Few-Shot Domain Adaptation for Effective Data Drift Mitigation in Network Management

Seyed Soheil Johari*, Massimo Tornatore†, Raouf Boutaba*, Aladdin Saleh‡
*University of Waterloo, Canada, {ssjohari | rboutaba}@uwaterloo.ca
†Politecnico di Milano, Italy, massimo.tornatore@polimi.it
‡Rogers Communications Canada Inc., Canada, aladdin.saleh@rci.rogers.com

***Abstract*—Machine Learning (ML) models are increasingly employed for critical network management tasks such as traffic prediction, anomaly detection, root cause analysis, and resource allocation. A major issue in the reliability of these models is data drift, which refers to the discrepancy between training data (source domain) and test/operational data (target domain) caused by changes in network conditions and configurations. Domain adaptation, which aims to develop robust models that can generalize well across different but related domains, is a promising solution to the data drift issue. However, existing domain adaptation methods often fall short in few-shot scenarios, where target domain data is limited due to high data collection costs or restricted operational network access. Furthermore, the existing methods require the network management ML models to be frequently retrained or fine-tuned over time to adapt to the changes in data distributions, leading to high operational costs. To address these limitations, we propose a novel, model-agnostic domain adaptation approach specifically designed for few-shot scenarios and network data. In our approach, network management ML models are trained exclusively on source domain data with all the input features included, while a two-step method aligns test data samples from the target domain with the source domain during inference. The first step employs our proposed causal-inference-based feature separation (FS) method, which introduces a novel perspective by treating domain shift as soft interventions (interventions that adjust the probability distribution of features rather than making absolute changes) on a set of specific features. FS effectively separates domain-variant and domain-invariant features directly in the input space, even with limited target training data. In the second step of our approach, we propose a Generative Adversarial Network (GAN)-based reconstruction method, trained exclusively on source data, to reconstruct the domain-variant features given the domain-invariant features. During inference, the GAN model maps the domain-variant features of the target domain samples to the source domain distribution, allowing the use of domain-variant features without causing cross-domain performance degradation. Since our approach trains the network management ML models exclusively on source domain data, it eliminates the need for retraining or fine-tuning these models as network conditions or data distributions evolve over time, significantly reducing costs and operational overhead. Comprehensive evaluations on two public 5G network datasets demonstrate an average 52% improvement in mitigating data drift compared to state-of-the-art methods in terms of F1-score.**

***Index Terms*—Data Drift, Domain Adaptation, Network Management, Causal Inference**

I. Introduction

The deployment and management of 5G and beyond networks present significant challenges due to their complexity and dynamic nature [1], [2]. With the rise of Virtualized Network Functions (VNFs) [3], operational data has surged in volume and diversity, creating both opportunities and challenges for effective network management. To navigate these complexities and leverage the vast amount of available data, Machine Learning (ML) models are being employed to address a variety of management tasks, such as traffic prediction [4], anomaly detection [5], and network security [6].

However, these ML models inherently rely on temporal data, which is subject to continuous changes over time due to system upgrades, configuration adjustments, workload shifts, and evolving user demands [7], [8]. As a result, the issue of data drift [9], a discrepancy between training data (source domain) and testing data (target domain), becomes a significant challenge, leading to a substantial decline in model accuracy. In 5G and beyond networks, this challenge is intensified by the reliance on simulation data (e.g., digital twins [10]), as replicating an exact network and all possible patterns is impractical [8]. Consequently, the developed models need to be refined to effectively support actual network operations. Domain adaptation (DA) [11] emerges as a promising solution in this regard, as it aims to bridge the gap between the training and target domains, ensuring that models remain accurate and reliable despite the inevitable changes in data distribution.

However, existing DA methods [12]–[16] often fall short in few-shot scenarios where target domain training data is limited due to the high cost of data collection and restricted access to operational networks. Furthermore, these approaches require frequent retraining or fine-tuning of the different network management models whenever network conditions or configurations evolve, leading to considerable redeployment costs and operational inefficiencies. To overcome these limitations, we propose a novel, model-agnostic DA approach tailored for few-shot scenarios and network data that requires no retraining or fine-tuning of the network management models over time. In our approach, network management ML models are trained exclusively on source domain data with all the input features included. To ensure effective cross-domain performance, we introduce a two-step method to align test data samples from the target domain with the source domain before they are processed by the network management models during inference.

The first step employs our proposed causal-inference-based feature separation (FS) method, which introduces a new perspective by treating domain shift as soft interventions

[17] on specific features (in causal inference, an intervention involves intentionally manipulating certain features to observe their effects on others, and a soft intervention is a type of intervention that modifies the probability distribution of features rather than making absolute changes). In the FS method, source domain data is treated as observational data, i.e., data collected without interventions, while target domain data is interpreted as interventional data, i.e., data gathered after certain features are altered by interventions. The features impacted by these interventions (the intervention targets) are identified through causal inference and are considered as the domain-variant features. To the best of our knowledge, this novel view of domain shift as soft interventions and the identification of intervention targets through causal discovery as domain-variant features is introduced by us in this work.

We show that even with very limited target domain training data, the FS method effectively separates data features into domain-variant and domain-invariant categories in the input space. Our findings demonstrate that domain-variant features are the primary contributors to performance degradation due to data drift. By isolating these features and focusing on domain-invariant features, we can significantly improve the robustness and cross-domain performance of network management ML models. *This direct feature separation approach is particularly beneficial for DA in network data, where features are affected differently by domain shifts. In contrast, existing DA methods designed for image or text data rely heavily on spatial or contextual structures, making input-space feature separation impractical for those applications.*

In the second step, we propose a reconstruction method based on a Generative Adversarial Network (GAN) model [18] to leverage information from domain-variant features without causing performance degradation when applying the network management models to different domains. A GAN is trained on the source domain data to learn to reconstruct the variant features from the invariant features. During inference, the trained GAN maps the variant features of a target (test) data sample to the source domain distribution. This allows the network management models, trained exclusively on source data with all the input features included, to incorporate information from variant features without causing performance degradation when applying the network management models to different domains (i.e., cross-domain performance degradation). Since the GAN is trained solely on source data without using any target domain samples, it is ideally suited for few-shot settings. Our proposed GAN model is completely different from the GAN models in adversarial DA methods [12]–[15] in terms of architecture, training, and purpose. The existing GAN models in adversarial DA are trained on both source and target domain data to learn domain-independent latent representations from both domains. In contrast, in our approach, separation of domain-variant and invariant features is achieved by our FS method, and our GAN model is only trained on the source domain data to learn how to reconstruct a portion of the input features (domain-variant features) given the remaining portion of the features (domain-invariant features).

As mentioned, in our approach, the network management ML models are trained exclusively on source domain data. If the data domain evolves further over time, requiring adaptation to new changes in the data distribution, it may be necessary to rerun the FS method and retrain the GAN model to identify and address newly emerging domain-variant features. However, the network management ML models themselves do not require retraining/fine-tuning, as they remain trained solely on the source data. The FS method and the GAN model handle the alignment of new test samples to the source domain, ensuring compatibility with the existing ML models. Compared to existing DA methods, this aspect of our approach significantly reduces the cost and resource requirements associated with retraining and redeploying network management ML models, as there are numerous network management models trained on network data for a variety of tasks (such as traffic prediction, anomaly detection, root cause analysis, and resource allocation).

To the best of our knowledge, this is the first work to combine causal inference with a GAN-based approach for few-shot domain adaptation, enabling separation of domain-variant and invariant features while preserving valuable variant information. Our contributions are as follows:

- **A Few-Shot DA Approach Combining Causal Feature Separation and GAN-Based Reconstruction**: We propose a novel few-shot DA approach, tailored for network data, that combines causal inference-based feature separation with a GAN-based reconstruction method to address cross-domain performance degradation.
- **Model-Agnostic Framework**: Our DA approach is model-agnostic, allowing the use of various ML model architectures for network management tasks. While the focus of our experimental evaluation is on classification tasks, the model-agnostic nature of our method allows it to be applied to a variety of other network management tasks such as regression, forecasting, and resource allocation. This ensures that our approach is adaptable to different network management scenarios and constraints.
- **No need for retraining or fine-tuning the network management models**: In our approach, all network management ML models are trained exclusively on the source domain data and remain unchanged over time, regardless of further domain shifts or evolving network conditions. This eliminates the need for retraining or fine-tuning, simplifying maintenance and reducing operational costs.
- **Comprehensive Evaluation**: We thoroughly evaluate our DA methods using two public 5G datasets, one for network failure classification and the other for network fault detection. These datasets include data from a source domain (e.g., network digital twin) and a target domain (e.g., real network) with varying network traffic trends. The results demonstrate that our approach achieves an average 52% improvement in mitigating data drift compared to state-of-the-art DA and few-shot learning techniques in terms of F1-score performance.

II. Related Works

Data Drift Mitigation in Network Management: Various approaches have been proposed to mitigate the impact of data drift to maintain the performance of ML models in network management. For example, [7] presents a framework to address concept drift in cellular networks using explainability techniques and strategic retraining. The work in [8] tackles data drift in large-scale ML deployments by selecting the most similar training data batch for each test sample, enhancing accuracy and efficiency in Microsoft Azure. [15] proposes a DA method using labeled and unlabeled data to combat ML model performance degradation due to dynamic network changes, evaluated on 5G and cloud testbed data. [14] leverages GANs for adversarial DA to train effective network intrusion detection models for both homogeneous and heterogeneous feature spaces. These approaches are particularly effective in scenarios where there is an abundance of samples from the target domain. However, they often overlook scenarios with limited data availability from the target domain, which is the focus of our work.

Few-Shot Learning Methods for DA: Numerous studies have utilized few-shot learning techniques to make the most of limited data in various network management tasks [19], [20]. Methods like Matching Networks [21] and Prototypical Networks [22] have been particularly successful in achieving high performance with minimal training samples. These approaches can also facilitate DA by training on data from a source domain and then leveraging a few samples from the target domain to achieve robust generalization to the new domain. However, our evaluation shows that while these methods perform well with minor distribution shifts, they are less effective when the shift is significant, as their success relies on relatively similar distributions between source and target domains.

Causal Inference-Based DA: Causal DA uses causal inference to handle domain shifts by embedding probabilistic relationships from multiple domains into a larger causal structure. For example, [23] frames DA as Bayesian inference on graphical models to encode changes in joint distributions. [24] uses causal models to represent relationships between features and class labels, adapting to domain changes by transferring knowledge through these models. However, both [23] and [24] are designed for scenarios with multiple source domains and are not applicable to our single-source DA problem. Specifically, the central idea of modeling the target conditional distribution as a mixture of source conditionals in [23] and [24] requires at least two source domains to form a mixture.

The method in [25] leverages causal inference for DA using a copula entropy-based conditional independence test to find invariant representations, but it is unsuitable for our problem as it requires prior context information about interventions, fitting better with simulated scenarios where the characteristics of interventions can be directly measured. [16] proposes identifying invariant conditional distributions across domains without prior causal graph knowledge, using a joint causal inference framework for accurate predictions despite distribution shifts. Specifically, this method is designed for DA in medical data analysis, where the number of features is relatively small, making it not suited for high-dimensional data. [26] addresses few-shot supervised DA by leveraging causal modeling and nonlinear independent component analysis for data augmentation in the target domain, using the augmented samples for training. We included both [16] and [26] in our evaluations, demonstrating that our proposed method significantly outperforms these approaches in our few-shot DA problem.

DA with Training Exclusively on Source Domain Data: The majority of existing DA approaches include target domain data in the training of the network management models, and thus, by design, require retraining or fine-tuning of the network management models for a new target domain. However, there are some works (e.g., [27]–[29]) that train the model only on source data, but their DA procedure relies on test-time adaptation during inference using all available target domain samples, which significantly increases the inference time of the network management models and requires their fine-tuning (for example, [27] adapts batch normalization statistics during inference by minimizing entropy of predictions on target data). This increased inference time is undesirable for network management models, which often operate in real-time. Moreover, these methods are unsuitable for few-shot scenarios since they rely on numerous target domain samples for adaptation.

III. Problem Statement

Given a source domain $\mathcal{D}_A$ (e.g., a network digital twin) and a target domain $\mathcal{D}_C$ (e.g., the real network), our objective is to develop a robust DA framework to mitigate the impact of data drift on ML network management models. Let $\mathcal{X}_A = \{x_A^i\}_{i=1}^{n_A}$ and $\mathcal{Y}_A = \{y_A^i\}_{i=1}^{n_A}$ denote the feature space and label space of the source domain, respectively. Similarly, $\mathcal{X}_C = \{x_C^j\}_{j=1}^{n_C}$ and $\mathcal{Y}_C = \{y_C^j\}_{j=1}^{n_C}$ represent the feature space and label space of the target domain, respectively. n_A and n_C are the number of samples in $\mathcal{D}_A$ and $\mathcal{D}_C$, respectively. The number of training samples from the target domain is limited, due to practical constraints such as the high cost of data collection, limited access to operational networks, and infrequent occurrence of certain network events in real-world environments.

The goal is to train a model using both the source domain data and the limited target domain data that can perform effectively in the target domain. The key challenge is that the source domain $\mathcal{D}_A$ and the target domain $\mathcal{D}_C$ exhibit distinct characteristics and data distributions due to factors such as differing network configurations, traffic patterns, and operational environments. This discrepancy prevents ML models trained solely on $\mathcal{D}_A$ from performing well on $\mathcal{D}_C$. Furthermore, the limited samples in the target domain intensifies the difficulty, as the models trained only on the available $\mathcal{D}_C$ data are insufficient to achieve satisfactory results. We emphasize that our work targets distribution shifts arising from normal network condition changes, such as configuration updates or evolving usage patterns, rather than adversarial or malicious behaviors.

Our specific focus within this broader problem is the tasks of network failure classification and network fault detection (critical aspects of root cause analysis in network management). For the task of classifying network failures, each data sample $x \in \mathcal{X}$ comprises a collection of various performance metrics monitored from the network at a specific time instance related to a particular failure scenario. Its corresponding label $y \in \mathcal{Y}$ indicates the type of the failure, belonging to a finite set $\mathcal{C}$ that includes all possible types of failures. The final objective is to develop a DA framework to transfer knowledge from $\mathcal{D}_A$ to $\mathcal{D}_C$ using the limited target samples (e.g., one sample per each fault type) to improve the performance of the network failure classification model $f : \mathcal{X} \to \mathcal{Y}$ on $\mathcal{D}_C$. For the fault detection task, the setting is similar, except the label $y \in \mathcal{Y}$ is binary: 0 for normal and 1 for faulty network status.

IV. Datasets Description

A. 5GC Dataset

The first public dataset is from [30] and is generated in a cloud-native 5G mobile core network deployment using VNFs on the OpenStack infrastructure. The dataset comprises data collected from two mobile core domains: the source domain $\mathcal{D}_A$, representing a network digital twin, and the target domain $\mathcal{D}_C$, representing the real network. Domain $\mathcal{D}_A$ data serves as training data, while domain $\mathcal{D}_C$ data is split into additional training data (with a small number of samples) and test data. The primary challenge arises from the differences in network traffic patterns between these domains, which complicate the development of accurate models for domain $\mathcal{D}_C$ using only domain $\mathcal{D}_A$ data. The goal is to create a network failure classification model that operates reliably in $\mathcal{D}_C$ by leveraging $\mathcal{D}_A$ data and limited samples from $\mathcal{D}_C$. The dataset has 16 labels: one for normal operations and 15 for various failure scenarios derived from five fault types—bridge deletion, interface down, interface packet loss, memory stress, and vCPU overload—applied to three VNFs: AMF (Access and Mobility Management Function), AUSF (Authentication Server Function), and UDM (Unified Data Management).

The dataset includes 442 performance metrics covering network traffic, interface status, memory usage, CPU usage, system load, and specific 5G core metrics. These metrics provide a comprehensive view of network performance, including data volume, unicast packet counts, interface status and speed, memory statistics, CPU usage, system load, and 5G core registration metrics. The source domain training data includes 3,645 samples for the 16 classes. In our experiments, target domain training data is limited to 1, 5, or 10 samples per failure scenario, totaling 16, 80, or 160 samples, respectively (the normal class is also considered as a fault type here). While the original dataset [30] has 700 training samples from the target domain, we intentionally restrict our training data from the target domain to these three cases to simulate different few-shot DA scenarios. The target domain test data includes 873 samples, approximately evenly distributed among all classes.

B. 5GIPC Dataset

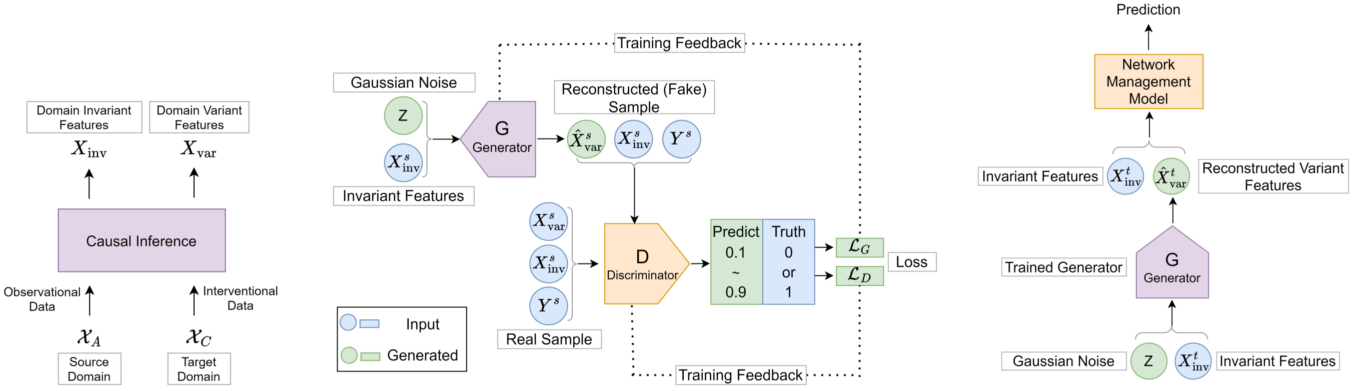
The second public dataset is from [31] and was generated in an NFV-based test environment that simulates a 5G IP core network. The NFV testbed's target topology consists of five VNFs: two IP core nodes (TR-01 and TR-02), two internet gateway routers (IntGW-01 and IntGW-02), and a route reflector (RR-01), each hosted on a separate Virtual Machine (VM). Various performance metrics, such as different resource utilization metrics and incoming/outgoing packet rates, are collected from each VNF at one-minute intervals. The network management task in this dataset is fault detection. To achieve this, the following fault scenarios are periodically injected into one of the VNFs to generate faulty samples: node failure, interface failure, packet loss, and packet delay. Normal samples are labeled as 0, while faulty samples (regardless of the fault type) are labeled as 1. Thus, fault detection is framed as a binary classification problem. We frame the task as fault detection rather than fault classification to differentiate it from the task in the first dataset.

For investigating the issue of data drift, we used the Gaussian Mixture Model (GMM) as a clustering algorithm to partition the dataset into two clusters based on the underlying distribution of the data. The larger cluster was designated as the source domain dataset $\mathcal{D}_A$, representing the majority of the data, while the smaller cluster was considered as the target domain dataset $\mathcal{D}_C$. This separation enabled us to simulate a domain adaptation scenario where data drift occurs, allowing us to evaluate the performance of fault detection models when trained on the source domain and tested on the target domain. The source domain dataset comprises 5,315 normal samples and 100, 226, 874, and 619 samples for the four fault types: node failure, interface failure, packet loss, and packet delay, respectively. The target domain dataset is split into additional training data (with a small number of samples) and test data. The target training dataset is limited to 1, 5, or 10 samples per fault type, totaling 5, 25, or 50 samples, respectively (the normal class is also considered as a fault type here). The target test dataset contains 2,060 normal samples and 95, 124, 311, and 546 samples for the same fault types.

V. Proposed Domain Adaptation Framework

In this section, we begin by detailing the first step of our DA approach, i.e., our novel FS method that considers the domain shift as soft interventions on a set of features to separate the performance metrics (features) into domain-variant and domain-invariant. (In causal inference, an intervention refers to an intentional manipulation of certain features to observe their effects on others. A soft intervention is a type of intervention where the changes are not absolute but rather adjust the probability distribution of the features. Observational data is the data collected without interventions, recording natural feature occurrences. In contrast, interventional data is the data gathered after certain features are altered by interventions.)

We formulate our FS problem by treating the source data as observational data and the target data as interventional data under soft interventions on certain features. In this context, the source data represents the original, unaltered data samples, whereas the target data represents the data points after some features have been adjusted due to the domain shift.



(a) Separating domain-variant and invariant features.

(b) GAN model training on the source domain data.

(c) Inference process for a test sample from the target domain.

Fig. 1: Overview of our proposed DA framework: a) Our causal inference-based method separates features into domain-variant and invariant sets, b) The GAN model is trained on source domain data to reconstruct variant features from invariant features, and c) During inference, the trained GAN generator maps the target sample's variant features to the source domain, enabling the network management models to utilize the new generated sample without being affected by data drift.

The intervened features in this formulation are identified as domain-variant features: those features that change between the source and target domains due to the soft interventions (or domain shift). The intervened features (the unknown intervention targets) are identified through causal inference to determine which features have been subtly altered (softly intervened upon) due to the domain shift. By focusing on the invariant features (those that remain consistent across both domains), we can mitigate the adverse effects of data drift and improve model performance on the target domain.

Next, we introduce the second step of our DA approach, i.e., the GAN-based reconstruction method to retain the benefits of variant features while maintaining robust performance across different domains. We train a conditional GAN on the source domain to learn how to reconstruct variant features from invariant ones within the source domain. During inference, the GAN maps variant features of a test target sample to the source domain, transforming the target sample into a source-like sample. This transformed sample is used for prediction with the network management model trained on the source domain data with all features included.

A. Separating Domain-Variant and Invariant Features

Our FS method (Fig. 1(a)) separates domain-variant and domain-invariant features by treating the domain shift as soft interventions on the variant features, which are initially unknown. We consider the source data $\mathcal{X}_A$ as observational data and the target data $\mathcal{X}_C$ as interventional data under soft interventions on a subset of features. To identify the unknown intervened features, we employ a modified version of the Ψ-FCI algorithm [17], adapted to our scenario with no latent confounders (as we have numerous observable features), to determine the intervened features.

1) Formulating the Problem: Following the Ψ-FCI algorithm, we combine the datasets from the source and target domains into a single dataset $\mathcal{D}^*$, where each sample $x_A \in \mathcal{X}_A$ from the source domain is labeled with $F = 0$ and each sample $x_C \in \mathcal{X}_C$ from the target domain is labeled with $F = 1$. Here, F is an introduced F-NODE [32]–[34] representing the effect of an intervention on the data, which in our problem is the domain indicator as we are considering the domain shift as an intervention. V represents the set of all observed features.

$$P^*(V \mid F = 0) = P_A(V), \quad P^*(V \mid F = 1) = P_C(V) \quad (1)$$

where P_A and P_C are the distributions of the source and target domains, respectively. The combined distribution P^* is constructed to incorporate both P_A and P_C, representing the overall data distribution under both observational and interventional conditions.

2) Learning the Causal Graph and Identifying Intervened Features: We apply the PC algorithm [35] to learn the causal graph structure of the features in the combined dataset $\mathcal{D}^*$, thereby identifying the intervened features. A causal graph is a graphical representation that depicts the causal relationships between different features in a dataset. By understanding these relationships, particularly the neighbors of the F-node, we can determine which features have been influenced by interventions due to domain shift. While the Ψ-FCI algorithm utilizes the FCI method [35] for constructing the causal graph, we use the PC algorithm instead, as we assume there are no latent confounders in our scenario. Additionally, we constrain the F-node to have no outgoing edges in the causal graph, as this node was manually added to the dataset. By running the PC algorithm on the combined dataset, we perform conditional independence tests to determine the features X that satisfy:

$$X \perp\!\!\!\perp F \mid \text{Pa}(X) \quad (2)$$

Here, $\text{Pa}(X)$ denotes the parent set of X [35]. This conditional independence statement indicates that X is one of the intervened features if it does not hold true.

3) Separation of Features: Features that do not satisfy the above conditional independence condition are identified as the intervened features R, such that:

$$P_A(R \mid \text{Pa}(R)) \neq P_C(R \mid \text{Pa}(R)) \tag{3}$$

These intervened features are directly influenced by the domain shift and are identified as the neighbors of the F-NODE in the learned causal graph. Once all the intervened features are identified, we can separate the set of domain-invariant features (X_{inv}) and the set of domain-variant features (X_{var}):

$$X_{\text{inv}} = V \setminus R, \quad X_{\text{var}} = R \tag{4}$$

B. Domain-Variant Features in the Datasets

We analyze domain-variant features identified by our method as those exhibiting significant variability between the source and target domains. For instance, in the 5GC dataset, one identified feature is the number of outgoing bytes on an AMF interface, while in the 5GIPC dataset, one example is the CPU utilization of the IntGW-01 gateway router. Another example is the number of incoming unicast packets on an SMF interface, which can vary based on network traffic patterns and device communication. Memory usage metrics for certain VNFs are also among the identified domain-variant features.

These features are sensitive to environmental changes, which can introduce noise and bias, degrading model performance on the target domain. Including such features in training may cause overfitting to source domain characteristics, reducing generalizability, while indiscriminate exclusion risks losing critical information. Therefore, a data-driven approach is necessary to accurately identify domain-variant features.

C. GAN-Based Method for Leveraging Variant Features

1) GAN Architecture and Training: Fig. 1(b) illustrates the training procedure of our GAN model, comprising a generator and a discriminator. The generator G takes as input a combination of the invariant features and Gaussian noise (a noise vector sampled from a standard normal distribution), and reconstructs the variant features as output. The discriminator D receives the generator's output concatenated with the original invariant features and the one-hot encoded label (e.g., network failure type) as its input, and it distinguishes between real and generated samples. For a sample $x^s \in \mathcal{X}_A$ from the source domain, let X^s_{inv} denote the values of its invariant features, X^s_{var} denote the values of its variant features, and Y^s denote its one-hot encoded label. The input to the generator is:

$$G_{\text{input}} = [X^s_{\text{inv}}, Z] \tag{5}$$

where Z is Gaussian noise. The generator's output is:

$$\hat{X}^s_{\text{var}} = G([X^s_{\text{inv}}, Z]) \tag{6}$$

Discriminator's input is the concatenation of the invariant features, the generator's output, and the one-hot encoded label Y^s:

$$D_{\text{input}} = [X^s_{\text{inv}}, \hat{X}^s_{\text{var}}, Y^s] \tag{7}$$

The discriminator is trained to distinguish between real samples $[X^s_{\text{inv}}, X^s_{\text{var}}, Y^s]$ and generated samples $[X^s_{\text{inv}}, \hat{X}^s_{\text{var}}, Y^s]$. The generator is trained to fool the discriminator by generating variant features that are indistinguishable from real ones. The objective functions for the generator and discriminator are:

$$\begin{aligned} \mathcal{L}_D = &- \mathbb{E}[\log D([X^s_{\text{inv}}, X^s_{\text{var}}, Y^s])] \\ &- \mathbb{E}[\log(1 - D([X^s_{\text{inv}}, \hat{X}^s_{\text{var}}, Y^s]))] \end{aligned} \tag{8}$$

$$\mathcal{L}_G = -\mathbb{E}[\log D([X^s_{\text{inv}}, \hat{X}^s_{\text{var}}, Y^s])] \tag{9}$$

Generator and discriminator are trained iteratively (the generator minimizing $\mathcal{L}_G$ and the discriminator minimizing $\mathcal{L}_D$).

2) Inference with Trained Generator: We assume that the network management model (e.g., the failure classification model) deterministically predicts $P(Y|X_{\text{var}}, X_{\text{inv}})$, while our trained GAN model estimates $P(X_{\text{var}}|X_{\text{inv}})$. During inference, the goal is to predict $P(Y|X_{\text{inv}})$ for target samples using only their domain-invariant features. By the law of total probability:

$$P(Y|X_{\text{inv}}) = \int P(Y|X_{\text{var}}, X_{\text{inv}}) P(X_{\text{var}}|X_{\text{inv}}) \, d_{X_{\text{var}}}$$

Since $P(X_{\text{var}}|X_{\text{inv}})$ is modeled by the GAN, we can sample $X^{(i)}_{\text{var}}$ values from the GAN's generator and approximate $P(Y|X_{\text{inv}})$ using Monte Carlo sampling. Specifically, M samples $\{X^{(i)}_{\text{var}}\}^M_{i=1} \sim P(X_{\text{var}}|X_{\text{inv}})$ are drawn from the GAN, conditioned on X_{inv}. The network management model then computes $P(Y|X^{(i)}_{\text{var}}, X_{\text{inv}})$ for each sample $X^{(i)}_{\text{var}}$, and $P(Y|X_{\text{inv}})$ is approximated by averaging:

$$P(Y|X_{\text{inv}}) \approx \frac{1}{M} \sum_{i=1}^{M} P(Y|X^{(i)}_{\text{var}}, X_{\text{inv}}).$$

To minimize inference time, we set the noise vector Z to a small size relative to the data dimension, ensuring $P(Y|X_{\text{inv}})$ can be estimated with small M. Our experiments show that with a small noise vector, the network management model's predictions for different GAN-generated samples (for the same input sample) are effectively identical. Thus, we set $M = 1$, enabling prediction of $P(Y|X_{\text{inv}})$ (for a single input sample) without increasing inference time. By setting $M = 1$, our inference procedure, shown in Fig. 1(c), becomes as follows:

Generate Variant Features for Target Samples: For each sample $x^t \in \mathcal{X}_C$ from the target domain, extract the values of its invariant features X^t_{inv} and pass them through the trained generator along with Gaussian noise Z:

$$\hat{X}^t_{\text{var}} = G([X^t_{\text{inv}}, Z]) \tag{10}$$

Construct Augmented Sample: Combine the generated variant features $\hat{X}^t_{\text{var}}$ with the original invariant features X^t_{inv} to form a new sample resembling the source domain distribution:

$$\hat{x}^t = [X^t_{\text{inv}}, \hat{X}^t_{\text{var}}] \tag{11}$$

Predict with Source-Trained Classifier: Apply the network management model f (the failure classification or the fault detection model in our datasets), trained on the source domain data with all the input features included (and with samples with the same feature order as $\hat{x}^t$), to the newly created sample $\hat{x}^t$:

$$\hat{y}^t = f(\hat{x}^t) \tag{12}$$

By transforming the variant features of the target domain samples to resemble those of the source domain, our approach enables the network management model to leverage the full information content of the features without suffering from cross-domain performance degradation. While our focus is on failure classification and fault detection tasks, the newly created sample can naturally be utilized for other network management tasks (regression, forecasting, etc.) as well.

3) GAN Architecture Details: For the architecture of the generator and discriminator models, we follow the architecture of CTGAN [36]. The CTGAN model, designed specifically for tabular data, uses fully-connected neural networks for both the generator and the discriminator. The generator has two hidden layers with batch normalization and ReLU activation, followed by output layers using *tanh* for continuous values and Gumbel softmax for discrete values. The discriminator also has two fully-connected layers with leaky ReLU activation and dropout, ending with a sigmoid output layer. This architecture is suitable for tabular data because it can handle the lack of local structure typically found in image data, allowing the network to learn complex interdependencies between diverse data types (continuous and discrete).

In our implementation, the GAN for the 5GC dataset (442 input features) uses a noise dimension of 30, with the generator and discriminator structured as two layers of 256 neurons each. For the 5GIPC dataset (116 input features), the noise dimension is 15, and the generator and discriminator are adjusted to two layers of 128 neurons each. Both setups use learning rates of 2×10^{-4} for the generator and discriminator, with a decay of 1×10^{-6} to regularize and prevent overfitting.

VI. Evaluation

In this section, we evaluate our proposed DA approach on two public datasets (described in Section IV). We compare our proposed approach against state-of-the-art DA and few-shot learning methods in different few-shot scenarios. Since our approach is based on causal inference and tailored for few-shot DA, we have included state-of-the-art causal learning DA methods and few-shot learning methods in the compared approaches. Additionally, domain-independent latent representation learning DA methods are included, as they are among the most widely used approaches in DA, particularly in network management [14], [15].

A. Compared Approaches

The compared approaches include several baselines and state-of-the-art methods in DA and few-shot learning, grouped into the following four categories:

1) Naive Baselines:

SrcOnly: Trains the model only on source domain data and evaluates it directly on the target domain without adaptation.

TarOnly: Trains the model only on target domain data.

S&T: Combines both source and target domain data for training without specific adaptation, assigning higher weights to target domain samples for better performance.

Fine-Tune: First trains the model on source domain data and then fine-tunes it with limited target domain data, optimizing all model parameters.

2) Domain-Independent Latent representation Learning:

CORAL [37], [39]: Correlation Alignment (CORAL) minimizes domain shift by aligning the second-order statistics of source and target distributions.

DANN [14], [15]: Domain-Adversarial Neural Network (DANN) uses adversarial training to learn domain-independent features by training a domain classifier to distinguish domains and a feature extractor to confuse the classifier.

SCL [38]: Combines supervised contrastive learning (SCL) with domain adversarial training for DA in classification tasks.

3) Few-Shot Learning Methods:

MatchNet [22]: An attention-based few-shot learning method that first trains on source domain, mapping support and query samples into embeddings. Limited labeled target samples form a support set, and new target samples are classified by comparing their embeddings to this support set.

ProtoNet [21]: Learns a metric space where classification is based on distances to class prototypes. The model is trained on source domain data to compute mean embeddings for each class, and new prototypes are formed by updating the source prototypes with limited labeled target data. Target samples are classified based on their proximity to these prototypes.

4) Causal Learning-Based Approaches:

CMT [26]: Causal Mechanism Transfer (CMT) leverages causal modeling and nonlinear independent component analysis for data augmentation in the target domain. The new augmented data is used to train the classification model.

ICD [16]: This method utilizes a joint causal inference framework to predict invariant conditional distributions (ICD) in the data. Although it is designed for DA in medical data analysis, where the number of features is relatively small, we can adopt their methodology to separate the data features into variant and invariant categories, similar to our FS algorithm. We then train the network management model only on the invariant features.

FS (ours): Our Feature Separation (FS) method that employs causal inference to distinguish domain-variant and invariant features (Section V-A), with the network management model trained only on invariant features and only on source data.

FS+GAN (ours): In addition to separating variant and invariant features, this version leverages our GAN-based method to retain the benefits of variant features (Section V-C). Here, the network management model is trained only on source data with all input features included.

All the compared approaches, except for our FS and FS+GAN methods, incorporate target domain samples in the training of the network management models. In contrast, FS and FS+GAN train the network management models exclusively on source domain data, using target domain data only to identify domain-variant features without including it in the training process of the network management models. Additionally, if the FS method identifies new domain-variant features over time based on updated target domain data, it requires retraining the network management models to accommodate the revised feature set. However, FS+GAN is the

Results for the 5GC Dataset

	Method	#Samples from target domain: 1				#Samples from target domain: 5				#Samples from target domain: 10			
		TNet	MLP	RF	XGB	TNet	MLP	RF	XGB	TNet	MLP	RF	XGB
Causal Learning	FS+GAN (ours)	**89.7**	**89.6**	**84.5**	**83.6**	**93.1**	**92.5**	**89.2**	**89.3**	**93.4**	**92.7**	**89.3**	**89.6**
	FS (ours)	86.8	86.4	81.7	81.0	88.2	86.7	82.0	82.1	88.6	87.4	82.5	82.9
	CMT [26]	63.7	61.0	57.6	58.1	71.8	70.3	68.6	68.1	76.2	74.5	71.7	71.5
	ICD [16]	34.2	35.7	32.9	32.8	65.8	63.2	62.6	62.5	74.9	72.0	71.3	71.3
Naive Baselines	SrcOnly	10.6	11.8	22.4	22.6	10.6	11.8	22.4	22.6	10.6	11.8	22.4	22.6
	TarOnly	16.5	15.6	25.6	26.0	56.1	54.5	57.3	57.5	60.8	59.2	59.4	59.5
	S&T	37.0	35.4	32.3	32.7	59.5	58.8	61.5	61.6	66.0	64.2	63.7	64.1
	Fine-tune	-	37.8	-	-	-	56.5	-	-	-	64.5	-	-
Domain Independent	CORAL [37]	38.5	37.9	36.3	36.4	64.7	62.5	62.1	62.2	70.9	69.5	69.2	69.6
	DANN [14]	33.6				61.9				71.3			
	SCL [38]	31.7				60.4				71.6			
Few-shot Learning	MatchNet [22]	43.8				68.9				72.3			
	ProtoNet [21]	45.4				65.3				70.8			

Results for the 5GIPC Dataset

	Method	#Samples from target domain: 1				#Samples from target domain: 5				#Samples from target domain: 10			
		TNet	MLP	RF	XGB	TNet	MLP	RF	XGB	TNet	MLP	RF	XGB
Causal Learning	FS+GAN (ours)	**80.5**	**79.0**	**80.2**	**79.7**	**85.5**	**85.0**	**85.8**	**85.5**	**86.1**	**85.7**	**86.5**	**86.3**
	FS (ours)	76.5	75.8	76.3	76.1	81.3	80.8	81.2	80.9	82.5	82.0	82.7	82.4
	CMT [26]	70.3	69.5	70.2	70.0	73.2	72.5	73.3	72.9	74.1	73.7	74.2	74.0
	ICD [16]	66.8	65.8	66.3	65.9	71.5	71.4	71.8	71.4	74.0	72.5	73.3	73.2
Naive Baselines	SrcOnly	51.3	51.6	53.5	53.7	51.3	51.6	53.5	53.6	51.3	51.6	53.5	53.6
	TarOnly	56.2	55.5	55.8	55.6	59.2	58.8	59.5	59.3	62.5	62.0	62.3	62.1
	S&T	61.6	61.0	61.7	61.3	64.8	64.3	65.0	64.7	67.7	67.0	67.2	67.3
	Fine-tune	-	58.2	-	-	-	61.0	-	-	-	63.2	-	-
Domain Independent	CORAL [37]	66.2	65.8	66.2	65.8	68.5	68.0	67.8	68.3	70.5	69.8	70.3	70.2
	DANN [14]	70.7				75.8				78.0			
	SCL [38]	69.8				75.7				77.8			
Few-shot Learning	MatchNet [22]	68.5				70.8				72.7			
	ProtoNet [21]	70.7				73.5				74.8			

TABLE I: F1-score performance of the DA methods on target test data using four classification models (TNet, MLP, RF, XGB) with 1, 5, and 10 target training samples per fault type.

only approach among the compared methods that, by design, eliminates the need for retraining or fine-tuning the network management models, as they are trained solely on source data with all the features (domain-variant invariant) included.

B. Evaluation Results

We evaluate the compared approaches on our dataset, with the training data comprising all source domain samples and a small number of target domain samples. We consider three few-shot scenarios with 1, 5, or 10 training samples per fault type from the target domain (the normal class is also considered as a fault type here). For each scenario, target samples are randomly selected, and experiments are repeated 20 times with different selections to report average performance. All approaches are evaluated on a test dataset containing only target domain samples. For our methods, we normalized feature values to the range [-1, 1], while for other methods we followed their suggested normalization. Among the compared approaches, DANN, SCL, MatchNet, and ProtoNet are model-specific, while others are model-agnostic. For model-agnostic approaches, we used four classification models for failure classification in the 5GC dataset and fault detection in the 5GIPC dataset: TNet [40] (a deep learning model specialized for tabular data), MLP, Random Forest (RF) [41], and XGBoost (XGB) [41]. However, the focus of our paper is not on identifying the best classification model, but rather on evaluating different DA approaches in mitigating the problem of data drift. For model-specific methods, we used the same models as in the original works. Table I reports the F1-scores of all approaches on the target domain test data.

a) Naive Baselines: As expected, the SrcOnly baseline achieves very poor performance, with F1-scores ranging from 10.6 to 22.6 on the 5GC dataset and from 51.3 to 53.7 (random performance) on the 5GIPC dataset. Note that when SrcOnly was cross-validated on the source domain (i.e., when the test dataset was also from the source domain), it achieved F1-scores exceeding 98.1 for 5GC and 94.3 for 5GIPC. This highlights that its poor performance on the target test dataset is due to the data drift issue. The TarOnly baseline performs better than SrcOnly, particularly with more target domain samples. The S&T approach, which combines source and target data, shows improvement over SrcOnly and TarOnly, but is still outperformed by more sophisticated methods. Fine-tune, which fine-tunes a pre-trained model on the source domain with limited target data, shows reasonable improvement over TarOnly, but performs slightly worse than S&T. The Fine-Tune approach is only applicable to the MLP model. In Fine-Tune, we re-optimize all the MLP parameters rather than just updating the last layer, as this resulted in better performance.

b) Domain-Independent Latent representation Learning Methods: CORAL shows moderate performance improvement over naive baselines by aligning the second-order statistics of source and target domains. However, its effectiveness diminishes with fewer target samples, demonstrating F1-scores ranging from 36.3 to 70.9 on the 5GC dataset and from 65.8 to 70.5 on the 5GIPC dataset. DANN has outperformed naive baselines, but its performance is stil limited in the few-shot learning setup. The F1-scores for DANN increase significantly with more target samples, but it does not match the effectiveness of our proposed methods. SCL, which combines contrastive learning with adversarial training, shows performance similar to DANN, indicating that the addition of contrastive learning has not significantly enhanced the effectiveness of adversarial training in our few-shot scenario.

c) Few-Shot Learning Methods: MatchNet and ProtoNet performed better than naive baselines on both datasets and outperformed domain-independent representation learning methods in most cases of the 5GC dataset. However, their effectiveness is constrained by the significant distribution differences between the source and target domains (few-shot learning methods are often biased toward the source domain data). Their F1-scores improve with more target samples, but remain below those of our proposed methods.

d) Causal Learning-Based Approaches: ICD delivers competitive performance compared to domain-independent representation learning and few-shot learning methods on the 5GC dataset. However, it performs worse than these methods on the 5GIPC dataset, particularly in scenarios with fewer target domain samples. ICD's reliance on predicting invariant causal mechanisms works relatively well with more target samples but struggles with fewer samples, as evidenced by its F1-scores ranging from 32.8 to 74.9 on the 5GC dataset and from 65.8 to 74.0 on the 5GIPC dataset. Our experiments show that ICD identifies much less domain-variant features than our FS method, failing to effectively eliminate variant features that degrade performance. CMT performs better than ICD, especially in the case of having only one target sample per fault type. Its performance improves with more target samples, showing F1-scores ranging from 57.6 to 76.2 on the 5GC dataset and from 69.5 to 74.2 on the 5GIPC dataset.

Our proposed FS method, which focuses on invariant features for robust performance against data drift, demonstrates substantial improvement over state-of-the-art methods, with F1-scores ranging from 81.0 to 88.6 on the 5GC dataset and from 75.8 to 82.7 on the 5GIPC dataset. Our FS+GAN method further enhances performance by leveraging GAN-based augmentation to utilize variant features without degrading cross-domain performance. This method achieves the highest F1-scores across all scenarios, ranging from 83.6 to 93.4 on 5GC dataset and from 79.0 to 86.5 on 5GIPC dataset. Our FS+GAN method shows a significant improvement over the CMT method, the best performing method among the state-of-the-art approaches. On the 5GC (5GIPC) dataset, the highest F1-score achieved by CMT is 76.2 (74.2), while FS+GAN achieves up to 93.4 (86.5), marking an improvement of 17.2 (12.3) points. On average, FS+GAN attains an F1-score of 89.8 (83.8) on the 5GC (5GIPC) dataset, compared to 67.6 (72.3) by CMT, representing an average improvement of 32.9% (15.9%) in F1-score over the state-of-the-art. In terms of mitigating data drift, FS+GAN demonstrates a significant average improvement in F1-score over the SrcOnly baseline, with improvements of 72.8 (31.5) points on the 5GC (5GIPC) dataset, compared to improvements of 50.1 (19.8) points achieved by CMT. This translates to a 45.3% (59.1%) improvement in data drift mitigation compared to the state-of-the-art on the 5GC (5GIPC) dataset, with an overall average improvement of 52% across both datasets.

Method	5GC Dataset			5GIPC Dataset		
	#Samples from target:			#Samples from target:		
	1	5	10	1	5	10
FS+GAN	**89.7**	**93.1**	**93.4**	**80.5**	**85.5**	**86.1**
FS+NoCond	89.3	91.7	93.0	**80.5**	84.1	84.9
FS+VAE	88.4	90.1	91.3	79.3	82.8	83.0
FS+VanillaAE	87.6	89.1	89.5	77.4	81.6	83.0

TABLE II: Ablation study of different reconstruction strategies, reporting F1-scores with the TNet classification model.

C. Sensitivity Analysis

We perform a sensitivity analysis to evaluate the robustness and performance of our proposed methods under various conditions and configuration, as described in the following:

Varying Number of Samples from the Target Domain: In Table I, we evaluated the performance of our methods, FS and FS+GAN, by varying the number of target domain training samples (1, 5, and 10 samples per fault type). Our experiments showed that with more target samples, our FS method was able to identify more domain-variant features, mitigating data drift more effectively. Specifically, FS identified 35, 68, and 75 domain-variant features in the 5GC dataset and identified 23, 31, and 37 domain-variant features in the 5GIPC dataset with 1, 5, and 10 target samples per fault type, respectively. FS+GAN's performance also improved with more target samples, as it benefits from the enhanced feature separation.

Different Failure Classification Models: In Table I, we evaluated our proposed methods using different failure classification models (TNet, MLP, RF, and XGB) to ensure their generalizability. The results demonstrate that our methods are model-agnostic and can improve classification performance regardless of the underlying model architecture. Additionally, the experiments demonstrated that TNet consistently outperforms all other classifiers, while MLP, RF and XGB perform similarly to each other but are slightly less effective than TNet.

Variance Across Different Random Target Sample Selections: The variance in performance due to different random selections of target samples was analyzed for our methods. Our proposed methods (FS and FS+GAN) exhibited minimal variance, consistently remaining within ±2.6. So, the observed variance values are small enough to not affect the interpretation of the performance analysis presented in Table I.

D. Running Time of Our Proposed Methods

We implemented our approaches on a server with dual 20-core Intel Xeon Silver 4114 2.20GHz CPUs, 187 GB of memory, and an NVIDIA Tesla P40 GPU. Our FS method took

DA Method	Performance on *Target_1*			Performance on *Target_2*		
	#Sample from target:			#Samples from target:		
	1	5	10	1	5	10
FS+GAN_1	**78.6**	**83.8**	**85.0**	74.8	79.1	80.2
FS+GAN_2	74.4	79.5	81.7	**76.7**	**84.1**	**85.3**

TABLE III: F1-score of the TNet fault detection model, trained only on *Source*, for different DA scenarios.

about 42 minutes to process the 5GC dataset, and 35 minutes to process the 5GIPC dataset. The computational overhead of FS is primarily due to the conditional independence tests needed to construct the causal graph. However, these tests focus solely on direct relationships with the F-node, rather than constructing the entire causal graph, making the process more efficient. Training the conditional GAN model on the source domain data took around 12 minutes for 5GC and 7 minutes for 5GIPC. This efficiency is due to the generator only reconstructing variant features, a small subset of the total features. The GAN training used a batch size of 64 over 500 epochs. The FS algorithm and GAN training are performed once and reused for multiple target domain instances during inference. The inference time for a target sample involves only a single feed-forward pass through the generator, averaging roughly 0.05 seconds per sample. Although FS and GAN retraining may be needed over time, both are lightweight compared to full model retraining and are infrequently triggered, making the approach scalable for large networks with moderate shift frequency. Additionally, our approach is energy-efficient in operational settings, as it avoids repeated retraining or fine-tuning of the network management models.

E. Ablation Study on Different Reconstruction Strategies

We conduct an ablation study to evaluate different reconstruction strategies for domain-variant features within our FS+GAN method. We compare FS+GAN with two alternative techniques: one using a Variational Autoencoder (VAE) [42] and another using a vanilla Autoencoder (VanillaAE) [43] to reconstruct the variant features (instead of the GAN model). Additionally, we examine a version of FS+GAN, termed FS+NoCond, where the discriminator is not conditioned on the label ($[X^s_{\text{inv}}, \hat{X}^s_{\text{var}}]$ becomes the input of discriminator instead of $[X^s_{\text{inv}}, \hat{X}^s_{\text{var}}, Y^s]$ in FS+NoCond). The neural network architecture of the VAE and VanillaAE matches our generator model (described in Section V-C3).

Table II shows the F1-score results of these comparisons, reporting only the performance with the best-performing TNet classification model. Our findings indicate that both the VAE and vanilla Autoencoder-based methods do not significantly enhance the performance of the FS method compared to FS+GAN. FS+GAN consistently achieves higher F1-scores across different scenarios, demonstrating its superior capability in reconstructing domain-variant features. Additionally, the FS+NoCond version performs worse than FS+GAN. This highlights the importance of conditioning the discriminator on the label to fully utilize the GAN framework.

F. No Retraining or Fine-Tuning Required for Network Management ML Models

In the experiments summarized in Table I, we observed that our FS+GAN method identified varying numbers of domain-variant features depending on the number of target domain samples. Despite these variations, the network management models trained solely on source domain data consistently delivered strong performance across all scenarios and datasets. This observation implicitly demonstrates that our approach eliminates the need for retraining or fine-tuning the network management models. To further validate this capability, we conducted an additional set of experiments using the 5GIPC dataset. Here, the dataset was divided into three clusters using GMM: one large cluster serving as the source domain (denoted as *Source*), and the other two smaller clusters representing two distinct target domains (*Target_1* and *Target_2*). We applied DA using our FS+GAN method, resulting in two variations: FS+GAN_1 (trained on *Source* and *Target_1*) and FS+GAN_2 (trained on *Source* and *Target_2*).

Table III presents the performance of the TNet fault detection model, trained exclusively on *Source*, when evaluated on the *Target_1* and *Target_2* datasets with domain adaptation performed by FS+GAN_1 and FS+GAN_2, respectively. The results demonstrate that the TNet model achieves excellent performance on both target domains when DA is performed by their corresponding FS+GAN method (FS+GAN_1 for *Target_1* and FS+GAN_2 for *Target_2*), with F1-scores ranging between 76.7 and 85.3. These findings highlight the robustness of our approach against evolving data distributions over time, without requiring retraining or fine-tuning of the network management models. Furthermore, the results reveal that TNet achieves competitive performance on *Target_2* when DA is performed using FS+GAN_1 (compared to using FS+GAN_2), and similarly, competitive performance on *Target_1* when DA is performed using FS+GAN_2 (compared to using FS+GAN_1). This robustness arises because the majority of domain-variant features identified by FS+GAN_1 and FS+GAN_2 were common across the target domains. This suggests that FS+GAN only needs to be updated when the data distribution undergoes significant changes.

VII. Conclusion

In this paper, we presented a novel approach to address the critical issue of data drift in machine learning models for 5G and beyond mobile networks. Our method leverages causal inference to separate domain-variant and domain-invariant features, ensuring robust performance across different domains even with limited target domain data. Additionally, we introduced a GAN-based technique to harness the information in domain-variant features without causing cross-domain performance degradation. Our comprehensive evaluation demonstrated a substantial improvement of 52% in mitigating data drift compared to state-of-the-art methods, highlighting the effectiveness of our approach. This work represents a significant advancement in the development of reliable ML models for practical network management tasks.

REFERENCES

[1] S. Sharma, R. Miller, and A. Francini, "A cloud-native approach to 5g network slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 120–127, 2017.

[2] S. Mwanje, G. Decarreau, C. Mannweiler, M. Naseer-ul Islam, and L. C. Schmelz, "Network management automation in 5g: Challenges and opportunities," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2016, pp. 1–6.

[3] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE communications magazine*, vol. 53, no. 2, pp. 90–97, 2015.

[4] M. Chen, X. Wei, Y. Gao, L. Huang, M. Chen, and B. Kang, "Deep-broad learning system for traffic flow prediction toward 5g cellular wireless network," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 940–945.

[5] S. S. Johari, N. Shahriar, M. Tornatore, R. Boutaba, and A. Saleh, "Anomaly detection and localization in nfv systems: an unsupervised learning approach," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2022, pp. 1–9.

[6] B. Bousalem, V. F. Silva, R. Langar, and S. Cherrier, "Ddos attacks detection and mitigation in 5g and beyond networks: A deep learning-based approach," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 1259–1264.

[7] S. Liu, F. Bronzino, P. Schmitt, A. N. Bhagoji, N. Feamster, H. G. Crespo, T. Coyle, and B. Ward, "Leaf: Navigating concept drift in cellular networks," *Proceedings of the ACM on Networking*, vol. 1, no. CoNEXT2, pp. 1–24, 2023.

[8] A. Mallick, K. Hsieh, B. Arzani, and G. Joshi, "Matchmaker: Data drift mitigation in machine learning for large-scale systems," *Proceedings of Machine Learning and Systems*, vol. 4, pp. 77–94, 2022.

[9] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014.

[10] N. Apostolakis, L. E. Chatzieleftheriou, D. Bega, M. Gramaglia, and A. Banchs, "Digital twins for next-generation mobile networks: Applications and solutions," *IEEE Communications Magazine*, vol. 61, no. 11, pp. 80–86, 2023.

[11] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A brief review of domain adaptation," *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pp. 877–894, 2021.

[12] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*. PMLR, 2015, pp. 1180–1189.

[13] Q. Zhou, W. Zhou, S. Wang, and Y. Xing, "Unsupervised domain adaptation with adversarial distribution adaptation network," *Neural Computing and Applications*, vol. 33, pp. 7709–7721, 2021.

[14] A. Singla, E. Bertino, and D. Verma, "Preparing network intrusion detection deep learning models with minimal data using adversarial domain adaptation," in *Proceedings of the 15th ACM Asia conference on computer and communications security*, 2020, pp. 127–140.

[15] H. Larsson, F. Moradi, J. Taghia, X. Lan, and A. Johnsson, "Domain adaptation for network performance modeling with and without labeled data," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2023, pp. 1–9.

[16] S. Magliacane, T. Van Ommen, T. Claassen, S. Bongers, P. Versteeg, and J. M. Mooij, "Domain adaptation by using causal inference to predict invariant conditional distributions," *Advances in neural information processing systems*, vol. 31, 2018.

[17] A. Jaber, M. Kocaoglu, K. Shanmugam, and E. Bareinboim, "Causal discovery from soft interventions with unknown targets: Characterization and learning," *Advances in neural information processing systems*, vol. 33, pp. 9551–9561, 2020.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[19] L. Du, Z. Gu, Y. Wang, L. Wang, and Y. Jia, "A few-shot class-incremental learning method for network intrusion detection," *IEEE Transactions on Network and Service Management*, 2023.

[20] Y. Hu, J. Wu, G. Li, J. Li, and J. Cheng, "Privacy-preserving few-shot traffic detection against advanced persistent threats via federated meta learning," *IEEE Transactions on Network Science and Engineering*, 2023.

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

[22] 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.

[23] K. Zhang, M. Gong, P. Stojanov, B. Huang, Q. Liu, and C. Glymour, "Domain adaptation as a problem of inference on graphical models," *Advances in neural information processing systems*, vol. 33, pp. 4965–4976, 2020.

[24] K. Zhang, M. Gong, and B. Schölkopf, "Multi-source domain adaptation: A causal view," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[25] J. Ma, "Causal domain adaptation with copula entropy based conditional independence test," *arXiv preprint arXiv:2202.13482*, 2022.

[26] T. Teshima, I. Sato, and M. Sugiyama, "Few-shot domain adaptation by causal mechanism transfer," in *International Conference on Machine Learning*. PMLR, 2020, pp. 9458–9469.

[27] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," *arXiv preprint arXiv:2006.10726*, 2020.

[28] L. Zancato, A. Achille, T. Y. Liu, M. Trager, P. Perera, and S. Soatto, "Train/test-time adaptation with retrieval," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 15 911–15 921.

[29] J. Zhang, L. Qi, Y. Shi, and Y. Gao, "Domainadaptor: A novel approach to test-time adaptation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 18 971–18 981.

[30] ITU AI for Good, "AI for Good Challenge - Network Fault Management," https://challenge.aiforgood.itu.int/match/matchitem/84, [Online; accessed 21-July-2024].

[31] Itu ai/ml in 5g challenge. IEICE Rising Group. [Online]. Available: https://www.ieice.org/ rising/AI-5G/

[32] J. Pearl, "Causal diagrams for empirical research," *Biometrika*, vol. 82, no. 4, pp. 669–688, 1995.

[33] L. G. Neuberg, "Causality: models, reasoning, and inference, by judea pearl, cambridge university press, 2000," *Econometric Theory*, vol. 19, no. 4, pp. 675–685, 2003.

[34] A. Ikram, S. Chakraborty, S. Mitra, S. Saini, S. Bagchi, and M. Kocaoglu, "Root cause analysis of failures in microservices through causal discovery," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 158–31 170, 2022.

[35] P. Spirtes, C. Glymour, and R. Scheines, *Causation, prediction, and search*. MIT press, 2001.

[36] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional gan," *Advances in neural information processing systems*, vol. 32, 2019.

[37] K. A. Lee, Q. Wang, and T. Koshinaka, "The coral+ algorithm for unsupervised domain adaptation of plda," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5821–5825.

[38] J.-W. Kim, S. Bae, W.-Y. Cho, B. Lee, and H.-Y. Jung, "Stethoscope-guided supervised contrastive learning for cross-domain adaptation on respiratory sound classification," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 1431–1435.

[39] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[40] L. Du, F. Gao, X. Chen, R. Jia, J. Wang, J. Zhang, S. Han, and D. Zhang, "Tabularnet: A neural network architecture for understanding semantic structures of tabular data," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 322–331.

[41] S. Raschka and V. Mirjalili, *Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2*. Packt publishing ltd, 2019.

[42] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[43] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.