Boosting for transfer learning with multiple sources

Yi Yao Gianfranco Doretto
Visualization and Computer Vision Lab, GE Global Research, Niskayuna, NY 12309
yaoy@ge.com doretto@research.ge.com

Abstract

Transfer learning allows leveraging the knowledge of source domains, available a priori, to help training a classifier for a target domain, where the available data is scarce. The effectiveness of the transfer is affected by the relationship between source and target. Rather than improving the learning, brute force leveraging of a source poorly related to the target may decrease the classifier performance. One strategy to reduce this negative transfer *is to import knowledge from multiple sources to increase the chance of finding one source closely related to the target. This work extends the boosting framework for transferring knowledge from multiple sources. Two new algorithms,* MultiSource-TrAdaBoost*, and* TaskTrAdaBoost*, are introduced, analyzed, and applied for object category recognition and specific object detection. The experiments demonstrate their improved performance by greatly reducing the negative transfer as the number of sources increases.* TaskTrAdaBoost *is a fast algorithm enabling rapid retraining over new targets.*

1. Introduction

A common assumption of traditional machine learning algorithms is that the probability distributions of the training and testing data are the same. Under such an assumption, when presented with a new set of data with a different distribution, training samples need to be collected for learning new classifiers. Let us consider a classic computer vision problem, such as object category recognition, which is known to require a large number of training samples to ensure good generalization [10]. When tasked with the problem of recognizing a new object category, new training data has to be collected and labeled so as to represent the new distribution. However, one would save time if he/she could leverage useful information from existing annotated data and/or classifiers of old object categories.

In addition, different reasons may impede easy access to new data, and only a small number of samples may be available. Training a new classifier under such conditions would dramatically increase the risk of overfitting the new data, leading to poor generalization. It would be more efficient if one could regularize the learning in this scenario by exploiting the knowledge previously accumulated from similar problems.

Transfer learning [16, 22] represents a family of algorithms that relaxes the identical distribution assumption of the traditional machine learning approach. As the name suggests, transfer learning algorithms leverage and transfer informative knowledge from old data domains (*sources*) to a new data domain (*target*). The transferred knowledge helps improving the learning in the target domain when the training samples are scarce. Among the works in this area *TrAdaBoost* [4] is becoming a popular boosting based algorithm that is most closely related to our work.

In general, the ability to transfer knowledge from a source to a target depends on how they are related. The stronger the relationship, the more usable will be the previous knowledge. On the other hand, brute force transferring in case of weak relationships may lead to performance deterioration of the resulting classifier. This is known as *negative transfer*. In order to avoid this effect one would have to answer the question “when to transfer”. Limited work has been done in this area [16]. One strategy to decrease the risk for negative transfer is to import knowledge not from one, but from multiple sources. In this way, the chance to borrow beneficial knowledge closely related to the target domain significantly increases. From another point of view, this implies that answering the question of “when to transfer” becomes less important.

TrAdaBoost relies only on one source, and therefore is intrinsically vulnerable to negative transfer. This work formally states the problem of transfer learning from multiple sources to improve the training of a target classifier (Section 3). Two boosting based approaches addressing this problem are proposed. The first one, called *MultiSource-TrAdaBoost*, extends the *TrAdaBoost* framework for handling multiple sources (Section 4). The second one, called *Task-TrAdaBoost*, introduces a training process with two phases (Section 5). One is dedicated to the summarization of the knowledge from multiple sources. The other one is devoted to transferring knowledge to the target, and has a very low time complexity, which enables rapid retraining when presented with a new target. The theoretical performance of

978-1-4244-6985-7/10/$26.00 ©2010 IEEE

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.

these two algorithms, and their dynamic behavior, are discussed in relation to *AdaBoost* (Section 6). The algorithms are general, and have the potential for significantly improving the performance of several computer vision applications. The approaches are deployed and evaluated within the context of object category recognition, and specific object detection (Section 7). A thorough comparison against *TrAdaBoost*, and the baseline traditional machine learning algorithm *AdaBoost*, is also provided.

2. Related work

Transfer learning has been deployed over a wide variety of applications, such as sign language recognition [7], text classification [25], WiFi localization [20], and adaptive updating of land-cover maps [3]. The approaches to transfer learning are categorized based on the means used for importing knowledge from source to target, and can be based on *instance-transfer* [4, 1], *feature-representation-transfer* [17, 24], *parameter-transfer* [8, 2, 19], and *relational-knowledge-transfer*. For more details we refer the reader to the following surveys: [16, 22].

In the instance-transfer approach samples from the source are directly applied for training the target classifier. *TrAdaBoost* [4] falls into this category, as well as *MultiSourceTrAdaBoost*, the first proposed approach. In feature-representation-transfer the focus is to find a representation of the feature space that minimizes the differences between source and target. Parameter-transfer approaches assume that the target could share parameters with a related source. *TaskTrAdaBoost*, the second proposed method, falls into this category. Relational-knowledge-transfer assumes that data within the source is correlated and the goal is to export this correlation to the target [16].

In addition to these approaches, [17] applied the sparse prototype representation to transductive transfer learning, where no labeled data in the target domain is available. [15] presented a learning method where high-level semantic attributes, describing shape and color, are exploited to transfer knowledge from multiple sources to the target.

Support vector machines (SVM) have been modified for transfer learning. In [27] an SVM is derived by adjusting existing classifiers according to the target data. [14] derived more adaptable decision boundaries by training a target SVM with the help of weighted support vectors learned from multiple sources. [6] performed video concept detection by learning an SVM where the kernel is derived by minimizing the distribution mismatch between the labeled source data and the unlabeled target data.

Although SVM-based transfer learning has been extended to leverage knowledge from more than one source, to the best of the knowledge of the authors, this is the first work that extends boosting-based transfer learning to multiple sources. Some related work has extended boosting for multi-task learning [26] and on-line incremental learning [12]. In this context, [21] presented a multi-class boosting-based classification framework that jointly selects weak classifiers shared among different tasks. In contrast, here the interested is in boosting a single target classifier by leveraging the (instance-based, or parameter-based) knowledge transferrable from multiple sources. Also, [21] assumes a comparable number of training samples for every task. In contrast, the proposed method focuses on scenarios where target training samples are scarce.

3. Problem statement

In this section we introduce some notation and define the type of transfer learning problem we intend to approach. Formally, a *domain* $\mathcal{D}$ is made of a feature space $\mathcal{X}$, and a marginal probability distribution $P(X)$, where $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$, and $\mathbf{x}_i \in \mathcal{X}$. A *task* $\mathcal{T}$ is made of a label space $\mathcal{Y} \doteq \{+1, -1\}$, and a boolean function $f : \mathcal{X} \rightarrow \mathcal{Y}$. Learning the task $\mathcal{T}$ for the domain $\mathcal{D}$, in traditional machine learning, amounts to estimating a classifier function $\hat{f} : \mathcal{X} \rightarrow \mathcal{Y}$, from the given *training data* $D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n) | \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$, that best approximates f, according to certain criteria.

Let us now indicate with $\mathcal{D}_T \doteq (\mathcal{X}, P_T(X))$ a *target domain* for which we would like to learn the *target task* $\mathcal{T}_T \doteq (\mathcal{Y}, f_T)$, from the *target training data* $D_T = \{(\mathbf{x}_1^T, y_1^T), \cdots, (\mathbf{x}_{n_T}^T, y_{n_T}^T)\}$. Let us also indicate with $\mathcal{D}_S \doteq (\mathcal{X}, P_S(X))$ a *source domain*, and with $\mathcal{T}_S \doteq (\mathcal{Y}, f_S)$ a *source task*, for which we have available the *source training data* $D_S = \{(\mathbf{x}_1^S, y_1^S), \cdots, (\mathbf{x}_{n_S}^S, y_{n_S}^S)\}$. Improving the learning of the target classifier function $\hat{f}_T : \mathcal{X} \rightarrow \mathcal{Y}$ by exploiting the knowledge of the source task $\mathcal{T}_S$, in the source domain $\mathcal{D}_S$, is a so-called *inductive transfer learning* problem [16, 22].

Performing inductive transfer learning, as opposed to traditional machine learning, should be advantageous in the cases when the size of the target training data D_T, is very small in absolute terms, and also relative to the size of the source training data D_S, i.e. $n_T \ll n_S$. In fact, under such conditions, traditional machine learning would suffer from serious overfitting problems. From here comes the idea of attempting to regularize the learning problem by transferring knowledge from a source domain, where resources have already been allocated to collect abundant training data for learning the source task. The *TrAdaBoost* algorithm [4] has become a popular boosting-based solution for this case of the inductive transfer learning problem.

The source and target domains and tasks may differ either because their marginal probability distributions differ ($P_T \neq P_S$), or their boolean functions differ ($f_T \neq f_S$), or both differ. *TrAdaBoost* provides a framework for automatically discovering which part of knowledge is specific for the source domain or task, and which part may be common between source and target domains or tasks, and provides a way to attempt to transfer this knowledge from the source

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.

to the target.

The effectiveness of any inductive transfer learning method depends on the source domain and task, and on how they relate to the target domain and task. It is reasonable to expect a transfer method to take advantage of strong relationships. The most effective transfer would occur when $\mathcal{D}_T = \mathcal{D}_S$, and $\mathcal{T}_T = \mathcal{T}_S$, which reduces inductive transfer learning to traditional machine learning. On the other hand, a weak relationship may cause the transfer method to not only be ineffective, but also to decrease the performance of the target task, when compared to traditional machine learning performance. This effect is known as *negative transfer*.

In order to increase *positive transfer*, and avoid the negative, one can think of transferring knowledge not from one but from multiple sources. In this case the transfer learning method could identify and take advantage of the source, among the ones that have been made available, that is found to be the most closely related to the target. Or even better, it could take advantage of the best pieces of knowledge, coming from various available sources, that are found to be the most closely related to the target. Therefore, in this work *we make the assumption that N source domains $\mathcal{D}_{S_1}, \cdots, \mathcal{D}_{S_N}$, with source tasks $\mathcal{T}_{S_1}, \cdots, \mathcal{T}_{S_N}$, and source training data $D_{S_1}, \cdots, D_{S_N}$, are available, and would like to exploit them to improve the learning of the target classifier function $\hat{f}_T : \mathcal{X} \rightarrow \mathcal{Y}$.*

Since *TrAdaBoost* transfers knowledge only from one source, its performance heavily relies on the relationship between source and target. In Section 4 we extend *TrAdaBoost* from handling only one source to handling multiple sources, making it much less vulnerable to negative transfer. In Section 5 we further expand this boosting-based approach by introducing a two-step learning procedure that, given the same sources, can transfer knowledge to a new target with minimal computational complexity.

4. TrAdaBoost with multiple sources

In this section we are going to present an extension of *TrAdaBoost* [4] to multiple sources. We recall that *AdaBoost* [11] is a traditional machine learning algorithm, which assumes that the domains and tasks from where the training and testing data come from are the same (i.e. there is no distinction between source and target because $\mathcal{D}_S = \mathcal{D}_T$, and $\mathcal{T}_S = \mathcal{T}_T$). *AdaBoost* at every iteration increases the accuracy of the selection of the next weak classifier by carefully adjusting the weights of the training instances. In particular, it gives more importance to misclassified instances because they are believed to be the "most informative" for the next selection.

TrAdaBoost assumes that there is abundant source training data to learn a classifier, but the target domain and task are different from the source ($\mathcal{D}_S \neq \mathcal{D}_T$, and $\mathcal{T}_S \neq \mathcal{T}_T$). Therefore, the *TrAdaBoost* learning paradigm allows to exploit a small target training data set D_T, in conjunction with the source training data set D_S, for driving the boosting of a target classifier $\hat{f}_T$. The target training instances drive the selection of a week classifier in the same way as *AdaBoost* does. On the other hand, at every iteration the source training instances are given less importance when they are misclassified. This is because they are believed to be the most dissimilar to the target instances, and therefore their impact to the next weak classifier selection should be weakened.

Algorithm 1: *MultiSourceTrAdaBoost*

```
Input: Source training data $D_{S_1}, \cdots, D_{S_N}$, target training data
       $D_T$, and the maximum number of iterations $M$
Output: Target classifier function $\hat{f}_T : \mathcal{X} \rightarrow \mathcal{Y}$
Set $\alpha_S \doteq \frac{1}{2} \ln \left(1 + \sqrt{2 \ln \frac{n_S}{M}}\right)$, where $n_S \doteq \sum_k n_{S_k}$
Initialize the weight vector $(\mathbf{w}^{S_1}, \cdots, \mathbf{w}^{S_N}, \mathbf{w}^T)$, where
$\mathbf{w}^{S_k} \doteq (w_1^{S_k}, \cdots, w_{n_{S_k}}^{S_k})$, and $\mathbf{w}^T \doteq (w_1^T, \cdots, w_{n_T}^T)$ to the
desired distribution
for $t \leftarrow 1$ to $M$ do
    Empty the set of candidate weak classifiers, $\mathcal{F} \leftarrow \emptyset$
    Normalize to 1 the weight vector $(\mathbf{w}^{S_1}, \cdots, \mathbf{w}^{S_N}, \mathbf{w}^T)$
    for $k \leftarrow 1$ to $N$ do
        Find the candidate weak classifier $h_t^k : \mathcal{X} \rightarrow \mathcal{Y}$ that
        minimizes the classification error over the combined set
        $D_{S_k} \bigcup D_T$, weighted according to $(\mathbf{w}^{S_k}, \mathbf{w}^T)$
        Compute the error of $h_t^k$ on $D_T$:
            $\epsilon_t^k \doteq \sum_j \frac{w_j^T [y_j^T \neq h_t^k(\mathbf{x}_j^T)]}{\sum_i w_i^T}$
        $\mathcal{F} \leftarrow \mathcal{F} \cup (h_t^k, \epsilon_t^k)$
    Find the weak classifier $h_t : \mathcal{X} \rightarrow \mathcal{Y}$ such that
        $(h_t, \epsilon_t) \doteq \arg \min_{(h,\epsilon) \in \mathcal{F}} \epsilon$
    Set $\alpha_t \doteq \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$, where $\epsilon_t < 1/2$
    Update the weight vector
        $w_i^{S_k} \leftarrow w_i^{S_k} e^{-\alpha_S |h_t(\mathbf{x}_i^{S_k}) - y_i^{S_k}|}$
        $w_i^T \leftarrow w_i^T e^{\alpha_t |h_t(\mathbf{x}_i^T) - y_i^T|}$
return $\hat{f}_T(\mathbf{x}) \doteq \text{sign}\left(\sum_t \alpha_t h_t(\mathbf{x})\right)$
```

We now extend *TrAdaBoost* to the case where abundant training data is available from multiple sources, each of which is different from the target ($\mathcal{D}_{S_k} \neq \mathcal{D}_T$, and $\mathcal{T}_{S_k} \neq \mathcal{T}_T$). The strategy for assigning the importance to the source and target training instances remains the same as explained above. However, we no longer have to find a week classifier by leveraging only one source, and a mechanism has been introduced such that *every weak classifier is selected from the source that appears to be the most closely related to the target, at the current iteration.* Clearly, this approach greatly reduces the effects of negative transfer caused by the imposition to transfer knowledge from a single source, potentially loosely related to the target. More precisely, at every iteration each source, independently from the others,

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2: Phase-I of *TaskTrAdaBoost*

```
Input: Source training data $D_{S_1}, \cdots, D_{S_N}$, the maximum number
    of iterations $M$, and the regularizing threshold $\gamma$
Output: Set of candidate weak classifiers $\mathcal{H}$
Empty the set of candidate weak classifiers, $\mathcal{H} \leftarrow \emptyset$
for $k \leftarrow 1$ to $N$ do
    Initialize the weight vector $\mathbf{w}^{S_k} \doteq (w_1^{S_k}, \cdots, w_{n_{S_k}}^{S_k})$, to the
    desired distribution
    for $t \leftarrow 1$ to $M$ do
        Normalize to 1 the weight vector $\mathbf{w}^{S_k}$
        Find the candidate weak classifier $h_t^k : \mathcal{X} \rightarrow \mathcal{Y}$ that
        minimizes the classification error over the set $D_{S_k}$,
        weighted according to $\mathbf{w}^{S_k}$
        Compute the error $\epsilon \leftarrow \sum_j w_j^{S_k} [y_j^{S_k} \neq h_t^k(\mathbf{x}_j^{S_k})]$
        $\alpha \leftarrow \frac{1}{2} \ln \frac{1-\epsilon}{\epsilon}$, where $\epsilon < 1/2$
        if $\alpha > \gamma$ then
            $\mathcal{H} \leftarrow \mathcal{H} \cup h_t^k$
        Update the weights $w_i^{S_k} \leftarrow w_i^{S_k} e^{-\alpha y_i^{S_k} h_t^k(\mathbf{x}_i^{S_k})}$
return $\mathcal{H}$
```

Algorithm 3: Phase-II of *TaskTrAdaBoost*

```
Input: Target training data $D_T$, the set of candidate weak classifiers
    $\mathcal{H}$, and the maximum number of iterations $M$
Output: Target classifier function $\hat{f}_T : \mathcal{X} \rightarrow \mathcal{Y}$
Initialize the weight vector $\mathbf{w}^T \doteq (w_1^T, \cdots, w_{n_T}^T)$, to the desired
distribution
for $t \leftarrow 1$ to $M$ do
    Normalize to 1 the weight vector $\mathbf{w}^T$
    Empty the current weak classifier set $\mathcal{F} \leftarrow \emptyset$
    foreach $h \in \mathcal{H}$ do
        Compute the error of $h$ on $D_T$

            $\epsilon \leftarrow \sum_j w_j^T [y_j^T \neq h(\mathbf{x}_j^T)]$        (1)

        if $\epsilon > 1/2$ then
            $h \leftarrow -h$
            Update $\epsilon$ via (1)
        $\mathcal{F} \leftarrow \mathcal{F} \cup (h, \epsilon)$
    Find the weak classifier $h_t : \mathcal{X} \rightarrow \mathcal{Y}$ such that

        $(h_t, \epsilon_t) \doteq \arg \min_{(h,\epsilon) \in \mathcal{F}} \epsilon$

    $\mathcal{H} \leftarrow \mathcal{H} \setminus h_t$
    Set $\alpha_t \doteq \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$
    Update the weights $w_i^T \leftarrow w_i^T e^{-\alpha_t y_i^T h_t^k(\mathbf{x}_i^T)}$
return $\hat{f}_T(\mathbf{x}) \doteq \text{sign}\left(\sum_t \alpha_t h_t(\mathbf{x})\right)$
```

combines its training data with the target training data to propose a *candidate weak classifier*. The final weak classifier is then chosen from the source that minimizes the target classification error.

A detailed description of the proposed extension is given in Algorithm 1, and is called *MultiSourceTrAdaBoost*, where N source training data sets are given as input, and M week classifiers are extracted to compose $\hat{f}_T$. As it can be seen from line 10, the weighting update of the source training instances is the same as in *TrAdaBoost*, and the weighting update of the target training instances is the same as in *AdaBoost*. At every iteration the inner loop computes: (a) N candidate week classifiers from N different training data sets, $\{D_{S_k} \bigcup D_T\}$; (b) how each weak classifier relates to the target training data by computing the classification error. Line 8 then selects the weak classifier corresponding to the source that minimizes the target classification error. Finally, when $N = 1$ the algorithm reduces to *TrAdaBoost*.

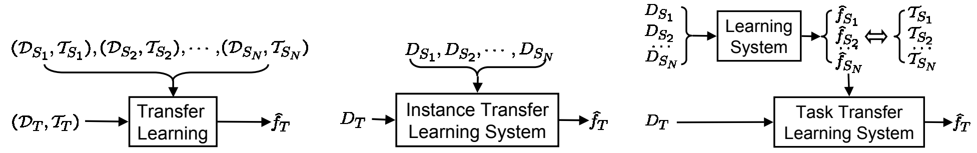
5. Boosting for transferring source tasks

Figure 1(a) depicts the conceptualization of inductive transfer learning, which is intended as the exploitation of the knowledge from different sources to improve the learning of a classifier that is meant to work in a target domain, to address the target task that was defined on it. *MultiSourceTrAdaBoost* is one particular implementation of this concept. More precisely, it tries to identify which training instances, coming from the various source domains, can be reused, together with the target training instances, to boost the target classifier. Figure 1(b) depicts this situation, which is typically referred to as an *instance-transfer approach*. Another way of implementing inductive transfer learning is by admitting that the target classifier model will share some parameters with the most closely related sources. Therefore, this *parameter-transfer approach* tries to identify which parameters, coming from various sources, can be reused, together with the target training data, to improve the target classifier learning.

In this section we introduce an inductive transfer learning framework made of two phases. Phase-I deploys traditional machine learning to extract suitable parameters that summarize the knowledge from the sources. Phase-II is a parameter-transfer approach for boosting the target classifier $\hat{f}_T$. More precisely, phase-I extracts the parameters that constitute the models of the source task classifiers $\hat{f}_{S_1}, \cdots, \hat{f}_{S_N}$. Therefore, the source tasks are described explicitly, and not implicitly through the labeled source training data. For this reason, this instance of parameter-transfer approach can be thought of as a *task-transfer approach*, where *sub-tasks*, coming from the various source tasks, can be reused, together with the target training instances, to boost the target classifier, because they are believed to be closely related to the target task. The sub-tasks will be represented under the form of weak classifiers. Figure 1(c) depicts this situation.

A detailed description of the proposed approach, which is called *TaskTrAdaBoost*, is given in Algorithm 2 for phase-I, and in Algorithm 3 for phase-II. Phase-I is nothing but *AdaBoost* run for each of the source training data. The output $\mathcal{H}$ is a collection of all the candidate weak classifiers that are being computed, and that are the most discriminative. In fact, we constrain the coefficient α to be greater

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.



$(\mathcal{D}_{S_1},\mathcal{T}_{S_1}),(\mathcal{D}_{S_2},\mathcal{T}_{S_2}),\cdots,(\mathcal{D}_{S_N},\mathcal{T}_{S_N})$
$(\mathcal{D}_T,\mathcal{T}_T)$
Transfer Learning
$\hat{f}_T$
$D_{S_1},D_{S_2},\cdots,D_{S_N}$
D_T
Instance Transfer Learning System
$\hat{f}_T$
D_{S_1}
D_{S_2}
D_{S_N}
Learning System
$\hat{f}_{S_1}$
$\hat{f}_{S_2}$
$\hat{f}_{S_N}$
⇔
$\mathcal{T}_{S_1}$
$\mathcal{T}_{S_2}$
$\mathcal{T}_{S_N}$
D_T
Task Transfer Learning System
$\hat{f}_T$

(a) (b) (c)

Figure 1. ***Transfer learning approaches.*** (a) Inductive transfer learning. (b) Instance-transfer based approach (*MultiSourceTrAdaBoost*). (c) Parameter-transfer based approach (*TaskTrAdaBoost*).

than a given regularizing threshold γ. We do so because we are not interested in transferring parameters that may lead the target classifier to overfitting the data. Phase-II is again an *AdaBoost* loop over the target training data D_T. However, at every iteration, from $\mathcal{H}$ it is picked the weak classifier with the lowest classification error on the target training data, ensuring the transfer of the knowledge that is more closely related to the target task. Moreover, the update of the weights of the target training instances drives the search for the transfer of the next sub-task that is needed the most for boosting the target classifier.

6. Algorithm comparisons

Decision boundaries Figure 2 shows a data distribution, and a sketch of how various learning algorithms would attempt to separate the instances. Squares are negative samples. Crosses are positive samples from one source, D_{S_1}. Circles are positive samples from another source, D_{S_2}. Orange stars, and orange crosses are positive training and testing instances in the target domain, respectively.

Figure 2(a) shows that a traditional machine learning algorithm, such as *AdaBoost*, would overfit the target training data with decision boundaries unable to guarantee good generalization. Figure 2(b) shows the decision boundaries obtained by *TrAdaBoost* when D_{S_1} and D_{S_2} are used jointly, which means that the sources are seen as one. In this case there is no overfitting. Figure 2(c) shows how *MultiSourceTrAdaBoost* improves the decision boundaries. Each source separately combines with the target, virtually producing the dashed boundaries on the left. On the right, the boundary parts more closely related to the target are transferred to produce tighter target decision boundaries. Finally, Figure 2(d) shows how *TaskTrAdaBoost* would behave. Phase-I learns the dashed boundaries between S_1, and everything else, as well as the dashed boundaries between S_2, and everything else. At every iteration phase-II grabs the most useful pieces of the dashed boundaries to build the tight target decision boundaries.

Performance analysis The convergence properties of *MultiSourceTrAdaBoost* can be inherited directly from *TrAdaBoost* [4], whereas for *TaskTrAdaBoost* they can be inherited directly from *AdaBoost* [11, 18]. Moreover, because the condition of $\epsilon_t < 0.5$ is satisfied in both algorithms, the *prediction error* ϵ over the target training data D_T is bounded by $\epsilon \leq 2^M \prod_{t=1}^{M} \sqrt{\epsilon_t(1-\epsilon_t)}$, and the upper bound of the associated *generalization error* is given by $\epsilon + O(\sqrt{M d_{VC}/n_T})$, where d_{VC} is the VC-dimension of the weak classifier model [23].

We now make an observation regarding how the cardinality $|\mathcal{H}|$, of the set of candidate weak classifiers $\mathcal{H}$, affects the performance of *TaskTrAdaBoost*. If $|\mathcal{H}|$ is of the same order of magnitude of M, the offering of weak classifiers may be too limiting. Therefore, the probability to chose weak classifiers with higher classification error ϵ_t increases, leading to a higher prediction error ϵ. On the other hand, an overly rich $\mathcal{H}$ ($|\mathcal{H}|$ very big), would very much increase the probability to choose weak classifiers with low classification error, leading to a low prediction error. However, the VC-dimension d_{VC} would increase as well, leading to a higher risk for overfitting, as well as poorer generalization. This is the reason for inserting the regularizing threshold γ in phase-I, which allows to strike a balance between prediction and generalization performance.

The set $\mathcal{H}$ plays also another role in *TaskTrAdaBoost*. More precisely, the fact that it limits the freedom in picking the weak classifiers leads to a greater prediction error, in comparison with *MultiSourceTrAdaBoost*. On the other hand, in the generalization error this effect is compensated because this reduced freedom also leads to a smaller VC-dimension d_{VC}, and therefore a lower upper bound.

Finally, since we have $n_T << n_S$, the convergence rate of *TaskTrAdaBoost* has a reduced upper bound [11, 4], compared to *MultiSourceTrAdaBoost*, which means that it requires fewer iterations.

7. Experimental results

The performance of the proposed methods are investigated based on two applications: object category recognition and specific object detection. In object category recognition, it is assumed that we are given a small number of training samples of a *target object category*, and abundant training samples of other *source object categories*. When presented with a test sample, we verify whether it belongs to the target object category. As for specific object detection, it is assumed that we are given a small number of training samples of a *target object*, and abundant training samples of other *source objects* of the same category and of other categories (background). When presented with a test image, we want to verify whether it contains the target object, and where it is located in the image. We use *AdaBoost* and *TrAdaBoost* as the baseline methods for performance comparison. All the

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.

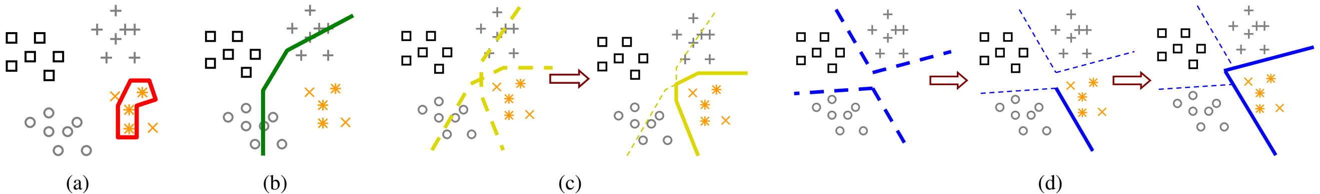

(a) (b) (c) (d)

Figure 2. ***Decision boundaries.*** Representation of the decision boundaries between the positive and negative samples in the target domain, as computed by *AdaBoost* (a), *TrAdaBoost* (b), *MultiSourceTrAdaBoost* (c), and *TaskTrAdaBoost* (d). Orange crosses and stars represent the target positive samples. Dashed lines represent candidate decision boundaries. Solid lines are the learned boundaries.

algorithms use a linear SVM as the basic learner to build a weak classifier.

For every experiment, we provide the receiver operating characteristic (ROC) curve of the classifier output for performance comparison. We also compute the area under the ROC curve A_{ROC} as a quantitative performance evaluation.

7.1. Object category recognition

Data sets For object category recognition, we have used the Caltech 256 data set [13], which contains 256 object categories. Among them, we have used the 136 categories that have more than 100 samples. We have also used the background data set, collected via the Google image search engine, along with the remaining categories as our augmented background data set.

Experimental setup The "bag-of-words" method [9] is used to map images into the feature space for classification. We designate the target category and randomly draw the positive samples that form the target data. The number of positive samples for training n_T^+, is limited from 1 to 50, for testing is 50. We treat the remaining categories as the repository from which to draw positive samples for the source data. We vary the number of source categories, or domains, N, from 1 to 10 to investigate the performance of the classifiers with respect to the variability of the domains. The number of positive samples for one source of data is 100. The negative samples of both source and target data are randomly drawn from the augmented background data set. The number of negative training samples in the target data is given by $5n_T^+$. The number of negative testing samples in the target data is 250. The number of negative training samples in the source data is 500. For each target object category, the performance of the classifier is evaluated over 20 random combinations of N source object categories. Given the target and source categories, the performance of the classifier is obtained by averaging over 20 trials of experiments. The overall performance of the classifier is averaged over 20 target categories.

Results Figure 3 compares *AdaBoost*, *TrAdaBoost*, *MultiSourceTrAdaBoost*, and *TaskTrAdaBoost* based on the area under the ROC with different number of positive target training samples ($n_T^+ \in \{1, 5, 15, 50\}$) and source domains ($N \in \{1, 2, 3, 5\}$). Figure 3(a) assumes $N = 3$ and shows the behavior of the algorithms as n_T^+ increases. Since *AdaBoost* does not transfer any knowledge from the source, its performance heavily depends on n_T^+. For a very small n_T^+ it performs slightly better than chance, according to the A_{ROC}. *TrAdaBoost* combines the three sources into one and improves upon *AdaBoost* due to the transfer learning mechanism. By incorporating the ability to transfer knowledge from multiple individual domains, *MultiSourceTrAdaBoost* and *TaskTrAdaBoost* demonstrate a significant improvement in recognition accuracy, even for a very small n_T^+. In addition, the performance of *AdaBoost* and *TrAdaBoost* strongly depends on the selection of source domains and target positive samples, as revealed by the standard deviation of A_{ROC}. On the other hand, a much smaller standard deviation is observed from both of the proposed algorithms. As expected, the performance gaps among all the approaches dwindle as n_T^+ increases. They show a significant decrease when $n_T^+ = 50$, for the given dataset with a limited amount of positive testing samples.

Figure 3(b) assumes that $N = 1$. It shows that *MultiSourceTrAdaBoost* reduces to *TrAdaBoost* and therefore they have the same performance. Moreover, it shows that *TaskTrAdaBoost* outperforms *MultiSourceTrAdaBoost* when n_T^+ is very small, and underperforms it for a larger n_T^+. These are the effects of a low VC-dimension offered by *TaskTrAdaBoost*, which is leveraging only one source. When n_T^+ is very small it helps avoiding overfitting more than other approaches. When n_T^+ increases it limits the ability to build the desired decision boundaries.

Figure 3(c) assumes $n_T^+ = 1$, and shows that as the number of source domains increases, the A_{ROC} of *MultiSourceTrAdaBoost* and *TaskTrAdaBoost* increases, and the corresponding standard deviations decrease, indicating an improved performance in both accuracy and consistency. *TrAdaBoost* is incapable of exploring the decision boundaries separating multiple source domains, resulting in a maintained performance regardless of the number of source domains. With $N = 3$ and $n_T^+ = 1$, *MultiSourceTrAdaBoost* and *TaskTrAdaBoost* have an A_{ROC} of 0.966 ± 0.047 and 0.972 ± 0.043, respectively, in comparison with an A_{ROC} of 0.848 ± 0.111 from *TrAdaBoost*.

Time complexity With C_h and C_w we indicate the time complexity to compute a weak classifier, and update the weight of one training instance. The time complexity of *AdaBoost* is approximately $C_h O(M) + C_w O(M n_T)$, the time complexity of *TrAdaBoost* is $C_h O(M) + C_w O(M n_S)$,

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.

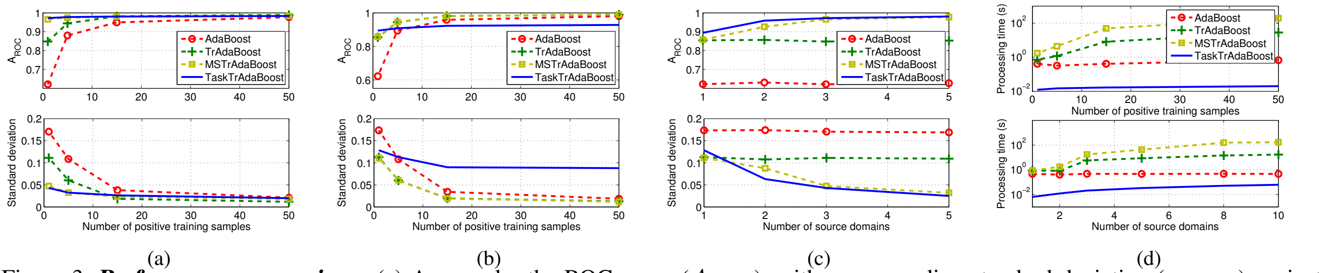


Figure 3. ***Performance comparison.*** (a) Area under the ROC curve (A_{ROC}), with corresponding standard deviation ($\sigma_{A_{ROC}}$) against the number of positive target training samples n_T^+ with $N = 3$ sources; (b) A_{ROC} and $\sigma_{A_{ROC}}$ against n_T^+ with $N = 1$; (c) A_{ROC} and $\sigma_{A_{ROC}}$ against N with $n_T^+ = 1$ (d) Processing time against n_T^+ with $N = 2$ (top), and against N with $n_T^+ = 1$ (bottom). *MSTrAdaBoost* represents *MultiSourceTrAdaBoost*.

and the time complexity of *MultiSourceTrAdaBoost* is $C_h O(MN) + C_w O(M n_S)$, which is roughly the same as that of phase-I of *TaskTrAdaBoost*, whereas phase-II has a time complexity of $C_w O(M|\mathcal{H}| n_T)$. Therefore, since we typically have $C_h \gg C_w$ and $|\mathcal{H}| n_T \approx n_S$, the phase-II of *TaskTrAdaBoost* is very fast and deployable when there is a strong need for rapid retraining over a new target domain. Figure 3(d) plots the recorded average training time per experiment trial against n_T^+ and N of all the algorithms.

7.2. Specific object detection

Data sets For this experiment we have collected a data set made of two video sequences of highway traffic. The first one (`Sequence A`) includes vehicle images from a fixed view point, whereas the second one (`Sequence B`) includes vehicle images from different view points. For each video, we manually annotated the ground-truth vehicle locations, and sizes, by recording rectangular regions of interest (ROIs) around each vehicle moving along the highway, resulting in a total of about 70000 different ROIs, corresponding to 400 different vehicles. The sizes of the ROIs vary from about 30×20 to 120×40 pixels, depending on the type of the vehicle and the view points. The average number of annotated ROIs per vehicle is approximately 100 and 400 for `Sequence A` and `Sequence B`, respectively.

Experimental setup Considering the small sizes of the ROIs, in this experiment we have chosen the region moment descriptor [5] for mapping image ROIs onto the feature space. We fix the number of source domains to $N = 50$. Positive samples are selected from the annotated ROIs and negative samples are randomly cropped from the background. For a target object, n_T^+ varies from 1 to 50, whereas the remaining positive samples are used for testing. The overall ROC curves are obtained by averaging the performances over 50 target vehicles. The remaining experimental setup is the same as for the object category recognition. The trained classifiers have been deployed to perform specific object detection in video by using a multiscale sliding window scheme, which reveals position and scale of the detected vehicle (see Figure 5).

Results Figure 4 compares the performances among classifiers based on ROC curves, and Table 1 lists the corresponding A_{ROC} values. For $n_T^+ = 50$, all the classifiers produce comparable performances. As n_T^+ decreases and becomes 1 we observe significant performance gaps. As expected, the proposed approaches can effectively exploit the decision boundaries between multiple sources and outperform *TrAdaBoost*. The reduced time complexity of the phase-II of *TaskTrAdaBoost* makes it a good candidate for applications that need rapid retraining for the detection of a new target object. Finally, Figure 5 shows example frames with the detection of specific vehicles using *TaskTrAdaBoost*.

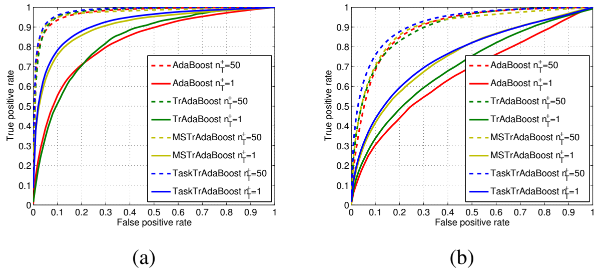


Figure 4. ***ROC curves.*** (a) `Sequence A` and (b) `Sequence B`.

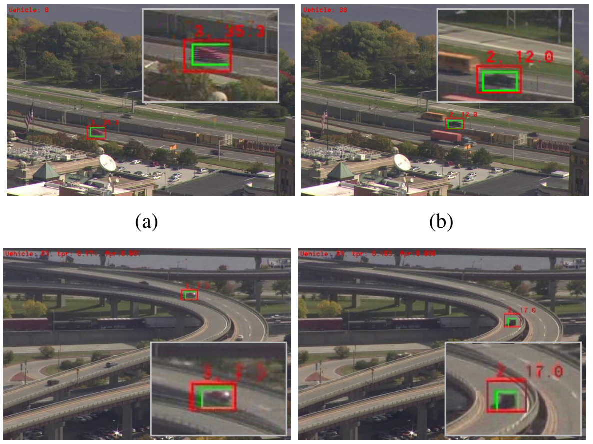
8. Conclusions

This work extends the boosting framework for inductive transfer learning when knowledge from multiple sources is available to help training a target classifier. Considering multiple sources directly addresses the problem of *negative transfer* because the chance to import knowledge from a source related to the target increases significantly. *MultiSourceTrAdaBoost*, an instance-transfer approach, and *TaskTrAdaBoost*, a parameter-transfer (or more specifically a task-transfer) approach, are introduced. They are applied to the problem of object category recognition and specific object detection when the target data available is scarce. Compared to *TrAdaBoost*, an impressive performance increase in terms of recognition and detection rates is observed, even with only one positive target training sample,

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.

Sequence A	$n_T^+ = 50$	$n_T^+ = 1$
AdaBoost	0.976 ± 0.021	0.837 ± 0.107
TrAdaBoost	0.979 ± 0.020	0.846 ± 0.107
MultiSourceTrAdaBoost	0.976 ± 0.030	0.909 ± 0.087
TaskTrAdaBoost	0.985 ± 0.014	0.923 ± 0.068
Sequence B	$n_T^+ = 50$	$n_T^+ = 1$
AdaBoost	0.904 ± 0.062	0.674 ± 0.131
TrAdaBoost	0.896 ± 0.063	0.705 ± 0.129
MultiSourceTrAdaBoost	0.900 ± 0.056	0.754 ± 0.108
TaskTrAdaBoost	0.922 ± 0.040	0.765 ± 0.136

Table 1. ***ROC area.*** Area under the ROC curves, A_{ROC}, and corresponding standard deviations.



(a) (b)

(c) (d)

Figure 5. ***Specific object detection.*** Example frames with specific vehicles detected. (a)-(b) `Sequence A` and (c)-(d) `Sequence B`. Green and red boxes depict the ground-truth and detected ROIs, respectively. Gray boxes show a zoom-in view of the detected ROIs.

showing the effectiveness of both of the approaches in exploiting multiple sources, with a slight advantage of *TaskTrAdaBoost*. Moreover, as the number of sources increases, a dramatic decrease in performance variability is observed, showing that the approaches tend to become independent of the source choices, and therefore they properly address the negative transfer problem. The framework is general, and applicable to help the learning in a wide variety of computer vision problems. Finally, an important property of *TaskTrAdaBoost* is its speed when it is presented with a new target task to learn. This aspect makes it a good candidate for applications where the source knowledge needs to be quickly reused, for instance for the on-line retraining for the detection of a new specific object.

References

[1] S. Bickel, M. Bruckner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *Int'l Conf. on Machine Learning*, 2007.

[2] E. Bonilla, K. M. Chai, and C. Williams. Multi-task gaussian process prediction. In *Annual Conf. on Neural Information Processing Systems*, pages 153–160, 2008.

[3] L. Bruzzone and M. Marconcini. Toward the automatic updating of land-cover maps by a domain-adaption SVM classifier and a circular validation strategy. *IEEE Trans. on Geoscience and Remote Sensing*, 47(4):1108–1122, Apr. 2009.

[4] W. Dai, Q. Yang, G. Xue, and Y. Yu. Boosting for transfer learning. In *Int'l Conf. on Machine Learning*, Corvallis, OR, 2007.

[5] G. Doretto and Y. Yao. Region Moments: Fast invariant descriptors for detecting small image structures. In *CVPR*, 2010.

[6] L. Duan, I. W. Tsang, D. Xu, and S. J. Maybank. Domain transfer SVM for video concept detection. In *CVPR*, 2009.

[7] A. Farhadi, D. Forsyth, and R. White. Transfer lerning in sign language. In *CVPR*, 2007.

[8] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE TPAMI*, 28(4):594–611, Apr. 2006.

[9] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *CVPR*, volume 2, pages 524–531, June 20–25, 2005.

[10] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

[11] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Science*, 55:119–139, 1997.

[12] H. Grabner and H. Bischof. On-line boosting and vision. In *CVPR*, pages 260–267, New York, NY, Jun. 2006.

[13] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[14] W. Jiang, E. Zavesky, S.-F. Chang, and A. Loui. Cross-domain learning methods for high-level visual concept classification. In *ICIP*, 2008.

[15] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object class by between-class attibute transfer. In *CVPR*, pages 951–957, Miami Beach, FL, Jun. 2009.

[16] S. J. Pan and Q. Yang. A survey on transfer learning. Technical Report HKUST-CS08-08, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China, Nov. 2008.

[17] A. Quattoni, M. Collins, and T. Darrell. Transfer learning for image classification with sparse prototype representation. In *CVPR*, 2008.

[18] R. E. Schapire. A brief introduction to boosting. In *Int'l Conf. on Artificial Intelligence*, 1999.

[19] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, pages 373–380, Tokyo, Japan, Oct. 2009.

[20] Z. Sun, Y. Chen, J. Qi, and J. Liu. Adaptive localization through transfer learning in indoor Wi-Fi environment. In *Int'l Conf. on Machine Learning and Applications*, 2008.

[21] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE TPAMI*, 29(5):854–869, May 2007.

[22] L. Torrey and J. Shavlik. *Transfer learning*. IGI Global, 2009.

[23] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, 1982.

[24] C. Wang and S. Mahadevan. Manifold alignment using procrustes analysis. In *Int'l Conf. on Machine Learning*, 2008.

[25] P. Wang, C. Domeniconi, and J. Hu. Using wikipedia for co-clustering based cross-domain text classification. In *IEEE Int'l Conf. on Data Mining*, 2008.

[26] X. Wang, C. Zhang, and Z. Zhang. Boosted multi-task learning for face verification with applications to web image and video search. In *CVPR*, 2009.

[27] J. Yang, R. Yan, and A. G. Hauptmann. Cross-domain video concept detection using adaptive SVMs. In *ACM Multimedia*, 2007.

Authorized licensed use limited to: Jinan University. Downloaded on August 07,2020 at 02:29:27 UTC from IEEE Xplore. Restrictions apply.