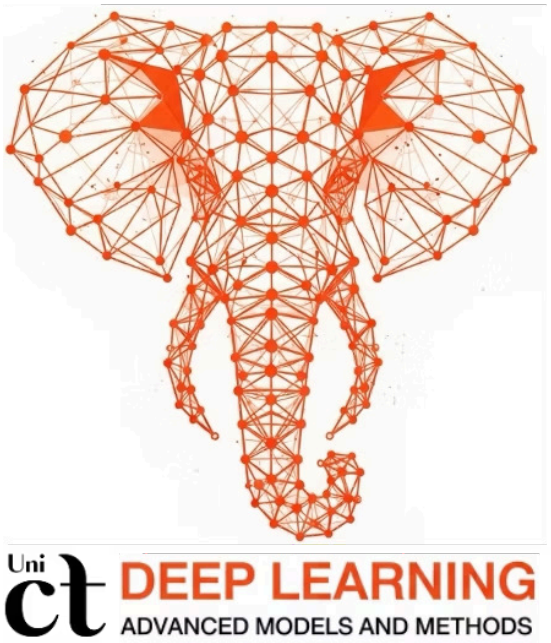
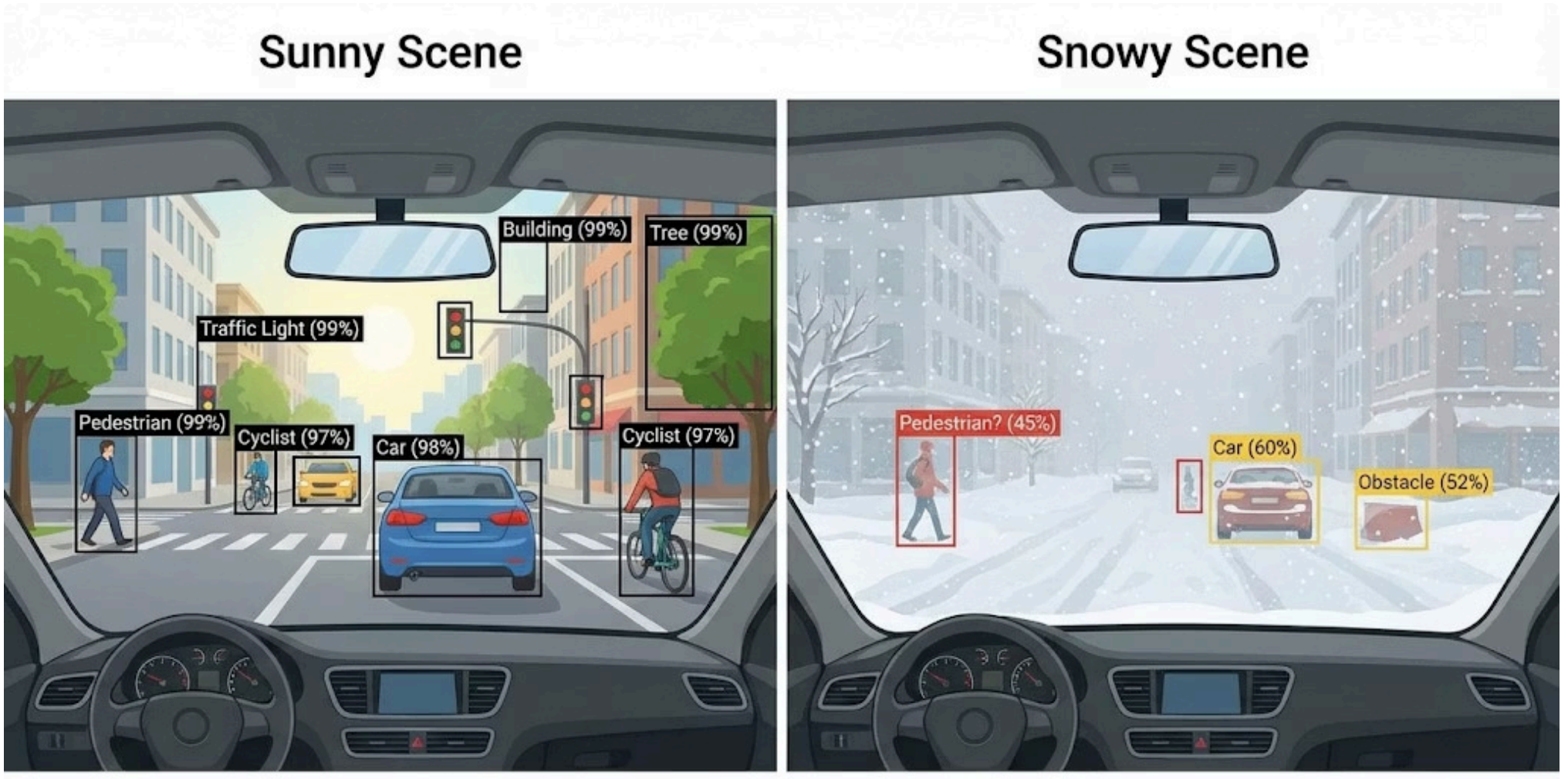


Deep Learning
Advanced Models & Methods

Prof. Antonino Furnari (antonino.furnari@unict.it)
Corso di Laurea Magistrale in Informatica
Dip. di Matematica e Informatica
Università di Catania

Deep Domain Adaption

The Motivation: Why Domain Adaptation?



The Challenge

Deep learning models excel on training data but struggle with distribution shifts. An object detection model trained on sunny scenes may fail entirely on snowy scenes.

One solution is to **collect data and labels in the new domain** and fine-tune. While this generally work, it is **expensive**.

The Solution

Domain adaptation leverages a crucial insight: **while the data distribution shifts, the underlying task remains constant.** It **transfers structural knowledge** from a labeled source domain to an unlabeled target domain, enabling **generalization across varying environments**.

Defining the Players: Source vs. Target Domains

Source Domain (D_s)

The source domain contains abundant labeled training data: $\{(x_i^s, y_i^s)\}_{i=1}^{N_s}$

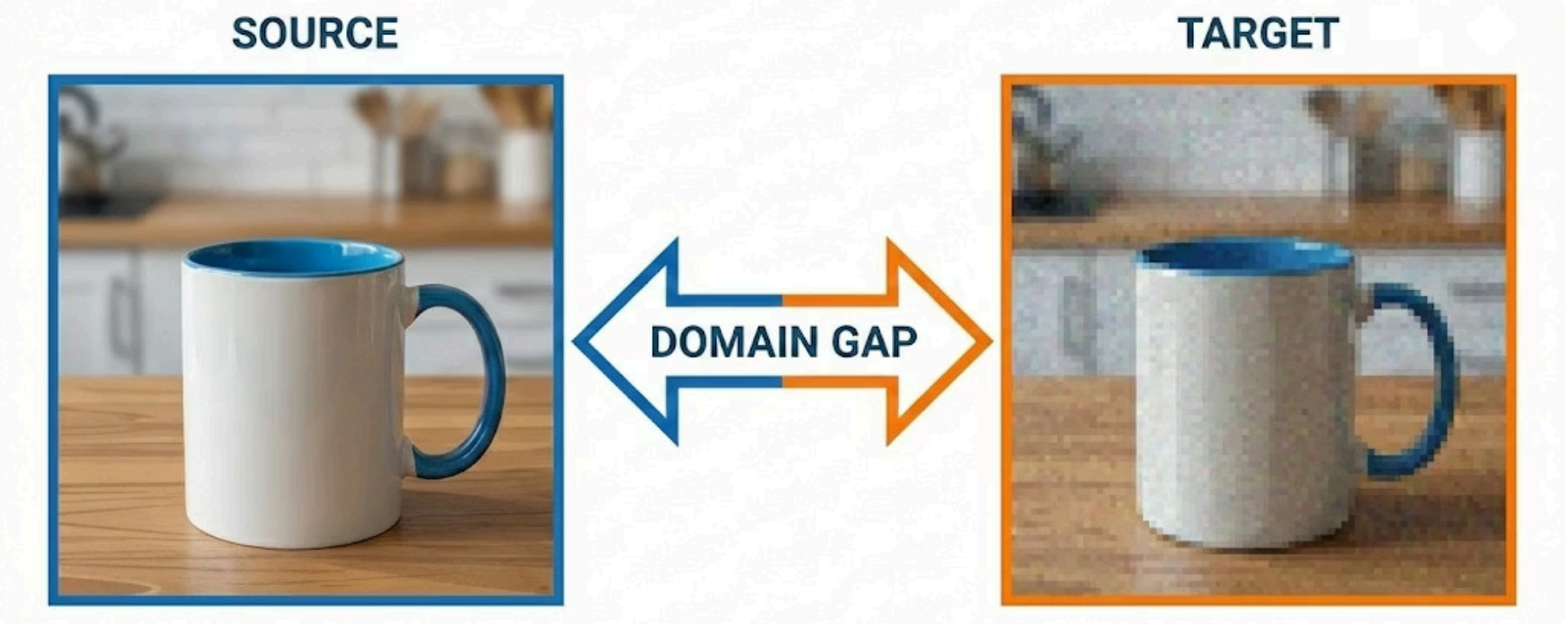
- High-quality annotations typically available
- Often collected under controlled conditions
- Example: Professional product photography for training a model

Target Domain (D_t)

The target domain contains data, which may or may not be labeled: $\{(x_j^t)\}_{j=1}^{N_t}$ or $\{(x_j^t, y_j^t)\}_{j=1}^{N_t}$

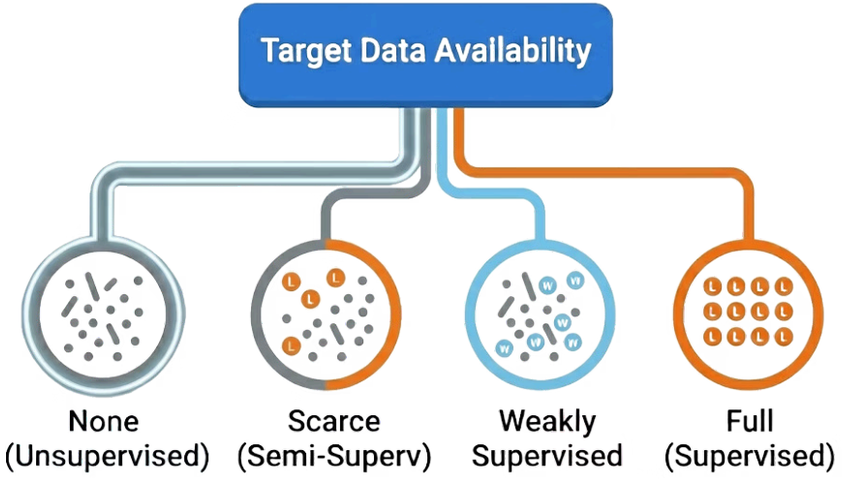
- Can have limited or no manual annotations
- Reflects real-world deployment conditions
- Example: Consumer webcam photos where the model will be deployed

The **domain gap** usually refers to the visual and statistical differences between these distributions—variations in lighting, texture, resolution, or sensor characteristics—despite the underlying semantic content remaining identical. A coffee mug photographed professionally versus captured by a webcam represents the same object class, yet exhibits drastically different visual features between D_s and D_t.



The Taxonomy of Supervision

The availability of target domain labels fundamentally determines which adaptation methods are applicable. This course focuses primarily on the most challenging and practical scenario: **unsupervised domain adaptation**.



Unsupervised DA (UDA)

Source labeled, Target completely unlabeled ($Y_t = \emptyset$). This represents the most common real-world scenario and the primary focus of this lecture.

Semi-Supervised DA

Target domain has sparse labels (typically <5% of data). Combines supervised learning signals with unsupervised adaptation techniques.

Supervised DA

Target domain is fully labeled. Reduces to standard transfer learning or fine-tuning approaches. Less challenging but resource-intensive.

Weakly Supervised DA

Target has noisy, incomplete, or tag-based annotations (e.g., image-level tags rather than pixel-level masks).

Wang, M., & Deng, W. (2018). "Deep visual domain adaptation: A survey." *Neurocomputing*.

Taxonomy of Domain Shift: A Recap

Understanding how data distributions change between source and target domains is crucial for selecting appropriate adaptation strategies. We identify three primary types of shift:

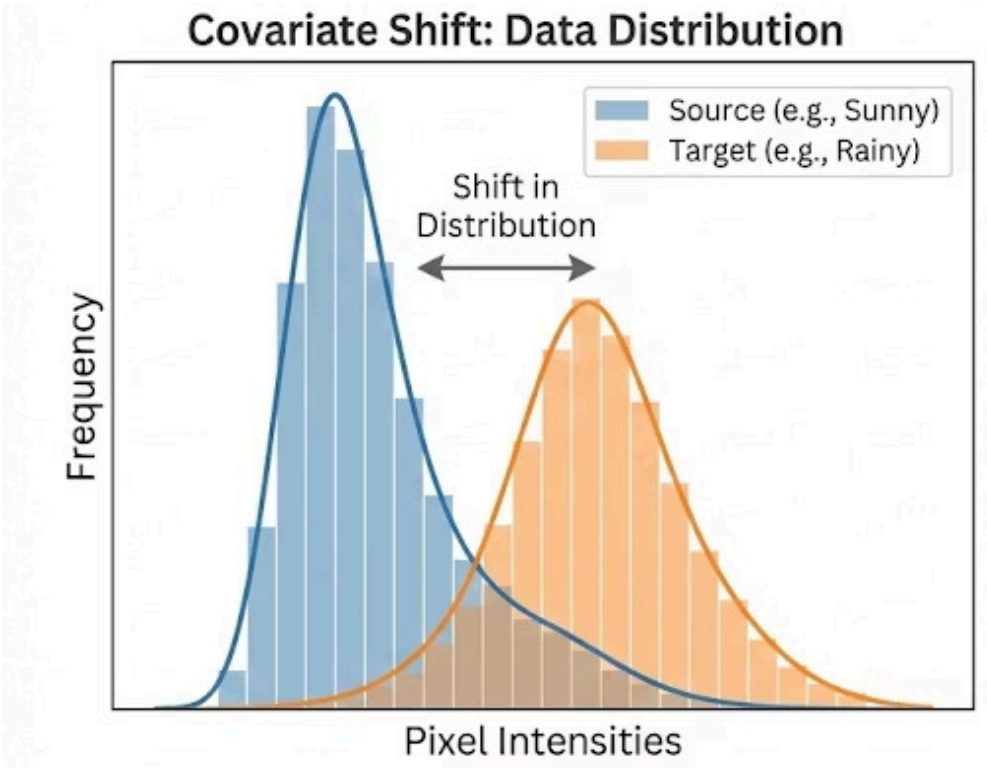
Covariate Shift: $P(X)$ Changes

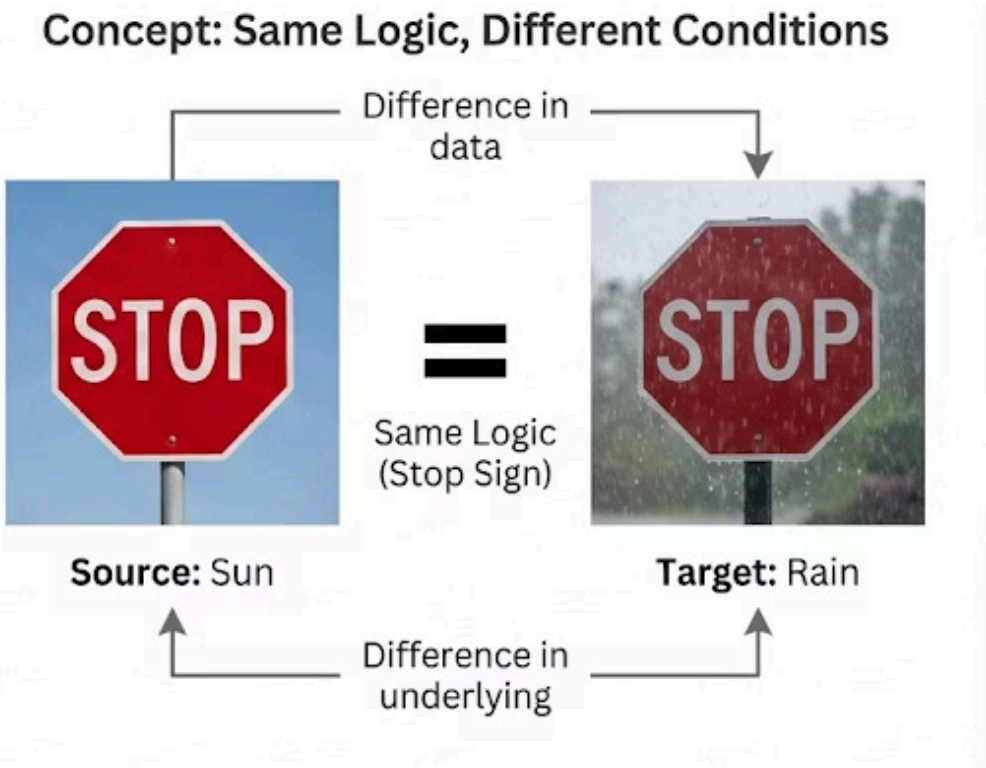
Label Shift: $P(Y)$ Changes

Concept Shift: $P(Y|X)$ Changes

We look at these more in details int he next slides.

Taxonomy of Shift I: Covariate Shift





Mathematical Definition

Covariate shift occurs when the input distribution changes between domains, but the conditional relationship between inputs and labels remains constant:

$$P_s(X) \neq P_t(X) \text{ but } P_s(Y|X) = P_t(Y|X)$$

This is the most common type of distribution shift in practice. The marginal distribution of features $P(X)$ differs, but the decision boundary—the fundamental mapping from inputs to outputs—stays unchanged.

Concrete Example: Autonomous Driving

Consider training a perception system in sunny California versus deploying in rainy London. The pixel intensity distributions differ dramatically (bright vs. dark, high vs. low contrast), yet a pedestrian remains visually identifiable as a pedestrian regardless of weather conditions.

Adaptation Strategy

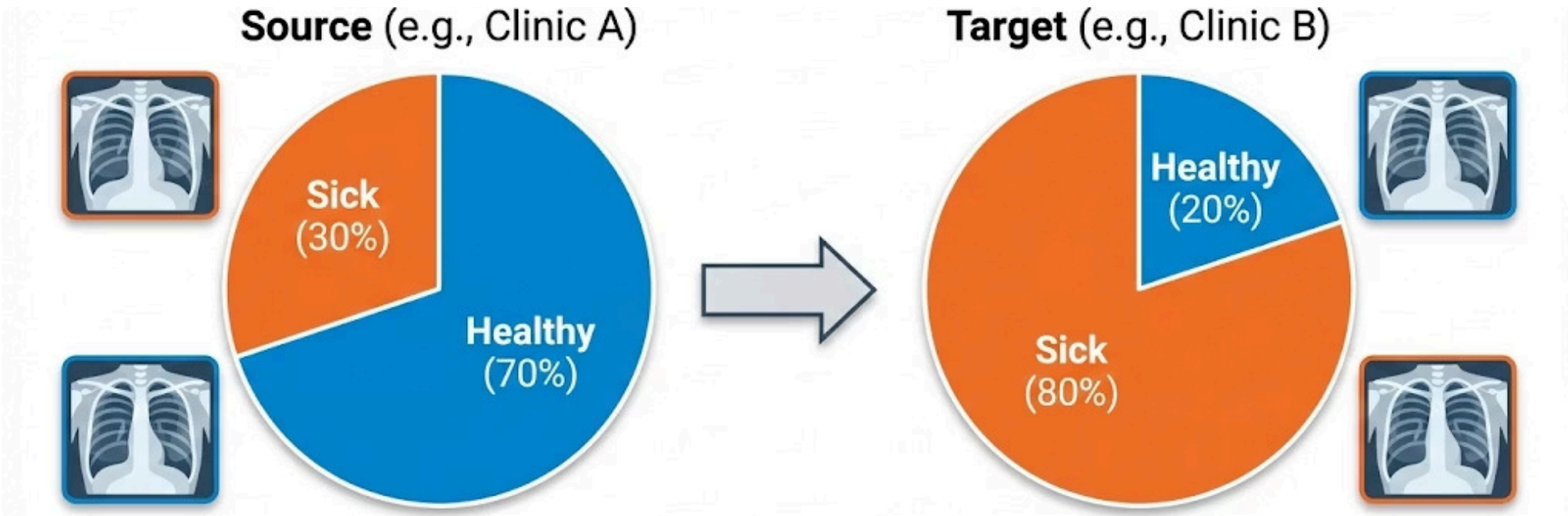
The classifier learned on source data can theoretically transfer perfectly if we can align the input distributions—this insight motivates discrepancy-based adaptation methods. In practice, we want to find a mapping ϕ such that:

$$P_s(\phi(X)) \approx P_t(\phi(X))$$

Then classify with:

$$P(Y|\phi(X))$$

Taxonomy of Shift II: Label Shift



Mathematical Definition

Label shift (also called *prior shift*) occurs when the class distribution changes while the class-conditional feature distributions remain stable:

$$P_s(Y) \neq P_t(Y) \text{ but } P_s(X|Y) = P_t(X|Y)$$

Concrete Example: Medical Diagnosis

Training (Source): General population screening with 90% healthy patients, 10% diseased.

Deployment (Target): Pandemic hospital setting with 40% healthy, 60% diseased.

The visual appearance of disease on medical scans ($X|Y$) remains unchanged—a pneumonia infiltrate looks identical. However, the prior probability of encountering each class has shifted dramatically.

Adaptation Strategy

Label shift requires reweighting techniques. Importance weighting by the ratio $P_t(Y)/P_s(Y)$ can correct for prior mismatch without retraining the feature extractor.

Taxonomy of Shift III: Concept Shift

The Hardest Challenge

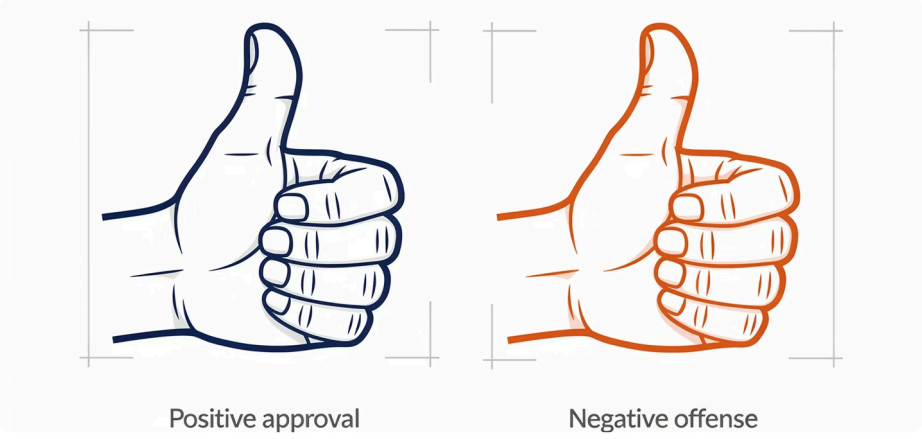
Concept shift represents the most fundamental type of distribution change. Here, the *meaning* of the input itself changes across domains:

$$P_s(Y|X) \neq P_t(Y|X)$$

Unlike covariate or label shift, no amount of distribution alignment can resolve concept shift—the relationship between features and labels has fundamentally changed. This violates the core assumption of domain adaptation.

Cultural Example

The "thumbs up" gesture (X) maps to "approval" (Y_s) in Western cultures but can signal "offense" (Y_t) in parts of the Middle East. The visual input is identical; the semantic output differs completely.



Adaptation Strategy

⊗ True concept shift cannot be solved by standard DA methods. Domain adaptation assumes task consistency—when this breaks down, we need task adaptation or multi-task learning approaches instead.

Scenarios I: Closed Set Domain Adaptation

Closed set domain adaptation represents the standard and most well-studied scenario in the field. The key assumption: source and target domains share an **identical label space**.

Mathematical Formulation

The class sets are equivalent: $C_s = C_t$

Both domains contain exactly the same categories, with no unknown or missing classes.

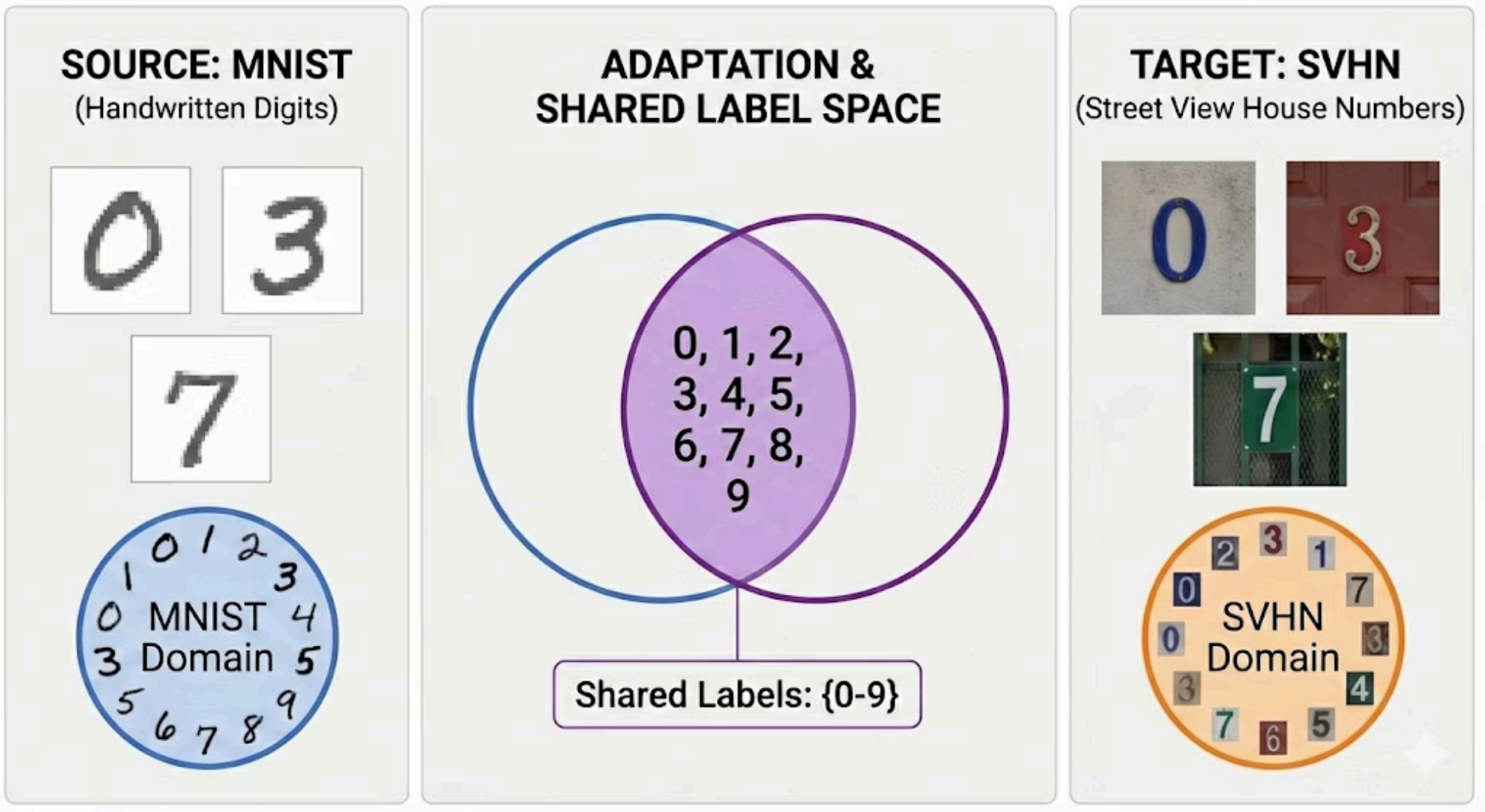
Canonical Example

MNIST → SVHN: Adapting handwritten digit recognition (digits 0-9) to street view house numbers (digits 0-9).

The label space remains {0, 1, 2, ..., 9} in both cases. Only the visual rendering differs—handwritten strokes versus natural scene text.

The Challenge

Despite identical semantics, **dramatic visual style differences create the domain gap**: stroke thickness, background clutter, perspective distortion, and color variations all change between domains.



This scenario enables clean theoretical analysis and serves as the foundation for most domain adaptation algorithms. The vast majority of methods—discrepancy-based, adversarial, and self-training approaches—assume closed set conditions.

Scenarios II: Partial Domain Adaptation

Mathematical Formulation

Partial domain adaptation relaxes the closed set assumption by considering cases where the target contains a **subset** of source classes:

$$C_t \subset C_s$$

Example: Adapting from large datasets

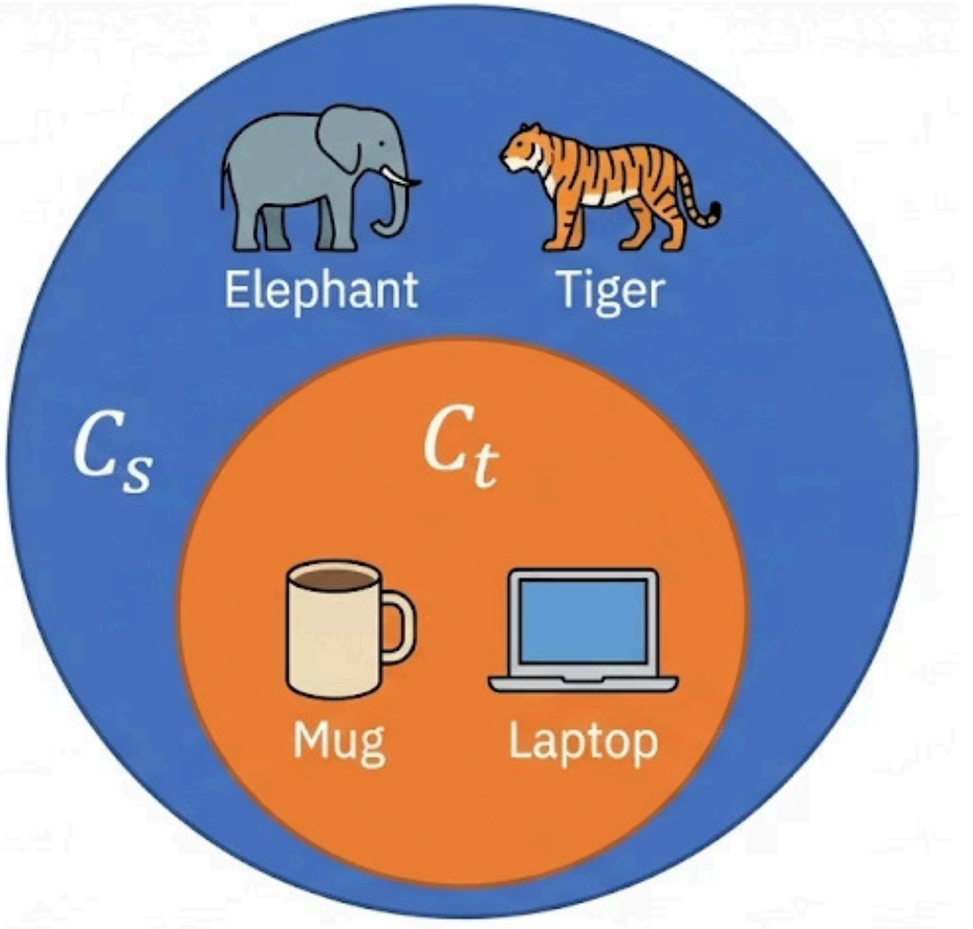
This arises naturally when adapting from large-scale pre-trained models (ImageNet with 1,000 classes) to specialized target tasks (office objects with 10 classes).

The Negative Transfer Problem

The critical challenge: **negative transfer** from irrelevant source classes. Standard adaptation methods attempt to align all source samples with target samples. When 990 of 1,000 source classes are absent in the target, this alignment corrupts the feature space.

Adaptation Strategy

- **Weighting:** Down-weight outlier source samples during adaptation
- **Filtering:** Explicitly detect and exclude irrelevant classes
- **Progressive:** Gradually expand the set of transferred classes



Cao, Z., et al. (2018). "Partial adversarial domain adaptation." *ECCV*.

Scenarios III: Open Set Domain Adaptation

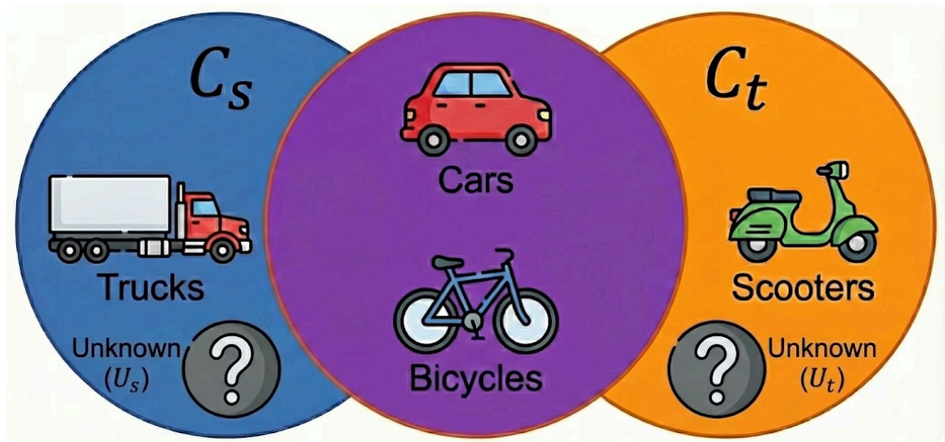
Open set domain adaptation addresses the realistic scenario where the target domain contains **unknown classes** not present in the source—the most challenging and practical setting.

Mathematical Setup

The label spaces partially overlap with additional target classes:

$$C_t \not\subseteq C_s \text{ and } C_t \cap C_s \neq \emptyset$$

Both domains contain "unknown" elements not belonging to any of the classes of interest.



Example

Source trains on {Cars, Bicycles, Trucks, Unknown}; Target encounters {Cars, Bicycles, *Scooters,* Unknown}.

The Unknown Detection Challenge

The model must learn a **reject option**—confidently stating "I don't know" when encountering novel classes. Naive adaptation would force-fit scooters into the car or bicycle category, producing silent failures.

Safety-Critical Implications

Open set DA is essential for deployment safety. Autonomous vehicles, medical diagnosis systems, and security applications cannot assume a closed world. They must detect distributional novelty.

Panareda Busto, P., & Gall, J. (2017). "Open set domain adaptation." *ICCV*.

MODULE 2 DISCREPANCY METHODS

Discrepancy-Based Approaches

Measuring & Minimizing Distribution Distance

The discrepancy-based strategy rests on a simple but powerful idea: if we can *measure* the statistical distance between source and target distributions, we can explicitly minimize it during training.

The Discrepancy Strategy

Core Principle

Jointly optimize two objectives:

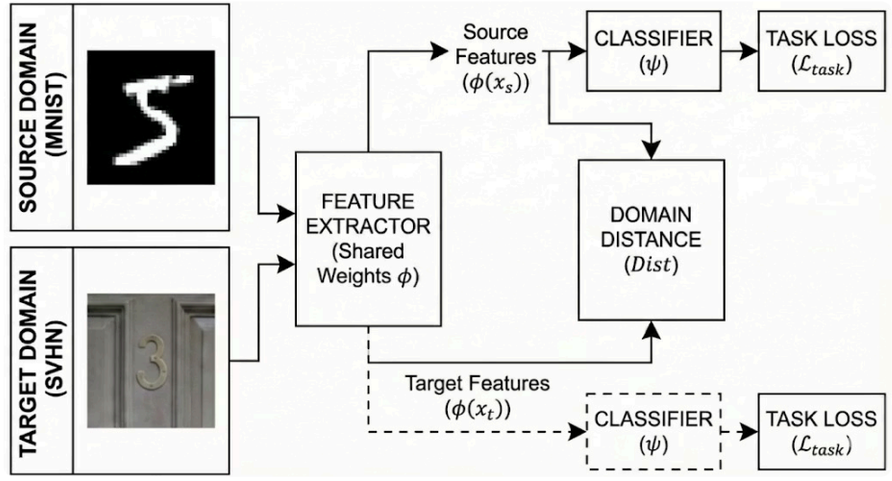
1. **Classification Loss:** Standard supervised learning on labeled source data
2. **Distribution Alignment:** Minimize distance between source and target feature distributions:

$$L_{total} = L_{task}(\psi(\phi(X_s)), Y_s) + L_{task}(\psi(\phi(X_t)), Y_t) + \lambda \cdot Dist(\phi(X_s), \phi(X_t))$$

Where ϕ is a feature extractor, ψ is the task head (e.g., a classifier), and Dist is the distance function. The hyperparameter λ controls the trade-off between task performance and domain invariance.

In practice, models use variations of this loss function. The task term on the target features is applied only when target labels are available (e.g., semi-supervised domain adaptation).

General Architecture



Key Distance Metrics

- **MMD (Maximum Mean Discrepancy):** First-order moment matching in kernel space
- **CORAL:** Second-order moment matching (covariance alignment)
- **Wasserstein Distance:** Optimal transport between distributions

Recap: Kernel Methods for Point Similarity

Kernel methods provide a powerful way to measure the similarity between data points and even entire distributions, by implicitly mapping them into a higher-dimensional feature space where linear relationships become apparent.

The Kernel Trick: General Formulation

A kernel function $k(x, y)$ computes the dot product of data points x and y in a high-dimensional feature space $\mathcal{H}$ (a Reproducing Kernel Hilbert Space) without explicitly performing the mapping $\phi : x \to \phi(x)$.

$$k(x, y) = \langle \phi(x), \phi(y) \rangle$$

This allows algorithm to operate on data that may behave non-linear (e.g., non-linearly separable) in the original space, providing a measure of similarity that goes beyond simple Euclidean distance.

Gaussian Radial Basis Function (RBF) Kernel

The Gaussian RBF kernel is one of the most widely used kernels, defining similarity based on proximity in the input space. Its formula is:

$$k(x, y) = \exp\left(-\gamma \|x - y\|^2\right)$$

Here, $\|x - y\|^2$ is the squared Euclidean distance between points, and $\gamma > 0$ is a bandwidth parameter. A larger γ means points must be very close to be considered similar, while a smaller γ gives a broader influence.

In a 1D visualization, the Gaussian RBF kernel assigns a high similarity score (close to 1) when points are near each other and a low score (close to 0) when they are far apart, acting like a smooth "bump" function around each data point.

We can see the process of measuring similarity as placing a Gaussian around one point and seeing where it intersect the other point.

Maximum Mean Discrepancy (MMD)

Maximum Mean Discrepancy provides a powerful non-parametric test for determining whether two samples originate from the same distribution, without requiring explicit density estimation.

Mathematical Formulation

MMD measures the distance between mean embeddings in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$. For a given kernel function $k(x, y)$, the squared MMD can be expressed directly:

$$MMD^2(P_s, P_t) = \mathbb{E}_{x,x' \sim P_s}[k(x, x')] - 2\mathbb{E}_{x \sim P_s, y \sim P_t}[k(x, y)] + \mathbb{E}_{y,y' \sim P_t}[k(y, y')]$$

Given finite samples $X_s = \{x_i\}_{i=1}^n$ from P_s and $X_t = \{y_j\}_{j=1}^m$ from P_t, an unbiased empirical estimator for MMD^2 is:

$$MMD_u^2(X_s, X_t) = \frac{1}{n(n-1)} \sum_{i \neq j} k(x_i, x_j) - \frac{2}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} k(x_i, y_j) + \frac{1}{m(m-1)} \sum_{i \neq j} k(y_i, y_j)$$

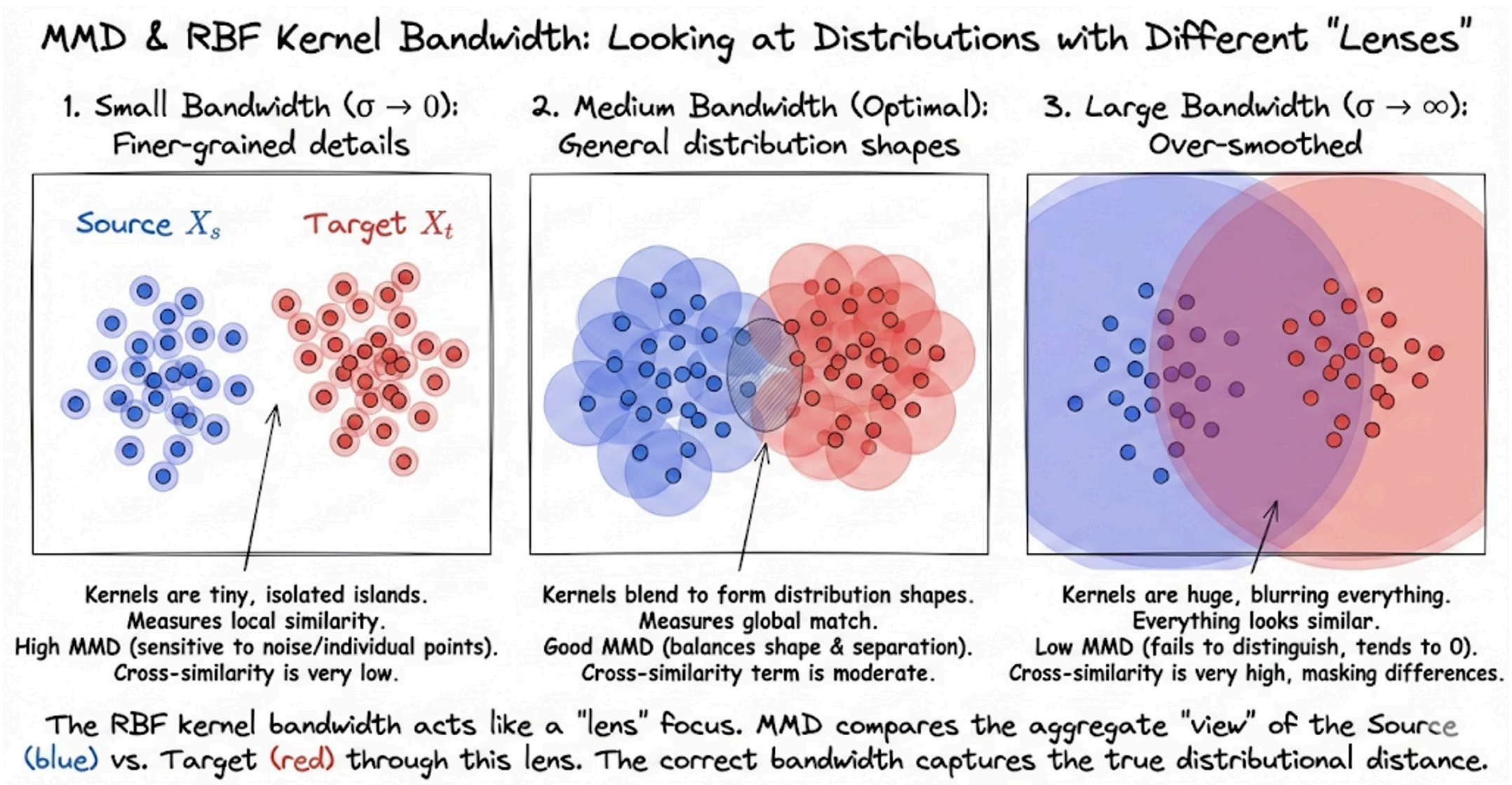
Interpretation

We can see the kernel as a function measuring a sort of distance between two data points. In practice, the MMD has three terms:

- **First term**: self-similarity between elements in the source domain (a sort of measure of variance)
- **Second term**: cross-similarity between elements in the source and target domain (interaction between the two distributions)
- **Third term**: self-similarity between elements in the target domain (a sort of measure of variance)

Intuitively, if the two distributions are overlapped, the interaction term will be large (similarities of pairs of points are large) compared to the self-similarities (measuring the variance of the distributions).

The kernel measures "how" the similarity is computed. For instance RBF kernels with low bandwidth will look at general distribution shapes, while higher bandwidth will look at finer-grained details. This also connects to Kernel Density Estimation.



Properties

Universality Property

For universal kernels (kernels with given mathematical properties, such as the Gaussian RBF), MMD is a **metric** on probability distributions:

$$MMD(P_s, P_t) = 0 \iff P_s = P_t$$

Zero discrepancy implies identical distributions.

Computational Advantages

- Symmetric, unlike KL divergence
- Works with finite samples
- No explicit density estimation required
- Unbiased empirical estimator available

Method: Deep Adaptation Networks (DAN)

Deep Adaptation Networks (DAN) pioneered the integration of MMD into deep learning architectures, enabling end-to-end adaptation through multi-layer feature alignment.

Architecture Design

Shared Backbone (AlexNet): The backbone is entirely shared. Early convolutional layers are frozen, preserving low-level features (edges, textures) that transfer well across domains. Later layer are fine-tuned but not adapted.

Adaptation Layers: The final fully-connected layers are task-specific. DAN applies **Multi-Kernel MMD (MK-MMD)** loss to these high-level representations.

Why Multiple Kernels?

Why Multiple Kernels? A single Gaussian kernel with fixed bandwidth σ may fail to capture multi-scale patterns. Different bandwidths capture different "scales" of distribution differences:

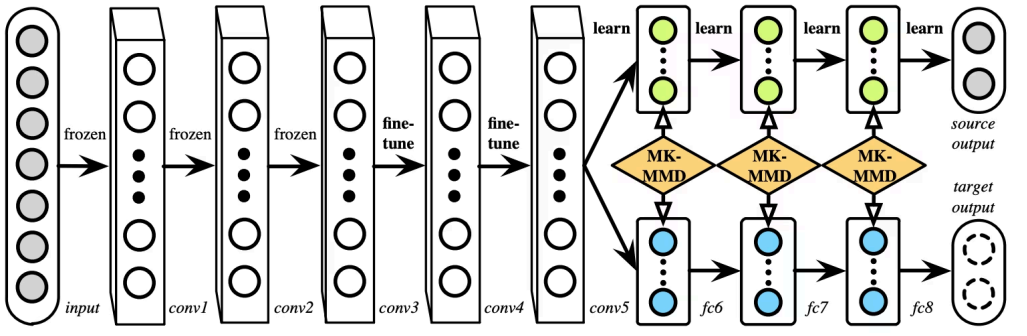
- Small σ: detects fine-grained mismatches (high granularity)
- Large σ: detects coarse-grained mismatches (low granularity)

DAN combines 5 kernels with bandwidths {0.25, 0.5, 1, 2, 4} to ensure robustness:

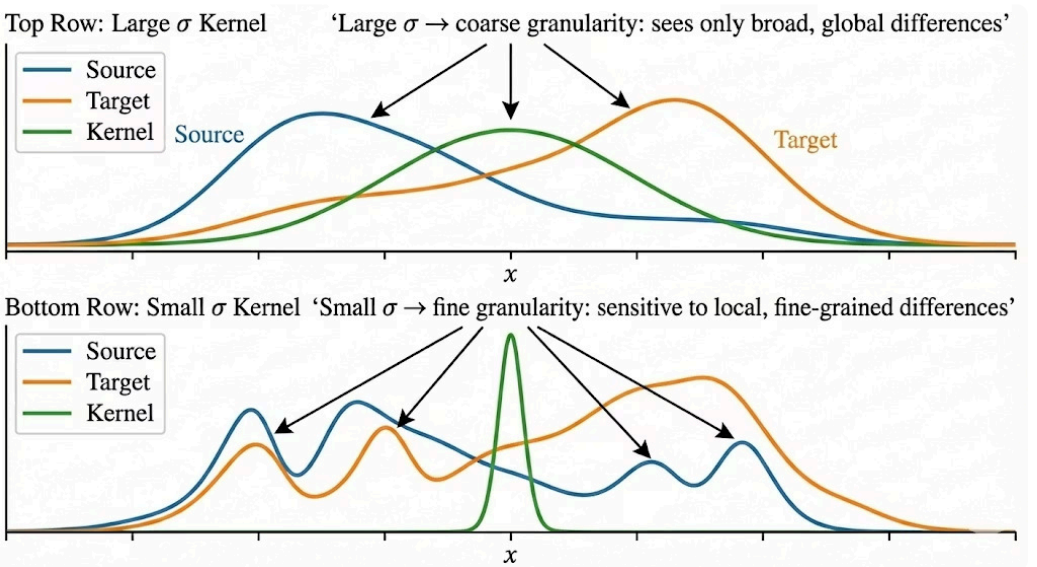
$$MK - MMD = \sum_{k=1}^{5} MMD_{\sigma_k}^2$$

In practice, DAN uses a linear-time unbiased estimator of MMD and automatically learns non-negative weights over multiple Gaussian kernels, making the MK-MMD loss both scalable and statistically well-behaved.

Model Architecture (from paper)



Effect of multiple-bandwidth kernels



Long, M., et al. (2015). "Learning transferable features with deep adaptation networks." *ICML*.

Theory: Correlation Alignment (CORAL)

While MMD aligns first-order statistics (means), CORAL recognizes that **second-order statistics** (covariance structure) also encode domain-specific information.

Geometric Intuition

Source and target feature distributions may have different *shapes*:

- Source: tilted elongated ellipsoid (high variance in certain directions)
- Target: horizontal elongated ellipsoid (decorrelation)

Simply matching means is insufficient—we must align the **covariance structure**.

CORAL Transform

Let matrices C_S and C_T be the covariance matrices of source and target features. If we transform source features with the linear transformation A (i.e., $x'_s = Ax_s$), then the covariance matrix of the transformed features will be $A^T C_s A$. CORAL aims to find the linear transformation A minimizing the divergence between the two covariance matrices:

$$\min_A \{|A^\top C_S A - C_T|_F^2\}$$

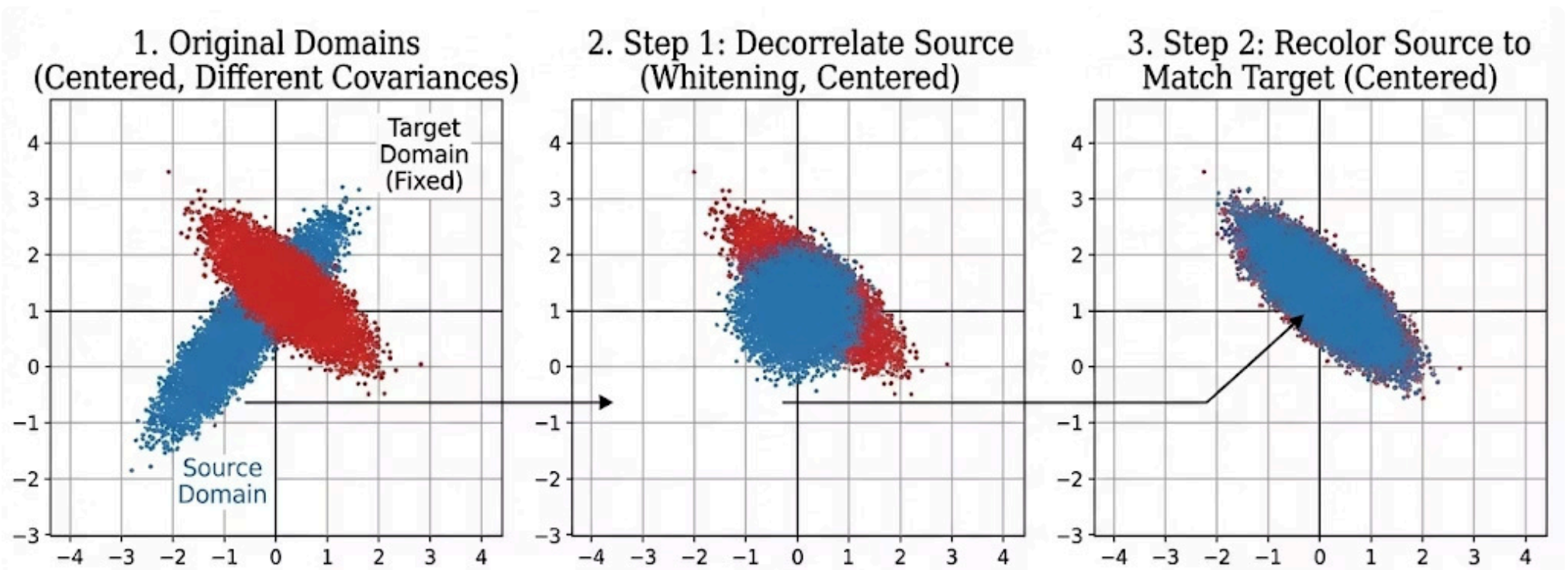
Where $|\cdot|_F$ is the Frobenius norm (a specific norm measuring the "magnitude" of a matrix - we will omit the details).

Whitening–Recoloring Solution

Closed-form via:

1. **Whiten** source features: $\tilde{X}_S = X_S(C_S + \lambda I)^{-1/2}$
2. **Re-color** with target covariance: $X_S^* = \tilde{X}_S(C_T + \lambda I)^{1/2}$

This implements the optimal linear feature transformation (with small regularization $\lambda > 0$). Assume features are all centred around the mean.



Sun, Baochen, Jiashi Feng, and Kate Saenko. "Return of frustratingly easy domain adaptation." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. No. 1. 2016.

Deep CORAL

Extend CORAL to deep networks by aligning covariances of layer activations during end-to-end training.

Motivation

- CORAL is a closed-form linear transform—powerful but limited to shallow, fixed feature representations.
- Deep CORAL integrates covariance alignment loss into a deep neural network, enabling joint learning of features and domain-invariant representations.

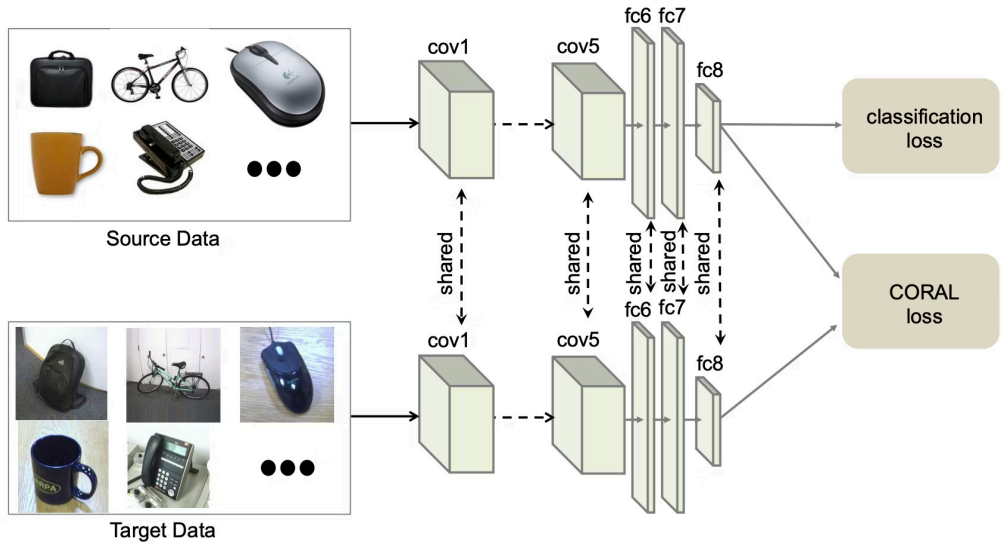
Deep CORAL Loss

Add to the task loss (e.g., classification) a regularization term:

$$L_{\text{Deep CORAL}} = \frac{1}{4d^2}\|C_S - C_T\|_F^2$$

where:

- C_S, C_T are covariances of activations (at some intermediate layer) for source and target batches
- d is the feature dimension
- The factor $1/(4d^2)$ normalizes the loss to be scale-invariant



Training

Minimize:

$$L_{\text{total}} = L_{\text{task}} + \lambda\, L_{\text{Deep CORAL}}$$

where λ is a hyperparameter controlling the strength of domain alignment.

Sun, Baochen, and Kate Saenko. "Deep coral: Correlation alignment for deep domain adaptation." *European conference on computer vision*. Cham: Springer International Publishing, 2016.

The Adversarial Philosophy

Learning to Fool Discriminators

Adversarial adaptation takes a fundamentally different approach: rather than using fixed distance metrics (MMD, CORAL), we **learn** the distance measure itself through a discriminator network.

The Minimax Game

Two networks compete in a game-theoretic framework:

- **Discriminator (D):** Learns to classify whether features come from source or target
- **Feature Extractor (F):** Learns to produce features that fool D (make domains indistinguishable)

Feature-Level Alignment

Methods like DANN and ADDA align latent representations—the high-dimensional feature space learned by deep networks.

Pixel-Level Alignment

Methods like CycleGAN align raw visual appearance, translating images from source style to target style.

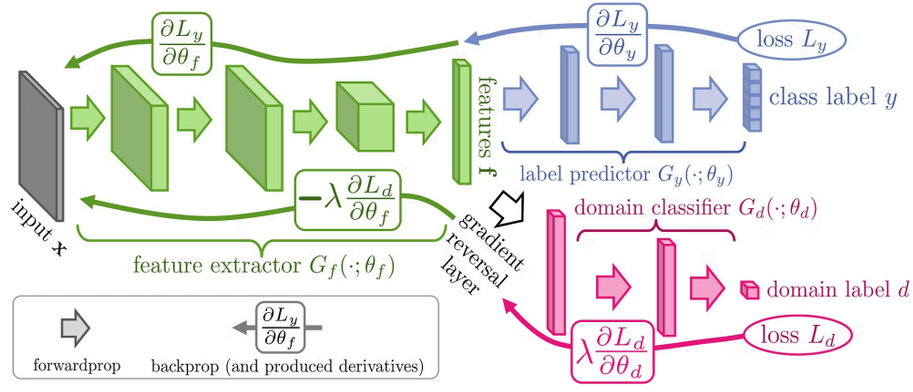
This strongly connect to literature on Generative Adversarial Networks.

Domain-Adversarial Neural Networks (DANN)

DANN introduced the elegant concept of domain confusion through adversarial training, enabling end-to-end learning of domain-invariant features.

Architecture Components

1. **Feature Extractor (G_f):** Shared CNN backbone extracting representations from both domains
2. **Label Predictor (G_y):** Classifies source samples using standard supervised loss
3. **Domain Classifier (G_d):** Binary classifier distinguishing source from target features



The Gradient Reversal Layer (GRL)

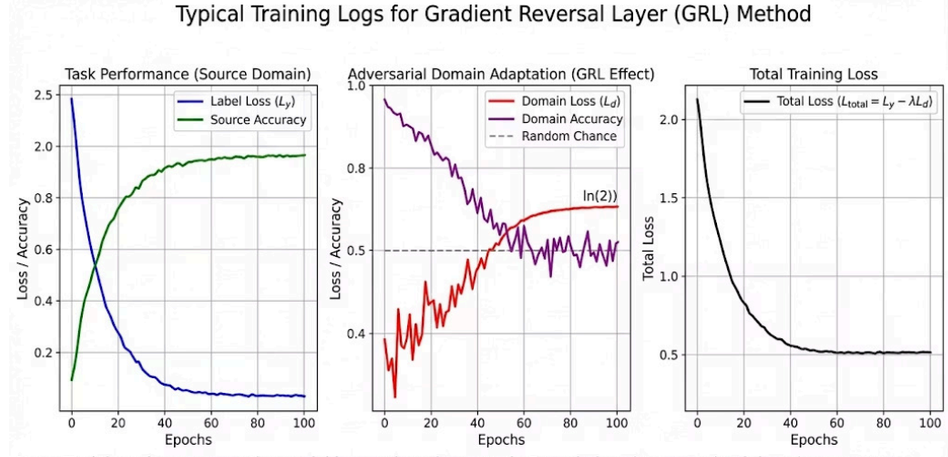
The critical innovation: during backpropagation, the GRL multiplies gradients from D by -1. This creates competing objectives:

- D tries to *maximize* domain classification accuracy
- F tries to *minimize* domain classification accuracy

$$\min_F \max_D \mathcal{L}_C - \lambda \mathcal{L}_D$$

Training Dynamics

At equilibrium, D cannot distinguish domains (accuracy ~50%), meaning F has learned domain-invariant features. The hyperparameter λ is gradually increased during training.



Note: Task loss decreases as the model learns the primary task. Domain loss *increases* (and domain accuracy decreases towards random) due to the adversarial GRL, indicating the feature extractor is learning domain-invariant features.

Ganin, Y., et al. (2016). "Domain-adversarial training of neural networks." *JMLR*, 17(1), 2030-2096.

Adversarial Discriminative Domain Adaptation (ADDA)

ADDA extends the adversarial framework by introducing **asymmetric mapping**—using separate networks for source and target feature extraction.

01

Pre-train Source Model

Train a source CNN S using standard supervised learning on labeled source data. This initializes a strong feature extractor for the source domain.

02

Initialize Target Model

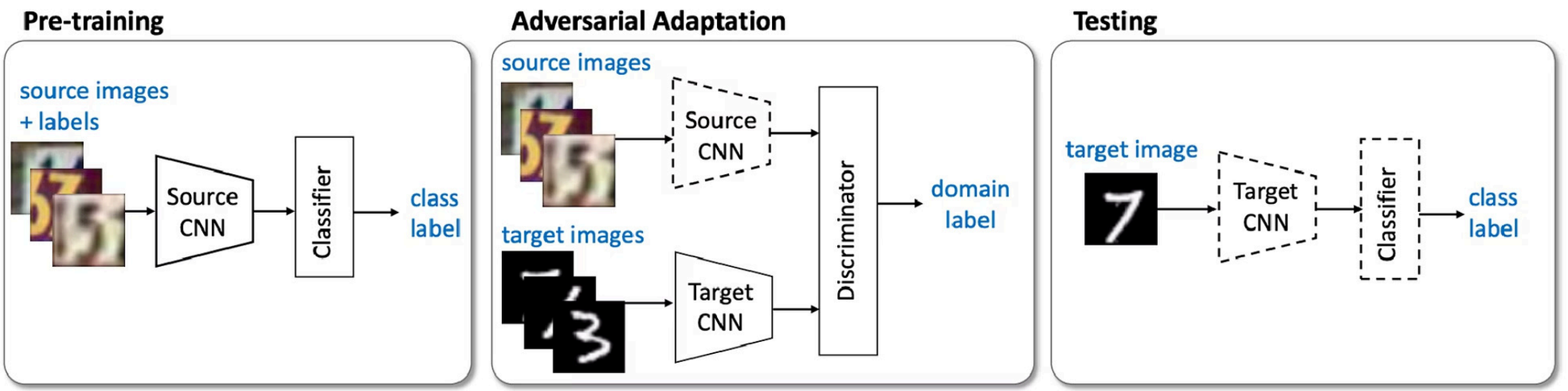
Copy source weights to initialize target CNN T. Crucially, S and T are **not** weight-tied—they can diverge during adaptation.

03

Adversarial Alignment

Freeze S. Train T adversarially to produce features indistinguishable from $S(x_s)$ according to discriminator D.

$$\min_T \max_D \mathcal{L}_{adv}(S(x_s), T(x_t), D)$$



Key Insight: Untied weights allow asymmetric transformations, beneficial when source and target domains require fundamentally different feature mappings (e.g., synthetic to real).

Tzeng, E., et al. (2017). "Adversarial discriminative domain adaptation." *CVPR*.

Pixel-Level Adaptation

While methods seen so far aim to align features, when dealing with images, we can align the source images in pixel-space rather than their features.

The Interpretability Problem

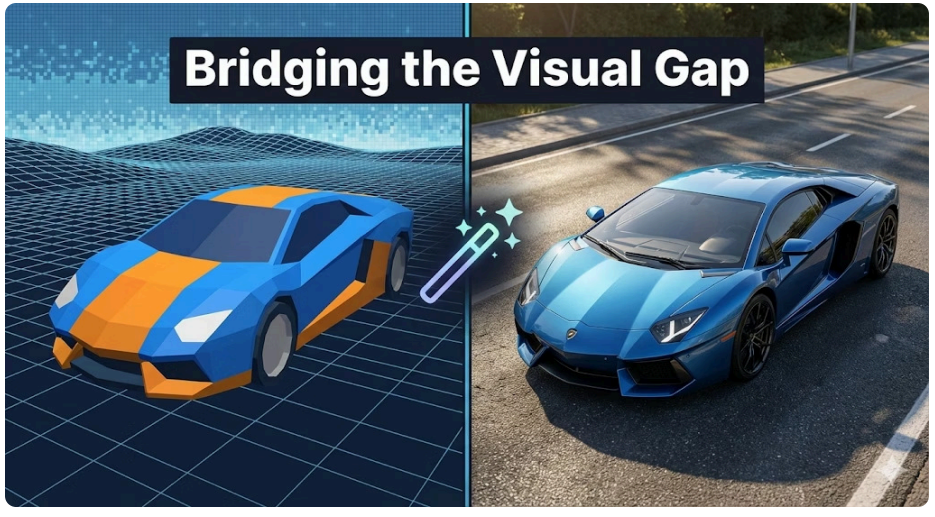
Feature-level adaptation (DANN, ADDA) operates in high-dimensional latent spaces. We optimize for domain confusion but cannot visualize what "domain-invariant" actually means—the learned representations remain a black box.

The Sim-to-Real Gap

Synthetic data (Grand Theft Auto 5, CAD models) provides free, perfect labels but suffers from unrealistic visual appearance: flat textures, simplistic lighting, absent atmospheric effects. The semantic content is correct, but the low-level statistics are wrong.

The Pixel-Level Solution

If we can **translate** source images to look like target images, we transform unsupervised domain adaptation into standard supervised learning. Train on "target-style" images using source labels.



Pixel-level adaptation is particularly effective for large appearance shifts (synthetic→real) but may be overkill for small shifts where feature-level methods suffice.

Pixel-Level Adaptation: CycleGAN

CycleGAN enables image-to-image translation **without paired training data**—a critical advantage since obtaining pixel-aligned image pairs across domains is often impossible.

The Cycle Consistency Principle

Learn two generators: $G : S \to T$ and $F : T \to S$. Enforce that translating $S \to T \to S$ recovers the original image:

$$\mathcal{L}_{cyc} = \|F(G(x_s)) - x_s\| + \|G(F(x_t)) - x_t\|$$

Adversarial Loss	Cycle Loss	Identity Loss
Generators produce realistic target-style images: $\mathcal{L}_{GAN}(G, D_T)$	Translations preserve semantic content: $\mathcal{L}_{cyc}$	Generators preserve color when unnecessary: $\mathcal{L}_{id}$

The Full CycleGAN Objective

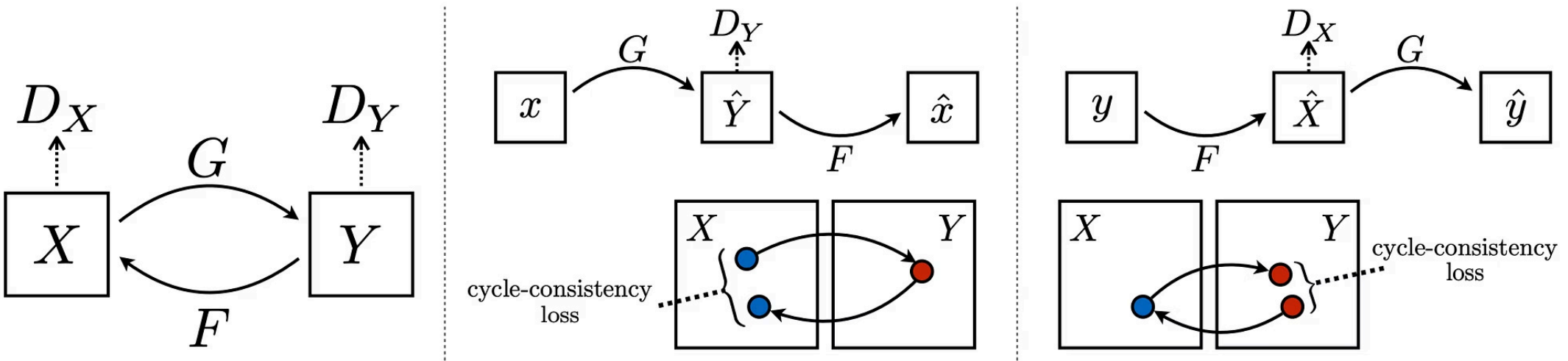
The total objective function combines adversarial losses for both generation directions, cycle consistency losses, and an optional identity mapping loss, weighted by hyperparameters:

$$\mathcal{L}_{total}(G, F, D_S, D_T) = \mathcal{L}_{GAN}(G, D_T, X, Y) + \mathcal{L}_{GAN}(F, D_S, Y, X) + \lambda_{cyc}\mathcal{L}_{cyc}(G, F, X, Y) + \lambda_{identity}\mathcal{L}_{identity}(G, F, X, Y)$$

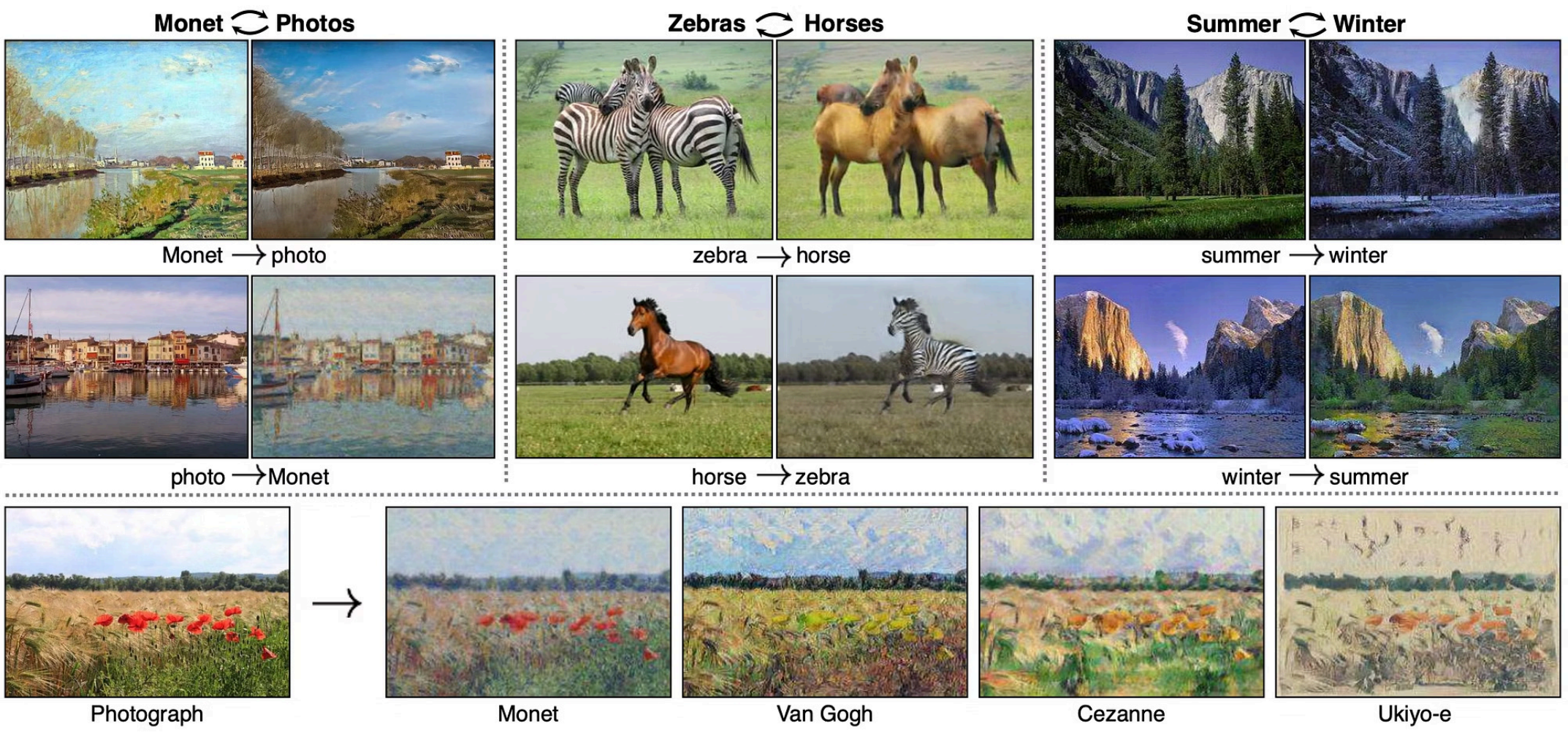
Where:

- $\mathcal{L}_{GAN}(G, D_T, X, Y)$: Adversarial loss for generator G and discriminator D_T, encouraging G to generate images $G(x)$ that look like they came from domain Y, while D_T aims to distinguish them from real images in Y.
- $\mathcal{L}_{GAN}(F, D_S, Y, X)$: Adversarial loss for generator F and discriminator D_S, encouraging F to generate images $F(y)$ that look like they came from domain X, while D_S aims to distinguish them from real images in X.
- $\mathcal{L}_{cyc}(G, F, X, Y) = \|F(G(x)) - x\|_1 + \|G(F(y)) - y\|_1$: Cycle consistency loss, ensuring that translating an image from one domain to another and back reconstructs the original image. This prevents the network from learning mappings that simply transform input images to noise.
- $\mathcal{L}_{identity}(G, F, X, Y) = \|G(y) - y\|_1 + \|F(x) - x\|_1$: Identity loss, which ensures that if a real image from domain Y is fed to generator G (which maps X to Y), it should ideally remain unchanged. Similarly for F and domain X. This helps preserve color composition and prevents unnecessary changes.

The weighting hyperparameters λ_{cyc} (typically 10) and $\lambda_{identity}$ (typically 0.5 λ_{cyc}) balance the importance of these objectives.



Results



Domain Adaptation Pipeline

CycleGAN is used in a domain adaptation pipeline as follows:

1. Use G to translate source images to target style.
2. Train classifier on translated images using source labels.
3. Deploy on real target data.

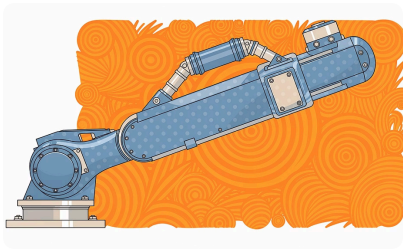
Zhu, J.-Y., et al. (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks." *ICCV*.

Beyond Feature Alignment and Adversarial Learning

Domain Randomization, Self-Training, Reconstruction-Based and Ensembles of Methods

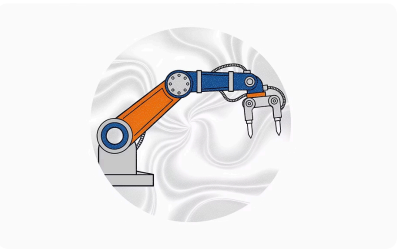
Domain Randomization

Domain randomization takes a radically different approach: instead of adapting to a *specific* target domain, train on synthetic data with **extreme visual variability** to encourage invariance to superficial appearance.



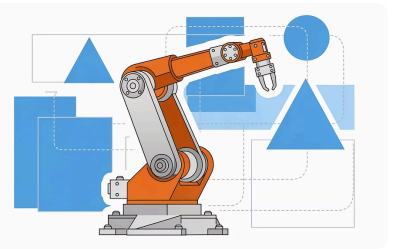
Randomize Textures

Apply random colors, patterns, and materials to objects and backgrounds during training.



Randomize Lighting

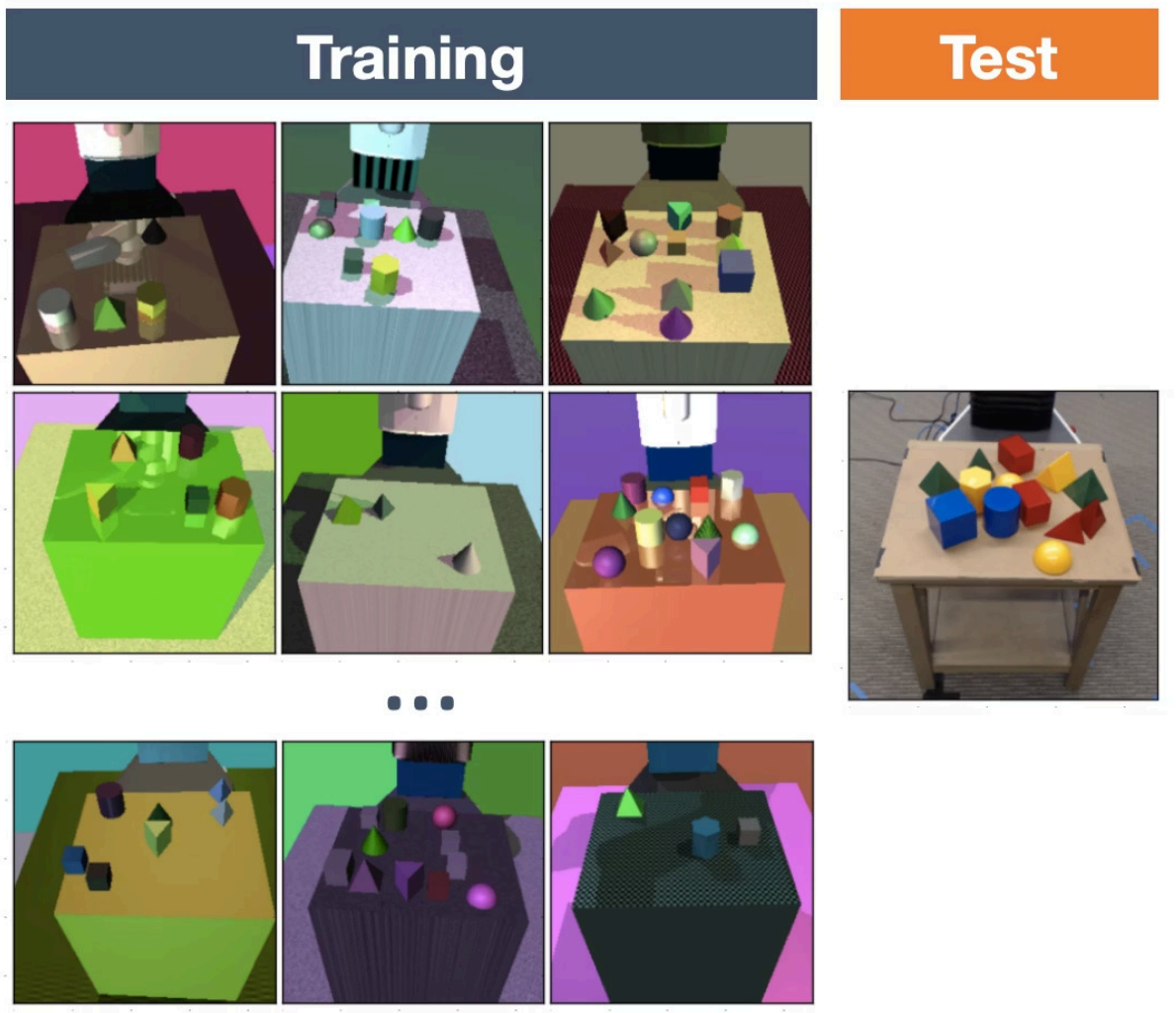
Vary light positions, colors, and intensities to remove lighting-specific cues.



Randomize Cameras

Change viewpoints, focal lengths, and sensor noise patterns.

Core Philosophy: "If the model can recognize an object under polka dots and neon lighting, it can recognize it under any realistic condition." By training on a distribution *wider* than reality, we encompass the target domain without explicitly observing it.



Tobin, J., et al. (2017). "Domain randomization for transferring deep neural networks from simulation to the real world." IROS.

Self-Training & Teacher-Student

Leveraging unlabeled target domain data is crucial when labeled data is scarce. Self-training and teacher-student models provide effective strategies for this by generating pseudo-labels for target samples, enhancing generalization.

Iterative Pseudo-Labeling

This method involves using a model trained on source data to predict labels for unlabeled target data. These high-confidence predictions, or pseudo-labels, are then treated as ground truth to retrain or fine-tune the model, iteratively improving performance.

Mean Teacher Approach

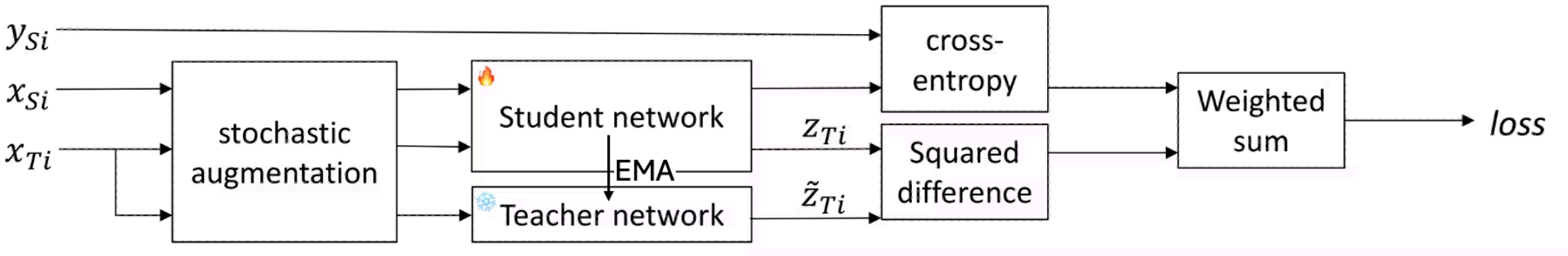
The Mean Teacher model addresses the instability of pseudo-labeling by maintaining two neural networks: a 'Student' model and a 'Teacher' model. The Teacher's parameters are an Exponential Moving Average (EMA) of the Student's past parameters, making it a more stable target for supervision. At each training step, the teacher parameters are updated as a weighted average of the previous teacher parameters and the current student parameters, providing temporal smoothing and stability.

$$\theta_{teacher} = \alpha \cdot \theta_{teacher} + (1 - \alpha) \cdot \theta_{student}$$

Here, α is the EMA decay rate (typically 0.99 or 0.999), which determines the influence of past teacher parameters versus current student parameters. This temporal smoothing makes the teacher a more reliable source of pseudo-labels than the student itself.

$$L = L_{sup} + ||Student(x) - Teacher(x)||^2$$

The total loss function combines a supervised loss (L_{sup}) on labeled data and an unsupervised consistency loss, encouraging the Student's predictions to match the (more stable) Teacher's predictions on unlabeled data.



Tarvainen, A., & Valpola, H. (2017). Mean teachers are better role models. *NeurIPS*.

French, G., Mackiewicz, M., & Fisher, M. (2017). Self-ensembling for visual domain adaptation. arXiv preprint arXiv:1706.05208.

Reconstruction-Based Adaptation (DRCN)

Deep Reconstruction-Classification Networks (DRCN) introduce an auxiliary task—pixel-level reconstruction—to regularize feature learning and prevent overfitting to source-specific shortcuts.

Encoder (Shared)

Maps input images x to latent representations $f(x)$. Learned from both source and target domains.

Classifier Head

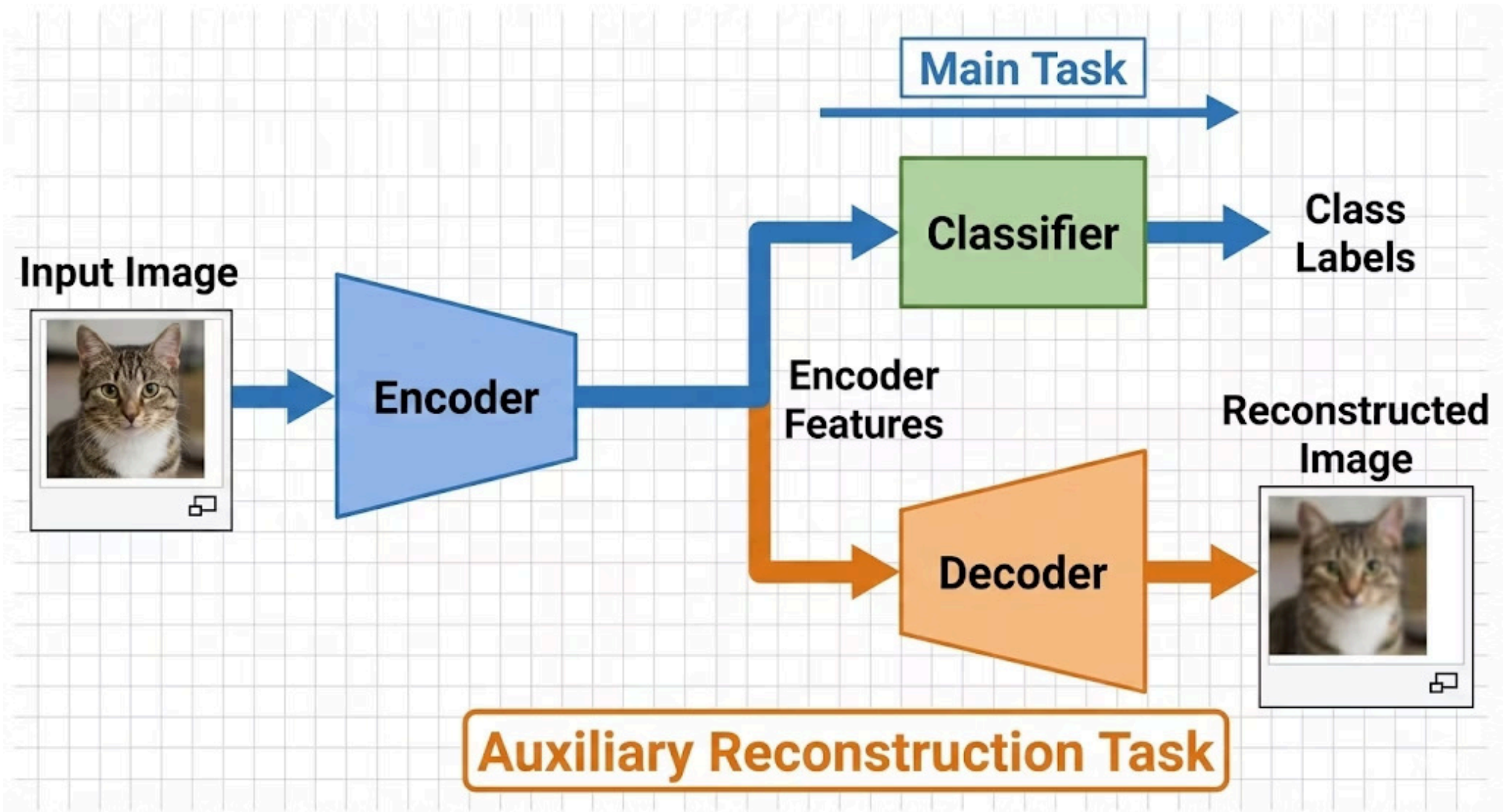
Supervised classification on source data: $\mathcal{L}_{class} = CE(y_s, C(f(x_s)))$

Decoder Head

Reconstructs input pixels from latent features: $\mathcal{L}_{recon} = \|x - D(f(x))\|^2$

Applied to both domains.

Intuition: If the encoder can reconstruct target images, it must have learned meaningful representations of target data—not just source-specific features. Reconstruction acts as an unsupervised regularizer.

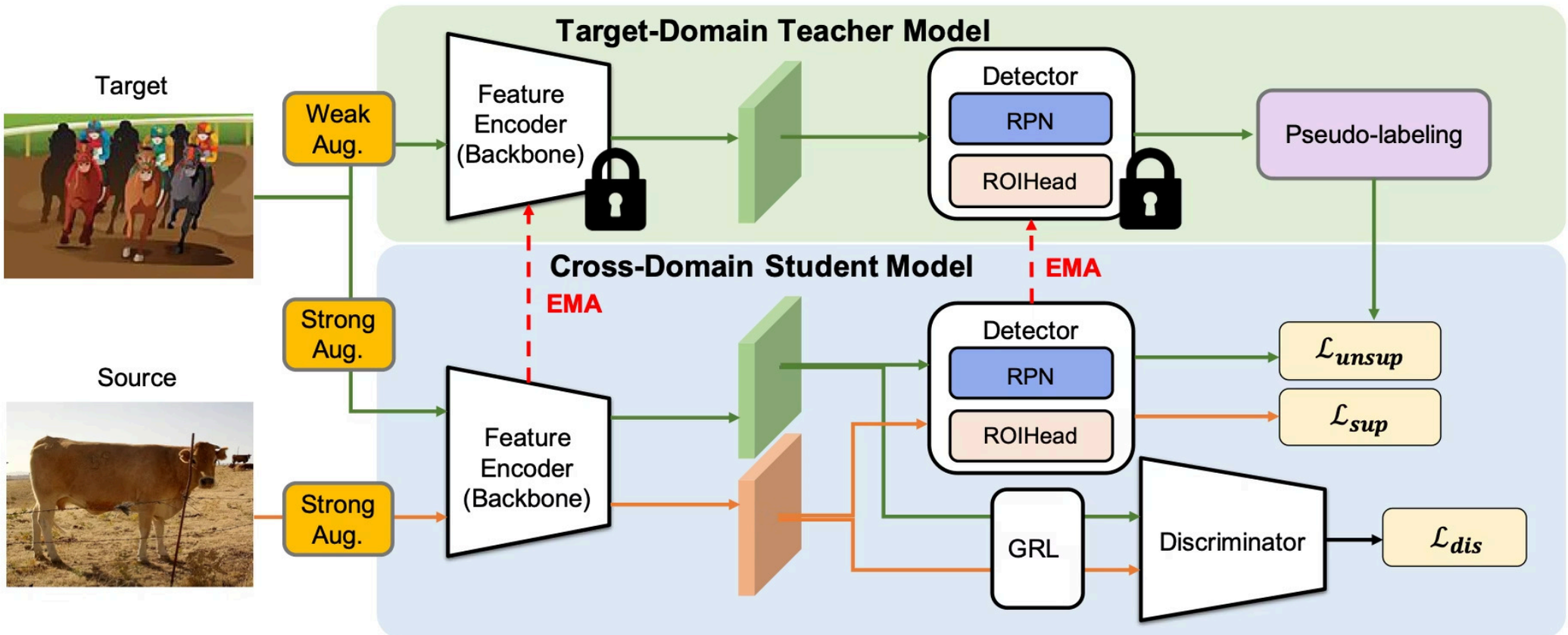


Ghifary, M., et al. (2016). "Deep reconstruction-classification networks for unsupervised domain adaptation." *ECCV*.

An Ensamble of Methods

In practice, these techniques can be combined, even in scenarios different from simple classification.

For example, consider the "Adaptive Teacher" adaptation for cross-domain object detection.



Li, Y. J., Dai, X., Ma, C. Y., Liu, Y. C., Chen, K., Wu, B., ... & Vajda, P. (2022). Cross-domain adaptive teacher for object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 7581-7590).

MODULE 4 EMERGING PARADIGMS

Emerging Paradigms in Domain Adaptation

Beyond Standard Assumptions

Recent research challenges traditional domain adaptation assumptions—access to source data, offline training, and task-specific models—leading to more practical and flexible adaptation frameworks.

Source-Free Domain Adaptation (SFDA)

Source-Free DA addresses a critical real-world constraint: source data may be unavailable due to privacy regulations (GDPR), proprietary restrictions, or storage limitations. Only the pre-trained source model is accessible.

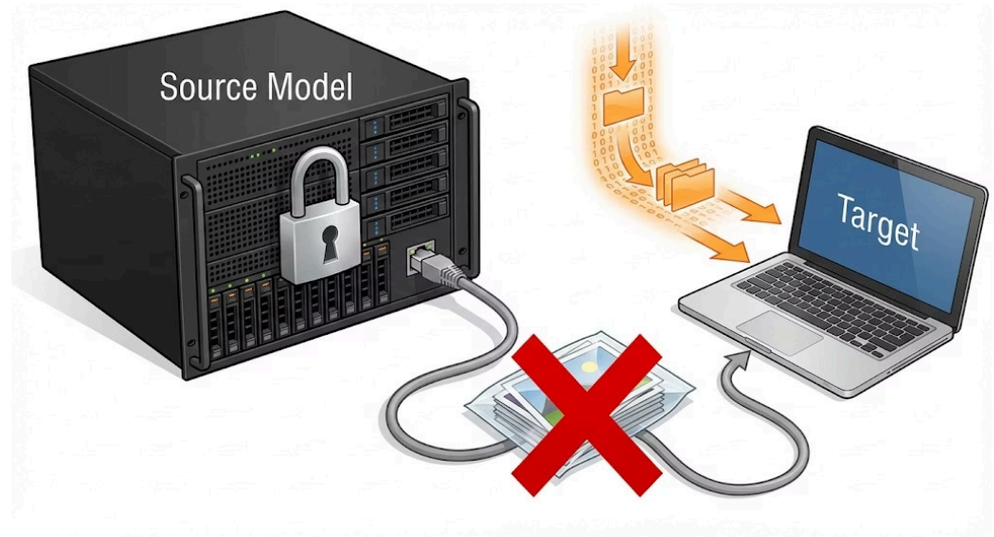
Problem Formulation

Given: Pre-trained source model f_S (feature extractor + classifier)

Given: Unlabeled target data $\{x_t\}$

Not Given: Source training data $\{(x_s, y_s)\}$

Goal: Adapt f_S to perform well on target domain



SHOT: Self-Supervised Adaptation

A prominent SFDA method that:

1. **Freezes** the classifier head (preserves source decision boundary)
2. **Updates** the feature extractor to maximize:
   - Prediction confidence (entropy minimization)
   - Class diversity (avoid collapse to single class)

$$\mathcal{L} = \mathcal{H}(y_t) - \mathcal{H}(\bar{y}_t)$$

First term: minimize prediction entropy (confident predictions). Second term: maximize class distribution entropy (diverse predictions).

Liang, J., et al. (2020). "Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation." *ICML*.

Test-Time Adaptation (TTA)

Test-Time Adaptation confronts deployment scenarios where the model encounters continuously shifting distributions in production. Traditional offline adaptation is insufficient—the model must adapt **online** with each incoming batch.

Deployment Start

Model trained on clear daytime driving conditions.

Night Driving

Drastically different visual conditions require immediate adaptation.

1 — 2 — 3 — 4

Dusk Transition

Lighting changes gradually. Model performance degrades.

Rain Onset

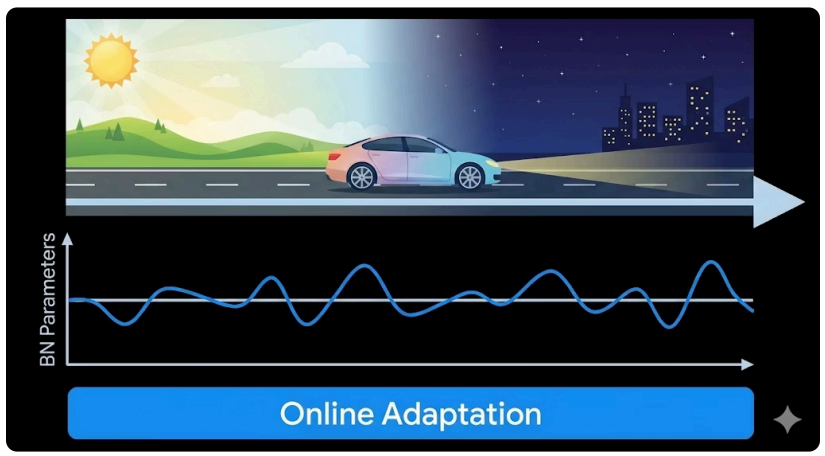
Additional compounding shift. Continuous adaptation needed.

TENT: Test-Time Entropy Minimization

TENT enables efficient online adaptation by updating **only Batch Normalization parameters** (scale γ and shift β) to minimize prediction entropy on the current test batch.

Why BN-only?

- Minimal parameters to update (~1% of model)
- Fast optimization (seconds per batch)
- BN statistics encode distribution-specific information
- Prevents catastrophic forgetting of source knowledge

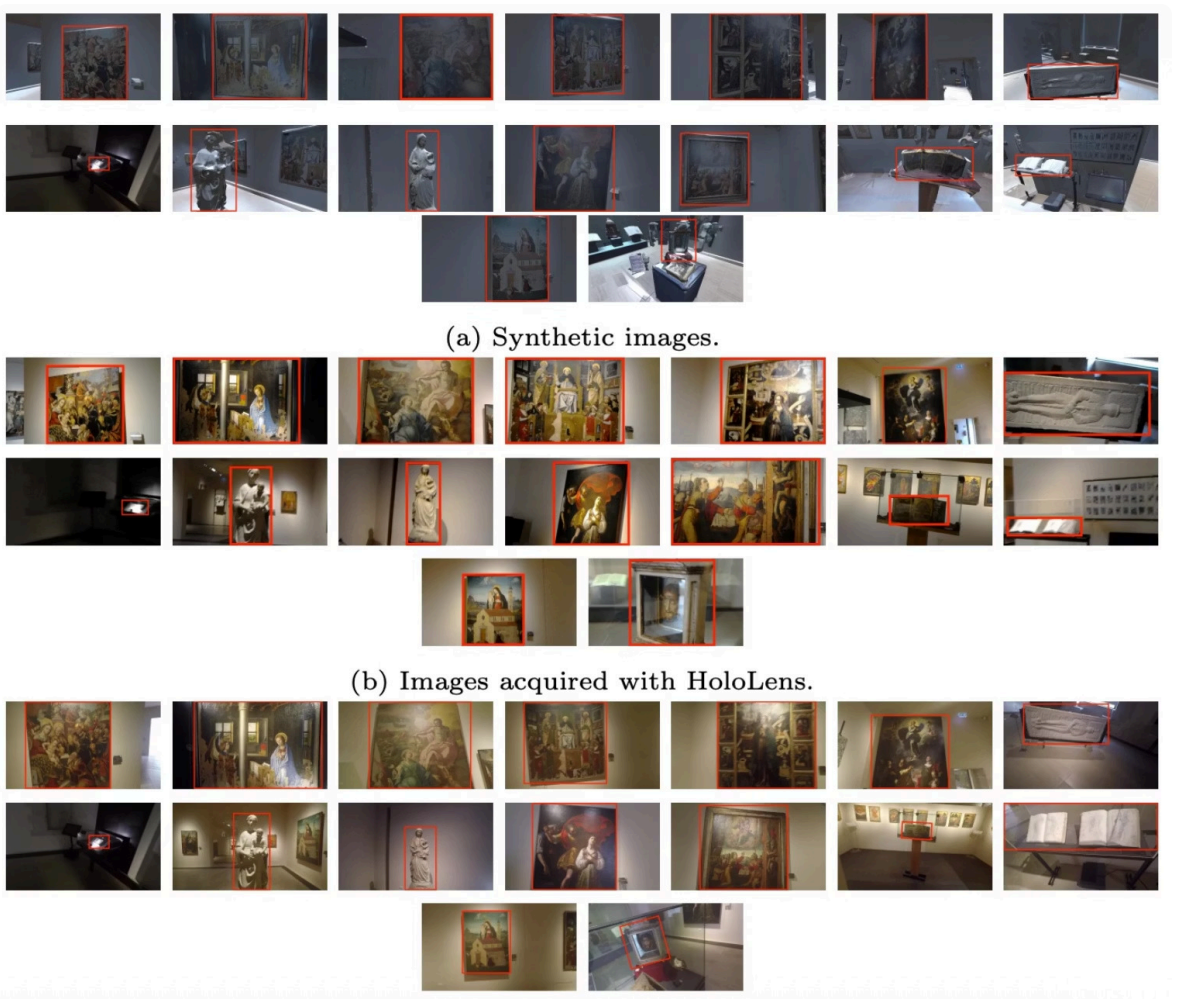


Wang, D., et al. (2021). "Tent: Fully test-time adaptation by entropy minimization." *ICLR*.

First-Hand Experience with Domain Adaptation for Object Detection

Problem

Detecting artworks in museums by learning from **labeled** synthetic data generated from a 3D model of the museum and **unlabeled** real data collected with two cameras.



(a) Synthetic images.

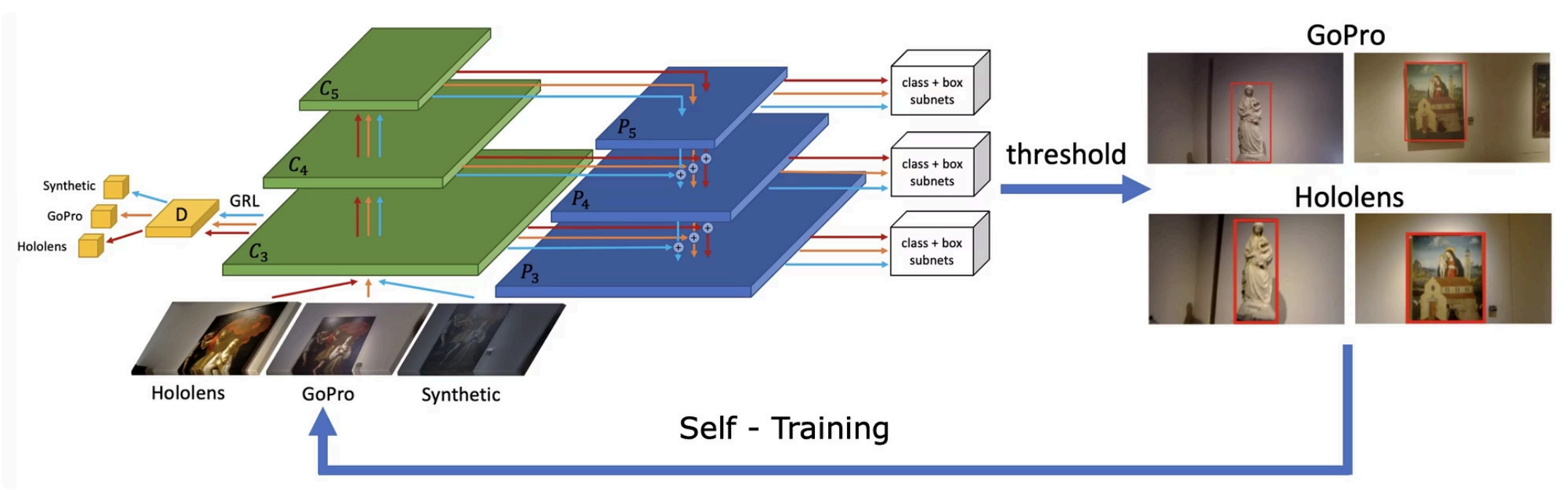
(b) Images acquired with HoloLens.

(c) Images acquired with GoPro.

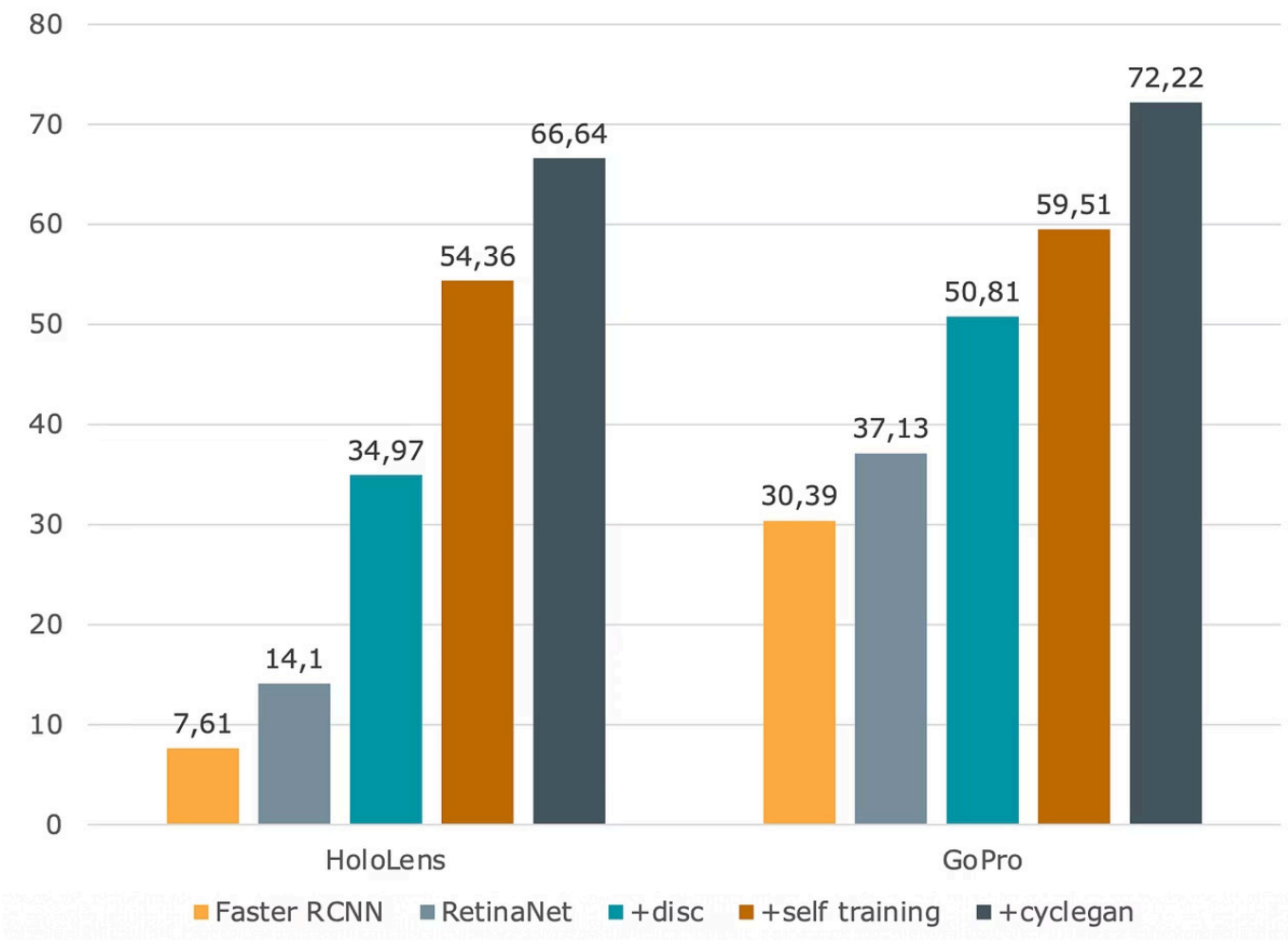
Domain-Adaptive Object Detection

We integrate different components:

- Gradient Reversal Layer (GRL) to classify the three domains (one synthetic, two real)
- A self-training loop, where the model generate pseudo-labels on unlabeled real images
- CycleGAN to minimize pixel-level gap



Results



RetinaNet suffers less from domain gap as compared to Faster RCNN.

Giovanni Pasqualino, Antonino Furnari, Giovanni Maria Farinella (2022). A multi camera unsupervised domain adaptation pipeline for object detection in cultural sites through adversarial learning and self-training. Computer Vision and Image Understanding (CVIU), pp. 103487.

Evaluation Pitfalls & Future Research Directions

Common Evaluation Mistakes

t-SNE Misleading

t-SNE visualizations can show perfect separation in 2D while 3D structure remains tangled. It preserves *local* not *global* structure.

A-Distance Metric

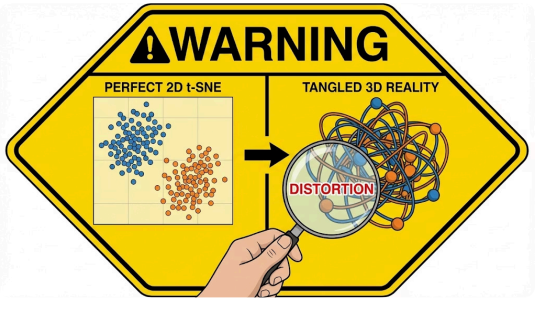
Measure domain divergence via discriminator error: $d_A = 2(1 - 2\epsilon)$ where ϵ is domain classification error. Lower is better.

Task Performance

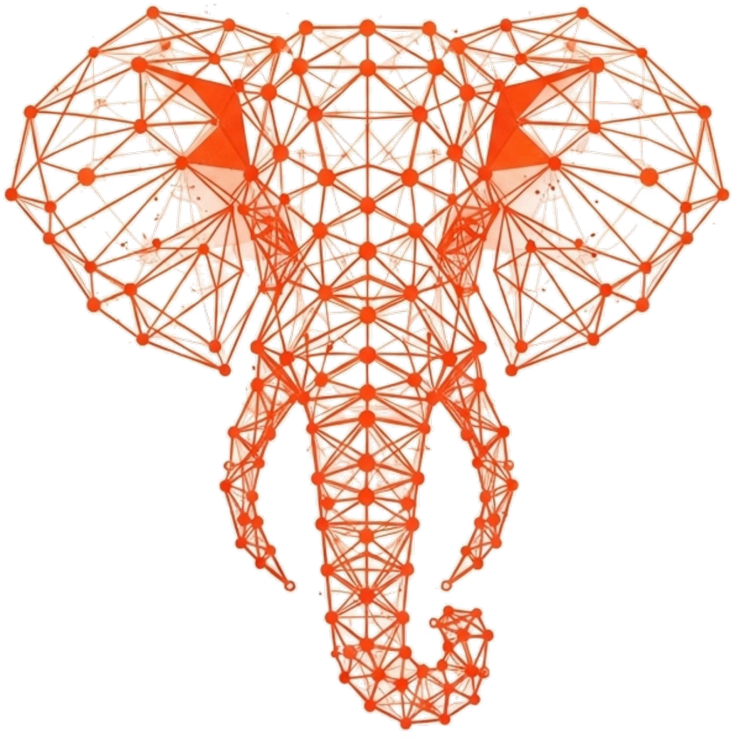
Always report **target domain accuracy** on held-out test data—not just source accuracy or qualitative alignment.

Future Research Frontiers

1. **Universal Domain Adaptation:** Unified framework handling open set, partial, and closed set scenarios simultaneously
2. **Safety & Robustness:** Certified adaptation with formal guarantees against adversarial shifts
3. **Continual Adaptation:** Lifelong learning across sequential domains without catastrophic forgetting
4. **Multi-modal Adaptation:** Leveraging language, vision, and other modalities jointly
5. **Causal DA:** Moving beyond correlation to understand causal mechanisms of domain shift



Conclusions and Next Steps



We Have Explored:

- Domain adaptation fundamentals, including the distinction between source and target domains and different types of supervision.
- Key types of domain shift: covariate, label, and concept shift.
- Various adaptation scenarios: closed set, partial, and open set adaptation.
- Discrepancy-based methods for domain alignment, such as MMD, CORAL, Deep CORAL, and DAN.
- Adversarial methods, including Domain-Adversarial Neural Networks (DANN) and Adversarial Discriminative Domain Adaptation (ADDA).
- Pixel-level adaptation techniques like CycleGAN for image-to-image translation across domains.
- Alternative approaches to domain adaptation: domain randomization, self-training, teacher-student models, and reconstruction-based methods.
- Emerging paradigms in domain adaptation, such as source-free DA and test-time adaptation.

Uni ct DEEP LEARNING
ADVANCED MODELS AND METHODS

References

1. Saenko et al., Adapting visual category models to new domains, ECCV 2010.
2. Ganin et al., Domain-Adversarial Training of Neural Networks (DANN), JMLR 2016.
3. Tzeng et al., Adversarial Discriminative Domain Adaptation (ADDA), CVPR 2017.
4. Long et al., Learning Transferable Features with Deep Adaptation Networks (DAN), ICML 2015.
5. Sun & Saenko, Deep CORAL: Correlation Alignment for Deep Domain Adaptation, ECCV 2016.
6. Zhu et al., Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, ICCV 2017.
7. Liang et al., Do We Really Need to Access the Source Data? Source Hypothesis Transfer (SHOT), ICML 2020.
8. Wang et al., Tent: Fully Test-Time Adaptation by Entropy Minimization, ICLR 2021.
9. Wang & Deng, Deep Visual Domain Adaptation: A Survey, Neurocomputing 2018.