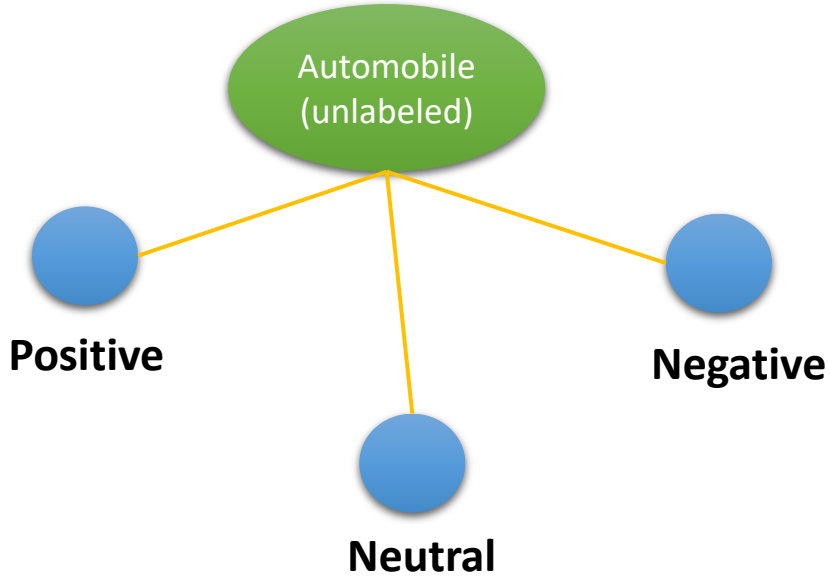
Learning Transferable Feature Representations Using Neural Networks

Himanshu, Shourya, Arun and Sriranjani

ACL'19

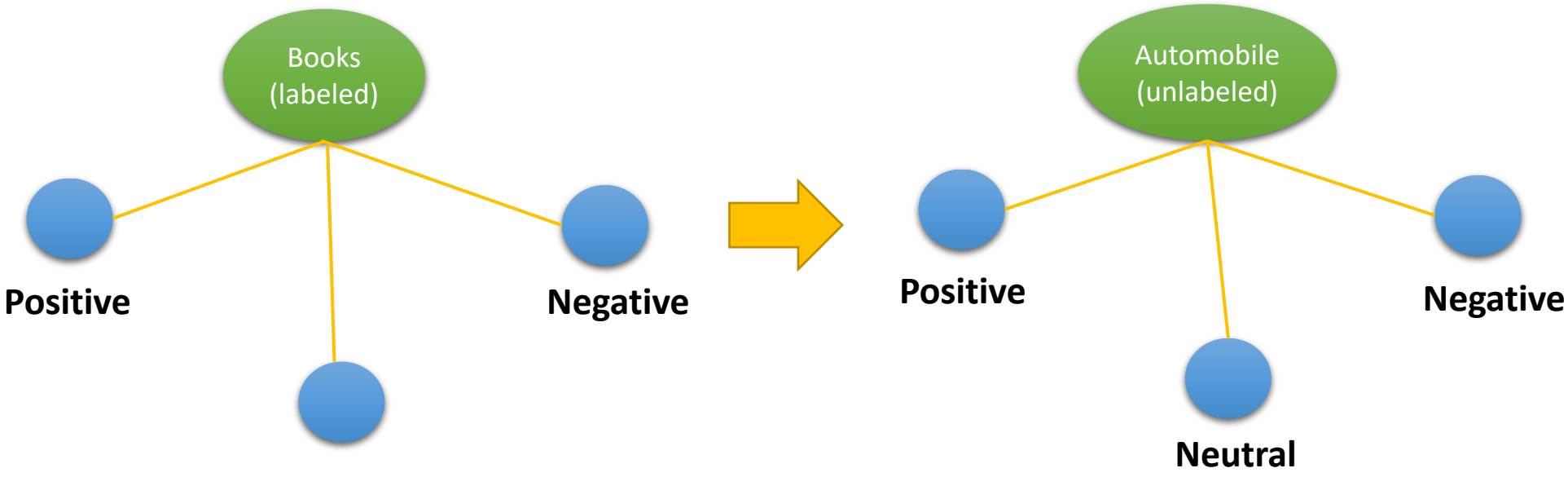
Problem

- How to build a ML model in an unlabeled domain?
- Sentiment analysis for reviews in an unlabeled domain (e.g., automobile)
- Label generation cost for a new ML task is a **BIG OBSTACLE**.
- Solution 1: Generate synthetic data
- *Example: Train a image classifier on synthetic or semi-synthetic images*

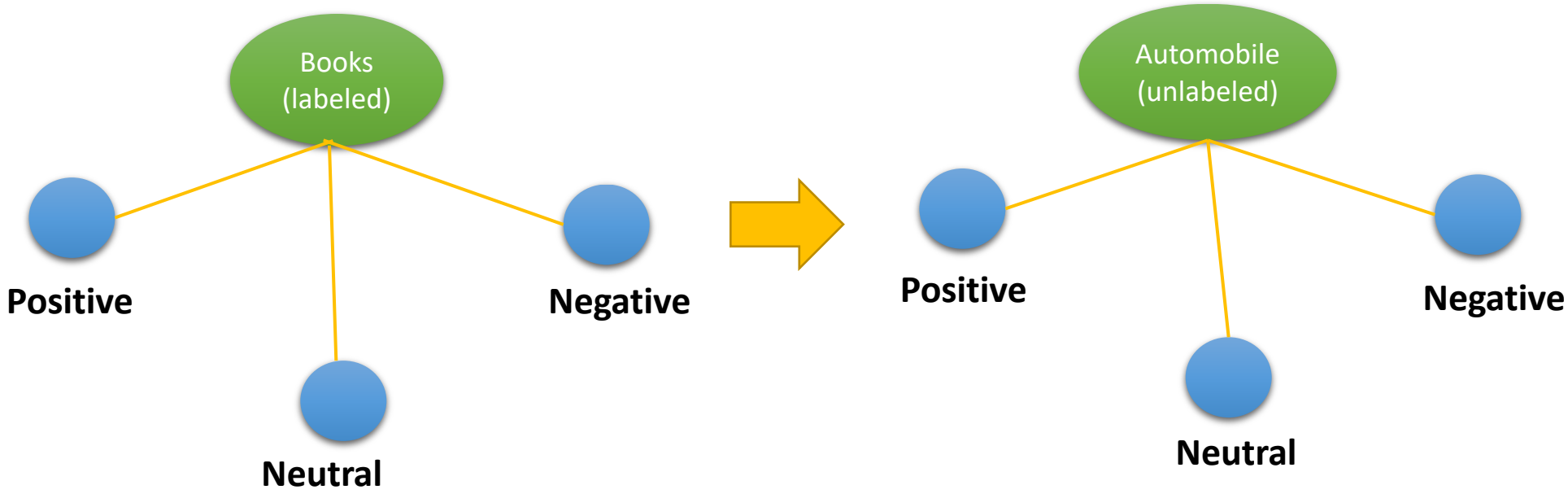


Problem

- Solution 2: Leverage labelled data from relevant domains
- *Example: Sentiment classifier for reviews of one product (e.g., automobile) from that of other products (e.g., books)*



Challenge



- **Shift in data distribution** from the actual data encountered at 'test time' <= violates i.i.d.
- Given: Labeled examples from books (source) domain, Unlabeled examples from Automobile (target) domain
- Task: Build a discriminative classifier in Automobile (target) domain
- We will call this problem as **Domain Adaptation (DA)**

Formalizing DA

Consider classification task:

- $\boldsymbol{X}$ is the input space (ex: TF-IDF vector for a movie review)
- $\boldsymbol{Y} = \{0, 1\}$ is the set of possible labels (ex: positive or negative sentiment)
- Source domain D_S defined over $\boldsymbol{X} \times \boldsymbol{Y}$ (ex: set of reviews from books domain)
- Target domain D_T defined over $\boldsymbol{X} \times \boldsymbol{Y}$ (ex: set of reviews from automobile domain)
- Labeled samples from Source S drawn i.i.d from D_S $S = \{(x_i^s, y_i^s)\}_{i=1}^{m} \sim (D_S)^m;$
- Unlabeled samples from Target T drawn i.i.d from D_T $T = \{x_i^t\}_{i=1}^{m'} \sim (D_T)^{m'}$

Goal: Build a classifier η: $\boldsymbol{X} \rightarrow \boldsymbol{Y}$ with a low target risk while having no information about the labels of D_T :

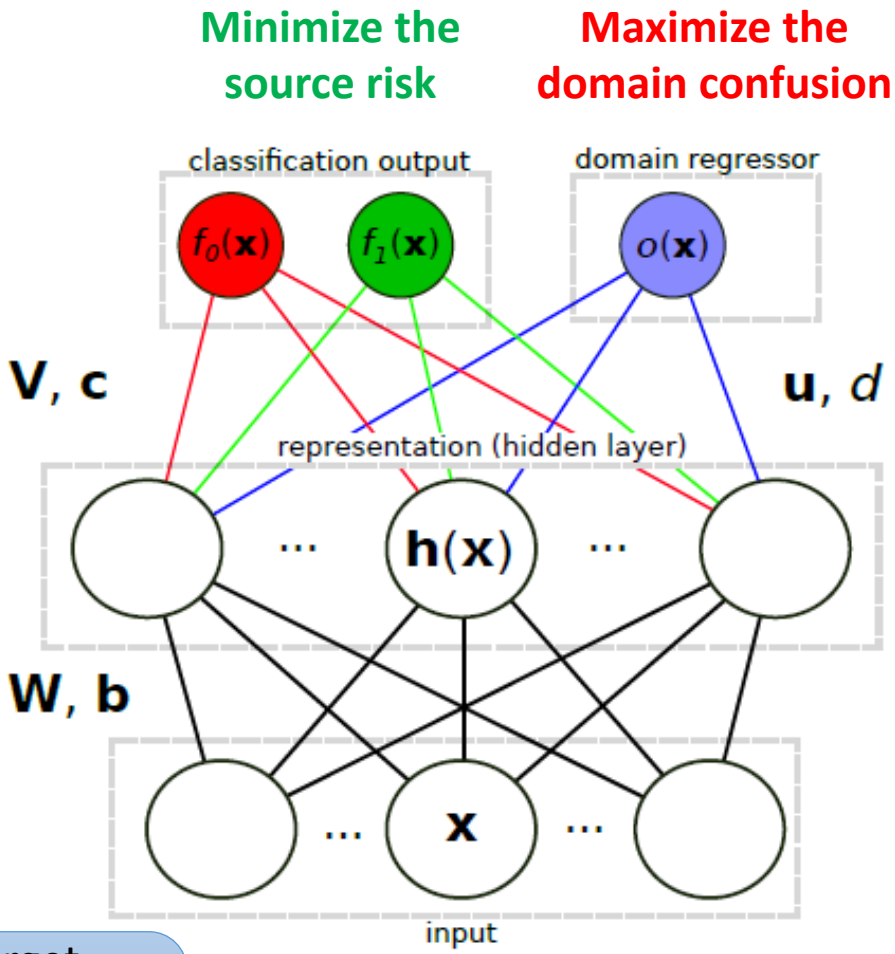
$$R_{\mathcal{D}_\mathrm{T}}(\eta) = \Pr_{(\mathbf{x},y)\sim\mathcal{D}_\mathrm{T}}\left(\eta(\mathbf{x}) \neq y\right)$$

Domain Adversarial Neural Networks (Ganin et al., JMLR'16)

- Use **deep learning** techniques to learn representations that are transferable across domains.

Learns features that combine:

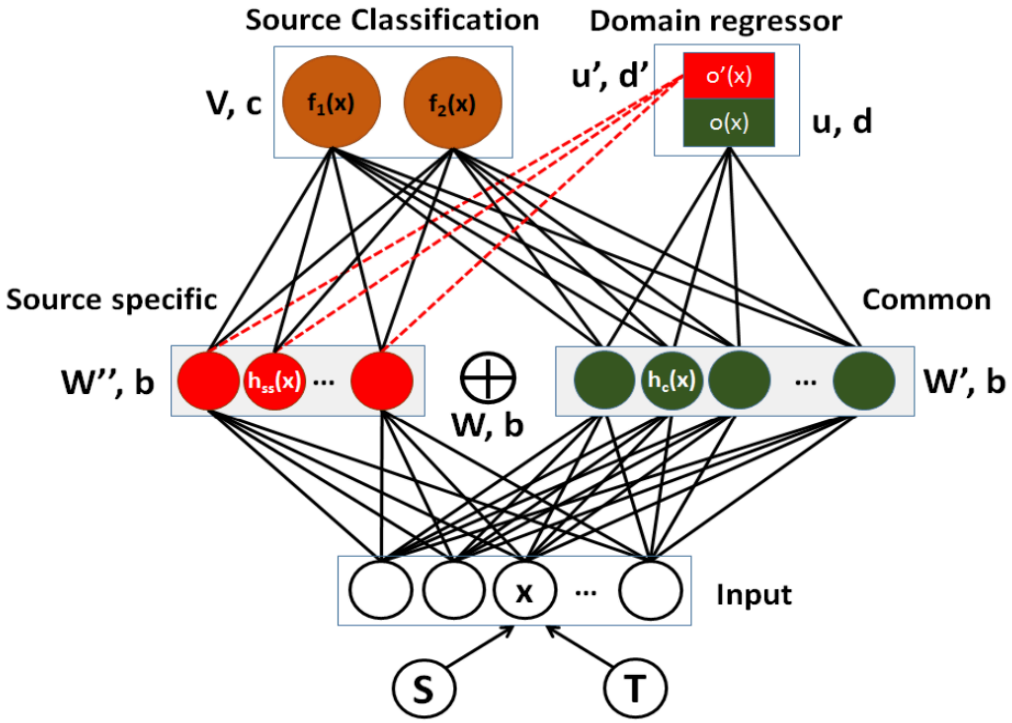
- **Domain-invariance:**
  - the domain regressor that discriminates between the source and the target domains during training.
  - makes sure the internal representation of neural network contains **no discriminative information about the origin of the input** (source or target).
- **Discriminativeness:**
  - the label predictor that predicts class labels during both training and testing
  - preserves low error rate on classification of source samples.



Intuition: When the source and target domain distributions are made similar, the source accuracy can correspond to target accuracy

Idea

- Example (books → automobile) :
  - Common features: word features such as ‘good’, ‘bad’ that are domain-invariant
  - Source specific features: word features like ‘unpredictable’
- **Goal:** Learn a common shared representation from the labeled source and unlabeled target domain samples while explicitly keeping away the source specific characteristics
- **Effect:** Reduces the impact of ‘negative transfer’

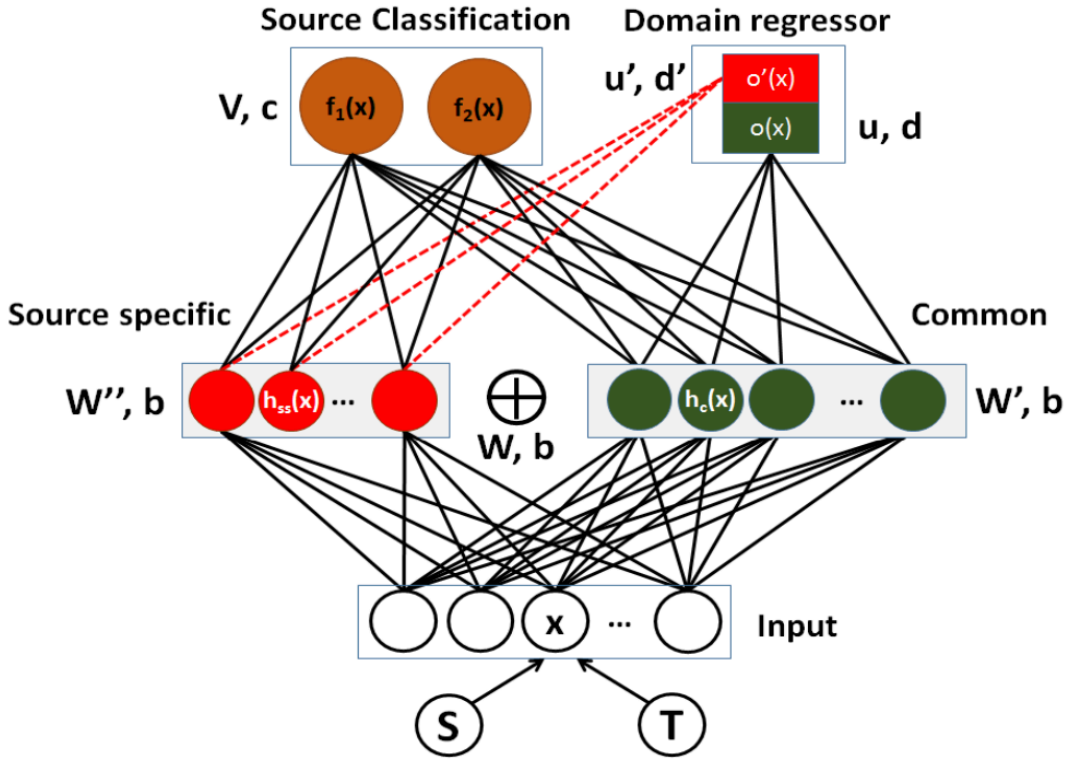


Source Classification

- Input: source instance, **x**
- Output: source class label, **y**

NN Equations:

- $h(x) = sigm(Wx+b)$
- $h(x) = \{h_{ss}(x) \oplus h_c(x)\}$
- $f(x) = softmax(Vh(x) + (c))$



- Objective function:

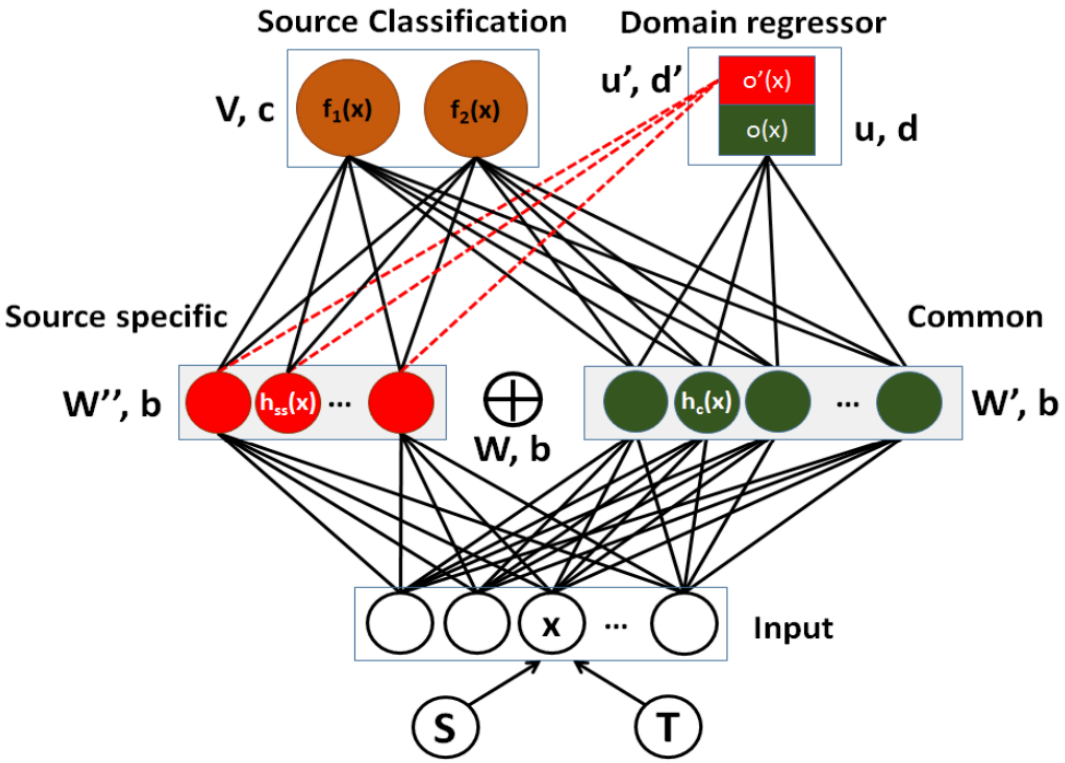
$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \left[\frac{1}{m} \sum_{i=1}^{m} \ell(f(x_i^s), y_i^s) + \lambda R(\mathbf{W}, \mathbf{b}) \right]$$

Domain Divergence

- Adapting to a target domain from a source domain depends on a measure of similarity between the two.
- H-divergence defines the capacity of a hypothesis class H to distinguish between examples generated by a pair of source-target tasks.

$$d_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) = 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_s} \left[\eta(\mathbf{x}^s) = 1\right] - \Pr_{\mathbf{x}^t \sim \mathcal{D}_t} \left[\eta(\mathbf{x}^t) = 1\right] \right|$$

- Higher the similarity between the tasks, less is H-divergence between them and vice-versa.



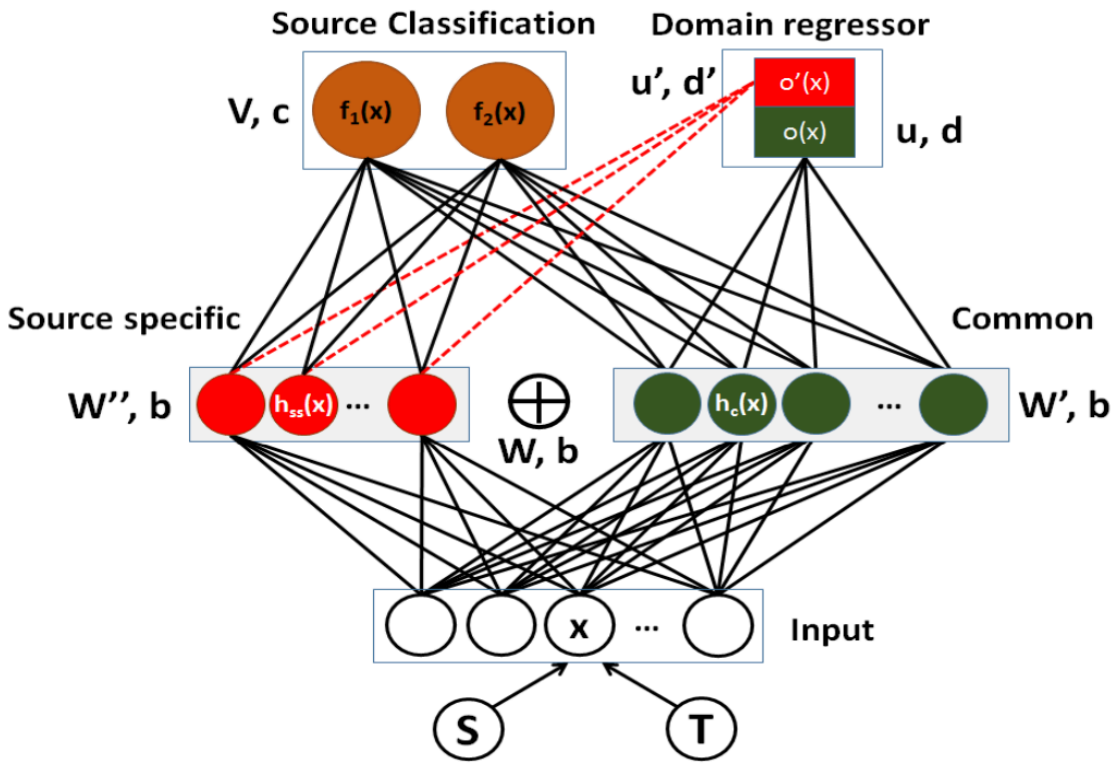
Domain Divergence

- Running over all hypothesis is hard.
- Approximate by a learning algorithm that discriminates between source and target examples.

$$\{(\mathbf{x}_i^s, 1)\}_{i=1}^m \cup \{(\mathbf{x}_j^t, 0)\}_{j=1}^{m'}$$

- Error of the classifier trained on the above dataset can be used as an approx. of H-divergence terms as Proxy A Distance (PAD)

$$\hat{d}_A = 2(1 - 2\epsilon)$$



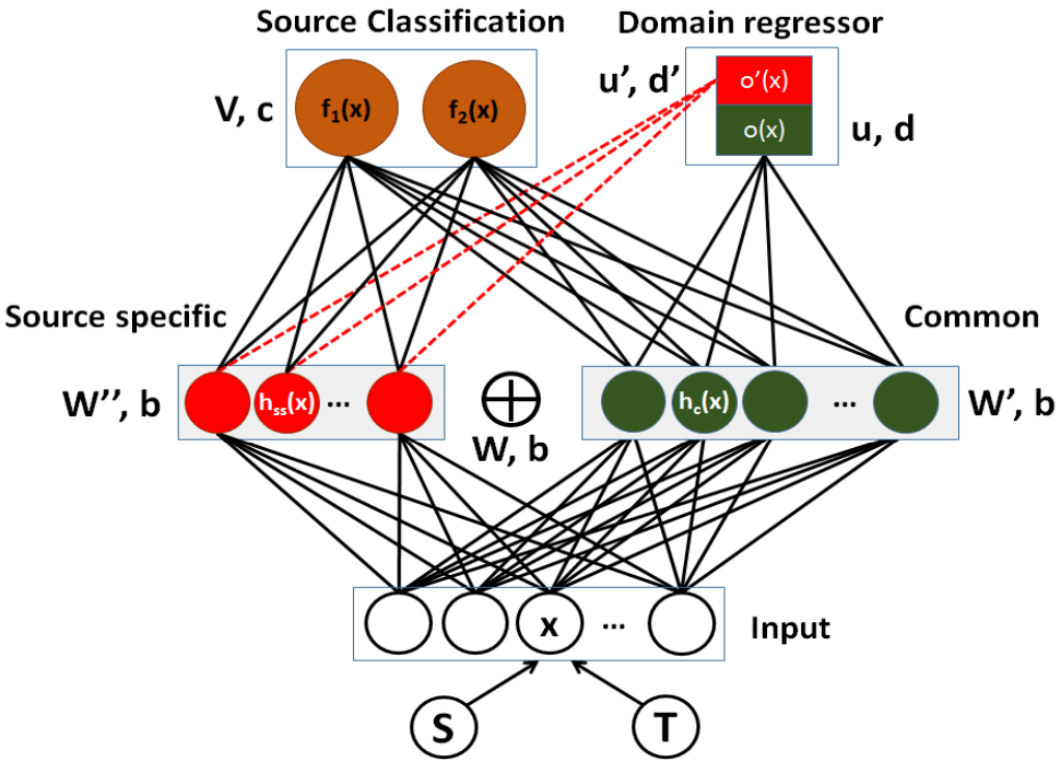
Domain Divergence

- Encourage the common representation to be domain invariant.

$$d_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) = 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_s} \left[\eta(\mathbf{x}^s) = 1\right] - \Pr_{\mathbf{x}^t \sim \mathcal{D}_t} \left[\eta(\mathbf{x}^t) = 1\right] \right|$$

$$\hat{d}^c_{\mathcal{H}}(h_c(S), h_c(T)) =$$

$$2\left(1 - \min_{\eta \in \mathcal{H}} \left[\frac{1}{m}\sum_{i=1}^{m} I[\eta(h_c(x_i^s)) = 1] + \frac{1}{m'}\sum_{i=1}^{m'} I[\eta(h_c(x_i^t)) = 0]\right]\right)$$



Overall objective

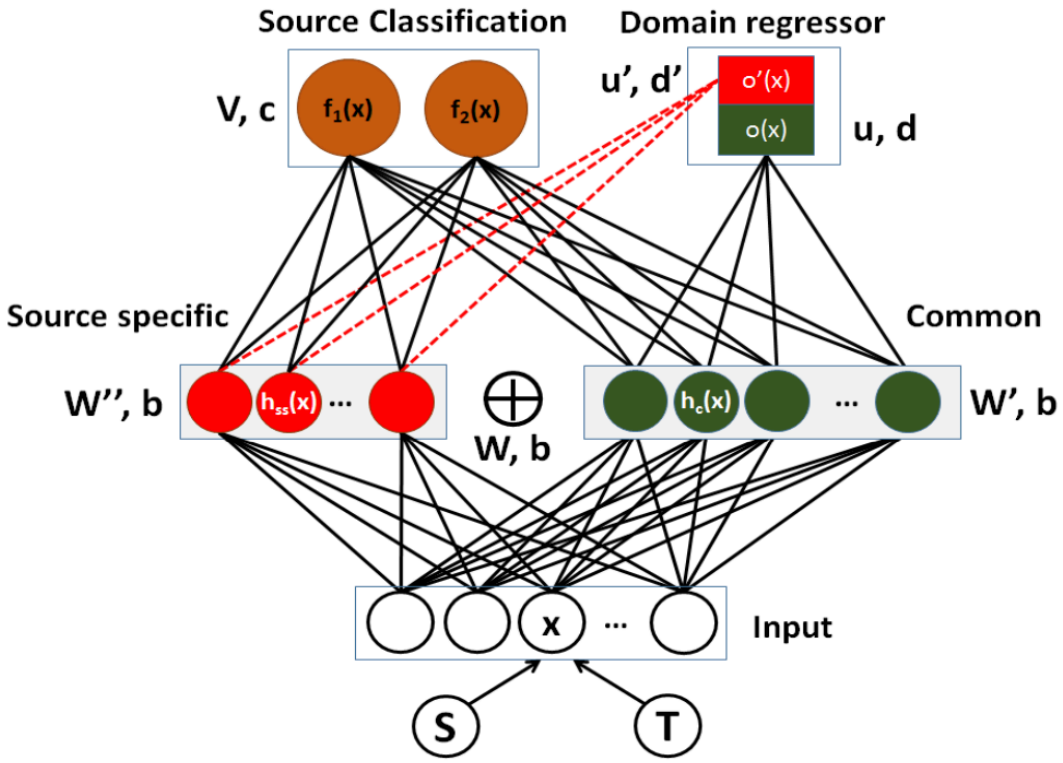
$$\min_{W,V,b,c}\Big[\frac{1}{m}\sum_{i=1}^{m}\ell(f(x_i^s),y_i^s)+$$

$$\lambda\max_{W',u,b,d}\Big(-\frac{1}{m}\sum_{i=1}^{m}\ell^d(o(x_i^s),1)$$

$$-\frac{1}{m'}\sum_{i=1}^{m'}\ell^d(o(x_i^t),0)\Big)+$$

$$\lambda\min_{W'',u',b,d'}\Big(-\frac{1}{m}\sum_{i=1}^{m}\ell^{d'}(o'(x_i^s),1)$$

$$-\frac{1}{m'}\sum_{i=1}^{m'}\ell^{d'}(o'(x_i^t),0)\Big)\Big]$$



Experiments

- Input: Bag-of-words (tf-idf)
- Dataset 1: Amazon Movie Reviews
- Domain: 1000 +ve / 1000 -ve
  - 700/700 – Training Set
  - 100/100 – Dev Set
  - 200/200 – Test Set
- Dataset 2: Online Social Media (OSM)

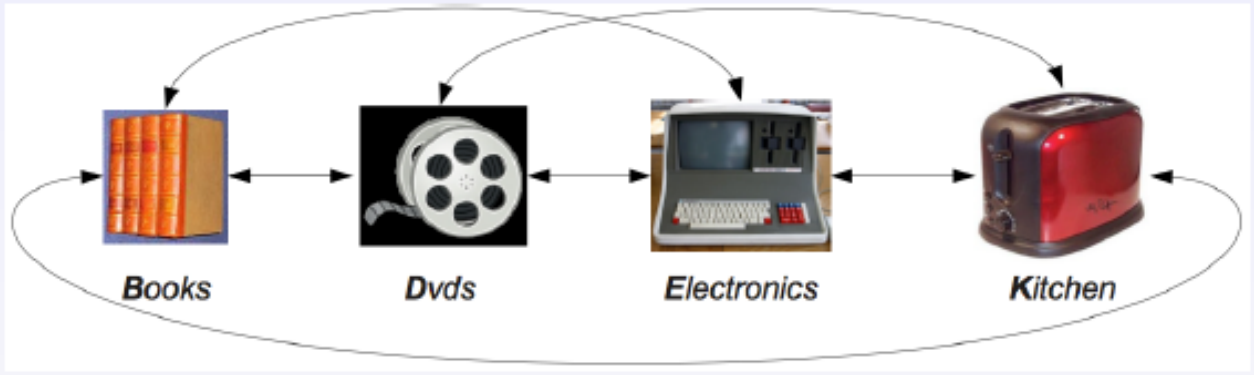


Table 1: Collections from the OSM dataset.

Target	Description	# Unlabelled	# Labelled
Col1	Mobile support	22645	5650
Col2	Obama Healthcare	36902	11050
Col3	Microsoft kinnect	20907	3258
Col4	X-box	36000	4580

Cross-domain performance on Amazon Reviews

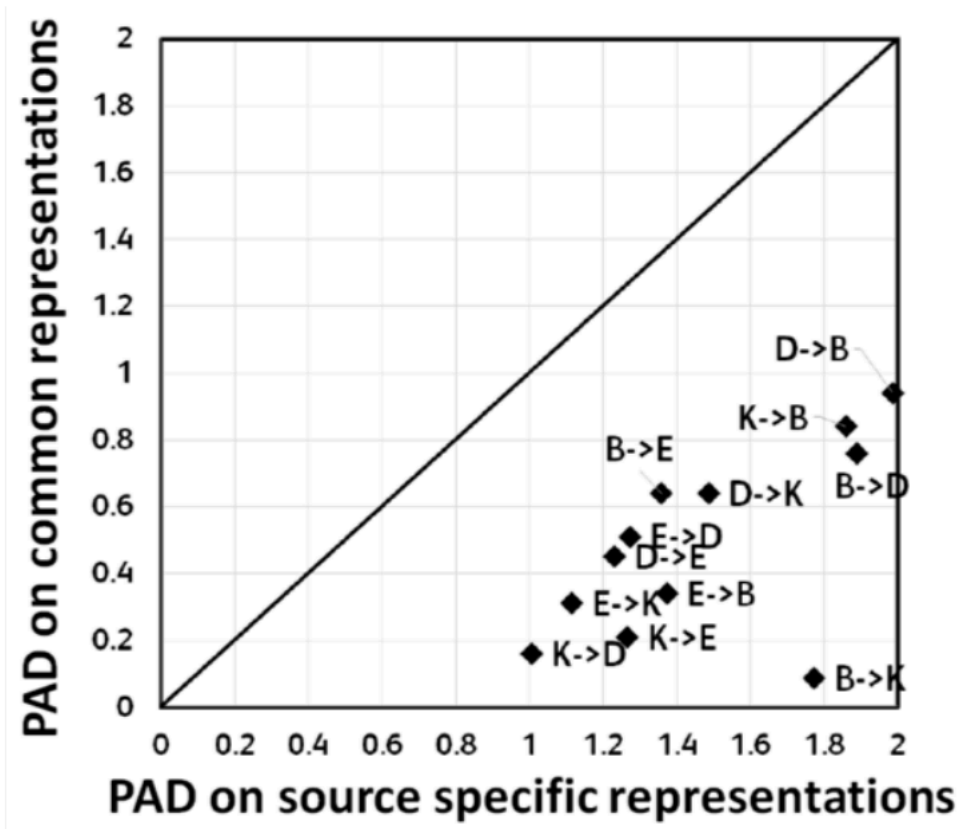
Method	D→ B	E→ B	K→ B	B→ D	E→ D	K→ D	D→ E	B→ E	K→ E	D→ K	B→ K	E→ K
SS	43.2	47.3	47.0	43.5	47.4	46.7	48.6	47.6	48.4	51.2	51.0	49.6
NN	76.8	71.4	74.2	71.4	72.0	67.3	71.6	72.4	77.6	76.5	77.8	79.6
SVM	77.9	73.2	75.4	73.2	73.5	70.4	73.6	71.5	78.2	77.7	78.8	82.3
SCL	78.7	75.3	76.8	78.2	75.0	73.1	75.3	75.8	84.0	77.1	79.3	85.4
SFA	80.5	75.9	76.6	77.6	75.3	74.2	75.4	77.0	84.2	78.1	80.3	85.8
PJNMF	81.8	77.2	78.8	79.4	76.3	75.8	76.4	77.8	84.4	79.0	81.6	86.4
SDA	81.1	76.6	76.8	78.2	75.4	75.4	75.8	77.4	83.8	78.4	80.8	87.2
mSDA	81.3	77.6	78.5	79.5	76.5	76.4	75.4	77.2	83.6	78.5	81.2	88.2
TLDA	81.5	78.0	80.6	79.8	76.6	76.4	76.2	78.0	84.2	79.4	81.8	87.6
BTDNNs	81.9	78.6	81.2	80.0	77.9	76.2	76.8	78.6	85.2	80.5	82.7	88.3
SS+Common	78.8	76.7	77.3	74.4	77.8	73.6	74.4	76.8	80.4	78.6	80.2	83.5
DANN	79.5	77.4	78.2	76.3	78.4	76.3	75.2	77.2	81.4	78.9	80.6	85.8
DSN	81.5	78.9	79.0	78.3	79.5	77.4	76.0	78.3	83.4	79.5	81.4	87.7
Proposed	**83.2**	**81.8**	**83.8**	**81.3**	**81.8**	**82.2**	**82.4**	**83.2**	**86.0**	**86.2**	**88.4**	**89.9**
Gold-standard	*84.6*	*84.6*	*84.6*	*83.4*	*83.4*	*83.4*	*86.7*	*86.7*	*86.7*	*90.2*	*90.2*	*90.2*

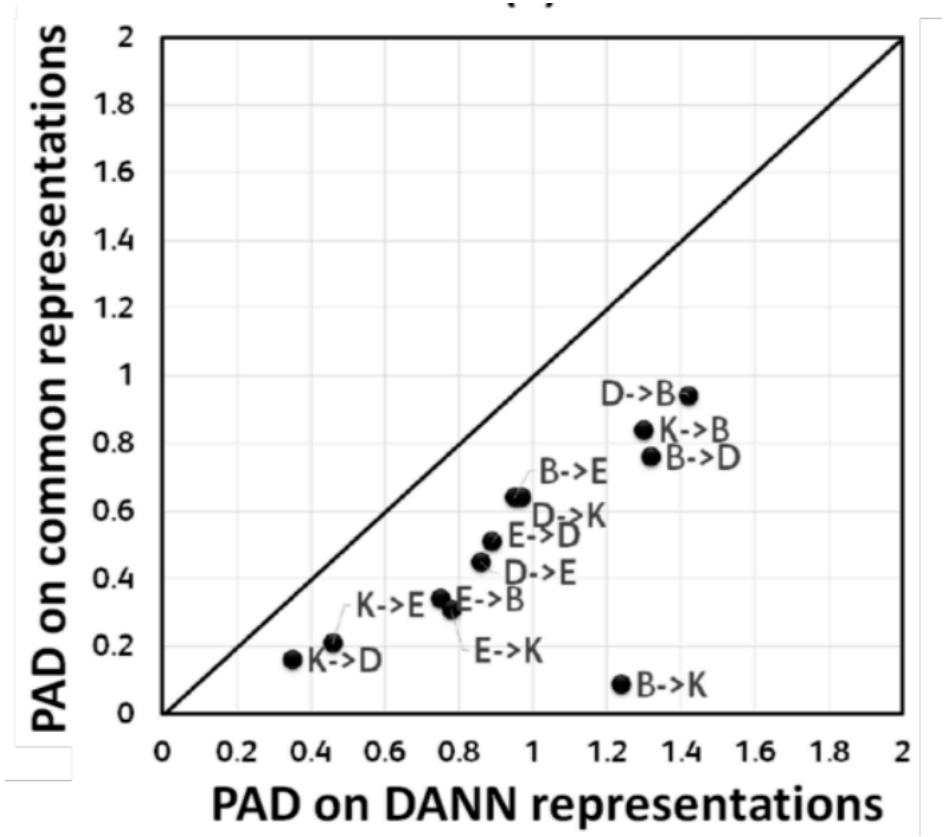
Cross-domain performance on OSM Reviews

Method	Col2→1	Col3→1	Col4→1	Col1→2	Col3→2	Col4→2	Col1→3	Col2→3	Col4→3	Col1→4	Col2→4	Col3→4
SS	35.0	39.4	35.6	32.8	40.2	38.6	40.7	41.9	42.5	45.0	44.9	42.4
NN	66.4	65.2	68.3	65.8	66.8	63.8	65.2	67.2	68.2	67.3	67.2	68.1
SVM	67.1	63.2	64.3	62.6	64.3	60.4	62.8	63.2	65.8	68.2	69.3	72.4
SCL	68.2	67.5	67.2	67.1	67.3	64.1	64.5	65.3	72.1	68.8	70.1	73.6
SFA	71.3	67.6	67.8	69.1	70.2	67.8	68.2	68.4	74.2	69.5	72.3	76.3
PJNMF	72.0	67.2	68.3	70.4	70.5	68.4	69.3	69.1	74.8	70.0	72.5	74.8
SDA	71.5	66.3	67.6	68.2	69.3	70.2	67.6	68.3	68.7	72.4	69.3	72.6
mSDA	72.1	67.5	68.2	69.0	70.4	70.8	68.3	69.1	69.2	73.0	70.2	73.1
TLDA	72.4	67.8	68.6	69.7	71.1	71.5	68.8	69.8	70.0	73.8	70.7	73.8
BTDNNs	73.1	68.3	69.0	70.2	71.6	72.1	69.4	70.2	70.6	74.2	71.3	74.2
SS+Common	68.7	67.9	67.7	67.5	67.8	64.9	65.0	65.7	72.6	69.4	70.7	74.2
DANN	69.6	69.5	69.8	70.0	68.7	66.2	66.3	66.6	73.4	70.6	71.4	75.7
DSN	72.9	68.6	69.4	70.5	72.0	72.2	69.5	70.3	70.8	74.3	71.5	74.6
Proposed	**77.6**	**74.5**	**75.5**	**76.2**	**77.8**	**78.2**	**75.2**	**75.7**	**76.1**	**80.1**	**77.9**	**80.9**
Gold-standard	*78.2*	*78.2*	*78.2*	*79.1*	*79.1*	*79.1*	*81.0*	*81.0*	*81.0*	*81.4*	*81.4*	*81.4*

The Common Representation

$$\hat{d}_A = 2(1 - 2\epsilon)$$





The Source Specific Representation

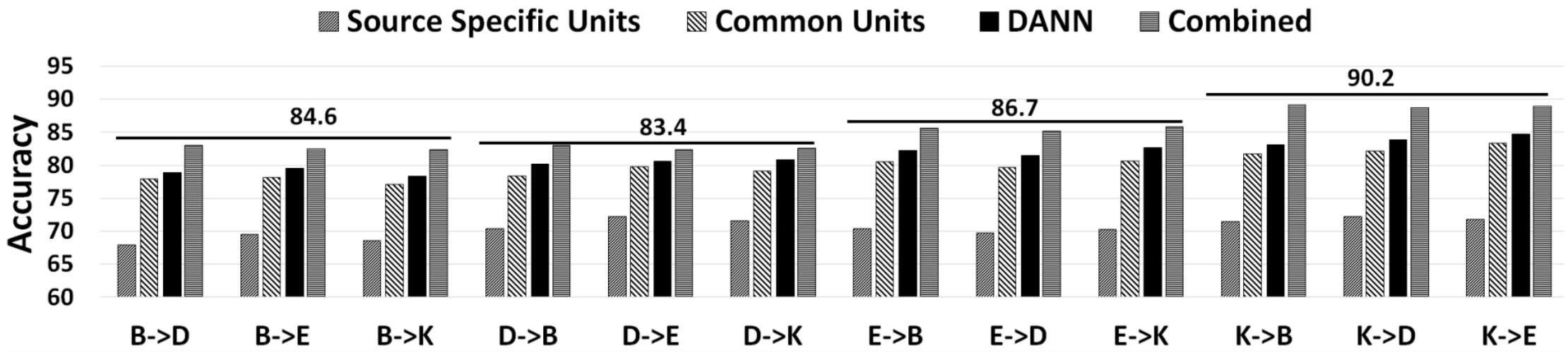


Figure 4: Compares the performance on the source classification task. For example, B→D here represent the performance of an algorithm on the source domain B when the representations are learned with B as labeled source and D as unlabeled target domain.

Conclusion

- A novel model learning a two-part representation where each part optimizes for different objective.
- Major Contribution: **Disentangling the source specific characteristics so as not to detract the capabilities of common representation for the cross-domain task.**
- Good for Target: **Common**
- Good for Source: **Source + Common**
- A tutorial to build DANN in PyTorch will be available soon: https://github.com/UBC-NLP/dlnlp2019_resources