IEEE ComSoc IEEE Communications Society
IEEE Open Journal of the Communications Society

Received 19 September 2025; revised 30 October 2025; accepted 18 November 2025. Date of publication 21 November 2025; date of current version 1 December 2025.

Digital Object Identifier 10.1109/OJCOMS.2025.3636436

A Foundation Model for Wireless Technology Recognition and Localization Tasks

MOHAMMAD CHERAGHINIA [1] (Graduate Student Member, IEEE), ELI DE POORTER [1], JARON FONTAINE [1], MEROUANE DEBBAH [2], AND ADNAN SHAHID [1] (Senior Member, IEEE)

[1]IDLab, Department of Information Technology, Ghent University - imec, 9052 Ghent, Belgium
[2]Center for 6G Technology, Khalifa University of Science and Technology, Abu Dhabi, UAE

CORRESPONDING AUTHOR: M. CHERAGHINIA (e-mail: mohammad.cheraghinia@ugent.be)

This work was supported in part by the SNS JU European Union's Horizon Europe Research and Innovation Program under Grant 101139194 (6GXCEL); in part by the Horizon Europe Program under the MSCA Staff Exchanges under Grant 101086218 (EVOLVE); and in part by the Scientific Research Flanders (FWO-Vlaanderen) through the SB-Ph.D. Fellowship under Grant 1S52025N and through the FWO Research Project PESSO under Grant G018522N.

ABSTRACT Wireless Technology Recognition (WTR) and localization are essential in modern communication systems, enabling efficient spectrum usage, coexistence across diverse technologies, and accurate positioning in dynamic environments. Real-world deployments must handle signals from different sampling rates, capturing devices, frequency bands, and propagation conditions. Traditional methods, such as energy detection and conventional Deep Learning (DL) models like Convolutional Neural Networks (CNNs), often fail to generalize across unseen technologies, environments, or tasks. In this work, we introduce a Transformer-based foundation model for both WTR and localization, pre-trained in a self-supervised manner on large-scale unlabeled aciq and Channel Impulse Response (CIR) timeseries data. The model aims for reusability and generalizability compared to single-task architectures. It leverages input patching for computational efficiency and employs a two-stage pipeline: self-supervised pre-training to learn general-purpose representations, followed by lightweight fine-tuning for task-specific adaptation. This enables the model to generalize to new wireless technologies and unseen environments using minimal labeled samples. Evaluations across short-range and long-range datasets show superior accuracy in WTR (up to 99.99%), Line-Of-Sight (LOS) detection (up to 100%), and ranging error correction (reducing Mean Absolute Error (MAE) by up to 50%), all while maintaining low computational complexity. These results underscore the potential of a reusable wireless foundation model for multi-task applications with minimal retraining.

INDEX TERMS Foundation model, transformers, wireless technology recognition, convolutional neural networks, patching, localization.

I. INTRODUCTION

THE EVOLUTION of wireless communication systems has shown an era of unique connectivity, including applications from autonomous vehicles [1] and smart homes [2] to localization ecosystems [3]. Wireless systems must handle the coexistence of diverse technologies, including long-range technologies (e.g., Sigfox, LoRa, IEEE 802.11ah, IEEE 802.15.4) for energy-efficient coverage and short-range technologies (e.g., LTE, WiFi, 5G) for high-data-rate links. Effective WTR specifies signals based on features like bandwidth, modulation, and temporal patterns, which is important for spectrum management [4], interference mitigation [5], and secure spectrum sharing [6]. On the other hand, precise localization in multipath environments, such as distinguishing between LOS and Non-Line-Of-Sight (NLOS) paths in Ultra-Wideband (UWB) systems and correcting ranging errors, is essential for reliable positioning in dynamic settings, such as warehouses or offices.

However, real-world wireless solutions face challenges due to data heterogeneity: signals vary in sampling rates, capture devices, frequency bands, and environmental conditions, leading to poor generalization in traditional models [7]. Task-specific approaches often require large labeled datasets for each scenario separately, which fail to generalize to unseen technologies, devices, or environments without extensive retraining. This inefficiency is worsened by the growing

complexity of wireless applications, where new standards emerge rapidly and environments change unpredictably [8]. All these make a task-specific model very impractical and non-reusable.

To overcome these limitations, foundation models are pre-trained on vast amounts of unlabeled data to learn general-purpose features, establishing a transformative paradigm. Inspired by models in natural language processing [9] and computer vision [10], foundation models enable learning across diverse tasks, reducing dependence on task-specific labeled data and enabling generalization to novel scenarios. In wireless communications, a foundation model is required to do various tasks such as WTR and localization, capturing patterns in raw In-phase and Quadrature (IQ) and CIR timeseries. Why is this essential? First, it enables data efficiency: by pre-training on diverse unlabeled signals, the model learns robust features that generalize to unseen classes or environments, thus overcoming the challenge of limited data availability in wireless datasets. Second, it enhances scalability and reusability: as the number of wireless technologies increases (e.g., toward 6G and beyond), a reusable foundation can adapt to new technologies without rebuilding models from scratch, aligning with ongoing research in multi-task learning. Third, it supports computational sustainability: through techniques like patching, it efficiently handles long sequences, avoiding the complexity of vanilla Transformer architectures while preserving dependencies. These reasons indicate that the foundation models are critical to streamline fine-tuning toward new wireless tasks in new conditions.

In this work, we propose a Transformer-based foundation model pre-trained in a self-supervised manner on IQ and CIR datasets. Our direct timeseries solution bypasses the computational overhead associated with spectrogram generation, using the critical temporal and phase features from raw IQ and CIR data. Unlike spectrograms, which require an additional transformation step, this raw data is readily available from many commercial radio front-ends (e.g., UWB transceivers like the Qorvo DW3000 series [11] or various software-defined radios [12]), making our solution directly applicable to existing hardware. Using patching for reduced complexity and a two-stage pipeline (self-supervised pre-training followed by lightweight fine-tuning), the model generalizes across short-range and long-range technology recognition tasks, as well as UWB localization tasks ((N)LOS classification and localization error correction). To assess our model under challenging conditions, we employ different datasets to include sampling rate and capturing device diversity. Furthermore, we evaluate its generalization capabilities by testing it in different environments for localization tasks and on unseen classes for the WTR tasks. Experimental results demonstrate its effectiveness: achieving up to 99.99% accuracy in multi-class WTR, 100% in LOS/NLOS detection, and up to 50% reduction in ranging MAE.

The main contributions of the paper are summarized as follows:

- We propose a foundation model that directly uses raw timeseries data (IQ and CIR), eliminating the overhead of data pre-processing. To our knowledge, this is the first wireless foundation model capable of processing heterogeneous input types like IQ samples and CIR (each with distinct data features) within a single, unified architecture.
- We propose a patching strategy designed for our Transformer architecture. This technique addresses the challenge of processing sequences by segmenting the input data into smaller patches, enabling the model to enhance its computational efficiency and accuracy.
- We introduce a foundation model that adapts across WTR and localization tasks by handling heterogeneous data from **multiple technologies, sampling rates, and environments**. Through an evaluation of pre-training strategies, we demonstrate that a self-supervised approach achieves competitive or superior performance to supervised methods, validating the model's effectiveness as a true wireless timeseries foundation model.
- We conduct an evaluation across a diverse range of downstream tasks, encompassing both **classification and regression**. Using short-range, long-range, and UWB datasets, we assess the model's generalization capabilities, specifically its performance in **unseen environments and wireless technologies** not present during pre-training. Our model is benchmarked against baselines across multiple metrics, including predictive accuracy and model complexity.

The remainder of the paper is organized as follows: Section II reviews related works and explains the wireless foundation models and related studies. Section III introduces the system model and description of pre-training and fine-tuning strategies. Section IV describes the methodology, including the architecture of the proposed solution and baseline models. Section V presents results, discussion, and computational analysis. Finally, Section VI concludes the paper and provides directions for future work.

II. RELATED WORKS

This section is an overview of the literature relevant to our work, including three key domains. We first review the application of AI in WTR and UWB localization, exploring both established and learning-based methods in comparison to our downstream tasks. We then delve into the evolving field of foundation models for wireless applications, providing the direct context for our proposed approach.

A. AI IN WTR

Traditional methods, such as energy detection, are simple but struggle in noisy and complex environments, while

more robust techniques, like cyclostationary detection, are computationally expensive [13], [14]. CNNs represent a major improvement by automatically learning features from raw IQ data. Studies show CNNs can achieve high accuracy 99% on low-power devices [15]. To address the large data requirements of CNNs, Semi-Supervised Learning (SSL) has been used to achieve over 70% accuracy with only 10% of the labeled data [16]. However, a key weakness of CNNs is their poor ability to generalize to new, unseen wireless technologies without extensive retraining, as they rely on convolutional kernels to capture local patterns from the training data [17], [18], [19].

Transformers are well-suited for sequence data like IQ samples due to their ability to model timeseries dependencies. Several studies [20], [21], [22], [23] have applied Transformers to IQ samples using various techniques like hybrid architectures, SSL, and different data windowing methods. While promising, these approaches often focus on modulation classification in simulated environments and do not consider data with different sampling rates and capturing devices. Vision-based models convert wireless signals into images (e.g., spectrograms) and then apply computer vision techniques. These methods have shown high accuracy [24], [25] but can be computationally heavy, requiring pre-processing time and very large models with millions of parameters [26].

Recent studies have explored self-supervised and meta-learning solutions to remove the need for extensive labeled datasets [27], [28], [29]. Some approaches convert timeseries signals into image-based representations, like time-frequency diagrams [27], or utilize multi-modal inputs (e.g., IQ, Amplitude/Phase, and Power Spectral Density) to tackle few-shot or open-set challenges [30]. While effective, these methods can introduce computational overhead and associated latency from the pre-processing transformations [31], unlike our approach that uses the direct CIR or IQ output of off-the-shelf devices. Other studies apply various contrastive and self-supervised learning frameworks directly to raw timeseries data [28], [29], [32], [33], but they typically focus on the singular task of modulation classification, often with an emphasis on improving robustness against noise [31], channel impairments, developing novel data augmentations [28], or handling out-of-distribution signals [29]. Our work differs by proposing a versatile, Transformer-based foundation model pre-trained on raw IQ and CIR timeseries. This approach is designed to handle multiple downstream tasks with heterogeneous data and not only enables multi-task capabilities but also avoids the computational limitations and associated delay of image conversion, extending beyond WTR to include localization and LOS detection. While this direct timeseries processing may ignore some explicit frequency information present in spectrograms, it preserves the critical phase and temporal details necessary for localization tasks, offering a more efficient and broadly applicable solution.

B. AI IN UWB LOCALIZATION

Several studies use Machine Learning (ML) to enhance UWB positioning accuracy by correcting ranging errors and LOS detection, mainly using the CIR. These approaches can be grouped into two main categories.

Many studies focus on directly predicting and correcting the ranging error. By extracting features such as signal strength from the CIR, models like Support Vector Machines (SVM) [34], Gaussian Processes, and autoencoders [35] have been used for error correction. On the other hand, many studies focus solely on LOS detection. Raw CIR-based approaches generally show superior performance for (N)LOS detection, with one study reporting a 27.9% increase in accuracy by using a CNN [36]. Similarly, other researchers have combined different neural network architectures to improve performance. Reference [37] applied a Gate Recurrent Unit (GRU) to extract spatial features from the raw CIR data before feeding them to a CNN for classification. Another hybrid model was proposed [38], which used a CNN to extract non-temporal features that were then fed into an Long Short-Term Memory (LSTM) network for the final NLOS classification, achieving an accuracy of 82.14%.

Other papers first classify the LOS detection and then apply a targeted error correction. Techniques like ensemble tree classifiers, fuzzy logic, and CNNs have been employed for the classification step. One study achieved 95.65% accuracy in (N)LOS detection and reduced the final error to a root mean square error of just 0.4790 m [39]. Another approach combined ML-based error correction with optimal anchor selection to reduce positioning error by 75% [40]. A critical limitation noted across all these papers is that their methods were not evaluated in new, unseen environments. This makes it difficult to determine how well these solutions would generalize to real-world deployment scenarios. To address this, the study by [41] proposes an automatic transfer learning framework that adapts a pre-trained neural network to new environments using a small number of data samples, significantly improving accuracy with minimal new data collection.

C. FOUNDATION MODELS

Foundation models have gained attention for learning generalizable representations from diverse datasets, enabling adaptation to multiple downstream tasks. Recent works include Large Wireless Localization Model (LWLM) [42], which pre-trains on Channel State Information (CSI) data using masked modeling and contrastive objectives for tasks like Time-of-Arrival (TOA) estimation, Angle-of-Arrival (AOA) estimation, Single-Base Station (BS) localization, and multi-BS localization. IQFM [43] operates on IQ streams with contrastive SSL and task-aware augmentations, supporting modulation classification, AoA prediction, beam prediction, and Radio Frequency (RF) fingerprinting. Wireless Foundation Model (WiFo) [44] employs masked autoencoders on CSI for unified time- and frequency-domain channel prediction. SpectrumFM [45] uses IQ data with

TABLE 1. Comparison of wireless foundation models.

Model	Input Data	Preprocessing Required	Downstream Tasks	Tasks Type
LWLM [42]	CSI	Yes	ToA/AoA estimation, single-base station localization	Regression
IQFM [43]	IQ	Minimal (augmentations)	modulation classification, AoA, beam prediction, RF fingerprinting	Classification
WiFo [44]	CSI	Yes	Channel prediction (time/freq.)	Regression
SpectrumFM [45]	IQ	Yes (CNN/Multihead Self-Attention (MHSA))	modulation classification, wireless technology classification, spectrum sensing, anomaly detection	Classification
WirelessGPT [46]	CSI	Yes (cross-domain embeddings)	Channel estimation, channel prediction, human activity recognition, wireless environment reconstruction	Classification & Regression
6G WavesFM [47]	Spectrograms, CSI (image-like modalities)	Yes	5G NR positioning, MIMO-OFDM channel estimation, human activity sensing, RF signal classification	Classification & Regression
Ours	IQ and CIR	None	wireless technology recognition (2 tasks), LOS detection, ranging error correction	Classification & Regression

masked reconstruction and prediction tasks for modulation classification, wireless technology classification, spectrum sensing, and anomaly detection. Similarly, WirelessGPT [46] is pre-trained on wireless channel datasets using an unsupervised masked autoencoding approach to capture spatio-temporal-frequency correlations, for downstream tasks like channel estimation, channel prediction, and human activity recognition. 6G WavesFM [47] also uses a masked modeling approach on a Vision Transformer (ViT) pre-trained on image-like modalities, including spectrograms and CSI, to perform tasks such as RF signal classification, human activity sensing, 5G positioning, and channel estimation.

Unlike these, our model directly processes raw IQ and CIR timeseries without preprocessing (e.g., spectrogram or CSI conversion), avoiding pre-processing overhead and preserving critical phase and temporal details essential for localization tasks. IQ samples represent complex-valued raw signal amplitudes over time, capturing modulation and phase information, while CIR timeseries reflect multipath channel delays and amplitudes in the time domain; including these heterogeneous data types in a single Transformer-based model enables unified representation learning across diverse signal types. In contrast, previous models rely on uniform data types (e.g., CSI in LWLM and WiFo, or IQ in IQFM and SpectrumFM) for their respective downstream tasks, limiting adaptability in multi-modal wireless scenarios. While WavesFM processes multiple modalities, these are all pre-processed into a uniform "image-like" representation, unlike our approach, which handles raw heterogeneous timeseries. This enables better generalization to unseen technologies and environments with minimal labeled data, outperforming baselines in multi-task efficiency.

To highlight differences, Table 1 compares key aspects, showing our comparison with the other wireless foundation model studies.

III. SYSTEM DESCRIPTION

In our proposed system model, a central large model is first pre-trained on a dataset containing long-range (e.g., Sigfox, LoRA, 802.11ah, Zigbee), short-range (e.g., LTE, WiFi, 5G) radio technologies, and UWB LOS detection and error correction in different environments (e.g., Office, Hallway, lab, Industrial lab (IIOT)). To focus our pre-training on applicable patterns in each downstream task, we excluded the following technologies and environments in pre-training:

1) *Zigbee*, which enables low-power and short-range communication suitable for dense sensor deployments in smart home environments [2].
2) *LTE*, which offers wide-area, high-reliability connectivity required for autonomous vehicle operations and vehicular communication [1].
3) *Localization environments* have different device settings and multi-path patterns. We consider including the office and IIOT in the pre-training.

As depicted in Fig. 1, the foundation model is then fine-tuned to include these unseen technologies and environments using a small set of labeled data samples related to each application. This fine-tuning process ensures that the model can extend its functionality to previously unseen classes using limited data, improving its generalization across different use cases.

We define the pre-trained and fine-tuned models as follows (modified from [48]):

Definition 1 (Pre-trained Model): A *pre-trained model* is a ML model trained on a large-scale dataset to learn general representations and serve as a foundation for downstream tasks.

Definition 2 (Fine-tuned Model): A *fine-tuned model* starts from a pre-trained model and is further trained on a specific labeled dataset to adapt its parameters for a particular task or domain.

A. PROBLEM STATEMENT

In this work, we explore two distinct pre-training strategies (a) supervised and (b) self-supervised learning, to investigate their impact on downstream performance. This subsection is organized into two parts: the first covers the pre-training phase, divided into separate parts for supervised

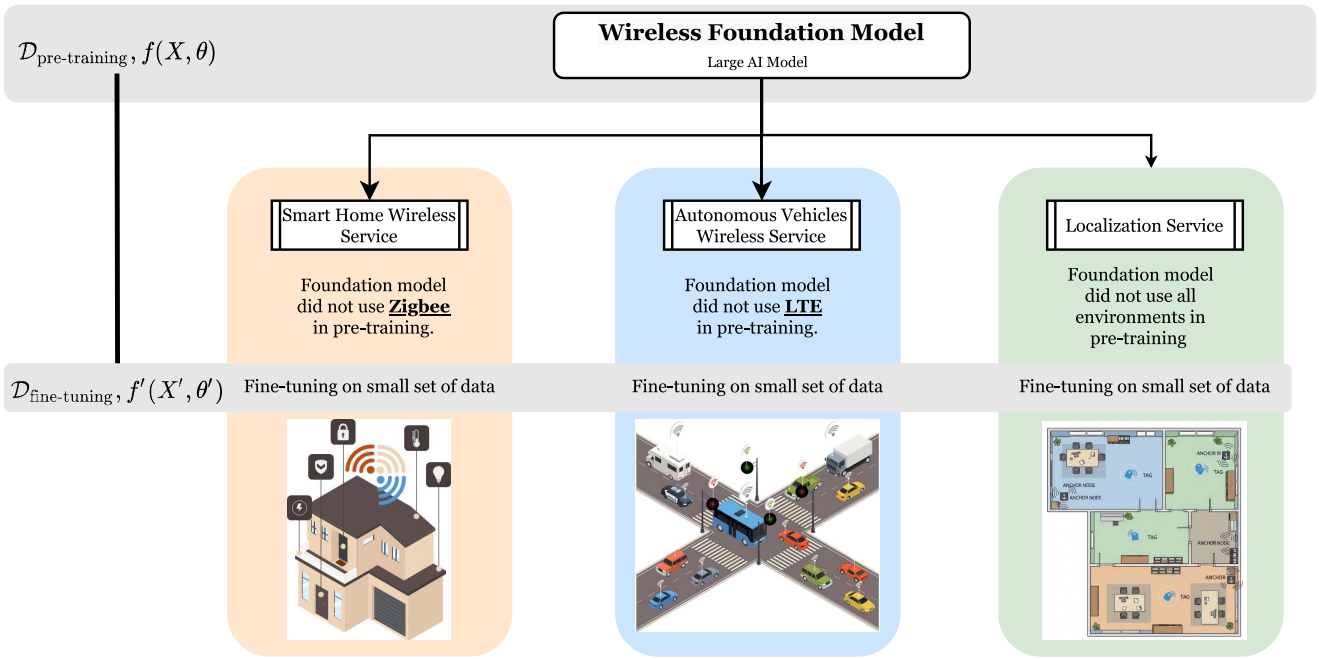


FIGURE 1. System model for wireless technology recognition and localization. Fine-tuning necessity on unseen classes in different wireless applications and generalization necessity in diverse environments for localization applications.

and self-supervised methods; and the second addresses the fine-tuning phase.

1) PRE-TRAINING

We examine two strategies for pre-training the model–supervised and self-supervised learning:

a) Supervised Pre-training: We define the **supervised pre-training dataset** as:

$$\mathcal{D}_{\text{sup}} = \left\{ (X_{i_k}, y_k) \,\middle|\, \begin{array}{l} y_k \in \mathcal{Y}_{\text{sup}} = \{1, \ldots, K\}, \\ i_k \in \{1, \ldots, M_k\} \end{array} \right\}, \tag{1}$$

where X_{i_k} represents the data samples belonging to the k-th class, and this class contains a total of M_k data samples and y_k denotes their respective labels. Here, $\mathcal{Y}_{\text{sup}} = \{1, 2, \ldots, K\}$ is the set of classes present during pre-training and K is the total number of the classes.

The model $f(X;\theta)$ is trained to minimize the classification loss. During this stage, the objective is to learn parameters θ that capture features and patterns within the classes available in the pre-training set and can be expressed as:

$$\theta^{\star} = \arg\min_{\theta} \mathcal{L}_{\text{sup}}\big(\mathcal{D}_{\text{sup}}; \theta\big). \tag{2}$$

$\mathcal{L}_{\text{sup}}$ is the training loss function can be written as [49]:

$$\mathcal{L}_{sup}(\theta) = -\sum_{(X_{i_k}, y_k) \in \mathcal{D}_{sup}} \sum_{k=1}^{K} \mathbb{1}(y_k = k) \times \log\big(f_k\big(X_{i_k}; \theta\big)\big), \tag{3}$$

where $f_k(X_{i_k};\theta)$ denotes the model's predicted probability for class k given input X_{i_k}, and $\mathbb{1}(y_k = k)$ is an indicator function that is 1 if $y_k = k$ and 0 otherwise.

b) Self-supervised Pre-training: The self-supervised component uses self-supervised reconstruction with patch-level masking, where a random subset of timeseries patches is masked during training. This encourages the model to reconstruct missing segments from data and learn robust temporal features.

The self-supervised dataset is defined as:

$$\mathcal{D}_{\text{unsup}} = \{X_n\}_{n=1}^{N}, \tag{4}$$

where each $X_n \in \mathbb{R}^{C \times T}$ is a multichannel timeseries signal. A masking function $\mathcal{M}(\cdot)$ randomly removes a fixed percentage of the non-overlapping patches in X_n, producing a corrupted input:

$$\widetilde{X}_n = \mathcal{M}(X_n). \tag{5}$$

For example, with 50% masking, half of the patches are zeroed out or replaced with a learnable token.

A reconstruction model $f_{\text{rec}}(\cdot;\theta)$ is then trained to recover the original signal from the masked version:

$$\theta^{\star}_{\text{unsup}} = \arg\min_{\theta} \mathcal{L}_{\text{unsup}}\big(\mathcal{D}_{\text{unsup}}; \theta\big), \tag{6}$$

$$\mathcal{L}_{\text{unsup}}(\theta) = \sum_{n=1}^{N} \big\| f_{\text{rec}}\big(\widetilde{X}_n; \theta\big) - X_n \big\|^2, \tag{7}$$

where $\|\cdot\|^2$ denotes the Mean Squared Error (MSE) between the reconstructed and original signals. This encourages the model to learn informative and generalizable representations from raw timeseries data without requiring labels.

2) FINE-TUNING

This phase uses a smaller, task-specific labeled dataset, denoted as $\mathcal{D}_{\text{fine-tuning}}$, which is significantly smaller than the pre-training dataset ($|\mathcal{D}_{\text{fine-tuning}}| \ll |\mathcal{D}_{\text{pre-training}}|$). A principle of this stage is the separation of data to prevent leakage and ensure an unbiased evaluation of the model's generalization:

$$\mathcal{D}_{\text{pre-training}} \cap \mathcal{D}_{\text{fine-tuning}} = \emptyset.$$

During fine-tuning, the pre-trained model $f(X;\theta)$ is transformed into a fine-tuned model $f'(X';\theta')$. To preserve general-purpose representations learned during pre-training, we employ a layer-freezing strategy. The initial layers of the encoder are kept frozen, retaining the foundational knowledge from the pre-trained model. Only the final encoder layer and a newly added task-specific head are unfrozen and trained on the new data.

The overall objective is to find the optimal parameters $\theta'^{\star}$ that minimize the task-specific loss on the fine-tuning dataset:

$$\theta'^{\star} = \arg\min_{\theta'} \mathcal{L}_{\text{fine-tuning}}\big(\mathcal{D}_{\text{fine-tuning}};\theta'\big). \tag{8}$$

We consider two distinct fine-tuning tasks: classification and regression.

a) Classification fine-tuning: For the classification task, the goal is to categorize wireless technologies. The fine-tuning dataset is defined as:

$$\mathcal{D}_{\text{cls-fine-tuning}} = \left\{ \left(X'_{i_k}, y'_k\right) \mid \begin{array}{l} y'_k \in \mathcal{Y}_{\text{fine-tuning}} = \{1,\ldots,K'\}, \\ i_k \in \{1,\ldots,M'_k\} \end{array} \right\}, \tag{9}$$

where X'_{i_k} are samples from the k-th class, y'_k are their labels, M'_k is the number of samples in class k, and K' is the total number of classes. The set of classes extends the pre-training classes K with U new, unseen classes, such that $K' = K + U$ and $\mathcal{Y}_{\text{fine-tuning}} = \mathcal{Y}_{\text{pre-training}} \cup \mathcal{Y}_{\text{unseen}}$. The fine-tuning dataset includes a small number of labeled samples from previously seen classes to support continual recognition.

The model's classification head is a linear layer with a softmax activation. The optimization uses the cross-entropy loss:

$$\mathcal{L}_{\text{cls-fine-tuning}}\big(\theta'\big) = -\sum_{\left(X'_{i_k},y'_k\right)\in\mathcal{D}_{\text{cls}}} \sum_{k=1}^{K'} \mathbb{1}\big(y'_k = k\big) \times \log\big(f'_k(X'_{i_k};\theta')\big), \tag{10}$$

where $f'_k(X'_{i_k};\theta')$ is the model's predicted probability for class k.

b) Regression fine-tuning: For the regression task, the objective is to predict the localization error. The fine-tuning dataset consists of input samples and their target vectors:

$$\mathcal{D}_{\text{reg-fine-tuning}} = \big\{\big(X'_i, \mathbf{z}'_i\big)\big\}_{i=1}^{N'}, \tag{11}$$

where X'_i is the i-th input sample, $\mathbf{z}'_i \in \mathbb{R}^d$ is its label (localization error), and N' is the total number of samples.

For this task, the pre-trained model's head is replaced with a new regression head, a linear layer with one output neuron. The optimization objective is to minimize the MSE between the predicted and true coordinates:

$$\mathcal{L}_{\text{reg-fine-tuning}}\big(\theta'\big) = \frac{1}{N'}\sum_{i=1}^{N'} \|f'\big(X'_i;\theta'\big) - \mathbf{z}'_i\|_2^2, \tag{12}$$

where $f'(X'_i;\theta')$ is the model's predicted coordinate vector for sample X'_i. This approach allows the model to leverage its learned spatial-temporal features to perform precise localization.

B. DATASET DESCRIPTION

The datasets used in this study encompass a much broader range of wireless conditions, including more technologies and capturing conditions than prior studies. We consider two types of technologies: long-range and short-range, each with varying wireless technologies, sampling rates, center frequencies, and different capturing devices.

For the short-range datasets, we consider two publicly accessible datasets. Reference [13], which includes LTE, Wi-Fi, and DVB-T signals captured at a 10 MHz sampling rate using an Anritsu MS 2690A device in different locations in Ghent, Belgium. To ensure a diversity of capturing conditions, the measurements were performed at three distinct locations (north, east, and west sides) inside a large 12×80 m office building. The dataset was compiled from measurements of three different wireless technologies under distinct environmental and traffic conditions. The Wi-Fi signal, operating at 5540 MHz, was captured inside an active office environment with two access points. Measurement points were intentionally varied in distance from the transmitters to create a diverse range of signal strengths. In contrast, the LTE (806 MHz) and DVB-T (482 MHz) signals were recorded in an outdoor-to-indoor scenario, originating from a nearby commercial cell tower and a local TV broadcast tower, respectively. The captured signals also feature authentic background traffic. The Wi-Fi and LTE measurements include live, uncontrolled data from active users, while the DVB-T signal consists of a continuous, stable broadcast stream without the bursty traffic typical of data networks.

Another short-range dataset from [50] provides Wi-Fi (IEEE 802.11ax), LTE, and 5G-NR signals. These were sampled at 20 MHz using a USRP B200mini-i SDR to capture high-frequency technologies at center frequencies of 5.825 GHz (Wi-Fi), 1.8425 GHz (LTE), and 628 MHz (5G-NR), respectively. All captures were performed over the air with active data traffic to better represent real-world signal characteristics. The Wi-Fi and LTE signals were generated within controlled testbeds. These environments used dedicated access points and client devices to create active, known data transmissions. Specific antennas were

TABLE 2. Overview of the datasets, showcasing the diversity of technologies, sampling rate, and capturing devices used for pre-training.

Dataset & Data Type	Technologies	Sampling Rate	Center Frequencies	Capture Device
[13] short-range	LTE, Wi-Fi, DVB-T	10 MHz	806 MHz (LTE), 2412/5540 MHz (Wi-Fi), 482 MHz (DVB-T)	Anritsu MS 2690A
[15] long-range	Sigfox, LoRa, IEEE 802.15.4g, IEEE 802.11ah	1 MHz	863.0 - 868.4 MHz (Sub-GHz)	RTL-SDR dongles
[50] short-range	Wi-Fi (IEEE 802.11ax), LTE, and 5G	20 MHz	5.825 GHz (Wi-Fi), 1.8425 GHz (LTE), 628 MHz (5G-NR)	USRP B210 SDR
[41] UWB CIR	UWB (LOS/NLOS), Ranging error correction	64 MHz PRF (∼1 ns resolution)	Ch. 1-3 (3.5-4.5 GHz), Ch. 5 (6.5 GHz)	Qorvo DW1000 (Wi-PoS & DWM1001-DEV)

employed for these captures: a VERT2450 omni-directional antenna (3 dBi gain) for Wi-Fi and an ANT-LTE-WS-SMA dipole antenna (5.9 dBi gain) for LTE. In contrast, the 5G-NR signal was captured from a live, public commercial mobile network in an uncontrolled outdoor environment.

For the long-range datasets, we consider the dataset from [15], which focuses on sub-GHz technologies, including Sigfox, LoRa, IEEE 802.15.4g, and IEEE 802.11ah, with a sampling rate of 1 MHz and RTL-SDR dongles as capture devices operating in parallel to cover the 868 MHz band. The data was generated in a lab-style test-bed environment using various development boards as transmitters.

To evaluate LOS and NLOS detection, we utilize the comprehensive UWB dataset from [41]. This dataset comprises over 80,000 CIR measurements gathered from three distinct environments: a large industrial warehouse (21,085 samples), a multi-room office (44,894 samples), and a university building (15,208 samples). The data was collected using Qorvo DW1000-based transceivers, specifically the Wi-PoS and DWM1001-DEV platforms, ensuring a variety of hardware configurations. Each measurement is labeled as LOS or NLOS, providing a robust basis for classification tasks. In addition to classification labels, the dataset also includes the ground truth and measured ranges for each sample, enabling the development of models for ranging error correction.

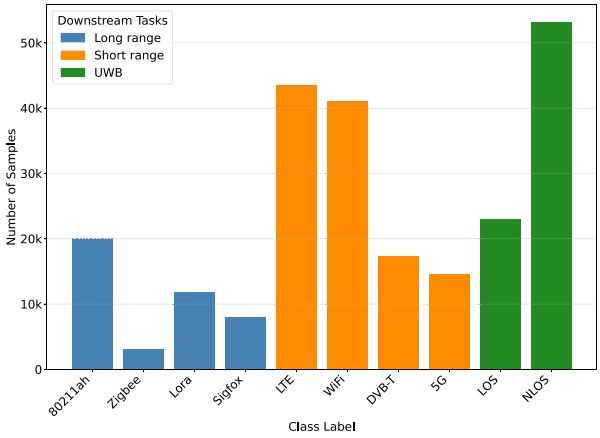


FIGURE 2. Distribution of the dataset used for the proposed foundation model.

A summary of the datasets is provided in Table 2. Furthermore, to provide a more detailed statistical overview, Figure 2 illustrates the total number of samples available for each distinct technology across all combined datasets. In summary, our dataset contains ten distinct classes, captured with varying sampling rates, at diverse locations, and using different devices, resulting in a broad and more realistic representation of wireless environments compared to existing datasets.

IV. METHODOLOGY

This section introduces our proposed method, providing an overview of its structure and functionality. In addition, we briefly describe the baseline methods used for comparison. Finally, we present the data preparation and implementation details, highlighting the main steps and technical considerations involved in the experimental setup.

A. PROPOSED MODEL FOR WIRELESS TECHNOLOGY RECOGNITION

We present an optimized model inspired by the Vanilla Transformer [9] and PatchTST [51] architecture for efficient long-term time series analysis in the context of WTR and localization. This model employs a combination of patch-based segmentation, channel independence, and attention mechanisms, effectively capturing local and global dependencies while minimizing computational costs.

1) MODEL ARCHITECTURE OVERVIEW

The proposed model processes each time series independently, applying a patch-based segmentation approach followed by a Transformer encoder to capture long-term dependencies as depicted in Fig. 3. IQ and CIR samples have real and imaginary parts that are known as two channels of data $x = (x^{(r)}, x^{(i)})$ and are treated independently, without cross-channel interaction, preserving the temporal structure within each channel. This channel-wise independence allows the model to handle large multivariate datasets more efficiently by reducing the complexity involved in attention calculations.

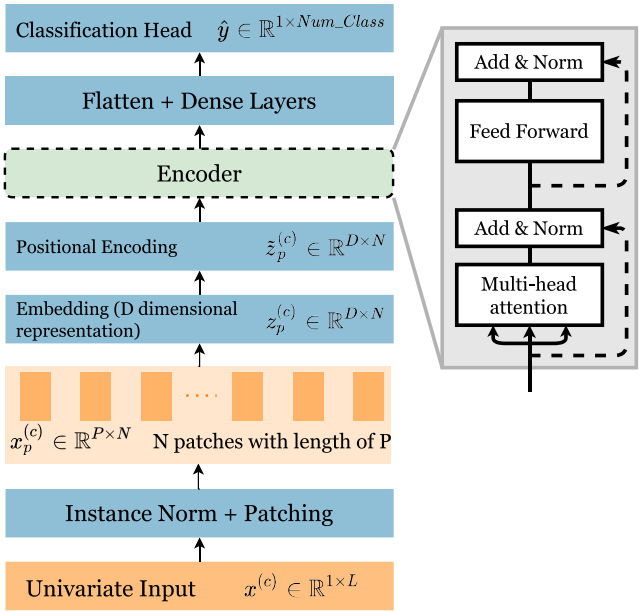


FIGURE 3. Proposed Transformer model with patching for wireless technology classification.

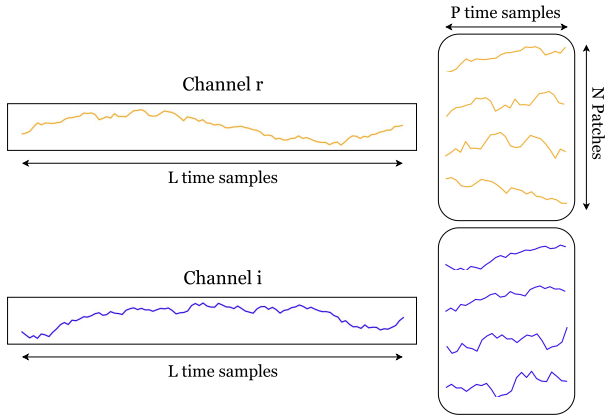


FIGURE 4. Illustration of the patching process applied to IQ timeseries data. Each channel (r and i) consists of *L* time samples and is independently divided into *N* overlapping patches, each of length *P*. These patches are then used as input tokens for the transformer-based model.

2) PATCHING STRATEGY

To address the issue of high computational cost for long time sequences, we segment each time series channel into overlapping patches as illustrated in Fig. 4. Given a time series $x^{(r)}$ or $x^{(i)}$ of length L, each channel is divided into patches of length P with a stride S. This segmentation reduces the input sequence length from L to approximately $\frac{L}{S}$, significantly lowering the computational complexity of the model. A smaller stride increases patch overlap, allowing smoother transitions between patches but also increasing computational load.

For a single channel (either r or i), let the segmented patches be represented by:

$$x_p^{(c)} = \left\{x_t^{(c)}, x_{t+1}^{(c)}, \ldots, x_{t+P-1}^{(c)}\right\} \quad \forall t = \{1, S, 2S, \ldots, L-P+1\}, \tag{13}$$

where $c \in \{r, i\}$. These patches $x_p^{(c)} \in \mathbb{R}^{P\times N}$, where $N = \frac{L}{S}$, serve as the primary input tokens to the Transformer. The patching reduces the effective sequence length from L to N, which lowers the complexity of the attention mechanism from $\mathcal{O}(L^2)$ to $\mathcal{O}(N^2)$.

Each patch $x_p^{(c)}$ is then projected into a latent space using a linear projection matrix $W_p \in \mathbb{R}^{D\times P}$, mapping each patch to a D-dimensional representation [9]:

$$z_p^{(c)} = W_p x_p^{(c)} + b_p, \tag{14}$$

where $b_p \in \mathbb{R}^D$ is the bias term. This transformation effectively captures the local temporal patterns within each patch, and a positional encoding $E_{pos} \in \mathbb{R}^{D\times N}$ is an addition to the projected patches to preserve the temporal sequence order since the Transformer architecture lacks inherent order awareness:

$$\tilde{z}_p^{(c)} = \text{Dropout}\left(z_p^{(c)} + E_{pos}\right), \tag{15}$$

where dropout is applied to the summed embedding to prevent overfitting and improve generalization.

3) MULTI-HEAD SELF-ATTENTION MECHANISM

The positionally encoded patch representations $\tilde{z}_p^{(c)}$, derived from the input patches of channel c, are passed to the Transformer encoder's multi-head self-attention module [9]. This mechanism is designed to capture both local and global dependencies across the sequence.

For each attention head $h = \{1, \ldots, H\}$, where H is the number of attention heads, the attention mechanism computes query, key, and value matrices Q_h, K_h, and V_h as follows:

$$Q_h = W_Q^{(h)} \tilde{z}_p^{(c)}, \tag{16}$$

$$K_h = W_K^{(h)} \tilde{z}_p^{(c)}, \tag{17}$$

$$V_h = W_V^{(h)} \tilde{z}_p^{(c)}, \tag{18}$$

here $\tilde{z}_p^{(c)}$ represents the input sequence of positionally encoded patch embeddings from channel c, and $W_Q^{(h)}, W_K^{(h)}, W_V^{(h)} \in \mathbb{R}^{D\times d_k}$ are learnable weight matrices, and $d_k = D/H$ is the dimension of each attention head.

The attention scores are computed as:

$$\text{Attention}(Q_h, K_h, V_h) = \text{Dropout}\left(\text{Softmax}\left(\frac{Q_h K_h^\top}{\sqrt{d_k}}\right) V_h\right). \tag{19}$$

The scaling factor $\sqrt{d_k}$ is used to avoid large dot products, which can lead to vanishing gradients in the softmax function.

The outputs from all attention heads are concatenated along the feature dimension and linearly transformed using a shared output projection matrix $W_O \in \mathbb{R}^{D\times D}$:

$$\hat{z}_p^{(c)} = W_O[\text{Attention}_1, \ldots, \text{Attention}_H]. \tag{20}$$

where $\hat{z}_p^{(c)} \in \mathbb{R}^{D\times N}$ is the final attended patch representation. This multi-head setup allows the model to attend to information from different representation subspaces simultaneously, improving its capacity to model complex temporal dependencies.

This mechanism enables the model to assign greater importance to the most relevant patches, enhancing its ability to generalize across diverse wireless signals while maintaining interpretability through the learned attention patterns.

4) CLASSIFICATION HEAD

The encoded features from both channels are concatenated and flattened:

$$f = \text{Flatten}\left(\left[\hat{z}_p^{(r)} \| \hat{z}_p^{(i)}\right]\right), \tag{21}$$

which is then passed through a two-layer fully connected network:

$$\hat{y} = W_{c1}\,\sigma(W_{c2} f + b_{c2}) + b_{c1}, \tag{22}$$

where σ is the ReLU activation, and $W_{c1}, W_{c2}, b_{c1}, b_{c2}$ are learnable parameters. The output $\hat{y} \in \mathbb{R}^C$ represents class logits, which are passed through a softmax during inference to generate the predicted class probabilities.

5) REGRESSION HEAD

Similar to the classification head, the encoded features from both channels are first concatenated and flattened as in eq. (21). The combined feature vector is then processed by a two-layer fully connected network:

$$\hat{\mathbf{z}} = W_{r1}, \sigma(W_{r2} f + b_{r2}) + b_{r1}, \tag{23}$$

where σ is the ReLU activation, and $W_{r1}, W_{r2}, b_{r1}, b_{r2}$ are the learnable parameters of the regression head. The output $\hat{\mathbf{z}} \in \mathbb{R}^d$ is the predicted continuous vector, where d is the dimension of the output.

B. BASELINE MODELS (WTR)

We consider four baseline models to evaluate the performance of our proposed method in classifying wireless communication technologies across heterogeneous classes: 802.11ah, Zigbee, LoRa, Sigfox, LTE, WiFi, DVB-T, and 5G.

1) CNN

We implement a CNN model as a baseline based on a research paper in WTR [18]. The architecture is composed of three convolutional layers, followed by two fully connected layers. Each convolutional layer uses ReLU activation, followed by batch normalization, max-pooling, and dropout to reduce overfitting and enhance feature extraction. The fully connected layers further process the flattened features, and the final layer uses a softmax activation for multi-class classification.

This model was originally designed and optimized for a small number of wireless technology classes. In our study, we include this model as a baseline to investigate how such specifically tailored models perform when scaled to more complex classification tasks involving a larger number of heterogeneous technologies.

2) AUTOENCODER + CNN

We employ a convolutional autoencoder to learn a compact latent representation of wireless signal data in a self-supervised manner [16]. The architecture consists of an encoder with two convolutional layers, followed by a fully connected layer. The decoder mirrors this structure using transposed convolutional layers to reconstruct the input from the latent space. Batch normalization and ReLU activations are applied throughout the network to stabilize training and improve learning dynamics. Given that the input consists of 4096 time samples, the dimensionality of the latent space in the autoencoder is set to 1024.

After training the autoencoder using only unlabeled examples, the encoder is frozen and reused as a generic feature extractor. The latent representation serves as input to a CNN model. We used the model explained in IV-B.1.

3) LSTM

Recurrent Neural Networks (RNN), particularly LSTM networks, are widely used for modeling sequential data due to their ability to capture dependencies in time series. Inspired by the architecture presented in [52], we process the input IQ samples using three stacked LSTM layers, followed by fully connected layers and a softmax output for classification. Dropout regularization is applied between layers to prevent overfitting, and ReLU activations are used in the dense layers to ensure non-linearity.

While LSTMs are theoretically well-suited for modeling signal dynamics, our experiments show that this architecture is significantly slower than other models in both pre-training and adaptation. The sequential nature of LSTM computations prevents efficient parallelization, which becomes a bottleneck when dealing with large datasets or high-throughput applications.

4) TRANSFORMER-BASED MODELS

We also include two Transformer-based models in our comparison: the vanilla Transformer and the iTransformer. We do not propose these models, but they serve as state-of-the-art baselines for comparison with our method. These models represent different tokenization strategies for timeseries data: (1) individual time samples as tokens (vanilla Transformer), (2) patches of time samples (our method), and (3) entire channels as tokens (iTransformer). By comparing across these architectures, we aim to highlight how tokenization granularity and model complexity influence performance.

Vanilla Transformer: The vanilla Transformer [9], originally developed for language modeling, has been adapted for timeseries classification by treating each time step as

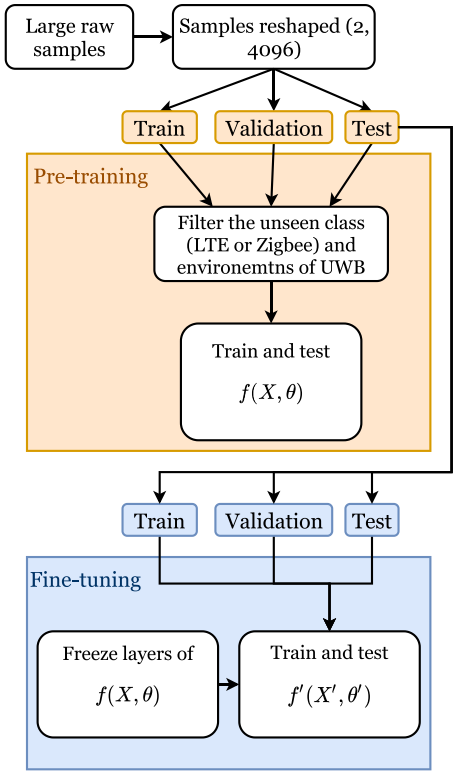


FIGURE 5. Flow-chart depicting the pre-training and fine-tuning strategy.

a separate token. This results in long input sequences and leads to quadratic complexity $\mathcal{O}(N^2)$ in the attention mechanism, significantly increasing computational cost for high-resolution signals. The model applies multi-head self-attention followed by feed-forward layers, layer normalization, and residual connections. While effective in learning sequence dependencies, its deep architecture and token-level granularity make it computationally heavy.

iTransformer: The iTransformer [53] is a more efficient alternative that keeps the core Transformer components unchanged but inverts the input dimensions. Instead of treating time steps as tokens, it treats the channels (or variates) of the time series as tokens by embedding all time points of each channel. The attention mechanism is then applied across these channel tokens to capture multivariate correlations. The feed-forward layers are applied independently to each token to learn non-linear features. This simple yet effective design makes the model less complex while also improving generalization and allowing flexible use of different lookback windows.

C. BASELINE MODEL (UWB)

To evaluate the performance of our proposed method in LOS detection and localization error correction, we use a baseline inspired by the work of [41]. This study, focused on UWB ranging error correction and LOS signal classification, proposes two parallel architectures: a feature-based Deep Neural Network (DNN) that operates on 12 features extracted from the UWB transceiver, and a CNN that learns directly from raw CIR data. The core of their work is an automated Transfer Learning (TL) framework that uses Bayesian optimization to adapt a pre-trained model to a new, unseen environment with smaller samples. We include this baseline to evaluate performance on a practical task where models must overcome data scarcity and generalize across different physical environments and hardware configurations, a common challenge in wireless sensing applications.

TABLE 3. Summary of key parameters and values.

Parameter	Description	value
L	Input sequence length	4096
S	Patch stride	128
P	Patch size	128
c	Number of input channels	2
D	Dimensionality of the encoder layers	128
-	Number of encoder layers	4

D. DATA PREPARATION FOR PRE-TRAINING AND FINE-TUNING

To evaluate the generalization capabilities of our models, we designed structured data handling to simulate unseen class scenarios, as depicted in Fig. 5. First, a large dataset containing IQ and CIR samples was prepared. Each sample was reshaped into a uniform shape of (2, 4096), where '2' represents the real and imaginary channels, as introduced in Section IV-A.1. This reshaping ensures that all data instances maintain a consistent input format, simplifying the pre-training and adaptation process.

To evaluate the model's generalization capability in the presence of an unseen class or environment during pre-training, we excluded specific classes for WTR and environments for UWB from the dataset. In particular, either Zigbee (the class with the fewest samples) or LTE (the class with the most samples) was designated as the unseen class. Zigbee is also a class with a single sampling rate and collected from a single source, while LTE includes two sampling rates and data gathered from multiple sources. Following this exclusion, the dataset was divided into three subsets: training, validation, and test. The model was trained on the training set, tuned using the validation set, and finally evaluated on the test set. Importantly, the test set was kept completely separate from the training data to ensure an unbiased assessment of the model's performance.

The unseen class was included in the test data in the fine-tuning phase. Now containing the previously unseen class, the extended test dataset was again split into new train, validation, and test sets. During fine-tuning, layers of the models were frozen, with only the dense output layer being trained. In our simulations, our proposed model uses the parameters mentioned in Table. 3.

E. IMPLEMENTATION

We implemented the models using PyTorch and TensorFlow, leveraging a single Tesla V100-SXM3-32GB GPU for both training and fine-tuning processes in the IDLab GPULab [54]. For the baseline models and our framework, we constructed custom HDF5 datasets to handle the large-scale, multivariate wireless communication timeseries data efficiently. The HDF5 structure allows for optimized storage, retrieval, and scalability, ensuring efficient data loading during training and evaluation. We configured data loaders with a batch size of 64. For pre-training, we used the AdamW optimizer with a learning rate of 1×10^{-4} and a weight decay of 1×10^{-4}, applied to all model parameters. For fine-tuning, we used AdamW with a lower learning rate of 1×10^{-5} and a weight decay of 1×10^{-5}. In both phases, we employed a ReduceLROnPlateau scheduler that monitors the validation loss and reduces the learning rate by a factor of 0.7 if no improvement is observed for 3 consecutive epochs.

V. RESULTS AND DISCUSSION

This section presents an evaluation of our proposed method and the baseline models under various scenarios. We begin by analyzing the performance of models during the supervised training stage. Next, we discuss the WTR fine-tuning results, where the models are adapted to recognize all classes, even when some were not present during pre-training and UWB fine-tuning results, where our model is adapted for regression and classification tasks considering unseen environment scenarios and compared with the baseline model. Finally, we provide a complexity analysis to assess the computational efficiency of our method in comparison to existing approaches.

A. WTR SUPERVISED RESULTS

In this section, we present the results of supervised pre-training under two main scenarios:

1) *8-class scenario:* assumes that all eight wireless technologies are available during training, which works as the baseline to compare with fine-tuning results.
2) *7-class scenario:* introduces a constraint where one technology is excluded during training to simulate an unseen class. This may be due to the unavailability of class-specific samples or an insufficient number of pre-training samples.

Table 4(a) presents the classification results for the 8-class scenario, where all wireless technologies are included during training. Traditional models such as CNN and LSTM achieve accuracies of 54.89% and 75.62%, respectively. Their lower performance is due to the increased number of classes and the presence of heterogeneous sampling rates, which challenge their ability to generalize across diverse IQ data. Incorporating an autoencoder with CNN improves the performance to 78.01%. Among Transformer-based models, the vanilla Transformer and iTransformer obtain 71.03% and 74.19% accuracies, respectively. In contrast, our proposed method achieves a significantly higher accuracy of 99.47%, demonstrating its strong learning capability in a supervised setting with heterogeneous input data. These results highlight that classical models and even standard Transformer variants struggle when exposed to more diverse classes and varying sampling rates. Their architectural limitations become clear when analyzing the tokenization strategies. The vanilla Transformer, which considers each time sample as a token, creates long sequences from IQ and CIR data. This high complexity makes it very difficult for the self-attention mechanism to capture features. On the other hand, the iTransformer, which treats each channel as a single token, is overly simplistic. This approach flattens all temporal features within a channel, resulting in a loss of sequential information. Our patch-based approach avoids both of these extremes, capturing the underlying features of the data.

TABLE 4. Supervised pre-training results: These supervised results are presented to establish a performance baseline for evaluating the fine-tuning results of our foundation model.

(a) 8-class scenario (all classes seen during training)

Model	Accuracy
CNN	54.89%
LSTM	75.62%
Autoencoder + CNN	78.01%
iTransformer	74.19%
Vanilla Transformer	65.32%
Our method	**99.47%**

(b) 7-class scenario with excluded technologies

Model	LTE Excluded	Zigbee Excluded
CNN	63.97%	57.77%
LSTM	72.15%	73.82%
Autoencoder + CNN	83.53%	79.54%
iTransformer	79.84%	74.43%
Vanilla Transformer	74.76%	66.42%
Our method	**99.69%**	**99.66%**

Table 4(b) presents results for the 7-class scenario, where one class is held out during training. This experiment is designed to evaluate the model's ability to generalize to unseen technologies in the fine-tuning stage. We investigate two cases: (1) excluding LTE, the most dominant class in terms of sample count and sampling rate diversity, and (2) excluding Zigbee, the least represented class with only one sampling rate.

When either LTE or Zigbee is excluded from the training set, most models exhibit improved performance compared to the full 8-class scenario. For example, CNN achieves accuracies of 63.97% and 57.77% when LTE and Zigbee are excluded, respectively. Similarly, LSTM reaches 72.15% accuracy in the LTE-excluded case and 73.82% in the Zigbee-excluded case. Interestingly, LSTM is the only model that shows a

TABLE 5. Fine-tuning results on different downstream tasks.

Model	LTE Excluded		Zigbee Excluded	
	8-class	short-range	8-class	long-range
CNN	39.50%	62.38%	43.45%	46.08%
LSTM	57.41%	52.43%	69.50%	81.36%
Autoencoder + CNN	71.60%	61.87%	79.36%	97.45%
iTransformer	59.10%	52.79%	74.43%	91.10%
Our method (Supervised)	62.79%	86.4%	**98.3%**	99.52%
Our method (Self-supervised)	**93.42%**	**91.14%**	96.84%	**99.99%**

performance drop in the LTE-exclusion setup compared to the 8-class case. For all other models, the performance trend aligns with CNN, showing increased accuracy upon exclusion. This improvement is more prominent when excluding LTE, likely because the LTE dataset introduces higher variability due to its different sampling rates, making it harder for traditional models to generalize. This observation reinforces that the performance of baseline models is fragile and degrades significantly as task complexity increases, which accounts for the large performance gap noted in the 8-class scenario. Despite these variations, our proposed method consistently outperforms all baselines, achieving 99.69% and 99.66% accuracy in the LTE- and Zigbee-excluded scenarios, respectively. These results highlight that our method performs well even when facing eight or seven technologies. The comparison between the two exclusion cases further emphasizes the robustness of our approach, especially in handling variations due to sampling rate.

These results highlight that our method performs well even when facing eight or seven technologies. The comparison between the two exclusion cases further emphasizes the robustness of our approach, especially in handling variations due to sampling rate.

B. WTR FINE-TUNING RESULTS

We now assess the model performance during the fine-tuning stage, where a limited number of labeled samples from the previously unseen class are provided. This step reflects a realistic scenario in which a pre-trained model is adapted to recognize new wireless technologies using only a small amount of labeled data. We focus on **7-class scenario** (two exclusions) earlier: the **LTE-excluded** and **Zigbee-excluded** setups.

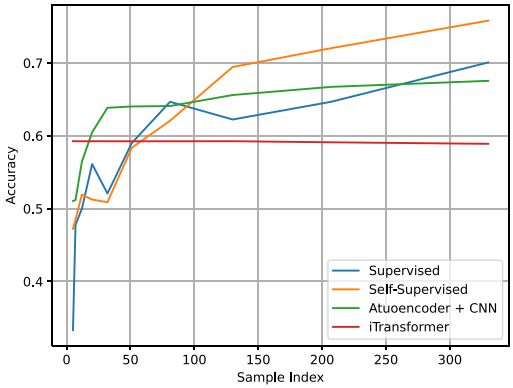
Table 5 summarizes the fine-tuning results on downstream tasks using all available data for fine-tuning. The performance trends highlight the model's sensitivity to the pre-training data, which results from the differing complexity of the excluded classes. LTE represents a "hard" class due to its diversity and spectral overlap with other signals, while Zigbee is an "easy" class with clean, distinct patterns. This difference explains the results. When the "hard" LTE class is excluded, our self-supervised model shows its strength, achieving 93.42% on the 8-class task, far surpassing all other models, including our supervised version (62.79%). This demonstrates that self-supervised learning builds more generalizable representations, which are essential for adapting to a complex, unseen class. Conversely, when the "easy" Zigbee class is excluded, all models are pre-trained on a more complex dataset (containing LTE), providing a robust foundation. Consequently, both our supervised and self-supervised models achieve outstanding accuracy, with the supervised model reaching 98.3% in the 8-class task and the self-supervised model peaking at 99.99% in the long-range task. This illustrates that the diversity of the pre-training data is key to achieving high performance.

These results highlight the strong generalization capability of the self-supervised model, which, despite only receiving labeled data during fine-tuning, performs well across all tasks. Notably, in the LTE-excluded 8-class scenario, the supervised model underperforms (62.79%) compared to its self-supervised counterpart (93.42%), indicating the difficulty of adapting to this specific unseen class using limited supervision. This further emphasizes the effectiveness of self-supervised pre-training in enabling the model to better adapt to new, unseen technologies with minimal labeled data. One possible reason for the lower performance in the LTE-excluded setting is that LTE data in the dataset spans multiple sampling rates and shares spectral characteristics with 5G, making it harder for a purely supervised model to learn distinctive features from a few fine-tuning samples.

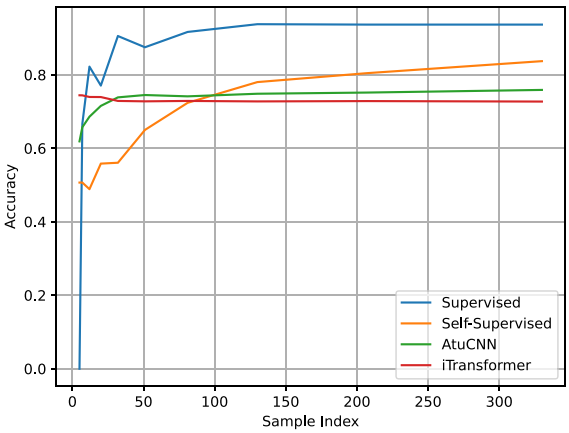
Figures 6(a) and 6(b) illustrate the fine-tuning accuracy of different models on two excluded-class scenarios: LTE and Zigbee. The x-axis indicates the number of samples used for fine-tuning per class, while the y-axis shows the resulting accuracy. For these experiments, the model is fine-tuned and evaluated on the complete 8-class scenario. Among the models compared, we specifically focus on the performance of our supervised and self-supervised solutions in relation to Autoencoder + CNN and iTransformer, which were selected for this analysis due to their superior performance in prior evaluations. In contrast, LSTM and Vanilla Transformer models were excluded from this comparison due to their significantly higher computational demands, which rendered them impractical for large-scale fine-tuning experiments.

In Figure 6(a), the supervised setting achieves competitive performance, especially when the number of available samples is small. However, the self-supervised version outperforms all baselines as the number of samples increases, eventually surpassing 75% accuracy with more than 300 samples. In comparison, Autoencoder + CNN reaches a plateau around 67% accuracy, and the supervised iTransformer converges to slightly below 70%.

In Figure 6(b), the Zigbee-excluded scenario, shows a more pronounced advantage for supervised fine-tuning. The supervised pre-training consistently achieves over 85% accuracy with sufficient data, outperforming all baselines. The self-supervised variant starts at lower accuracy but gradually improves, eventually closing the gap with Autoencoder + CNN. While Autoencoder + CNN maintains steady but lower performance around 74%, the iTransformer again remains below 72%.

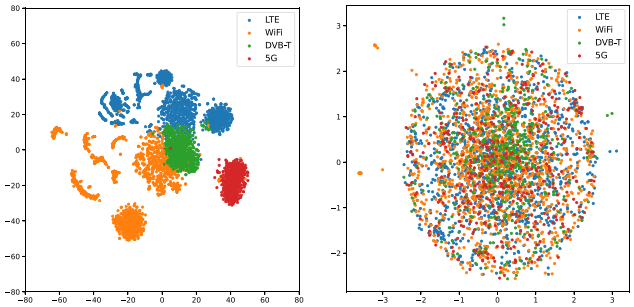


(a) LTE-excluded case.



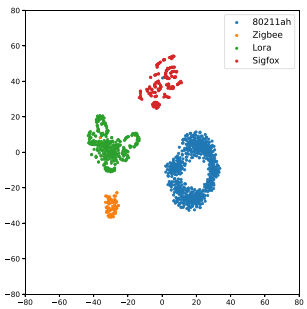
(b) Zigbee-excluded case.

FIGURE 6. **Fine-tuning accuracy when a class was excluded during pre-training. The x-axis shows the number of samples per class used for fine-tuning.**

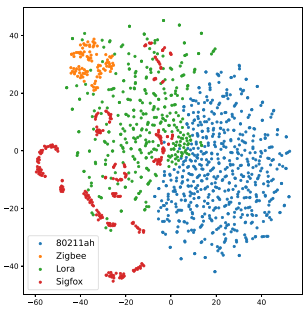


(a1) Fine-tuned Short-range

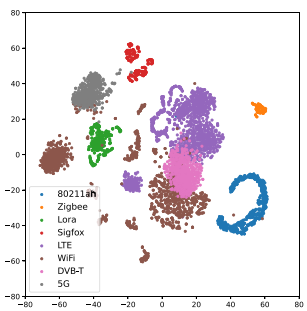
(a2) Untrained Short-range



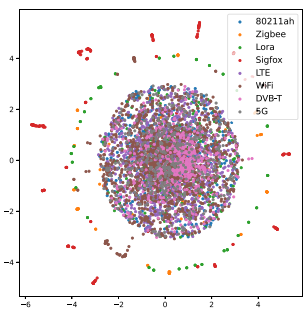
(b1) Fine-tuned Long-range



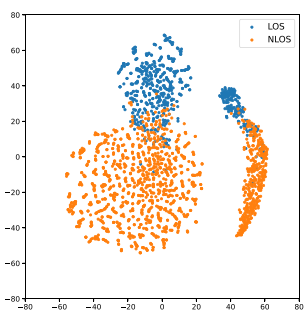
(b2) Untrained Long-range



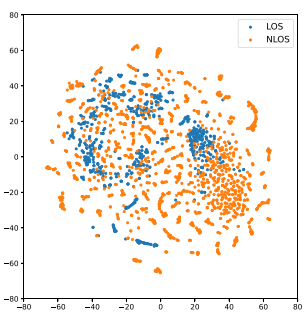
(c1) Fine-tuned 8-class



(c2) Untrained 8-class



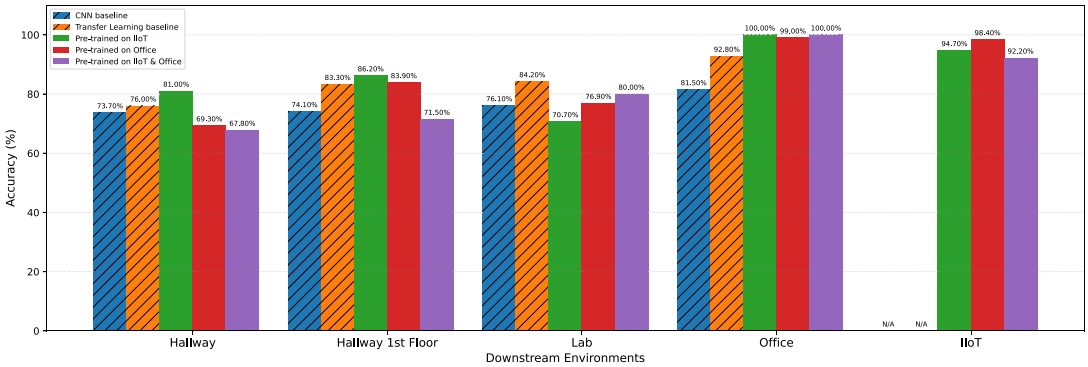
(d1) Fine-tuned UWB (N)LOS



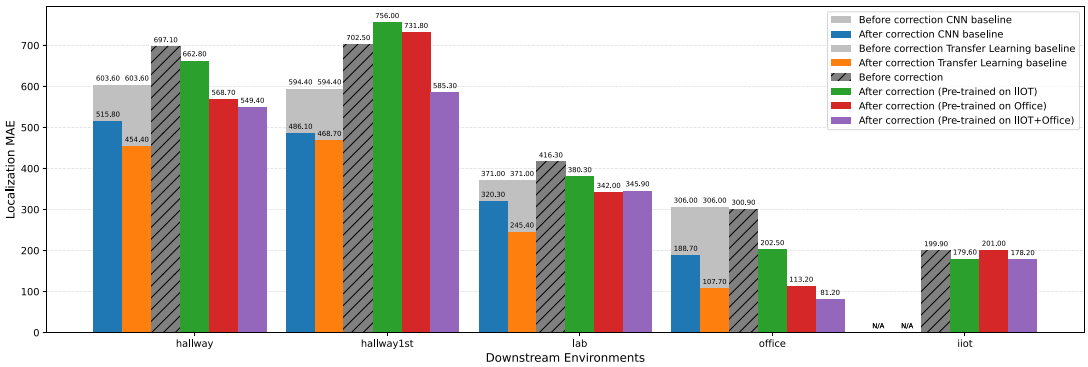
(d2) Untrained UWB (N)LOS

FIGURE 7. **t-SNE visualization of feature embeddings from the self-supervised model (all classes seen) after it was fine-tuned for four distinct downstream tasks. The clear clustering in each plot demonstrates the model's strong ability to learn discriminative features for different classification scenarios.**

Figure 7 provides a t-SNE visualization and comparison of the feature embeddings learned by the self-supervised model before and after being fine-tuned for four distinct WTR downstream classification tasks. These plots illustrate the model's capability to learn different representations for various wireless recognition scenarios. Figures 7(a1) and 7(a2) present the feature distribution for the short-range classification task, where the model produces clusters for LTE, Wi-Fi, DVB-T, and 5G, depicting the separation despite potential spectral overlap. In Figures 7(b1) and 7(b2), for the long-range classification task, the embeddings form distinct clusters for 802.11ah, Zigbee, Lora, and Sigfox are visualized. This clear separation shows better classification performance, likely due to the distinct signal features of these sub-GHz technologies. Figures 7(c1) and 7(c2) depict the most challenging 8-class scenario, which combines all technologies. Even in this complex setting, the model successfully generates obvious clusters for most of the classes, demonstrating its robustness and its ability to learn effective features across a diverse set of wireless technologies. Finally, Figures 7(d1) and 7(d2) display the feature distribution for the UWB LOS and NLOS classification task. The Fine-tuned UWB (N)LOS features show clear separation and clustering between the LOS and NLOS samples, indicating the model's success in distinguishing these classes.



(a) (N)LOS classification accuracy.



(b) Localization Mean Absolute Error (MAE) for the ranging error correction task.

FIGURE 8. **Performance of the fine-tuned foundation model on UWB downstream tasks, compared against a Transfer Learning baseline. The results show (a) classification accuracy and (b) regression MAE across five environments. Our model is evaluated with different pre-training datasets (IIOT, Office, or both). To test generalization, the Hallway, Hallway 1st Floor, and Lab environments were unseen during any pre-training phase.**

These results confirm that both supervised and self-supervised variants of our proposed model exhibit strong generalization to unseen technologies when fine-tuned with limited data. Supervised fine-tuning offers the highest performance in data-rich settings, whereas self-supervised initialization provides a notable advantage in low-data regimes. Autoencoder + CNN, while competitive, consistently underperforms as more samples become available. These findings underline the effectiveness of transformer-based models, particularly when pre-trained using self-supervised objectives, for recognizing unseen wireless technologies.

C. UWB FINE-TUNING RESULTS

In this section, we evaluate the performance of our foundation model when fine-tuned for UWB downstream tasks. We assess its capabilities in two distinct scenarios: (N)LOS classification, a classification task, and ranging error correction, a regression task. The model's performance is compared with a specialized, state-of-the-art transfer learning study designed specifically for these UWB tasks in separate models. To test generalization, the evaluation includes environments that were entirely unseen during the pre-training phase, providing a clear measure of the model's adaptability.

Figure 8 presents the (N)LOS classification accuracy and localization MAE results, respectively. Our model was pre-trained using three different data configurations: the IIoT dataset alone, the Office dataset alone, or both, and then fine-tuned for the downstream environments. These results are compared against the Transfer Learning Baseline [41], which uses raw CIR data. A key distinction is that the baseline model is pre-trained in a supervised manner for a single task before knowledge is transferred. In contrast, our approach is based on a self-supervised foundation model designed to learn general representations applicable to multiple tasks, including WTR and UWB localization. Another difference lies in the size of the input data. The baseline model inputs 150 time samples of raw CIR. Our model, however, takes an input of 4096 time samples, where the 150 CIR samples are included and the other parts are zero-padded. This forces the model to identify and learn from sparse, localized features within a much larger input sequence. We have also added results for a CNN trained from scratch to the figure as a task-specific baseline. This model uses the same architecture as the transfer learning model, providing a direct comparison

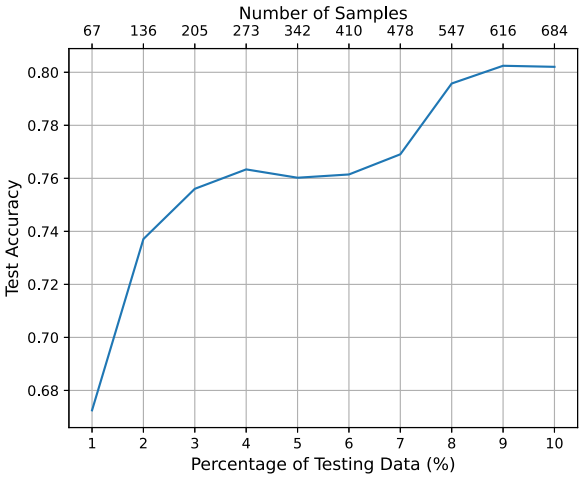
against the transfer learning results and our foundation model results.

In the unseen environments (Hallway, Hallway 1st Floor, and Lab), which were not included in the pre-training, our model demonstrates comparative performance as depicted in Figure 8(a), showing that our foundation model can be a valid alternative for task-specific AI solutions. For instance, when pre-trained on the Office dataset, our model achieves 81.00% accuracy in the Hallway, surpassing the baselines' 76.00% and 73.7%. Similarly, pre-training on the IIoT dataset yields 86.20% accuracy in the Hallway 1st Floor, outperforming the baselines' 83.30% and 74.1%. This highlights that the features learned during our model's self-supervised pre-training are highly transferable to new, unseen physical spaces. For the seen environments (Office and IIoT), our fine-tuned model achieves high accuracy, reaching 100.00% on the Office dataset (when pre-trained on Office) and 98.40% on the IIoT dataset (when pre-trained on IIoT). This significantly exceeds the baselines' 92.80% and 81.5% accuracy on the Office environment. Notably, the baseline shows N/A accuracy for the IIoT environment, as this dataset was used as its source domain for pre-training and was not evaluated as a separate downstream task in the original work.

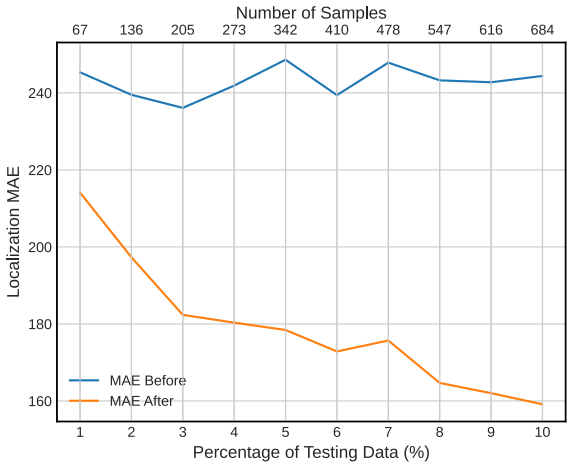
For the UWB ranging error correction task, Figure 8(b) compares localization MAE before and after correction. Hatched gray bars indicate the initial MAE, solid gray bars the MAE before transfer learning and CNN baselines, and orange bars the transfer learning and blue bars the CNN results. Green, red, and purple bars show our fine-tuned foundation model using different pre-training datasets. Differences in reported errors (our model and baselines) arise from random sample selection.

In unseen environments (hallway, hallway1st, lab), the non-foundation model baseline still often achieves lower MAE. For instance, in hallway1st it reduces MAE to 468.70 mm and 486.1 mm, while our best model achieves 585.30 mm. This is expected, as the baseline is optimized with a Bayesian framework for this specific task, whereas our model is a general-purpose foundation model adapted via self-supervised pre-training. Still, our model reduces error in all cases, showing the value of general features learned during pre-training. In seen environments (office, IIOT), our model is competitive or superior. In the office, pre-training on IIOT + Office yields 81.20 mm MAE, outperforming the baselines' 107.70 mm and 188.7 mm. In IIoT, our model reduces the error from 199.90 mm to 178.20 mm, even though the baseline was not evaluated here. These results highlight that fine-tuning a broad, self-supervised model can leverage rich general representations to achieve competitive performance.

To further investigate the data efficiency of our framework, we analyze the model's performance on both UWB tasks while varying the number of samples used for fine-tuning. Figure 9 depicts the performance with data availability, from 1% (67 samples) to 10% (684 samples) of the total data



(a) (N)LOS classification accuracy



(b) Localization MAE before and after error correction

FIGURE 9. **Impact of the number of fine-tuning samples on UWB downstream task performance. These results are the average of the results for all environments, considering that all of them were seen in pre-training.**

from all environments. For the (N)LOS classification task in Figure 9(a), the test accuracy shows a clear positive correlation with the number of fine-tuning samples. The model's accuracy climbs from just under 70% with only 67 samples to over 80% when fine-tuned with 684 samples. This robust improvement highlights the model's ability to effectively use additional data to enhance its classification accuracy when the data includes information from different environments.

A similar trend is observed for the error correction task. Figure 9(b) shows that the corrected error is consistently lower than the initial error, demonstrating the model's benefit regardless of sample size. As the number of fine-tuning samples increases, the localization MAE decreases. With just 2% of the data (136 samples), the model already achieves a significant error reduction from over 200 mm to nearly 180 mm. This demonstrates that even a small, targeted

TABLE 6. Effect of patch size on classification accuracy, training time per epoch, and GPU memory usage.

Patch	Accuracy (%)	Time (s/epoch)	Memory (MB)
8	99.42	201.2	8960
32	99.62	47.54	1900
64	99.79	45.15	1210
128	**99.55**	**44.14**	**980**
256	98.91	44.60	850
512	86.99	44.42	800
1024	75.33	44.30	770

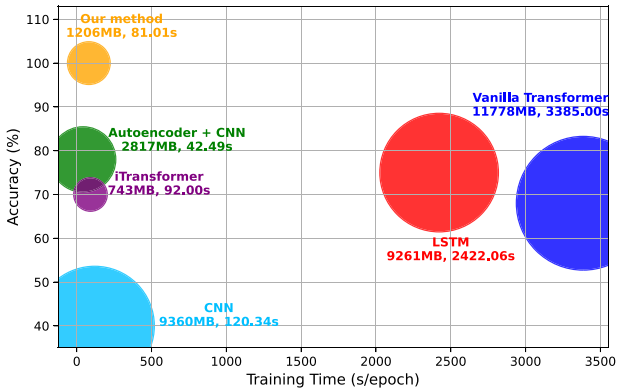


FIGURE 10. Accuracy vs. training time per epoch for different models, with bubble size representing memory usage. Our method achieves the highest accuracy with low training time and memory footprint, outperforming baseline models in all three aspects.

dataset is sufficient to effectively adapt the foundation model for the regression task.

D. COMPLEXITY ANALYSIS

The original Transformer architecture [9] exhibits a computational complexity of $O(N^2)$ in both time and space, where N represents the number of input tokens. In timeseries modeling tasks like ours, N equals the input sequence length L. This quadratic complexity poses a significant bottleneck when dealing with long sequences, as it leads to substantial computational time and memory consumption.

To address this challenge, we used a patching technique that reduces the complexity of the model. By segmenting the input sequence into patches of length P with a stride S, the effective number of tokens becomes $N \approx L/S$. This method changes computational complexity to $O((L/S)^2)$, reducing the quadratic growth.

To further analyze the impact of patching on efficiency, we evaluate the effect of varying patch sizes P on classification accuracy (assuming that the stride is equal to the patch size in this evaluation), training time per epoch, and GPU memory usage, as shown in Table 6. As the patch size increases, we observe a drop in computational cost. For example, increasing P from 8 to 128 reduces the training time by over 75% (from 201.2s to 44.14s) and memory usage by almost 9× (from 8960MB to 980MB), while maintaining a high classification accuracy above 99.5%.

However, very large patch sizes (e.g., $P = 512$ or $P = 1024$) lead to a significant degradation in accuracy, indicating a loss of important temporal resolution. To balance this trade-off between efficiency and performance, we selected a patch size of $P = 128$ as the optimal configuration. It achieves 99.55% accuracy while minimizing both training time and GPU memory usage.

Figure 10 presents the complexity comparison results for the 8-class pre-training scenario, corresponding to the accuracies reported in Table 4(a). Our method achieves the highest accuracy while maintaining significantly lower training time (81.01 seconds per epoch) and minimal GPU memory usage (1206MB), clearly outperforming all baseline models. In contrast, models such as the Vanilla Transformer and LSTM exhibit extremely high computational costs, with training times of 3385s and 2422s per epoch and memory usage of 11778MB and 9261MB, respectively–yet they achieve much lower accuracies of 67.6% and 75.62%.

Even efficient models such as Autoencoder + CNN or iTransformer deliver lower accuracy. This clear performance gap highlights the efficiency and scalability of our patch-based Transformer. By reducing the effective sequence length through patching, our model avoids the $O(L^2)$ time and memory complexity typical of full-length Transformer attention, achieving a more practical $O((L/S)^2)$ complexity with minimal compromise in performance.

VI. CONCLUSION & FUTURE WORKS

This work introduced a patching-based Transformer approach for WTR and localization, aiming to overcome the task-specific models' limitations in generalization to unseen wireless technologies and environments. We showed the lack of adaptability in task-specific models across complex data with different sampling rates and capturing devices. The proposed foundation model achieved better performance than these methods while maintaining a lower computational complexity. Unlike standard Transformer models, which suffer from high training times, our approach offers efficiency comparable to baseline methods. Evaluations were done on a heterogeneous dataset containing eight wireless technology classes with varying sampling rates and UWB data from different challenging environments. With each fine-tuning, the foundation model achieved strong generalization performance, demonstrating its suitability for real-world applications and reusability in a wider range of applications.

Future work will focus on integrating more diverse and real-world datasets to further assess the model's scalability and robustness. Deploying the model in live communication systems will help bridge the gap between experimental validation and practical deployment, ensuring stable operation in dynamic wireless environments. Additionally, inspired by the concept of foundation models [55], we aim to

extend this framework into a unified wireless physical-layer model capable of handling multiple tasks such as interference detection and modulation classification. As a complementary direction, we also plan to explore the use of agentic AI systems to autonomously adapt and optimize wireless technology recognition in evolving environments, further enhancing the model's applicability and resilience.

REFERENCES

[1] M. Noor-A-Rahim, G. G. M. N. Ali, Y. L. Guan, B. Ayalew, P. H. J. Chong, and D. Pesch, "Broadcast performance analysis and improvements of the LTE-V2V autonomous mode at road intersection," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 9359–9369, Oct. 2019.

[2] S. Zhihua, "Design of smart home system based on ZigBee," in *Proc. Int. Conf. Robots Intell. Syst. (ICRIS)*, 2016, pp. 167–170.

[3] M. Cheraghinia et al., "A comprehensive overview on UWB radar: Applications, standards, signal processing techniques, datasets, radio chips, trends and future research directions," *IEEE Commun. Surveys Tuts.*, vol. 27, no. 4, pp. 2283–2324, 4th Quart., 2025.

[4] M. Kulin, T. Kazaz, I. Moerman, and E. De Poorter, "End-to-end learning from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications," *IEEE Access*, vol. 6, pp. 18484–18501, 2018.

[5] S. A. Rajab, W. Balid, M. O. Al Kalaa, and H. H. Refai, "Energy detection and machine learning for the identification of wireless MAC technologies," in *Proc. Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2015, pp. 1440–1446.

[6] F. Hu, B. Chen, and K. Zhu, "Full spectrum sharing in cognitive radio networks toward 5G: A survey," *IEEE Access*, vol. 6, pp. 15754–15776, 2018.

[7] F. Zhu et al., "Wireless large AI model: Shaping the AI-native future of 6G and beyond," 2025, *arXiv:2504.14653*.

[8] M. Goswami, K. Szafer, A. Choudhry, Y. Cai, S. Li, and A. Dubrawski, "MOMENT: A family of open time-series foundation models," in *Proc. 41st Int. Conf. Mach. Learn. (ICML)*, Jul. 2024, pp. 16115–16152. [Online]. Available: https://proceedings.mlr.press/v235/goswami24a.html

[9] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 1–11. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[10] M. Awais et al., "Foundation models defining a new era in vision: A survey and outlook," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 4, pp. 2245–2264, Apr. 2025.

[11] "DWM3000—6.5 & 8.0 GHz ultra-wideband (UWB) module." Qorvo. 2021. Accessed: Sep. 17, 2025. [Online]. Available: https://www.qorvo.com/products/p/DWM3000

[12] T. Ulversoy, "Software defined radio: Challenges and opportunities," *IEEE Commun. Surveys Tuts.*, vol. 12, no. 4, pp. 531–550, 4th Quart., 2010.

[13] W. Liu, M. Kulin, T. Kazaz, A. Shahid, I. Moerman, and E. De Poorter, "Wireless technology recognition based on RSSI distribution at sub-nyquist sampling rate for constrained devices," *Sensors*, vol. 17, no. 9, p. 2081, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/9/2081

[14] J. Lunden, V. Koivunen, A. Huttunen, and H. V. Poor, "Collaborative cyclostationary spectrum sensing for cognitive radio systems," *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4182–4195, Nov. 2009.

[15] J. Fontaine, A. Shahid, R. Elsas, A. Seferagic, I. Moerman, and E. De Poorter, "Multi-band sub-GHz technology recognition on NVIDIA's Jetson nano," in *Proc. IEEE 92nd Veh. Technol. Conf.*, 2020, pp. 1–7.

[16] M. Camelo et al., "A semi-supervised learning approach towards automatic wireless technology recognition," in *Proc. IEEE Int. Symp. Dyn. Spectrum Access Netw. (DySPAN)*, 2019, pp. 1–10.

[17] X. Li, F. Dong, S. Zhang, and W. Guo, "A survey on deep learning techniques in wireless signal recognition," *Wireless Commun. Mobile Comput.*, vol. 2019, no. 1, 2019, Art. no. 5629572. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2019/5629572

[18] A. Shahid et al., "A convolutional neural network approach for classification of LPWAN technologies: Sigfox, LoRA and IEEE 802.15.4g," in *Proc. 16th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON)*, 2019, pp. 1–8.

[19] J. Fontaine et al., "Towards low-complexity wireless technology classification across multiple environments," *Ad Hoc Netw.*, vol. 91, Aug. 2019, Art. no. 101881. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870518309685

[20] W. Kong, Q. Yang, X. Jiao, Y. Niu, and G. Ji, "A transformer-based CTDNN structure for automatic modulation recognition," in *Proc. 7th Int. Conf. Comput. Commun. (ICCC)*, 2021, pp. 159–163.

[21] W. Kong, X. Jiao, Y. Xu, B. Zhang, and Q. Yang, "A transformer-based contrastive semi-supervised learning framework for automatic modulation recognition," *IEEE Trans. Cogn. Commun. Netw.*, vol. 9, no. 4, pp. 950–962, Aug. 2023.

[22] W. Zhang, K. Xue, A. Yao, and Y. Sun, "CTRNet: An automatic modulation recognition based on transformer-CNN neural network," *Electronics*, vol. 13, no. 17, p. 3408, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/17/3408

[23] J. Cai, F. Gan, X. Cao, and W. Liu, "Signal modulation classification based on the transformer network," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 3, pp. 1348–1357, Sep. 2022.

[24] P. Wang, Y. Cheng, B. Dong, R. Hu, and S. Li, "WIR-transformer: Using transformers for wireless interference recognition," *IEEE Wireless Commun. Lett.*, vol. 11, no. 12, pp. 2472–2476, Dec. 2022.

[25] M. Hu, J. Ma, Z. Yang, J. Wang, J. Lu, and Z. Wu, "Feature fusion convolution-aided transformer for automatic modulation recognition," *IEEE Commun. Lett.*, vol. 27, no. 10, pp. 2643–2647, Oct. 2023.

[26] A. Aboulfotouh, E. Mohammed, and H. Abou-Zeid, "6G WavesFM: A foundation model for sensing, communication, and localization," 2025, *arXiv:2504.14100*.

[27] X. Hao, Z. Feng, S. Yang, M. Wang, and L. Jiao, "Automatic modulation classification via meta-learning," *IEEE Internet Things J.*, vol. 10, no. 14, pp. 12276–12292, Jul. 2023.

[28] K. Davaslioglu, S. Boztaş, M. C. Ertem, Y. E. Sagduyu, and E. Ayanoglu, "Self-supervised RF signal representation learning for nextG signal classification with deep learning," *IEEE Wireless Commun. Lett.*, vol. 12, no. 1, pp. 65–69, Jan. 2023.

[29] Y. Tai, Y. Zeng, and Y. Gong, "Self-supervised learning and nearest neighbors for out-of-distribution modulation classification," *IEEE Wireless Commun. Lett.*, vol. 14, no. 5, pp. 1466–1470, May 2025.

[30] H. Zhang, F. Zhou, Q. Wu, and C. Yuen, "FSOS-AMC: Few-shot open-set learning for automatic modulation classification over multipath fading channels," *IEEE Internet Things J.*, vol. 12, no. 12, pp. 18718–18731, Jun. 2025.

[31] M. Du, J. Pan, and D. Bi, "A contrastive learner for automatic modulation classification," *IEEE Trans. Wireless Commun.*, vol. 24, no. 4, pp. 3575–3589, Apr. 2025.

[32] D. Liu, P. Wang, T. Wang, and T. Abdelzaher, "Self-contrastive learning based semi-supervised radio modulation classification," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, 2021, pp. 777–782.

[33] X. Yun and X. Zhou, "Exploring self-supervised learning for radio signal recognition," in *Proc. IEEE 23rd Int. Conf. High Perform. Comput. Commun., 7th Int. Conf. Data Sci. Syst., 19th Int. Conf. Smart City, 7th Int. Conf. Depend. Sens., Cloud Big Data Syst. Appl. (HPCC/DSS/SmartCity/DependSys)*, 2021, pp. 2425–2430.

[34] H. Wymeersch, S. Marano, W. M. Gifford, and M. Z. Win, "A machine learning approach to ranging error mitigation for UWB localization," *IEEE Trans. Commun.*, vol. 60, no. 6, pp. 1719–1728, Jun. 2012.

[35] J. Fontaine, M. Ridolfi, B. Van Herbruggen, A. Shahid, and E. De Poorter, "Edge inference for UWB ranging error correction using autoencoders," *IEEE Access*, vol. 8, pp. 139143–139155, 2020.

[36] C. Jiang, S. Chen, Y. Chen, D. Liu, and Y. Bo, "An UWB channel impulse response de-noising method for NLOS/LOS classification boosting," *IEEE Commun. Lett.*, vol. 24, no. 11, pp. 2513–2517, Nov. 2020.

[37] Q. Liu, Z. Yin, Y. Zhao, Z. Wu, and M. Wu, "UWB LOS/NLOS identification in multiple indoor environments using deep learning methods," *Phys. Commun.*, vol. 52, Jun. 2022, Art. no. 101695. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1874490722000544

[38] C. Jiang, J. Shen, S. Chen, Y. Chen, D. Liu, and Y. Bo, "UWB NLOS/LOS classification using deep learning method," *IEEE Commun. Lett.*, vol. 24, no. 10, pp. 2226–2230, Oct. 2020.

[39] V. C. S. R. Rayavarapu and A. Mahapatro, "NLOS identification and mitigation in uwb positioning with bagging-based ensembled classifiers," *Ann. Telecommun.*, vol. 77, no. 5, pp. 267–280, 2022.

[40] B. Van Herbruggen, J. Fontaine, and E. D. Poorter, "Anchor pair selection for error correction in time difference of arrival (TDoA) ultra wideband (UWB) positioning systems," in *Proc. Int. Conf. Indoor Position. Indoor Navig. (IPIN)*, 2021, pp. 1–8.

[41] J. Fontaine et al., "Transfer learning for UWB error correction and (N)LOS classification in multiple environments," *IEEE Internet Things J.*, vol. 11, no. 3, pp. 4085–4101, Feb. 2024.

[42] G. Pan, K. Huang, H. Chen, S. Zhang, C. Häger, and H. Wymeersch, "Large wireless localization model (LWLM): A foundation model for positioning in 6G networks," 2025, *arXiv:2505.10134*.

[43] O. Mashaal and H. Abou-Zeid, "IQFM a wireless foundational model for I/Q streams in AI-native 6G," 2025, *arXiv:2506.06718*.

[44] B. Liu, S. Gao, X. Liu, X. Cheng, and L. Yang, "WiFo: Wireless foundation model for channel prediction," *Sci. China Inf. Sci.*, vol. 68, no. 6, 2025, Art. no. 162302.

[45] F. Zhou et al., "SpectrumFM: A foundation model for intelligent spectrum management," 2025, *arXiv:2505.06256*.

[46] T. Yang et al., "WirelessGPT: A generative pre-trained multi-task learning framework for wireless communication," *IEEE Netw.*, vol. 39, no. 5, pp. 58–65, Sep. 2025.

[47] A. Aboulfotouh, E. Mohammed, and H. Abou-Zeid, "6G WavesFM: A foundation model for sensing, communication, and localization," *IEEE Open J. Commun. Soc.*, vol. 6, pp. 6792–6807, 2025.

[48] X. Han et al., "Pre-trained models: Past, present and future," *AI Open*, vol. 2, pp. 225–250, Feb. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651021000231

[49] A. Mao, M. Mohri, and Y. Zhong, "Cross-entropy loss functions: Theoretical analysis and applications," in *Proc. 40th Int. Conf. Mach. Learn.*, Jul. 2023, pp. 23803–23828. [Online]. Available: https://proceedings.mlr.press/v202/mao23b.html

[50] S. Subray, "Real-world wireless communication dataset: IEEE 802.11ax, LTE, and 5G-NR signals." 2023. [Online]. Available: https://dx.doi.org/10.21227/b82j-sy57

[51] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," 2023, *arXiv:2211.14730*.

[52] A. Emam, M. Shalaby, M. A. Aboelazm, H. E. A. Bakr, and H. A. Mansour, "A comparative study between CNN, LSTM, and CLDNN models in the context of radio modulation classification," in *Proc. 12th Int. Conf. Electr. Eng. (ICEENG)*, 2020, pp. 190–195.

[53] Y. Liu et al., "iTransformer: Inverted transformers are effective for time series forecasting," in *Proc. 12th Int. Conf. Learn. Represent.*, 2024, pp. 1–25. [Online]. Available: https://openreview.net/forum?id=JePfAI8fah

[54] "IDLab GPU lab." Accessed: Nov. 19, 2024. https://idlab.technology/infrastructure/gpulab/

[55] J. Fontaine, A. Shahid, and E. De Poorter, "Towards a wireless physical-layer foundation model: Challenges and strategies," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, 2024, pp. 1–7.