Contrastive learning with self-reconstruction for channel-resilient modulation classification

Erma Perenda*, Sreeraj Rajendran†, Gerome Bovet‡, Mariya Zheleva§ and Sofie Pollin*
{erma.perenda, sofie.pollin}@esat.kuleuven.be, sreeraj.rajendran@sirris.be gerome.bovet@armasuisse.ch, mzheleva@albany.edu
* WaveCore, ESAT, KU Leuven, †Sirris, Belgium
‡Cyber-Defence Campus, armasuisse Science&Technology,
§Department of Computer Science, University at Albany - SUNY

IEEE INFOCOM 2023 - IEEE Conference on Computer Communications | 979-8-3503-3414-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/INFOCOM53939.2023.10228908

***Abstract*—Despite the substantial success of deep learning for Automatic Modulation Classification (AMC), models trained on a specific transmitter configuration and channel model often fail to generalize well to other scenarios with different transmitter configurations, wireless fading channels, or receiver impairments such as clock offset. This paper proposes Contrastive Learning with Self-Reconstruction called CLSR-AMC to learn good representations of signals resilient to channel changes. While contrastive loss focuses on the differences between individual modulations, the reconstruction loss captures representative features of the signal. Additionally, we develop three data augmentation operators to emulate the impact of channel and hardware impairments without exhaustive modeling of different channel profiles. We perform extensive experimentation with commonly used realistic datasets. We show that CLSR-AMC outperforms its counterpart based on contrastive learning for the same amount of labeled data by significant average accuracy gains of 24.29%, 17.01%, and 15.97% in the Additive White Gaussian Noise (AWGN), Rayleigh, and Rician channels, respectively.**

I. Introduction

RF Sensing will be an integral part of future wireless networks, enabling the learning and building of intelligence in the network to support emerging applications such as autonomous vehicles, smart homes, and human-computer interaction. A recent research direction, Integrated Sensing And Communication (ISAC), has attempted to efficiently unify sensing and communication systems so that they can share the same frequency band and hardware, and also benefit from each other, i.e., communication-assisted sensing and sensing-assisted communication [1,2]. Adopting Automatic Modulation Classification (AMC), as an intermediate step between signal detection and signal demodulation, in the ISAC receiver would benefit in further improving spectral efficiency and reducing receiver complexity [1] in a similar way as it is done in cognitive radios [3,4]. Besides cognitive radios, AMC has been crucial for many spectrum monitoring and security applications. Thus, a reliable and robust modulation classifier is essential to support those applications.

AMC has been studied for more than four decades. In general, three methodological streams can be distinguished: (1) Likelihood-Based (LB), Feature-Based (FB), and Deep Learning (DL). LB methods define AMC as a multi-hypothesis testing problem and can reach the optimal classification accuracy under the assumption of the perfect knowledge of the signal and channel models [5,6], however, it incurs prohibitive computational cost as the number of modulation classes increases [5]. On the other hand, FB methods are developed ad-hoc, which does not guarantee optimality in the Bayesian sense [7]–[9]. The human-crafted discriminative features are extracted from underlying raw data (e.g., In-phase/Quadrature (I/Q)). However, those features may have different values under different transmitter and channel parameters leading to performance degradation [10]. In contrast to LB, FB is more favorable to deploy in practical systems due to its relatively easy implementation and lower complexity. Due to its ability to automatically extract discriminative features and perform classification under lower computational cost, DL has been preferred over the other two [6,11].

The great successes of DL for AMC are achieved under the assumption that the training dataset (source/reference domain) and test dataset (target domain) share the same data distribution [12]–[15]. In other words, transmitter configuration parameters (signal shaping and sampling frequency) and channel conditions are assumed to be known as a-priori. However, this assumption is too strong and does not hold in practice. DL-based AMC only learns to model signals that are represented in the training data as accurately as possible. However, it cannot make predictions with high-level confidence about signals coming from unknown channel conditions or with unknown transmitter configurations [10]. The wireless channel is inherently dynamic, with infinite possible channel realizations. Moreover, wireless communications systems are constantly evolving, and each new release/generation aims to increase spectral efficiency either by introducing new modulation formats (e.g., 256/1024 Quadrature Amplitude Modulation (QAM) in Wi-Fi 6, up to 256 Amplitude Phase Shift Keying (APSK) in satellite systems) or adopting self-optimization algorithms which adapt transmitter configuration parameters (e.g., bandwidth, coding rate, center frequency) according to the current channel conditions [16,17]. Furthermore, for AMC applications such as signal interception in the military, there is no cooperation between transmitter and interceptor to obtain transmitter configuration parameters. Thus, the question arises: *How to perform robust modulation classification when training on all target scenarios is impossible?*

Unsupervised Domain Adaptation (DA) is a relatively new branch in DL, which aims to align the data distributions of source and target domains. It has achieved great success in image processing [18,19], and a few methods have been adopted for AMC [20]–[22]. Those methods assume that a large amount of labeled data exists for the source domain while there is a large amount of unlabeled data for the target

domains. Although unsupervised DA can boost the classification performance for target unknown domains, it requires retraining the whole Deep Neural Network (DNN) architecture every time a new domain arises. Moreover, a domain detector is necessary to prevent the out-dating of the trained classifier. Further, unsupervised DA assumes that the unlabeled data are class-balanced, however, this might not hold for data collected in the wild. As there are infinite combinations of transmitter and channel parameters, it is impossible to guarantee that within such a large amount of unlabeled data, each class has a balanced and sufficient amount of high-quality data for each combination of transmitter and channel parameters.

In this paper, we propose data augmentation by using simple spatial transformations of the signal constellations to generate domain-diverse high-quality data by using a large amount of labeled data from one source domain. Note that the domain denotes one combination of transmitter and channel parameters in this paper. As the source domain, we choose a simple AWGN with a Signal-Noise Ratio (SNR) of 18 dB. Unlike more complex channel models such as Rayleigh and Rician, the collection of labeled data for AWGN is cheaper as it does not have many hyperparameters such as path delay profiles and Doppler spread. Spatial transformation operators emulate the impact of channel and hardware impairments on the signal constellations. The cleaner the signals, the better the precision of data augmentation. We choose 18 dB as it specifies the wireless environment (e.g., Wi-Fi and LTE) with good data speeds. It is a trade-off between collecting a high-SNR trace but not requiring the extreme case of ideal/infinite SNR that requires a testbed with an expensive isolation chamber. In other words, 18 dB allows the collection of training traces in practical, real-world conditions while facilitating sufficient improvement through the transform operations. Having the pairs with different applied spatial transformations, we adopt contrastive learning to learn good feature encodings that are transformation-invariant. Contrastive learning [23] compensates different transformations by minimizing the distances between signal pairs that belong to the same modulation (positive pairs) and maximizing the distances between signal pairs with different modulations (negative pairs). Additionally, we add self-reconstruction to reconstruct the original signal from its spatial transformed version to support learning transformation-invariant features. Our proposed framework is Contrastive Learning with Self-Reconstruction for AMC, referred to as CLSR-AMC.

This paper makes the following contributions:

- We are the first to evaluate the robustness of contrastive loss in cases where the source and target domains are characterized with different channel conditions (i.e., cross-channel scenarios).
- We design a composite loss function that comprises contrastive, reconstruction, and cross-entropy losses, and show its benefits over models that treat these losses independently.
- We show that the proposed data augmentation significantly improves the supervised classification performance up to approx. 40% in unknown channel conditions.

II. Related work

There are three possible directions to construct a reliable and robust AMC: (1) labeling, (2) blind estimation of unknown transmitter and channel parameters, and (3) unsupervised DA. Below we summarize the pros and cons of each direction. As the proposed CLSR-AMC relies on contrastive learning, we summarize its applications for AMC to date.

A. Labeling

Fully supervised State-of-the-Art (SoA) DL-based AMC models [12]–[15] perform well only with sufficient training data that covers all possible combinations of channel and transmitter parameters. Two techniques can be adopted to improve classification performance in cases where a small (but not sufficient) amount of labeled data exists: (1) Transfer Learning (TL) and (2) DL-based data augmentation. TL trains a DNN on one or multiple domains with sufficient labeled data and then uses a smaller amount of labeled data from other domains to retrain only a tiny part of the DNN [10]. In contrast, DL-based data augmentation methods generate additional high-quality labeled data required for AMC training from a small amount of seed data [24]–[26]. However, all those techniques are unrealistic due to: (1) the labeling process, which even for a small amount of data, is costly, time-consuming and tedious and usually requires the participation of human experts; (2) the fact that the number of possible combinations of transmitter and channel parameters is infinite.

B. Blind estimation of transmitter/channel parameters

Blind estimation of unknown transmitter and channel parameters helps to recover signals and reduce channel impairments. Modulation classification and blind estimation of channel and transmitter parameters can be done separately [27] or jointly [28,29]. In [27], signal recovery utilizes the cyclostationary features for Carrier Frequency Offset (CFO) and symbol rate estimation. Afterward, the modulation classifier extracts statistical features of the recovered signal and adopts a decision tree as a classifier. The main drawback of cyclostationary features is that they cannot distinguish higher-order modulations belonging to the same family, such as QAM, PSK, and APSK. Further, a high number of samples is required, which results in high computational costs. A costly LB approach that jointly considers modulation classification and symbol decoding under the assumption of perfect synchronization in an AWGN channel with high SNR values is presented in [28]. In contrast, [29] does not have such an assumption and combines the features learned from both raw and recovered signals. Signal recovery is made by linear signal processing operations where CFO, noise, and fading are compensated sequentially. CFO, noise filter parameters, and equalization filter parameters for fading compensation are estimated using a fully supervised DNN; thus, their estimation accuracy heavily depends on available labeled datasets. Finally, [30] considers the dynamic optimization of the transmit

filter to adapt the signal to new and unseen channel conditions while keeping the modulation classifier fixed. However, this approach is not applicable to our problem setting as it assumes full cooperation between transmitter and receiver.

C. Unsupervised DA

Unsupervised DA has been actively studied in image processing, and generally, three research streams can be identified: discrepancy-based, adversarial-based and reconstruction-based [18,19]. As discussed in the introduction, the main drawback of unsupervised DA is the requirement for a large amount of class-balanced unlabeled data for target domains.

1) Discrepancy-based DA: uses a certain criterion, such as Maximum Mean Discrepancy (MMD) (measures the distance between feature means), to fine-tune the DNN with unlabeled target data to diminish the shift between source and target domains. Although the discrepancy-based DA techniques have been widely applied in image processing [31,32], to our knowledge, none has been adopted for AMC.

2) Adversarial-based DA: adopts a domain discriminator, which minimizes the distance between source and target distributions through an adversarial objective. In [22], Domain-Adversarial Neural Network (DANN) is proposed for AMC cross-channel scenario. DANN consists of a modulation classifier, a domain classifier, and a shared feature encoder. DANN integrates a Gradient Reversal Layer (GRL), which treats domain invariance as a binary classification problem while simultaneously maximizing domain confusion loss. In contrast to DANN, Adversarial Discriminative Domain Adaptation (ADDA) [21] and Adversarial Transfer Learning Architecture (ATLA) [20] separately train feature encoders for source and target domains. ADDA addressed the cross-channel scenario with a large amount of unlabeled target data, while ATLA addressed the cross-sampling rate scenario (different sampling frequencies have been adopted in source and target domains) with a small amount of labeled target data. ATLA is a supervised method but utilizes an adversarial-based adaptation approach.

3) Reconstruction-based DA: assumes that the data reconstruction of the source or target samples can help improve the performance of DA. One example is Deep Reconstruction Classification Network (DRCN) [33], which combines a shared encoder with two pipelines. The first pipeline connects the encoder with a supervised classifier trained with source labels. The second pipeline connects the encoder with a decoder which minimizes the reconstruction error of source and target data in an unsupervised fashion. A Long-Short Term Memory (LSTM)-based DRCN for AMC is given in [34], while a Convolutional Neural Network (CNN)-based DRCN for AMC is given in [35].

D. Contrastive learning and AMC

Contrastive learning has been developed in the context of Siamese networks. A Siamese neural network consists of two or more identical sub-networks, to learn similarities and differences between data inputs [36]. There have been a few approaches for AMC based on Siamese networks and contrastive learning [37]–[39]. In [37], contrastive loss is used to measure the similarity of weights at different levels of CNN. Fully supervised CNN-based Siamese networks with contrastive loss were presented in [38]. [39] pre-trains the feature encoder through self-supervised contrastive training (Semi-CLR) using a large amount of unlabeled data. The Siamese network pairs are realized through utilizing the random rotations of the input I/Q data sample. Afterward, the feature encoder is frozen, and the classifier is trained using a small amount of labeled data. Both [38] and [39] showed that Siamese networks with contrastive loss outperform the supervised approaches for the same amount of labeled training data. However, both consider only one signal and channel realization without evaluating the robustness of the feature encoder to unseen signal and channel realizations. Moreover, the Siamese networks and classifier are trained independently, while our paper shows that their joint training ensures better classification performance. Further, Semi-CLR does not consider the class attribution. Semi-CLR considers only augmented versions of one sample as positive pairs, while all other samples in the training batch are treated as negative pairs. However, it is highly likely that the batch contains samples belonging to the same class which should be treated as positive pairs but not negative pairs. To solve this problem, our CLSR-AMC uses class information to create correct pairs. In sum, CLSR-AMC enhances the Semi-CLR framework by (1) adding a self-reconstruction task, (2) adopting a weighted sum of reconstruction, classification, and contrastive losses as a loss function, and (3) employing supervised contrastive learning.

III. Methodology

This section describes the core of CLSR-AMC, including the signal model fed to the input of AMC, preliminaries, the proposed structure, and the optimization process.

A. Signal model

Assume that one active transmitter sends a vector of complex symbols $s \in \mathbb{C}^{N_s}$. The symbols are encoded by adopting modulation format m from a pool of known modulations $\mathcal{M}$. The encoded symbols are shaped with a pulse of duration T and upconverted to center frequency f_c, forming the transmitted signal $s(t)$. This signal is transmitted over a dynamic wireless fading channel modeled with an impulse response h. Assuming one antenna at the receiver, after down-conversion, the distorted and noise-corrupted received signal, $r(t)$, is:

$$r(t) = e^{j(\phi_0 - 2\pi\Delta f t)} s(t - \Delta t) \circledast h(t, \tau) + v(t), \tag{1}$$

where Δt is the timing offset, Δf is the common frequency offset, ϕ_0 is the phase offset, $v(t)$ is AWGN with mean 0 and variance $2\sigma_v^2$, and τ represents the delays of the multipath wireless channel. The received signal, $r(t)$, is sampled in the time domain with Nyquist frequency $1/T_r$. The sampling instance at timestep t_k is given as $t_k = \Delta t + k \cdot T_r$. Thus, the length of source samples N_s and received samples N_r are related as $N_r = N_s \cdot \lceil \frac{T}{T_r} \rceil$.

The N_r raw I/Q samples are referred to as an instance, represented as a matrix S with dimensions $2 - by - N_r$, where the first row holds I values, and the second row holds the corresponding Q values. The AMC's task is to select a modulation format $\hat{m}$ correctly from $\mathcal{M}$ by examining the received signal, $r(t)$ represented by an instance.

B. Problem definition

Our goal is to enable robust modulation classification with limited training data across different channel conditions and SNR. Particularly, we target a source domain where labeled data is empirically collected for a single SNR (18 dB) and single channel (AWGN) across all target modulations, and then augmented to match a large number of realistic cases. There are multiple target domains with unlabeled data. The unlabeled target data is not available during training. The source domain, denoted by $\mathcal{D}_s$, is composed of n_s labeled instances $\{s_j, m_j\}_{j=1}^{n_s}$, where $s_j \in \mathcal{S}_s$ and $m_j \in \mathcal{M}_s$ denote the instance space and modulation space of the source domain, respectively. Note that the instance space $\mathcal{S}_s$ is set of n_s matrices S, $\mathcal{S}_s = \{S_i | i = 1, ..., n_s\}$. Similarly, the target domains, denoted by $\mathcal{D}_t$, contain n_t instances $\{s_j, m_j\}_{j=1}^{n_t}$, where $s_j \in \mathcal{S}_t$ and $m_j \in \mathcal{M}_t$. In this paper, only homogeneous classification tasks are considered, therefore, there is $\mathcal{M}_s = \mathcal{M}_t = \mathcal{M} = \{1, 2, ..., M\}$, where M is the number of modulations.

Given the labeled source domain $\mathcal{D}_s$, the objective of a deep DA network is to learn a functional mapping $g : \mathcal{S} \rightarrow \mathcal{M}$, where $\mathcal{S} = \mathcal{S}_s \cup \mathcal{S}_t$. The functional mapping g can be decomposed into a feature encoder and a modulation predictor. The feature encoder maps the instances to a latent feature space $\mathcal{Z}$. The modulation predictor maps the latent features to the modulation space. The details of how the functional mapping g is found are explained in the following text.

C. Contrastive Learning with Self-Reconstruction for AMC

Our CLSR-AMC framework includes data augmentation and DNN architecture. Both are detailed below. Additionally, the definitions of losses adopted for optimizing the weights are given. The optimization of DNN architecture and its hyperparameters is detailed, as well.

1) Data augmentation: Data augmentation aims to impersonate various degrees of channel impact on signal constellations such as rotation and distortion. CNN-based AMCs learns spatial features [10], and any spatial transformation of the signal constellations may help the feature encoder to learn better feature encodings. Adding Gaussian noise and rotation of the signal constellations have been previously used for data augmentation in AMC [39,40]. In addition, we add a novel augmentation method named *concatenation and downsampling* to augment the data with real-world Rayleigh/Rician channel effects. We briefly summarize each of them below.

1) **Adding Gaussian noise:** The received signal is distorted by adding Gaussian noise with zero mean value and random variance σ^2.

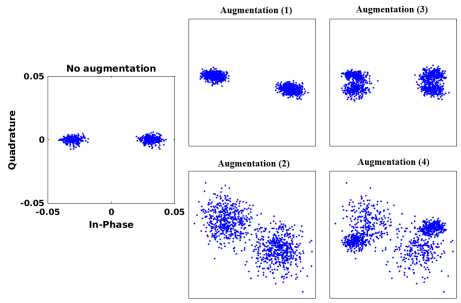

Figure 1: An example of data augmentation output for BPSK.

2) **Rotation of the signal constellation:** Rotation emulates the impact of phase offset. The phase offset might be introduced by fading channels or local oscillators. Augmented I/Q values by rotation with random angle θ are calculated as

$$s_r = \begin{bmatrix} \hat{I} \\ \hat{Q} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} I \\ Q \end{bmatrix}. \quad (2)$$

3) **Concatenation and downsampling (CaD):** One signal is augmented twice by using previous operations. The augmented signal versions are first downsampled by factor 2 and then concatenated. This operation impersonates the impact of Rician fading with one Line-Of-Sight (LOS) signal path.

The allowed ranges for σ and θ define the type of augmentation: (1) weak or (2) strong. Weak augmentation adds a small amount of white Gaussian noise with $\sigma \in [0.006, 0.02]$ and small rotation with $\theta \in [-30 : 10 : 30]^\circ$. On the other hand, strong augmentation adds a bit higher amount of white Gaussian noise with $\sigma \in [0.02, 0.05]$ and a rotation with $\theta \in [-90 : 30 : 90]^\circ$. Data augmentation is run for each instance in the source domain. One run of data augmentation on a certain instance outputs four new augmented instances: (1) pure weak augmented instance, (2) pure strong augmented instance, (3) concatenation of two weakly augmented instances, and (4) concatenation of one weakly augmented instance and one strongly augmented instance. Fig. 1 shows the output of data augmentation for a simple BPSK signal at $SNR = 18$ dB. Generated augmented instances are merged with non-augmented instances, forming the new labeled dataset used for the training of CLSR-AMC. Each augmented instance tracks its corresponding non-augmented instance, which is used for reconstruction loss.

2) Architecture overview: The DNN architecture of CLSR-AMC is illustrated in Fig. 2. The architecture consists of three branches: (1) contrastive learning, (2) classification, and (3) reconstruction. The contrastive learning and reconstruction branches are used only for training, while the classification branch operates alone in the testing stage. Each branch shares the same feature encoder but with different inputs. The contrastive learning branch uses both inputs to force the feature encoder to learn features invariant to augmentations. The classification branch uses the first input, while the reconstruction branch uses the second input. However, each

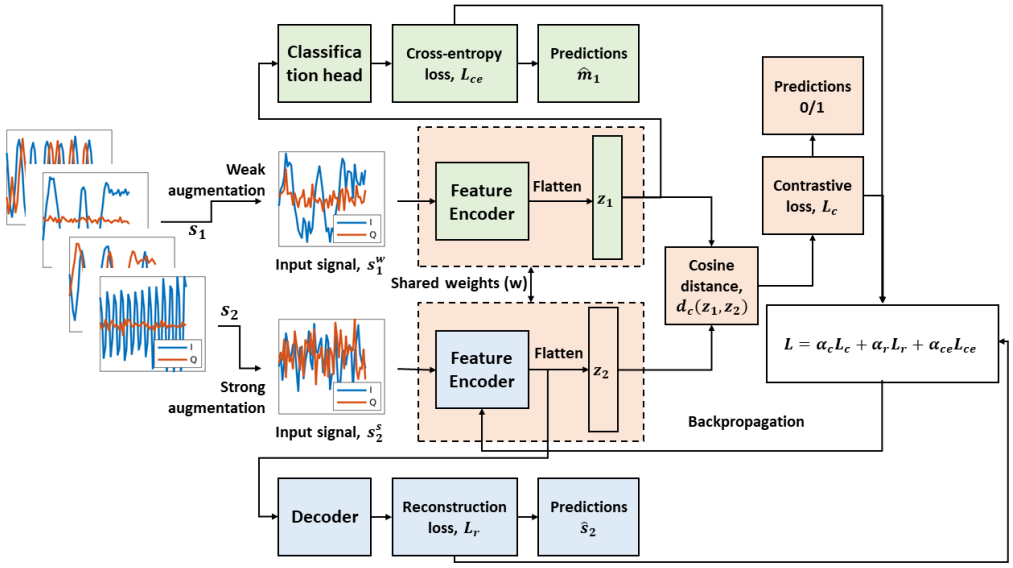

Figure 2: CLSR-AMC framework. The classification branch consists of blocks shaded in green. The reconstruction branch consists of blocks shaded in blue. The contrastive learning branch consists of blocks shaded in orange. All branches are used in the training stage, but only the classification branch is used in the testing stage.

branch contributes to the loss function as described below. The structure and hyperparameters optimization of each branch are given below.

3) Loss function: The weights of the feature encoder are updated over training epochs to minimize the loss function, which is given as a weighted sum of contrastive loss L_c, reconstruction loss L_r and cross-entropy loss L_{ce}.

$$L = \alpha_c L_c + \alpha_r L_r + \alpha_{ce} L_{ce}, \tag{3}$$

where α_c, α_r and α_{ce} are positive weight coefficients for contrastive loss, reconstruction loss and cross-entropy loss, respectively. The weight coefficient values range between 0 and 1 such that $\alpha_c + \alpha_r + \alpha_{ce} = 1$.

a) Cross-entropy loss: Categorical cross-entropy [41] measures the difference between two probability distributions. Softmax is utilized to convert the learned embeddings into the probability of the input signal belonging to each candidate modulation. When used as a loss function, the two underlying distributions are the predictions and the true classes of the samples. Categorical cross-entropy can be written as:

$$L_{ce} = -\frac{1}{N_B} \sum_{i=1}^{N_B} \sum_{j=1}^{M} y_{i,j} \cdot \log(\hat{y}_{i,j}), \tag{4}$$

where $y_{i,j}$ represents the ground truth, $\hat{y}_{i,j}$ is the prediction, M is the number of modulations, and N_B is the training batch size.

b) Reconstruction loss: This paper defines reconstruction loss as Mean-Squared Error (MSE) between the non-augmented instance and the consequently reconstructed instance by the decoder. It can be expressed as

$$L_r = \frac{1}{N_B} \sum_{i=1}^{N_B} (s_i - \hat{s}_i)^2, \tag{5}$$

where N_B is the training batch size.

c) Contrastive loss: Contrastive loss [23] runs over pairs of samples, unlike loss functions that sum over samples, such as cross-entropy loss. Mathematically, it is given below:

$$L_c = \frac{1}{2N_B} \sum_{i=1}^{N_B} (1 - y_i) \cdot (d_{c_i})^2 + (y_i) \cdot \{max(0, q - d_{c_i})\}^2, \tag{6}$$

where N_B is the training batch size, the value of y_i is the true label (0 for positive pairs, 1 for negative pairs), d_c is the distance measure between feature embeddings of the input samples, and $q > 0$ is a hyperparameter called margin, which controls whether the distance of negative pairs contributes to the loss [23]. As in [39], this paper adopts a cosine distance to measure the similarity between feature embeddings of input samples, s_i, s_j, and is given as follows:

$$d_c(z_i, z_j) = \frac{z_i^T z_j}{\|z_i\|^2 \|z_j\|^2}, \tag{7}$$

where $\|\cdot\|$ denotes the l_2 norm.

4) Optimization: The CLSR-AMC framework adopts Aggregated Residual Transformations for Deep Neural Networks (ResNeXt)-based architecture for the feature encoder. ResNeXt is CNN-based architecture that adds identity shortcuts to resolve the vanishing gradient problem in CNNs [42]. The ResNeXt-based methods outperform traditional CNN-based AMC, as shown in [10,13]. The decoder shares the same structure as the feature encoder, where *Pooling* layers are replaced with *Upsampling* layers, and *Convolutional* layers are replaced with *Convolutional transpose* layers. The classification head consists of a few *Dense* layers. The number of ResNeXt blocks in the feature encoder and decoder, and the number of *Dense* layers in the classification head are optimized by adopting a Genetic Algorithm (GA) inspired by [43]. The hyperparameters include the number of filters,

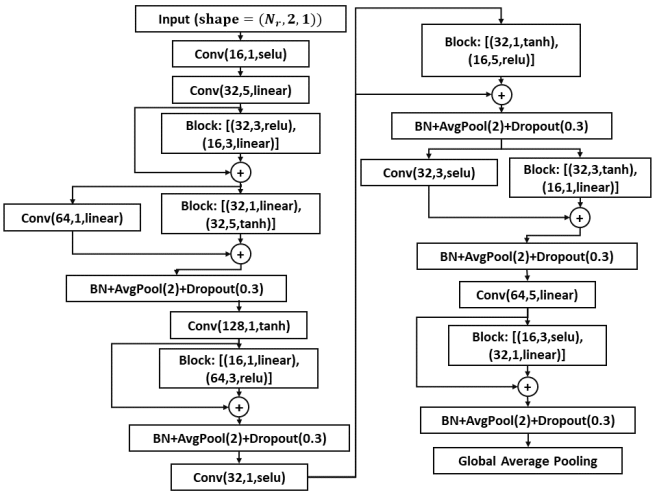

Figure 3: DNN architecture of the feature encoder. (BN denotes *Batch Normalization* layer. AvgPool denotes *Average Pooling* layer.)

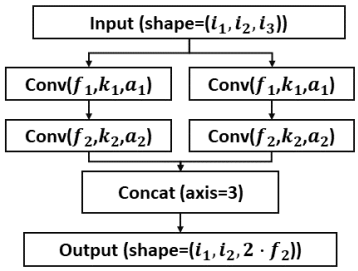

Figure 4: ResNeXt-based block structure with two parallel branches. Each branch is a serial fusion of two *Convolutional* layers

kernel size, activation type of *Convolutional* layers, number of hidden units and activation type of the *Dense* layers are also optimized by GA. Besides the DNN structure and its hyperparameters optimization, there are also additional hyperparameters of CLSR-AMC, including the margin value for the contrastive loss and the weight coefficients in the loss function (Eq. 3). The GA proposed in [43] is modified to support the optimization of the CLSR-AMC architecture (Fig. 2) and additional hyperparameters while keeping the core functions of GA the same as in [43]. The best-found architecture for the feature encoder is shown in Fig. 3. It consists of six blocks with the structure shown in Fig. 4. Each block has two parallel branches, each with two *Convolutional* layers with f filters, kernel size k and activation a. The classification head consists of two *Dense* layers with the number of dense units 71, 72 and activations *relu*, *linear*, respectively. GA found out that CLSR-AMC with $q = 0.79$, $\alpha_c = 0.75$, $\alpha_r = 0.11$ and $\alpha_{ce} = 0.14$ achieves the best performance.

IV. Experimental setup

1) Baselines: This work employs four baselines from the literature: (1) LSTM-based DRCN (LSTM-DRCN) [34], (2) supervised 1D-CNN classifier given in [13] (1D-CNN), (3) supervised ResNeXt classifier with Spatial Transformer Networks (STN) module given in [10] (ResNeXt-STN), and (4) Semi-supervised CLR framework for AMC given in [39] (Semi-CLR). LSTM-DRCN is a supervised reconstruction-based DA whose loss function is a weighted sum of reconstruction and cross-entropy losses. ResNeXt-STN includes an STN module that learns spatial transformations for data augmentation. Semi-CLR is a contrastive learning-based modulation classifier that treats independently contrastive loss and cross-entropy loss.

2) Datasets: Two modulations sets are used: (1) *a Basic set*, containing $N_{mod} = 11$ modulation formats typically used in the literature: BPSK, QPSK, 8-PSK, 16/64-QAM, PAM4, GFSK, CPFSK, BFM, DSB-AM and SSB-AM; and (2) *an Extended set*, containing the basic set and nine higher-order modulations: OQPSK, 32/128/256-QAM, 16/32/64/128/256-APSK ($N_{mod} = 20$). Both sets are synthetically generated in MATLAB. The training and validation datasets contain I/Q samples generated with an upsampling factor of 4, Raised Cosine (RC) filter with a roll-off factor of 0.35, and SNR of 18 dB under simple AWGN. The testing datasets have the same upsampling factor and RC filter configuration as the training dataset. The testing datasets contain data from multiple domains:

- AWGN with SNR ranging from -6 dB to 20 dB.
- Rayleigh channel with a path profile: delays of $[0, 4.5, 8.5]$ μs and gains of $[0, -1, -5]$ dB. AWGN with SNR in range $[-6, 20]$ dB is added to the Rayleigh channel. The maximum Doppler shift is set to 4 Hz.
- Rician channel with K factor of 4, a path profile with delays of $[0, 0.25, 3, 8]$ μs and gains of $[0, -2, -10, -3]$ dB. AWGN with SNR in range $[-6, 20]$ dB is added to the Rician channel. The maximum Doppler shift is set to 4 Hz.

For each combination of SNR and modulation type, 1000 testing instances are generated with a size of $N = 128$ and $N = 1024$ for the basic set and the extended set, respectively. The extended set requires longer signal observation because of the higher-order modulations [10].

3) Implementation details: Both CLSR-AMC and the chosen baselines are implemented in TensorFlow [44]. The training is performed over $N_{epochs} = 80$ epochs and a batch size of 256 on a GPU server with eight Nvidia RTX 2080Ti cards. Adam [45] with learning rate of 0.001 is adopted for weights optimization.

V. Performance evaluation

CLSR-AMC assumes a single source domain using AWGN channel model with $SNR = 18$ dB. A cross-channel scenario evaluates classification performance when the SNR value changes and/or the channel model changes from the source to the target domain. In this section, various simulation experiments are presented, illustrating the performance of the proposed framework for cross-channel scenarios. First, we examined the impact of weight coefficient values in the proposed loss function (Eq. 3). Second, we assessed whether the classification performance increases with multiple runs of data augmentation per instance. Third, we compare CLSR-AMC to baselines from the literature for both datasets.

A. Impact of loss function weights on performance

CLSR-AMC adopts the loss function as the weighted sum of contrastive α_c, reconstruction α_r, and cross-entropy α_{ce}

Table I: CLSR-AMC average accuracy (%) vs weight values in the loss function (Eq. 3).

Weights	Basic dataset		
	AWGN	Rayleigh	Rician
GA optimized	81.65	68.76	61.06
$\alpha_{ce} = 1, \alpha_r = \alpha_c = 0$	80.95	68.93	59.71
$\alpha_c = 1, \alpha_r = \alpha_{ce} = 0$	55.95	52.01	45.84
$\alpha_r = 1, \alpha_c = \alpha_{ce} = 0$	41.41	37.75	34.51
$\alpha_{ce} = 0, \alpha_c = \alpha_r = 0.5$	62.30	57.50	50.87

losses. The weight values are optimized by GA [43] and the resulting values are $\alpha_c = 0.75$, $\alpha_r = 0.11$ and $\alpha_{ce} = 0.14$. Note that these values are optimal for the GA best-found feature encoder DNN architecture (see Fig. 3), and they may differ for the other architectures. With each new run, GA may find another best DNN architecture and its optimal weight values. To justify gains due to the GA best-found weight values, we run four experiments with manually set values for edge cases: (1) $\alpha_{ce} = 1$, $\alpha_r = \alpha_c = 0$, (2) $\alpha_c = 1$, $\alpha_r = \alpha_{ce} = 0$, (3) $\alpha_r = 1$, $\alpha_c = \alpha_{ce} = 0$, and (4) $\alpha_{ce} = 0$, $\alpha_r = \alpha_c = 0.5$. In the last three experiments, CLSR-AMC is first trained with defined weight values. Afterward, the feature encoder is frozen, and the classification head is re-trained with the labeled dataset. All presented results are on the basic dataset. The accuracies are averaged for the entire SNR range of $[-6, 20]$ dB and summarized in Table I.

Table I shows that CLSR-AMC with GA-optimized weights achieves the best results. CLSR-AMC behaves as the supervised classifiers with the augmented dataset when the cross-entropy loss is only active. Contrastive loss learns better classification features than reconstruction loss by achieving 14.54%, 14.26%, and 11.33% higher accuracy in the AWGN, Rayleigh, and Rician channels, respectively. For a case when CLSR-AMC is trained jointly for contrastive and reconstruction losses, it achieves higher accuracy by 6.35%, 5.49%, and 5.03% in AWGN, Rayleigh, and Rician channels, respectively, than in the case when it is trained only with contrastive loss.

B. Impact of the amount of augmented data on performance

One run of data augmentation per instance outputs four augmented instances. We choose the four augmentations per instance to ensure that each instance in the source domain has its weak and strong augmentations for both operations rotation and CaD. To assess whether CLSR-AMC achieves better classification performance if the number of data augmentation runs per instance increases, we run three experiments with (1) 2, (2) 3, and (3) 4 runs of data augmentation per instance. The accuracies are averaged over the entire SNR range, $SNR = [-6, 20]$ dB, and the results are presented in Table II.

Table II shows that accuracy values differ by a maximum of 2% in each channel and for each considered number of data augmentation runs. One run of data augmentation per instance is enough if there are 1000 labeled instances for each modulation in the source domain. However, more data augmentation runs might be needed when only 2 labeled instances per modulation are available, as shown later.

Table II: CLSR-AMC average accuracy (%) vs no. of data augmentation runs.

No. of augmentation runs	Basic dataset		
	AWGN	Rayleigh	Rician
1	81.65	68.76	61.06
2	81.88	69.30	62.10
3	80.98	70.53	61.17
4	81.72	70.36	63.34

C. CLSR-AMC vs baselines for cross-channel scenarios

As CSLR-AMC includes both data augmentation and weighted loss function, it is necessary to identify performance gain due to the data augmentation and performance gain due to the weighted loss function. To assess the benefit of the proposed weighted loss function, we compare CLSR-AMC with Semi-CLR and supervised baselines (1D-CNN and ResNeXt-STN) for different amounts of available labeled data. On the other hand, to assess the benefit of data augmentation, we compare CLSR-AMC with supervised baselines (1D-CNN, LSTM-DRCN, ResNeXt-STN) when we equip them with data augmentation. The comparison is made for both the basic and extended datasets.

1) CLSR-AMC: Performance gain due to the loss function: Semi-CLR originally assumes that it has two labeled instances for each SNR/modulation pair and a large amount of class-balanced unlabeled data. In contrast, CLSR-AMC assumes that it has 1000 labeled instances for each modulation at SNR of 18 dB. First, we will evaluate the robustness of Semi-CLR to channel model changes. Second, we will evaluate the benefits of the proposed weight loss function in CLSR-AMC under a limited amount of labeled data. Hence, we run three experiments depending on which data are available to Semi-CLR for training: (1) 1000 labeled instances for each modulation at $SNR = 18$ dB in AWGN, (2) 1000 labeled instances for each modulation at $SNR = 18$ dB and 1000 unlabeled instances for each modulation/SNR pair over the entire SNR range, $SNR = [-6, 20]$ dB in AWGN, and (3) two labeled instances and 1000 unlabeled instances for each modulation/SNR pair over the entire SNR range, $SNR = [-6, 20]$ dB in AWGN. For the first two experiments, CLSR-AMC is trained with its original settings, while for the third experiment, CLSR-AMC is trained under the same settings as Semi-CLR but without the usage of unlabeled data. For each experiment, 1D-CNN and ResNeXt are trained with available labeled data plus augmented data as the output of one run of data augmentation per each instance in the labeled dataset. The classification accuracies are averaged for the entire SNR range and summarized in Table III.

CLSR-AMC significantly outperforms Semi-CLR for each experiment and both datasets. CLSR-AMC and supervised classifiers perform better when a large amount of labeled data is available just for one SNR value than in a case where are only two labeled instances per each SNR value. As data augmentation provides instances for other SNR values and channel models, they can generalize well for the entire SNR range in each channel. In contrast, one run of data augmentation is not enough when there are only two samples per SNR value. The supervised classifiers are overfitting, and

Table III: Average accuracy (%) for CLSR-AMC vs baselines in different cross-channel scenarios.

Exp.	Tested for	Basic dataset				Extended dataset			
		CLSR-AMC	Semi-CLR	1D-CNN	ResNeXt-STN	CLSR-AMC	Semi-CLR	1D-CNN	ResNeXt-STN
#1	AWGN	**81.65**	46.10	73.46	79.05	69.60	26.25	63.85	**72.26**
	Rayleigh	68.76	41.67	63.48	**69.94**	58.02	23.97	55.07	**63.09**
	Rician	61.06	36.25	59.06	**62.95**	44.83	17.50	**44.89**	42.66
#2	AWGN	**81.65**	42.94	73.46	79.05	69.60	30.26	63.85	**72.26**
	Rayleigh	68.76	41.56	63.48	**69.94**	58.02	27.89	55.07	**63.09**
	Rician	61.06	40.39	59.06	**62.95**	44.83	23.10	**44.89**	42.66
#3	AWGN	**70.54**	46.25	49.46	46.04	**55.84**	30.15	39.24	44.40
	Rayleigh	**60.88**	43.87	47.72	43.62	**47.31**	28.73	37.28	40.21
	Rician	**53.63**	37.66	44.74	44.57	**39.82**	22.17	33.37	32.03

for the basic dataset their performance significantly drops by $33.01/24\%$, $26.32/15.76\%$, and $18.38/14.32\%$ for ResNeXt-STN/1D-CNN in the AWGN, Rayleigh, and Rician channels, respectively. The performance deterioration is similar for the extended dataset for both ResNeXt-STN and 1D-CNN. On the contrary, CLSR-AMC can perform quite well even under such a limited amount of data. Its accuracy drops are much lower: $11.11/13.76\%$, $7.88/10.71\%$, and $7.43/5.01\%$ for the basic/extended dataset in the AWGN, Rayleigh, and Rician channels, respectively. Semi-CLR performs the worst for the first experiment, where only labeled data are available without unlabeled data. Compared to Semi-CLR for the basic dataset, CLSR-AMC has a higher average accuracy of 24.29%, 17.01%, and 15.97% in the AWGN, Rayleigh, and Rician channels, respectively. CLSR-AMC also achieves similar accuracy gains for the extended dataset. Table III shows that CLSR-AMC benefits from the proposed weighted loss function in a case with a small amount of labeled data.

2) CLSR-AMC: Performance gain due to the data augmentation: The comparison with supervised baselines (1D-CNN, LSTM-DRCN, and ResNeXt-STN) is suitable to assess the performance gain due to the data augmentation. Thus, we run three experiments: (1) supervised baselines are trained for AWGN with 1000 labeled instances for each modulation/SNR pair over the entire SNR range, $SNR = [-6, 20]$ dB, (2) supervised baselines are trained for AWGN with 1000 labeled instances for each modulation at $SNR = 18$ dB, and (3) supervised baselines are trained for AWGN with 1000 labeled instances for each modulation at $SNR = 18$ dB and leveraging the proposed data augmentation where four augmented instances are added per each labeled instance. CLSR-AMC is trained only for AWGN and $SNR = 18$ dB with its data augmentation. Such trained models are evaluated in the AWGN, Rayleigh, and Rician channels.

Fig. 5 and Fig. 6 show accuracy versus SNR in different channels for the basic and extended datasets, respectively. All baselines experience significant accuracy drops by approx. 30%, 40% and 60% in AWGN with the unknown SNR, Rayleigh, and Rician channels, respectively (see Figs. 5 and 6, middle column). The proposed data augmentation helps each baseline to boost accuracy up to approx. 40% for unknown channel conditions (see Figs. 5 and 6, right column). CLSR-AMC performs similarly in fading channels as the supervised baselines with the augmented dataset. CLSR-AMC achieves the accuracy gain due to contrastive loss in the AWGN channel for $SNR > 10$ dB. The accuracy gains are up to 7.8%, 13.8%, and 6.8% versus ResNeXt-STN, 1D-CNN, and LSTM-DRCN, respectively, for the basic dataset (see Fig. 5 right top). The accuracy gains are up to 4.8%, 13.0%, and 5% versus ResNeXt-STN, 1D-CNN, and LSTM-DRCN, respectively, for the extended dataset (see Fig. 6 right top). A gain due to reconstruction loss can be noticed for low SNR values in AWGN and is equal to 4%, 5%, and 10.2% versus ResNeXt-STN, 1D-CNN, and LSTM-DRCN, respectively, for the basic dataset. In the Rayleigh channel, LSTM-DRCN and CLSR-AMC achieve the best performance due to reconstruction loss in their loss functions. Fig. 5 and Fig. 6 show that reconstruction loss can deal with the Rayleigh fading but not with the Rician fading.

VI. Conclusions

Supervised DL-based modulation classifiers are very sensitive to changes in the transmitter and channel parameters as such variations cause large data distribution shifts. A few unsupervised DA methods have been adopted to combat distribution shifts, but under the unrealistic assumption of a large and complete class-balanced unlabeled dataset that covers each possible combination of transmitter and channel parameters. That assumption does not hold in the wild because the possible combinations of transmitter and channel parameters is unbounded. In this paper, we proposed CLSR-AMC, which assumes that only a large amount of labeled data is collected in the simplest channel conditions, AWGN with an SNR commensurate with that of a real-world indoors link (in our case 18 dB). CLSR-AMC applies simple data augmentation operations to emulate the impact of fading channels or in general data distribution changes to the signal constellations. We show that with four augmentations per labeled instance, the classification performance improves up to approx. 40% in unknown channel conditions. We also demonstrate that the proposed weighted sum of contrastive, reconstruction and cross-entropy losses provides better results than when loss functions are treated independently. The weighted loss function outperforms the fully supervised baselines in the high SNR, simple AWGN channel regime by 6.8% and 5%, when trained on the same augmented and labeled basic and extended datasets, respectively. This means all networks are trained on exactly the same data, but our proposed method learns the

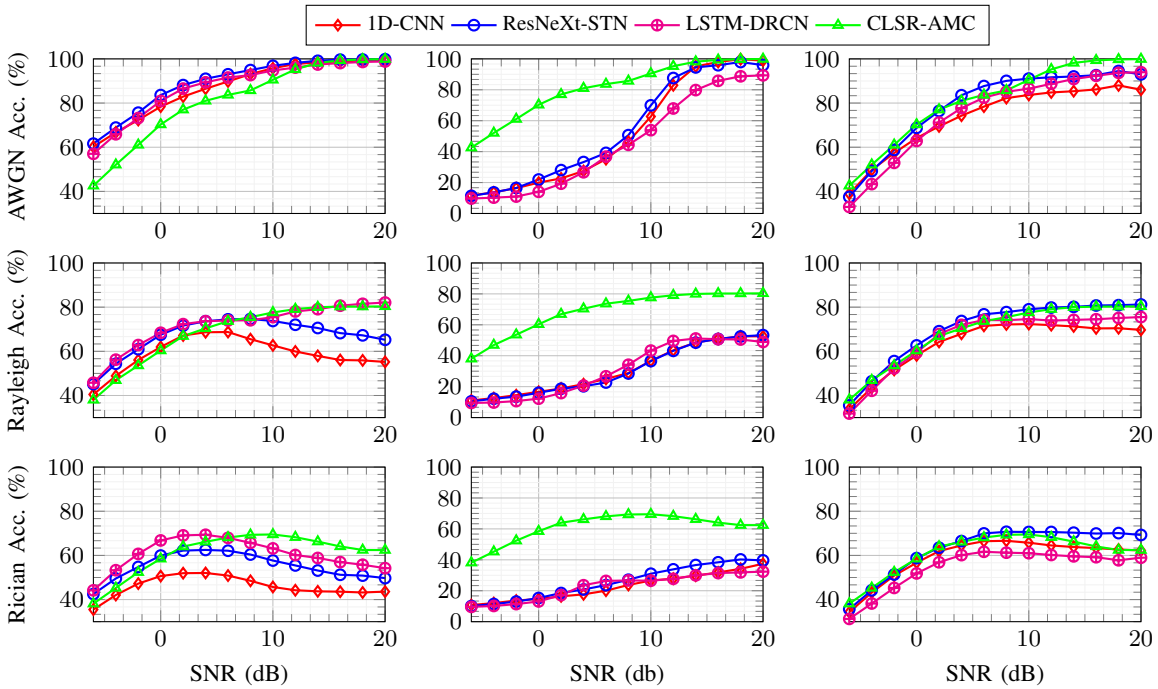

Figure 5: CLSR-AMC comparison with the supervised classifiers in different channels (AWGN top, Rayleigh middle, Rician bottom) and the basic dataset when classifiers are trained for: AWGN $SNR = [-6, 20]$ dB (left); AWGN $SNR = [18]$ dB (middle); AWGN $SNR = [18]$ dB including proposed data augmentation for all baselines (right). CLSR-AMC is trained for AWGN $SNR = [18]$ dB and includes data augmentation.

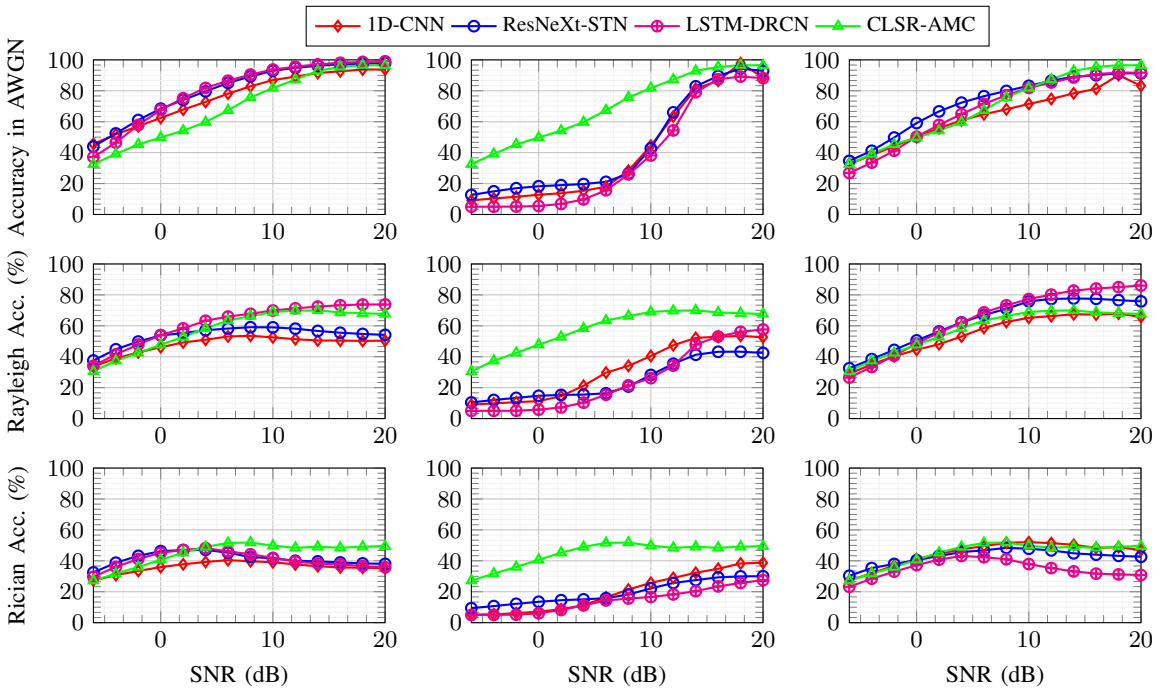

Figure 6: CLSR-AMC comparison with the supervised classifiers in different channels (AWGN top, Rayleigh middle, Rician bottom) and the extended dataset when classifiers are trained for: AWGN $SNR = [-6, 20]$ dB (left); AWGN $SNR = [18]$ dB (middle); AWGN $SNR = [18]$ dB including proposed data augmentation for all baselines (right). CLSR-AMC is trained for AWGN $SNR = [18]$ dB and includes data augmentation.

differences between domain variations and class variations better. Additionally, our results show that the reconstruction loss can combat the impact of the Rayleigh fading, but not the Rician fading. This paper covers only distribution shifts due to channel parameters changes. Future work should cover distribution shifts due to hardware impairments and transmitter parameters' changes by adding corresponding augmentation operations in CLSR-AMC to emulate their impacts to the signal constellations.

VII. Acknowledgment

This work has been partially supported by the Swiss Federal Office for Defence Procurement (armasuisse) (project code Aramis CYD-C-2020003) and NSF CNS-1845858.

REFERENCES

[1] F. Liu, Y. Cui, *et al.*, "Integrated Sensing and Communications: Toward Dual-Functional Wireless Networks for 6G and Beyond," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 6, pp. 1728–1767, 2022.

[2] A. Liu, Z. Huang, *et al.*, "A Survey on Fundamental Limits of Integrated Sensing and Communication," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 994–1034, 2022.

[3] Y.-C. Liang, K.-C. Chen, *et al.*, "Cognitive radio networking and communications: an overview," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 7, pp. 3386–3407, 2011.

[4] T. Yücek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *IEEE Communications Surveys & Tutorials*, vol. 11, pp. 116–130, 2009.

[5] J. L. Xu, W. Su, and M. Zhou, "Likelihood-Ratio Approaches to Automatic Modulation Classification," *IEEE Trans. Systems, Man, and Cybernetics Part C: Applications and Reviews*, vol. 41, pp. 455–469, July 2011.

[6] T. Huynh-The, Q.-V. Pham, *et al.*, "Automatic Modulation Classification: A Deep Architecture Survey," *IEEE Access*, vol. 9, pp. 142950–142971, 2021.

[7] S. Majhi, R. Gupta, *et al.*, "Hierarchical Hypothesis and Feature-Based Blind Modulation Classification for Linearly Modulated Signal," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 12, 2017.

[8] Y. Zhang, J. Wang, *et al.*, "Wireless Signal Classification Based on High-Order Cumulants and Machine Learning," in *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pp. 246–250, 2018.

[9] F. Yang, B. Hao, *et al.*, "A Method of High-Precision Signal Recognition Based on Higher-Order Cumulants and SVM," in *2018 5th International Conference on Systems and Informatics (ICSAI)*, pp. 455–459, 2018.

[10] E. Perenda, S. Rajendran, *et al.*, "Learning the unknown: Improving modulation classification performance in unseen scenarios," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10, 2021.

[11] B. Jdid, K. Hassan, *et al.*, "Machine Learning Based Automatic Modulation Recognition for Wireless Communications: A Comprehensive Survey," *IEEE Access*, vol. 9, pp. 57851–57873, 2021.

[12] S. Rajendran, W. Meert, *et al.*, "Deep Learning Models for Wireless Signal Classification with Distributed Low-Cost Spectrum Sensors," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 3, pp. 433–445, 2018.

[13] T. O'Shea, T. Roy, and T. C. Clancy, "Over the Air Deep Learning Based Radio Signal Classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, 2018.

[14] Y. Ren, W. Jiang, and Y. Liu, "Complex-valued Parallel Convolutional Recurrent Neural Networks for Automatic Modulation Classification," in *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 804–809, 2022.

[15] S. Chang, S. Huang, *et al.*, "Multitask-Learning-Based Deep Neural Network for Automatic Modulation Classification," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2192–2206, 2022.

[16] E. Meshkova, Z. Wang, *et al.*, "Designing a Self-Optimization System for Cognitive Wireless Home Networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 684–702, 2017.

[17] L. Cheng and B. A. Huberman, "Dynamic Bandwidth Allocation in Small-Cell Networks: An Economics Approach," in *2020 IEEE Radio and Wireless Symposium (RWS)*, pp. 225–227, 2020.

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

[19] L. Zhang and X. Gao, "Transfer Adaptation Learning: A Decade Survey," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–22, 2022.

[20] K. Bu, Y. He, *et al.*, "Adversarial Transfer Learning for Deep Learning Based Automatic Modulation Classification," *IEEE Signal Processing Letters*, vol. 27, pp. 880–884, 2020.

[21] L. Li, Q. Peng, *et al.*, "Deep Modulation Recognition in an Unknown Environment," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 1045–1048, 2019.

[22] Q. Wang, P. Du, *et al.*, "Adversarial unsupervised domain adaptation for cross scenario waveform recognition," *Signal Processing*, vol. 171, p. 107526, 2020.

[23] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, pp. 1735–1742, 2006.

[24] I. Lee and W. Lee, "UniQGAN: Unified Generative Adversarial Networks for Augmented Modulation Classification," *IEEE Communications Letters*, vol. 26, no. 2, pp. 355–358, 2022.

[25] B. Tang, Y. Tu, *et al.*, "Digital Signal Modulation Classification With Data Augmentation Using Generative Adversarial Nets in Cognitive Radio Networks," *IEEE Access*, vol. 6, pp. 15713–15722, 2018.

[26] M. Patel, X. Wang, and S. Mao, "Data augmentation with conditional GAN for automatic modulation classification," *Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning*, 2020.

[27] E. Rebeiz, F.-L. Yuan, *et al.*, "Energy-Efficient Processor for Blind Signal Classification in Cognitive Radio Networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 2, pp. 587–599, 2014.

[28] E. Kazikli, B. Dulek, and S. Gezici, "Optimal Joint Modulation Classification and Symbol Decoding," *IEEE Transactions on Wireless Communications*, vol. 18, no. 5, pp. 2623–2638, 2019.

[29] S. Hanna, C. Dick, and D. Cabric, "Signal Processing-Based Deep Learning for Blind Symbol Decoding and Modulation Classification," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 82–96, 2022.

[30] S. D'Oro, F. Restuccia, and T. Melodia, "Can You Fix My Neural Network? Real-Time Adaptive Waveform Synthesis for Resilient Wireless Signal Classification," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10, 2021.

[31] E. Tzeng, J. Hoffman, *et al.*, "Deep Domain Confusion: Maximizing for Domain Invariance," *CoRR*, vol. abs/1412.3474, 2014.

[32] M. Long, Y. Cao, *et al.*, "Transferable Representation Learning with Deep Adaptation Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 3071–3085, 2019.

[33] M. Ghifary, W. B. Kleijn, *et al.*, "Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation," in *Computer Vision – ECCV 2016*, (Cham), pp. 597–613, Springer International Publishing, 2016.

[34] Z. Ke and H. Vikalo, "Real-Time Radio Modulation Classification With An LSTM Auto-Encoder," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4935–4939, 2021.

[35] J. Qiao, W. Chen, *et al.*, "Blind Modulation Classification Under Uncertain Noise Conditions: A Multitask Learning Approach," *IEEE Communications Letters*, vol. 26, no. 5, pp. 1027–1031, 2022.

[36] D. Chicco, "Siamese Neural Networks: An Overview," *Methods in molecular biology*, vol. 2190, pp. 73–94, 2021.

[37] S. Huang, Y. Jiang, *et al.*, "Automatic Modulation Classification Using Contrastive Fully Convolutional Network," *IEEE Wireless Communications Letters*, vol. 8, no. 4, pp. 1044–1047, 2019.

[38] Z. L. Langford, L. Eisenbeiser, and M. Vondal, "Robust Signal Classification Using Siamese Networks," *Proc. ACM Workshop on Wireless Security and Machine Learning*, 2019.

[39] D. Liu, P. Wang, *et al.*, "Self-Contrastive Learning based Semi-Supervised Radio Modulation Classification," in *MILCOM 2021*, pp. 777–782, 2021.

[40] L. Huang, W. Pan, Y. Zhang, L. Qian, N. Gao, and Y. Wu, "Data Augmentation for Deep Learning-Based Radio Modulation Classification," *IEEE Access*, vol. 8, pp. 1498–1506, 2020.

[41] S. Huang and T. Le, *Principles and Labs for Deep Learning*. Elsevier Science, 2021.

[42] S. Xie, R. Girshick, *et al.*, "Aggregated Residual Transformations for Deep Neural Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5987–5995, 2017.

[43] E. Perenda, S. Rajendran, *et al.*, "Evolutionary Optimization of Residual Neural Network Architectures for Modulation Classification," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 542–556, 2022.

[44] M. Abadi, A. Agarwal, *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[45] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *San Diego: The International Conference on Learning Representations (ICLR)*, vol. 1, no. 1, 2015.