Domain adaptation and counterfactual prediction

Presenter: Zhe Wang

https://qdata.github.io/deep2Read

Zhe Wang

201909

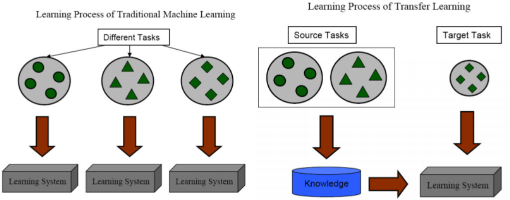
Content

1. Domain Adaptation
   - Distance based
   - Adversarial based
   - Reconstruction based
2. Counterfactual Inference

Transfer Learning[1]

Definition (Transfer Learning)

Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.



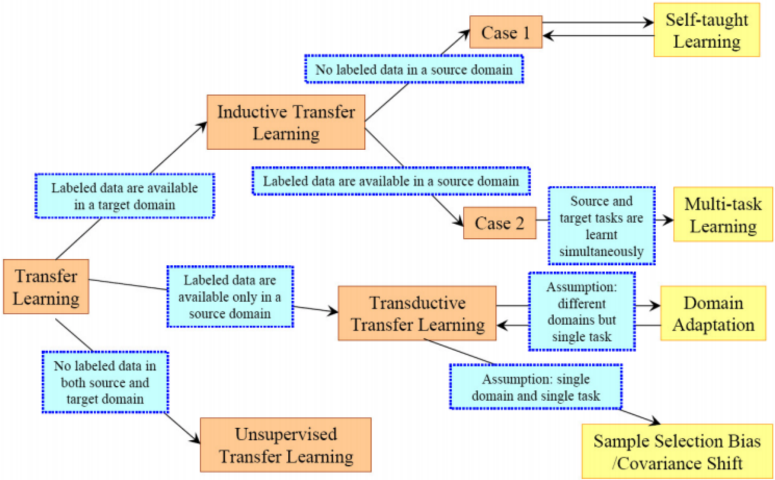
(a) Traditional Machine Learning (b) Transfer Learning

Fig. 1. Different Learning Processes between Traditional Machine Learning and Transfer Learning

[1] A Survey on Transfer Learning, Pan and Yang, IEEE TKDE, 2009

Category of transfer learning

Determined by the availability of labels, the relationship between $\mathcal{D}_S$ and $\mathcal{D}_T$, $\mathcal{T}_S$ and $\mathcal{T}_T$, transfer learning can be categorized as follows:



Domain Adaptation[2]

Setting:

- A set of labeled data $\{x_s, y_s\}_{s=1}^{m}$ from the source domain $\mathcal{D}_S$,
- A set of unlabeled data $\{x_t\}_{t=1}^{n}$ from the target domain $\mathcal{D}_T$,
- The source domain and the target domain share the same task, i.e. $\mathcal{T}_S = \mathcal{T}_T$.

Why we need the DA?
Deep model trained on one dataset may have infinite error bound on another similar dataset.

Based on the results of [Yosinski, et.al, NeurIPS2014]:

- In shallow convolutional layers can learn generic features that tend to be transferable in shallow layers.
- In middle layers, features are slightly domain-biased, and the transferability drops.
- In deep layers, features are more task or domain specific and are not safely transferable to novel tasks.

[2] Deep Visual Domain Adaptation: A Survey, Wang and Deng, NeuralComputing

Domain Adaptation

Methods:

- Discrepancy based
- Adversarial based
- Reconstruction based

Main idea:
Learn features that are both **predictive** and **invariant** across different domains.

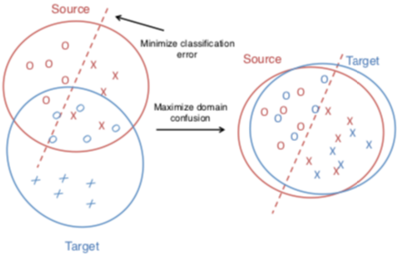
In the paper, maximum mean discrepancy(MMD) is used to measure the discrepancy of two distributions.

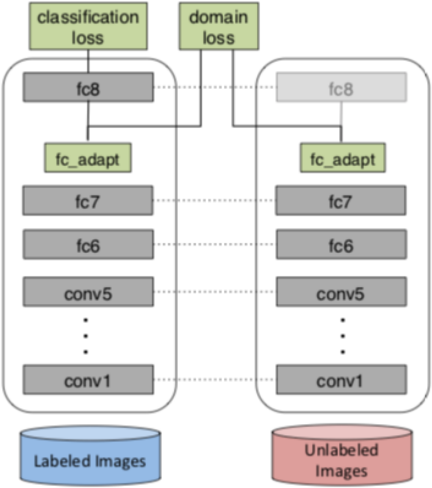
MMD:

$$MMD(P_x, P_y) = \left|\left|\frac{1}{|x_i|}\sum_{x_i \in P_x} \phi(x_i) - \frac{1}{|x_j|}\sum_{x_j \in P_y} \phi(x_j)\right|\right|$$

Motivation

Using the MMD as a regularization to find the invariant features which are also predictive.

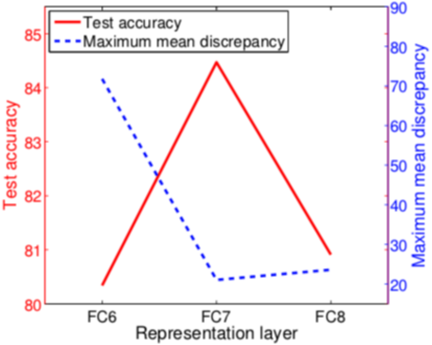



classification loss
domain loss
fc8
fc8
fc_adapt
fc_adapt
fc7
fc7
fc6
fc6
conv5
conv5
conv1
conv1
Labeled Images
Unlabeled Images

Where to insert the MMD regularization?
Starting from a pretrained model (such as AlexNet trained on ImageNet), find the layer has the smallest MMD on $\mathcal{D}_S, \mathcal{D}_t$. Insert the regularization there.

Loss function:

$$L = L_c(X_s, Y_s) + \lambda MMD^2(X_s, X_t)$$



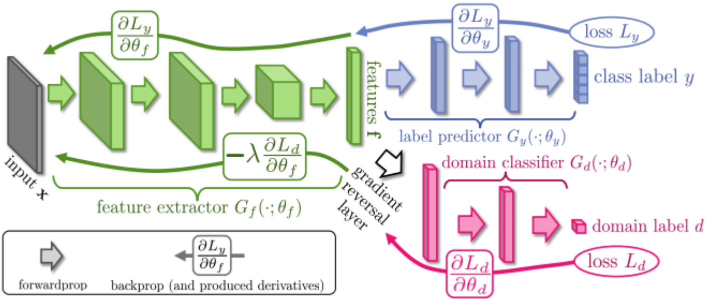
Adversarial based method[4]

Main idea: Adding a classifier to distinguish data from two domains.

Three modules:
Feature extractor: $G_f(\cdot, \theta_f)$, label predictor: $G_y(\cdot, \theta_y)$ and domain classifier: $G_d(\cdot, \theta_d)$.

- For label predictor, the inputs are the features and labels from source domain, the goal is to correctly predict the labels.
- For domain classifier, the inputs are features from both source domain and target domain, the goal is to correctly distinguish two sets.
- For feature extractor, the goal is to 1) generate predictive features for source domain. 2) fool the domain classifier.

[4] Unsupervised Domain Adaptation by Backpropagation, Ganin and Lempitsky, ICML2015



Energy function:

$$E(\theta_f, \theta_y, \theta_d) = L_y(G_y(G_f(x_s, \theta_f); \theta_y), y_s) - \lambda L_d(G_d(G_f(x; \theta_f); \theta_d), y_d)$$

Based on the idea, energy function is optimized to seek the saddle point:

$$\begin{aligned}(\hat{\theta}_f, \hat{\theta}_y) &= \arg\min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d) \\ \hat{\theta}_d &= \arg\max_{d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d)\end{aligned} \tag{1}$$

If gradient descent based optimizer is used:

$$\theta_f \leftarrow \theta_f - \mu(\frac{\partial L_y}{\partial \theta_f} - \lambda \frac{\partial L_d}{\partial \theta_f})$$
$$\theta_y \leftarrow \theta_y - \mu \frac{\partial L_y}{\theta_y} \quad (2)$$
$$\theta_d \leftarrow \theta_d - \mu \frac{\partial L_d}{\theta_d}$$

To avoid training different module alternatively, a new layer called **gradient reversal layer**(GRL) is defined as:

$$GRL.forward(x) = x, GRL.backward(\frac{dl}{dx}) = \frac{dl}{dx}(-\lambda I)$$

The new loss function is:

$$E(\theta_f, \theta_y, \theta_d) = L_y(G_y(G_f(x_s, \theta_f); \theta_y), y_s) + \lambda L_d(G_d(GRL(G_f(x; \theta_f)); \theta_d), y_d)$$

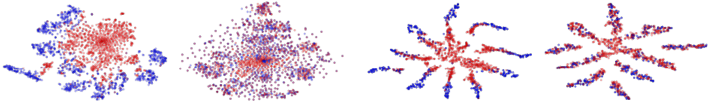
Now, all parameters can be jointly trained with gradient descent.

Experiment results

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%)
Proposed approach	**.7666** (52.9%)	**.9109** (79.7%)	**.7385** (42.6%)	**.8865** (46.4%)
Train on target	.9596	.9220	.9942	.9980

MNIST → MNIST-M: top feature extractor layer

SYN NUMBERS → SVHN: last hidden layer of the label predictor



(a) Non-adapted (b) Adapted (a) Non-adapted (b) Adapted

Reconstruction based method[5]

Motivation

- Shared representations are vulnerable to contamination by noise that is correlated with the underlying shared distribution
- There should be a subspace for each domain contains domain specific noise, and a common subspace contains shared features.
- The features in private subspace should be independent of features in common space.

[5] Domain Separation Networks, Bousmalis, et.al, NeurIPS2016

Several modules:

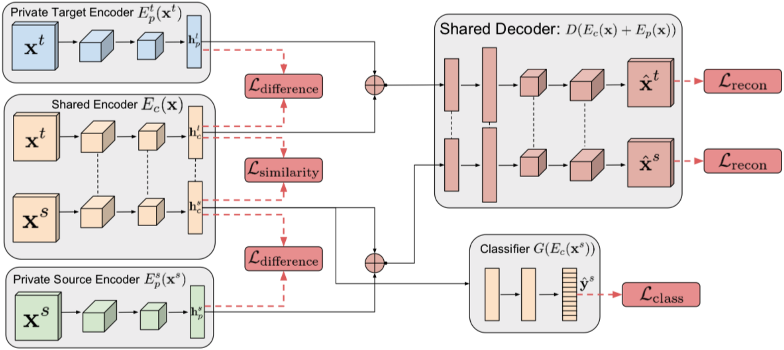
Shared encoder $E_c(\cdot, \theta_c)$ as common feature extractor

Private encoder $E_p^s(\cdot, \theta_{pt})$ as private feature extractor for $\mathcal{D}_S$,

Private encoder $E_p^t(\cdot,_{ps})$ as private feature extractor for $\mathcal{D}_T$,

Shared decoder $D_c(E_c(x) + E_p(x), \theta_d)$ as a decoder.

Task-related module, such as classifier $G(\cdot, \theta_g)$.



Loss functions:

- for L_{class}, general cross entropy is used
- for l_{recon}, general L2 loss is used
- for $l_{difference}$, it measures the difference of common features and private features, to force the independence,

$$L_{diff} = ||(H_c^s)^T H_p^s||_F^2 + ||(H_c^t)^T H_p^t||_F^2$$

- for $l_{similarity}$, it can be set as a domain classifier with gradient reverse layer or a MMD module. For domain classifier, the loss is defined as:

$$L = \sum_{i=0}^{n_s+n_t} \{d_i \log \hat{d}_i + (1 - d_i) \log(1 - \hat{d}_i)\}$$

The final loss is the linear combination of the four losses.

Experiment Results

Model	**MNIST to MNIST-M**	**Synth Digits to SVHN**	**SVHN to MNIST**	**Synth Signs to GTSRB**
Source-only	56.6 (52.2)	86.7 (86.7)	59.2 (54.9)	85.1 (79.0)
CORAL [26]	57.7	85.2	63.1	86.9
MMD [29, 17]	76.9	88.0	71.1	91.1
DANN [8]	77.4 (76.6)	90.3 (91.0)	70.7 (73.8)	92.9 (88.6)
DSN w/ MMD (ours)	80.5	88.5	72.2	92.6
DSN w/ DANN (ours)	**83.2**	**91.2**	**82.7**	**93.1**
Target-only	98.7	92.4	99.5	99.8

Content

1. Domain Adaptation
   - Distance based
   - Adversarial based
   - Reconstruction based
2. Counterfactual Inference

What is a counterfactual problem?

Example 1: for a patient $x \in X$ the set T of interventions of interest might be two different treatments $t = 0$ *or* $t = 1$, and the set of outcomes might be $Y = [0, 200]$ indicating blood sugar levels. But for each x, we only know the result of one treatment, for example $Y_{t=0}(x)$ and need to predict $Y_{t=1}(x)$.

Example 2: For an ad slot on a webpage x, the set of interventions T might be all possible ads on the inventory, and the potential result could be $Y = \{click, \ no - click\}$. Again, for each x we only know the result for one intervention $Y_{T=t_0}(x)$, and need to predict the remaining $Y_{T=t}(x)$

Set up

- Let T be the set of potential interventions or actions we are considering,
- X the set of contexts,
- and Y the set of possible outcomes,
- in this work, they only consider the binary action set $T = 0, 1$ corresponding to control group and treated group, respectively.
- For each context $x \in X$, the outcome of one of the two actions is observed.
- We refer to the observed outcomes as the factual outcome $y^F(x)$, and counterfactual outcome $y^{CF}(x)$ respectively.

Quantity of interest

Individualized treatment effect (ITE) for context x is defined as:

$$ITE(x) = Y_1(x) - Y_0(x)$$

Average treatment effect (ATE) is defined as:

$$ATE = E_{x \sim p(x)}[ITE(x)]$$

Suppose we have n observed samples $\{(x_i, t_i, y_i^F)\}$, where $y_i^F = t_i Y_1(x_i) + (1 - t_i) Y_0(x_i)$.

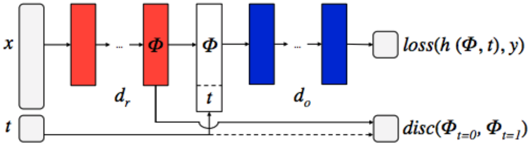
Note $\hat{P}^F = \{(x_i, t_i)\}_{i=1}^n$ and $\hat{P}^{CF} = \{(x_i, 1 - t_i)\}_{i=1}^n$

Generally, source domain $\hat{P}^F$ is different from target domain $\hat{P}^{CF}$, thus it is a special case of domain adaptation.

Model

The model contains two parts, the first part is a representation extractor $\Phi : X \rightarrow R^d$, the second part is a predictor $h : R^d \times T \rightarrow R$.
The learned representation balances three objectives:

- enable low-error prediction on factual domain (source domain).
- enable low-error prediction on unobserved counterfactual domain.
- the distribution of treatment populations are similar. (the feature distribution from two domains are similar).



- The prediction loss on factual domain(source domain) is :

$$\frac{1}{n}\sum_{i=1}^{n}|h(\phi(x_i), t_i) - y_i^F|$$

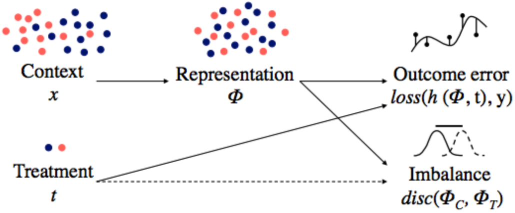
- The prediction loss on counter factual domain(target domain) can't be calculated directly, since y_i^{CF} is unknown. Let $j(i)$ be the nearest neighbor of x_i among the group that received the opposite treatment from unit i, the prediction loss on counterfactual domain is approximated as:

$$\frac{1}{n}\sum_{i=1}^{n}|h(\phi(x_i), 1 - t_i) - y_{j(i)}^F|$$

- The discrepancy distance is noted as $disc_{\mathcal{H}}$

The final loss is:

$$B_{H,\alpha,\gamma}(\phi, h) = \frac{1}{n}\sum_{i=1}^{n}|h(\phi(x_i), t_i) - y_i^F| + \frac{\gamma}{n}\sum_{i=1}^{n}|h(\phi(x_i), 1 - t_i) - y_{j(i)}^F| + \alpha disc_{\mathcal{H}}(\hat{P}^F, \hat{P}^{CF}) \tag{3}$$



Algorithm:

Algorithm 1 Balancing counterfactual regression

1: **Input:** $X, T, Y^F; \mathcal{H}, \mathcal{N}; \alpha, \gamma, \lambda$
2: $\Phi^*, g^* = \underset{\Phi \in \mathcal{N}, g \in \mathcal{H}}{\arg\min}\ B_{\mathcal{H},\alpha,\gamma}(\Phi, g) \quad (2)$
3: $h^* = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (h(\Phi, t_i) - y_i^F)^2 + \lambda \|h\|_{\mathcal{H}}$
4: **Output:** h^*, Φ^*

Error bound

Assume

$$
\begin{aligned}
\hat{\beta}^F(\phi) &= \arg\min_{\beta\in\mathcal{H}} L_{P^F_\phi}(\beta) + \lambda||\beta||_2^2 \\
\hat{\beta}^{CF}(\phi) &= \arg\min_{\beta\in\mathcal{H}} L_{P^{CF}_\phi}(\beta) + \lambda||\beta||_2^2
\end{aligned} \tag{4}
$$

under some technique assumptions, for both $Q = P^F, Q = P^{CF}$ we have:

$$
\begin{aligned}
C(L_Q(\hat{\beta}^F(\phi)) - L_Q(\hat{\beta}^{CF}(\phi)))^2 &\leqslant disc_{\mathcal{H}}(\hat{\beta}^F(\phi), \hat{\beta}^{CF}(\phi)) + \\
&\min_{h\in H} \frac{1}{n}\sum_{i=1}^{n}(|\hat{y}_i^F(\phi,h) - y_i^F| + |\hat{y}_i^{CF}(\phi,h) - y_i^{CF}|) \\
&\leqslant disc_{\mathcal{H}}(\hat{\beta}^F(\phi), \hat{\beta}^{CF}(\phi)) + \\
\min_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}(|\hat{y}_i^F(\phi,h) - y_i^F| &+ |\hat{y}_i^{CF}(\phi,h) - y_{j(i)}^F|) + \frac{c_1}{n}\sum_{i:t_i=1} d_{i,j(i)}
\end{aligned} \tag{5}
$$

Experiment results

Table 1. IHDP. Results and standard errors for 1000 repeated experiments. (Lower is better.) Proposed methods: BLR, BNN-4-0 and BNN-2-2. † (Chipman et al., 2010)

	ϵ_{ITE}	ϵ_{ATE}	PEHE
LINEAR OUTCOME			
OLS	4.6 ± 0.2	0.7 ± 0.0	5.8 ± 0.3
DOUBLY ROBUST	3.0 ± 0.1	0.2 ± 0.0	5.7 ± 0.3
LASSO + RIDGE	2.8 ± 0.1	0.2 ± 0.0	5.7 ± 0.2
BLR	2.8 ± 0.1	0.2 ± 0.0	5.7 ± 0.3
BNN-4-0	3.0 ± 0.0	0.3 ± 0.0	5.6 ± 0.3
NON-LINEAR OUTCOME			
NN-4	2.0 ± 0.0	0.5 ± 0.0	1.9 ± 0.1
BART†	2.1 ± 0.2	0.2 ± 0.0	1.7 ± 0.2
BNN-2-2	$\mathbf{1.7 \pm 0.0}$	0.3 ± 0.0	$\mathbf{1.6 \pm 0.1}$

Table 2. News. Results and standard errors for 50 repeated experiments. (Lower is better.) Proposed methods: BLR, BNN-4-0 and BNN-2-2. † (Chipman et al., 2010)

	ϵ_{ITE}	ϵ_{ATE}	PEHE
LINEAR OUTCOME			
OLS	3.1 ± 0.2	0.2 ± 0.0	3.3 ± 0.2
DOUBLY ROBUST	3.1 ± 0.2	0.2 ± 0.0	3.3 ± 0.2
LASSO + RIDGE	2.2 ± 0.1	0.6 ± 0.0	3.4 ± 0.2
BLR	2.2 ± 0.1	0.6 ± 0.0	3.3 ± 0.2
BNN-4-0	2.1 ± 0.0	0.3 ± 0.0	3.4 ± 0.2
NON-LINEAR OUTCOME			
NN-4	2.8 ± 0.0	1.1 ± 0.0	3.8 ± 0.2
BART†	5.8 ± 0.2	0.2 ± 0.0	3.2 ± 0.2
BNN-2-2	$\mathbf{2.0 \pm 0.0}$	0.3 ± 0.0	$\mathbf{2.0 \pm 0.1}$