atheros CSI Tool* is available for Atheros NICs [32]. Such tools extract the CSI from a particular OFDM-based preamble frame. Specifically, the tools leverage an internal *sounding mechanism* found on the 802.11n and later releases to enable spatial diversity channels between the AP and the NIC. The mechanism occurs via an exchange of *sounding preambles* using the *HT* frame format. The tool then extracts CSI from either 30 or 56 SC indices from said preambles. Said *HT* preambles are used for MIMO technology for multiple antenna-capable NICs and, thus, can be sent in different spatial streams and in extended channel bandwidths of 40 MHz [33]. Before the *sounding mechanism* is triggered, a prior established communication is expected between the AP and NIC for 802.11 protocol compatibility matching, e.g., the AP can have compatibility up to the 802.11ac release, and the NIC only up to 802.11g. Thus, the connection employs technology relevant to the oldest release among both. Due to this dedicated connection, most state-of-the-art CSI-based IPS report a single AP [13]–[15], [18], [20], [23], [24], [26]. Another limitation is the availability of said tools to a few NIC vendors. Furthermore, the *Atheros CSI Tool* requires firmware modifications to both the NIC and the AP, making the setup complex. Finally, the *sounding mechanism* has to be triggered via pings between the NIC and AP using an imposed Java program [13].

Because the *sounding mechanism* was introduced in a later OFDM-based release (802.11n), not all OFDM-capable single-antenna NICs have said mechanism, and thus, do not recognize *HT frames*. Nonetheless, all OFDM legacy NICs are, per the standard, required to transmit and receive *non-HT* frames. This is because broadcast OFDM beacon frames are sent on OFDM legacy *non-HT* frame format. Therefore, passive measurement collection on *non-HT* frames for non-modified WLAN infrastructure and legacy compatibility is anticipated. Each captured beacon frame contains 52 SCs for CSI measurements, as opposed to the *sounding mechanism* containing 56 SCs. Table 2 shows the SCs frequency indices that are transmitted in the full and decimated *sounding* frames, corresponding to 30 and 56 SCs, versus the full and decimated SCs that are available from *SDR-Fi*, corresponding to 28 and 52 SCs. The decimated beacon frame assimilates the decimated sounding SCs index pattern. For this assimilation, we removed the indices −28, 27, and 28 that are not present in *non-HT* frames, and added SC 26, thus having 28 instead of 30 SCs. We use said decimated frame and the full beacon frame as a comparison against the *HT* frame SCs for our proposed methods.

TABLE 2. OFDM Subcarrier channels index for different frames.

Frame Type	SC frequency indices
Full sounding frame (*HT*)	-28, -27, -26, -25, …, -1, 1, 2, 3, …, 28
Decimated sounding frame (*HT*)	-28, -26, -24, -22, -20, -18, -16, -14, -12, -10, -8, -6, -4, -2, -1, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 28
Full beacon frame (*non-HT*)	-26, -25, -24, -23, …, -1, 1, 2, 3, …, 26
Decimated beacon frame[a] (*non-HT*)	-26, -24, -22, -20, -18, -16, -14, -12, -10, -8, -6, -4, -2, -1, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 26

[a]Decimated from the *non-HT* full beacon frame to assimilate the decimated sounding frame SC index pattern.

B. STATE-OF-THE-ART CSI-BASED LOCALIZATION

Since the development of the *CSI Tool*, most CSI-based IPS have been explored using 30 complex-valued SCs per antenna. Few reported IPS used *Atheros CSI Tool* to obtain 56 SCs due to setup complexity [29], [30]. *PinLoc* first explored the stability of CSI for FP. They divided an indoor area into 1 m x 1 m spots and attempted war driving (access point mapping) with a robot to build the radio-map; in the online phase, they assessed a log-likelihood function to detect such spots using either phase or magnitude [23]. *FIFS* attempted grouping 30 SC magnitudes into four values to use as a prior distribution per RP on a MLE [22]. *CSI-MIMO* aggregates 90 SCs from three antennas to 30 SCs and uses the difference of normalized amplitudes and phases between consecutive SCs as a signature [24]. *PhaseFi* averaged 90 SCs into 30 and applied a phase calibration assuming the SCs are symmetric. They feed the calibrated phases into a DL-based autoencoder network. It trains the network in a layer-by-layer manner using the greedy algorithm with a stack of RBMs [14]. *BiLoc* used similar method to compute the phases but in the 5 GHz band [27]. Similar network architecture is also exploited in [13], but they use 30 SC magnitudes instead. *ConFi* is the first to leverage a CSI-based CNN. They generate "images" by organizing 30 SC amplitudes into a time-frequency matrix [18]. Similarly, *CiFi* uses a 2D CNN but obtains phase differences from 30 SCs to form an angle-of-arrival image as input [26]. Authors in [15] used 30 SC magnitudes in a far less complex 1D CNN. Another category is so-called device-free localization. Authors in [25] form a radio-map with pairs of APs and NICs at fixed locations, assuming the user is device-free. The radio-map construction consists of placing several heights, shapes, and orientations of users per RP. In the online phase, if the user disrupts the line-of-sight and matches the radio-map patterns, a location is estimated with potential false alarms. An SDR in [28] is proposed by using two USRP units synchronized by a clock and using similar OFDM symbol structure as the WLAN standard. They collected 52 SCs and applied a 1D CNN as in this work. Nevertheless, the reported system does not represent a full asynchronous WLAN OFDM receiver. Finally, authors in [20] convert CSI values into CIR via FFT. They attempt a super-resolution algorithm to obtain high-granularity path-delay signatures from the estimated CIR for localization.

III. PROPOSED OFDM SOFTWARE RECEIVER

A. RECEIVER OVERVIEW

SDR is defined as having a front-end where received signals are digitized and a software aspect where digitized data are processed. SDR solutions become popular because of providing full control of receiver modules, so the researchers can integrate and test their methods without redesigning all receiver chains. This becomes an advantage for SDR-based research for fast-prototyping [12], [34]. The proposed single-antenna SDR mimics legacy OFDM-based NICs that are capable of passively listening for available networks and decoding *non-HT* beacon frames.

The receiver is implemented by the Software Communications and Navigation Systems Laboratory at The University of Texas at San Antonio. We implemented an OFDM-based WLAN receiver in a platform-based environment by using LabVIEW, a platform from National Instruments that uses a visual programming language [34]. LabVIEW interfaces between the host PC, where the software resides, and the front-end. The receiver uses LabVIEW built-in function blocks along with custom baseband blocks compiled in C++ as DLLs. Such custom blocks are integrated into the receiver via a *Call Library Function Node block*. Further acceleration features that are inherent from LabVIEW are also utilized for real-time operation [12].

The architecture of the receiver is based on the *producer-consumer loop* design. This design allows LabVIEW to handle real-time continuous data acquisition from the front-end in the *producer loop*. After this, the *consumer loop* dequeues the raw data from a FIFO buffer and sends it to the custom baseband blocks. LabVIEW uses native NI-USRP drivers to interface with the front-end seamlessly. The core of the receiver occurs at the baseband processing blocks in the *consumer loop*. The reader is directed to [12] for additional software architecture details.

B. OFDM BEACON FRAME FORMAT

This section presents an overview of the OFDM PPDU beacon frame format. Fig. 2 shows the detailed structure of the frame. The L-STF is two OFDM symbols long and is used for packet detection and coarse CFO estimation and correction. The L-LTF is also two symbols long and is used for symbol timing, fine CFO estimation and correction, and channel estimation. The L-SIG field is a single OFDM symbol and contains the RATE and LENGTH fields along with a parity bit. These fields contain the modulation type and total length in symbols, which are used to decode the DATA field. The DATA field composed of multiple symbols contains a MAC header, which specifies the frame type and subtype. We filter only management type and beacon frame subtype.

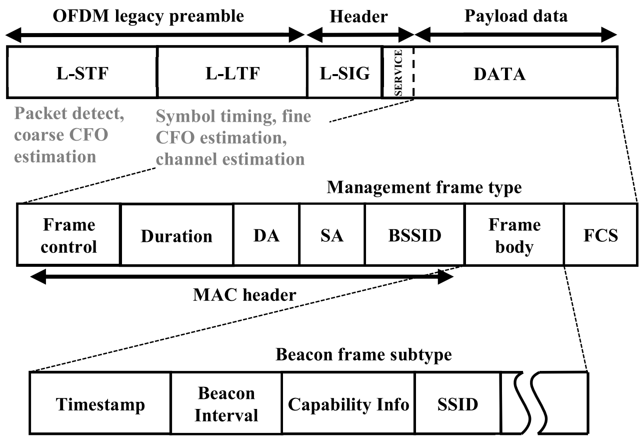


FIGURE 2. OFDM legacy PPDU beacon frame format and description fields.

TABLE 3. Implemented baseband blocks on proposed receiver.

Baseband block	Machine state	LabVIEW/custom block	Method description
Packet detect	1	Custom DLL	[39]
Coarse CFO estimator	1	Custom DLL	[40]
Symbol timing	2	Custom DLL	FFT-based cross-correlation
Fine CFO estimator	2	Custom DLL	[40]
Channel estimate (CSI)	2	LabVIEW-based	Zero-forcing equalizer
OFDM demodulator	3, 4	Custom DLL	IFFT
Equalizer	3, 4	LabVIEW-based	-
Pilot phase correction	3, 4	LabVIEW-based	-
Symbol demodulator	3, 4	LabVIEW-based	LabVIEW blocks
Deinterleaver	3, 4	LabVIEW-based	LabVIEW blocks
Viterbi decoder	3, 4	Custom DLL	[41]
Parity check	3	LabVIEW-based	-
Descrambler	4	LabVIEW-based	Polynomial
Checksum	4	Custom DLL	CRC32

C. BASEBAND BLOCKS

The *consumer loop* is built as a sequential state machine, which is divided into four main states: (1) the *packet detect state*, (2) *timing sync state*, (3) *L-SIG decode state*, and (4) *payload decode state*. These four states are based on the corresponding fields L-STF, L-LTF, L-SIG, and DATA, respectively. Table 3 lists the baseband blocks that were implemented based on the state machine, whether it is a LabVIEW-based or custom DLL block, and the method used. Some states use certain blocks more than once. We use an optimized FFT library for most of our methods [38].

State (1) is the most computationally intensive, as it runs a sliding window continually for packet detection. Specifically, an FFT-based autocorrelation method from [39] is used on the L-STF. Afterward, a coarse CFO estimation method is used [40]. For state (2), symbol timing is achieved via FFT-based cross-correlation with the L-LTF local replica. A fine CFO is implemented as in state (1). The channel estimation uses a frequency-domain zero-forcing equalizer with the L-LTF in LabVIEW blocks. This is where the CSI is measured (the 52 SCs), which is used thereafter to equalize the rest of the payload data. In state (3), the L-SIG symbol is decoded to obtain the RATE and LENGTH for further demodulation. The L-SIG symbol is first equalized, and a phase correction is applied based on the pilot tones. Afterward, OFDM symbol demodulation occurs via a DLL IFFT. Subsequently, a quadrature amplitude demodulator found in LabVIEW built-in blocks is used. Similarly, deinterleaving, Viterbi decoding, and parity check are applied prior to L-SIG header extraction. The Viterbi decoder is ported to a custom DLL from a highly efficient single-input multiple-data routine [41]. Once the L-SIG is decoded, state (4) applies similar steps as in (3) but with a polynomial-based data descrambler and a checksum computation of the frame. The checksum uses a cyclic redundancy check, the CRC32, with a known polynomial in the standard. If it passes, it then proceeds to measure RSS, and log beacon frame data from a particular AP such as MAC address, and SSID. Overall, all these baseband block implementations are described in the standard [33].

D. SLEEP-MODE FOR REAL-TIME OPERATION

As of the current version of *SDR-Fi*, the receiver is not able to operate in real-time since the most demanding operation of constant correlation in a sliding window manner occurs in state (1). Specifically, this state makes the receiver run three times slower than real-time operation. However, since a continuous operation of the receiver is not required, a *sleep-mode* has been implemented based on the beacon broadcast interval of 102.4 milliseconds per AP [17]. The receiver is put to ''sleep'' after detecting a beacon frame packet. This is done by commercial NICs to save power and, thus, is imitated in the SDR implementation. With *sleep-mode*, the receiver achieves real-time packet collection rates up to 9.5 Hz, which is close to the theoretical maximum of 9.76 Hz [17].

IV. NEURAL NETWORK MODELS

In this work, we adopt a FFNN and a 1D convolutional neural network for classification. The models for the proposed IPS are less complex compared to the algorithms discussed in [13], [15], and [18]. Furthermore, the parameters for these models are selected based on the data set. The details of the custom network models are discussed in the following subsections.

A. SDR-BASED CSI MEASUREMENTS

As has been demonstrated previously in [13] and [18], the signature in CSI measurements contains fine-grained information of a relevant location. It has been observed that different locations can be characterized separately based on specific patterns in the CSI waveforms. In terms of neural networks, each CSI pattern denotes a location or a class. This is why

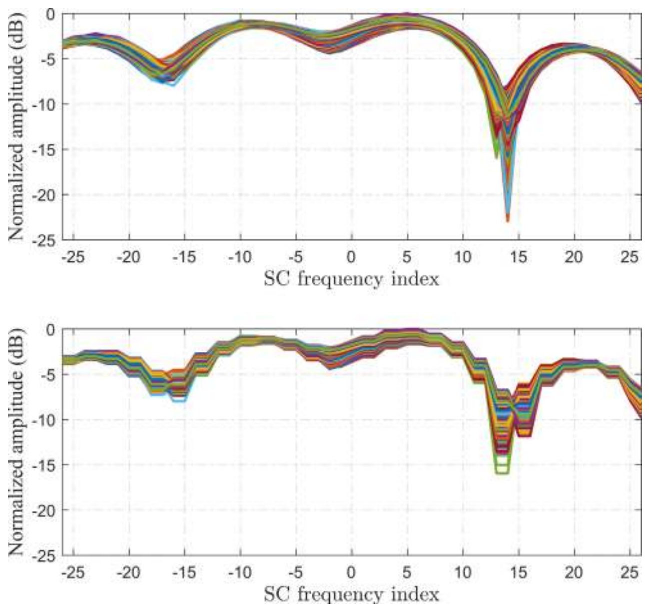


FIGURE 3. Normalized CSI magnitudes for a single RP for 52 SCs (top), and 28 SCs (bottom).

neural networks have been attempted to learn said patterns for location estimation. The CSI measurement for each SC is denoted as complex-valued:

$$CSI_i = |CSI_i|\, e^{j\angle CSI_i} \tag{1}$$

where $|CSI_i|$ and $\angle CSI_i$ are the magnitude and phase of the i-th SC, respectively [13]. As mentioned in Section II-B, many conjugations of the CSI magnitude and phase have been used in previous reported IPS. In this work, we use CSI magnitudes as inputs for our network configuration along with calibrated RSS values in dBm. For calibration, we test RSS values alongside a reported commercial NIC. We also explore the effects of having the full SCs from a beacon frame for network training. Fig. 3 shows a batch of 2000 CSI normalized measurements collected for a specific RP in our experiments. The top plot shows 52 SCs, and the bottom shows 28 SCs. The bottom plot closely relates to the 30 SCs available from the *CSI Tool*.

B. FEED-FORWARD NEURAL NETWORK

A FFNN works similar to the neurons of a human brain, through interactive learning. The neurons learn by associating patterns through complex interconnections. The architecture of the proposed FFNN is a specific type of pattern recognition network called *PatternNet* from MATLAB [42]. The overall training of the FFNN occurs through a multilayer backpropagation training algorithm called SCG [43]. The outputs of each layer are weights and biases, which are mapped to a nonlinear activation function called hyperbolic tangent sigmoid, or *tansig*. This activation function calculates the layer's outputs based on its net inputs within a range [−1,1]. Overall, the SCG algorithm adjusts the weights and biases to minimize the gradient.

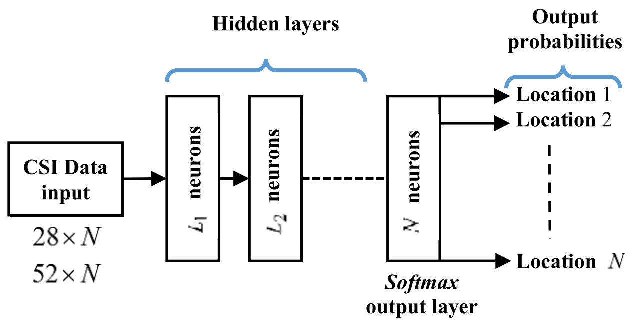


FIGURE 4. Structure of the FFNN.

In the context of IPS, the neural network is trained to classify inputs, which are CSI measurements, to N target classes, which are the radio-map locations. Fig. 4 shows an overview of the FFNN structure. The input dimensions are corresponding to the SCs, i.e., either $28 \times N$ or $52 \times N$. Each hidden layer has L_k neurons, where typically $L_1 > L_2 > L_i, k \in \{1 \ldots K\}$, for K layers. The output layer always has N neurons. Initially, the inputs are computed and passed as weights and biases assigned to the neurons of the first hidden layer. During the training, the gradient is minimized based on a MSE loss function defined as:

$$\varepsilon = \frac{1}{2N} \sum_{j=1}^{N} \left(RP_j - RP'_j\right)^2 \tag{2}$$

where ε is the MSE, RP_j is the true j-th location, and RP'_j is the estimated location. The interaction between HLs occurs in a similar way: the newly adjusted weights of the neurons are passed as a vector input to each neuron of the next hidden layer. The overall training process runs until the minimum gradient is reached; this signifies a well-trained model. As for the data, the model randomly divides the data into training, validation, and testing subsets. The output layer implements a *softmax* function to map the output weights to the estimated classes (N locations). Therefore, each output neuron corresponds to an RP, and these final weights are the estimated probabilities for each location. The *softmax* is based on an exponential function that returns values in the range of [0, 1]:

$$y_j = \frac{e^{\mathbf{w}_j^T x_i}}{\sum_{j=1}^{N} e^{\mathbf{w}_j^T x_i}} \tag{3}$$

where y_j is the output weight corresponding to the j-th location, $\mathbf{w}_j$ is a weight vector from the last hidden layer connected to the output neuron, and x_i is the i-th output of the last hidden layer.

C. CONVOLUTIONAL NEURAL NETWORK

CNNs are proven to be effective in 2D image classification and pattern detection [44]. However, we describe a 1D CNN for flattened CSI-based inputs to simplify our proposed model. Similar to [15], we optimize our 1D CNN further for our data sets.

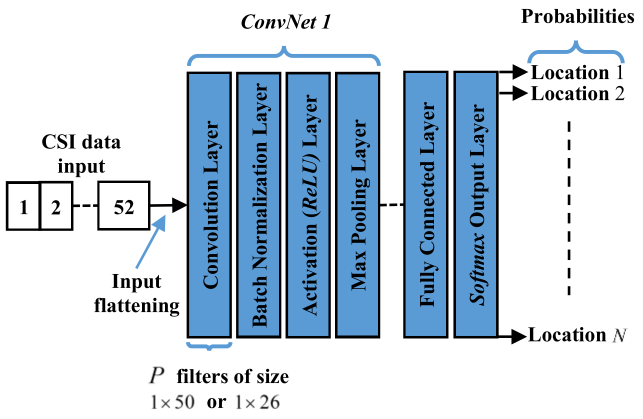


FIGURE 5. Structure and workflow of the 1D CNN.

The basis of CNN is similar to FFNN in terms of a layered structure. However, typical CNN comprises additional layers such as a convolutional layer, pooling layer, activation layer, and fully connected layer. Fig. 5 shows the structure and workflow of a 1D CNN. The structure uses so-called convolutional blocks (*ConvNet*), which consist of all or some of these four layers: a convolutional layer, which processes the input samples based on a 1D filter [44]; a batch normalization layer; an activation layer; and a pooling layer, which resamples the data according to the next input size. The model can have several convolutional blocks; however, we observed a degradation depending on the experimental scenario.

On each convolutional block, the convolutional layer consists of optimized filters of sizes 1×50 or 1×26 for our flattened 1D CSI data. Additionally, we use P of these filters. Afterward, the batch normalization is applied between the convolutional layer and the activation layer. This achieves faster training and reduces initial parameter sensitivity. In the activation layer, a nonlinear transfer function is applied as a threshold to remove negative values. The rectified linear unit is chosen as the activation function for all the layers except the output layer. It is defined as follows:

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{4}$$

where x is the input element. The result is then downsampled with a *max-pooling layer* to produce more compact features.

The last two layers consist of a fully connected layer and a *softmax* output layer. The fully connected layer resamples the features from previous *ConvNets* to match the output layer. In this case, it has N neurons matching the locations. As in the FFNN, the *softmax* layer outputs probabilities of N locations based on the learned features in the CNN hierarchy.

The training of the model utilizes a SGDM [45] algorithm. It accumulates the gradient of the past steps to update the parameters while minimizing the loss function. Similar to the FFNN, the minimization occurs at each iteration along the direction of steepest descent. The momentum term reduces the oscillation during the minimization. The training function is defined as follows:

$$\mathbf{w}_{l+1} = \mathbf{w}_l - \alpha \nabla L(\mathbf{w}_l) + \gamma (\mathbf{w}_l - \mathbf{w}_{l-1}) \tag{5}$$

where $\mathbf{w}_l$ is the weight vector of the l-th iteration, $L(\mathbf{w}_l)$ is the loss function, α is a learning rate, ∇ is a gradient operator, and γ is the contribution of the previous gradient step to the current iteration. The algorithm evaluates the gradient and updates the weights in each iteration using a subset of the training data. The cross-entropy with $L2$ regularization (weight decay) is chosen as the loss function to avoid overfitting [46], [47]. It is defined as follows:

$$L_R(\mathbf{w}) = \frac{-1}{M}\left[\sum_{i=1}^{M}\sum_{j=1}^{N} I\{z_i = j\} \log \frac{e^{\mathbf{w}_j^T x_i}}{\sum_{l=1}^{N} e^{\mathbf{w}_l^T x_i}}\right] + \frac{\lambda}{2}\sum_{i=1}^{L}\sum_{j=1}^{N} w_{i,j}^2 \tag{6}$$

where M is the training set size, N is the number of RPs, $I\{\cdot\}$ is an indicator function, x_i is the i-th RP location output of the previous layer, z_i is the RP location where CSI measurements are collected, λ is a regularization factor, and L is the size of $\mathbf{w}$, i.e., number of neurons in the fully connected layer. The training process in the *ConvNet* is iterated until the features are appropriately discriminative. These features are then fed to the fully connected layer. With the downsampling and the shared weights between layers, the CNN reduces the number of trainable parameters, i.e., weights. Finally, similar to FFNN, the model performs cross-validation by dividing the data into randomized sets such as training, validation, and testing.

V. EXPERIMENTAL VALIDATION

This section presents an experiment methodology followed by a description of the performance metrics used in the experimental validation. We present several testing configurations and a comparative analysis of the results. In this work, we evaluate the performance of the IPS on a 2D coordinate system.

A. EXPERIMENT METHODOLOGY

We perform data collection at two representative scenarios for our experimental validation: (a) at the indoor laboratory (cluttered indoor area); and (b) at an adjacent (less cluttered) covered outdoor area. For the lab, we survey 69 RPs with a granularity of 60.96 cm (2 ft) between RPs. We choose 16 TPs in-between RPs for validation. The indoor lab depicts a cluttered environment with desks, chairs, and lockers. Fig. 6 shows the indoor layout with a single AP. The total area is around 60 m^2 (646 ft^2), and the sides are around 10 m $\times$ 6 m (32.8 ft $\times$ 19.7 ft). The adjacent outdoor area depicts an open space, and we use the same indoor AP for the survey. We use 36 RPs spaced at 76.2 cm (2.5 ft) apart. Likewise, we use eight interlaced TPs for validation. The adjacent outdoor area is 3.81 m $\times$ 6 m (12.5 ft $\times$ 20 ft).

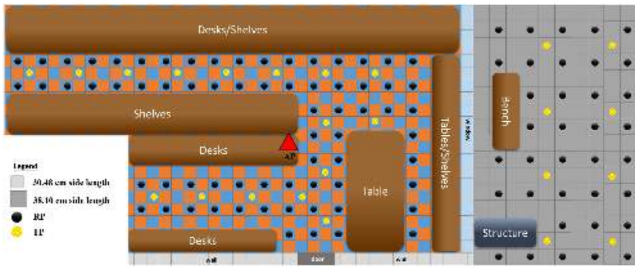


FIGURE 6. Indoor laboratory layout (left) and adjacent (right) outdoor layout for testing.

We define a sample as a successful collection of a beacon frame at the receiver, providing an RSS measurement, and 28 or 52 CSI measurements. For each RP and TP, we collect 2000 and 500 samples, respectively. We use a NI-USRP 9232 front-end along with an ASUS ROG GL552VW laptop (quad-core Intel i7-6700HQ processor, 32 GB RAM, and Windows 10), and the proposed *SDR-Fi* v4.3. With this version, it takes us around 3.5 min to collect 2000 samples, and 1 min to collect 500 samples. A total of 210,000 samples are collected for RPs, and 12,000 samples for TPs.

B. PERFORMANCE METRICS

In the training phase of the neural network, the RPs are used as classes and the SCs as features to generate a network model. In the online phase, each TP sample is evaluated against the trained model. The model returns probability weights for each known trained class, which corresponds to 69 classes (RPs) for indoors, and 36 for outdoors. This means each TP evaluation returns, e.g., 69 probability values such that when combined they add up to 1. To compute the performance metric, we use the mean error by using the weights of each output class (based on the RPs) per TP evaluation, thus, computing a 2D centroid location, i.e. $(\hat{x}_{TP}, \hat{y}_{TP})$. Furthermore, we subtract the coordinates from the known TP coordinates used in said evaluation. We evaluate K TPs to compute the mean error ε as follows:

$$\varepsilon = \frac{1}{K}\sum_{l=1}^{K}\sqrt{\left(\hat{x}_{TP,l} - x_{TP,l}\right)^2 + \left(\hat{y}_{TP,l} - y_{TP,l}\right)^2} \tag{7}$$

Additionally, we generate a CDF based on all K evaluated TP distance errors and compute the mean, standard deviation, 50th percentile, and 90th percentile. We perform the same metric evaluation for all our comparative results in the following subsections.

C. TESTING CONFIGURATIONS

For our IPS evaluation, we use the MATLAB Deep Learning Toolbox [42] to construct customized FFNN and 1D CNN configurations. We evaluate both networks along with CSI-based *DeepFi*, and RSS-based *Horus* methods, respectively. Table 4 shows several configurations evaluated for both our networks. For FFNN, we evaluate 2, 3, and 4 HL configurations. We vary the neurons in the hidden layers

TABLE 4. Testing configurations for the performance evaluation.

Type	Configuration	Value
Inputs	No. SCs	28, 52
	Training samples	500, 1000, 2000
	testing samples	500
General	Training function	SGD, SGDM
	Loss function	MSE
FFNN	Hidden layers	2, 3, 4
	Neurons	300, 150, 100, 69, 36
	Training, testing, validation (%)	90% training, 0% testing, 10% validation (indoors); 75% training, 15% validation, 10% testing (outdoors).
1D CNN	Convolutional layers	1 (indoor), 3 (outdoors)
	Filter size	1 × 50 (indoors), 1 × 26 (outdoors).
	Number of filters	512, 1024, 2048
	Learning rate	0.01
DeepFi (optimized)	Indoors (neurons)	200/50/30/69
	Outdoors (neurons)	70/50/30/36

as the highest value being the first as listed in Table 4, e.g., 300/150/100 would be a 3 HL configuration. Further, the training of the FFNN models for indoors use 90% of samples for training and 10% for validation, whereas for outdoors we use 75% for training, 15% for validation, and 10% for testing. These percentages are used for hyperparameter tuning during the training stage, and we have selected them for optimal performance. For all the 1D CNN variations, the learning rate is set at 0.001. For *DeepFi*, we optimize the configuration to adjust to our indoor and outdoor environments. For indoors, we configure four HLs and use 200/50/30/69 neurons, respectively. Similarly, for outdoors, we use 4 HLs and 70/50/30/36 neurons. Thus, we report an optimized version of *DeepFi* in our results. Additionally, *DeepFi* originally uses 90 SCs from 3 antennas; in this work, we input the SC measurements from the single-antenna SDR into *DeepFi*, e.g., 52 SCs.

D. RESULTS ANALYSIS

1) COMPARISON WITH EXISTING METHODS

We compare the proposed neural networks with *Horus* and optimized *DeepFi*. We use our most optimized configurations for FFNN and 1D CNN. Specifically, we use two HLs with 300/150 neurons per layer for the indoor setting, and 4 HLs with 300/150/100/36 neurons for outdoors, for the FFNN. For 1D CNN, we use one convolutional layer and 2048 filters for indoors, and three convolutional layers and 2048 filters per layer for outdoors. We use 52 SCs as feature inputs and 2000 training samples per RP. Table 5 shows results for the indoor environment. The best performance is observed for the 1D CNN method with 0.99 m mean error while FFNN achieves 1.37 m mean error, both outperforming *DeepFi* and *Horus*. The 1D CNN and FFNN outperform *DeepFi* by 42.5% and 20.7%, respectively. Fig. 7 shows the CDF plot for all four methods. We can observe that 1D CNN and FFNN outperform *DeepFi* and *Horus* by comparing their 50th percentiles.

TABLE 5. Comparative positioning performance results of proposed methods for indoor environment.

	Mean (m)	Std. dev. (m)	50th pctl. (m)	90th pctl. (m)
1D-CNN	0.9914	0.6342	0.7750	1.9985
FFNN	1.3678	0.8736	1.2353	2.6525
DeepFi	1.7238	1.4180	1.3171	3.6594
Horus	2.83	1.7305	2.7379	4.7922

TABLE 6. Comparative positioning performance results of proposed methods for outdoor environment.

	Mean (m)	Std. dev. (m)	50th pctl. (m)	90th pctl. (m)
1D-CNN	1.5210	0.8358	1.5598	2.7518
FFNN	1.4671	0.8298	1.3751	2.7450
DeepFi	1.8603	0.9471	1.6667	3.20
Horus	2.4073	1.2976	2.1504	4.3352

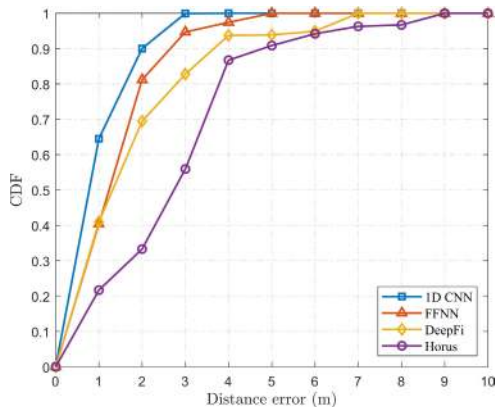


FIGURE 7. Indoor CDF distance error comparison with existing methods.

Similarly, Table 6 shows results for the outdoor environment. In the less cluttered scenario, we observe performance degradation in mean error with 1.52 m for 1D CNN and 1.47 m for FFNN when compared with indoors. The same degradation is observed for *DeepFi*. Fig. 8 shows the CDF for the outdoor environment. In contrast, *Horus* shows an improvement in the outdoor vs indoor setting from 2.83 m to 2.40 m; thus exploiting the less-cluttered outdoor environment for RSS-based FP.

We can see both 1D CNN and FFNN performing similarly but still outperforming *DeepFi* and *Horus*. Overall, we prove our hypothesis of unique signatures in the high-grained CSI measurements by observing a more stable performance in the more cluttered environment (indoors) against a noisier outdoor setting (see Fig. 7 and Fig. 8). Conversely, an improvement is seen for RSS-based *Horus* in this same trend.

2) IMPACT ON THE NUMBER OF SCS

We compare the FFNN and 1D CNN against varying SCs obtainable by *SDR-Fi*. We use the same configuration as in Section V-D.1, but we vary the number of SCs between the decimated 28 SCs (see Section II-A) and the full 52 SCs. Table 7 shows mean error and std. dev. results for both 52 and 28 SCs evaluation on the indoor environment. For 1D CNN,

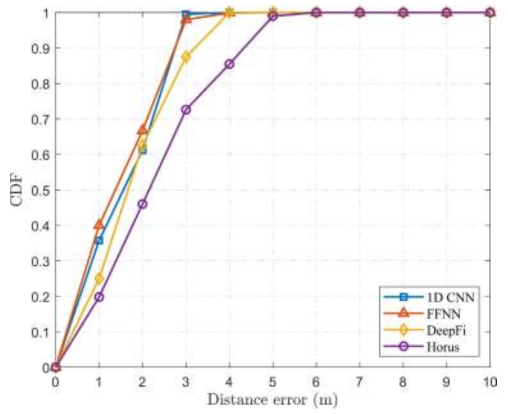


FIGURE 8. Outdoor CDF distance error comparison with existing methods.

TABLE 7. Comparison results for number of SCs for indoor environment.

	28 SCs		52 SCs	
	Mean (m)	Std. dev. (m)	Mean (m)	Std. dev. (m)
1D CNN	1.1966	0.8061	0.9914	0.6342
FFNN	1.8778	1.3543	1.3678	0.8736

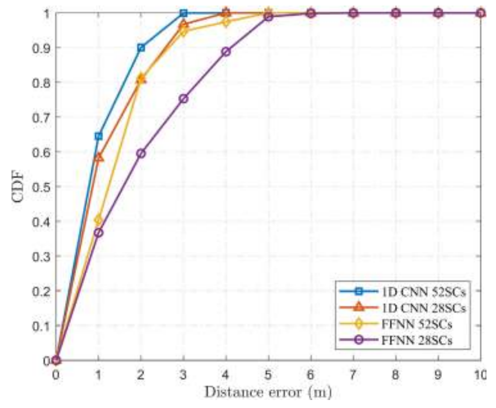
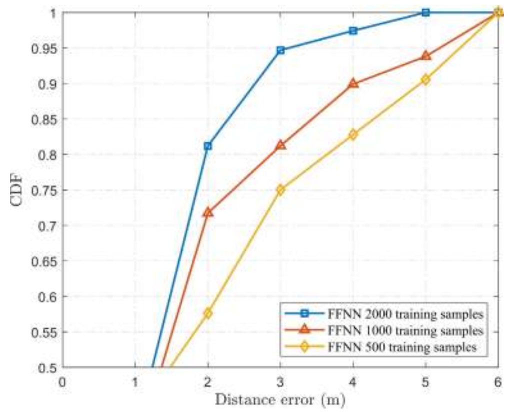


FIGURE 9. Indoor CDF distance error comparison with varying SCs for both models.

we see a mean error improvement of 17.2% when using all 52 SCs; however, the improvement on the std. dev. is 21.3%. For FFNN, we see an improvement of 27.2% and 35.5% for the mean and std. dev., respectively. Fig. 9 shows the CDF for the four-combination scenarios. We can clearly observe improvement in the CDF curves when using the 52 SCs from the OFDM symbol for both 1D CNN and FFNN, against 28 SCs. Also, we observe an improvement on the std. dev. (as opposed to the mean) due to a finer resolution in the SCs. In other words, the higher number of SCs translates to overall precision improvement because the network trains with an enhanced channel estimate.

3) IMPACT ON THE NUMBER OF TRAINING SAMPLES

We assess the number of training samples entered into both the FFNN and 1D CNN models for the indoor setting. We use the same network configurations as in Section V-D.1, but we use 500, 1000, and 2000 training samples per RP.



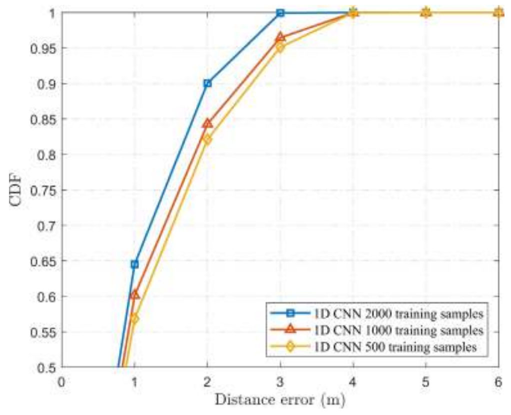


FIGURE 10. Indoor CDF distance error comparison with varying training samples for FFNN (top), and 1D CNN (bottom).

TABLE 8. Comparison results for training samples for indoor environment.

	500 samples		1000 samples		2000 samples	
	Mean (m)	Std. dev. (m)	Mean (m)	Std. dev. (m)	Mean (m)	Std. dev. (m)
1D CNN	1.2171	0.8224	1.1472	0.8005	0.9914	0.6342
FFNN	2.039	1.5794	1.7843	1.3712	1.3678	0.8736

Table 8 shows the comparative results for different training samples on the models. We observe a gain of 18.5% in the mean error for 1D CNN and 32.9% for FFNN when increasing training samples from 500 to 2000. We conclude that the 1D CNN is at its optimum performance for CSI-based training, as not much gain is observed. Additionally, 1D CNN training performance with 500 samples outperforms FFNN training with 2000 samples. Thus, the 1D CNN has better compatibility for CSI-based training. Fig. 10 shows a zoomed-in CDF plot for different training samples for FFNN on the top, and 1D CNN on the bottom. This also shows more curve variations for FFNN against 1D CNN when training samples are increased.

4) IMPACT ON THE NUMBER OF HIDDEN LAYERS

We assess the impact on the number of hidden layers for the FFNN. We test with 2, 3, and 4 HLs. Table 9 shows the comparative results for our indoor setting. We observe a

TABLE 9. Comparison results for hidden layers on the FFNN for indoor environment.

	2 HLs (300/150 neurons)	3 HLs (300/150/100 neurons)	4 HLs (300/150/100/69 neurons)
Mean (m)	1.3678	1.6237	1.7957
Std. dev. (m)	0.8736	1.0584	1.282

TABLE 10. Comparison results for number of convolutional layer filters.

	512 filters	1024 filters	2048 filters
Mean (m)	1.3505	1.2703	0.9914
Std. dev. (m)	1.0404	0.9349	0.6342

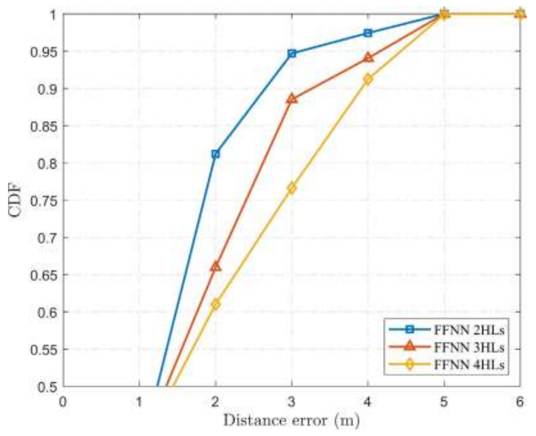


FIGURE 11. Indoor CDF distance error comparison for FFNN with varying HLs.

decrease in performance as the hidden layers are increased. We conclude that the complexity does not add to better location estimation in the FFNN. In fact, we see a similar performance for 4 HLs as in *DeepFi* (see Table 5). Thus, our 2 HL model is better tuned for the indoor setting. Fig. 11 shows a zoomed-in CDF comparison for different HLs. For the indoor setting, an evident underperformance is observed as the complexity of the network is increased.

5) IMPACT ON THE NUMBER OF CONVOLUTIONAL LAYER FILTERS

To achieve the best-optimized model for our 1D CNN, we assess the impact on the number of *ConvNet* filters. The 1D CNN configuration for indoors is listed in Section V-D.1, but we vary the number of filters as in Table 4. Table 10 shows the comparative results for our indoor setting. We see a gain of 26.6% for the mean error, and 39% for the std. dev. when comparing 512 filters and 2048 filters. We see an improvement in precision (std. dev.) with the increase in filters on the 1D CNN. Fig. 12 shows the CDF comparison for these varying filters. We also observe that 1D CNN training is more susceptible to number of filters compared to number of training samples (see Fig. 10 bottom).

VI. RELATED WORK

Indoor positioning has observed increasing research in the last two decades for numerous types of technologies such as WLAN, Bluetooth, FM radio, ultrasound, magnetic fields, among others [7]. Nonetheless, WLAN-based has been the most predominant due to its extensive deployment and availability. Beginning from [9], FP-based IPS have presented numerous methods in recent years, with DL being the most recent (and best performing) for classification [18], and RSS and CSI the most used features. We narrow our proposed methods within the CSI-based FP using DL for classification. Several testing methodologies have been proposed, having a broad range of options such as number of APs, evaluation on RPs or evaluation on TPs, the type of survey environment whether open space or cluttered and the type of metric used for reporting accuracy, e.g., mean error and CDF percentiles. In this light, we consider the *best representative testing methodology* for overall performance as follows: (a) 1 AP or more, the former to show a potential worst-case scenario for emergency situations; (b) performance evaluation on TPs rather than RPs; (c) both a cluttered and a less cluttered area; and (d) mean error as well as CDF percentile as performance metrics. Said methodology suitably evaluates an IPS for an improved representation of its performance. We present a comprehensive list of representative FP-based IPS in Table 11 by categorizing them into FP features, classification method, number of APs used, testing evaluation on either RPs or TPs, and their self-reported accuracy metrics. To the best of our knowledge, we present the best accuracy in terms of the aforementioned *representative testing methodology*, with 0.99m mean error and 0.77m 50th percentile error, for our CNN scheme. Additionally, all CSI-based IPS in Table 11 except *SDR-Fi* and *ResNet1D* use the *CSI Tool* and *Atheros CSI Tool*, which rely on dedicated connections to a single AP and complex setup and firmware modifications to both the AP and NIC, as mentioned in Section II-A. Additionally, having all 56 SCs from the HT frame did not provide any substantial gain in accuracy against the 52 SCs from the reported *non-HT* frame in SDR mode [29], [30]. As an example, *BP* obtained mean error accuracy of 1.57m using a *representative testing methodology* and 56 SCs from *Atheros CSI Tool*, but we obtained 1.37m mean error using

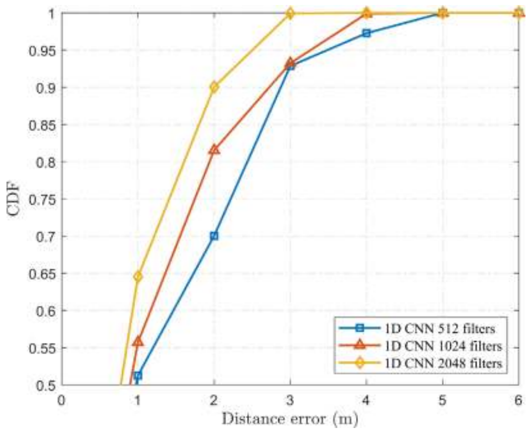


FIGURE 12. Indoor CDF distance error comparison for varying convolutional layer filters.

TABLE 11. **Self-reported accuracy comparison for representative WLAN IPS.**

Reference	Year	FP feature	Classification method	No. APs	Evaluation mode	Best reported accuracy
RADAR [9]	2000	RSS-based	KNN	3	RPs	Mean 2-3m
Horus [10]	2005	RSS-based	MLE	4-6	RPs	90th 1.4m
PinLoc [23]	2011	CSI-based (30 SCs)	MLE	1	TPs	Mean 1.1m
FIFS [22]	2012	CSI-based (30 SCs)	MLE	1[a]	RPs	Mean 2.5m[a]
CSI-MIMO [24]	2014	CSI-based (30 SCs)	MLE	1	single TP[b]	Mean 0.95m
PhaseFi [14]	2016	CSI-based (30 SCs)	DL	1	TPs	Mean 2.0m
BP [29]	2016	CSI-based (56 SCs)[c]	DL	1	TPs	Mean 1.57m
DeepFi [13]	2017	CSI-based (30 SCs)	DL	1	TPs	Mean 1.8m
BiLoc [27]	2017	CSI-based (30 SCs)	DL	1	TPs	Mean 1.57m
ConFi [18]	2017	CSI-based (30 SCs)	2D-CNN	1	TPs	Mean 1.37m
CiFi [26]	2018	CSI-based (30 SCs)	2D-CNN	1	TPs	Mean 1.7m
SFP [20]	2018	CSI-based (30 SCs)	MLE	1	TPs	50th 1.1m, 90th 2.5m
DfP [25]	2018	CSI-based (30 SCs)	MLE	3-4 pairs[d]	RPs	Mean 0.6m
DNNFi [30]	2018	CSI-based (56 SCs)[c]	DL	1	TPs	50th 1m, 90th 1.9m
DNN [15]	2019	CSI-based (30 SCs)	1D-CNN	1 pair[e]	RPs	99th 0.92m
ResNet1D [28]	2019	CSI-based (52 SCs)	1D-CNN	1[f]	RPs	95.68% accuracy
SDR-Fi (proposed)	2019	CSI-based (52 SCs)	1D-CNN	1	TPs	Mean 0.99m, 50th 0.77m

[a]The self-reported accuracy is for 1 AP scenario. FIFS also reported 0.65m mean accuracy for 6 APs scenario. Nonetheless, CSI-based accuracy for concurrent APs has not been reported, as the *CSItool* requires dedicated connection to a single AP at a time [31].
[b]This method was evaluated and self-reported accuracy on a single TP.
[c]Atheros CSI Tool with 56 SCs is used.
[d]This method uses transmitter-receiver pairs between fixed locations of APs and NICs. The device-free user is tracked via line-of-sight obstruction or variations. Thus, same RPs are used for training and testing.
[e]Same as [25], this method uses RPs for training and testing for a device-free user.
[f]This SDR uses two OFDM-based USRPs as transmitter and receiver, with clock synchronization.

52 SCs. In terms of classification, 1D CNN has the best performance overall, but any method is suitable. For number of APs, device-free methods such as in [25] and [15] require fixed location of APs and NICs, thus, requiring pairs of transmitters and receivers for the radio-map collection (see Section II-B). For the evaluation, some reported using the same RPs for training and testing, but we obtained centimeter accuracy in such tests since the classification for RPs and TPs is the same. *CSI-MIMO* reported a single TP for testing, but we evaluated our system with a single TP and obtained a mean error of 0.22m for our best TP. Hence, using several TPs renders a better statistical figure for the overall performance. Finally, the accuracy is reported in several metrics, but these numbers can vary among CDF values and the overall mean error. *ResNet1D* reported accuracy in percentage since they trained and evaluated the system with RPs only, but *SDR-Fi* obtained 99.99% accuracy with this same testing methodology. *ConFi* provided the second-lowest self-reported mean accuracy of 1.37m with a *representative testing methodology* similar to that proposed in this work.

VII. CONCLUSION AND FUTURE WORK

This paper presents *SDR-Fi*, a fast-prototyping SDR receiver capable of extracting RSS and CSI measurements passively from WLAN OFDM-based beacon frames for an IPS. As opposed to commercial NICs that only provide RSS, and hacked NIC versions, e.g., *CSI Tool,* that provide CSI from a restricted *sounding mechanism* and limited NIC vendors, *SDR-Fi* captures beacon frames in a passive manner. The receiver achieves packet collection rates up to 9.5 Hz and is able to capture from simultaneous visible APs. The demonstrated SDR capabilities for fast-prototyping evidently enhance potential research areas to assess real-life WLAN FP-based IPS with no modifications to the WLAN infrastructure.

In this work, we proposed two of the latest state-of-the-art deep learning models: a feed-forward neural network, and a 1D convolutional neural network. These models used partial and full SCs as features from *non-HT* beacon frames collected from the SDR. We used RPs for model training and TPs for online navigation. Hence, the models estimated the closest location for a TP sample. A good set of parameters was selected for the DL models through exhaustive experiments. We verified the efficacy of the models in location estimation compared to existing methods. We presented comparative results of our optimized 1D CNN and FFNN models against a CSI-based *DeepFi* method, and a RSS-based *Horus* method. Our 1D CNN model demonstrated an accuracy of 0.99 m and 0.63 m std. dev., and our FFNN model demonstrated an accuracy of 1.37 m and 0.87 m std. dev., both for an indoor cluttered scenario with a single AP. This one-meter accuracy is suitable for indoor navigation. Additionally, we analyzed comparative results on the impact for varying number of SCs, training samples, number of hidden layers for the FFNN, and number of filters for the 1D CNN. We concluded that the full number of SCs provided from *SDR-Fi* offer improved precision (std. dev.) over the commonly known *CSI Tool,* thus, strengthening fast-prototyping SDR research.

For future work, multiple AP scenarios are proposed, as the receiver can passively capture broadcast beacon frames from visible networks. The passive CSI measurement collection provides a possibility for vendor network cards to provide such measurements in the future; this can leverage future IPS deployment at a fractional cost. Furthermore, LSTM methods for human activity recognition can be explored with *SDR-Fi*. Methods related to sparse processing [48] can be applied to a 3D space model, i.e., SCs, APs, and RPs, respectively. Similarly, legacy WLAN technologies such as DSSS are also to be explored with techniques similar to those seen in previous work [12].

REFERENCES

[1] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*, 2nd ed. Lincoln, MA, USA: Ganga-Jamuna Press, 2006.

[2] A. Basiri, E. S. Lohan, T. Moore, A. Winstanley, P. Peltola, C. Hill, P. Amirian, and P. F. Silva, "Indoor location based services challenges, requirements and usability of current solutions," *Comput. Sci. Rev.*, vol. 24, pp. 1–12, May 2017.

[3] X. Li and K. Pahlavan, "Super-resolution TOA estimation with diversity for indoor geolocation," *IEEE Trans. Wireless Commun.*, vol. 3, no. 1, pp. 224–234, Jan. 2004.

[4] M. E. Rusli, M. Ali, N. Jamil, and M. M. Din, "An Improved indoor positioning algorithm based on rssi-trilateration technique for Internet of Things (IoT)," in *Proc. Int. Conf. Comput. Commun. Eng. (ICCCE)*, Kuala Lumpur, Malaysia, Jul. 2016, pp. 72–77.

[5] P. Kułakowski, J. Vales-Alonso, E. Egea-López, W. Ludwin, and J. García-Haro, "Angle-of-arrival localization based on antenna arrays for wireless sensor networks," *Comput. Electr. Eng.*, vol. 36, no. 6, pp. 1181–1186, Nov. 2010.

[6] X. Shen, K. Xu, X. Sun, J. Wu, and J. Lin, "Optimized indoor wireless propagation model in WiFi-RoF network architecture for RSS-based localization in the Internet of Things," in *Proc. Int. Topical Meeting Microw. Photon. Jointly held Asia–Pacific Microw. Photon. Conf.*, Singapore, Oct. 2011, pp. 274–277.

[7] S. He and S.-H. G. Chan, "Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 466–490, Jan. 2015.

[8] X. Gan, B. Yu, L. Huang, and Y. Li, "Deep Learning for Weights Training and Indoor Positioning Using Multi-sensor Fingerprint," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sapporo, Japan, Sep. 2017, pp. 1–7. doi: 10.1109/IPIN.2017.8115923.

[9] P. Bahl and V. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proc. 19th Annu. Joint Conf. IEEE Comput. Commun. Societies*, vol. 2, Mar. 2000, pp. 775–784.

[10] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proc. 3rd Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, Seattle, WA, USA, Jun. 2005, pp. 205–218.

[11] Z. Wu, C. Li, J. Ng, and K. Leung, "Location estimation via support vector regression," *IEEE Trans. Mobile Comput.*, vol. 6, no. 3, pp. 311–321, Mar. 2007.

[12] E. Schmidt and D. Akopian, "Fast Prototyping of an SDR WLAN 802.11b Receiver for an Indoor Positioning Systems," in *Proc. 31st Int. Tech. Meeting Satell. Division Inst. Navigat. (ION GNSS)*, Miami, FL, USA, Sep. 2018, pp. 674–684.

[13] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, Jan. 2017.

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

[15] C.-H. Hsieh, J.-Y. Chen, and B.-H. Nien, "Deep learning-based indoor localization using received signal strength and channel state information," *IEEE Access*, vol. 7, pp. 33256–33267, 2019.

[16] X. Wang, Z. Yu, and S. Mao, "DeepML: Deep LSTM for indoor localization with smartphone magnetic and light sensors," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.

[17] E. Schmidt, M. A. Mohammed, and D. Akopian, "A performance study of a fast-rate WLAN fingerprint measurement collection method," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 10, pp. 2273–2281, Oct. 2018. doi: 10.1109/TIM.2018.2819378.

[18] H. Chen, Y. Zhang, W. Li, X. Tao, and P. Zhang, "ConFi: Convolutional neural networks based indoor Wi-Fi localization using channel state information," *IEEE Access*, vol. 5, pp. 18066–18074, 2017.

[19] C. Nerguizian, C. Despins, and S. Affes, "Geolocation in mines with an impulse response fingerprinting technique and neural networks," *IEEE Trans. Wireless Commun.*, vol. 5, no. 3, pp. 603–611, Mar. 2006.

[20] Y.-J. Lin, P.-H. Tseng, Y.-C. Chan, J. He, and G.-S. Wu, "A super-resolution-assisted fingerprinting method based on channel impulse response measurement for indoor positioning," *IEEE Trans. Mobile Comput.*, to be published. doi: 10.1109/TMC.2018.2883092.

[21] C. Steiner and A. Wittneben, "Efficient training phase for ultrawideband-based location fingerprinting systems," *IEEE Trans. Signal Process.*, vol. 59, no. 12, pp. 6021–6032, Dec. 2011.

[22] J. Xiao, K. Wu, Y. Yi, and L. Ni, "FIFS: Fine-grained indoor fingerprinting system," in *Proc. 21st Int. Conf. Comput. Commun. Netw. (ICCCN)*, Munich, Germany, Jul./Aug. 2012, pp. 1–7. doi: 10.1109/ICCCN.2012.6289200.

[23] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka, "Precise indoor localization using PHY layer information," in *Proc. 10th ACM Workshop Hot Topics Netw. (HotNets-X)*, New York, NY, USA, Nov. 2011, Art. no. 18.

[24] Y. Chapre, A. Ignjatovic, A. Seneviratne, and S. Jha, "CSI-MIMO: Indoor Wi-Fi fingerprinting system," in *Proc. 39th Annu. IEEE Conf. Local Comput. Netw.*, Edmonton, AB, USA, Sep. 2014, pp. 202–209.

[25] S. Shi, S. Sigg, L. Chen, and Y. Ji, "Accurate location tracking from CSI-based passive device-free probabilistic fingerprinting," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5217–5230, Jun. 2018.

[26] X. Wang, X. Wang, and S. Mao, "Deep convolutional neural networks for indoor localization with CSI images," *IEEE Trans. Netw. Sci. Eng.*, to be published. doi: 10.1109/TNSE.2018.2871165.

[27] X. Wang, L. Gao, and S. Mao, "BiLoc: Bi-modal deep learning for indoor localization with commodity 5GHz WiFi," *IEEE Access*, vol. 5, pp. 4209–4220, 2017.

[28] F. Wang, J. Feng, Y. Zhao, X. Zhang, S. Zhang, and J. Han, "Joint activity recognition and indoor localization with WiFi fingerprints," *IEEE Access*, vol. 7, pp. 80058–80068, 2019.

[29] J. Li, Y. Li, and X. Ji, "A novel method of Wi-Fi indoor localization based on channel state information," in *Proc. 8th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Yangzhou, China, Oct. 2016, pp. 1–7. doi: 10.1109/WCSP.2016.7752710.

[30] G.-S. Wu and P.-H. Tseng, "A deep neural network-based indoor positioning method using channel state information," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Maui, HI, USA, Mar. 2018, pp. 290–294. doi: 10.1109/ICCNC.2018.8390298.

[31] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: Gathering 802.11n traces with channel state information," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 1, p. 53, Jan. 2011.

[32] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity Wi-Fi," in *Proc. ACM Mobicom*, Paris, France, Sep. 2015, pp. 53–64. doi: 10.1145/2789168.2790124.

[33] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, Standards 802.11-2012, 2012.

[34] E. Schmidt, Z. Ruble, D. Akopian, and D. J. Pack, "Software-defined radio GNSS instrumentation for spoofing mitigation: A review and a case study," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 8, pp. 2768–2784, Aug. 2019. doi: 10.1109/TIM.2018.2869261.

[35] M. T. Hoang, B. Yuen, X. Dong, T. Lu, R. Westendorp, and K. Reddy, "Recurrent Neural Networks For Accurate RSSI Indoor Localization," Mar. 2019, *arXiv:1903.11703*. [Online]. Available: https://arxiv.org/abs/1903.11703

[36] J.-S. Choi, W.-H. Lee, J.-H. Lee, J.-H. Lee, and S.-C. Kim, "Deep learning based NLOS identification with commodity WLAN devices," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3295–3303, Apr. 2018.

[37] S. Yousefi, H. Narui, S. Dayal, S. Ermon, and S. Valaee, "A survey on behavior recognition using WiFi channel state information," *IEEE Commun. Mag.*, vol. 55, no. 10, pp. 98–104, Oct. 2017.

[38] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proc. IEEE*, vol. 93, no. 2, pp. 216–231, Feb. 2005.

[39] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Trans. Commun.*, vol. 45, no. 12, pp. 1613–1621, Dec. 1997.

[40] P. H. Moose, "A technique for orthogonal frequency division multiplexing frequency offset correction," *IEEE Trans. Commun.*, vol. 42, no. 10, pp. 2908–2914, Oct. 1994.

[41] P. Karn. *Forward Error Correcting Codes*. Accessed Aug. 15, 2018. [Online]. Available: http://www.ka9q.net/code/fec/

[42] T. MathWorks. *Deep Learning Toolbox*. Accessed: Sep. 5, 2018. [Online]. Available: https://www.mathworks.com/products/deep-learning.html

[43] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Netw.*, vol. 6, no. 4, pp. 525–533, Nov. 1993, doi: 10.1016/S0893-6080(05)80056-5.

[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012. doi: 10.1145/3065386.

[45] N. S. Keskar and R. Socher, "Improving generalization performance by switching from adam to SGD," Dec. 2017, *arXiv:1712.07628*. [Online]. Available: https://arxiv.org/abs/1712.07628

[46] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Ann. Oper. Res.*, vol. 134, no. 1, pp. 19–67, 2005.

[47] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.

[48] A. Khalajmehrabadi, N. Gatsis, D. Pack, and D. Akopian, "A joint indoor WLAN localization and outlier detection scheme using LASSO and elastic-net optimization techniques," *IEEE Trans. Mobile Comput.*, vol. 16, no. 8, pp. 2079–2092, Aug. 2017.

ERICK SCHMIDT (S'17) received the B.S. degree (Hons.) in electronics and computer engineering from the Monterrey Institute of Technology and Higher Education, Monterrey, Mexico, in 2011, and the M.S. degree from The University of Texas at San Antonio, San Antonio, TX, USA, in 2015, where he is currently pursuing the Ph.D. degree in electrical engineering.

From 2011 to 2013, he was a Systems Engineer with Qualcomm Incorporated, San Diego, CA, USA. His current research interests include software-defined radio, indoor navigation, global navigation satellite systems, spoofing mitigation algorithms, and fast-prototyping methods and accelerators for baseband communication systems.

Mr. Schmidt is a Student Member of the Institute of Navigation.

DEVASENA INUPAKUTIKA (S'18) received the B.Tech. degree in electronics and communications engineering from MITS University, India, in 2010, and the M.Sc. degree in robotics and automation from the University of Salford, Manchester, U.K., in 2013. She is currently pursuing the Ph.D. degree in electrical engineering with The University of Texas at San Antonio, TX, USA.

From 2010 to 2012, she was a Software Engineer with Accenture, India, and from 2013 to 2017, she was a Research Software Engineer at Southampton, U.K. Her current research interests include mobile -cloud computing and algorithms, wearable technology and IoT, and web and mobile application development.

RAHUL MUNDLAMURI received the bachelor's degree in electronics and communications engineering from JNT University, India, in 2014, and the master's degree from the University of Houston, Houston, TX, USA, in 2016. He is currently pursuing the Ph.D. degree in electrical engineering with The University of Texas at San Antonio, TX.

From 2017 to 2018, he was a Data Engineer with BGS Technologies, Austin, TX. His current research interest includes software-defined radio.

DAVID AKOPIAN (M'02–SM'04) received the Ph.D. degree from the Tampere University of Technology, Finland, in 1997. He was a Senior Research Engineer and a Specialist with Nokia Corporation, from 1999 to 2003. From 1993 to 1999, he was a Researcher and an Instructor with the Tampere University of Technology. He is currently a Professor with The University of Texas at San Antonio (UTSA). He authored and coauthored more than 30 patents and 140 publications. His current research interests include digital signal processing algorithms for communication and navigation receivers, positioning, dedicated hardware architectures and platforms for software-defined radio, and communication technologies for healthcare applications.

He is elected as a Fellow of US National Academy of Inventors, in 2016. He served in organizing program committees of many IEEE conferences and co-chairs an annual conference on Multimedia and Mobile Devices. His research has been supported by National Science Foundation, National Institutes of Health, USAF, US Navy, and Texas foundations.