Deep Learning: Domain Adaptation

Vasileios Ilias Drouzas
Giannis Trantalidis

June 29, 2024

Athens University of Economics and Business
MSc in Data Science

Contents

1 Introduction to Domain Adaptation

Domain adaptation is a subset of transfer learning aimed at addressing the problem of domain shift, where the distribution of the source domain differs from that of the target domain. This report delves into various techniques and methodologies employed to mitigate this issue, with a focus on adapting deep learning models to new domains effectively.

2 Datasets Used

2.1 MNIST

MNIST dataset is a widely used dataset consisting of 60,000 grayscale images of handwritten digits, each 28x28 pixels. The digits are centered in the images and presented in a grayscale format. Primarily used for training and testing in the field of machine learning, MNIST is particularly valuable for image processing and computer vision tasks such as digit recognition. The dataset is divided into a training set of 60,000 examples and a test set of 10,000 examples. The digits have been size-normalized and centered in a fixed-size image (Figure 1).

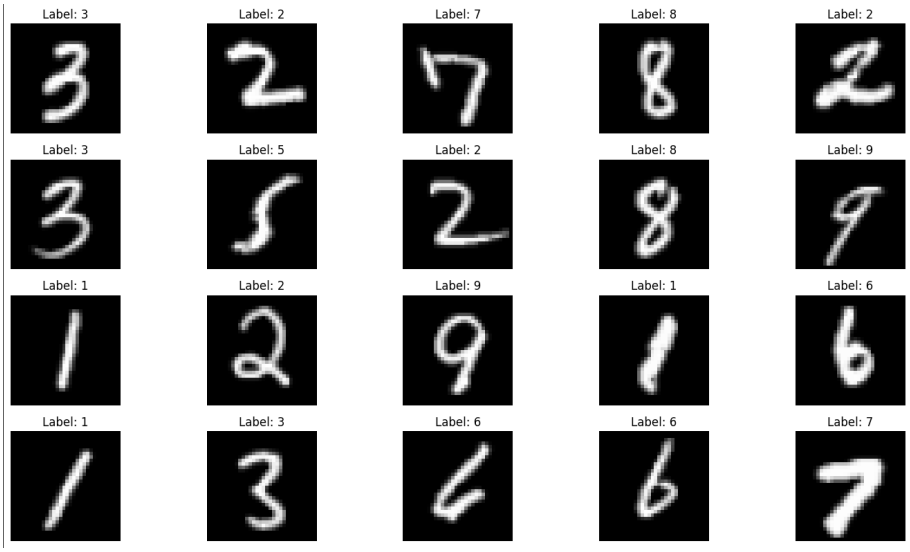


Figure 1: MNIST dataset.

2.2 SVHN (Street View House Numbers)

SVHN (Street View House Numbers) dataset contains 73,257 RGB images of house numbers captured in natural scenes, each 32x32 pixels. Unlike MNIST, the digits in SVHN are non-centered and presented in three RGB channels. This dataset is useful for developing algorithms capable of recognizing digits in natural scenes and is often used to test the robustness of models to real-world conditions. SVHN includes a training set of 73,257 digits, a test set of 26,032

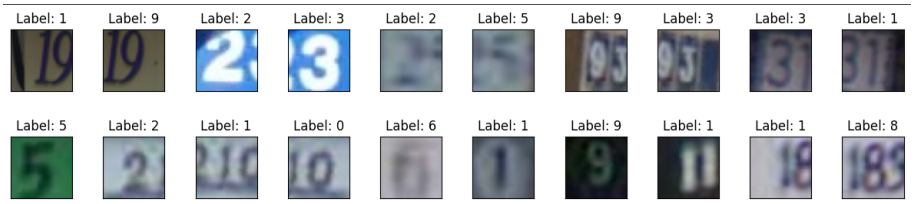


Figure 2: SVHN dataset.

digits, and an extra set of 531,131 less difficult samples. Each image comes with bounding box annotations for each digit, providing additional challenges for object detection tasks. The images are in PNG format, and the dataset is provided in MATLAB format for ease of use. The non-centered and varying illumination conditions in the images make this dataset more challenging compared to MNIST. (Figure 2).

2.3 USPS

USPS dataset is a digit dataset scanned from U.S. postal envelopes, consisting of 9,298 samples. The images are 16x16 pixels and in grayscale format. The USPS dataset is often used for comparing algorithms' performance on smaller, less complex datasets and is useful for tasks requiring lower-resolution images. It contains 7,291 training images and 2,007 test images. The images have been resized to 16x16 pixels and converted to grayscale, maintaining simplicity for quick prototyping. The dataset is typically available in CSV format or as binary files. The lower resolution and simpler preprocessing make it a good dataset for testing basic image processing and classification algorithms. (Figure 3).

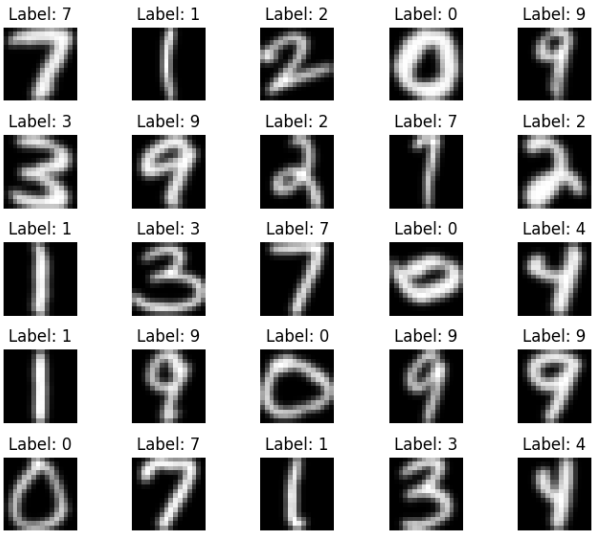


Figure 3: USPS dataset.

3 Pre-processing Techniques

3.1 Dimension Matching

One of the primary challenges in domain adaptation is ensuring that the input dimensions of different datasets match. The MNIST dataset consists of 28x28 pixel images, while the SVHN dataset consists of 32x32 pixel images. To make these datasets compatible:

- **Padding**: We pad the MNIST images with zeros to increase their size from 28x28 pixels to 32x32 pixels. This ensures that the dimensions of MNIST images match those of SVHN.
- **Implementation Details**: Padding is applied uniformly around the original images, adding two pixels of padding to each side (top, bottom, left, and right) to increase the dimensions evenly and maintain the central position of the digits.

3.2 Channel Matching

The MNIST dataset consists of grayscale images (single channel), while the SVHN dataset consists of RGB images (three channels). To address this discrepancy:

- **Channel Replication**: The single grayscale channel of MNIST images is repeated three times to create a three-channel image. This is done to simulate RGB images, allowing grayscale images to be processed by models designed for three-channel inputs.
- **Implementation Details**: The grayscale pixel values are copied into three separate channels (red, green, and blue), resulting in an image where all three channels have identical values.

3.3 Normalization

Normalization is a crucial step in pre-processing to ensure that the input data is on a similar scale, which can lead to faster convergence during training and better performance:

- **Scaling to Range [-1, 1]**: All pixel values in the datasets are scaled to the range [-1, 1]. This involves two main steps:
    1. **Rescaling**: First, pixel values are rescaled from their original range of [0, 255] (for MNIST, which is grayscale, and SVHN, which is RGB) to [0, 1].
    2. **Shifting**: The rescaled values are then shifted to the range [-1, 1] by multiplying by 2 and subtracting 1. This ensures that the pixel values are centered around zero, which is beneficial for training neural networks as it can help with the stability of gradient descent.

4 Algorithms

Here, we examine the algorithms that were tested: Source-only (simple CNN), DANN, deep CORAL.

4.1 Simple CNN (Source Only)

A basic convolutional neural network was implemented as a baseline, consisting of:

- **Convolutional Layer**: 3 input channels, 16 output channels, 3x3 kernel size.
- **Max-Pooling Layer**
- **Fully Connected Layer**
- **Loss Function**: CrossEntropyLoss
- **Optimizer**: Adam (learning rate: 0.001)

4.2 Domain-Adversarial Neural Network (DANN)

DANN aims to learn domain-invariant features through the following components:

- **Feature Extractor:**
  - Extracts features from the input images.
  - Implemented by the `FeatureExtractor` class.
  - Composed of convolutional layers, ReLU activations, max pooling layers, batch normalization, and an adaptive average pooling layer to reduce spatial dimensions.
- **Label Classifier:**
  - Predicts class labels from the extracted features.
  - Implemented by the `LabelClassifier` class.
  - Composed of fully connected layers, ReLU activations, and batch normalization.
- **Domain Classifier with Gradient Reversal Layer:**
  - Predicts the domain (source or target) from the extracted features.
  - Implemented by the `DomainClassifier` class.
  - Uses a gradient reversal layer (`GradReverse` function) to reverse the gradient sign during backpropagation. . During the backward pass, the gradient is multiplied by a negative scalar ($-\lambda$), where $\lambda = 1$. This forces the feature extractor to learn domain-invariant features.
  - Composed of fully connected layers, ReLU activations, batch normalization, and a final sigmoid activation to output domain probabilities.

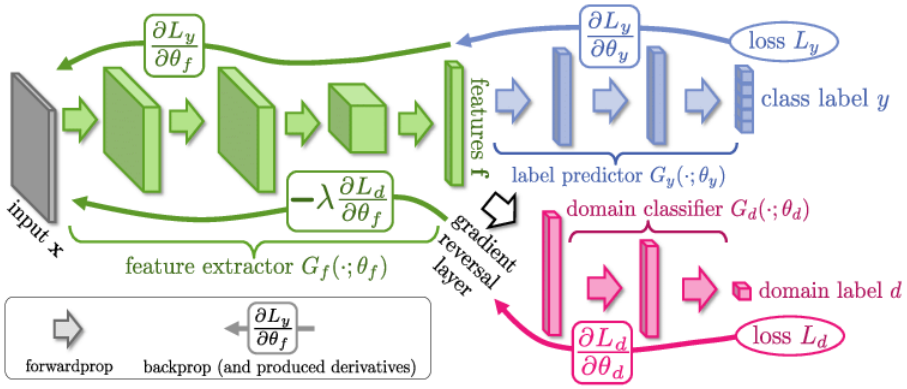


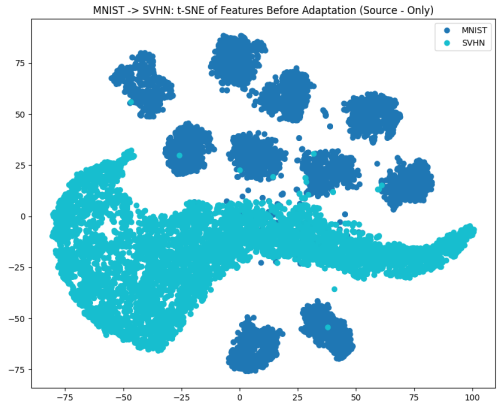
Figure 4: Domain Adversarial Neural Network (DANN).

Training process:

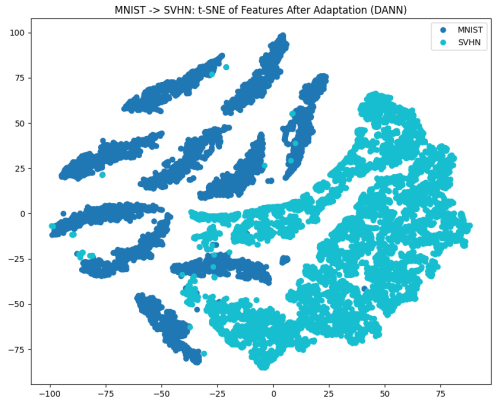
- **Feature Extraction:** The `FeatureExtractor` processes input images to generate feature representations.
- **Label Prediction:** The `LabelClassifier` predicts class labels based on these features.
- **Domain Prediction:** The `DomainClassifier`, with the gradient reversal layer, predicts the domain of the features. The gradient reversal layer ensures that during backpropagation, the feature extractor is trained to produce features that confuse the domain classifier, encouraging domain-invariant feature learning.

While examining these algorithms, we will also test the feature representations (t-SNE) of the two datasets. The more overlap is noticed, the more successful adaptation has been performed from the source to the target dataset. In Figures 5, 6, we demonstrate the feature representation after performing DANN. DANN manages to perform well on the SVHN $\rightarrow$ MNIST case, but not on the opposite one. However, DANN excels in both ways regarding the MNIST and USPS datasets.

MNIST → SVHN

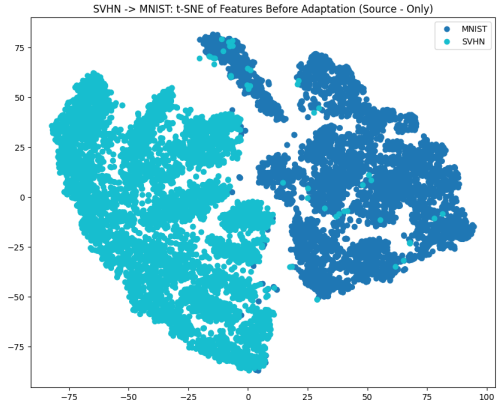


(a) Non-adapted

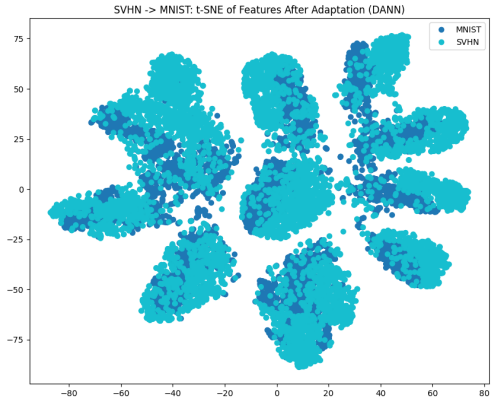


(b) Adapted

SVHN → MNIST



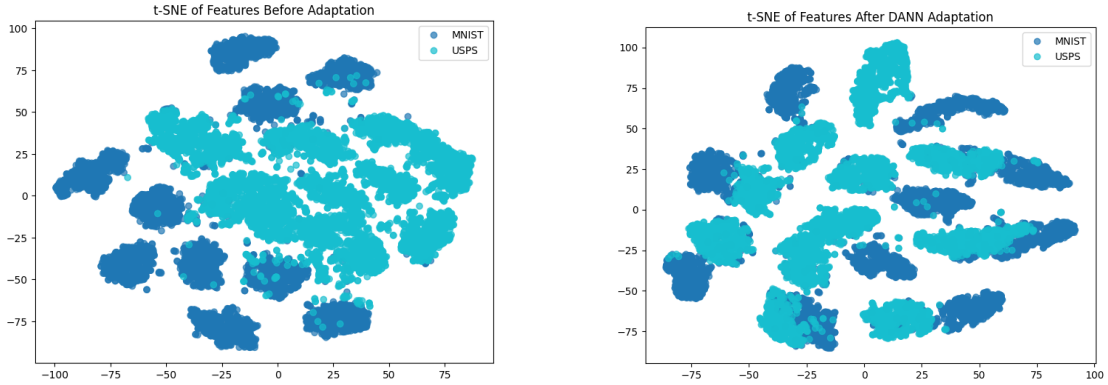
(a) Non-adapted



(b) Adapted

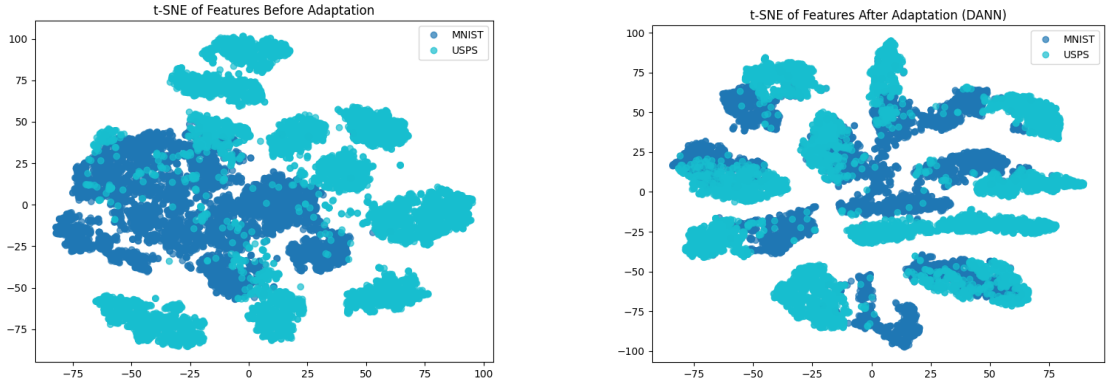
Figure 5: The effect of adaptation (MNIST, SVHN).

MNIST → USPS



(a) Non-adapted (b) Adapted

USPS → MNIST



(a) Non-adapted (b) Adapted

Figure 6: The effect of adaptation (MNIST, USPS).

4.3 Deep CORAL

CORAL minimizes domain shift by aligning the second-order statistics (covariances) of the source and target domains using:

- **CORAL Loss**: Squared Frobenius norm between batch covariances, normalized to minimize the domain shift.

During the training process, we use as the loss a combination of these three losses:

- Class Loss: ensures the model is good at predicting the correct labels for the source domain.
- CORAL Loss: reduces the feature distribution discrepancy between the source and target domains.

- Domain Loss: encourages the feature extractor to produce domain-invariant features.

The rest of the CORAL model is identical to DANN (same feature extractor, domain/label classifier, gradient reversal layer etc.)

5 Benchmarking Results

The performance of different algorithms was evaluated on various domain adaptation tasks, with the following accuracy scores:

Algorithm	**MNIST → SVHN**	**SVHN → MNIST**	**MNIST → USPS**	**USPS → MNIST**
Source Only	**18.83**%	53.42%	51.97%	33%
DANN	18.70%	**81.29**%	**83.91**%	**92.02**%
CORAL	18.57%	63.84%	82.81%	60.80%

Table 1: Benchmarking Results

- **DANN** consistently outperforms Source Only and CORAL across all tasks. Its adversarial learning framework enables it to learn robust domain-invariant representations, significantly improving adaptation accuracy.

  Domain-Adversarial Neural Networks (DANN) excel in learning features that are invariant across different domains. The superior performance of DANN across all adaptation tasks underscores its ability to handle complex domain shifts and generalize well to new datasets, making it a powerful tool for domain adaptation in machine learning.

- **CORAL** shows competitive performance, particularly in aligning statistical features (second-order statistics) between domains. However, it generally falls slightly short compared to DANN, especially in tasks with more complex domain shifts.

  CORrelation ALignment (CORAL) focuses on aligning the covariance matrices of source and target domains to reduce domain shift. By minimizing the distribution mismatch between domains at a statistical level, CORAL aims to enhance adaptation performance. Nonetheless, CORAL remains a valuable approach for domains with clear statistical differences and provides a solid baseline for comparison against more advanced methods like DANN.

- **Source Only** performs the poorest in most cases, highlighting the necessity of domain adaptation techniques to handle domain shift effectively.

  Source Only models, which are trained solely on the source domain data without adaptation, typically struggle when applied directly to target domains with significant differences. This is also the case here, our baseline source-only CNN model fails to generalize well beyond the source domain distribution.

These results underscore the effectiveness of advanced domain adaptation techniques like DANN and CORAL in improving model performance across different datasets and domains. State-of-the-art scores of several methods are shown in Figure (Figure 7).

Methods	Source Target	MNIST USPS	USPS MNIST	SVHN MNIST	MNIST SVHN
Source Only		78.9	57.1±1.7	60.1±1.1	20.23±1.8
w/o augmentation					
CORAL [43]		81.7	-	63.1	-
MMD [48]		81.1	-	71.1	-
DANN [10]		85.1	73.0±2.0	73.9	35.7
DSN [2]		91.3	-	82.7	-
CoGAN [25]		91.2	89.1±0.8	-	-
ADDA [49]		89.4±0.2	90.1±0.8	76.0±1.8	-
DRCN [11]		91.8±0.1	73.7±0.1	82.0±0.2	40.1±0.1
ATT [37]		-	-	86.20	52.8
ADA [13]		-	-	97.6	-
AutoDIAL [3]		97.96	97.51	89.12	10.78
SBADA-GAN [35]		97.6	95.0	76.1	**61.1**
GAM [16]		95.7±0.5	98.0±0.5	74.6±1.1	-
MECA [32]		-	-	95.2	-
DWT		**99.09**±0.09	**98.79**±0.05	**97.75**±0.10	28.92 ±1.9
Target Only		96.5	99.2	99.5	96.7

Figure 7: State-of-the-art scores.

6 Conclusions

Adapting from MNIST to SVHN is a challenging task due to the significant differences in the nature of the two datasets (MNIST being grayscale handwritten digits and SVHN being colored images of house numbers).

For the opposite task (SVHN to MNIST), DANN is highly effective in adapting features from the more complex SVHN dataset to the simpler MNIST dataset. The domain difference is not that evident between MNIST & USPS, with good results for both directions.

DANN shows the most consistent and significant improvements across different tasks, particularly in scenarios where the source and target datasets have different complexities.

CORAL is also effective, particularly for tasks like MNIST $\rightarrow$ USPS, though it generally lags behind DANN.

Source Only generally performs poorly, highlighting the need for domain adaptation methods to handle domain shifts effectively.

7 References

[1] Ganin, Yaroslav, Ustinova, Evgeniya, Ajakan, Hana, Germain, Pascal, Larochelle, Hugo, Laviolette, François, Marchand, Mario, and Lempitsky, Victor. **"Domain-Adversarial Training of Neural Networks."** Journal of Machine Learning Research, vol. 17, no. 59, 2016.

[2] Sun, Baochen, and Saenko, Kate. **"Deep CORAL: Correlation Alignment for Deep Domain Adaptation."** Computer Vision–ECCV 2016 Workshops, Springer, 2016.

[3] Papers With Code. Domain Adaptation.

[4] Bhagwat, S., & Karmalkar, S. (2019). **"Optimal power allocation and user scheduling in NOMA systems: A solution based on bipartite graph matching"**. arXiv.