Domain Adversarial Neural Nets

Presenter: Shafiq Joty

School of Computer Science and Engineering
Nanyang Technological University

May 8, 2018

Outline

Adversarial Neural Nets

Adversarial nets have been used in **two ways**:

1. **Generative modeling**: GANs [Goodfellow et al., 2014].
   - Generate real-like image samples from random vectors
   - One neural net is pit against another:
     (*i*) Generator, (*ii*) Discriminator
   - Other Examples: image-to-image translation, Unsup. NMT

Adversarial Neural Nets

Adversarial nets have been used in **two ways**:

1. **Generative modeling**: GANs [Goodfellow et al., 2014].
    - Generate real-like image samples from random vectors
    - One neural net is pit against another:
      (*i*) Generator, (*ii*) Discriminator
    - Other Examples: image-to-image translation, Unsup. NMT
2. **Transfer learning/domain adaptation**:
   DANNs [Ganin et al., 2016]
    - Map samples from two domains into a common feature space
    - Generally a **three-player** game:
      (*i*) Encoder, (*ii*) Classifier (*iii*) Discriminator
    - Examples: MNIST $\Rightarrow$ USPS, X-lingual NER

Unsupervised Domain Adaptation

Training Data:

$$\mathcal{D}_s = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \quad (1)$$
$$\mathcal{D}_t = \{\mathbf{x}_i\}_{i=1}^{M} \quad (2)$$

- $\mathcal{D}_s$ denotes source (labeled) dataset.
- $\mathcal{D}_t$ denotes target (unlabeled) dataset.
- Distribution (domain) shift from $\mathcal{D}_s$ to $\mathcal{D}_t$

Examples

NLP Sentiment analysis (Movie $\Rightarrow$ Book), NER (Eng $\Rightarrow$ Ger)
CV Image classification: MNIST $\Rightarrow$ USPS

DANN Approach: Learn a common feature space for $\mathcal{D}_s$ and $\mathcal{D}_t$

DANN Framework

Three players

(i) Encoder; (ii) Classifier; (iii) Discriminator

DANN Framework

Three players

(i) Encoder; (ii) Classifier; (iii) Discriminator

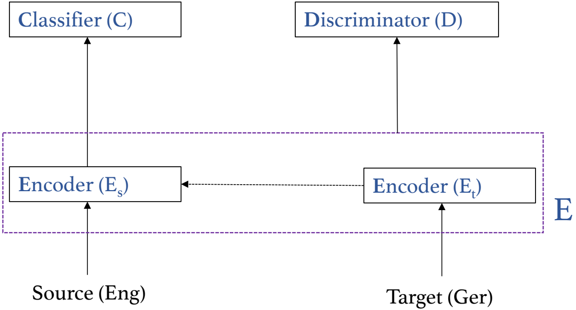


Figure: DANN Framework

DANN Design Choices

Three players

(i) Encoder; (ii) Classifier; (iii) Discriminator

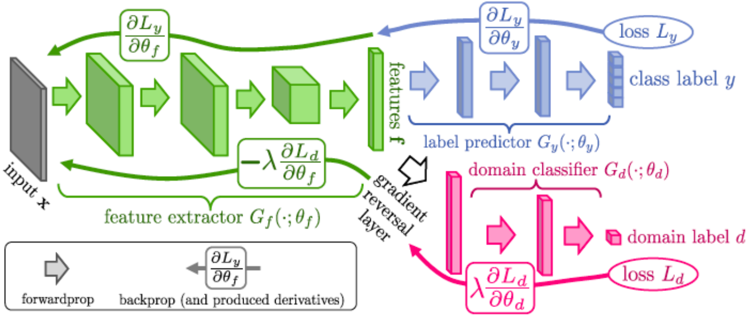
Encoder

- Shared [Ganin et al., 2016] vs. Separate [Tzeng et al., 2017]
- Train Encoders concurrently or in steps.
- Generative [Volpi et al., 2017] vs. Discriminative [Tzeng et al., 2017]

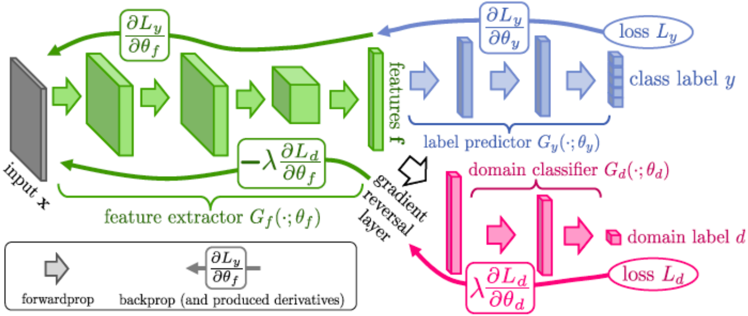
Adversary Loss

- Flip gradient [Ganin et al., 2016]
- GAN loss [Tzeng et al., 2017]

DANN Design Choices



DANN Design Choices



Choices:

- Shared **encoder**. Train source/target concurrently.
- **Discriminatively** trained source.

DANN Adversary

- **Discriminator:**
  $-\mathbb{E}_{x_s \sim X_s} \log D\left(E(x_s)\right) - \mathbb{E}_{x_t \sim X_t} \log\left(1 - D\left(E(x_t)\right)\right).$

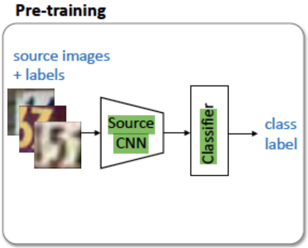
DANN Adversary

- **Discriminator:**
  $-\mathbb{E}_{x_s \sim X_s} \log D\left(E(x_s)\right) - \mathbb{E}_{x_t \sim X_t} \log\left(1 - D\left(E(x_t)\right)\right).$

- **Flip Gradient Adversary** (to Encoder):
  $\Rightarrow$ **Encoder (for source):** $\nabla_{\theta_e} \mathbb{E}_{x_s \sim X_s} \log D\left(E(x_s)\right)$
  $\Rightarrow$ **Encoder (for target):** $\nabla_{\theta_e} \mathbb{E}_{x_t \sim X_t} \log\left(1 - D\left(E(x_t)\right)\right)$

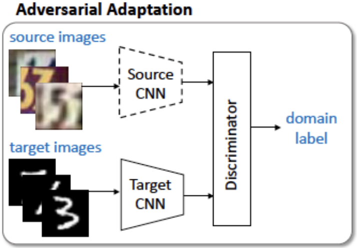
ADDA [Tzeng et al., 2017]

Step 1: Pre-training on Source



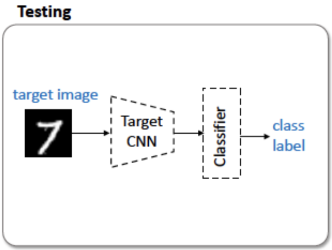
ADDA [Tzeng et al., 2017]

Step 2: Adapt Target towards (fixed) Source



ADDA [Tzeng et al., 2017]

Step 3: Testing



ADDA Design Choices

ADDA Design Choices:

- Separate **encoder** for source and target.
- Train source/target separately.
- **Discriminatively** trained source.
- GAN loss for **adversary**.

ADDA Adversary

- **Discriminator:**
  $-\mathbb{E}_{x_s \sim X_s} \log D\left(E(x_s)\right) - \mathbb{E}_{x_t \sim X_t} \log\left(1 - D\left(E(x_t)\right)\right).$
- **GAN Adversary**:
  $\Rightarrow$ **Encoder (for target):** - $\nabla_{\theta_e} \mathbb{E}_{x_t \sim X_t} \log\left(D\left(E(x_t)\right)\right)$

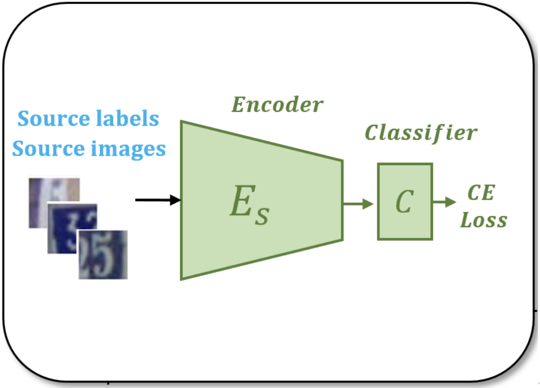
ADDA Adversary

- **Discriminator:**
  $-\mathbb{E}_{x_s \sim X_s} \log D(E(x_s)) - \mathbb{E}_{x_t \sim X_t} \log(1 - D(E(x_t)))$.
- **GAN Adversary**:
  $\Rightarrow$ **Encoder (for target):** - $\nabla_{\theta_e} \mathbb{E}_{x_t \sim X_t} \log(D(E(x_t)))$
- Recall Adversary for Flip Gradient [Ganin et al., 2016]
  $\Rightarrow$ **Encoder (for source):** $\nabla_{\theta_e} \mathbb{E}_{x_s \sim X_s} \log D(E(x_s))$
  $\Rightarrow$ **Encoder (for target):** $\nabla_{\theta_e} \mathbb{E}_{x_t \sim X_t} \log(1 - D(E(x_t)))$

Feature Augmented DANN [Volpi et al., 2017]

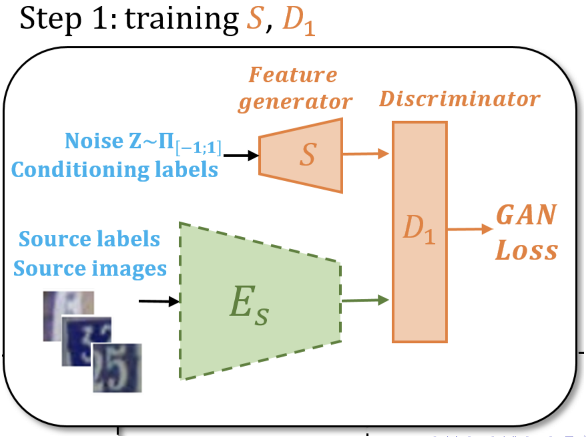
Step 0: Pre-training on Source

Step 0: training E_s, C



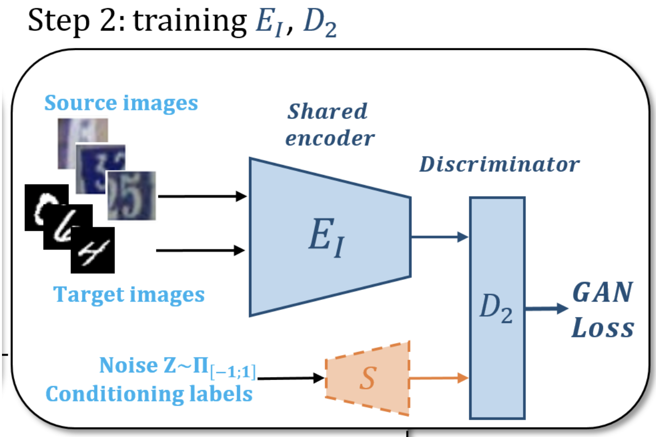
Feature Augmented DANN [Volpi et al., 2017]

Step 1: Feature Generation with CGAN



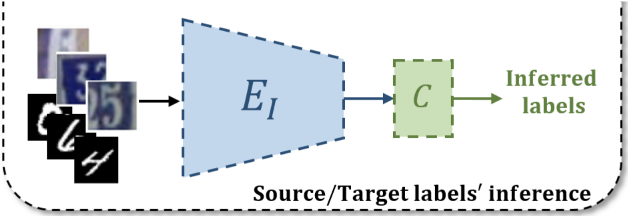
Feature Augmented DANN [Volpi et al., 2017]

Step 2: Feature Adaptation



Feature Augmented DANN [Volpi et al., 2017]

Step 3: Testing



FA-DANN Design Choices

FA-DANN Design Choices:

- Same shared **encoder** for source and target.
- Train source/target separately.
- **Discriminatively** trained source + **Generative** features.
- GAN loss for **adversary**.

Results

- ADDA [Tzeng et al., 2017]

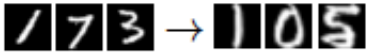
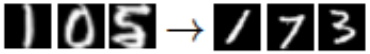
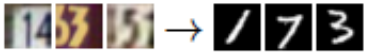
Method	MNIST → USPS 	USPS → MNIST 	SVHN → MNIST 
Source only	0.752 ± 0.016	0.571 ± 0.017	0.601 ± 0.011
Gradient reversal	0.771 ± 0.018	0.730 ± 0.020	0.739 [19]
Domain confusion	0.791 ± 0.005	0.665 ± 0.033	0.681 ± 0.003
CoGAN	0.912 ± 0.008	0.891 ± 0.008	did not converge
ADDA (Ours)	0.894 ± 0.002	0.901 ± 0.008	0.760 ± 0.018

Table 2: Experimental results on unsupervised adaptation among MNIST, USPS, and SVHN.

Results

- FA-DANN [Volpi et al., 2017]

	SVHN→MNIST	MNIST→USPS$_{P1}$	MNIST→USPS$_{P2}$	USPS→MNIST	SYN→SVHN	NYUD
Source	0.682	0.723	0.797	0.627	0.885	0.139
DANN [9, 10]	0.739	0.771 ± 0.018 [35]	-	0.730 ± 0.020 [35]	0.911	-
DDC [35]	0.681 ± 0.003	0.791 ± 0.005	-	0.665 ± 0.033		-
DSN [3]	0.827	-	-	-	0.912	-
ADDA [35]	0.760 ± 0.018	0.894 ± 0.002	-	**0.901 ± 0.008**		0.211
Tri [29]	0.862	-	-	-	**0.931**	-
DTN [33]	0.844*	-	-	-	-	-
PixelDA** [2]	-	-	0.959	-	-	-
UNIT [18]	**0.905***	-	0.960	-	-	-
CoGANs [19]	no conv. [35]	0.912 ± 0.008	0.957 [18]	0.891 ± 0.008	-	-
LS-ADDA	0.743 ± 0.028	0.914 ± 0.000	0.912 ± 0.003	**0.910 ± 0.004**	0.908 ± 0.004	no conv.
Ours (DI)	0.851 ± 0.026	0.914 ± 0.000	0.954 ± 0.002	0.879 ± 0.005	0.925 ± 0.002	0.287 ± 0.002
Ours (DIFA)	**0.897 ± 0.020**	**0.923 ± 0.001**	0.962 ± 0.002	0.897 ± 0.005	**0.930 ± 0.002**	**0.313 ± 0.002**
Target	0.992	0.999	0.999	0.975	0.913	0.468 [35]

References I

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016).
Domain-adversarial training of neural networks.
Journal of MLR, 17(59):1–35.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014).
Generative adversarial nets.
In *Advances in neural information processing systems*, pages 2672–2680.

References II

Tzeng, E., Hoffman, J., Darrell, T., and Saenko, K. (2017).
Adversarial discriminative domain adaptation.
In *Computer Vision and Pattern Recognition (CVPR)*.

Volpi, R., Morerio, P., Savarese, S., and Murino, V. (2017).
Adversarial feature augmentation for unsupervised domain adaptation.
CoRR, abs/1711.08561.