Towards a Unified Few-Shot Learning Evaluation Framework for RF Fingerprinting

Sai Shi*†, Vahid Mahzoon*†, Xuyu Wang‡, Shiwen Mao§, Jie Wu†, Slobodan Vucetic†
† *Computer and Information Science Department, Temple University*, Philadelphia, USA
‡ *Knight Foundation School of Computing and Information Sciences, Florida International University*, Miami, USA
§ *Department of Electrical and Computer Engineering, Auburn University*, Auburn, USA

***Abstract*—Radio frequency (RF) fingerprinting is a technique used to identify a wireless device based on its specific and unique hardware characteristics. In recent years, deep learning has been utilized for RF fingerprinting due to its superiority in feature extraction and higher classification accuracy. However, one major challenge of deep learning-based RF fingerprinting is that wireless signals are highly sensitive to environmental conditions, causing the device fingerprints captured in one environment to not transfer well to another. Hence, deep learning models are found to perform well in the same condition but lose their ability to classify devices in the new condition. In this paper, we examine three transfer learning techniques to mitigate the domain shift problem in RF fingerprinting and compare them with two well-defined baselines. The three RF fingerprinting datasets under various scenarios are examined to explore how environmental factors impact RF fingerprinting, such as transmitter locations, transmitter distance, and device configurations. We identify the most challenging scenarios and study how environmental factors lead to model deterioration through t-SNE visualization.**

***Index Terms*—RF fingerprinting, deep learning, transfer learning, domain adaptation, meta-learning**

I. Introduction

Wireless technology has transformed how we communicate, access data, and interact with the world. Wireless communication transmits information from one point (a transmitter) to another (a receiver) without using any physical connections, such as wires or cables. Each wireless device introduces variations in transmitted signals due to hardware imperfections, creating unique characteristics in the physical layer, which can be used as fingerprints or signatures for the device [1]. RF fingerprinting [2], [3] is the identification of individual wireless devices based on these characteristics of the physical layer without relying on standard identifiers such as MAC or IP addresses. Given one or more wireless devices, the task of RF fingerprinting is to uniquely classify each of them based on the emitted signals captured by a receiver. RF fingerprinting improves security, identification, and monitoring in wireless communication systems because hardware-based fingerprints are difficult to disguise and robust against software modifications [4]. Hence, RF fingerprinting is actively investigated for its potential in applications such as intrusion detection [5], [6], device authentication [7], and user tracking [8].

* Equal contribution.
Corresponding authors: Slobodan Vucetic (vucetic@temple.edu), Shiwen Mao (smao@auburn.edu)

Traditional RF fingerprinting methods [1], [2] rely on handcrafted physical-layer features, which are manually selected based on domain expertise. These methods may not capture all the complexities of real-world RF signals and are not sufficiently robust. Deep learning [9], [10] has achieved success in solving many big-data problems due to its superior ability to automatically extract complex features without domain knowledge. Deep learning methods are adept at benefiting from large datasets and could classify a large number of devices [11]. Hence, deep learning algorithms have been used in RF fingerprinting and have achieved state-of-the-art accuracy [12]–[14]. Evaluation of deep learning methods in RF fingerprinting often assumes that the environmental conditions are stable. Under this assumption, a labeled set for RF fingerprinting can be randomly split into training and test sets and a deep learning model can be trained on the training set and evaluated on the test set.

Despite the success in applying deep learning models for RF fingerprinting under stable conditions, one major challenge is that RF fingerprinting is sensitive to changes in wireless environments or deployment settings [1], [15], [16]. This leads to performance deterioration when deploying a deep learning model under a new condition unseen during training. These environmental factors are often dynamic, unpredictable, and difficult to control due to the complex nature of wireless systems and devices. In this work, we specifically focus on changes in channel conditions. Specifically, channel conditions [17] relate to changes in the operating environment that affect signal propagation, such as whether the devices are deployed in an indoor or outdoor environment, the distance between the transceiver and the receiver, and the time that data are collected. When the above factors change conditions significantly, a deep learning model trained on historical data could fail to classify the same set of devices under the new condition [15], [18].

Transfer Learning (TL) [19]–[22] addresses domain shifts by reusing or fine-tuning a model trained on one domain (the source), which has plentiful labeled data, for use in a different but related target domain, which may have few or no labels. Instead of training from scratch on the target data, whose distribution often differs due to factors like channel conditions, TL leverages knowledge encoded in a pre-trained source model to improve efficiency and performance. Formally, given

979-8-3315-0898-2/25/$31.00 ©2025 IEEE

a large labeled source dataset and a smaller (or unlabeled) target dataset, TL seeks a predictor that achieves high accuracy on both source and target data. In RF fingerprinting, TL has been applied to mitigate performance degradation caused by environmental variations, such as changes in deployment location or time, by adapting a source-trained model to new channel conditions [18], [23]–[25].

In our work, we focus on a typical setting of TL where very few labeled data samples are available for the target task since labeling RF signals for device fingerprinting is labor intensive. This setting is often called few-shot learning (FSL) [26]. FSL enables a model to quickly adapt to new tasks or classes with minimal data, leveraging prior knowledge learned from other related tasks. Recently, FSL has attracted increasing attention in RF fingerprinting research [27]–[30].

One type of FSL algorithm is meta-learning [31], also known as "learning to learn". It has been widely used to solve FSL problems because it enables a deep-learning model to generalize across many tasks with fast adaptation by identifying patterns in how learning occurs. Meta-learning approaches can be categorized as optimization-based [32], metric-based [33], and model-based [34]. In this work, we focus on metric-based meta-learning methods for several reasons: (1) they achieve strong performance in FSL without requiring complex training procedures; (2) they offer computational efficiency through their simple implementation; and (3) their distance-based classification mechanism is particularly suitable for RF fingerprinting, where signal similarity in the feature space often correlates with device identity. Specifically, we implemented two representative fundamental approaches in metric-based meta-learning: Prototypical Network (PTN) [35] and Matching Network [36]. PTN uses class prototypes for classification, while Matching Networks leverage attention mechanisms to compare query samples with support examples. They both provide insights on how to leverage the embedding space for device identification.

Domain adaptation (DA) [37] addresses the few-shot learning challenge from a different perspective by focusing on the domain-shift problem—where marginal and conditional distributions change between source and target datasets while the underlying task remains the same. This domain shift is relevant in RF fingerprinting, where environmental factors such as channel conditions can significantly alter signal characteristics while the classification task remains unchanged. DA focuses on aligning feature distributions between the source and target domains to minimize the domain shift. Depending on how feature alignment is achieved, DA approaches can be categorized [38] as divergence-based, adversarial-based, and reconstruction-based. In this work, we focus on divergence-based DA methods because of three reasons: (1) they offer a strong theoretical foundation on transferability across domains; (2) they provide transparent optimization objectives through explicit distance metrics; and (3) they achieve competitive performance with significantly lower computational requirements compared to adversarial approaches. The core idea of divergence-based DA is to minimize a statistical distance between source and target distributions so that the model can perform well in both domains. Examples of such distance metrics are Maximum Mean Discrepancy (MMD) [39], Kullback–Leibler divergence, and Wasserstein distance [40]. In this work, we use MMD as the distance metric to guide the optimization process since it provides flexibility in measuring distances in high-dimensional feature spaces using kernel functions.

However, existing TL studies in RF fingerprinting often employ different datasets, data-split strategies, and baselines—many compare only to zero-shot or fine-tune baselines—making it difficult to draw consistent or generalizable conclusions. Meta-learning works speak in terms of support/query sets instead of standard training/test splits, so they rarely evaluate alongside domain-adaptation approaches under a unified protocol.

We therefore ask: How do metric-based meta-learning (PTN, MN) and divergence-based DA (MMD) methods compare—alongside zero-shot and fine-tune baselines—in a consistent few-shot evaluation framework for RF fingerprinting? To answer this, we implement all four approaches, fix datasets, data splits, and baselines, and benchmark them on three real-world RF datasets under six domain-shift scenarios.

In summary, our work makes the following contributions:

- Unified evaluation framework. We design a few-shot TL protocol for RF fingerprinting that fixes datasets, data splits, and baselines (zero-shot and fine-tune), enabling apples-to-apples comparisons across methods.
- Empirical benchmarking. We implement and compare Prototypical Networks, Matching Networks, MMD-based domain adaptation, and fine-tuning on three public RF datasets under six realistic domain shifts.
- Representation analysis. We visualize learned embeddings via t-SNE and quantify clustering quality using silhouette scores to reveal how each method mitigates domain discrepancy.
- Practical guidelines. We distill recommendations on method choice, shot-count, and evaluation design to help future RF fingerprinting TL studies achieve fair and reproducible results.

II. Related Work

A. Deep Learning an RF Fingerprinting

Deep learning methods have been widely used in RF fingerprinting. Wu et al. [12] used LSTM to capture temporal dynamics in baseband signals and classify six devices, Cekic et al. [13] used complex-valued Deep Neural Networks to process baseband signals, and Shen et al. [14] used CNN with deep metric learning to distinguish between 60 LoRA devices. Jian et al. [11] demonstrated that CNN can scale to hundreds of transmitters.

B. Transfer Learning for RF Fingerprinting

Despite the success of deep learning methods in RF fingerprinting, many experimental studies [15], [16], [22] have shown that domain shifts are common in the wireless

environment and deep learning methods are very sensitive to environmental changes. In these cross-domain settings, transfer learning [18], [23], [24] has been used to enable knowledge transfer in RF fingerprinting between historical and new conditions. Tian et al. [23] fine-tuned a ConvMixer network under varying SNR conditions, achieving improved accuracy on 10 devices. Kuzdeba et al. [24] fine-tuned a CNN across different numbers of devices. Both methods compare their performance only with zero-shot baselines, leaving open how fine-tuning performs relative to other TL paradigms.

Domain Adaptation (DA) transfer learning techniques align feature distributions between source and target domains, often when target labels are scarce. Elmaghbub et al.'s [29] adversarial-based domain adaptation algorithm focuses on a single environmental change scenario and is evaluated solely against unsupervised baselines. Yang et al. [41] applied KL-based alignment with adaptive pseudo-labeling for WiFi transmitter shifts, demonstrating improved t-SNE separation but without comparing with meta-learning benchmarks.

Few-shot learning (FSL) enables rapid adaptation to new tasks using only a few labeled target examples and have been popular in RF fingerprinting research [25], [27]–[30], [41], [42], adopting support/query training episodes. Mackey et al. [25] applied prototypical networks (PTN) on three RF fingerprinting datasets, but only compared them with zero-shot baselines. Zhao et al. [27] enhanced PTN with data augmentation, but they compared it only with weak and zero-shot baselines. Wang et al. [28] introduced time-frequency mask augmentation to improve cross-domain accuracy but tested only against PTN and fine-tuning. Zhang et al. [42] proposed a meta-learning scheme using siamese network, evaluating a single environmental shift and excluding other meta-learning baselines.

C. Research Gaps

While prior studies demonstrate the potential of TL and meta-learning to mitigate domain shifts in RF fingerprinting, they have several limitations. They tend to evaluate on limited datasets or scenarios and often compare with weak or inconsistent baselines. Some studies focus on specific factors, such as data collection time or receiver impact, without analyzing a broader impact of diverse environmental factors. Others propose methods like data augmentation or domain alignment but lack deeper analysis into how these methods mitigate domain shift. Terminology mismatches and varied K-shot conventions further impair reproducibility. Importantly, no prior work provides an apples-to-apples comparison of zero-shot, fine-tuning, metric-based meta-learning, and divergence-based DA within a unified few-shot evaluation framework. Motivated by these gaps, we aim to establish a standardized and reproducible FSL evaluation framework for RF fingerprinting.

III. Methodology

A. Problem Formulation

RF fingerprinting is a classification problem where the goal is to identify the transmitter of a received signal based on transmitter-specific characteristics. A received signal, denoted as X, is influenced not only by the transmitter identity Y but also by environmental conditions during the data collection process represented by Z. Thus, the received signal can be expressed as a function $X = g(Y, Z)$. The objective is to approximate the inverse mapping $Y = f(X)$ directly from data, without explicit access to Z. When Z remains constant between training and testing, this problem is not exceedingly challenging and reduces to a standard classification task. However, in practical scenarios, Z is dynamic, and variations in conditions lead to a distribution shift between training and testing environments, making domain shift a fundamental challenge.

In transfer learning, the **source domain** $\mathcal{D}_S = \{X_S, P_S(X, Y)\}$, where X_S is the input space and $P_S(x, y)$ is the joint probability distribution over inputs $x \in X_S$ and labels $y \in \mathcal{Y}$, provides labeled signals under specific conditions, while the **target domain** $\mathcal{D}_T = \{X_T, P_T(X, Y)\}$ represents a different environment with a shifted signal distribution which has limited labeled data. Given a large labeled dataset from the source task $\{(x_S^i, y_S^i)\}_{i=1}^{N_S}$, and a small labeled set from the target task $\{(x_T^j, y_T^j)\}_{j=1}^{N_T}$, the goal is to learn a predictor $f_\theta(X)$ that maximizes the accuracy over the target domain while maintaining high accuracy over source domain. Often, it is also preferred that the predictor retains its accuracy on the source domain.

We explore three approaches: Prototypical Networks (PTN) and Matching Network (MN), two meta-learning methods for learning robust representations, and Maximum Mean Discrepancy (MMD), a domain adaptation technique to align distributions between the source and target domains. We explain in detail each stage of our methodology in the next subsections.

B. Evaluation Framework

As mentioned in the introduction, literature on transfer learning in RF fingerprinting has issues with benchmarking. One potential reason is that model training in meta-learning is only based on source data, so it reminds of a zero-shot scenario. However, meta-learning also requires access to labeled examples in target data to enable inference. Despite not fine-tuning a model on target data, meta-learning still belongs to a class of FSL algorithms and could be compared to other FSL algorithms. To clarify this issue, we propose a unified evaluation framework that facilitates comparisons across different FSL algorithms.

In both source and target domains, we partition the data into training and test subsets. In FSL, the models are trained using training examples from both source and target domains. For target domain, we assume only K labeled examples are available per class. The models are evaluated separately on test data from source and target domains. When K is small, the observed accuracy is sensitive to randomness of that selection. That is why we repeat each experiment 100 times, each on a different random selection of target training data.

Algorithm 1 Few-Shot Task Generation For Meta-Learning

Input: Dataset D, Labels $\mathcal{Y}$, C-classes, K-supports, N-queries
Output: Few-shot tasks $\mathcal{T} = \{\tau_i\}_{i=1}^{T}$, $\tau_i = (\mathcal{S}_i, \mathcal{Q}_i)$
for $i = 1$ **to** T **do**
    $\mathcal{C} \leftarrow$ Sample a subset of C classes from $\mathcal{Y}$
    Initialize $\mathcal{S}_i, \mathcal{Q}_i \leftarrow \emptyset$
    for each class $c \in \mathcal{C}$ **do**
        $D_c \leftarrow$ All instances of class c in D
        $\mathcal{S}_i \leftarrow$ Sample K examples from D_c
        $\mathcal{Q}_i \leftarrow$ Sample N examples from $D_c \setminus \mathcal{S}_i$
    end for
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{(\mathcal{S}_i, \mathcal{Q}_i)\}$
end for

C. Preprocessing

In an RF transmission system, the modulated waveform is represented as the I component aligned with a reference carrier signal and the Q component phase-shifted by 90 degrees. Each raw I/Q signal is represented as (I) and (Q) vectors of length L. To provide an input to RF fingerprinting neural network, we preprocess the signal following the approach used in [15]: The raw I/Q signal is segmented into fixed-length, non-overlapping frames consisting of W consecutive I/Q samples. Each frame is normalized to have unit energy to standardize signal amplitude across all frames.

D. CNN-based Feature Extraction

We used a CNN-based deep learning model described in [15] as the backbone for all transfer learning methods, because they are good in capturing the temporal relationship from the input frames. The model consists of six convolutional (Conv) layers, each followed by batch normalization, LeakyReLU activation, and max pooling. The final convolutional output is passed through two fully connected (FC) layers for classification, followed by a softmax layer that produces probability distributions over transmitter classes.

E. Meta-learning Methodology

Few-Shot Task Generation For Meta-Learning Algorithms

Training and evaluation of meta-learning algorithms for FSL is typically based on the process of task generation [32], [35]. In this process, a model is trained on a series of small classification tasks, each constructed from a subset of training data. Each task consists of a support set and a query set, each containing a few labeled examples per class for a subset of all classes, as detailed in Algorithm 1.

Prototypical Network

Prototypical Network (PTN) [35], [43] is a FSL meta-learning algorithm where the goal is to generalize across tasks by learning robust embeddings for each class. Algorithm 2 outlines the pseudo-code for training PTN. The algorithm takes as input the labeled source training data. Then, it first generates a set of *few-shot tasks* $\mathcal{T}$ from source training data using Algorithm 1. Initially, the CNN model f_θ is randomly initialized. In each iteration, a new task τ_i is sampled and all its support and query examples are processed to extract their embeddings. These embeddings are captured from the output of the last FC layer, as this representation demonstrated good performance during our empirical evaluation.

Once the embeddings are computed, the algorithm constructs prototype (proto_c) for each class by averaging the embedding of the K support examples belonging to that class Next, the query examples are classified by computing their distance to the prototypes using a distance metric δ. Softmax operation is then applied to convert these distances into class probabilities. The *cross-entropy loss* is then computed based on the predicted probabilities and the true labels of examples in query set. Finally, the CNN parameters θ are updated using gradient descent with a learning rate β. This methodology is visualized in Figure 1.

Given a PTN trained on source domain, it can be applied on target data, assuming each class in target data has K labeled examples. PTN algorithm then construct class-dependent prototypes by averaging the embeddings of target labeled examples of each class. Target test examples are classified by measuring their distance to the generated prototypes, typically using Euclidean distance, and assigning the label of the nearest prototype.

Algorithm 2 Training of Prototypical Network

Input: Source training data (X_S, Y_S), C, K, N, distance metric δ
Output: Optimized CNN model f_θ
$\mathcal{T} = \{\tau_i\}_{i=1}^{T} \leftarrow$ Alg. 1 (X_S,Y_S,C, K, N)
Initialize θ randomly for CNN model f_θ
while training not converged **do**
    Sample episode $\tau_i = (\mathcal{S}_i, \mathcal{Q}_i)$
    $E_S = \{e_i = f_\theta(x_i) | x_i \in S_i\}$ ▷ Support set embedding
    $E_Q = \{e_i = f_\theta(x_i) | x_i \in Q_i\}$ ▷ Query set embedding
    for each class $c \in \{1, \ldots, N\}$ **do**
        $\text{proto}_c \leftarrow \frac{1}{K} \sum_{(e_i, y_i = c) \in E_S} e_i$
    end for
    for each query embedding $e_j \in E_Q$ **do**
        $d_j \leftarrow [\delta(e_j, \text{proto}_1), \ldots, \delta(e_j, \text{proto}_N)]$
        $p_j \leftarrow \text{softmax}(-d_j)$
    end for
    $L \leftarrow -\frac{1}{|\mathcal{Q}_i|} \sum_{j=1}^{|\mathcal{Q}_i|} \log p_j[y_j]$ ▷ Cross-entropy
    Update $\theta \leftarrow \theta - \beta \nabla_\theta L$
end while

Matching Network

Matching Network (MN) [36] is similar to PTN. The key difference lies in how the query examples are classified. PTNs classify query examples by first computing a prototype for each class, by averaging the embeddings of its support examples. Classification is then based on the closest prototype. In contrast, MNs do not rely on prototypes. Instead, they compare each query example directly with every support example by

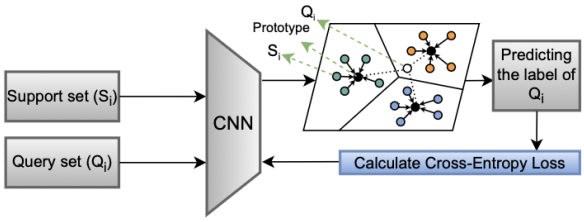


Fig. 1: Prototypical Network

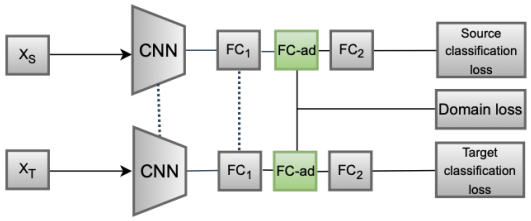


Fig. 2: Domain Adaptation

computing pairwise similarity scores based on distance between their embeddings. These scores determine the influence of each support sample on the classification outcome. The final prediction is derived by computing a weighted sum of the support labels, where the weights correspond to the calculated similarity scores.

F. Domain Adaptation Methodology

Maximum Mean Discrepancy

Domain adaptation aims to transfer knowledge from a labeled source domain to a target domain, which has limited labeled data, by learning domain-invariant feature representations. To achieve this, the model aligns the feature distributions of the source and target domains while maintaining semantic consistency in classification. The Maximum Mean Discrepancy (MMD) is employed to measure and minimize the difference between the source and target distributions. Specifically, following the work in [44], a domain confusion loss is integrated with the classification loss to train the CNN. As shown in Figure 2, both the source and target data pass through a CNN architecture where all parameters are shared between two domains for all Conv layers and the first FC layer. To ensure that the feature representations from the source and target domains are better aligned in the shared feature space, an adaptation layer is added between the first and second FC layer, which uses ReLU activation function, a batch normalization layer, and a dropout layer with $p = 0.5$. The MMD loss is computed using the feature representations produced by this adaptation layer. The training process optimizes the network using a combination of four losses:

$$\begin{aligned}\mathcal{L} = \mathcal{L}_C(\mathbf{X}_S, \mathbf{y}_S) + \alpha \cdot \mathcal{L}_C(\mathbf{X}_T, \mathbf{y}_T) \\ + \lambda_g \cdot \text{MMD}^2_{\text{global}}(\mathbf{X}_S, \mathbf{X}_T) \\ + \lambda_c \sum_{i=1}^{N} \text{MMD}^2_{\text{class}}(\mathbf{X}_S^i, \mathbf{X}_T^i)\end{aligned} \tag{1}$$

where $\mathcal{L}_C(\mathbf{X}_S, \mathbf{y}_S)$ is the classification loss on the source domain, which is computed using the cross-entropy loss to ensure accurate predictions on the labeled source domain data. $\mathcal{L}_C(\mathbf{X}_T, \mathbf{y}_T)$ is the target classification loss on the target domain based on cross-entropy on limited labeled target domain data. The third and fourth terms are the global-level MMD loss and class-level MMD loss, respectively. Global-level MMD aligns the overall feature distributions of the entire source and target domains without considering class-specific structure. Class-level MMD computes MMD separately for each class and then sums the MMD losses for all classes, which refines the alignment at a finer level and prevents misalignment within individual classes. α, λ_g, and λ_c are the hyperparameters to control the trade-off between these losses. Specifically, MMD loss is computed using the following equation:

$$\text{MMD}(\mathbf{X}_S, \mathbf{X}_T) = \left\| \frac{1}{|\mathbf{X}_S|} \sum_{x_s \in \mathbf{X}_S} \phi(x_s) - \frac{1}{\mathbf{X}_T} \sum_{x_t \in \mathbf{X}_T} \phi(x_t) \right\| \tag{2}$$

where $\phi(.)$ is a kernel function that maps the original data into a higher-dimensional space if necessary where distributions can be more easily compared. In this work, we leveraged linear kernel for simple implementation and fast training. Linear MMD represents the discrepancy between the mean embeddings of source and target distributions.

IV. Evaluation

A. Datasets

We utilized 3 publicly available datasets for our study. The first dataset, LoRa RF dataset [15], consists of both time-domain IQ samples from 25 identical Pycom IoT devices, which encompasses over 16,300 transmissions. This dataset was collected under seven different scenarios with various network deployments. In our experiments, we focused on IQ samples collected from four deployments: different days, different locations, different configurations, and different distances. Distance refers to the distance between transmitter and receiver, and configuration refers to variations in the LoRa protocol parameters. The details of each scenario setting are shown in Table I. Under the scenario of different configurations, the varying parameters across four configurations are spread factor (7, 8, 11, 12) and bit rate (5470, 3125, 537, 293 bps), while other parameters are unchanged. All transmissions under the same scenario are 1 minute apart from one another. After data pre-processing with a window size of 8192 samples, the number of examples per transmission is 1,220, each with an input shape of (8192, 2).

TABLE I: Specifications of four different scenarios in LoRa dataset

Scenarios	Number of Devices	Number of Days	Transmissions per Device	Duration per Transmission	Distances	Environment
Diff Days Indoor	25	5	10	20s	5m	Indoor
Diff Locations	25	1	3	20s	5m	Room, Office, Outdoor
Diff Distances	25	1	4	20s	5,10,15,20m	Outdoor
Diff Configurations	25	1	4	20s	5m	Indoor

The second RF dataset, the WiSig dataset [45], [46], was collected over four days at the Orbit Testbed and contains IQ signals from 174 WiFi cards captured using 41 USRP receivers. We followed the same preprocessing step as proposed in [25] and used the data from a single receiver ("node3-19") focusing on the 130 devices that were active across all four days. The third RF dataset, the CORES dataset [46], [47], collected at the Orbit Testbed, comprises signals from 163 consumer WiFi cards arranged in a grid. These signals were recorded using a USRP N210 over four days. In our experiment, we considered 58 devices that were consistently present across all sessions.

B. Experimental Design

Given these datasets, we designed 6 domain shift experiments as shown in Table II to compare different transfer learning algorithms.

TABLE II: Different domain shift experiments

Setting	Domain Shift	Dataset	Source	Target
LoRa-Day	Day	LoRa	Day 1, 2	Day 3, 4
WiSig-Day	Day	WiSig	Day 1, 2	Day 3, 4
CORES-Day	Day	CORES	Day 1, 3	Day 2, 4
LoRa-Config	Configuration	LoRa	Config 1	Config 2
LoRa-Location	Location	LoRa	Office	Outdoor
LoRa-Distance	Distance	LoRa	15m	20m

To evaluate the models, we used the evaluation framework mentioned in methodology. For evaluation on target domain, we used $K = 3$, $N = 5$ and $T = 100$. The following models were evaluated:

Zero-shot Baseline: A CNN model was trained in a standard supervised manner on the training subset of the source domain. During evaluation, the model was not exposed to any labeled examples from target domain.

Finetune Baseline: The pretrained baseline model was fine-tuned on K labeled samples per class from target domain for several epochs.

PTN and MN: The models were trained on source domain using Algorithm 2. For accuracy evaluation, the models generate prototypes given K labeled samples per class from target or source domains.

MMD: The model is trained using source training examples and K labeled examples per class from the target domain.

To train PTN and MN, we set C equal to the number of all classes in a dataset, $K = 3$, $N = 12$ and $T = 1,000$. For baselines, PTN, and MN training, we used early stopping on validation data with a patience of 50 and a learning rate of 0.001. For MMD, Stochastic Gradient Descent (SGD) was used as the optimizer with a step decay factor of 0.8 and a step size of 10. The initial learning rate was 0.01 for the adaptation layer and last FC layer, and 0.001 for all Conv layers and the first FC layer, with a patience of 50. After hyperparameter tuning, we set $\alpha = 1$, $\lambda_g = 0.7$, and $\lambda_c = 0.1$ to compute the total loss. For all trainings, we used a batch size of 16.

C. Experimental Results on Different PTN's Hyperparameters

Table III compares PTN's performance on target task for three different distance functions (Manhattan, Euclidean, and Cosine) and two different K-shot values in day-based settings across three datasets. The results indicate that Euclidean distance outperforms both Manhattan and Cosine distances. Also, it reveals that the 3-shot setting (only using 3 labeled samples from each label) is superior to the 1-shot setting across all distance metrics and settings. Based on the results, we selected Euclidean distance and a 3-shot setting as the optimal configuration, which was used for the remainder of this study.

D. Experimental Results on Different Domain Shift Settings

We compared all FSL algorithms on domain shift settings outlined in Table II, and the results are presented in Figure 3. Each plot in the figure represents a specific domain shift setting, showing the average accuracy across 100 experiment repetitions, with error bars indicating the standard deviation.

PTN and MN are the superior methodologies across all settings, maintaining high accuracy on the source task while also achieving strong performance on the target task, thanks to their ability to learn robust label representations. Following these, MMD is the next best methodology, demonstrating solid performance on both tasks. In contrast, the zero-shot baseline approach suffers a significant drop in target task accuracy, highlighting the domain shifts between the source and target task. Similarly, the finetune baseline method, after fine-tuning on the target task, experiences catastrophic forgetting, leading to a substantial decline in source task accuracy.

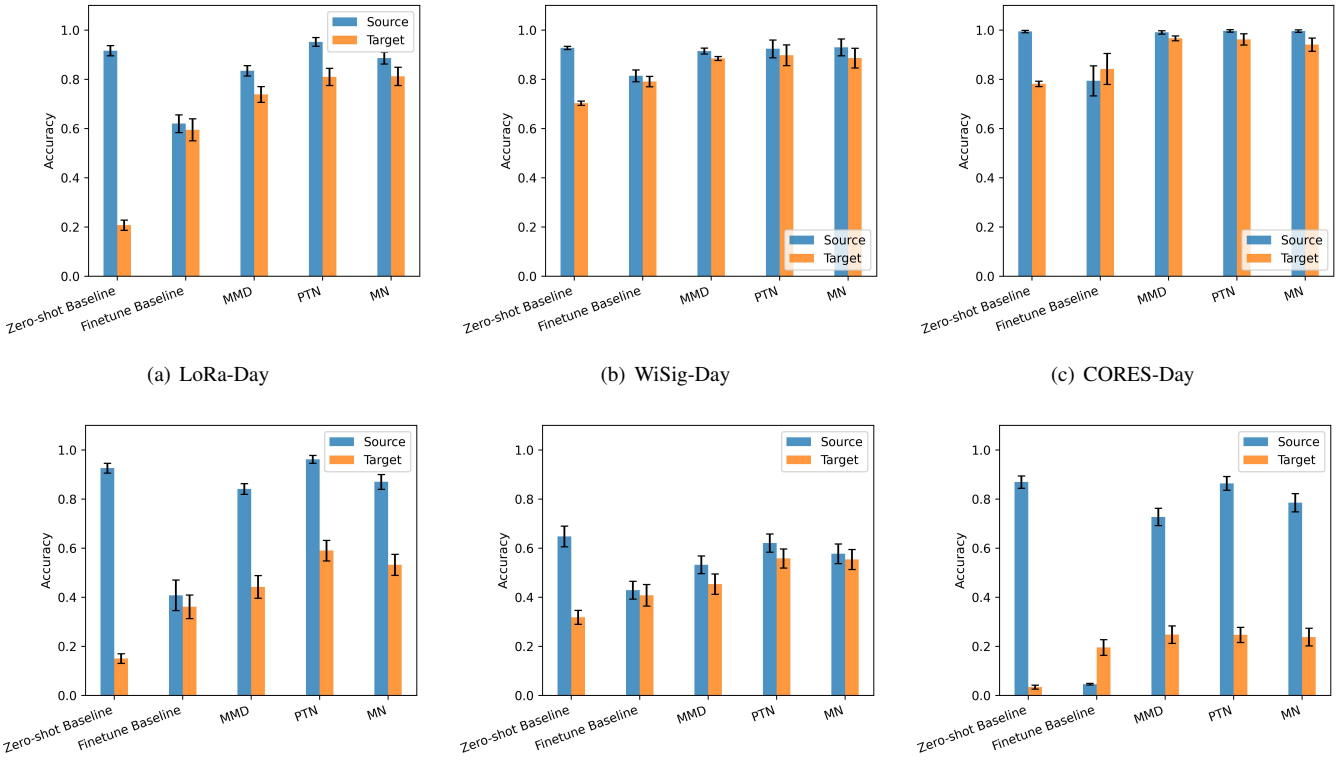
Among the six settings, the LoRa-Config and LoRa-Location settings stand out as the most challenging. In these cases, the accuracy on the target tasks are low, suggesting that these types of domain shifts pose substantial challenges for transfer learning. In contrast, the WiSig-Day and CORES-Day settings exhibit smaller gaps between source and target accuracy, reflecting less severe domain shifts.

E. Analysis of Learned Feature Embeddings

To evaluate the effectiveness of domain adaptation and understand how well target domain examples are represented in the feature space, we visualize the learned embeddings using t-distributed Stochastic Neighbor Embedding (t-SNE).

TABLE III: Comparison of PTN performance across different distance functions and k-shot values for three datasets.

Distance Function	LoRa-Day Dataset		WiSig-Day Dataset		CORES-Day Dataset	
	K=1	K=3	K=1	K=3	K=1	K=3
Euclidean Distance	0.742	0.810	0.768	0.898	0.927	0.962
Manhattan Distance	0.721	0.784	0.746	0.881	0.910	0.941
Cosine Distance	0.737	0.792	0.739	0.872	0.932	0.953



(a) LoRa-Day (b) WiSig-Day (c) CORES-Day

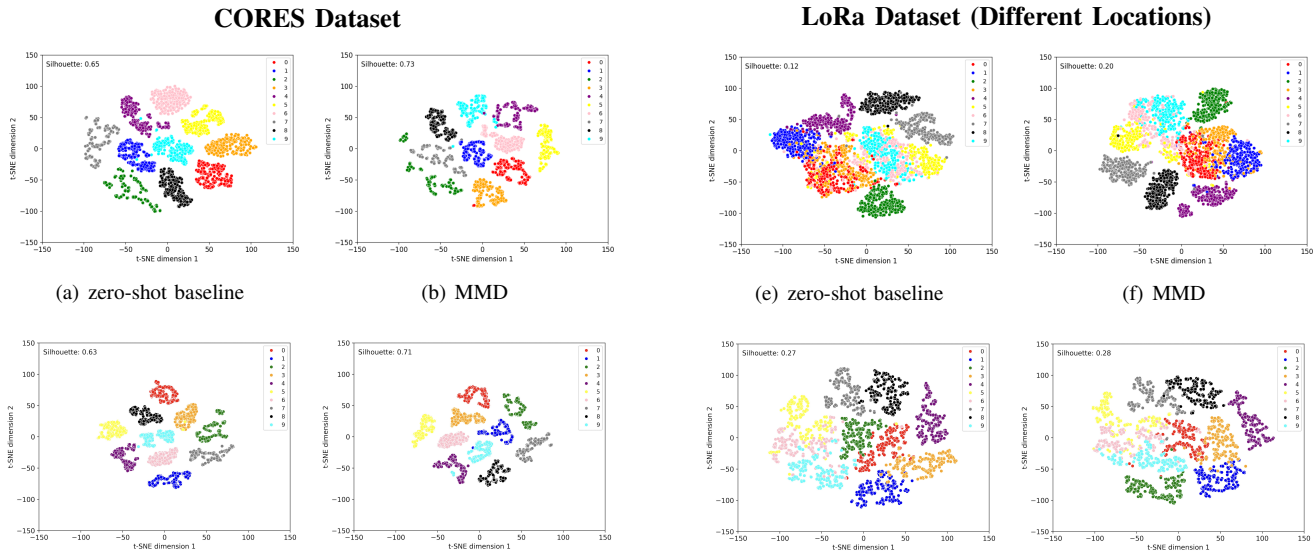
(d) LoRa-Location (e) LoRa-Distance (f) LoRa-Config

Fig. 3: Comparing accuracy of different methodologies over different domain shift settings

We compare the t-SNE plots of the zero-shot baseline, MMD-based domain adaptation, PTN, and MN in the target domain. t-SNE is a widely used nonlinear dimensionality reduction technique for projecting high-dimensional data into a 2D space, enabling visual inspection of cluster structures and class separability.

In our experiments, feature embeddings are extracted from the output before the final softmax layer of CNN. The visualization results for the CORES dataset and one scenario of the LoRa dataset are presented in Fig. 4, respectively. To ensure fair comparisons, we used the same subset of target examples across all methods when generating the t-SNE plots. We evaluate the visualizations based on two factors: the intra-class compactness, which represents how tightly samples from the same class cluster together, and inter-class separation, which represents how distinctly clusters of different classes are separated in space. It can be seen from Fig. 4 that the MMD, PTN, and MN approaches all produce more compact intra-class clusters compared to the zero-shot baseline, indicating that they encourage the model to group semantically similar samples more cohesively. Additionally, the distinct class boundaries are maintained using MMD, PTN and MN.

The zero-shot baseline of Fig. 4 indicates that RF fingerprinting in the LoRa data set is more challenging than the CORES dataset, as significant class mixing is observed. Samples from the same class are scattered, and many different classes overlap in the embedding space. However, with MMD, PTN, and MN, the visualizations show improved structure: certain classes (e.g., 0, 3, and 5) exhibit higher intra-class compactness, and others (e.g., 5 vs. 6 vs. 9, and 0 vs. 3) demonstrate better inter-class separation. These improvements suggest that FSL methods enhance the model's ability to generalize to the target domain. Additionally, from the results of PTN and MN in Fig. 4, it can be seen that these two methods yield less compact clusters but slightly better separation between clusters compared to MMD. This observation aligns with their underlying learning objectives: PTN and MN explicitly maximize the inter-class distance using feature embeddings, while MMD focuses on domain alignment without explicitly optimizing the inter-class separation.



(a) zero-shot baseline (b) MMD (e) zero-shot baseline (f) MMD

(c) PTN (d) MN (g) PTN (h) MN

Fig. 4: t-SNE plots comparing predictions from different methodologies across 10 classes on LoRa dataset (setting: different locations) and CORES dataset

To better interpret the t-SNE visualizations, we quantize the cluster compactness and separation of all methods by computing the silhouette score [48] using feature embeddings before t-SNE dimensionality reduction. Silhouette score is scale-invariant measurement on how similar a data point is to its own cluster compared to other clusters. For a point i, it can be defined as:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{3}$$

where $a(i)$ represents the mean distance between i and all other points in the same cluster; $b(i)$ represents the mean distance between i and all points in the nearest neighboring cluster. The overall silhouette score is the mean of $s(i)$ for all points. A higher silhouette score indicates a better group clustering performance.

The results of silhouette scores of all methods on 10 classes have been shown at the top-left corner of each figure. This additional metric further validates our conclusions on the success of our methods in mitigating the domain shift in RF fingerprinting. For example, Fig.4 indicates that FSL methods achieve higher silhouette scores compared to the baselines. Specifically, MMD shows the best score for smaller domain shift scenario, while PTN and MN show the best scores for larger domain shift scenario.

V. Conclusion

In this paper, we proposed a unified framework to evaluate and compare domain adaptation and meta-learning approaches for addressing the domain shift challenge in RF fingerprinting. Specifically, we applied two state-of-the-art meta-learning methods, Prototypical Networks and Matching Networks, alongside a domain adaptation approach based on Maximum Mean Discrepancy (MMD), across three real-world datasets with varying domain shift settings. Our results demonstrate that these methodologies outperform baseline models and we found the meta-learning methodologies superior in our domain shift settings. Furthermore, we analyzed the embeddings learned by these methods and provide visualizations to illustrate how these methods adapt the target domain within the feature space.

VI. Acknowledgment

This work is supported in part by the NSF (CNS-2317190, CNS-2107190, CNS-2107164).

References

[1] S. U. Rehman, K. W. Sowerby, S. Alam, and I. Ardekani, "Radio frequency fingerprinting and its challenges," in *2014 IEEE Conference on Communications and Network Security*, 2014, pp. 496–497.

[2] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar, "A review of radio frequency fingerprinting techniques," *IEEE Journal of Radio Frequency Identification*, vol. 4, no. 3, pp. 222–233, 2020.

[3] A. Jagannath, J. Jagannath, and P. S. P. V. Kumar, "A comprehensive survey on radio frequency (rf) fingerprinting: Traditional approaches, deep learning, and open challenges," *Computer Networks*, vol. 219, p. 109455, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622004893

[4] O. Ureten and N. Serinken, "Wireless security through rf fingerprinting," *Canadian Journal of Electrical and Computer Engineering*, vol. 32, no. 1, pp. 27–33, 2007.

[5] J. Hall, M. Barbeau, E. Kranakis *et al.*, "Enhancing intrusion detection in wireless networks using radio frequency fingerprinting." *Communications, internet, and information technology*, vol. 1, 2004.

[6] J. Hall, M. Barbeau, and E. Kranakis, "Radio frequency fingerprinting for intrusion detection in wireless networks," *IEEE Transactions on Defendable and Secure Computing*, vol. 12, pp. 1–35, 2005.

[7] J. Zhang, G. Shen, W. Saad, and K. Chowdhury, "Radio frequency fingerprint identification for device authentication in the internet of things," *IEEE Communications Magazine*, vol. 61, no. 10, pp. 110–115, 2023.

[8] P. Mirowski, D. Milioris, P. Whiting, and T. K. Ho, "Probabilistic radio-frequency fingerprinting and localization on the run," *Bell Labs Technical Journal*, vol. 18, no. 4, pp. 111–133, 2014.

[9] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, no. 1, p. 7068349, 2018.

[10] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Computer Science Review*, vol. 40, p. 100379, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574013721000198

[11] T. Jian, B. C. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis, "Deep learning for rf fingerprinting: A massive experimental study," *IEEE Internet of Things Magazine*, vol. 3, no. 1, pp. 50–57, 2020.

[12] Q. Wu, C. Feres, D. Kuzmenko, D. Zhi, Z. Yu, X. Liu, and X. 'Leo'Liu, "Deep learning based rf fingerprinting for device identification and wireless security," *Electronics Letters*, vol. 54, no. 24, pp. 1405–1407, 2018.

[13] M. Cekic, S. Gopalakrishnan, and U. Madhow, "Wireless fingerprinting via deep learning: The impact of confounding factors," in *2021 55th Asilomar Conference on Signals, Systems, and Computers*, 2021, pp. 677–684.

[14] G. Shen, J. Zhang, A. Marshall, and J. R. Cavallaro, "Towards scalable and channel-robust radio frequency fingerprint identification for lora," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 774–787, 2022.

[15] A. Elmaghbub and B. Hamdaoui, "Lora device fingerprinting in the wild: Disclosing rf data-driven fingerprint sensitivity to deployment variability," *IEEE Access*, vol. 9, pp. 142 893–142 909, 2021.

[16] A. Mohammadian and C. Tellambura, "Rf impairments in wireless transceivers: Phase noise, cfo, and iq imbalance – a survey," *IEEE Access*, vol. 9, pp. 111 718–111 791, 2021.

[17] Y. Ren, L. Peng, W. Bai, and J. Yu, "A practical study of channel influence on radio frequency fingerprint features," in *2018 IEEE International Conference on Electronics and Communication Engineering (ICECE)*, 2018, pp. 1–7.

[18] L. Sun, L. Zhang, and Z. Liu, "Transfer learning-based radio frequency fingerprinting across receiving systems," in *2024 IEEE International Conference on Signal, Information and Data Processing (ICSIDP)*, 2024, pp. 1–6.

[19] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[20] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[21] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*. Springer, 2018, pp. 270–279.

[22] C. T. Nguyen, N. Van Huynh, N. H. Chu, Y. M. Saputra, D. T. Hoang, D. N. Nguyen, Q.-V. Pham, D. Niyato, E. Dutkiewicz, and W.-J. Hwang, "Transfer learning for wireless networks: A comprehensive survey," *Proceedings of the IEEE*, vol. 110, no. 8, pp. 1073–1115, 2022.

[23] T. Tian, Y. Wang, H. Dong, Y. Peng, Y. Lin, G. Gui, and H. Gacanin, "Transfer learning-based radio frequency fingerprint identification using convmixer network," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, 2022, pp. 4722–4727.

[24] S. Kuzdeba, J. Robinson, and J. Carmack, "Transfer learning with radio frequency signals," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2021, pp. 1–9.

[25] S. Mackey, T. Zhao, X. Wang, and S. Mao, "Cross-domain adaptation for rf fingerprinting using prototypical networks," in *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, 2022, pp. 812–813.

[26] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, 2020.

[27] T. Zhao, X. Wang, and S. Mao, "Cross-domain, scalable, and interpretable rf device fingerprinting," in *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, 2024, pp. 2099–2108.

[28] H. Wang, H. Zha, Y. Lin, and H. Jiang, "Tfma: Few-shot rf fingerprint identification with time-frequency mask augmentation," in *2023 IEEE 23rd International Conference on Communication Technology (ICCT)*, 2023, pp. 1651–1655.

[29] A. Elmaghbub, B. Hamdaoui, and W.-K. Wong, "Adl-id: Adversarial disentanglement learning for wireless device fingerprinting temporal domain adaptation," in *ICC 2023 - IEEE International Conference on Communications*, 2023, pp. 6199–6204.

[30] H. Li, Y. Tang, D. Lin, Y. Gao, and J. Cao, "A survey of few-shot learning for radio frequency fingerprint identification," in *Artificial Intelligence for Communications and Networks*, X. Wang, K.-K. Wong, S. Chen, and M. Liu, Eds. Cham: Springer International Publishing, 2021, pp. 433–443.

[31] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial intelligence review*, vol. 18, pp. 77–95, 2002.

[32] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.

[33] D. Wang, M. Zhang, Y. Xu, W. Lu, J. Yang, and T. Zhang, "Metric-based meta-learning model for few-shot fault diagnosis under multiple limited data conditions," *Mechanical Systems and Signal Processing*, vol. 155, p. 107510, 2021.

[34] M. Huisman, J. N. Van Rijn, and A. Plaat, "A survey of deep meta-learning," *Artificial Intelligence Review*, vol. 54, no. 6, pp. 4483–4541, 2021.

[35] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[36] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, 2016.

[37] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto, "Few-shot adversarial domain adaptation," *Advances in neural information processing systems*, vol. 30, 2017.

[38] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018.

[39] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. 25, pp. 723–773, 2012. [Online]. Available: http://jmlr.org/papers/v13/gretton12a.html

[40] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[41] L. Yang, Q. Li, X. Ren, Y. Fang, and S. Wang, "Mitigating receiver impact on radio frequency fingerprint identification via domain adaptation," *IEEE Internet of Things Journal*, vol. 11, no. 13, pp. 24 024–24 034, 2024.

[42] Z. Zhang, G. Li, J. Shi, H. Li, and A. Hu, "Real-world aircraft recognition based on rf fingerprinting with few labeled ads-b signals," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 2, pp. 2866–2871, 2023.

[43] N. Niknami, V. Mahzoon, and J. Wu, "Ptn-ids: Prototypical network solution for the few-shot detection in intrusion detection systems," in *2024 IEEE 49th Conference on Local Computer Networks (LCN)*. IEEE, 2024, pp. 1–9.

[44] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[45] S. Hanna, S. Karunaratne, and D. Cabric, "Wisig: A large-scale wifi signal dataset for receiver and channel agnostic rf fingerprinting," *IEEE Access*, vol. 10, pp. 22 808–22 818, 2022.

[46] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh, "Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols," in *IEEE Wireless Communications and Networking Conference, 2005*, vol. 3. IEEE, 2005, pp. 1664–1669.

[47] S. Hanna, S. Karunaratne, and D. Cabric, "Open set wireless transmitter authorization: Deep learning approaches and dataset considerations," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 1, pp. 59–72, 2020.

[48] K. R. Shahapure and C. Nicholas, "Cluster quality analysis using silhouette score," in *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, 2020, pp. 747–748.