Safe Multiclass Transfer Learning

Abstract—**In transfer learning, domain adaptation tries to exploit the knowledge from a source domain with a plentiful data to help learn a classifier for the target domain with a different distribution and little labeled training data. In this paper, we investigate this problem under the setting of *Hypothesis Transfer Learning* (HTL) where we can only access the source model instead of the data. In real world application, this scenario is common due to many reasons such as data credential. As reported in [1], negative transfer could happen when the source and target domains are not related, especially for the multi-class scenario. As we are not able to verify the source hypotheses in HTL, negative transfer could be an important issue. In this paper, to study the problem of negative transfer in HTL, we propose a reformulation of HTL using the framework of Feature Augmentation (FA). Specifically, we first augment feature space of the target domain by adding the auxiliary features using the outputs of the models trained from the source domain. With this FA, we find the major theoretic reason that makes previous HTL work suffer from negative transfer, which is the in-proper selection of transfer parameter. To better estimate the transfer parameter, we propose a novel objective function to learn a better transfer parameter. Experiment results show that our method can alleviate negative transfer and outperform other transfer methods in the different scenarios.**

I. Introduction

The success of transfer learning suggests that exploiting the knowledge of the existing models properly can greatly help us to learn new data. Transfer learning on image recognition is a very popular topic in recent years. Domain adaptation for image recognition tries to exploit the knowledge from a source domain with a plentiful data to help learn a classifier for the target domain with a different distribution and little labeled training data. In domain adaptation, the source and target domains share the same label but their data are drawn from the different distributions.

In domain adaptation, the knowledge of the source domain can be transferred in 3 different approaches: *instance transfer*, *model transfer* and *feature representation transfer* [2]. In this paper, we focus on the method that transfers the knowledge from the source model. Some recent works show that exploiting the knowledge from the source model can boost the performance of the target model effectively. For example, Tommasi et al. [3] show that it is possible to learn a good target model with just a few positive examples. Kuzborskij et al. [4] show that leveraging the knowledge from the source models is more effective than the source data. Moreover, in some real applications, we can only obtain the source models and it is difficult to access their training data because of various reasons such as the data credential. Recently, a framework called Hypothesis Transfer Learning (HTL) [5] has been proposed to handle this situation. HTL assumes only source models (called the *hypotheses*) trained on source domain can be utilized and there is no access to source data, nor any knowledge about the relatedness of the source and target distributions. In HTL, a number of works [4] [6] have been attempted with Least Square Support Vector Machine (LS-SVM) [7] which is widely used in pattern recognition [8]. Previous approaches show that the source models can be evaluated effectively with LS-SVM via Leave-One-Out cross-validation [3].

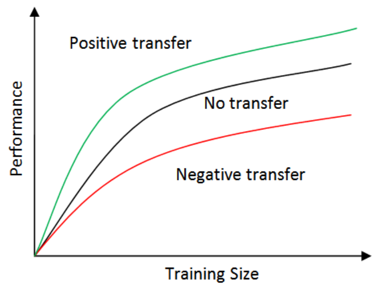


Fig. 1: Relying on the related source knowledge can improve the performance of the target model while forcing the target model to rely on the unrelated source could suffer from negative transfer.

In domain adaptation, different source domains can make different contribution to the performance of the target model. The theoretical research [1] [9] shows that the utility of the source domain decreases as the distributions of the source and target data become less similar (or *less related*) . Moreover, when the source and target tasks are not related, negative transfer may happen. In transfer learning, *negative transfer* refers to the phenomenon where the source knowledge hurts the learning process and degrades the performance of the target model comparing to a method without using any source knowledge [2]. Previous works of HTL [3] [4] [6] just consider the scenario where the target task is adding a new category to the source task (so called *from N classes to N+1 classes*). Therefore, the source and target domains are still very related. Moreover, their algorithms only focus on the performance on the newly added category, i.e. binary classification scenario, while paying less attention to the performance of the target model on all classes in the target data (the multi-class scenario). In domain adaptation, the source and target domains can be related in different levels (from strongly related to weakly related), negative transfer could happen when they are weakly related. This could be a more severe issue when we consider the performance of the target model on the whole target data (see Figure 2).

How to safely utilize the hypotheses to avoid negative transfer is still an open question in transfer learning [10].

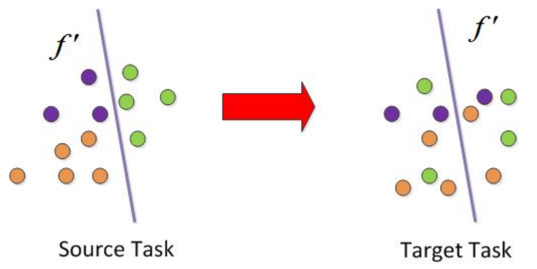


Fig. 2: Negative transfer happens when we transfer source hypothesis f' to target one. Points with different color represent different categories. The data distribution would change in different domains and negative transfer could happen when we consider the multi-class scenario.

To avoid negative transfer, we have to evaluate the utility of each hypothesis to keep useful knowledge and reject unrelated information. This approach can be achieved by setting different weights (called transfer parameters) to each hypothesis. Previous works of HTL use Leave-One-Out error to estimate the transfer parameters and avoid negative transfer [3] [4]. However, they try to solve a convex optimization problem which minimizes an upper bound of the leave-one-out error on the training set with a fixed regularization term [1]. As a result, when the source and target domains are not related, previous methods suffer from negative transfer if the regularization term is not set properly (see experiments in Section V). In this paper, we propose our method, called Safe Multiclass Transfer Learning (SMTLe), that can alleviate negative transfer and leverage correct hypotheses to improve the performance of the target model simultaneously.

The main contributions of this paper include: (1) We propose a novel algorithm SMTLe within the HTL framework that can safely utilize the hypotheses to alleviate negative transfer. We use a novel objective function with a L2 regularization term that can better estimate the transfer parameters and alleviate negative transfer. (2) We also show that by using sub-gradient descent, we can obtain the $O(\log(t)/t)$ optimal solution with t iteration.

Our framework consists of two major phases, augmenting the target data and estimating the transfer parameter. In Phase I, inspired by the previous method [11], we reformulate the previous HTL problem as a feature augmentation approach which reconstructs the target data by adding auxiliary features using the outputs of the source hypotheses. We show that with proper values for the transfer parameter, the target model can get improved performance by exploiting the knowledge from the source model and alleviate negative transfer. In Phase II, based on the closed-form leave-one-out (LOO) error for model evaluation, we propose our novel objective function that can better estimate the transfer parameter and alleviate negative transfer with the L2 regularization term. We prove that transfer parameters learned from our novel objective function can alleviate negative transfer. Moreover, we show that we can always find a $\mathcal{O}(\log(t)/t)$ optimal solution with t iterations using sub-gradient descent while previous methods are not able to get any guaranteed convergence rate, which is the main reason why they suffer from negative transfer.

In our experiment, initially, the data of the source and target domains are drawn from the same distribution (dataset). By adding the different levels of noise to the source data, we can generate several sources with different relatedness to the target domain. Experiment results show that when the source and target domain are related (no noise or very little noise is added), all the transfer methods can get improved result and our method outperforms the other baselines. As the source and target domain become less related, the baseline methods suffer from negative transfer while our method can still exploit knowledge from the source domain and the improve the performance of the target model.

The rest of this paper is organized as follow. In Section II we introduce the issues in transfer learning and some related work regarding these issues. In Section III, we reformulate the HTL in Phase I and propose our method of feature augmentation. We show that we can better analysis the performance of the transfer learning algorithm with feature augmentation. Then, we propose a novel objective function for transfer parameter estimation, called SMTLe in Section IV. We show that the estimated transfer parameter can evaluate the utility of the source hypothesis and alleviate negative transfer autonomously. In Section V, we show the performance comparison between SMTLe and other baselines on a variety of experiments on MNIST and USPS datasets.

II. Related Work

The motivation of transferring knowledge between different domains is to apply the previous information from the source domain to the target one, assuming that there exists certain relationship, explicit or implicit, between the feature space of these two domains [2]. Technically, previous work can be categorized into solving the following three issues: *what*, *how* and *when* to transfer [3].

What to transfer. Previous work tried to answer this question from three different aspects: (1) selecting transferable instances, (2) learning transferable feature representations and (3) transferable model parameters. Instance-based transfer learning assumes that part of the instances in the source domain could be re-used to benefit the learning for the target domain. Lim et al. [12] proposed a method of augmenting the training data by borrowing data from other classes for object detection. Learning transferable features means to learn common feature that can alleviate the bias of data distribution in the target domain. Recently, Long et al. [13] proposed a method that can learn transferable features with deep neural network and showed some impressive results on the benchmarks. Model transfer approach assumes that the parameters of the model for the source task can be transferred to the target task. Yang et al. [14] proposed Adaptive SVMs by transferring parameters by incorporating the auxiliary classifier

[1]In their original papers, this value is fixed to be 1. In our experiments, we found that this setting leads to degraded performance.

trained from the source domain. On top of Yang's work, Ayatar et al. [15] proposed PMT-SVM that can determine the transfer regularizer according to the target data automatically. Tommasi et al. [3] proposed Multi-KT that can utilize the parameters from multiple source models for the target classes . Kuzborskij et al. [4] proposed a similar method to learn new categories by leveraging over the known source.

When and how to transfer. The question *when to transfer* arises when we want to know if the information acquired from the previous task is relevant to the new one (i.e. in what situation, knowledge should not be transferred). *How to transfer* the prior knowledge effectively should be carefully designed to prevent inefficient and negative transfer. Some previous works [16] [17] [18] consist in using generative probabilistic method. Bayesian learning methods can predict the target domain by combining the prior source distribution to generate a posterior distribution. Alternatively, some previous max margin methods [4] [19] show that it is possible to learn from a few examples by minimizing the Leave-One-Out (LOO) error for the training model. Cawley et al. [20] show that there is a closed-form implementation of LOO cross-validation that can generate unbiased model estimation for LS-SVM.

Our work corresponds to the context above. In this paper, we propose SMTLe based on model transfer approach with LS-SVM. We address our work on how to prevent negative transfer while only the source model is accessible for domain adaptation. Compared to other works, we propose a new perspective which takes insight to negative transfer. Based on that, we propose our novel objective function and show that SMTLe can better leverage the knowledge from different source models. As a result, SMTLe can achieve a better performance and alleviate negative transfer.

III. Transfer knowledge with feature augmentation

In this section, we focus on Phase I of our framework and introduce our approach for feature augmentation. We propose a new perspective for transfer learning in HTL and analyze the reason why negative transfer could happen.

A. Feature augmentation in HTL

We define our transfer task in the following way: suppose we have N visual categories. In our source task, N source binary classifiers $f'_n(x)$ for $n = 1, ..., N$, are trained from a distribution $\mathcal{D}_s$ to distinguish whether an object belongs to each of the N categories. In our target task, we have another small dataset (x, y) drawn from another distribution $\mathcal{D}_t$ with the same N categories as those in source task. We want to train N target binary classifiers $f_n(x)$ for $n = 1, ..., N$ using this small target dataset so that they can perform well on future data of the target domain.

Vapnik et al. [11] proposed a diagram to transfer the *privileged knowledge* from the teacher to the student. The privileged knowledge is used as the auxiliary feature for training the student model. Inspired by this idea, we propose our feature augmentation approach for HTL in domain adaptation. We treat the outputs of the source models as the auxiliary features and use them to train the target model. The difference between the auxiliary feature in our work and the privileged information is that we can use the auxiliary feature in both training and testing procedures while by assumption, privileged knowledge can only be used for training.

Inspired by [11], when there are a group of N source models $F'(x) = \{f'_i(x)|i = 1, ..., n\}$, we can augment the target feature space as $\hat{x} = [\phi(x), f'_1(x), ..., f'_N(x)]$ when we want to leverage the knowledge from multiple source models (Multi-adaptation) or $\hat{x} = [\phi(x), f'_r(x)], r \in 1, ..., N$ (Single-adaptation). Here $\phi(x)$ can be any feature mapping that maps the example into another space. Thus the target binary model for category n can be represented as $f_n = \hat{w}_n\hat{x}$, where $\hat{w}_n = [w''_n, \beta_1, ..., \beta_N]$ (Multi-adaptation) or $[w''_n, \beta_r]$ (Single-adaptation). Here, we call β the *Transfer Parameter*.

$$f_n(x) = w''_n\phi(x) + \sum_k^N \beta_k w'_k\phi(x) \tag{1}$$

An intuitive interpretation of Eq. (1) is that, when the target model decides whether an object belongs to a certain category, it also considers the decisions of the source models as a reference (the second part of the right hand side in Eq. (1)). The transfer parameter β here is to control the weight of the decision from the source model, i.e. the amount of the knowledge transferred from the source domain.

The best f_n can be found by solving the following optimization problem:

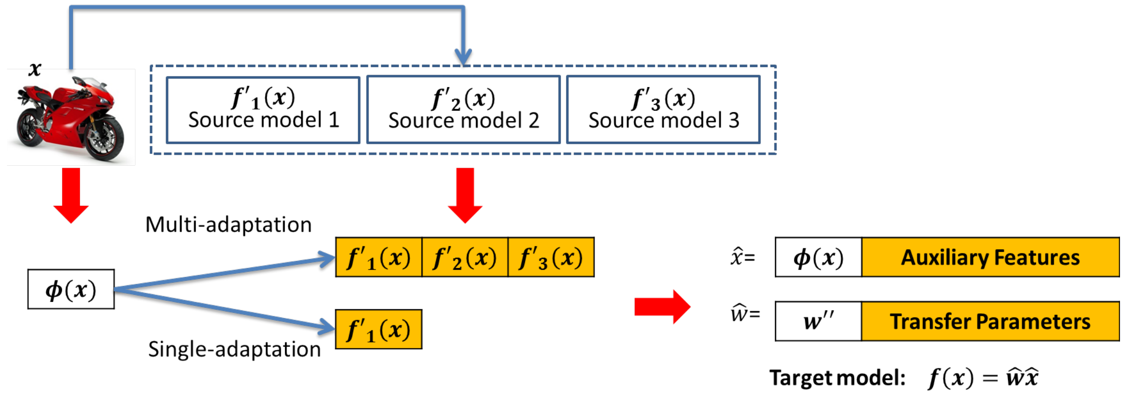
$$\textbf{min} \quad \Omega(w_n) + \frac{C}{2}\sum_i^l \mathcal{L}(f_n(x_i), Y_{in}) \tag{2}$$

Here $\Omega(w)$ is the regularization term to encourage good generalization performance and avoid overfitting. Y_{in} is the encoded label for binary classifier following $Y_{in} = 1$ if $y_i = n$ and -1 otherwise. $\mathcal{L}(\cdot)$ is the loss function. When we consider to use Least Square SVM as the classifier, we have $\mathcal{L}(f, y) = (f - y)^2$ and $\Omega(w) = \frac{1}{2}||w||^2$.

Apparently, $\hat{w}$ can be solved directly by minimizing the optimization problem (2). However, if we are not able to regularize $\hat{w}$ properly, the target model could easily suffer from negative transfer when the source and target are weakly related (see the experiment result for the method Feature+ in Section V).

Compared to the previous methods in HTL [3] [15], we have the following advantages:

- Wider range selection of the source model. Most of the previous works in HTL only support the linear source model [3] [15]. With the idea of feature augmentation, we only require the source model to be a function that can output the decision score of an example. Therefore,



x
$f'_1(x)$ Source model 1
$f'_2(x)$ Source model 2
$f'_3(x)$ Source model 3
Multi-adaptation
$\phi(x)$
$f'_1(x)$ $f'_2(x)$ $f'_3(x)$
$f'_1(x)$
Single-adaptation
$\hat{x}=$ $\phi(x)$ Auxiliary Features
$\hat{w}=$ w'' Transfer Parameters
Target model: $f(x) = \hat{w}\hat{x}$

Fig. 3: The transfer learning process can be considered as the augmentation of the target data where the decision scores of the source models are appended as the auxiliary features. The transfer parameters can be considered as a part of the corresponding hyperplane. We can consider 2 augmentation strategies: multi-adaptation and single-adaptation.

we can exploit the knowledge of the different types, such as Neural Networks and inference models [2].

- Insight into the performance of the target model. By feature augmentation, we turn the HTL domain adaptation problem into a classical learning problem and we can better analyze some important issues existed in transfer learning such as negative transfer. In the next subsection, we show the statistical analysis of how to improve the performance of the target model in our framework.

B. *Statistical analysis of the feature augmentation*

From the perspective of the feature augmentation, we can turn the problem of domain adaptation with HTL into a traditional learning problem, i.e. to find the optimal values for the elements of the hyperplane hyperplane $\hat{w} = [w''_n, \beta_1, ..., \beta_N]$ [3]. According to the principle of Structural Risk Minimization (SRM) [21], the risk of a linear classifier $f(x) = wx$ on the unseen test data $R(f)$ (generalization risk) is bounded by:

$$R(f) \leq R_{emp}(f) + \mathcal{O}\left(\sqrt{\frac{h}{l}}\right) \tag{3}$$

Here the first part on the right-hand side of the inequation $R_{emp}(f)$ is the empirical risk (training error) of the classifier and the second part is the confidence interval. h and l denote the VC dimension and number of training data of the classifier, respectively. According to [7], the VC dimension h is bounded by $h \leq \min([||w||^2R^2], m) + 1$ where R and m is the radius of the smallest ball containing data x and the dimension of x. $||w||$ is the *2-norm* of the hyperplane.

As we discussed above, we use the outputs of the source models as the auxiliary features to augment the target data. Let R and $\hat{R}$ denote the radius of the data before and after augmentation. We should have $R^2 \leq \hat{R}^2$. When the auxiliary features can't provide any useful information to reduce the empirical risk $R_{emp}(f)$, i.e. the source model is unrelated if we failed to limit $||\hat{w}||^2$, the VC dimension h will increase. Thus, the generalization risk $R(f)$ would increase and the target model suffers from negative transfer. For example, when the source models are unrelated, the optimal solution is to set the transfer parameters to 0s which is equivalent to the method without transferring any source knowledge (no transfer model). In contrast, if we can significantly decrease the empirical risk $R_{emp}(f)$ of the target model trained on the augmented data (augmented model), even though its VC dimension h increases, its generalization risk still decreases and the augmented model can get improved performance, i.e. positive transfer.

From the analysis above, we can see that in order to get improved performance and alleviate negative transfer, we should evaluate the source models and carefully set the values of the transfer parameters to control the amount of knowledge from the source models. The optimal values of the transfer parameters should be able to control the VC dimension h and reduce the empirical risk $R_{emp}(f)$ simultaneously. In next section, we introduce our method SMTLe to estimate the transfer parameters that can improve the performance of the target model and alleviate negative transfer .

IV. SMTLe

In this section, we focus on Phase II of our framework, estimating the transfer parameter. We introduce an algorithm, called SMTLe, that can effectively estimate unbiased transfer parameter from a small training set and alleviate negative transfer.

A. *Leave-One-Out estimation of LS-SVM*

In the previous section, we introduce a novel perspective for HTL by feature augmentation. We show that how to set the values of the transfer parameters can significantly affect the performance of the target model. To achieve better

[2]Previous works, such as [3] [15], can be considered as the special case of our problem where the linear model is used as the source. We can obtain similar objective function when we consider the transfer parameter as the hyper-parameter that can be defined according to the background knowledge.

[3]We take Mult-adaptation for instance. The conclusion can be applied to Single-adaptation without any modification.

performance of the target model, we have to reduce the training error and limit the VC dimension of the target model to improve its performance and alleviate negative transfer. In this part, we introduce the Leave-One-Out cross-validation (LOO-CV) error to estimate the training error of each target binary model.

As we discussed above, we have to choose the proper transfer parameters $\boldsymbol{\beta}$ to minimize the empirical risk on the target training set to exploit the source knowledge. In this paper, we choose the Leave-One-Out (LOO) cross-validation error to estimate the empirical risk for the following reasons: (1) It is proven that LOO error has a low bias on small training data regime [5]. The Leave-One-Out error is an almost unbiased estimator of the generalization error [22]. (2) For LS-SVM, we can obtain unbiased LOO-CV error in closed form which means we can estimate the values of the transfer parameters in a more efficient way.

Let $K(X,X)$ be the kernel matrix and

$$\psi = \left[K(X,X) + \frac{1}{C}\mathrm{I}\right] \tag{4}$$

The unbiased LOO estimation for sample x_i can be written as [20]:

$$\hat{Y}_{in} = Y_{in} - \frac{\alpha_{in}}{\psi_{ii}^{-1}} \quad \text{for} \quad n = 1, ..., N \tag{5}$$

Here ψ^{-1} is the inverse of matrix ψ and ψ_{ii}^{-1} is the ith diagonal element of ψ^{-1}.

Let $F'(X) = [f'_1(X), ..., f'_N(X)]$ be the output matrix of the source models and define $\boldsymbol{\alpha'}$ and $\boldsymbol{\alpha''}$ as follow:

$$\boldsymbol{\alpha'} = \psi^{-1} \times Y \quad \boldsymbol{\alpha''} = \psi^{-1} \times F'(X) \tag{6}$$

The matrix $\boldsymbol{\alpha} = \{\alpha_{in} | i = 1, ...l; n = 1, ..., N\}$ in Eq. (5) can be calculated as:

$$\boldsymbol{\alpha} = \boldsymbol{\alpha}' - \boldsymbol{\alpha}''\boldsymbol{\beta^T} \tag{7}$$

Now, we already have an effective way to measure the performance of each target binary model with different $\boldsymbol{\beta}$ for our task. In the next subsection, we expand it to the multi-class scenario to estimate the optimal transfer parameters.

B. Loss Function of SMTLe

In this subsection, we propose a novel objective function according to our multi-class prediction loss function for transfer parameter estimation. We show that we can effectively obtain the optimal $\boldsymbol{\beta}$ that alleviates negative transfer.

For the multi-class scenario, we use One-versus-all strategy to assign the label to class j if $j \equiv \arg\max_{n=1,...,N} \{f_n(x)\}$. Let us call ξ_i the multi-class prediction error for example x_i. ξ_i can be defined as [23]:

$$\xi_i(\beta) = \max_{n \in \{1,...,N\}} \left[1 - \varepsilon_{ny_i} + \hat{Y}_{in}(\beta_n) - \hat{Y}_{iy_i}(\beta_{y_i})\right] \tag{8}$$

Where $\varepsilon_{ny_i} = 1$ if $n = y_i$ and 0 otherwise. The intuition behind this loss function is to enforce the distance between the true class and other classes to be at least 1.

From Eq. (8) we can see that, different from the binary scenario where 0 is used as the hard threshold to distinguish the two classes, our multi-class loss only depends on the gap between the decision function value of the correct label ($\hat{Y}_{y_i}$) and the maximum among the decision function value of the other labels $\hat{Y}_{in}(n \neq y_i)$. To reduce ξ_i for a specific example x_i, we only have to increase the gap between $\hat{Y}_{in}(n \neq y_i)$ and $\hat{Y}_{iy_i}$.

Instead of optimizing ξ_i directly, we add the extra regularization terms for $\boldsymbol{\beta}$. Then we define our objective function as:

$$\begin{aligned} \textbf{min} \quad & \frac{\lambda}{2}\sum_{n=1}^{N}\|\beta_n\|^2 + \sum_{i=1}^{l}\xi_i \\ \textbf{s.t.} \quad & 1 - \varepsilon_{ny_i} + \hat{Y}_{in}(\beta_n) - \hat{Y}_{iy_i}(\beta_{y_i}) \leq \xi_i; \\ & \lambda_1, \lambda_2 \geq 0 \end{aligned} \tag{9}$$

Here λ is the regularization parameter. This objective function can improve the performance of the target model on the unseen test data from two aspects: improve the generalization ability by limiting the VC dimension and reduce the empirical risk compared to no transfer model.

As we discussed in Section III, regularizing the transfer parameters could improve the performance of the target model. Moreover, by adding the regularization term, the objective function (9) turns to be strongly convex. Therefore, we are able to use sub-gradient descent [24] to guarantee its convergence to be $\mathcal{O}(\log(t)/t)$ optimal in t iterations (see proof in Appendix A). This promises we can find the optimal transfer parameters effectively. We can also show that this objective function can achieve lower empirical risk compared to no transfer model (see Appendix B). This is very important when the source and target domains are not very related.

By adding a dual set of variables in objective function (9), one for each constraint in, we get the Lagrangian of the optimization problem:

$$\begin{aligned} & L(\beta, \xi, \eta) = \frac{\lambda}{2}\sum_{n=1}^{N}\|\beta_n\|^2 + \sum_{i=1}^{l}\xi_i \\ & + \sum_{i,n}\eta_{i,n}\left[1 - \varepsilon_{ny_i} + \hat{Y}_{in}(\beta_n) - \hat{Y}_{iy_i}(\beta_{y_i}) - \xi_i\right] \\ & \textbf{s.t.} \quad \forall i, n \quad \eta_{i,n} \geq 0 \end{aligned} \tag{10}$$

To obtain the optimal values for the problem above, we introduce our method using sub-gradient descent [25] and summarize it in Algorithm. 1.

V. EXPERIMENT

In this section, we show empirical results of our algorithm for different transferring situations on two image benchmark datasets: MNIST[4] [26] and USPS[5]. We test the performance of our algorithm in different scenarios and show that SMTLe can outperform the other baseline transfer methods when the source and target domains are related and alleviate negative transfer while other baseline methods suffer.

[4]http://yann.lecun.com/exdb/mnist/

[5]http://www.cs.nyu.edu/ roweis/data.html

Algorithm 1 SMTLe

```
Input: $\lambda, \psi, \alpha', \alpha'', T$,
Output: $\beta = \{\beta^1, ..., \beta^n\}$
 1: $\beta^0 \leftarrow 1$
 2: for $t = 1$ to $T$ do
 3:     $\hat{Y} \leftarrow Y - \left(\psi^{-1} \circ I\right)^{-1} (\alpha' - \alpha''\beta)$
 4:     $\Delta_\beta = 0$
 5:     for $i = 1$ to $l$ do
 6:         $\Delta_\beta \leftarrow \Delta_\beta + \lambda\beta$
 7:         $l_{ir} = \max(1 - \varepsilon_{y_i r} + \hat{Y}_{ir} - \hat{Y}_{iy_i})$
 8:         if $l_{ir} > 0$ then
 9:             $\Delta_\beta^{y_i} \leftarrow \Delta_\beta^{y_i} - \frac{\alpha''_{iy_i}}{\psi_{ii}^{-1}}$
10:             $\Delta_\beta^{r} \leftarrow \Delta_\beta^{r} + \frac{\alpha''_{ir}}{\psi_{ii}^{-1}}$
11:         end if
12:     end for
13:     $\beta^t \leftarrow \beta^{(t-1)} - \frac{\Delta_\beta}{\lambda \times t}$
14: end for
```

A. Dataset

The MNIST database [27] is a large database of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning. It was created by re-mixing the samples from NIST's original datasets. Data in MNIST were normalized to fit into a 28x28 pixel bounding box and anti-aliased, which introduced grayscale levels. In our experiment, we use a sub-set of MNIST dataset, containing 6,000 examples for 10 classes (from digit 0 to 9). We also use another handwritten digital dataset USPS [28]. USPS contains 11,000 images and the data is evenly distributed among 10 classes, i.e. 1,100 examples for each digit. Each digit is represented as a 16x16 greyscale image.

For each of the datasets, we randomly split it into 3 sets: the large source dataset (100 examples per class for both dataset) to train the source models, the small target training set (5/10/15/20/25 examples per class for both datasets) to train target model and the large target testset (4700 and 9700 examples for MNIST and USPS, respectively) to evaluate the performance of the target model.

B. Baseline methods and experiment setup

We compare our algorithm with two kinds of baselines. The first one is the methods without leveraging any prior knowledge (no transfer baselines). The second consists of some previous methods in HTL.

We select 2 no transfer baselines: **No transfer (NT):** LS-SVM trained only on target data. Any transfer algorithm that performs worse than it suffers from negative transfer. **Batch:** We combined the source and target data, assuming that we have full access to all data, to train the LS-SVM. The result of this baseline might be considered as the best performance achieved when no noise is added to the source data.

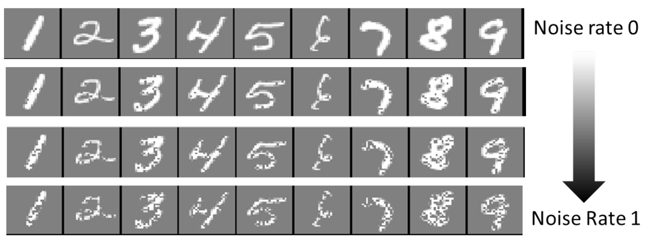


Fig. 4: We add the noise (Salt & Pepper noise) to the source data to generate the source domain with different relatedness to the target domain. From the images we can see that the source domain is still related to the target domain in different level of the noise rate

We select the 3 HTL baseline methods, **MKTL [6]**, **Multi-KT [3]**, **PMT-SVM** [15] as our transfer baselines[6]. MKTL also use feature augmentation for the target data, but with a more aggressive strategy. For Multi-KT, there are 3 different strategies, Weighted Multi-KT, Single-KT and Average-KT. We use all the 3 strategies in our experiments and denote them as MKT_m, MKT_s and MKT_a respectively. For PMT-SVM, we use grid search on $\{0.1, 0.2, ..., 1\}$ for the weights of its projection matrix and report the best performance. Also, we include the method **Feature+** discussed in Section III which solves the hyperplane $\hat{w}$ directly. For our method **SMTLe**, we implement 2 adaptation strategies single-adaptation (SMTLe_s) and multi-adaptation (SMTLe_m). For SMTLe_s, we choose the model that distinguishes the corresponding class in the source, i.e. choose the model to distinguish the class digit 1 in the source as the single source model for target binary model of the class digit 1. For SMTLe_m, we choose all the source models for each target binary model.

For all the experiments in this section, we adopt the same strategy as [4] and [3], using RBF kernels on RBF hyperparameter $\gamma = \{2^{-5}, 2^{-4}, ..., 2^8\}$. The penalty parameter C is tuned via cross-validation on $\{10^{-5}, 10^{-4}, ..., 10^8\}$ and the optimal value is reused for all the algorithms. The transfer regularization parameter λ in SMTLe is also set via cross-validation on $\{10^{-3}, 10^{-2}, ..., 10\}$ respectively. For both datasets, we use the raw pixels as the input feature.

As the baseline methods can only use the linear model as their source, in this paper, the SVM model trained from the source data is used for all the transfer methods. To generate different source models with different relatedness to the target domain. We add different levels of noise to the source training data and train the source models from the noisy source data. When there is no noise added to the source data, the source and target domain are identical (strongly related). As we add more noise, the distribution of the data in source and target domain become more different, i.e. the source and target domain become less related (see Figure 4).

[6]Algorithms are implemented by the original authors. MKTL and Multi-KT can be downloaded from http://tatianatommasi.wix.com/tatianatommasi#!3/ckra. PMT-SVM can be found from https://www.robots.ox.ac.uk/ vgg/software/tabularasa/

C. Experiment result & analysis

We perform the algorithms on different scenarios where the different levels of noise is added to the source data to train the source models. Still we use LS-SVM to train the source models. For each dataset, we use the accuracy across the 10 classes as the criterion to evaluate the performance of the algorithms. We randomly split the original datasets into 3 sets and run 10 times to report the average performance.

We show the performance of different algorithms on the two datasets under different noise levels in Table I where only 5 positive examples for each class are used in target training set. We show more results with different training sizes in the target set in Table II and Table III. From the experiments results we can see that more transfer methods suffer from negative transfer in MNIST than in USPS. This may be caused by the different data dimension (784 in MNIST VS 256 in USPS). As we discussed in Section III, if there are more parameters to be determined and the source knowledge is not leveraged properly, the target model is more likely to suffer from negative transfer. The rest of the analysis is based on the results from Table I and we can observe similar results on other tables.

From Table I we can see that when there is no noise in the source data, most transfer algorithms can leverage the knowledge of the source model. Our methods SMTLe_s and SMTLe_m achieve the best performance among all transfer methods, but still slightly underperform the Batch method which can access all the data from both source and target training data. We may expect better performances of PMT-SVM as we reduce the interval of the grid search, which is however, more computational expensive. It is worthy to notice that the MKT_m in MNIST and MKT_s in USPS suffer from negative transfer even though there is no noise in the source data. Arguably, this is caused by their less expressive manner of using the regularization term. In fact, they limit the transfer parameters within the ball of unit radius (its default setting). The results could be improved if a better regularization can be found. The results in Table I also reflects that, as we add more noise into the source data, some of the transfer algorithms start to suffer from negative transfer. It is not surprising that all methods (except for NT as no source knowledge is used) get a decreased performance. However, our methods can still outperform the other methods. The performance of MKTL in MNIST drops almost 39%. MKTL tries to use aggressive augmentation approach and there are many parameters to learn. Therefore, it is more difficult to find a good solution when the source and target domain are not very related. In our methods, We use a simpler feature augmentation approach. The optimal transfer parameters can both limit the VC dimension and reduce empirical risk at the same time. Compared to Feature+ which suffers from negative transfer when the noise level increases, our two methods can alleviate negative transfer and achieve lower generalization risk even though the source models become less related to the target domain. We can also see that there are just 3 methods that don't suffer from negative transfer in both datasets, i.e. SMTLe_s, SMTLe_m and MKT_a. However, we can see that MKT_a is a more conservative method which is reluctant to leverage the knowledge from the source model. In fact, it assigns identical weights to all source models. Therefore, it is unable to fully exploit the knowledge from the source models.

In summary, in this section, we empirically evaluate the performance of our method in different scenarios where there is different relatedness between the source and target domains. Comparing with the baseline methods, we can see that our method can effectively leverage the knowledge from the source models and alleviate negative transfer when other baseline methods suffer.

VI. Conclusion

In this paper, we present a novel method called SMTLe that is able to transfer knowledge of the source model in domain adaptation. Inspired by previous work, we propose a novel perspective on the work of HTL and show the reasons why positive and negative transfer would happen in the different scenarios. Based on our analysis, we propose our method SMTLe that can safely leverage the knowledge from the source models to achieve the improved target model performance by limiting the VC dimension of the transfer problem and reduce the empirical risk as well. Experiment results show that SMTLe can leverage related source knowledge and alleviate negative transfer in different scenarios and outperform other baseline methods.

In our perspective on the domain adaptation problem, the feature augmentation approach can fit a wider range of source classifiers. We can leverage the knowledge from any source model that can output the decision score/confidence, such as the Neural Networks and the inference model. Meanwhile, there are still many open issues to solve before we could maximize the utility of different kinds of source classifiers. For example, how to better exploit the knowledge from a deep neural network with our feature augmentation framework and achieve good positive transfer performance and avoid negative transfer simultaneously. These challenges could lead to our future interest.

Acknowledgment

The authors would like to thank...

Appendix A
Convergence Analysis

Theorem 1.

$$L(\beta_{T+1}) \leq L(\beta^*) + \frac{G^2(1+\ln(T))}{2\lambda T} \tag{11}$$

Proof. As $L(\beta)$ is strongly convex and Δ_t is in its sub-gradient set at β_t, according to the definition of λ-strong convexity [29], the following inequality holds:

$$\langle \beta_t - \beta^*, \Delta_t \rangle \geq L(\beta_t) - L(\beta^*) + \frac{\lambda}{2}||\beta_t - \beta^*||^2 \tag{12}$$

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	**87.23**	**86.40**	**85.81**	**84.63**	**81.51**	**79.67**	**78.10**
SMTLe_m	82.94	80.20	79.05	76.56	72.10	69.57	68.09
MKT_m	61.96*	57.14*	55.18*	50.67*	45.16*	43.72*	42.73*
MKT_s	86.63	82.33	79.78	74.61	67.51	65.52	64.06
MKT_a	62.94	63.04	63.03	62.91	62.93	62.98	62.89
MKTL	70.06	55.63*	60.42*	41.57*	40.44*	42.22*	31.44*
PMT	63.54	63.54	63.54	63.54	63.54	63.54	63.54
Feature+	66.99	61.72*	59.13*	54.27*	46.46*	44.44*	42.67*
NT	62.87	62.87	62.87	62.87	62.87	62.87	62.87
Batch	87.39	84.42	82.40	76.83	62.86*	57.33*	51.16*

(a) Results on MNIST

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	**91.12**	89.79	89.23	88.06	86.22	85.33	84.60
SMTLe_m	90.78	**89.85**	**89.42**	**88.55**	**86.80**	**85.96**	**84.91**
MKT_m	86.80	84.80	83.57	81.02	75.96	74.53*	72.73*
MKT_s	64.18*	61.39*	61.80*	62.93*	64.85*	65.29*	65.55*
MKT_a	75.76	75.76	75.75	75.79	75.75	75.78	75.84
MKTL	90.24	88.13	86.20	86.07	81.82	80.18	80.14
PMT	75.89	75.89	75.9	75.88	75.88	75.88	75.87
Feature+	88.42	86.56	85.28	83.23	79.79	78.68	77.17
NT	75.75	75.75	75.75	75.75	75.75	75.75	75.75
Batch	91.65	90.38	89.58	87.36	82.55	80.52	77.84

(b) Results on USPS

TABLE I: Experiment results on 5 examples per class in target training set. We show the percentage of accuracy across the 10 classes on different noise level. We use "*" to denote the results suffer from negative transfer

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	**88.13**	**86.98**	**86.62**	**85.44**	**84.26**	**83.61**	**82.79**
SMTLe_m	87.29	85.20	83.81	81.47	77.74	76.45	76.26
MKT_m	65.01*	61.51*	59.02*	53.92*	49.24*	48.34*	47.51*
MKT_s	87.82	85.88	84.00	80.85	75.87	73.81	72.30
MKT_a	72.37	72.32	72.36	72.30	72.32	72.32	72.25
MKTL	79.63	68.80*	69.55*	59.74*	52.04*	50.47*	42.47*
Feature+	77.60	73.33	70.73*	64.90*	56.86*	54.40*	52.3*
PMT	72.86	72.87	72.87	72.87	72.86	72.86	72.87
NT	72.22	72.23	72.23	72.23	72.23	72.23	72.23
Batch	87.46	84.78	82.96	78.04	65.96*	60.97*	55.65*

(a) 10 examples per class

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	88.63	**87.52**	**87.12**	**86.33**	**85.57**	**85.04**	**84.65**
SMTLe_m	**88.92**	86.98	85.99	84.01	80.69	80.08	79.28
MKT_m	67.03*	63.34*	61.31*	56.26*	51.76*	50.72*	49.69*
MKT_s	88.08	86.31	85.40	83.38	79.52	77.83	77.04
MKT_a	76.19	76.16	76.19	76.12	76.14	76.16	76.11
MKTL	83.75	74.27*	77.46	66.39*	61.57*	60.0*	55.28*
Feature+	81.87	78.54	76.63	72.25*	64.45*	61.88*	59.67*
PMT	76.78	76.78	76.78	76.79	76.78	76.78	76.78
NT	76.09	76.10	76.10	76.10	76.10	76.10	76.10
Batch	87.72	85.20	83.57	79.29	68.95*	64.36*	59.8*

(b) 15 examples per class

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	88.87	87.81	**87.28**	**86.50**	**85.98**	**85.64**	**85.27**
SMTLe_m	**89.39**	**87.97**	87.06	85.47	82.63	81.95	81.32
MKT_m	68.03*	64.37*	62.7*	58.98*	54.41*	53.52*	52.93*
MKT_s	88.04	86.42	85.58	83.69	81.29	80.66	79.56
MKT_a	78.68	78.64	78.65	78.62	78.63	78.63	78.60
MKTL	86.75	81.14	82.46	72.57*	65.4*	69.53*	61.56*
Feature+	83.80	80.99	79.29	75.54*	68.46*	66.12*	63.97*
PMT	78.43*	78.43*	78.44*	78.44*	78.43*	78.43*	78.43*
NT	78.58	78.59	78.59	78.60	78.60	78.59	78.60
Batch	87.80	85.41	83.89	80.10	71.22*	67.35*	63.28*

(c) 20 examples per class

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	89.22	88.16	87.67	**86.94**	**86.46**	**86.26**	**85.95**
SMTLe_m	**89.69**	**88.7**	**87.72**	86.33	83.95	83.23	82.73
MKT_m	69.15*	66.04*	64.23*	60.71*	56.0*	54.9*	54.53*
MKT_s	88.21	86.6	85.87	84.14	82.02	81.51	81.04
MKT_a	80.35	80.33	80.36	80.33	80.33	80.33	80.31
MKTL	87.50	84.39	81.88	72.97*	70.57*	70.17*	62.88*
Feature+	85.23	82.76	81.21	77.66*	71.59*	69.39*	67.48*
PMT	79.58*	79.59*	79.59*	79.59*	79.59*	79.59*	79.58*
NT	80.27	80.29	80.28	80.29	80.29	80.29	80.29
Batch	88.02	85.80	84.33	80.92	73.36*	69.99*	66.27*

(d) 25 examples per class

TABLE II: Results on MNIST with 10/15/20/25 positive examples for each class

For the term $\langle \beta_t - \beta^*, \Delta_y \rangle$, it can be written as:

$$\begin{aligned}\langle \beta_t - \beta^*, \Delta_t \rangle &= \left\langle \beta_t - \frac{1}{2}\eta_t \Delta_t + \frac{1}{2}\eta_t \Delta_t - \beta^*, \Delta_t \right\rangle \\ &= \frac{1}{2}\left\langle [(\beta_t - \eta_t\Delta_t) - \beta^*] + (\beta_t - \beta^*) + \eta_t\Delta_t, \Delta_t \right\rangle \\ &= \frac{1}{2}\left\langle (\beta_{t+1} - \beta^*) + (\beta_t - \beta^*), \Delta_t \right\rangle + \frac{1}{2}\eta_t \Delta_t^2 \\ &= \frac{1}{2}\left\langle \beta_{t+1} + \beta_t - 2\beta^*, \Delta_t \right\rangle + \frac{1}{2}\eta_t \Delta_t^2\end{aligned} \tag{13}$$

Then we have:

$$\begin{aligned}||\beta_t - \beta^*||^2 - ||\beta_{t+1} - \beta^*||^2 &= (\beta_t - \beta_{t+1})(\beta_t + \beta_{t+1} - 2\beta^*) \\ &= \langle \beta_{t+1} + \beta_t - 2\beta^*, \eta_t \Delta_t \rangle\end{aligned} \tag{14}$$

Using the assumption $||\Delta_t|| \le G$, we can rearrange (12) and plug (13) and (14) into it, we have:

$$\begin{aligned}&Diff_t = L(\beta_t) - L(\beta^*) \\ &\le \frac{||\beta_t - \beta^*||^2 - ||\beta_{t+1} - \beta^*||^2}{2\eta_t} - \frac{\lambda}{2}||\beta_t - \beta^*||^2 + \frac{1}{2}\eta_t \Delta_t^2 \\ &\le \frac{||\beta_t - \beta^*||^2 - ||\beta_{t+1} - \beta^*||^2}{2\eta_t} - \frac{\lambda}{2}||\beta_t - \beta^*||^2 + \frac{1}{2}\eta_t G^2 \\ &= \frac{\lambda(t-1)}{2}||\beta_t - \beta^*||^2 - \frac{\lambda t}{2}||\beta_{t+1} - \beta^*||^2 + \frac{1}{2}\eta_t G^2\end{aligned} \tag{15}$$

Due to the strong convexity, for each pair of $L(\beta_t)$ and $L(\beta_{t+1})$ for $t = 1, ..., T$, according to (12), we have:

$$\begin{aligned}L(\beta_{t+1}) - L(\beta_t) &\le \langle \beta_{t+1} - \beta_t, \Delta_t \rangle - \frac{\lambda}{2}||\beta_{t+1} - \beta_t||^2 \\ &= -\eta_t \Delta_t^2 (1 - \frac{1}{2t}) \le 0\end{aligned} \tag{16}$$

Therefore, we have the following sequence $L(\beta^*) \le L(\beta_T) \le L(\beta_{T-1}) \le ... \le L(\beta_1)$. For the sequence $Diff_t$ for $t =$

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	**91.12**	89.79	89.23	88.06	86.22	85.33	84.60
SMTLe_m	90.78	**89.85**	**89.42**	**88.55**	**86.80**	**85.96**	**84.91**
MKT_m	86.80	84.80	83.57	81.02	75.96	74.53*	72.73*
MKT_s	64.18*	61.39*	61.80*	62.93*	64.85*	65.29*	65.55*
MKT_a	75.76	75.76	75.75	75.79	75.75*	75.78	75.84
MKTL	90.24	88.13	86.20	86.07	81.82	80.18	80.14
Feature+	88.42	86.56	85.28	83.23	79.79	78.68	77.17
PMT	75.89	75.89	75.90	75.88	75.88	75.88	75.87
NT	75.75	75.75	75.74	75.76	75.76	75.76	75.74
Batch	91.65	90.38	89.58	87.36	82.55	80.52	77.84

(a) 10 examples per class

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	**91.58**	90.58	89.84	88.82	86.82	86.28	85.45
SMTLe_m	91.46	**90.72**	**90.29**	**89.61**	**88.22**	**87.89**	**87.19**
MKT_m	89.06	87.98	87.31	85.46	81.94	80.29	78.51*
MKT_s	74.12*	71.44*	71.63*	72.16*	73.36*	73.56*	73.56*
MKT_a	79.57	79.57	79.55	79.58	79.57*	79.59	79.58
MKTL	88.74	89.45	88.86	87.63	84.53	82.30	84.41
Feature+	89.98	88.6	87.64	85.97	83.30	82.36	81.00
PMT	79.56	79.54*	79.54	79.55*	79.56*	79.55*	79.55
NT	79.55	79.56	79.54	79.57	79.57	79.58	79.54
Batch	91.75	90.61	89.88	88.04	84.25	82.69	80.80

(b) 15 examples per class

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	91.95	91.18	90.64	89.55	87.80	87.16	86.46
SMTLe_m	**92.01**	**91.25**	**90.79**	**90.20**	**89.03**	**88.70**	**88.30**
MKT_m	89.90	89.05	88.64	87.12	82.78	81.21*	79.44*
MKT_s	79.39*	76.88*	76.84*	77.31*	78.02*	78.13*	78.03*
MKT_a	81.92	81.92	81.92	81.93	81.94*	81.94	81.91
MKTL	90.67	89.08	89.31	88.84	85.26	85.64	85.03
Feature+	90.95	89.66	88.89	87.49	85.03	84.14	83.06
PMT	81.49*	81.48*	81.48*	81.50*	81.51*	81.49*	81.50*
NT	81.89	81.92	81.87	81.91	81.94	81.93	81.89
Batch	91.91	90.85	90.20	88.55	85.40	84.21	82.76

(c) 20 examples per class

	Noise Level						
	0.0	0.2	0.3	0.5	0.8	0.9	1.0
SMTLe_s	92.18	91.43	90.93	89.97	88.41	87.83	87.22
SMTLe_m	**92.27**	**91.68**	**91.29**	**90.62**	**89.58**	**89.17**	**88.81**
MKT_m	90.35	89.67	89.35	88.08	84.91	83.27*	81.37*
MKT_s	82.40*	80.24*	80.12*	80.46*	80.93*	80.98*	80.83*
MKT_a	83.69	83.70	83.66	83.70	83.71	83.72	83.69
MKTL	91.55	90.46	90.01	88.49	87.36	86.47	87.02
Feature+	91.38	90.26	89.56	88.38	86.42	85.63	84.67
PMT	82.89*	82.87*	82.88*	82.88*	82.89*	82.88*	82.9*
NT	83.66	83.69	83.65	83.68	83.71	83.70	83.65
Batch	92.11	91.08	90.50	89.06	86.34	85.30	84.27

(d) 25 examples per class

TABLE III: Results on USPS with 10/15/20/25 positive examples for each class

$1, ..., T$, we have:

$$\sum_{t=1}^{T} Diff_t = \sum_{t=1}^{T} L(\beta_t) - TL(\beta^*) \geq T\left[L(\beta_T) - L(\beta^*)\right] \tag{17}$$

Next, we show that

$$\begin{aligned} &\sum_{t=1}^{T} Diff_t = \\ &\sum_{t=1}^{T} \left\{ \frac{\lambda(t-1)}{2} \|\beta_t - \beta^*\|^2 - \frac{\lambda t}{2} \|\beta_{t+1} - \beta^*\|^2 + \frac{1}{2}\eta_t G^2 \right\} \\ &= -\frac{\lambda T}{2} \|\beta_{T+1} - \beta^*\|^2 + \frac{G^2}{2\lambda} \sum_{t=1}^{T} \frac{1}{t} \\ &\leq \frac{G^2}{2\lambda} \sum_{t=1}^{T} \frac{1}{t} \leq \frac{G^2}{2\lambda}(1 + \ln(T)) \end{aligned} \tag{18}$$

Combining (17) and rearranging the result, we have:

$$L(\beta_{T+1}) \leq L(\beta^*) + \frac{G^2(1 + \ln(T))}{2\lambda T}$$

□

APPENDIX B
REDUCED TRAINING ERROR

Assume that $\bar{\xi}_i$ is the multi-class loss of example x_i without utilizing any prior knowledge, i.e. $\beta = \mathbf{0}$. Let β^* be the optimal solution for Eq. (10) and ξ_i^* be the multi-class loss with respective to example x_i. Then for every example $x_i \in \mathcal{X}$, we have:

$$\sum_i \xi_i^* \leq \sum_i \bar{\xi}_i$$

Proof. When $\beta = \mathbf{0}$, from Eq. (8) we can get:

$$\bar{\xi}_i = \max_n \left[\varepsilon_{ny_i} - 1 + \frac{({\alpha'}_{iy_i} - {\alpha'}_{in})}{\psi_{ii}^{-1}}\right]$$

For simplification, let $\delta_i = 1$ if $i = N+1$ and 0 otherwise, and $\theta_{ij} = \alpha''_{ij}\left(1 - \delta_j\right)/\psi_{ii}^{-1}$. To find the minimum of the primal problem, we require:

$$\frac{\partial L}{\partial \xi_i} = 1 - \sum_n \eta_{in} = 0 \Rightarrow \sum_n \eta_{in} = 1 \tag{19}$$

$$\frac{\partial L}{\partial \beta_n} = 0 \Rightarrow \beta_n^* = \frac{1}{\lambda} \sum_{i,n} \frac{\eta_{in}{\alpha''}_{in}}{\psi_{ii}^{-1}} \left(\delta_{y_i} - \delta_n\right) \tag{20}$$

As the strong duality holds,the primal and dual objectives coincide. Plug Eq. (20) into Eq. (10), we have:

$$\sum_{i,n} \eta_{in} \left[1 - \varepsilon_{ny_i} + \hat{Y}_{in}\left(\beta_n^*\right) - \hat{Y}_{iy_i}\left(\beta_{y_i}^*\right) - \xi_i^*\right] = 0$$

Expand the equation above, we have:

$$\sum_{i,n} \eta_{in} \left[\varepsilon_{n,y_i} - 1 + \frac{({\alpha'}_{iy_i} - {\alpha'}_{in})}{\psi_{ii}^{-1}} - \xi_i\right] = \lambda \sum_r \|\beta_r^*\|^2 \geq 0$$

Rearranging the above, we obtain:

$$\sum_{i,n} \eta_{in} \left[\varepsilon_{n,y_i} - 1 + \frac{({\alpha'}_{iy_i} - {\alpha'}_{in})}{\psi_{ii}^{-1}}\right] \geq \sum_{i,n} \eta_{in}\xi_i = \sum_i \xi_i \tag{21}$$

The left-hand side of Inequation (21) can be bounded by:

$$\begin{aligned}
&\sum_{i,n} \eta_{in} \left[\varepsilon_{ny_i} - 1 + \frac{(\alpha'_{iy_i} - \alpha'_{in})}{\psi_{ii}^{-1}} \right] \\
&\leq \sum_i \left(\sum_n \eta_{in} \max_r \left\{ \varepsilon_{ry_i} - 1 + \frac{(\alpha'_{iy_i} - \alpha'_{ir})}{\psi_{ii}^{-1}} \right\} \right) \\
&= \sum_i \left(\sum_n \eta_{in} \bar{\xi}_i \right) = \sum_i \bar{\xi}_i \qquad (22)
\end{aligned}$$

□

References

[1] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, no. 10, pp. 1345–1359, 2010.

[3] T. Tommasi, F. Orabona, and B. Caputo, "Learning categories from few examples with multi model knowledge transfer," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 5, pp. 928–941, 2014.

[4] I. Kuzborskij, F. Orabona, and B. Caputo, "From n to n+ 1: Multiclass transfer incremental learning," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 3358–3365.

[5] I. Kuzborskij and F. Orabona, "Stability and hypothesis transfer learning," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 942–950.

[6] L. Jie, T. Tommasi, and B. Caputo, "Multiclass transfer learning from unconstrained priors," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1863–1870.

[7] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[8] C. M. Bishop, "Pattern recognition," *Machine Learning*, 2006.

[9] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira *et al.*, "Analysis of representations for domain adaptation," *Advances in neural information processing systems*, vol. 19, p. 137, 2007.

[10] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowledge-Based Systems*, vol. 80, pp. 14 – 23, 2015, 25th anniversary of Knowledge-Based Systems.

[11] V. Vapnik and R. Izmailov, "Learning using privileged information: Similarity control and knowledge transfer," *Journal of Machine Learning Research*, vol. 16, pp. 2023–2049, 2015.

[12] J. J. Lim, "Transfer learning by borrowing examples for multiclass object detection," Ph.D. dissertation, Massachusetts Institute of Technology, 2012.

[13] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France*, 2015, pp. 97–105.

[14] J. Yang, R. Yan, and A. G. Hauptmann, "Cross-domain video concept detection using adaptive svms," in *Proceedings of the 15th international conference on Multimedia*. ACM, 2007, pp. 188–197.

[15] Y. Aytar and A. Zisserman, "Tabula rasa: Model transfer for object category detection," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2252–2259.

[16] J. Davis and P. Domingos, "Deep transfer via second-order markov logic," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 217–224.

[17] X. Wang, T.-K. Huang, and J. Schneider, "Active transfer learning under model shift," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1305–1313.

[18] T. Zhou and D. Tao, "Multi-task copula by sparse graph regression," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 771–780.

[19] T. Tommasi, F. Orabona, and B. Caputo, "Safety in numbers: Learning categories from few examples with multi model knowledge transfer," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3081–3088.

[20] G. C. Cawley, "Leave-one-out cross-validation based model selection criteria for weighted ls-svms," in *Neural Networks, 2006. IJCNN'06. International Joint Conference on*. IEEE, 2006, pp. 1661–1668.

[21] V. N. Vapnik, "An overview of statistical learning theory," *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp. 988–999, 1999.

[22] A. Elisseeff, M. Pontil *et al.*, "Leave-one-out error and stability of learning algorithms with applications," *NATO science series sub series iii computer and systems sciences*, vol. 190, pp. 111–130, 2003.

[23] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *The Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2002.

[24] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[25] ——, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[26] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 951–958.

[27] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[28] J. J. Hull, "A database for handwritten text recognition research," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 5, pp. 550–554, 1994.

[29] R. T. Rockafellar, *Convex analysis*. Princeton university press, 2015.