A Transfer Learning Method for Deep Networks with Small Sample Sizes

A Transfer Learning Method for Deep Networks with Small Sample Sizes

Xin Zheng[1], Luyue Lin[1], Shouzhi Liang[1], Bo Rao[1] and Ruidian Zhan[2]

[1] School of Automation, Guangdong University of Technology, Guangzhou, China
[2] Microelectronics Foshan Ltd., Foshan, China
Email: xinzheng9209@gmail.com

Abstract. Transfer learning is that a machine learning model learns knowledge from more than one domain, and it is applied to the context of small sample size. Some of approaches concentrate on the correlation determination among all domains while some pay more attention on knowledge transfer among all domains. In this paper, on the basic of SVM with hinge loss, a new regularized transfer learning deep network with a specific regularization is proposed, in which a deep network learns high level representation with respect to the given samples. And a part of parameters in SVM are shared such that the similarity of data distribution can be well captured. Besides, a modified regularized SVM is exploited such that the gradient based method is feasible, which yields a parallel implementation of the proposed method. After that, in the experiment part, the comparison of our approach with state-of-the-art approaches manifests the competitive performance and the feasibility in classification.
Keywords. Transfer learning; deep networks; parallelization.

1. Introduction

Classification algorithms are used in variety of areas, including images classification [1] and text categorization [2]. These classification methods are based on the assumption that all of the training data and test data are drawn independently from identically distribution [1]. The number of training samples is sufficient for us to construct a predictive classifier. However, it is worth noting that this assumption may not be feasible in applications. For example, the training samples may be not sufficient enough to construct a classifier or the existing training data are outdated [3]. The reasons behind these cases probably include the follow three folds. First, annotating data is usually an expensive labor process, and experts are not willing to annotate all images [1]. Secondly, it is extremely costly to obtain sufficient training samples, such as medical image analysis [4]. Thirdly, it is unrealistic for us to obtain sufficient training samples, like visual object tracking [5].

To address this problem, people have proposed a kind of methods, called transfer learning [1, 3]. These methods require training data is drawn from multiple domains, including target domain and related source domains. It is noteworthy that the source domain data is relative to the target domain data. The target domain data is small size. These methods exploit the training data from source domains to assist to construct the task learning in the target domain. For example, in Ref. [2], a novel transfer learning framework called TrAdaBoost has been presented, which extends Adaboost and TrAdaBoost and allows users to utilize a small amount of newly labeled data to leverage the old data to construct a high-quality classification model for the new data. Besides, some other transfer learning methods straddles both multi-task [6] and transfer learning, which is referred as multi-task transfer learning [7, 8]. In Ref. [8], Zheng et al. have proposed a multi-task-based transfer learning method with dictionary

learning. In multi-task learning, people care about the performance of each task, while in multi-task transfer learning, people care about the result of task learning in the target domain [7, 8]. In addition, some deep network model-based transfer learning methods are also studied, such as fully convolutional networks (FCN) [9], weakly-shared deep transfer networks (weakly-shared DTN) [10] and Learning structure and strength of CNN filters (SSF-CNN) [11].

In this paper, motivated by the multi-task transfer learning methods [9, 10], we propose a deep regularized transfer learning method named Dratle to solve the problem of training deep network with small sample sizes. In the proposed Dratle method, we construct a support vector machine (SVM) model for each task with respect to each domain. These SVMs are embedded in multi-task framework such that the source domain data can assist to construct the predictive SVM in the target domain. In our approach, the SVM is improved to make the SVM and the deep network can be simultaneous optimized. Besides, a regularization term is constructed for the deep network in order that the similarity of target data distribution and source data distribution can be well captured. The basic contributions of the paper can be summarized as follows:

(1) We build a revised SVM for transfer learning such that the SVM model can be optimized by gradient-based method. Moreover, the SVM model and deep network can be optimized simultaneously.

(2) We propose a regularization for deep network and shared parameter for the SVM such that the relationship between the source domain and the target domain can be well determined.

(3) We conduct experiments to investigate the performance of our proposed Dratle method. And the comparison of Dratle with existing approaches manifests the feasibility and the competitive performance in classification.

2. Related Work

2.1. Multi-task Transfer Learning

Multi-task transfer learning is that the data is generated from multiple domains, including source domains and target domain. For the existing multi-task transfer learning methods, we can summarize them into two groups, including the non-deep network-related methods and the deep network-related methods.

In non-deep network-related methods, people modify the shadow models for the transfer learning setting, including the logistic regression-based method [7, 12] and the SVM-based method [8], Bayesian method [12]. In Ref. [7], Saha et. al. proposed a multi-task transfer learning (MTTL) method to augment the data from the source domain to assist the classification task in the target domain. In Ref. [8], Zheng et. al. proposed a multi-task-based transfer learning with dictionary learning (DMTTL). In DMTTL method, the dictionary learning model is exploited to learn a discriminative sparse code to enhance the classification accuracy.

The deep network-related methods embed the deep network into the multi-task transfer learning framework. For example, in Ref. [13], Kandemir et. al. adopted a two-layer feed-forward deep Gaussian process as the task learner of source and target domains. Based on the pre-training and fine-tune strategy, some transfer learning methods have been proposed, including [9, 11, 14]. Besides, some parameter sharing methods are also proposed such as Weakly-shared DTN [10] and SSF-CNN [11]. SSF-CNN [13] is a method to learn structure and strength of CNN filters based on the pre-training model, where it fine-tunes coefficients for each filter respectively.

The proposed method is the deep network-related methods but it differs from the existing deep network-related methods. We construct the regularized deep network such that the relationship between the source domain data and the target domain data can be well determined. Besides, we construct a set of SVM models for each task with respect to each domain, which are embedded in multi-task framework. This multi-task framework yields the parameter sharing such that the source domain data can assist to construct the predictive SVM in the target domain.

2.2. Support Vector Machine

Support vector machine is firstly proposed in Ref. [15], which is served as a binary classifier. And many modifications are proposed to improve the performance of SVM such as introduction of kernel function [16]. In binary SVM, the optimal hyperplane in feature space is formulated by w and b. And the objective of SVM is

$$\begin{aligned} &\min_{\boldsymbol{w},b,\xi_i} \cdot \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_i \xi_i \\ &s.t. \cdot\ y_i(\boldsymbol{w}^T\boldsymbol{x} + b) \le 1 - \xi_i, i = 1,2,\cdots,n \\ &\quad \cdot\ \xi_i \ge 0, i = 1,2,\cdots,n \end{aligned} \tag{1}$$

where ξ_i relaxes the hard margin constrain. There are a variety of SVM extensions, for example, a regularized multi-task SVM is proposed for multi-task learning setting in Ref. [6]. For parallelization, SVM with optimizing methods based on gradient have been proposed, such as Pegasos [17], P-packSVM [18], where Pegasos [17] is a method that considers the sub-gradient for optimization and has been proposed with convergence analysis and complexity analysis. P-packSVM [18] has embraced the best known stochastic gradient descent method to optimize the primal objective which achieves a parallel implementation.

In this paper, we exploit a set of SVM models for each task with respect to each domain, which are embedded in multi-task framework. This multi-task framework yields the parameter sharing. The shared parameters are to determine the similarity among multiple domains samples and the data from source domain can assist to construct the predictive SVM in the target domain. In addition, we modify the objective function of the SVM model so that the simultaneous optimization for deep network and SVM models are available.

3. The Proposed Method

3.1. Objective Function

Assume that we are given two sets of data from two domains respectively. Namely, the source domain denoted as $\mathcal{D}_s = \mathcal{X}_s \times \mathcal{Y}_s = \{(\boldsymbol{x}_{1s}, y_{1s}), (\boldsymbol{x}_{2s}, y_{2s}), \ldots, (\boldsymbol{x}_{ns}, y_{ns})\}$, while the other one is target domain denoted as $\mathcal{D}_t = \mathcal{X}_t \times \mathcal{Y}_t = \{(\boldsymbol{x}_{1t}, y_{1t}), (\boldsymbol{x}_{2t}, y_{2t}), \ldots, (\boldsymbol{x}_{nt}, y_{nt})\}$, where $\mathcal{X}$ denotes the sample space and $\mathcal{Y}$ is the label space. As for arbitrary i−th sample $\boldsymbol{x}_{is} = [x_1, x_2, \ldots, x_d]^T \in \mathbb{R}^d$ with its labels $y_{is} \in \{-1,1\}$ from the source domain, y_{is} is 1 if and only if the label is associated with instance $\boldsymbol{x}_{is}$, otherwise y_{is} is -1. For $(\boldsymbol{x}_{it}, y_{it})$, we have the same explanation.

Given the arbitrary i−th sample, we adopt two deep networks as the non-linear feature mapping for source domains and target domains respectively so that a high-level feature representation can be achieved. Let $\psi_s(\boldsymbol{x}_{is}; \boldsymbol{\Theta}_s) = \boldsymbol{s}_{is} = [s_1, s_2, \ldots, s_{d_2}]^T \in \mathbb{R}^{d_2}$ with parameter $\boldsymbol{\Theta}_s$ denote the mapping w.r.t. the source domain. Let $\psi_t(\boldsymbol{x}_{it}; \boldsymbol{\Theta}_t) = \boldsymbol{s}_{it} = [s_1, s_2, \ldots, s_{d_2}]^T \in \mathbb{R}^{d_2}$ with parameter $\boldsymbol{\Theta}_t$ denote the mapping w.r.t. the target domain.

Given the arbitrary i-th sample from the target domain $\boldsymbol{x}_{it}$, its corresponding feature representation is $\boldsymbol{s}_{it} = \psi_t(\boldsymbol{x}_{it})$. Then the classification task in the target domain is set as SVM-based binary classifier as follow.

$$y_{it} = \begin{cases} +1, (\boldsymbol{w} + \boldsymbol{v}_t)^{\mathrm{T}}\boldsymbol{s}_{it} + b_t \ge 0 \\ -1, (\boldsymbol{w} + \boldsymbol{v}_t)^{\mathrm{T}}\boldsymbol{s}_{it} + b_t < 0 \end{cases} \tag{2}$$

where $\boldsymbol{w}$ is the shared parameter for source tasks and target task, while $\boldsymbol{v}_t$ is the specific parameter for the target task. For the classification task in the source domain is similar to (2) with the shared parameter $\boldsymbol{w}$ and specific parameter for the source task $\boldsymbol{v}_t$.

The motivation of above formulations is presented as follow. Firstly, in this method, given the sample, the deep network can learn a high-level feature representation so as to improve the classification accuracy [5]. Besides, considering the multi-task transfer learning setting, there is a relationship between the source domain and the target domain. Moreover, the classifier corresponding to each domain is of

similarity, and the shared parameter is to well capture this consistency [8]. Considering the variety of all tasks, the parameter $\boldsymbol{v}_t$and $\boldsymbol{v}_s$ are constructed to capture the own data distribution characteristic of each domain.

Besides, the similarity of data from two domains is also important, we take the regularization term $\|\bar{\boldsymbol{s}}_{it} - \bar{\boldsymbol{s}}_{is}\|^2$ into consideration. $\bar{\boldsymbol{s}}_{it}$ is the average value over all $\boldsymbol{s}_{it}$ and $\bar{\boldsymbol{s}}_{is}$ is the average value over all $\boldsymbol{s}_{is}$. The motivation is that the high similarity of $\boldsymbol{s}_{it}$ and $\boldsymbol{s}_{it}$ can help to construct the classifier w.r.t. the target task. We also exploit the l-2 norm regularization to limit the complexity of the model. The Dratle model is optimized by integrating deep networks, SVM and the regularization terms mentioned above. Then, we have the following expression.

$$\begin{aligned} \min_{\boldsymbol{w},b_s b_t,\xi_{is}\xi_{it},\boldsymbol{\theta}_s,\boldsymbol{\theta}_t,\boldsymbol{v}_t,\boldsymbol{v}_s} & \cdot \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{\lambda_t}{2}\|\boldsymbol{v}_t\|^2 + \frac{\lambda_s}{2}\|\boldsymbol{v}_s\|^2 + c\left(\sum_i \xi_{is} + \sum_i \xi_{it}\right) \\ & \cdot +\gamma_0\|\boldsymbol{\theta}_t\|_F^2 + \gamma_0\|\boldsymbol{\theta}_s\|_F^2 + \gamma_1\|\bar{\boldsymbol{s}}_{it} - \bar{\boldsymbol{s}}_{is}\|^2 \\ s.t. & \cdot y_{is}\left((\boldsymbol{w}+\boldsymbol{v}_s)^{\mathrm{T}}\boldsymbol{s}_{is} + b_s\right) \geq 1 - \xi_{is}\ \forall is \\ & \cdot y_{it}\left((\boldsymbol{w}+\boldsymbol{v}_t)^{\mathrm{T}}\boldsymbol{s}_{it} + b_t\right) \geq 1 - \xi_{it}\ \forall it \end{aligned} \tag{3}$$

where λ_s, λ_t, λ, c, γ_0 and γ_1 are the trade-off parameters to balance the effect of those respective regularizations such that all these regularizations are in the same order of magnitude.

3.2. Optimization and Pseudo-Codes

In this section, the optimization of Dratle is presented. The initial parameters are set as random values, including $\boldsymbol{w}, b_s b_t, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t, \boldsymbol{v}_t, \boldsymbol{v}_s$. And an end-to-end optimization is utilized to minimize the objective. Consider the hinge loss and the idea of mini-batch gradient descent. The objective in (3) can be written as

$$\begin{aligned} & \cdot \mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_s, \boldsymbol{v}_t, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t, b_t, b_s) \\ =& \cdot \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{\lambda_t}{2}\|\boldsymbol{v}_t\|^2 + \frac{\lambda_s}{2}\|\boldsymbol{v}_s\|^2 + \gamma_0\|\boldsymbol{\theta}_t\|_F^2 + \gamma_0\|\boldsymbol{\theta}_s\|_F^2 + \gamma_1\|\bar{\boldsymbol{s}}_{it} - \bar{\boldsymbol{s}}_{is}\|^2 \\ + & \cdot \frac{1}{|\mathcal{B}_s|}\sum_{(\boldsymbol{x}_{is}, y_{is})\in\mathcal{B}_s} \max\left(0, 1 - y_{is}\left((\boldsymbol{w}+\boldsymbol{v}_s)^{\mathrm{T}}\boldsymbol{s}_{is} + b_s\right)\right) \\ + & \cdot \frac{1}{|\mathcal{B}_t|}\sum_{(\boldsymbol{x}_{it}, y_{it})\in\mathcal{B}_t} \max\left(0, 1 - y_{it}\left((\boldsymbol{w}+\boldsymbol{v}_t)^{\mathrm{T}}\boldsymbol{s}_{it} + b_t\right)\right) \end{aligned} \tag{4}$$

where $\bar{\boldsymbol{s}}_{is} = \frac{1}{|\mathcal{B}_s|}\sum_{\boldsymbol{x}_{is}\in\mathcal{B}_s}\boldsymbol{s}_{is}$, $\bar{\boldsymbol{s}}_{it} = \frac{1}{|\mathcal{B}_t|}\sum_{\boldsymbol{x}_{it}\in\mathcal{B}_t}\boldsymbol{s}_{it}$. $\mathcal{B}_s$ denotes a mini-batch of samples drawn from $\mathcal{D}_s$ and $\mathcal{B}_t$ for $\mathcal{D}_t$. And the objective function of this method is

$$\min_{\boldsymbol{w},b_s b_t,\boldsymbol{\theta}_s,\boldsymbol{\theta}_t,\boldsymbol{v}_t,\boldsymbol{v}_s} \mathcal{L}(\boldsymbol{w}, \boldsymbol{v}_s, \boldsymbol{v}_t, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t, b_t, b_s) \tag{5}$$

Algorithm 1. Optimization of Dratle

Require: Specify the trade-off parameters λ_s, λ_t, λ, γ_0 and γ_1.
Require: Initialize the parameters in Dratle.
for all h=1:max training step **do**
  Load $\mathcal{B}_s, \mathcal{B}_t$ from two domain
  for all samples in $\mathcal{B}_s$ **do**
    Feed $\boldsymbol{x}_{is}$ into deep networks $\psi_s(\boldsymbol{x}_{is}; \boldsymbol{\theta}_s)$.
    Compute $(\boldsymbol{w}+\boldsymbol{v}_s)^{\mathrm{T}}\boldsymbol{s}_{is} + b_s$.
  end for
  for all samples in $\mathcal{B}_t$ **do**
    Feed $\boldsymbol{x}_{it}$ into deep networks $\psi_t(\boldsymbol{x}_{it}; \boldsymbol{\theta}_t)$.

9: Compute $(\boldsymbol{w}+\boldsymbol{v}_t)^{\mathrm{T}}\boldsymbol{s}_{it}+b_t$.
10: **end for**
11: Compute the $\nabla_{\widetilde{\boldsymbol{w}}}\mathcal{L}$ according to (**6**) and (**7**).
12: Compute the $\nabla_{\widetilde{\boldsymbol{v}}}\mathcal{L}$ according to (**8**) and (**9**).
13: Compute the Compute gradients with respect to $\boldsymbol{\Theta}_s, \boldsymbol{\Theta}_t$ according to(**10**), (**11**) and backpropagation algorithm.
14: Update the parameters in Dratle based on the Adam method.
15: **end for**

Given a mini-batch of samples $\mathcal{B}_s$ drawn from $\mathcal{D}_s$, and $\mathcal{B}_t$ drawn from $\mathcal{D}_t$. The gradient of $\mathcal{L}$ with respect to $\boldsymbol{w}$ represents as following expression.

$$\nabla_{\widetilde{\boldsymbol{w}}}\mathcal{L}=\boldsymbol{w}+\frac{1}{|\mathcal{B}_s\cup\mathcal{B}_t|}\sum_{\boldsymbol{x}_i\in\mathcal{B}_s\cup\mathcal{B}_t}\frac{\mathrm{d}}{\mathrm{d}\widetilde{\boldsymbol{w}}}\Delta_i \tag{6}$$

where $\Delta_i=\max(0,1-y_i\,((\boldsymbol{w}+\boldsymbol{v})^T\boldsymbol{s}_i+b)\,)$, and

$$\frac{\mathrm{d}}{\mathrm{d}\widetilde{\boldsymbol{w}}}\Delta_i=-\begin{cases}\tilde{\boldsymbol{s}}_i y_i, & y_i(\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{v}})^T\tilde{\boldsymbol{s}}_i\le 1\\ 0, & y_i(\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{v}})^T\tilde{\boldsymbol{s}}_i>1\end{cases} \tag{7}$$

Here, $\tilde{\boldsymbol{s}}_i=[\boldsymbol{s}_i^T,1]^T$, $\widetilde{\boldsymbol{v}}=[\boldsymbol{v}^T,b]^T$, and $\widetilde{\boldsymbol{w}}=[\boldsymbol{w}^T,0]^T$. $\boldsymbol{v}=\boldsymbol{v}_s$, $b=b_s$, $\boldsymbol{s}_i=\boldsymbol{s}_{is}$ if $\boldsymbol{x}_i\in\mathcal{B}_s$ and $\boldsymbol{v}=\boldsymbol{v}_t$, $b=b_t$, $\boldsymbol{s}_i=\boldsymbol{s}_{it}$ if $\boldsymbol{x}_i\in\mathcal{B}_t$. And the gradient of $\mathcal{L}$ with respect to $\boldsymbol{v}$ represents as following expression.

$$\nabla_{\widetilde{\boldsymbol{v}}}\mathcal{L}=\lambda\boldsymbol{v}+\frac{1}{|\mathcal{B}|}\sum_{\boldsymbol{x}_i\in\mathcal{B}}\frac{\mathrm{d}}{\mathrm{d}\widetilde{\boldsymbol{v}}}\Delta_i \tag{8}$$

$$\frac{\mathrm{d}}{\mathrm{d}\widetilde{\boldsymbol{v}}}\Delta_i=-\begin{cases}\tilde{\boldsymbol{s}}_i y_i, & y_i(\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{v}})^T\tilde{\boldsymbol{s}}_i\le 1\\ 0, & y_i(\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{v}})^T\tilde{\boldsymbol{s}}_i>1\end{cases} \tag{9}$$

where if $\boldsymbol{x}_i\in\mathcal{B}_s$, $\lambda=\lambda_s$, $\mathcal{B}=\mathcal{B}_s$; otherwise, if $\boldsymbol{x}_i\in\mathcal{B}_t$, $\lambda=\lambda_t$, $\mathcal{B}=\mathcal{B}_t$.

Furthermore, the gradient of $\mathcal{L}$ with respect to $\boldsymbol{s}_i$ is shown as following formulas.

$$\nabla_{\boldsymbol{s}_i}\mathcal{L}=\frac{1}{|\mathcal{B}|}\sum_{i}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{s}_i}\Delta_i \tag{10}$$

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{s}_i}\Delta_i=-\begin{cases}y_i(\boldsymbol{w}+\boldsymbol{v}), & y_i(\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{v}})^T\tilde{\boldsymbol{s}}_i\le 1\\ 0, & y_i(\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{v}})^T\tilde{\boldsymbol{s}}_i>1\end{cases} \tag{11}$$

Once the gradient of $\mathcal{L}(\boldsymbol{w},\boldsymbol{v}_s,\boldsymbol{v}_t,\boldsymbol{\Theta}_s,\boldsymbol{\Theta}_t,b_t,b_s)$ with respect to $\boldsymbol{s}_i$ is visible, we can implement the backpropagation algorithm to compute the gradient with respect to parameters in latent layers in deep networks. And finally, the gradient-based optimization method can be implemented to optimize all parameters in Dratle method. Here, we adopt the mini-batch Adam method [19] to update parameters at each iteration, where Adam is an optimizer based on gradient descent and adaptive estimates of lower-order moments. Besides, the Adam method is straightforward to implement and has computational efficiency for little space complexity. Finally, the pseudo-codes of the proposed Dratle method is as **Algorithm 1**.

For the above formulas, the optimal value of $\widetilde{w}$, $\tilde{v}$ and Θ are denoted as $\widetilde{w}^*$, $\tilde{v}^*$ and Θ^* respectively. We can conclude that the regret bound of the proposed Dratle method with Adam

optimizer is $R(T) = \sum_{t=1}^{T}[\mathcal{L} - \mathcal{L}^*] = O(\sqrt{T})$, and therefore $\frac{R(T)}{T} = O(\frac{1}{\sqrt{T}})$. Similar to works in Ref. [17], given the loss value ϵ, the complexity of runtime is $O\left(\frac{1}{\epsilon^2}\right)$.

4. Experiment

4.1. Baselines and Data Sets

In the experiment, we compared Dratle with following famous transfer learning methods, such as DMTTL [8], SVM with deep network (SVM) [20], multi-task SVM (MTSVM) [6], multi-task transfer learning method (MTTL) [7], weakly-shared deep transfer learning (WSDTL) [10]. In the above methods, deep network-related methods include SVM, WSDTL and the proposed Dratle method. The non-deep network-related methods are DMTTL, MTSVM and MTTL.

In the experiment, we study the performance of Dratle based on transfer learning data sets such as 20 Newsgroups and Reuters. The detail information about the data set is shown in table 1. The 20 Newsgroups is a popular data set for experiment in text classification. It is comprised of 20 sub-classes which is grouped into 7 classes, including comp, rec, sci, misc, talk, alt, and soc. Here, we exploit 4 classes, including comp, rec, sci and talk. Besides, these 4 classes achieve the first 3 settings in table 1.

As for Reuters data set, there are 5 classes, such as Exchanges, Orgs, People Places and Topics. Each class has a number of sub-classes. Here, we exploit three classes, including Orgs, People and Places. Also, these 3 classes achieve the second 3 settings in table 1.

Table 1. Data settings.

Settings	**Source domain**		**Target domain**	
	Positive class	Negative class	Positive class	Negative class
C v.s. R_C	**C**omp	Rec, sci, talk	**C**omp	Rec, sci, talk
R v.s. R_R	**R**ec	Comp, sci, talk	**R**ec	Comp, sci, talk
S v.s. R_S	**S**ci	Comp, rec, talk	**S**ci	Comp, rec, talk
O v.s. R_O	**O**rgs	People, Places	**O**rgs	People, Places
E v.s. R_E	P**e**ople	Orgs, Places	P**e**ople	Orgs, Places
L v.s. R_L	P**l**aces	Orgs, People	P**l**aces	Orgs, People
M0 v.s. R_M0	0 in **M**NIST	1~9 in MNIST	0 in USPS	1~9 in USPS
M9 v.s. R_M9	9 in **M**NIST	0~8 in MNIST	9 in USPS	0~8 in USPS
U0 v.s. R_U0	0 in **U**SPS	1~9 in USPS	0 in MNIST	1~9 in MNIST
U9 v.s. R_U9	9 in **U**SPS	0~8 in USPS	9 in MNIST	0~8 in MNIST

In addition, we also conduct transfer learning in the image data sets, including MNIST and USPS. These data sets are composed of digit images with ten labels from 0 to 9. Among them, MNIST and USPS are grayscale image set. Here we conduct the experiment between them. We randomly select images belong to the labels from 0 to 9 as the positive classes, while the rest classes are negative classes. These 2 classes achieve the last 4 settings in table 1.

The first six settings in table 1 have the same explanation as follows. For the first setting C v.s. R_C, the alphabet C denotes Comp class and R_C is the rest classes including rec, sci and talk. The target domain is a sub-class in comp, rec, sci and talk, and rest sub-classes are set as source domain. In the last four settings, M0 v.s. R_M0 denotes that the positive class is 0 in MINIST dataset while the rest classes 1 to 9 are set as negative class. Besides, the source domain and target domain are highlighted respectively in table 1.

4.2. Parameters Settings

In this experiment, we exploit the five-fold cross validation method to search for the optimal trade-off parameters settings. The data from the data set is normalized into a range from 0 to 1. For the baselines, we following their parameter settings, including the trade-off parameters searching interval and their parameter settings optimization methods. As for deep network-related methods, like SVM, WSDTL and the proposed Dratle method, they share the same deep network structure as shown in table 2, where fc denotes the fully connected layer.

Table 2. The deep network structures.

Sample of d-dimension	512-d fc activated by Relu	256-d fc activated by Relu	128-d fc activated by Relu	1-d fc

The settings of proposed method are presented as follows. The regularization parameters for SVMs, λ_s, λ_t, and λ, are searched in the set $\{1^{-3}, 1^{-2}, 1^{-1}, 1^0, 1^1, 1^2, 1^3, \}$. The γ_0 to regularize the deep network is searched in the set $\{1^{-5}, 1^{-4}, 1^{-3}, 1^{-2}, 1^{-1}\}$. To enhance the similarity of $\boldsymbol{s}_{it}$ and $\boldsymbol{s}_{is}$, the optimal value of γ_1 is searched in the set $\{2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^0, 2^1, 2^2, 2^3\}$.

4.3. Experiment Result

Let r denote the percentage of used training samples from the target domain. The results of all baselines with all data sets is shown in table 3. Furthermore, we implement these methods with different sizes of source training set. Besides, we use the setting, 0 v.s. R_0, with increasing the r from 0.01 to 0.5, and the accuracy is shown in figure 1. Based on these works, we have the follow four observations.

(1) The proposed Dratle method delivers the highest accuracy in most cases. For example, when r is 0.05, we can see that in the C v.s. R_C setting, the accuracy of Dratle is 83.7%. Moreover, the accuracy of O v.s. R_O is 85.7% and that of M0 v.s. R_M0 is 87.4%. The outperformance of Dratle method manifest the advance of the proposed Dratle method. The reason is that, in the Dratle method, the shared parameter and the similarity regularization as (4) works well.

(2) Deep network related methods achieve better performance than non-deep networks methods. We can see that the SVM, WSDTL, and the proposed Dratle method can achieve higher accuracy than DMTTL, MTSVM and MTTL. The reason is that, although they are fed with the same feature, these deep network-related methods are able to learn a high-level feature representation, which can improve the classification accuracy.

Table 3. Result of experiment with $r = 0.05$. Ablation experiment is conducted in the last three columns.

Settings	**DMTTL**	**MTTL**	**MTSVM**	**WSDTL**	**SVM**	**Dratle**
C v.s. R_C	0.825	0.796	0.725	0.829	0. 827	**0.837**
R v.s. R_R	0.849	0.802	0.712	0.862	0.839	**0.881**
S v.s. R_S	0.824	0.813	0.703	0.833	0.828	**0.873**
O v.s. R_O	0.770	0.698	0.585	0.798	0.807	**0.857**
E v.s. R_E	0.788	0.714	0.623	0.794	0.801	**0.829**
L v.s. R_L	0.806	0.639	0.610	0.819	0.828	**0.861**
M0 v.s. R_M0	0.759	0.735	0.698	0.775	0.795	**0.874**
M9 v.s. R_M9	0.674	0.542	0.564	0.692	0.682	**0.785**
U0 v.s. R_U0	0.571	0.514	0.503	0.584	0.586	**0.774**
U9 v.s. R_U9	0.544	0.546	0.465	0.546	0.551	**0.706**

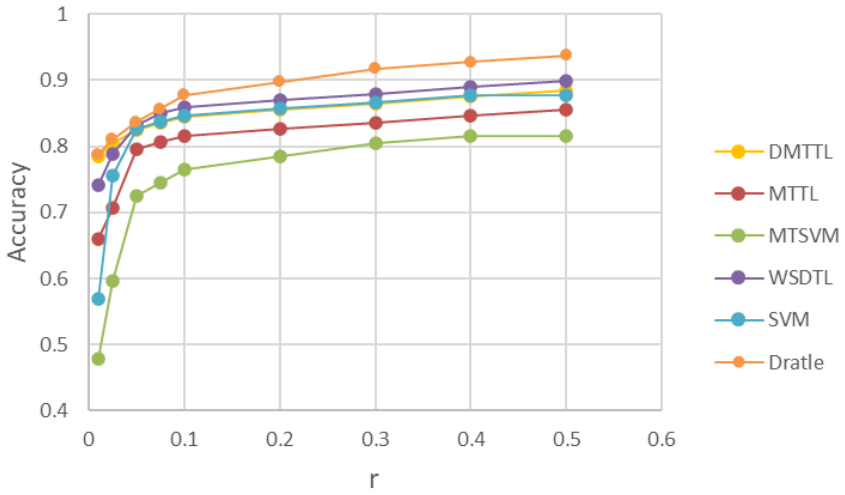

Figure 1. Accuracy curves on 0 v.s. R_0 for 6 methods.

(3) The proposed Dratle outperforms in ablation experiment. Comparing Dratle and SVM method, we can easily to conclude that the shared parameter and the similarity regularization as (4) is able to improve the performance of classification task in the target domain. The reason is that these term in (4) is able to utilize the source domain data to assist constructing a predictive classifier in the target domain. The outperformance of Dratle over WSDTL manifest the feasibility of similarity regularization in (4). The reason is that although both Dratle and WSDTL exploit the parameter sharing mechanism, but the Dratle also exploit the similarity regularization in (4) so that the relationship between the source domain and the target domain can be well determined.

(4) From figure 1, we can see that if the ratio r increases, the accuracy of the Dratle also increases. The reason behind this is that target domain data will contain more information and the classifier is able to effectively capture the target domain data distribution. These data assist to construct a more predictive classifier in the target domain and the generalization ability of target domain classifier is enhanced. In addition, with different training samples, Dratle always outperforms over other methods.

5. Conclusion and Future Work

In this paper, we proposed a multi-task transfer learning method called Dratle based on the SVM and deep network. In the proposed Dratle, we use the sharing parameter and similarity regularization method to well determine the relationship between the source domain and the target domain. We also revised the SVM so as that the gradient-based optimization method is feasible, which yields the end-to-end optimization for this transfer learning based deep network. Besides, in the experiment, the proposed method performs better in the benchmark transfer learning data set. In the future, we will pay more attention on Dratle method with outlier detection and data stream application.

Acknowledgments

This work was supported by the Key-Area Research & Development Program of Guangdong Province under Grant 2019B010142001.

References

[1] Shao L, Zhu F and Li X 2014 Transfer learning for visual categorization: A *survey IEEE Transactions on Neural Networks and Learning Systems* **26** (5) 1019-1034.

[2] Dai W, Yang Q, Xue G R and Yu Y 2007 Boosting for transfer learning *Proceedings of the 24th International Conference on Machine Learning* pp 193-200.

[3] Pan S J, Ni X, Sun J T, Yang Q and Chen Z 2010 Cross-domain sentiment classification via spectral feature alignment *Proceedings of the 19th International Conference on World Wide Web* pp 751-760.
[4] Calimeri F, Marzullo A, Stamile C and Terracina G 2017 Biomedical data augmentation using generative adversarial neural networks *International Conference on Artificial Neural Networks* (Cham: Springer) pp 626-634.
[5] Lin L, Liu B and Xiao Y 2020 COB method with online learning for object tracking *Neurocomputing* **393** 142-155.
[6] Evgeniou T and Pontil M 2004 Regularized multi-task learning *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* pp 109-117.
[7] Saha B, Gupta S, Phung D and Venkatesh S 2016 Multiple task transfer learning with small sample sizes *Knowledge and Information Systems* **46** (2) 315-342.
[8] Zheng X, Lin L, Liu B, Xiao Y and Xiong X (2020 A multi-task transfer learning method with dictionary learning *Knowledge-Based Systems* **191** 105233.
[9] Long J, Shelhamer E and Darrell T 2015 Fully convolutional networks for semantic segmentation *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* pp 3431-3440.
[10] Shu X, Qi G J, Tang J and Wang J 2015 Weakly-shared deep transfer networks for heterogeneous-domain knowledge propagation *Proceedings of the 23rd ACM International Conference on Multimedia* pp 35-44.
[11] Keshari R, Vatsa M, Singh R and Noore A 2018 Learning structure and strength of CNN filters for small sample size training *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* pp 9349-9358.
[12] Raina R, Ng A Y and Koller D 2006 Constructing informative priors using transfer learning *Proceedings of the 23rd International Conference on Machine Learning* pp 713-720.
[13] Kandemir M 2015 Asymmetric transfer learning with deep gaussian processes *International Conference on Machine Learning pp* 730-738.
[14] Bengio Y 2012 Deep learning of representations for unsupervised and transfer learning *Proceedings of ICML Workshop on Unsupervised and Transfer Learning* pp 17-36.
[15] Cortes C and Vapnik V 1995 Support-vector networks *Machine Learning* **20** (3) 273-297.
[16] Smola A J and Schölkopf B 1998 Learning with Kernels Vol 4 (GMD-Forschungszentrum Informationstechnik).
[17] Shalev-Shwartz S, Singer Y, Srebro N and Cotter A 2011 Pegasos: Primal estimated sub-gradient solver for SVM *Mathematical Programming* **127** (1) 3-30.
[18] Zeyuan A Z, Weizhu C, Gang W, Chenguang Z and Zheng C 2009 P-packSVM: Parallel primal gradient descent kernel SVM *2009 Ninth IEEE International Conference on Data Mining* pp 677-686.
[19] Kingma D P and Ba J 2014 Adam: A method for stochastic *optimization arXiv preprint arXiv:14126980*.
[20] Tang Y 2013 Deep learning using linear support vector machines *arXiv preprint arXiv:13060239.*