




Transfer Learning by Learning Projections from Target to Source

Antoine Cornuéjols¹(✉) , Pierre-Alexandre Murena^{1,2}, and Raphaël Olivier³

¹ UMR MIA-Paris, AgroParisTech, INRA, Université Paris-Saclay,
75005 Paris, France

antoine.cornuejols@agroparistech.fr

² Telecom ParisTech - Université Paris-Saclay, 75013 Paris, France

³ University Carnegie-Mellon, Pittsburgh, USA

<http://www.agroparistech.fr/mia/equipes/membres/page:antoine>

Abstract. Using transfer learning to help in solving a new classification task where labeled data is scarce is becoming popular. Numerous experiments with deep neural networks, where the representation learned on a source task is transferred to learn a target neural network, have shown the benefits of the approach. This paper, similarly, deals with hypothesis transfer learning. However, it presents a new approach where, instead of transferring a representation, the source hypothesis is kept and this is a translation from the target domain to the source domain that is learned. In a way, a change of representation is learned. We show how this method performs very well on a classification of time series task where the space of time series is changed between source and target.

Keywords: Transfer learning · Boosting

1 Introduction

While transfer learning has a long history, dating back at least to the study of analogy reasoning, it has enjoyed a spectacular rise of interest in recent years, thanks largely to its use and effectiveness in learning new tasks with deep neural networks using an architecture learned on a source task. This approach is called Hypothesis Transfer Learning [6]. The justification for this strategy is that, in the absence of enough data in the target domain to learn anew a good hypothesis, it might be effective to transfer the intermediate representations learned on the source task. This is indeed the case, for instance, in face analysis when the source task is to guess the age of the person, and the target task is to recognize the gender. Technically, with neural networks, this amounts to keeping the first layers of the source neural network in the target network and learning only the last layers, the ones that combine intermediate representations of the examples in order to make a prediction.

Let \mathcal{X} , \mathcal{Y} and \mathcal{Z} be the input, output and feature spaces respectively. Let F be a class of representation functions, where $f \in F: \mathcal{X} \rightarrow \mathcal{Z}$. Let G be a class

of decision functions that use descriptions of the examples in the feature space: $g \in G : \mathcal{Z} \rightarrow \mathcal{Y}$. Then, in the context of deep neural networks, the hypothesis class is $\mathcal{H} := \{h : \exists f \in F, g \in G \text{ st. } h = g \circ f\}$ and while f is kept (at least approximately) from the source problem to the target one, only g remains to be learned to solve the target problem.

In this paper, we adopt a dual perspective: we propose to keep the decision function g fixed, and learn *translation functions* from the target input space to the source input space, $\pi : \mathcal{X}_{\mathcal{T}} \rightarrow \mathcal{X}_{\mathcal{S}}$, such that the target hypothesis space becomes $\mathcal{H}_{\mathcal{T}} := \{h_{\mathcal{T}} : \exists \pi \in \Pi, f \in F, g \in G \text{ st. } h_{\mathcal{T}} = g \circ f \circ \pi\}$, which, given that $h_{\mathcal{S}} = g \circ f$ might be considered as the source hypothesis, may be re-expressed as: $\mathcal{H}_{\mathcal{T}} := \{h_{\mathcal{T}} : \exists \pi \in \Pi, f \in F, g \in G \text{ st. } h_{\mathcal{T}} = h_{\mathcal{S}} \circ \pi\}$.

Indeed, for some problems, it might be much more easy to learn a translation (also called *projections* in this paper) from the target input space $\mathcal{X}_{\mathcal{T}}$ to the source input space $\mathcal{X}_{\mathcal{S}}$ than to learn a new target decision function. Furthermore, this allows one to tackle problems with different input spaces $\mathcal{X}_{\mathcal{S}}$ and $\mathcal{X}_{\mathcal{T}}$.

In the following, Sect. 2 presents **TransBoost** a new algorithm for transfer learning. The theoretical analysis of Sect. 3 provides a PAC-learning bound on the generalization error on the target domain. Controlled experiments are described in Sect. 4 together with an analysis of the results. The new approach is put in perspective in Sect. 5 before we conclude in Sect. 6.

2 A New Algorithm for Transfer Learning

Suppose that we have a system that is able to recognize poppy fields in satellite images. We might imagine that knowing how to translate a biopsy image into a satellite image, we could, using the recognition function defined on satellite image, decide if there is cancerous cells in the biopsy.

Ideally then, one could translate a target query: “what is the label of $\mathbf{x}^{\mathcal{T}} \in \mathcal{X}_{\mathcal{T}}$ ” into a source query “what is the label of $\pi(\mathbf{x}^{\mathcal{T}}) \in \mathcal{X}_{\mathcal{S}}$ ” where $h_{\mathcal{S}}$ is the source hypothesis which, applied to $\pi(\mathbf{x}^{\mathcal{T}}) \in \mathcal{X}_{\mathcal{S}}$, provides the answer we are looking for. Notice here that we suppose that $\mathcal{Y}_{\mathcal{S}} = \mathcal{Y}_{\mathcal{T}}$, but not $\mathcal{X}_{\mathcal{S}} = \mathcal{X}_{\mathcal{T}}$.

The goal is then to learn a good translation $\pi : \mathcal{X}_{\mathcal{T}} \rightarrow \mathcal{X}_{\mathcal{S}}$. However, defining a proper space of candidate projections Π might be problematic, not to mention the risk of overfitting if the space of functions $h_{\mathcal{S}} \circ \Pi$ has too high a capacity. It might be more easy and manageable to discover “weak projections” from $\mathcal{X}_{\mathcal{T}}$ to $\mathcal{X}_{\mathcal{S}}$ using a boosting learning scheme.

Definition 1. A *weak projection* w.r.t. source decision function $h_{\mathcal{S}}$ is a function $\pi : \mathcal{X}_{\mathcal{T}} \rightarrow \mathcal{X}_{\mathcal{S}}$ such that the decision function $h_{\mathcal{S}}(\pi(\mathbf{x}^{\mathcal{T}}))$ has better than random classification performance on the target training set $S_{\mathcal{T}}$.

In this setting, the training set $S_{\mathcal{T}} = \{(\mathbf{x}_i^{\mathcal{T}}, y_i^{\mathcal{T}})\}_{1 \leq i \leq m}$ is used to learn *weak projections* (Fig. 1).

Once the concept of weak projection is assumed, it is natural to use a boosting algorithm in order to learn a set of such weak projections and to combine them to get a final good classification on elements of \mathcal{T} . This is what does the

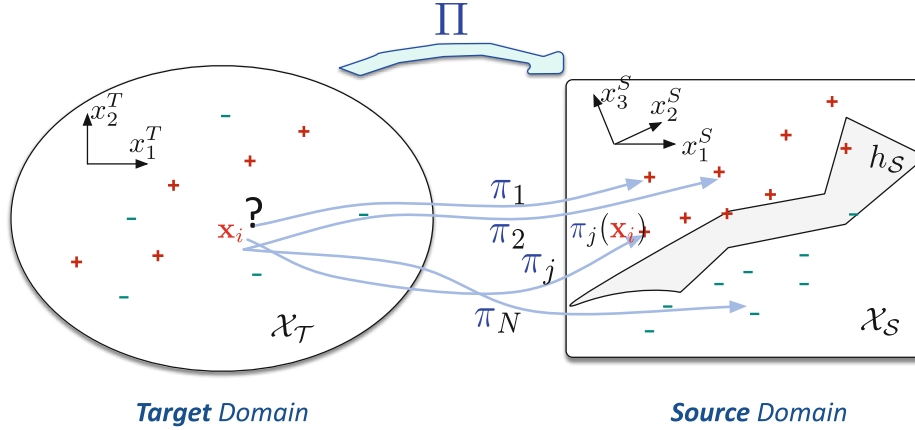


Fig. 1. The principle of prediction using TransBoost. A given target example \mathbf{x}_i^T is projected in the source domain using a set of identified weak projections π_j and the prediction for \mathbf{x}_i^T is computed as: $H_T(\mathbf{x}_i^T) = \text{sign}\left\{\sum_{j=1}^N \alpha_j h_S(\pi_j(\mathbf{x}_i^T))\right\}$.

TransBoost algorithm (see Algorithm 1). It does rely on the property of the boosting algorithm to find and combine weak rules to get a strong(er) rule.

3 Theoretical Analysis

Here, we study the question: can we get *guarantees* about the performance of the learned decision function H_T in the target space using TransBoost?

We tackle this question in two steps. First, we suppose that we learn a single projection function $\pi \in \Pi : \mathcal{X}_T \rightarrow \mathcal{X}_S$ so that $h_T = h_S \circ \pi$, and we find bounds on the generalization error on the target domain given the generalization error on the source domain. Second, we turn to the TransBoost algorithm in order to justify the use of a boosting approach.

3.1 Generalization Error Bounds When Using a Single Projection

For this analysis, we suppose the existence of a source input distribution $\mathbf{P}_{\mathcal{X}_S}$ in addition to the target input distribution $\mathbf{P}_{\mathcal{X}_T}$. We consider the binary classification setting $\mathcal{Y} = \{-1, +1\}$, and we note \bar{h}_S and \bar{h}_T respectively the source and the target labelling functions. We note $R_S(h)$ (resp. $R_T(h)$) the risk of a hypothesis h on the source (resp. target) domain: $R_S(h) = \mathbb{E}_{\mathbf{x}_S \sim \mathbf{P}_{\mathcal{X}_S}} [h_S(\mathbf{x}^S) \neq \bar{h}_S(\mathbf{x}^S)]$ (resp. $R_T(h) = \mathbb{E}_{\mathbf{x}_T \sim \mathbf{P}_{\mathcal{X}_T}} [h_T(\mathbf{x}^T) \neq \bar{h}_T(\mathbf{x}^T)]$). Let $\hat{R}_S(h)$ and $\hat{R}_T(h)$ be the corresponding empirical risks, with m_S training points for \mathcal{S} and m_T training points for \mathcal{T} . Let $d_{\mathcal{H}}$ be the VC dimension of the hypothesis space \mathcal{H} .

In the following, what is learned is a projection $\pi \in \Pi : \mathcal{X}_T \rightarrow \mathcal{X}_S$ in order to get a target hypothesis of the form $h_T = \hat{h}_S \circ \pi$, where $\hat{h}_S = \text{ArgMin}_{h \in \mathcal{H}_S} \hat{R}_S(h)$ is the source hypothesis. Our aim is to upper-bound $R_T(\hat{h}_T)$, the risk of the learned hypothesis on the target domain in terms of:

Algorithm 1. Transfer learning by boosting

Input: $h_S : \mathcal{X}_S \rightarrow \mathcal{Y}_S$ the source hypothesis
 $\mathcal{S}_T = \{(\mathbf{x}_i^T, y_i^T)\}_{1 \leq i \leq m}$: the target training set

Initialization of the distribution on the training set: $D_1(i) = 1/m$ for $i = 1, \dots, m$;

for $n = 1, \dots, N$ **do**

Find a projection $\pi_i : \mathcal{X}_T \rightarrow \mathcal{X}_S$ st. $h_S(\pi_i(\cdot))$ performs better than random on $D_n(\mathcal{S}_T)$;

Let ε_n be the error rate of $h_S(\pi_i(\cdot))$ on $D_n(\mathcal{S}_T)$:

$\varepsilon_n \doteq \mathbf{P}_{i \sim D_n}[h_S(\pi_n(\mathbf{x}_i)) \neq y_i]$ (with $\varepsilon_n < 0.5$) ;

Computes $\alpha_i = \frac{1}{2} \log_2\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$;

Update, for $i = 1 \dots, m$:

$$\begin{aligned} D_{n+1}(i) &= \frac{D_n(i)}{Z_n} \times \begin{cases} e^{-\alpha_n} & \text{if } h_S(\pi_n(\mathbf{x}_i^T)) = y_i^T \\ e^{\alpha_n} & \text{if } h_S(\pi_n(\mathbf{x}_i^T)) \neq y_i^T \end{cases} \\ &= \frac{D_n(i) \exp(-\alpha_n y_i^{(T)} h_S(\pi_n(\mathbf{x}_i^{(T)})))}{Z_n} \end{aligned}$$

where Z_n is a normalization factor chosen so that D_{n+1} be a distribution on \mathcal{S}_T ;

end

Output: the final target hypothesis $H_T : \mathcal{X}_T \rightarrow \mathcal{Y}_T$:

$$H_T(\mathbf{x}^T) = \text{sign} \left\{ \sum_{n=1}^N \alpha_n h_S(\pi_n(\mathbf{x}^T)) \right\} \quad (1)$$

- the empirical risk $\widehat{R}_S(\widehat{h}_S)$ of the source hypothesis,
- the generalization error of a hypothesis \widehat{h}_S in \mathcal{H}_S learned from m_S examples, which depends on $d_{\mathcal{H}_S}$,
- the generalization error of a hypothesis $\widehat{h}_T = h_S \circ \pi$ in \mathcal{H}_T learned from m_T examples, which depends on $d_{\mathcal{H}_T} = d_{h_S \circ \pi}$,
- a term that expresses the “proximity” between the source and the target problems.

For the latter term, we adapt the theoretical study of McNamara and Balcan [9] on the transfer of representation in deep neural networks. We suppose that $P_S, P_T, h_S, h_T = \widehat{h}_S \circ \pi$ ($\pi \in \Pi$), \widehat{h}_S and Π have the property:

$$\forall \widehat{h}_S \in \mathcal{H}_S : \quad \text{Min}_{\pi \in \Pi} R_T(\widehat{h}_S \circ \pi) \leq \omega(R_S(h_S)) \quad (2)$$

where $\omega : \mathbb{R} \rightarrow \mathbb{R}$ is a non-decreasing function.

Equation (2) means that the best target hypothesis expressed using the learned source hypothesis has a true risk bounded by a non-decreasing function of the true risk on the source domain of the learned source hypothesis.

We are now in position to get the desired theorem.

Theorem 1. Let $\omega : \mathbb{R} \rightarrow \mathbb{R}$ be a non-decreasing function. Suppose that $P_{\mathcal{S}}, P_{\mathcal{T}}, h_{\mathcal{S}}, h_{\mathcal{T}} = \widehat{h}_{\mathcal{S}} \circ \pi (\pi \in \Pi)$, $\widehat{h}_{\mathcal{S}}$ and Π have the property given by Eq. (2). Let $\widehat{\pi} := \text{ArgMin}_{\pi \in \Pi} \widehat{R}_{\mathcal{T}}(\widehat{h}_{\mathcal{S}} \circ \pi)$, be the best apparent projection.

Then, with probability at least $1 - \delta$ ($\delta \in (0, 1)$) over pairs of training sets for tasks \mathcal{S} and \mathcal{T} :

$$\begin{aligned} R_{\mathcal{T}}(\widehat{h}_{\mathcal{T}}) &\leq \omega(\widehat{R}_{\mathcal{S}}(\widehat{h}_{\mathcal{S}})) + 2\sqrt{\frac{2d_{\mathcal{H}_{\mathcal{S}}}\log(2em_{\mathcal{S}}/d_{\mathcal{H}_{\mathcal{S}}}) + 2\log(8/\delta)}{m_{\mathcal{S}}}} \\ &\quad + 4\sqrt{\frac{2d_{h_{\mathcal{S}} \circ \Pi}\log(2em_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + 2\log(8/\delta)}{m_{\mathcal{T}}}} \end{aligned} \quad (3)$$

Proof. Let $\pi^* = \text{ArgMin}_{\pi \in \Pi} R_{\mathcal{T}}(h_{\mathcal{S}} \circ \pi)$. With probability at least $1 - \delta$:

$$\begin{aligned} R_{\mathcal{T}}(h_{\mathcal{S}} \circ \widehat{\pi}) &\leq \widehat{R}_{\mathcal{T}}(h_{\mathcal{S}} \circ \widehat{\pi}) + 2\sqrt{\frac{2d_{h_{\mathcal{S}} \circ \Pi}\log(2em_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + 2\log(8/\delta)}{m_{\mathcal{T}}}} \\ &\leq \widehat{R}_{\mathcal{T}}(h_{\mathcal{S}} \circ \pi^*) + 2\sqrt{\frac{2d_{h_{\mathcal{S}} \circ \Pi}\log(2em_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + 2\log(8/\delta)}{m_{\mathcal{T}}}} \\ &\leq R_{\mathcal{T}}(h_{\mathcal{S}} \circ \pi^*) + 4\sqrt{\frac{2d_{h_{\mathcal{S}} \circ \Pi}\log(2em_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + 2\log(8/\delta)}{m_{\mathcal{T}}}} \\ &\leq \omega(R_{\mathcal{S}}(\widehat{h}_{\mathcal{S}})) + 4\sqrt{\frac{2d_{h_{\mathcal{S}} \circ \Pi}\log(2em_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + 2\log(8/\delta)}{m_{\mathcal{T}}}} \\ &\leq \omega(\widehat{R}_{\mathcal{S}}(\widehat{h}_{\mathcal{S}})) + 2\sqrt{\frac{2d_{\mathcal{H}_{\mathcal{S}}}\log(2em_{\mathcal{S}}/d_{\mathcal{H}_{\mathcal{S}}}) + 2\log(8/\delta)}{m_{\mathcal{S}}}} \\ &\quad + 4\sqrt{\frac{2d_{h_{\mathcal{S}} \circ \Pi}\log(2em_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + 2\log(8/\delta)}{m_{\mathcal{T}}}} \end{aligned}$$

This follows from the fact that [10] (p. 48) using m training points and a hypothesis class of VC dimension d , with probability at least $1 - \delta$, for all hypotheses h simultaneously, the true risk $R(h)$ and empirical risk $\widehat{R}(h)$ satisfy $|(R(h) - \widehat{R}(h))| \leq 2\sqrt{\frac{2d\log(2em/d) + 2\log(4/\delta)}{m}}$. For $h_{\mathcal{S}} \circ \Pi$, this yields the first and third inequalities with probabilities at least $1 - \delta/2$. For $\mathcal{H}_{\mathcal{S}}$, this yields the fifth inequality with probability at least $1 - \delta/2$. Applying the union bound archives the desired results. The second inequality follows from the definition of $\widehat{\pi}$, and the fourth inequality is where we inject our assumption about the transferability (or proximity) between the source and the target problem. \square

We can thus control the generalization error on the transfer domain by controlling $d_{h_{\mathcal{S}} \circ \Pi}$, $m_{\mathcal{S}}$ and ω which measures the link between the domain and the target domain. The number of target training data $m_{\mathcal{T}}$ is typically supposed to be small in transfer learning and thus cannot be employed to control the error.

3.2 Boosting Projections from Target to Source

The above analysis bounds the generalization error of the learned target hypothesis $h_S \circ \hat{\pi}$ in terms, among others, of the VC dimension of the space $h_S \circ \Pi$. The problem of controlling the capacity of such a space of functions in order to prevent under or over-fitting is the same as in the traditional supervised learning setting. The difficulty lies in choosing the right space Π of projection functions from \mathcal{X}_T to \mathcal{X}_S .

The space of hypothesis functions considered is:

$$L(h_S \circ \Pi_B) := \left\{ \mathbf{x} \mapsto \text{sign} \left[\sum_{n=1}^N \alpha_n (h_S \circ \pi_n(\mathbf{x}^T)) \right] : \forall n, \alpha_n \in \mathbb{R}, \text{ and } \pi_n \in \Pi_B \right\}$$

where Π_B is a space of weak projections satisfying definition (1).

Now, from [11] (p. 109), the VC dimension of the space $h_S \circ \Pi_B$ satisfies:

$$d_{L(h_S \circ \Pi_B)} \leq N(d_{h_S \circ \Pi_B} + 1)(3 \log(N(d_{h_S \circ \Pi_B} + 1)) + 2)$$

If $d_{h_S \circ \Pi_B} \ll d_{h_S \circ \Pi}$, then $d_{L(h_S \circ \Pi_B)}$ can also be much less than $d_{h_S \circ \Pi}$, and theorem (1) provides tighter bounds.

Using the **TransBoost** method, we can thus gain both on the theoretical bounds on the generalization error and on the ease of finding an appropriate space of projections $\mathcal{X}_T \rightarrow \mathcal{X}_S$.

4 Design of the Experiments

4.1 The Main Dimensions of Experiments in Transfer Learning

There are two dimensions that can be expected to govern the efficiency of transfer learning:

1. The *level of signal* in the target data.
2. The *relatedness* between the source and the target domains.

Regarding the *first dimension*, one can expect that if there is no signal in the target data (i.e. the examples are labelled randomly), then no regularity can be extracted, directly or using transfer. In fact, only overfitting of the training data can potentially occur. If, on the contrary, the target learning task is easy, then there cannot be much advantage in using transfer learning. A question therefore arises as to whether there might be an optimal level of signal in the target data so as to maximally benefit from transfer learning.

The *second dimension* is tricky. Here, we intuitively expect that the closer the source and target domains (and problems), the more profitable transfer learning should be. However, how should we measure the “relatedness” of the source and target problems? In the domain adaptation setting, closeness can be measured through a measure of the divergence between the source distribution and the target one, since they are defined on the same input space. In transfer learning,

the input spaces can be different, so that it is much more difficult to define a divergence between distributions. This is why we resorted to the function ω in our theoretical analysis. In our experiments, we control relatedness through the information shared between source and target (see below).

4.2 Experimental Setup

In our study, we devised an experimental setup that would allow us to control the two dimensions above.

In the **target domain**, the learning task is to classify time series of length $t_{\mathcal{T}}$ into two classes: $h_{\mathcal{T}} : \mathbb{R}^{t_{\mathcal{T}}} \rightarrow \{-1, +1\}$. By controlling the level of noise and the difference between the distributions governing the two classes, we can control the signal level, that is the difficulty of extracting information from the target training data. We control the amount of information by varying the size $m_{\mathcal{T}}$ of the target training set.

Likewise, the source input space is the space of sequences of real measurements of length $t_{\mathcal{S}}$. Therefore, we have $h_{\mathcal{S}} : \mathbb{R}^{t_{\mathcal{S}}} \rightarrow \{-1, +1\}$.

Varying $|t_{\mathcal{S}} - t_{\mathcal{T}}|$ is a way of controlling the information potentially shared in the two domains. With $t_{\mathcal{S}} = t_{\mathcal{T}}$, the two input domains are the same.

Note that learning to classify times series is not a trivial task. It has many applications, some of them involving to classify time series of length different from the length for which exists a classifier.

4.3 Description of the Experiments

Time series were generated according to the following equation:

$$x_t = \underbrace{t \times \text{slope} \times \text{class}}_{\text{information gain}} + \underbrace{x_{max} \sin(\omega_i \times t + \varphi_j)}_{\text{sub shape within class}} + \underbrace{\eta(t)}_{\text{noise factor}} \quad (4)$$

The fact that the noise factor is generated according to a Gaussian distribution induces a distribution over the data (class $\in \{-1, +1\}$).

The **level of signal in the training data** is governed by:

1. the *slope factor*: the higher the value of the *slope* factor, the easier the discrimination between the two classes at each additional time step
2. the *number of different shapes* in each class of sequences, each shape controlled by ω_i and φ_j , and the importance of this factor in the equation being weighted by x_{max}
3. the *noise factor* $\eta(t)$
4. the *length* of the time series, that is the number of measurements
5. the *size* of the training set

In our experiments, the noise factor is generated according to a Gaussian distribution of mean = 0 and standard deviation in $\{0.001, 0.002, 0.02, 0.2, 1\}$.

Figure 2 illustrates what can be obtained with slope = 0.01 with 3 subclasses in the +1 class, and 2 subclasses in the -1 class.

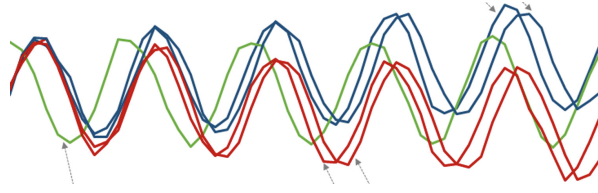


Fig. 2. A synthetic data set \mathcal{S} with 5 times series where η is Gaussian ($\mu = 0, \sigma = 0.2$).

In the experiments reported here, we kept the size of the training set constant. In each experiment, 900 times series of length 200 were generated according to the equation described above: 450 times series in each class -1 or $+1$. We varied the difficulty of learning by varying the slope from almost non existent: 0.001 to significant: 0.01. Similarly, we varied the length $t_{\mathcal{T}}$ of the target training set in $\{20, 50, 70, 100\}$ thus providing increasing levels of signal.

A *target training data set* of 300 time series was drawn equally balanced between the two classes. Note that this relatively small number corresponds to transfer learning scenarios where the training data is limited in the target domain. The remaining 600 time series were used as a *test set*. The source hypothesis was learned using the complete time series generated as explained above.

In these experiments, the *set of projections* Π was chosen as a set of “hinge functions”, defined by three parameters, the slope of the first linear part, the time t where the hinge takes place, and the slope of the second linear part. The set is explored randomly by the algorithm and a projection is retained if its error rate on the current weighted data is lower than 0.45. We explored other, richer, spaces of projections without gaining superior performances. This simple set seems to be sufficient for this learning task.

In order to better assess the value of **TransBoost**, its performance was compared (1) to a classifier (Gaussian SVM as implemented in Scikit Learn) acting directly on the target training data, (2) to a boosting algorithm operating in the target domain with base classifiers being Gaussian SVMs, and (3) to a baseline transfer learning method that consists in finding a regression from the target input space to the source input space using a SVR regression. In this last method the regression acts as a translation from $\mathcal{X}_{\mathcal{T}}$ to $\mathcal{X}_{\mathcal{S}}$ and the class of an example $\mathbf{x}^{\mathcal{T}}$ is given by $h_{\mathcal{S}}(\text{regression}(\mathbf{x}^{\mathcal{T}}))$.

Table 1 provides representative examples of the results obtained. Each cell of the table shows the average performance (and the standard deviations) computed from 100 experiments repeated under the same conditions. The experimental conditions are organized according to the level of signal in the training data. In the experiments corresponding to this table, the source hypotheses were learned according to the first protocol defined above.

Several lessons can be drawn. First of all, in most situations, **TransBoost** brings *very significant gains* over learning without transfer or using transfer learning with regression. Figures 3 and 4 that sum up a larger set of experimental

Table 1. Comparison of the error rate (lower is better) between: learning directly in the target domain (columns $h_{\mathcal{T}}$ (train) and $h_{\mathcal{T}}$ (test)), using **TransBoost** (columns $H_{\mathcal{T}}$ (train) and $H_{\mathcal{T}}$ (test)), learning in the source domain (column $h_{\mathcal{S}}$ (test)) and, finally, mapping the time series with a SVR regression and using $h_{\mathcal{S}}$ (naïve transfer, column $H'_{\mathcal{T}}$ (test)). Test errors are highlighted in the orange columns. Bold numbers indicate where **TransBoost** significantly dominates both learning without transfer and learning with naïve transfer.

slope, noise, $t_{\mathcal{T}}$	$h_{\mathcal{T}}$ (train)	$h_{\mathcal{T}}$ (test)	$H_{\mathcal{T}}$ (train)	$H_{\mathcal{T}}$ (test)	$h_{\mathcal{S}}$ (test)	$H'_{\mathcal{T}}$ (test)
0.001, 0.001, 20	0.46 ± 0.02	0.50 ± 0.08	0.08 ± 0.03	0.08 ± 0.02	0.05	0.49 ± 0.01
0.005, 0.001, 20	0.46 ± 0.02	0.49 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	0.01	0.45 ± 0.01
0.005, 0.002, 20	0.46 ± 0.02	0.49 ± 0.03	0.03 ± 0.02	0.04 ± 0.02	0.02	0.43 ± 0.01
0.005, 0.02, 20	0.44 ± 0.02	0.48 ± 0.03	0.09 ± 0.01	0.10 ± 0.01	0.01	0.47 ± 0.01
0.001, 0.2, 20	0.46 ± 0.02	0.50 ± 0.01	0.46 ± 0.02	0.51 ± 0.02	0.11	0.49 ± 0.01
0.01, 0.2, 20	0.42 ± 0.03	0.47 ± 0.03	0.34 ± 0.02	0.35 ± 0.02	0.02	0.35 ± 0.01
0.001, 0.001, 50	0.46 ± 0.02	0.50 ± 0.01	0.08 ± 0.03	0.08 ± 0.02	0.06	0.41 ± 0.01
0.005, 0.001, 50	0.25 ± 0.07	0.28 ± 0.09	0.01 ± 0.01	0.01 ± 0.01	0.01	0.28 ± 0.01
0.005, 0.002, 50	0.27 ± 0.07	0.30 ± 0.08	0.02 ± 0.01	0.02 ± 0.01	0.02	0.28 ± 0.01
0.005, 0.02, 50	0.26 ± 0.07	0.30 ± 0.08	0.04 ± 0.01	0.04 ± 0.01	0.01	0.31 ± 0.01
0.001, 0.2, 50	0.44 ± 0.02	0.50 ± 0.01	0.38 ± 0.03	0.44 ± 0.02	0.15	0.43 ± 0.01
0.01, 0.2, 50	0.10 ± 0.03	0.12 ± 0.04	0.10 ± 0.02	0.11 ± 0.02	0.03	0.15 ± 0.02
0.001, 0.001, 100	0.43 ± 0.03	0.47 ± 0.03	0.07 ± 0.02	0.07 ± 0.02	0.02	0.23 ± 0.01
0.005, 0.001, 100	0.06 ± 0.03	0.07 ± 0.03	0.01 ± 0.01	0.01 ± 0.01	0.01	0.07 ± 0.02
0.005, 0.002, 100	0.08 ± 0.03	0.10 ± 0.04	0.02 ± 0.01	0.02 ± 0.01	0.02	0.07 ± 0.01
0.005, 0.02, 100	0.08 ± 0.03	0.09 ± 0.03	0.02 ± 0.01	0.03 ± 0.01	0.01	0.07 ± 0.01
0.001, 0.2, 100	0.04 ± 0.03	0.46 ± 0.02	0.28 ± 0.02	0.31 ± 0.01	0.16	0.31 ± 0.01
0.01, 0.2, 100	0.03 ± 0.01	0.05 ± 0.02	0.04 ± 0.01	0.05 ± 0.01	0.02	0.05 ± 0.01

conditions make this even more striking. In both tables, the x -axis reports the error rate obtained using **TransBoost**, while the y -axis reports the error rate of the competing algorithm: either the hypothesis $h_{\mathcal{T}}$ learnt on the target training data alone (Fig. 3), or the hypothesis $H'_{\mathcal{T}}$ learned on the target data projected on the source input space using a SVR regression (Fig. 4). The remarkable efficiency of **TransBoost** in a large spectrum of situations is readily apparent.

Secondly, as expected, **Transboost** is less dominant when either the data is so noisy that no method can learn from the data (high level of noise or low slope): this is apparent on the right part of the graphs 3 and 4 (near the diagonal), or when the task is so easy (large slope and/or low noise) that nothing can be gained from transfer learning (left part of the two graphs).

We did not report here the results obtained with boosting directly in the target input space $\mathcal{X}_{\mathcal{T}}$ since the learning performance was almost the same as the performance as the one of the SVM classifier. This shows that this is not boosting in itself that brings a gain.

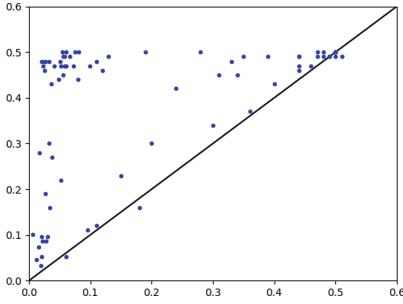


Fig. 3. Comparison of error rates. y -axis: test error of the SVM classifier (without transfer). x -axis: test error of the TransBoost classifier with 10 boosting steps. The results of 75 experiments (each one repeated 100 times) are summed up in this graph.

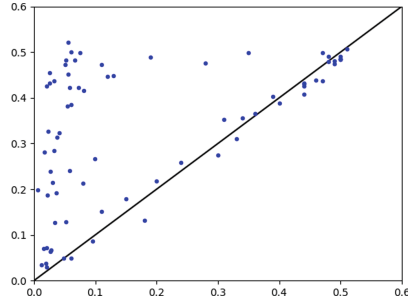
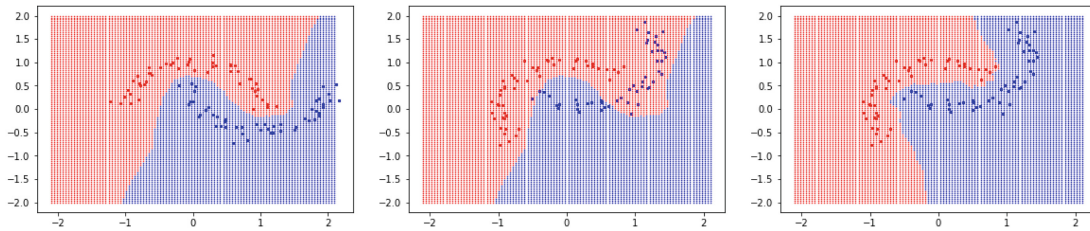


Fig. 4. Comparison of error rates. y -axis: test error of the “naive” transfer method. x -axis: test error of the TransBoost classifier with 10 boosting steps. The results of 75 experiments (each one repeated 100 times) are summed up in this graph.

4.4 Additional Experiments

We show here, in Figs. 5, 6 and 7 qualitative results obtained on the classical half-moon problem. It is apparent that **Transboost** brings satisfying results.



(a) kNN source model trained on the data source : it fits to the data the source
 (b) kNN source model trained on the data target : it does not fit to the data source
 (c) kNN source model trained on the data target : it fits to the data target

Fig. 5. Experiments on the half-moon problem.

5 Comparison to Previous Works

In the theoretical analysis of Ben-David *et al.* [1,2], one central idea is that a *common representation space* should be found in which the projections of the source data $\{(\mathbf{x}_i^S)\}_{1 \leq i \leq m}$ and of the target data $\{(\mathbf{x}_i^T)\}_{1 \leq i \leq m}$ should be as undistinguishable as possible using discriminative functions from the hypothesis space \mathcal{H} . The intuition is that if the domains become indistinguishable, a classifier constructed for the source domain should work also for the target domain. It has been at the core of many proposed methods so far [3,5,7,12].

In [8] a scenario in which multiple sources are available for a single target domain is studied. For each source $i \in \{1, \dots, k\}$, the input distribution D_i is

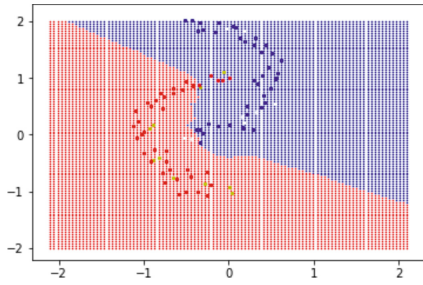


Fig. 6. A KNN model trained on the few target data points (in yellow). (Color figure online)

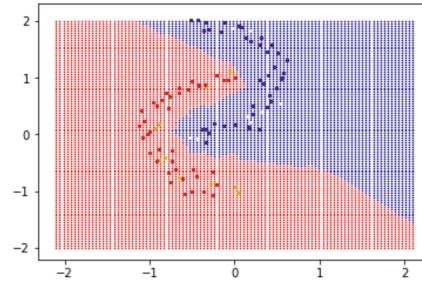


Fig. 7. A KNN model transboosted on the few target data points.

known as well as a hypothesis h_i with loss bounded by ε on D_i . It is further assumed that the target input distribution is a mixture of the k source distributions D_i . The adaptation problem is thus seen as finding a combination of the hypotheses h_i . It is shown that guarantees on the loss of the combined target hypothesis can be given for some forms of combinations. However, the authors do not show how to learn the parameters of these combinations. In [4], the authors present a system called **TrAdaboost**, which uses a boosting scheme to eliminate data points that seem irrelevant for the new task defined over the same space \mathcal{X} . Despite the use of boosting, the scope is quite different from ours.

Finally, the authors in [6] study a scheme seemingly very close to ours. They define *Hypothesis Transfer Learning algorithms* as algorithms taking as input a training set in the target domain and a source hypothesis in the source domain, and producing a target hypothesis:

$$A^{\text{htl}} : (\mathcal{X}_T \times \mathcal{Y}_T)^m \times \mathcal{H}_S \rightarrow \mathcal{H}_T \subseteq \mathcal{Y}^{\mathcal{X}}$$

One goal of the paper is to identify the effect of the source hypothesis on the generalization properties of A^{htl} . However, the scope of the analysis is limited in several ways. First, it focusses on linear regression with the Regularized Least Square algorithm. Second, the formal framework necessitates that in fact $\mathcal{X}_T = \mathcal{X}_S$ and $\mathcal{Y}_T = \mathcal{Y}_S$. It is thus more an analysis of domain adaptation than of transfer learning. Third, the transfer learning algorithm in effect tries to find a weight vector \mathbf{w}^T as close as possible to the source weight vector \mathbf{w}^S while fitting the target data set. There is therefore a parameter λ to set. More importantly, the consequence is that the analysis singles out the performance of the source hypothesis on the target domain as the most significant factor controlling the expected error on the target problem. Again, therefore, the target hypothesis cannot be much different from the source one, which seems to defeat the whole purpose of transfer learning.

6 Conclusion

This paper has presented a new transfer learning algorithm, **TransBoost**, that uses the boosting mechanism in an original way by selecting and combining weak

projections from the target domain to the source domain. The algorithm inherits some nice features from boosting. There is only one parameter to set: the number of boosting steps, and guarantees on the training error and on the test error are easily derived from the ones obtained in the theory of boosting.

References

1. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. *Mach. Learn.* **79**(1), 151–175 (2010). <https://doi.org/10.1007/s10994-009-5152-4>
2. Ben-David, S., Blitzer, J., Crammer, K., Pereira, F., et al.: Analysis of representations for domain adaptation. In: *Advances in Neural Information Processing Systems*, vol. 19, p. 137 (2007)
3. Bickel, S., Brückner, M., Scheffer, T.: Discriminative learning for differing training and test distributions. In: *Proceedings of the 24th International Conference on Machine Learning*, pp. 81–88. ACM (2007)
4. Dai, W., Yang, Q., Xue, G.R., Yu, Y.: Boosting for transfer learning. In: *Proceedings of the 24th International Conference on Machine Learning*, pp. 193–200. ACM (2007)
5. Jiang, J., Zhai, C.: Instance weighting for domain adaptation in NLP. In: *ACL*, vol. 7, pp. 264–271 (2007)
6. Kuzborskij, I., Orabona, F.: Stability and hypothesis transfer learning. In: *ICML* (3), pp. 942–950 (2013)
7. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: learning bounds and algorithms. *arXiv preprint [arXiv:0902.3430](https://arxiv.org/abs/0902.3430)* (2009)
8. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation with multiple sources. In: *Advances in Neural Information Processing Systems*, pp. 1041–1048 (2009)
9. McNamara, D., Balcan, M.F.: Risk bounds for transferring representations with and without fine-tuning. In: *International Conference on Machine Learning*, pp. 2373–2381 (2017)
10. Mohri, M., Rostamizadeh, A., Talwalkar, A.: *Foundations of Machine Learning*. MIT Press, Cambridge (2012)
11. Shalev-Shwartz, S., Ben-David, S.: *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, Cambridge (2014)
12. Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P.V., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: *Advances in Neural Information Processing Systems*, pp. 1433–1440 (2008)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

