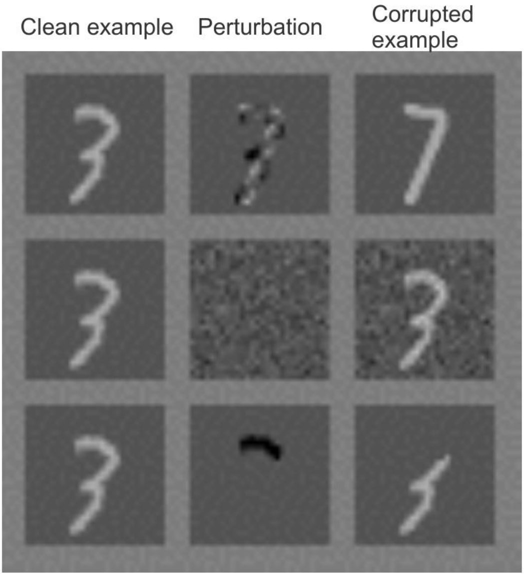
Deep Learning

Alexandre Allauzen, Michèle Sebag
CNRS & Université Paris-Sud

Oct. 17th, 2018

Credit for slides: Sanjeev Arora; Yoshua Bengio;Yann LeCun; Nando de Freitas; Pascal Germain; Léon Gatys; Weidi Xie; Max Welling; Victor Berger; Kevin Frans; Lars Mescheder et al.; Mehdi Sajjadi et al.; Pascal Germain et al.; Ian Goodfellow; Arthur Pesah

What happens when perturbing an example ?



Anything !

Informed perturbations

Goodfellow et al. 15

For x' perturbed from x

$$F(x',\theta) \approx F(x,\theta) + \langle x - x', \nabla_x F(x,\theta)\rangle$$

Nasty small perturbations ?

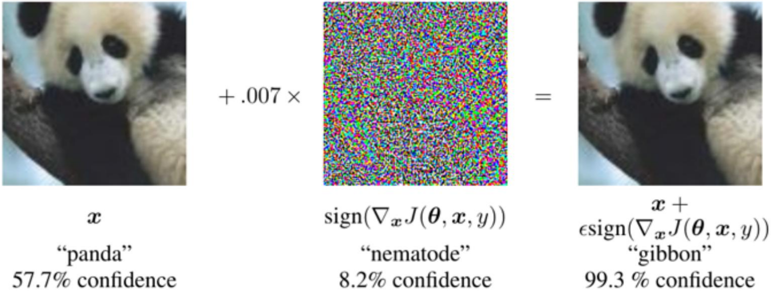
$$\text{Maximize } \langle x - x', \nabla_x F(x,\theta)\rangle$$

subject to

$$\|x - x'\|_\infty \leq \epsilon$$

Example

Goodfellow et al. 15



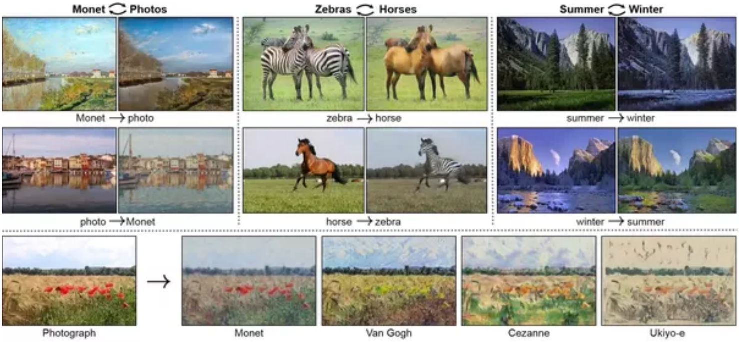
Example 2

Karpathy et al. 15

The lesson of adversarial examples

- Good performances do not imply that the NN got it !
- Small modifications are enough to make it change its diagnosis
- Terrible implications for autonomous vehicles !
- An arms race: modify the learning criterion; find adversarial examples defeating the modified criterion; iterate
- More in the Course/Seminar !

What is domain adaptation ?



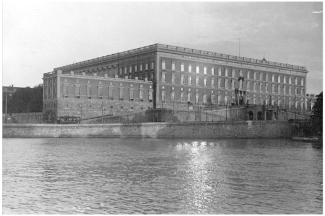
some differences should make **no** difference

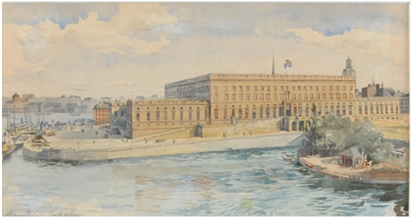
Domain adaptation:

- Learning from poor data by leveraging other (not really, not much different) data
- Teaching the learner to overcome these differences

Have you been to Stockholm recently ?

... you recognize the castle ...





regardless of light, style, angle...

Formally

Domain Adaptation

- Task: classification, or regression
- A source domain — source distribution $\mathcal{D}_s$
- A target domain — target distribution $\mathcal{D}_t$

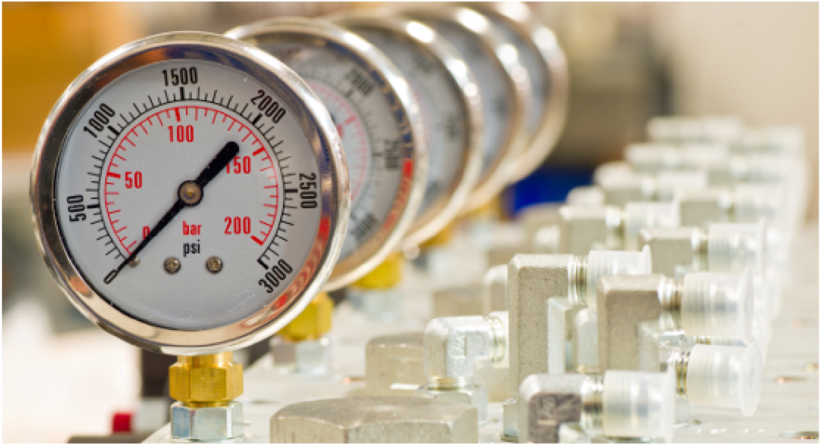
Idea

- Source and target are "sufficiently" related
- ... one wants to use source data to improve learning from target data

Applications

1. Calibration
2. Physiological signals
3. Reality gap (simulation vs real-world)
4. Lab essays
5. Similar worlds

Application 1. Calibration

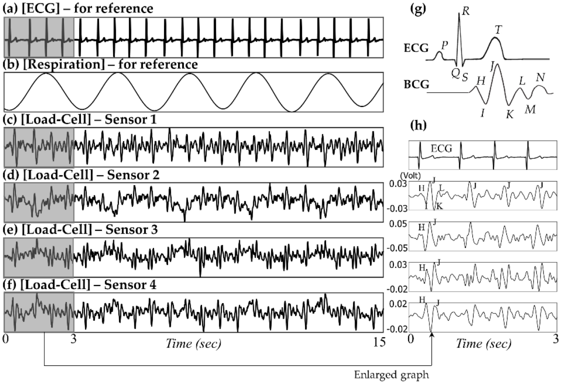


Different devices

- same specifications (in principle)
- in practice response function is biased
- Goal: recover the output complying with the specifications.

Application 2. Physiological signals

Won Kyu Lee et al. 2016



Different signals

- Acquired from different sensors (different price, SNR),
- Goal: predict from poor signal

Application 3. Bridging the reality gap

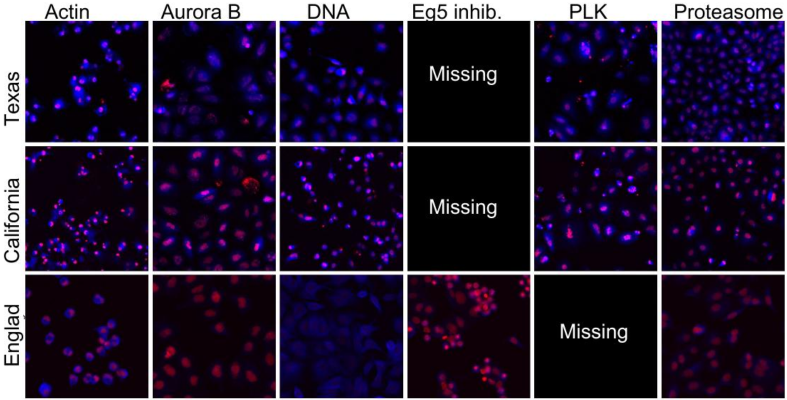
Source world aimed to model target world

- Target (expensive): real-world
- Source (cheap, approximate): simulator
- Goal: getting best of both worlds

In robotics; for autonomous vehicles; for science (e.g. Higgs boson ML challenge); ...

Application 4. Learning across labs

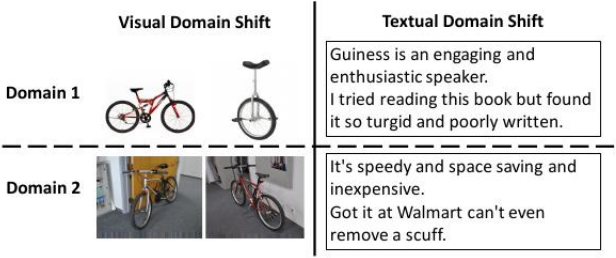
Schoenauer et al. 18



Many labs, many experiments in quantitative microscopy

- Each dataset: known and unknown perturbations; experimental bias
- Goal: Identify drugs in datasets: *in silico* discovery.

Application 5. Bridges between worlds

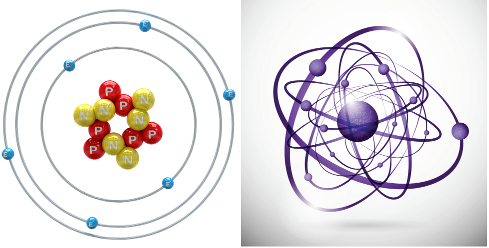


Different domains

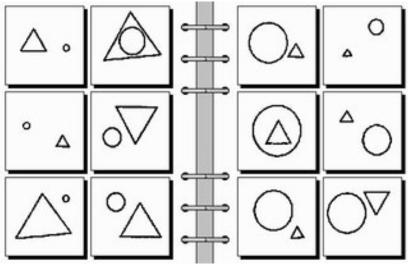
- Supposedly related
- One (source) is well-known;
- The other (target) less so: few or no labels
- Goal: Learn faster/better on the target domain

At the root of domain adaptation; Analogical reasoning

Hofstadter 1979: Analogy is at the core of cognition



Solar system $\leftrightarrow$ Atom and electrons



Bongard IQ tests

Roots of domain adaptation, 2

Training on male mice; testing on male and female mice ?

Relaxing the iid assumption:

when training and test distributions differ

- Class ratios are different — Kubat et al. 97; Lin et al, 02; Chan and Ng 05
- Marginals are different: Covariate shift

  Shimodaira 00; Zadrozny 04; Sugiyama et al. 05; Blickel et al. 07

Adversarial examples

Domain Adaptation: Formal background

Introduction
- Position of the problem
- Applications
- Settings

Key concept: distance between source and target distributions

Some Domain Adaptation Algorithms
- Domain Adversarial Neural Network
- Evaluating DA algorithms
- DANN improvements and relaxations

Settings: Domain adaptation wrt Transfert learning

Notations

	Joint dis.	Marginal Instance dis.	Conditional dis.
Source	$\mathcal{D}_s$	$P_s(X)$	$P_s(Y\|X)$
Target	$\mathcal{D}_t$	$P_t(X)$	$P_t(Y\|X)$

The settings

- Same instance distributions $P_s(X) = P_t(X)$
  - Same conditional distributions $P_s(Y|X) == P_t(Y|X)$ **Usual setting**
  - Different conditional distributions $P_s(Y|X) \neq P_t(Y|X)$ **Concept drift** **Inductive transfert learning**
- Different instance distributions $P_s(X) \neq P_t(X)$
  - Same conditional distributions $P_s(Y|X) == P_t(Y|X)$ **Domain adaptation** **Transductive transfert learning**
  - Different conditional distributions $P_s(Y|X) \neq P_t(Y|X)$ **Concept drift** **Unsupervised transfert learning**

NB: For some authors, all settings but the usual one are Transfer learning.
NB: Multi-task, $dom(Y_s) \neq dom(Y_t)$
NB: A continuum from Domain Adaptation to Transfer Learning to Multi-task learning

Examples of concept drift

- Which speed reached depending on the actuator value ?
  decreases as the motor is aging

- The concept of “chic” ?
  depends on the century nice, cool, ...

Related: Lifelong learning

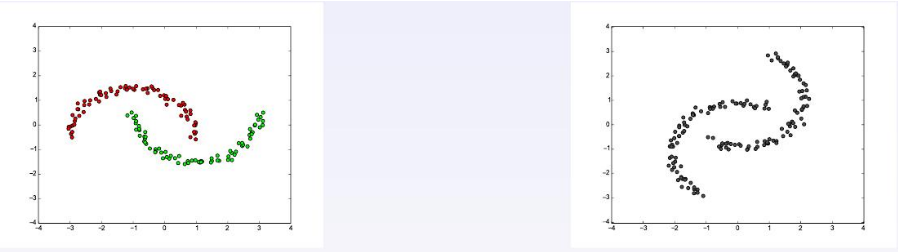
- player increases its abilities through time
- poker hands were generated in order
- instance is a market state in 30 minutes
- synthetic data with three drift points of abrupt concept change

Dataset	instances	attributes	Reference
Chess	503	8	(Žliobaite, 2010)
Poker	100,000	10	(Olorunnimbe et al., 2015)
Electricity	45,312	8	(Baena-García et al., 2006)
Stagger	70,000	3	(Gama et al., 2014)

AutoML2 challenge data sets

Shameless ad for AutoML3: AutoML for Lifelong ML-2018

Toy example of domain adaptation: the intertwining moons



Settings, 2

General assumptions

- Wealth of information about source domain
- Scarce information about target domain

Domain Adaptation aims at alleviating the costs

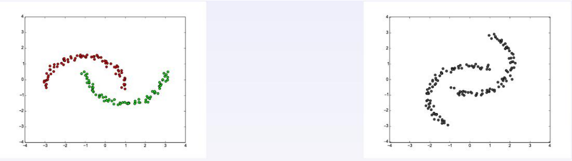
- of labelling target examples
- of acquiring target examples

No target labels	**Unsupervised Domain Adaptation**
Partial labels	**Partially unsupervised Domain Adaptation**
Few samples	**Few-shot Domain Adaptation**

Key Concept: Distance between source and target marginal distributions

1. The larger, the more difficult the domain adaptation
2. Can we measure it ? for theory

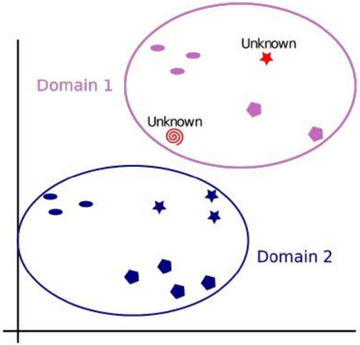
   if so, turn the measure into a loss, to be minimized
3. Can we reduce it ? for algorithms



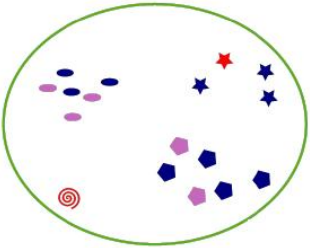
The 2 moons problem

Domain adaptation, intuition

What we have



What we want



Distance between source and target marginal distributions, followed

Main strategies

- Reduce it in original space $\mathcal{X}$

  Importance sampling

- Modify source representation

  Optimal transport

- Map source and target onto a third **latent** space

  Domain adversarial

- Build generative mechanisms in latent space

  Generative approaches

Milestone: defining distances on distributions

Discrepancy between source and target marginal distributions

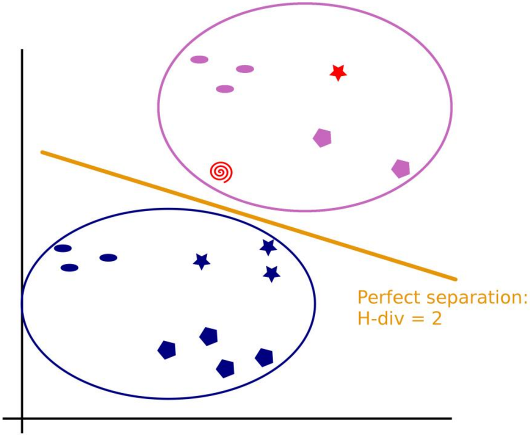
Ben-David 06, 10

$\mathcal{H}$ Divergence between P_s and P_t

$$d_X(P_s, P_t) = 2 \sup_{h \in \mathcal{H}} \left| Pr_{x \sim P_s}(h(x) = 1) - Pr_{x \sim P_t}(h(x) = 1) \right|$$

This divergence is high if there exists h separating P_s and P_t.

Perfect separation case

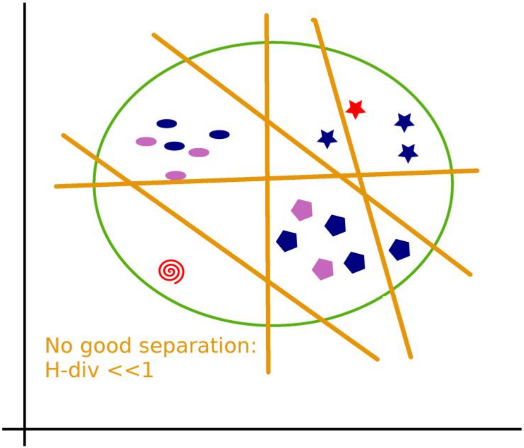


Discrepancy between source and target marginal distributions, 2

Ben-David 06, 10

$$d_X(P_s, P_t) = 2 \sup_{h \in \mathcal{H}} |Pr_{x \sim P_s}(h(x) = 1) - Pr_{x \sim P_t}(h(x) = 1)|$$

Perfect mixt case



$\Rightarrow$ what is learned on source

Discrepancy between source and target marginal distributions, 3

Ben-David et al. 2006, 2010

Approximation of $\mathcal{H}$ divergence

Proxy A-distance (PAD)

$$\widehat{d_X(P_s, P_t)} = 2\left(1 - \min_h \left(\frac{1}{n}\sum_i 1_{h(x_i)=0} + \frac{1}{n'}\sum_j 1_{h(x'_j)=1}\right)\right)$$

The divergence can be approximated by the ability to empirically discriminate between source and target examples.

Comment

Estimation of distribution differences $\rightarrow$ two-sample tests.

Bounding the domain adaptation risk

Ben-David et al. 2006, 2010

Notations

- $R_s(h) = \mathbb{E}_{\mathcal{D}_s}\mathcal{L}(h)$ risk of h under source distribution
- $R_t(h) = \mathbb{E}_{\mathcal{D}_t}\mathcal{L}(h)$ risk of h under target distribution

Theorem

With probability $1-\delta$, if $d(\mathcal{H})$ is the VC-dimension of $\mathcal{H}$,

$$R_t(h) \leq \widehat{R_s(h)} + \widehat{d_X} + C\sqrt{\frac{4}{n}(d(\mathcal{H})log\frac{2}{d} + log\frac{4}{\delta})} + \text{Best possible}$$

and

$$\text{Best possible} = \inf_h \left(R_S(h) + R_T(h)\right)$$

What we want (risk on h wrt $\mathcal{D}_T$) is bounded by:

- empirical risk on source domain
- + Proxy A-distance
- + error related to possible overfitting
- + min error one can achieve on both source and target distribution.

Interpretation

Ben-David et al. 2006, 2010

The regret

With probability $1-\delta$, if $d(\mathcal{H})$ is the VC-dimension of $\mathcal{H}$,

$$R_t(h) - \text{Best possible} \leq \widehat{R_s(h)} + C\sqrt{\frac{4}{n}(d(\mathcal{H})log\frac{2}{d(\mathcal{H})} + log\frac{4}{\delta})} + \widehat{d_X}$$

Hence a domain adaptation strategy:

- Choose $\mathcal{H}$ with good potential
- Minimize $\widehat{d_X}$: through transporting source data; or mapping source and target toward another favorable space.

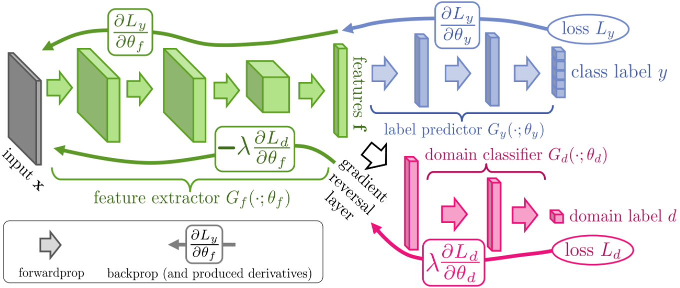
Extending Adversarial Ideas to Domain Adaptation

Input

$$\mathcal{E}_s = \{(x_{s,i}, y_i), i = [[1, n]]\}$$
$$\mathcal{E}_t = \{(x_{t,j}), j = [[1, m]]\}$$

Principle

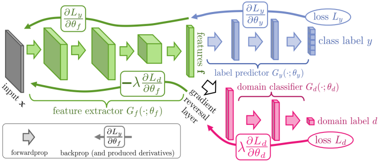
- What matters is the distance between $\mathcal{D}_s$ and $\mathcal{D}_t$ Ben David et al. 2010
- Strategy: mapping both on a same latent space in an indistinguishable manner



Ganin et al., 2015; 2016

Domain Adversarial Neural Net

Ganin et al. 2015; 2016



Adversarial Modules

- Encoder G_f — green
  $x_s \mapsto G_f(x_s); x_t \mapsto G_f(x_t)$
- Discriminator G_d: trained from $\{(G_f(x_{s,i}), 1)\} \cup \{(G_f(x_{t,j}), 0)\}$ — red

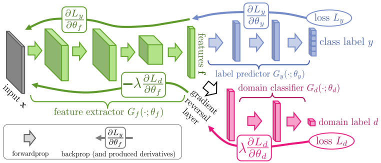
$$\text{Find} \quad \max_{G_f} \min_{G_d} \mathcal{L}(G_d, G_f)$$

And a Classifier Module

- G_y: $\mathcal{L}(G_y) = \sum_i \ell(G_y(G_f(x_{s,i})), y_i)$ — blue
- NB: needed to prevent trivial solution $G_f \equiv 0$

DANN, 2

Ganin et al. 2015; 2016



Training

1. Classifier: backprop from $\nabla(\mathcal{L}(G_y))$ blue
2. Encoder: backprop from $\nabla(\mathcal{L}(G_y))$ and $-\nabla(\mathcal{L}(G_d))$ green
3. Discriminator: backprop from $\nabla(\mathcal{L}(G_d))$ red

The algorithm

Algorithm 1 Shallow DANN – Stochastic training update

Input:
— samples $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ and $T = \{\mathbf{x}_i\}_{i=1}^{n'}$,
— hidden layer size D,
— adaptation parameter λ,
— learning rate μ,

Output: neural network $\{\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}\}$

$\mathbf{W}, \mathbf{V} \leftarrow \text{random\_init}(\,D\,)$
$\mathbf{b}, \mathbf{c}, \mathbf{u}, d \leftarrow 0$
while stopping criterion is not met **do**
  for i from 1 to n **do**
    `# Forward propagation`
    $G_f(\mathbf{x}_i) \leftarrow \text{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}_i)$
    $G_y(G_f(\mathbf{x}_i)) \leftarrow \text{softmax}(\mathbf{c} + \mathbf{V}G_f(\mathbf{x}_i))$

    `# Backpropagation`
    $\Delta_{\mathbf{c}} \leftarrow -(\mathbf{e}(y_i) - G_y(G_f(\mathbf{x}_i)))$
    $\Delta_{\mathbf{V}} \leftarrow \Delta_{\mathbf{c}}\, G_f(\mathbf{x}_i)^\top$
    $\Delta_{\mathbf{b}} \leftarrow \left(\mathbf{V}^\top \Delta_{\mathbf{c}}\right) \odot G_f(\mathbf{x}_i) \odot (1 - G_f(\mathbf{x}_i))$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{b}} \cdot (\mathbf{x}_i)^\top$

    `# Domain adaptation regularizer...`
    `# ...from current domain`
    $G_d(G_f(\mathbf{x}_i)) \leftarrow \text{sigm}(d + \mathbf{u}^\top G_f(\mathbf{x}_i))$
    $\Delta_d \leftarrow \lambda(1 - G_d(G_f(\mathbf{x}_i)))$
    $\Delta_{\mathbf{u}} \leftarrow \lambda(1 - G_d(G_f(\mathbf{x}_i)))G_f(\mathbf{x}_i)$
    $\text{tmp} \leftarrow \lambda(1 - G_d(G_f(\mathbf{x}_i))) \times \mathbf{u} \odot G_f(\mathbf{x}_i) \odot (1 - G_f(\mathbf{x}_i))$
    $\Delta_{\mathbf{b}} \leftarrow \Delta_{\mathbf{b}} + \text{tmp}$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{W}} + \text{tmp} \cdot (\mathbf{x}_i)^\top$

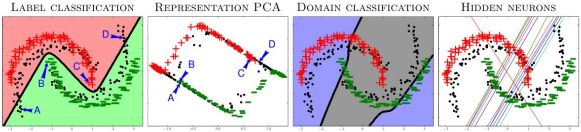
    `# ...from other domain`
    $j \leftarrow \text{uniform\_integer}(1, \ldots, n')$
    $G_f(\mathbf{x}_j) \leftarrow \text{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}_j)$
    $G_d(G_f(\mathbf{x}_j)) \leftarrow \text{sigm}(d + \mathbf{u}^\top G_f(\mathbf{x}_j))$
    $\Delta_d \leftarrow \Delta_d - \lambda G_d(G_f(\mathbf{x}_j))$
    $\Delta_{\mathbf{u}} \leftarrow \Delta_{\mathbf{u}} - \lambda G_d(G_f(\mathbf{x}_j))G_f(\mathbf{x}_j)$
    $\text{tmp} \leftarrow -\lambda G_d(G_f(\mathbf{x}_j)) \times \mathbf{u} \odot G_f(\mathbf{x}_j) \odot (1 - G_f(\mathbf{x}_j))$
    $\Delta_{\mathbf{b}} \leftarrow \Delta_{\mathbf{b}} + \text{tmp}$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{W}} + \text{tmp} \cdot (\mathbf{x}_j)^\top$

    `# Update neural network parameters`
    $\mathbf{W} \leftarrow \mathbf{W} - \mu\Delta_{\mathbf{W}}$
    $\mathbf{V} \leftarrow \mathbf{V} - \mu\Delta_{\mathbf{V}}$
    $\mathbf{b} \leftarrow \mathbf{b} - \mu\Delta_{\mathbf{b}}$
    $\mathbf{c} \leftarrow \mathbf{c} - \mu\Delta_{\mathbf{c}}$

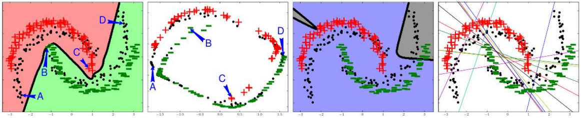
    `# Update domain classifier`
    $\mathbf{u} \leftarrow \mathbf{u} + \mu\Delta_{\mathbf{u}}$
    $d \leftarrow d + \mu\Delta_d$
  end for
end while

Note: In this pseudo-code, $\mathbf{e}(y)$ refers to a "one-hot" vector, consisting of all 0s except for a 1 at position y, and $\odot$ is the element-wise product.

The intertwinning moons



(a) Standard NN. For the "domain classification", we use a *non adversarial* domain regressor on the hidden neurons learned by the Standard NN. (This is equivalent to run Algorithm 1, without Lines 22 and 31)

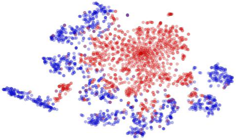


(b) DANN (Algorithm 1)

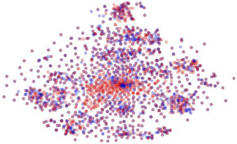
- left: the decision boundary
- 2nd left: apply PCA on the feature layer
- 3rd left: discrimination source vs target
- right: each line corresponds to hidden neuron = .5

Mixing the distributions in latent space

MNIST $\rightarrow$ MNIST-M

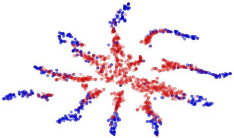


(a) Non-adapted

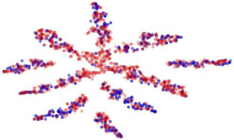


(b) Adapted

Syn $\rightarrow$ SVHN



(a) Non-adapted



(b) Adapted

Evaluation

Top: SVHN; Bottom: MNIST

Usual practice

- The reference experiment: adapting from Street View House Numbers (SVHN, source) to MNIST (handwritten digits)
- Score: accuracy on the test set of MNIST.
- **Caveat**: reported improvements might come from:
  1. algorithm novelty;
  2. neural architecture;
  3. hyperparameter tuning ?
- Lesion studies are required !

Experimental setting

Ganin et al., 16

The datasets

- MNIST: as usual
- MNIST-M: blend with patches randomly extracted from color photos from BSDS500
- SVHN: Street-View House Number dataset
- Syn Numbers: figures from WindowsTM fonts, varying positioning, orientation, background and stroke colors, blur.
- Street Signs: real (430) and synthetic (100,000)

Results

Ganin et al., 16

Method (Source / Target)	MNIST / MNIST-M	Syn Numbers / SVHN	SVHN / MNIST	Syn Signs / GTSRB
Source only	.5225	.8674	.5490	.7900
SA (Fernando et al., 2013)	.5690 (4.1%)	.8644 (−5.5%)	.5932 (9.9%)	.8165 (12.7%)
DANN	**.7666** (52.9%)	**.9109** (79.7%)	**.7385** (42.6%)	**.8865** (46.4%)
Train on target	.9596	.9220	.9942	.9980

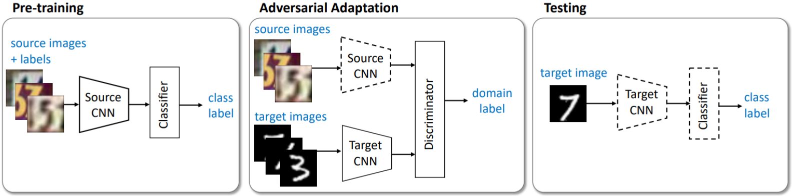
Score DANN: 74%

Decoupling the encoder: ADDA

Tzeng et al., 2017

Adversarial Discriminative Domain Adaptation (ADDA)

- DANN used a single encoder G_f for both source and target domains
- ADDA learns $G_{f,s}$ and $G_{f,t}$ independently, both subject to G_d (domain discriminator); and $G_{f,s}$ subject to G_y
- Rationale: makes it easier to handle source and target with different dimensionality, specificities,...



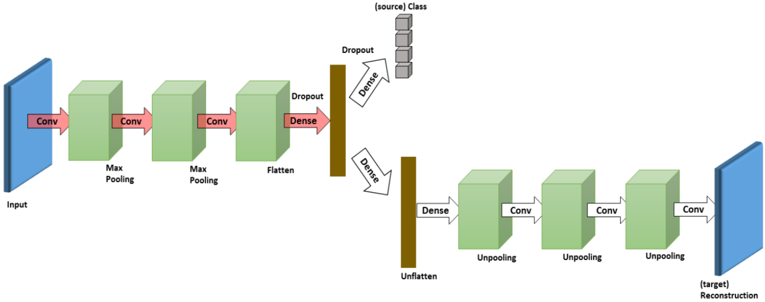
Score DANN: 74%
Score ADDA: 76%

Replacing domain discrimination with reconstruction: DRCN

Ghifary et al., 2016

Deep Reconstruction-Classification Networks (DRCN)

- DANN used a discriminator G_d to discriminate $G_f(x_t)$ and $G_f(x_s)$
- DRCN replaces G_d with a decoder s.t. $G_d(G_f(x_t)) \approx x_t$
- Rationale: The latent space preserves all information from target, while enabling classification on source.



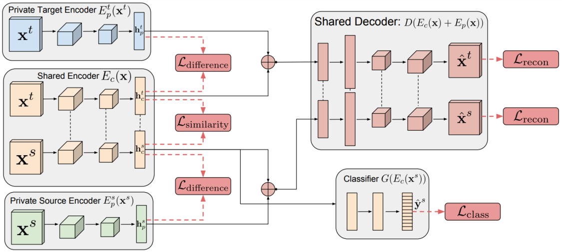
Score DANN: 74%
Score ADDA: 76%
Score DRCN: 82%

Hybridizing ADDA and DRCN: Deep Separation Networks

Bousmalis et al., 2016

Deep Separation Networks (DSN)

- Encoder:
  - A shared part $G_{f,u}$
  - A private source part $G_{f,s}$
  - A private domain part $G_{f,t}$
- Discriminator $\rightarrow$ Decoder
  - $G_d(G_{f,u}(x_s), G_{f,s}(x_s)) \approx x_s$
  - $G_d(G_{f,u}(x_t), G_{f,t}(x_t)) \approx x_t$



(. . . stands for "shared weights")

Score DANN: 74%
Score ADDA: 76%
Score DRCN: 82%
Score DSN: 82.7%

Not covered...

- Optimal transport — Couturi Peyre 18, Courty et al. 17,18
- Generative Networks and domain to domain translations
  Taigman et al. 16; Sankaranarayanan et al. 17; Liu et al. 17
  Choi et al. 17; Anoosheh et al., 2017; Shu et al. 18
- Partial domain adaptation — Motiian et al. 17a, b; Schoenauer-Sebag 18

Conclusions

Theory and Validation

- Most theoretical analysis relies on Ben David et al. 06; 10
- When using feature space, something is underlooked (see DRCN).
- Comprehensive ablation studies needed to assess the mixture of losses/architectures
- Assessing the assumptions

Applications

- Many applications on vision The Waouh effect ?
- Reinforcement learning !
- Natural Language processing !

Take home message

What is domain adaptation:

- Playing with tasks and distributions
- Making assumptions about how they are related
- Testing your assumptions

Domain adaptation is like playing Lego with ML