Domain-Adversarial Training of Neural Networks

Y. Ganin and E. Ustinova and H. Ajakan and P. Germain and H. Larochelle and F. Laviolette and M. Marchand and V. Lempitsky. 2016

Extends from

Unsupervised Domain Adaptation by Backpropagation

Y. Ganin and V. Lempitsky. 2015

Presentation by Nikos Efthymiadis
Visual Recognition Group
Czech Technical University in Prague

Motivation

- Early deep learning approach in domain adaptation
  - The first that used adversarial learning for domain adaptation
  - It created a branch in the practice

Motivation

- Early deep learning approach in domain adaptation
  - The first that used adversarial learning for domain adaptation
  - It created a branch in the practice
- As a research practice model
  - Inspiration from theory and not on only from pure intuition
  - Strong arguments on why it works

Motivation

- Early deep learning approach in domain adaptation
  - The first that used adversarial learning for domain adaptation
  - It created a branch in the practice
- As a research practice model
  - Inspiration from theory and not on only from pure intuition
  - Strong arguments on why it works
- Was the state of the art at that time

Setting

Unsupervised Single-Source Domain Adaptation

Source domain

MNIST

4 0 1 *Labels*

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Target domain

MNIST-M

No Labels

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Setting

Unsupervised Single-Source Domain Adaptation

Source domain

MNIST

4 0 1 *Labels*

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Target domain

MNIST-M

No Labels

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Setting

Unsupervised Single-Source Domain Adaptation

Source domain

MNIST

4 0 1

Labels

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Target domain

MNIST-M

No Labels

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Setting

Unsupervised **Single-Source Domain Adaptation**

Source domain

MNIST

4 *0* *1* *Labels*

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Target domain

MNIST-M

No Labels

Task: Assign {0, 1, 2, ..., 9}

$x \in [0,1]^{256 \times 256 \times 3}$

Domain Alignment

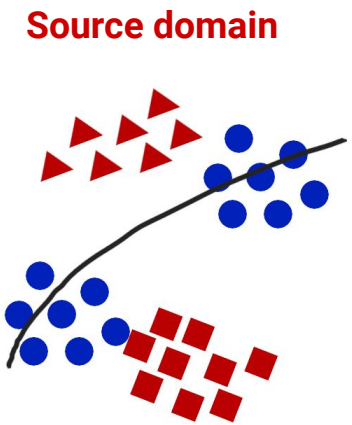


Figure 1. The domain alignment concept schematically. The color defines the domain and the shape defines the class. We don’t know the class of the target domain

Domain Alignment

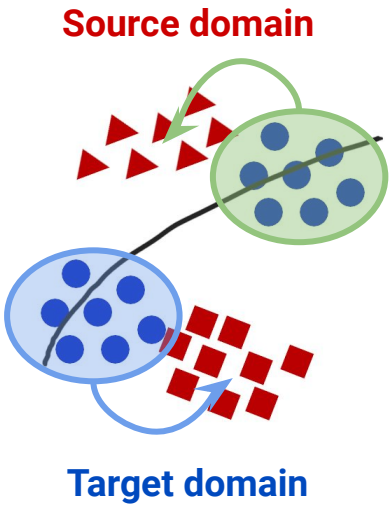


Figure 1. The domain alignment concept schematically. The color defines the domain and the shape defines the class. We don’t know the class of the target domain

Domain Alignment

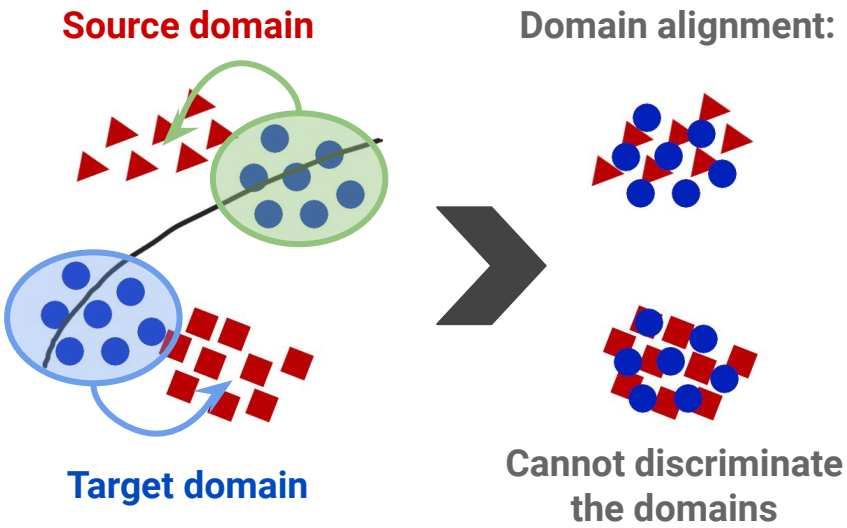


Figure 1. The domain alignment concept schematically. The color defines the domain and the shape defines the class. We don’t know the class of the target domain

Domain Alignment

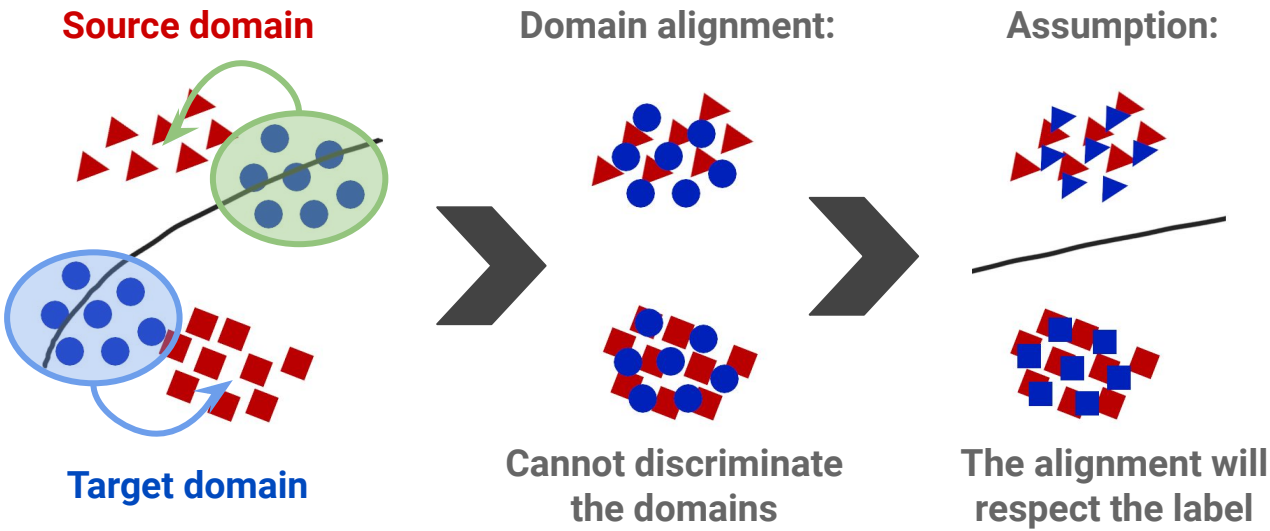


Figure 1. The domain alignment concept schematically. The color defines the domain and the shape defines the class. We don't know the class of the target domain

Setting

- Input space X

Setting

- Input space X
- Label space $Y = \{0, 1\}$

Setting

- Input space X
- Label space $Y = \{0, 1\}$
- Source domain D_S and target domain D_T are distributions over $X \times Y$

Setting

- Input space X
- Label space $Y = \{0, 1\}$
- Source domain D_S and target domain D_T are distributions over $X \times Y$
- Unsupervised setting: labeled source sample S from D_S and unlabeled target sample T from D_T^X

Setting

- Input space X
- Label space $Y = \{0, 1\}$
- Source domain D_S and target domain D_T are distributions over $X \times Y$
- Unsupervised setting: labeled source sample S from D_S and unlabeled target sample T from D_T^X
- D_T^X is the marginal distribution of D_T over X

Setting

- Input space X
- Label space $Y = \{0, 1\}$
- Source domain D_S and target domain D_T are distributions over $X \times Y$
- Unsupervised setting: labeled source sample S from D_S and unlabeled target sample T from D_T^X
- D_T^X is the marginal distribution of D_T over X
- Goal: Find a classifier $h : X \to Y, h \in H$ with small target risk $R_{D_T}(h) = Pr_{(x,y)\sim D_T}(h(x) \neq y)$

Distance Between Distributions

- H-divergence: $d_H(D_S^X, D_T^X) = 2 \sup_{h \in H} |Pr_{x \sim D_S^X}[h(x) = 1] - Pr_{x \sim D_T^X}[h(x) = 1]|$

  It searches for the hypothesis and the example with the biggest disagreement under the two distributions. It is small if we are unable to tell from which distribution every sample comes from.

Distance Between Distributions

- H-divergence: $d_H(D_S^X, D_T^X) = 2 \sup_{h \in H} |Pr_{x \sim D_S^X}[h(x) = 1] - Pr_{x \sim D_T^X}[h(x) = 1]|$

  It searches for the hypothesis and the example with the biggest disagreement under the two distributions. It is small if we are unable to tell from which distribution every sample comes from.

- Empirical H-divergence: $\hat{d}_H(S, T) = 2 \left(1 - \min_{h \in H} \left[\frac{1}{n} \sum_{i=1}^{n} I[h(x_i) = 0] + \frac{1}{n'} \sum_{i=n+1}^{N} I[h(x_i) = 1]\right]\right)$

  S is the source sample of size n, T is the target sample of size n', I is the indicator function. This holds for a symmetric hypothesis space H. Proof in *Ben-David et al. 2010*.

Distance Between Distributions

- H-divergence: $d_H(D_S^X, D_T^X) = 2 \sup_{h \in H} |Pr_{x \sim D_S^X}[h(x) = 1] - Pr_{x \sim D_T^X}[h(x) = 1]|$

  It searches for the hypothesis and the example with the biggest disagreement under the two distributions. It is small if we are unable to tell from which distribution every sample comes from.

- Empirical H-divergence: $\hat{d}_H(S,T) = 2\left(1 - \min_{h \in H}\left[\frac{1}{n}\sum_{i=1}^{n} I[h(x_i) = 0] + \frac{1}{n'}\sum_{i=n+1}^{N} I[h(x_i) = 1]\right]\right)$

  S is the source sample of size n, T is the target sample of size n', I is the indicator function. This holds for a symmetric hypothesis space H. Proof in *Ben-David et al. 2010*.

- Proxy distance: Construct a new dataset $U = \{(x_i, 0)\}_{i=1}^{n} \cup \{(x_i, 1)\}_{i=n+1}^{N}$, train a classifier h' that discriminates domains and it's risk ε is going to approximate min part. Then: $\hat{d}_H(S,T) = 2(1 - 2\varepsilon)$

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d.

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample,

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability $1-\delta$, for every hypothesis h:

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + f(m,m',d,\delta)$$

$$\lambda \geq \inf_{h^* \in H} [R_{D_S}(h^*) + R_{D_T}(h^*)]$$

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + f(m,m',d,\delta)$$

$$\lambda \geq \inf_{h^* \in H} [R_{D_S}(h^*) + R_{D_T}(h^*)]$$

Function of the data size, the uncertainty δ and the VC dimension d. Irrelevant to the training.

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + f(m, m', d, \delta)$$

$$\lambda \geq \inf_{h^* \in H} [R_{D_S}(h^*) + R_{D_T}(h^*)]$$

Function of the data size, the uncertainty δ and the VC dimension d. Irrelevant to the training.

λ is small if there is a hypothesis h that performs well on both domains. We assume there is for this task.

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + f(m, m', d, \delta)$$

$$\lambda \geq \inf_{h^* \in H} [R_{D_S}(h^*) + R_{D_T}(h^*)]$$

Function of the data size, the uncertainty δ and the VC dimension d. Irrelevant to the training.

λ is small if there is a hypothesis h that performs well on both domains. We assume there is for this task.

The ***H*-divergence** as described

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + f(m,m',d,\delta)$$

$$\lambda \geq \inf_{h^* \in H} [R_{D_S}(h^*) + R_{D_T}(h^*)]$$

Function of the data size, the uncertainty δ and the VC dimension d. Irrelevant to the training.

λ is small if there is a hypothesis h that performs well on both domains. We assume there is for this task.

The ***H*-divergence** as described

The **empirical source error** which is easily calculable.

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + f(m, m', d, \delta)$$

$$\lambda \geq \inf_{h^* \in H} [R_{D_S}(h^*) + R_{D_T}(h^*)]$$

Function of the data size, the uncertainty δ and the VC dimension d. Irrelevant to the training.

λ is small if there is a hypothesis h that performs well on both domains. We assume there is for this task.

The ***H*-divergence** as described

The **empirical source error** which is easily calculable.

Approach

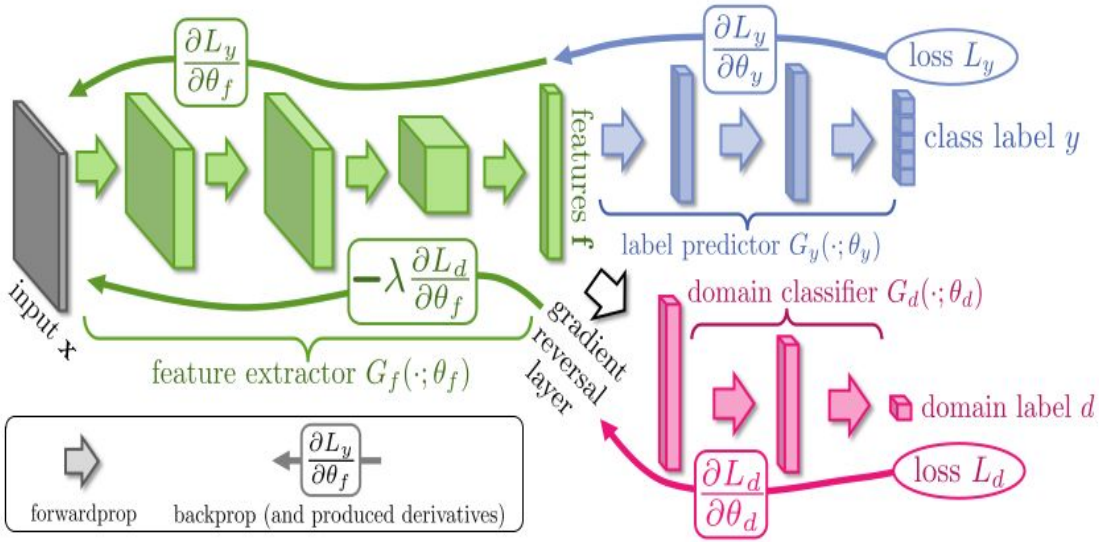


Figure 2. The proposed architecture. Image from *Ganin et al. 2016*.

Approach

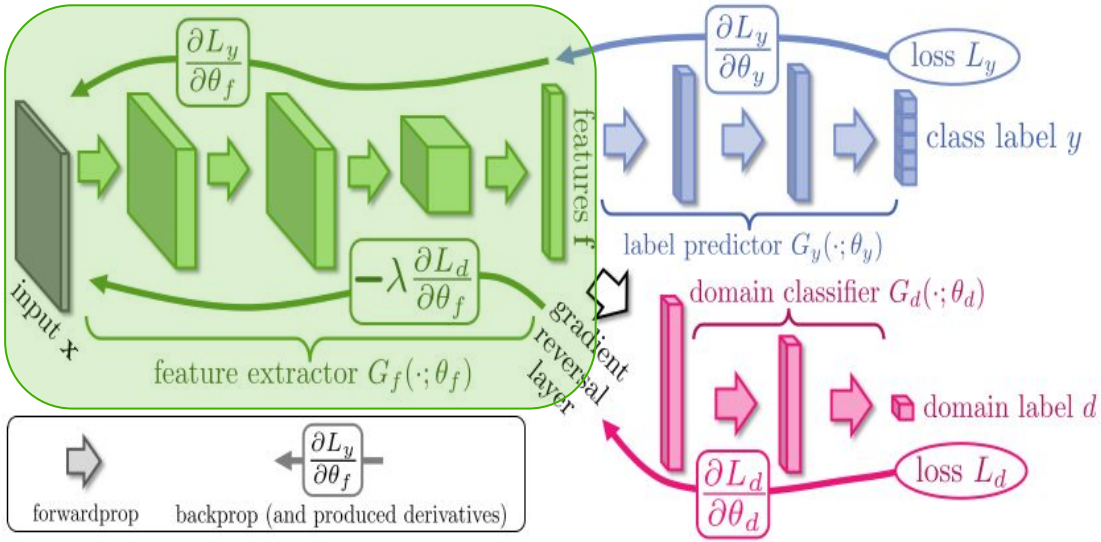


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

- The feature extractor learns a map of the input x to a new space through G_f

Approach

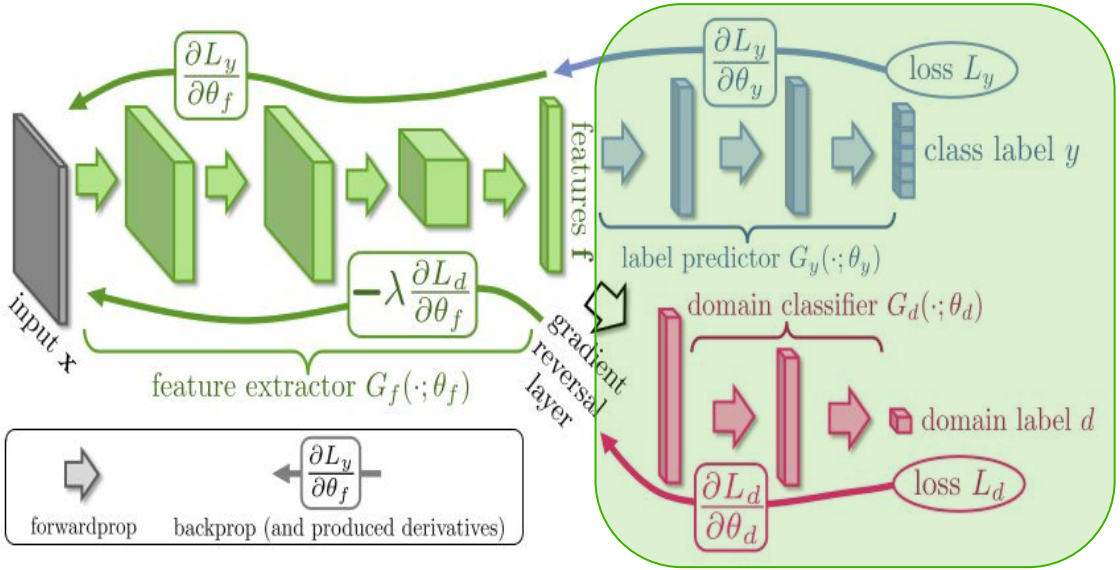
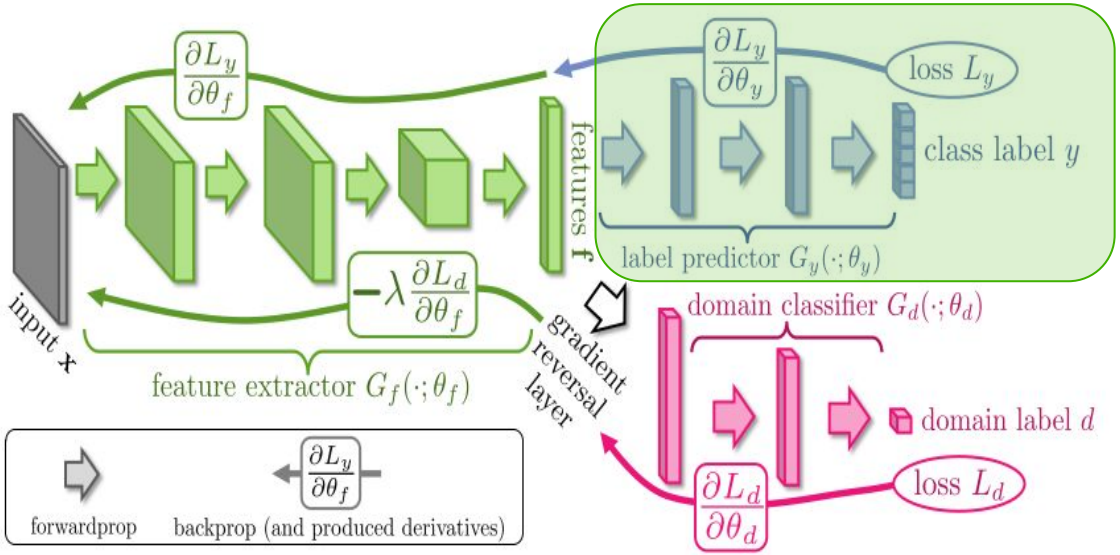


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

- The feature extractor learns a map of the input x to a new space through G_f
- The $G_f(x)$ are passed to a class classifier and a domain classifier

Approach



- The feature extractor learns a map of the input x to a new space through G_f
- The $G_f(x)$ are passed to a class classifier and a domain classifier
- The class classifier returns the gradient as usual

Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

Approach

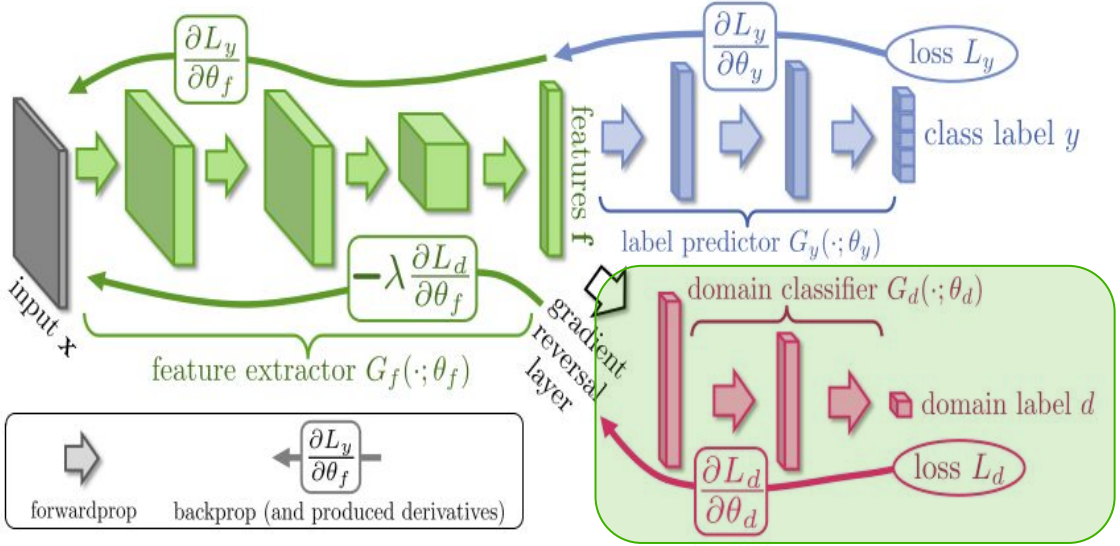


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

- The feature extractor learns a map of the input x to a new space through G_f
- The $G_f(x)$ are passed to a class classifier and a domain classifier
- The class classifier returns the gradient as usual
- The domain classifier uses a gradient reversal layer to return a gradient of the opposite direction

Approach

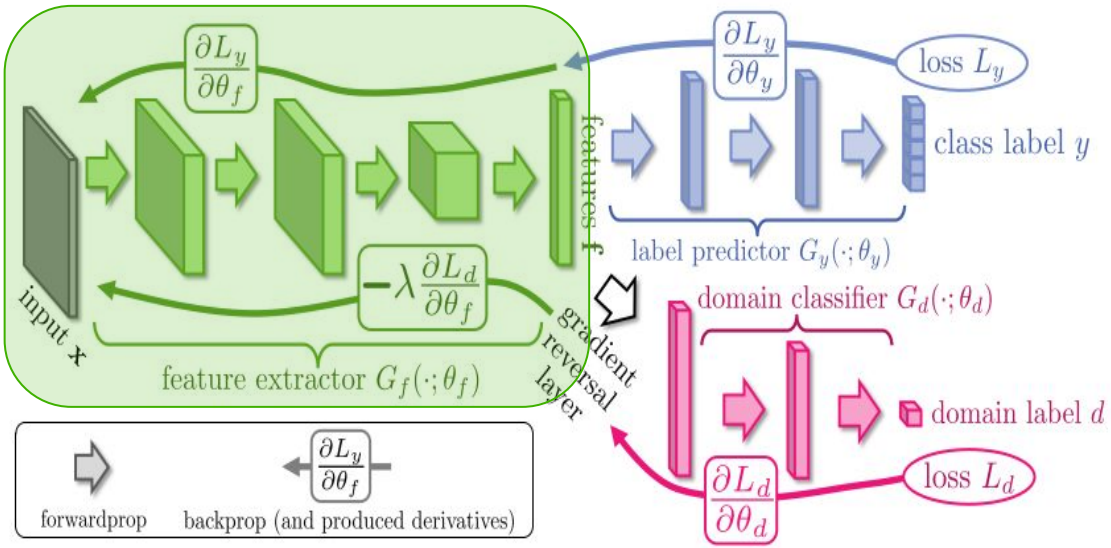


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

- The feature extractor learns a map of the input x to a new space through G_f
- The $G_f(x)$ are passed to a class classifier and a domain classifier
- The class classifier returns the gradient as usual
- The domain classifier uses a gradient reversal layer to return a gradient of the opposite direction
- This is making the feature extractor to map the input to a space where the domains are not discriminatable and therefore aligned

Approach

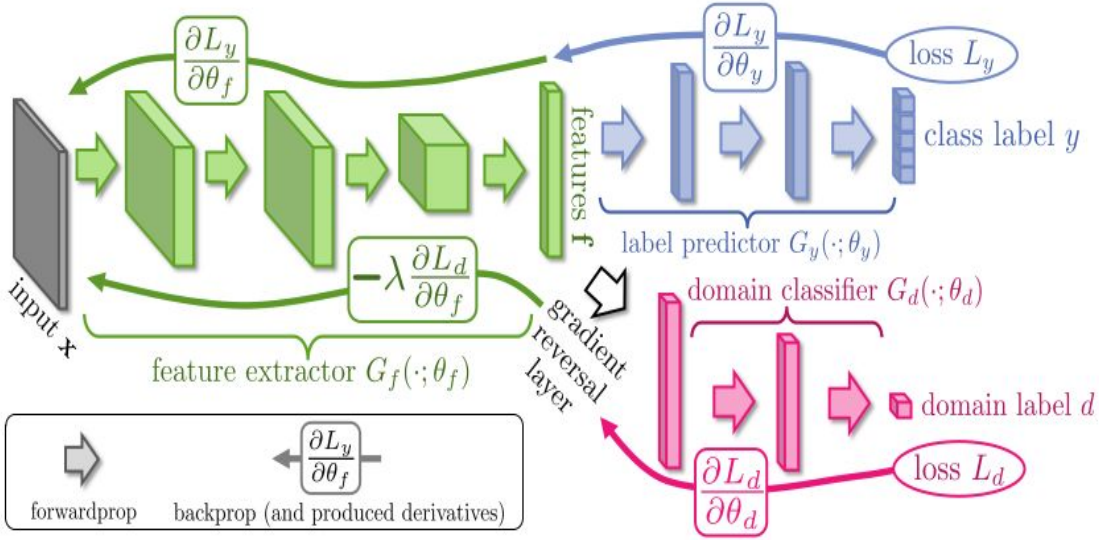


Figure 2. The proposed architecture. Image from *Ganin et al. 2016*.

$$\mathcal{L}_y^i(\theta_f,\theta_y) = \mathcal{L}_y\big(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y),y_i\big)$$
$$\mathcal{L}_d^i(\theta_f,\theta_d) = \mathcal{L}_d\big(G_d(G_f(\mathbf{x}_i;\theta_f);\theta_d),d_i\big)$$

$$E(\theta_f,\theta_y,\theta_d) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_y^i(\theta_f,\theta_y) - \lambda\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_d^i(\theta_f,\theta_d) + \frac{1}{n'}\sum_{i=n+1}^{N}\mathcal{L}_d^i(\theta_f,\theta_d)\right)$$

$$(\hat{\theta}_f,\hat{\theta}_y) = \operatorname*{argmin}_{\theta_f,\theta_y} E(\theta_f,\theta_y,\hat{\theta}_d)$$
$$\hat{\theta}_d = \operatorname*{argmax}_{\theta_d} E(\hat{\theta}_f,\hat{\theta}_y,\theta_d)$$

Approach

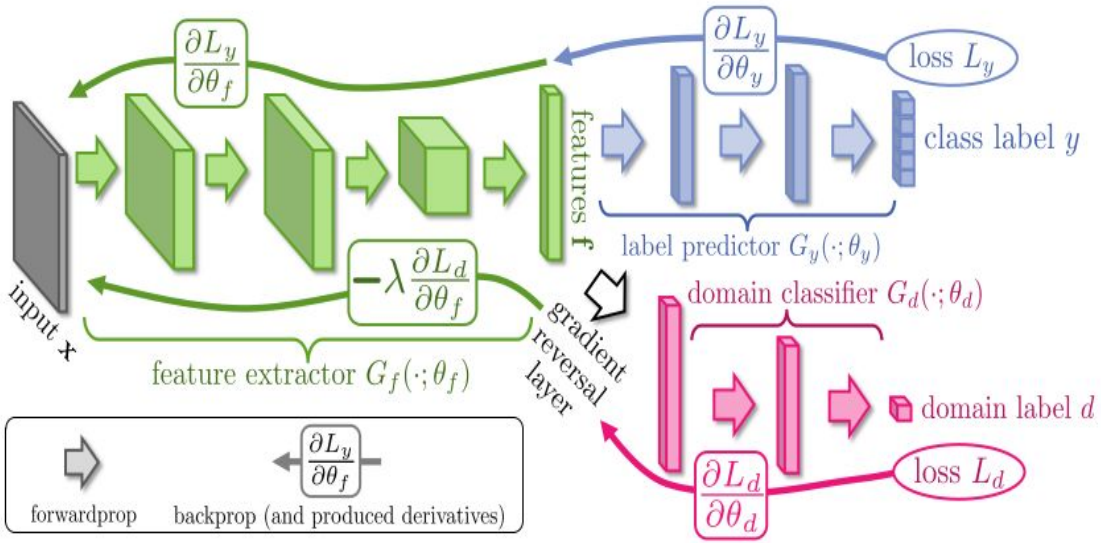


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

$$\mathcal{L}_y^i(\theta_f,\theta_y) = \mathcal{L}_y\big(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y),y_i\big)$$
$$\mathcal{L}_d^i(\theta_f,\theta_d) = \mathcal{L}_d\big(G_d(G_f(\mathbf{x}_i;\theta_f);\theta_d),d_i\big)$$

$$E(\theta_f,\theta_y,\theta_d) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_y^i(\theta_f,\theta_y) - \lambda\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_d^i(\theta_f,\theta_d) + \frac{1}{n'}\sum_{i=n+1}^{N}\mathcal{L}_d^i(\theta_f,\theta_d)\right)$$

$$(\hat\theta_f,\hat\theta_y) = \operatorname*{argmin}_{\theta_f,\theta_y} E(\theta_f,\theta_y,\hat\theta_d)$$
$$\hat\theta_d = \operatorname*{argmax}_{\theta_d} E(\hat\theta_f,\hat\theta_y,\theta_d)$$

$$\tilde E(\theta_f,\theta_y,\theta_d) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_y\left(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y),y_i\right)$$
$$-\lambda\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_d\left(G_d(\mathcal{R}(G_f(\mathbf{x}_i;\theta_f));\theta_d),d_i\right) + \frac{1}{n'}\sum_{i=n+1}^{N}\mathcal{L}_d\left(G_d(\mathcal{R}(G_f(\mathbf{x}_i;\theta_f));\theta_d),d_i\right)\right)$$
$$\mathcal{R}(\mathbf{x}) = \mathbf{x},$$
$$\frac{d\mathcal{R}}{d\mathbf{x}} = -\mathbf{I},$$

Approach

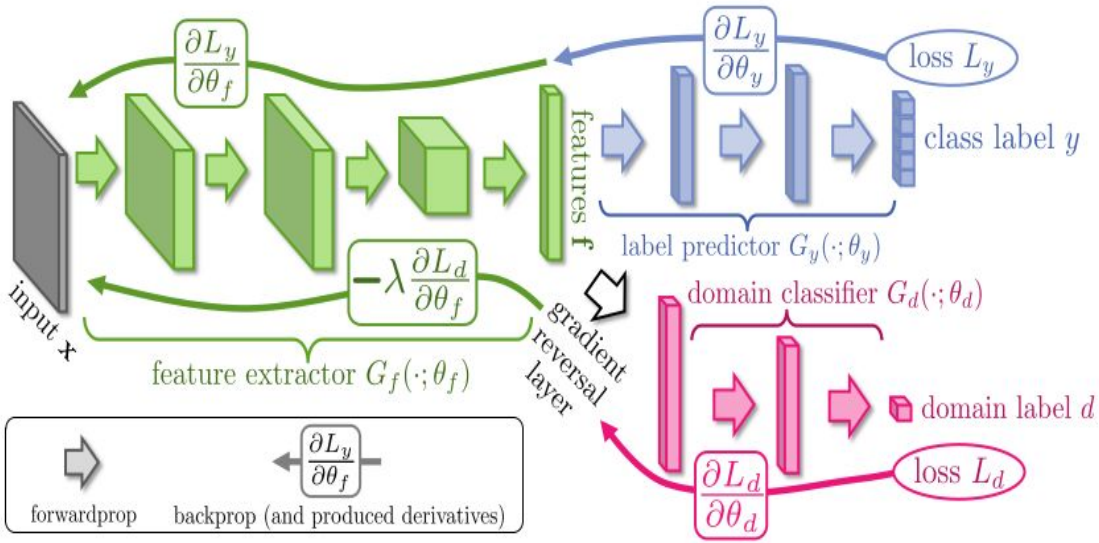


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

Learning rate $\mu_p = \frac{\mu_0}{(1+\alpha \cdot p)^\beta}$

$p \in [0, 1]$ progress of training

$\mu_0 = 0.01, \alpha = 10, \beta = 0.75$

Approach

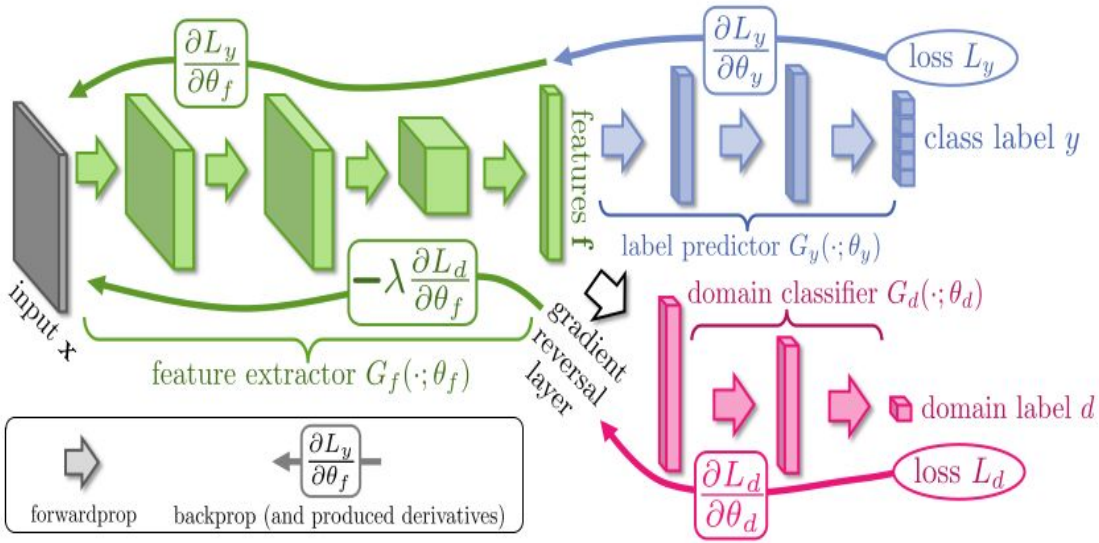


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

Learning rate $\mu_p = \frac{\mu_0}{(1+\alpha \cdot p)^\beta}$

$p \in [0, 1]$ progress of training

$\mu_0 = 0.01, \alpha = 10, \beta = 0.75$

For Feature Extractor updating DA parameter $\lambda_p = \frac{2}{1+e^{-\gamma \cdot p}} - 1$

$\gamma = 10$

Approach

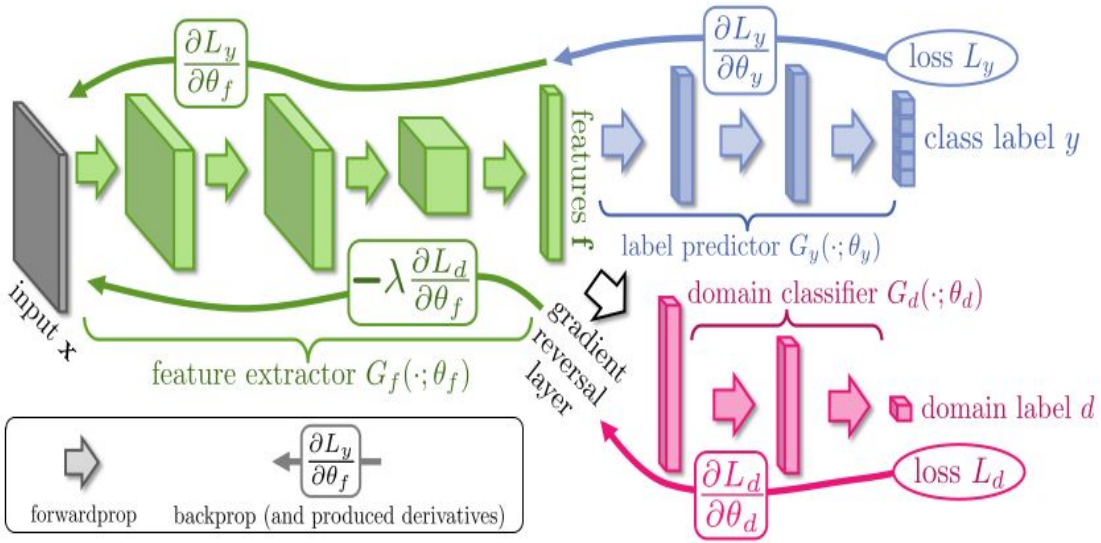


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

Learning rate $\mu_p = \frac{\mu_0}{(1+\alpha \cdot p)^\beta}$

$p \in [0, 1]$ progress of training

$\mu_0 = 0.01, \alpha = 10, \beta = 0.75$

For Feature Extractor updating DA parameter $\lambda_p = \frac{2}{1+e^{-\gamma \cdot p}} - 1$

$\gamma = 10$

For Domain Classifier updating

$\lambda = 1$

Approach

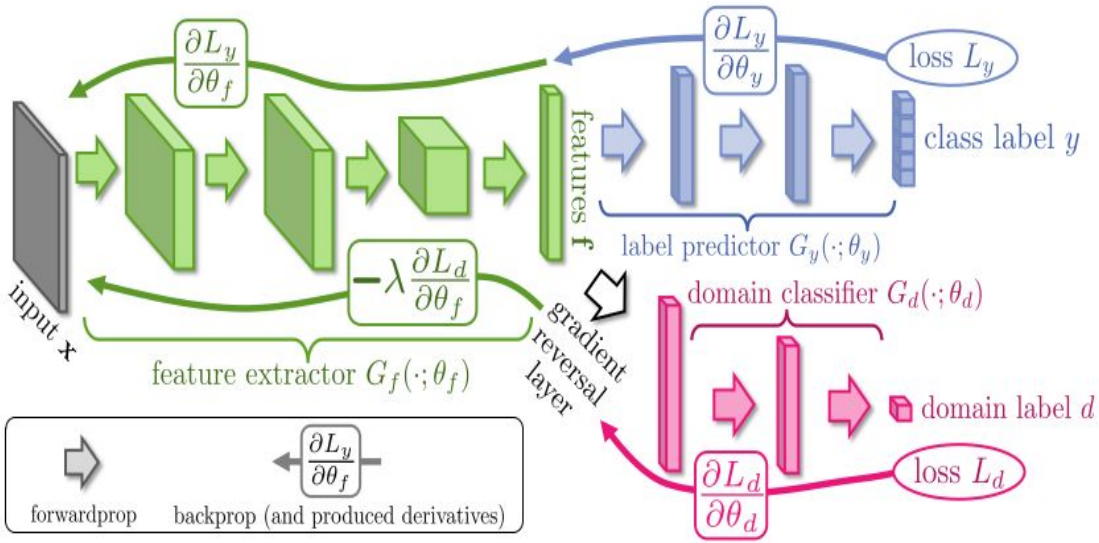


Figure 2. The proposed architecture. Image from *Ganin et al. 2016.*

Learning rate $\mu_p = \frac{\mu_0}{(1+\alpha \cdot p)^\beta}$

$p \in [0, 1]$ progress of training

$\mu_0 = 0.01, \alpha = 10, \beta = 0.75$

For Feature Extractor updating

DA parameter $\lambda_p = \frac{2}{1+e^{-\gamma \cdot p}} - 1$

$\gamma = 10$

For Domain Classifier updating

$\lambda = 1$

Batch size 128

64-Source & 64-Target

Results

Method	Source / Target	MNIST / MNIST-M	Syn Numbers / SVHN	SVHN / MNIST	Syn Signs / GTSRB
Source only		.5749	.8665	.5919	.7400
SA (Fernando et al., 2013)		.6078 (7.9%)	.8672 (1.3%)	.6157 (5.9%)	.7635 (9.1%)
Proposed approach		**.8149** (57.9%)	**.9048** (66.1%)	**.7107** (29.3%)	**.8866** (56.7%)
Train on target		.9891	.9244	.9951	.9987

Table 1. Classification accuracies for digit image classifications for different source and target domains. MNIST-M corresponds to difference-blended digits over non-uniform background. The first row corresponds to the lower performance bound (i.e. if no adaptation is performed). The last row corresponds to training on the target domain data with known class labels (upper bound on the DA performance). Table from Ganin et al. 2016.

Method	Source / Target	Amazon / Webcam	DSLR / Webcam	Webcam / DSLR
GFK(PLS, PCA) (Gong et al., 2012)		.464 ± .005	.613 ± .004	.663 ± .004
SA (Fernando et al., 2013)		.450	.648	.699
DA-NBNN (Tommasi & Caputo, 2013)		.528 ± .037	.766 ± .017	.762 ± .025
DLID (S. Chopra & Gopalan, 2013)		.519	.782	.899
DeCAF$_6$ Source Only (Donahue et al., 2014)		.522 ± .017	.915 ± .015	–
DaNN (Ghifary et al., 2014)		.536 ± .002	.712 ± .000	.835 ± .000
DDC (Tzeng et al., 2014)		.594 ± .008	.925 ± .003	.917 ± .008
Proposed Approach		**.673 ± .017**	**.940 ± .008**	**.937 ± .010**

Table 2. Accuracy evaluation of different DA approaches on the standard OFFICE dataset. Table from Ganin et al. 2016.

Bibliography

T. Batu, L. Fortnow, R. Rubinfeld, W. Smith, and P. White. *Testing that distributions are close*. In FOCS, volume 41, pages 259–269, 2000.

S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. *Analysis of representations for domain adaptation*. In NIPS, pages 137–144, 2006.

S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira and J.W. Vaughan. *A theory of learning from different domains*. In Machine Learning, 2010.

Y. Ganin and V. Lempitsky. 2015. *Unsupervised domain adaptation by backpropagation*. In Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research), Vol. 37. PMLR.

Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand and V. Lempitsky, *Domain-Adversarial Training of Neural Networks*, JMLR, 2016

Discussion

Background

- Let X be an **instance set**, Z be a **feature space** and $R : X \to Z$ a **representation** that maps them
- We define a distribution D over X and a target function $f : X \to [0, 1]$
- We also define a distribution D' over Z and a target function $f' : Z \to [0, 1]$ using the representation R
- Specifically: $P_{D'}[B] = P_D[R^{-1}(B)]$ and $f'(z) = E_D[f(x)|R(x) = z]$
- A **domain** is a distribution D over the instance X. We can define the corresponding distribution D' over Z
- We assume two domains: The **source** domain with D_S, D'_S and the **target** domain with D_T, D'_T. f, f' are common
- The goal is to approximate f' by estimating a **hypothesis** function $h : Z \to [0, 1], h \in H$ from the hypothesis space H
- The **source error** is defined as $\varepsilon_S(h) = E_{z \sim D'_S}|f'(z) - h(z)|$ and the **target error** as $\varepsilon_T(h) = E_{z \sim D'_T}|f'(z) - h(z)|$

Distance Between Distributions

- Variational Distance: $d_{L_1}(D_S, D_T) = 2 \sup_{B \in \mathcal{B}} |Pr_{D_S}[B] - Pr_{D_T}[B]|$

  Is the largest possible difference between the probabilities that the two distributions can assign to the same event.

  Supremum is over all measurable subsets under D_S, D_T. Cannot be computed for real valued distributions from finite samples. *Batu et al. 2000*

- H-Divergence: $d_H(D_S, D_T) = 2 \sup_{h \in H} |Pr_{D_S}[h(x) = 1] - Pr_{D_T}[h(x) = 1]|$

  Limits the supremum over the hypothesis set. For H of finite VC dimension it can be estimated from finite samples.

Target Error Bound

Theorem 2 *Ben-David et al. 2006*

Let R be a fixed representation and H be a hypothesis space of VC dimension d. If a random labeled sample S of size m is generated by applying R to a D_S i.i.d. sample and an unlabeled sample T of size m' is generated by applying R to a D_T^X i.i.d. sample, then with probability 1-δ, for every hypothesis h:

$$R_{D_T}(h) \leq R_S(h) + \hat{d}_H(S,T) + \lambda + \frac{4}{m}\sqrt{d\log\left(\frac{2em}{d}\right) + \log\left(\frac{4}{\delta}\right)} + 4\sqrt{\frac{d\log(2m') + \log\left(\frac{4}{\delta}\right)}{m'}}$$

$$\lambda \geq \inf_{h^* \in H}[R_{D_S}(h^*) + R_{D_T}(h^*)]$$

The **dataset size m, m'** and **uncertainty δ** trade-off. For complete certainty: while δ approaches zero the terms approach to infinity. When the dataset sizes m, m' approach to infinity, the terms approach zero.