Unsupervised Domain Adaptation

Kintan Saha Pratham Gupta
Gavish Bansal Krishna Agarwal

Indian Institute of Science, Bangalore

April 14, 2025

Project Motivation

Our goal is to explore the field of **Unsupervised Domain Adaptation (UDA)** and implement some of the state-of-the-art methods.
Flow of the presentation:

- Introduction to UDA
- Domain-Adversarial Training of Neural Networks (DANN)
- Survey of Other Methods of UDA
  - Divergence Based Methods
  - Reconstruction Based Methods
  - Ensemble Methods

What is UDA?

Domain Adaptation (DA) is a subfield of machine learning that focuses on transferring knowledge from a source domain to a target domain with different data distributions.

Key Idea: The goal is to learn a model that performs well on the target domain, even when only labeled data from the source domain is available.

Challenges: The main challenge in UDA is the domain shift, which occurs when the source and target domains have different distributions. This can lead to poor performance of standard learning techniques.

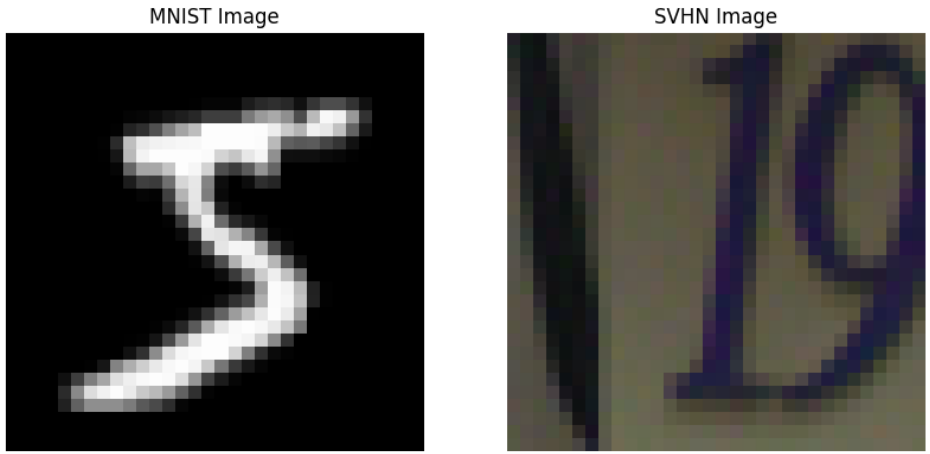


Figure: Example of domain shift between source and target domains

What is UDA?

In **Unsupervised Domain Adaptation (UDA)**, only the source domain data is labeled, while the target domain data is unlabeled.

Applications:

- When shifting a model trained on **synthetic data** to **real-world data**, the model may not perform well due to the differences in data distributions.
- For systems like **Security Cameras** and **Autonomous Driving**, the inputs to each camera belong to different domains. So we wish to adapt the model to the new domain.

$\mathcal{H}$-Divergence

$$d_{\mathcal{H}}(\mathcal{D}_S^X, \mathcal{D}_T^X) = 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{x \sim \mathcal{D}_S^X}[\eta(x) = 1] - \Pr_{x \sim \mathcal{D}_T^X}[\eta(x) = 1] \right|$$

- Measures the ability of a hypothesis $\eta \in \mathcal{H}$ to distinguish between source and target distributions.
- High $d_{\mathcal{H}}$ implies easy domain discrimination — bad for transfer.
- For symmetric hypothesis classes, it can be estimated via:

$$\hat{d}_A = 2(1 - 2\epsilon)$$

- Here, ϵ is the generalization error of a domain classifier.

DANN Architecture

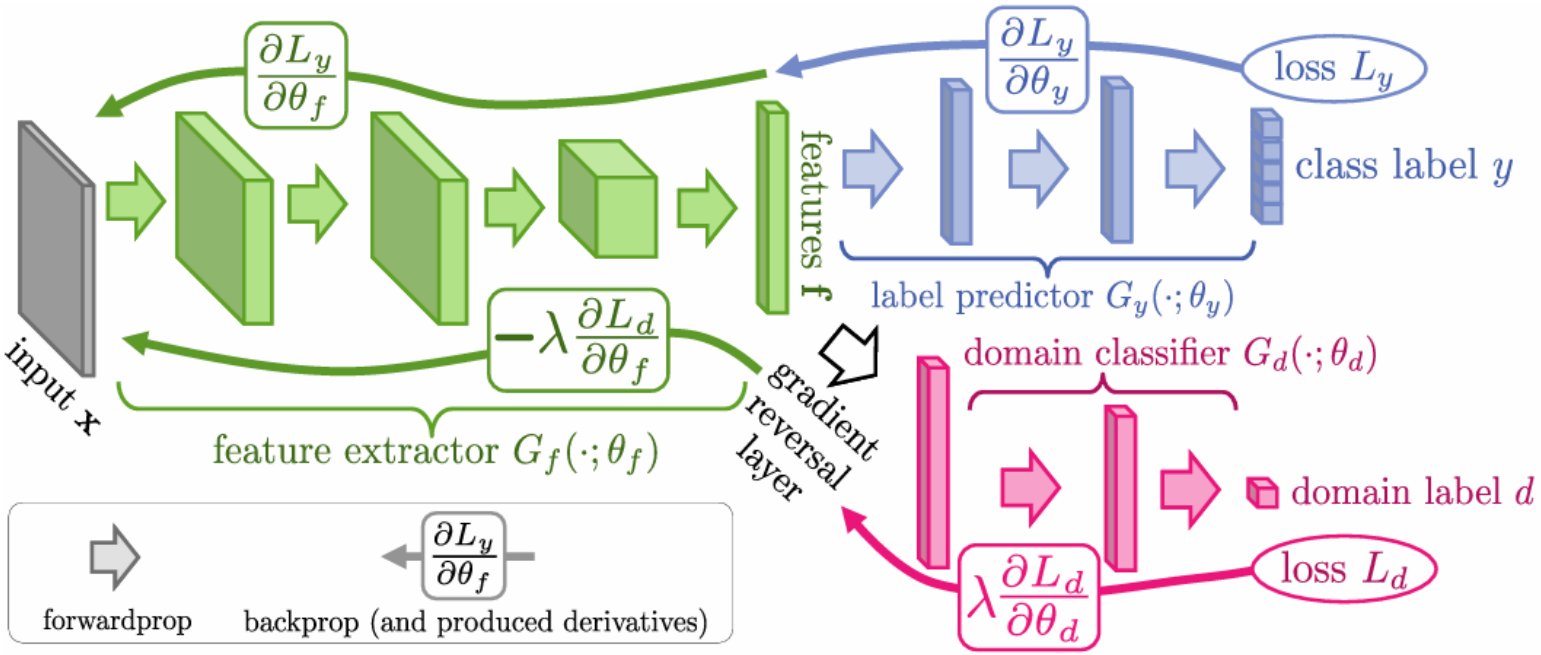


Figure: Domain-Adversarial Neural Network (DANN) Architecture

DANN: Key Idea

- Simultaneous optimization of:
  1. **Label predictor** — classifies source data correctly
  2. **Domain classifier** — fails to distinguish source from target
- The **feature extractor** is trained to:
  - Minimize label prediction loss (task objective)
  - Maximize domain classification loss (encourages invariance)

DANN Architecture Overview

- **Feature extractor** G_f: shared across tasks
- **Label predictor** G_y: trained on labeled source data
- **Domain classifier** G_d: trained to classify domain
- **Gradient Reversal Layer (GRL)**: connects G_f and G_d

Gradient Reversal Layer (GRL)

- **Forward pass:** identity function: $R(x) = x$
- **Backward pass:** gradient scaled by $-\lambda$:

$$\frac{dR}{dx} = -\lambda I$$

- GRL enables adversarial training by reversing gradients from the domain classifier.
- This pushes G_f to generate domain-invariant features.

Objective Function

$$\min_{\theta_f, \theta_y} \max_{\theta_d} \mathcal{E}(\theta_f, \theta_y, \theta_d)$$

$$\mathcal{E} = \frac{1}{n} \sum_{i=1}^{n} L_y(G_y(G_f(x_i)); y_i) - \lambda \sum_{j=1}^{n+n'} L_d(G_d(G_f(x_j)); d_j)$$

- **Label loss:** minimized w.r.t. θ_f, θ_y
- **Domain loss:** maximized w.r.t. θ_f, minimized w.r.t. θ_d
- λ balances the two objectives

Experiments

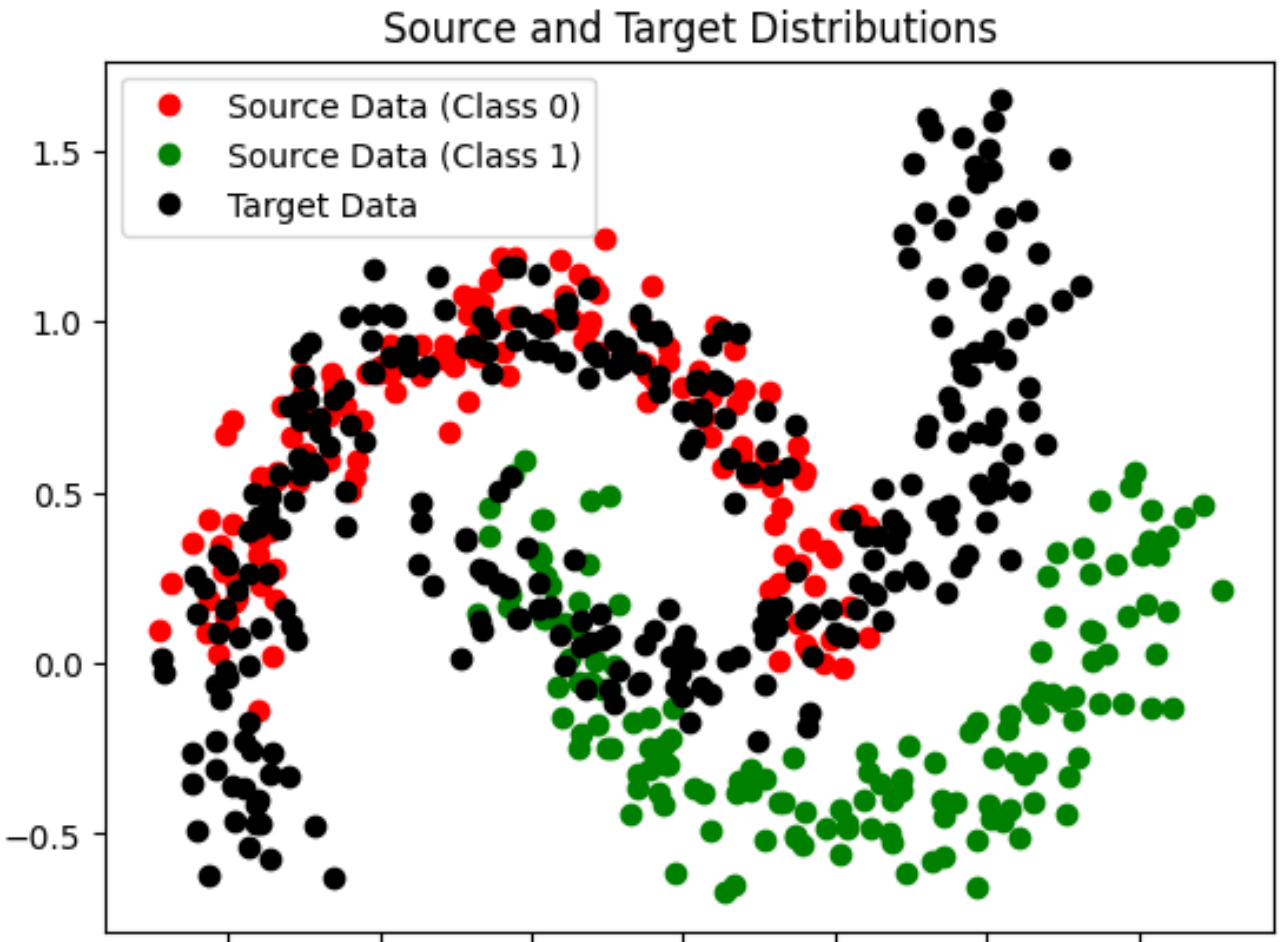
The experiments section is divided into 3 parts:

- Experiments on shallow DNN
- Experiments on image classification
- Experiments on image reconstruction

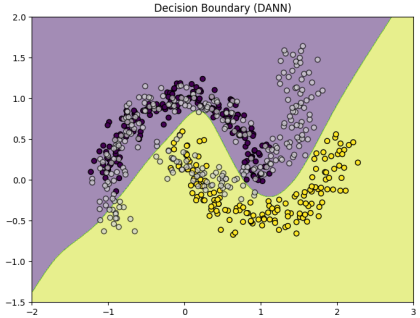
Experiments on Shallow DNN

Intertwining moons: Dataset used for evaluating domain adaptation.

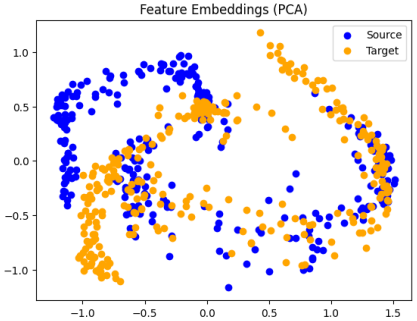
The structure of source and target distributions is shown below:



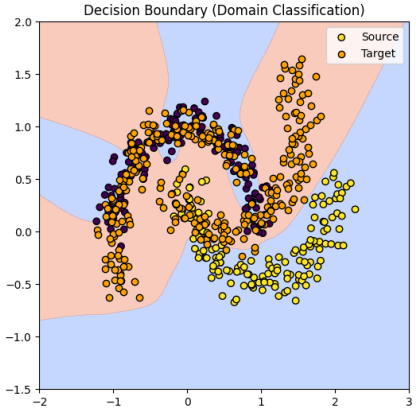
Results on the inter-twinning moons problem



Label Classification

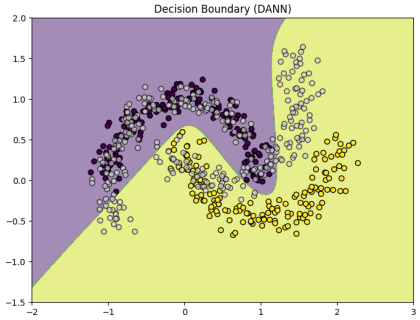


Representation PCA

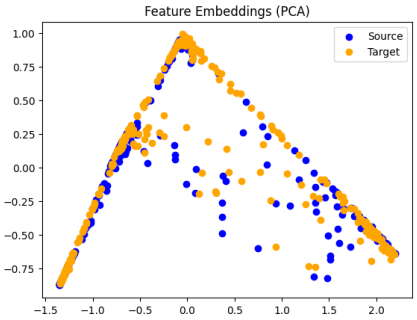


Domain Classification

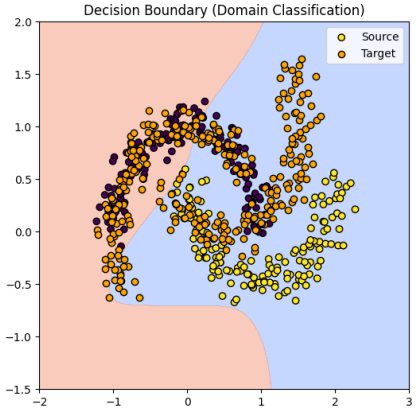
(a) Standard NN.



Label Classification



Representation PCA



Domain Classification

Domain Adaptation on Amazon review Datasets

Source	Target	DANN	NN	SVM	DANN(p)	NN(p)	SVM(p)
books	dvd	**72.8%**	**72.8%**	71.95%	78.4%	79.0%	**79.9%**
books	electronics	**70.9%**	69.5%	69.5%	73.3%	74.7%	**74.8%**
books	kitchen	**71.85%**	**71.85%**	71.7%	**77.9%**	77.8%	76.9%
dvd	books	**71.55%**	66.9%	69%	72.3%	72.0%	**74.3%**
dvd	electronics	**69.8%**	69%	66.63%	**75.4%**	73.2%	74.8%
dvd	kitchen	**70.05%**	67.25%	69.5%	**78.3%**	77.8%	74.6%
electronics	books	**65.95%**	63.8%	65.45%	**71.3%**	70.9%	70.5%
electronics	dvd	**68.05%**	67.15%	67%	**73.8%**	73.3%	72.6%
electronics	kitchen	**79.25%**	78.45%	78.75%	**85.4%**	**85.4%**	84.7%
kitchen	book	68.25%	**68.75%**	68.05%	**70.9%**	70.8%	70.7%
kitchen	dvd	**68.90%**	65.15%	68.5%	**74.0%**	73.9%	73.6%
kitchen	electronics	78.75%	77.4%	**79.25%**	**84.3%**	84.1%	84.2%

Table: Results of DANN, NN, SVM on Amazon Review Dataset compared with paper results.

MNIST → MNIST-M: Top Feature Extractor Layer

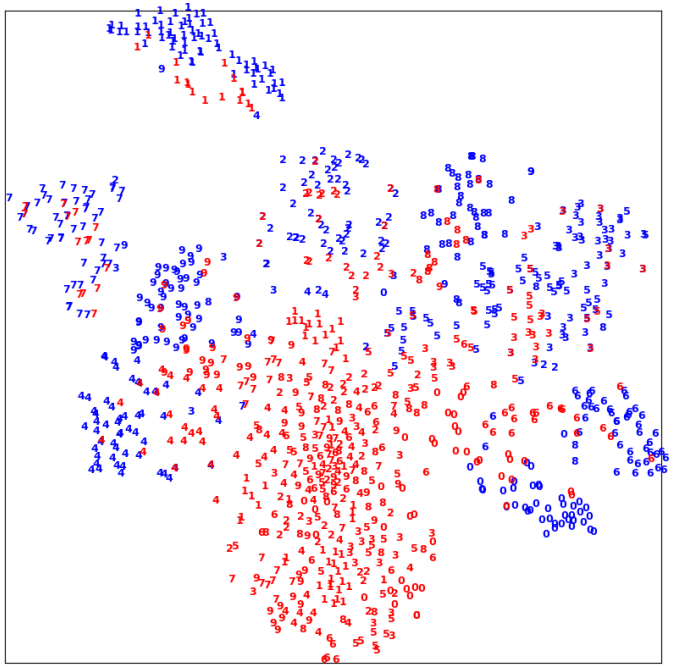


Figure: Without DANN

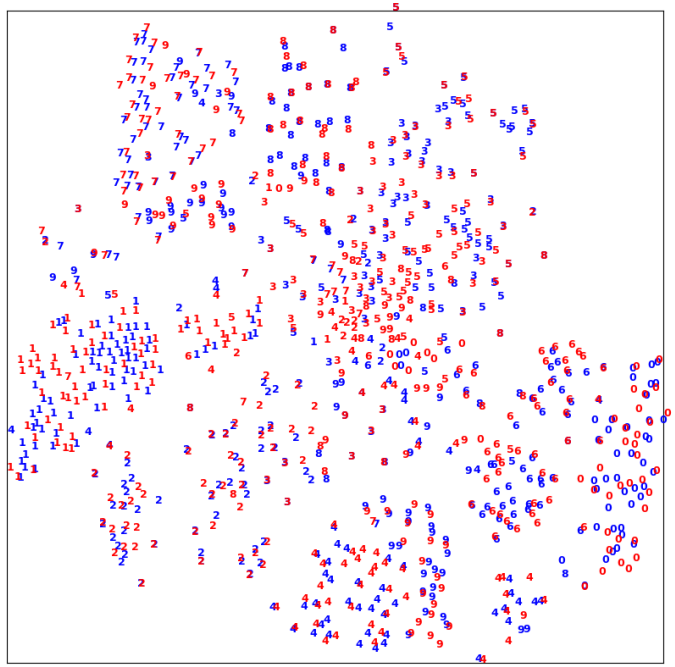


Figure: With DANN

Figure: t-SNE visualizations of extracted features. Blue points: source domain (MNIST); Red points: target domain (MNIST-M). Adaptation aligns distributions better.

Domain Invariant Feature Learning

Key Idea: The goal of domain invariant feature learning is to learn a representation that is invariant to the domain shift. Basically, aligning source and target domains by creating a domain invariant feature representation which follows the same distribution regardless of source and target.

Algorithms Explored:

- CORAL(Correlation Alignment) and DeepCORAL
- MMD(Maximum Mean Discrepancy)
- DSN(Domain Separation Networks)

CORAL

Key Idea: CORAL aligns the second-order statistics of the source and target distributions. It minimizes the distance between the covariance matrices of the source and target features. This is done by transforming the source data to a new feature representation by finding a linear transformation $A(\mathcal{D}_s \rightarrow \mathcal{D}_S A)$.

Mathematical Definition:

$$\mathcal{L}_{CORAL} = ||\widehat{C_S} - C_T||_F^2$$

Where $\widehat{C_S}$ and C_T are the covariance matrices of the transformed source($\mathcal{D}_S A$) and target(D_T) features. We find an A which minimises this CORAL loss.

CORAL: Experiments

Testing the CORAL method for basic robustness on randomly synthesised linearly separable data random as well as Gaussian datasets. We checked the accuracy for source and target domains for 2 scenerios in both the cases- same covariances and different covariances.

Dataset Type	**Covariance Condition**	**Accuracy (%)**
Univariate Random	Same Covariance	100.0
Univariate Random	Different Covariance	100.0
Multivariate Normal	Same Covariance	95.80
Multivariate Normal	Different Covariance	87.39

The accuracies are averaged over 100 iterations of data generation, model training, and testing in each case in the table.

DeepCORAL

Key Idea: DeepCORAL extends the CORAL method to deep neural networks. It uses a deep network to learn a feature representation that minimizes a linear combination of the CORAL loss and the classifier loss.

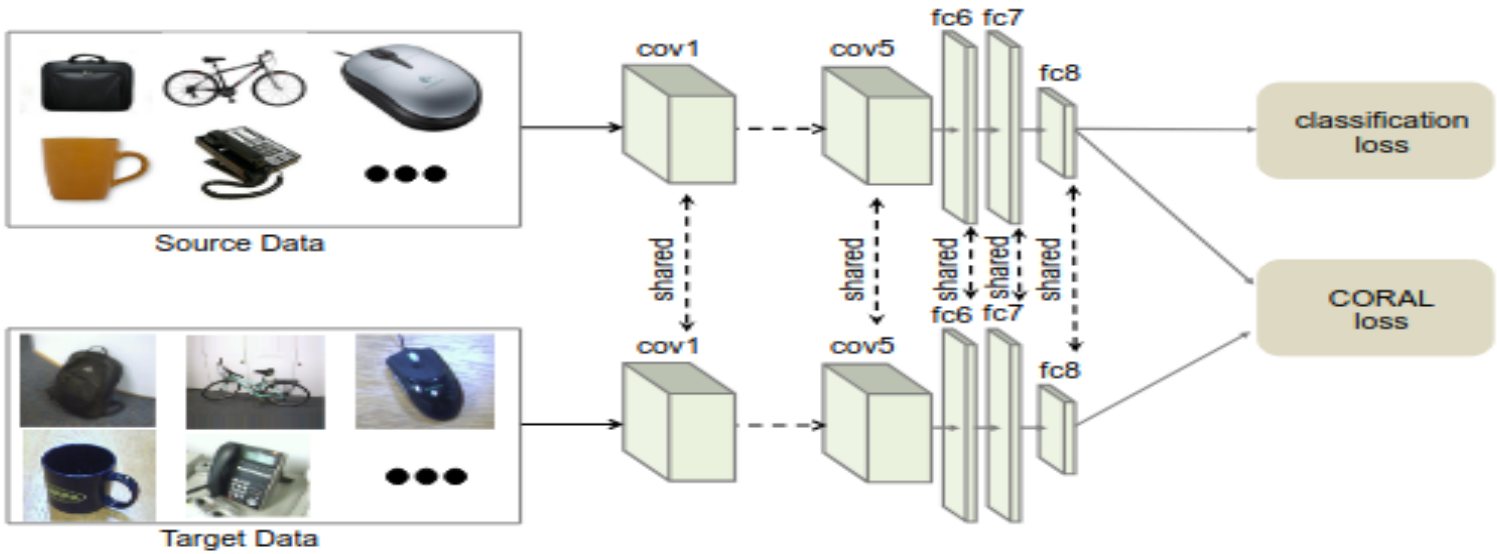


Figure: Sample architecture on a CNN(AlexNet) with a classifier layer.

- $\mathcal{L}_{CORAL} = \frac{1}{4d^2}||\widehat{C_S} - C_T||_F^2$
- Total Loss $= \mathcal{L}_{CORAL} + \sum_i \lambda_i \mathcal{L}_{CORAL}$

DeepCORAL: Experiments

Dataset: Office dataset: Amazon, DSLR, Webcam. A computer vision dataset capturing pictures of office objects under different conditions.

Table: Classification accuracy (source → target) of DeepCORAL on the Office computer vision dataset. A: Amazon, D: DSLR, W: Webcam.

Result Source	**Office (Amazon, DSLR, Webcam)**					
	A → W	**D → W**	**W → D**	**A → D**	**D → A**	**W → A**
Our Code	62.05	95.32	99.56	64.44	52.77	56.49
Survey Paper	66.4±0.4	95.7±0.3	99.2±0.1	66.8±0.6	52.8±0.2	51.5±0.3

We have compared the accuracies obtained in our experiments with those mentioned in the survey paper.

Maximum Mean Discrepancy

Concept: MMD is a statistical test to measure the distance between two distributions.

Key Idea: MMD is defined as the squared distance between the mean embeddings of two distributions in a reproducing kernel Hilbert space (RKHS).

Mathematical Definition:

$$\text{MMD}^2(\mathcal{D}_S, \mathcal{D}_T) = \left\| \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) - \frac{1}{m} \sum_{j=1}^{m} \phi(y_j) \right\|_{\mathcal{H}}^2$$

$$\text{MMD}_k^2(P, Q) := \mathbb{E}_{x,x'}[k(x, x')] + \mathbb{E}_{y,y'}[k(y, y')] - 2\mathbb{E}_{x,y}[k(x, y)]$$

Estimation of MMD

MMD can be estimated using the empirical distributions of the samples. The empirical MMD is given by:

$$X = \{x_1, x_2, \ldots, x_m\} \text{ and } Y = \{y_1, y_2, \ldots, y_m\}$$

$$\widehat{\text{MMD}}^2(P, Q) = \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} k(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} k(y_i, y_j) - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} k(x_i, y_j)$$

Key Idea: The empirical MMD is an unbiased estimator of the true MMD. From this we can define a test statistic:

$$T = \frac{\hat{\text{MMD}}^2(X, Y)}{\sqrt{\hat{V}_m(X, Y)}}$$

Where V_m is unbiased estimator of the variance of the MMD.
Key Idea: If T is large, then the two distributions are likely different.

Experiments on MMD

We have used synthetic data to test MMD. The experiment uses two different distributions:

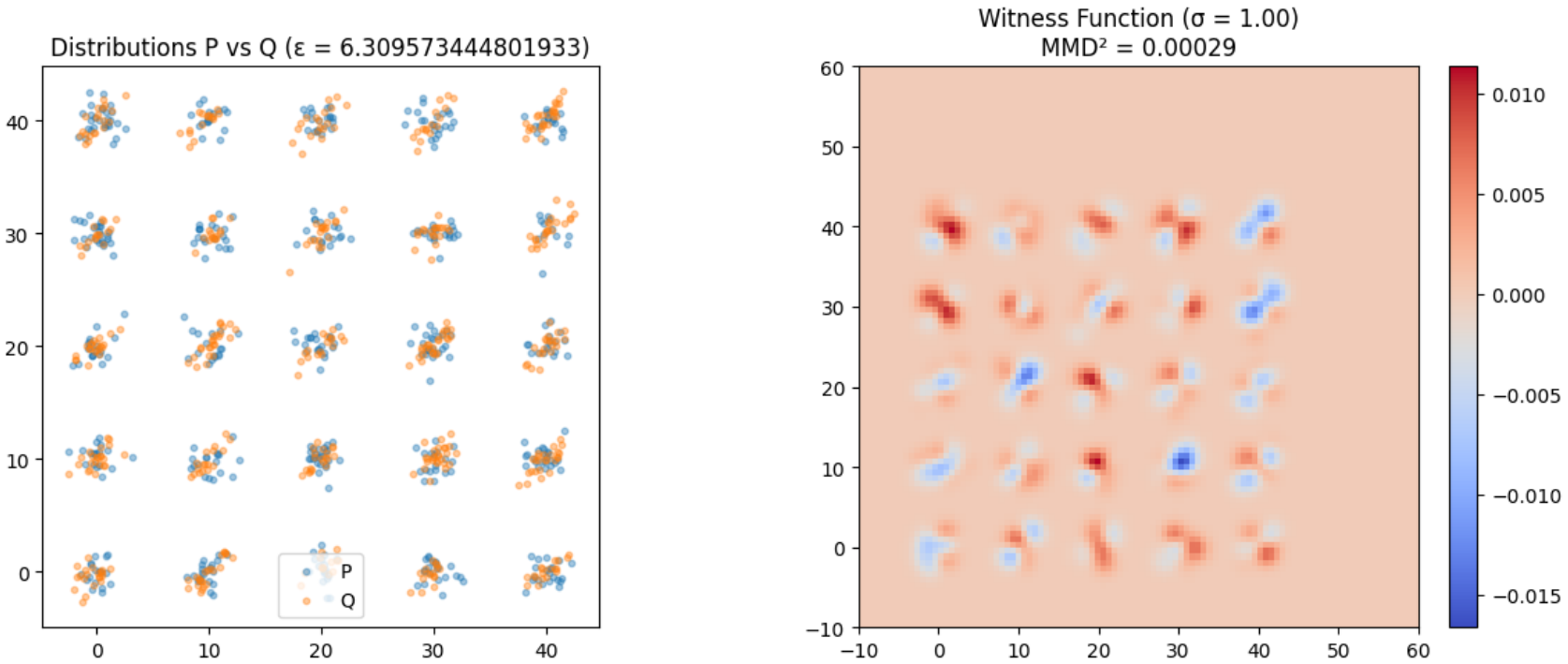


Figure: Synthetic datasets used for MMD experiments.

Results of MMD

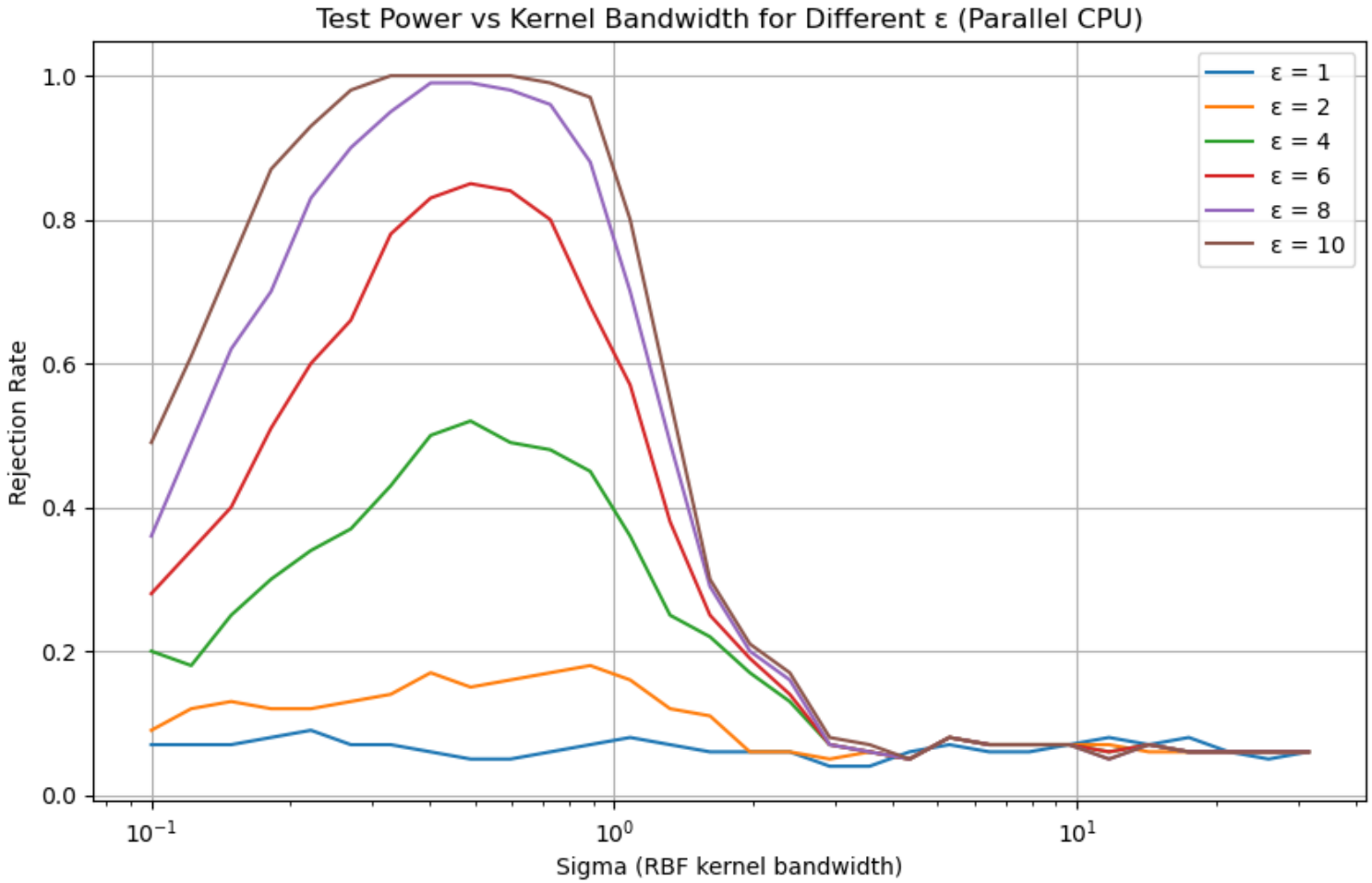
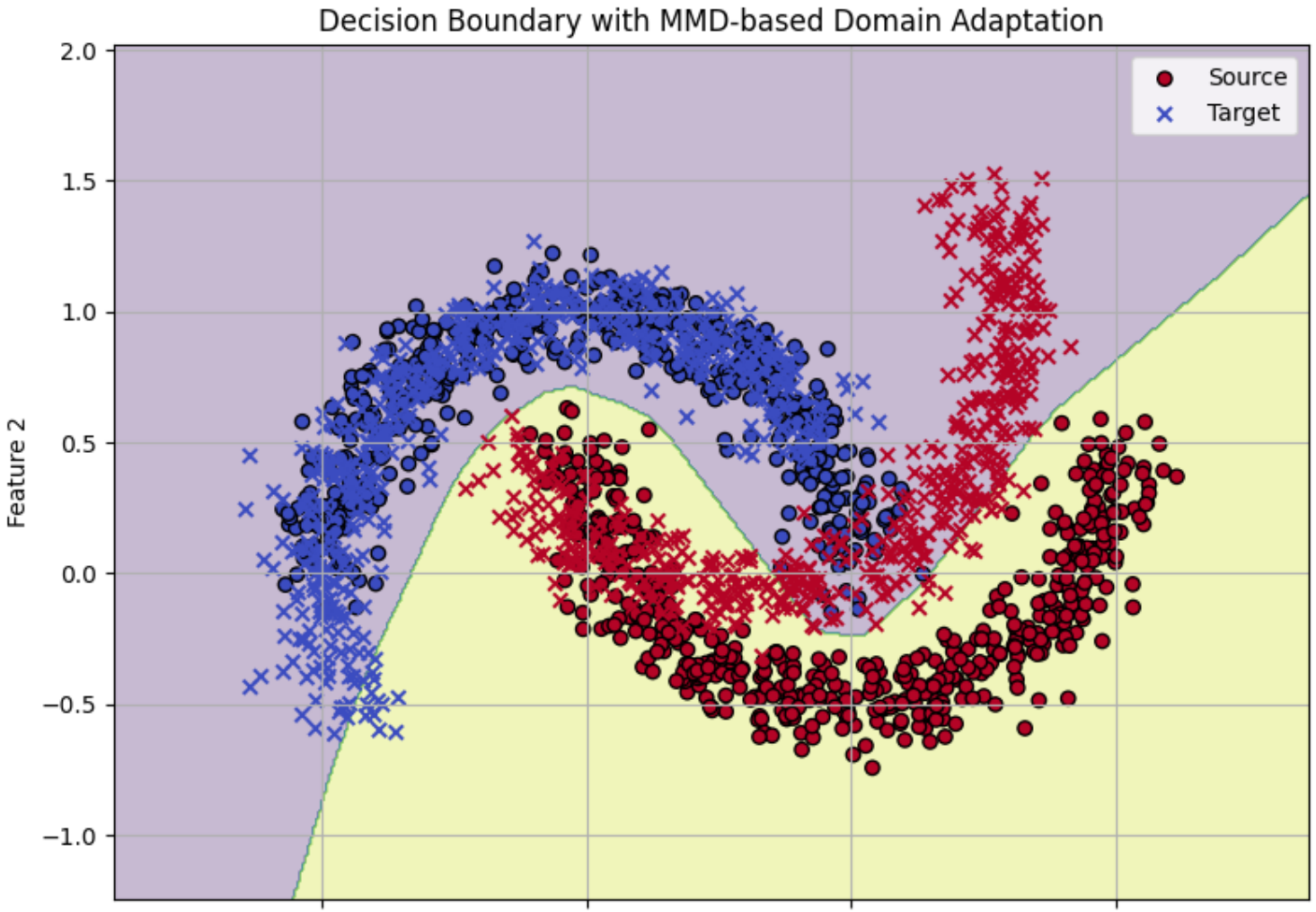


Figure: Test power vs ϵ (epsilon). As the difference between distributions increases, MMD becomes more effective at distinguishing them.

MMD in UDA

Key Idea: MMD can be used as a loss function to minimize the distance between the source and target distributions.



Domain Separation Network

Motivation: Obtaining large datasets is very difficult and expensive. So we wish to use the data from a synthetically generated source and then adapt it to the real world.

Idea: In this method we learn two types of features:

- **Domain Invariant Features:** These features are common to both the source and target domains. They are used to classify the data.
- **Domain Specific Features:** These features are specific to the source or target domain. They are used to separate the two domains.

DSN Architecture

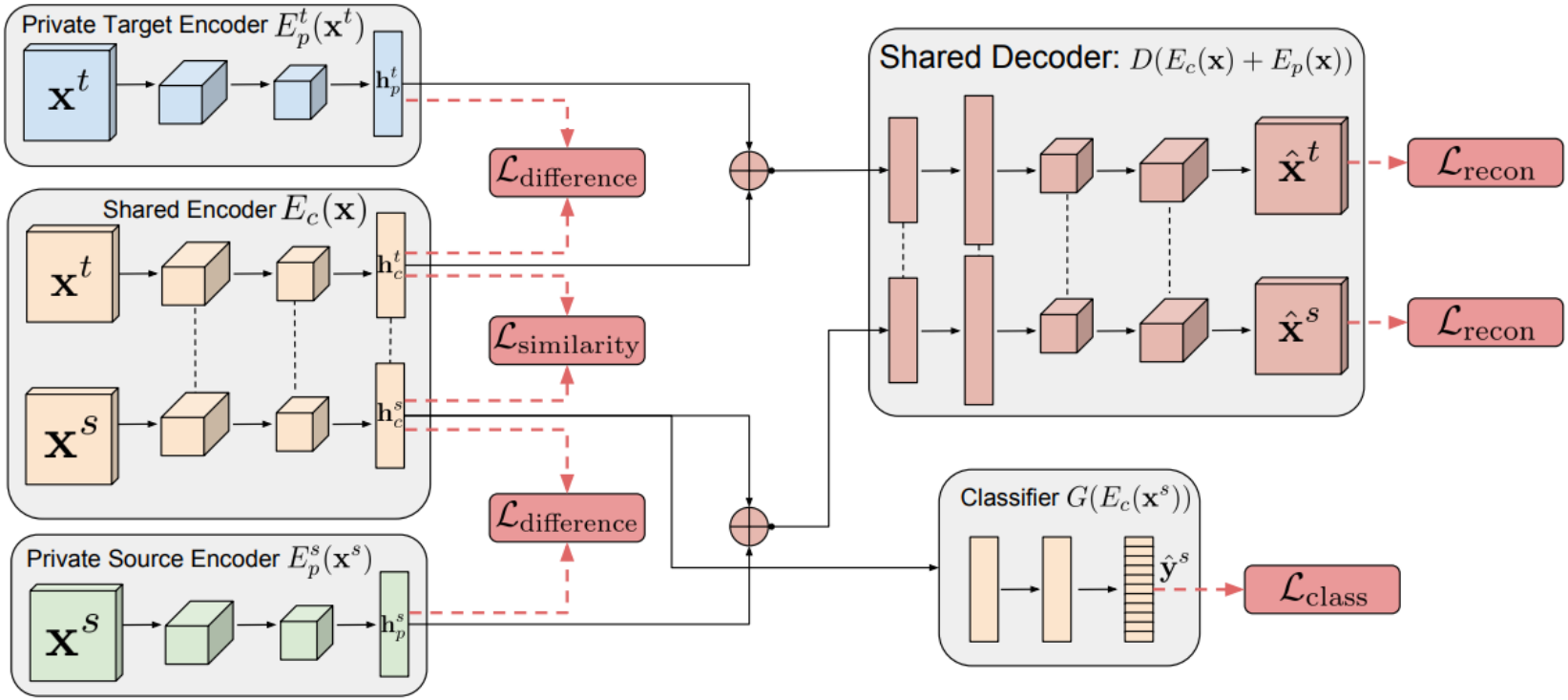


Figure: Domain Separation Network Architecture. The network learns both domain-invariant and domain-specific features through separate encoders.

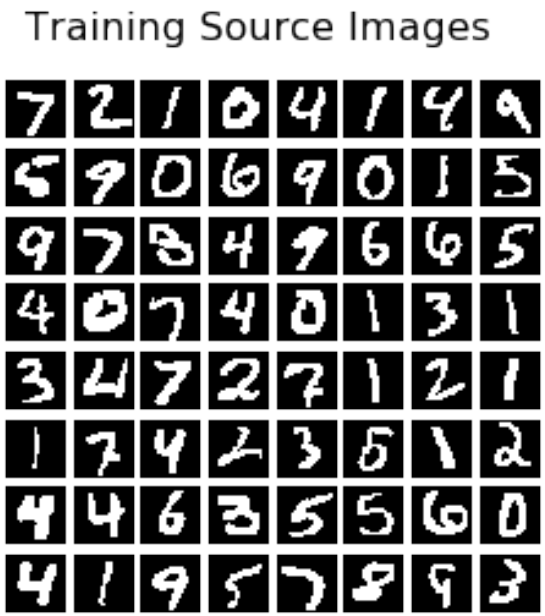
Experiments on DSN

Dataset: We used the MNIST and MNIST-M datasets for the experiments.

Method	**Accuracy**
DSN (Ours)	81.6%
DSN (Paper)	83.2%

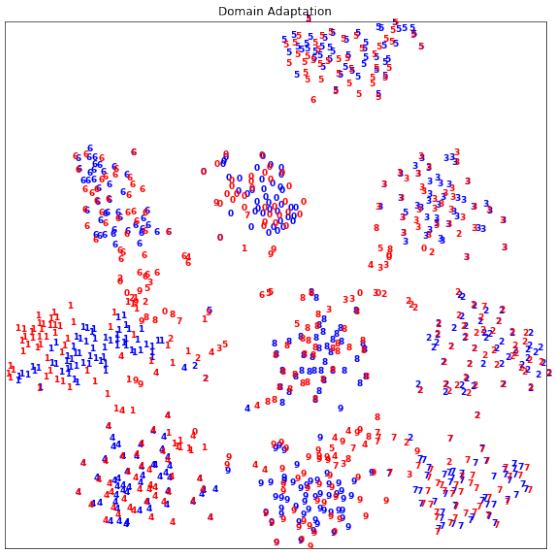
Table: Results comparing our DSN implementation with original paper

Results of DSN



(a) Source Images

(b) Target Images



(c) Classification Results

Results of DSN

(a) Reconstructed Source Images

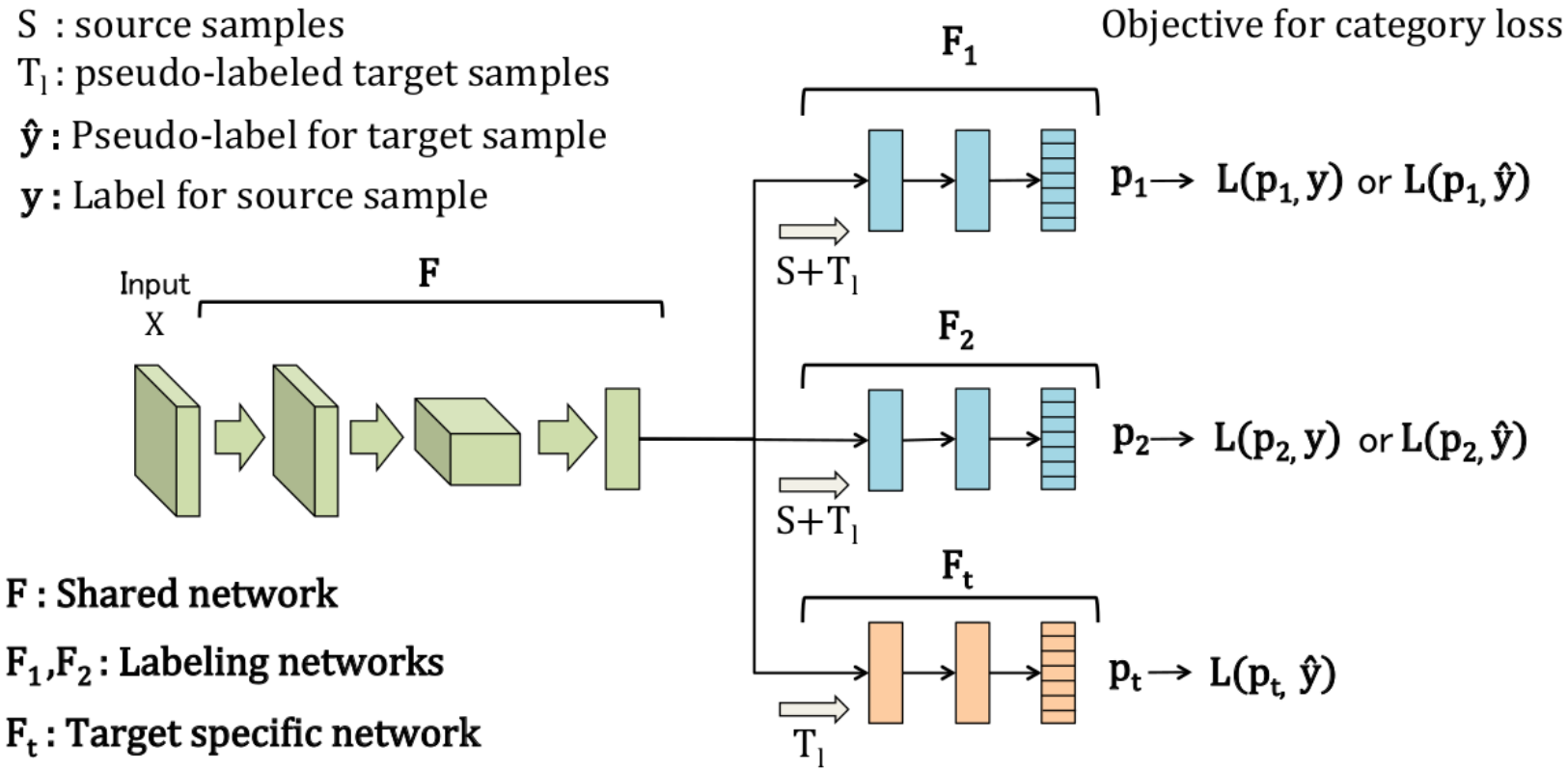
(b) Reconstructed Target Images

Figure: Experiments on DSN with MNIST and MNIST-M datasets

Asymmetric Tri-training

Ensemble model approach to UDA. Utilizing a feature extractor F and three classifiers F_1, F_2, F_t.

Key Idea: Use two classifiers to pseudo-label the target domain and train a third classifier on the pseudo-labeled data.



MNIST - SVHN Domain Adaptation

Dataset: MNIST and SVHN are two datasets used for image classification. MNIST contains handwritten digits, while SVHN contains street view house numbers.

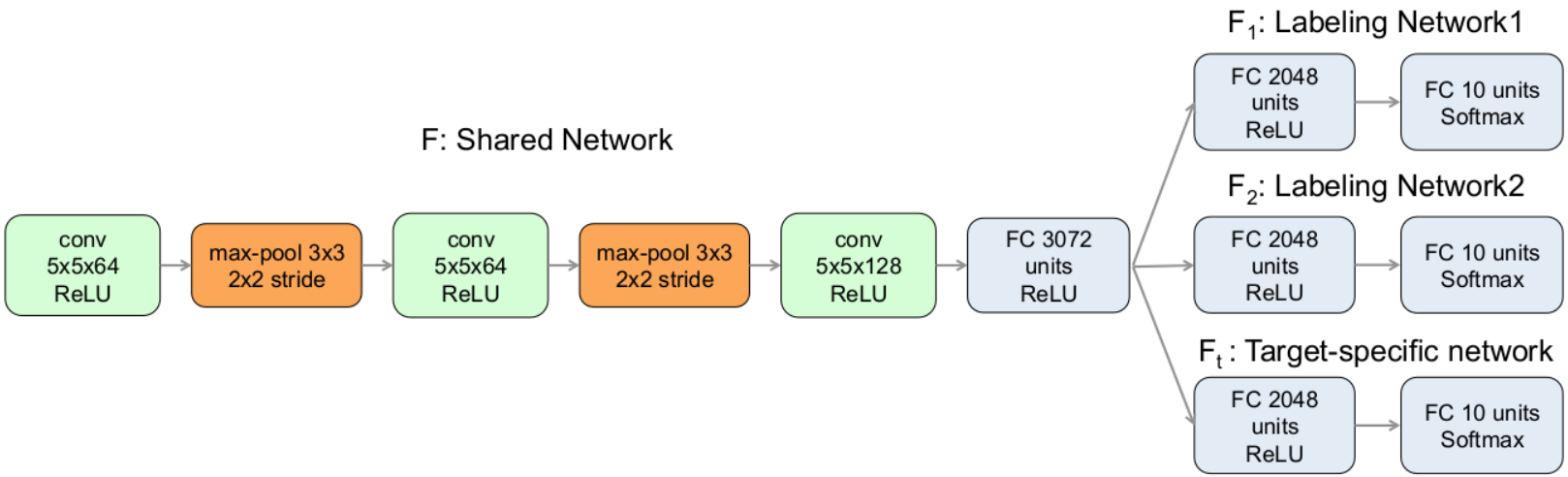


Figure: The architecture used for training SVHN

Algorithm Overview

Key Points:

- **Loss Function for F1 and F2:**

$$\mathcal{L}_{\Theta_F,\Theta_{F1},\Theta_{F2}} = \frac{1}{n}\sum_{i=1}^{n}[L_y(F_1(F(x_i)); y_i) + L_y(F_2(F(x_i)); y_i)] + \lambda|W_1^T W_2|$$

- **Loss Function for Ft:**

$$\mathcal{L}(\theta_F, \theta_{F_t}) = \mathcal{L}_{CE}(F_t(x_t), \hat{y}_t)$$

Algorithm 1 *iter* denotes the iteration of the training. The function *Labeling* indicates the labeling method. We assign pseudo-labels to samples when the predictions of F_1 and F_2 agree, and at least one of them is confident of their predictions.

Input: data
$\mathbf{X^s} = \{(x_i, t_i)\}_{i=1}^{m}$, $\mathbf{X^t} = \{(x_j)\}_{j=1}^{n}$
$\mathbf{X^t}_l = \emptyset$
for $j = 1$ **to** *iter* **do**
  Train F, F_1, F_2, F_t with a mini-batch from the training set $\mathcal{S}$
end for
$N_t = N_{init}$
$\mathbf{X^t}_l$ = Labeling($F, F_1, F_2, \mathbf{X^t}, N_t$)
$\mathcal{L} = \mathbf{X^s} \cup \mathbf{X^t}_l$
for K steps **do**
  for $j = 1$ **to** *iter* **do**
    Train F, F_1, F_2 with mini-batch from training set $\mathcal{L}$
    Train F, F_t with mini-batch from training set $\mathbf{X^t}_l$
  end for
  $\mathbf{X^t}_l = \emptyset$, $N_t = K/20 * n$
  $\mathbf{X^t}_l$ = Labeling($F, F_1, F_2, \mathbf{X^t}, N_t$)
  $\mathcal{L} = \mathbf{X^s} \cup \mathbf{X^t}_l$
end for

Results of ATT on MNIST and SVHN Domain Adaptation

Table: Results of ATT on MNIST and SVHN Domain Adaptation.

Method	**MNIST→SVHN**	**SVHN → MNIST**
Our w/o Batch Normalization	36.9%	76.8%
Ours w/o Multi-view Loss	15.2%	76.5%
Ours	15.2%	71.4%
Papers w/o Batch Normalization	39.8%	79.8%
Papers w/o Multi-view Loss	49.7%	86.0%
Papers	52.8%	85.8%

Contributions

Pratham Gupta:

- Codes for MMD and ATT algorithms
- Complete Main Section of Report
- MMD and ATT in Appendix Section of Report
- Presentation Slides of Introduction, DSN, MMD and ATT
- Video Editing
- Maintained Repository of the project
- Worked on DANN Re-identification Code

Gavish Bansal:

- Codes for Sentiment Analysis and Re-identification Code
- Complete Appendix Section of DANN and Theory paper in Report
- Prepared Presentation for DANN part

Contributions (contd.)

Kintan Saha:

- Code for the inter-twined moons setup
- Code for image classification experiments of the DANN paper
- Writing the Appendix section for the DANN paper and the theory paper
- Helping in preparing the presentation slides for the DANN part

Krishna Agarwal:

- Codes for CORAL, DeepCORAL, and DSN
- Helped in writing sections of CORAL, DeepCORAL, and DSN in the main report
- Writing the Appendix section for the Survey paper
- Presentation slides for CORAL and DeepCORAL

Thank You!

Any Questions?