Deep Multi-Task Learning with Adversarial-and-Cooperative Nets

Pei Yang[1,2] , **Qi Tan**[3] , **Jieping Ye**[4] , **Hanghang Tong**[2] and **Jingrui He**[2]
[1]South China University of Technology
[2]Arizona State University
[3]South China Normal University
[4]University of Michigan
cs.pyang@gmail.com, tanqi@scnu.edu.cn, jpye@umich.edu,
hanghang.tong@asu.edu, jingrui.he@asu.edu

Abstract

In this paper, we propose a deep multi-Task learning model based on Adversarial-and-COoperative nets (TACO). The goal is to use an adversarial-and-cooperative strategy to decouple the task-common and task-specific knowledge, facilitating the fine-grained knowledge sharing among tasks. TACO accommodates multiple game players, i.e., feature extractors, domain discriminator, and tri-classifiers. They play the MinMax games adversarially and cooperatively to distill the task-common and task-specific features, while respecting their discriminative structures. Moreover, it adopts a divide-and-combine strategy to leverage the decoupled multi-view information to further improve the generalization performance of the model. The experimental results show that our proposed method significantly outperforms the state-of-the-art algorithms on the benchmark datasets in both multi-task learning and semi-supervised domain adaptation scenarios.

1 Introduction

Domain shift [Blitzer *et al.*, 2007] usually refers to the difference of distributions between the data collected from multiple sources. For example, the images from different domains may encounter domain shift caused by changes in the camera, image resolution, lighting, background, and viewpoint [Saenko *et al.*, 2010]. It was shown that even state-of-the-art deep convolutional neural network (CNN) [Donahue *et al.*, 2014] learned on millions of instances are susceptible to domain shift. On the other hand, it is expensive or unrealistic to collect a large amount of labeled data for each task or domain. Therefore, deep multi-task learning [Ruder, 2017; Yang and Hospedales, 2017; Long *et al.*, 2017a] becomes a promising research direction since it conjoins the strength of deep learning [LeCun *et al.*, 2015] and multi-task learning [Caruana, 1993]. The main challenge of deep multi-task learning is how to decouple task-variance and task-invariance from deep features while respecting their discriminativeness. Task-invariance captures the common knowledge shared across tasks, while task-variance characterizes the specific property of each task which is distinctive from others.

In this paper, we propose a deep multi-Task learning method based on Adversarial-and-COoperative nets (TACO) to address this issue. The main idea is to adopt an adversarial-and-cooperative strategy to disentangle between task-invariant and task-variant information, and use the decoupled multi-view features to enhance the discriminativeness of the model. The TACO network consists of multiple game players, i.e., feature extractors, domain discriminator, and triple types of classifier. They play the MinMax games adversarially and cooperatively to extract both task-common and task-specific knowledge from data while maintaining their discriminative structures. On one hand, the feature extractors compete with the domain discriminator to learn the task-common knowledge in a domain-adversarial way. It encourages the extractors to produce the task-invariant features, which are indistinguishable by the domain discriminator. On the other hand, the extractors play with the classifiers to learn the task-specific knowledge in a classifier-adversarial fashion. For each task, it encourages each extractor to generate the task-specific features, which are exclusive to its own classifier. Furthermore, the task-common and task-specific features naturally form multiple views for the data. Therefore, in order to make advantage of the complementary benefit from multiple views, we build three types of classifiers, i.e., task-specific classifier, task-common classifier, and within-task classifier. The triple classifiers cooperate with each to enhance discriminativeness of the models. We conduct the detailed experiments on both multi-task learning and semi-supervised domain adaptation scenarios. The empirical study shows TACO significantly outperforms the state-of-the-art algorithms. The main contributions of this work are summarized as follows:

- Adversarial-and-cooperative nets to decouple task-invariance and task-variance for the fine-grained knowledge sharing among tasks.
- Divide-and-combine strategy to leverage the decoupled multi-view features to improve the performance.
- Experiments on the benchmark data demonstrating the effectiveness of the proposed method.

The rest of the paper is organized as follows. Section 2 reviews the related work. The proposed method is presented in Section 3, followed with the experimental results in Section 4. We conclude the paper in Section 5.

2 Related Work

Multi-task learning [Caruana, 1993] boosts the performance of each task by sharing knowledge among related tasks. A number of multi-task learning methods have been proposed, such as: multi-task feature learning [Argyriou *et al.*, 2008], clustered multi-task learning [Zhou *et al.*, 2011], low-dimensional subspace learning [Ji and Ye, 2009], multi-task relationship learning [Zhang and Yeung, 2010], robust multi-task learning [Chen *et al.*, 2013], and sparsity-regularized multi-task learning including weighted Lasso [Lee *et al.*, 2016], tree-guided fused lasso [Han and Zhang, 2015], generalized Schatten norm [Yang *et al.*, 2017], etc.

Deep multi-task learning or deep domain adaptation becomes to receive attentions owning to its ability of learning hierarchical features from data and sharing knowledge across domains. The related approaches can be roughly categorized into four types: domain-adversarial networks [Ganin and Lempitsky, 2015; Tzeng *et al.*, 2015; Bousmalis *et al.*, 2016; Tzeng *et al.*, 2017; Pei *et al.*, 2018], adaptation methods based on maximum mean discrepancy (MMD) [Long *et al.*, 2017b; Ghifary *et al.*, 2014; Long *et al.*, 2015; Tzeng *et al.*, 2014], deep models with tensor prior [Yang and Hospedales, 2017; Long *et al.*, 2017a], and methods based on both feature-level and instance-level adapations [Hoffman *et al.*, 2018; Yang *et al.*, 2019]. First, the domain-adversarial neural network [Ganin and Lempitsky, 2015] used adversarial training [Goodfellow *et al.*, 2014] to promote the emergence of domain-invariant features via the use of a gradient reversal layer. In [Tzeng *et al.*, 2015], they simultaneously aligned domain via domain confusion and aligned source and target classes via soft labels. The domain separation networks [Bousmalis *et al.*, 2016] explicitly modeled both private and shared components of domain representations. The adversarial discriminative domain adaptation model [Tzeng *et al.*, 2017] combined discriminative modeling, untied weight sharing, and a domain-adversarial loss into a unified framework. The multi-adversarial domain adaptation approach [Pei *et al.*, 2018] captured multimode structures to enable fine-grained alignment of domains based on multiple domain discriminators. Second, the joint adaptation networks [Long *et al.*, 2017b] adopted adversarial training strategy to maximize a joint MMD criterion. The domain adaptive neural network [Ghifary *et al.*, 2014] incorporated MMD measure as a regularization embedded in the supervised back-propagation training. DDC [Tzeng *et al.*, 2014] had an MMD loss at one layer while DAN [Long *et al.*, 2015] had MMD losses at multiple layers. Third, the deep multi-task learning with tensor factorization model [Yang and Hospedales, 2017] learned shared feature subspace from multilayer parameter tensors, while multilinear relationship networks [Long *et al.*, 2017a] learned multilinear task relationships from multiplayer parameter tensors. Some recent methods learn feature-level and instance-level adaptations simultaneously. The cycle-consistent adversarial adaptation model [Hoffman *et al.*, 2018] enforced both structural and semantic consistency during adaptation using a cycle-consistency loss and semantics loss. The task-adversarial co-generative nets [Yang *et al.*, 2019] simultaneously learns the marginal distribution of task-invariant features across tasks and the joint distributions of examples with labels for each task.

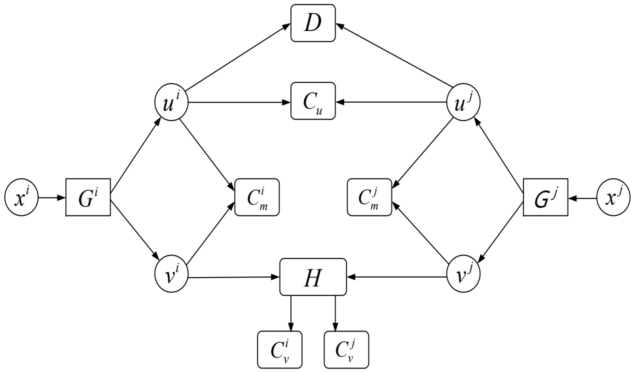


Figure 1: The high-level architecture of the TACO model.

Most domain-adversarial methods rely on adversarial training to learn the task-invariant knowledge only. In contrast, TACO decouples the task-common and task-specific features in an adversarial-and-cooperative way, and leverages their complementary correlations to facilitate the fine-grained knowledge sharing across tasks.

3 The TACO Method

In this section we introduce the TACO model, together with the network architecture and the learning algorithm.

3.1 Model

Suppose we have the data from T related tasks. Each task refers to a multi-class classification problem. There are limited labeled instances in each task. The goal is to predict the unlabeled data for each task as accurately as possible by using TACO to manipulate knowledge sharing among tasks.

Figure 1 shows the high-level architecture of the TACO method. Let G^i and G^j be the feature extractors for the i^{th} and j^{th} tasks respectively, where $1 \leq i, j \leq T$. Each feature extractor G accepts the instance x and generates its task-common features u and task-specific features v. The task index (if applicable) is omitted for brevity. We have three types of classifiers, i.e., task-common classifier C_u, task-specific classifier C_v, and within-task classifier C_m. D is the domain discriminator. H refers to the router, which will be introduced in details in the next subsection. Note that we assume that there are two tasks in Figure 1 for simplicity. But the proposed method is directly ready for the multiple ($T \geq 2$) tasks.

TACO is composed of multiple game players, i.e., feature extractors, domain discriminator, and three types of classifiers. Each game player may include multiple feed-forward layers. Let θ_g^i, θ_d, θ_u, θ_v^i, and θ_m^i be the learning parameters for G^i, D, C_u, C_v^i, and C_m^i, respectively, where $1 \leq i \leq T$. Denote $\theta_g = \{\theta_g^i | 1 \leq i \leq T\}$, $\theta_v = \{\theta_v^i | 1 \leq i \leq T\}$, and $\theta_m = \{\theta_m^i | 1 \leq i \leq T\}$. Let L_u, L_v^i, and L_m^i be the loss for the task-common classifier C_u, task-specific classifier C_v^i, and within-task classifier C_m^i respectively. The domain discrimination loss is denoted by L_d.

TACO aims to learn both task-common and task-specific knowledge, together with their mutual correlations, in a unified model.

The first question is how to extract the task-common features. We learn the commonality in a domain-adversarial way. The feature extractors try to make the task-common features indistinguishable by the domain discriminator, while the domain discriminator attempts to predict the domain label as accurately as possible. Specifically, in order to learn the task-invariant features, we seek the parameters θ_g of the feature extractors that maximize the loss of the domain discriminator, while simultaneously seeking the parameters θ_d of the domain discriminator that minimize the discriminating loss. On the other hand, we minimize the loss of task-common classifier to make sure that the task-common features maintain the label information. Therefore, the MinMax game among the domain discriminator D, the task-common classifier C_u, and the feature extractors $G^i (1 \leq i \leq T)$ can be formulated as follows:

$$\min_{\theta_u,\theta_g} \max_{\theta_d} L_u(\theta_g,\theta_u) - \alpha L_d(\theta_g,\theta_d) \tag{1}$$

where α is a non-negative trade-off parameter. In this MinMax game, the feature extractors, domain discriminator, and the task-common classifier work in a domain-adversarial way to generate the features which are task-invariant and discriminative.

The second question is how to learn the task-specific features. Different from the task-common features shared by tasks, the task-specific features are exclusive to its own task. Intuitively, since the task-specific features contain knowledge for its own task only, the classifiers of the other tasks should perform bad on these features. Therefore, we learn the task-variance in a classifier-adversarial way. For each task, it encourages the feature extractor to produce the task-specific features which minimize the loss of its own task-specific classifier, while maximizing the loss of the task-specific classifiers for all the other tasks. Specifically, we seek the parameters θ_g^i of feature extractor G^i and the parameters θ_v^i of task-specific classifier C_v^i that minimize the task-specific loss L_v^i, while simultaneously seeking the parameters $\tilde{\theta}_g^i = \{\theta_g^j | j \neq i, 1 \leq j \leq T\}$ of the features extractors for the other tasks that maximize L_v^i. Hence, the MinMax game played among two groups, i.e., $\{G^i, C^i\}$ and $\{G^j | j \neq i, 1 \leq j \leq T\}$, can be formulated as:

$$\min_{\theta_v^i,\theta_g^i} \max_{\tilde{\theta}_g^i} L_v^i(\theta_g^i, \tilde{\theta}_g^i, \theta_v^i) \tag{2}$$

The multiple players work in the classifier-adversarial way to generate the features which are task-specific and discriminative. In total, the task-specific adversarial loss for all the tasks is as follows:

$$\min_{\theta_v,\theta_g} \max_{\theta_g} \sum_{i=1}^{T} L_v^i(\theta_g^i, \tilde{\theta}_g^i, \theta_v^i) \tag{3}$$

The third question is how to further enhance the discriminativeness of the models by leveraging the decoupled knowledge. The task-common and task-specific features learned above naturally form multiple views for each task. To fully take advantage of mutually benefit from multiple views, we build three types of classifiers, i.e., task-common classifier, task-specific classifier, and within-task classifier. Both the task-common and the task-specific classifiers are introduced as above. For within-task classifier $C_m^i (1 \leq i \leq T)$, it is trained with the concatenated features, i.e., the combination of task-common and task-specific features. We seek the parameter of θ_g^i of feature extractor G^i and the parameter θ_m^i of the within-task classifier C_m^i, which minimizes the loss L_m^i. Thus, the loss for the within-task classifiers can be formulated as:

$$\min_{\theta_m,\theta_g} \sum_{i=1}^{T} L_m^i(\theta_g^i, \theta_m^i) \tag{4}$$

The tri-classifiers cooperate with each other to fully leverage the complementarity among multiple views. We follow the conventional way of ensemble learning to combine the predictions. The three classifiers determine the final prediction of each instance by voting. If neither agrees with each other, the prediction will be determined by the classifier with the highest confidence.

In summary, the overall objective function for TACO is formulated as follows:

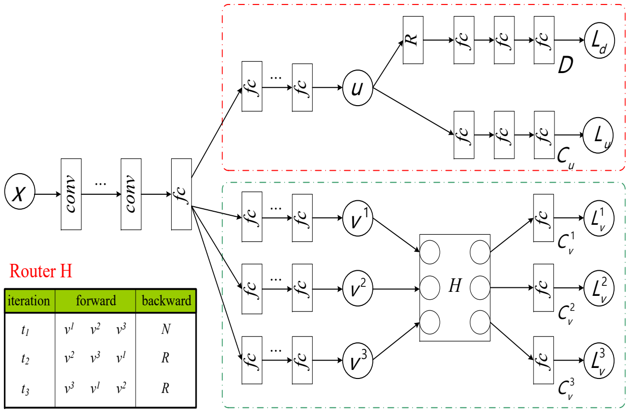
$$\begin{aligned} \min_{\{\theta_u,\theta_v,\theta_m,\theta_g\}} \max_{\{\theta_d,\theta_g\}} & L_u(\theta_g,\theta_u) - \alpha L_d(\theta_g,\theta_d) \\ & + \sum_{i=1}^{T} \left[L_v^i(\theta_g^i, \tilde{\theta}_g^i, \theta_v^i) + L_m^i(\theta_g^i, \theta_m^i) \right] \end{aligned} \tag{5}$$

TACO accommodates multiple game players, i.e., feature extractors, domain discriminator, and tri-classifiers, in a unified model. The feature extractors involve in both MinMax games, where they cooperate to generate the task-common features, while competing with each other to generate the task-specific features. In such a way, the multi-players work in an adversarial-and-cooperative way to distill the task-common and task-specific features. Moreover, it uses a divide-and-combine strategy to leverage the decoupled multi-view information to improve the generalization performance.

3.2 Network

Note that various deep networks can be used as the base architecture to build the TACO model. Here we take two of the most popular CNN networks, i.e., AlexNet [Krizhevsky *et al.*, 2012] and VGGnet [Simonyan and Zisserman, 2015], as the base networks to instantialize the TACO model.

The CNN-based TACO architecture is shown in Figure 2. x represents the input data from all tasks. The backbone is composed of several convolutional layers (e.g., conv1 - conv5 in AlexNet), max-pooling layers, dropout layers, and one fully-connected layer (fc). These layers focus on extracting the common low-level features from all the tasks. From here on, the network is divided into two branches. The top branch is made of several fully-connected layers, normalization layers, and one gradient reversal layer *R* [Ganin and Lempitsky, 2015]. It aims to learn the task-common features. The bottom branch consists of a few fully-connected layers, normalization layers, and one router layer *H*. The goal is to distill the task-specific knowledge for each task. For brevity, the within-task classifiers are omitted in Figure 2. The softmax function is used to compute the domain discrimination loss and the label classification losses.



iteration	forward			backward
t_1	v^1	v^2	v^3	N
t_2	v^2	v^3	v^1	R
t_3	v^3	v^1	v^2	R

Figure 2: The CNN-based TACO network.

We present a novel neural component, i.e, router *H*, to facilitate the classifier-adversarial training. It can be viewed as a data router combined with the gradient reversal layer. The gradient reversal layer [Ganin and Lempitsky, 2015] leaves the input unchanged during forward propagation and reverses the gradient by multiplying it with a negative scalar during the backpropagation. The H layer has T input and output gates respectively. We assume $T = 3$ here.The table in the bottom-left of Figure 2 shows the rules of feed-forward and backpropagation for H. Each H has T iteration periods. At the first iteration, the task-specific features, v^1, v^2, and v^3, are routed to their own task-specific classifiers, C_v^1, C_v^2, and C_v^3, respectively. At the other iterations, the task-specific features are fed to the other task-specific classifiers. For example, at t_2 iteration, v^2, v^3, and v^1, are routed to C_v^1, C_v^2, and C_v^3, respectively. The 'R' in the backward column means that the gradients should be reversed during the backpropagation, while 'N' means the normal backpropagation of gradients.

3.3 Algorithm

The TACO algorithm is showed in Algorithm 1. It consists of two MinMax games played among feature extractor, domain discriminator, and tri-classifiers. Line 1 initializes the net weights with pretrained nets such as AlexNet [Krizhevsky *et al.*, 2012] and VGGnet [Simonyan and Zisserman, 2015]. Lines 3-4 sample a batch of data and forward pass it through the nets. Lines 5-8 optimize the domain discriminator and tri-classifiers, while Lines 9-13 update the feature extractor. The feature extractor involves in both MinMax games, which is updated as follows:

- At t_1 round, we update the feature extractor θ_g^i:

$$\theta_g^i = \arg\min_{\theta_g^i} \Big[L_u(\theta_g, \theta_u) - \alpha L_d(\theta_g, \theta_d) + L_v^i(\theta_g^i, \tilde{\theta}_g^i, \theta_v^i) + L_m^i(\theta_g^i, \theta_m^i) \Big] \tag{6}$$

- At $t_k (2 \le k \le T)$ round, we update the feature extractor θ_g^i using the loss for other tasks:

$$\theta_g^i = \arg\max_{\theta_g^i} L_v^j(\theta_g^j, \theta_g^i, \theta_v^j) \tag{7}$$

where $j = (i + k - 1)\%T$ (% is a mod operator).

Algorithm 1 The **TACO** Algorithm

```
Input:   multi-task data $x^i (1 \le i \le T)$, trade-off parameter
         $\alpha$, maximum iteration $\tau_{max}$, batch size $b$.
Output:  predictions for unlabeled data.
 1: Initialize the net weights with pretrained nets.
 2: for number of iterations $\tau_{max}$ do
 3:    Sample a batch of $b$ instances from each task;
 4:    Forward pass the batch through the network and compute the losses;
 5:    Update domain discriminator $D$ by ascending along its stochastic gradient $\nabla_{\theta_d} - \alpha L_d(\theta_g, \theta_d)$;
 6:    Update task-common classifier $C_u$ by descending along its stochastic gradient $\nabla_{\theta_u} L_u(\theta_g, \theta_u)$;
 7:    Update each task-specific classifier $C_v^i$ by descending along its stochastic gradient $\nabla_{\theta_v^i} L_v^i(\theta_g^i, \tilde{\theta}_g^i, \theta_v^i)$;
 8:    Update each within-task classifier $C_m^i$ by descending along its stochastic gradient $\nabla_{\theta_m^i} L_m^i(\theta_g^i, \theta_m^i)$;
 9:    if at $t_1$ round then
10:       Update each feature extractor $G^i$ by descending along its stochastic gradient using Eq. 6;
11:    else
12:       Update each feature extractor $G^i$ by ascending along its stochastic gradient using Eq. 7;
13:    end if
14: end for
15: Combine tri-classifiers to get the predictions.
16: Augment the training data with the pseudo labels.
17: Re-train the nets and get the final predictions.
```

The TACO algorithm is implemented using the Caffe framework [Jia *et al.*, 2014]. It can be trained using the min-batch stochastic gradient descent method. We adopt learning rate decaying strategy.

4 Experiments

We evaluate the proposed TACO algorithm in two scenarios with domain shift, i.e., multi-task learning and semi-supervised domain adaptation. The experiments are conducted on three image datasets, which are the standard benchmarks for multi-task learning and domain adaptation.

The Office-Home[1] [Venkateswara *et al.*, 2017] dataset consists of 15500 images from 4 different domains: Artistic images (A), Clip art (C), Product images (P) and Real-world images (R). For each domain, the dataset contains images of 65 object categories collected in office and home settings.

The Office-31[2] [Saenko *et al.*, 2010] dataset is a collection of 4652 images in 31 categories collected from three distinct domains, Amazon (A), DSLR (D), and Webcam (W). The Amazon domain consists of images at medium resolution typically taken in an environment with studio lighting conditions. The DSLR domain consists of images that are captured with a digital camera in realistic environments. The Webcam domain is composed of images recorded with a simple webcam at low resolution.

[1] http://hemanthdv.org/OfficeHome-Dataset/

[2] https://people.eecs.berkeley.edu/~jhoffman/domainadapt/

Method	5%					10%					20%				
	A	C	P	R	Avg	A	C	P	R	Avg	A	C	P	R	Avg
STL	35.8	31.2	67.8	62.5	49.3	51.0	40.7	75.0	68.8	58.9	56.1	54.6	80.4	71.8	65.7
MTFL	40.1	30.4	61.5	59.5	47.9	50.3	35.0	66.3	65.0	54.2	55.2	38.8	69.1	70.0	58.3
RMTL	42.3	32.8	62.3	60.6	49.5	49.7	34.6	65.9	64.6	53.7	55.2	39.2	69.9	70.5	58.6
MTRL	42.7	33.3	62.9	61.3	50.1	51.6	36.3	67.7	66.3	55.5	55.8	39.9	70.2	71.2	59.3
DMTRL	49.2	34.5	67.1	62.9	53.4	57.2	42.3	73.6	69.9	60.8	58.3	56.1	79.3	72.1	66.5
MRN	53.3	36.4	70.5	**67.7**	57.0	59.9	42.7	76.3	73.0	63.0	58.5	55.6	80.7	72.8	66.9
TACO	**59.7**	**48.3**	**75.3**	65.5	**62.3**	**66.9**	**63.8**	**81.1**	**79.9**	**72.9**	**71.1**	**70.6**	**86.6**	**79.0**	**77.1**

Table 1: Classification accuracy on Office-Home based on VGGnet.

Method	5%					10%					20%				
	A	W	D	C	Avg	A	W	D	C	Avg	A	W	D	C	Avg
STL	88.9	73.0	80.4	88.7	82.8	92.2	80.9	88.2	88.9	87.6	91.3	83.3	93.7	**94.9**	90.8
MTFL	90.0	78.9	90.2	86.9	86.5	92.4	85.3	89.5	89.2	89.1	93.5	89.0	95.2	92.6	92.6
RMTL	91.3	82.3	88.8	89.1	87.9	92.6	85.2	93.3	87.2	89.6	94.4	87.0	96.7	93.4	92.4
MTRL	86.4	83.0	95.1	89.1	88.4	91.1	87.1	97.0	87.6	90.7	90.0	88.8	99.2	94.3	93.1
DMTRL	91.2	88.3	92.5	85.6	89.4	92.2	91.9	97.4	86.8	92.0	92.6	97.6	94.5	88.4	93.3
MRN	92.5	**97.5**	97.9	87.5	93.8	93.6	**98.6**	98.6	87.3	94.5	94.4	98.3	**99.9**	89.1	95.5
TACO	**93.1**	92.4	**99.3**	**89.8**	**94.2**	**94.9**	96.5	**99.3**	**90.3**	**95.1**	**95.6**	**98.4**	99.2	91.6	**96.0**

Table 2: Classification accuracy on Office-Caltech based on AlexNet.

The Office-Caltech dataset consists of 2533 images selected from the 10 common categories shared by the Office-31 dataset and the Caltech-256[3] dataset. The Caltech-256 dataset is a collection of 30607 images in 256 categories downloaded from Google Images. Hence, it yields four learning tasks corresponding to four domains: Amazon (A), Webcam (W), DSLR (D), and Caltech (C).

The initial learning rate is set to 0.001, and momentum is 0.9. The training iteration is set as $\tau_{max} = 1000$, and batch size $b = 20$. We empirically set the parameter $\alpha = 0.1$.

4.1 Multi-task Learning

We first evaluate TACO on both Office-Home and Office-Caltech datasets in the multi-task learning scenario.

Protocols and Baselines

We follow the standard protocol [Zhang and Yeung, 2010; Long *et al.*, 2017a] for multi-task learning and randomly select 5%, 10%, and 20% samples from each task as trainset and use the rest as testset, respectively. A half of trainset is randomly chosen to select the optimal parameters. We repeat five random experiments and report the average classification accuracy on the testset.

We compare TACO with various methods: multi-task feature learning (MTFL) [Argyriou *et al.*, 2008], multi-task relationship learning (MTRL) [Zhang and Yeung, 2010], robust multi-task learning (RMTL) [Chen *et al.*, 2013], deep single-task learning (STL), deep multi-task learning with tensor factorization (DMTRL) [Yang and Hospedales, 2017], and multilinear relationship networks (MRN) [Long *et al.*, 2017a]. The former three are shallow multi-task learning algorithm, while the latter two are deep multi-task learning methods.

AlexNet [Krizhevsky *et al.*, 2012] or VGGnet [Simonyan and Zisserman, 2015] pre-trained on ImageNet is used as base networks. STL is based on AlexNet [Krizhevsky *et al.*, 2012] or VGGnet [Simonyan and Zisserman, 2015], and fine-tuned for each task independently.

Performance Comparison

Table 1 shows the classification accuracy on Office-Home dataset using VGGnet [Simonyan and Zisserman, 2015] as the base networks. Table 2 shows the results on Office-Caltech dataset using AlexNet [Krizhevsky *et al.*, 2012] as the base networks. The results of the comparison methods are quoted from the related papers [Argyriou *et al.*, 2008; Chen *et al.*, 2013; Zhang and Yeung, 2010; Yang and Hospedales, 2017; Long *et al.*, 2017a].

We have the following observations from the results. First, the deep single-task learning method STL performs better than the shallow multi-task learning approaches such as MTFL [Argyriou *et al.*, 2008], MTRL [Zhang and Yeung, 2010], and RMTL [Chen *et al.*, 2013] on the Office-Home dataset when the sampling ratio is relatively large (e.g., 10% or 20%), verifying the superiority of CNN for feature learning. However, when the labeled data turn sparser (e.g., 5% in Office-Home) or the domains are more similar (as in Office-Caltech), the shallow multi-task learning methods become to show their advantages by borrowing the strength from the related tasks. Second, all the deep multi-task learning methods including TACO, DMTRL [Yang and Hospedales, 2017], and MRN [Long *et al.*, 2017a] outperform both the deep single-task learning method STL and the shallow multi-task learning approaches. It demonstrates that deep multi-task learning can further promote the performance by simultaneously learning the hierarchical features from data and sharing knowledge across tasks. Third, our proposed TACO method outperforms all the comparison methods on both datasets in most case. The performance superiority of TACO over the baselines becomes more significant on the relatively difficult problem (i.e., Office-Home dataset). The strength of TACO is that it disentangles the task-common and task-specific features while respecting the discriminative structures, and further enhances the performance by leveraging the complementary correlations between the decoupled views.

[3] http://www.vision.caltech.edu/Image_Datasets/Caltech256/

4.2 Semi-supervised Domain Adaptation

We then evaluate TACO on the Office-31 dataset in the semi-supervised domain adaptation [Ganin and Lempitsky, 2015; Long *et al.*, 2015] scenario, where labeled data in target domain are much sparser than that in source domain.

Protocols and Baselines

Since the Office-31 dataset has three domains, we build six domain adaptation problems in total, i.e., $A \rightarrow W$, $A \rightarrow D$, $W \rightarrow A$, $W \rightarrow D$, $D \rightarrow A$, and $D \rightarrow W$. We follow the standard protocol [Saenko *et al.*, 2010] for this dataset to sample the labeled instances. For source domain, we sample 20 instances per category for the Amazon domain, and 8 instances per category for the DSLR and Webcam domains. For target domain, 3 labeled instances are randomly picked out for each category. The remainder in each domain are used as testset. We repeat five random experiments and report the average classification accuracy on the testset of target domain.

We compare TACO with state-of-the-art methods: domain adaptive neural network (DaNN) [Ghifary *et al.*, 2014], deep domain confusion (DDC) model [Tzeng *et al.*, 2014], deep adaptation network (DAN) [Long *et al.*, 2015], and DCSL [Tzeng *et al.*, 2015] which simultaneously aligns domain and classes. We also compare TACO with three CNN baselines, i.e., CNN-S, CNN-T, and CNN-M, which are trained using source labeled data, target labeled data, and both source and target labeled data, respectively. AlexNet [Krizhevsky *et al.*, 2012] pre-trained on ImageNet is used as base networks to build the deep learning models.

Performance Comparison

Table 3 shows the classification performance on the Office-31 dataset. The results of the comparison methods are quoted from the related papers [Ghifary *et al.*, 2014; Tzeng *et al.*, 2014; Long *et al.*, 2015; Tzeng *et al.*, 2015].

We have the following observations from the results. First, CNN-S performs better on two adaptation problems, i.e., $W \rightarrow D$ and $D \rightarrow W$, but worse on the other four problems, comparing to CNN-T. It suggests Webcam (W) and DSLR (D) are similar with each other, but different from Amazon (A). CNN-M improves upon CNN-S and CNN-T by augmenting the training data. It also outperforms DaNN [Ghifary *et al.*, 2014] that is trained on a denoising auto-encoder, although DaNN introduced the MMD loss to bridge the domain gap. Second, the deep domain adaptation methods can promote the accuracy by aligning the domains with MMD-based regularizations [Tzeng *et al.*, 2014; Long *et al.*, 2015] or using domain-adversarial training [Tzeng *et al.*, 2015], or aligning the classes with soft labels [Tzeng *et al.*, 2015]. Third, our proposed TACO method outperforms the comparison algorithms in all six adaptation problems. The results demonstrate that TACO based on the adversarial-and-cooperative mechanism is also effective in semi-supervised domain adaptation scenarios where the labeled data are much sparser in the target domain than that in the source domain. Again, it verifies that by decoupling the task-variance and task-invariance, we are able to better manipulate the knowledge sharing among tasks.

Method	A→W	A→D	W→A	W→D	D→A	D→W	Avg
DDC	84.1	-	-	96.3	-	95.4	-
DAN	85.7	-	-	96.4	-	97.2	-
DaNN	53.6	59.9	37.3	83.5	38.2	71.2	57.3
CNN-S	56.5	64.6	42.7	93.6	47.6	92.4	66.2
CNN-T	80.5	81.8	59.9	81.8	59.9	80.5	74.1
CNN-M	82.5	85.2	65.2	96.3	65.8	93.9	81.5
DCSL	82.7	86.1	65.0	97.6	66.2	95.7	82.2
TACO	**87.1**	**90.2**	**70.8**	**98.2**	**71.4**	**97.6**	**85.9**

Table 3: Classification accuracy on Office-31 based on AlexNet.

4.3 Ablation Study and Convergence

We conduct an ablation study to investigate the impact of TACO's key components on performance. We compare TACO with its two reduced versions, TACO$_\alpha$ and TACO$_\beta$, as well as the baseline STL. TACO$_\alpha$ involves the generation of task-common features only, while TACO$_\beta$ involves the generation of both task-common and task-specific features. Figure 3 shows the ablation study results on Office-Home dataset. The training ratio is fixed to 10%, and the number of iteration is set to 1000. It shows that TACO$_\alpha$ outperforms STL by learning the task-invariant features, which help smooth the domain shift. TACO$_\beta$ improves upon TACO$_\alpha$ by disentangling the task-variance and task-invariance, which allows for the fine-grained knowledge sharing among tasks. Finally, TACO obtains the best performance by further making use of mutual benefit from multiple views. The results suggest each of TACO's key components is indispensable. Also, Figure 3 shows that TACO runs stably and converges fast.

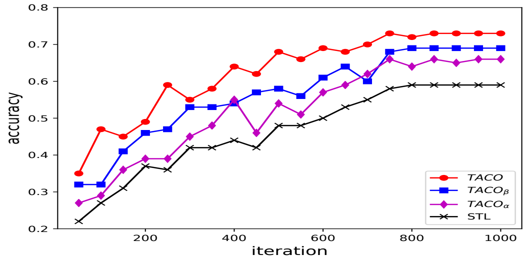


Figure 3: Ablation study on Office-Home dataset.

5 Conclusion

We propose a novel TACO method for deep multi-task learning. TACO adopts an adversarial-and-cooperative mechanism to learn both task-common and task-specific features. It further leverages the disentangled multi-view features to enhance the discriminativeness.The experiments on the benchmark data demonstrate the effectiveness of the approach.

Acknowledgements

This work is supported by National Natural Science Foundation of China (No. 61473123), Natural Science Foundation of Guangdong (No. 2017A030313370 and 2018A030313356), National Science Foundation (No. IIS-1552654, IIS-1813464, IIS-1651203, and CNS-1629888), U.S. Department of Homeland Security (No. 17STQAC00001-02-00), and an IBM Faculty Award. The views are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the governments.

References

[Argyriou *et al.*, 2008] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[Blitzer *et al.*, 2007] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

[Bousmalis *et al.*, 2016] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NIPS*, pages 343–351, 2016.

[Caruana, 1993] Rich Caruana. Multitask learning: A knowledge-based source of inductive bias. In *ICML*, pages 41–48, 1993.

[Chen *et al.*, 2013] Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. A convex formulation for learning a shared predictive structure from multiple tasks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(5):1025–1038, 2013.

[Donahue *et al.*, 2014] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML*, pages 647–655, 2014.

[Ganin and Lempitsky, 2015] Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189, 2015.

[Ghifary *et al.*, 2014] Muhammad Ghifary, W. Bastiaan Kleijn, and Mengjie Zhang. Domain adaptive neural networks for object recognition. In *PRICAI*, pages 898–904, 2014.

[Goodfellow *et al.*, 2014] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[Han and Zhang, 2015] Lei Han and Yu Zhang. Learning tree structure in multi-task learning. In *KDD*, pages 397–406, 2015.

[Hoffman *et al.*, 2018] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, pages 1994–2003, 2018.

[Ji and Ye, 2009] Shuiwang Ji and Jieping Ye. An accelerated gradient method for trace norm minimization. In *ICML*, pages 457–464, 2009.

[Jia *et al.*, 2014] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint:1408.5093*, 2014.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[Lee *et al.*, 2016] Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *ICML*, pages 230–238, 2016.

[Long *et al.*, 2015] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, pages 97–105, 2015.

[Long *et al.*, 2017a] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Philip S. Yu. Learning multiple tasks with multilinear relationship networks. In *NIPS*, pages 1593–1602, 2017.

[Long *et al.*, 2017b] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, pages 2208–2217, 2017.

[Pei *et al.*, 2018] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *AAAI*, 2018.

[Ruder, 2017] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017.

[Saenko *et al.*, 2010] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226, 2010.

[Simonyan and Zisserman, 2015] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, pages 1106–1114, 2015.

[Tzeng *et al.*, 2014] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014.

[Tzeng *et al.*, 2015] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, pages 4068–4076, 2015.

[Tzeng *et al.*, 2017] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, pages 2962–2971, 2017.

[Venkateswara *et al.*, 2017] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5385–5394, 2017.

[Yang and Hospedales, 2017] Yongxin Yang and Timothy M. Hospedales. Deep multi-task representation learning: A tensor factorisation approach. In *ICLR*, 2017.

[Yang *et al.*, 2017] Pei Yang, Qi Tan, and Jingrui He. Multi-task function-on-function regression with co-grouping structured sparsity. In *KDD*, pages 1255–1264, 2017.

[Yang *et al.*, 2019] Pei Yang, Qi Tan, Hanghang Tong, and Jingrui He. Task-adversarial co-generative nets. In *KDD*, 2019.

[Zhang and Yeung, 2010] Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. In *UAI*, pages 733–442, 2010.

[Zhou *et al.*, 2011] Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. In *NIPS*, pages 702–710, 2011.