POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Compressing Wi-Fi Channel State Information for Optimized Human Sensing

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Author: **Paolo Cerutti**

Student ID: 233322
Advisor: Prof. Alessandro E. C. Redondi
Co-advisors: Marco Cominelli, Fabio Palmese
Academic Year: 2024-25

Abstract

The rapid proliferation of IoT devices has created a massive flood of data that these systems are not designed to manage. With limited storage and processing capabilities, many IoT devices struggle to keep up with the high-dimensional data they generate, especially in applications that require long-term data preservation, such as digital forensics and real-time human activity recognition. This thesis addresses these critical challenges by investigating lossy compression techniques for *Channel State Information* (CSI) data to balance the need for reduced storage requirements with the preservation of essential evidentiary and analytical details.

The research investigates several methods for dimensionality reduction and quantization. Conventional methods include *Principal Component Analysis* (PCA), *Scalar Quantization* (SQ), and *Vector Quantization* (VQ), which are evaluated both individually and in combination to determine the optimal balance between storage efficiency and classification performance. In addition, advanced deep learning techniques using *Variational Autoencoders* (VAE) are explored to automatically extract latent representations for later quantization. Huffman coding is applied as a lossless compression step to further increase storage reduction. Experiments have been performed in two real-world scenarios. In the *Presence Detection* scenario—a binary classification task—the system captured CSI data to detect the presence of a person in the environment. In the *Activity Recognition* scenario—a multi-class problem—CSI data was collected to detect a range of static and dynamic human actions. In both cases, in-depth experiments evaluated the impact of different compression configurations on classification accuracy and storage requirements. The results highlight the success in preserving data fidelity for legal evidence and optimizing storage in resource-constrained IoT environments. This work demonstrates that lossy compression methods can effectively support long-term forensic data preservation and human activity recognition.

Keywords: Human Sensing, Channel State Information, Lossy Compressions, IoT Forensics

Abstract in lingua italiana

La rapida diffusione di dispositivi IoT ha creato un'enorme quantità di dati che questi sistemi non sono stati concepiti per gestire. Tali dispositivi, infatti, presentano limitate capacità di calcolo e di archiviazione, rendendoli inadatti a mantenere il passo con l'elevato grado di dimensionalità dei dati generati; specialmente in contesti come la scienza digitale forense dove è essenziale la conservazione a lungo termine dei dati. La tesi affronta queste problematiche sperimentando l'utilizzo di tecniche di compressione con perdita per i dati CSI *(Channel State Information)*, al fine di trovare un equilibrio tra la riduzione del costo di archiviazione dei dati e la conservazione di dettagli essenziali per l'analisi forense.

La ricerca si concentra sull'impiego di metodi di riduzione della dimensionalità e quantizzazione, valutandone l'efficacia nel trovare un equilibrio tra efficienza di memorizzazione e prestazioni di classificazione. In particolare si esaminano l'*Analisi delle Componenti Principali* (PCA), la *Quantizzazione Scalare* (SQ) e la *Quantizzazione Vettoriale* (VQ), sia singolarmente che in combinazione tra di loro. Inoltre, si esplorano tecniche di deep learning basate su *Autoencoder Variazionali* (VAE) per l'estrazione automatica di rappresentazioni latenti, successivamente soggette a quantizzazione. Infine, alla fine del processo, viene applicata la Codifica di Huffman, un metodo di compressione senza perdita, per ridurre ulteriormente lo spazio di archiviazione. Gli esperimenti sono stati condotti in due scenari reali. Nel primo scenario (*Presence Detection*), il sistema ha acquisito dati CSI per rilevare la presenza di una persona all'interno della stanza. Nel secondo scenario (*Activity Recognition*), sono stati raccolti dati CSI per la rilevazione di azioni umane sia statiche che dinamiche. In entrambi i casi, gli esperimenti hanno valutato l'impatto delle diverse configurazioni di compressione sull'accuratezza della classificazione e sui requisiti d'archiviazione. I risultati ottenuti evidenziano il successo nel preservare l'accuratezza dei dati e nell'ottimizzare l'archiviazione dei dati in sistemi IoT. La tesi dimostra che i metodi di compressione con perdita possono svolgere un ruolo significativo nella conservazione di dati destinati all'utilizzo forense e che possono essere impiegati in metodi di rilevamento delle attività umane.

Parole chiave: Rilevamento umano, Channel State Information, compressione con perdita di dati, analisi forense dell'IoT

Contents

Abstract **i**

Abstract in lingua italiana **iii**

Contents **v**

1 Introduction **1**

2 Background **5**

2.1 Internet of Things Forensics 5
2.2 Existing Approaches to Activity Recognition 7
2.3 Channel State Information 8
2.3.1 Wireless Signal Propagation 8
2.3.2 Mathematical Representation 10
2.3.3 Estimation Techniques 11
2.3.4 Data Collection Techniques 12
2.3.5 Multiple-Input Multiple-Output Systems 12
2.3.6 CSI for Classification 14

3 Related Works **17**

3.1 Human Behavior Recognition 17
3.2 CSI Compression 19

4 Tools and Methods **21**

4.1 Compression Methodology 21
4.1.1 Principal Component Analysis 22
4.1.2 Scalar Quantization 25
4.1.3 Vector Quantization 26
4.1.4 Huffman Coding 27
4.1.5 Variational Autoencoder 28

4.2 Scenarios and Experimental Setup 30
4.2.1 Presence Detection Scenario 30
4.2.2 Experimental Methods 31
4.2.3 Activity Recognition Scenario 37
4.2.4 Experimental Methods 38
4.3 Implementation Details 39

5 Results **43**
5.1 Performance Analysis 43
5.1.1 Presence Detection 43
5.1.2 Activity Recognition 46
5.2 Storage Analysis 47
5.3 Encoding Analysis 49

6 Conclusions and Future Developments **53**

Bibliography **55**

List of Figures **61**

List of Tables **63**

1 | Introduction

The growth of the Internet of Things (IoT) has transformed modern life, turning everyday environments into connected ecosystems capable of exchanging data in real time. This expansion not only changed industries and consumer experiences, but also opened new opportunities in research areas such as digital forensics and human activity recognition. As billions of IoT devices are deployed in smart homes, cities, and industrial environments, the volume of data generated has created a need for efficient data processing and storage methods. [1]

Digital forensics is the process of discovering, collecting, analyzing, and preserving electronic evidence for legal proceedings. With the rise of IoT, traditional digital forensics practices are evolving into a more complex discipline known as IoT forensics. This new domain extends investigations to a wide range of devices, from smart home appliances to medical implants and remote sensors, and requires specialized techniques for handling vast, heterogeneous data sources. In particular, as IoT devices are integrated into private spaces, they not only record and transmit data that reveals sensitive aspects of daily life, but also introduce unique security challenges. Vulnerabilities in these devices can potentially allow unauthorized access to homes or control surveillance systems. At the same time, because these devices operate continuously and are deeply embedded in our routines, they provide a rich source of evidence that can be invaluable in reconstructing events and human behavior in forensic investigations when a crime occurs.

Traditional methods for activity recognition, including wearable sensors and camera systems, have provided valuable insights but are hampered by significant limitations. Wearable devices require continuous user compliance, while camera-based systems are often constrained by line-of-sight issues and heightened privacy concerns. In contrast, WiFi-based sensing offers a promising, non-invasive alternative that leverages the ubiquity of wireless signals indoors. In particular, Channel State Information (CSI) provides a detailed characterization of how wireless signals propagate, capturing both amplitude and phase information across multiple subcarriers. This dataset can capture subtle variations induced by human motion, from simple gestures to complex behaviors, making it

an attractive candidate for a wide range of applications, including digital forensics.

One critical aspect of IoT forensics is that the collected evidence must often be preserved indefinitely; once data is identified as potentially valuable for an investigation, it must be maintained in an unaltered state for days, months, or even longer awaiting legal trial. Additionally, unlike traditional evidence stored on static media, IoT data is often distributed across cloud services and edge devices, where storage limitations and frequent overwrites pose additional challenges.

Because forensic cases require long-term preservation of digital evidence, data collection and storage pipelines must be optimized to minimize resource usage while maintaining data integrity. In many cases, lossless compression techniques such as `gzip` are used to preserve full data fidelity. However, when dealing with high-dimensional data, such as CSI streams, these traditional methods may not provide sufficient reduction in storage requirements. As a result, while lossless approaches are fundamental to ensuring that evidence remains unaltered, their compression ratios may be insufficient for applications where data must be kept indefinitely. This gap motivates the exploration of lossy compression strategies that aim to discard non-critical redundancy while preserving the essential features necessary for forensic analysis and accurate human activity recognition.

However, CSI-based sensing comes with its own set of challenges. While it provides detailed insight into wireless channels by accounting for signal scattering, environmental attenuation, and multipath effects, CSI is inherently high-dimensional. The massive amounts of data generated can quickly overwhelm memory systems in IoT devices, which often have limited storage and computing capabilities. This limitation is significant in forensic applications, where data integrity and long-term availability are critical. Optimized compression techniques are essential not only to reduce storage footprints, but also to ensure that forensic evidence remains reliable and accessible throughout the legal process.

This thesis aims to address these needs by optimizing the storage requirements of WiFi-based sensing through lossy compression of Channel State Information (CSI) for efficient forensic analysis and passive human activity detection.

To address these challenges, this study investigates the performance and storage implications of various CSI data compression techniques for human activity recognition. Both traditional and deep learning-based approaches are explored. Techniques such as Principal Component Analysis (PCA), Scalar Quantization (SQ), and Vector Quantization (VQ) are used to reduce dimensionality and discard redundant information. These methods are supplemented by more advanced machine learning techniques such as Variational

Autoencoders (VAE), which provide automated feature extraction and recently emerged for handling complex data sets. In addition, Huffman coding is applied as a lossless final compression step to further reduce data size without sacrificing essential information. By comparing these methods in two different scenarios (binary classification for presence detection and multi-class classification for human activity recognition), the study aims to identify optimal strategies that balance the need for high accuracy with storage and transmission constraints.

The first experimental scenario involves a simplified setting where a smart camera and a standard access point are used to generate and capture CSI data in a controlled indoor environment. Here, the primary goal is to detect the presence or absence of a person, which is a binary classification problem. In contrast, the second scenario scales up to a more complex environment using routers equipped with Multiple Input Multiple Output (MIMO) technology. This scenario investigates the detection of various human activities, including both static and dynamic actions.

Preliminary results indicate that when computational resources are abundant, a combination of VQ and Huffman coding (VQ + ENC) provides maximum data reduction with minimal performance degradation. In contrast, in resource-constrained environments, the use of SQ combined with PCA and Huffman coding (SQ + PCA + ENC) provides a balanced trade-off between computational complexity and storage efficiency.

The rest of the work is structured as follows:

- **Chapter 2** provides the background knowledge necessary to understand the research. It begins with an overview of IoT forensics, followed by a discussion of existing approaches to activity recognition. A significant portion of the chapter is dedicated to CSI, including its wireless signal propagation properties, mathematical representation, estimation and data collection techniques, and its role in classification tasks.
- **Chapter 3** reviews related work in the field, focusing on human behavior recognition using wireless signals and existing CSI compression techniques. This chapter provides an analysis of previous research efforts. It also highlights the gaps that this work aims to address.
- **Chapter 4** describes the tools and methods used in this study. It introduces different compression techniques, such as Principal Component Analysis, Scalar and Vector Quantization, Huffman Coding, and Variational Autoencoders. It also describes the experimental setup, including scenarios for presence and activity recognition,

and the implementation details of the proposed approach.

- **Chapter 5** presents the results of the study and evaluates the performance of the proposed methods. It includes performance analysis for presence and activity recognition, as well as storage and coding analysis to evaluate the efficiency of the compression techniques applied.
- Finally, **Chapter 6** summarizes the work, summarizes the main findings, and discusses possible future developments. It highlights the contributions of the research and suggests directions for further improvements and applications in real-world scenarios.

2 | Background

2.1. Internet of Things Forensics

Digital forensics is the process of discovering, collecting, analyzing, and preserving electronic data for legal purposes. A subset of this field, IoT forensics, is an evolving domain focused on on investigating crimes involving IoT devices. It plays a crucial role in in criminal investigations for extracting vital digital evidence from IoT devices to solve crimes. In incident response, it helps identify the source and extent of security breaches involving connected devices and legal processes rely on IoT forensics to gather and preserve data that can be presented in court as evidence.

Unlike traditional digital forensics, which primarily examines computers, smartphones, tablets, and servers, IoT forensics extends the examination to a wider range of devices. These can include infant or patient monitoring systems, in-vehicle infotainment systems, traffic lights, smart home appliances, and even medical implants in humans and animals. In addition to the devices themselves, critical evidence may be found in IoT networks, routers, cloud storage that store voluminous amounts of data and firewalls that manage IoT communications.

IoT devices exchange information with millions of other connected devices worldwide. This large-scale, open communication makes them attractive targets for malicious actors. Often, attackers do not target IoT devices directly, but instead exploit them as tools to launch attacks on other systems. A major challenge is that IoT manufacturers tend to prioritize cost, size, and usability over security and forensic capabilities, leaving many devices vulnerable. Unlike cybersecurity measures, which focus on preventing or mitigating attacks, forensic techniques are concerned with identifying the source of an attack or those responsible and tracing the origins of an incident. [2]

IoT forensics can be classified into three main categories:

- **Device Forensics**: Extracting evidence from IoT devices such as sensors, health implants, smart meters, and other smart technologies.

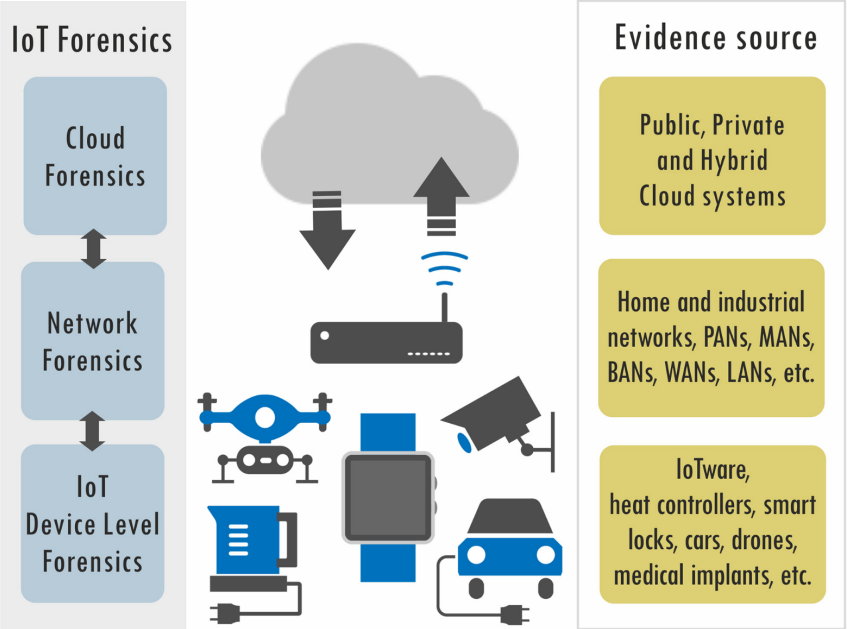


Figure 2.1: Components of the IoT Forensics. *Source:* [2]

- **Network Forensics**: Analyzing network logs, traffic patterns, and communication traces, including data from Bluetooth or other network models.
- **Cloud Forensics**: Investigating cloud services used by IoT devices, which can store vast amounts of potential evidence.

Although IoT forensics opens new opportunities for digital investigations, it also introduces significant challenges due to the complexity, diversity, and security limitations of IoT systems.

One of the main difficulties is data extraction and preservation, which involves collecting information from multiple IoT devices and storage media; the dynamic nature of IoT devices can result in a constantly changing environment and data sources, further complicating data extraction and analysis. The complexity of IoT networks and data storage technologies can make it difficult to store and interpret massive amounts of data, especially when working with remote or distributed devices.

The focus of this study is on **network forensics**, as we will analyze the information that can be extracted from the communication between two WiFi devices. Specifically, the work concentrates on finding a solution for storing massive amounts of data that can later be used in a forensic context.

2.2. Existing Approaches to Activity Recognition

The field of activity recognition is focused on the monitoring and understanding of human behaviors. Traditional approaches have relied on the use of wearable devices equipped with motion sensors, a method known as active monitoring. The sensor data collected is processed either locally or transmitted to a server for activity classification. This active monitoring method achieves 90% accuracy in the recognition of activities such as sleeping, sitting, standing, walking, and running [3]. However, this approach is hindered by user compliance and scalability issues.

Alternatively, camera-based systems have been used for passive activity recognition but their reliance on line-of-sight and privacy concerns make them unsuitable for many environments.

Therefore, there is a need for a passive monitoring system based on a wireless signal that does not violate people's privacy. Because of its availability in indoor areas, recently, WiFi has been the focus of much research on activity recognition. Such systems consist of a WiFi access point and one or several WiFi-enabled devices located in different parts of the environment. *Received Signal Strength* (RSS) is a commonly used metric. RSS measures the power of a received radio signal and has been extensively applied in identification tasks. When a person is between a WiFi device and an access point, the signal attenuates, resulting in detectable RSS variations thus enabling classification process. However, RSS has limitations; it cannot capture real-time signal fluctuations due to movement, as it remains unstable even in a static environment. [4]

In addition to RSS, modified WiFi hardware such as the *Universal Software Radio Peripheral* (USRP) system has been used for passive tracking. This system measures Doppler shifts in *Orthogonal Frequency Division Multiplexing* (OFDM) signals caused by human motion using a technique called *Frequency Modulated Carrier Wave* (FMCW). The Doppler shift correlates with distance and can be used to estimate the location of a target. Movement toward the receiver produces a positive Doppler shift, while movement away from the receiver produces a negative shift.

However, this approach requires modified hardware, which typically results in higher costs and reduced availability. To address this, efforts have been made to use commercial WiFi access points for passive detection without modifying the WiFi system. One effective approach is based on WiFi Channel State Information (CSI), which captures dynamic changes in the environment due to human movement. CSI-based behavior detection is based on the observation that different human motions create different wireless channel

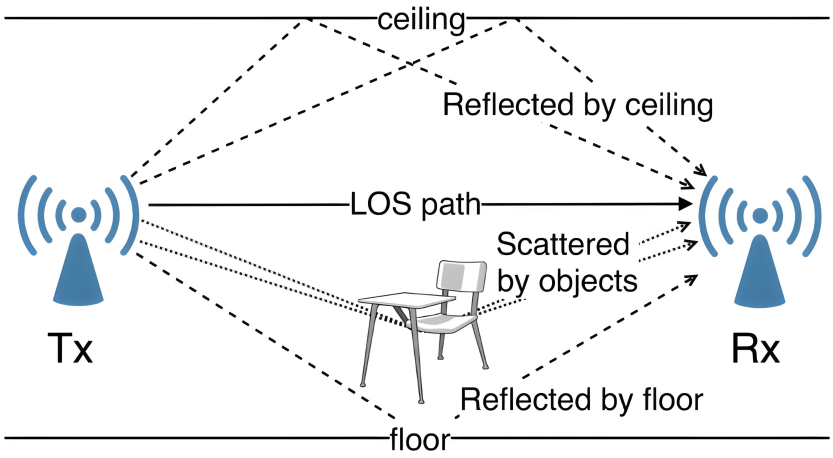


Figure 2.2: Radio signal propagation in indoor environment, *Source:* [7]

disturbances. By analyzing the correlation between CSI dynamics and user activity, human behavior can be identified. [5, 6]

2.3. Channel State Information

Channel State Information is a fundamental concept in wireless communications, providing a detailed representation of how signals propagate from a transmitter to a receiver. Unlike RSS, CSI provides a more robust and reliable measure by accounting for signal scattering, environmental attenuation, and distance. It captures both the amplitude and phase of the signal across multiple subcarriers, enabling a more in depth understanding of the channel characteristics.

2.3.1. Wireless Signal Propagation

Wireless signals do not travel along a single path; instead, they propagate via multiple paths due to reflection, scattering, and diffraction. These interactions create multiple copies of the signal, each arriving at the receiver with a slight delay, different amplitude, and phase shift. This phenomenon is called *multipath propagation*. Because these copies can interfere constructively or destructively, the net received signal can vary significantly over time and frequency, impacting performance and reliability.

The multipath channel, also called *Channel Impulse Response* (CIR) in the time domain, is typically modeled as a sum of delayed, attenuated, and phase-shifted impulses. Math-

ematically, it can be written as:

$$h(t) = \sum_{n=1}^{N} a_n \, e^{j\phi_n} \, \delta(t - \tau_n))$$

where a_n is the amplitude, ϕ_n is the phase shift and τ_n is the delay of the nth path, while $\delta(\cdot)$ represents the Dirac delta function (an impulse function). [8]

In the frequency domain, the multipath causes frequency-selective fading, which is characterized by the *Channel Frequency Response* (CFR), which is essentially the Fourier transform of CIR:

$$H(f) = \sum_{n=1}^{N} a_n e^{j\phi_n} e^{-j2\pi f \tau_n}.$$

It represents how different frequency components are affected by the channel; each path contributes a term with an exponential phase shift $e^{-j2\pi f \tau_n}$due to its delay τ_n.

CIR and CFR are measured by decoupling the transmitted signal from the received signal. In the time domain the received signal $r(t)$ is the *convolution* of the transmitted signal $s(t)$ and the channel impulse response $h(t)$:

$$r(t) = s(t) \otimes h(t).$$

This means that received signal is obtained by adding copies of the transmitted signal, each shifted and scaled according to the CIR. If we know both $r(t)$ and $s(t)$, we can mathematically extract $h(t)$ by performing a process called *deconvolution*, which is computationally complex.

However, as represented in Figure 2.3, it is possible to take advantage of the the Fourier transform, as convolution in the time domain corresponds to multiplication in the frequency domain:

$$R(f) = S(f)H(f)$$

where $R(f)$ and $S(f)$ are the received and transmitted signal spectrum respectively while $H(f)$ is the channel frequency response. This means that instead of performing complex deconvolution in the time domain, we can just divide in the frequency domain:

$$H(f) = \frac{R(f)}{S(f)}$$

which is simpler that performing deconvolution. Then, $h(t)$ can be retrieved by performing

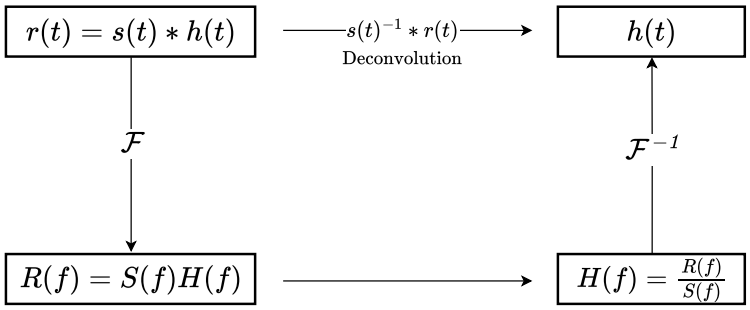


Figure 2.3: The process of obtaining the Channel Impulse Response

the inverse Fourier transform, with additional steps required in real-world applications:

$$h(t) = \frac{1}{P_s}\mathcal{F}^{-1}\left\{S^*(f)R(f)\right\}$$

where $S^*(f)$ is the complex conjugate of $S(f)$, and P_s is an approximation of the transmitted signal power. [9, 10]

2.3.2. Mathematical Representation

CSI was originally introduced to enhance data transmission reliability, it enables adaptive signal rate adjustments based on channel conditions. As wireless signals propagate through multiple paths, obstacles and reflections alter their amplitude and phase. The analysis of these variations facilitates the estimation of channel quality and the optimization of data rates.

CSI is represented as a complex matrix, capturing the amplitude and phase response of a signal's transmission channel for each subcarrier. In IEEE 802.11 standards, CSI is obtained from the PHY layer using Orthogonal Frequency Division Multiplexing (OFDM) technology. Mathematically, the wireless channel is represented as:

$$\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{N}$$

where $\mathbf{H}$ is the channel matrix representing CSI, $\mathbf{Y}$ and $\mathbf{X}$ are the received and transmitted signal vectors, respectively, and $\mathbf{N}$ denotes the noise which is often modeled as *circular symmetric complex normal* $\mathcal{N}_{\mathcal{C}}$.

Recent advances in the wireless community make it possible to get the sampled version of CFR from commercial-off-the-shelf Wi-Fi Network Interface Cards (NICs). The extracted CFR are often referred to as a series of complex numbers. The CSI for the ith subcarrier

is given by:

$$H(i) = |H(i)|e^{j\angle H(i)}$$

where $|H(i)|$ represents the amplitude, and $\angle H(i)$ represents the phase.

2.3.3. Estimation Techniques

Accurate CSI acquisition is crucial for optimizing wireless communication systems. Channel estimation techniques are employed to determine the CSI, which can be broadly categorized into:

- **Pilot-Based Estimation**: This method is widely used due to its simplicity and effectiveness. Known pilot symbols are transmitted, allowing the receiver to estimate the channel response from the received pilot signals. Expressing the pilot sequence as

  $$\mathbf{P} = [p_1, \ldots, p_N]$$

  the received signal is modeled as:

  $$\mathbf{Y} = [y_1, \ldots, y_N] = \mathbf{HP} + \mathbf{N}.$$

  With this notation, channel estimation means that $\mathbf{H}$ should be recovered from the knowledge of $\mathbf{Y}$ and $\mathbf{P}$.

- **Least Squares (LS) Estimation**: This approach minimizes the squared error between the received and expected pilot signals. However, it may not take noise into account effectively. With $\mathbf{P}$ representing the matrix of pilot symbols and $\mathbf{Y}$ the received signals, the LS estimate of the channel matrix $\mathbf{H}$ is given by:

  $$\mathbf{H_{LS}} = \mathbf{YP}^*(\mathbf{PP}^*)^{-\mathbf{1}}$$

  where $\mathbf{P}^*$ (or $\overline{\mathbf{P^T}}$) is the conjugate transpose of $\mathbf{P}$, meaning it is obtained by taking the transpose of $\mathbf{P}$ and applying complex conjugation to each of its entries. The error is minimized when $\mathbf{PP}^*$ is a scaled identity matrix.

- **Minimum Mean Square Error (MMSE) Estimation**: This technique incorporates statistical information about the channel and noise to minimize the mean square error. It generally provides better performance than LS estimation but at the cost of increased computational complexity.

- **Other Techniques**: The work of Almamori and Mohan [11] and Essai Ali and Taha

[12] explores new methods for CSI estimation using Kalman filters and Recurrent Neural Networks (RNNs) respectively.

2.3.4. Data Collection Techniques

To analyze CSI data, it must first be extracted at the *physical layer* (PHY), which manages the transmission and reception of raw data bits over a communication channel. Since CSI is measured close to the hardware level and not directly accessible to users, its extraction depends on the firmware and chipset of specific radio devices. As a result, CSI extractors are typically designed for particular manufacturers' chipsets.

The first tool enabling CSI extraction was the Intel 5300 Network Card, which collected CSI data from 30 subcarriers. Later, the Atheros CSI Tool was introduced, allowing detailed extraction of PHY wireless communication data from Atheros Wi-Fi Network Interface Cards. This tool, based on the open-source `ath9k` Linux kernel driver, supports all types of Atheros 802.11n WiFi chipsets [13, 14].

New tools and frameworks have emerged to improve CSI extraction and support modern Wi-Fi standards. Nexmon CSI is an extraction framework for Broadcom and Cypress Wi-Fi chips used in Raspberry Pi devices and smartphones. It extends CSI support to 802.11ac, enabling real-time data collection for applications such as location and motion sensing [15].

Another advancement is AX-CSI, the first tool capable of extracting CSI from 802.11ax consumer devices using the Broadcom 43 684 Wi-Fi chipset. It supports up to 160 MHz wide CSI with 4×4 MIMO [16].

2.3.5. Multiple-Input Multiple-Output Systems

Multiple-Input Multiple-Output (MIMO) systems are a core component of modern wireless communications, using multiple antennas at both the transmitter and receiver ends. This architecture enables the simultaneous transmission and reception of multiple data streams over the same frequency band, improving system capacity, spectral efficiency and reliability. By taking advantage of multipath propagation, MIMO systems can take advantage of spatial diversity and spatial multiplexing techniques. The result is increased data throughput and improved robustness to fading and interference. [17]

This technology make use of multiple antennas to achieve the following results:

- **Spatial Multiplexing**: MIMO systems split a data stream into multiple parallel

streams, each transmitted from a different antenna. This increases the data rate without requiring additional spectral resources.

- **Diversity Gain**: Multiple antennas provide several independent paths for signal propagation, which can be combined to mitigate the effects of fading and improve link reliability.
- **Beamforming**: With the knowledge of channel conditions, MIMO systems can adjust the phase and amplitude of signals across the antennas to direct the energy towards the intended receiver, enhancing signal strength and reducing interference. This makes accurate and up-to-date CSI essential for MIMO performance.

MIMO and OFDM

When MIMO is combined with OFDM, the benefits become even more pronounced. OFDM splits the available frequency band into numerous narrow subcarriers, each experiencing relatively flat fading. This segmentation simplifies equalization and makes the system highly adaptive to frequency-selective fading. In a MIMO-OFDM system, each subcarrier can exploit spatial diversity independently, enabling robust performance even in environments with severe multipath propagation. [18]

OFDM was first introduced in Wi-Fi standards with IEEE 802.11a in 1999, operating in the 5 GHz band with 64 subcarriers, each spaced 312.5 kHz apart. This marked a significant improvement over previous technologies. Since then, several upgrades have been made, leading to the development of IEEE 802.11ax (Wi-Fi 6) in 2019. Wi-Fi 6 introduced *Orthogonal Frequency-Division Multiple Access* (OFDMA), which allows multiple users to share the same channel simultaneously, enhancing the efficiency of frequency usage. The standard supports channel bandwidths of 20 MHz, 40 MHz, 80 MHz, and 160 MHz, with corresponding subcarrier counts of 256, 512, 1024, and 2048. To accommodate the increased number of subcarriers, the subcarrier spacing was reduced to 78.125 kHz, further improving the overall system's performance and adaptability in complex wireless environments.

CSI in MIMO-OFDM Systems

Despite the advantages offered by this technologies, the integration of MIMO with OFDM increases system complexity. Accurate CSI is critical for optimal beamforming and spatial multiplexing. However, acquiring reliable CSI is challenging as the number of antennas grows and the channel varies rapidly over time.

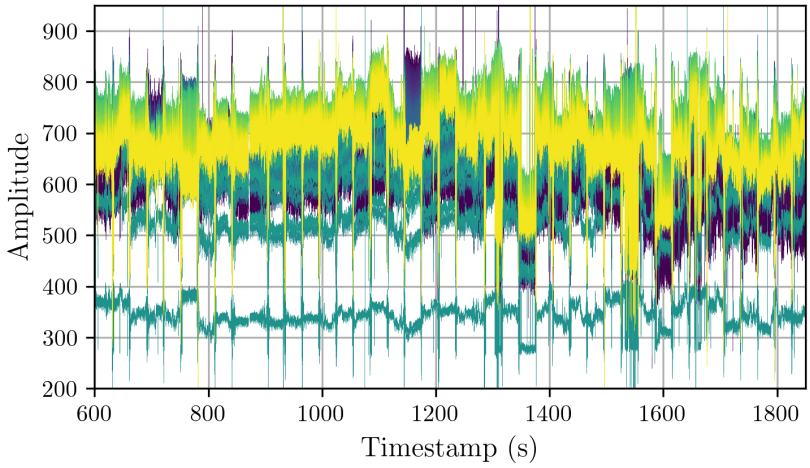


Figure 2.4: Example of a CSI amplitude capture

Specifically, for a MIMO system with **M** transmit antennas and **N** receive antennas, the receiver estimates the CFR for each antenna pair. This results in $\mathbf{M} \times \mathbf{N}$ complex-valued coefficients per subcarrier in systems using OFDM. So, the channel is represented as a matrix of CFR values:

$$\mathbf{H}(f) = \begin{bmatrix} H_{11}(f) & H_{12}(f) & \cdots & H_{1M}(f) \\ H_{21}(f) & H_{22}(f) & \cdots & H_{2M}(f) \\ \vdots & \vdots & \ddots & \vdots \\ H_{N1}(f) & H_{N2}(f) & \cdots & H_{NM}(f) \end{bmatrix}$$

Where each element $H_{ij}(f)$ represents the complex CFR between the jth transmit antenna and the ith receive antenna at frequency f. The complete CSI set, comprising all these CFR measurements, provides a full description of the channel's behavior across the entire MIMO link.

2.3.6. CSI for Classification

The behavior recognition system, based on signal processing techniques, consists of four main components: data collection, base signal selection, signal pre-processing, and behavior classification.

After collecting CSI data, selecting an appropriate base signal is crucial. Next, signal processing techniques are applied to eliminate noise from the CSI stream, ensuring more accurate data. Once processed, human behaviors can be recognized using three main

approaches: pattern-based, model-based, and deep learning-based methods, as described in *Section 3.1.*

Signal Selection Accurate and comprehensive data collection that captures user movements is essential for effective behavior recognition. Without adequate data, it is hard to mine behavior profiles or build a mathematical model to characterize the correlation between human behavior and corresponding signals. For this reason, the selection of the base signal plays a crucial role in behavior recognition as it decides the resolution of data and the accuracy of behavior identification. [6]

In related research, the following types of signals are commonly used:

- **Amplitude**: The amplitude of CSI is generally stable and useful for feature extraction and classification, as it quantifies signal power attenuation after multi-path effects. Many studies emphasize amplitude because of its direct correlation with path loss and fading. Figure 2.4 shows how amplitude data appears across multiple subcarriers.
- **Phase**: Theoretically, the phase contains more information than the amplitude and can be used to provide details into timing, propagation delays, and even localization. However, the phase of CSI is affected by *Carrier Frequency Offset* (CFO) and *Sampling Frequency Offset* (SFO), which result from transmitter-receiver clock misalignment. CFO occurs due to discrepancies in central frequencies between transmitter and receiver clocks, while SFO is caused by variations in the receiver's analog-to-digital (ADC) converter. These errors make raw phase information unreliable.
- **Combination of Amplitude and Phase**: A combination of amplitude and phase might be a useful approach in that it can effectively utilize the advantages of both features and improve recognition accuracy.
- **Phase Difference**: Compared to phase, phase difference has several benefits such as space diverse and less noise. It is sensitive to human movements and can be leveraged to identify behaviors. However, the approach usually uses an antenna array to measure phase difference, which leads to this method unavailable without an antenna array.

Signal Pre-processing As shown in Figure 2.4, CSI measurements consist of helpful signals, disordered noise, and a few outliers. These distortions mainly come from environments, signal interference, and some moving persons. Therefore, data pre-processing is significant since it can remove outliers, filter noise, calibrate the phase, and retain

valuable data. Specifically, to reduce invalid data and enhance the accuracy of behavior recognition, It is needed to eliminate the noise caused by multi-path effects and hardware devices.

Common pre-processing methods used in research include:

- **Low-Pass Filter**: A standard method that allows only signals below the cutoff frequency to pass through. Since human activities generally cause low-frequency variations in CSI signals, low-pass filtering effectively removes high-frequency noise.
- **Hampel Filter**: An efficient method for detecting and eliminating outliers that deviate significantly from neighboring data points. It identifies outliers and replaces them with the local mean using a moving average window, thereby reducing the impact of invalid data.
- **Principal Component Analysis**: Widely used dimensionality reduction technique that transforms data into a new coordinate system. This transformation simplifies complex datasets while preserving the most significant information as explained in depth in *Section 4.1.1*
- **Phase Sanitation**: A corrective process that mitigates CFO and SFO errors by applying a linear transformation, ensuring more accurate phase-based measurements.

3 | Related Works

3.1. Human Behavior Recognition

Human behavior recognition using CSI has gained significant attention due to its potential to provide accurate and device-free sensing solutions. A recent survey by Wang et al. [6] classified the techniques used in the field into three groups: model-based, pattern-based, and deep learning-based.

Pattern-based approaches focus on identifying distinct patterns in CSI signals to recognize human behavior. The construction of features plays a crucial role in distinguishing between different activities. These methods rely on sample data to establish behavioral patterns. CSI data pre-processing is essential to remove errors and noise caused by hardware imperfections and environmental factors.

Model-based approaches utilize mathematical or physical models to establish relationships between CSI variations and human activities. The main challenge is to extract effective signal features to build models that accurately capture signal changes caused by human actions. Compared to pattern-based methods, model-based approaches require less data, making them advantageous in certain scenarios.

Deep learning-based approaches have gained popularity due to their ability to automatically handle complex feature extraction. Unlike traditional methods, deep learning does not require a predefined feature extraction step, as feature learning is integrated into network structures. While often considered a subset of pattern-based techniques, deep learning methods differ in that they rely on large datasets to train networks with millions of parameters, enabling automatic feature extraction and classification by neural network layers.

CSI-based detection falls into two main categories: **specific behavior detection** and **activity inference**. The former identifies actions such as walking, sitting, falling, and hand gestures, while the latter infers behavioral intent, such as smoking detection or crowd counting.

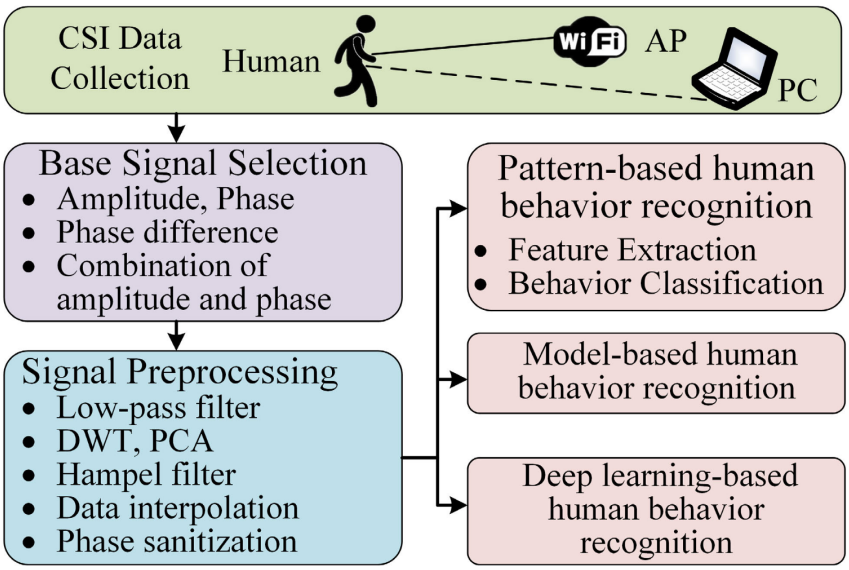


Figure 3.1: CSI-based behavior recognition framework. *Source:* [6]

Wu et al. [19] introduced TW-See, a through-wall system that identifies seven activities such as walking, hand swinging, or boxing. It pre-process data by applying a low-pass filter and extracts features and associates human motion with CSI changes by applying PCA. A sliding window algorithm detects activity boundaries from the extracted features, which are then fed into a three-layer back-propagation neural network. Experiments show that it can identify activity in a through-the-concrete-wall scenario with an average accuracy of 94.46%.

Wang et al. [20] developed WiSDAR, a system that recognizes eight activities such as walking, falling or sitting in different environments. It uses PCA for noise reduction, and CNN and Long Short-Term Memory neural network for classification. The experimental results show that WiSDAR achieves 96% stable identification precision for eight kinds of activity identification.

Zou et al. [21] proposed DeepHare, a device-free human behavior recognition system using Autoencoder Long-term Recurrent Convolutional Network to classify five daily activities (sit, stand, walk, run, lie down) across three environments (conference room, office, apartment). The system processes CSI data applying low-pass and median filters to reduce noise. Trained on 400 000 CSI samples, DeepHare achieves 97.6% accuracy in activity recognition.

The previously mentioned studies explore the potential of CSI analysis to maximize the performance of human behavior recognition. This work focuses specifically on **pattern-based** and deep **learning-based behavior recognition techniques** to investigate the

impact of compression applied to CSI data on performance and storage.

3.2. CSI Compression

CSI compression plays a crucial role in *Multiple Input Multiple Output* (MIMO) systems, not only for communication but also for applications in wireless sensing. MIMO technology uses multiple antennas to transmit and receive data simultaneously, improving the speed, reliability, and efficiency of wireless networks. However, one of the primary challenges in MIMO systems is the increasing demand for CSI feedback, which can consume significant up-link bandwidth. Several studies have investigated various CSI compression techniques to mitigate this problem.

The work of Örnhag et al. [22] investigates the impact of traditional, non-deep learning compression techniques on CSI feedback reduction in massive MIMO systems. Massive MIMO is a key technology in 5G, improving spectral and energy efficiency by using numerous antennas at the base station. However, accurate down-link CSI is required, and in frequency division duplex systems, this information must be sent back to the base station, increasing up-link bandwidth usage. To address this, compression techniques are applied at the user equipment before transmission and then decompressed at the base station. The study highlights the limitations of existing datasets, evaluation metrics, and quantization approaches. It also demonstrates that traditional transform coding can outperform many deep learning-based methods when using the same bit allocation, and that not all neural network architectures scale efficiently when quantized.

Similarly, Ferguson et al. [23] explores CSI compression in OFDM systems, which are widely used in modern wireless communication. Traditional OFDM systems use fixed modulation schemes on all subcarriers, leading to inefficiencies as signal-to-noise ratios (SNRs) vary due to fading and interference. Adaptive bit loading can optimize performance by adjusting modulation levels based on individual subcarrier SNRs. However, this requires frequent CSI feedback, which increases exponentially in multi-antenna MIMO systems. To reduce bandwidth consumption, the study applies Huffman coding to quantized subcarrier gains and shows that a reduction to 7 bits per sample provides optimal feedback reduction, while 5 bits per sample achieves maximum compression.

In addition, Mismar and Kaya [24] investigates the use of deep autoencoders for CSI compression in massive MIMO systems. Although autoencoders perform lossy compression, the study finds that they are effective for practical CSI feedback applications. By evaluating performance under different channel models and compression ratios, the study identifies an optimal trade-off between compression efficiency and reconstruction accu-

racy. In particular, the study shows that the runtime complexity of autoencoders remains independent of the compression ratio, allowing adaptive compression based on real-time channel conditions and SNRs variations.

Beyond communication, CSI is also essential in wireless sensing applications, especially in human activity and gesture recognition. Yang et al. [25] addresses the challenge of high-dimensional CSI measurements in WiFi sensing. Traditional WiFi sensing systems have limited scalability due to computational constraints on edge devices. The study proposes EfficientFi, a framework that combines edge and cloud computing to process CSI for large-scale sensing applications. It uses a deep neural network to compress CSI at the edge, efficiently transmit it to the cloud, and perform sensing tasks. Experiments demonstrate that EfficientFi reduces CSI data from 1.368 Mb/s to 0.768 kb/s while preserving data accuracy, significantly reducing communication overhead while maintaining over 98% classification accuracy in behavior recognition tasks.

As presented here, CSI compression plays an important role and shows potential in several fields. This work takes inspiration from these methods and applies compression to reduce the size of CSI data for forensic analysis. What differentiates this study is its focus on achieving high compression using various techniques and comparing them to identify the most effective method that improves storage efficiency without compromising performance.

4 | Tools and Methods

To conduct analysis in real-world scenarios, it is crucial to minimize the vast amount of data generated by IoT devices. This reduction not only optimizes storage but also reduces infrastructure and management costs. Therefore, this study explores various data compression techniques and their combinations to achieve efficient storage without compromising accuracy.

The following section is organized as follows: First, we introduce the tool used in our research. Next, we present the scenarios considered for evaluation. Finally, we detail the methods used to integrate the tools into these scenarios.

4.1. Compression Methodology

Forensic data storage in resource-constrained IoT systems presents a unique challenge, which requires a careful balance between efficient compression and preservation of critical information for forensic analysis. The volume of high-dimensional data generated by IoT devices, especially in the form of CSI, necessitates methods that can drastically reduce storage requirements without compromising the evidentiary value of the data. Traditional lossless compression methods, while ensuring complete data fidelity, often fall short in reducing storage needs, especially when long-term preservation is required for legal proceedings. This limitation forces the use of lossy compression techniques that are able to discard less critical components to achieve significant data reduction while preserving the essential signals needed for both forensic investigations and tasks such as human activity recognition.

The reasoning behind the selection of these methods is based on the double objective of forensic data integrity and analysis performance. On the one hand, it is essential that the data remain unaltered and legally admissible, as details may become important in reconstructing events during an investigation. On the other hand, the size and complexity of the data require aggressive reduction techniques to ensure that storage resources are not overloaded. This trade-off necessitates the use of methods such as dimensionality

reduction, which reduces the data to its most meaningful components, and quantization techniques, which eliminate redundancy while preserving the core characteristics of the signal. In addition, advanced machine learning approaches provide ways to deal with the noise and complexity typical of IoT environments. These methods work in combination with lossless techniques that are applied as a final step to ensure that the remaining data maintains its integrity and extends the compression.

The selection and combination of these tools is based on the need to create a robust, efficient pipeline that meets the needs of IoT forensics. The goal is to optimize storage without sacrificing the reliability and detail of the data required for accurate analysis and legal proceedings. This chapter delves into the details of these tools, exploring the choices involved and how each method contributes to a balanced approach that supports both the forensic and analytical objectives of the study.

4.1.1. Principal Component Analysis

Principal Component Analysis (PCA) is a widely used dimensionality reduction technique that transforms data into a new coordinate system, where the first few dimensions, or *principal components* (PCs), capture the most of the variance. This transformation simplifies complex datasets while preserving the most significant information, making PCA useful for applications such as noise reduction, data visualization, feature extraction and data compression. By selecting PCs that retain the most variation, PCA effectively reduces redundancy in datasets with correlated variables.

Given a set of variables, the first principal component is a linear combination of the original variables that maximizes variance. The second principal component captures the next highest variance after removing the effect of the first component, and and so on for the remaining components. PCA is particularly useful when variables in the dataset are highly correlated, as it reduces redundancy and improves computational efficiency. [26]

Mathematical Formulation

Data Standardization and Covariance To ensure that each variable contributes equally, PCA starts by standardizing the data. For a given variable x, standardization is performed by centering (subtracting the mean μ) and scaling (dividing by the standard deviation σ):

$$z = \frac{x - \mu}{\sigma}$$

Once the data is standardized, the covariance matrix $\mathbf{C}$ is computed:

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X}$$

where $\mathbf{X}$ is the matrix of centered data with n number of samples and p features. Each element c_{ij} in the covariance matrix captures how much variables i and j vary together, with diagonal elements representing variances and off-diagonals capturing covariances.

Eigenvalue Decomposition The principal components are obtained by solving an eigen decomposition of the covariance matrix:

$$\mathbf{Cv} = \lambda\mathbf{v}$$

Eigenvectors ($\mathbf{v}$) represent the directions in the feature space while Eigenvalues (λ) quantify the amount of variance along these directions. The eigenvalues are sorted in descending order, this way the principal components are ranked by their "importance". The eigenvector associated with the largest eigenvalue becomes the first principal component, and so on.

Select Principal Components Once the eigenvectors are computed, a projection matrix $\mathbf{W}$ of size $p \times p$ is constructed. Selecting the top k eigenvectors corresponding to the largest eigenvalues results in a matrix $\mathbf{W_k}$ of size $p \times k$. This matrix is then used to project the original data into a lower-dimensional space that retains most of the variance.

Transform Data Finally, the original data is then projected onto this new space:

$$\mathbf{Y} = \mathbf{XW_k}$$

Here, $\mathbf{Y}$ is the transformed dataset with reduced dimensions.

The reduced dimension can be quantified in terms of storage gains compared to the original data size. The size of the dataset in memory can be calculated as follows:

$$\text{Original Data Size} = n \times p \times \texttt{Sizeof}(\textit{datatype}) \tag{4.1}$$

$$\begin{aligned}\text{Reduced Data Size} &= \text{PCA Data Size} + \text{Transformation Matrix Size} && (4.2)\\ &= \big(n \times k \times \texttt{Sizeof}(\textit{datatype})\big) + \big(p \times k \times \texttt{Sizeof}(\textit{datatype})\big) && (4.3)\\ &= \big(n + p\big)\big(k \times \texttt{Sizeof}(\textit{datatype})\big) && (4.4)\end{aligned}$$

where `Sizeof`(*datatype*) represents the memory required to store a single numerical value (e.g., 4 bytes for a 32-bit float). The percentage reduction in storage, or storage gain, is then given by:

$$\text{Storage Gain} = \frac{\text{Original Data Size} - \text{Reduced Data Size}}{\text{Original Data Size}} \times 100 \tag{4.5}$$

Explained Variance

The amount of total variability captured by each principal component is expressed by the Explained Variance Ratio. This ratio is computed by dividing the variance λ_i (the eigenvalue) of the ith principal component by the total variance:

$$\text{Explained Variance Ratio for PC}_i = \frac{\lambda_i}{\sum_{j=1}^{p} \lambda_j}.$$

Consequently, the Cumulative Explained Variance for the top k components is then obtained by summing the individual ratios:

$$\text{Cumulative Explained Variance}_i = \frac{\sum_{j=1}^{k} \lambda_i}{\sum_{j=1}^{p} \lambda_j}.$$

Reconstruction

The transformed dataset is now represented as a matrix with principal components as its columns, which is ideal for various types of analysis. However, there may be situations where reconstructing the original dataset is necessary; for example, when you need to interpret the original features or perform analyses that require the initial variables. In such cases, it is possible to reverse the projection process described earlier to recover an approximation of the original data. Mathematically, this is done by just applying the transpose W_k^T to the transformed data:

$$\hat{\mathbf{X}} = \mathbf{Y}\mathbf{W}_k^T = \mathbf{X}\mathbf{W}_k\mathbf{W}_k^T.$$

where $\hat{\mathbf{X}}$ is the reconstructed data, that is in the subspace spanned by the selected k principal components. For data reconstruction, the transformation matrix $\mathbf{W}_k$ is essential. Consequently, when storing the compressed data, it is important to also save $\mathbf{W}_k$.

Reconstruction Error Since the reconstruction process in PCA relies on the transformation matrix $\mathbf{W}_k$, this method is by construction lossy. The reconstruction error quantifies how much information is lost when the dimensionality of the data is reduced by keeping only k components. While this error can be measured using the Frobenius norm (the square root of the sum of the squares of all its entries), one result of PCA is that the reconstruction error directly corresponds to the sum of the eigenvalues of the discarded components' covariance matrix:

$$\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 = \sum_{i=k+1}^{p} \lambda_i.$$

Here, the index i starts from $k+1$ because the eigenvalues λ_i are sorted in descending order.

PCA in Sensing

As presented in *Chapter 3*, for behavioral recognition tasks, PCA allows to filter out noise and redundant data from signals that are captured by off-the-shelf devices. By identifying the directions of maximum variance, PCA helps in uncovering the most informative features that distinguish different behaviors. This extraction of key features not only improves the interpretability of the data but also enhances the accuracy of subsequent classification or clustering algorithms applied in the recognition process. [27]

In this study, PCA is utilized as a lossy compression technique, discarding components that contribute less to the overall variance.

4.1.2. Scalar Quantization

Scalar Quantization (SQ) is a fundamental compression technique in digital signal processing and information theory. It involves mapping each scalar data value from a continuous range to a discrete set of levels. This process reduces the complexity of the data while preserving as much fidelity as possible. Quantization is widely used in digital signal processing, data compression, and image processing, among others.

In this study, we utilized the a well-known and efficient scalar quantization method which is the *Lloyd-Max Quantizer* (LMQ). It which minimizes the *Mean Squared Error* (MSE) by optimally positioning decision boundaries and quantization levels. Given a probability density function of the input data, LMQ refines its quantization scheme to achieve the optimal balance between accuracy and compression.

The Lloyd-Max algorithm operates iteratively to optimize quantization parameters. The process follows these steps assuming M quantization levels:

1. **Initialization**: Choose an initial set of **M** centroids $c_1, c_2, \ldots, c_M$, spaced uniformly or derived from the input data distribution. This points are the values chosen to represent the quantization levels or centers of the decision region.

2. **Determine Thresholds**: Compute thresholds τ_j for each quantization level using the midpoint formula:
$$\tau_j = \frac{1}{2}(c_j + c_{j+1})$$
This points are the values that separate adjacent quantization regions. These boundaries define the intervals within which the data points are assigned to a particular representation point.

3. **Update Centroids**: Adjust centroids c_j by computing the conditional mean of the input values **I** within each threshold:
$$c_j = E[\ \mathbf{I} \mid \mathbf{I} \in (\tau_{j-1}, \tau_j)]$$
where $\tau_0 = -\infty$ and $\tau_M = +\infty$.

4. **Iteration**: Repeat steps 2 and 3 until the improvement in MSE is negligible.

The MSE decreases or remains the same for each execution of step 2 and step 3. Since the MSE is nonnegative, the algorithm converges to a limit $\epsilon > 0$. This limit represents the best achievable quantization error for the given number of levels M and the input data distribution. [28]

Regarding storage gains with this method: the original data size can be expressed by equation 4.1. Given M quantization levels, each element requires $\log_2(M)$ bits for representation. Therefore, the size of the quantized data is:

$$\text{Quantized Data Size} = n \times k \times \log_2(M) + \texttt{Sizeof}(\textit{Codebook})$$

The resulting storage gains can then be computed using equation 4.5.

4.1.3. Vector Quantization

A vector quantizer converts a sequence of vectors into a digital representation for efficient transmission or storage. Its primary objective is to reduce the bit rate while maintaining an acceptable level of data fidelity. While scalar quantizers (one-dimensional quantiz-

ers) are simpler and perform well at high bit rates, vector quantizers provide superior performance by encoding vectors rather than individual scalar values. Optimal vector quantization strategies have been analyzed in-depth, particularly in their applications for data compression and signal processing. [29]

The key benefit of VQ is its ability to adapt to the distribution of data, ensuring that frequently occurring patterns are represented with higher accuracy while reducing redundancy in less common areas. Frequently occurring data points incur minimal error because they are well-represented by centroids, whereas rarer ones may experience higher distortion since fewer centroids are available to accurately capture their variability. This characteristic makes VQ highly suitable for lossy data compression. [30]

One of the most common methods for implementing VQ is through *k-means clustering*, where a set of input data points is partitioned into a predefined number of clusters, k. The algorithm iteratively assigns points to the nearest centroid and recalculates the centroids until convergence, minimizing distortion (sum of squared distances between data points and their nearest centroids). This process ensures efficient data representation.

In information theory, the mapping between centroid indices (or *codes*) and centroids is referred to as the *codebook*. This codebook allows data points to be quantized efficiently, thereby reducing redundancy and the expected distortion while preserving data fidelity.

After quantization as each vector is represented by the index of the closest codebook vector the size of the quantized data is:

$$\text{Quantized Data Size} = n \times \log_2(M) + \texttt{Sizeof}(\textit{Codebook}).$$

The resulting storage gains can then be computed using equation 4.5.

4.1.4. Huffman Coding

Huffman coding is a lossless data compression technique that optimizes the representation of symbols based on their frequency. This method constructs an optimal binary tree for encoding, making it widely used in file compression formats (ZIP, GZIP), multimedia encoding (JPEG, MP3), and telecommunications. It is valued for its simplicity, efficiency, and its ability to achieve near-optimal compression for many types of data.

The algorithm operates by first analyzing the frequency of each symbol in the dataset. It then constructs a binary tree, known as the *Huffman tree*, where each symbol corresponds to a leaf node. More frequent symbols are positioned closer to the root, resulting in shorter binary codes, while less frequent symbols are located deeper in the tree, leading

to longer codes. The process of constructing the Huffman tree is iterative and follows a greedy approach merging the two least frequent symbols until a single tree remains. The resulting *prefix-free code* ensures accurate decoding by preventing ambiguity. [31]

Huffman coding operates with an implementation efficiency of $\mathcal{O}(n \log n)$. Variable-length codes allow symbols to be represented by different bit lengths. By employing them, Huffman coding approaches the theoretical compression bound defined by *Shannon's Source Coding Theorem*. The theorem states that the average length of the encoded symbols cannot be shorter than the theoretical limit defined by the probabilities of the symbols. Mathematically:

$$\mathbb{E}\big[\ell(d(x))\big] \;\geq\; \mathbb{E}\big[-\log_2(P(x))\big]$$

where ℓ is the codeword length, $d(x)$ is the coding function, and $P(x)$ represents the symbol probability. The negative logarithm of the probability quantifies the information content required to encode the symbol, ensuring that codeword lengths won't be shorter than the limit described by Shannon's Theorem. By adhering to these principles, Huffman coding minimizes redundancy while preserving data integrity. [32]

In Huffman encoding, the storage requirements differ from other methods due to the use of lossless compression with variable-length codes. Both the Huffman tree (or codebook) and the compressed data need to be stored. For m distinct symbols out of n total symbols, the storage requirement can be expressed as:

$$\begin{aligned}\text{Reduced Data Size} &= \text{Compressed Data Size} + \text{Huffman Tree Size} \\ &= \big(n \times \texttt{Avg}(\textit{code length})\big) + \big(m \times \texttt{Sizeof}(\textit{tree node})\big)\end{aligned}$$

The resulting storage gains can then be computed using equation 4.5.

4.1.5. Variational Autoencoder

Autoencoders are neural networks used for unsupervised learning and dimensionality reduction. They work by compressing input data into a compact latent representation with an encoder and then reconstructing it back to its original form using a decoder. However, standard autoencoders may overfit to training data and fail to generate meaningful variations of the input. Variational Autoencoders (VAEs) address this problem by sharing the same architecture but using a probabilistic approach. Figure 4.1 shows the architecture of a VAE.

Instead of mapping each input to a single point in latent space, VAEs learn a distribution over latent space. This means that the encoder produces the parameters of a Gaussian

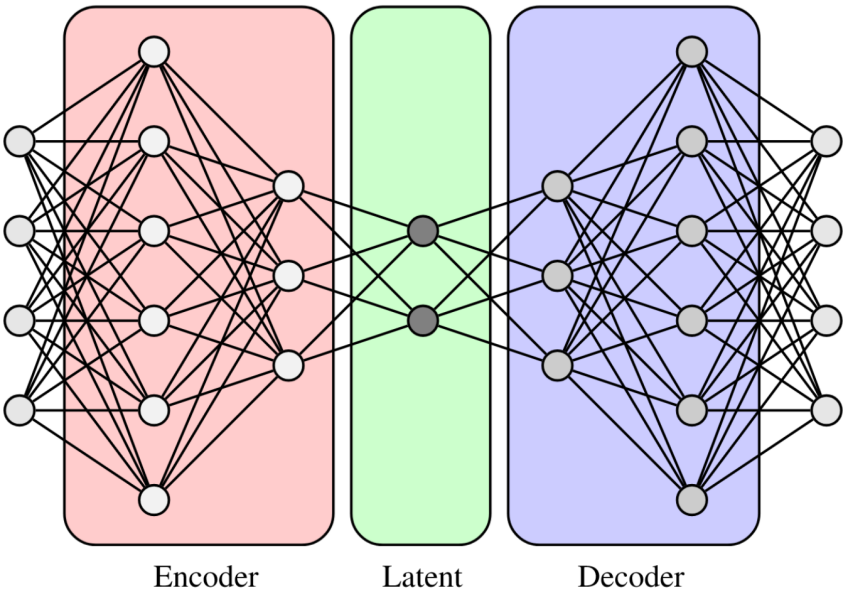


Figure 4.1: Components of a variational autoencoder. *Source:* [24]

distribution and allows the model to sample from this distribution, introducing variability that encourages the learning of more general and useful features. The decoder takes a sample from the latent distribution and maps it back to the input space. This also enables VAEs to generate new, meaningful samples rather than just memorizing training data. The advantage is that similar inputs are encoded into distributions that overlap in latent space, making interpolation between data points more meaningful. [33]

Training a VAE involves optimizing two key terms:

1. **Reconstruction Loss**: Measures how accurately the decoder reconstructs the input from the latent space. Typically measured using *Mean Squared Error* (MSE) or another distance metric.
2. **Kullback–Leibler (KL) Divergence**: Encourages the latent distribution to be similar to a standard normal distribution and prevents the model from collapsing into a deterministic mapping, so that the latent space remains meaningful for generation.

The loss function used to train a VAE is a combination of both terms:

$$\mathcal{L} = \text{Reconstruction Loss} + \beta \cdot \text{KL Divergence}$$

where β is a weighting factor that controls the trade-off between reconstruction accuracy and regularization.

Variational autoencoders not only serve as dimensionality reduction tools but also as generative models that are able of producing new, similar data by sampling from the latent space. This property is particularly useful in our case of unsupervised feature learning.

4.2. Scenarios and Experimental Setup

In order to evaluate the trade-off between accuracy and storage, two different use cases were considered. Both scenarios used a dataset captured from consumer IoT devices generating network traffic. The following subsections describe the scenarios in detail including the experiments descriptions and additional details regarding system architecture and testbed setup.

As previously stated, the primary objective of this work is to understand the impact of CSI compression on sensing. To that end, we explored a wide range of method combinations, starting with the approach used for recognition. The first scenario was examined utilizing both feature-based and deep learning-based methods, while the second scenario focused solely on deep learning techniques. For each method, we conducted various studies across different configurations to identify which combination of tools had the most significant effect on performance and storage efficiency.

4.2.1. Presence Detection Scenario

The objective of this use case is to determine the presence of a person in the room, which is also known as a binary classification problem.

The experiment conducted included a Teckin TC100 smart camera that generated network traffic on the 2.4 GHz frequency on a 20 MHz channel, resulting in 64 OFDM subcarriers. The *Access Point* (AP) used in the experiment was a Linksys WRT3200ACM. In the experimental setup, the smart camera (transmitter) and the sniffer were placed on opposite sides of the room. The person alternated between being present and absent, moving within a designated area. To ensure that variations in packet generation were not influenced by motion detection, the camera's field of view did not overlap with the movement area and the camera's audio was disabled. Experiments were repeated multiple times while recording ground truth timestamps of the person's presence. During the experiment, 1200 balanced samples were collected. [34]

The experimental setup is illustrated in Figure 4.2, showing the placement of the smart camera, sniffer, and movement area.

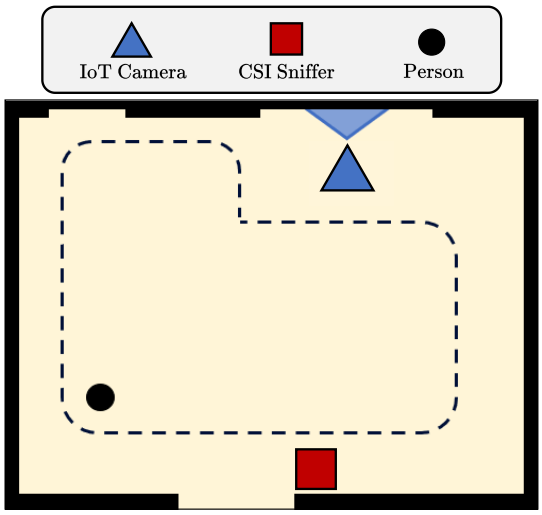


Figure 4.2: Sketch of device placement in the room

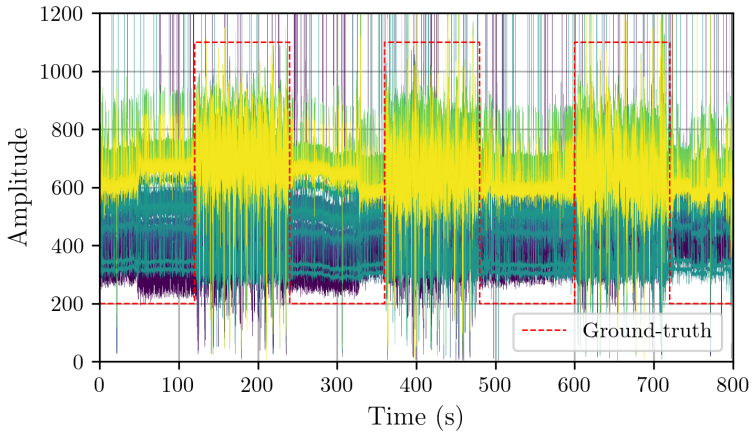
The sniffer's system architecture consists of two components: the AP and a Raspberry Pi acting as the CSI collector. The AP runs `OpenWrt`, an open-source embedded operating system used to route network traffic, while the Raspberry Pi is equipped with the Nexmon CSI extractor. This extractor enables CSI feature extraction from sniffed Wi-Fi frames, giving users control over settings such as the sniffing channel and device filters. The feature happens thanks to the *Legacy Long Training Field* (L-LTF) found in every transmitted frame. The L-LTF contains information that allows the device to detect the signal for each OFDM subcarrier, enabling the receiver to estimate the CSI matrix by comparing received signals with this reference. [35]

The output is a CSV file containing timestamps and CSI data for each OFDM subcarrier, represented as complex numbers.

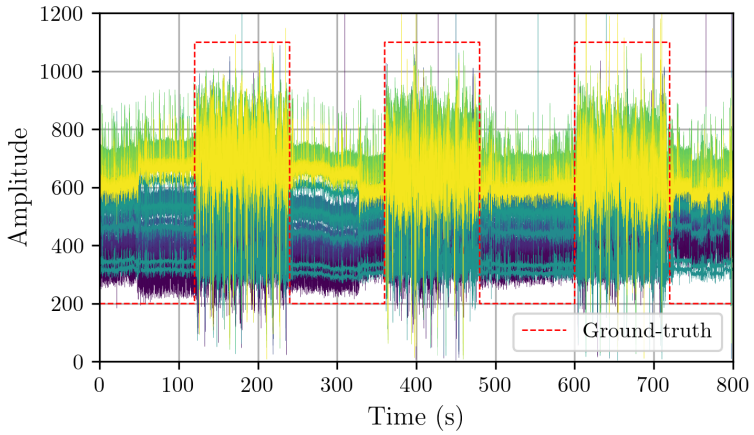
4.2.2. Experimental Methods

After the data collection phase, the CSI data was pre-processed following the procedure presented in Palmese and C. Redondi [34]. First, the phase information was discarded, keeping only the amplitude component of each received subcarrier.

Outlier Removal Then, for every captured frame at timestamp t, and each subcarrier i, any CSI amplitude value A_t^i that was greater than the local mean μ within the window w by more than a predefined threshold was replaced by the previous CSI amplitude A_{t-1}^i. The threshold is $\lambda = 3$ times the standard deviation σ of the window. Mathematically,



(a) CSI subcarrier amplitudes before outlier removal



(b) CSI subcarrier amplitudes after outlier removal

Figure 4.3: Comparison of CSI subcarrier amplitudes before and after the outlier removal process, showing the effectiveness of filtering out anomalous data points.

this is expressed as:

$$\bar{A}_t^i = \begin{cases} A_t^i & \frac{\left|A_t^i - \mu_{t-w:t}^i\right|}{\sigma_{t-w:t}^i} < \lambda \\ A_{t-1}^i & \text{otherwise} \end{cases}$$

where $\mu_{t-w:t}^i$ and $\sigma_{t-w:t}^i$ are the mean and standard deviation of subcarrier i in the preceding window w. Figure 4.3 shows the difference in CSI subcarrier amplitudes before and after the outlier removal process. The figure also displays the ground truth, which shows when the person is inside the room walking.

Feature-Based Subcarrier Aggregation After outlier removal, the CSI amplitude samples were aggregated into a single value. Specifically, frames were grouped into non-

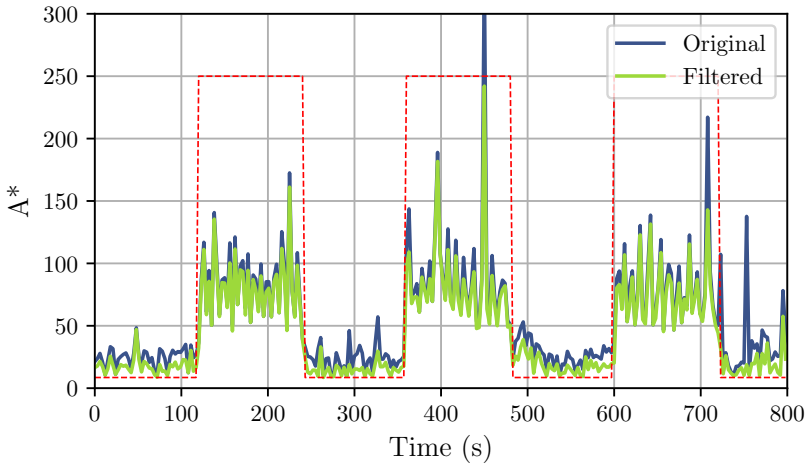


Figure 4.4: Comparison of A^* before and after the outlier removal process

overlapping 3-second time windows. For each time window k, the standard deviation σ_k^i of the CSI samples was computed for each subcarrier i. Finally, the average of all computed standard deviations across subcarriers was obtained for each window:

$$A_k^* = \mu([\sigma_k^i \ \forall i \in I])$$

where I is the set of available subcarriers. On a 20 MHz channels in the 2.4 GHz band, legacy frames use 64 available subcarriers indexed in the range $[-32, 31]$. However, not all the subcarriers are used for data. Indeed, 4 of these are pilot subcarriers not carrying data (with index -21, -7, 7, 21), and 8 of these are guard subcarriers with null amplitude: the central one with index 0 and the 7 outermost with index -32,-31,-30,-29,29,30,31. Thus, we discard such subcarriers and only keep the 52 carrying data, having index in the range $I = [-28, \ldots, -22, -20, \ldots, -8, -6, \cdots - 1, 1, \ldots, 6, 8, \ldots, 20, 22, \ldots, 28]$.

Threshold Classification The result of this is a one-dimensional time series A_k^* that can be easily used for threshold classification. To choose the proper threshold τ the *Receiver Operating Characteristic* (ROC) curve has been obtained from an experiment with 1000 different values of τ. The *Area Under the Curve* (AUC) was used to assess classification performance, selecting the τ value that maximized the trade-off between the true positive rate (TPR) and false positive rate (FPR).

Once the optimal threshold τ was established, presence detection was performed using a

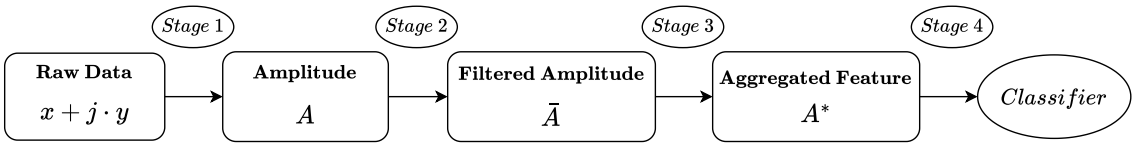


Figure 4.5: Presence detection scenario data analysis pipeline

simple decision rule:

$$\text{Presence} = \begin{cases} 1 & A_k^* \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

This approach ensures a robust classification mechanism by distinguishing significant variations in CSI amplitude from background noise. The final classification results were validated against ground truth labels confirming the effectiveness of the proposed methodology.

Deep Learning-Based Approach Feature extraction was replaced with an automated process using a VAE. Instead of relying on a threshold algorithm to determine whether a person was present in the room, a *Multi-Layer Perceptron* (MLP) was employed for binary classification. Except for the differences already outlined, the dataset underwent the same pre-processing steps as before. The training data was then used to train the VAE to ensure effective handling of the test data. Figure 4.7 illustrates the structure of the VAE and MLP. More information on the hyperparameters and architecture of both models can be found in *Section 4.3*.

Experiments Performed Within each group, the data was further divided into training and testing sets, each containing multiple windows of 32 frames. The training dataset had 56 000 samples while the test dataset had 14 700 samples. The average storage requirement for a frame after pre-processing was 330 bits.

With the test dataset, we conducted a comprehensive series of experiments aimed at investigating the trade-offs between accuracy loss and storage optimization for efficient forensic analysis. Our goal was to assess how these processing steps influence both the classifier's performance and the required storage. The experiments were designed as follows:

- **PCA**:
  - *Purpose*: To evaluate how much of the original data variance is needed to maintain classification performance and to determine the minimum number of principal components necessary to preserve classification accuracy.

  - *Methodology*: Classification was performed iteratively by first using a single principal component and then progressively adding more components. The cumulative explained variance was monitored to identify the point at which additional components provided diminishing returns.
  - *Expected Outcome*: Determine the minimum number of principal components required to achieve near-optimal classification accuracy.

- **SQ**:
  - *Purpose*: To evaluate the effect of reducing precision on each subcarrier independently, balancing the trade-off between data compression and classification accuracy.
  - *Methodology*: Classification was performed iteratively on data that was quantized subcarrier by subcarrier. We increased the quantization resolution progressively up to 256 bits to observe how different quantization levels influence the results.
  - *Expected Outcome*: Determine the quantization level that provides significant storage savings with minimal impact on classification performance.

- **VQ**:
  - *Purpose*: To investigate whether grouping subcarriers before quantization better preserves features relationships, thus enhancing classification accuracy while reducing storage.
  - *Methodology*: Subcarrier data were grouped and quantized as vectors, with quantization levels increased up to 256 bits, and classification performance was evaluated at each level.
  - *Expected Outcome*: Assess if vector quantization provides superior compression by retaining feature information compared to scalar quantization.

- **PCA + SQ**:
  - *Purpose*: To explore the combined effect of dimensionality reduction and scalar quantization on both accuracy and storage optimization.
  - *Methodology*: After applying PCA, each principal component was quantized individually subcarrier by subcarrier with increasing quantization levels up to 256 bits, followed by classification.

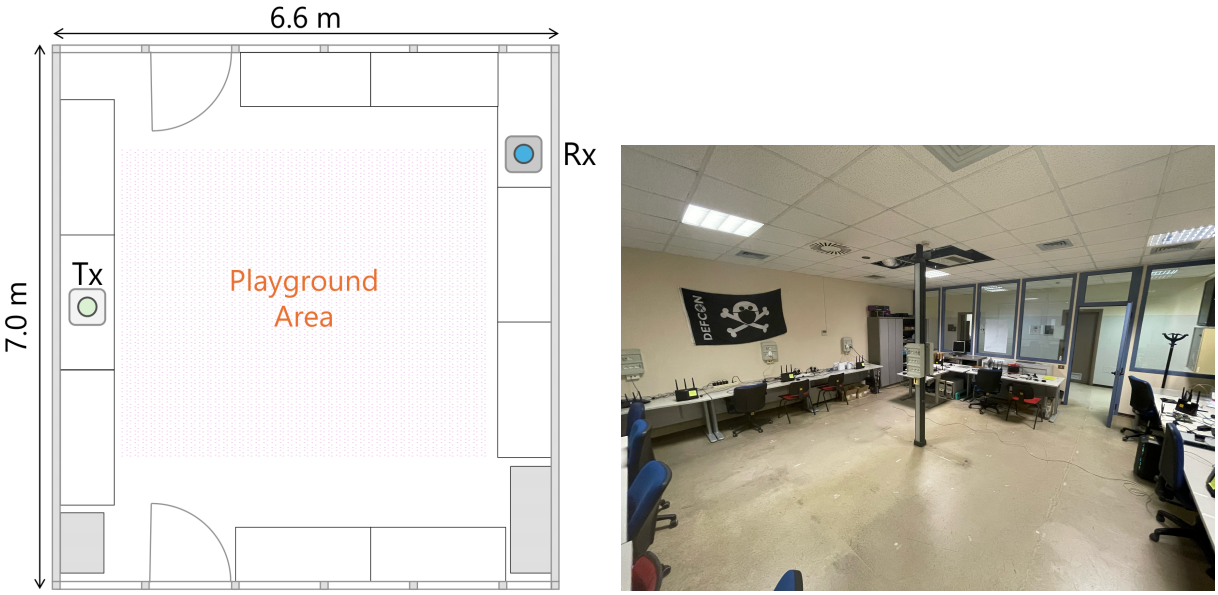
  - *Expected Outcome*: Assess whether applying PCA prior to scalar quantization leads to a more efficient representation that keeps essential features even at lower quantization levels.

- **PCA + VQ**:
  - *Purpose*: To determine if the combination of PCA and vector quantization further enhances storage efficiency while sustaining classification performance.
  - *Methodology*: PCA was first applied for dimensionality reduction, followed by grouping subcarriers and applying vector quantization with levels up to 256 bits. Classification was then performed on the quantized data.
  - *Expected Outcome*: Identify an optimal configuration that leverages both PCA and vector quantization to minimize storage requirements with minimal accuracy degradation.

- **VAE_SQ and VAE_VQ**:
  - *Purpose*: To explore a deep learning-based approach where the latent space representation of the data is subjected to quantization, aiming to achieve efficient storage and robust classification.
  - *Methodology*: A Variational Autoencoder (VAE) was employed to learn a compact latent representation of the input data. This latent space was then quantized using two methods:
    * *VAE_SQ:* Scalar quantization was applied to each element of the latent vector.
    * *VAE_VQ:* Vector quantization was applied to groups within the latent representation.

    Quantization levels were progressively increased up to 256 bits, and the impact on classification performance was evaluated.
  - *Expected Outcome*: Evaluate whether the VAE's compact latent representation can withstand quantization, thus reducing storage while preserving the features needed for classification.

In each experiment, **Huffman Encoding** (ENC) was applied to further compress the results.



(a) Sketch of the test environment (b) Photograph of the laboratory setup

Figure 4.6: Test environment for the data collection campaign, showing the router placement and laboratory setup.

4.2.3. Activity Recognition Scenario

The objective of this use case is to identify the activity that a subject is performing, which is also known as a multi-class classification problem.

The testbed consisted of two Asus RT-AX86U routers, each equipped with four antennas and MIMO capabilities. One router generated dummy Wi-Fi traffic using the injection feature of AX-CSI, while the other router collected CSI from received frames. This scenario is part of a larger experimental study involving seven different experiments, as described in Cominelli et al. [36]. The data were collected in a $45m^2$ laboratory, a representative indoor space for CSI sensing experiments. For the analysis, we focused on five out of twelve activities performed by a participant, including both static (sitting, empty room) and dynamic (walking, jumping, running) movements.

The testbed configuration is depicted in Figure 4.6, providing a visual representation of the router placement and laboratory environment.

The dataset for this scenario is structured in three dimensions. The first dimension represents time, capturing 12 000 consecutive CSI samples at a rate of 150 samples per second. The second dimension corresponds to subcarriers, covering 2048 OFDM subcarriers within a 160 MHz 802.11ax frame. Finally, the third dimension represents receiving antennas. Although data were collected from four independent receiver antennas to ensure robust

multi-antenna reception, we only consider the data from the first antenna for simplicity. All values within the dataset are stored in a `double complex` format (16 bytes), preserving the full fidelity of the CSI signal for high-precision analysis. [37]

4.2.4. Experimental Methods

In this scenario, to ensure an effective and leak-free data split, the pre-processing procedure divided the dataset into training and testing subsets systematically. Each activity was segmented into five distinct windows, each lasting 16 seconds. Within these windows, 9 seconds were allocated for training, while 3 seconds were designated for testing. The remaining time in each window functioned as a buffer, preventing overlap between training and testing data and ensuring a clear separation between consecutive activity windows.

To increase data utilization, the dataset was processed using a sliding window approach, extracting segments of 450 CSI frames at a time with a step size of 1. This method allowed for maximal data usage during training while still maintaining a sufficient volume of data for testing. By structuring the data in this way, the pre-processing step optimized both the consistency and reliability of the activity recognition model.

For what concerns the classification, the analysis pipeline is based on a Variational Autoencoder coupled with a Multi-layer Perceptron, as sketched in Figure 4.7 and detailed in the next section. The VAE consists of three convolutional layers that extract features, followed by a fully connected layer that compresses the information into a smaller set of neurons before mapping it to the latent space. The MLP, instead, is composed of three fully connected hidden layers with ReLU activations and dropout mechanisms to mitigate overfitting.

As for the experiments conducted in this scenario, we follow the same methodology as in the presence detection scenario. To summarize, these experiments include Principal Component Analysis (PCA), Scalar Quantization (SQ), Vector Quantization (VQ), and their combinations: PCA + SQ, PCA + VQ, as well as Variational Autoencoder-based methods (VAE_SQ and VAE_VQ).

For this study, the training dataset comprised 33 750 samples, while the test dataset contained 11 250 samples. After pre-processing, the average storage requirement for a single frame was calculated to be 16 593 bits, reflecting the efficiency of the applied techniques in data compression and feature extraction.

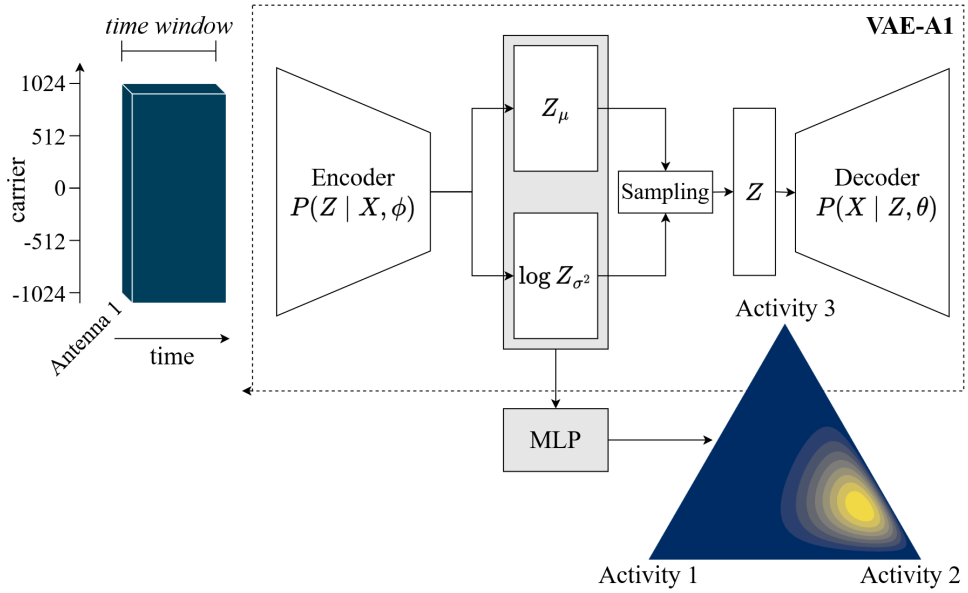


Figure 4.7: Overview of the VAE architecture and its interaction with the MLP

4.3. Implementation Details

This section provides a detailed explanation of the VAE and the MLP used in both scenarios, discussing their architectural choices, hyperparameters, and theoretical foundation to ensure effective classification.

Overview of the Variational Autoencoder The input to the VAE is a three-dimensional tensor with a shape of $(32, 56, 1)$ in the presence detection scenario and $(450, 2048, 1)$ in the activity recognition one. Each dimension of this tensor carries specific significance in representing the collected CSI data. The first dimension, consisting of the time steps, captures the temporal evolution of the signal. The second dimension corresponds to the number of subcarriers. Finally, the third dimension represents the number of antennas used in the system. In this particular scenario, the setup always includes a single antenna, meaning this dimension remains fixed at 1 throughout all experiments.

The encoder maps CSI sequences into two Gaussian-distributed variables with parameters:

- z_mean: Represents the mean (expected value) of the latent representation.
- z_log_var: Represents the logarithm of the variance, used to model uncertainty.

Since each normal distribution in the latent space is defined by two parameters (mean and variance), the latent space has a dimension of 4. This means that the VAE encodes the input signal into a lower-dimensional probabilistic representation, which helps extract

meaningful features while reducing noise.

As summarized in Table 4.1 and Table 4.3, the encoder extracts hierarchical features through convolutional layers (Conv2D), followed by a fully connected layer (Dense) to condense information into 16 neurons before projecting into the latent space. The convolutional layers play a critical role in feature extraction by learning patterns and dependencies in the CSI data. By applying filters over the input, these layers capture features such as signal variations caused by human motion. The dense layer instead acts as a feature aggregator, combining the extracted spatial features into a more compact and meaningful representation. This step is essential for dimensionality reduction. This transformation helps the VAE encode meaningful variations in human activity while discarding redundant information.

Overview of the Multi-Layer Perceptron The MLP is trained using the VAE's latent space representation rather than raw CSI data. It is trained as a binary classifier for presence detection, determining whether a person is present in the environment. In the case of activity recognition, the MLP is trained as a multi-class classifier to identify the specific activity being performed by the person.

As summarized in Table 4.2 and Table 4.4, The MLP consists of:

- **Input Layer**: Accepts the 4-dimensional latent space representation from the VAE.
- **Hidden Layers**: Three fully connected layers with ReLU activations and dropout to prevent overfitting.
- **Output Layer**: A final Dense layer with softplus activation, producing a binary probability distribution for presence detection or a 5-class distribution for human activity recognition.

VAE Architecture for Presence Detection

Layer	Filters/Neurons	Stride	Activation
Encoder	**Input Size:** $(32, 56, 1)$		
Conv2D	$(3, 3) \times 64$	$(1, 1)$	ReLU
Conv2D	$(3, 3) \times 128$	$(1, 1)$	ReLU
Flatten	-	-	-
Dense	16	-	ReLU
Latent Space	**Latent Dimension:** 4		
Decoder			
Dense	46592	-	ReLU
Reshape	$(28, 52, 32)$	-	-
Conv2DTranspose	$(3, 3) \times 32$	$(1, 1)$	ReLU
Conv2DTranspose	$(3, 3) \times 32$	$(1, 1)$	ReLU

Table 4.1: Architecture of the VAE in the presence detection scenario.

MLP Architecture for Presence Detection

Layer	# Neurons	Dropout Rate	Activation
Input	4	-	-
Dense	32	-	ReLU
Dropout	-	40%	-
Dense	64	-	ReLU
Dropout	-	40%	-
Dense	32	-	ReLU
Dense (Output)	2	-	Softplus

Table 4.2: Architecture of the MLP in the presence detection scenario.

VAE Architecture for Activity Recognition

Layer	Filters/Neurons	Stride	Activation
Encoder	**Input Size:** $(450, 2048, 1)$		
Conv2D	$(5, 8) \times 32$	$(5, 8)$	ReLU
Conv2D	$(5, 8) \times 32$	$(5, 8)$	ReLU
Conv2D	$(2, 4) \times 32$	$(2, 4)$	ReLU
Flatten	-	-	-
Dense	16	-	ReLU
Latent Space	**Latent Dimension:** 4		
Decoder			
Dense	2304	-	ReLU
Reshape	$(9, 8, 32)$	-	-
Conv2DTranspose	$(2, 4) \times 32$	$(2, 4)$	ReLU
Conv2DTranspose	$(5, 8) \times 32$	$(5, 8)$	ReLU
Conv2DTranspose	$(5, 8) \times 32$	$(5, 8)$	ReLU

Table 4.3: Architecture of the VAE in the activity recognition scenario.

MLP Architecture for Activity Recognition

Layer	# Neurons	Dropout Rate	Activation
Input	4	-	-
Dense	32	-	ReLU
Dropout	-	40%	-
Dense	64	-	ReLU
Dropout	-	40%	-
Dense	32	-	ReLU
Dense (Output)	5	-	Softplus

Table 4.4: Architecture of the MLP in the activity recognition scenario.

5 | Results

This section presents the results through three analyses: *Performance Analysis*, *Storage Analysis* and *Encoding Analysis*. Organizing the results this way helps show the balance between storage efficiency and performance. The comparisons allow for an accurate evaluation of each technique under different conditions, helping to understand their relative advantages and limitations.

5.1. Performance Analysis

The Performance Analysis focuses on the difference between the scores obtained in the experiments and those obtained by testing with uncompressed data. Rather than presenting raw performance scores, the results are expressed using a score-loss metric, which quantifies the performance degradation caused by compression. This approach provides a clearer understanding of how different compression techniques impact the model's overall accuracy and effectiveness.

5.1.1. Presence Detection

Our initial analysis focused on determining the number of principal components required to achieve 95% of the cumulative explained variance, a threshold we considered a reasonable starting point. As illustrated in Figure 5.1, out of the 56 available components, only $N_{PCA} = 7$ were necessary to exceed this threshold, while increasing to $N_{PCA} = 16$ brought the explained variance close to 98%. This result was encouraging, as it implied that at least 40 components could be discarded with minimal impact on the analysis. Consequently, we concentrated our study on scenarios employing $N_{PCA} = [2, 4, 8, 16]$ components. Although we did explore additional components, we present only these values for clarity, as they represent the optimal trade-off between accuracy loss and storage gains, a balance that will be further discussed later in this chapter.

Feature Based Method As shown in Figure 5.2, a series of experiments were conducted to compare the performance of PCA in combination with Scalar Quantization and

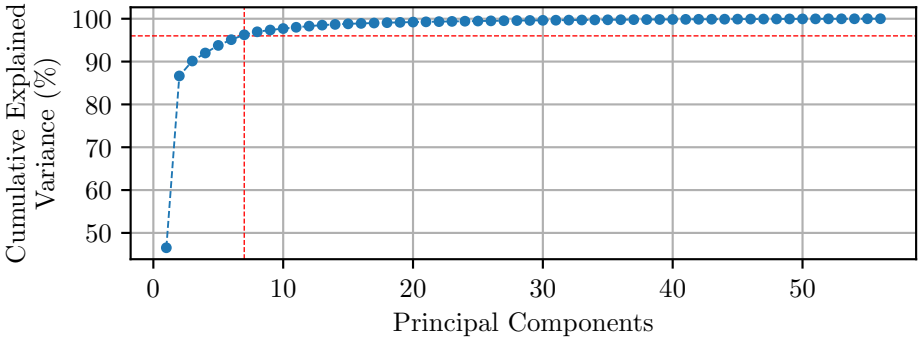
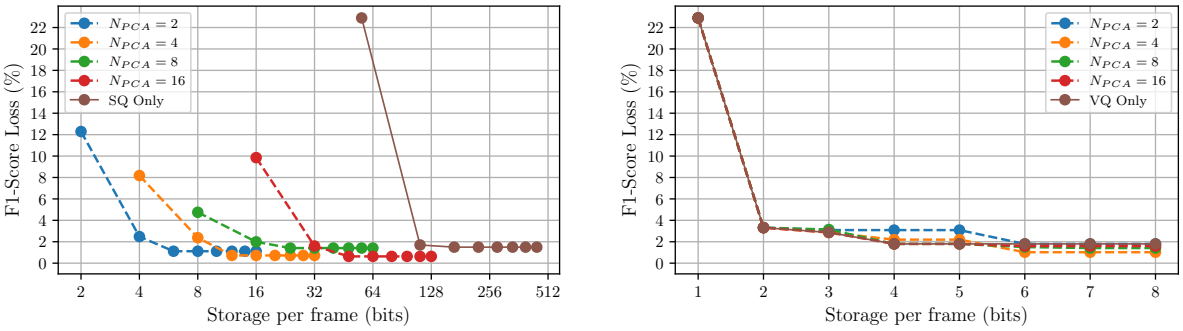


Figure 5.1: Cumulative Variance of PCA's components in presence detection scenario



(a) Relative F1-Score loss with PCA and SQ

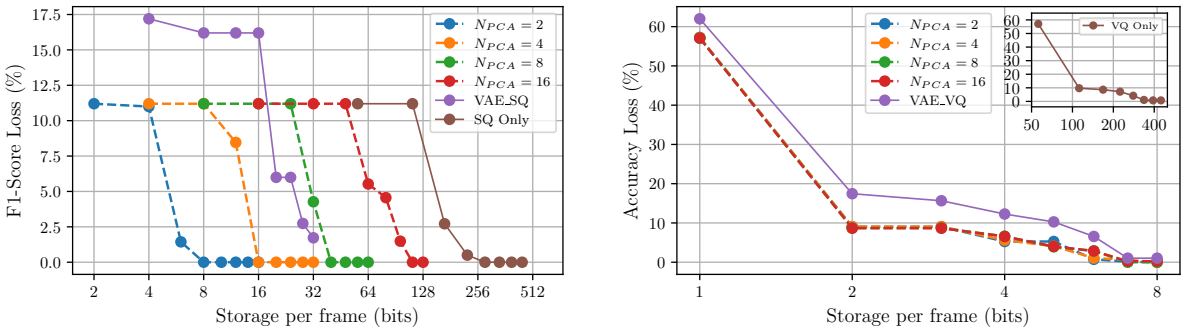
(b) Relative F1-Score loss with PCA and VQ

Figure 5.2: Performance results with the feature based method

Vector Quantization against a traditional feature-based method. These experiments evaluate PCA's ability to reduce storage requirements while maintaining model performance, and examine how the number of principal components affects this balance.

The results indicate that when using Scalar Quantization, PCA effectively reduces storage needs while maintaining high performance levels, with an observed performance loss of approximately 1%. Furthermore, increasing the number of principal components results in a slight improvement in performance. However, this improvement comes at a significant cost in terms of storage requirements. Thus, further increases in the number of components lead to diminishing returns in performance.

In contrast, when PCA is applied in conjunction with Vector Quantization, there is little to no impact on performance. The graph illustrates that for all PCA configurations, regardless of the number of principal components, the performance is very similar that of the quantization-only baseline. This suggests that VQ compresses feature information in a way that makes PCA's dimensionality reduction ineffective. As a result, PCA with SQ



(a) Relative F1-Score loss with PCA and SQ

(b) Relative F1-Score loss with PCA and VQ

Figure 5.3: Performance results with the deep learning based method

provides a balance between storage efficiency and performance preservation, while PCA with VQ provides little additional benefit.

Deep Learning Based Method The results using deep learning-based methods, shown in Figure 5.3, are consistent with previous findings: PCA remains effective when combined with SQ, but has minimal impact when combined with VQ.

Additionally, the plots illustrate a novel experiment where quantization is applied directly to the output of a Variational Autoencoder. The purpose of this experiment was to explore whether the latent space representation produced by the VAE could serve as an effective compressed form of the data, potentially outperforming the combined use of PCA and quantization. However, the results indicate that this approach does not lead to superior performance. Instead, the experiment shows a performance loss ranging from 5% to 15% in most cases, with the lowest loss being about 2% at the highest quantization levels (256 bits). In contrast, PCA maintains relatively high accuracy even at much lower quantization levels (8-16 bits).

The performance degradation in the VAE-based quantization approach is likely due to challenges in forming well-separated clusters within the latent space. Unlike PCA, which preserves variance and emphasizes key features, VAEs learn probabilistic representations that may not align with optimal clustering for classification. This suggests that while VAEs have potential for data compression, further refinements are necessary to improve their clustering properties and effectiveness with quantization.

A direct comparison between the two sets of experiments highlights the superiority of VQ in terms of compression efficiency. Even in the worst-case scenario, VQ is able to reduce storage requirements from 128 bits down to just 8 bits, while maintaining a performance

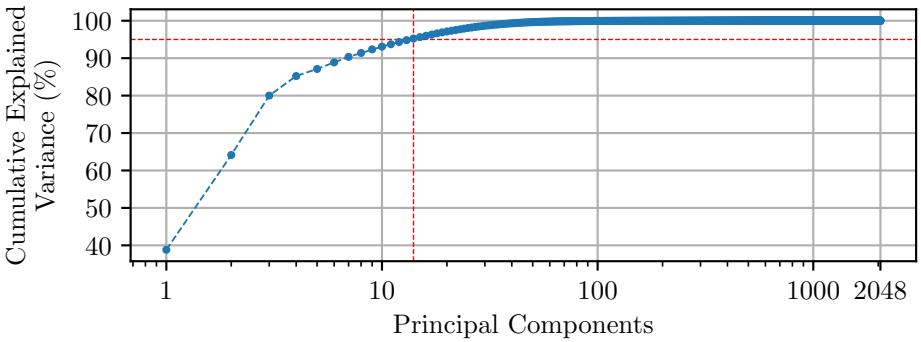
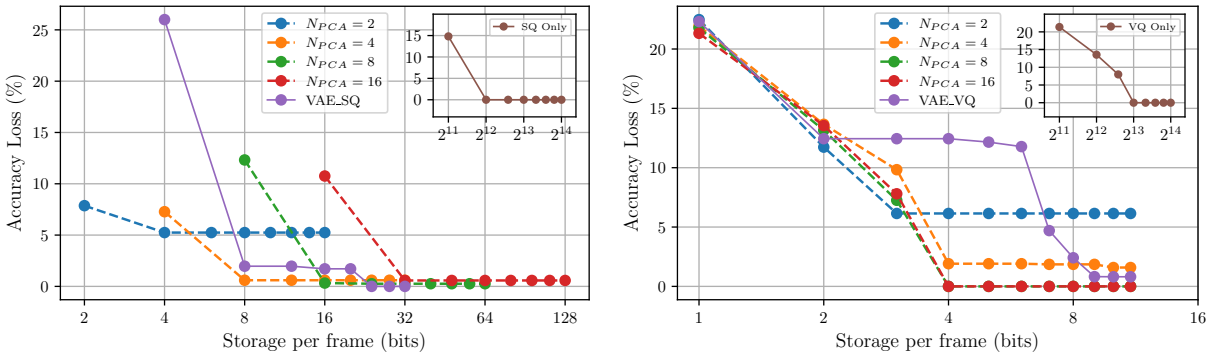


Figure 5.4: Cumulative Variance of PCA's components in activity recognition scenario



(a) Relative F1-Score loss with PCA and SQ

(b) Relative F1-Score loss with PCA and VQ

Figure 5.5: Performance results in the activity recognition scenario

loss comparable to SQ, typically around 2-3%. The overall performance degradation observed for both methods is likely related to the nature of the compression process itself.

5.1.2. Activity Recognition

In a similar fashion to our previous scenario, we performed the same analysis in this scenario, maintaining the 95% cumulative explained variance threshold as our baseline. As illustrated in Figure 5.4, out of the 2048 available components, only $N_{PCA} = 14$ were required to exceed this threshold. Notably, the results are not that different, if not more encouraging, as we achieved a similar reduction in dimensionality, effectively discarding nearly 2034 components, with minimal impact on the analysis. Although we explored additional components, we focused our study on scenarios employing $N_{PCA} = [2, 4, 8, 16]$ components.

In the scenario presented in Figure 5.5, the poor performance of using quantization alone is worsened by the significantly larger number of features (2048 vs. 56 in the previous

scenario). This reinforces the effectiveness of PCA when combined with quantization. Across all tested scenarios, PCA with quantization remains the most effective approach in maintaining performance while reducing storage requirements.

However, a distinction emerges when quantizing the output of a Variational Autoencoder. When SQ is applied to the VAE-compressed representation, the performance remains nearly identical to that of the uncompressed data, demonstrating that this approach can achieve a competitive balance between accuracy and storage efficiency. The compressifon performance of the VAE-SQ method is comparable to that of PCA combined with SQ, suggesting that for certain applications, VAE compression followed by SQ could serve as an alternative to traditional PCA-based methods.

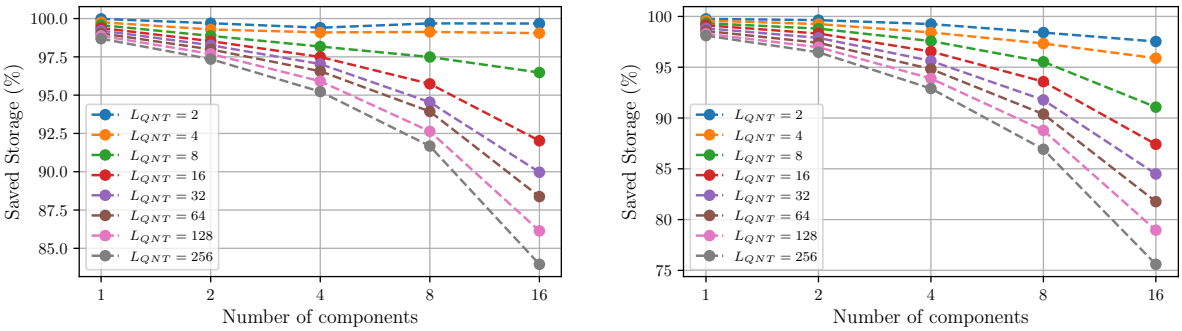
Conversely, the results for Vector Quantization tell a different story. When VQ is applied to the VAE output, performance deteriorates significantly, with an observed accuracy loss of approximately 12%. However, this performance loss decreases as the quantization level increases, reducing to 2-3% at the highest quantization level (256 bits). This aligns with previous observations that VAE representations are more sensitive to low-bit quantization than PCA-processed data.

5.2. Storage Analysis

The Storage Analysis studies the difference in storage requirements between the original, uncompressed data and the data after undergoing the full processing pipeline, which includes PCA, SQ/VQ, and ENC.

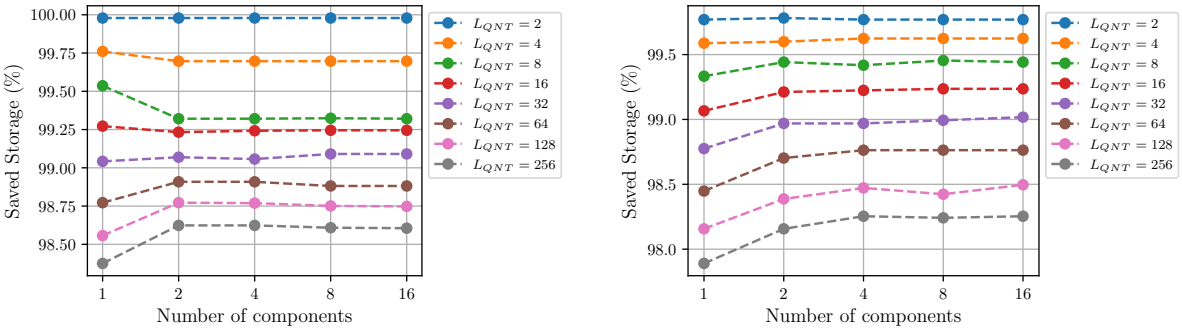
Scalar Quantization Figure 5.6 illustrates the impact of quantization and Huffman encoding on data compression compared to the original uncompressed data in the case of Scalar Quantization for both scenarios. As deductible from the graphs the behavior is similar in both scenarios. The number of PCA components significantly affects the percentage of storage saved. As the number of components increases, the storage savings decreases at all quantization levels. Lower quantization levels maintain higher storage savings, while higher quantization levels result in less compression. The difference in saved storage between different quantization levels becomes more pronounced as the number of components increases.

Given the previously discussed impact on performance, the configuration $N_{PCA} = 4$ and $L_{QNT} = 16$ seems to be an optimal balance for both scenarios, achieving less than 1% performance loss while saving over 97% of storage compared to the original uncompressed data. Increasing the number of components further reduces storage savings, with the



(a) Analysis in the presence detection scenario (b) Analysis in the activity recognition scenario

Figure 5.6: Storage analysis with SQ and ENC vs original data



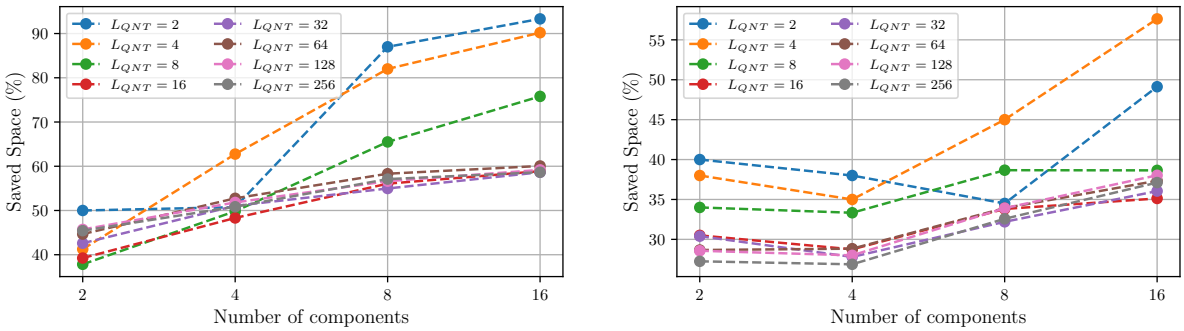
(a) Analysis in the presence detection scenario (b) Analysis in the activity recognition scenario

Figure 5.7: Storage analysis with VQ and ENC vs original data

lowest being around 85% in the presence detection scenario and around 75% in the activity recognition scenario, which is still a significant reduction.

Vector Quantization Figure 5.7 shows the impact of quantization and Huffman encoding on vector quantization for both scenarios. The number of PCA components has little effect on storage savings, which remain consistently high, with a minimum around 98% in both scenarios. This is because VQ compresses data as a whole rather than component by component. Thus, the primary factor influencing storage savings is the quantization level.

With the configuration $N_{PCA} = 4$ and $L_{QNT} = 16$, storage savings reach 99%, largely due to VQ rather than encoding. In this setup, experiments perform well, with only a 2% performance loss. However, the deep learning-based method in the presence detection scenario experiences a 7% performance loss. To achieve a similar 2% loss, a higher quantization level of $L_{QNT} = 128$ would be needed, which remains viable as storage savings



(a) Analysis in the presence detection scenario

(b) Analysis in the activity recognition scenario

Figure 5.8: Analysis of the impact of Huffman coding on SQ data

would only slightly decrease to 98.75%, a negligible reduction.

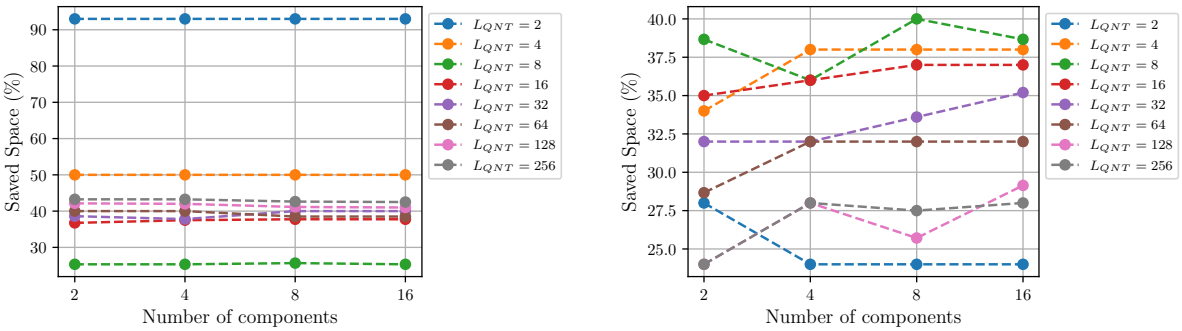
5.3. Encoding Analysis

This analysis examines the impact of applying Huffman coding at the end of a compression pipeline, after PCA and quantization, to maximize storage savings. With this type of compression, the gains depend on how the redundancy is distributed in the data.

Scalar Quantization As shown in Figure 5.8, the effectiveness of Huffman coding depends on both the number of PCA components and the quantization levels. In the presence detection scenario shown in Figure 5.8a, fewer quantization levels, such as $L_{QNT} = 2$, 4, and 8, result in high redundancy, allowing Huffman coding to achieve up to 90% compression. On the other hand, higher quantization levels, from $L_{QNT} = 16$ to $L_{QNT} = 256$, spread the data over more symbols, reducing redundancy and thus compression savings.

Increasing the number of components improves compression, especially with aggressive quantization. However, as quantization becomes finer, additional components offer diminishing returns. In fact, compression efficiency does not increase linearly with the number of components or quantization levels. At higher quantization levels, savings stabilize around 60%.

In the activity recognition scenario shown in Figure 5.8b, the compression performance is somewhat similar to the other scenario. The percentage of space saved ranges from a maximum of 58% to a minimum of 25%, which is lower than the other scenario, but the behavior remains consistent. When the quantization resolution is very low, using only $L_{QNT} = 2$ or $L_{QNT} = 4$ levels, the data has significant redundancy, allowing the encoder



(a) Analysis in the presence detection scenario (b) Analysis in the activity recognition scenario

Figure 5.9: Analysis of the impact of Huffman coding on VQ data

to reduce the storage requirement by about 38-49%. With fewer quantization levels, much of the redundant information can be removed.

As the number of quantization levels increases, from $L_{QNT} = 8$ to $L_{QNT} = 256$, the representation becomes more detailed. This additional detail results in a larger set of unique symbols and less redundancy, so the compression gains drop to about 27-38%.

In summary, optimal compression balances the number of components and quantization resolution. Both scenarios show that lower quantization levels lead to higher compression gains due to increased redundancy, while higher quantization levels lead to smaller but more predictable savings.

Vector Quantization With vector quantization, the picture is quite different from the scalar case, as shown in Figure 5.9. In the presence detection scenario shown in Figure 5.9a, the compression performance remains consistent across different quantization settings, regardless of the number of components. This consistency indicates that vector quantization effectively captures the joint statistics of the components, so increasing the number of dimensions does not significantly affect compressibility.

The primary factor affecting compression performance is the level of quantization. Extremely low settings, such as $L_{QNT} = 2$, results in savings exceeding 90%. This is probably due to the high redundancy of the data at such low resolutions. Higher settings, such as $L_{QNT} = 4$ or 8, result in lower savings, about 50% and 25%, respectively. At the highest resolutions, the savings gradually improve and stabilize around 42-43%.

The differences observed in the presence detection scenario compared to scalar quantization are also evident in the activity recognition scenario, as shown in Figure 5.9b. The actual storage required after coding remains consistent across different numbers of com-

ponents for each quantization level. For example, at the lowest resolution of $L_{QNT} = 2$ levels, the effective storage is nearly identical whether 2, 4, 8, or 16 components are used.

In summary, vector quantization in this activity recognition task results in stable and predictable compression performance that is largely independent of the number of components. Instead, the quantization resolution primarily dictates the trade-off between representational detail and space savings, with lower resolutions achieving higher relative compression gains, except at the lowest resolution.

The fact that the compression performance is not affected, or only slightly affected, by the number of components for each quantization setting highlights the strength of vector quantization. By considering groups of values together, it captures correlations between components, resulting in more predictable and stable compression performance across different data dimensions.

6 | Conclusions and Future Developments

This thesis aims to address the needs of digital forensics and human activity recognition in IoT environments by exploring lossy compression strategies for CSI data. As outlined in the introduction, the growth of IoT devices has led to a volume of high-dimensional data. Traditional methods struggle with prohibitive storage requirements, thus necessitating the development of innovative approaches that preserve critical information while achieving significant compression.

Our experimental results confirmed that dimensionality reduction using Principal Component Analysis combined with quantization techniques is a compelling solution. In both the presence detection and multi-class activity recognition scenarios, PCA combined with Scalar Quantization maintained high accuracy typically within 1–3% of the uncompressed data while achieving storage savings of up to 97%. In contrast, PCA with Vector Quantization provided even more robust storage compression, with savings remaining consistently high across different configurations. Despite a slight performance penalty in some cases, VQ's ability to effectively capture and compress the data components stands out as a significant strength.

The investigation also included deep learning-based compression via Variational Autoencoders. While VAEs offer promising opportunities for automated feature extraction, our experimental exploration did not completely identify a configuration of VAE parameters and architecture that could improve performance or match the efficiency achieved by PCA-based methods. In our tests, the latent space representations generated by VAEs did not consistently produce well-separated clusters under aggressive quantization, resulting in higher performance degradation. This may indicate that while deep learning approaches have potential, significant architectural and training refinements are needed before they can serve as valid alternatives in this context.

Beyond performance metrics, our study revealed important trade-offs between data fidelity and storage efficiency. For digital forensic applications, the integrity of the preserved ev-

idence is crucial. Our experiments indicate that with careful tuning, such as selecting an optimal number of PCA components and appropriate quantization levels, it is possible to preserve the necessary detail for reliable forensic analysis without incurring prohibitive storage requirements. The use of Huffman coding as a final coding step further demonstrated that additional gains in storage efficiency can be achieved, particularly in high redundancy settings.

Looking ahead, there are several directions for future work. One key area is to improve the VAE architecture. Exploring alternative network designs, refining loss functions, or adding constraints to the latent space may help to form more distinct and useful clusters, narrowing the performance gap with PCA-based methods. Another direction is to expand the scope of activity recognition. Expanding beyond the five activities studied in this work and investigating the effect of integrating additional antennas or advanced MIMO configurations could improve spatial resolution and recognition performance. In addition, incorporating phase information through linear transformation techniques could reveal new features, while variation of window sizes and sampling rates could provide further knowledge on how to optimize the trade-off between accuracy loss and storage efficiency.

In summary, the research demonstrates that lossy compression techniques particularly PCA combined with quantization offer a viable and effective strategy for managing the vast amounts of CSI data generated in modern IoT systems. These methods not only secure the long-term preservation of forensic evidence but also facilitate efficient real-time human activity recognition.

Bibliography

[1] Transforma Insights. Global iot connections to hit 29.4 billion in 2030, 2025. URL https://transformainsights.com/news/global-iot-connections-294.

[2] Maria Stoyanova, Yannis Nikoloudakis, Spyridon Panagiotakis, Evangelos Pallis, and Evangelos K. Markakis. A survey on the internet of things (iot) forensics: Challenges, approaches, and open issues. *IEEE Communications Surveys & Tutorials*, 22(2): 1191–1221, 2020. doi: 10.1109/COMST.2019.2962586.

[3] Olga Politi, Iosif Mporas, and Vasileios Megalooikonomou. Human motion detection in daily activity tasks using wearable sensors. In *2014 22nd European Signal Processing Conference (EUSIPCO)*, pages 2315–2319, 2014.

[4] Fadel Adib, Zach Kabelac, Dina Katabi, and Robert C. Miller. 3d tracking via body radio reflections. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 317–329, Seattle, WA, April 2014. USENIX Association. ISBN 978-1-931971-09-6. URL https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/adib.

[5] Siamak Yousefi, Hirokazu Narui, Sankalp Dayal, Stefano Ermon, and Shahrokh Valaee. A survey on behavior recognition using wifi channel state information. *IEEE Communications Magazine*, 55(10):98–104, 2017. doi: 10.1109/MCOM.2017.1700082.

[6] Zhengjie Wang, Kangkang Jiang, Yushan Hou, Wenwen Dou, Chengming Zhang, Zehua Huang, and Yinjing Guo. A survey on human behavior recognition using channel state information. *IEEE Access*, 7:155986–156024, 2019. doi: 10.1109/ACCESS.2019.2949123.

[7] Mohan Liyanage, Chii Chang, Satish Srirama, and Seng Loke. Indoor people density sensing using wi-fi and channel state information. *Advances in Modelling and Analysis A*, 61:37–47, 03 2018. doi: 10.18280/ama_b.610107.

[8] Zheng Yang, Yi Zhang, Guoxuan Chi, and Guidong Zhang. Hands-on wireless sensing with wi-fi: A tutorial, 2022. URL https://arxiv.org/abs/2206.09532.

[9] Neal Patwari and Sneha K. Kasera. Robust location distinction using temporal link signatures. In *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, MobiCom '07, page 111–122, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595936813. doi: 10.1145/1287853.1287867. URL https://doi.org/10.1145/1287853.1287867.

[10] Zheng Yang, Zimu Zhou, and Yunhao Liu. From rssi to csi: Indoor localization via channel response. *ACM Comput. Surv.*, 46(2), December 2013. ISSN 0360-0300. doi: 10.1145/2543581.2543592. URL https://doi.org/10.1145/2543581.2543592.

[11] Aqiel Almamori and Seshadri Mohan. Estimation of channel state information for massive mimo based on received data using kalman filter. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 665–669, 2018. doi: 10.1109/CCWC.2018.8301698.

[12] M. H. Essai Ali and I. B. M. Taha. Channel state information estimation for 5g wireless communication systems: recurrent neural networks approach. *PeerJ. Computer Science*, 7:e682, 2021. doi: 10.7717/peerj-cs.682. URL https://doi.org/10.7717/peerj-cs.682.

[13] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Tool release: gathering 802.11n traces with channel state information. *SIGCOMM Comput. Commun. Rev.*, 41(1):53, January 2011. ISSN 0146-4833. doi: 10.1145/1925861.1925870. URL https://doi.org/10.1145/1925861.1925870.

[14] Yaxiong Xie, Zhenjiang Li, and Mo Li. Precise power delay profiling with commodity wifi. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, page 53–64, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3619-2. doi: 10.1145/2789168.2790124. URL http://doi.acm.org/10.1145/2789168.2790124.

[15] Francesco Gringoli, Matthias Schulz, Jakob Link, and Matthias Hollick. Free your csi: A channel state information extraction platform for modern wi-fi chipsets. In *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, WiNTECH '19, page 21–28, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450369312. doi: 10.1145/3349623.3355477. URL https://doi.org/10.1145/3349623.3355477.

[16] Francesco Gringoli, Marco Cominelli, Alejandro Blanco, and Joerg Widmer. Ax-csi: Enabling csi extraction on commercial 802.11ax wi-fi platforms. In *Proceedings of the 15th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation &*

CHaracterization, WiNTECH '21, page 46–53, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450387033. doi: 10.1145/3477086.3480833. URL https://doi.org/10.1145/3477086.3480833.

[17] Mboli Sechang Julius. Mimo: State of the art and the future in focus, 2016. URL https://arxiv.org/abs/1610.05209.

[18] Nickolas LaSorte, W. Barnes, and Hazem Refai. The history of orthogonal frequency division multiplexing. In *Proceedings of the Global Communications Conference*, pages 3592–3596, 01 2008. doi: 10.1109/GLOCOM.2008.ECP.690.

[19] Xuangou Wu, Zhaobin Chu, Panlong Yang, Chaocan Xiang, Xiao Zheng, and Wenchao Huang. Tw-see: Human activity recognition through the wall with commodity wi-fi devices. *IEEE Transactions on Vehicular Technology*, 68(1):306–319, 2019. doi: 10.1109/TVT.2018.2878754.

[20] Fangxin Wang, Wei Gong, and Jiangchuan Liu. On spatial diversity in wifi-based human activity recognition: A deep learning-based approach. *IEEE Internet of Things Journal*, 6(2):2035–2047, 2019. doi: 10.1109/JIOT.2018.2871445.

[21] Han Zou, Yuxun Zhou, Jianfei Yang, and Costas J. Spanos. Towards occupant activity driven smart buildings via wifi-enabled iot devices and deep learning. *Energy and Buildings*, 177:12–22, 2018. ISSN 0378-7788. doi: https://doi.org/10.1016/j.enbuild.2018.08.010. URL https://www.sciencedirect.com/science/article/pii/S037877881831329X.

[22] Marcus Valtonen Örnhag, Stefan Adalbjörnsson, Püren Güler, and Mojtaba Mahdavi. A critical look at recent trends in compression of channel state information. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10095581.

[23] Sean Ferguson, Fabrice Labeau, and Alexander M. Wyglinski. Compression of channel state information for wireless ofdm transceivers. In *2010 IEEE 72nd Vehicular Technology Conference - Fall*, pages 1–5, 2010. doi: 10.1109/VETECF.2010.5594193.

[24] Faris B. Mismar and Aliye Özge Kaya. Adaptive compression of massive mimo channel state information with deep learning. *IEEE Networking Letters*, pages 1–1, 2024. doi: 10.1109/LNET.2024.3475269.

[25] Jianfei Yang, Xinyan Chen, Han Zou, Dazhuo Wang, Qianwen Xu, and Lihua Xie. Efficientfi: Toward large-scale lightweight wifi sensing via csi compression. *IEEE Internet of Things Journal*, 9(15):13086–13095, 2022. doi: 10.1109/JIOT.2021.3139958.

[26] Ian Jolliffe. Principal component analysis. *Encyclopedia of statistics in behavioral science*, 2005.

[27] Wei Wang, Alex X. Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, page 65–76, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336192. doi: 10.1145/2789168.2790093. URL https://doi.org/10.1145/2789168.2790093.

[28] Robert Gallager. Course materials for 6.450 principles of digital communications i. MIT OpenCourseWare, Fall 2006. http://ocw.mit.edu/, Massachusetts Institute of Technology.

[29] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression*. The Springer International Series in Engineering and Computer Science. Springer US, 1991. ISBN 9780792391814. URL https://books.google.it/books?id=DwcDm6xgItUC.

[30] R. Gray. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1984. doi: 10.1109/MASSP.1984.1162229.

[31] David A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952. doi: 10.1109/JRPROC.1952.273898.

[32] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948. doi: https://doi.org/10.1002/j.1538-7305.1948.tb01338.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1948.tb01338.x.

[33] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022. URL https://arxiv.org/abs/1312.6114.

[34] Fabio Palmese and Alessandro E. C. Redondi. Collecting channel state information in wi-fi access points for iot forensics. In *2023 21st Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 176–183, 2023. doi: 10.1109/MedComNet58619.2023.10168865.

[35] Jian Liu, Hongbo Liu, Yingying Chen, Yan Wang, and Chen Wang. Wireless sensing for human activity: A survey. *IEEE Communications Surveys & Tutorials*, 22(3):1629–1645, 2020. doi: 10.1109/COMST.2019.2934489.

[36] Marco Cominelli, Francesco Gringoli, and Francesco Restuccia. Exposing the csi: A

systematic investigation of csi-based wi-fi sensing capabilities and limitations. *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 81–90, 2023. URL `https://api.semanticscholar.org/CorpusID:256503746`.

[37] Marco Cominelli, Francesco Gringoli, Lance Kaplan, Mani Srivastava, and Federico Cerutti. Accurate passive radar via an uncertainty-aware fusion of wi-fi sensing data. In *Accurate Passive Radar via an Uncertainty-Aware Fusion of Wi-Fi Sensing Data*, pages 1–8, 06 2023. doi: 10.23919/FUSION52260.2023.10224098.

List of Figures

2.1 Components of the IoT Forensics 6
2.2 Radio signal propagation in indoor environment 8
2.3 The process of obtaining the Channel Impulse Response 10
2.4 Example of a CSI amplitude capture 14

3.1 CSI-based behavior recognition framework 18

4.1 Components of a variational autoencoder 29
4.2 Sketch of device placement in the room, presence scenario 31
4.3 Comparison of CSI amplitudes before and after the outlier removal process 32
4.4 Comparison of A^* before and after the outlier removal process 33
4.5 Presence detection scenario data analysis pipeline 34
4.6 Environment for the data collection campaign, activity recognition scenario 37
4.7 Overview of the VAE architecture and its interaction with the MLP 39

5.1 Cumulative Variance of PCA's components in presence detection scenario . 44
5.2 Performance results with the feature based method 44
5.3 Performance results with the deep learning based method 45
5.4 Cumulative Variance of PCA's components in activity recognition scenario 46
5.5 Performance results in the activity recognition scenario 46
5.6 Storage analysis with SQ and ENC vs original data 48
5.7 Storage analysis with VQ and ENC vs original data 48
5.8 Analysis of the impact of Huffman coding on SQ data 49
5.9 Analysis of the impact of Huffman coding on VQ data 50

List of Tables

4.1 Architecture of the VAE, presence detection 41
4.2 Architecture of the MLP, presence detection 41
4.3 Architecture of the VAE, activity recognition 42
4.4 Architecture of the MLP, activity recognition 42