Domain Adaptation

Domain Adaptation (DA) allows machine learning methods trained on data sampled from one distribution to be applied to data sampled from another.

One-Step DA

The source and target domains are directly related, allowing for knowledge transfer in one step.

Multi-Step DA

Build a series of intermediate bridges to connect two seemingly unrelated domains, then perform one-step DA via this bridge.

One-Step Domain Adaptation

One-step domain adaptation can be categorised into three approaches, each of which can be further split into subcategories as shown in table I.

TABLE I
DIFFERENT DEEP APPROACHES TO ONE-STEP DA

One-step DA Approaches	Brief Description	Subsettings
Discrepancy-based	fine-tuning the deep network with labeled or unlabeled target data to diminish the domain shift	class criterion [118], [86], [79], [98] [53], [45], [75], [139], [130], [29], [118], [28]
		statistic criterion [74], [130], [73] [75], [120], [32], [109], [87], [144]
		architecture criterion [69], [54], [68], [95], [128], [89]
		geometric criterion [16]
Adversarial-based	using domain discriminators to encourage domain confusion through an adversarial objective	generative models [70], [4], [57]
		non-generative models [119], [118], [26], [25], [117] [85]
Reconstruction-based	using the data reconstruction as an auxiliary task to ensure feature invariance	encoder-decoder reconstruction [5], [33], [31], [144]
		adversarial reconstruction [131], [143], [59]

1.1 Discrepancy-Based DA

Assumes that fine-tuning the deep network model with labelled or unlabelled target data can diminish the shift between the two domains.

1.1.1 Class Criterion

Uses the class label information as a guide for transferring knowledge between different domains. When such samples are unavailable, some other techniques can be adopted to substitute for class labelled data, such as pseudo labels and attribute representation.

Soft Label Loss

Introduced by Geoff Hinton (of course), a "soft" softmax can be used when training on the new domain as shown in equation 1.1, where T is the temperature that's normally set to 1 in standard softmax. Larger values of T produce a softer probability distribution over classes.

$$q_i = \frac{exp(z_j/T)}{\sum_j exp(z_j/T)} \tag{1.1}$$

Originally used in a paper about distilling knowledge in other networks, using soft labels rather than hard labels can preserve relationships between classes across domains. It is recommended that this is used in conjunction with the domain confusion loss (discussed later).

From [18]:

The bottle soft label will have a higher weight on mug than on keyboard, since bottles and mugs are more visually similar. Thus, soft label training with this particular soft label directly enforces the relationship that bottles and mugs should be closer in feature space than bottles and keyboards.

... we ensure that the parameters for categories without any labelled target data are still updated to output non-zero probabilities.

Embedded Metric Learning

Unified Deep Supervised Domain Adaptation and Generalization[12] uses a siamese network to process source and target domain examples simultaneously, and applies a *Contrastive Semantic Alignment Loss* to the embedded examples, minimising the dissimilarity across domains.

Deep Transfer Metric Learning[9] performs something similar, and uses training objectives that:

1. maximise inter-class variations and minimise intra-class variations at arbitrary layers
2. minimise the distribution divergence between the source domain and the target domain at the penultimate layer of the network

1.1.2 Statistic Criterion

Aligns the statistical distribution shift between the source and target domains using some mechanisms.

A frequently used metric is **Maximum Mean Discrepancy (MMD)**, which estimates the distance between different distributions.

Deep Domain Confusion: Maximizing for Domain Invariance[17] applies an *adaptation layer* (bottleneck) at some point in the network, then jointly minimises classification error and MMD at the adaptation layer. The minimisation of MMD between source and target domains is known as **domain confusion**.

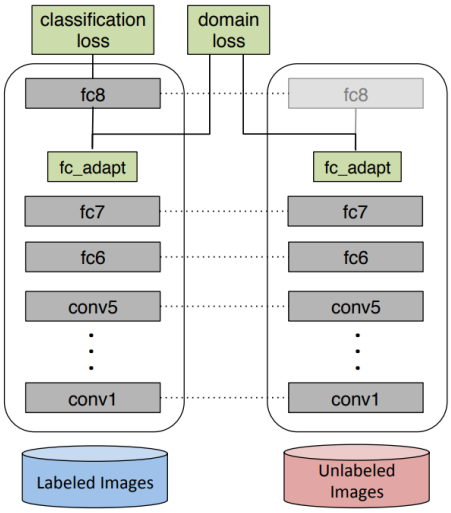

Figure 1.1: Architecture of Deep Domain Confusion[17]

Learning Transferable Features with Deep Adaptation Networks[11] takes it one step further by applying domain confusion to several layers of a network.

Deep CORAL[16] is similar to the previous systems, but instead of minimising MMD, minimises the difference in second-order statistics between the source and target feature activations (the learned feature covariances). Similarly, this can be applied to any layer of a network.

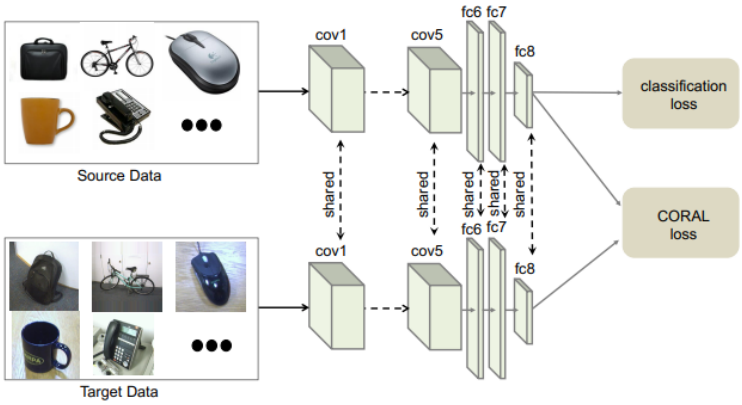

Figure 1.2: Deep CORAL architecture[16]

1.1.3 Architecture Criterion

> *Aims at improving the ability of learning more transferable features by adjusting the architectures of deep networks. The techniques that are proven to be cost effective include adaptive batch normalisation (BN), weak-related weight and domain-guided dropout.*

Instead of transfer-learning by fine-tuning weights to a target domain, *Beyond Sharing Weights for Deep Domain Adaptation*[14] adopts a dual-stream network architecture where select layers in the target classifier has distinct weights, which are regularised such that they have a similar distribution as those in the source domain model.

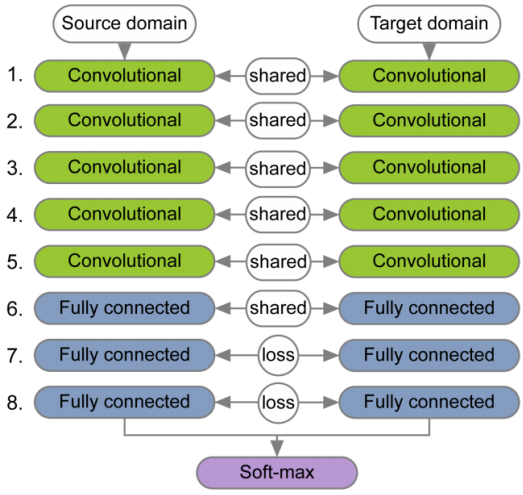

Figure 1.3: *Beyond Sharing Weights* architecture. The layers indicated by "loss" do not share weights, but are optimised to have similar distributions.

Revisiting Batch Normalization For Practical Domain Adaptation[10] hypothesizes that "the label related knowledge is stored in the weight matrix of each layer, whereas domain related knowledge is represented by the statistics of the Batch Normalization layer". They recommend the usage of **Adaptive Batch Normalization** to normalise features between domains:

$$BN(X^t) = \lambda\left(\frac{x - \mu(X^t)}{\sigma(X^t)}\right) + \beta \tag{1.2}$$

where λ and β are parameters learned from the *target* data.

AutoDIAL: Automatic DomaIn Alignment Layers[3] feeds source/target domain images into a network in parallel and inserts **Domain Alignment (DA) layers** (similar to batch norm) to enforce feature similarity across domains (figure 1.4). A learned parameter controls this "degree of domain alignment" as features pass through each of the DA layers.

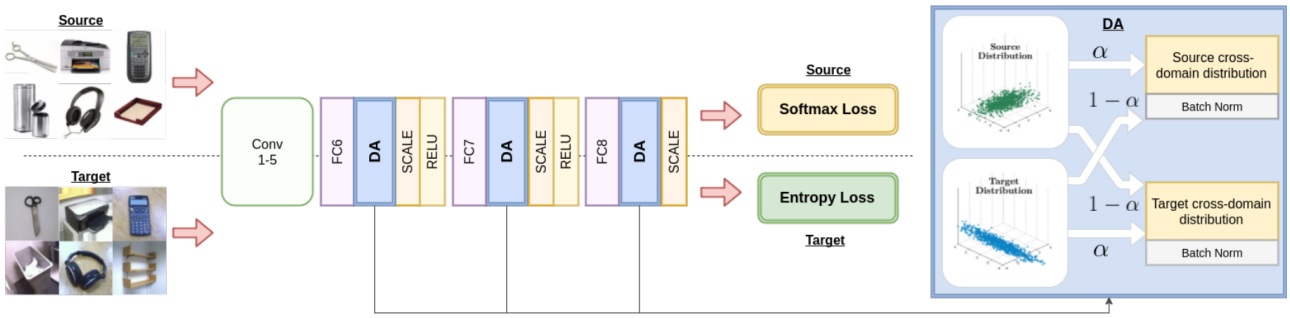

Figure 1.4: AutoDIAL aligns source/target distributions according to a learnt α at each DA layer

The survey[20] states that *Improved Texture Networks*[19] shows **Instance Normalisation** performs better at DA than standard Batch Normalisation... but I couldn't find that in the paper.

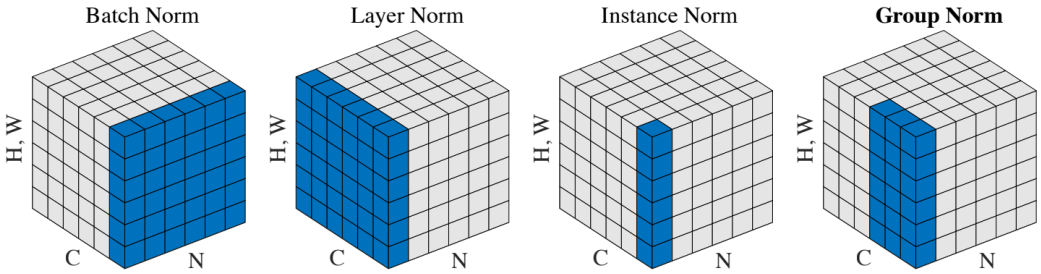

Figure 1.5:

Learning Deep Feature Representations with Domain Guided Dropout for Person Re-identification[22] introduces **Domain Guided Dropout**, which is essentially a method of applying dropout to irrelevant/unhelpful neurons per-domain. It's used like so:

1. Train on all domains
2. For each entry in a chosen feature vector:
3. The impact score of this neuron is $\mathcal{L}(g(x)_{\setminus i}) - \mathcal{L}(g(x))$, where $g(x)_{\setminus i}$ is the feature vector after zeroing-out the i-th neuron response to zero, averaged over all images in each domain
4. Continue training, but use the scores to guide dropout for each domain

This comes in two flavours: deterministic and stochastic. Deterministic applies the resultant mask when the score is < 0, stochastic uses the score to sample from a Bernoulli distribution.

1.1.4 Geometric Criterion

> *Bridges the source and target domains according to their geometrical properties. This criterion assumes that the relationship of geometric structures can reduce the domain shift.*

Domain adaptation for object recognition: An unsupervised approach[13] and *Geodesic flow kernel for unsupervised domain adaptation*[1] map features into an intermediate subspace to find the correlation between domains. They then interpolate between these spaces, starting with the source domain and moving towards the target domain.

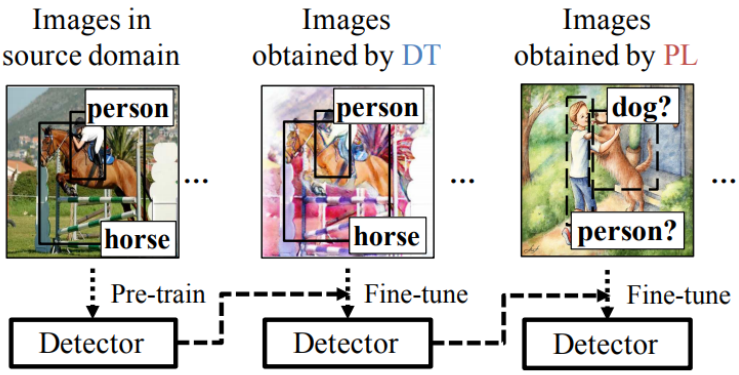

Figure 1.6: A three-step training approach to go from source to target images. DT = Domain Transfer GAN; PL = Pseudo-Labels (from detector)

1.2 Adversarial-Based DA

1.2.1 Generative Models

> *The typical case is to use source images, noise vectors or both to generate simulated samples that are similar to the target samples and preserve the annotation information of the source domain*

Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation[8] uses two techniques to generate target domain examples. Perform image-to-image translation from the source to target domain using CycleGAN = accurate bounding boxes, inaccurate visual features. Perform object detection to build "pseudo-labels" = inaccurate bounding boxes, accurate visual features.

1.2.2 Non-Generative Models

> *The feature extractor learns a discriminative representation using the labels in the source domain and maps the target data to the same space through a domain-confusion loss.*

A fairly typical example is *Unsupervised Domain Adaptation by Backpropagation*[5], where the model is trained to predict the domain from which an image is sampled. They reverse gradients at the boundary, for apply adversarial training.

Domain Adaptive Faster R-CNN for Object Detection in the Wild[4] applies it to Faster R-CNN by adding a domain-classifier to the region-proposals, in addition to the **Image-Level Domain Classifier**,

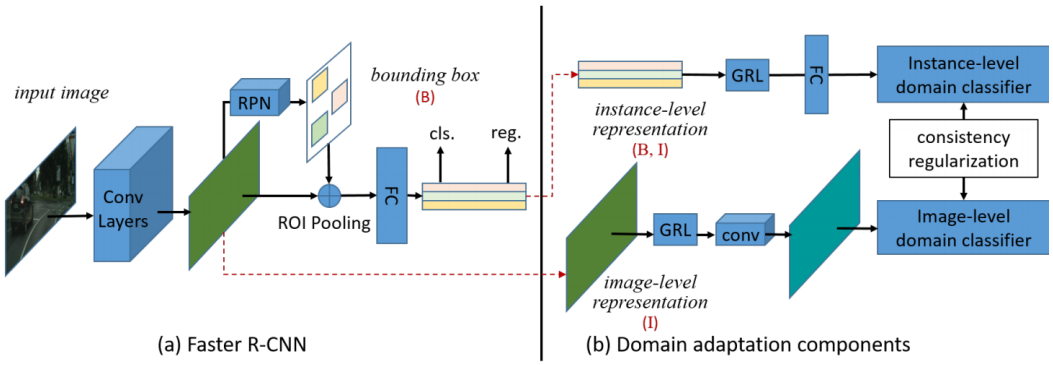

Figure 1.7: (a): Faster R-CNN; (b, upper): a Faster R-CNN-specific DA solution; (b, lower): conv-net used to predict image domain, with gradient-reversal-layer (GRL) at the boundary

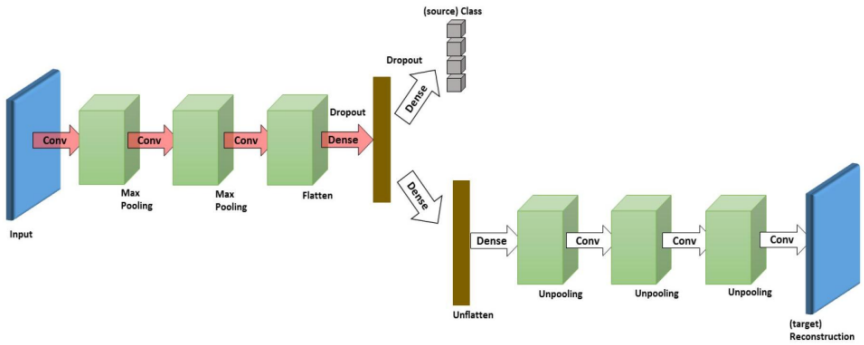

Figure 1.8: The deep reconstruction classification network (DRCN) adds the task of reconstruction during training

Instead of reversing gradients, *Simultaneous Deep Transfer Across Domains and Tasks*[18] sets the targets for the domain classifier to be a uniform distribution over all domains – thereby enforcing confusion.

1.3 Reconstruction-Based DA

Assumes that the data reconstruction of the source or target samples can be helpful for improving the performance of DA

1.3.1 Encoder-Decoder Reconstruction

By using stacked autoencoders (SAEs), encoder-decoder reconstruction methods combine the encoder network for representation learning with a decoder network for data reconstruction

Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach[7] uses Stacked Denoising Auto-encoders (SDA) to learn a less domain-dependant representation – an SDA is a stacked auto-encoder with stochastically-introduced noise added to the inputs. This gives good results, but doesn't scale well to high-dimensional data.

Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation[6] jointly learns the original (classification) task, and reconstruction of the inputs (figure 1.8).

Domain Separation Networks[2] explicitly use source, target and shared encoders (figure 1.9). They have a bunch of losses, including feature similarity, feature difference, reconstruction loss and the classification loss. Requires that same-class different-domain images are passed in pairs during training.

1.3.2 Adversarial Reconstruction

The reconstruction error is measured as the difference between the reconstructed and original images within each image domain by a cyclic mapping obtained via a GAN discriminator

The typical approach is "dual learning", where a network learns to map $X \rightarrow Y$ and $Y \rightarrow X$ simultaneously, using reconstruction loss for each side [21][23].

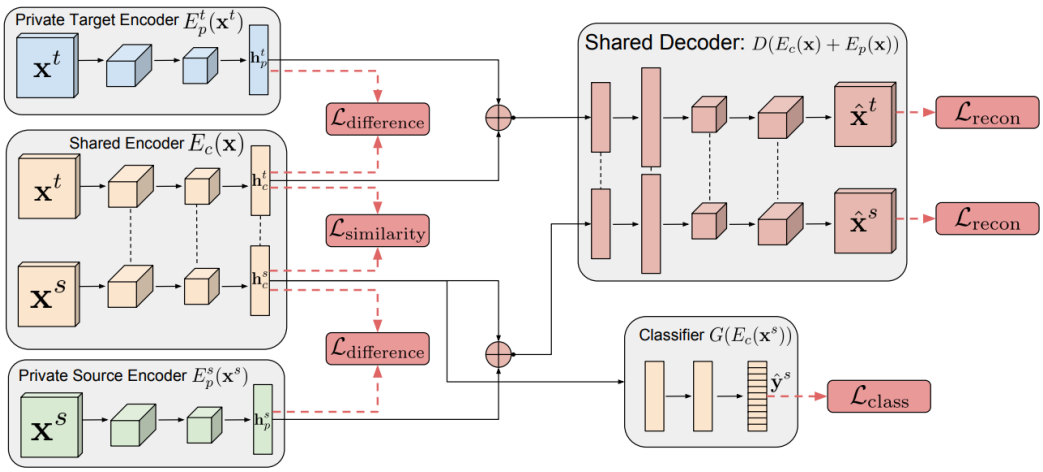

Figure 1.9: Domain Separation Networks architecture

Multi-Step Domain Adaptation

Multi-step domain adaptation can be categorised into three approaches as shown in table II.

TABLE II
DIFFERENT DEEP APPROACHES TO MULTI-STEP DA

Multi-step Approaches	Brief Description
Hand-crafted	users determine the intermediate domains based on experience [129]
Instance-based	selecting certain parts of data from the auxiliary datasets to compose the intermediate domains [114], [16]
Representation-based	freeze weights of one network and use their intermediate representations as input to the new network [96]

2.3.3 Hand-Crafted

Users determine the intermediate domains based on experience.

The intermediate domain can be selected by experience, that is, it is decided in advance. For example, when the source domain is image data and the target domain is composed of text data, some annotated images will clearly be crawled as intermediate domain data.

2.3.4 Instance-Based

Selecting certain parts of data from the auxiliary datasets to compose the intermediate domains to train the deep network

2.3.5 Representation-Based

Transfer is enabled via freezing the previously trained network and using their intermediate representations as input to the new one.

Progressive Neural Networks[15] extends the network for each domain, freezing existing weights when training on the new domain (figure 2.10).

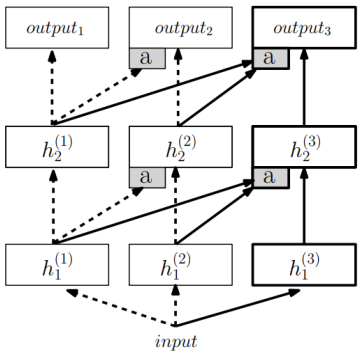

Figure 2.10:

Application to Swimming Project

While domain adaptation is something we wish to improve across the board, it is our crop-grid unit that is most in need of it. Unfortunately, that's also the most difficult for two main reasons:

1. High-dimensionality - *large cost for any additional operations/layers*
2. Spatially-dependant features - *location matters (unlike classification) making it harder to deal with feature distributions*

Maximum Mean Discrepancy (MMD) or Deep CORAL

- These two options make distributions of features similar at either a particular layer, or by some (learned) weighted amount at each layer.
- As our features have spatial dimensions, we would need to find a way to align our regions between source and target.
- This will likely not be possible, as we need to train simultaneously on source and target domains

Adaptive Batch Normalisation / Instance Normalisation / *Group Normalisation*

- Should be fairly easy to implement
- Likely no overhead, as we currently use batch normalisation.
- Although statistics do change between venues/videos, a single frame is a good representation of the entire video

Adversarial Domain classifier

- Relatively easy to implement in the regression unit
- May not be possible in crop-grid unit due to high-dimensionality

Reconstruction loss

- Relatively easy to implement in the regression unit
- May not be possible in crop-grid unit due to high-dimensionality