Holden Deliverable

D5.1 – Data structure definition for massive RF data input to DL

Grant Agreement number	101099491
Action Acronym	HOLDEN
Action Title	Ehtical Design of Holography with Dense wireless Networks
Type of action	HORIZON-EIC-2022-PaTHFINDEROPEN-01
Version date of the Annex I against which the assessment will be made	13/12/2022
Start date of the Project	1/6/2023
Due date of the delierable	30/11/2024
Actual date of submission	29/11/2024
Lead beneficiary for the deliverable	AALTO
Dissemination level of the deliverable	Public

Action coordinator's scientific representative

Prof. Stephan Sigg
AALTO – KORKEAKOULUSÄÄTIÖ,
Aalto Unviersity School of Electrical Engineering, Department of Information and Communications Engineering
stephan.sigg@aalto.fi

Funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

European Innovation Council

Authors in alphabetical order		
Name	**Beneficiary**	**e-mail**
Dariush Salami	AALTO	dariush.salami@aalto.fi
Ying Liu	AALTO	ying.2.liu@aalto.fi
Stephan Sigg	AALTO	stephan.sigg@aalto.fi

Change history				
Version	**Date**	**Status**	**Partner**	**Description**
1.0	15.11.2024	Final	Aalto	First final draft
1.1	22.11.2024	Final	Aalto	Second final draft
1.2	22.11.2024	Final	Aalto	Third final draft

Abstract

This deliverable explores the use of point cloud representations in Radio Frequency (RF) sensing technologies and their suitability for processing massive RF data with deep learning models. We explore the integration of point cloud data from millimeter-wave radar, WiFi, and Radio-Frequency IDentification (RFID) systems for applications such as gesture recognition, addressing the privacy risks inherent in these technologies. Our research introduces a novel framework that employs graph-based autoencoders with Message Passing Neural Network (MPNN) and multi-head self-attention mechanisms to anonymize user identity while maintaining high accuracy in gesture recognition tasks. By transforming point cloud data into graph structures that preserve essential gesture attributes but obscure personal identifiers, our approach significantly mitigates the risk of unauthorized identification. Extensive experimental results confirm the efficiency of our method, demonstrating reduced identification accuracy while retaining strong gesture recognition performance. Additionally, we discuss the benefits of point cloud representations, including their reduced memory and computational requirements compared to other data formats, making them ideal for handling large-scale RF data. We also outline training and validation strategies, such as cross-validation, performance metrics, and specialized loss functions like Chamfer Distance, Earth Mover's Distance (EMD), and Hausdorff Distance, to enhance model robustness and generalization. This deliverable aims to provide a comprehensive framework for developing ethical and efficient RF sensing systems, ensuring their secure and effective deployment in diverse real-world applications.

Contents

1 Introduction

Radio Frequency (RF) data can be represented in various forms, each with its own set of characteristics, use cases, and limitations. These forms include various modalities such as Wi-Fi, different types of radars, Radio-Frequency IDentification (RFID) tags, etc. Each modality provides unique data representations; for example, Wi-Fi can offer signal strength and phase information, radars can provide range-Doppler maps and point clouds, while RFID tags can supply proximity and motion data. Understanding these representations is essential for selecting the appropriate data structure for deep learning applications. The choice of data representation directly impacts the effectiveness of deep learning models, influencing their ability to learn relevant features and make accurate predictions. By leveraging the strengths of each modality, we can optimize the performance and accuracy of RF-based deep learning models. Selecting the right representation allows for better feature extraction, reduces computational complexity, and enhances the model's capability to generalize across different scenarios and environments.

1.1 RF sensing data modalities

RF sensing encompasses a variety of techniques that utilize radio waves to detect, measure, and analyze physical properties and activities in the environment. These techniques are becoming increasingly important in applications ranging from smart home automation to advanced medical diagnostics and autonomous vehicles. The diversity RF RF sensing modalities allows for a wide array of data representations, each providing unique insights and capabilities. In this section, we will explore the different types of RF sensing modalities, including Wi-Fi, radar, and RFID tags, highlighting their distinct characteristics, data forms, and practical applications.

1.1.1 Wi-Fi Sensing

Wi-Fi sensing leverages existing Wi-Fi infrastructure to perform various sensing tasks such as localization, gesture recognition, and activity detection. The data is typically derived from the analysis of Channel State Information (CSI) and Received Signal Strength Indicator (RSSI). CSI provides detailed information about the signal propagation environment, including amplitude and phase information across multiple subcarriers. This fine-grained data allows for precise sensing capabilities, making Wi-Fi a versatile and widely used modality in RF sensing applications.

Data representation granularity in Wi-Fi sensing is of high importance and it depends on the specific application. It is demonstrated in [5, 9] that the choice of data representation, and the level of detail it captures, directly impacts the system's ability to extract meaningful information and achieve the desired level of accuracy [1, 9]. For instance, applications like gesture recognition, which aim to capture the temporal evolution of hand movements, may benefit from a relatively coarse representation. Using features like sparse point clouds reduces data size and allows for efficient processing, especially on resource-constrained devices [23]. Conversely, activity recognition applications might necessitate a finer-grained representation using features like Doppler or phase information, which are more sensitive to subtle body movements [9].

1.1.2 Radar Sensing

Radar sensing utilizes radio waves to detect and measure various characteristics of objects within its range, including distance, speed, and movement. Different types of radars, such as Frequency Modulated Continuous Wave (FMCW) radar, pulse radar, and Doppler radar, offer varying capabilities. Radar

data is often represented in forms such as range-Doppler maps, range profiles, and point clouds. These representations provide detailed spatial and velocity information, which are invaluable for applications like gesture recognition, vital sign monitoring, and automotive sensing.

As an example, [12, 16, 18] examine mmWave radar for diverse applications, highlighting the crucial role of data representation in achieving optimal performance. While all of them utilize mmWave radar data, the specific representation and granularity are dictated by the target application. For gesture recognition [12, 16], the raw Analog to Digital Conversion (ADC) data is transformed into point clouds, which significantly reduces data size and facilitates efficient processing. This representation proves advantageous due to the inherent sparsity of mmWave radar point clouds, making direct processing methods like the proposed architectures highly effective. In contrast, for water quality analysis [18], the focus shifts to directly processing the In-phase and Quadrature (IQ) signals. This approach, utilizing convolutional neural networks, achieves remarkable accuracy in detecting contaminant concentrations and identifying water types.

The granularity of data representation is directly influenced by the specific demands of each application. Gesture recognition, focusing on capturing the temporal evolution of hand movements, benefits from a relatively coarse representation using sparse point clouds. This allows for computationally efficient processing on resource-constrained devices like Raspberry Pi. Conversely, water quality analysis requires a finer-grained representation, utilizing the raw IQ signals to discern subtle variations in signal characteristics induced by different contaminants and water types. This highlights how the choice of data representation granularity directly impacts the system's ability to extract meaningful information and achieve the desired level of accuracy.

1.1.3 RFID Sensing

RFID sensing employs RFID tags and readers to identify and track objects wirelessly. RFID systems can be passive, active, or semi-passive, each offering different ranges and data capabilities. The data from RFID sensing includes tag identity, proximity, and movement information. This modality is particularly useful in applications such as inventory management, asset tracking, and human-computer interaction. RFID tags can provide continuous and real-time data, which is essential for dynamic and responsive systems.

For complex and fine-grained gesture recognition using RFID [3], the challenge lies in capturing subtle finger movements and differentiating between similar gestures. To address this, a multi-tag array is used to enhance sensing ability and capture comprehensive spatial-temporal information. Raw phase and Received Signal Strength (RSS) data streams are pre-processed to remove static interferences and augment data diversity. This processed data is then fed into a Multimodal Convolutional Neural Network (MCNN), enabling the extraction of high-level representations that encode both intra-modal and cross-modal interactions. Adversarial learning is further employed to extract domain-independent and gesture-discriminative features, achieving robust recognition across different users and environments. In contrast, for human gait analysis [7], the focus shifts towards extracting unique patterns from the RF signal variations caused by an individual's gait. Multivariate Variational Mode Decomposition (MVMD) is employed to decompose the received RF signal and isolate the subtle gait-related components from the background noise and static object interference. The system then extracts time-domain and frequency-domain features from the decomposed signal and utilizes a Support Vector Machine (SVM) for human identification. This highlights how the selection of specific data representations and feature extraction techniques are tailored to the unique characteristics and challenges of each application.

Having explored the various RF sensing data modalities, it is crucial to understand how the data from these different modalities are represented and processed. Each modality, with its unique characteristics, generates distinct types of data that must be appropriately structured for effective analysis and applica-

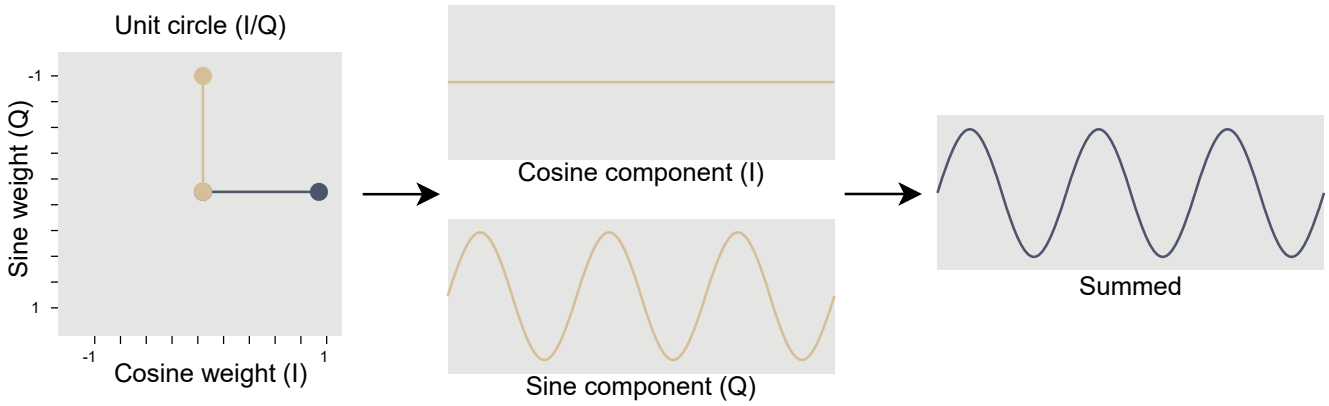


Figure 1: A visualization of the logic behind IQ sample representation. The leftmost figure illustrates the contribution of the I and Q components, with the light brown line representing the weight of each component, while the black line shows their combined sum, always equaling 1. The middle figure depicts the sine and cosine components, each multiplied by their respective weights. Finally, the rightmost figure shows the summation of these two weighted components, resulting in the final signal.

tion. In the next section, we will provide an overview of the common RF data representations, delving into the specific formats and structures used to encapsulate the rich information captured by RF sensing technologies. This understanding will lay the groundwork for effectively utilizing RF data in deep learning models.

1.2 Overview of RF Data Representations

In RF sensing, the way data is represented plays a pivotal role in the effectiveness of subsequent data processing and analysis. The representation of RF data can vary significantly depending on the modality, such as Wi-Fi, radar, or RFID, and each has its own unique set of characteristics and applications. This section provides an overview of the common RF data representations, examining the specific formats and structures used to encapsulate the diverse information captured by different RF sensing technologies.

1.2.1 IQ Samples

IQ samples are a fundamental representation of RF signals, capturing both the amplitude and phase information of the signal. This representation is critical in many RF sensing applications due to its ability to preserve the complete waveform information of the RF signal. IQ sampling involves splitting an RF signal into two components: the in-phase (I) component and the quadrature (Q) component. These components are obtained by mixing the RF signal with a reference signal that is phase-shifted by $90°$, producing two orthogonal signals. The I component corresponds to the cosine of the phase angle, while the Q component corresponds to the sine of the phase angle. Together, they provide a comprehensive representation of the signal's amplitude and phase.

Fig.1 illustrates the process of IQ sample representation, providing a step-by-step breakdown of how the I and Q components combine to form the final signal. In the leftmost figure, the light brown line represents the individual weight of each of the I and Q components, while the black line shows their sum, which is always equal to 1, highlighting the balance between the two components. The middle figure shows the sine and cosine components, which are multiplied by their respective weights, reflecting the influence of each component's magnitude. Finally, the rightmost figure demonstrates the summation of these two weighted components, resulting in the final signal, which captures both the amplitude and phase information necessary for further processing and analysis in RF sensing applications.

Characteristics IQ samples capture both the magnitude and phase of the RF signal, which allows for detailed analysis of the signal's properties. This combination of amplitude and phase information is critical for many RF sensing applications, as it provides a comprehensive view of the signal's behavior. Plus, they are often represented as complex numbers, where the real part corresponds to the in-phase (I) component and the imaginary part corresponds to the quadrature (Q) component. This complex number representation enables efficient processing and storage of the RF signal data.

Moreover, IQ sampling can provide high-resolution data, which is essential for applications requiring precise measurements and fine-grained signal analysis. The ability to capture detailed signal information makes IQ sampling suitable for advanced RF sensing techniques, such as radar systems and communication technologies. However, these samples are typically in a raw format, meaning that substantial processing is required to extract meaningful features. This raw data format can pose challenges in terms of storage and computation, especially for real-time applications, but it offers the flexibility needed for detailed analysis and feature extraction.

Common Use Cases in RF Sensing IQ samples are widely used in various RF sensing applications. In wireless communication systems, RF samples play a crucial role in modulating and demodulating signals, which enables efficient data transmission and reception. In radar systems, RF samples are essential for generating range-Doppler maps, which help detect and track objects by analyzing their distance and velocity. Additionally, RF samples are used in spectrum analysis, allowing for the identification of signal characteristics such as frequency, bandwidth, and modulation type. Finally, advanced signal processing techniques, such as beamforming and direction finding, rely on RF samples to accurately determine the direction and strength of a signal, further enhancing the capabilities of RF sensing systems.

For instance, the authors in [26] capture the transient response of the IQ signal from the reader, using it as an indicator of the material's condition. They argue that this approach offers higher sensitivity and robustness compared to traditional power-based RFID sensing methods, which typically rely on measurements such as RSSI. Their study demonstrates that features extracted from the transient response of the IQ signal, such as skewness, can effectively characterize corrosion progression and detect cracks in metallic samples. In the context of Wi-Fi sensing, [8] utilizes IQ data obtained from a Wi-Fi signal analyzer to perform human activity recognition. The authors propose a method that transforms the raw IQ data into time-frequency images using the Choi-Williams distribution and the Margenau-Hill spectrogram. Offset parameters and Principal Component Analysis (PCA) features are then extracted from these images and fed into a random forest classifier to identify various human activities. This work highlights the potential of leveraging the rich information embedded in IQ data to create discriminative features for activity recognition.

Furthermore, [18] uses IQ data captured from an mmWave FMCW radar to assess water quality. The authors develop a complex-valued 2D Convolutional Neural Network (CNN) that directly processes the raw IQ signals to detect varying concentrations of contaminants and classify different water types. They compare this approach with other methods that rely on Fast Fourier Transform (FFT)-based feature extraction and demonstrate that end-to-end processing of the raw IQ signals achieves superior accuracy. The authors attribute this superior performance to the CNN's ability to learn intricate patterns and subtle signal variations directly from the granular IQ data, which are critical for accurate water quality assessment.

Limitations Despite their advantages, IQ samples have several limitations. One significant challenge is the high data volume generated by IQ sampling, which can be difficult to store, transmit, and process, particularly in real-time applications. Additionally, extracting meaningful information from IQ samples

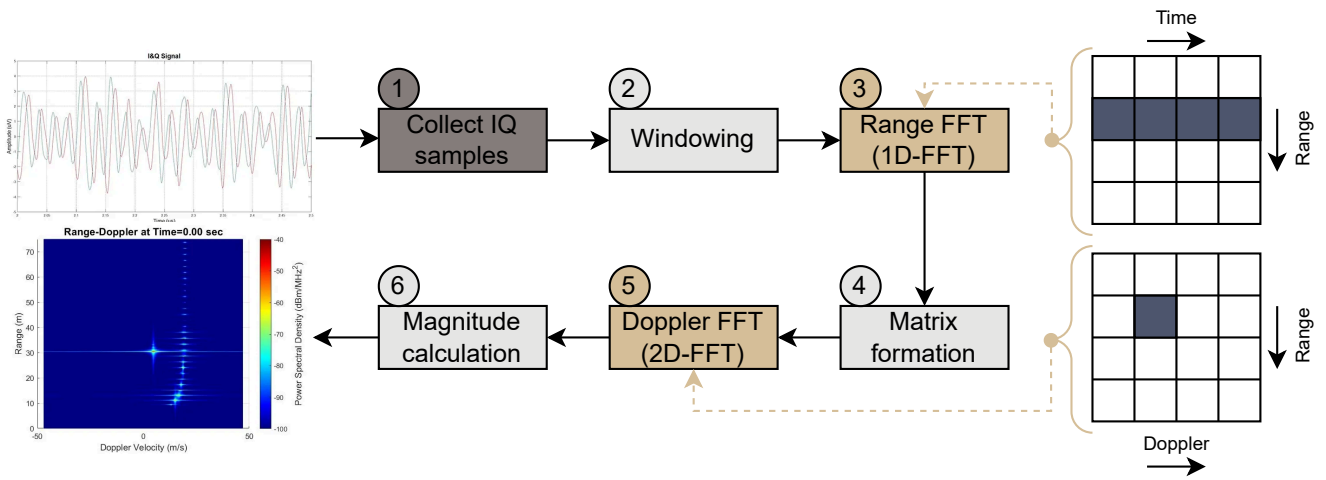


Figure 2: This figure illustrates the generation of Range-Doppler maps from IQ samples using FFT operations in two dimensions. The block shown in dark brown represents the RF sensing medium, the light boxes correspond to pre-processing steps, and the brown boxes represent the FFT operations.

requires considerable computational resources and sophisticated signal processing algorithms, making the process complex. Furthermore, IQ samples can be sensitive to noise and interference, which can negatively impact the accuracy and reliability of the signal analysis. Finally, accurate IQ sampling often necessitates careful calibration of the RF hardware to ensure that the I and Q components are properly aligned and free from distortions. Since IQ samples can capture detailed information about the signal, including potentially identifiable features of the user, they may not be privacy-preserving. For instance, in an activity recognition task, it may be unnecessary to know the user's identity, but such information might be inadvertently embedded in the IQ representation, posing privacy risks.

1.2.2 Range-Doppler Images

Range-Doppler images are a crucial representation in RF sensing, particularly for radar systems. These images map the range and velocity of objects, providing a two-dimensional view of the detected targets. The range dimension corresponds to the distance between the radar and the target, while the Doppler dimension represents the relative velocity of the target. By processing the reflected RF signals, radar systems can generate Range-Doppler images that reveal the movement and position of objects within the radar's field of view.

As shown in Fig.2, the following steps are required to generate Range-Doppler maps:

1. **Collect IQ Samples**: Gather the IQ samples from the RF sensing system. These samples represent the received signal after mixing the incoming echoes with a reference signal.

2. **Windowing**: Apply a window function (e.g., Hamming, Hann) to the IQ samples to reduce spectral leakage. This step helps in managing the trade-off between main-lobe width and side-lobe levels.

3. **Range FFT (1D-FFT)**: Perform an FFT along the range dimension. This transforms the time-domain IQ samples into the frequency domain, where each frequency bin corresponds to a specific range bin. This step results in a range profile for each radar pulse.

4. **Matrix formation**: Arrange the range profiles into a two-dimensional matrix, where each row represents a range profile for a given pulse, and each column represents a different range bin.

5. **Doppler FFT (2D-FFT)**: Perform an FFT along the Doppler dimension (i.e., along the columns of the matrix). This step converts the range profiles into a Range-Doppler map, where each element represents a specific range and Doppler shift (velocity).
6. **Magnitude Calculation**: Calculate the magnitude of the complex-valued elements in the Range-Doppler map to obtain the intensity values, which represent the power of the reflected signals.

Characteristics Range-Doppler images offer detailed spatial and velocity information, making them invaluable for applications that require precise tracking and detection of moving objects. They are typically represented as two-dimensional matrices where each element corresponds to a specific range and Doppler shift, allowing for fine-grained analysis of target movements. The high resolution of Range-Doppler images enables the detection of small and slow-moving objects, which is essential for applications such as automotive radar, surveillance, and remote sensing. However, generating and interpreting Range-Doppler images requires significant computational resources and advanced signal processing techniques.

Common Use Cases in RF Sensing Range-Doppler images are widely used in various RF sensing applications. In radar systems, they are employed to create range-Doppler maps, which help detect and track objects by analyzing their distance and velocity. This capability is particularly useful in automotive radar for collision avoidance and autonomous driving, where precise object detection and tracking are critical. Additionally, Range-Doppler images are used in surveillance systems to monitor and track the movement of individuals and vehicles. In remote sensing, they facilitate the analysis of the Earth's surface and atmosphere, enabling applications such as weather monitoring and environmental studies.

For instance, researchers propose a human activity recognition system using mmWwave radar, utilizing range-Doppler images to classify activities such as boxing, jumping, and squatting [6]. The range-Doppler data is first denoised using the Cell Average Constant False-Alarm Rate (CA-CFAR) algorithm and then fed into a 3D-CNN to obtain features. These features are then fused with features extracted from the point cloud data using a 3D-CNN and Long Short-Term Memory (LSTM) network to create a comprehensive activity feature, which is then used for classification. Moreover, another group of researchers focuse on hand gesture recognition using a mmWwave radar and explore the benefits of utilizing range-angle images in addition to range-Doppler images. Range-Doppler images, generated using 2D-FFT, capture the range and velocity information of a moving object. However, they are less effective at capturing horizontal movements. Range-angle images, created using 3D-FFT applied to multiple range-Doppler images outputs from different receive antennas, provide information about the object's range and angle. This additional angle information allows for better differentiation between gestures involving horizontal movement, leading to improved accuracy. The authors demonstrate that using range-Angle images as input to a CNN-LSTM model results in better performance than using range-Doppler images alone, especially for gestures with horizontal components.

In the WiFi domain, a team of researchers [2] introduce a WiFi-based passive Inverse Synthetic Aperture Radar (ISAR) system for generating high-resolution cross-range profiles of moving vehicles. The system uses a WiFi access point as the illuminator of opportunity and employs a multi-stage processing scheme that includes clutter cancellation, range compression, and 2D-Cross-Correlation Function (CCF) evaluation to detect moving targets. Once a target is detected and tracked, its range-compressed data is used for ISAR processing. This involves estimating the target's motion parameters to compensate for range and Doppler cell migration and form the cross-range profile. The system is shown to be effective for both mono-static and bi-static observation geometries. Moreover, another group of researchers utilize WiFi routers to analyze the variations in CSI to distinguish between different gestures using range-Doppler images.

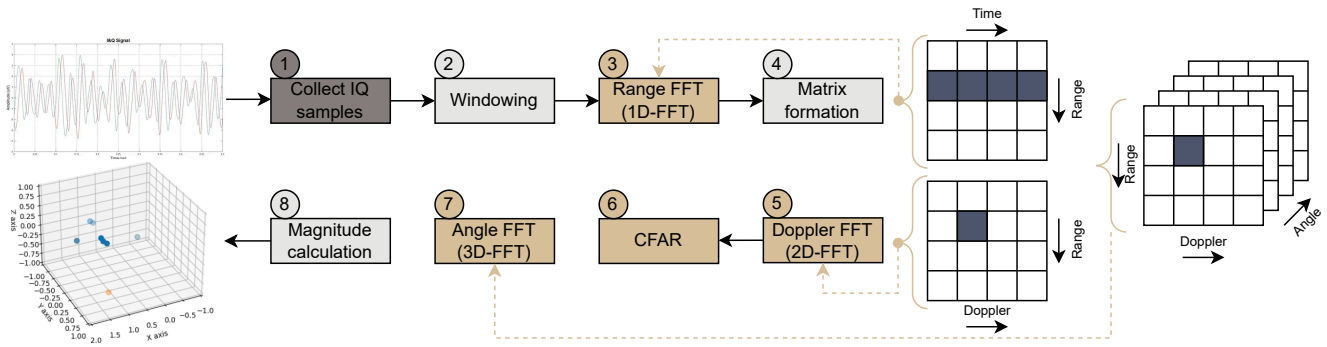


Figure 3: This figure illustrates the generation of point clouds from IQ samples using FFT operations in three dimensions. The block shown in dark brown represents the RF sensing medium, the light boxes correspond to pre-processing steps, and the brown boxes represent the FFT operations.

Limitations Despite their advantages, Range-Doppler images have several limitations. One major challenge is the high computational cost associated with generating and processing these images. The need for real-time analysis in many applications, such as automotive radar, exacerbates this issue, requiring powerful hardware and optimized algorithms. Additionally, Range-Doppler images can be sensitive to noise and interference, which can degrade the quality of the images and affect the accuracy of the analysis. Another limitation is the potential for privacy concerns, as these images can reveal detailed information about the movement and behavior of individuals. Ensuring the privacy and security of the data captured by radar systems is crucial, especially in applications involving human subjects.

1.2.3 Point Clouds

Point clouds are a fundamental representation in RF sensing, particularly for applications involving multi-dimensional imaging and spatial analysis. They consist of a collection of data points defined in a coordinate system, where each point represents a sample of the sensed environment. Each point in the cloud has coordinates that describe various dimensions, which can include spatial coordinates (such as x, y, z), intensity, color, or other attributes relevant to the specific application.

As shown in Fig.3, the following steps are required to generate point cloud images:

1. **Collect IQ Samples**: Gather the IQ samples from the RF sensing system. These samples represent the received signal after mixing the incoming echoes with a reference signal.
2. **Windowing**: Apply a window function (e.g., Hamming, Hann) to the IQ samples to reduce spectral leakage. This step helps in managing the trade-off between main-lobe width and side-lobe levels.
3. **Range FFT**: Perform a FFT along the range dimension. This transforms the time-domain IQ samples into the frequency domain, where each frequency bin corresponds to a specific range bin. This step results in a range profile for each pulse.
4. **Matrix formation**: Arrange the range profiles into a two-dimensional matrix, where each row represents a range profile for a given pulse, and each column represents a different range bin.
5. **Doppler FFT**: Perform a FFT along the Doppler dimension (i.e., along the columns of the matrix). This step converts the range profiles into a Range-Doppler map, where each element represents a specific range and Doppler shift (velocity).

6. **cfar**: Apply Constant Constant False Alarm Rate (CFAR) detection to the Range-Doppler map to identify potential targets by distinguishing signal reflections from noise.

7. **Angle FFT**: Perform a FFT along the angle dimension using data from multiple receive antennas. This step generates angle profiles that help in determining the direction of the detected targets.

8. **Magnitude calculation**: Calculate the magnitude of the complex-valued elements in the Range-Doppler map and angle profiles to obtain the intensity values, which represent the power of the reflected signals.

Characteristics Point clouds provide a highly detailed and accurate 3D representation of the sensed environment. They are valuable for applications that require precise spatial analysis and object recognition. Each point in the cloud captures specific information about the location and attributes of the surface it represents, allowing for a comprehensive understanding of the scene. The resolution and accuracy of point clouds depend on the quality of the sensing system and the effectiveness of the signal processing techniques used.

Common Use Cases in RF Sensing Point clouds are widely used in various RF sensing applications. In automotive radar systems, they enable Advanced Driver Assistance Systems (ADASs) and autonomous driving by providing detailed 3D maps of the surroundings, helping in obstacle detection and path planning. In surveillance and security, point clouds enhance monitoring capabilities by offering precise 3D information about intruders or objects in restricted areas. Additionally, in the field of robotics, point clouds facilitate navigation and manipulation by providing robots with accurate spatial information about their environment.

For instance, researchers have utilized point clouds for human activity recognition using mmWave radar [6]. By combining point cloud data with range-Doppler images, they were able to achieve high accuracy in classifying various activities such as walking, running, and sitting. The integration of point clouds provided additional spatial context, enhancing the overall recognition performance. Similarly, in the context of gesture recognition, point clouds generated from RF sensing data have been used to identify the hand movement of people which is valuable for human-robot interactions [12, 16]. Moreover, in the context of Wi-Fi, a group of researchers [10] have focused on utilizing Wi-Fi signals for tasks such as gesture recognition, human motion analysis, and vehicle tracking. They process CSI data collected from Wi-Fi signals using a Transformer encoder, which employs a multi-head attention mechanism to produce point clouds. These point clouds are then used to perform the intended tasks, enabling a detailed spatial representation of the environment. The approach leverages the spatial and temporal diversity of Wi-Fi signals to achieve accurate and robust performance in various applications.

Limitations While point clouds offer numerous advantages, they also come with certain limitations. Generating high-quality point clouds requires sophisticated hardware and advanced signal processing algorithms, which can be computationally intensive and costly. Additionally, the accuracy of point clouds can be affected by factors such as noise, interference, and environmental conditions. Ensuring robust and reliable data collection and processing is crucial to overcoming these challenges. Moreover, the large volume of data generated by point clouds necessitates efficient data storage and management solutions to handle and analyze the information effectively. Despite these limitations, point cloud representation remain a powerful tool in RF sensing, providing invaluable insights and capabilities across a wide range of applications.

(a) Original (b) Permuted (c) Translated (d) Rotated (e) scaled

Figure 4: Different features of point clouds which affect the way they need to be processed are shown in this figure. To simplify visualization, the text "HOLDEN" is shown in the form of a point cloud in a 2D space.

2 Point Cloud Representation

As we described in Section 1, point clouds are invaluable in a variety of applications due to their detailed and flexible representation of spatial data. In this section, we dive deeper into the nature of point clouds, exploring their structure and the reasons behind their suitability for deep learning algorithms. The richness of point clouds, with their ability to capture fine-grained spatial information and additional attributes like intensity and color, makes them particularly well-suited for tasks that require precise 3D modeling and analysis. By leveraging deep learning techniques, we can effectively process and interpret point clouds, enabling advanced applications such as object recognition, gesture detection, and environmental mapping.

2.1 Mathematical Definition of Point Clouds

Mathematically, a point cloud P can be defined as a set of points $\{p_i\}$ D-dimensional space, where each point p_i is represented as a vector:

$$P = \{p_i \in R^D | i = 1, 2, \ldots, N\}. \tag{1}$$

Here, N denotes the total number of points in the point cloud, and D represents the dimensionality of the space. Each point p_i can be described as:

$$p_i = (x_{i1}, x_{i2}, ..., x_{iD}). \tag{2}$$

In most common applications, $D = 3$, and each point p_i is characterized by three spatial coordinates (x_i, y_i, z_i). However, depending on the application, D can be higher, incorporating additional attributes such as intensity, color, time, or frequency.

2.2 Permutation and Geometric Transformation Invariance in Point Clouds

Point clouds, as a representation of 3D spatial data, exhibit certain unique properties that make them both powerful and challenging to work with in the context of deep learning and other computational techniques. Two critical aspects of point clouds that significantly influence their processing and analysis are permutation invariance and geometric transformation invariance.

2.2.1 Permutation Invariance

Permutation invariance refers to the property that the ordering of points in a point cloud does not affect the overall representation of the data. This is shown in Fig.4b. Although the set of points shown in two different colors are swapped with each other, the underlying semantic meaning of the shape has not changed. Mathematically, any permutation of a set of point cloud $P' = \{p_{\sigma(1)}, p_{\sigma(2)}, ..., p_{\sigma(n)}\}$ (where σ is a permutation function) should represent the same point cloud. This property is crucial because,

unlike grid-based representations like images, there is no inherent order in the points of a point cloud. Deep learning models designed to process point clouds must, therefore, be capable of handling this permutation invariance to ensure that the features extracted are consistent regardless of the order in which points are presented.

2.2.2 Geometric Transformation Invariance

Geometric transformation invariance pertains to the ability of a point cloud representation to remain consistent under various geometric transformations, such as translation, rotation, and scaling. For practical applications, it is often necessary that the point cloud representation remains unchanged (or appropriately transformed) when subjected to these operations. This invariance is crucial for ensuring that the models trained on point clouds are robust and can generalize well to real-world scenarios where objects and environments may appear in different orientations and scales.

1. **Translation Invariance:** A point cloud should retain its structural information when all points are translated by the same vector as shown in Fig.4c. Mathematically, if T is a translation vector, a point cloud transformed by T is $P' = \{p_1 + T, p_2 + T, ..., p_n + T\}$. Effective models must account for or neutralize the effects of translation to focus on the intrinsic geometry of the point cloud.

2. **Rotation Invariance:** Similarly, point clouds should remain consistent when rotated as shown in Fig.4d. A rotation matrix R applied to a point cloud P results in $P' = \{Rp_1, Rp_2, ..., Rp_n\}$. Models must either learn features that are inherently rotation-invariant or use data augmentation techniques to train the model under various rotations to achieve robustness.

3. **Scaling Invariance:** Scaling invariance requires that the point cloud representation should be unaffected by uniform scaling transformations as shown in Fig.4e. If S is a scaling factor, a scaled point cloud $P' = \{Sp_1, Sp_2, ..., Sp_n\}$ should ideally be equivalent to the original P in the context of the task at hand. Achieving scaling invariance can be more challenging but is often addressed through normalization techniques or scale-invariant feature extraction methods.

2.3 Memory Requirements Comparison

In RF sensing systems, different data representations are utilized to capture and process information from the environment as discussed in Section1. These representations include IQ samples, Range-Doppler maps, and point clouds, each with distinct memory requirements due to their inherent structural differences.

IQ samples are stored in a 4D dense matrix encompassing range, Doppler, angle, and time dimensions. Each sample comprises complex numbers representing the in-phase (I) and quadrature (Q) components. Consequently, the memory requirement for IQ samples is substantial, as it scales with the number of range bins (N_r), Doppler bins (N_d), angle bins (N_θ), and time steps (N_t). Mathematically, the memory needed is given by $4N_r \times N_d \times N_\theta \times N_t$ bytes, indicating the storage of large volumes of data, especially in high-resolution systems.

Range-Doppler maps, on the other hand, reduce the dimensionality by focusing on range and Doppler information at each time step, forming a 2D array for every time instance. Although this reduces the overall memory requirement compared to IQ samples, the total memory still depends on the number of range bins, Doppler bins, and time steps. The memory required for Range-Doppler maps is $4 \times N_r \times N_d \times N_t$ bytes. When considering Range-Doppler-angle cubes, the memory requirement remains comparable to IQ samples due to the addition of the angle dimension, resulting in $4 \times N_r \times N_d \times N_\theta \times N_t$ bytes.

Point clouds, however, offer a sparse representation, significantly reducing memory requirements. After clutter removal and CFAR processing, only a few points per radar frame remain, each representing spatial coordinates and possibly additional attributes. The memory needed for point clouds is given by $12 \times N_p$ bytes, where N_p is the number of points per radar frame. This sparse nature of point clouds leads to a dramatic reduction in memory usage compared to the dense representations of IQ samples and Range-Doppler maps. Typically, N_p is much smaller than the product of the range, Doppler, angle, and time dimensions, making point clouds a highly efficient representation for many applications.

In summary, the different representations of RF sensing data—IQ samples, Range-Doppler maps, and point clouds—vary significantly in memory requirements and intrinsic features. IQ samples and Range-Doppler maps demand substantial storage, especially for high-resolution systems. In contrast, point clouds provide a sparse and memory-efficient alternative, maintaining essential spatial information with far less storage. Additionally, point clouds are permutation and geometric transformation invariant, making them robust and suitable for deep learning applications. This efficiency and robustness, combined with lower memory demands, highlight the advantages of point clouds in advanced RF sensing systems.

3 Integration with Deep Learning Models

Integrating point cloud data with deep learning models involves several key steps to ensure effective processing and analysis. This section explores the essential aspects of preparing point cloud data for deep learning, including storage formats and standards, data augmentation, feature extraction, and the selection of suitable models. By understanding these components, we can harness the full potential of deep learning techniques to analyze and interpret complex 3D point cloud data, enabling a wide range of applications from object recognition to spatial analysis.

3.1 Storage Formats and Standards

Point cloud data can be stored in various formats, each with unique characteristics tailored to specific applications. The three most commonly used formats are Point Cloud Data (PCD) [15], PLY [21], and LASer (LAS) [4], each designed to handle 3D point cloud data but differing in structure, data organization, and target applications. All three formats are designed to store 3D point cloud data, typically including spatial coordinates (x, y, z) for each point and potentially additional attributes such as color, intensity, and normal vectors. They all offer flexibility, allowing users to define custom data types and structures to accommodate various applications and data sources.

Despite these similarities, the formats differ significantly. The PLY format includes a header section that defines the elements and properties present in the data, followed by lists of elements such as vertices, faces, and edges, each with corresponding properties. In contrast, the PCD format features a header section specifying the file version, data type, dimensions, and other metadata, followed by the point cloud data stored in American Standard Code for Information Interchange (ASCII), binary, or binary compressed formats. LAS files are organized into three main parts: a header containing metadata, Variable Length Records (VLRs) for additional information such as spatial reference systems, and point records that hold the actual point cloud data. Different point formats within LAS define the specific dimensions stored for each point.

Regarding data organization, PCD files can store both organized and unorganized point cloud data. Organized datasets resemble an image-like structure with rows and columns, which is beneficial for efficient nearest neighbor operations. The supported data types also vary among these formats. PLY supports various scalar and list data types, including char, uchar, short, ushort, int, uint, float, and

double. PCD supports common data types such as char, short, int, float, and double, and introduces the ”COUNT” field to enable the storage of n-D histogram descriptors at each point. LAS primarily uses signed and unsigned integers and floating-point numbers for storing point data, with the specific data types depending on the chosen point format, and later versions offering more diverse options.

The target applications for each format also differ. PLY is widely used in computer graphics and for exchanging graphical objects between different programs. PCD, being the native format of the Point Cloud Library (PCL) [15], caters to various applications in 3D perception, robotics, and computer vision, focusing on efficient handling of large point clouds and features like organized datasets and n-D histogram descriptors. LAS is primarily used for exchanging lidar point cloud data, particularly in geospatial applications.

For RF data, the PCD format is generally the most suitable choice. This is because PCD is specifically designed for 3D perception applications and is the native format of the PCL, which is widely used in robotics and computer vision. PCD supports efficient handling of large point clouds and offers flexibility in terms of both organized and unorganized datasets, making it ideal for RF data where the spatial organization and density of points can vary significantly. Furthermore, PCD's support for n-D histogram descriptors allows for storing additional features, such as intensity or Doppler shift, which are crucial for RF sensing applications. While LAS is optimized for lidar data and PLY is better suited for graphics and visualization, PCD's structure and capabilities align well with the needs of RF data processing, making it the preferred format in this context.

3.2 Preparing Point Cloud Data for Deep Learning

Preparing point cloud data for deep learning involves several essential steps to ensure the data is in a suitable format for training and inference. This process typically includes data augmentation to enhance the diversity of training samples, as well as feature extraction to highlight relevant patterns in the point cloud data. These steps help improve model performance and generalization. In the following sections, we will explore key techniques for preparing point cloud data and how they contribute to more effective deep learning applications.

3.2.1 Data Augmentation

Data augmentation is a critical step in preparing point cloud data for deep learning models [12–14,16,20]. It involves generating additional training samples by applying various transformations to the original data, such as rotation, scaling, translation, and adding noise. These augmentations help improve the robustness of the model by exposing it to a wider variety of scenarios, thus preventing overfitting and enhancing its generalization capabilities.

Random Translation Random translation involves shifting the entire point cloud by a random value within a specified range. This technique helps the model become invariant to small translations and better generalize to scenarios where the object or scene of interest may be located at different positions in space. By shifting the point cloud, the model learns to focus on the relative positioning of points rather than their absolute locations, making it more robust to small shifts that might occur in real-world sensing environments.

Random Scaling Random scaling applies a scaling factor to the entire point cloud, which alters the size of the object or scene represented. The scaling factor is chosen randomly based on the application, which means that the point cloud can shrink or expand by a factor. This technique mimics scenarios

Table 1: Effect of data augmentation.

	Data augmentation			
Training set	✗	✗	✓	✓
Test set	✗	✓	✓	✗
Acc. (%)	93.89	31.20	94.4	95.01

where the object might be perceived from different distances or perspectives, such as a sensor moving closer or farther away from the scene. Scaling ensures that the model does not overfit to a specific object size and can generalize better across varying scales encountered during deployment.

Random Point-Wise Translation (Jitter) Based on a Gaussian Distribution Random point-wise translation, or jitter, adds small random noise to each point in the cloud according to a Gaussian distribution. This technique introduces slight random displacements to each point. Jittering helps simulate the noise or inaccuracies that may arise in real-world sensors, such as imperfections in the measurement process or environmental disturbances. This augmentation ensures that the deep learning model becomes more resilient to small errors in the data, improving its robustness to noise in real-world applications.

Random Clipping Random clipping is a technique that removes points that are outside a specified range. This method is useful in training models that must focus on the central region of interest, eliminating any redundant or irrelevant points that may exist on the boundaries. Clipping helps improve computational efficiency and can prevent the model from learning unnecessary features, thus improving the focus on key spatial structures within the point cloud.

Random Permutation of Points Random shuffling rearranges the order of points in the point cloud while maintaining their spatial relationships. The key to this technique is that the relative positioning of points is preserved, which is crucial for applications that rely on the spatial arrangement of data, such as object detection or classification. Temporal features, if present (such as time-varying point clouds from moving sensors), should also be preserved. Shuffling helps to reduce overfitting by ensuring the model does not learn to rely on the sequential order of points. It increases the robustness of deep learning models to variations in how the point cloud data might be captured or processed.

By incorporating these data augmentation techniques, point cloud data becomes more diverse and comprehensive, allowing deep learning models to perform better across different conditions and enhance their generalization capabilities. To give an example, in Pantomime [12], point clouds generated from mmWave FMCW radars were used for gesture recognition, with data augmentation applied during the training phase. As shown in Table 1, the impact of various augmentation schemes is evident. When neither the training nor test data are augmented, the model achieves a $93.89\,\%$ accuracy. However, when tested on augmented data without augmentation during training, the accuracy drops significantly to just $31.2\,\%$. This significant difference highlights the importance of training on augmented data to account for real-world variations such as translation, disorientation, or differences in body size. The model achieves the highest accuracy—almost $1.1\,\%$ higher than the baseline—when only the training set is augmented. This demonstrates that data augmentation not only improves the model's performance but also enables it to become invariant to geometric transformations, which is crucial when handling point clouds for tasks like gesture recognition.

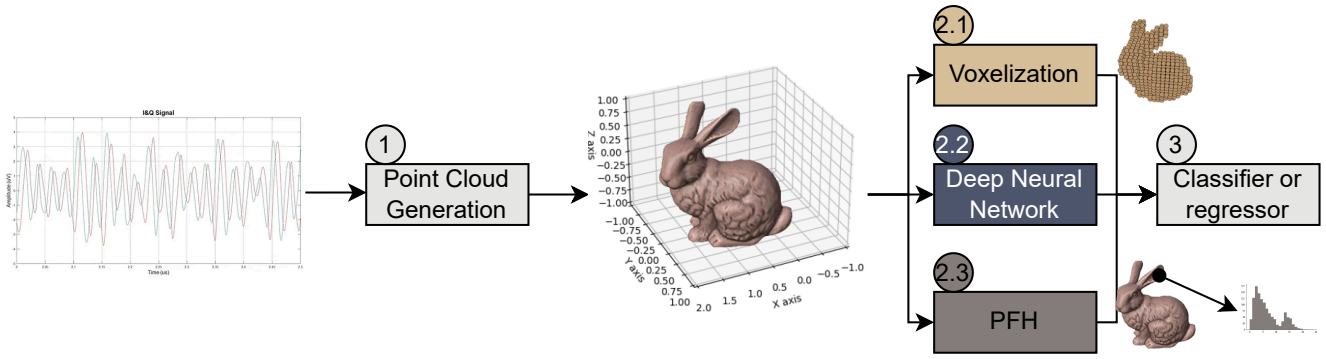


Figure 5: This figure illustrates three different methods of feature extraction for point clouds generated from RF sensing systems: voxelization, end-to-end deep neural networks, and PFH.

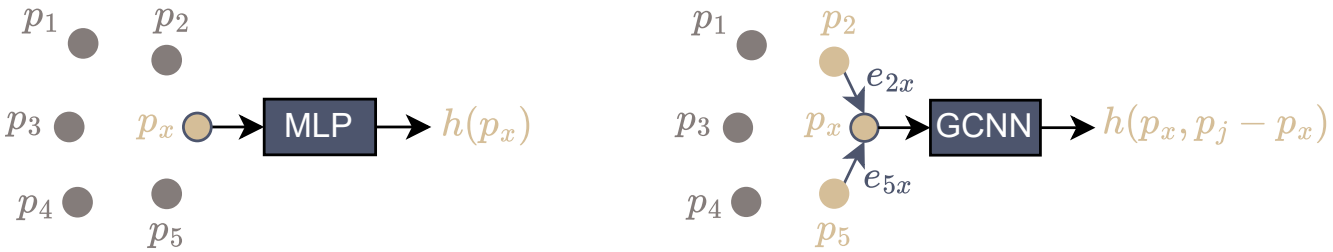
3.2.2 Feature Extraction

Feature extraction is crucial for transforming raw point cloud data into meaningful representations that can be efficiently processed by deep learning models. Several techniques have been developed to convert point cloud data into a format that captures important geometric and spatial features. Among these, voxelization, Point Feature Histograms (PFH), and end-to-end deep learning approaches are widely used, each offering different advantages and trade-offs, particularly when applied to RF-based sensing technologies like radar or Wi-Fi.

Voxelization Voxelization involves dividing the 3D space of a point cloud into a grid of smaller cubic units called voxels as shown in Fig.5. Each voxel aggregates the points that fall within its boundaries, simplifying the data into a regular grid structure. This transformation makes it easier to apply deep learning models, particularly 3D convolutional networks, which require structured grid-like input. In the context of RF sensing, voxelization allows the representation of RF data from radar signals, such as range and Doppler information, as volumetric data, facilitating feature extraction related to spatial characteristics, motion, or material properties. However, a key challenge is balancing the resolution of the grid with computational resources. High-resolution voxel grids improve accuracy but come with significant memory and computational costs, which is particularly critical in RF applications where data can be large and high-dimensional.

Point Feature Histogram (PFH) PFH focus on capturing local geometric properties by analyzing the relationships between points within a defined neighborhood as shown in Fig.5. PFH computes histograms of geometric features, such as angles between normal vectors, to describe the local surface geometry. For RF-based applications, PFH can be adapted to extract features from RF signals such as the intensity, phase, or Doppler shifts observed from different points in the point cloud. PFH is effective for capturing detailed structural features, such as the motion of objects or changes in signal characteristics in human or vehicle tracking. While PFH provides compact and meaningful representations of local features, it requires careful parameter selection, such as the size of the neighborhood and the number of histogram bins, to ensure a balance between detail and computational efficiency, especially in real-time PFH data collection.

End-to-End Deep Learning Approaches End-to-end deep learning approaches directly process raw point cloud data, learning relevant features from the data itself without the need for explicit manual feature extraction as shown in Fig.5. These methods, including models like PointNet [13], PointNet++ [14],



(a) Models using MLPs

(b) Models using GNNs

Figure 6: Two different approaches to process point clouds using neural networks: (a) using MLPs, (b) using GNNs.

Pantomime [12], and Tesla [16], are capable of learning spatial relationships, local patterns, and global structures during training. In RF sensing, this approach can be used to learn intricate relationships between raw signal data—such as time-domain measurements from radar or Wi-Fi CSI data—and 3D spatial features such as position, velocity, or material properties of objects. For instance, deep learning models can directly learn to segment or classify objects based on raw radar point clouds, such as distinguishing between human bodies, vehicles, or walls. The main advantage of this approach is that the model automatically extracts the most relevant features for the specific task, such as classification or tracking, without relying on pre-defined feature engineering. Additionally, end-to-end models are more flexible and adaptable to varying point cloud complexities, including changes in scale, orientation, and occlusion. Although these models require large amounts of labeled data and substantial computational resources for training, they eliminate the need for intermediate feature extraction steps, making them more efficient in terms of performance and computational complexity compared to voxelization or PFH-based methods, particularly when dealing with dynamic RF environments.

In conclusion, while voxelization and PFH-based techniques are useful in certain applications, end-to-end deep learning models offer a more efficient solution in terms of both performance and computational complexity for RF-based sensing applications. These models simplify the process by learning directly from the raw data, removing the need for explicit feature extraction and enabling better scalability to larger datasets and more complex point cloud scenarios, such as in real-time radar or Wi-Fi-based tracking, localization, and sensing.

3.3 Examples of Deep Learning Models for Point Cloud Data

Deep learning models for processing point clouds can generally be classified into two main types based on the underlying techniques they use: models based on symmetric functions such as Multi-Layer Perceptrons (MLPs), and those using Graph Neural Networkss (GNNs).

Models Using MLPs Symmetric functions, like MLPs, are used in models such as PointNet [13], PointNet++ [14], Pantomime [12], and PointGest [19]. These models rely on symmetric functions because the order of the points in a point cloud is irrelevant; the network needs to process the point cloud in a way that treats it as a set rather than a sequence. PointNet [13] introduced this idea by employing symmetric functions, such as max-pooling, to aggregate features from individual points, making the model invariant to permutations of points. PointNet++ [14] extends this concept by introducing hierarchical architectures to capture local features at different scales. These models have been widely applied in applications such as gesture recognition, where raw point clouds from sensors like LiDAR or mmWave radars are processed to extract relevant features for tasks like object classification or segmentation. Pantomime and PointGest, which are designed for gesture recognition using point clouds generated from mmWave

FMCW radars, also leverage similar techniques to process point clouds for accurate gesture identification. These models are particularly effective because they are able to handle the temporal and unordered nature of point clouds and make them suitable for real-world applications, where spatial data can vary in size, orientation, time, and scale.

This approach, shown in Fig.6a, can be mathematically described as follows:

1. **Input Representation:** The input is a point cloud that is represented in Eq.1.
2. **Feature Transformation**: Each point p_i is independently processed by a series of MLP layers to transform it into a higher-dimensional feature space. Let f_{MLP} denote the MLP transformation function:
$$h_i = f_{\text{MLP}}(p_i), \quad h_i \in \mathbb{R}^k. \tag{3}$$
3. **Symmetric Aggregation**: A symmetric function, such as max-pooling, is used to aggregate the features of all points into a single global feature vector:
$$h_{\text{global}} = \max_{i=1}^{N} \{h_i\}. \tag{4}$$
4. **Classification/Regression:** The global feature vector h_{global} is passed through additional MLP layers to perform tasks like classification or regression:
$$y = f_{\text{MLP-final}}(h_{\text{global}}). \tag{5}$$

Moreover, to capture the global spatial features of the input point clouds, sampling and grouping mechanisms can also be applied [14].

Models Using GNNs On the other hand, GNNs have been increasingly used for point cloud processing due to their ability to model complex relationships between points in a more structured way. Models like Tesla [16], Dynamic Graph Convolutional Neural Network (DGCNN) [25], and multi-view GNN for radar-generated point clouds [17] utilize graph-based approaches to capture local connectivity between points and their neighborhoods, making them suitable for tasks that require an understanding of the spatial relationships between points. In these models, the point cloud is represented as a graph, where points are nodes, and edges are formed based on their proximity in time or space. The GNNs perform message-passing operations on the graph, propagating information between neighboring points to update their features iteratively. This allows the model to capture global, local and even temporal structures effectively. Tesla focuses on temporal point cloud classification, while DGCNN excels in classification and segmentation tasks by dynamically constructing graphs based on local point neighborhoods. Multi-view GNN for radar-generated point clouds [17], designed for multi-view gesture recognition tasks, uses a similar graph-based approach to identify the gesture using multiple radars. These models are particularly beneficial when the point cloud data has complex, non-Euclidean relationships, which is often the case in applications like robotic navigation, 3D object detection, and localization.

The graph-based approach can be described as follows:

1. **Input Representation:** A point cloud P (described in Eq.1) represented as a graph $G = (V, E)$, where each node $v_i \in V$ corresponds to a point p_i in the point cloud. The edges E represent the connections between neighboring points.

2. **Graph Construction**: Define an adjacency matrix A based on the distances between points. Although typically K-Nearest Neighbors (KNN) are used to define the edges [25], more advanced methods like temporal KNN can also be utilized for that purpose [16]:

$$A_{ij} = \begin{cases} 1 & \text{if } p_j \text{ is one of the K nearest neighbors of } p_i \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

3. **Message Passing**: Each node (point) feature is updated by aggregating features from its neighbors. Let $h_i^{(l)}$ be the feature of node i at layer l. The update rule for a GNN layer can be written as:

$$h_i^{(l+1)} = \sigma\left(W^{(l)} h_i^{(l)} + \sum_{j \in \mathcal{N}(i)} \phi(h_i^{(l)}, h_j^{(l)}, e_{ij})\right). \tag{7}$$

   Here, σ is a non-linear activation function, $W^{(l)}$ is a learnable weight matrix, $\mathcal{N}(i)$ denotes the set of neighbors of node i, and ϕ is a message function that combines the features of node i and its neighbors j, possibly including edge features e_{ij}.

4. **Pooling and Readout:** After several message passing layers, a global pooling operation (e.g., max-pooling, mean-pooling) is applied to aggregate node features into a global feature vector:

$$h_{\text{global}} = \text{Pooling}(\{h_i^{(L)} \mid i \in V\}) \tag{8}$$

5. **Classification/Regression:** The global feature vector h_{global} is passed through fully connected layers for the final prediction:

$$y = f_{\text{MLP-final}}(h_{\text{global}}) \tag{9}$$

Moreover, incorporating attention mechanisms [22] into Eq.7 and Eq.8 can significantly enhance the robustness and performance of the model. Attention mechanisms allow the model to dynamically focus on the most relevant parts of the input data, effectively prioritizing important features while mitigating the impact of noise and irrelevant information. This approach has been shown to improve model accuracy and generalization across various tasks. Recent studies [16, 17, 24] demonstrate the effectiveness of integrating attention mechanisms with graph neural networks and symmetric function-based models, leading to superior performance in complex tasks such as point cloud classification, segmentation, and object detection.

Both approaches, whether using symmetric functions or graph-based networks, have proven successful for different types of point cloud data and applications, with the choice of model depending on the specific nature of the task and the type of relationships within the point cloud that need to be captured.

3.4 Training and Validation Strategies

3.4.1 Cross-Validation

Cross-validation is a crucial strategy for training deep learning models on point cloud data. By dividing the dataset into multiple folds and training the model on different subsets of the data, cross-validation helps in assessing the model's performance and robustness. This technique ensures that the model generalizes well to unseen data and reduces the risk of overfitting. Typically, k-fold cross-validation is employed, where the dataset is partitioned into k equal-sized folds. The model is trained on $k-1$ folds and validated on the remaining fold, with this process repeated k times. The average performance across all k trials provides a robust estimate of the model's generalization ability.

3.4.2 Performance Metrics

Evaluating the performance of deep learning models on point cloud data requires appropriate metrics. Common performance metrics include accuracy, precision, recall, and F1-score for classification tasks, and mean Intersection over Union (IoU) for segmentation and localization tasks. Accuracy measures the overall correctness of the model, while precision and recall evaluate the model's ability to correctly identify positive instances and its sensitivity to detecting all relevant instances, respectively. The F1-score, the harmonic mean of precision and recall, balances these two aspects. Mean IoU assesses the overlap between predicted and ground truth segments or bounding boxes in localization tasks, offering a comprehensive view of segmentation quality.

3.4.3 Loss Functions and Distance Metrics

Loss functions play a vital role in training deep learning models by quantifying the difference between predicted and actual values. For point cloud data, specialized loss functions such as Chamfer Distance, Earth Mover's Distance (EMD), and Hausdorff Distance are commonly used.

Chamfer Distance Chamfer Distance measures the average nearest-neighbor distance between two point sets P and Q, ensuring that each point in one set has a close counterpart in the other set:

$$\text{Chamfer Distance}(P, Q) = \frac{1}{2n} \sum_{p \in P}^{n} \|p - \text{NN}(p, Q)\| + \frac{1}{2m} \sum_{q \in Q}^{m} \|q - \text{NN}(q, P)\|, \tag{10}$$

where $\text{NN}(p, P) = \text{argmin}_{p' \in P} \|p - p'\|$ is the nearest neighbor function.

Earth Mover's Distance (EMD) The EMD between two point clouds P and Q is calculated as the average distance between pairs of points based on an optimal correspondence $\pi \in \Pi(P, Q)$. Here, $\Pi(P, Q)$ represents the set of $n \times m$ matrices where the sum of each row and column equals one. The assignment π is a matrix where $\Pi_{i,j}$ is a value between 0 and 1, indicating the degree of correspondence between point p_i and q_j. Formally, the EMD can be expressed as:

$$\text{EMD}(P, Q) = \min_{\pi \in \Pi(P,Q)} \sum_{i=1}^{n} \sum_{j=1}^{m} \pi_{i,j} \|p_i - q_j\| \tag{11}$$

Hausdorff Distance Hausdorff Distance measures the greatest of all the distances from a point in one set to the closest point in the other set, capturing the most significant dissimilarity between the two sets:

$$\text{Hausdorff Distance}(P, Q) = \frac{1}{2} \max_{p \in P} \|p - \text{NN}(p, Q)\| + \frac{1}{2} \max_{q \in Q} \|q - \text{NN}(q, P)\| \tag{12}$$

These loss functions are crucial for accurately training models to handle the unique challenges posed by point cloud data, such as varying point densities and geometric transformations.

In summary, effective training and validation strategies, coupled with appropriate performance metrics and loss functions, are essential for developing robust deep learning models for point cloud data. Cross-validation ensures generalization, while metrics like accuracy, precision, recall, F1-score, and mean IoU provide comprehensive evaluations of model performance. Specialized loss functions like Chamfer Distance, EMD, and Hausdorff Distance further refine the training process, enabling the models to learn intricate spatial relationships and geometrical properties of point clouds. This combination of strategies

and metrics forms a solid foundation for advancing deep learning applications in the context of 3D point cloud data.

4 Ethical Considerations

Privacy preservation in mmWave radar-based gesture recognition systems is a significant ethical concern. While radar systems offer certain privacy advantages over camera-based systems by not capturing visual data, they can still inadvertently reveal sensitive information. This section delves into the privacy risks associated with radar-based human sensing systems, specifically focusing on gesture recognition, and proposes strategies to mitigate these concerns. Additionally, we discuss the applicability of these strategies to other RF modalities, such as WiFi and RFID tags, and the benefits of using point cloud representations from a privacy perspective.

As we discussed before, mmWave radar technology, known for its high spatial resolution and ability to operate in various environmental conditions, is increasingly used in gesture recognition applications. However, the precision that makes these systems effective can also lead to privacy issues. For instance, detailed radar data can be used to extract sensitive biometric information, enabling potential user identification based on unique gesture patterns. This dual capability poses a challenge in maintaining user privacy while ensuring accurate gesture recognition.

Our research highlights the privacy risks by demonstrating that gesture data, although primarily intended for gesture recognition, can be exploited to identify individuals. This is particularly concerning in applications where user anonymity is critical, such as in healthcare and smart home environments. We investigated these risks using various datasets designed for gesture recognition, extracting data to identify different test subjects based on their gesture patterns observable in point cloud data. The findings underscore the inherent privacy concerns in radar-based gesture recognition systems. For example, our experiments showed that the identification accuracy of users based on their gesture data could be as high as $87\,\%$, which is alarmingly high considering the data was not intended for user identification.

To address these privacy issues, we propose a novel framework that obfuscates user identity while preserving gesture recognition accuracy. Our approach utilizes a graph-based autoencoder incorporating Message Passing Neural Network (MPNN) and multi-head self-attention mechanisms similar to what is used in Tesla [16]. This framework effectively balances the need for privacy protection with the functionality of gesture recognition systems.

By converting point cloud data into a form that retains gesture attributes but conceals user identity information, our method mitigates the risk of unauthorized identification. This is achieved by transforming the unordered structure of gesture point clouds into a directed graph that captures inter-frame temporal dynamics. The graph-based autoencoder processes this structured representation, encoding latent features that preserve essential gesture information while masking identity-related features. Our experimental results demonstrated that using this obfuscation method reduced the user identification accuracy from $87\,\%$ to below $25\,\%$, while the gesture recognition accuracy remained above $92\,\%$.

Point cloud representations, inherently focusing on the spatial distribution of points rather than visual or personally identifiable features, offer a natural advantage from a privacy perspective. They abstract the raw data into a form that is less directly tied to an individual's identity, focusing on the geometric properties of gestures rather than detailed visual attributes. This abstraction helps in reducing the risk of privacy breaches while retaining essential information for accurate recognition.

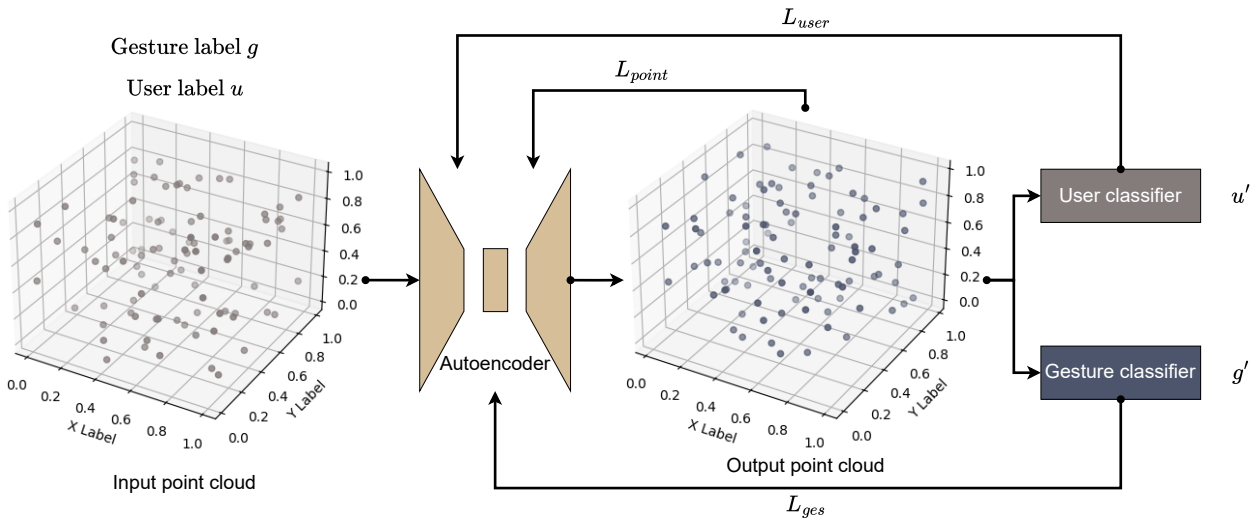


Figure 7: Schematic view of the proposed method to preserve the indentity of the user while performing the desired task using RF sensing.

4.1 System model

Given a point cloud p, let u represent the user identity attributes to be obfuscated and g denote the gesture attributes to be preserved. The objective is to find a function Φ that applies perturbations to the input point cloud p, resulting in a perturbed point cloud $p' = \Phi(p)$ with three desired properties: (1) Reduction in User Identification Performance: The performance of a user identification classifier U on p' is significantly diminished, ensuring that the classifier can no longer effectively distinguish between users. (2) Preservation of Gesture Recognition Accuracy: The performance of a gesture recognition classifier G on p' remains close to its performance on the original point cloud p, maintaining the usability of gesture-based applications. (3) Retention of Point Cloud Structure: The structure and coordinates of p' do not deviate significantly from those of p, preserving the overall geometric integrity of the point cloud for other potential applications.

Fig. 7 illustrates the proposed framework, which includes three sub-networks: an autoencoder AE for reconstructing input point clouds, a pre-trained gesture recognition classifier G, and a pre-trained user identification classifier U, with the weights of G and U fixed during the training of AE. The autoencoder is trained to minimize gesture recognition loss and user de-identification loss by passing the reconstructed point clouds through G and U, respectively. This ensures that AE effectively obscures user identity attributes while preserving essential gesture features, achieving the objectives outlined above.

4.1.1 Autoencoder to Reconstruct Point Cloud

Based on semi-adversarial methods [11], we employ an autoencoder with specific constraints to generate modified point clouds. we focus on reconstructing point clouds that represent gestures from different users, where each point cloud is composed of a series of frames, each containing an unordered set of points in 3D space. The unordered nature of these point clouds makes traditional convolution techniques challenging to apply. Moreover, unlike previous research that primarily relies on spatial properties for image and static point cloud reconstruction, our work must address both the spatial-temporal properties of the point clouds. This introduces a set of unique challenges, as the points in each frame that compose the entire point cloud are temporally dependent and subtle. Consequently, any perturbations

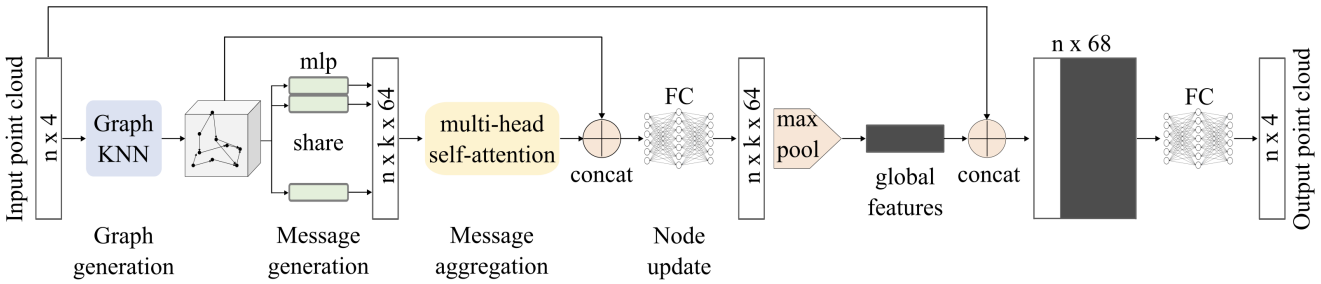


Figure 8: Graph-Based Autoencoder for Spatio-Temporal Point Cloud Reconstruction in Gesture Recognition.

applied must maintain temporal consistency and preserve the subtle spatial-temporal features critical for accurate gesture recognition. To model the spatio-temporal features of the point clouds, we designed a specialized graph-based autoencoder AE, as shown in Fig. 8. First, we generate a directed graph based on the initial features of the point cloud, using the edges between points to capture the temporal properties. Then, we apply a self-attention mechanism to extract a high-dimensional feature from the directed graph. Subsequently, the high-dimensional feature is concatenated with the initial features of each point, ultimately generating the target point cloud.

Given an input point cloud $p = \{p_1, p_2, \dots p_n\} \subseteq \mathbb{R}^4$, where each point p_i is a 4-dimensional vector: $p_i = (x_i, y_i, z_i, t_i)$. Here, (x_i, y_i, z_i) are the 3D coordinates and t_i is the frame index of the i-th point. First, we utilize the Graph KNN algorithm proposed in [16] to generate a directed graph $G = (V, E)$, where $V = p = \{p_1, p_2, \dots p_n\}$ is the set of nodes and $E = \{(p_i, p_j) \mid p_i, p_j \in V\}$ is the set of directed edges. The set of edges E is established by connecting each point p_i to its k nearest neighbors within the set of points from the previous frames. Let Vp_i represent the set of all points from frames that precede the frame containing point p_i, defined as $Vp_i = \{p_j \in p \mid x_j < x_i\}$. If p_j belongs to the set Vp_i, the distance between p_i and p_j is defined as the Euclidean distance. If p_j is not in Vp_i, the distance is considered to be infinite. Then, the distance can be represented as

$$D_{p_i,p_j} = \begin{cases} \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} & p_j \in V_{p_i}, \\ \infty & p_j \notin V_{p_i}. \end{cases} \tag{13}$$

After calculating the Euclidean distances between p_i and each point p_j in V_{p_i}, the k nearest points to p_i are selected. These points are then connected to p_i, establishing directed edges that reflect the temporal features within the point cloud within the point cloud.

The original point cloud p combined with the set of established directed edges e forms the graph $G = (p, e)$. Subsequent to the formation of the directed graph G, we utilize a Message Passing Neural Network (MPNN) to extract the features of the nodes within the graph, which is designed to handle graph-structured data by utilizing the directional edges to guide information flow between nodes.

The initial node features are represented by $h_i = f_{\text{mlp}}(p_i)$, where f_{mlp} refers to a multi-layer perceptron. The choice of an MLP to encode node features is driven by its ability to capture complex non-linear relationships within the data, which enhances the initial embeddings and facilitates a deeper understanding of the graph's intricate structure. To derive the representation h_i' after the MPNN, three primary stages are involved: message generation, message aggregation, and node updating.

During the message generation phase, for each edge (p_i, p_j), we define the message generation function M as a fully connected network. All nodes share the same network with unified weights. This approach minimizes the number of trainable parameters and ensures generalizability across various node configurations within the graph. The concatenated vector $(h_i, (h_j - h_i))$ is employed as the input to capture the local dependencies as well as the global structure. Specifically, for node p_i and its

neighboring node p_j, the message generation process is described by:

$$m_{ij} = M\left(\text{concat}\left(h_i, (h_j - h_i)\right)\right), \tag{14}$$

where M is the message generation function that maps the concatenated features to produce the message m_{ij}.

In the message aggregation phase, a multi-headed self-attention mechanism is utilized. This mechanism enables the model to differentially focus on various parts of the input messages across each head. For each attention head b, the messages m_{ij} are linearly transformed by the corresponding weight matrices to generate sets of queries Q, keys K, and values V. Specifically, the queries $Q_b^{(i)}$, keys $K_b^{(i)}$, and values $V_b^{(i)}$ for point p_i are computed as follows:

$$Q_b^{(i)} = W_b^Q m_{ij}, \quad K_b^{(i)} = W_b^K m_{ij}, \quad V_b^{(i)} = W_b^V m_{ij}, \tag{15}$$

where W_b^Q, W_b^K, W_b^V are the weight matrices for the b-th head. Attention score α_{ij} between node p_i and its neighbor p_j is then calculated by:

$$\alpha_{ij}^{(b)} = \frac{\exp\left(\frac{Q_b^{(i)} \cdot K_b^{(j)T}}{\sqrt{k}}\right)}{\sum_{j' \in \mathcal{N}(i)} \exp\left(\frac{Q_b^{(i)} \cdot K_b^{(j')T}}{\sqrt{k}}\right)}, \tag{16}$$

where $\mathcal{N}(i)$ denotes the set of neighboring nodes of p_i, and k is the dimension of the queries $Q_b^{(i)}$ and keys $K_b^{(i)}$, reflecting the number of neighbors each node has in the graph G.

The attention scores are leveraged to aggregate messages from the respective neighbors for each attention head b:

$$z_i^{(b)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(b)} V_b^{(i)}. \tag{17}$$

This aggregation mechanism effectively captures the weighted significance of each neighbor's features. Subsequently, to synthesize the diverse representational focuses of each head, the vectors $z_i^{(b)}$ from all heads are concatenated and then linearly transformed:

$$z_i = \text{concat}(z_i^{(1)}, z_i^{(2)}, ..., z_i^{(B)}) W^O, \tag{18}$$

where W^O is the output transformation matrix that integrates the multi-headed outputs into a unified feature representation for node p_i, with B indicating the number of attention heads.

In the node updating phase, node features are refined using the aggregated messages. The update mechanism employs a fully connected network, denoted as f_{update}, which processes the concatenated current node features h_i and the aggregated messages z_i through linear transformations followed by non-linear activations. The updated node features are given by:

$$h_i' = f_{\text{update}}[\text{concat}(h_i, z_i)]. \tag{19}$$

Following the feature update, a max pooling operation is applied across the entire point set to aggregate the features, enhancing the representation with global contextual information. The result of the max pooling across all nodes is computed as:

$$H_{\text{max}} = \max_{0 \le i < n} h_i', \tag{20}$$

where max denotes the max pooling operation.

These aggregated features, represented by H_{max}, are subsequently concatenated with the initial position feature p_i, forming an enriched feature set. This set undergoes further processing through multiple fully connected layers, designated as f_{out}, to produce the de-identified point cloud p'. The final transformation is represented by:

$$p' = f_{out}\left[\text{concat}\left(p_i, H_{\text{max}}\right)\right]. \tag{21}$$

4.1.2 De-identification

The primary goal of the de-identification process is to modify the point clouds generated by the autoencoder to effectively obscure user identity while maintaining the accuracy of gesture recognition. To achieve this, a specialized de-identification loss $\mathcal{L}_{user}$ is employed, with the objective of maximizing the misclassification rate of the user identification classifier U on the generated point clouds. To encourage incorrect predictions of user identity, the de-identification loss $\mathcal{L}user$ is defined as the negative of the negative log-likelihood (NLL) loss:

$$\mathcal{L}_{user} = \frac{1}{N}\sum_{i=1}^{N}\log(u'_{i,u_i}) \tag{22}$$

where $u'i$ denotes the predicted probability vector, and $u'i, u_i$ indicates the predicted probability that the i-th point cloud belongs to its actual class u_i. By maximizing the negative log of this probability, we encourage the classifier to make incorrect predictions of user identity, thereby achieving the effect of de-identification.

During the training process, we observed that the original de-identification loss function could lead to numerical stability issues, prompting optimizations to ensure its positivity and stability. To address this, we introduced the logarithmic function $\log(1+x)$ and an offset δ to regulate the rate of constraint growth, which are crucial for preventing gradient explosion during the training phase. The enhanced de-identification constraint function is formulated as follows:

$$\mathcal{L}^{'}_{user} = -\log\left(1+\mathcal{L}_{user}\right) + \delta, \tag{23}$$

where δ is an offset introduced to ensure that the loss function maintains a positive value, thereby supporting consistent optimization behavior. It is important to note that while δ shifts the overall value of the loss function, it does not influence the gradient and thus does not affect the direction or magnitude of gradient updates during backpropagation. This formulation not only enhances the de-identification effect but also ensures the stability and efficiency of training, enabling the autoencoder to function effectively and reliably under various training conditions.

4.1.3 Gesture Recognition Preservation

While the de-identification process aims to obscure user identity, it is essential to ensure that this modification does not compromise the gesture recognition capability of the generated point clouds. This is crucial for maintaining the usability and functionality of the data in downstream tasks, such as gesture recognition. To address the impact of de-identification on gesture recognition accuracy, we introduce a gesture recognition loss.

This loss is implemented by passing the generated point clouds through a gesture recognition classifier. During the autoencoder training process, the gesture recognition classifier imposes constraints to ensure that the generated point clouds retain essential gesture features. Additionally, during the

evaluation phase, it examines both the original and generated point clouds to measure the effect of de-identification on gesture recognition accuracy. In this study, we utilize Tesla [16] as our gesture recognition classifier. Similar to the user identification constraint, we implement this constraint by minimizing the gesture recognition loss during the autoencoder training process, which is defined as the negative log-likelihood loss:

$$\mathcal{L}_{\text{ges}} = -\frac{1}{N}\sum_{i=1}^{N}\log(g'_{i,g_i}) \tag{24}$$

where g'_i represents the predicted probability vector and g'_{i,g_i} denotes the predicted probability of the i-th point cloud belonging to its actual gesture class g_i.

Minimizing $\mathcal{L}_{\text{ges}}$ guides the autoencoder to preserve essential gesture-related information in the point clouds while applying de-identification modifications. This ensures that the autoencoder focuses on altering only the identity-relevant features, leaving those necessary for precise gesture recognition intact.

4.1.4 Point Cloud Reconstruction

In addition to the objectives of de-identification and gesture recognition preservation discussed above, we also introduce a point cloud reconstruction loss. This loss is designed to help the autoencoder preserve the inherent spatio-temporal features of the input point clouds during reconstruction. By enhancing the quality of the reconstructed point clouds, we aim to ensure that any modifications for de-identification do not significantly alter the underlying data structure. This is particularly important for tasks requiring high-fidelity representations, as it supports the usability and integrity of the processed data in downstream applications.

To accurately reconstruct the input point clouds, we employ the Chamfer Distance (CD), which preserves the detailed spatial relationships within the point clouds and ensures high-quality reconstruction. The Chamfer Distance $\mathcal{L}_{cd}$ between the original point cloud p and the reconstructed point cloud p' is defined as:

$$\mathcal{L}_{cd}(p,p') = \sum_{p_m\in p}\min_{p_n\in p'}\|p_m-p_n\|_2^2 + \sum_{p_n\in p'}\min_{p_m\in p}\|p_n-p_m\|_2^2, \tag{25}$$

in which, the operation $\|p_n - p_m\|_2^2$ represents the squared Euclidean distance between points p_m and p_n, ensuring that the nearest point pairs between p and p' are considered. This loss function identifies the nearest point pairs between the original and reconstructed point clouds and sums their squared Euclidean distances. By minimizing $\mathcal{L}_{cd}$, the autoencoder is trained to retain the overall structure and geometry of the point clouds during reconstruction.

4.1.5 Training Loss

To achieve a balanced optimization between point cloud reconstruction, gesture recognition preservation, and de-identification, we propose an integrated training loss function. The purpose of this combined loss function is to impose multiple constraints during training, ensuring that the model can accurately reconstruct the input point clouds, preserve task-relevant features, and effectively obscure user identity.

The final training loss $\mathcal{L}_{\text{final}}$ is formulated as a weighted combination of the point cloud reconstruction loss, gesture recognition loss, and user identification loss:

$$\mathcal{L}_{\text{final}} = \alpha\cdot\mathcal{L}_{\text{cd}} + \beta\cdot\mathcal{L}_{\text{ges}} + \gamma\cdot H(A_{\text{user}}(t)-\tau)\cdot\mathcal{L}_{user}, \tag{26}$$

where α, β, and γ are scaling factors that adjust the relative importance of each loss component in the overall training process and $A\text{user}(t)$ denotes the user identification accuracy at training time t, and

$H(x)$ is the Heaviside step function, defined as $H(x) = 1$ if $x \geq 0$, and $H(x) = 0$ otherwise, which is used to activate the user identification loss $\mathcal{L}_{user}$ only when the user identification accuracy exceeds the threshold τ.

In this combined training loss, the point cloud reconstruction loss $\mathcal{L}_{\text{cd}}$ ensures that the autoencoder accurately reconstructs the input point clouds while preserving their spatial and geometric structure. The gesture recognition loss $\mathcal{L}_{\text{ges}}$ acts as a constraint to maintain gesture-related features in the generated point clouds, enabling effective use in gesture recognition tasks even after de-identification. The user identification loss $\mathcal{L}_{user}$ is triggered only when the user identification accuracy exceeds the threshold τ, allowing the de-identification mechanism to be incorporated to reduce the user identification accuracy of the generated point clouds.

4.2 Generalization to Other Modalities

The privacy preservation framework we developed for millimeter-wave radar can also be applied to other RF sensing modalities which utilize point cloud representations, such as WiFi and RFID tags. These modalities similarly collect detailed spatial and temporal data that can inadvertently reveal sensitive information about users. WiFi sensing systems, which leverage CSI to recognize gestures and track movements, face similar privacy challenges. Detailed CSI data can be used to identify individuals based on their unique movement patterns. By applying our graph-based autoencoder framework to WiFi CSI data which is converted to a point cloud representation, we can obfuscate identifying features while retaining the critical information needed for gesture recognition and movement tracking. This ensures that WiFi-based sensing systems can protect user privacy effectively.

RFID tags are used for various applications, including tracking and identifying objects or individuals. RFID-based systems can inadvertently capture detailed movement patterns and behavioral data. Using a similar approach to the one we developed for radar and WiFi data, we can transform RFID data into a privacy-preserving format. By encoding the temporal dynamics and spatial relationships in RFID data into graph structures and applying autoencoding techniques, we can mitigate the risk of user identification while maintaining system functionality.

In conclusion, as RF sensing technologies like millimeter-wave radar, WiFi, and RFID tags continue to evolve, addressing the ethical implications of privacy is paramount. Our proposed framework offers a promising solution by integrating advanced neural architectures to achieve a balance between privacy preservation and accurate sensing capabilities. Further research and development in this area are essential to enhance privacy protections and ensure the ethical deployment of RF-based sensing technologies across various applications and modalities. Point cloud representations, due to their focus on spatial rather than visual or detailed biometric data, provide a robust foundation for balancing privacy with functionality, making them an ideal choice for privacy-sensitive applications.

5 Future Work

The data structures and representations established in this deliverable form a solid foundation for processing massive RF datasets. However, there is considerable scope for refinement. Future work can focus on creating adaptive data structures capable of dynamically adjusting to the requirements of specific applications, such as optimizing granularity for tasks ranging from real-time gesture recognition to large-scale environmental modeling. Hierarchical representations, which enable multi-scale data processing, could significantly enhance computational efficiency without compromising accuracy. Additionally, advancing interoperability between RF sensing modalities, such as mmWave, RFID, and Wi-Fi, would facilitate broader integration across applications. As privacy concerns grow, embedding anonymization

and encryption mechanisms directly into data structures will ensure that human-centric sensing systems align with ethical and legal requirements.

The landscape of RF sensing and holography is rapidly evolving, with several emerging technologies offering opportunities to advance the Holden project. Edge computing and federated learning present viable solutions for localized data processing, minimizing latency and enhancing privacy for dense network applications. Similarly, quantum computing is exptected to revolutionize optimization techniques for RF data compression and real-time holographic rendering. In artificial intelligence, advancements in models like graph neural networks and transformers provide powerful tools for handling the complexity of point cloud data. Interdisciplinary innovations, particularly in materials science, are expected to yield low-power, high-sensitivity RF sensors, enabling efficient deployment in dense network environments.

6 Conclusion

This deliverable has provided a detailed framework for managing massive RF data inputs, emphasizing their use in deep learning applications relevant to holographic design. The findings highlight the effectiveness of point cloud representations as a robust, memory-efficient structure for dense network environments. The methodologies outlined demonstrate how advanced models, such as PointNet and GNN, can integrate point clouds into holographic applications. Memory optimization techniques further solidify the feasibility of these approaches. Moreover, the deliverable underscores the importance of privacy-preserving mechanisms and ethical considerations, ensuring that the developed technologies align with societal values. Recommendations for future efforts include enhancing interoperability, exploring interdisciplinary innovations, and scaling computational frameworks for broader applications.

This deliverable marks a significant milestone in the Holden project, establishing a comprehensive approach to processing and analyzing massive RF datasets for holographic applications. By focusing on ethical design and cutting-edge technologies, it bridges the gap between technical feasibility and societal expectations. The frameworks and findings presented here not only advance the technical goals of the project but also lay the groundwork for transformative applications in healthcare, autonomous systems, and smart infrastructure. This work reinforces Holden's commitment to responsible innovation, setting a precedent for future research and development in dense wireless networks and ethical holography.

References

[1] Chen Chen, Gang Zhou, and Youfang Lin. Cross-domain wifi sensing with channel state information: A survey. *ACM Computing Surveys*, 55(11):1–37, 2023.

[2] Fabiola Colone, Debora Pastina, Paolo Falcone, and Pierfrancesco Lombardo. Wifi-based passive isar for high-resolution cross-range profiling of moving targets. *IEEE Transactions on Geoscience and Remote Sensing*, 52(6):3486–3501, 2013.

[3] Cao Dian, Dong Wang, Qian Zhang, Run Zhao, and Yinggang Yu. Towards domain-independent complex and fine-grained gesture recognition with rfid. *Proceedings of the ACM on Human-Computer Interaction*, 4(ISS):1–22, 2020.

[4] Lewis Graham. The las 1.4 specification. *Photogrammetric Engineering & Remote Sensing*, 78(2), 2012.

[5] Steven M Hernandez and Eyuphan Bulut. Wifi sensing on the edge: Signal processing techniques and challenges for real-world systems. *IEEE Communications Surveys & Tutorials*, 25(1):46–76, 2022.

[6] Yuchen Huang, Wei Li, Zhiyang Dou, Wantong Zou, Anye Zhang, and Zan Li. Activity recognition based on millimeter-wave radar by fusing point cloud and range–doppler information. *Signals*, 3(2):266–283, 2022.

[7] Shang Jiang, Jianguo Jiang, Siye Wang, Yanfang Zhang, Yue Feng, Ziwen Cao, and Yi Liu. Rf-gait: Gait-based person identification with cots rfid. *Wireless Communications and Mobile Computing*, 2022(1):3638436, 2022.

[8] Yier Lin and Fan Yang. Iq-data-based wifi signal classification algorithm using the choi-williams and margenau-hill-spectrogram features: A case in human activity recognition. *Electronics*, 10(19):2368, 2021.

[9] Yongsen Ma, Gang Zhou, and Shuangquan Wang. Wifi sensing with channel state information: A survey. *ACM Computing Surveys (CSUR)*, 52(3):1–36, 2019.

[10] Tuomas Määttä, Sasan Sharifipour, Miguel Bordallo López, and Constantino Álvarez Casado. Spatio-temporal 3d point clouds from wifi-csi data via transformer networks. *arXiv preprint arXiv:2410.16303*, 2024.

[11] Vahid Mirjalili, Sebastian Raschka, and Arun Ross. Privacynet: Semi-adversarial networks for multi-attribute face privacy. *IEEE Transactions on Image Processing*, 29:9400–9412, 2020.

[12] Sameera Palipana, Dariush Salami, Luis A Leiva, and Stephan Sigg. Pantomime: Mid-air gesture recognition with sparse millimeter-wave radar point clouds. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies*, 5(1):1–27, 2021.

[13] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[14] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[15] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, pages 1–4. IEEE, 2011.

[16] Dariush Salami, Ramin Hasibi, Sameera Palipana, Petar Popovski, Tom Michoel, and Stephan Sigg. Tesla-rapture: A lightweight gesture recognition system from mmwave radar sparse point clouds. *IEEE Transactions on Mobile Computing*, 22(8):4946–4960, 2022.

[17] Dariush Salami, Ramin Hasibi, Stefano Savazzi, Tom Michoel, and Stephan Sigg. Angle-agnostic radio frequency sensing integrated into 5g-nr. *IEEE Sensors Journal*, 2024.

[18] Dariush Salami, Anni Juvakoski, Riku Vahala, Michael Beigl, and Stephan Sigg. Water quality analysis using mmwave radars. In *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pages 412–415. IEEE, 2023.

[19] Dariush Salami, Sameera Palipana, Manila Kodali, and Stephan Sigg. Motion pattern recognition in 4d point clouds. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2020.

[20] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719, 2020.

[21] Greg Turk. The ply polygon file format. *Recuperado de*, 1994.

[22] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[23] Fei Wang, Sanping Zhou, Stanislav Panev, Jinsong Han, and Dong Huang. Person-in-wifi: Fine-grained person perception using wifi. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5452–5461, 2019.

[24] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10296–10305, 2019.

[25] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019.

[26] Aobo Zhao, Gui Yun Tian, and Jun Zhang. Iq signal based rfid sensors for defect detection and characterisation. *Sensors and Actuators A: Physical*, 269:14–21, 2018.