Computer Vision II, Spring 2025
Assignment 1

Ting Zhou

April, 2025

1 [10 points] Train your classification model using PyTorch (source-only training for domain adaptation)

1.1 some instances of Office-Home dataset

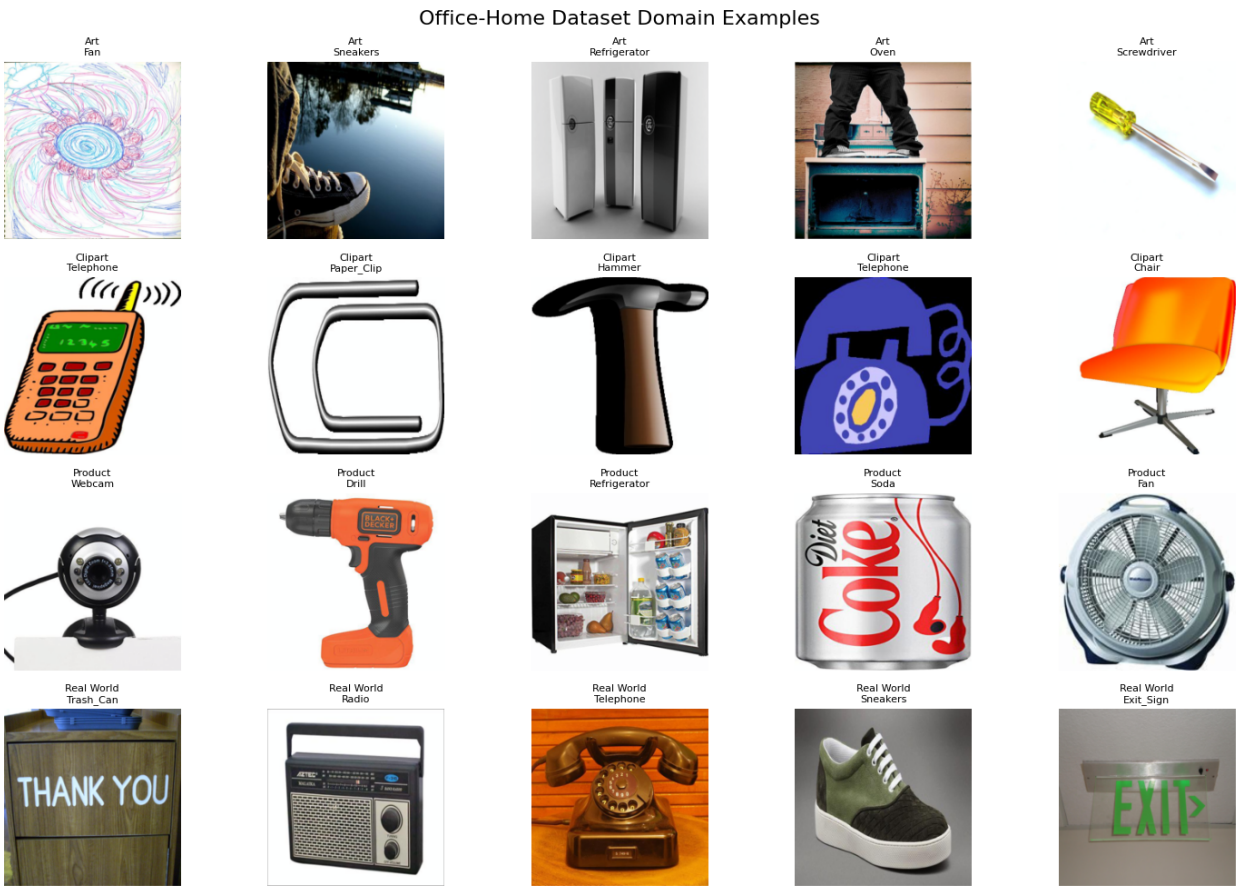

Figure 1: some instances per domain.

1.2 Loss function

Principle

- Supervised Learning on Source Domain:The model $f = h \circ g$, consisting of a feature extractor g and a classifier h, is trained on labeled source domain data $\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$ to minimize classification error.
- Implicit Regularization:Techniques like data augmentation, Dropout, and weight decay are used to prevent overfitting to the source domain and improve generalization.
- Assumption The source and target domains share a similar underlying feature space, so optimizing the source classification task indirectly improves performance on the target domain.

Loss Function The loss function in Source-only DA consists only of the supervised classification loss (no explicit domain alignment loss is used):

$$\mathcal{L}_{\text{source}} = \mathbb{E}_{(x^s, y^s) \sim \mathcal{D}_s} \left[\mathcal{L}_{\text{cls}}(h(g(x^s)), y^s) \right]$$

where:
• $\mathcal{L}_{\text{cls}}$ is the classification loss (e.g., cross-entropy):

$$\mathcal{L}_{\text{cls}}(p, y) = -\sum_{c=1}^{C} y_c \log p_c$$

• $p = h(g(x^s))$ is the predicted probability distribution.

1.3 implementation details

data augmentation
train:

- Enhance data diversity through color perturbation and flipping, and enhance the generalization ability of the model by using random cropping.
- Convert to a tensor.
- Normalize using the three-channel standardized parameters of a specific dataset parameters.

test:

- Use center cropping processing (no random enhancement).
- Convert to a tensor.
- Normalize using the three-channel standardized parameters of a specific dataset parameters.

network structure:

- The pre-trained ResNet50 is used as the feature extractor to remove its original classification layer (fc), and the output dimension is 2048.
- The custom classifier contains a fully connected layer (2048 to 65) and a regularization layer (Dropout).

batch size 64
loss and Optimizer

- Only cross-entropy loss (source domain supervision) is used, and there is no domain alignment loss.
- The Adam optimizer is used to jointly optimize the feature extractor and classifier.

learning rate Initial value 0.001.
training steps epoch=100,train steps=$2724 \times 100/64 = 4257$.

1.4 Result

The source domain is selected as "Art" and the target domain is selected as "real world". The characteristics of the two domains are quite different. It can be observed from the figure that the training loss has decreased from 4.3 to 0.9. The training accuracy rate rose from 4% to 75%, and the test accuracy rate rose from 3% to 30%. The increase in the test accuracy rate was slow.

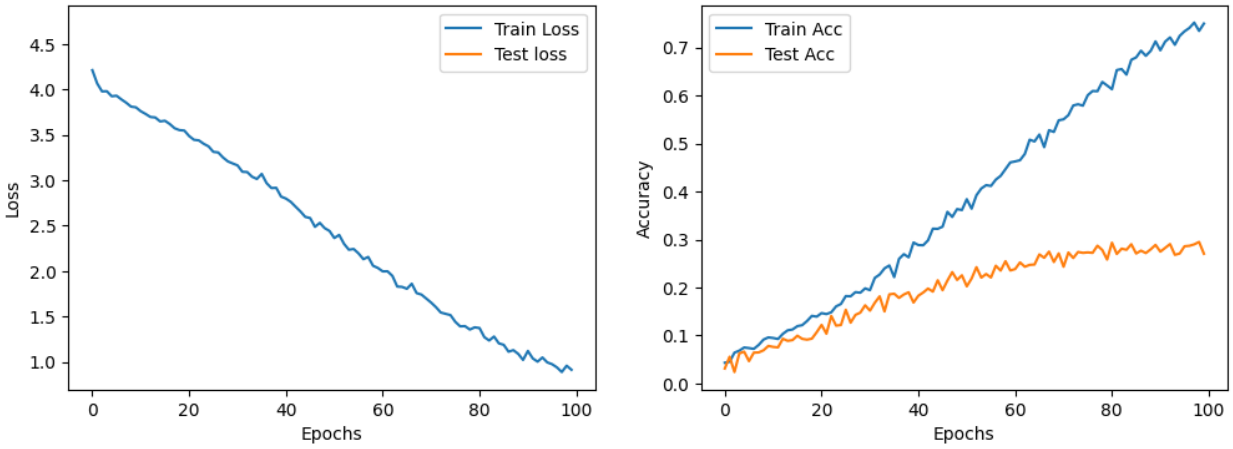

Figure 2: Loss and Accuracy of source-only DA

2 [30 points] Domain-Adversarial Training (DANN) for Domain Adaptation

2.1 Loss function

DANN achieves domain adaptation through adversarial training with three key components:

Feature Extractor(G_f):Learn to generate domain-invariant features to align the feature distributions of the source domain and the target domain.

Domain Discriminator(G_d):Attempt to distinguish whether the features come from the source domain or the target domain (similar to the discriminator of GAN).

Gradient Reversal Layer(GRL):During backpropagation, the gradient of the domain discriminator is reversed, forcing the feature extractor to "deceive" the discriminator, thereby eliminating the domain differences.

The total loss combines:

1. Source Classification Loss (Supervised):

$$\mathcal{L}_y = \mathbb{E}_{(x_s,y_s)\sim\mathcal{D}_s}\left[-\sum_{i=1}^{C} y_s^i \log G_y(G_f(x_s))^i\right]$$

   • C: Number of classes • Ensures features remain discriminative

2. Domain Adversarial Loss (Unsupervised Alignment):

$$\mathcal{L}_d = \mathbb{E}_{x_s\sim\mathcal{D}_s}\left[\log G_d(G_f(x_s))\right] + \mathbb{E}_{x_t\sim\mathcal{D}_t}\left[\log(1 - G_d(G_f(x_t)))\right]$$

   • G_d maximizes $\mathcal{L}_d$ (better domain discrimination) • G_f minimizes $\mathcal{L}_d$ via GRL (better domain confusion)

3. Total Loss:

$$\mathcal{L} = \mathcal{L}_y + \lambda\mathcal{L}_d$$

   • λ: Trade-off parameter (typically annealed during training)

2.2 implementation

data augmentation:

The source domain and the target domain are handled in the same way as the source domain in train phase of Problem one.The target domain is handled in the same way as the target domain of Problem One, and the 10-crop data augening strategy is used.

network structure:

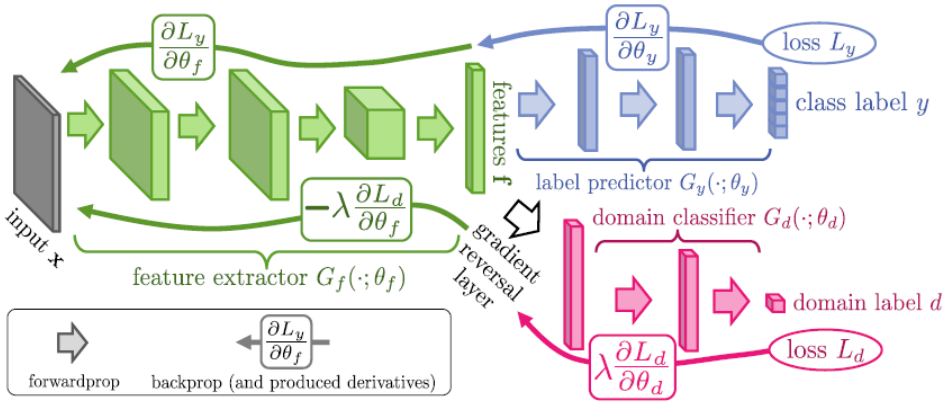

Figure 3: The proposed architecture includes a deep feature ertractor (green) and a deeplabel predictor (blue), which together form a standard feed-forward architecture.Unsupervised domain adaptation is achieved by adding a domain classifer (red)connected to the feature extractor via a gradient reversal layer that multipliesthe gradient by a certain negative constant during the backpropagation-basedtraining.Otherwise, the training proceeds standardly and minimizes the labelprediction loss (for source examples) and the domain classifcation loss (for allsamples). Gradient reversal ensures that the feature distributions over the twodomains are made similar (as indistinguishable as possible for the domain classifer), thus resulting in the domain-invariant features.

Feature Extractor:Extract shared features from the input data, which are applicable to both the Source Domain and the Target Domain. It is composed of convolutional layers (CNN) or fully connected layers, and the backbone of CNN is ResNet50.

Label Predictor:Task classification is carried out based on the extracted features, and only the source domain labels are used for supervised training.

Domain Discriminator:Determine whether the features come from the source domain or the target domain. Force the feature extractor to generate dome-invariant features through adversarial training. Gradient Reversal Layer (GRL) $\rightarrow$ Fully connected layer (ReLU) $\rightarrow$ Sigmoid output layer.

GRL:When propagating forward, the output of GRL is equal to the input. During backpropagation, GRL inverts the gradient of the domain discriminator, enabling the feature extractor to maximize the error of the domain discriminator (against the target), thereby eliminating the domain difference.

loss and Optimizer

EntropyLoss (Entropy Loss Function):This function minimizes the entropy of predicted probability distributions to encourage high-confidence predictions. It first filters out probabilities below 1e-6 for numerical stability, then computes the standard entropy formula $-\sum(p \times \log p)$ on the remaining values, and finally returns the batch-averaged entropy. Mathematically, it is expressed as $\mathcal{L}_{\text{ent}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{c=1}^{C} p_{i,c}\log p_{i,c}$, where entropy approaches 0 for near one-hot predictions and reaches its maximum ($\log C$ for uniform distributions.

DANN Loss (Domain Adversarial Loss) This function trains a domain discriminator using binary cross-entropy to make features indistinguishable between source and target domains. During implementation, features first pass through a Gradient Reversal Layer (GRL), which preserves forward outputs but multiplies gradients by λ during backpropagation. The discriminator predicts domain labels (1 for source, 0 for target), and the loss computes binary cross-entropy between predictions and true domain labels. Formally,$\mathcal{L}_{\text{adv}} = -\frac{1}{2B}\left(\sum_{i=1}^{B}\log D(f_i^s) + \sum_{j=1}^{B}\log\left(1 - D(f_j^t)\right)\right)$.

The feature extractor minimizes this loss (via GRL) to produce domain-invariant features, while the discriminator maximizes it to improve domain classification.

Use the Settings of the SGD optimizer and include the learning rate scheduling strategy (inv type).

trade off 1.
learning rate Initial value 0.0003.
batch size source batch size 36;target batch size36;test batch size 4.
training steps 1251,About 10 epochs

2.3 Result

The source domain is selected as "Art" and the target domain is selected as "real world". During the training process, approximately one epoch after every 125 batches, the model's performance was tested, and the best acc and its corresponding model were saved. Finally, the loss and acc corresponding to 1250 batches were plotted as shown in the figure4.It can be observed from the figure that the training loss has decreased from 4.9 to 1. The accuracy rate of the test set can reach up to 67%.

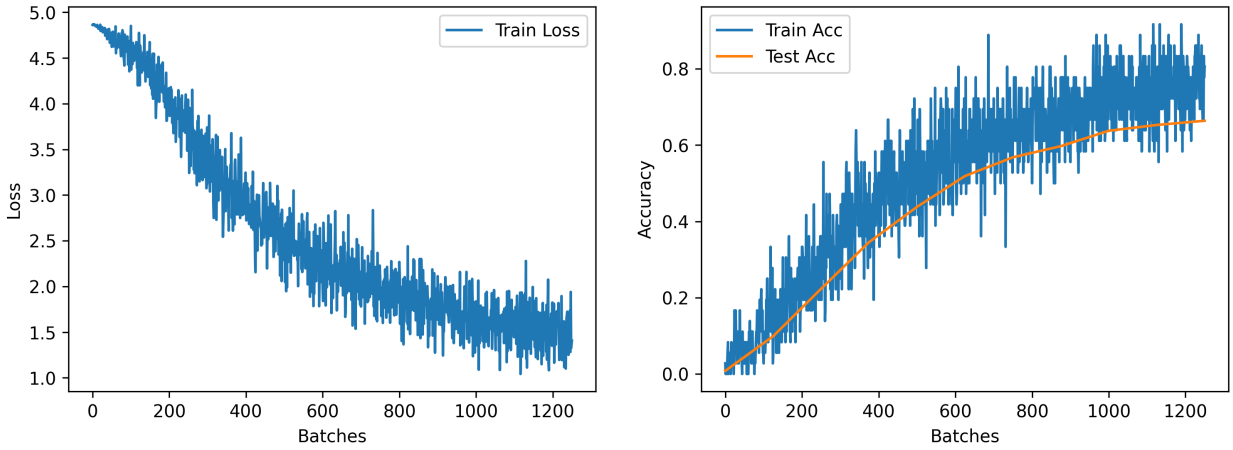

Figure 4: Loss and Accuracy of DANN

3 [30 points] Unpaired Image-to-Image Translation (CycleGAN) for Unsupervised Style Transfer

3.1 Loss function

The loss function in CycleGAN consists of three main components: adversarial loss, cycle-consistency loss, and identity loss.

- Adversarial Loss (GAN Loss):To ensures generated images match the target domain distribution. For generator $G : X \to Y$ and discriminator D_Y:

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x)))]$$

  Symmetric loss for $F : Y \to X$ and D_X.

- Cycle-Consistency Loss: To preserves consistency between original and reconstructed images (i.e., $x \to G(x) \to F(G(x)) \approx x$).

$$\mathcal{L}_{\text{cycle}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]$$

- Identity Loss: To encourages generators to preserve color/tone when inputs already belong to the target domain

$$\mathcal{L}_{\text{identity}}(G, F) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(y) - y\|_1] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(x) - x\|_1]$$

- Total Loss Function:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{GAN}}(G, D_Y) + \mathcal{L}_{\text{GAN}}(F, D_X) + \lambda \mathcal{L}_{\text{cycle}}(G, F) + \beta \mathcal{L}_{\text{identity}}(G, F)$$

Hyperparameters: • λ (default: 10): Controls cycle-consistency importance. • β (default: 0.5 or 0): Weight for identity loss.

3.2 implementation

network structure

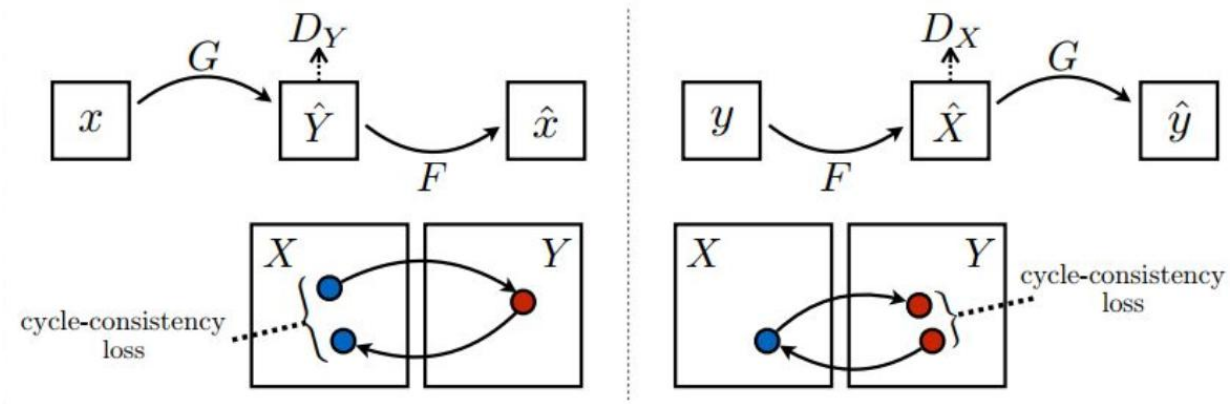

Figure 5: Architecture of CycleGAN

Generators (G and F):

- Architecture:ResNet-based encoder-decoder with residual blocks.
  - Encoder:Downsamples the input image via convolutional layers.
  - Transformer:Contains 6–9 residual blocks (for 256x256 images) to capture hierarchical features while preserving spatial information.
  - Decoder:Upsamples features to reconstruct the output image using transposed convolutions or interpolation.
  - purpose:enerator G maps images from domain X to Y;Generator F maps images from domain Y to X.

Discriminators (DX and DY)

- Architecture:PatchGAN(Markovian discriminator).
  - Processes overlapping local image patches (e.g., 70x70 pixels per patch).
  - Outputs a matrix of probabilities indicating whether each patch is real or fake.
  - purpose:DY distinguishes real images in domain Y from fake ones generated by G;DX distinguishes real images in domain X from fake ones generated by F.

Generators and discriminators are trained alternately.
Optimizer Adam Optimizer.
Hyperparameters $\lambda = 10$,$\beta = 5$.
learning rate Initial value 0002.
batch size batch size 1.
training steps epochs=35.

3.3 Result

I choose product as domain A and Art as domain B. the loss decay (adversarial loss and cycle consistency loss) over the training process is shown in Figures 5 and 6. The training performance is shown in Figures 7 and 8.

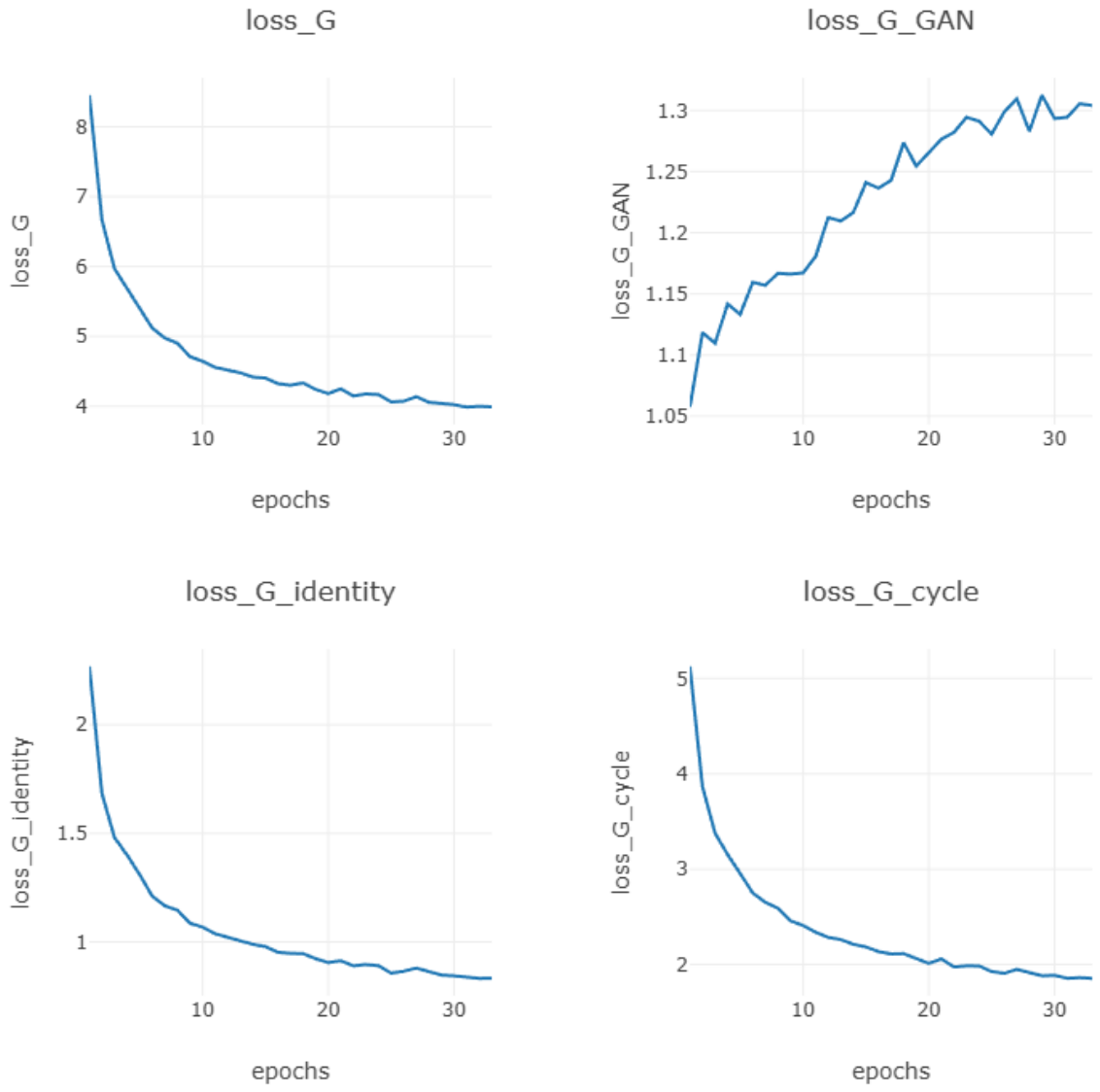

Figure 6: combine loss of GenerateAtoB and GenerateBtoA

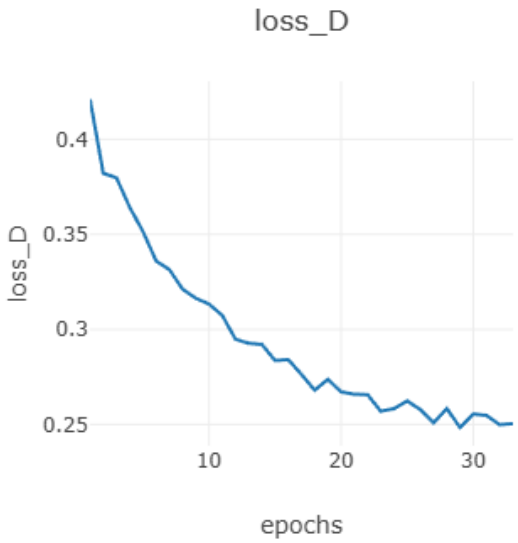

Figure 7: combine loss of DiscriminatorB and DiscriminatorA

examples of style-transferred images.

Figure 8: Product to Art

Figure 9: Art to Product

4 [30 points] Cycle-consistent Adversarial Learning(CyCADA) for Domain Adaptation

4.1 Loss function

CyCADA (Cycle-Consistent Adversarial Domain Adaptation) integrates CycleGAN's cycle-consistent adversarial loss and DANN's (Domain-Adversarial Neural Network) domain adaptation loss to achieve multi-level cross-domain alignment.

DANN's Domain-Adversarial Loss: Feature-Level Alignment

- Goal:Minimize feature distribution differences between the labeled source domain and unlabeled target domain.
- Implementation:
  - A domain discriminator (D) is trained adversarially to distinguish between source and target features.
  - Loss Function

$$L_{\text{domain}} = \mathbb{E}_{x\sim\text{source}}[\log D(f(x))] + \mathbb{E}_{x\sim\text{target}}[\log(1 - D(f(x)))]$$

    * Feature extractor (f) minimizes L_{domain} (fools the discriminator).
    * Discriminator (D) maximizes L_{domain} (correctly classifies domains).

CycleGAN's Adversarial Cycle-Consistency Loss: Pixel-Level Alignment:

- Goal:Transform source images into the target domain's style while preserving semantic content.
- Implementation:
  - Adversarial Loss (GAN Loss):
    * Generator $G_{S \hookrightarrow T}$ converts source images to the target domain.
    * Discriminator D_T distinguishes real vs. generated target images.

$$L_{\text{adv}} = \mathbb{E}_{x \sim \text{target}}[\log D_T(x)] + \mathbb{E}_{x \sim \text{source}}[\log(1 - D_T(G_{S \hookrightarrow T}(x)))]$$

  - Cycle-Consistency Loss:Ensures that translating an image back to its original domain preserves content:

$$L_{\text{cycle}} = \mathbb{E}_{x \sim \text{source}}[\|G_{T \hookrightarrow S}(G_{S \hookrightarrow T}(x)) - x\|_1] + \mathbb{E}_{x \sim \text{target}}[\|G_{S \hookrightarrow T}(G_{T \hookrightarrow S}(x)) - x\|_1]$$

Combined Total Loss Function: CyCADA's final loss combines all components with balancing weights: ($\lambda_{\text{domain}}, \lambda_{\text{cycle}}, \lambda_{\text{adv}}$):

$$L_{\text{total}} = \underbrace{L_{\text{task}}}_{\text{Task Loss}} + \lambda_{\text{domain}} \underbrace{L_{\text{domain}}}_{\text{Feature Alignment}} + \lambda_{\text{cycle}} \underbrace{L_{\text{cycle}}}_{\text{Semantic Consistency}} + \lambda_{\text{adv}} \underbrace{L_{\text{adv}}}_{\text{Image Alignment}}$$

Key Synergies

- Dual-Level Alignment:
  - DANN (Feature-Level): Aligns high-level semantic features (e.g., object boundaries).
  - CycleGAN (Pixel-Level): Aligns low-level visual styles (e.g., textures, lighting).
  - Cycle-Consistency: Preserves content during translation.
- Task-Driven Adaptation:
  - L_{task} (e.g., classification loss) ensures source labels remain useful.
  - Generated target-style images improve generalization.
- Dynamic Weighting:
  - High $\lambda_{\text{domain}} \rightarrow$ Strong feature alignment (useful for semantic shifts).
  - High $\lambda_{\text{adv}} \rightarrow$ Strong style transfer (useful for visual shifts).

4.2 implementation

network structure
CyCADA integrates **feature-level alignment**, **pixel-level adaptation**, and **semantic consistency** through the following components:

- **Feature Extractor** (G_f):
  - Input: Source image X_s or target image X_t.
  - Output: Feature maps $f_s = G_f(X_s)$ or $f_t = G_f(X_t)$.
  - Architecture: Standard CNN (e.g., ResNet).
- **Classifier** (G_c):
  - Input: Features f_s or f_t.
  - Output: Class probabilities $G_c(f_s)$.
  - Loss: Supervised cross-entropy on labeled source data:

$$\mathcal{L}_{\text{task}} = \mathbb{E}_{(X_s, y_s)}\left[-\log G_c(G_f(X_s))\right] \tag{1}$$

- **Domain Discriminator** (G_d):
  - Input: Features f_s or f_t.
  - Output: Domain probability $G_d(f)$ (0: source, 1: target).
  - Loss: Adversarial loss with Gradient Reversal Layer (GRL):

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{X_s}\left[\log G_d(G_f(X_s))\right] + \mathbb{E}_{X_t}\left[\log(1 - G_d(G_f(X_t)))\right] \tag{2}$$

- **Pixel-Level Generators**:
  - $G_{s\to t}$: Translates X_s to target domain.
  - $G_{t\to s}$: Translates X_t to source domain.
  - Discriminator D_t: Distinguishes real vs. translated target images.
  - Cycle-consistency loss:

$$\mathcal{L}_{\text{cyc}} = \mathbb{E}_{X_s}\left[\|G_{t\to s}(G_{s\to t}(X_s)) - X_s\|_1\right] + \mathbb{E}_{X_t}\left[\|G_{s\to t}(G_{t\to s}(X_t)) - X_t\|_1\right] \tag{3}$$

  - GAN loss for pixel alignment:

$$\mathcal{L}_{\text{GAN}} = \mathbb{E}_{X_t}\left[\log D_t(X_t)\right] + \mathbb{E}_{X_s}\left[\log(1 - D_t(G_{s\to t}(X_s)))\right] \tag{4}$$

- **Semantic Consistency Loss**:

$$\mathcal{L}_{\text{sem}} = \mathbb{E}_{X_s}\left[\|G_c(G_f(X_s)) - G_c(G_f(G_{s\to t}(X_s)))\|_2\right] \tag{5}$$

Total Objective

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{cyc}}\mathcal{L}_{\text{cyc}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}} \tag{6}$$

where λ_{adv}, λ_{cyc}, λ_{sem}, λ_{GAN} are hyperparameters.
Optimizer Adam Optimizer.
learning rate Initial value 0002.
batch size batch size 64.
training steps pretrain epoch=50,epochs=100.

4.3 Result

The source domain is selected as "Art" and the target domain is selected as "real world". Train phase:

Figure 10: Loss

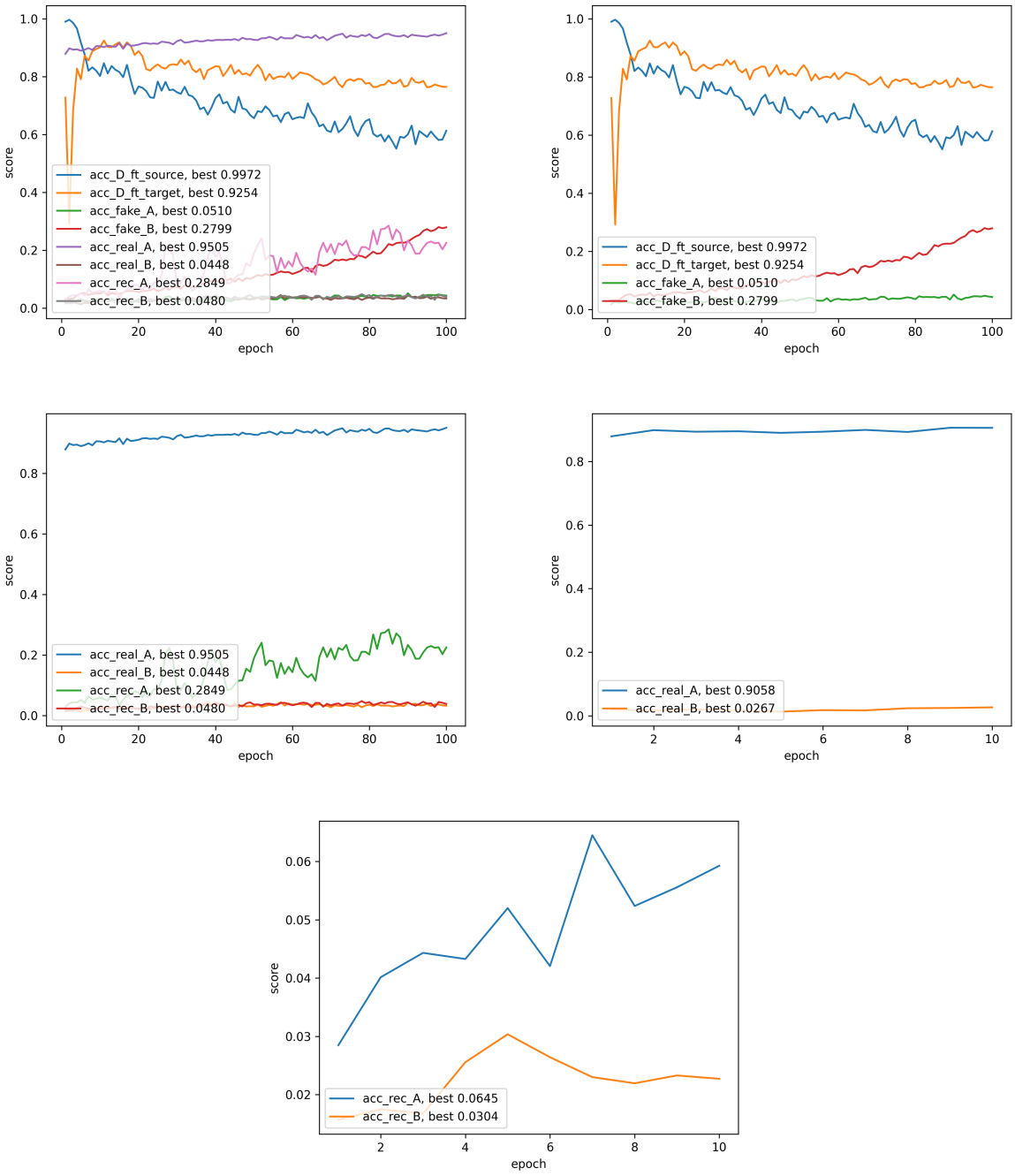

Figure 11: scores

Test phase:

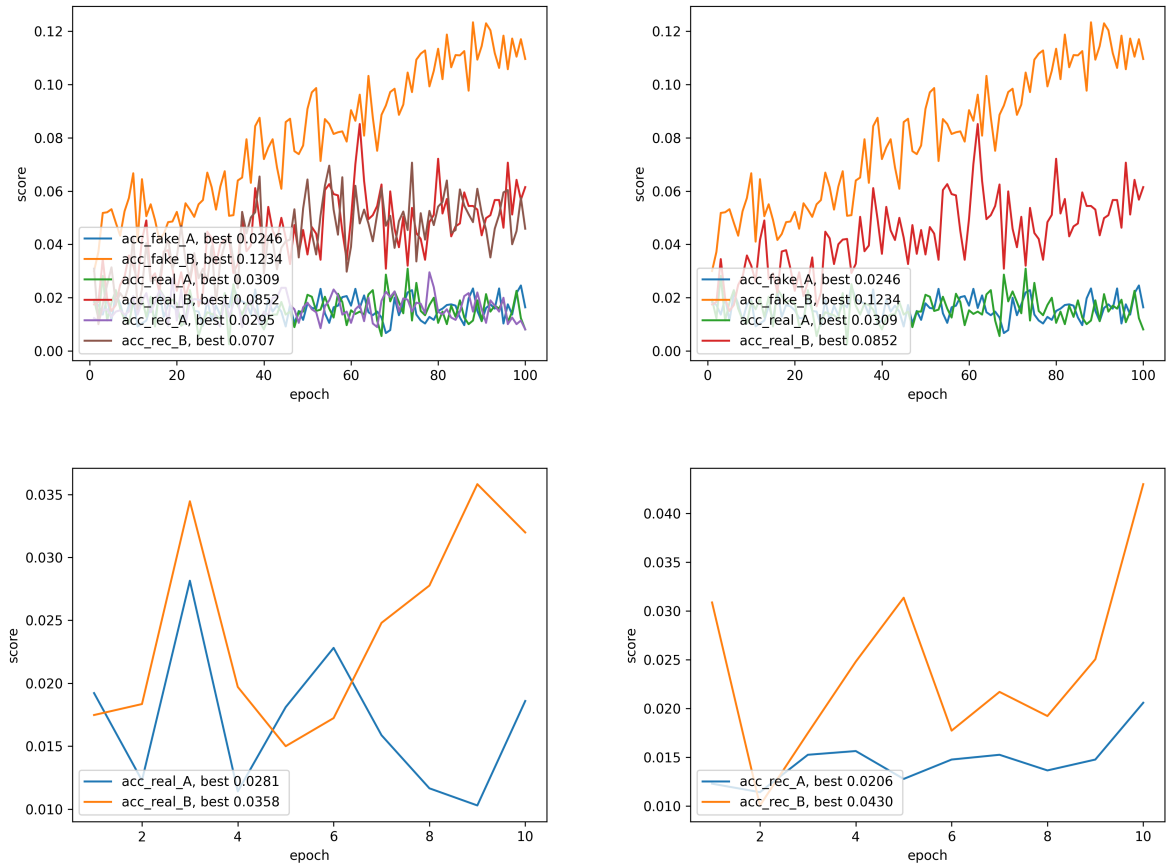

Figure 12: scores