CSIscan: Learning CSI for Efficient Access Point Discovery in Dense WiFi Networks

Kunal Sankhe, Dheryta Jaisinghani, and Kaushik Chowdhury
Institute for the Wireless Internet of Things, Northeastern University, Boston, MA, USA
Emails: sankhe.ku@husky.neu.edu, dheryta@ieee.org, and krc@ece.neu.edu

***Abstract*—Network densification through the deployment of WiFi access points (APs) is a promising solution towards achieving high connectivity rates required for emerging applications. A critical first step is to discover an AP before an active association between the client and the AP can be established. Legacy AP discovery procedures initiated by the client result in high latency in the order of a few 100 ms and waste spectrum, especially when clients need to frequently switch between multiple APs. We propose CSIscan that exploits the broadcast nature of WiFi channels by embedding discovery related information within an AP's ongoing regular transmissions. The AP does this by intelligently distorting the transmitted OFDM frame by inducing perturbations in the preamble, and these injected 'bits' of information are detected via changes in the perceived channel state information (CSI). A deep learning framework allocates the optimal level of distortion on a per-subcarrier basis that keeps the resulting packet error rate to less than 1%. Existing clients perceive no changes in their ongoing communication, while potential new clients quickly obtain discovery information at the same time. We experimentally demonstrate that CSIscan reduces the overall WiFi latency from 150 ms to 10 ms and improves spectrum utilization with $\sim 72\%$ reduction in the probe traffic. We show that CSIscan delivers up to 40 discovery information bits in the outgoing WiFi packet in an indoor environment.**

I. Introduction

Given the growth of multimedia applications, data traffic is poised to increase annually at an unprecedented rate. Cisco predicts the majority of the total IP traffic by 2022 will be over WiFi [1], thus making it the dominant wireless technology of the future. To support this traffic, total public WiFi hotspots will grow globally to 549 million by 2022, suggesting that network densification is inevitable. While upcoming standards, such as 802.11ax [2] improve spectrum utilization through efficient time/frequency resource allocation via OFDMA, higher-order modulation schemes like 1024-QAM, spatial reuse via multi-user(MU) MIMO, and interference management via basic service set coloring, a crucial first step involved in discovering APs has not evolved over a decade. Relying on costly and legacy handshake of standard management packets, such as beacons and probes for the discovery of APs, will not scale with network densification in the coming years. For example, in *passive discovery*, an AP broadcasts beacons typically in 100 ms intervals that quickly adds up in hot spots with hundreds of APs. In *active discovery* the AP must respond to every probe request sent by potential clients, creating a management traffic flooding situation [3]. The problem exacerbates in crowded or ultra-dense environments, such as sports arenas and large conferences. As reported in [4]–[6], excessive attempts to discover APs not only lead to severe energy wastage in client devices but also reduce the overall WiFi throughput.Thus, a fundamental change is needed in the way clients discover APs. The proposed CSIscan takes a step towards this direction by addressing the problems of *low spectrum utilization* and *WiFi latency*.

978-1-7281-6992-7/20/$31.00 ©2020 IEEE

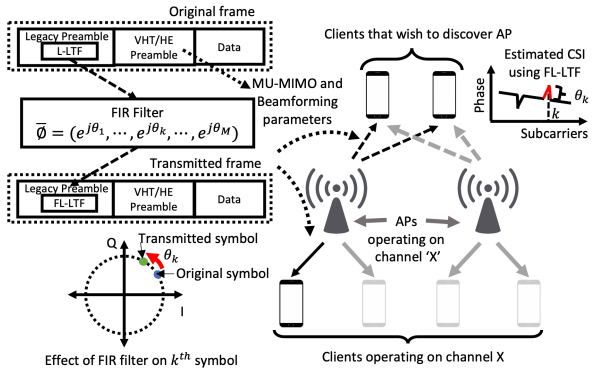


Figure 1: Overview of CSIscan: A WiFi network with APs and clients operating on channel X. AP embeds discovery information by controlled perturbation of the legacy preamble (L-LTF) through an FIR filter. Clients that wish to discover and associate with an AP analyze CSI to extract discovery information.

A. Spectrum Utilization, WiFi Latency

A measurement campaign in our university campus (Sec. II) showed that up to $\sim$19% of the spectrum airtime is occupied by management traffic; of which $\sim$80% are merely redundant AP discovery packets. To increases the discovery likelihood at distant locations, these packets are sent at the lowest modulation thereby reducing the effective bit rate. As the discovery traffic grows over time, the efficiency of spectrum utilization is considerably reduced. Despite tremendous growth in PHY data rates, high WiFi latency for a client to reach AP remains a bottleneck for emerging standards. Recent studies show 25 ms of WiFi latency causes 2.5 seconds delay in page load time for popular websites such as, Google and Facebook [7], which is unacceptable to most users [8]. Therefore, we examine the impact of active discovery, an inevitable precursor to establish a WiFi connection, on the WiFi latency. Legacy AP discovery methods are shown to be time-consuming [9], [10]. This problem further worsens in standards, such as 802.11ad/ay, that increase the number

of channels to be scanned in tri-band (2.4/5/60 GHz) APs or require beamforming with precise mapping of Tx and Rx antenna sectors during the beam-searching algorithm. Thus, triggering discovery during ongoing communication results in spikes in latency, affecting the QoS of existing applications [11]. A simple experimental setup in Sec. II-B shows the problems faced by clients in sub-6GHz dual-band operation. WiFi latency increases by up to 30x when the client triggers active discovery. This shows that legacy AP discovery will not scale as new unlicensed or shared spectrum gets added to existing WiFi bands; worsening the problem for high-density network deployments with a large number of APs present in close proximity, and where several thousands of WiFi clients keep switching APs to get better connectivity.

B. Proposed Solution

We propose CSIscan, wherein the AP encodes information by introducing subtle perturbations within the preamble at the PHY layer for all outgoing packets. These modifications are achieved by introducing phase shifts $[-\phi_{max}, \phi_{max}]$ in symbols of the maximum number of OFDM subcarriers N_{max} (see Fig. 1) The connection of already associated clients is not impacted with packet error rate (PER) staying within 1%. However, new clients do not need to rely on discovery packets, as *any* outgoing packet from AP has relevant discovery information encoded. Following this, the client sends a targeted discovery request in a specific channel, after analyzing all the discovery information in modified packet preambles. While the proposed CSIscan is suitable for AP discovery, we envision that embedding information in the preamble has a wide range of applications that have stringent timing constraints on latency. For example, CSIscan can deliver time-critical emergency messages to industrial robots in the domain of Industrial IoT with latency constraints of < 10ms.

•**Research Challenges in CSIscan:** CSIscan introduces modifications to the long training field (L-LTF) portion of the preamble, available in the latest as well as legacy standards, at the AP-side by passing corresponding IQ samples through a finite impulse response (FIR) filter. At the client side, these changes are observed in the estimated CSI, i.e., information is decoded by analyzing the locations and height of CSI phase patterns. The first research question we tackle is how to map the location (index of the modified subcarrier) and height (giving the phase shift by which the preamble symbol is distorted) in the CSI phase pattern reliably to specific filter coefficients used by the AP. Secondly, modifications to the legacy preambles through FIR filtering has a similar effect to impairments caused by the wireless fading channel. CSIscan does not alter traditional channel equalization algorithms used by the client that compensate the combined effect of both natural and synthetic impairments. Therefore, we need to distort IQ samples in a principled manner, such that these changes are not mistaken for random channel effects, and yet adequately compensated through equalization. For an ideal, static channel, CSIscan finds a deterministic bound (i.e., preset ϕ_{max} and N_{max}) on the permissible modification to the IQ samples to guarantee the PER constraint of 1%.

•**Role of Learning in CSIscan:** To address these challenges, CSIscan leverages a Convolutional Neural Network (CNN) that exploits spatial and temporal shift-invariant features to learn the channel and provides as an output the optimal combination of N_{max} and ϕ_{max}. This ensures the maximum number of bits are encoded without impacting the PER of transmissions. CSIscan trains the CNN with CSI measurements of ongoing transmissions to select N_{max} and ϕ_{max} at the AP. This allows the AP to automatically adapt its information overlay rate that benefits the discovery process. Deploying CSIscan in the real world network requires retraining of the CNN, which is a time-consuming and resource-hungry process. It is often impractical to collect and store vast amounts of CSI measurements to use them as training data on-site. We overcome this issue through *supervised domain adaptation*, a sub-discipline of machine learning that deals with scenarios in which a model trained on a source distribution is used in a different (but related) target distribution [12]. We first train the CNN using the CSI measurements by passing the data transmissions through a rich set of channels simulated using indoor channel models. In the actual network deployment, we fine-tune the CNN model through domain adaptation using the model trained with simulated data and partially retrain it.

C. Summary of Contributions

1) We present CSIscan, a spectrum-efficient and low-latency AP discovery method for WiFi clients. The key innovation behind CSIscan is to overlay discovery information by inducing synthetic IQ variations into the legacy preambles of ongoing transmissions from APs without impacting its PER beyond the pre-set threshold. We develop an encoding scheme to map discovery information into the coefficients of an FIR filter used by the AP. (Section III)
2) We design a CNN to determine the optimal selection of N_{max} and ϕ_{max} under varying channel conditions. For a realistic network deployment, we train the CNN model using supervised domain adaptation. This reduces average training time by 64 ms and achieves 93% prediction accuracy, even with limited available training data. (Section IV)
3) We implement a proof-of-concept testbed to demonstrate and evaluate CSIscan using a USRP based AP and client in realistic indoor environments. CSIscan improves the spectrum utilization of the network by reducing 72% of the discovery traffic. The long-tail (99^{th} percentile) WiFi latency at the client is reduced by 95%. CSIscan successfully delivers up to 40 bits in each outgoing packet to clients as distant as 16 ft from the AP without reducing PER beyond 1%. (Section V)

II. Study of AP-client Discovery

We conduct a measurement study to provide real-world evidence of problems due to active discovery and how CSIscan tackles them.

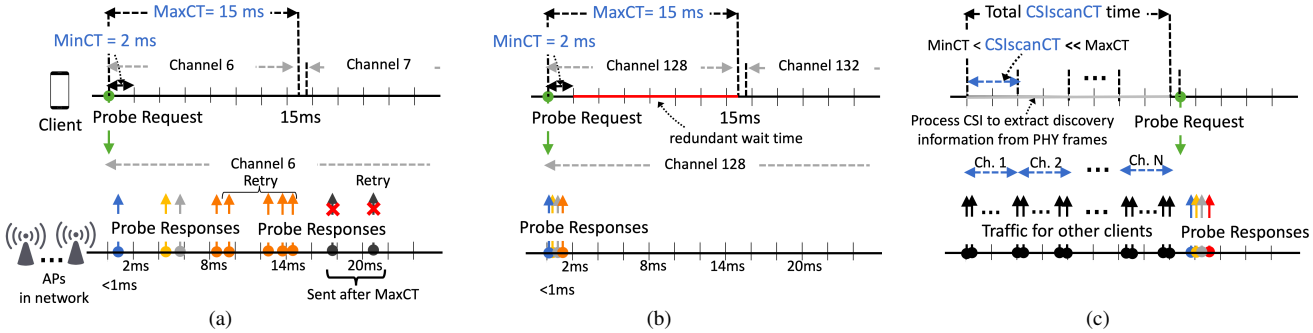


Figure 2: Demonstrating Active Discovery and CSIscan. a) In 2.4 GHz, probe responses are received at the client after MaxCT, most of which are retried. b) In 5 GHz, most probe responses arrive within MinCT at the client, resulting in unnecessary wait times. c) In CSIscan, the client decodes CSI from preambles of all packets, to choose one strong channel, and sends a single probe request on that channel only.

A. Active Discovery

Fig. 2a and 2b show the process of active discovery in 2.4 and 5 GHz, respectively. The client sends a probe request on a particular channel and waits for at least MinChannelTime (MinCT) to detect any channel activity. If the client senses any activity during MinCT, it waits until MaxChannelTime (MaxCT) to potentially receive probe responses from operational APs. After MaxCT, the client switches to the next channel and sends a new probe request. MinCT and MaxCT are timing parameters set in the client device that directly impacts its AP discovery time and number of discovered APs. Many prior works propose strategies to find optimal bounds on MinCT and MaxCT [13], [14]. These strategies are client-centric to expedite the discovery and do not reduce low bit rate probe traffic in the network. Further, the optimal selection of these parameters is application dependent as well as dynamics of the network including the deployment and load [15]. Stationary clients have been shown to trigger active discovery frequently resulting in fake handovers and injecting unnecessary probe traffic in the network [4], [16]. Next, we study the impact of active discovery from two perspectives: (i) the spectrum wastage and (ii) the latency at the client.

B. Impact of Active Discovery

1) Spectrum Wastage: We collect network traffic traces in a university building that has 35 WiFi APs and $\approx$ 200 students at any given time. We use a laptop with Ubuntu 16.04 OS and Atheros AR9464 WiFi chipset in monitor mode, and without loss of generality, we capture traffic on channel 6 (2.4 GHz) and channel 128 (5 GHz). Overall, we observe an average of 3917 and 992 probe packets per minute, in 2.4 GHz and 5 GHz, respectively. We delve deeper to understand the significance of these numbers by studying the collected traffic in sub-traces of 22ms duration each. We choose this duration empirically to determine the statistic for the probe traffic and metrics such as inter-packet arrival time, by processing traffic files with minimal computational time. We observe:

At 2.4 GHz – 62% of the probe responses reach a client after MaxCT. This shows that in a network with high congestion or dense deployment, the AP may not find transmission opportunities before MaxCT, despite hearing probe request from the client. Since after MaxCT, the client moves on to the next channel, probe responses sent by APs waste spectrum. Next, 75% of the total probe responses are retry packets. In a highly congested network, either the probe response or the corresponding acknowledgment (ACK) may be lost as shown in Fig. 2a. Without an ACK, AP retransmits the probe response, further congesting the network. During an active discovery, a probe request from the client on one channel can be unintentionally decoded in adjacent channels. Upon hearing such requests, not only APs operating on the client's chosen channel but, also others in adjacent channels reply with probe responses; thereby, exaggerating the amount of probe traffic. These problems in 2.4 GHz reduce the overall spectrum utilization. Standards operating in 5 GHz bands may circumvent this issue partly by using fully non-overlapping channels. However, future standards like 802.11ax support 2.4 GHz as well. Therefore, the problem needs to be addressed.

At 5 GHz – Interestingly, we observe that 72% probe responses from APs were transmitted within MinCT after the client triggers the probe request, as shown in Fig. 2b. This is because most of 5 GHz clients use 6 Mbps rate to transmit these probe packets. However, here the client continues to listen on the channel for MaxCT despite receiving most of the probe response within MinCT. Thus, the client wastes a significant amount of airtime in waiting for probe responses.

2) WiFi Latency at the client: Consider a client and an AP, where the former triggers active discovery every 2 minutes. We study the impact of the interleaved discovery initiation on the latency observed at the client to reach the AP. We disable all background traffic as well as the default network managers to obtain an accurate measurement. We use the Ping [17] utility to record latency [11]. Fig. 3 plots CDF of the WiFi latency at the client when active discovery is – i) not triggered, ii) triggered periodically. With no active discovery, the latency remains at a median value of $\sim$5.5 ms; whereas with periodic active discovery, it is $\sim$ 5.7 ms. However, the 97^{th} percentile and 99^{th} percentile are around 35 ms and 125 ms, respectively with the latter. The zoomed-in inset shows that the latency shoots up from 20 ms to 150 ms immediately after active discovery is triggered. Unfortunately, we observe increased latency for the next few seconds as well. This is due to momentary deferring of discovery packets, because kernel transmit/receive queues are filled-up with ping packets. The

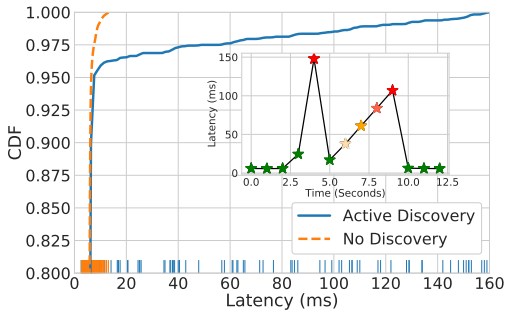


Figure 3: WiFi latency at the client to reach the AP. In the absence of active discovery, the latency remains within 15 ms; however, when the discovery is triggered it shoots up to 150 ms.

long tail of latency caused by the active discovery is critical because it can lead to multiple seconds of application-level delays resulting in longer page loads, disrupt real-time, or time-critical services like VoIP, ultimately degrading the users' quality of experience [18].

C. Motivation for CSIscan for WiFi AP Discovery

As opposed to receiving a beacon every 100 ms, or sending a probe request every 15 seconds [19], a client in densely populated WiFi network receives any MAC layer packet every 250 μs. Irrespective of the packet type, a WiFi client performs preamble detection and CSI estimation for decoding every transmitted packet. Then, it extracts MAC layer information from the PHY layer payload to decide if the packet is destined for the client. If not, then the client discards the packet. CSI estimation is performed to negate the channel distortions caused by multipath and fading. CSIscan exploits the broadcast nature of WiFi channels with a simple idea – *Since every WiFi client decodes the preamble of every packet transmission, the client no longer needs to wait for a beacon or probe if the AP embeds discovery information in these preambles.* This approach makes discovery packets redundant, instead it leverages packets of any type that are anyways transmitted. The subtle perturbations to the preamble at the PHY introduced by the APs cause variations in CSI, which are estimated by the client and used to extract discovery-related information. Of course, any distortion of the regular packet preamble must not impact the packet-error-rate of the ongoing communication.

The client listens on each channel for CSIscan channel time (`CSIscanCT`) and obtains discovery-related information by passively processing CSI of incoming packets on that channel (see Fig. 2c). The value of `CSIscanCT` is derived from measuring the median of the inter-packet arrival time of traffic at an AP and broadcast in its beacons. After processing the discovery information from all the channels, the client selects an appropriate channel to transmit a single probe request to discover a viable AP on that channel; thereby, limiting the number of probe responses. Thus, CSIscan reduces spectrum wastage due to excessive probe traffic by eliminating redundant probe traffic on other channels, and enabling a client to spend time considerably lower than `MaxCT`.

III. Design of CSIscan

In this section, we discuss how CSIscan modifies the outgoing packets, the challenges associated with implementing CSIscan, and how learning helps to solve them.

A. Encoding and Decoding with CSIscan

1) Intentional modifications at AP using FIR filtering: A WiFi transmitter includes a known LTF sequence in the preamble that is leveraged at a receiver to estimate CSI using channel estimation methods [20]. CSI estimation is a prerequisite step for data decoding that quantifies the distortions encountered by wireless signals due to fading and shadowing. Since Legacy LTF (L-LTF) is a mandatory field in all outgoing packets irrespective of standards, CSIscan leverages L-LTF towards developing a standard-independent information embedding technique.

CSIscan modifies L-LTF in real-time using discrete causal FIR filters. These filters are advantageous because they – i) do not depend on future inputs, only on past and present ones, ii) are represented as a weighted and finite term sum enabling accurate prediction of the output for any given input. CSIscan designs FIR filter in the frequency-domain that is described as a finite sequence of M complex filter coefficients. As L-LTF is composed of two time-domain OFDM symbols, we first convert the filter into the time-domain using inverse FFT to create its impulse response and then apply it through convolution with L-LTF symbols. The filter coefficients only distort the phase of L-LTF symbols at the subcarrier level and not their amplitude. This is required, as amplitude distortion changes the average power of the transmitted signal that adversely affects ongoing communication.

2) Mapping discovery information into FIR filter coefficients: We define the FIR filter $\bar{\phi}$ with 64 complex coefficients, i.e., $\bar{\phi} = [\phi_{-32}, \cdots, \phi_{-1}, \phi_0, \phi_1, \cdots, \phi_{31}]$, where $\phi_k = e^{\frac{j\pi\theta_k}{180}}$ and θ_k is the phase shift (in degrees) introduced in k^{th} subcarrier of observed CSI post-filtering. We match the indexing scheme $\{-32, \cdots, 31\}$ to that of 802.11 OFDM subcarrier mapping. We choose 64 coefficients, since the legacy preamble in 802.11ax/ac is always constructed using 64 subcarriers in 20 MHz of OFDM channel.

Each L-LTF OFDM symbol is constructed by mapping a known sequence of ± 1 into 52 out of 64 subcarriers during IFFT operation, whereas the remaining 12 are null subcarriers. CSIscan intentionally injects phase distortions in this sequence at the subcarrier level by carefully choosing coefficients of $\bar{\phi}$. CSIscan treats 12 indexes as null subcarriers ($\{-32, \cdots, -27, 0, 27, \cdots, 31\}$) as *don't care*. This implies that information bits are mapped in the remaining 52 subcarriers with indexes $\{-26, \cdots, -1, 1, \cdots, 26\}$. We divide 52 subcarriers into two fields a) `discovery-rate` and b) `discovery-data`. The `discovery-rate` conveys chosen bounds for phase shifts and the number of subcarriers that are essential to design the FIR filter at the AP and decode embedded bits at clients. These bounds ultimately determine the total number of embedded discovery bits, henceforth referred to as rate of discovery information. The

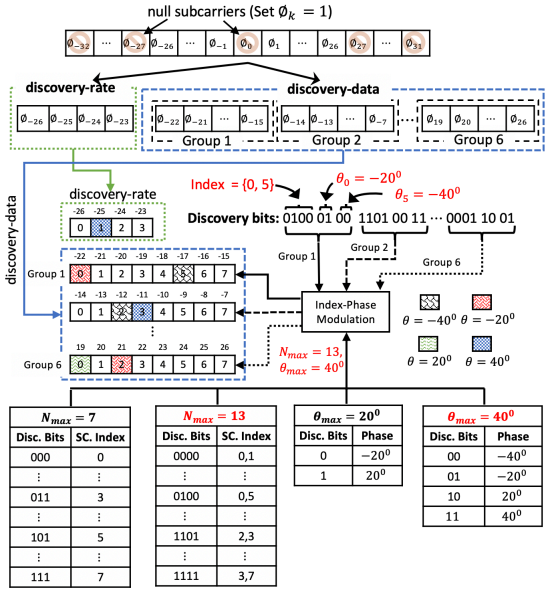


Figure 4: Illustration of CSIscan encoding scheme. Discovery bits 01000100 are mapped in subcarriers with indexes from -22 to -15. The first 4 bits 0100 indicate locations of subcarriers, i.e., 0 and 5 within the group of 8 subcarriers (refer mapping table for N_{max} = 13). Subsequent bits 01 represents the phase shift of -20° while 00 represents the phase shift of -40° at 0^{th} and 5^{th} subcarriers, respectively (see table for $\theta_{max} = 40^\circ$).

`discovery-rate` field spans first 4 subcarriers with indexes $\{-26, -25, -24, -23\}$. The `discovery-data` field spans the remaining subsequent 48 subcarriers with indexes $\{-22, \cdots, 26\}$. CSIscan selects a subset of subcarriers from this field to encode discovery bits. Some examples of information that can be embedded in the discovery bits are AP load and channel utilization.

We further divide `discovery-data` field into 6 groups sequentially, with each group consisting of 8 subcarriers. This is done to reduce the computational complexity of decoding operation at clients. Consider, N_{max} and θ_{max} are the upper bounds selected while designing the FIR filter to keep the PER of ongoing communication under control. Out of the total number of subcarriers N_{max}, we reserve one subcarrier in `discovery-rate` field. We divide the remaining $N_{max}-1$ subcarriers into 6 groups, i.e, $L = \lfloor \frac{N_{max}-1}{6} \rfloor$ subcarriers in each group. Consider, $\boldsymbol{\theta} = \{\theta_1, \theta_2, \cdots, \theta_M\}$ is the set of feasible phase shifts, such that $|\theta_i| \leq \theta_{max}$ for $i \in 1, \cdots, M$. We define separate mapping tables for N_{max} and θ_{max}. The table for N_{max} maps a block of bits to the position of subcarriers within a group, whereas the table for θ_{max} maps bits to a phase shift from the feasible set. CSIscan maps a block of 'k' information bits into different phase shifts in various subcarrier locations within a group of 8 subcarriers using the mapping table of N_{max} and θ_{max}, where $k = \lfloor \log_2 \binom{8}{L} \rfloor + L \times \lfloor \log_2 \binom{M}{1} \rfloor$, and M is the cardinality of the set $\boldsymbol{\theta}$. Thus, CSIscan transmits $R = 6{*}k = 6{*}\big(\lfloor \log_2 \binom{8}{L} \rfloor + L \times \lfloor \log_2 \binom{M}{1} \rfloor\big)$ discovery bits over each filtered packet.

In Fig. 4, we illustrate the encoding scheme by choosing $N_{max} = 13$ and $\theta_{max} = 40^\circ$. Now, we have $L = \lfloor \frac{13-1}{6} \rfloor = 2$ subcarriers in each group. Let $\boldsymbol{\theta} = \{-40^\circ, -20^\circ, 20^\circ, 40^\circ\}$ is the feasible set. Using mapping table for $N_{max} = 13, \theta_{max} = 40^\circ$, CSIscan maps the block of $k = \lfloor \log_2 \binom{8}{2} \rfloor + 2 \times \lfloor \log_2 \binom{4}{1} \rfloor = 8$ discovery information bits into subcarrier indexes and phase shifts. First, $\lfloor \log_2 \binom{8}{2} \rfloor = 4$ bits decide the indexes of $L = 2$ subcarriers. For each selected subcarrier, we choose the next $\lfloor \log_2 \binom{4}{1} \rfloor = 2$ bits to select the phase shifts to be introduced into a sequence of ± 1. For the illustrative example shown in Fig. 4, FIR filter coefficients for corresponding '0^{th}' and '5^{th}' subcarriers from the first group are selected as $\phi_{-22} = e^{\frac{j\pi(-20^\circ)}{180}}$ and $\phi_{-17} = e^{\frac{j\pi(-40^\circ)}{180}}$. We select 0° phase shifts for the remaining subcarriers.

3) Decoding using CSI phases: For decoding the packet, recall that a WiFi client first estimates CSI using filtered L-LTF signal and extracts embedded information in the phases of the subcarriers by processing CSI phases. While raw CSI phases change periodically between $[-\pi, \pi]$, unwrapped CSI phases for a stable wireless channel are nearly linear. The client performs a statistical detrending operation on CSI phases without impacting phase shifts caused by FIR filtering. The detrending operation first identifies a linear trend in CSI phases by approximating into a best straight-fit line using the least square method and later subtracts it from original CSI phases.

Fig. 5 shows unwrapped CSI phases after the detrending operation, where we observe the jumps in CSI phases at certain subcarriers. To decode the information, the client uses the same groups of the subcarriers that are used while encoding. In each group of 8 subcarriers, the client estimates phase shifts at each subcarrier using thresholds. The presence of phase shifts indicates the indexes of the subcarriers within the group where L-LTF symbol is intentionally distorted. The client demaps the estimated phase shifts and the indexes of subcarriers to recover information bits using the same mapping table used during encoding. If any mismatch is found between the subcarrier indexes and their corresponding phase shifts with the entries in the mapping table, the discovery information is discarded.

In Fig. 5, using threshold of $\pm 10^\circ$, $\pm 30^\circ$, we segregate the phase shift in each subcarrier into -40°, -20°, 0°, 20°, 40°. We show phase shifts other than 0° with red markers in the figure. For example, the marker with the label $(-22^\circ, -17.5^\circ)$, indicates the phase shift of -17.5° introduced at the -22^{nd} subcarrier index. This phase shift -17.5° is approximated to

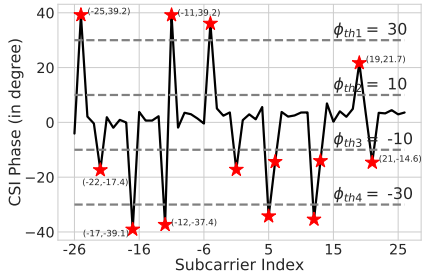
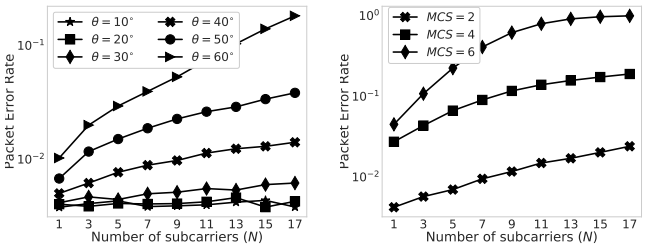


Figure 5: CSI phases estimated using filtered L-LTF. These phases represent the phase shifts (red markers) in L-LTF symbols at subcarriers introduced with FIR filter.



(a) (b)

Figure 6: a) PER is directly proportional to the increase in θ and number of subcarriers for a fixed MCS value. b) PER also increases with increase in MCS value for a fixed θ.

-20° using thresholds. We observe exact -20° phase shifts at -22^{nd} subcarrier in Fig. 4. Thus, WiFi clients extract discovery information by processing CSI phase information.

Next, we discuss the challenges and the solution to enhance the reliability of CSIscan under varying channel conditions.

B. Effective CSI Decoding via Learning

Off-the-shelf WiFi receivers are incapable of fully compensating the channel distortions. Synthetic distortions in pilot subcarriers disturb the tracking of phase noise, which is caused by oscillator irregularities and compensated using tracking algorithms [21]. These two factors ultimately impact PER observed at the receiver, defined as the ratio between the number of packets in error to the total number of transmitted packets.CSIscan must find an upper bound on synthetic distortions caused by FIR filtering to keep PER in control. We consider a single bit error as a packet error to study the worst-case performance of varying number of subcarriers N_{max}, phases θ_{max}, and modulation and coding schemes (MCS) on PER using simulated dataset. A transmitter generates PHY layer packets of fixed MCS index value containing legacy preamble, VHT preamble, and MAC layer data. We consider an average signal-to-noise ratio (SNR) of 30 dB at the receiver for each transmission. The receiver decodes all received packets and compares the decoded bits in each packet with the transmitted bits to determine the PER. We iterate this experiment for 1 million channel realizations for different configurations of N_{max} and θ_{max}.

1) Impact of varying N_{max}, θ_{max}, MCS on PER: Fig. 6a shows the average PER for a fixed MCS = 2 (QPSK, 3/4 coding rate) with respect to the number of subcarriers N_{max} for different phase shifts θ_{max}. For a fixed θ_{max}, we observe increase in PER with respect to N_{max}. Further, for a fixed N_{max}, PER also increases with θ_{max}. For a given PER constraint, this plot presents an upper bound on N_{max} and θ_{max} to be used by AP while encoding discovery information. We repeat above experiment by a varying MCS index of the generated PHY packets and the number of subcarriers, while introducing the constant phase shift of $\theta_{max} = 40^\circ$. Fig. 6b shows the results. As expected, the PER increases as the MCS increases. However, the dependency of PER on MCS imposes CSIscan to take MCS also into account while determining the upper bounds on N_{max} and θ_{max}. Further, the PER also varies with respect to the average SNR at the receiver, implying that it too must be considered while choosing N_{max} and θ_{max}.

For a static wireless channel, CSIscan can find deterministic bounds on N_{max} and θ_{max} to maintain PER at a WiFi client within a permissible range. However, these bounds are difficult to estimate under time-varying channels. The pessimistic approach of embedding the least number of discovery bits by choosing the least supported values of N_{max} and θ_{max} is undesirable. For instance, using bounds $N_{max} = 1$ and $\theta_{max} = 10^\circ$, CSIscan can convey just 3 bits of discovery information in each packet. However, it misses the opportunity of embedding > 3 bits of discovery information in more than 99% packets (PER $< 1\%$ for $N_{max} = 1$ and $\theta_{max} = 10^\circ$ as shown in Fig. 6a). On the other hand, an overly optimistic approach of always transmitting the maximum number of discovery bits could severely hamper the ongoing communication of the client. For instance, the choice of bounds $N_{max} = 17$ and $\theta_{max} = 60$ will severely degrade the communication of the client with the PER $> 30\%$ as shown in Fig. 6a.

2) Leveraging Learning: In order to maximize the opportunity of embedding discovery bits with the preamble, we need to find an optimal combination of N_{max}, θ_{max}. While various throughput maximization strategies are proposed in the wireless domain [22], [23], they are ill-suited for more practical and dense WiFi deployments. Most of these approaches consider statistical models of the environment, such as fading channel that are close approximations at best, but not quite accurate. Moreover, noisy inputs such as inaccurate CSI further affect the performance.Another challenge is that satisfying PER constraint of the client requires to consider not only the combinations of N_{max} and θ_{max}, but also MCS value chosen for the ongoing transmission and the channel quality. The ensuing large number of possible permutations is beyond the computational capability of AP to meet stringent time constraints. Therefore, we propose to solve the problem by leveraging machine learning. Our objective of learning is to enable CSIscan to find optimal bounds for N_{max} and θ_{max} for a given value of MCS.

IV. CNN based Preamble Modification

In this section, we describe the design of CSIscan, the CNN classifier, its training, and architecture.

A. Overall System Design of CSIscan

CSIscan leverages the previous transmission from an already connected client to infer at the AP *whether or not to embed* discovery information in the next outgoing transmission (See Fig. 7a).With 802.11ax/ac, the AP regularly acquires the latest CSI of the client through sounding frames for MIMO/beamforming operation. The AP also estimates SNR using RSSI measurements and the value of the noise floor. The combined knowledge of CSI and SNR is sufficient to accurately determine the quality of a wireless channel between the AP and client. CSIscan feeds CSI and SNR information obtained from the previous transmission in conjunction with a data rate (specified through MCS index value) opted for

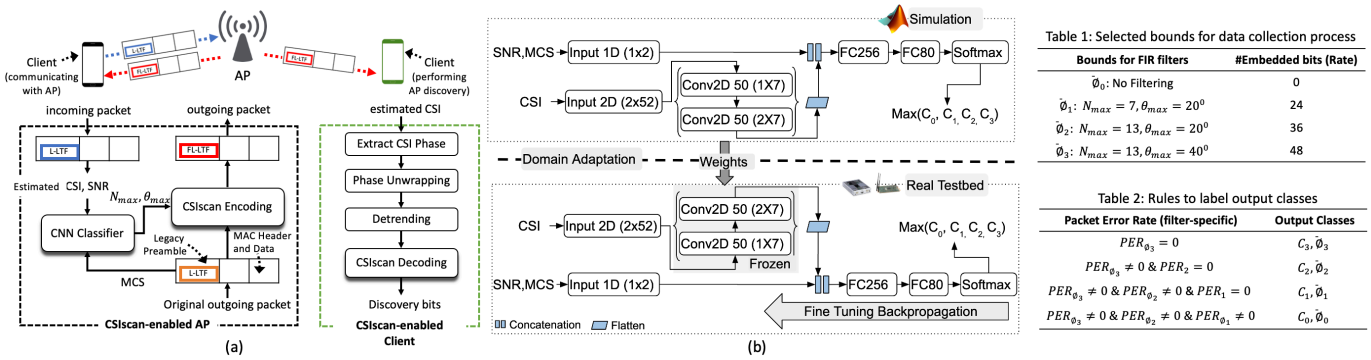


Table 1: Selected bounds for data collection process

Bounds for FIR filters	#Embedded bits (Rate)
$\breve{\emptyset}_0$: No Filtering	0
$\breve{\emptyset}_1$: $N_{max} = 7, \theta_{max} = 20^0$	24
$\breve{\emptyset}_2$: $N_{max} = 13, \theta_{max} = 20^0$	36
$\breve{\emptyset}_3$: $N_{max} = 13, \theta_{max} = 40^0$	48

Table 2: Rules to label output classes

Packet Error Rate (filter-specific)	Output Classes
$PER_{\emptyset_3} = 0$	$C_3, \breve{\emptyset}_3$
$PER_{\emptyset_3} \neq 0$ & $PER_2 = 0$	$C_2, \breve{\emptyset}_2$
$PER_{\emptyset_3} \neq 0$ & $PER_{\emptyset_2} \neq 0$ & $PER_1 = 0$	$C_1, \breve{\emptyset}_1$
$PER_{\emptyset_3} \neq 0$ & $PER_{\emptyset_2} \neq 0$ & $PER_{\emptyset_1} \neq 0$	$C_0, \breve{\emptyset}_0$

Figure 7: a) Operation of CSIscan b) Proposed CNN architecture

the next transmission to a trained CNN classifier. The CNN classifier infers the optimal bound on N_{max} and θ_{max} that are leveraged by the CSIscan encoding module to embed discovery information into the coefficients of the FIR filter. The clients that wish to discover an AP exploit these modifications to extract discovery information embedded within the estimated CSI. After unwrapping and detrending operation, the client recovers discovery bits using CSIscan decoding.

B. CNN Classifier

1) Data collection for training CNN classifier: Generating a massive amount of labeled data to improve CNN accuracy [24] from a large number of WiFi devices in real networks not only raises concerns for privacy violation, but is also time-consuming. Instead,we leverage a simulation environment that closely matches a real WiFi deployment. Our training dataset consists of simulated 802.11ac traffic using a MATLAB WLAN toolbox. CSIscan uses domain adaptation to seamlessly transition from a pure simulation environment to actual deployment.

We define three filter configurations each with a distinct bound of N_{max} and θ_{max} (see Table 1 in Fig. 7). A simulated AP transmits four packets (three filtered packets and a non-filtered packet) sequentially over the same instance of the wireless channel, defined by `wlanTGacChannel`. The simulated client receives these packets at average SNR randomly selected between 0 to 35 dB. After packet detection, the client estimates CSI and then recovers information bits. It compares the decoded bits in each packet with the transmitted bits to determine the packet error. We repeat this experiment for 1 million distinct channel realizations. For each k^{th} realization, we record the $CSI_{\phi_0}^{(k)}$ computed from the non-filtered packet. This CSI simulates the explicit CSI of the client obtained by AP through the exchange of channel sounding packets. We measure PER – $PER_{\phi_1}^{(k)}$, $PER_{\phi_2}^{(k)}$, and $PER_{\phi_3}^{(k)}$ from three filtered packets. Using these PER values and the rules defined in Table 2 of Fig. 7, we generate four output class labels $C_0^{(k)}, C_1^{(k)}, C_2^{(k)}, C_3^{(k)}$. We store $CSI_{\phi_0}^{(k)}$, $MCS^{(k)}$ and $SNR^{(k)}$ as input labels along with output class labels $C_0^{(k)}, C_1^{(k)}, C_2^{(k)}, C_3^{(k)}$ into dataset.

2) Classifier Architecture: We use CNN architecture depicted in Fig. 7b that consists two 2D convolution layers and two fully connected layers. Inputs to CNN are – (a) an estimated CSI represented as a two dimensional real-valued tensor of size 2×52 and (b) a tuple of SNR and MCS values. Input (a) is fed to the first convolutional layer (`Conv2D`) that has 50 filters each of size 1×7. Each filter learns a 7-sample variation in time over the I or Q dimension separately, to generate 50 distinct feature maps. Similarly, the second `Conv2D` layer has 50 filters each of size 2×7, where each filter learns variations, again of 7 activation values, over both I and Q dimensions of the 50-dimensional activation volume obtained after the first `Conv2D` layer. Each `Conv2D` layer is followed by a Rectified Linear Unit (`ReLU`) activation. The output of second `Conv2D` layer is flattened and concatenated with the input (b). The concatenated layer is then fed to the first fully connected (`FC256`) layer with 256 neurons. A second fully connected layer (`FC80`) is added to extract non-linear combinations high level features extracted from previous layers, that are finally passed to a `Softmax` classifier layer. We use cross validation to find hyperparameters of CNN, including the number of `Conv2D` filters, size, and depth of the model to ensure its generalization. We train the model with Adam optimizer with a learning rate of $lr = 0.0001$.

• **Domain adaptation for real network deployment:** Even though advantageous, the training of our CNN model with millions of simulated channel realizations is insufficient to learn and generalize for the infinite possibilities of channel variations in real environments. Fortunately, initial convolution layers of CNN are able to extract and retain necessary features to classify different bounds. We exploit these learned features to perform well in a completely new environment using a technique called *domain adaptation* – wherein a classifier trained in a source distribution is used to solve the same task in different (but related) target distribution [25], [26]. Although wireless channels in simulated and real environments follow a different distribution, our task of selecting optimum bounds remains the same, enabling us to transfer the knowledge from simulated to the real environment. We use a popular supervised *fine-tuning* approach for domain adaptation where we freeze first 'l' layers of the model trained with source data and retrain the same model with target data to fine-tune its last few layers using backpropagation. In our case, after training the classifier with a large amount of simulated data, we freeze two convolution layers of the classifier and then fine-tune other layers by retraining with real data.

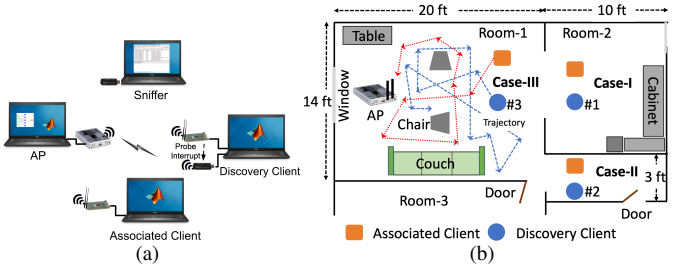


Figure 8: a) Experiment setup; b) Indoor environment with a stationary WiFi AP, associated, and discovery clients, either stationary or mobile. The devices are placed at different locations in the floor map.

V. Performance Evaluation

We first demonstrate the performance enhancement achieved with CSIscan compared to active discovery. Next, we establish the practical usage of CSIscan in different stationary and mobility scenarios.

A. Experimental Setup

• **Device Configuration:** The experiment setup consists of a stationary WiFi AP, sniffer, two clients – associated and discovery client that can be stationary or mobile (see Fig. 8a). We consider three scenarios – i) stationary, LOS (Case-I): the associated client and discovery client #1 are both stationary and are placed at a distance of ∼16 ft in LOS of the AP; b) stationary, NLOS (Case-II): the associated client and discovery client #2 are both stationary and are placed in NLOS of AP at a distance of ∼18 ft; and c) mobile (Case-III): the associated and discovery client #3 are both mobile and are moving at a brisk walking pace of ∼4 miles per hour. All three cases are evaluated in the live WiFi network of the university with ∼30 COTS clients present in lab rooms with 3 APs per lab, typically representing the indoor environment. While Case-I is evaluated to establish the advantages of CSIscan in stationary LOS, Case-II and Case-III establish that CSIscan is capable of serving NLOS and mobile clients as well.

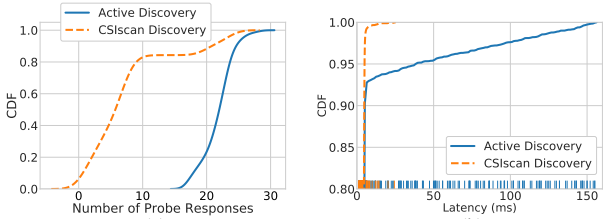
We set up the AP using USRP X310 software-defined radio (SDR) with the output power set at 20 dBm, similar to COTS WiFi APs. The AP transmits 802.11ac compliant WiFi MAC layer packets. We configure associated and discovery clients using USRP B210 SDRs, where the discovery client follows either active or CSIscan discovery protocol. For the CSIscan client, the challenge is to trigger the discovery process, once it successfully decodes discovery bits embedded within CSI of the received packet. We rely on the default wireless interface card of the laptop to execute the scanning procedure. The active discovery client simply relies on the onboard wireless card to trigger a scan. Although we use a USRP-based platform to evaluate CSIscan in this paper, its practical integration in COTS WiFi devices is feasible with firmware level modifications using tools like Nexmon [27]. It enables firmware modification for Broadcom/Cypress WiFi chips through a custom-built C-based firmware patching framework. In particular, the firmware should include the logic for CSI decoding and modify the existing logic for best channel selection in kernel drivers and network managers. Further, we envision next-generation COTS APs to be AI-powered with an integrated machine learning engine to perform various learning tasks. Major AP vendors have already introduced AI-empowered devices in the market [28]–[30]. When the packet reaches the client host through the USRP, our program interrupts the discovery code in the userspace that triggers a scan with the Linux `iw` utility [31].

• **Discovery Initiation:** The exact time instant when to trigger discovery procedure is decided by sophisticated algorithms at the client that reside in applications such as Network Managers and Kernel drivers. Since, *when* to initiate the process is out-of-the-scope for this work, we take a conservative approach here by disabling all such network management applications. We instruct the client to trigger the discovery process, active or CSIscan, whenever a packet from the AP is received by the client. The AP is programmed to embed information in any outgoing packet every 100 ms. Packets from AP are transmitted generally in the order of few μ seconds. Since this embedding interval of 100 ms is several order of magnitude higher than μ seconds, we choose this value to demonstrate the worst-case performance of CSIscan.

B. Experiments

• **Probe Responses and Latency:** Once the client receives a packet via the USRP, it sends probe request(s). With active discovery, the client sends a probe request on all supported 39 WiFi channels. With CSIscan, the client sends a targeted probe request on AP's channel x after successful decoding. If decoding fails, it follows active discovery. Here we make an assumption that the client has obtained discovery information across all channels and decided to discover AP on channel x ; since the logic of choosing the best channel is already present in kernel drivers and network managers. We record the probe responses for each probe request generated by the client. The client measures the latency using the `ping` utility.

• **CNN Accuracy:** We evaluate the classification accuracy of the proposed CNN model and its accuracy after domain adaptation. This is needed because the model is trained in a simulated environment, while it is actually used in the live environment. We verify the performance of the CNN model with data collected in MATLAB simulations. Our dataset consists of $\sim$ $80K$ training and $10K$ validation examples. We use other $10K$ examples to test the performance of the trained model. For realistic channel conditions, we collect over-the-air data with USRP-based AP and clients. Similar to data collection process described in Sec. IV-B for simulations, AP sequentially transmits four packets (three filtered and a non-filtered packet). These packets are transmitted within coherence time, and therefore, the channel is assumed to be constant. After receiving the packet, the client first extracts CSI and then decodes the packet. We record the CSI estimated from the non-filtered packet, along with MCS and SNR as input labels, while PERs of filtered packets are used to create output labels following rules defined in Table 2 of Fig. 7.



(a) (b)

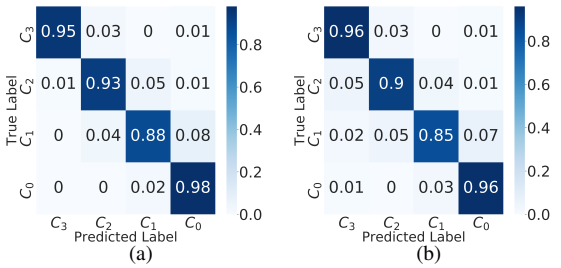
Figure 9: a) CSIscan reduces probe responses by 72% and b) tail latency from 150 ms to mere 10 ms.

• **Bit stuffing efficiency and PER:** We define bit stuffing efficiency as the number of discovery bits successfully transmitted per packet. We evaluate it for various MCS rates with the selection of – i) a fixed bound or ii) adaptive bound, as chosen by the trained CNN. The client extracts discovery bits by decoding estimated CSI from received packets. We measure the #successfully decoded discovery bits embedded per packet, averaging over 10000 collected packets. These packets are originally filtered at the AP with three bounds, i) $N_{max} = 7, \theta_{max} = 20$ ii) $N_{max} = 13, \theta_{max} = 20$, and iii) $N_{max} = 13, \theta_{max} = 40$. With CSIscan, the AP dynamically selects the bounds based on the channel condition. Next, we analyze the impact on the client's ability to compensate phase noise with and without FIR filtering of L-LTF. Typically, the distortion caused by phase noise is characterized by a common phase error (CPE) that represents the common rotation of all constellation points of equalized symbols in the complex plane. We track CPE over several OFDM symbols using the algorithm proposed in [21].Further, we measure PER at the associated client to quantify the impact on its communication due these bounds.

C. Results

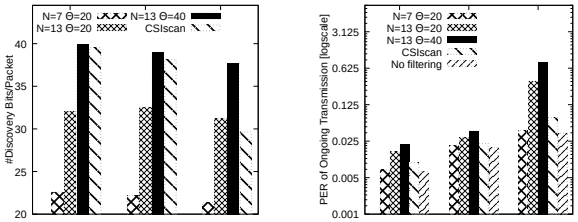
• **Probe Responses and Latency:** Fig. 9a plots the CDF of the number of probe responses received by the client for every probe request sent either with active discovery or CSIscan. We observe 22 (median) probe responses with active discovery, whereas only 6 are present for CSIscan. Thus, CSIscan reduces probe traffic by ∼72%. The reduction in probe responses reduces discovery traffic, thereby improving spectrum utilization. Fig. 9b compares the WiFi latency experienced by the client with active discovery and CSIscan. While median WiFi latency for active discovery is as low as ∼4 ms, its 99^{th} percentile is ∼132 ms. This long-tail distribution is not visible with CSIscan. The 99^{th} percentile is ∼6 ms. Thus, CSIscan alleviates the problem of long-tail latency caused by frequent discovery process.

• **CNN Accuracy:** For the simulation environment, the CNN model has an accuracy of 95.20% for correctly selecting one out of four bounds (see confusion matrix in Fig. 10a). Post domain adaptation the CNN model, fine-tuned with real data, has classification accuracy of 93.87% (see confusion matrix in Fig. 10b). For fine-tuning, we use nearly $12K$ examples of real data, as compared to $80K$ of simulation data, thus requiring only 15% of the total training data for training of the CNN model. This reduction in training data also shortens the



(a) (b)

Figure 10: Confusion matrices for a) Simulation and b) Live environment. In simulation, CNN achieves accuracy of 95.20%. Post domain adaptation, it achieves accuracy of 93.87% in live environment.

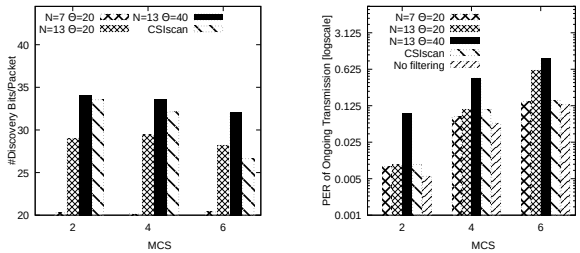


(a) (b)

Figure 11: Case-I – Stationary discovery client and associated client placed in LOS of AP. a) Bit stuffing efficiency of the discovery client (#1), and b) PER of the associated client. CSIscan chooses optimal combination of N, θ, while inducing ∼0 PER at associated client.

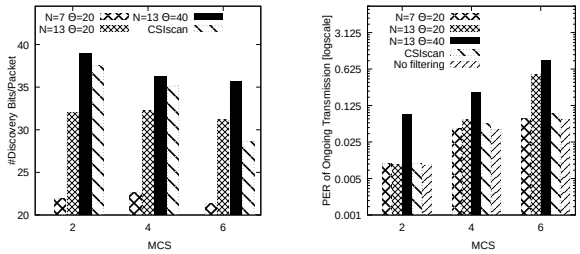
training time by 64 ms on average. For the sake of simplicity, we train the CNN model offline in a centralized manner to find the dynamic bound on data collected at clients. However, this approach raises privacy concerns in the sharing of data used to train the CNN model. Federated learning [32] can be a promising solution here, wherein the model is trained at the clients without sharing the raw data and then disseminated.

• **Bit stuffing efficiency and Packet Error Rate:** Fig.11-13 plot bit stuffing efficiency for the discovery client and corresponding PER of the associated client for various MCS values with fixed bound or adaptive bound in three deployment scenarios. The overly pessimistic approach of always choosing lower bound $N_{max} = 7$ and $\theta_{max} = 20$ reduces the bit stuffing efficiency to approximately average 22 discovery bits for the stationary client #1 (Fig. 11a), 20 discovery bits for the stationary client #2 (Fig. 12a), and 21 discovery bits for the mobile client #3 per each packet (Fig. 13a). On the other hand, the selection of higher bound $N_{max} = 13$ and $\theta_{max} = 40$ severely impacts PER of the client at higher MCS rate. At MCS= 6, PER is $> 75\%$ for client #1 (Fig. 11b) as well as client #3 (Fig. 13b), whereas PER is $> 95\%$ for client #2 (Fig. 12b). At MCS= 6, even the moderate bound of $N_{max} = 13$ and $\theta_{max} = 20$ results in $> 50\%$ PER for clients #2 and #3 (see Fig. 12b and Fig. 13b respectively). CSIscan dynamically adjusts its bound to embed maximum discovery information while guaranteeing the increase in PER at the associated client is $< 1\%$, as compared to PER with no filtering. Especially at a higher MCS, CSIscan avoids embedding information if the channel is not good enough; whereas if the channel condition permits, it transmits maximum information.



(a) (b)

Figure 12: Case-II – Stationary discovery client and associated client, placed in NLOS of AP. a) Bit stuffing efficiency of the discovery client (#2), and b) PER of the associated client.



(a) (b)

Figure 13: Case-III – Mobile discovery client and associated client. a) Bit stuffing efficiency of the discovery client (#3), and b) PER of the associated client. CSIscan delivers discovery information to mobile client without significant impact on PER of the associated client.

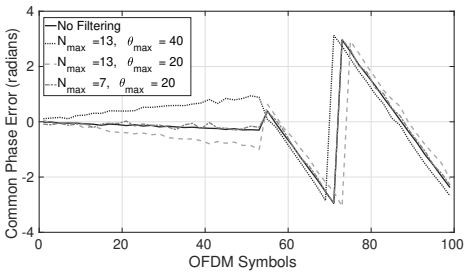


Figure 14: Tracking the compensation of phase noise quantified via common phase error (CPE). The associated client tracks CPE despite reception of L-LTF filtered packets from AP with any chosen bounds.

Despite L-LTF filtering, the associated clients compensate phase noise successfully for any chosen bounds. We confirm this by evaluating CPE over several OFDM symbols with and without filtering for the associated client in Case-II (Fig. 14). We observe a similar trend for the client in Case-I and Case-III. While CPE for lower bound of N_{max} = 7 and θ_{max} = 20 closely matches with that of without filtering, the client tracks CPE after few OFDM symbols for higher bounds showing offsets in initial symbols. Thus, CSIscan does not adversely affect the client's phase tracking functionality.

VI. Related work

Since the last decade, there have been significant efforts in quantifying the cost of excessive discovery traffic [33]–[36] and proposing protocols for a faster network discovery [37]–[40]. The IEEE 802.11ai is an upcoming fast initial link setup (FILS) amendment that improves the way APs are discovered by clients [41], [42]. FILS clients follow the Probe Deferral and Suppression mechanism that mandates the client to listen for broadcast beacons or probe responses for up to 20 ms before sending probe requests. This is an additional wait time over and above discovery latency, and is critical for URLLC and TSN services. Prior to 802.11ai, the IEEE 802.11k standard attempted to reduce AP discovery time for roaming clients [43], [44], where the client requests for Neighbor Report containing a list of neighbor APs with their channel numbers. Unfortunately, with this approach, the client (a) needs to be associated with the AP, (b) still has to scan all the channels reported in the Neighbor Report, and (c) needs periodic updates for a fresh Neighbor Report. This leads to an optimization problem– a longer period may result in a lost connection, while a shorter period increases network overhead.

Deep learning [45], particularly through the use of CNNs, has shown remarkable performance in image and speech applications [46], [47], and is steadily gaining traction in applications within wireless the domain [48]–[52]. More specific use cases of CNN in the PHY layer of wireless communications has been seen in Modulation Classification [53]–[55], Protocol Identification [56] and RF Fingerprinting [57]–[61]. The versatility of the CNN motivates us to adapt it in our work for the discovery problem within dense networks.

CSI has been widely used for various sensing applications [62] such as localization [63], [64]. Further, authors in [65] show how malicious users can infer CSI of clients by overhearing sounding frames. More related to our work is [66] that modifies symbols in short training field to embed information. However, this approach is very sensitive to changing frequency offsets in practical radios. Further, it embeds fixed number of bits irrespective of channel conditions. In contrast, CSIscan demonstrates a practical application of AP discovery by embedding discovery information in L-LTF field of the preamble. It dynamically adjusts the number of discovery bits by learning the channel and quickly becomes operational in a real network deployment with domain adaptation.

VII. Conclusion

We presented a novel technique, CSIscan, to embed relevant information in CSI of ongoing WiFi transmissions using CNNs. With the help of a prototype developed with USRP-based SDR, we demonstrate the feasibility of CSIscan-based WiFi discovery in a live network. CSIscan reduces probe responses by 72% and latency from 150 ms to 10 ms. The selection of optimal bounds for subcarriers and phases results in embedding upto 40 bits in the preamble while keeping the PER below 1%. Our robust CNN architecture with domain adaptation achieves an accuracy of 93.87% for unseen wireless environments. While we have demonstrated the feasibility of embedding information in CSI, our next step is to implement the solution in a mobile device like a COTS smartphone. Although we have considered AP discovery as an application of the proposed technique of embedding information, the technique can be extended to many other applications in the wireless domain. The dataset pertaining to this study is available at the authors' website [67].

Acknowledgment

This work is supported by the Office of Naval Research under grant N000141612651 and NSF award CNS 1923789.

REFERENCES

[1] Cisco. Mobile visual networking index (vni) forecast. Whitepaper, February 1999.

[2] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi. A tutorial on IEEE 802.11 ax high efficiency WLANs. *IEEE Communications Surveys & Tutorials*, 2018.

[3] D. Jaisinghani, V. Naik, S. K. Kaul, and S. Roy. Realtime detection of degradation in WiFi network's goodput due to probe traffic. In *Proceedings of WiOpt*, 2015.

[4] X. Hu and L. Song and D. V. Bruggen and A. Striegel. Is There WiFi Yet?: How Aggressive Probe Requests Deteriorate Energy and Throughput. In *Proceedings of ACM IMC*, 2015.

[5] Improving WiFi Connections in High-Density Environments, Oct 2017. Accessed on 4 Aug. 2020.

[6] H. Balbi, D. Passos, R. Carrano, L. Magalhães, and C. Albuquerque. Association stability and handoff latency tradeoff in dense IEEE 802.11 networks: A case study. *Computer Communications*, 2020.

[7] K. Sui, Y. Zhao, D. Pei, and L. Zimu. How bad are the rogues' impact on enterprise 802.11 network performance? In *Proceedings of IEEE INFOCOM*, 2015.

[8] Viki Green. Impact of slow page load time on website performance, Jan 2016. https://medium.com/@vikigreen/impact-of-slow-page-load-time-on-website-performance-40d5c9ce568a.

[9] A. Croitoru, D. Niculescu, and C. Raiciu. Towards Wifi Mobility without Fast Handover. In *Proceedings of USENIX NSDI*, 2015.

[10] S. Sur, I. Pefkianakis, X. Zhang, and K. Kim. WiFi-Assisted 60 GHz Wireless Networks. In *Proceedings of ACM MobiCom*, 2017.

[11] K. Sui, M. Zhou, D. Liu, M. Ma, D. Pei, Y. Zhao, Z. Li, and T. Moscibroda. Characterizing and improving wifi latency in large-scale operational networks. In *Proceedings of MobiSys*, 2016.

[12] P. Rai, A. Saha, H. Daumé III, and S. Venkatasubramanian. Domain adaptation meets active learning. In *Proceedings of NAACL HLT*, 2010.

[13] A. Mishra, M. Shin, and W. Arbaugh. An empirical analysis of the IEEE 802.11 MAC layer handoff process. *CCR*, 2003.

[14] H. Velayos and G. Karlsson. Techniques to reduce the IEEE 802.11 b handoff time. In *Proceedings of IEEE INFOCOM*, 2004.

[15] G. Castignani, A. Arcia-Moret, and N. Montavont. An evaluation of the resource discovery process in ieee 802.11 networks. In *Proceedings of MobiOpp*, 2010.

[16] Dwell Time In Probe Request Presentation - IEEE Mentor, May 2016. IEEE Standards Association - 802.11 WLANs WG Group Information.

[17] ping(8) - Linux man page. https://linux.die.net/man/8/ping, accessed on 2 Apr. 2019.

[18] Storage: How 'tail latency' impacts customer-facing applications, Aug 2019. https://www.computerweekly.com/opinion/Storage-How-tail-latency-impacts-customer-facing-applications, accessed on 25 Feb 2020.

[19] J. Freudiger. How talkative is your mobile device? An experimental study of Wi-Fi probe requests. In *Proceedings of ACM WiSec*, 2015.

[20] Yinsheng Liu, Zhenhui Tan, Hongjie Hu, Leonard J Cimini, and Geoffrey Ye Li. Channel estimation for OFDM. *IEEE Communications Surveys & Tutorials*, 2014.

[21] A. Van Zelst and T. Schenk. Implementation of a MIMO OFDM-based wireless LAN system. *IEEE Transactions on signal processing*, 2004.

[22] V. Tsibonis, L. Georgiadis, and L. Tassiulas. Exploiting wireless channel state information for throughput maximization. In *Proceedings of IEEE INFOCOM*, 2003.

[23] N. Karakoç and T. Duman. Rate Selection for Wireless Random Access Networks Over Block Fading Channels. *IEEE Transactions on Communications*, 2019.

[24] Do we need more training data?, author=Zhu, X. and Vondrick, C. and Fowlkes, C. and Ramanan, D. *International Journal of Computer Vision*, 2016.

[25] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. A survey on deep transfer learning. In *Proceedings of Springer ICANN*, 2018.

[26] M. Wang and W. Deng. Deep visual domain adaptation: A survey. *Elsevier Neurocomputing*, 2018.

[27] GitHub. seemoo-lab/nexmon, July 2020. https://github.com/seemoo-lab/nexmon, accessed on 15 Jan. 2020.

[28] Cisco. Artificial Intelligence Machine Learning, May 2020. https://www.cisco.com/c/en/us/solutions/artificial-intelligence.html, accessed on 20 July. 2020.

[29] Aruba. 802.11ax AI-powered Wi-Fi 6 Solutions. https://www.arubanetworks.com/ai-powered-wi-fi/, accessed on 20 July. 2020.

[30] Mist Systems. Mist Platform Solution Brief, July 2020. https://linux.die.net/man/8/iw, accessed on 20 July. 2020.

[31] iw(8) - Linux man page. https://linux.die.net/man/8/iw, accessed on 2 Apr. 2019.

[32] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 2019.

[33] M. Barbera, A. Epasto, A. Mei, V. Perta, and J. Stefa. Signals from the crowd: uncovering social relationships through smartphone probes. In *Proceedings of IMC*, 2013.

[34] H. Hong, C. Luo, and M. Chan. Socialprobe: Understanding social interaction through passive wifi monitoring. In *Proceedings of MOBIQUITOUS*, 2016.

[35] R. Raghavendra, E. Belding, K. Papagiannaki, and K. Almeroth. Unwanted link layer traffic in large IEEE 802.11 wireless networks. *IEEE Transactions on Mobile Computing*, 2010.

[36] B. Hart, M. Swartz, J. Suhr, and M. Silverman. Stadium Measurements. *802.11ax Working Group (Doc 1223)*, 2014.

[37] H. Wu, K. Tan, J. Liu, and Y. Zhang. Footprint: cellular assisted Wi-Fi AP discovery on mobile phones for energy saving. In *Proceedings of WINTECH*, 2009.

[38] G. Ananthanarayanan and I. Stoica. Blue-Fi: enhancing Wi-Fi performance using bluetooth signals. In *Proceedings of MobiSys*, 2009.

[39] J. Teng, C. Xu, W. Jia, and D. Xuan. D-scan: Enabling fast and smooth handoffs in ap-dense 802.11 wireless networks. In *Proceedings of IEEE INFOCOM*, 2009.

[40] J. Choi, G. Lee, Y. Shin, J. Koo, M. Jang, and S. Choi. Blend: BLE beacon-aided fast wifi handoff for smartphones. In *Proceedings of IEEE SECON*, 2018.

[41] IEEE 802.11 Working Group et al. IEEE 802.11 ai-2016.

[42] E. Au. The Latest Progress on IEEE 802.11mc and IEEE 802.11ai [Standards]. *IEEE Vehicular Technology Magazine*, 2016.

[43] IEEE Std 802.11k 2008. IEEE Standard for Information Technology – Local and Metropolitan Area Networks– Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs. Amendment to IEEE Std 802.11, 2008.

[44] M. Sanchez and A. Boukerche. On IEEE 802.11 K/R/V amendments: Do they have a real impact? *IEEE Wireless Communications*, 2016.

[45] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[46] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[47] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. Alsaadi. A survey of deep neural network architectures and their applications. *Elsevier Neurocomputing*, 2017.

[48] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari, and F. Adachi. Deep learning for physical-layer 5G wireless techniques: Opportunities, challenges and solutions. *IEEE Wireless Communications*, 2019.

[49] R. Balakrishnan, K. Sankhe, S. Somayazulu, R. Vannithamby, and J. Sydir. Deep Reinforcement Learning Based Traffic-and Channel-Aware OFDMA Resource Allocation. In *Proceedings of IEEE GLOBECOM*, 2019.

[50] T. Wang, C. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin. Deep learning for wireless physical layer: Opportunities and challenges. *China Communications*, 2017.

[51] K. Sankhe, F. Restuccia, S. D'Oro, T. Jian, Z. Wang, A. Al-Shawabka, J. Dy, T. Melodia, S. Ioannidis, and K. Chowdhury. Impairment Shift Keying: Covert Signaling by Deep Learning of Controlled Radio Imperfections. In *Proceedings of IEEE MILCOM*, 2019.

[52] H. Ye, G. Li, and B. Juang. Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wireless Communications Letters*, 2017.

[53] Timothy James O'Shea, Tamoghna Roy, and T Charles Clancy. Over-the-air deep learning based radio signal classification. *IEEE Journal of Selected Topics in Signal Processing*, 2018.

[54] N. Soltani, K. Sankhe, S. Ioannidis, D. Jaisinghani, and K. Chowdhury. Spectrum Awareness at the Edge: Modulation Classification using Smartphones. In *Proceedings of IEEE DySPAN*, 2019.

[55] T. O'Shea and J. Hoydis. An introduction to deep learning for the physical layer. *IEEE Transactions on Cognitive Communications and Networking*, 2017.

[56] A. Selim, F. Paisana, J. A Arokkiam, Y. Zhang, L. Doyle, and L. DaSilva. Spectrum monitoring for radar bands using deep convolutional neural networks. In *Proceedings of IEEE GLOBECOM*, 2017.

[57] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury. Deep learning convolutional neural networks for radio identification. *IEEE Communications Magazine*, 2018.

[58] K. Merchant, S. Revay, G. Stantchev, and B. Nousain. Deep learning for RF device fingerprinting in cognitive communication networks. *IEEE Journal of Selected Topics in Signal Processing*, 2018.

[59] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury. ORACLE: Optimized Radio clAssification through Convolutional neuraL nEtworks. In *Proceedings of IEEE INFOCOM*, 2019.

[60] K. Sankhe, M. Belgiovine, F. Zhou, L. Angioloni, F. Restuccia, S. D'Oro, T. Melodia, S. Ioannidis, and K. Chowdhury. No Radio Left Behind: Radio Fingerprinting Through Deep Learning of Physical-Layer Hardware Impairments. *IEEE Transactions on Cognitive Communications and Networking*, 2019.

[61] T. Jian, B. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis. Deep Learning for RF Fingerprinting: A Massive Experimental Study. *IEEE Internet of Things Magazine*, 2020.

[62] Y. Ma, G. Zhou, and S. Wang. WiFi sensing with channel state information: A survey. *ACM Computing Surveys*, 2019.

[63] K. Joshi, D. Bharadia, M. Kotaru, and S. Katti. WiDeo: Fine-grained Device-free Motion Tracing using RF Backscatter. In *Proceedings of USENIX NSDI*, 2015.

[64] X. Wang, L. Gao, S. Mao, and S. Pandey. CSI-based fingerprinting for indoor localization: A deep learning approach. *IEEE Transactions on Vehicular Technology*, 2016.

[65] X. Zhang and E. Knightly. CSIsnoop: Attacker inference of channel state information in multi-user WLANs. In *Proceedings of ACM MobiHoc*, 2017.

[66] H. Rahbari and M. Krunz. Exploiting frame preamble waveforms to support new physical-layer functions in OFDM-based 802.11 systems. *IEEE Transactions on Wireless Communications*, 2017.

[67] Dataset for CSIscan. Website: http://genesys-lab.org/csiscan, accessed on 21 Aug. 2020.