Transferable Adversarial Training:A General Approach to Adapting Deep Classifiers (2019)

Hong Liu, Mingsheng Long, Jianmin Wangn, Michael I. Jordan

Summary

Contributions

One of the main problems of Adversarial domain adaptation (DANN), is that we enforce a very strong assumption over the features extractor, so that we will have domain invariant representations z, in this case $p(z)$ is independent of the domain, but the true labels $\hat{y}$ might be, and this will reduce the transferability of z since we suppress domain specific details, and have lower performance on target domain (high risk). This can also be see in traditional bound of the target risk bellow.

$$\epsilon_t(C) \leq \epsilon_s(C) + d_{\mathcal{H}\Delta\mathcal{H}}(X_s, X_t) + \lambda$$

But training the classifier on the source domain, we reduce the source risk $\epsilon_s(C)$, and by using a discriminator over domain, we push the feature extractor to produce invariance representations. So we reduce the discrepancy between both domains $d_{\mathcal{H}\Delta\mathcal{H}}(X_s, X_t)$. But the last term, the transferability term, where we see is the optimal classifier is good on both domains is ignored, and in DANN, λ increases since we suppress domain specific information.

The authors propose to make the task of the feature extractor a little bit easier, by first taking its outputs, applying a series of updates, to align the features between domains, and then feeding them to the discriminator, to in this case we help the feature extractor by creating new and transferable features at each training iteration.

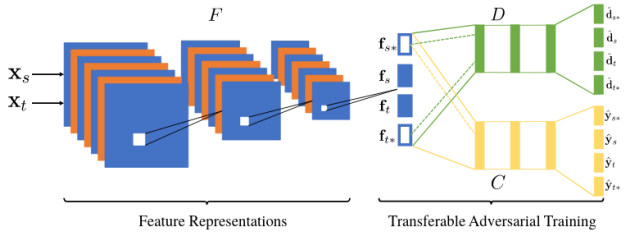


Figure 3. Transferable Adversarial Training (TAT) for adapting deep classifiers. The feature extractor F yields representations $\mathbf{f}_s$ and $\mathbf{f}_t$ of the source and target data, which are *fixed* in the training process to guarantee adaptability λ. The dashed lines indicate the adversarial generation of transferable examples $\mathbf{f}_{s*}$ and $\mathbf{f}_{t*}$ through maximizing the errors of the category classifier C and domain discriminator D. We adversarially train the classifiers with transferable examples: C to minimize the source error and D to distinguish source from target.

Method

In unsupervised domain adaptation, we have observations $\{\mathbf{x}_s^{(i)}, \mathbf{y}_s^{(i)}\}_{i=1}^{n_x}$ from a source domain of distribution $P(\mathbf{x}_s, \mathbf{y}_s)$, and observations $\{\mathbf{x}_t^{(i)}\}_{i=1}^{n_t}$ from a target domain of distribution $Q(\mathbf{x}_t, \mathbf{y}_t)$, with a distribution shift $P \neq Q$. The goal if to adapt the classifier $\mathbf{y} = C(\mathbf{f})$ on the target domain under a feature extractor $\mathbf{f} = F(\mathbf{x})$.

Normally we train the classifier on the source labeled examples, and the feature extractor and and discriminator D on target and source in a minimax game.

$$\ell_d(\theta_D, \mathbf{f}) = -\frac{1}{n_s}\sum_{i=1}^{n_s}\log[D(\mathbf{f}_s^{(i)})] - \frac{1}{n_t}\sum_{i=1}^{n_t}\log[1 - D(\mathbf{f}_t^{(i)})]$$

$$\ell_c(\theta_C, \mathbf{f}) = \frac{1}{n_s} \sum_{i=1}^{n_s} \ell_{ce}(C(\mathbf{f}_s^{(i)}), \mathbf{y}_s^{(i)})$$

Now instead of feeding the discriminator the original features $\mathbf{f}$, the authors propose to first create transferable features $\mathbf{f}_*$, which are created by updating the outputs of the features extractor for K steps as follows

$$\mathbf{f}_{t^{k+1}} arrow \mathbf{f}_{t^k} + \beta \nabla_{\mathbf{f}_{t^k}} \ell_d(\theta_D, \mathbf{f}_{t^k}) - \gamma \nabla_{\mathbf{f}_{t^k}} \ell_2(\mathbf{f}_{t^k}, \mathbf{f}_{t^0})$$
$$\mathbf{f}_{s^{k+1}} arrow \mathbf{f}_{s^k} + \beta \nabla_{\mathbf{f}_{s^k}} \ell_d(\theta_D, \mathbf{f}_{s^k}) - \gamma \nabla \mathbf{f}_{k^k} \ell_2(\mathbf{f}_{s^k}, \mathbf{f}_{s^0}) + \beta \nabla_{\mathbf{f}_{s^k}} \ell_c(\theta_C, \mathbf{f}_{s^k})$$

and are then fed into the discriminator

$$\ell_{d,adv}(\theta_D, \mathbf{f}_*) = -\frac{1}{n_s} \sum_{i=1}^{n_s} \log[D(\mathbf{f}_{s*}^{(i)})] - \frac{1}{n_t} \sum_{i=1}^{n_t} \log[1 - D(\mathbf{f}_{t*}^{(i)})]$$

We can also see the updated features $\mathbf{f}_*$ as perturbed features, and force a consistency over the classifier's outputs by adding a consistency to get the same classification using perturbed and original features.

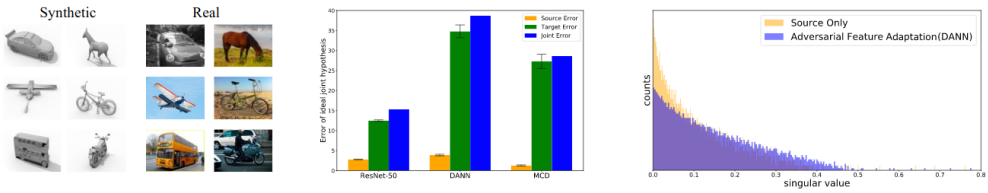
$$\ell_{c,adv}(\theta_C, \mathbf{f}_*) = \frac{1}{n_s} \sum_{i=1}^{n_s} \ell_{ce}(C(\mathbf{f}_{s*}^{(i)}), \mathbf{y}_{s*}^{(i)}) + \frac{1}{n_t} \sum_{i=1}^{n_t} |C((\mathbf{f}_{t*}^{(i)})) - C((\mathbf{f}_t^{(i)}))$$

The proposed algorithm can be summarized bellow:

Algorithm 1 Transferable Adversarial Training (TAT)

Input: original features $\{\mathbf{f}_s^{(i)}, \mathbf{y}_s^{(i)}\}_{i=1}^{n_s}$ and $\{\mathbf{f}_t^{(i)}\}_{i=1}^{n_t}$.
Output: learned model parameters $\theta = (\theta_C, \theta_D)$.
Initialize $\theta = (\theta_C, \theta_D)$ randomly.
for `iter` $= 1$ **to** `MaxIter` **do**
  Sample a mini-batch of $\{(\mathbf{f}_s, \mathbf{y}_s)\}$ and $\{\mathbf{f}_t\}$ uniformly from the training dataset in terms of original features.
  for $k = 0$ **to** $K - 1$ **do**
    $\mathbf{f}_{s^{k+1}} \leftarrow \mathbf{f}_{s^k} + \beta \nabla_{\mathbf{f}_{s^k}} \ell_d(\theta_D, \mathbf{f}_{s^k}) - \gamma \nabla_{\mathbf{f}_{s^k}} \ell_2(\mathbf{f}_{s^k}, \mathbf{f}_{s^0}) + \beta \nabla_{\mathbf{f}_{s^k}} \ell_c(\theta_C, \mathbf{f}_{s^k})$
    $\mathbf{f}_{t^{k+1}} \leftarrow \mathbf{f}_{t^k} + \beta \nabla_{\mathbf{f}_{t^k}} \ell_d(\theta_D, \mathbf{f}_{t^k}) - \gamma \nabla_{\mathbf{f}_{t^k}} \ell_2(\mathbf{f}_{t^k}, \mathbf{f}_{t^0})$
  end for
  $\theta_C \leftarrow \theta_C - \alpha \nabla_{\theta_C} [\ell_c(\theta_C, \mathbf{f}) + \ell_{c,adv}(\theta_C, \mathbf{f}_*))]$
  $\theta_D \leftarrow \theta_D - \alpha \nabla_{\theta_D} [\ell_d(\theta_D, \mathbf{f}) + \ell_{d,adv}(\theta_D, \mathbf{f}_*))]$
end for

Results



(a) VisDA-2017 (b) Test Error (c) Singular Values of Weight Matrices

Figure 2. Motivation of our approach. (a) Learning from synthetic to real data on VisDA-2017 dataset (Peng et al., 2017). (b) The error of ideal joint hypothesis. (c) The distributions of the singular values of the deep classifier's adapted weight matrices. *Best viewed in color.*

Table 1. Classification accuracies (%) on Office-31 for unsupervised domain adaptation with ResNet-50.

METHOD	A→W	D→W	W→D	A→D	D→A	W→A	AVG.
RESNET-50 (HE ET AL., 2016)	68.4±0.2	96.7±0.1	99.3±0.1	68.9±0.2	62.5±0.3	60.7±0.3	76.1
DAN (LONG ET AL., 2015)	80.5±0.4	97.1±0.2	99.6±0.1	78.6±0.2	63.6±0.3	62.8±0.2	80.4
DANN (GANIN ET AL., 2016)	82.6±0.4	96.9±0.2	99.3±0.2	81.5±0.4	68.4±0.5	67.5±0.5	82.7
ADDA (TZENG ET AL., 2017)	86.2±0.5	96.2±0.3	98.4±0.3	77.8±0.3	69.5±0.4	68.9±0.5	82.9
VADA (SHU ET AL., 2018)	86.5±0.5	98.2±0.4	99.7±0.2	86.7±0.4	70.1±0.4	70.5±0.4	85.4
GTA (SANKARANARAYANAN ET AL., 2018)	89.5±0.5	97.9±0.3	99.7±0.2	87.7±0.5	72.8±0.3	71.4±0.4	86.5
MCD (SAITO ET AL., 2018)	88.6±0.2	98.5±0.1	**100.0**±.0	92.2±0.2	69.5±0.1	69.7±0.3	86.5
CDAN (LONG ET AL., 2018)	**93.1**±0.1	98.6±0.1	**100.0**±.0	92.9±0.2	71.0±0.3	69.3±0.3	87.5
TAT	92.5±0.3	**99.3**±0.1	**100.0**±.0	**93.2**±0.2	**73.1**±0.3	**72.1**±0.3	**88.4**

Table 2. Classification accuracies (%) on Image-CLEF for unsupervised domain adaptation with ResNet-50.

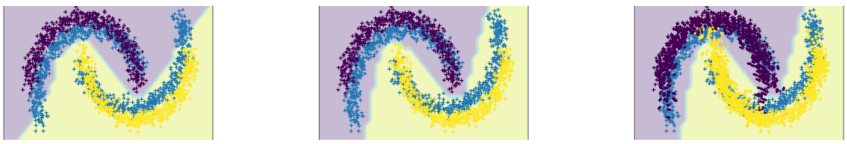
METHOD	I→P	P→I	I→C	C→I	C→P	P→C	AVG.
RESNET-50 (HE ET AL., 2016)	74.8±0.3	83.9±0.1	91.5±0.3	78.0±0.2	65.5±0.3	91.2±0.3	80.7
DAN (LONG ET AL., 2015)	74.5±0.4	82.2±0.2	92.8±0.2	86.3±0.4	69.2±0.4	89.8±0.4	82.5
DANN (GANIN ET AL., 2016)	75.0±0.3	86.0±0.3	96.2±0.4	87.0±0.5	74.3±0.5	91.5±0.6	85.0
CDAN (LONG ET AL., 2018)	76.7±0.3	90.6±0.3	97.0±0.4	90.5±0.4	74.5±0.3	93.5±0.4	87.1
TAT	**78.8**±0.2	**92.0**±0.2	**97.5**±0.3	**92.0**±0.3	**78.2**±0.4	**94.7**±0.4	**88.9**

Table 3. Classification accuracies (%) on Office-Home for unsupervised domain adaptation (ResNet-50).

METHOD	AR→CL	AR→PR	AR→RW	CL→AR	CL→PR	CL→RW	PR→AR	PR→CL	PR→RW	RW→AR	RW→CL	RW→PR	AVG.
RESNET-50 (HE ET AL., 2016)	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DAN (LONG ET AL., 2015)	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DANN (GANIN ET AL., 2016)	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
CDAN (LONG ET AL., 2018)	49.0	69.3	74.5	54.4	66.0	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8
TAT	**51.6**	**69.5**	**75.4**	**59.4**	**69.5**	**68.6**	**59.5**	**50.5**	**76.8**	**70.9**	**56.6**	**81.6**	**65.8**

Table 4. Classification accuracies (%) on Multi-Domain Sentiment Dataset for unsupervised domain adaptation.

METHOD	B→D	B→E	B→K	D→B	D→E	D→K	E→B	E→D	E→K	K→B	K→D	K→E	AVG.
SVM (BOW) (CHANG & LIN, 2011)	79.9	74.8	76.9	74.3	74.8	74.6	70.5	72.6	84.7	70.7	73.6	84.2	76.0
DANN (BOW) (GANIN ET AL., 2016)	78.4	73.3	77.9	72.3	75.4	78.3	71.3	73.8	85.4	70.9	74.0	84.3	76.3
MSDA (BOW) (CHEN ET AL., 2012)	83.5	74.5	**84.6**	**83.6**	79.5	**87.1**	78.8	**80.9**	85.3	**80.2**	**82.3**	86.9	82.3
TAT (BOW)	**84.5**	**80.1**	83.6	81.9	**81.9**	84.0	**83.2**	77.9	**90.0**	75.8	77.7	**88.2**	**82.4**
DANN (MSDA) (GANIN ET AL., 2016)	82.9	80.4	84.3	82.5	80.9	84.9	77.4	78.1	88.1	71.8	78.9	85.6	81.3
TDA (MSDA) (LONG ET AL., 2016)	84.1	85.0	87.5	84.9	85.7	88.6	82.0	82.7	87.7	**81.5**	83.3	86.8	85.0
TAT (MSDA)	**86.8**	**85.9**	**88.6**	**86.4**	**86.4**	**89.4**	**83.7**	**83.5**	**90.4**	81.4	**84.7**	**89.2**	**86.3**



(a) Source Only Model (b) TAT (c) Transferable Examples

Figure 4. Behaviors on the *two moon* problem. Purple and yellow "+"s indicate source samples, blue "+"s are target samples, while dots are transferable examples. (a) The source only model. (b) The decision boundary of TAT. (c) The distribution of the transferable examples.

Table 6. Classification accuracies (%) of the TAT variants on Office-31 for unsupervised domain adaptation (ResNet-50).

METHOD	A→W	D→W	W→D	A→D	D→A	W→A	AVG.
TAT (W\O C)	87.2±0.2	98.3±0.2	99.9±0.1	88.4±0.2	71.0±0.1	69.8±0.1	85.8
TAT (W\O D)	88.5±0.3	98.7±0.2	100.0±.0	90.6±0.2	71.4±0.2	72.0±0.1	86.8
TAT	**92.5**±0.3	**99.3**±0.1	**100.0**±.0	**93.2**±0.2	**73.1**±0.3	**72.1**±0.3	**88.4**