An explainable multi-source unsupervised domain adaptation framework using contrastive learning and adaptive clustering for remote sensing scene classification

Binu Jose A, Pranesh Das, Ebrahim Ghaderpour, Paolo Mazzanti

Department of Computer Science and Engineering,
National Institute of Technology Calicut, Kerala, India
&
Department of Earth Sciences,
Sapienza University of Rome, Piazzale Aldo Moro 5, 00185, Rome, Italy

25 October, 2025

DARES'25 colocated with ECAI 2025, Bologna, Italy

Presentation Overview

Domain Adaptation

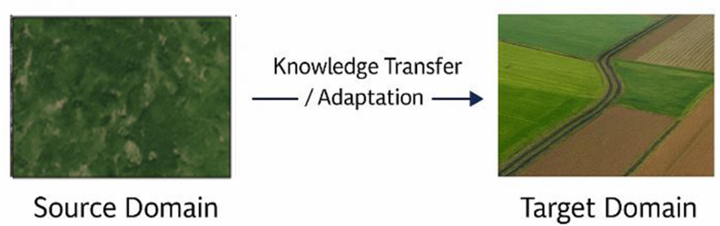


Figure 2: Domain Adaptation

- Source: Labeled satellite images from dry-season(with classes farmland, forest, industrial, parking, residential, river).
- Target: Unlabeled another satellite images during monsoon.
- Aim is to predict the label of the target image.

Types of domain adaptation

Table 1: Types of domain adaptation

Types of adaptation	**Source**	**Target**
Supervised domain adaptation [1]	Labelled	mostly labelled
Semi-Supervised domain adaptation [2]	Labelled	mostly unlabelled
Unsupervised domain adaptation [3]	Labelled	fully unlabelled

Unsupervised Domain adaptation (UDA)

- Let $\mathcal{X}$ be the input space and $\mathcal{Y} = \{1, \ldots, C\}$ the label space. *Source has labels, target is unlabeled:*

$$\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s} \sim P_s(x, y), \qquad \mathcal{D}_t = \{x_j^t\}_{j=1}^{n_t} \sim P_t(x).$$

- We learn $h = f_\phi \circ g_\theta : \mathcal{X} \to \Delta^{C-1}$ (feature extractor g_θ and classifier f_ϕ) with loss ℓ (e.g., cross-entropy).
- Source and target loss

$$R_s(h) = \mathbb{E}_{(x,y)\sim P_s}[\ell(h(x), y)], \qquad R_t(h) = \mathbb{E}_{(x,y)\sim P_t}[\ell(h(x), y)].$$

- Goal: minimize $R_t(h)$ using $\mathcal{D}_s$ and $\mathcal{D}_t$.

Pseudo-Labeling for UDA

Pseudo-Label

- A pseudo-label is a label we assign to an unlabeled sample using a model, so we can train with a supervised loss on that sample.

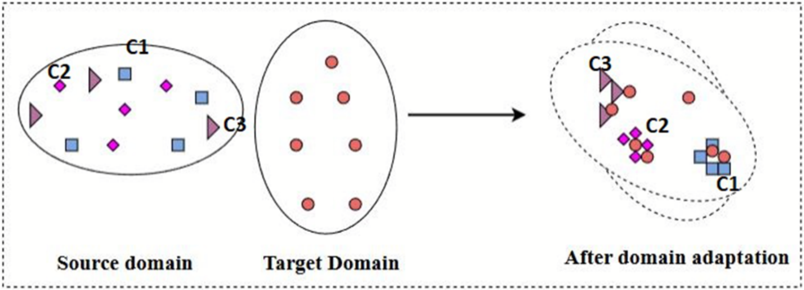


Figure 3: Illustration of Pseudo-labeling

Why contrastive learning

- Contrastive learning (CL) learns features that are both domain-invariant and class-discriminative from largely unlabeled data.
- This is essential for multi-source UDA

Multi-source UDA

- Real targets differ in many ways (sensor, season, resolution, geography).
- Data come from multiple heterogeneous sources makes the real deployment of domain adaptation.
- Both source and target share same set of labels.

Why density based clustering

- The effectiveness of UDA depends on the quality of pseudo-labels assigned to target data.
- To obtain reliable pseudo-labels, clustering algorithms groups the similar target features and use the assignments
- Density-based clustering discovers arbitrarily shaped clusters and marks low-density points as noise.
- Incremental density-based clustering [4] is an enhanced approach that updates the clustering results incrementally as new data points arrive.
- It enables efficient updates to cluster structures without requiring a complete re-clustering process.

Literature Survey

Table 2: Representative multi-source unsupervised domain adaptation (MS-UDA) methods.

Method	Year	Source $\rightarrow$ Target dataset	Feature Extractor	Main Contribution	Limitation
M^3SDA [5]	2019	DomainNet / Office-Home	ResNet-50	Aligns higher-order *moments* across multiple sources and target; introduced the large-scale DomainNet benchmark for MS-UDA.	Global (class-agnostic) alignment can underperform on fine-grained classes; requires source data; sensitive to class imbalance and negative transfer.
MFSAN [6]	2019	Office-31 / ImageCLEF-DA / Digit-Five	ResNet-50 / CNN	Two-stage scheme: (i) domain-specific distribution alignment per source–target pair and (ii) *classifier* alignment with domain-specific heads.	Many source-specific branches/head losses increase complexity; fusion thresholds/hyperparameters sensitive; assumes closed-set label space.
LtC-MSDA [7]	2020	DomainNet / Office-Home / Office-31	ResNet-50	Builds a prototype *graph* across sources; *Relation Alignment Loss* enforces cross-domain relational consistency for knowledge aggregation.	Prototype purity and memory bank maintenance are non-trivial; tuning of graph/temperature terms; limited open-/universal-set handling.

Table 2: Representative multi-source unsupervised domain adaptation (MS-UDA) methods.

Method	Year	Source → Target dataset	Feature Extractor	Main Contribution	Limitation
T-SVDNet [8]	2021	Office-Home / DomainNet	ResNet-50	Exploits high-order *prototypical correlations*; imposes tensor low-rank (T-SVD/TLR) constraints plus uncertainty-aware source weighting.	Tensor ops add compute/memory; rank/weighting hyperparameters sensitive; assumes clusterable class structure; requires source data at adapt time.
PTMDA [9]	2022	Office-Home / DomainNet (typ.)	ResNet-50	Builds *pseudo target domains* via group-specific adversarial subspaces with metric constraints; adds matching normalization to stabilize alignment.	Adversarial + metric objectives increase training cost; subspace design may be dataset-specific; relies on pseudo-label quality.
MCC-DA [10]	2023	Digit-Five / Office-31 / DomainNet	ResNet-50	*Decentralized* MS-UDA: collaborative *contrastive* alignment among per-domain models without sharing raw data; periodic model aggregation.	Requires model exchange/synchronization; robustness depends on contrastive partner selection; more training rounds than centralized baselines.

Table 2: Representative multi-source unsupervised domain adaptation (MS-UDA) methods.

Method	Year	Source $\rightarrow$ Target dataset	Feature Extractor	Main Contribution	Limitation
RRL [11]	2023	DomainNet / Office-Home / Digits	ResNet-50	*Riemannian* representation learning: minimizes average empirical *Hellinger* distance with theoretical bounds on MS-UDA risk.	Computing Riemannian distances adds overhead; metric choices affect stability; closed-set assumption; needs access to sources.
SUMDA [12]	2024	DomainNet / Office-Home (reported)	ResNet-50 (typ.)	Cross-*source* alignment strategy that leverages inter-source complementarities (e.g., uncertainty-/consistency-aware weighting) for robust MS-UDA.	Details depend on implementation; still sensitive to noisy pseudo labels and inter-source imbalance; please refine to match your exact paper.
Hy-MSDA [13]	2024	Remote sensing (e.g., AID / NWPU)	Hybrid CNN + ViT	Hybrid backbone with *consistency learning* and *dynamic source weighting* tailored to multi-source scene classification.	Transformer components increase compute; remote-sensing-specific tuning; generalization beyond RS benchmarks to be established.

Research Gap

- Domain invariance:Pulls together multiple views of the same underlying scene, reducing texture/season/sensor bias.
- Label noise in pseudo-label: unreliable clusters-guided pseudo-labels degrade training.
- Rather than completely rejecting the uncertain clusters, class-aware pseudo labels can be generated.
- Interpretability: Explanations of what regions/classes aligned are not interpreted

Problem Statement

- Design a multi-source UDA framework for scene classification using adaptive density based clustering and class aware pseudo label refinement by considering uncertain clusters

Objectives

- Class-aware refinement of pseudo labels by top-k selection per class based on pseudo-label confidence
- Provide explainability techniques such as Grad-CAM/Attention Rollout maps for prediction of target sample.

Problem Formulation

- Let there be M labeled source domains $\mathcal{S}_1, \ldots, \mathcal{S}_M$ and one unlabeled target domain $\mathcal{T}$. Each source $\mathcal{S}_m$ is characterized by a joint distribution $P_m(X, Y)$ over inputs $X \in \mathcal{X}_m$ and labels $Y \in \mathcal{Y}_m$, from which we observe a labeled sample $\mathcal{D}_m = \{(x_i^{(m)}, y_i^{(m)})\}_{i=1}^{n_m}$.
- The target domain $\mathcal{T}$ has a (generally different) joint distribution $P_T(X, Y)$ over a shared set of classes k over m domains such that $C_{k1} = C_{k2} = ... = C_{km} = C_t$, from which we observe an unlabeled sample $\mathcal{D}_T = \{x_j^{(T)}\}_{j=1}^{n_T}$
- The objective of multi-source UDA is to mitigate this distribution shift and train a model that can accurately predict the label ${y_j}^t$ for any target sample ${x_j}^t$.

Framework

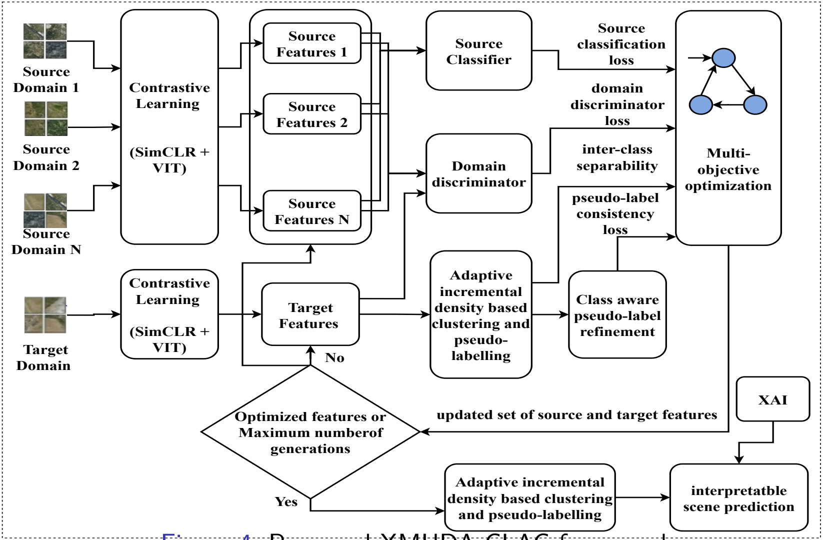


Figure 4: Proposed XMUDA-CLAC framework

Adaptive Incremental Clustering — Part I/IV

Algorithm 1: Adaptive incremental cluster formation with dynamic density estimation and NN-based merging (Part I/IV)

Target feature set $F_t = \{f_1, \ldots, f_n\}$; initial clusters $\mathcal{C}$; scaling factor α_1; adaptive range constants n_1, n_2, n_3, n_4 Updated cluster list $\mathcal{C}_{\text{updated}}$ `// Precompute global statistics`

Compute pairwise distance matrix D across F_t;

Compute global distance mean $T = \frac{1}{n(n-1)} \sum_{i \neq j} D_{ij}$;

Adaptive Incremental Clustering — Part II/IV

Algorithm 1: (Part II/IV)

foreach $f_{new} \in F_t$ **do**

    `// Estimate Local Distance Characteristics`

    Let $S = \{ D(f_{\text{new}}, f_j) \mid f_j \in F_t \}$;

    if $\mathrm{mean}(S) \leq T$ **then**

        Select $k \sim \mathrm{Uniform}(n_1, n_2)$;

    else

        Select $k \sim \mathrm{Uniform}(n_3, n_4)$;

    Sort S in ascending order; set $\epsilon = S[k]$;

    `// Infer Local Density`

    $N_\epsilon = \{ f_j \in F_t \mid D(f_{\text{new}}, f_j) \leq \epsilon \}$;

    Compute local density $\rho = \dfrac{|N_\epsilon|}{\epsilon}$;

    Compute adaptive threshold $\mathrm{MinPts} = \alpha_1 \cdot \rho$;

Adaptive Incremental Clustering — Part III/IV

Algorithm 1: (Part III/IV)

foreach $f_{new} \in F_t$ *(cont.)* **do**
    `// Decision: Assign or Evaluate`
    if $|N_\epsilon| \geq \mathrm{MinPts}$ **then**
        Identify intersecting clusters $\mathcal{C}_{\text{near}} = \{\, C_i \in \mathcal{C} \mid N_\epsilon \cap C_i \neq \emptyset \,\}$;
        if $|\mathcal{C}_{near}| = 0$ **then**
            Create new cluster $C_{\text{new}} = \{f_{\text{new}}\}$ and add to $\mathcal{C}$;
        else if $|\mathcal{C}_{near}| = 1$ **then**
            Append f_{new} to the matched cluster;
        else
            foreach *pair* $(C_a, C_b) \subseteq \mathcal{C}_{near}$ **do**
                Extract features: proximity, compactness, cross-similarity;
                Compute merge_score $\leftarrow \mathrm{Net}_1(\cdot)$, $\theta \leftarrow \mathrm{Net}_2(\cdot)$;
                if $merge\_score \geq \theta$ **then**
                    Merge $C_a \cup C_b$ and add f_{new};

The cluster proximity ($\grave{S}_p$), density ($\grave{S}_d$), and feature similarity ($\grave{S}_f$) are defined using Eq. (1), (2) and (3) respectively:

$$\grave{S}_p = 1 - \frac{\grave{d}_p}{\grave{d}_{max}} \tag{1}$$

$$\grave{S}_d = \frac{\min(\grave{\rho}_i, \grave{\rho}_j)}{\max(\grave{\rho}_i, \grave{\rho}_j) + \varepsilon} \tag{2}$$

$$\grave{S}_f = \frac{\grave{\mu}_i \cdot \grave{\mu}_j}{\|\grave{\mu}_i\| \, \|\grave{\mu}_j\|} \tag{3}$$

where $\grave{d}_p$ is the centroid distance between clusters C_i and C_j, $\grave{d}_{max}$ is the maximum possible distance between C_i and C_j, $\grave{\rho}_i$ and $\grave{\rho}_j$ are the cluster densities of clusters C_i and C_j, ε is a constant, $\grave{\mu}_i$ and $\grave{\mu}_j$ are the mean feature vectors of clusters C_i and C_j and $\|\grave{\mu}_i\|$ and $\|\grave{\mu}_j\|$ are the vector norms.

Adaptive Incremental Clustering — Part IV/IV

Algorithm 1: (Part IV/IV)

foreach $f_{new} \in F_t$ *(cont.)* **do**
  else
    `// Handle potential noise`
    if *none of* N_ϵ *belongs to any cluster* **then**
      Mark f_{new} as temporary noise;
    else
      Find nearest neighbor $f_{nn} \in N_\epsilon \cap C_j$; assign f_{new} to cluster of f_{nn};

`// Noise Re-Assessment Phase`
foreach *point* p *previously labelled as noise* **do**
  Recompute neighbors N_p within local ϵ_p;
  if $|N_p| \geq \mathrm{MinPts}_p$ **then**
    Assign p to the nearest valid cluster;

return $\mathcal{C}_{\text{updated}}$;

Class-aware Adaptive Pseudo-Labeling — Part I/IV

Algorithm 1: Class-aware adaptive pseudo-labeling refinement (Part I/IV)

Input: Source features F_s with labels Y_S, Target features F_t, Target clusters C_T, Top-k value k, temperature τ, contrastive weight λ_{proto}, scaling factor α_2

Output: Refined pseudo-labels and trained classifier

```
// Initialize Source Class Prototypes
```

for *each source class* $s \in Y_S$ **do**

  Compute initial prototype $\mu_s^{(0)} = \frac{1}{|F_s^s|} \sum_{x \in F_s^s} f(x)$;

```
// Iteratively update pseudo-labels and prototypes
```

for *each training epoch* m **do**

Class-aware Adaptive Pseudo-Labeling — Part II/IV

Algorithm 1: (Part II/IV)

for *each training epoch m (cont.)* **do**
    `// Assign Soft Pseudo-Labels with Class-Wise Top-k Filtering`
    Initialize $\mathcal{P}[s] = \emptyset$ for each class s;
    `// Pseudo-label generation`
    for *each target sample $x_i \in F_t$* **do**
        Identify cluster c_i of x_i from C_T;
        Compute soft probabilities:

$$P(y_i = s) = \frac{\exp(\mathrm{sim}(f(x_i), \mu_s^{(m-1)})/\tau)}{\sum_{j=1}^{Q} \exp(\mathrm{sim}(f(x_i), \mu_j^{(m-1)})/\tau)}$$

        Let $s^* = \arg\max_s P(y_i = s)$;
        `// Class-aware refinement (cluster confidence)`
        Compute cluster-level confidence γ_{c_i} (e.g., mean intra-cluster similarity/density);
        Compute class-aware threshold $\tau_{c_i} = \alpha_2 \cdot \mathrm{mean}(\gamma_{c_i})$;
        if $\gamma_{c_i} \geq \tau_{c_i}$ **then**
            Add $(x_i, P(y_i = s^*), f(x_i), \textbf{weight} = 1.0)$ to $\mathcal{P}[s^*]$;
        else
            Add $(x_i, P(y_i = s^*), f(x_i), \textbf{weight} = 0.5)$ to $\mathcal{P}[s^*]$;

Class-aware Adaptive Pseudo-Labeling — Part III/IV

Algorithm 1: (Part III/IV)

```
for each training epoch m (cont.) do
    // Top-k selection
    for each class s do
        Sort P[s] by confidence and retain top-k samples;
    // Update prototypes from top-k target samples
    for each class s do
        Compute μ_s^(m) = Σ_{(x_i,w_i)∈P[s]_top-k} w_i f(x_i) / Σ_{(x_i,w_i)∈P[s]_top-k} w_i ;
```

for *each training epoch m (cont.)* **do**

- // Top-k selection
- **for** *each class s* **do**
  - Sort $\mathcal{P}[s]$ by confidence and retain top-k samples;
- // Update prototypes from top-k target samples
- **for** *each class s* **do**
  - Compute $\mu_s^{(m)} = \dfrac{\sum_{(x_i,w_i)\in\mathcal{P}[s]_{\text{top-}k}} w_i\, f(x_i)}{\sum_{(x_i,w_i)\in\mathcal{P}[s]_{\text{top-}k}} w_i}$;

Class-aware Adaptive Pseudo-Labeling — Part IV/IV

Algorithm 1: (Part IV/IV)

```
for each training epoch m (cont.) do
    // Prototype contrastive loss
    for each pseudo-labelled sample (x_i, f(x_i)) do
        L_proto(x_i) = - log [ exp(sim(f(x_i), μ_{s*}^{(m)})/τ) / Σ_{j=1}^{Q} exp(sim(f(x_i), μ_j^{(m)})/τ) ]
    // Classifier training
    Compute cross-entropy loss L_cls over confident samples;
    Total loss: L_total = L_cls + λ_proto · L_proto;
    Update network parameters using L_total;
return refined pseudo-labels s*;
```

for *each training epoch m (cont.)* **do**

`// Prototype contrastive loss`

for *each pseudo-labelled sample* $(x_i, f(x_i))$ **do**

$$\mathcal{L}_{\text{proto}}(x_i) = -\log \frac{\exp(\mathrm{sim}(f(x_i), \mu_{s^*}^{(m)})/\tau)}{\sum_{j=1}^{Q} \exp(\mathrm{sim}(f(x_i), \mu_j^{(m)})/\tau)}$$

`// Classifier training`

Compute cross-entropy loss $\mathcal{L}_{\text{cls}}$ over confident samples;

Total loss: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cls}} + \lambda_{\text{proto}} \cdot \mathcal{L}_{\text{proto}}$;

Update network parameters using $\mathcal{L}_{\text{total}}$;

return refined pseudo-labels s^*;

Deep learning-based pareto front generation

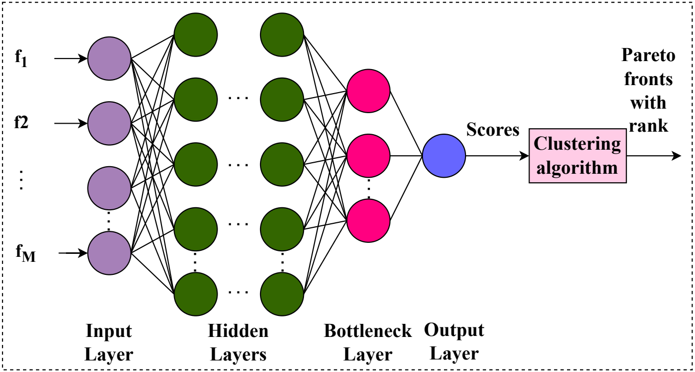


Figure 5: Deep learning-based pareto front generation.

How the Pareto fronts with ranks are generated

1. **Scores from the network.** For each solution with objectives $\boldsymbol{f}_i$, compute a score/embedding:

$$s_i = h_\phi(\boldsymbol{f}_i).$$

2. **Cluster scores.** Run MOC algorithm on $\{s_i\}$ to obtain clusters $C_1, \ldots, C_K$ (with centers $\{\mu_k\}$).
3. **Order clusters to form fronts.** Define a monotone *front quality* q_k:
   - If $d >= 1$: $q_k = \text{median}\{s_i : i \in C_k\}$
4. **Rank.** Sort clusters by q_k ascending:

$$q_{(1)} \leq q_{(2)} \leq \cdots \leq q_{(K)} \Rightarrow \text{Front } 1 = C_{(1)}, \text{ Front } 2 = C_{(2)}, \ldots$$

Every solution $i \in C_{(r)}$ receives **rank** r.

Table 3: Details of the Datasets

	AID (A) [14]	NWPU-RESISC45 (N) [15]	PatternNet (P) [16]	UC Merced (U) [17]
No of classes	30	45	38	21
Resolution (m)	0.5–8	0.2–30	0.062 - 4.693	0.3
Pixel size	600 × 600	256 × 256	256 x 256	256 × 256
Farmland	370	700	800	100
Forest	250	700	800	100
Parking	390	700	800	100
Residential	410	700	800	100
River	410	700	800	100

Table 4: Hyper-parameters, roles, search spaces, selection rules, and chosen values.

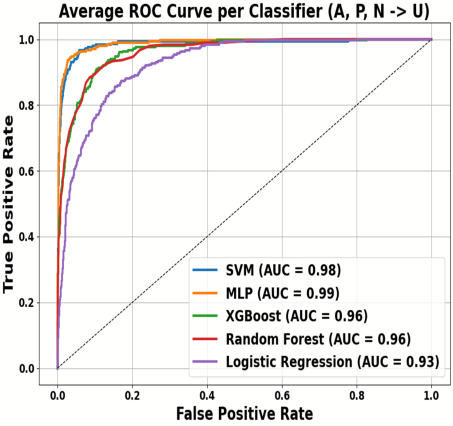
Param	Role	Search space	Selection rule	Chosen
n_1, n_2	dense-region neighbor rank	$[3, 8], [8, 12]$	best proxy rank-sum	$5, 10$
n_3, n_4	sparse-region neighbor rank	$[15, 30], [40, 60]$	best proxy rank-sum	$20, 50$
λ_{proto}	prototype-contrastive weight	$[0.1, 0.6]$	entropy is minimized	0.3
Top-k	per-class target selection	$\{10, 20, 50\}$	stability vs. coverage	20

Classification accuracy

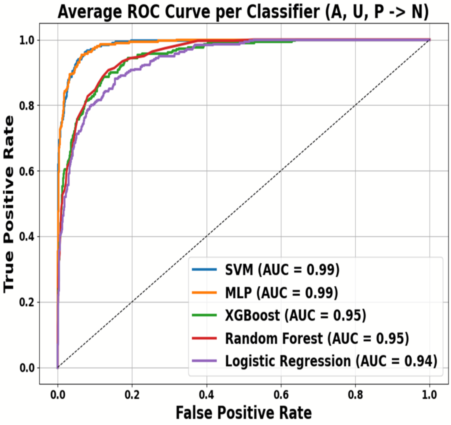
Table 5: Comparison of classification accuracy on multi-source domain adaptation methods across various domain combinations

Domain	M^3SDA [5]	MFSAN [6]	LCt-MSDA [7]	T-SVDNet [8]	MCC-DA [10]	PTMDA [9]	SUMDA [12]	RRL [11]	Hy-MSDA [13]	**XMUDA-CLAC**
(A $\rightarrow$ U)	0.887	0.912	0.873	0.854	0.940	0.944	0.944	0.946	0.959	**0.965**
(P $\rightarrow$ N)	0.870	0.907	0.868	0.855	0.931	0.928	0.939	0.937	0.953	**0.962**
(U $\rightarrow$ P)	0.879	0.910	0.870	0.859	0.938	0.930	0.933	0.933	0.947	**0.954**
(A, P $\rightarrow$ U)	0.883	0.919	0.890	0.865	0.950	0.951	0.949	0.944	0.968	**0.972**
(A, N $\rightarrow$ U)	0.895	0.920	0.905	0.860	0.945	0.948	0.950	0.951	0.972	**0.977**
(U, P $\rightarrow$ N)	0.917	0.940	0.908	0.881	0.967	0.968	0.965	0.962	**0.978**	**0.978**
(A, P, N $\rightarrow$ U)	0.901	0.923	0.898	0.869	0.957	0.954	0.954	0.955	0.974	**0.976**
(A, U, P $\rightarrow$ N)	0.922	0.928	0.886	0.884	0.964	0.960	0.963	0.955	0.977	**0.978**

AUC-ROC curve



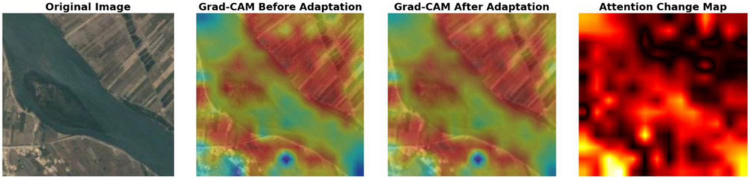
(a) (A, P, N → U)



(b) (A, U, P → N)

Figure 6: Classifier performance (AUC-ROC) across tasks (a) (A, P, N → U) (b) (A, U, P → N).

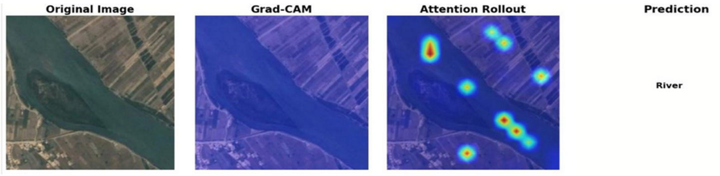
Visualization



(a) (b) (c) (d)

Figure 7: Visualization of attention shift before and after domain adaptation in (A, U, P $\rightarrow$ N)

Visualization



(a) (b) (c) (d)

Figure 8: Visual explanation of target scene prediction Using Grad-CAM and Attention Rollout (A, U, P $\rightarrow$ N)

Summary

- This framework effectively extracts domain-invariant features and generates high-confidence pseudo-labels for the unlabelled target domain.
- The robustness to class imbalance and feature drift is further enhanced through class-aware pseudo-label refinement
- This approach is applicable for shared classes, but for a generalized setting, we go for Universal UDA.

References I

[1] Han-Kai Hsu et al. "Progressive domain adaptation for object detection". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2020, pp. 749–757.

[2] Yu-Chu Yu and Hsuan-Tien Lin. "Semi-supervised domain adaptation with source label adaptation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 24100–24109.

[3] Yixin Zhang, Zilei Wang, and Weinan He. "Class relationship embedded learning for source-free unsupervised domain adaptation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 7619–7629.

[4] Rudolf Scitovski and Kristian Sabo. "DBSCAN-like clustering method for various data densities". In: *Pattern Analysis and Applications* 23.2 (2020), pp. 541–554.

References II

[5] Xingchao Peng et al. "Moment matching for multi-source domain adaptation". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 1406–1415.

[6] Yongchun Zhu, Fuzhen Zhuang, and Deqing Wang. "Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 5989–5996.

[7] Hang Wang et al. "Learning to combine: Knowledge aggregation for multi-source domain adaptation". In: *European Conference on Computer Vision*. Springer. 2020, pp. 727–744.

References III

[8] Ruihuang Li et al. “T-svdnet: Exploring high-order prototypical correlations for multi-source domain adaptation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9991–10000.

[9] Chuan-Xian Ren et al. “Multi-source unsupervised domain adaptation via pseudo target domain”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 2122–2135.

[10] Yikang Wei et al. “Multi-source collaborative contrastive learning for decentralized domain adaptation”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 33.5 (2022), pp. 2202–2216.

[11] Sentao Chen, Lin Zheng, and Hanrui Wu. “Riemannian representation learning for multi-source domain adaptation”. In: *Pattern Recognition* 137 (2023), p. 109271.

References IV

[12] Mengmeng Li et al. "Cross-domain urban land use classification via scenewise unsupervised multisource domain adaptation with transformer". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17 (2024), pp. 10051–10066.

[13] Kai Xu et al. "Enhancing Remote Sensing Scene Classification with Hy-MSDA: A Hybrid CNN-Transformer for Multi-Source Domain Adaptation". In: *IEEE Transactions on Geoscience and Remote Sensing* (2024).

[14] Gui-Song Xia et al. "AID: A benchmark data set for performance evaluation of aerial scene classification". In: *IEEE Transactions on Geoscience and Remote Sensing* 55.7 (2017), pp. 3965–3981.

[15] Gong Cheng, Junwei Han, and Xiaoqiang Lu. "Remote sensing image scene classification: Benchmark and state of the art". In: *Proceedings of the IEEE* 105.10 (2017), pp. 1865–1883.

References V

[16] Weixun Zhou et al. "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval". In: *ISPRS journal of photogrammetry and remote sensing* 145 (2018), pp. 197–209.

[17] Yi Yang and Shawn Newsam. "Bag-of-visual-words and spatial extensions for land-use classification". In: *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*. 2010, pp. 270–279.

Thank you

Email: binujose_p200050cs@nitc.ac.in,
praneshdas@nitc.ac.in,
ebrahim.ghaderpour@uniroma1.it,
paolo.mazzanti@uniroma1.it