A Dual Transfer Framework for Cooperative Spectrum Sensing in Cognitive Radio

Lusi Li, Rui Ning, Ziyu Wang, and Shuai Hao
Department of Computer Science, Old Dominion University, Norfolk, VA 23509, USA
Email: lusili@cs.odu.edu, rning@cs.odu.edu, zwang007@odu.edu, shao@cs.odu.edu

Abstract—**Introducing machine learning (ML) into cooperative spectrum sensing (CSS) in cognitive radio has yielded effective data-driven solutions for the spectrum shortage problem. Significant quantities of labeled data are required for ML-based CSS model training. However, collecting and annotating data repeatedly under dynamic spectrum environmental conditions is time-consuming, costly, and impractical. To this end, we propose a novel non-parametric dual transfer framework for CSS (DTCSS) to solve the poor generalization sensing performance caused by insufficient labeled data in the target environment with different wireless signals and propagation. DTCSS features a unique design that employs a two-stage learning approach. The offline training stage aims to transfer domain-level and class-level knowledge from the existing environment to the target environment and train a target detector. The objective of the online sensing phase is to utilize the trained detector to deduce the spectrum status of the target environment. DTCSS is robust and effective without hyperparameter tuning. Results from simulations indicate that DTCSS can achieve competitive sensing performance.**

Index Terms—**Cooperative spectrum sensing, machine learning, dual transfer learning, non-parametric training, robustness**

I. Introduction

The demand for scarce wireless spectrum has become urgent due to the widespread deployment of 5G and beyond networks, the rapid emergence of the Internet of Things, and the ever-increasing number of various wireless devices. Traditional static spectrum allocation only permits licensed primary users (PUs) to use the allocated spectrum bands, resulting in inefficient spectrum utilization. Cognitive radio (CR) is then proposed as one promising solution to the spectrum shortage. The core concept of CR is spectrum reuse, which permits secondary users (SUs) to periodically sense spectrum bands and access the idle bands without interfering with PUs. Spectrum sensing as a technology to identify the presence or absence of PUs by an individual SU is, therefore, essential for CR networks [1]. Nevertheless, the performance of SS is frequently diminished by fading channel conditions and other destructive effects. To boost the sensing quality, cooperative spectrum sensing (CSS) is employed in which a fusion center fuses the sensing results from multiple SUs to determine the status of PUs for subsequent spectrum access.

Numerous CSS techniques have been proposed [2]. Most conventional CSS methods concentrate on developing decision statistics to sense PU signals by analyzing the differences between signal and channel noise characteristics. Typical conventional CSS methods include the OR rule, AND rule, and CSU rule based on the energy detector. Nevertheless, the development of statistics requires prior knowledge of PU signals and the spectrum environment, which is not always available in practice due to dynamics, security, and privacy concerns. CSS has recently incorporated machine learning (ML) techniques to improve sensing performance in a data-driven manner due to its superior performance in extracting data features and identifying patterns. Support vector machine (SVM) and Kmeans are known to be more adaptable techniques. The fundamental concept is to train a mapping model from the training sensing data to the status of the PUs (presence or absence) and then make a decision based on the target sensing data. The model's success depends on a large quantity of labeled training data collected under dynamic spectrum environmental conditions and similar training and test data distributions. For instance, a model trained in one spectrum environment may not adapt well to another with different PU signal distributions. The difference between various environmental conditions is known as domain shift [3].

Intuitively, to make ML models operational and reduce domain shift, new batches of labeled training data must be recollected for the target spectrum environmental condition to retrain models from scratch. Unfortunately, two obstacles must be conquered. First, manually annotating data collected in a highly dynamic environment is time-consuming and labor-intensive. In addition, retraining models requires additional training time, resulting in a significant sensing delay. Consequently, it is impractical to repeatedly collect and annotate sensing data and then train a separate model for each environmental condition.

These obstacles severely limit the practical application of ML models. For CR networks, designing a blind, robust, and efficient CSS framework to avoid data annotation and model retraining has become an urgent matter. To this end, we propose to incorporate transfer learning into CSS. Transfer learning (TL) is a promising strategy for improving learning performance on an unlabeled target domain by leveraging knowledge from a well-labeled source environmental condition [4]. For instance, the authors of [5] designed a TL-based method for deep sensing in SS. To facilitate knowledge transfer, [6] proposed a TL-based method with multiple hyperparameters. Motivated by them, we propose a novel non-parametric dual transfer framework for CSS (DTCSS) that is inspired by the advantages of TL. The most significant contributions are highlighted below:

- To alleviate the sensing delay in DTCSS, we employ a two-stage learning strategy. Offline training aims to transfer both domain- and class-level knowledge to the target environment. The objective of the online sensing phase is to infer the spectrum status of the target environment.
- To alleviate the domain shift in terms of domain and class levels, we design a dual feature alignment component coupled with the offline training stage. The intra-class compactness is improved.
- DTCSS is a non-parametric framework that does not require tuning any hyperparameters. Compared to other CSS schemes, it can achieve comparable sensing performance on various transfer tasks.

The rest of this paper is organized as follows. Section II describes the system model. The proposed framework is presented in Section III. The performance results and analysis can be found in Section IV. Section V gives the conclusions.

II. System Model

We consider a cooperative CR network, where M SUs and Z PUs are deployed in a square area. The SUs carry out spectrum sensing to sense spectrum holes by distinguishing the activity status (absence/presence) of the PUs over a spectrum channel of interest. Thus the sensing problem can be formulated as a binary hypothesis test problem. Specifically, each SU utilizes an energy detector to sense the channel and then reports the obtained local energy level to a fusion center, which would fuse the energy levels from the M SUs and predict the channel status. The channel has a bandwidth b. In a sensing duration τ, the m-th SU collects $b\tau$ signal samples of the channel as the local sensing results. Let $S_m[k]$ be the k-th signal sample captured by the m-th SU, and we have the test problem as follows

$$S_m[k] = \begin{cases} \omega_m[k] & H_0 \\ \sum_{z=1}^{Z} a_z g_{z,m} P_z[k] + \omega_m[k] & H_1 \end{cases} \tag{1}$$

where the hypothesis H_0 represents the absence of all the PUs' signals which means the channel is idle; the hypothesis H_1 indicates the presence of at least one PU's signal which means the channel is busy; $m = 1, 2, ..., M$; $k = 1, 2, ..., b\tau$; $z = 1, 2, ..., Z$; a_z indicates the state indicator of the z-th PU, where $a_z = 0$ indicates the absence of the z-th PU's signal and $a_z = 1$ indicates the presence of the z-th PU's signal; $g_{z,m}$ is the channel gain from the z-th PU to the m-th SU; $P_z[k]$ indicates the k-th signal sample transmitted by the z-th PU; $\omega_m[k]$ indicates the local noise. Then the m-th SU in the i-th sensing duration creates a local energy estimate $x_{i,m}$ as follows:

$$x_{i,m} = \sum_{k=1}^{b\tau} |S_m[k]|^2 \tag{2}$$

Hence we have an energy sample from M SUs in the i-th sensing duration as $x_i = (x_{i,1}, x_{i,2}, ..., x_{i,M})^T$ fused at the fusion center. The CSS problem at the fusion center can be formulated as a binary classification problem. Next, the fusion center begins to train a target detector and determine the presence or the absence of the PUs based on the energy vectors.

III. Proposed Method

The notations in our proposed method are denoted as follows. We are given a labeled source domain $\Theta_s = (X^{(s)}, Y^{(s)}) = \{(x_i^{(s)}, y_i^{(s)})\}_{i=1}^{N_s}$, where $x_i^{(s)} \in \mathbb{R}^M$ is the i-th energy sample of total N_s samples collected by M SUs in the source spectrum environmental condition. We are also given an unlabeled target domain $\Theta_t = X^{(t)} = \{x_j^{(t)}\}_{j=1}^{N_t}$ for training, where $x_j^{(t)} \in \mathbb{R}^M$ is the j-th target energy sample of total N_t samples collected by M SUs in the target spectrum environmental condition. Assume Θ_s and Θ_t are collected under different spectrum environment conditions in the same considered CR network. $Q(x^{(s)})$ and $Q(x^{(t)})$ denote the marginal distributions of Θ_s and Θ_t. $Q(y^{(s)}|x^{(s)})$ and $Q(y^{(t)}|x^{(t)})$ indicate the conditional distributions of Θ_s and Θ_t. The marginal and conditional distributions can be measured by the differences between domains and that between classes, respectively. Under different spectrum environment conditions, we, without loss of generality, posit that Θ_s and Θ_t have different joint distributions, i.e., $Q(x^{(s)}) \neq Q(x^{(t)})$ and $Q(y^{(s)}|x^{(s)}) \neq Q(y^{(t)}|x^{(t)})$. We aim to transfer the knowledge from Θ_s to Θ_t and then make inferences for a new target samples $\chi^{(t)}$. Fig. 2 shows the procedure of our proposed method. We will introduce the designed dual feature alignment components in terms of domain and class levels. Then we will describe the two-stage component.

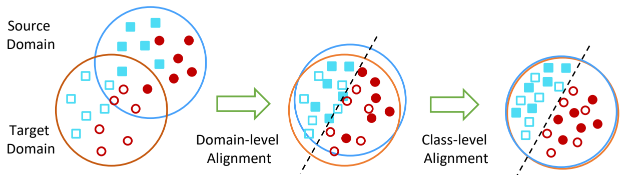


Fig. 1: The procedure of our proposed DTCSS method

A. Feature Alignment

Due to the differences across spectrum environmental conditions, a detector trained by source domain data would have an unsatisfactory sensing performance for target domain data. In this paper, we aim to dual transfer both domain-level and class-level knowledge by aligning the data marginal and conditional distributions. To conduct distribution alignments, there have been many proposed distribution measurements. Among them, correlation alignment (CORAL) as a simple yet efficient strategy could measure the distribution distance across domains in the original feature space by aligning the second-order statistics (i.e., co-variance matrices) [7]. We use CORAL strategy to perform the alignments.

The domain-level alignment aims to reduce marginal distribution discrepancy by performing target re-correlation at the domain level. That is to add the correlation of the target data to the source data in the feature space. The transformed source domain features at the domain level $G^{(d)}$ can be found as follows:

$$G^{(d)} = X^{(s)} * C_s^{-1/2} * C_t^{1/2} \tag{3}$$

where C_s and C_t indicate the domain-level co-variance matrices of source and target data, respectively. They can be calculated by

$$C_s = \frac{1}{N_s - 1}[(X^{(s)})^T X^{(s)} - \frac{1}{N_s}(\mathbf{1}^T X^{(s)})^T(\mathbf{1}^T X^{(s)})], \tag{4}$$

$$C_t = \frac{1}{N_t - 1}[(X^{(t)})^T X^{(t)} - \frac{1}{N_t}(\mathbf{1}^T X^{(t)})^T(\mathbf{1}^T X^{(t)})]. \tag{5}$$

where $\mathbf{1}$ is a column vector with all ones.

As we know that the marginal distribution adaptation cannot guarantee the conditional distribution adaptation for each class. We can further have the class-level alignment to add the correlation of the target data to the source features at the class level. The transformed source class features $G^{(c)}$ at the class level for class c are given by:

$$G^{(c)} = G_c^{(d)} * C_{s,c}^{-1/2} * \hat{C}_{t,c}^{1/2} \tag{6}$$

where $c = \{0, 1\}$ in which $c = 0$ means the channel is idle and $c = 1$ means the channel is busy; $G_c^{(d)}$ indicates the transformed source data samples by Eq. (3) in class c; $C_{s,c}^{-1/2}$ is the co-variance matrix of class c of the source domain; $\hat{C}_{t,c}^{1/2}$ is the estimated co-variance matrix with the pseudo label c of the target domain. Similarly, they can be computed as

$$C_{s,c} = \frac{1}{N_{s,c} - 1}[(G_c^{(d)})^T G_c^{(d)} - \frac{1}{N_{s,c}}(\mathbf{1}^T G_c^{(d)})^T(\mathbf{1}^T G_c^{(d)})] \tag{7}$$

$$\hat{C}_{t,c} = \frac{1}{\hat{N}_{t,c} - 1}[(\hat{X}_c^{(t)})^T \hat{X}_c^{(t)} - \frac{1}{\hat{N}_{t,c}}(\mathbf{1}^T \hat{X}_c^{(t)})^T(\mathbf{1}^T \hat{X}_c^{(t)})] \tag{8}$$

where $N_{s,c}$ is the number of the source samples in class c; $\hat{X}_c^{(t)}$ and $\hat{N}_{s,c}$ are the estimated target samples and the corresponding amount in class c, respectively. Since we do not know the target data labels, we would obtain the pseudo labels at the offline training stage.

B. Offline Training

The offline training stage aims to transfer the knowledge from source domain to target domain by performing both domain-level and class-level alignments.

Specifically, we can use $X^{(s)}$ and $X^{(t)}$ to perform the domain-level alignment and the transformed source features $G^{(d)}$ can be learned by Eq. (3). To obtain the predicted pseudo labels for class-level alignment, given $G^{(d)}$ and $X^{(t)}$, we firstly estimate each source class center $g_d^{(c)}$ by

$$g_d^{(c)} = \frac{1}{N_{s,c}} \sum_{j=1}^{N_{s,c}} G_j^{(d)}. \tag{9}$$

where $G_j^{(d)}$ is j-th source feature vector of $G^{(d)}$. Then we measure the distance Ω_{ic} between each target sample $x_i^{(t)}$ and each source class center $g_d^{(c)}$ as follows:

$$\Omega_{ic} = ||x_i^{(t)} - g_d^{(c)}||_f^2 \tag{10}$$

where $||\cdot||_f$ is the Frobenius norm.

Given the distance matrix Ω, we introduce an indicator matrix F to denote an initial classifier. We let $F \in \mathbb{R}^{N_t \times 2}$, where 2 means we have 2 classes; its each entry is $F_{ic} \in [0, 1]$ denoting the probability of $x_i^{(t)}$ belonging to class c; $i = \{1, 2, ..., N_t\}$. Inspired by the Softmax layer of a neural network, $x_i^{(t)}$ would have the class label $\hat{y}^{(t)}$ by

$$\hat{y}_i^{(t)} = \text{argmax}_c(F_{i0}, F_{i1}) \tag{11}$$

where $F_{i0} + F_{i1} = 1$. Before that, the initial classifier F can be learned by minimizing the objective function

$$\min_F \sum_{c=0}^{1} \sum_{i=1}^{N_t} F_{ic}\Omega_{ic} \quad s.t.\ 0 \le F_{ic} \le 1,\ F_{i0} + F_{i1} = 1 \tag{12}$$

We can solve Eq. (12) by a linear programming strategy and generate the pseudo target labels $\hat{y}^{(t)}$ with Eq. (11).

After the domain-level alignment and the initial classifier learning, we align the marginal distributions of the two domains and obtain the pseudo target labels. The next step is to perform the class-level alignment and learn the target classifier. The co-variance matrices for each class of the source and target domains can be computed by Eq. (7) and Eq. (8), respectively. Then we have the class-level transformed source features $G^{(c)}$ with $c = \{0, 1\}$ by Eq. (6). As such, the source class center can be updated using $G^{(c)}$ as follows:

$$g_c^{(c)} = \frac{1}{N_{s,c}} \sum_{j=1}^{N_{s,c}} G_j^{(c)}. \tag{13}$$

C. Online Sensing

After the offline training stage, we have the updated source class centers in the aligned feature space. For the online sensing stage, given a new target energy sample $\chi^{(t)}$, we firstly compute the distance matrix Ω^*

$$\Omega_{1c}^* = ||\chi^{(t)} - g_c^{(c)}||_f^2 \tag{14}$$

Similarly, we can have the target classifier F^* and the target label $\Upsilon^{(t)}$ given by

$$\min_{F^*} \sum_{c=0}^{1} F_{1c}^*\Omega_{1c}^* \quad s.t.\ 0 \le F_{1c}^* \le 1,\ F_{10}^* + F_{11}^* = 1 \tag{15}$$

$$\Upsilon^{(t)} = \text{argmax}_c(F_{10}^*, F_{11}^*) \tag{16}$$

Algorithm 1 summarizes the proposed two-stage learning framework. We are given labeled source data from the past or existing environmental condition and unlabeled target data from the target or new environmental condition for offline training. The offline training stage first performs domain-level alignment to transfer the domain-level knowledge from the source domain to the target domain and then obtain the pseudo target labels. Then it performs class-level alignment to transfer the class-level knowledge and obtain the updated source class centers. At the online sensing stage, we are given a new energy sample collected at the fusion center and predict the channel status.

Algorithm 1 DTCSS Framework

```
Input: Labeled source training data $(X^{(s)}, Y^{(s)})$, target training data $X^{(t)}$, and target testing data $\chi^{(t)}$.
Output: The labels of the target testing data $\Upsilon^{(t)}$
   Stage 1: offline training
1: Perform domain-level alignment by Eq. (3) to have $G^{(d)}$.
2: Calculate the source class centers by Eq. (9) and the distance matrix $\Omega$ by Eq. (10).
3: Solve Eq. (12) to learn the indicate matrix $F$ and compute the pseudo target labels $\hat{y}^{(t)}$ by Eq. (11).
4: Perform class-level alignment by Eq. (6) to have $G^{(c)}$.
5: Update the source class centers by Eq. (13).
   Stage 2: online sensing
6: Calculate distance matrix $\Omega^*$ by Eq. (14) with testing target data $\chi^{(t)}$.
7: Solve Eq. (15) to learn the indicate matrix $F^*$ and compute the target labels $\Upsilon^{(t)}$ by Eq. (16).
8: return  Target testing labels $\Upsilon^{(t)}$.
```

D. Complexity Analysis

The computational complexity of our proposed DTCSS framework can be found as follows. For the offline training stage, the domain-level alignment takes at most $O(N_s^3)$, and the class-level alignment takes at most $O(C^3N_s^3)$, where $C = 2$ indicating the channel is idle or busy. For the online sensing stage, it takes $O(N_t^2)$ given that Eq. (15) is a linear programming problem. Thus the overall DTCSS framework takes about $O(N_s^3 + N_t^2)$.

IV. Experiment

A. Simulation Protocol

It has been challenging to improve sensing robustness under dynamic spectrum environmental conditions. Given labeled source signal data under one environmental condition and unlabeled target signal data under another environmental condition, several experiments are conducted to evaluate the performance of our proposed DTCSS framework on the target domain. In the experiments, for each domain, we produce one type of digitally modulated signal (such as Gaussian, BPSK, QPSK, PAM4, and 16-PSK) under additive white Gaussian noise (AWGN) channels as positive samples and the noise as negative samples. The simulation parameters in terms of spectrum environmental conditions can be found below: the equal presence probability of the PUs' signals, the noise power spectral density [−153, −174] dBm, the transmission power 200 mW, the unity shadow fading and multi-path fading components, the path-loss exponent $[4, 5]$, the bandwidth 5 MHz, the sensing time interval $10\mu s$ for each observation, and the number of samples in a sensing time interval $b\tau = 50$.

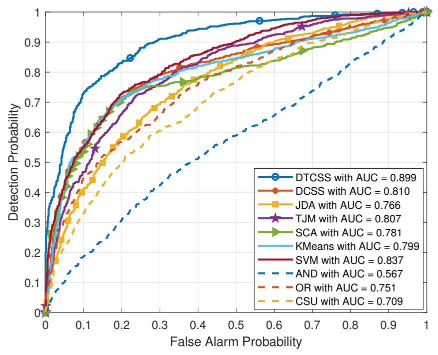
The baselines are compared with DTCSS: three TL-based methods (joint distribution adaptation (JDA) [8], scatter component analysis (SCA) [9], and transfer joint matching (TJM) [10]), two ML-based methods (KMeans, SVM), and three conventional CSS methods (OR rule, AND rule, CSU rule). Therein, JDA aligns both marginal and conditional distributions; TJM aligns marginal distribution with source sample selection; SCA aligns scatters in subspace. For JDA, TJM, and SCA, their hyper-parameters are tuned to obtain the best results. Unlike them, our proposed method has no hyper-parameters to tune and is robust to dynamic environments. Note that we also add a baseline DCSS, the variant of DTCSS without class-level alignment, to verify the importance of class-level alignment. All the experiments are implemented in Matlab. For evaluation metrics, we use the area under the curve (AUC) values and the receiver operating characteristic (ROC) curves to measure the sensing performance of all the methods. The ROC curve is a curve of the probability of false alarm and probability of detection. The method can achieve a larger AUC value, indicating it performs better.

B. Experimental Results

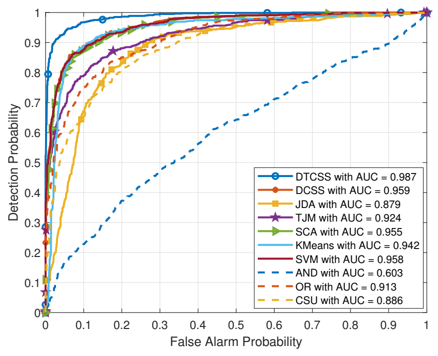
The first CR network scenario considers one PU with coordinates (0, 0) and three SUs, which are randomly distributed in a 2000×2000 area. The Monte-Carlo simulations are used to generate $2,000$ observations for the source domain, target training domain, and target testing data, respectively. We have two transfer tasks: Gaussian $\rightarrow$ 16-PSK and 16-PSK $\rightarrow$ Gaussian.

For the first transfer task, the source domain has the observed Gaussian signal as the positive samples; the target domain has the observed 16-PSK signal as the positive samples. For the second transfer task, we interchange the two domains of the first task. The sensing performance by different baselines and DTCSS on the two transfer tasks are shown in Fig. 2 (a) and (b), respectively. From the ROC curves and the corresponding AUC values, our proposed DTCSS method outperforms the other baselines and enhances the robustness of the two tasks. Roughly, the TL-based and ML-based methods have better performance than the conventional CSS approaches, which verifies the superiority of data-driven schemes. Specifically, DTCSS can improve the robustness by alleviating the domain shift at the domain and class levels through the designed two-stage learning strategy since it performs better than DCSS, which is the DTCSS without the second-stage learning. Moreover, DTCSS performs better than the other TL-based methods, indicating that the effective knowledge is transferred to the target domain. Additionally, for the conventional CSS approaches, OR rule and CSU rule have comparable performance and perform better than AND rule. Note that the source domain with the 16-PSK signal shown in Fig.2 (b) makes better guidance than that with the Gaussian signal shown in Fig.2 (a) in the knowledge transfer.

To explore the effects of the number of SUs on sensing performance, we conduct experiments with different numbers of SUs on the transfer tasks: Gaussian $\rightarrow$ 16-PSK. We have



(a) Sensing performance on the transfer task: Gaussian $\rightarrow$ 16-PSK



(b) Sensing performance on the transfer task: 16-PSK $\rightarrow$ Gaussian

Fig. 2: Sensing performance (ROC curves and AUC values) by different methods with three SUs for two transfer tasks.

one fixed PU and 10 SUs, which are randomly distributed in the same area. We also generate 2000 observations for each domain in which each observation is a 10-dimensional energy vector. We select the first M estimated energy levels as a vector for the case with different numbers of SUs, where M can be 2, 3, 5, 8, and 10. Therein, the sensing results for M = 3 are also shown in Fig. 2 (a). Roughly, from Table II, the averaged AUC values over five runs become larger as more SUs, demonstrating the CSS strategy's benefits.

TABLE I: AUC values with different numbers of SUs

# of SUs	2	3	5	8	10	Average
DTCSS	**0.786**	**0.899**	**0.919**	**0.942**	**0.963**	**0.902**
DCSS	0.735	0.810	0.849	0.892	0.912	0.840
JDA	0.685	0.766	0.841	0.854	0.865	0.802
TJM	0.718	0.807	0.836	0.862	0.902	0.825
SCA	0.748	0.781	0.883	0.891	0.923	0.845
KMeans	0.716	0.799	0.856	0.863	0.882	0.823
SVM	0.739	0.837	0.855	0.879	0.902	0.842
AND	0.523	0.567	0.572	0.589	0.554	0.561
OR	0.637	0.751	0.828	0.834	0.896	0.789
CSU	0.634	0.709	0.723	0.729	0.882	0.735

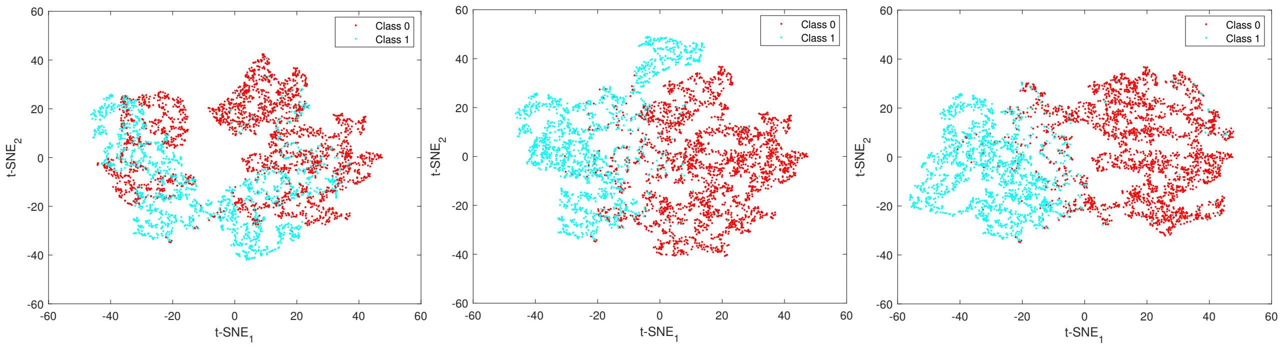
To verify the effects of the number of labeled source training samples, we conduct experiments by TL-based and ML-based methods on the transfer task: Gaussian $\rightarrow$ 16-PSK with different training sizes while fixing the target domain under the first scenario. In this case, the traditional CSS methods (OR rule, AND rule, and CSU rule) have the same performance as before shown in Fig.2 (a) since they only work on the target domain without the need for training samples. We generate 3,000 samples in total and randomly select 500, 1000, 2,000, 2,500, and 3,000 from them as the labeled source training samples. Table III shows the best AUC values in terms of different labeled training sizes. It can be observed that the obtained AUC values increase as the size of source training data increases. DTCSS has superior performance even given a small amount of labeled source data since it can learn domain-invariant features and adapt the source features to the target domain.

TABLE II: AUC values with different numbers of labeled source training samples (# of samples for short)

# of samples	500	1000	2000	2500	3000	Average
DTCSS	**0.827**	**0.885**	**0.899**	**0.914**	**0.931**	**0.891**
DCSS	0.783	0.795	0.810	0.846	0.835	0.814
JDA	0.727	0.745	0.766	0.854	0.867	0.792
TJM	0.732	0.760	0.807	0.852	0.874	0.805
SCA	0.749	0.762	0.781	0.884	0.906	0.816
KMeans	0.697	0.761	0.799	0.847	0.858	0.792
SVM	0.675	0.715	0.837	0.851	0.864	0.788

To verify the effectiveness of the two-stage learning strategy of DTCSS, as an example, we use the t-SNE technique [11] to visualize the source and target domain data learned at different stages on the transfer task of 16-PSK $\rightarrow$ Gaussian. In Fig. 2, the red dots indicate class 0, and the blue dots represent class 1 for the source and target domains. The dots with the same color belong to the same class for the two domains and are expected to be aligned well. Fig. 2 (a) shows the visualization of the original source and target data. Note that the dots with different colors are mixed together, indicating that the domain shift exists between the source and target domains. Fig. 2 (b) shows the visualization of the initially aligned domain data by the variant DCSS. It can be observed that the dots with the same color are roughly aligned. Fig. 2 (c) shows the visualization of the aligned domain data by our proposed DTCSS method. We can notice that both the separation across different classes and the compactness within the same class are improved. Fig. 2 verifies that DTCSS aligns the two domains well and simultaneously ensures the intra-class compactness as well as inter-class separation.

The second CR network scenario considers M = 5 SUs and Z = 2 PUs with coordinates (0,0) and (1000, 500), which have equal presence probabilities and are activated independently from each other. We fix $N_s = N_t = 2000$. To evaluate the model robustness, the sensing performance by the TL-based methods is tested on the different transfer tasks. Table IV shows the averaged AUC values by different approaches over five runs. It can be observed that different AUC values are obtained by the approaches when the source domain and the target domain are interchanged. That's because different



(a) Original domain data before training (b) Initially aligned data after training by DCSS (c) Aligned domain data after training by DTCSS

Fig. 3: t-SNE visualization of the source and target domain data at different stages on the transfer task: 16-PSK → Gaussian. For both domains, the red dots in Class 0 means the channel is idle and the blue dots in Class 1 means the channel is busy. The dots with the same color are expected to be aligned.

TABLE III: AUC values on different transfer tasks

Methods	DTCSS	DCSS	JDA	TJM	SCA
BPSK → Gaussian	**0.959**	0.895	0.836	0.843	0.903
Gaussian → BPSK	**0.885**	0.803	0.756	0.770	0.798
QPSK → PAM4	**0.973**	0.922	0.864	0.838	0.925
PAM4 → QPSK	**0.920**	0.854	0.823	0.806	0.891
BPSK → QPSK	**0.966**	0.908	0.895	0.917	0.925
QPSK → BPSK	**0.987**	0.917	0.908	0.923	0.938
QPSK → Gaussian	**0.941**	0.875	0.817	0.825	0.899
Gaussian → QPSK	**0.906**	0.834	0.709	0.783	0.815
Averaged AUCs	**0.942**	0.876	0.826	0.838	0.887
Offline training time	0.541s	**0.236s**	6.679s	11.39s	13.42s
Online testing time	**0.086s**	**0.086s**	2.873s	2.792s	4.502s
Total running time	0.627s	**0.322s**	9.552s	14.18s	17.92s

types of signals have different guidance in the process of knowledge transfer. When the two domains have more similar distributions, such as BPSK and QPSK signals as the positive samples, the approaches can achieve comparable performance on the interchanged tasks. In addition, we also assess the computational complexity. The averaged offline training time, online testing time, and total running time (seconds) by the TL-based methods over five runs on the eight tasks are shown in the last three rows of Table IV. The total running time by DTCSS is roughly two times that of its variant DCSS since it is a dual transfer learning method. DTCSS can take first place in terms of the testing time at the online sensing stage and second place in terms of the training time and total running time. It outperforms the baselines.

V. Conclusion

In this paper, a novel non-parametric dual transfer-based CSS (DTCSS) framework is proposed. DTCSS utilized a two-stage learning strategy to perform the knowledge transfer from labeled source environmental condition to unlabeled target condition, where the PU signal modulation and propagation differ. DTCSS aligns the two domains in terms of domain and class levels at the offline training stage and infers the channel status at the online sensing stage. By incorporating transfer learning into the CSS scheme, DTCSS improves efficiency and robustness. The experimental results demonstrated the effectiveness of our method. In the future, we will extend it to determine the modulation of PU signals.

VI. Acknowledgment

This work was supported in part by the National Science Foundation under Grants CNS-2245918, CNS-2153358, and DUE-2153358, DoD Center of Excellence in AI and Machine Learning (CoE-AIML) under Contract Number W911NF-20-2-0277, the Commonwealth Cyber Initiative, and InterDigital Communications, Inc.

References

[1] Z. Zhang, G. Zhu, and S. Cui, "Low-latency cooperative spectrum sensing via truncated vertical federated learning," in *2022 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2022, pp. 1858–1863.

[2] L. Li, H. Jiang, and H. He, "Imbalanced learning for cooperative spectrum sensing in cognitive radio networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

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

[4] K. Yang, J. Lu, W. Wan, G. Zhang, and L. Hou, "Transfer learning based on sparse gaussian process for regression," *Information Sciences*, vol. 605, pp. 286–300, 2022.

[5] Q. Peng, A. Gilman, N. Vasconcelos, P. C. Cosman, and L. B. Milstein, "Robust deep sensing through transfer learning in cognitive radio," *IEEE Wireless Communications Letters*, vol. 9, no. 1, pp. 38–41, 2019.

[6] L. Li, H. Jiang, and H. He, "Deep transfer cooperative sensing in cognitive radio," *IEEE Wireless Communications Letters*, pp. 1–1, 2021.

[7] B. Sun, J. Feng, and K. Saenko, "Correlation alignment for unsupervised domain adaptation," in *Domain Adaptation in Computer Vision Applications*. Springer, 2017, pp. 153–171.

[8] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2200–2207.

[9] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1414–1430, 2016.

[10] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer joint matching for unsupervised domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1410–1417.

[11] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.