Domain-Adversarial Neural Networks

Hana Ajakan[1], **Pascal Germain**[1], Hugo Larochelle[2],
François Laviolette[1], Mario Marchand[1]

[1] Département d'informatique et de génie logiciel, Université Laval, Québec, Canada
[2] Département d'informatique, Université de Sherbrooke, Québec, Canada

Groupe de recherche en apprentissage automatique de l'Université Laval
(GRAAL)

December 13, 2014

Outline

1 Domain Adaptation Setting

2 Domain Divergence

3 Neural Network for Domain Adaptation

4 Empirical Results

Outline

1 Domain Adaptation Setting

2 Domain Divergence

3 Neural Network for Domain Adaptation

4 Empirical Results

Our Domain Adaptation Setting

Binary classification tasks

- Input space: $\mathbb{R}^d$
- Labels: $\{0, 1\}$

Two different data distributions

- Source domain: $\mathcal{D}_S$
- Target domain: $\mathcal{D}_T$

A **domain adaptation** learning algorithm is provided with

a **labeled source sample** $S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m$,

an **unlabeled target sample** $T = \{\mathbf{x}_i^t\}_{i=1}^m \sim (\mathcal{D}_T)^m$.

The goal is to build a classifier $\eta : \mathbb{R}^d \rightarrow \{0, 1\}$ with a low **target risk**

$$R_{\mathcal{D}_T}(\eta) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x}^t, y^t) \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) \neq y^t\right].$$

Our Domain Adaptation Setting

Binary classification tasks

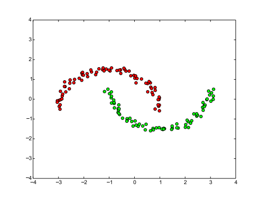
- Input space: $\mathbb{R}^d$
- Labels: $\{0, 1\}$

Two different data distributions

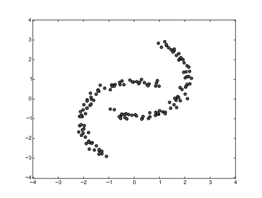
- Source domain: $\mathcal{D}_S$
- Target domain: $\mathcal{D}_T$

A **domain adaptation** learning algorithm is provided with

a **labeled source sample**
$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m,$



an **unlabeled target sample**
$T = \{\mathbf{x}_i^t\}_{i=1}^m \sim (\mathcal{D}_T)^m.$



The goal is to build a classifier $\eta : \mathbb{R}^d \to \{0, 1\}$ with a low **target risk**

$$R_{\mathcal{D}_T}(\eta) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x}^t, y^t) \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) \neq y^t \right].$$

Our Domain Adaptation Setting

Binary classification tasks

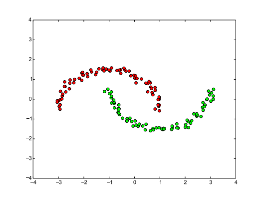
- Input space: $\mathbb{R}^d$
- Labels: $\{0, 1\}$

Two different data distributions

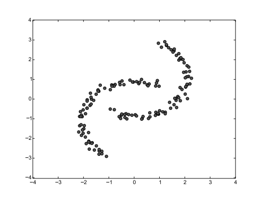
- Source domain: $\mathcal{D}_S$
- Target domain: $\mathcal{D}_T$

A **domain adaptation** learning algorithm is provided with

a **labeled source sample**
$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m$,



an **unlabeled target sample**
$T = \{\mathbf{x}_i^t\}_{i=1}^m \sim (\mathcal{D}_T)^m$.



The goal is to build a classifier $\eta : \mathbb{R}^d \to \{0, 1\}$ with a low **target risk**

$$R_{\mathcal{D}_T}(\eta) \stackrel{\text{def}}{=} \Pr_{(\mathbf{x}^t, y^t) \sim \mathcal{D}_T} [\eta(\mathbf{x}^t) \neq y^t] .$$

Outline

1 Domain Adaptation Setting

2 Domain Divergence

3 Neural Network for Domain Adaptation

4 Empirical Results

Divergence between source and target domains

Definition (Ben David et al., 2006)

Given two domain distributions $\mathcal{D}_S$ and $\mathcal{D}_T$, and a **hypothesis class** $\mathcal{H}$, the **$\mathcal{H}$-divergence** between $\mathcal{D}_S$ and $\mathcal{D}_T$ is

$$d_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \stackrel{\text{def}}{=} 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_S} \left[\eta(\mathbf{x}^s) = 1\right] - \Pr_{\mathbf{x}^t \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) = 1\right] \right| .$$

$$= 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_S} \left[\eta(\mathbf{x}^s) = 1\right] + \Pr_{\mathbf{x}^t \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) = 0\right] - 1 \right| .$$

The **$\mathcal{H}$-divergence** measures the ability of an hypothesis class $\mathcal{H}$ to **discriminate** between source $\mathcal{D}_S$ and target $\mathcal{D}_T$ distributions.

Divergence between source and target domains

Definition (Ben David et al., 2006)

Given two domain distributions $\mathcal{D}_S$ and $\mathcal{D}_T$, and a **hypothesis class** $\mathcal{H}$, the **$\mathcal{H}$-divergence** between $\mathcal{D}_S$ and $\mathcal{D}_T$ is

$$d_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \stackrel{\text{def}}{=} 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_S} \left[\eta(\mathbf{x}^s) = 1\right] - \Pr_{\mathbf{x}^t \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) = 1\right] \right| .$$

$$= 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_S} \left[\eta(\mathbf{x}^s) = 1\right] + \Pr_{\mathbf{x}^t \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) = 0\right] - 1 \right| .$$

The **$\mathcal{H}$-divergence** measures the ability of an hypothesis class $\mathcal{H}$ to **discriminate** between source $\mathcal{D}_S$ and target $\mathcal{D}_T$ distributions.

Divergence between source and target domains

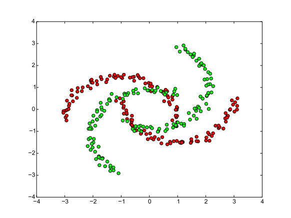
Definition (Ben David et al., 2006)

Given two domain distributions $\mathcal{D}_S$ and $\mathcal{D}_T$, and a **hypothesis class** $\mathcal{H}$, the $\mathcal{H}$**-divergence** between $\mathcal{D}_S$ and $\mathcal{D}_T$ is

$$d_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \stackrel{\text{def}}{=} 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_S} \left[\eta(\mathbf{x}^s) = 1 \right] - \Pr_{\mathbf{x}^t \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) = 1 \right] \right| .$$

$$= 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_S} \left[\eta(\mathbf{x}^s) = 1 \right] + \Pr_{\mathbf{x}^t \sim \mathcal{D}_T} \left[\eta(\mathbf{x}^t) = 0 \right] - 1 \right| .$$

The $\mathcal{H}$**-divergence** measures the ability of an hypothesis class $\mathcal{H}$ to **discriminate** between source $\mathcal{D}_S$ and target $\mathcal{D}_T$ distributions.



Bound on the target risk

Theorem (Ben David et al., 2006)

Let $\mathcal{H}$ be a hypothesis class of VC-dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_S)^m$ and $T \sim (\mathcal{D}_T)^m$, for every $\eta \in \mathcal{H}$:

$$R_{\mathcal{D}_T}(\eta) \;\leq\; R_S(\eta) + \frac{4}{m}\sqrt{d \log \tfrac{2e\,m}{d} + \log \tfrac{4}{\delta}} + \hat{d}_{\mathcal{H}}(S,T) + \frac{4}{m^2}\sqrt{d \log \tfrac{2\,m}{d} + \log \tfrac{4}{\delta}} + \beta$$

with $\beta \geq \inf\limits_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*)\right].$

Empirical risk on the source sample:

$$R_S(\eta) \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) \neq y_i^s].$$

Empirical $\mathcal{H}$-divergence:

$$\hat{d}_{\mathcal{H}}(S,T) \;\stackrel{\text{def}}{=}\; 2\max_{\eta \in \mathcal{H}} \left[\frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) = 1] + \frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^t) = 0] - 1\right].$$

Bound on the target risk

Theorem (Ben David et al., 2006)

Let $\mathcal{H}$ be a hypothesis class of VC-dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_S)^m$ and $T \sim (\mathcal{D}_T)^m$, for every $\eta \in \mathcal{H}$:

$$R_{\mathcal{D}_T}(\eta) \;\leq\; R_S(\eta) + \frac{4}{m}\sqrt{d \log \tfrac{2e\,m}{d} + \log \tfrac{4}{\delta}} + \hat{d}_{\mathcal{H}}(S,T) + \frac{4}{m^2}\sqrt{d \log \tfrac{2\,m}{d} + \log \tfrac{4}{\delta}} + \beta$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*)\right]$.

Empirical risk on the source sample:

$$R_S(\eta) \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) \neq y_i^s]\,.$$

Empirical $\mathcal{H}$-divergence:

$$\hat{d}_{\mathcal{H}}(S,T) \;\stackrel{\text{def}}{=}\; 2 \max_{\eta \in \mathcal{H}} \left[\frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) = 1] + \frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^t) = 0] - 1\right].$$

Bound on the target risk

Theorem (Ben David et al., 2006)

Let $\mathcal{H}$ be a hypothesis class of VC-dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_S)^m$ and $T \sim (\mathcal{D}_T)^m$, for every $\eta \in \mathcal{H}$:

$$R_{\mathcal{D}_T}(\eta) \;\le\; R_S(\eta) + \frac{4}{m}\sqrt{d \log \tfrac{2e\,m}{d} + \log \tfrac{4}{\delta}} + \hat{d}_{\mathcal{H}}(S,T) + \frac{4}{m^2}\sqrt{d \log \tfrac{2\,m}{d} + \log \tfrac{4}{\delta}} + \beta$$

with $\beta \ge \inf_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*) \right]$.

Empirical risk on the source sample:

$$R_S(\eta) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) \neq y_i^s]\,.$$

Empirical $\mathcal{H}$-divergence:

$$\hat{d}_{\mathcal{H}}(S,T) \stackrel{\text{def}}{=} 2\max_{\eta\in\mathcal{H}} \left[\frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) = 1] + \frac{1}{m}\sum_{i=1}^{m} I[\eta(\mathbf{x}_i^t) = 0] - 1 \right].$$

Bound on the target risk

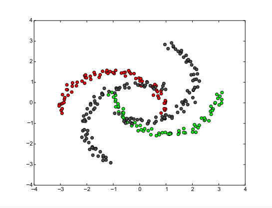
Theorem (Ben David et al., 2006)

Let $\mathcal{H}$ be a hypothesis class of VC-dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_S)^m$ and $T \sim (\mathcal{D}_T)^m$, for every $\eta \in \mathcal{H}$:

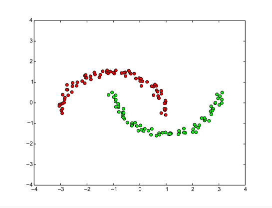
$$R_{\mathcal{D}_T}(\eta) \;\leq\; R_S(\eta) + \frac{4}{m}\sqrt{d\log\frac{2e\,m}{d} + \log\frac{4}{\delta}} + \hat{d}_{\mathcal{H}}(S,T) + \frac{4}{m^2}\sqrt{d\log\frac{2\,m}{d} + \log\frac{4}{\delta}} + \beta$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*)\right]$.

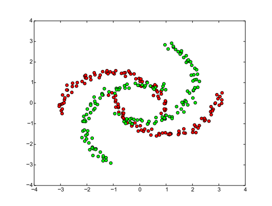
Target risk $R_{\mathcal{D}_T}(\eta)$ is low
if, given S and T,



$R_S(\eta)$ is small,
i.e., $\eta \in \mathcal{H}$ is good on



and $\hat{d}_{\mathcal{H}}(S,T)$ is small,
i.e., all $\eta' \in \mathcal{H}$ are bad on



Outline

1 Domain Adaptation Setting

2 Domain Divergence

3 Neural Network for Domain Adaptation

4 Empirical Results

Standard Neural Network

Let consider a neural network architecture with one hidden layer

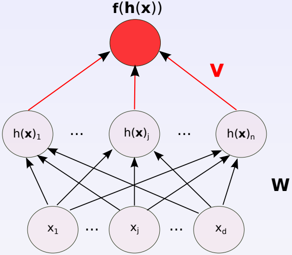
$$\mathbf{h}(\mathbf{x}) = \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}), \quad \text{and} \quad \mathbf{f}(\mathbf{x}) = \mathrm{softmax}(\mathbf{c} + \mathbf{V}\mathbf{h}(\mathbf{x})).$$

Given a training source sample

$$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m,$$

We optimize by back-propagation:

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \underbrace{\left[\frac{1}{m}\sum_{i=1}^{m} -\log\left(1 - y_i^s - \mathbf{f}(\mathbf{h}(\mathbf{x}_i^s))\right)\right]}_{\text{source loss}}.$$



The hidden layer learns a **representation** $\mathbf{h}(\cdot)$ from which linear hypothesis $\mathbf{f}(\cdot)$ can **classify source examples**.

Standard Neural Network

Let consider a neural network architecture with one hidden layer

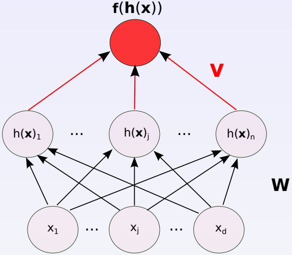
$$\mathbf{h}(\mathbf{x}) \;=\; \mathrm{sigm}(\mathbf{b}+\mathbf{W}\mathbf{x})\,, \quad \text{and} \quad \mathbf{f}(\mathbf{x}) \;=\; \mathrm{softmax}(\mathbf{c}+\mathbf{V}\mathbf{h}(\mathbf{x}))\,.$$

Given a training source sample

$$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m\,,$$

We optimize by back-propagation:

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \underbrace{\left[\frac{1}{m}\sum_{i=1}^{m} -\log\Big(1 - y_i^s - \mathbf{f}(\mathbf{h}(\mathbf{x}_i^s))\Big)\right]}_{\text{source loss}}\,.$$



The hidden layer learns a **representation h**$(\cdot)$ from which linear hypothesis **f**$(\cdot)$ can **classify source examples**.

Standard Neural Network

Let consider a neural network architecture with one hidden layer

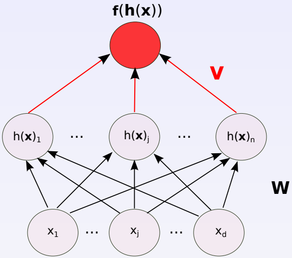
$$\mathbf{h}(\mathbf{x}) = \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}), \quad \text{and} \quad \mathbf{f}(\mathbf{x}) = \mathrm{softmax}(\mathbf{c} + \mathbf{V}\mathbf{h}(\mathbf{x})).$$

Given a training source sample

$$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m,$$

We optimize by back-propagation:

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \underbrace{\left[\frac{1}{m}\sum_{i=1}^{m} -\log\left(1 - y_i^s - \mathbf{f}(\mathbf{h}(\mathbf{x}_i^s))\right)\right]}_{\text{source loss}}.$$



The hidden layer learns a **representation** $\mathbf{h}(\cdot)$ from which linear hypothesis $\mathbf{f}(\cdot)$ can **classify source examples**.

Domain-Adversarial Neural Network (DANN)

Empirical $\mathcal{H}$-divergence

$$\hat{d}_{\mathcal{H}}(S,T) \stackrel{\text{def}}{=} 2 \max_{\eta \in \mathcal{H}} \left[\frac{1}{m} \sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) = 1] + \frac{1}{m} \sum_{i=1}^{m} I[\eta(\mathbf{x}_i^t) = 0] - 1 \right].$$

We estimate the $\mathcal{H}$-divergence by a logistic regressor that model the probability that a given input (either $\mathbf{x}^s$ or $\mathbf{x}^t$) is from the source domain:

$$o(\mathbf{h}(\mathbf{x})) \stackrel{\text{def}}{=} \mathrm{sigm}(d + \mathbf{w}^\top \mathbf{h}(\mathbf{x})).$$

Given a representation outputted by the hidden layer $\mathbf{h}(\cdot)$:

$$\hat{d}_{\mathcal{H}}\Big(\mathbf{h}(S), \mathbf{h}(T)\Big) \approx 2 \max_{\mathbf{w},d} \left[\frac{1}{m} \sum_{i=1}^{m} \log\big(o(\mathbf{h}(\mathbf{x}_i^s))\big) + \frac{1}{m} \sum_{i=1}^{m} \log\big(1 - o(\mathbf{h}(\mathbf{x}_i^t))\big) - 1 \right].$$

Domain-Adversarial Neural Network (DANN)

Empirical $\mathcal{H}$-divergence

$$\hat{d}_{\mathcal{H}}(S, T) \stackrel{\text{def}}{=} 2 \max_{\eta \in \mathcal{H}} \left[\frac{1}{m} \sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) = 1] + \frac{1}{m} \sum_{i=1}^{m} I[\eta(\mathbf{x}_i^t) = 0] - 1 \right].$$

We estimate the $\mathcal{H}$-divergence by a logistic regressor that model the probability that a given input (either $\mathbf{x}^s$ or $\mathbf{x}^t$) is from the source domain:

$$o(\mathbf{h}(\mathbf{x})) \stackrel{\text{def}}{=} \operatorname{sigm}(d + \mathbf{w}^\top \mathbf{h}(\mathbf{x})).$$

Given a representation outputted by the hidden layer $\mathbf{h}(\cdot)$:

$$\hat{d}_{\mathcal{H}}\Big(\mathbf{h}(S), \mathbf{h}(T)\Big) \approx 2 \max_{\mathbf{w}, d} \left[\frac{1}{m} \sum_{i=1}^{m} \log\big(o(\mathbf{h}(\mathbf{x}_i^s))\big) + \frac{1}{m} \sum_{i=1}^{m} \log\big(1 - o(\mathbf{h}(\mathbf{x}_i^t))\big) - 1 \right].$$

Domain-Adversarial Neural Network (DANN)

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \Bigg[\underbrace{\frac{1}{m}\sum_{i=1}^{m} -\log\big(1 - y_i^s - \mathbf{f}(\mathbf{h}(\mathbf{x}_i^s))\big)}_{\text{source loss}} + \underbrace{\lambda \max_{\mathbf{w},d} \left(\frac{1}{m}\sum_{i=1}^{m} \log\big(o(\mathbf{h}(\mathbf{x}_i^s))\big) + \frac{1}{m}\sum_{i=1}^{m} \log\big(1 - o(\mathbf{h}(\mathbf{x}_i^t))\big) \right)}_{\text{adaptation regularizer}} \Bigg],$$

where $\lambda > 0$ weights the domain adaptation regularization term.

Given a source sample

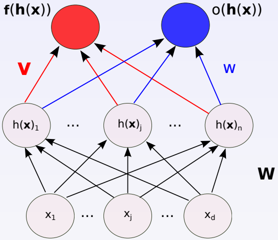
$$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m,$$

and a target sample

$$T = \{(\mathbf{x}_i^t)\}_{i=1}^m \sim (\mathcal{D}_T)^m,$$

DANN finds a representation $\mathbf{h}(\cdot)$:

- Good on S; but
- unable to discriminate between S and T.



Domain-Adversarial Neural Network (DANN)

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \Bigg[\underbrace{\frac{1}{m}\sum_{i=1}^{m} -\log\big(1-y_i^s-\mathbf{f}(\mathbf{h}(\mathbf{x}_i^s))\big)}_{\text{source loss}} + \underbrace{\lambda \max_{\mathbf{w},d}\Bigg(\frac{1}{m}\sum_{i=1}^{m}\log\big(o(\mathbf{h}(\mathbf{x}_i^s))\big)+\frac{1}{m}\sum_{i=1}^{m}\log\big(1-o(\mathbf{h}(\mathbf{x}_i^t))\big)\Bigg)}_{\text{adaptation regularizer}} \Bigg],$$

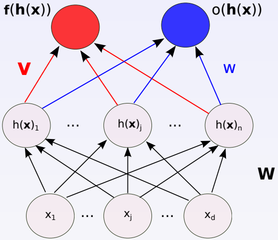
where $\lambda > 0$ weights the domain adaptation regularization term.

Given a source sample

$$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (\mathcal{D}_S)^m,$$

and a target sample

$$T = \{(\mathbf{x}_i^t)\}_{i=1}^m \sim (\mathcal{D}_T)^m,$$



DANN finds a representation $\mathbf{h}(\cdot)$:

- Good on S; but
- unable to discriminate between S and T.

Outline

1 Domain Adaptation Setting

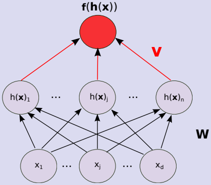
2 Domain Divergence

3 Neural Network for Domain Adaptation

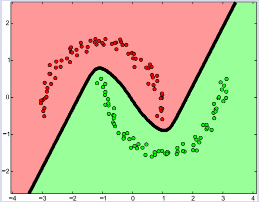
4 Empirical Results

Toy Dataset

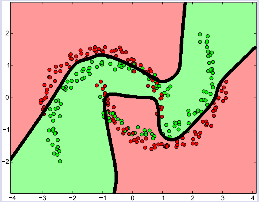
Standard Neural Network (NN)



Trained to classify source



Trained to classify domains

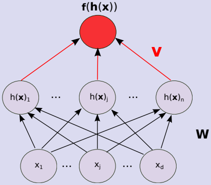


Domain-Adversarial Neural Networks (DANN)

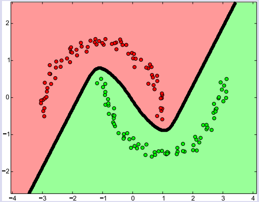
Classification output: $\mathbf{f}(\mathbf{h}(\mathbf{x}))$ Domain output: $o(\mathbf{h}(\mathbf{x}))$

Toy Dataset

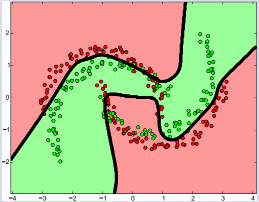
Standard Neural Network (NN)



Trained to classify source



Trained to classify domains



Domain-Adversarial Neural Networks (DANN)

Classification output: $\mathbf{f}(\mathbf{h}(\mathbf{x}))$

Domain output: $o(\mathbf{h}(\mathbf{x}))$

Amazon Reviews

Input: product review (bag or words) — **Output:** positive or negative rating.

Dataset	**DANN**	**NN**	**SVM**
books → dvd	0.201	**0.199**	0.206
books → electronics	**0.246**	0.251	0.256
books → kitchen	0.230	0.235	**0.229**
dvd → books	**0.247**	0.261	0.269
dvd → electronics	**0.247**	0.256	0.249
dvd → kitchen	**0.227**	**0.227**	0.233
electronics → books	**0.280**	0.281	0.290
electronics → dvd	**0.273**	0.277	0.278
electronics → kitchen	**0.148**	0.149	0.163
kitchen → books	**0.283**	0.288	0.325
kitchen → dvd	0.261	**0.261**	0.274
kitchen → electronics	0.161	0.161	**0.158**

Note: We use a *small labeled subset* of 100 target examples to select the hyperparameters.

Marginalized Stacked Denoising Autoencoders (mSDA)

Question

Does DANN can be combined with other representation learning techniques for domain adaptation?

With **mSDA**, Chen et al. (2012) obtained *state-of-the-art* results on Amazon Reviews.

1) Find a new common representation for source and target (unsupervised)
2) Learn a linear SVM on the new source representations.

Let replace **Step 2**:

1) Find a new common representation for source and target (unsupervised)
2) Learn with DANN on source representations and target representations.

Marginalized Stacked Denoising Autoencoders (mSDA)

Question

Does DANN can be combined with other representation learning techniques for domain adaptation?

With **mSDA**, Chen et al. (2012) obtained *state-of-the-art* results on Amazon Reviews.

1) Find a new common representation for source and target (unsupervised)
2) Learn a linear SVM on the new source representations.

Let replace **Step 2**:

1) Find a new common representation for source and target (unsupervised)
2) Learn with DANN on source representations and target representations.

Marginalized Stacked Denoising Autoencoders (mSDA)

Question

Does DANN can be combined with other representation learning techniques for domain adaptation?

With **mSDA**, Chen et al. (2012) obtained *state-of-the-art* results on Amazon Reviews.

1) Find a new common representation for source and target (unsupervised)
2) Learn a linear SVM on the new source representations.

Let replace **Step 2**:

1) Find a new common representation for source and target (unsupervised)
2) Learn with DANN on source representations and target representations.

Amazon Reviews

Input: product review (bag or words) — **Output:** positive or negative rating.

Dataset **(mSDA representations)**	**DANN**	**NN**	**SVM**
books $\rightarrow$ dvd	0.176	**0.171**	0.175
books $\rightarrow$ electronics	**0.197**	0.228	0.244
books $\rightarrow$ kitchen	0.169	**0.166**	0.172
dvd $\rightarrow$ books	0.176	**0.173**	0.176
dvd $\rightarrow$ electronics	**0.181**	0.234	0.220
dvd $\rightarrow$ kitchen	**0.151**	0.153	0.178
electronics $\rightarrow$ books	**0.237**	0.241	0.229
electronics $\rightarrow$ dvd	**0.216**	0.228	0.261
electronics $\rightarrow$ kitchen	**0.118**	0.126	0.137
kitchen $\rightarrow$ books	**0.222**	0.226	0.234
kitchen $\rightarrow$ dvd	**0.208**	0.214	0.209
kitchen $\rightarrow$ electronics	0.141	**0.136**	0.138

Note: We use a *small labeled subset* of 100 target examples to select the hyperparameters. The *noise parameter* of mSDA representations is fixed to 50%.

Future Work

Several paths to explore:

- Deeper neural networks architectures.
- Multiclass / Multilabels problems.
- Multisource domain adaptation.

Thank you!

Future Work

Several paths to explore:

- Deeper neural networks architectures.
- Multiclass / Multilabels problems.
- Multisource domain adaptation.

Thank you!