Received 21 June 2024; accepted 25 July 2024. Date of publication 5 August 2024; date of current version 9 August 2024.

Digital Object Identifier 10.1109/OJCOMS.2024.3436601

Spectrum Sensing With Deep Clustering: Label-Free Radio Access Technology Recognition

LJUPCHO MILOSHESKI [1,2] (Graduate Student Member, IEEE), MIHAEL MOHORČIČ [1,2] (Senior Member, IEEE), AND CAROLINA FORTUNA [1]

[1]Department of Communication Systems, Jožef Stefan Institute, 1000 Ljubljana, Slovenia
[2]Information and Communication Technologies, Jožef Stefan International Postgraduate School, 1000 Ljubljana, Slovenia

CORRESPONDING AUTHOR: L. MILOSHESKI (e-mail: ljupcho.milosheski@ijs.si)

This work was supported in part by the Slovenian Research and Innovation Agency under Grant P2-0016, and in part by the European Union's Horizon Europe Framework Programme under Grant 101096456 (NANCY). The NANCY Project was supported by the Smart Networks and Services Joint Undertaking and its members.

ABSTRACT The growth of the number of connected devices and network densification is driving an increasing demand for radio network resources, particularly Radio Frequency (RF) spectrum. Given the dynamic and complex nature of contemporary wireless environments, characterized by a wide variety of devices and multiple RATs, spectrum sensing is envisioned to become a building component of future 6G, including as a components within O-RAN or digital twins. However, the current SotA research for RAT classification predominantly revolves around supervised Convolutional Neural Network (CNN)-based approach that require extensive labeled dataset. Due to this, it is unclear how existing models behave in environments for which training data is unavailable thus leaving open questions regarding their generalization capabilities. In this paper, we propose a new spectrum sensing workflow in which the model training does not require any prior knowledge of the RATs transmitting in that area (i.e., no labelled data) and the class assignment can be easily done through manual mapping. Furthermore, we adaptat a SSL deep clustering architecture capable of autonomously extracting spectrum features from raw 1D Fast Fourier Transform (FFT) data. We evaluate the proposed architecture on three real-world datasets from three European cities, in the 868 MHz, 2.4 GHz and 5.9 GHz bands containing over 10 RATs and show that the developed model achieves superior performance by up to 35 percentage points with 22% fewer trainable parameters and 50% less floating-point operations per second (FLOPS) compared to an SotA AE-based reference architecture.

INDEX TERMS Analysis, clustering, machine learning, monitoring, self-supervised, radio frequency spectrum.

I. INTRODUCTION

THE EXPONENTIAL growth of the number of connected devices [1] and network densification is driving an increasing demand for radio network resources, particularly Radio Frequency (RF) spectrum. The allocation of new operational frequency bands, such as the 6.425-7.125 GHz range designated by the ITU for licensed mobile communications [2], only partially alleviates the growing demand for RF spectrum resources. Thus, it is crucial to enhance the utilization of the occupied bands through innovative spectrum-sharing strategies that go beyond the existing licensed [3] and license-exempt access schemes [4], meeting the complexity of the fast-evolving wireless networks. Such approaches will increasingly rely on accurate monitoring and understanding of the RF spectrum environment, achieved through spectrum sensing techniques. The process of spectrum sensing involves analyzing radio data for a variety of tasks [5], each playing an important role in the broader context of radio awareness, impacting the efficiency and effectiveness of wireless radio networks. Such tasks and corresponding purposes include Radio Access Technology (RAT) classification [6] for optimizing resources allocation and spectrum utilization, modulation classification [7] for improving data throughput and reducing error rates, anomaly

detection [8] enhancing network security and resilience, interference recognition [9] for identification of sources of interference, and Specific Emitter Identification (SEI) [10] ensuring secure and efficient spectrum sharing by preventing unauthorized access.

Given the dynamic and complex nature of contemporary wireless environments, characterized by a wide variety of devices and multiple RATs, spectrum sensing is envisioned to become a building component of future 6G networks [11]. As such, it will support the progress of Integrated Sensing and Communication (ISAC)-related systems and technologies [12], [13], facilitating their development and integration into the next generation of wireless communication systems. Consequently, spectrum sensing is anticipated to be a key component in the evolution of emerging wireless communication frameworks such as Open Radio Access Network (O-RAN) and Digital Spectrum Twin (DST) [14].

In the emerging O-RAN in which a large proportion of the radio functions are realized in software [15], [16], RAT monitoring plays an important role in optimizing radio and network parameters and spectrum utilization, largely through the employment of AI algorithms encapsulated as xApps and rApps [17]. By continuously assessing the usage and availability of different RATs, O-RAN can utilize intelligent applications to dynamically fine tune network settings and allocate spectrum resources or manage beam forming [18] in real time. Thus, it can enhance the overall efficiency and performance of the network and accommodate growing traffic demands, while better preserving privacy and being less data intensive [19] compared to SEI.

DSTs are digital representations of the real-world characteristics of the RF spectrum in a region based on historical data and measurement updates [20]. They are envisioned as tools to inform, forecast, and enhance spectrum utilization and interference management in wireless networks in a given geographical area. Spectrum sensing is a building component of DST, as outlined in [14]. It provides the measurement input necessary to refine the DST, which is initially developed based on empirical, theoretical, and/or ray-tracing models. Incorporating RAT monitoring into DST could further enhance its capabilities by adding detailed data on the various transmission technologies that are present in the area, such as RAT-specific activity patterns for a given time interval within the operational landscape. This integration could significantly improve the DST's awareness, accuracy and functionality, facilitating more effective spectrum management and optimization strategies.

The current SotA research for RAT classification predominantly revolves around supervised Convolutional Neural Network (CNN)-based approaches [21], [22], [23], [24], which have shown promising results, particularly in diverse signal-to-noise ratio (SNR) settings and various environmental conditions [25]. However, these methods have a number of shortcomings as follows. First, they heavily depend on extensive labeled datasets acquired in controlled settings where the transmission parameters need to be recorded in addition to the received signals requiring a more complex set-up for collection compared to unlabelled data. Second, it is unclear how they transfer to other environments characterized by (i) additional RATs compared to the original training data and (ii) different physical obstructions and noise variations. Third, the majority of the models have been developed and evaluated on a single dataset leaving open questions regarding their *generalization* capabilities.

Self-Supervised Learning (SSL) approaches, extensively explored in other application domains [26], are able to address the first two challenges mentioned above, because they do not require labels for training. As long as a spectrum sensor collecting data is available, that data can be directly used to train such models without requiring recorded transmission parameters. By analogy, this approach should be able to adapt the model in regular retraining periods to new types of RATs that may appear in the area. However, to be able to realize the classification functionality similar to their supervised counterparts, a manual cluster-to-label assignment step is required. Nevertheless, this process is rather efficient as the learnt clusters, together with additional insights, can be presented to a human decision maker for label assignment. The decision maker is also presented with a new cluster including a new technology that emerged from the learning process and can immediately identify it. The adoption of Self-Supervised Learning (SSL) in wireless communications research remains limited. Existing state-of-the-art (SotA) works primarily rely on variations of Autoencoder (AE) architectures [23], [27]. These methods emphasize instance learning, where the CNN part is trained solely or partially on reconstruction loss, focusing on distinguishing individual samples.

Recognizing the need for a more adaptable solution, particularly in scenarios where class information is unknown, we propose[1] the use of a domain-adapted DeepCluster [28] architecture. This approach diverges from the existing methods by basing the learning process of the CNN part on features shared among groups of samples rather than individual instances. This distinction could make the proposed architecture more general and effective in realistic environments, thus addressing the third challenge.

The main contributions of our work are:

- *New spectrum sensing workflow:* We propose a new spectrum sensing workflow in which the model training does not require any prior knowledge of the RATs transmitting in that area (i.e., no labelled data) and the class assignment can be easily done through manual mapping.
- *Development of an SSL Deep Clustering (DC) Architecture:* We introduce an adaptation of SSL deep clustering architecture capable of autonomously extracting spectrum features from raw 1D Fast Fourier Transform (FFT) data. Notably, this architecture is

[1] https://github.com/sensorlab/spire

inherently environment-agnostic and adaptable to varying numbers of operating transmission technologies.

- *Generic Unsupervised Model:* We go beyond the SotA in model development for RAT feature learning and subsequent classification by developing a more *generic* model on three real-world datasets from three European cities, in the 868 MHz, 2.4 GHz and 5.9 GHz bands containing over 10 RATs.
- *Efficiency and Performance Optimization:* The developed model boasts 22% fewer trainable parameters and requires 50% less floating-point operations per second (FLOPS) compared to an identical AE-based reference architecture. Besides the efficiency gains, our model achieves superior performance in label-based clustering evaluations by up to 35 percentage points (ppt) on TCD-L dataset (data of each technology acquired in six different sites), up to 4 ppt on ITS-L dataset (samples of each technology acquired at single site) and better separation of the feature space of the unlabeled UNB-U dataset (continuously sensed data in uncontrolled environment).

The structure of this paper is as follows: Section II reviews related work in the field. Section III describes the reference scenario and related assumption. The problem formulation is detailed in Section IV. The proposed and baseline architectures are presented in Section V. Section VI outlines the evaluation methodology, while Section VII discusses the outcomes of this evaluation. The paper concludes with Section VIII, summarizing the key findings and contributions.

II. RELATED WORK

Considering the disadvantages of the supervised learning approaches for spectrum sensing-related tasks, such as the necessity of labels and expert intervention in the setup of models in specific environments, research work towards employing unsupervised models has attempted to advance alternative approaches that do not necessitate labels.

The performance of pure unsupervised clustering models applied to spectrum data is studied in [29]. Authors evaluate classical unsupervised (clustering) approaches on data transformed by t-distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP). Three clustering algorithms are subjected to comparison: K-means, Agglomerative Hierarchical Clustering, and Hierarchical Density-Based Spatial Clustering. Although such an approach benefits from being general and simple for high-level spectrum analysis, feature extraction is based on t-SNE and UMAP, which is disadvantageous compared to CNN-based, deep feature learning.

A study of another lightweight and general approach for RAT classification is proposed in [30] as an alternative to models based on CNN. It introduces Dynamic Mode Decomposition (DMD) as a feature extraction technique capable of distinguishing RATs based on their bandwidth. The classification process involves a straightforward threshold method, demonstrating superior performance compared to a baseline CNN-based solution that operates on Gramian Angular Summation Field (GASF) images derived from time series data. Although the solution benefits from its simplicity, it faces the problem of manual adjustment of thresholds.

AEs are neural networks trained to encode input data into a compressed representation and then decode that representation back to an approximation of the original input. They are primarily used for dimensionality reduction or feature learning. However, in their original form, where learning is based on reconstruction loss only, they do not provide clustering-friendly embedded space. This led to research on modifications of this architecture by adding different loss functions, as discussed in [31]. AE with an additional clustering loss ([32], [33], [34]) in the embedded space is an SSL model designed for both dimensionality reduction and clustering tasks. Such a model combines the feature-learning capabilities of AEs with the grouping intuition of clustering algorithms by optimizing for a representation that serves both purposes. In general, the clustering loss in the embedded space could be calculated based on distances between samples or based on their distribution.

In [33], a deep learning, AE-based solution is employed for unsupervised feature learning and clustering of embedded features on radio data for the task of modulation recognition. Alongside the AE-specific reconstruction loss, the authors introduce a "Deep Clustering" loss in the embedded space. The total loss function aims to simultaneously improve reconstruction and clustering in the embedded space. The authors provide a detailed evaluation of their proposed model, considering that the number of clusters is the same as the number of different modulations, which is an optimistic setup because it requires knowledge of the exact number of modulations that exist in the data. In a real-world RAT monitoring scenario, it is much more realistic to assume an unknown number of classes/RATs (modulations in their case).

Deep Convolutional Embedded Clustering (DCEC) is another modification of AE for deep feature learning, initially dedicated to image processing proposed in [34]. In this work, the total loss function is a combination of the typical reconstruction loss and Kullback-Leibler divergence loss in the embedded space. While similar learning CNN modules could be used in both approaches, the different loss calculations may optimize the convolution filters to focus on different features in the input data.

Our previous work [35] involves a high complexity self-supervised model as it operates on waterfall plots/spectrograms (2D image-like matrices) employing off-the-shelf ResNet18 model, initially developed for Red-Green-Blue (RGB) images. This approach provides valuable information about the time-frequency occupancy of the spectrum with different RATs that operate in the observed band. Although a substantial reduction of complexity of the feature vectors is achieved by augmenting the feature

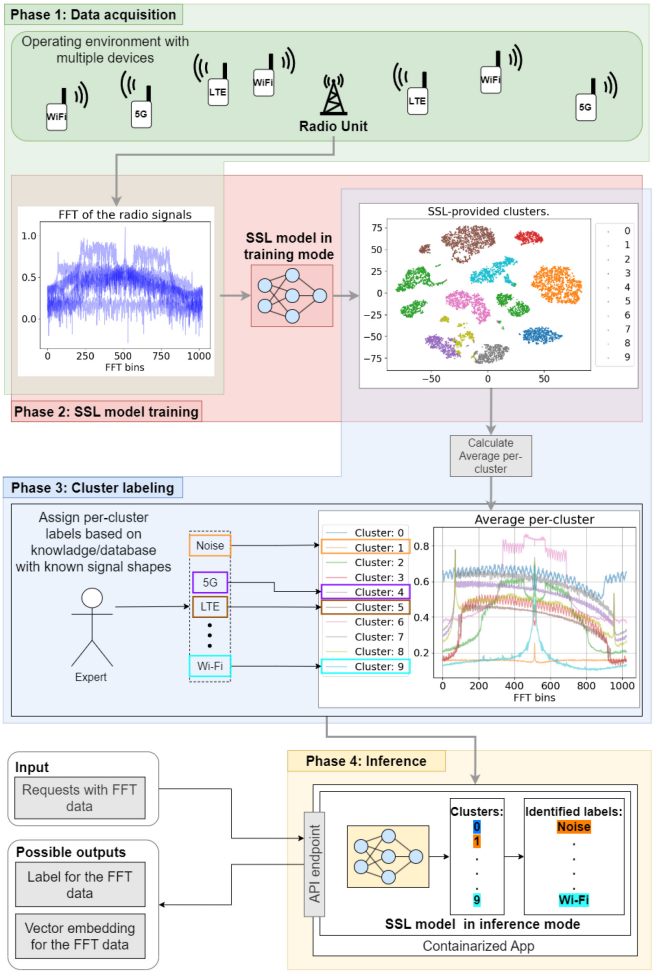


FIGURE 1. SSL enabled spectrum sensing workflow.

processing, the model itself keeps the same (high) complexity as its original implementation for feature learning of RGB images. While this could be advantageous, considering it is off-the-shelf, proven architecture, the high complexity could prevent application in the lower control layers of the O-RAN, for example, as xApp in the Near-real-time (Near-RT) Radio Intelligent Controller (RIC) due to the short control loop times.

In this manuscript, we propose a 1D-CNN-based adaptation of the DeepCluster architecture for feature learning and unsupervised classification of RATs by processing 1D FFT data instead of 2D spectrograms. While the loss of time-frequency dependency is a drawback, this issue can be mitigated through post-processing of the classified 1D swipes.

III. REFERENCE SCENARIO AND PROPOSED SPECTRUM SENSING WORKFLOW

In this paper, we assume a realistic scenario in which terminals, equipped with various RATs as depicted at the top of Figure 1, operate in a given area, possibly also in the same frequency band. For instance, a spectrum sensor may operate in ITS 5.9 GHz band where LTE, Wi-Fi, 5G NR, C-V2X PC5, and ITS-G5 technologies may co-exist [24]. Other examples may refer to monitoring the U.S. 6 GHz band, where Wi-Fi 6E and 5G New Radio Unlicensed (NR-U) share the unlicenseds spectrum, or the LTE/5G and military radar that are sharing the 150 MHz CBRS band at 3.5 GHz [36].

Furthermore, we work under the assumption that we have no prior information on what kind of RATs exist in the area under observation. Without having any prior knowledge on the transmitters in the sensed spectrum, we aim to identify them and recognize them in the future. In order to reach our aim, we propose a four-step workflow as depicted in Figure 1 and elaborated below.

A. PHASE 1: DATA ACQUISITION

The Phase 1, marked with the green area in Figure 1, represents the data acquisition process. In this phase, the Radio Unit collects data from the operating environment in the form of the FFT transformation of the captured radio signals.

B. PHASE 2: SSL MODEL TRAINING

The model training, marked with red in Figure 1, uses the data collected in Phase 1 and learns to group similar FFT shapes together. As a result of the grouping, clusters of similar shapes emerge with each cluster ideally containing samples from a single RAT. As the output of this phase, we obtain the trained model for feature extraction from FFT data and formed clusters.

C. PHASE 3: CLUSTER LABELLING

In this phase, a human expert is able to recognize the existing RATs by looking at the clusters and manually assign a label, i.e., specific RAT such as LTE, WiFi, etc., to each cluster. In this step, hours, days or weeks of spectrum sensing are summarized for the expert end-user to quickly glance at and understand what is happening in the monitored environment. The summary can be presented as an average per cluster sweep such as depicted in Figure 1 or as lower dimensional projections such as T-SNE.

D. PHASE 4: INFERENCE

The model trained on real-world samples in Phase 2 together with the per cluster labels provided in Phase 3 can be packaged as a classification service where the human-assigned labels represent the class of the sample assigned to a cluster. Therefore, the classifier can be encapsulated as a containerized application with an API endpoint, and deployed as a containerized application in a production level system where incoming sweeps would get labels assigned. Furthermore, the trained model could also be exposed as an embedding model, where a compact and low-dimensional representation of the input data is provided to the requester as illustrated in Phase 4 of Figure 1.

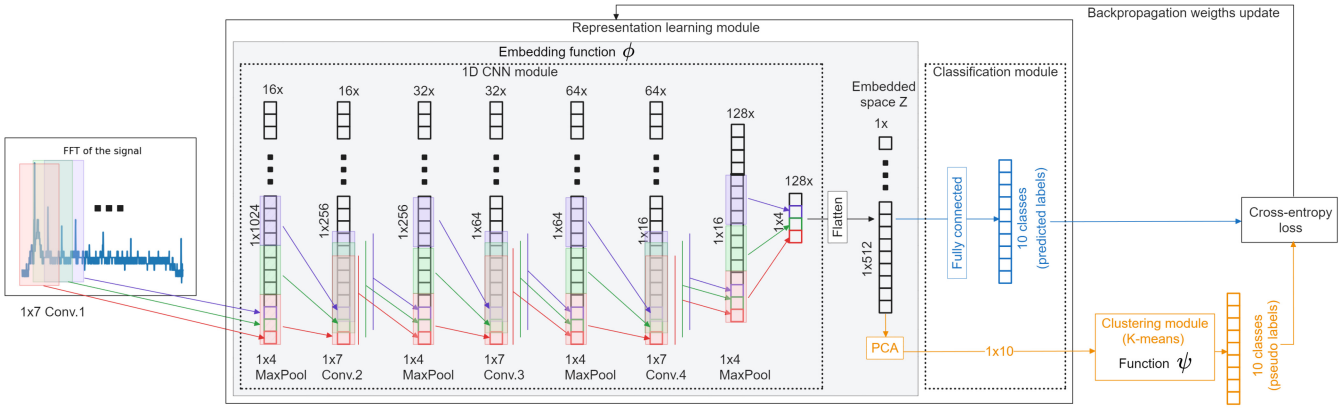


FIGURE 2. Proposed SSDC architecture.

IV. PROBLEM STATEMENT

In this paper, we formulate the spectrum sensing problem as a clustering problem in machine learning that partitions the set of electromagnetic energy measurements collected during individual sweeps into groups (i.e., clusters) that contain sweeps with similar shapes (i.e., energy level envelope). These are later identified and labeled by an expert. This corresponds to Phases 1-3 described in Section III and depicted in Figure 1. We represent a set of electromagnetic energy measurements corresponding to a sweep as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ and the set of identified RATs as $\mathbf{L} = \{L_1, L_2, \ldots, L_k\}$. To account for realistic environments where the number of types of transmissions is not known apriori, the number of clusters k is assumed to be unknown. This assumption uniquely distinguishes this work from the SotA where the number of classes or clusters are assumed apriori. The objective is to learn a composite function $\Phi : X \rightarrow Y$ that maps raw data points to their identified classes in the clustered space.

$$L = \Phi(X) \tag{1}$$

This function can be realized by decomposition into three sub-functions, the embedding function ϕ, the clustering function ψ, and the mapping function $\mathcal{M}$ and can be expressed as:

$$L = \mathcal{M}(\psi(\phi(X))) \tag{2}$$

A. 1. EMBEDDING FUNCTION

Denoting the ϕ as the embedding function that transforms raw data $\mathbf{X}$ into an embedded space and $Z = \{z_1, z_2, \ldots, z_n\}$ as the set of embedded representations of the data points, formally it is defined as:

$$\phi : X \rightarrow \mathbb{R}^d \tag{3}$$

where $\phi(x_i) = z_i$ and d is the dimensionality of the embedded space.

B. 2. CLUSTERING FUNCTION

Considering $C = \{C_1, C_2, \ldots, C_K\}$ as the set of clusters and K as the number of clusters where $K \leq n$, the clustering function ψ that assigns each embedded data point from $\mathbf{Z}$ to a cluster can be defined as:

$$\psi : \mathbb{R}^d \rightarrow \{1, 2, \ldots, K\} \tag{4}$$

where $\psi(z_i) = k$ indicates that the embedded point z_i is assigned to cluster C_k.

C. 3. MAPPING FUNCTION

Considering $L = \{l_1, l_2, \ldots, l_K\}$ as the set of expert identified labels, the mapping function denoted as $\mathcal{M}$, can be defined as:

$$\mathcal{M} : \{1, 2, \ldots, K\} \rightarrow L \tag{5}$$

where $\mathcal{M}(k) = l_j$ means that cluster C_k is mapped to the label l_j.

D. CONSTRAINTS

The constraint lies in the unknown number of clusters, $\mathbf{K}$, for which the clustering function ψ should be modeled. This is due to the uncontrolled nature of the monitored environment, where there is no prior information about the number of operating RATs. This scenario is general and mirrors a real-world deployment where, regardless of the operating band, new types of signals could appear during the operation of the system, either coming from unauthorized/unknown RATs in the licensed bands or from unknown RATs in the unlicensed bands.

V. PROPOSED SSL ARCHITECTURE

To solve the problem formalized in Section IV, we propose a SSL deep clustering architecture depicted in Figure 2. The architecture iteratively combines deep learning with clustering to learn meaningful data representations without labeled samples. The embedding function from Eq. (3) is realized by the CNN module, part of the representation learning module, while the clustering function from Eq. (4) is realized by the clustering module.

A. DESCRIPTION OF ARCHITECTURAL MODULES

1) CLUSTERING MODULE

The clustering module groups similar data points in the embedded space. To realize the clustering module, which in turn realizes the clustering function ψ, we chose the well-known K-means algorithm with the Euclidean distance metric. K-means has been shown to perform very well with embedded spaces [37], is easy to understand, and has low computational complexity. The Euclidean metric used as a loss function for the clustering is also computationally efficient, making it appropriate for large datasets, and it is also effective for lower-dimensionality embedded spaces [38]. It measures the straight-line distance between data points, which allows for a clear interpretation of how data points are assigned to clusters based on their proximity in the feature space.

2) REPRESENTATION LEARNING MODULE

The role of the representation learning module is to learn the mapping from the input raw data to the pseudo-labels generated by the clustering process. The CNN part of the module, together with the flattening and PCA transformation, realizes the embedding function ϕ, which transforms each input data point into a low-dimensional representation in the embedded space.

The CNN module of the SSDC architecture consists of four 1D convolutional layers, each followed by batch normalization, max-pooling, and *ReLU* activation layers, visualized in Figure 2. Batch normalization typically improves convergence speed [39], max-pooling reduces the dimensionality of the data through the layers while focusing on the most prominent features, and ReLU is important for catching the nonlinear dependencies in the data. Regarding the number and size of CNN filters, we followed the design principles discussed in [40] and [41], consisting of a stack of convolution layers followed by fully connected layers. Each consequent convolutional layer consists of double the number of filters of the previous layer and a vector size that is four times smaller than that of its predecessor. We increase the filter size to 7 (compared to 3 in [40]) so its size is large enough to neglect noise influence but small enough to be sensitive to the changes induced by the RAT-specific content in the FFT amplitudes.

Such an architecture totals 128, 406 parameters and 0.2 *GFLOPS*, which is significantly lower (up to approx. 100 times, i.e., two orders of magnitude) compared to the 11.7 million parameters and (9 times) 1.81 *GFLOPS* of a model dedicated for spectrograms processing in [35].

B. WORKFLOW OF THE SSDC ARCHITECTURE.

The SSDC follows an iterative workflow consisting of two branches executed successively, mutually optimizing the clustering module and the weights of the representation learning module. The working of the clustering branch (marked with orange color in Figure 2) consists of:

1) The 1D CNN module, which is part of the embedding function ϕ, and is set to evaluation mode, having randomly initialized weights. This module extracts features from the input data, processing the raw FFT through the convolutional layers and providing embedded 512×1 vector representations of the samples at its output. The embedded representation is further transformed to an even lower dimensionality of 1×10 using PCA transformation [42]. The goal is to preserve only the high-variety features of the embedded space before passing it to the clustering module. The CNN module, the flattening layer, and the PCA realize the embedding function defined in Eq. (3).
2) The feature vectors serve as an input to the clustering module, realizing the clustering function ψ in our architecture. The K-means assigns pseudo-labels to the processed samples according to:

$$g(\mathbf{z}_i) = \arg\min_j \|\mathbf{z}_i - \mu_j\|, \tag{6}$$

where g represents the cluster assignment function, $\mathbf{z}_i$ stands for the feature vector, and μ_j is the centroid of cluster C_j. The centroid μ_j is defined as:

$$\mu_j = \frac{1}{|C_j|} \sum_{\mathbf{z}_i \in C_j} \mathbf{z}_i. \tag{7}$$

The K-means clustering forms clusters by minimizing the within-cluster sum of squares, which is given by:

$$\min_{\mathbf{C},k} \sum_{j=1}^{k} \sum_{\mathbf{z}_i \in C_j} \|\mathbf{z}_i - \mu_j\|^2. \tag{8}$$

The output of the K-means algorithm is the partitioned set of feature vectors $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$ into k clusters. Ideally, each cluster $C_j \subseteq \mathbf{Z}$ would represent a single RAT. The provisioning of the pseudo-labels ends the clustering module optimization as part of a single iteration.

The representation learning module is set to the train mode in the second branch of the iteration, with the corresponding flow in Figure 2 marked with blue. Considering the pseudo-labels provided by the clustering branch described above, the weights of the representation learning module (including the CNN and the FC part) are trained in a classical supervised learning procedure.

1) The representation learning [42] module maps each of the input raw FFT samples to a single class at the output of the classifier, minimizing the difference between the pseudo-labels and the predicted labels using the Cross-entropy loss function defined as:

$$\mathcal{L}_{\mathrm{CE}} = -\sum_{i=1}^{n} \sum_{k=1}^{K} \mathbf{1}_{[g(\mathbf{z}_i)=k]} \log p_{i,k}. \tag{9}$$

2) Parameters θ are updated using gradient descent and backpropagation according to

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\mathrm{CE}}, \tag{10}$$

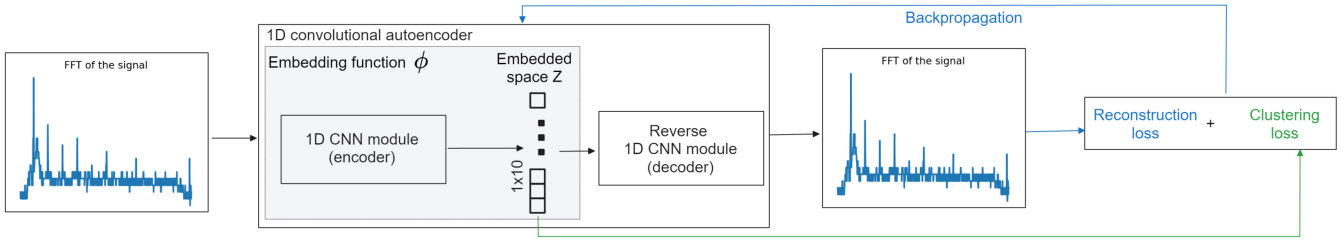


FIGURE 3. SotA autoencoder-based architecture [33].

where η is the learning rate, and $\nabla_\theta \mathcal{L}_{\text{CE}}$ represents the gradient of the loss function with respect to the network parameters θ.

The two separate procedures for optimizing the clustering and the representation learning module complete one full learning iteration. In summary, the clustering algorithm groups input data based on their distances in the embedded domain, providing pseudo-labels. The CNN and classification modules learn this distribution by enhancing the extracted features and predicting the pseudo-labels with each iteration. This iterative process between the CNN-based classification and clustering modules continues for a predefined number of training epochs, starting with randomly initialized CNN weights.

C. REFERENCE AUTOENCODER-BASED ARCHITECTURES

We compare our proposed SSDC architecture with two variants of Autoencoder (AE)-based SSL architecture, depicted in Figure 3. The AE-based architecture is composed of three main modules: encoder, embedding layer, and decoder. Regarding the problem formulation in Section IV, in this architecture, the encoder module represents the embedding function, and the clustering function works on the embedded space representations. Input for the encoder module is the raw data, which is processed by the sequential convolutional layer filters and converted into low-dimensional representation. This low-dimensional representation is flattened and transformed into a 1D vector representation in the embedding layer, which is in the middle of the architecture. The decoder block, which follows the embedding layer, has the same number of layers as the encoder and works in the opposite direction, increasing the dimensionality of the feature vector to the original size of the raw data. Providing the same dimensionality of the data at the output enables reconstruction loss in the learning process. In our work, we purposely use the same CNN module that was used in the SSDC architecture for building the AE-based architectures. This provides equal ground for comparison that will highlight the performance differences which arise from the architecture itself, and not from the variation in the trainable parameters in the representation learning module. However, the AE-based architectures, which are used as baselines, have different loss functions in the embedded space:

- Autoencoder with Modified Loss (AEML) has a custom loss function in the embedded domain based on the relative distances between the samples, as proposed in [33].
- Deep Convolutional Embedded Clustering (DCEC) utilizes the Kullback-Leibler divergence loss in the embedded domain concerning the distribution of the samples instead of their relative distances. DCEC was designed for image processing in [34].

1) WORKFLOW OF THE AE-BASED ARCHITECTURES

The training workflow of the reference AE-based models, AEML and DCEC, consists of two phases, pretraining and joint training.

1) Pre-training: The pretraining phase is the same as for the regular autoencoder. The encoder part takes high-dimensional input data and compresses it into a lower-dimensional embedded space (latent space), and the decoder part aims to reconstruct the original input data from the compressed form obtained by the encoder without considering the clustering loss. The goal is to produce a reconstruction as close as possible to the original input, thereby ensuring that the embedded space captures the essential features of the data.

During the pre-training phase, the autoencoder is trained to minimize the reconstruction error without considering the clustering loss. Thus, the weights update is performed using the reconstruction loss, which is the mean squared error (MSE) between the input x and its reconstruction $\hat{x}$, expressed as:

$$\mathcal{L}_{\text{recon}} = \frac{1}{n}\sum_{i=1}^{n} \|x_i - \hat{x}_i\|^2 \tag{11}$$

where x_i is the input data point, and $\hat{x}_i$ is the reconstructed data point.

2) Joint Training: In this second phase, the model is trained with a combined objective function that includes both the reconstruction loss, calculated as the mean squared error between the input and its reconstruction, and the clustering loss, i.e., a loss that measures how well the clustering assignments match the distribution of the data in the embedded space. This phase further refines the encoder weights to optimize both reconstruction and clusterability based on the combined loss function, expressed as:

$$\mathcal{L} = \alpha \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{cluster}} \tag{12}$$

where α and β are hyperparameters that control the weight of the reconstruction and clustering losses, respectively.

The clustering loss for AEML consists of multiple steps and is detailed in [33]. Here we introduce its general form, defined as:

$$\mathcal{L}_{\text{cluster}}^{\text{AEML}} = \text{CustomLoss}(Z) \tag{13}$$

where Z is the set of embedded representations of the data points and CustomLoss is a function that measures clustering quality based on the relative distances between the samples.

The clustering loss for DCEC, which utilizes the Kullback-Leibler divergence, is detailed in [34]. However, we also briefly introduce it here for completeness. It is calculated as:

$$\mathcal{L}_{\text{cluster}}^{\text{DCEC}} = \text{KL}(P\|Q) = \sum_{i=1}^{n}\sum_{k=1}^{K} p_{ik} \log \frac{p_{ik}}{q_{ik}} \tag{14}$$

where P is the target distribution, Q is the predicted distribution of the clusters, and p_{ik} and q_{ik} are the probabilities that point x_i belongs to cluster k.

VI. EVALUATION METHODOLOGY

In this section, we elaborate on the methodological details of the experiments carried out to assess the performance of the proposed architecture. First, we elaborate on the training data for model development, followed by a summary of the evaluation metrics used, and we end with considerations of the evaluation approach.

A. TRAINING DATA

To evaluate the performance of the proposed architecture we selected three real-world datasets, collected from research testbeds, in particular (i) the Technology Classification Dataset - Labeled (TCD-L) [25] containing DVB-T, LTE and WiFi transmissions, (ii) the ITS-L dataset [24] containing signals from 5 different RAT, i.e., LTE, 5G, WiFi, ITS-G5 and C-V2X, and (iii) the LOG-a-TEC dataset [43] comprised of LoRa, IEEE 802.15.4, SIGFOX and some proprietary technologies. This selection of three very diverse real-world datasets from the 868 MHz, 2.4 GHz and 5.9 GHz frequency bands enables the most extensive evaluation of unsupervised methods to date.

1) TECHNOLOGY CLASSIFICATION LABELED DATASET

We purposely selected the labeled Technology Classification Dataset - Labeled (TCD-L) [25], collected from different neighborhoods located in Ghent, Belgium. During the data collection process, the transmission times and settings were also recorded, therefore this is a real-world labelled dataset. However, we only use the labels for evaluation and not for training. The data captures the influence of the different environments on the different wireless technologies that are operating in the 2.4 GHz band, namely LTE, WiFi, and DVB-T. In the original dataset, samples are collected in different bands specific to each technology, while here, we normalize them in common 1024 FFT bins as if they were coexisting in the same channel. Samples of the captured transmissions by each SSDC are depicted in Figure 4, showing that each of the RATs has its own characteristic shape in the FFT domain.

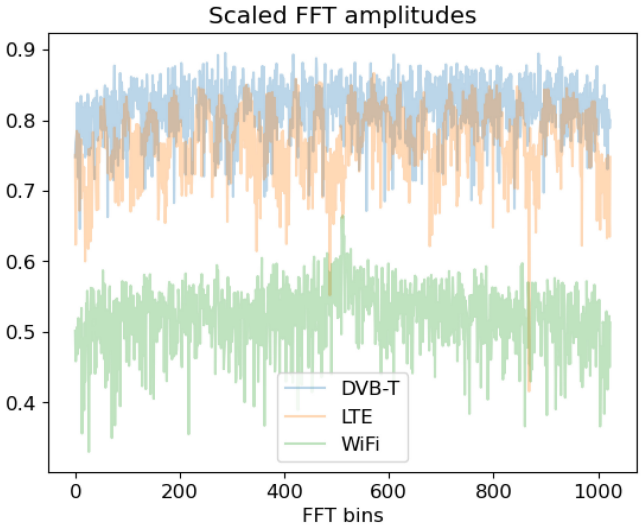


FIGURE 4. Samples for each signal type in the TCD-L dataset.

2) INTELLIGENT TRANSPORTATION SYSTEMS LABELED DATASET

The Intelligent Transportation Systems - Labeled (ITS-L) dataset [24] was collected in the region of Antwerp, Belgium. It contains signals from 5 different RATs (LTE, 5G, WiFi, ITS-G5 and C-V2X). The RAT technologies in the dataset are expected to coexist in the Intelligent Transportation Systems (ITS) 5.9 GHz band. By including the ITS-L, we expect to provide insights about the performance of the models when more types of RAT co-exist (i.e., 5 + noise compared to the 3 in the TCD-L), from which some are significantly different regarding their shape (e.g., WiFi in blue and C-V2X in red in Figure 5), and some are very similar (e.g., LTE in green and 5G in orange in Figure 5).

3) ULTRA NARROW BAND UNLABELED DATASET

Finally, we also included one unlabeled, continuously sensed dataset collected from the LOG-a-TEC testbed [43] in Ljubljana, Slovenia. This dataset consists of real-world spectrum traces in an ultra-narrow bandwidth of 192 kHz inside the European 868 MHz SRD band, which were sensed with a frequency of 5 PSD measurements per second, using 1024 FFT bins. According to an inspection of parts of the data, there are at least 4 technologies appearing in the recordings: LoRa, IEEE 802.15.4, SIGFOX, and some proprietary technology. The goal of employing such data is to check the performance of the evaluated architectures in real-world scenarios, which exposes them to a significantly wider

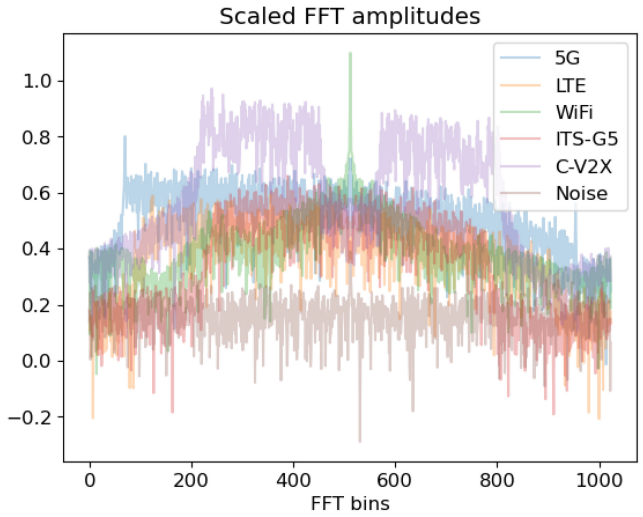


FIGURE 5. Samples for each signal type in the ITS dataset (ITS-L).

variety in the data that is being used, with all the artifacts and interference that occur during the acquisition. This makes the problem of SSDC monitoring much more challenging and closer to the potential deployment environment.

B. TRAINING CONFIGURATIONS

In this subsection, we provide the training configurations of the three architectures used in subsequent performance evaluation, the proposed SSDC architecture and the two AE-based architectures, AEML and DCEC.

1) CONFIGURATION OF THE CLUSTERING MODULE

Evaluations are performed by measuring the K-means clustering performance on the embedded vectors provided by the models created with the proposed architecture SSDC and two reference architectures, i.e., AEML and DCEC, based on multiple metrics described in the following Section VI-C. We approached the experimental evaluation assuming no information about the exact types and quantity of existing RATs in each of the datasets. Thus, for all three datasets, we set the upper number of possible RATs to 10, which is higher compared to the actual number of classes 3 and 6 in the respective labeled datasets TCD-L and ITS-L, and approximately 5 according to [43] in the unlabeled dataset UNB-U. Selection of an approximate number of classes (i.e., operating RATs) is also viable in real-world deployment based on the monitored frequency band, urban/rural area, etc. While such an approach closely imitates the real-world environment, following the problem formulation in Section IV, it also sets fair ground for comparison of the evaluated models. In this way, we avoid potential bias induced by training and evaluation with the actual number of classes, considering that the SSDC employs a clustering algorithm in its training pipeline, as detailed in Section V-A.1, while the AE-based models work without clustering feedback. Furthermore, a higher number of clusters may capture more details and variance within the data, potentially leading to more accurate representations of smaller, nuanced groups. However, this number (10 in our case) should not be too big since too many clusters might also mean overfitting to noise and outliers, making some clusters less meaningful.

2) CONFIGURATION OF THE CNN MODULES

To ensure fair comparisons across the three architectures, all models were constructed with the identical CNN module, as proposed and detailed in Section V-A.2. The implementation was written in Python programming language, using the Pytorch library. The training of the SSDC model was performed with the Adam optimizer with learning rate set to 10^{-3}, and weight decay set to 10^{-5}. The training iterations for the SSDC were set to 250 epochs. The AE-based models AEML and DCEC were configured following the description in Section V-C regarding the loss functions. The weight parameters α and β were set to 1, which gives the same importance for both components of the combined loss function, expressed with Eq. (12). The training was performed with the same configuration of the optimizer as for the SSDC model. Regarding the training iterations, considering the AE-based architectures work in two stages as described in Section V-C, the pre-training ran for 200 epochs, and the second stage of joint-training ran for 50 epochs, totaling 250 epochs, same as for the SSDC model. The embedded space dimensionality for all three models was set to 10.

C. CLUSTER QUALITY EVALUATION METRICS

For better understanding and analyzing the embedded space, we provide three different types of visualizations of the clustering in the embedded space, namely t-distributed Stochastic Neighbor Embedding (t-SNE) [44] of the embedded space, average of samples per cluster, and distribution of ground-truth classes per cluster.

Regarding the quantitative evaluation, we employ multiple existing metrics that are specific to the clustering approaches. According to the way the used metrics are calculated, they can be divided into two groups. The first group is based on labels, mainly evaluating the content/purity of the clusters regarding the different RATs. The second group of metrics is based on the distances in the embedded space, mainly considering the inter-cluster and intra-cluster distances.

1) VISUAL EVALUATION

t-SNE is useful for transforming high-dimensional data into a two-dimensional or three-dimensional space for visualization, thus revealing the structure and patterns in complex datasets. It is particularly suitable for examining the outcomes of the clustering algorithm and visualizing how various data classes are distributed. However, it is important to note that t-SNE visualizations should be taken cautiously due to their nonlinear transformation and dimensionality reduction. They may not necessarily reflect the same data structure that exists in the higher-dimensional embedded

space. Thus, the t-SNE visualizations will be considered as informative and complementary views, supporting the cross-interpretation of multiple metrics when drawing conclusions for the performance of such self-supervised models. Due to the large size of the data, the t-SNE embedded space visualization is performed with a randomly sampled subset of $10,000$ points instead of the entire dataset. These visualizations aim to showcase the approximate structure of the clusters created by each model within the feature space.

Calculation and visualization of an average of the samples per cluster will show what the dominant type of RAT for each cluster is and enable the expert mapping of the identified clusters to actual RAT, as described in Section III-D. The averaging will suppress the less common types of RATs and highlight the representative shape for each cluster. This visualization is particularly useful when unlabeled data is considered, which cannot be analyzed by the label-based metrics.

For the evaluation with labeled datasets, we also show the distribution of samples in each cluster regarding the ground-truth labels, providing one additional perspective of the clustering directly related to the label-based metrics described in the following section.

2) LABELS-BASED METRICS

Regarding the labels-based evaluation, we utilize 4 different metrics, namely Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), Clustering homogeneity score (CHS) and Clustering completeness score (CCS).

Normalized Mutual Information is a clustering metric that quantifies the mutual information between true class labels and cluster assignments while accounting for the scale of each. It ranges from 0 to 1, with higher values indicating better agreement between true classes and clusters. Given two cluster assignments U and V, NMI is defined as:

$$\mathrm{NMI}(U, V) = \frac{2 \cdot I(U; V)}{H(U) + H(V)}, \tag{15}$$

where: $I(U; V)$ is the mutual information between U and V and $H(U)$ and $H(V)$ are the entropies of U and V, respectively.

Adjusted Rand Index [45] metric compares the pairwise decisions (whether pairs of elements are in the same or different clusters) in the clustering outcome to the true labels, adjusting for chance grouping. Formally, it is expressed as:

$$\mathrm{ARI} = \frac{\mathrm{RI} - \mathbb{E}[\mathrm{RI}]}{\max(\mathrm{RI}) - \mathbb{E}[\mathrm{RI}]}, \tag{16}$$

where RI is the Rand Index and $\mathbb{E}[\mathrm{RI}]$ is the expected value of the Rand Index. It can range from -1 (indicating completely independent labeling) to 1 (perfectly matching labeling).

The Clustering homogeneity score [46], as a clustering metric, evaluates the quality of a clustering operation, assessing whether each cluster contains only members of a single class. It is defined as:

$$\mathrm{Homogeneity} = 1 - \frac{H(C|K)}{H(C)}, \tag{17}$$

where $H(C|K)$ is the conditional entropy of the classes given the cluster assignments, and $H(C)$ is the entropy of the classes. The CHS has a range of values between 0 and 1. A score of 0 is achieved when the clusters are completely mixed, meaning that each cluster contains an equal proportion of members from different classes. A score of 1 is achieved when each cluster contains members from only one class and no class is spread across multiple clusters, representing perfect homogeneity.

The Clustering completeness score [46] is a metric evaluating the quality of clustering results in terms of how well it groups together elements of the same class. Completeness is computed based on the conditional entropy of the clusters given the class labels. It essentially measures whether all members of a given class are assigned to the same cluster, regardless of how many other classes are also present in that cluster, defined as:

$$\mathrm{Completeness} = 1 - \frac{H(K|C)}{H(K)}, \tag{18}$$

where $H(K|C)$ is the conditional entropy of the cluster assignments given the classes and $H(K)$ is the entropy of the cluster assignments. CCS score ranges from 0 to 1. A score of 0 indicates that the algorithm has dispersed members of a single class across multiple clusters, failing to group them together, and a score of 1 signifies that all members of each class are perfectly grouped within a single cluster.

All four labels-based metrics provide quantitative measures to evaluate the performance of clustering algorithms when true labels are known, which is the case for the TCD-L and ITS-L datasets. Each metric provides a different perspective of the clustering performance, considering the distribution of samples across the clusters relative to the ground-truth labels.

3) DISTANCE-BASED METRICS

For measuring the clustering performance based on the state in the embedded space, we combined metrics used in [35] and [29], i.e., silhouette score, Davies-Bouldin score, and Calinski-Harabasz index.

The silhouette score [47] is a metric used to measure the goodness of a clustering technique. It quantifies how well-separated clusters are in a dataset, ranging from -1 to 1. A higher silhouette score indicates better-defined clusters, with scores closer to 1 indicating more cohesive and separated clusters, while scores near 0 suggest overlapping clusters. A silhouette score that trends toward -1 suggests that many points in the dataset have been placed in inappropriate clusters. Considering a sample i, the average distance between i and all other points in the same cluster $a(i)$, and the minimum average distance between i and points in a

different cluster $b(i)$, minimized over clusters, the silhouette score $s(i)$ is given by:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}. \tag{19}$$

The overall silhouette score for the dataset is the mean silhouette score of all samples.

The Davies-Bouldin [48] score is a clustering evaluation metric that measures the compactness and separation of clusters. Theoretically, this score ranges from 0 upwards, with no fixed upper limit. Formal definition is as follows: considering k clusters, with C_i being the centroid of cluster i and S_i being the average distance of all points in cluster i to the centroid C_i, and similarity measure R_{ij} between clusters i and j defined as:

$$R_{ij} = \frac{S_i + S_j}{\|C_i - C_j\|}, \tag{20}$$

the Davies-Bouldin score DB is the average of the maximum R_{ij} for each cluster:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} R_{ij}. \tag{21}$$

Low scores indicate good clustering quality, where clusters are compact (data points within clusters are close to each other) and well-separated (clusters are far apart from each other).

The Calinski-Harabasz [49] index is a clustering evaluation metric that quantifies the ratio of variance between clusters to variance within clusters. The score can theoretically be as low as zero, indicating extremely poor clustering where the within-cluster variance is as high as the total variance. However, in practice, any non-trivial clustering will result in a score greater than zero. There is no theoretical upper limit to the score. Higher scores indicate better clustering performance, with more significant separation between clusters compared to the variance within clusters. Formal definition is as follows: given k clusters and n data points, B_k as between-group dispersion matrix and W_k the within-group dispersion matrix, the Calinski-Harabasz index CH is defined as:

$$CH = \frac{\text{trace}(B_k)/(k-1)}{\text{trace}(W_k)/(n-k)}, \tag{22}$$

where $\text{trace}(B_k)$ is the trace of the between-group dispersion matrix and $\text{trace}(W_k)$ is the trace of the within-group dispersion matrix.

The Davies-Bouldin, silhouette, and Calinski-Harabasz scores work based on the clustering result and inter-sample distances. Such metrics are crucial to provide a quality assessment of the clustering result when labels are absent, which is the case with the unlabeled dataset UNB-U. Furthermore, evaluation with Davies-Bouldin, silhouette, and Calinski-Harabasz can also provide valuable insights when working with labeled data because, in general, they quantify the quality of the clustering result based on the inter-cluster and intra-cluster distances, independent of the ground-truth labels.

D. TRANSMISSION DETECTION EVALUATION METRIC

Based on the mapping of the clusters to specific RAT classes, performed as described in Section III-C, we can evaluate the monitoring by calculating the standard multiclass classification evaluation metrics *precision*, *recall* and *F1 score*. Since the problem of monitoring at this stage could be interpreted as multiclass classification, the evaluation with the aforementioned metrics is performed per class.

Thus, considering $\mathcal{C} = \{1, 2, \ldots, K\}$ is the set of classes, for each class $k \in \mathcal{C}$:

- Precision P_k is the ratio of true positives to the sum of true positives and false positives for class k:

$$P_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FP}_k}. \tag{23}$$

- Recall R_k is the ratio of true positives to the sum of true positives and false negatives for class k:

$$R_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k}. \tag{24}$$

- F1 score for class k is the harmonic mean of precision and recall for class k:

$$F1_k = \frac{2 \cdot P_k \cdot R_k}{P_k + R_k}. \tag{25}$$

This evaluation is performed using the models trained with the two labeled datasets, TCD-L and ITS, since these metrics necessitate ground-truth labels. For predicted labels, we use the identified labels for each formed cluster based on the visual evaluation. In the case of having clusters that contain multiple classes, we identify them based on the most dominant type of samples.

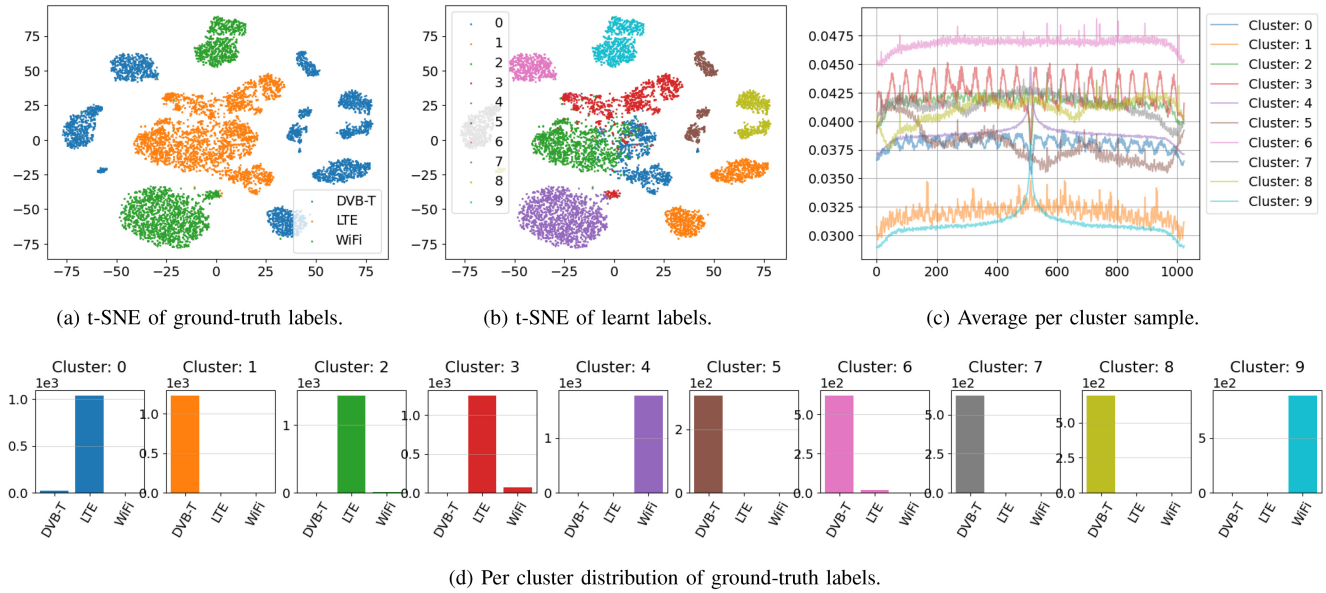
E. COMPUTATIONAL PERFORMANCE METRICS

As part of the evaluation, we also assess the potential deployment challenges of the models regarding their structure and size based on the number of trainable parameters and computational requirements based on the calculation of one billion floating-point operations per second (GFLOPS). GFLOPS is a measure of computational performance required or achieved by the model during the training and inference phases. It quantifies the number of floating-point operations (FLOPs) a model needs to execute per second in the billions (giga-). The calculations of the GFLOPS were carried out using the **fvcore**[2] library.

VII. RESULTS

The following results show the evaluation of models obtained from each of the three architectures, the proposed SSDC and the AEML and DCEC reference SotA architectures, with three different datasets, each of them addressing a specific challenge, as described in Section VI-A.

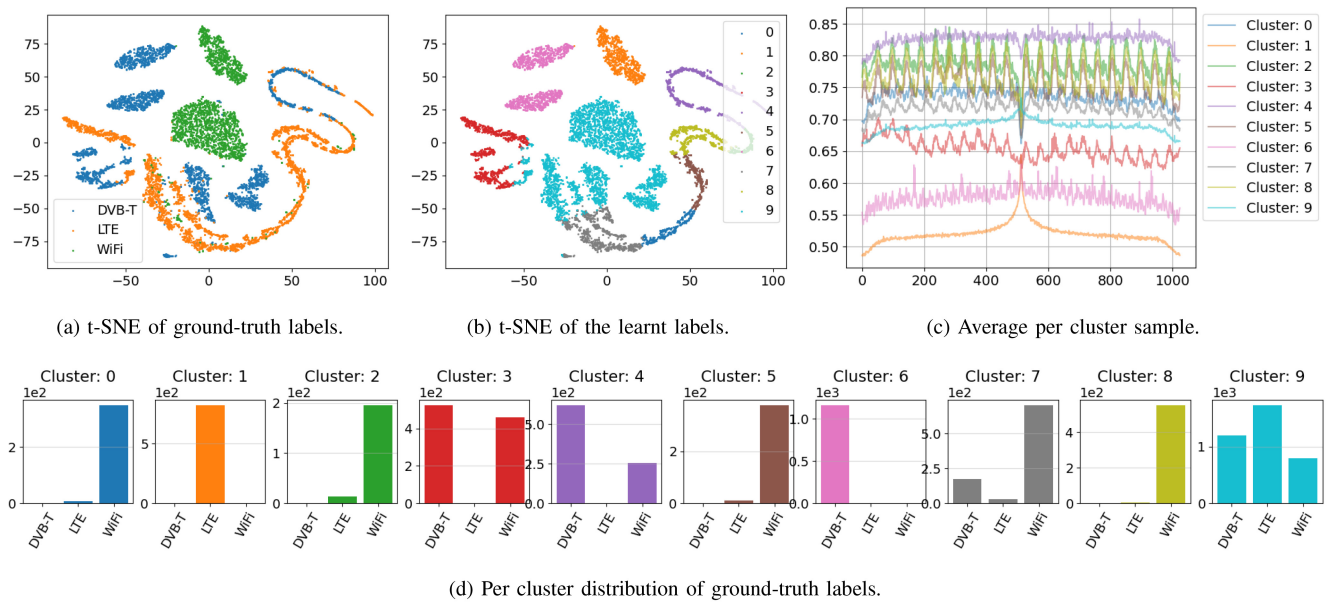
[2]https://github.com/facebookresearch/fvcore



(a) t-SNE of ground-truth labels. (b) t-SNE of learnt labels. (c) Average per cluster sample.

(d) Per cluster distribution of ground-truth labels.

FIGURE 6. Evaluation of the model learnt with the SSDC architecture and TCD-L dataset.



(a) t-SNE of ground-truth labels. (b) t-SNE of the learnt labels. (c) Average per cluster sample.

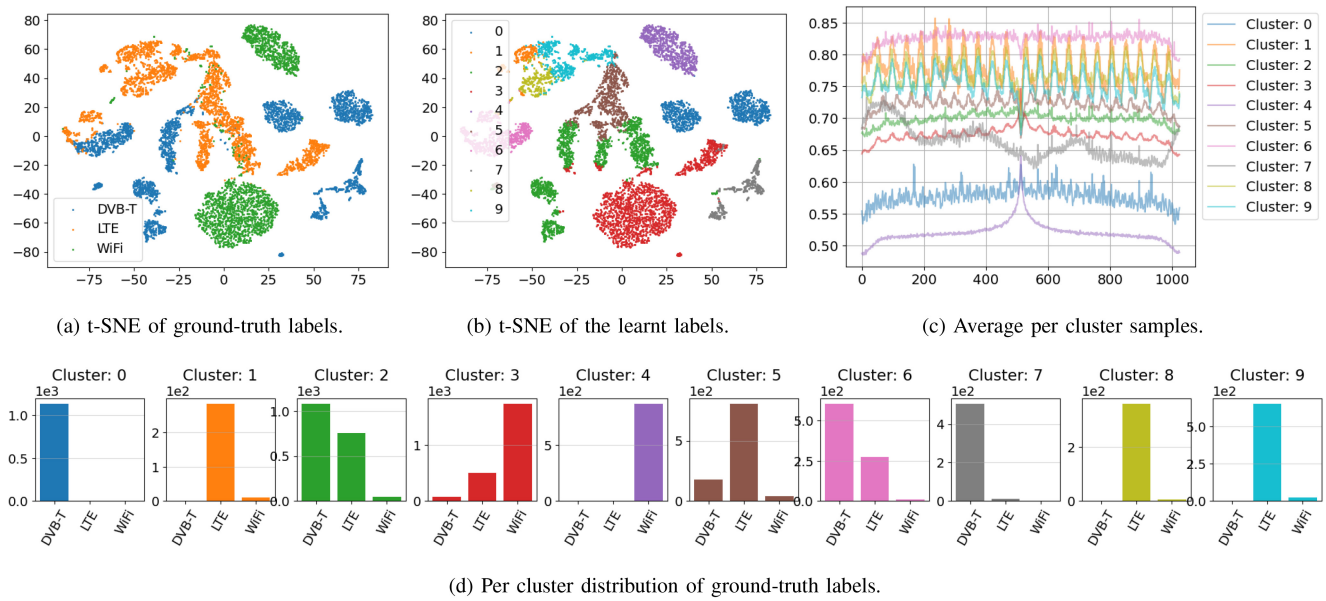
(d) Per cluster distribution of ground-truth labels.

FIGURE 7. Evaluation of the model learnt with the AEML architecture and TCD-L dataset.

A. EVALUATION RESULTS WITH TCD-L

1) VISUAL ANALYSIS

Figures 6(a), 7(a) and 8(a) depict the two-dimensional t-SNE projected features learnt by each of the three models developed. The number of depicted clusters is 10, as selected in Section VI-A, while the colors represent the three types of RAT, namely DVB-T (blue), LTE (orange), and WiFi (green). From Figure 6(a), it can be seen that while the model learns to create 10 clusters due to the clustering parameter selection, these clusters are well separated and homogeneous, with several of them containing the same technology. For instance, there are several blue clusters, all containing DVB-T, one large orange cluster containing LTE, and two large and one small green clusters containing WiFi. Although each model provides the same number of clusters, their shapes vary significantly. As can be seen in Figure 6(a), the SSDC model shows more tendency towards the creation of clusters with elliptic shapes, while the other two, the AEML in Figure 7(a) and DCEC in Figure 8(a) have more variety in the cluster shapes. The distribution of the data input into the clustering methods, i.e., the distribution of the learnt embeddings in our case, is less suitable for well-shaped spheric clusters for the AEML and DCEC models compared to the SSDC. These two characteristics could be considered important advantages for the SSDC model since it provides more "clustering-friendly" feature space.



(a) t-SNE of ground-truth labels.

(b) t-SNE of the learnt labels.

(c) Average per cluster samples.

(d) Per cluster distribution of ground-truth labels.

FIGURE 8. Evaluation of the model learnt with the DCEC architecture and TCD-L dataset.

In Figures 6(b), 7(b) and 8(b), we show the same feature space embedding, but this time colored with the 10 labels corresponding to the 10 clusters that were learnt by each model. As can be seen in the figures, each of the formed clusters is largely homogeneous, consisting of samples mostly of a single RAT technology. Looking at the largest cluster in Figure 6(b), we know from the ground-truth depiction in Figure 6(a) that it contains only LTE transmissions. It can be seen that it learnt to separate it into three different subclusters (0-blue, 2-green, and 3-red), where each of them contains samples from the same technology. This is also depicted by the clusters' distribution plot in Figure 6(d), with the same coloring as for the K-means result in Figure 6(b). Thus, the averaged clusters' samples in Figure 6(c), also colored with the corresponding colors of the 10 clusters provided by the K-means, are aligned with this, showing the specific shapes for each of the technologies. The additional insight provided by the average plot is that the K-means groups the samples based on the RAT technology, where signal strength has a significant impact on the features, thus leading to the creation of separate clusters of the same technology.

Looking at the results of the reference autoencoder-based models depicted in Figures 7 and 8 for the AEML and DCEC respectively, they seem to have more condensed clusters in the feature space compared to the proposed SSDC model, which means that the samples of each technology in Figures 7(a) and 8(a) are less dispersed in the feature space. However, the fragmented and free forms of the clusters lead to greater confusion/misclassification when performing the K-means clustering. Thus, the resulting K-means clusters, according to Figures 7(b) and 8(b), contain samples from different technologies. For example, cluster 3-red and cluster 9-cyan in Figure 7, which are part of the elongated groups of samples, contain both DVB-T and LTE transmissions. A similar effect of clusters containing a mix of RATs is even more prominent in the visualizations for the DCEC model instance, as can be seen in Figure 8. Clusters learnt by the K-means algorithm in the DCEC provided feature space, visualized in Figure 8(b) with assigned labels 2-green, 5-brown, and 6-pink, contain samples of DVB-T and LTE, and cluster 3-red contains LTE and WiFi, as can also be seen on the per cluster distribution of ground-truth labels in Figure 8(d). This phenomenon is also leading to less distinguishable average samples in Figure 8(c). For example, the shape of cluster 3-red, which has a dominant group of WiFi samples, does not have a smooth characteristic shape, which can be noticed for the average of cluster 4-purple, which has WiFi samples only.

2) QUANTITATIVE ANALYSIS

1) Evaluation with labels-based metrics: As it was described in Section VI, besides the analysis based on visual evaluation of the clustering results, we also performed a quantitative evaluation based on multiple clustering-specific metrics. The results are summarized in Table 1, which lists the evaluation metrics in the first column, followed by the models obtained with the proposed architecture in column two and the reference autoencoder-based architectures in columns three and four, respectively. From the first row of Table 1, it can be seen that NMI of the proposed SSDC is almost double the value of the reference AE-based models indicating stronger mutual dependence between the clustering results and the true labels, considering that 0 NMI means no dependence. The clustering assignments produced by K-means on the SSDC-provided feature-space more successfully identified the inherent groupings within the data that align closely with the ground truth categories, compared to the K-means

TABLE 1. Evaluation of clustering by K-means on the different embedded spaces with TCD-L.

Metric \ Model	SSDC	AEML	DCEC
NMI ↑	**0.6209**	0.3266	0.3998
ARI ↑	**0.3874**	0.1362	0.23
CHS ↑	**0.9465**	0.4595	0.5897
CCS ↑	**0.4620**	0.2533	0.3024
Silhouette ↑	0.3174	**0.6050**	0.4325
Davies-Bouldin ↓	1.8727	**0.5286**	1.0396
Calinski-Harabasz ↑	18029	**1259829**	357440

Legend: ↑ - higher the better, ↓ - lower the better.
Labels-based metrics, Distance-based metrics

TABLE 2. Performance of different models on multiclass classification.

Metric	Class	SSDC	AEML	DCEC
Precision	DVB-T	**0.99**	0.76	0.89
	LTE	**0.97**	0.89	0.81
	WiFi	**1.00**	0.57	0.94
Recall	DVB-T	**1.00**	0.63	0.79
	LTE	**1.00**	0.59	0.88
	WiFi	0.96	**0.97**	0.96
F-score	DVB-T	**1.00**	0.69	0.84
	LTE	**0.98**	0.71	0.84
	WiFi	**0.98**	0.72	0.95

Higher is better.

assignments on the feature-space of the reference AE-based models AEML and DCEC.

Considering the ARI metric values in Table 1, all three models have values above 0, meaning that their developed clusters are better than random clustering. Looking at each model's score separately, SSDC leads by a margin of 0.14 to the second DCEC. This indicates that SSDC has the greatest similarity between the clustering assignments and the ground-truth labels. The CHS scores show very high values for SSDC, meaning that the formed clusters contain mostly samples from a single class (RAT), while in the embedded spaces of the reference AE-based models, there is much more variety in the clusters, supporting the conclusions made from the visual analysis. Based on the CCS metric, the proposed SSDC is again much better (by a factor of 1.5 to 1.8). However, the values of all three models are in general low (the highest 0.4578 for SSDC) due to a large number of clusters (10) compared to the number of existing RATs (3) in this dataset.

In general, SSDC is superior according to the labels-based evaluation, DCEC is second, and the worst performance is achieved with the AEML model.

2) Evaluation with distance-based metrics: While all the labels-based metrics have shown a significant advantage for the proposed SSDC model, the distance-based metrics silhouette, Davies-Bouldin and Calinski-Harabasz (last three rows in Table 1) show a clear advantage of AEML. The higher silhouette score of the AEML model indicates better-separated clusters, meaning that there is better cohesion of the inter-cluster samples and better separation between clusters, compared to SSDC and DCEC. The advantage of better separation between the clusters and compactness of each cluster is already seen in Figures 6-8. The reference AE-based models provide more dense clusters, with larger distances between them, while SSDC has more dispersed clusters. Davies-Bouldin and Calinski-Harabasz scores (last two rows in Table 1), which relate to the within-cluster dispersion and between-cluster separation, again indicate the more compact and better-separated clusters of the AEML model compared to SSDC and DCEC.

In general, both the labels-based and distance-based metrics are in line with the observations in the visual analysis of the embedded space (Figures 6, 7 and 8) where better-separated feature space was provided by the AEML. However, the clusters were much cleaner with the proposed SSDC model mostly having single RAT per cluster. This means that the SSDC model successfully learns more representative features for the different RATs compared to the reference AEML and DCEC models, and also exhibits better generalization capabilities, considering that the TCD-L dataset is acquired across multiple environments.

3) Spectrum Monitoring Evaluation: As described in Section VI-D, we evaluate the spectrum monitoring with the different models based on the identified classes, using the standard classification metrics, precision, accuracy, and F1-score. The results are summarized in Table 2. Precision, recall, and F1-score are calculated per class for the three existing classes in the TCD-L dataset. The SSDC model outperforms the AE-based models by a significant margin, up to 40 ppt, in all but one case. There is a negligible difference in the Recall of the WiFi class, where the AEML model has 1 ppt better performance over the other two, SSDC and DCEC. This evaluation again confirms the superior generalization capabilities of the SSDC model compared to the AE-based models, considering that the TCD-L dataset contains data from different environments, as described in Section VI-A, and the potential to use it for training a classifier without labeled data.

B. EVALUATION RESULTS WITH ITS-L

Following the methodology detailed in Section VI, in this paragraph, we analyze the results of the evaluation with the second labeled dataset ITS-L. The evaluation is performed in the same order as for the previous TCD-L dataset, with a focus on the quantitative analysis.

1) EVALUATION WITH LABELS-BASED AND DISTANCE-BASED METRICS

The performance of the models is depicted in Table 3, where again, the first four rows with blue background show the performance measured with the labels-based matrices, and

TABLE 3. Evaluation of clustering by K-means on different embedded spaces, ITS-L.

Metric \ Model	SSDC	AEML	DCEC
NMI ↑	**0.8884**	0.3266	0.8478
ARI ↑	**0.7902**	0.1362	0.7887
CHS ↑	**0.9955**	0.4595	0.9405
CCS ↑	**0.8022**	0.2533	0.7718
Silhouette ↑	0.5892	0.6050	**0.6819**
Davies-Bouldin ↓	0.6953	**0.5286**	1.0396
Calinski-Harabasz ↑	309780	**1259829**	1102317

Legend: ↑ - higher the better, ↓ - lower the better.
Labels-based metrics, Distance-based metrics

TABLE 4. Performance of different models on multiclass classification.

Metric	Class	SSDC	AEML	DCEC
Precision	CV2X	0.94	**1.00**	**1.00**
	5G	**1.00**	0.99	0.95
	ITSG5	**1.00**	0.89	0.99
	LTE	**1.00**	0.93	0.99
	WiFi	**1.00**	0.97	0.98
	Noise	**1.00**	0.88	**1.00**
Recall	CV2X	**1.00**	0.99	0.99
	5G	**1.00**	0.89	0.98
	ITSG5	**0.96**	**0.96**	0.94
	LTE	0.99	0.96	**1.00**
	WiFi	0.98	0.83	**0.99**
	Noise	**1.00**	**1.00**	**1.00**
F1-score	CV2X	0.97	**0.99**	**0.99**
	5G	**1.00**	0.94	0.97
	ITSG5	**0.98**	0.92	0.96
	LTE	**1.00**	0.94	0.99
	WiFi	**1.00**	0.90	0.99
	Noise	**1.00**	0.94	**1.00**

Higher is better.

the last three rows with orange background show the values of the three distance-based metrics.

Regardless of the larger number (6) of different types of signals (5 RATs + noise) in this dataset, the performance of all three models follows a similar pattern as in the evaluation with the TCD-L dataset with 3 RATs. Regarding the labels-based metrics, the SSDC model outperforms the other two across all four metrics. DCEC is very close to the SSDC on the labels-based metrics, whereas AEML performs quite badly, which was not the case for the evaluation with the TCD-L dataset. The relatively high performance of the DCEC means that the AE structure is capable of capturing the different shapes of the signals. However, different loss functions used in the two AE-based architectures have led to the creation of different structures in the embedded domain. The deep clustering component of the loss function of the AEML model is more dominant compared to the reconstruction loss, leading to very compact clusters in the embedded domain while minimizing the influence of the shape of the samples in the learning process. This is confirmed by the high performance on the distance-based metrics of the AEML model, providing compact and well-distanced clusters containing samples from different RATs. The SSDC model shows the most balanced performance, with comparably distinguishable clusters as the AE-based models, which are also homogeneous, containing mostly single RAT samples. Overall, the SSDC successfully captures the RAT-specific features when there is a larger number of RATs, outperforming the AE-based models.

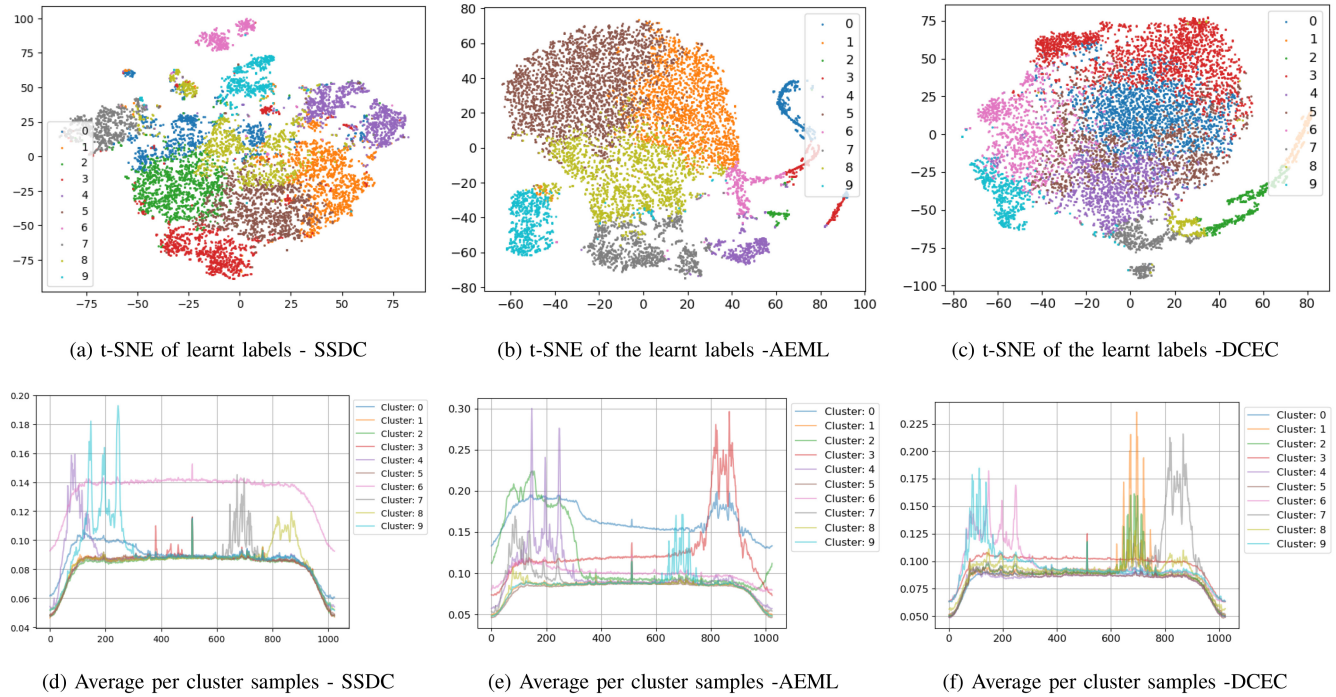
1) Spectrum Monitoring Evaluation: Spectrum monitoring performance of the three models on the ITS dataset, regarding the precision, recall, and F1-score, is summarized in Table 4. The highest scores are bolded in the table. While the SSDC model achieves the highest score in most cases across the evaluation with the three metrics across the six classes, it is evident that it has a very similar performance with the DCEC model. In general, we can notice a similar performance pattern as in the previous labels-based evaluation, demonstrated in Table 3. This means that the SSDC and DCEC can provide high-performance classifiers using completely unlabeled data, with only providing identification of the clusters, as part of the Phase 3, described in Section III-C.

C. EVALUATION RESULTS WITH UNB-U

1) VISUAL EVALUATION

Figure 9 depicts the visual evaluation of the three models with the unlabeled dataset. In this case, since the ground-truth labels are not available, we can only analyze as part of the visual evaluation the t-SNE projections (Figures 9(a), 9(b) and 9(c)) and the cluster averages (Figures 9(d), 9(e) and 9(f)). Considering that this dataset is acquired by continuous sensing, we expect it to have many noisy samples without clear distinction boundaries between them. Also, if there are significant amounts of samples with specific shapes, which would result from some specific RAT operating in the monitored band, we expect them to be grouped together in a separate cluster in the feature space.

Comparing the t-SNE visualizations, the feature space of the SSDC model appears to be separated into more distinctive and better-rounded clusters as shown in Figure 9(a) compared to AE-based models shown in Figures 9(b) and 9(c). This makes the SSDC-provided feature space advantageous for further analysis since it is easier to isolate each specific type of RAT, regardless of the type of clustering algorithm being used. Besides, the AE-based models contain some well-separated clusters, such as the cluster 9-cyan in Figure 9(b). There are also the elongated shapes that result in merged clusters that contain different RATs, such as the clusters 3-red and 0-blue in Figure 9(b) and clusters 2-green and 8-yellow in Figure 9(c). The same effect was also noticed in the evaluation with the TCD-L labeled dataset in Section VII-A.



(a) t-SNE of learnt labels - SSDC (b) t-SNE of the learnt labels -AEML (c) t-SNE of the learnt labels -DCEC

(d) Average per cluster samples - SSDC (e) Average per cluster samples -AEML (f) Average per cluster samples -DCEC

FIGURE 9. Feature-space and average clusters of the different models.

These observations are also confirmed with the per-cluster averages depicted in Figures 9(d), 9(e) and 9(f). The well-distinguished clusters are easily recognizable shapes that are specific for different RATs. This allows for further fine-grain analysis of the smaller clusters, which would not be possible for the other two, AEML and DCEC, which show only 3-5 distinguishable clusters. Looking at the averages of the clusters of each model, all of them appear to have formed clusters for mostly similar transmission types. However, the merged clusters which were notable in the feature space visualization in Figures 9(b) and 9(c) are also influencing the per-cluster averages in Figures 9(e) (e.g., cluster 0-blue and cluster 3-red) and 9(f) (e.g., cluster 6-pink and cluster 9-cyan).

Based on analysis of the same dataset (UNB-U) in [43], we can tell which RATs represent some of the formed clusters based on the shape of the per-cluster average. In Figure 9(d), cluster 6-pink are IEEE 802.15.4 transmissions, clusters 7-gray and 9-cyan are two different proprietary transmissions, and cluster 4-purple are LoRA transmissions.

2) QUANTITATIVE EVALUATION

Quantitative evaluation of the three models with UNB-U dataset with the distance-based metrics is summarized in Table 5. Results support the previous conclusions based on the visual analysis of the feature space and the per-cluster averages. The existence of some well-separated clusters has led the SSDC model to significantly outperform the other two AE-based reference models according to the silhouette score, i.e., AEML by almost 40% and DCEC by 150%, which was not the case in the labeled datasets. It also shows comparable performance based on the Davies-Bouldin score by ranking second. The relatively good performance of AE-based models results from the larger density of the formed clusters, which was also noticed in the evaluations with the labeled datasets. It is interesting to notice how the DCEC model performs significantly better on the Calinski-Harabasz score, which results from the good separation and density of some of the clusters on the margins of the feature space (i.e., cluster 1-orange, 9(c)).

TABLE 5. Evaluation of clustering by K-means on the different embedded spaces with UNB-U.

Metric \ Model	SSDC	AEML	DCEC
Silhouette ↑	**0.2056**	0.1484	0.0825
Davies-Bouldin ↓	1.5218	**1.3183**	1.631
Calinski-Harabasz ↑	57037	71520	**363927**

Legend: ↑ - higher the better, ↓ - lower the better..

Overall, judging by the distance-based metrics, all three models show comparable performance with the UNB-U unlabeled dataset. However, based on the visual evaluation, the clusters of the SSDC model appear to form a larger number of recognizable transmission types; thus, the SSDC model has more potential for further analysis because of the better segmentation of the feature space.

D. POTENTIAL DEPLOYMENT CHALLENGES

Regarding the architectural specifics, the SSDC model could be considered easier to configure (number of training epochs

TABLE 6. Computational complexity of the models.

Metric \ Model	SSDC	AEML / DCEC
Num. params.	**128406**	162827
GFLOPS	**0.196672**	0.38731776

and stopping criteria) compared to the reference AE-based models because of the following main characteristics.

1) SINGLE LOSS FUNCTION

Having two separate components that build the global loss function for the AEML and DCEC could lead to conflicting situations in the learning process. For example, lowering the deep clustering part of the loss in the embedded domain (increasing the distance between samples) could lead to worsening the reconstruction loss. Additionally, in AE-based models, there is no direct feedback from the actual classification process, such as the K-means for the SSDC model, which means that actual classification performance could be even degraded by creating unnecessary clusters and fragmenting the feature space while improving the deep clustering and reconstruction loss.

2) INHERENTLY DEVELOPED CLASSIFIER

The training of the SSDC inherently develops a classifier in parallel with the feature learning, which follows the assignments of the K-means algorithm. On the contrary, the AE-based models require another classification algorithm to work on the extracted features since they only provide encoding of the input data without class information.

3) LOWER COMPUTATIONAL COMPLEXITY

As we described in Section VI, the reference AE-based and the proposed SSDC architectures utilize the same CNN module. Consequently, during the inference mode (deployment scenario), all of the models will have a similar number of parameters. However, in the training mode, the SSDC models have around 22% fewer parameters than AE-based models because of the encoder-decoder structure, which requires symmetrical CNN module for decoding. Considering the GFLOPS calculation, as specified in Section VI, the proposed SSDC architecture requires approximately half of the amount of GFLOPS compared to the AE-based architecture. This means that tuning or retraining of the SSDC model will require significantly less computational resources, as summarized in Table 6. Smaller model sizes and lower computational requirements of the proposed SSDC model could significantly benefit the potential deployment in more restricted devices closer to the edge of the network, thus extending the range of potential applications.

VIII. CONCLUSION

In this paper, we investigated label-free RAT classification for spectrum sensing. We proposed a new spectrum sensing workflow, adequate for realistic set-ups that do not collect labels, based on a novel deep Self-Supervised Learning (SSL), that can be easily ported to various environments. This workflow enables the development of RAT classification models while avoiding the necessity of large labeled datasets. The SSL architecture is capable of autonomously learning low-dimensional features from raw FFT and enables the identification of different RATs without prior knowledge of the specific RATs in the monitored environment. The model is trained in a self-supervised manner, followed by manual cluster labeling by a human expert.

We evaluated the proposed architecture against state-of-the-art Autoencoder (AE)-based architectures using three real-world datasets acquired from three different frequency bands, 2.4 GHz, 5.9 GHz, and 868 MHz, containing signals from over ten different RATs. The trained model using the proposed architecture consistently outperformed the AE-based models by up to 31 ppt F1 score according to spectrum monitoring evaluation, while at the same time requiring 22% fewer trainable parameters and 50% fewer floating-point operations per second (FLOPS).

REFERENCES

[1] (Ericsson, Stockholm, Sweden). *Mobility Report*. Nov. 2023. Accessed: Feb. 20, 2025. [Online]. Available: https://ericsson.com/mobility-report

[2] W. K. Alsaedi, H. Ahmadi, Z. Khan, and D. Grace, "Spectrum options and allocations for 6G: A regulatory and standardization review," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 1787–1812, 2023.

[3] R. K. Saha and J. M. Cioffi, "Dynamic spectrum sharing for 5G NR and 4G LTE coexistence-a comprehensive review," *IEEE Open J. Commun. Soc.*, vol. 5, pp. 795–835, 2024.

[4] M. Parvini et al., "Spectrum sharing schemes from 4G to 5G and beyond: Protocol flow, regulation, ecosystem, economic," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 464–517, 2023.

[5] L. J. Wong, W. H. Clark, B. Flowers, R. M. Buehrer, W. C. Headley, and A. J. Michaels, "An RFML ecosystem: Considerations for the application of deep learning to spectrum situational awareness," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 2243–2264, 2021.

[6] Z. Ke and H. Vikalo, "Real-time radio technology and modulation classification via an LSTM auto-encoder," *IEEE Trans. Wireless Commun.*, vol. 21, no. 1, pp. 370–382, Jan. 2022.

[7] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 3, pp. 433–445, Sep. 2018.

[8] I. G. A. Poornima and B. Paramasivan, "Anomaly detection in wireless sensor network using machine learning algorithm," *Comput. Commun.*, vol. 151, pp. 331–337, Feb. 2020.

[9] Y. Ghanney and W. Ajib, "Radio frequency interference detection using deep learning," in *Proc. IEEE 91st Veh. Technol. Conf.*, 2020, pp. 1–5.

[10] J. Robinson, S. Kuzdeba, J. Stankowicz, and J. M. Carmack, "Dilated causal convolutional model for RF fingerprinting," in *Proc. 10th Annu. Comput. Commun. Workshop Conf. (CCWC)*, 2020, pp. 157–162.

[11] U. Demirhan and A. Alkhateeb, "Integrated sensing and communication for 6G: Ten key machine learning roles," *IEEE Commun. Mag.*, vol. 61, no. 5, pp. 113–119, May 2023.

[12] C. Ouyang, Y. Liu, H. Yang, and N. Al-Dhahir, "Integrated sensing and communications: A mutual information-based framework," *IEEE Commun. Mag.*, vol. 61, no. 5, pp. 26–32, May 2023.

[13] Z. Wei et al., "Integrated sensing and communication signals towards 5G-A and 6G: A survey," *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11068–11092, Jul. 2023.

[14] S. Tadik et al., "Digital spectrum twins for enhanced spectrum sharing and other radio applications," presented at the IEEE Int. Conf. Digit. Twins Parallel Intell., Jan. 2023.

[15] A. S. Abdalla, P. S. Upadhyaya, V. K. Shah, and V. Marojevic, "Toward next generation open radio access networks: What O-RAN can and cannot do!" *IEEE Netw.*, vol. 36, no. 6, pp. 206–213, Nov./Dec. 2022.

[16] C. Coletti et al., "O-RAN: Towards an open and smart RAN," O-RAN ALLIANCE, Alfter, Germany, White Paper, 2018.

[17] L. Bonati, M. Polese, S. D'Oro, S. Basagni, and T. Melodia, "Open, programmable, and virtualized 5G networks: State-of-the-art and the road ahead," *Comput. Netw.*, vol. 182, Dec. 2020, Art. no. 107516.

[18] M. Hoffmann and P. Kryszkiewicz, "Beam management driven by radio environment maps in O-RAN architecture," 2023, *arXiv:2303.11742*.

[19] S. D'Oro, L. Bonati, M. Polese, and T. Melodia, "OrchestRAN: Network automation through orchestrated intelligence in the open ran," in *Proc. IEEE Conf. Comput. Commun.*, 2022, pp. 270–279.

[20] G. D. Durgin, M. A. Varner, N. Patwari, S. K. Kasera, and J. Van der Merwe, "Digital spectrum twinning for next-generation spectrum management and metering," in *Proc. IEEE 2nd Int. Conf. Digit. Twins Parallel Intell. (DTPI)*, 2022, pp. 1–6.

[21] W. Zhang, M. Feng, M. Krunz, and A. H. Y. Abyaneh, "Signal detection and classification in shared spectrum: A deep learning approach," in *Proc. IEEE Conf. Comput. Commun.*, 2021, pp. 1–10.

[22] A. Scalingi, D. Giustiniano, R. Calvo-Palomino, N. Apostolakis, and G. Bovet, "A framework for wireless technology classification using crowdsensing platforms," in *Proc. IEEE Conf. Comput. Commun.*, 2023, pp. 1–10.

[23] H. Han, W. Li, Z. Feng, G. Fang, Y. Xu, and Y. Xu, "Proceed from known to unknown: Jamming pattern recognition under open-set setting," *IEEE wireless Commun. Lett.*, vol. 11, no. 4, pp. 693–697, Apr. 2022.

[24] M. Girmay, V. Maglogiannis, D. Naudts, M. Aslam, A. Shahid, and I. Moerman, "Technology recognition and traffic characterization for wireless technologies in ITS band," *Veh. Commun.*, vol. 39, Feb. 2023, Art. no. 100563.

[25] J. Fontaine et al., "Towards low-complexity wireless technology classification across multiple environments," *Ad Hoc Netw.*, vol. 91, Aug. 2019, Art. no. 101881.

[26] Y. Ren et al., "Deep clustering: A comprehensive survey," 2022, *arXiv:2210.04142*.

[27] N. Zhang, J. Shen, Y. Shi, and Y. Li, "CNN-Zero: A zero-shot learning framework for jamming identification," in *Proc. IEEE 22nd Int. Conf. Commun. Technol. (ICCT)*, 2022, pp. 1126–1131.

[28] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 132–149.

[29] G. Cerar, B. Bertalanič, M. Mohorčič, and C. Fortuna, "Learning to detect wireless spectrum occupancy using clustering approaches," in *Proc. 19th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, 2023, pp. 143–148.

[30] A. Elsebaay and H. H. Refai, "Wireless technology identification employing dynamic mode decomposition spectrum," in *Proc. Int. Telecommun. Conf. (ITC-Egypt)*, 2023, pp. 134–139.

[31] Q. Meng, H. Qian, Y. Liu, Y. Xu, Z. Shen, and L. Cui, "Unsupervised representation learning for time series: A review," 2023, *arXiv:2308.01578*.

[32] Q. Ma, J. Zheng, S. Li, and G. W. Cottrell, "Learning representations for time series clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[33] H. Zhou, J. Bai, Y. Wang, J. Ren, X. Yang, and L. Jiao, "Deep radio signal clustering with interpretability analysis based on saliency map," *Digit. Commun. Netw.*, to be published.

[34] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Proc. Int. Conf. Neural Inf. Process.*, 2017, pp. 373–382.

[35] L. Milosheski, G. Cerar, B. Bertalanič, C. Fortuna, and M. Mohorčič, "Self-supervised learning for clustering of wireless spectrum activity," *Comput. Commun.*, vol. 212, pp. 353–365, Dec. 2023.

[36] Z. Luo et al., "Spectrum sensing everywhere: Wide-band spectrum sensing with low-cost UWB nodes," *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 2112–2127, Jun. 2024.

[37] J. MacQueen et al., "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, 1967, pp. 281–297.

[38] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising Behavior of distance metrics in high dimensional space," in *Proc. 8th Int. Conf. Database Theory (ICDT)*, 2001, pp. 420–434.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[40] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 168–179, Feb. 2018.

[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[42] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[43] T. Gale, T. Šolc, R.-A. Moşoi, M. Mohorčič, and C. Fortuna, "Automatic detection of wireless transmissions," *IEEE Access*, vol. 8, pp. 24370–24384, 2020.

[44] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

[45] L. Hubert and P. Arabie, "Comparing partitions," *J. Classif.*, vol. 2, pp. 193–218, Dec. 1985.

[46] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proc. Joint Conf. Empir. Methods Nat. Lang. Process. Comput. Nat. Lang. Learn. (EMNLP-CoNLL)*, 2007, pp. 410–420.

[47] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987.

[48] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. pattern Anal. Mach. Intell.*, no. 2, pp. 224–227, Apr. 1979.

[49] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Statist.-Theory Methods*, vol. 3, no. 1, pp. 1–27, 1974.