LDAuCID

Lifelong Domain Adaptation via Consolidated Internal Distribution

Team Members

1.	Atharva Singh	220251
2.	Kawaljeet Singh	220514
3.	Nikhil Jain	220709
4.	Riyanshu Kumar	220904

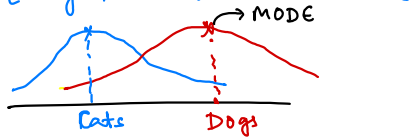
Imagine we are training a machine learning model to classify animals in images.

- **Step 1: Initial Training**
  We start with a labeled training dataset $D_S = (X_0, Y_0)$:
  - X_0: A collection of images containing only **dogs** and **cats**.
  - Y_0: Corresponding labels, e.g., "Dog" for dog images and "Cat" for cat images.

  The images are sampled from an unknown source distribution $p_0(x)$, meaning we do not know the exact rules governing how these images were generated but assume they are representative of dogs and cats in general.
- **Step 2: Expanding the Problem**
  Now, we are given new data points that include images of **foxes** and **wolves**, which were not part of the original training set.
  - The goal is to classify these new animals (foxes and wolves) correctly without compromising the model's accuracy on dogs and cats.
- **Challenge**:
  - How do we train the model effectively to generalize over this expanded set of classes while still maintaining high performance on the initial classes?
  - This requires adapting the model to work across the broader domain of **dogs, cats, foxes, and wolves**, even though the original data D_S was limited.

→ Input distribution $x \sim p(x)$
→ Assumptions – Gaussian Distribution

K classes → [dogs, cats, foxes, wolves, ...]



Problem Statement

1. **Source Domain (S):**
   - Training data is labeled: $D_S = (X_0, Y_0)$, with:
     - $X_0 \in \mathbb{R}^{d \times N}$: Feature matrix (d-dimensional, N samples).
     - $Y_0 \in \mathbb{R}^{k \times N}$: Labels for X_0.
   - Samples follow a fixed source distribution: $x_{0,i} \sim p_0(x)$.
2. **Model Training on Source Domain:**
   - A deep neural network f_θ (parameters θ) is trained using **Empirical Risk Minimization (ERM)**:

     $$\hat{\theta}_0 = \arg\min_\theta \sum_i L(f_\theta(x_{0,i}), y_{0,i}),$$

     where L is a suitable loss function.
   - If the dataset and model are sufficiently large/complex, $f_{\hat{\theta}_0}$ generalizes well on new data from $p_0(x)$.

   Continual Learning (CL) Objective:
   - Sequentially arriving **unlabeled target domains**: $T^t, t = 1, \ldots, T$
     - Datasets: $D_T^t = (X^t)$, $X^t \in \mathbb{R}^{d \times M_t}$, $x_i^t \sim p_t(x)$.
   - Goal: Update the source-trained model $\hat{f}_{\hat{\theta}_0}$ using unlabeled data while avoiding:

Notable Aspects

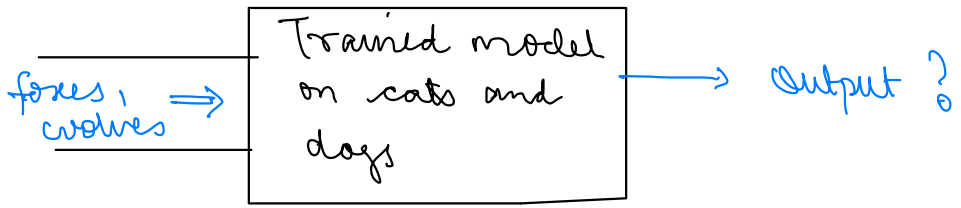
D1 domain 1 Cats and dog
D2 domain 2 foxes and wolves

Unsupervised Domain Adaptation (UDA)

- **Definition**:
  UDA trains models for a target domain with unannotated data by transferring knowledge from a source domain with labeled data.
- **Core Idea**:
  - Map data points from both domains into a shared latent embedding space.
  - Align source and target domain distributions within this space.
- **Key Methods**:
  - Use **probability distances** (e.g., Wasserstein distance) to measure and minimize cross-domain discrepancies.
  - Train encoders to align latent representations between domains.
- **Goal**: Enable effective domain alignment for improved performance in the target domain.

Continual Learning (CL)

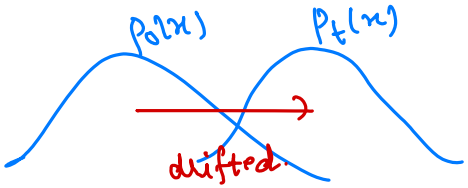
- **Definition**:
  Continual learning aims to enable models to learn adaptively and autonomously across tasks over extended time periods, even when the input distribution drifts.
- **Challenges**:
  - **Domain shift**: Changes in data distributions over time degrade model performance.
  - **Catastrophic forgetting**: Models forget previously learned tasks when updated for new tasks.
- **Key Strategies**:
  - **Weight consolidation**: Preserve important network weights related to past tasks during updates.
  - **Memory replay**: Store critical training data points in a memory buffer and replay them alongside new data to retain past knowledge.
- **Major Issue**: Identifying critical data points to retain knowledge effectively.



Challenges with Known Models

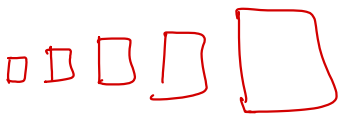
Drifted Distribution

- **Context**:
  Domain shift occurs when the training and testing data distributions differ, leading to challenges in maintaining model performance over time.
- **Impact**:
  - Performance degradation as the model becomes less effective with shifted distributions.
  - Necessitates frequent retraining to accommodate new data distributions.
- **Relation to Continual Learning**:
  - Drifted distributions are a natural challenge in CL, as models must adapt to such shifts over time.



Catastrophic Forgetting

- **Definition**:
  A phenomenon where a model forgets previously learned tasks due to retroactive interference during updates for new tasks.
- **Challenges**:
  - Loss of previously acquired knowledge.
  - Hinders models from retaining information across multiple tasks.
- **Key Strategies to Address**:
  - **Weight Consolidation**: Strengthen weights tied to past tasks.
  - **Memory Replay**: Replay samples from prior tasks to reinforce past knowledge.
  - **Internal Distribution Analysis**: Identify critical data points to focus retention efforts.



Proposed Solution

To address "domain shift" and "catastrophic forgetting," we decompose the base network $f_\theta(\cdot)$ into a deep encoder $\phi_v(\cdot) : \mathcal{X} \to \mathcal{Z} \subset \mathbb{R}^p$ and a classifier subnetwork $h_w(\cdot) : \mathcal{Z} \to \mathcal{Y}$, where $f_\theta = h_w \circ \phi_v$ and $\theta = (w, v)$. The method focuses on stabilizing the internal distribution in the embedding space $\mathcal{Z}$, assuming source classes are separable in $\mathcal{Z}$ after initial training. By minimizing the distributional shift between $\phi(p_0(x_0))$ (source) and $\phi(p_t(x_t))$ (target) at each step, the model generalizes well on target domains, even when trained only on the labeled source dataset. While common unsupervised domain adaptation (UDA) strategies rely on source data access, this is infeasible in continual learning (CL), making it impossible to explicitly compute $\phi(p_0(x_0))$.

Wasserstein Distance

Walkthrough

1. **Learning and Consolidating Multimodal Internal Distribution (Source Domain):**

   - **Training on Source Domain:**

     The model's encoder $\phi_v(\cdot)$ maps input data to a latent embedding space Z.

     In this space, the distribution $p_0^J(z)$ forms a **multimodal Gaussian Mixture Model (GMM)**, where:

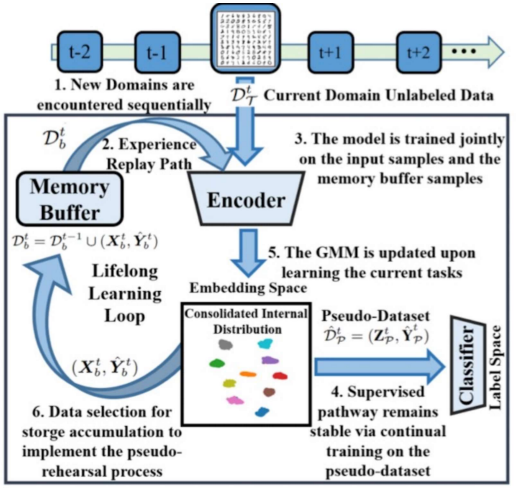
     $$p_J^0(\boldsymbol{z}) = \sum_{i=1}^{k} \alpha_j^0 \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_j^0, \boldsymbol{\Sigma}_j^0),$$

     - Each mode represents a class.
     - Data points of the same class are clustered around a mean μ_0^j with covariance Σ_0^j.

     Parameters of the GMM $(\alpha_0^j, \mu_0^j, \Sigma_0^j)$ are estimated using the labeled source data.

   - **Purpose of GMM:**

     The GMM serves as a compact representation of the learned distribution. It helps align new domain data (target domains) with this internal distribution to ensure generalizability.



Lifelong Domain Adaptation Agent

MAP estimate for the GMM parameters at stage 0 :

$$\hat{\alpha}_j^0 = \frac{|\boldsymbol{S}_j^0|}{N},\ \hat{\boldsymbol{\mu}}_j = \sum_{(\boldsymbol{x}_i^0, \boldsymbol{y}_i^0) \in \boldsymbol{S}_j^0} \frac{1}{|\boldsymbol{S}_j^0|} \phi_v(\boldsymbol{x}_i^0),\ \hat{\boldsymbol{\Sigma}}_j^0 = \sum_{(\boldsymbol{x}_i^0, \boldsymbol{y}_i^0) \in \boldsymbol{S}_j^0} \frac{1}{|\boldsymbol{S}_j^0|} \left(\phi_v(\boldsymbol{x}_i^0) - \hat{\boldsymbol{\mu}}_j^0\right)^\top \left(\phi_v(\boldsymbol{x}_i^0) - \hat{\boldsymbol{\mu}}_j^0\right). \quad (2)$$

Walkthrough

2. **Adapting to New Domains (Target Domains):**

- **Embedding Alignment with GMM:**
  For a new target domain, the encoder is updated to align the embeddings of unlabeled target data with the source GMM in the latent space Z.
  This alignment ensures that the new data adheres to the previously learned internal distribution.
- **Generating Pseudo-Labeled Data:**
  - Random samples are drawn from the GMM to create a pseudo-labeled dataset $\hat{D}_t^P = (Z_t^P, \hat{Y}_t^P)$.
  - Only samples with high model confidence (above a threshold τ) are used, avoiding noisy labels.
- **Objective Function:**
  At time t, the model minimizes the following loss:

$$\min_{v,w} \sum_{i=1}^{N} L(h_w(z_p^i), \hat{y}_{p,t}^i) + \lambda D(\phi_v(p_t(X_t)), \hat{p}_t^J(Z_t^P)),$$

  where:
  - $L(h_w(z), y)$: Classification loss ensuring predictions match pseudo-labels.
  - $D(\cdot, \cdot)$: A discrepancy measure (e.g., Sliced Wasserstein Distance, SWD) aligning target domain embeddings with the GMM.

$\hat{p}_j^t(z)$
internal distribution in embedding space
$z_i^t \sim \hat{p}_j^t(z)$
$z_i^t \xrightarrow[\text{model}]{} \hat{y}_i^t$
(high confidence > τ) used.
target domain prediction

Walkthrough

3. **Mitigating Catastrophic Forgetting:**

   - **Experience Replay with Representative Samples:**
     - A memory buffer stores a small set of representative samples from each task/domain.
     - Samples are selected using a **Mean of Features (MoF)** strategy:
       - For each class, the M_b samples closest to the GMM mean μ_t^j are chosen (informative samples).
       - The buffer D_t^b maintains these samples across all domains.
   - **Augmented Loss Function:**
     The loss is modified to include supervised learning on buffer samples and further alignments:

$$\min_{v,w} \underbrace{\sum_{i=1}^{N} L(h_w(z_p^i), \hat{y}_{p,t}^i)}_{\text{Pseudo-labeled data}} + \underbrace{\sum_{i=1}^{N_b} L(h_w(\phi_v(x_b^i)), \hat{y}_b^i)}_{\text{Buffered samples}} + \lambda \underbrace{D(\phi_v(p_t(X_t)), \hat{p}_t^J(Z_t^P))}_{\text{Target embeddings alignment}} + \lambda \underbrace{D(\phi_v(p_t(X_t^b)), \hat{p}_t^J(Z_t^P))}_{\text{Buffer embeddings alignment}}.$$

     - The second term prevents forgetting by retraining on buffered data.
     - The fourth term consolidates the internal distribution across all domains.

Algorithm

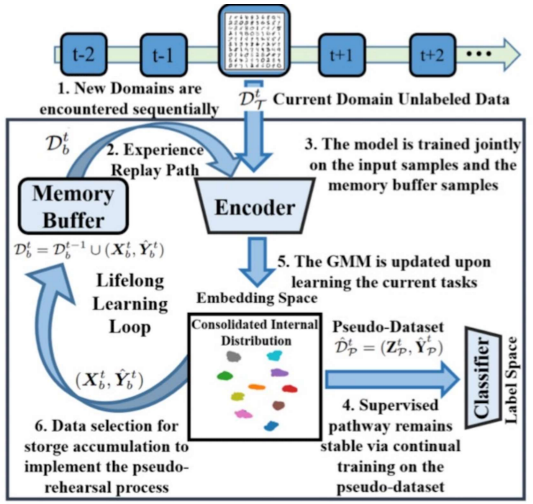
Algorithm 1 LDAuCID (λ, τ, N_b)

Source Training:
  Input: source labeled dataset $\mathcal{D}_{\mathcal{S}} = (\boldsymbol{X}_0, \boldsymbol{Y}_0)$
    $\hat{\theta}_0 = (\hat{\boldsymbol{w}}_0, \hat{\boldsymbol{v}}_0) = \arg\min_\theta \sum_i \mathcal{L}(f_\theta(\boldsymbol{x}_i^0), \boldsymbol{y}_i^0)$
  Internal Distribution Estimation:
    Use Eq. (2) and estimate $\alpha_j^0, \boldsymbol{\mu}_j^0$, and Σ_j^0
  Memory Buffer Initialization
    $\mathcal{D}_b^0 = (\boldsymbol{X}_b^0, \hat{\boldsymbol{Y}}_b^0)$
    Pick the N_b/k samples with the least $d_{j,l}^t = \|\mu_j^t - \phi(\boldsymbol{x}_l^t)\|_2^2$, $\hat{\boldsymbol{y}}_b^{0,i} = \arg\max f_{\hat{\theta}^t}(\boldsymbol{x}_b^{0,N_b})$
Continual Unsupervised Domain Adaptation:
for $t = 1, \ldots, T$ **do**
    Input: target unlabeled dataset $\mathcal{D}_{\mathcal{T}}^t = (\boldsymbol{X}_t)$
    Pseudo-Dataset Generation:
    $\hat{\mathcal{D}}_{\mathcal{P}}^t = (\mathbf{Z}_{\mathcal{P}}^t, \hat{\mathbf{Y}}_{\mathcal{P}}^t) =$
    $([\boldsymbol{z}_1^{p,t}, \ldots, \boldsymbol{z}_N^{p,t}], [\hat{\boldsymbol{y}}_1^{p,t}, \ldots, \hat{\boldsymbol{y}}_N^{p,t}])$, where:
    $\boldsymbol{z}_i^{p,t} \sim \hat{p}_J^{t-1}(\boldsymbol{z}), 1 \le i \le N_p$ and
    $\hat{\boldsymbol{y}}_i^{p,t} = \arg\max_j \{h_{\hat{\boldsymbol{w}}_t}(\boldsymbol{z}_i^{p,t})\}$ if with confidence τ: $\max_j \{h_{\hat{\boldsymbol{w}}_t}(\boldsymbol{z}_i^{p,t})\} > \tau$
  for $itr = 1, \ldots, ITR$ **do**
    draw data batches from $\mathcal{D}_{\mathcal{T}}^t$ and $\hat{\mathcal{D}}_{\mathcal{P}}$
    Update the model by solving Eq. (4)
  end for
  Internal Distribution Estimate Update:
    Use Eq. (2) similar to step 5 above.
  Memory Buffer Update
    $\mathcal{D}_b^t = \mathcal{D}_b^{t-1} \cup (\boldsymbol{X}_b^t, \hat{\boldsymbol{Y}}_b^t)$, where $(\boldsymbol{X}_b^t, \hat{\boldsymbol{Y}}_b^t)$ is computed similar to step 7 above.
end for

MAP estimate for the GMM parameters at stage 0 :

$$\hat{\alpha}_j^0 = \frac{|\boldsymbol{S}_j^0|}{N},\ \hat{\boldsymbol{\mu}}_j = \sum_{(\boldsymbol{x}_i^0, \boldsymbol{y}_i^0) \in \boldsymbol{S}_j^0} \frac{1}{|\boldsymbol{S}_j^0|} \phi_v(\boldsymbol{x}_i^0),\ \hat{\boldsymbol{\Sigma}}_j^0 = \sum_{(\boldsymbol{x}_i^0, \boldsymbol{y}_i^0) \in \boldsymbol{S}_j^0} \frac{1}{|\boldsymbol{S}_j^0|} (\phi_v(\boldsymbol{x}_i^0) - \hat{\boldsymbol{\mu}}_j^0)^\top (\phi_v(\boldsymbol{x}_i^0) - \hat{\boldsymbol{\mu}}_j^0). \quad (2)$$

Memory Budget = Nb samples



Lifelong Domain Adaptation Agent

$$\min_{v,w} \sum_{i=1}^{N} \mathcal{L}(h_w(\boldsymbol{z}_i^p), \hat{\boldsymbol{y}}_i^{p,t}) + \sum_{i=1}^{N_b} \mathcal{L}(h_w(\phi_v(\boldsymbol{x}_i^b)), \hat{\boldsymbol{y}}_i^o) + \lambda D(\phi_v(p_t(\boldsymbol{X}_t)), \hat{p}_J^\tau(\boldsymbol{Z}_{\mathcal{P}}^\tau)) + \lambda D(\phi_v(p_t(\boldsymbol{X}_b^\tau)), \hat{p}_J^\tau(\boldsymbol{Z}_{\mathcal{P}}^\tau))$$

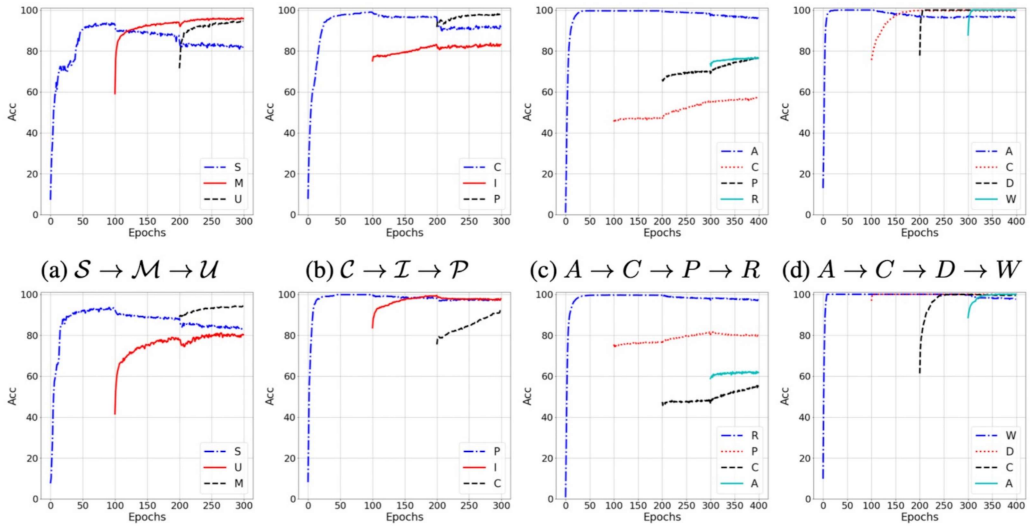
Results

2. Experimental Setup

- **Datasets and Tasks:**
  - Used four benchmark UDA datasets:
    1. **Digit Recognition Tasks:** MNIST (M), USPS (U), SVHN (S).
    2. **ImageCLEF-DA:** Caltech (C), ILSVRC (I), Pascal VOC (P).
    3. **Office-Home:** Artistic (A), Clip Art (C), Product (P), Real-World (R).
    4. **Office-Caltech:** Amazon (A), Caltech (C), DSLR (D), Webcam (W).
  - Sequential UDA tasks were designed to simulate continual learning scenarios.
- **Model Architecture:**
  - Used ResNet-50, VGG16, and Decaf6 depending on the dataset.
  - Learning curves tracked over 100 training epochs for each task to measure performance dynamics.

3. Key Results

- **Performance on Sequential UDA Tasks:**
  - LDAuCID effectively handles **domain shifts** with significant improvements in model accuracy across all target domains.
  - Demonstrated **jumpstart performance** on subsequent tasks due to knowledge transfer, starting from higher-than-random accuracies (e.g., +60% on Digit tasks).
- **Mitigation of Catastrophic Forgetting:**
  - After training on new domains, performance on previously learned tasks remained **relatively stable**.
  - Forgetting effects were more pronounced in more challenging datasets like SVHN, indicating room for improvement with larger memory buffers.
- **Comparison with Classic UDA Methods:**
  - Despite lifelong UDA constraints (limited access to source data), LDAuCID:
    - **Outperformed or matched** several classic UDA methods on standard UDA tasks.
    - Demonstrated competitive performance on unbalanced datasets (Office-Home) and significant gains on balanced datasets (ImageCLEF-DA).



(a) $\mathcal{S} \rightarrow \mathcal{M} \rightarrow \mathcal{U}$ (b) $\mathcal{C} \rightarrow \mathcal{I} \rightarrow \mathcal{P}$ (c) $A \rightarrow C \rightarrow P \rightarrow R$ (d) $A \rightarrow C \rightarrow D \rightarrow W$

(e) $\mathcal{S} \rightarrow \mathcal{U} \rightarrow \mathcal{M}$ (f) $\mathcal{P} \rightarrow \mathcal{I} \rightarrow \mathcal{C}$ (g) $R \rightarrow P \rightarrow C \rightarrow A$ (h) $W \rightarrow D \rightarrow C \rightarrow A$

Figure 2: Learning curves for sequential UDA tasks on (a,e) digit, (b,f) ImageClef, (c,g) Office-Home, and (d,h) Office-Caltech datasets. (Best viewed in color).