Some thoughts about transfer learning. What role for the source domain?

Antoine Cornuéjols

Some thoughts about transfer learning. What role for the source domain?

A. Cornuéjols

UMR MIA-Paris-Saclay, AgroParisTech, INRAe,
Université Paris-Saclay,
91120, Palaiseau, France
antoine.cornuejols@agroparistech.fr

Abstract

Transfer learning is called for when the training and test data do not share the same input distributions ($\mathbf{P}^{\mathcal{S}}_{\mathcal{X}} \neq \mathbf{P}^{\mathcal{T}}_{\mathcal{X}}$) or/and not the same conditional ones ($\mathbf{P}^{\mathcal{S}}_{\mathcal{Y}|\mathcal{X}} \neq \mathbf{P}^{\mathcal{T}}_{\mathcal{Y}|\mathcal{X}}$). In the most general case, the input spaces and/or output spaces can be different: $\mathcal{X}_{\mathcal{S}} \neq \mathcal{X}_{\mathcal{T}}$ and/or $\mathcal{Y}_{\mathcal{S}} \neq \mathcal{Y}_{\mathcal{T}}$. However, most work assume that $\mathcal{X}_{\mathcal{S}} = \mathcal{X}_{\mathcal{T}}$. Furthermore, a common held assumption is that it is necessary that the source hypothesis be good on the source training data and that the "distance" between the source and the target domains be as small as possible in order to get a good (transferred) target hypothesis.

This paper revisits the reasons for these beliefs and discusses the relevance of these conditions. An algorithm is presented which can deal with transfer learning problems where $\mathcal{X}_{\mathcal{S}} \neq \mathcal{X}_{\mathcal{T}}$, and that furthermore brings a fresh perspective on the role of the source hypothesis (it does not have to be good) and on what is important in the distance between the source and the target domains (translations between them should belong to a limited set). Experiments illustrate the properties of the method and confirm the theoretical analysis.

Determining beforehand a relevant source hypothesis remains an open problem, but the vista provided here helps understanding its role.

Keywords: Transfer learning, Domain Adaptation, Out of Distribution Learning

1. Introduction

Inference goes beyond querying a database to answer a question that is not explicitly answered there. It uses formal rules to derive conclusions, possibly unknown before the query, from premises. In this way, inference "completes" existing knowledge by creating new facts or beliefs. This is a multi-step process, using formal derivations composed of elementary inference steps, such as *modus ponens* or *ET-elimination*. **Deduction** is a logically correct inference. From true premises, it is guaranteed to lead to true conclusions. If we know that *John is either at home or at work*, and the premises indicate that *John is not at home*, we can conclude that he must be at work. A second type of inference is **abduction**. It is at the heart of the generation of explanations. To a first approximation, its scheme is as follows: from the knowledge that b and *if a then b*, abduction infers that a. In both cases, deduction and abduction, a derivation is a sequence of mechanical applications of elementary rules of inference. But as mechanical as it is, producing a unique derivation usually requires the rejection of a large number of alternative deductions, and making the right choices is one of the key problems in automating the inference process. While, in principle, only the computational complexity of the inference is affected by the exploration strategy, and not its results, this is not the case for abduction where different strategies can lead to different results.

In the **induction** framework, we seek to answer questions using a different form of knowledge and other types of inference mechanisms. If we focus on supervised induction, we are given a learning set $\{(\mathbf{x}_i, y_i)\}_{1 \leq i \leq m}$ of pairs where the $\mathbf{x}_i$ belong to an input space $\mathcal{X}$ and the associated y_i belong to an output space $\mathcal{Y}$. The question is now: how should we use this learning set to answer possible queries of the form "$x \in \mathcal{X} \rightarrow$?" where $\mathbf{x} \neq \mathbf{x}_i, \forall i \in \{1, \ldots, m\}$?

In order to get a feeling for the problem, it may be useful to consider a simple situation. Suppose that the input space $\mathcal{X}$ is described using three binary descriptors: a_1, a_2 and a_3 and that the output space is itself binary $\mathcal{Y} = \{0, 1\}$.

There are thus $2^3 = 8$ possible examples, and $2^8 = 256$ different ways of labelling these 8 examples, which means that there are 256 possible binary functions f over these examples, where $y = f(\mathbf{x})$ for a given example $\mathbf{x}$. Now suppose that 5 out of the 8 possible examples are provided with their associated output values in the training set (see Figure 1).

a_1	a_2	a_3	$f(\mathbf{x})$
0	0	0	+
0	0	1	−
0	1	0	+
0	1	1	**?**
1	0	0	+
1	0	1	**?**
1	1	0	**?**
1	1	1	−

Figure 1: *Suppose that an oracle has chosen the function f in order to label the examples, and that 5 examples with their associated labels have been provided in the training set.*

How should we answer the query for one unseen example, like $(0, 1, 1)$? The classical approach is to suppose that the same function that gives the labels of the known examples is used for the unseen ones. However, since there are 3 examples for which we do not know the answer, there remains $2^3 = 8$ possible functions, each one making no errors on the 5 known examples. And, of these 8 functions, 4 predict the answer '+' for the query $(0, 1, 1)$, and 4 predict '−'. Without additional knowledge, even in this simpliest of problems, we can not decide. Even worse, there exists a function among the remaining candidates that perfectly predicts the training examples and would be wrong on all unseen examples! So that the performance on the training data says nothing about the performance to be expected on the unseen examples.

If, however, an "expert" tells us that only descriptors a_1 and a_3 are relevant, then there are only 2^4 possible functions, and from the training examples, we get the following table where f is the only function defined over a_1 and a_3 satisfying the training examples:

a_1	a_3	$f(\mathbf{x})$
0	0	+
0	1	−
1	0	+
1	1	−

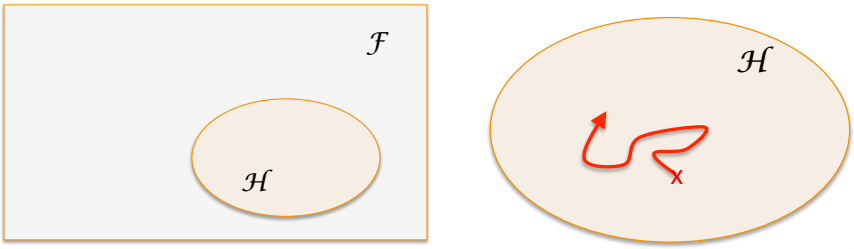
Following this, the only answer for $(0, 1, 1)$ then is '−'. Induction has been made possible because of the limitation on the hypothesis space that the learner must explore.

In the vocabulary of machine learning, such a limitation on the space of candidate hypotheses is called a *bias*. A distinction is made between "representation bias" where the space $\mathcal{H}$ is limited a priori, and "search bias" where it is the way the space of hypotheses is explored which induces a preference between candidate hypotheses. Of course, these two biases can be combined (see Figure 2). While the representation bias is well explored in the statistical learning theory[1], this is not the case for the search bias.

Classical induction is thus a machine that takes as input a set $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{1 \leq i \leq m}$ of training examples and some prior knowledge under the form of biases limiting the space of hypotheses and that outputs an hypothesis $h : \mathcal{X} \to \mathcal{Y}$ that allows one to get an answer $y \in \mathcal{Y}$ for any query $x \in \mathcal{X}$.

Note that there exists also *transduction* which is a machine that takes as input a set $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{1 \leq i \leq m}$ of training examples, some prior knowledge in the form of biases and a query $\mathbf{x}$ and that produces an answer $y \in \mathcal{Y}$ for that specific query. As described by Vapnik [1], transductive inference is "moving from particular to particular". It can

[1] And under the strong assumption that both the training examples and the test examples are drawn randomly from the same distribution, aka the i.i.d. (identically and independently distributed) assumption.



(a) (b)

Figure 2: (a) The *representation bias* limits the space of candidate functions $\mathcal{H}$ to a subset of $\mathcal{F}$, the space of all functions from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$. (b) The *search bias* induces an order on the candidate hypotheses that are considered by the learner.

thus be seen as a case of transfer learning, from some training examples to specific queries. What is interesting is that in this case, as strongly underlined by Vapnik, there is no need to search for an hypothesis, a function $h : \mathcal{X} \to \mathcal{Y}$ able to answer any question in the input space $\mathcal{X}$. Less information is thus necessary to answer specific queries (see also the chapter 24 from Vapnik in [2] for a more thorough discussion). This is in line with remarks from Sutton et al. [3] (see Section 2 below).

In **transfer learning**, the function f (or conditional dependence $\mathbf{p}_{Y|X}$ if we adopt the perspective of statistics) that associates outputs y_i with inputs $\mathbf{x}_i$ in the training set is no longer expected to be the correct one to produce answers to future queries. Since the distribution $\mathbf{p}_{XY}$ can be decomposed as $\mathbf{p}_{Y|X}\,\mathbf{p}_X$, the reasons for this situation are usually distinguished as follows:

1. A change in the distribution of data $\mathbf{p}_X$. For instance, disease properties and symptoms are the same in winter and summer, but the distribution of pathologies differs. Or there might be a change in the lighting or in vision applications. This is variously called *covariate shift*, *data drift*, *dataset shift* or *domain adaptation*
2. A change is the labelling function or the conditional distribution $\mathbf{p}_{Y|X}$. For instance, the definition of 'fraudulent' may change over time. According to different authors and/or times, this has been called *concept shift* or *concept drift*. However, the latter is sometimes associated with a change in $\mathbf{p}_{XY}$.
3. A change in $\mathbf{p}_{XY}$, the most general case where the distribution of the examples can change, as well as the labelling function. In some scenarios, the input space and/or the output space differ(s) between the "source" domain and the "target" one. In this paper, we refer to this general situation as *transfer learning*.

One further distinction is between the scenario where the training source data $S_{\mathcal{S}} = \{(\mathbf{x}^i_{\mathcal{S}}, y^i_{\mathcal{S}})\}_{1\leq i\leq m}$ is available and one where only the source hypothesis $h_{\mathcal{S}}$ is known. The latter case is called *Hypothesis Transfer Learning*.

Supervised Transfer learning is thus a machine that takes as input some source information, under the form of source data and/or a source hypothesis, a target domain $\mathcal{X}_{\mathcal{T}} \times \mathcal{Y}_{\mathcal{T}}$ and some target data, unsupervised or supervised, and that outputs an hypothesis $h_{\mathcal{T}} : \mathcal{X}_{\mathcal{T}} \to \mathcal{Y}_{\mathcal{T}}$ that allows one to get an answer $y^{\mathcal{T}} \in \mathcal{Y}_{\mathcal{T}}$ for any query $\mathbf{x}_{\mathcal{T}} \in \mathcal{X}_{\mathcal{T}}$.

Transfer learning is used when the target training data is not sufficient to select a target hypothesis $h_{\mathcal{T}}$ with enough confidence in its performance.

In the case of classical induction, where the environment is stationary, it seemed legitimate to assume that the same function that "explains" the training data is also the function that correctly predicts the results for the unseen examples. The statistical theory of supervised induction showed that this assumption is indeed reasonable as far as there are constraints on the set of hypotheses that the learner can consider, the biases mentioned above. Then, the performance of a hypothesis measured on the training data can be expected to be representative of its performance on the unseen examples.

But what should be assumed in the case of transfer learning? How can the source information help focus the search for a target hypothesis?

It is generally expected that the following ingredients play a role in the success or not of transfer learning:

1. The *performance of the source hypothesis* on the source domain.
2. The "*distance*" between the source and target domains.

3. The amount of *training data in the target domain*. The larger this quantity, the less interesting it is to use transfer learning.

This paper explores the role of the first two ingredients.

Specifically, (1) why should we expect that it is *necessary* for the source hypothesis to predict the source data well in order to help find a good hypothesis in the target domain? And (2) what should be the relationship between the source hypothesis and the function supposed to be at play in the target domain?

This paper is organized along some key ideas that have structured theories and the search for algorithmic approaches to transfer learning. In Section 2, we describe how the idea of *minimal adaptation* came about. Related to this idea, but still distinct, is the perspective, described in Section 3, that a *common representation space* should be sought between the source and target domains in order to achieve good transfer learning. At the same time, another principle, described in Section 4 looks at transfer learning *in the light of statistical learning* and looks for biases that might contribute to limiting the space of hypotheses to consider. Section 5 then presents an algorithm, original in its design, which sheds new light on the conditions for transfer learning, and specially on the role of the source hypothesis. It also proposes a new way to relate the source and target domain. The conclusion, in Section 6, underlines the lessons that can be drawn from this survey.

2. Transfer and the idea of minimal adaptation

... or why is it that a small "distance" between the source and the target is deemed to be a key factor for a successful transfer?

There is a common idea that when trying to solve a problem in a target domain using transfer learning, first of all it is important that the source domain be related to the target one, meaning not be too different or too distant from the target one and, second, that minimally adapting the solution (e.g. a hypothesis with low error rate) from the source domain is the way to go. It is then believed that doing so provides guarantees about the performance of the transferred solution (e.g. hypothesis). For instance, this view is quite prominent in the recent works that rely on the concept of "minimal transport" in order to both measure the distance between the source and target domains and to allow for the transport from the source solution to a target one.

During the pioneering days of inductive learning, the days of the perceptrons in the sixties, or later, in the seventies, of the version space and the candidate elimination algorithm, the unquestioned belief was that the hypothesis that had the best performance on the training set was to be also the best one for future data. Then came Vapnik and Valiant who showed independently [4, 1] that this needs to be qualified and that the criterion to be optimized must incorporate other factors than the empirical risk in order to have guarantees on the performance on unseen examples, aka the real risk.

Should we question in the same spirit the current common wisdom about transfer learning? More specifically, why are we more confident about the performance on the transferred solution if the source and target domains are "closed" to each other? Which begs the question: how to measure this distance? Why is it that we usually measure it using statistical tools? Finally, what is the nature of the theoretical guarantees we get using these tools?

It is always interesting to look at the limit cases when studying a phenomenon. One limit case in transfer learning is when the source and target domains are the same. The input spaces and the output spaces are the same and the underlying distributions of the examples also. Then what the theory tells about the performance to be expected?

We consider here the simplest case because it provides the gist of the message, which is the same in the more general one, but at the risk of being obscured by many technical terms.

Let us consider a hypothesis space $\mathcal{H}$ that is a enumerable set with cardinal $|\mathcal{H}|$ and m identically and independently distributed labelled examples used as a training set. Then the expected error rate of an hypothesis h, $\mathrm{err}(h)$, can be bounded by its error rate on the training set, $\widehat{\mathrm{err}}(h)$, as follows:

$$\forall h \in \mathcal{H}, \forall \delta \leq 1 \quad : \mathbf{P}^m\left[\mathrm{err}(h) \;\leq\; \widehat{\mathrm{err}}(h) \;+\; \frac{\log|\mathcal{H}| \;+\; \log\frac{1}{\delta}}{m}\right] \;>\; 1 \;-\; \delta \tag{1}$$

This means that the true error rate of h is bounded in probability by its empirical error rate on the training set plus a term that depends on the richness of $\mathcal{H}$, here measured by its cardinal, the number of training examples (the larger,

the tighter the bound) and a δ term that appears also in the overall probability. Indeed, the bound, hence the guarantee on the performance to be expected, can only be in probability, controlled by the δ factor. This comes from the fact that one cannot guarantee that the training set be "representative" of the underlying distribution of the examples. Even if this is unlikely, it could for instance happen that all the training examples had the same label. Then, one cannot ask the learner to provide an hypothesis with guaranteed performance. Thus, the δ parameter expresses that we want the guarantee to hold for more than $1 - \delta$ times all training samples that are drawn identically and independently from the underlying distribution. The smaller δ, the more demanding we are regarding the trueness of the bound. Thus, the lower is δ, the more we are willing to accept training distributions that differ from the testing distribution at the cost of being less certain about the bound.

Let us now try to interpret this in the light of transfer learning. In a way, δ encodes the distance between the distribution encountered during training and the true distribution, the one that will be encountered during testing. If, for some reasons, the training domain (i.e. source) differs from the testing one (i.e. target), then we end up with weak guarantees. And, conversely, it seems that in order to have strong guarantees, we need to have the training distribution as close as possible to the testing one.

In the line of this PAC (Probably Approximately Correct) learning framework, one paper [5] and variants of it have been very influential. It starts with a paragraph that states "(...), we expect the task performance in the target domain to depend on both the *performance in the source domain* and the *similarity* between the two domains" (italics mine). And then, it goes on with a derivation of a bound on the expected error on the target domain for any hypothesis given the expected error on the source domain, a term that measures a divergence between the source input distribution $\mathbf{p}^{\mathcal{S}}_{\mathcal{X}\mathcal{Y}}$ and the target input distribution $\mathbf{p}^{\mathcal{T}}_{\mathcal{X}\mathcal{Y}}$, a term similar to the one encountered in Equation (1) and a term λ which is the minimal combined error on the source and target domains for any hypothesis: let $h^\star = \text{ArgMin}_{h\in\mathcal{H}}(\epsilon_T(h) + \epsilon_S(h))$, then $\lambda = \epsilon_S(h^\star) + \epsilon_T(h^\star)$.

Formally, the theorem, in [5], states precisely that:

> Let $\mathcal{H}$ be a hypothesis space of VC-dimension d [1] and $\mathcal{U}_S, \mathcal{U}_T$ be unlabeled samples of size m' each, drawn from $\mathbf{P}^{\mathcal{S}}_{\mathcal{X}}$ and $\mathbf{P}^{\mathcal{T}}_{\mathcal{X}}$, respectively. Let $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}$ be the empirical distance on $\mathcal{U}_S, \mathcal{U}_T$, induced by the symmetric difference hypothesis space. With probability at least $1 - \delta$ (over the choice of the samples), for every $h \in \mathcal{H}$,
>
> $$\epsilon_T(h) \;\leq\; \epsilon_S(h) \;+\; 4\sqrt{\frac{2d\log(2m') + \log(\frac{4}{\delta})}{m'}} \;+\; \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T) \;+\; \lambda$$
>
> where e is the base of the natural logarithm.

Several remarks are worth mentioning here:

- The theorem is valid for sources and targets that share the same input space $\mathcal{X}$.
- It is limited to binary classification tasks (e.g. $\mathcal{Y} = \{0, 1\}$).
- It provides guarantees for the case where the same hypothesis h is used both in the source and in the target domains, which corresponds to a very crude transfer.
- The divergence term supposes that there are enough unlabeled data points in the source and target domains to be evaluated with the required accuracy. Analogy making, which can be considered as a limit case of transfer learning, is thus an instance where the theorem is inoperative.
- Finally, the theorem provides a sufficient condition, not a necessary one.

Even though there are limits to the analysis reported in [5] and in subsequent papers (see for instance [6]), and that the authors of the two papers make clear that their works deal only with domain adaptation where the distributions $\mathbf{P}^{\mathcal{S}}_{\mathcal{X}}$ and $\mathbf{P}^{\mathcal{T}}_{\mathcal{X}}$ may differ but are defined over the same input space $\mathcal{X}$ and where the labelling function is the same, the impression was that, in general for having a low error on the target domain, it was necessary (*i*) to have a hypothesis with low error on the source domain, and (*ii*) a small distance (or divergence) between the source and target domains.

The same types of bounds, but using optimal transport between distributions instead of the divergence of previous work[2], similarly relate the error of a hypothesis h on the target domain and the error of h on the source domain plus a term that evaluates the distance between the two domains. (See for instance [7]).

In summary, the theoretical analyses of transfer learning underlie the role of the distance between the source and the target domains in the performance of the transferred hypothesis. Its statistical nature leads also to expressing a supervised learning task as a probability distribution $\mathbf{P}_{\mathcal{X}\mathcal{Y}}$. It then becomes natural to look for tools that allow to compare probability distributions in order to evaluate the distance between domains.

Aside from these formal approaches of transfer learning, there are also heuristic reasons that tend to support enhancing the importance of a small distance between source and target.

One source might stem from the continuity of our experience in the world, what is defined as "temporal consistency" in [3]: "The right answers in nearby times may tend to be similar: in other words, the world maybe temporally coherent". Changes in the world from one moment to the next are usually very small, so that we only need to modify minimally our expectations and our decisions rules. In the more formal setting of online learning, the same assumption is applied.

In the online learning scenario, the learner is supposed to observe a succession of inputs $\mathbf{x}_t$, for each of which it must produce an output $\hat{y}_t$, usually using a current hypothesis h_t ($\hat{y}_t = h_t(\mathbf{x}_t)$) before receiving the true output y_t. In the case where the prediction at time t is incorrect: $\hat{y}_t \neq y_t$, the learner may chose to adapt its current hypothesis h_t. But except in the case of adversarial online learning, studied in for instance [8], the assumption is that the change of environment is limited from one moment to the next, as in the case of forecasting tomorrow's temperature. Therefore, the modifications envisioned for the current hypothesis are accordingly minimal, as for instance for the perceptron algorithm and more generally for most of the heuristically oriented online learning algorithms.

To sum up, both the formal analyses of transfer learning up to date and heuristic reasons point to the importance of having a small distance between the source and target in order to succeed. However, the stance on probability distributions and the associated divergence like measures of distance, may mislead us and blind us to other means of measuring what counts in transferring useful information from one domain or task to another one.

3. Transfer and the idea of indistinguishability

... or why is it that finding an intermediate representation space that renders the source and target data indistinguishable is so attractive for transfer learning?

The analysis sketched above serves as a justification for a significant amount of work of a heuristic nature. Indeed, if performance in the target task depends in part on the distance between the representation space of the source task and the representation space of the target task, then one should try to reduce it as much as possible, or even bring it to zero.

Reinforcing this conviction, a lot of recent work with neural networks, in computer vision noticeably, shows that the features learned for one task, say recognizing animals in the wild, remain useful for another recognition task, say detecting pedestrians in streets, and that this is therefore what should be transferred.

And even though some authors [9] warn that finding an invariant representation and getting a small error at the source is not enough to guarantee a small error at the target, the conventional wisdom remains intact.

In this perspective, researchers have proposed to jointly learn to reduce the gap between the representations of the source and target data (using a GAN approach for example) while still maximizing the classification performance (see the pioneering work [10]).

3.1. A general representation space?

At the same time that transfer learning was predominant in the field of computer vision, with methods trying to reuse as much as possible the features or parameters learned in a source task in order to solve a new task, with some additional fine tuning using target data, new developments in Natural Language Processing (NLP) were changing the

[2] The novelty is essentially that the geometry of the distributions is taken into account in optimal transport by means of an associated cost function which is based on the Euclidean distance between examples.

game. Prior to the advent of language models and generative methods, tasks such as question answering, sentiment analysis, machine translation, reading comprehension, and summarization were typically approached using specific algorithms trained on specific data sets. When trying to solve several tasks, the idea, as emphasized above, was to reuse a system pretrained on a given task and perform supervised fine-tuning to solve the new task.

However, in the text domain particularly and then in others as well, it has been recognized that context could be used as a source of supervisory signal, thus removing the need for large annotated data sets. Given a large text corpus, the principle is to learn a feature space, aka embedding, such that by using it, it becomes easy to predict the words in the context, e.g. the next word or a missing word. This intermediate representation has been shown to be useful in a number of different tasks without the need to adapt it to each one of them. This is one major realization, not foreseen initially but with consequences that are still being explored.

The evolution of ideas within this line of work has included self-taught learning in vision, Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs) and diffusion generative models. Language models such as BERT and GPT are eminent examples of recent achievements in this path. And this has proven to be key for improving the performance in a wide variety of NLP tasks.

In this series of works, the key idea is that intermediate representations which capture parts of the structure of the examples, are important for succeeding at many targeted tasks in a zero-shot setting, that is without any subsequent training in these tasks. This is why this approach is now generalized to many domains, like Computer Vision, Graph Learning, Speech understanding, Video analysis, code generation, etc.

One lesson to be learned here is that what is deemed important for the successful transfer of knowledge from the domain where the representation was learned to a new domain, is that the relationships captured about the "objects" in the original domain are relevant to the target one. This certainly encompasses many transfer learning tasks, such as the famous analogy between the solar system and the atomic model. If it doesn't exhaust the range of transfer learning tasks where domains concern different "objects", the idea of a general representation space broadens the way we can think about the notion of distance between domains.

Another line of research points to the importance of learning a representation space based on what is relevant for performing learning tasks in different contexts.

3.2. Invariant Risk Minimization: searching for invariant causal relationships

Under the covariate shift assumption, the marginal distribution $\mathbf{P}_X$ may change from the training environment to the test one, while the same conditional distribution $\mathbf{P}_{Y|X}$ is shared. Many techniques have been devised in order to tackle this problem. One common approach consists in "importance reweighting" which consists in increasing the weights of the training examples that are similar to the known test instances. Basically, each of the training example is reweighed by the relative probability of the training and test set before using importance-weighted empirical risk minimization. The latter can be estimated by density estimation, through kernel methods such as kernel mean matching, or through discriminative reweighing. One drawback is that density estimation must rely on making assumptions about the distributions.

Another approach, motivated by the absence of a prior assumption about the distribution, is to search a common representation space via techniques such as adversarial domain adaptation.

A more theory grounded approach underlines the difference between the causal factors or properties that "explain" the membership of an example to a category and other environmental features that are accidental to a specific environment. An archetypical example is the problem of classifying cows and camels. The former tend to be photographed in green pastures, while most pictures of camels happen to be in deserts. One consequence of such biased data sets is that one can get an apparently good predictive model which classifies green landscapes as cows and beige ones as camels. If only the causal properties of the categories could be identified, a classifier relying on these would not only be a good classifier, but it would also be ready to identify the membership of test examples in any environment, even ones that have never been encountered before, in contrast to the heuristically methods developed for covariate shift. This is what the Invariant Risk Minimization principle aims at achieving [11]. Without entering the details, the principle translates into an optimization criterion that tends to isolate the causal predictors from multiple training environments. It aims at finding a representation of the features, such that the predictor is *simultaneously optimal in all environments* when using this representation.

Although it has been remarked [12] that the optimization problem is highly non convex in a space of unknown latent variables and thus difficult to solve and furthermore that there are simple conditions under which the method fails to recover the optimal invariant predictor, the idea is still very appealing and attracts a lot of follow-up work [13].

On the one hand, the perspective here resembles that of the search for a generic representation useful for many different tasks. On the other hand, it distinguishes itself by the search for *a representation that is just good enough* to enable good performance in all the possible environments.

4. Transfer and the idea of maximal bias

... or how we come back to the statistical theory of learning, this time enhancing the role of the inductive bias

Transfer learning is needed when the target training data is too scarce to enable reliable learning. Under this condition, the risk is high to obtain a solution, a classifier, which overfits the training data and has a poor generalization performance. Following the recipe from the statistical analysis of supervised induction, one approach to this problem is to limit as much as possible the capacity of the hypothesis space in the target domain (see Section 2).

A special kind of transfer learning illustrates this. It is called *Learning Using Privileged Information* (LUPI) and was introduced by V. Vapnik and A. Vashitst in 2009 [14]. The idea is inspired by learning in a teacher-student setting. In this scenario, it is often the case that, at training time, the teacher provides each example $\mathbf{x}_i$ from a description space $\mathcal{X}$ together with some supplementary (or privileged) information $\mathbf{x}_i^\star$ from a space $\mathcal{X}^\star$ in addition to the associated label $y_i \in \mathcal{Y}$ yieding triplets $(\mathbf{x}_i, \mathbf{x}_i^\star, y_i)$. The catch is that, at test time, the student will have to make predictions using only the description $\mathbf{x} \in \mathcal{X}$ without having access to the privileged information anymore.

Such a scenario could occur when a student in medicine is learning to interpret tomographies and a teacher provides comments on the images under inspections, while at testing time, the future physician will only have access to the images to make a prognostic.

The question is then: is it possible, and how, to use the privileged information available at training time in order to build a better predictor $h : \mathcal{X} \to \mathcal{Y}$?

In [15], V. Vapnik and R. Izmailov propose two answers, both of them based on the statistical analysis of learning. We focus here on the second one, the more closely related to transfer learning.

In this solution, it is supposed that the teacher knows a simple decision rule in the privileged (i.e. source) space $\mathcal{X}^\star$. The VC dimension of the hypothesis space in $\mathcal{X}^\star$ can then be much smaller than the one in the input (i.e. target) space $\mathcal{X}$. The problem is consequently to be able to translate this rule which applies in $\mathcal{X}^\star$ into a rule usable in $\mathcal{X}$. The authors describe a convoluted way of doing so that does not concern us here. But there is an important message here. The LUPI scenario can be seen as a kind of transfer learning where the source task, training with privileged information, hence in a different input space to that of the target task, is used to *find a bias that reduces the capacity of the hypothesis space for the target task*.

The fundamental problem of transfer learning can be thus seen as the search for an appropriate bias in oder to learn using the (very) limited target training data.

5. The TransBoost algorithm and its implications

In the works cited previously, transfer learning methods suppose that the input space is the same for the source domain and the target one, the LUPI scenario being an exception. This allows the estimation of a distance or similarity between the empirical source and target distributions using classical tools from statistics (e.g. the Kullback-Leibler divergence). Furthermore, looking for a common representation space where the source and target descriptions of the examples can no longer be distinguished is more amenable.

If the source and target input spaces are distinct: $\mathcal{X}_\mathcal{S} \neq \mathcal{X}_\mathcal{T}$, what kind of distance could be used which would tell something about the difficulty of the transfer and in which way it could enter theoretical guarantees about the performance in the target domain?

In what follows, we present an algorithm for transfer learning, first presented in [16], that (*i*) allows transfer between domains that do not share the same input space (but assumes that $\mathcal{Y}_\mathcal{S} = \mathcal{Y}_\mathcal{T}$)[3], (*ii*) provides a way to

[3] Actually, the requirement is not that $\mathcal{Y}_\mathcal{S}$ is exactly equal to $\mathcal{Y}_\mathcal{T}$, but that they have the same number of labels. For instance, that both the source and target tasks are binary classification.

obtain formal guarantees on the performance in the target domain, while showing (*iii*), perhaps surprisingly, that the performance of the source hypothesis need not be good on the source domain to obtain high performance on the target domain, thus offering a fresh perspective for the design of new transfer learning algorithms.

5.1. A new perspective on transfer learning

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 $f \in F$: $\mathcal{X} \to \mathcal{Z}$. Let G be a class of decision functions that use descriptions of the examples in the feature space: $g \in G : \mathcal{Z} \to \mathcal{Y}$.

Then, in the classical context of deep neural networks, the hypothesis class for the source domain is $\mathcal{H}_\mathcal{S} := \{h_\mathcal{S} : \exists f \in F, g \in G \text{ st. } h_\mathcal{S} = g \circ f\}$. Retaining the idea of keeping the same representation in the source domain and the target one (see Section 3), f would be kept (at least approximately) during transfer, and only g would have to be learned to solve the target problem, yielding $\mathcal{H}_\mathcal{T} := \{h_\mathcal{T} : g' \in G \text{ st. } h_\mathcal{T} = g' \circ f\}$ where g' is adapted to fit the target training data.

But we could as well adopt a *dual perspective* where the decision function g is kept fixed ($g \in G : \mathcal{Z} \to \mathcal{Y}$.) and it is the representation function f which is transformed between the source and the target ($f_\mathcal{S} \in F_\mathcal{S}$: $\mathcal{X}_\mathcal{S} \to \mathcal{Z}$ and $f_\mathcal{T} \in F_\mathcal{T}$: $\mathcal{X}_\mathcal{T} \to \mathcal{Z}$).

Then we would have as before: $\mathcal{H}_\mathcal{S} := \{h_\mathcal{S} : \exists f_\mathcal{S} \in F_\mathcal{S}, g \in G \text{ st. } h_\mathcal{S} = g \circ f_\mathcal{S}\}$ and $\mathcal{H}_\mathcal{T} := \{h_\mathcal{T} : f_\mathcal{T} \in F_\mathcal{T} \text{ st. } h_\mathcal{T} = g \circ f_\mathcal{T}\}$, where now, the problem is to translate $f_\mathcal{S}$ into $f_\mathcal{T}$, or rather the inverse: $f_\mathcal{T}$ into $f_\mathcal{S}$ in order to be able to use g. Intuitively this would correspond to a good decision maker (here the function g) that would remain good in a new domain provided that the representation of the examples in the new domain are amenable to its decision process.

In the `TransBoost` approach, the adaptation goes one step further. The source hypothesis $h_\mathcal{S}$ where $h_\mathcal{S} = g \circ f_\mathcal{S}$ is used as such in the target domain, given that a right translation function from the target input space to the source input space, $\pi : \mathcal{X}^\mathcal{T} \to \mathcal{X}^\mathcal{S}$ has been learned. Therefore, in this perspective, the target hypothesis space becomes $\mathcal{H}_\mathcal{T} := \{h_\mathcal{T} : \exists \pi \in \Pi, \text{ st. } h_\mathcal{T} = h_\mathcal{S} \circ \pi\}$ (see Figure 3).

Remark: one could note that $h_\mathcal{T} = g \circ f_\mathcal{S} \circ f_\mathcal{S}^{-1} \circ f_\mathcal{T} = h_\mathcal{S} \circ f_\mathcal{S}^{-1} \circ f_\mathcal{T}$. But we find it more easy to learn directly projections $\pi : \mathcal{X}_\mathcal{T} \to \mathcal{X}_\mathcal{S}$, thus considering $h_\mathcal{T} = h_\mathcal{S} \circ \pi$, since we do not have $f_\mathcal{T}$.

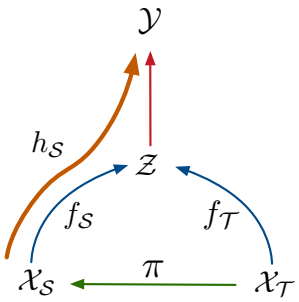


Figure 3: How to use the source hypothesis $h_\mathcal{S}$ in order to compute the target hypothesis $h_\mathcal{T}$ from $\mathcal{X}_\mathcal{T}$ to $\mathcal{Y}$: $h_\mathcal{T} = h_\mathcal{S} \circ \pi$.

Intuitively again, 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 PET scan image into a satellite image, we could use the recognition function defined on satellite image to decide if there are cancerous cells in the biopsy (see Figure 4).

Given a target query of the form: "*what is the label of* $\mathbf{x}_\mathcal{T} \in \mathcal{X}^\mathcal{T}$?", thanks to the translation π learned, we would translate it into a source query "*what is the label of* $\pi(\mathbf{x}_\mathcal{T}) \in \mathcal{X}^\mathcal{S}$?", and the source hypothesis $h_\mathcal{S}$ applied to $\pi(\mathbf{x}_\mathcal{T}) \in \mathcal{X}_\mathcal{S}$, would then provide the answer.

The goal of transfer learning would then be to learn a good translation $\pi : \mathcal{X}_\mathcal{T} \to \mathcal{X}_\mathcal{S}$.

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 (Fig. 5).

Definition 5.1. A **weak projection** w.r.t. source decision function $h_\mathcal{S}$ is a function $\pi : \mathcal{X}_\mathcal{T} \to \mathcal{X}_\mathcal{S}$ such that the decision function $h_\mathcal{S}(\pi(\mathbf{x}_\mathcal{T}))$ has *better than random classification performance* on the target training set $S_\mathcal{T}$.

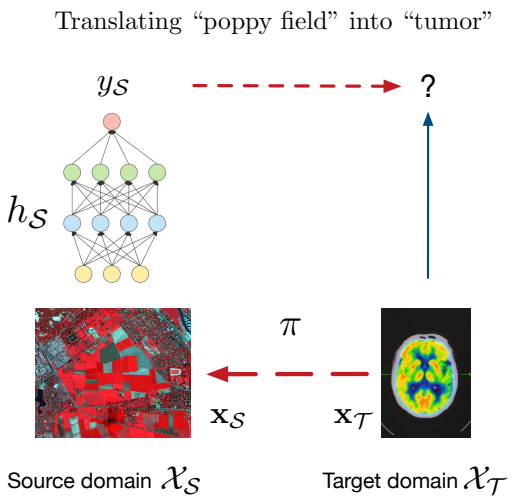


Figure 4: In order to classify a PET scan image (target task), a classifier for satellite images (source task) can be used if one knows how to translate appropriately PET scan images to satellite images.

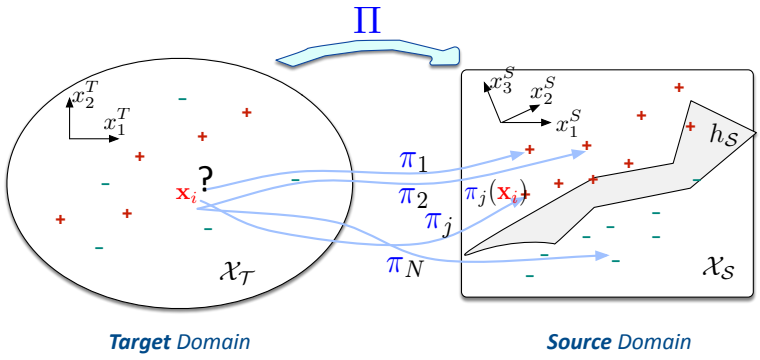


Figure 5: The principle of prediction using `TransBoost`. A given target example $\mathbf{x}_i^{\mathcal{T}}$ is projected in the source domain using a set of identified weak projections π_j and the prediction for $\mathbf{x}_i^{\mathcal{T}}$ is computed as: $H_{\mathcal{T}}(\mathbf{x}_i^{\mathcal{T}}) = \texttt{sign}\Big\{\sum_{j=1}^{N} \alpha_j h_{\mathcal{S}}(\pi_j(\mathbf{x}_i^{\mathcal{T}}))\Big\}$.

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

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 `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.

Roughly, the idea is to use an iterative strategy where, at each step n:

1. the algorithm searches the space Π of weak projections until one ($\pi^n_{\mathcal{T}\to\mathcal{S}}$, noted π_n from now on) is found that leads to better than random performance for the hypothesis $h_{\mathcal{S}} \circ \pi_n$ on the target training sample $\mathcal{S}_{\mathcal{T}}$,
2. the algorithm modifies the weights of the training examples in $\mathcal{T}$ such that examples that were correctly labelled by $h_{\mathcal{S}} \circ \pi_n$ have their weight reduced, while the other ones have their weight increased

When the termination criterion is met, at step N, the algorithm returns the decision function defined on the target domain of the form:

$$\mathcal{H}_{\mathcal{T}} = \texttt{sign}\Big[\sum_{n=1}^{N} \alpha_n \, h_{\mathcal{S}} \circ \pi_n\Big]$$

Algorithm 1: Transfer learning by boosting

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

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

for $n = 1, \ldots, N$ **do**
Find a projection $\pi_i : \mathcal{X}_{\mathcal{T}} \to \mathcal{X}_{\mathcal{S}}$ st. $h_{\mathcal{S}}(\pi_i(\cdot))$ performs better than random on $D_n(\mathcal{S}_{\mathcal{T}})$;
Let ε_n be the error rate of $h_{\mathcal{S}}(\pi_i(\cdot))$ on $D_n(\mathcal{S}_{\mathcal{T}})$: $\varepsilon_n \doteq \mathbf{P}_{i \sim D_n}[h_{\mathcal{S}}(\pi_n(\mathbf{x}_i)) \neq y_i]$ (with $\varepsilon_n < 0.5$) ;
Computes $\alpha_i = \frac{1}{2} \log_2(\frac{1-\varepsilon_i}{\varepsilon_i})$;
Update, for $i = 1 \ldots, m$:

$$\begin{aligned} D_{n+1}(i) &= \frac{D_n(i)}{Z_n} \times \begin{cases} e^{-\alpha_n} & \text{if } h_{\mathcal{S}}(\pi_n(\mathbf{x}_{\mathcal{T}}^i)) = y_{\mathcal{T}}^i \\ e^{\alpha_n} & \text{if } h_{\mathcal{S}}(\pi_n(\mathbf{x}_{\mathcal{T}}^i)) \neq y_{\mathcal{T}}^i \end{cases} \\ &= \frac{D_n(i)\, \exp(-\alpha_n\, y_{\mathcal{T}}^i\, h_{\mathcal{S}}(\pi_n(\mathbf{x}_{\mathcal{T}}^i)))}{Z_n} \end{aligned}$$

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

Output: the final target hypothesis $H_{\mathcal{T}} : \mathcal{X}_{\mathcal{T}} \to \mathcal{Y}_{\mathcal{T}}$:

$$H_{\mathcal{T}}(\mathbf{x}_{\mathcal{T}}) = \text{sign}\Big\{\sum_{n=1}^{N} \alpha_n\, h_{\mathcal{S}}(\pi_n(\mathbf{x}_{\mathcal{T}}))\Big\} \tag{2}$$

5.2. Theoretical analysis

Two questions arise about the method presented:

1. Can we get any *guarantees* about the performance of the learned decision function $H_{\mathcal{T}}$ in the target space using `TransBoost`?
2. What are the *ingredients* for a successful transfer learning? In other words, what properties, if any, should satisfy the source domain (i.e. the source hypothesis) and the projections with respect to the target learning problem?

I- Let us start with the first question. Inspired by the theoretical framework of boosting [17], we define the *weak transfer property*.

Definition 5.2. A set Π of projection functions $\pi : \mathcal{X}_{\mathcal{T}} \to \mathcal{X}_{\mathcal{S}}$ fulfills the **weak transfer property** w.r.t. source decision function $h_{\mathcal{S}}$ for some transfer learning problem from a source domain $\mathcal{D}_{\mathcal{S}}$ to a target domain $\mathcal{D}_{\mathcal{T}}$ if: for any marginal distribution $\mathbf{p}_{\mathcal{X}}^{\mathcal{T}}$ over $\mathcal{X}_{\mathcal{T}}$, any concept $c : \mathcal{X}_{\mathcal{T}} \to \{-1, +1\}$, any $\gamma > 0$, any $\delta > 0$, there exists a learning algorithm that takes as input $m(\delta)$ examples $\{(\mathbf{x}_{\mathcal{T}}^1, c(\mathbf{x}_{\mathcal{T}}^1)), \ldots, (\mathbf{x}_{\mathcal{T}}^m, c(\mathbf{x}_{\mathcal{T}}^m))\}$ where the $\mathbf{x}_{\mathcal{T}}^i$ are chosen independently according to any distribution over $\mathcal{X}_{\mathcal{T}}$ and can pickup a projection $\pi \in \Pi$ such that:

$$\mathbf{P}_{\mathcal{X}}^{\mathcal{T}}\Big[err(h_{\mathcal{S}}(\pi(\cdot))) > \frac{1}{2} - \gamma\Big] \leq \delta$$

The weak transfer property is modeled on the weak learnability definition, setting apart the projection π in the composed hypothesis $h_{\mathcal{S}}(\pi(\cdot))$.

A weaker demand is that for any distribution $\mathbf{P}_{\mathcal{X}}^{\mathcal{T}}$ over the training set in the target domain, there exists a weak projection $\pi \in \Pi$ such that

$$\widehat{err}_{S_{\mathcal{T}}}[h_{\mathcal{S}}(\pi(\cdot))] = \mathbf{Pr}_{i \sim D}[h_{\mathcal{S}}(\pi(\mathbf{x}_{\mathcal{T}}^i)) \neq y_i] < \frac{1}{2}$$

where $\widehat{err}_{S_\mathcal{T}}$ is the empirical error on the training set in the target domain weighted by a probability distribution D. This is called the *empirical weak transfer assumption*.

As soon as we have a weak projection π, wrt. $h_\mathcal{S}$ and a distribution $\mathbf{P}^\mathcal{T}_\mathcal{X}$ over the training set $S_\mathcal{T}$, we have a weak hypothesis $h_\mathcal{S}(\pi(\cdot))$ over the target domain in the traditional sense of the boosting framework. And thus, we inherit the properties of the AdaBoost algorithm: (*i*) *bound on the training error* that drops exponentially fast as a function of the number of the weak projections used, and (ii) *bounds on the generalization error*. The latter can be obtained through a VC-dimension analysis yielding for the hypothesis set:

$$\mathcal{H}_\mathcal{T} \;=\; \left\{ \mathtt{sign}\left[\sum_{n=1}^{N} \alpha_n \, h_\mathcal{S} \circ \pi_n \right] | \alpha_n \in \mathbb{R}, \pi_n \in \Pi, n \in [1, N] \right\}$$

the VC-dimension: $d_{\mathtt{VC}}(\mathcal{H}_\mathcal{T}) = d_{\mathtt{VC}}(h_\mathcal{S} \circ \Pi) \;\leq\; 2(d+1)(N+1)\log_2((N+1)\,e)$ with d the VC-dimension of the base classifiers $h_\mathcal{S} \circ \pi$ and e the mathematical constant (≈ 2.718). See for instance [18], p.131.

So, as soon as the source hypothesis $h_\mathcal{S}$ is given, the set Π of (weak) projections induces a set of (weak) classifiers $h_\mathcal{T} = h_\mathcal{S} \circ \pi$ (with $\pi \in \Pi$) over the target domain. Thus the weak transfer property provides the same guarantees on the learning performance of `TransBoost` as the ones of AdaBoost.

Indeed, controlling the versatility of the set Π of projections induces a control over the capacity of the set of target hypotheses $h_\mathcal{S} \circ \Pi$. And low capacity is more easy to obtain for weak projections.

II- Now, let us turn to the second question: *what are the ingredients for a successful transfer?*

Note that nowhere the performance of the source hypothesis $h_\mathcal{S}$ enters the analysis. What ensures the performance of the resulting target hypothesis $h_\mathcal{T}$ is the weak transfer property and the low capacity of $h_\mathcal{T}$ (or large margin property induced by) the set Π of projections. So that transfer does not depend on having a good source hypothesis, but on having a good, and limited, set of projections Π (e.g. as measured by the VC dimension of $h_\mathcal{S} \circ \Pi$).

5.2.1. Experimental Setup

Two dimensions are expected to govern the efficiency of transfer learning:

1. The *level of signal* in the target data. This is controlled by the number $m_\mathcal{T}$ of training data in the target domain. The larger this number, the less useful is transfer learning. It is also a function of the information in each example that can lead to a good prediction.
2. The *relatedness* between the source and the target domains. In the first experiments reported below, we control relatedness through the information shared between source and target (i.e. the number of shared measurements).

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 univariate time series of length $t_\mathcal{T}$ into two classes, therefore finding a classifier $h_\mathcal{T} : \mathbb{R}^{t_\mathcal{T}} \to \{-1, +1\}$. By controlling the level of noise and the difference between the distributions governing the two classes of time series: -1 and $+1$, we can control the signal level, that is the difficulty of extracting information from the target training data. We also control the amount of information by varying the length $t_\mathcal{T}$ (number of measurements for each example: here between 20 and 100).

Likewise, the **source input space** is a space of univariate sequences of real measurements of length $t_\mathcal{S}$ (here 200). In this space also, there are two classes of time series. Therefore, we have $h_\mathcal{S} : \mathbb{R}^{ts} \to \{-1, +1\}$.

In the first experiment, the source and target share the same classes of time series. Only, the source time series are known up to $t_\mathcal{S}$ measurements, while the target ones are known up to $t_\mathcal{T}$ measurements with $t_\mathcal{T} < t_\mathcal{S}$ (See Figure 6).

Varying $|t_\mathcal{S} - t_\mathcal{T}|$ is a way of controlling the information potentially shared in the two domains. The smaller the difference $t_\mathcal{S} - t_\mathcal{T}$, the larger the similarity. 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. Many applications involve to classify time series of length different from the length for which exists a classifier. [19, 20].

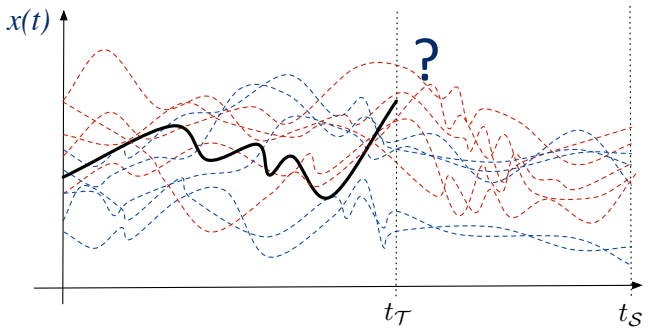


Figure 6: The source domain is composed of time series of length $t_{\mathcal{S}}$ for which the labels are known (here blue and red). The target domain is made of time series of length $t_{\mathcal{T}}$ with $t_{\mathcal{T}} \leq t_{\mathcal{S}}$ for which the label must be estimated.

5.2.2. *Description of the Experiments*

Time series were generated according to the following equation where the sign of the slope determines the class of the series : '+' if the sign is positive, '−' otherwise:

$$\mathbf{x}_t = \underbrace{t \times \text{slope} \times \text{class}}_{\text{information gain}} + \underbrace{\text{x}_{max}\ \sin(\omega_t \times t + \varphi_t)}_{\text{sinusoid}} + \underbrace{\eta(t)}_{\text{noise factor}} \tag{3}$$

In each class, the examples are drawn with varying values for the period ω_t of the signal (a sinusoid) and its phase φ_t, in addition to a noise factor, generated according to a Gaussian distribution. x_{max} weights the importance of the sinusoid.

To sum up, 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 measurements. In the experiments, the slope varied from almost non existent: 0.001 to significant: 0.01.
2. the *noise factor* $\eta(t)$. In our experiments, the noise factor is generated according to a Gaussian distribution of mean = 0 and standard deviation in $\{0.001, 0.002, 0.02, 0.2, 1\}$
3. the *length* of the time series, that is the number of measurements
4. the *size* $m_{\mathcal{T}}$ of the target training set

Figure 7 illustrates what can be obtained with 2 examples (with two sinusoids of different periods and phases) in the +1 class (slope = +0.08) , and similarly 2 examples in the −1 class (slope = −0.08).

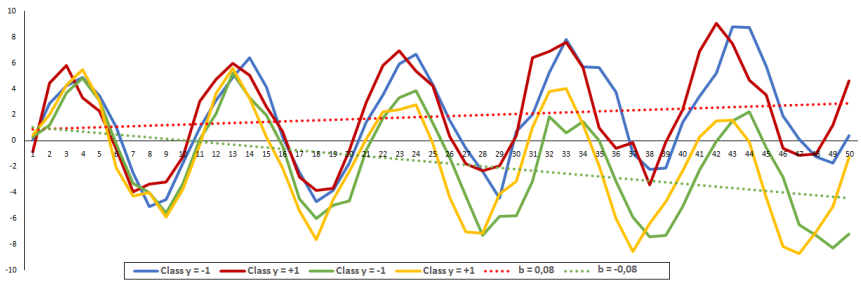


Figure 7: A synthetic data set $\mathcal{S}$ with 4 times series, two of the $y = +1$ with associated slope = +0.08, and two of the class $y = -1$, with associated slope = −0.08. The noise level is Gaussian with ($\mu = 0, \sigma = 0.2$).

In the experiments reported here, we kept the size of the **source** training set constant. In each experiment, 900 times series of length $t_{\mathcal{S}} = 200$ 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. The source hypothesis, Gaussian SVM as implemented in Scikit Learn, was learned using these time series [21].

The **target** training data set was obtained using the same generation process as for the source data (using equation 3), but with the length $t_{\mathcal{T}}$ varying in $\{20, 50, 70, 100\}$, thus providing increasing levels of signal since the target examples shared more and more features (time measurements) with the source ones.

A *target training data set* of 300 time series was drawn equally balanced between the two classes (150 '+', and 150 '−'). Note that this relatively small number corresponds to transfer learning scenarios where the training data is limited in the target domain. A remaining 600 time series not used for learning were employed as a *test set*.

Projecting the target examples into the source input space

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 (see Figure 8). 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.

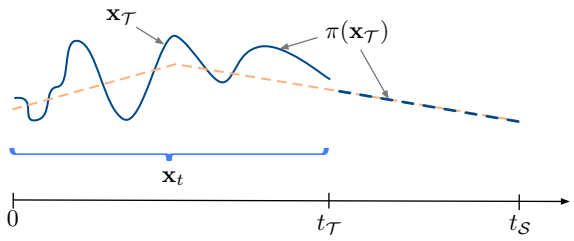


Figure 8: Example of a projection π (a hinge function with three parameters: the first slope, the second one and the time of the hinge) that is adjusted to the target exemple $\mathrm{x}_{\mathcal{T}}$ by least square. The resulting projection $\pi(\mathrm{x}_{\mathcal{T}})$ is the concatenation of $\mathrm{x}_{\mathcal{T}}$ and the remaining part of the adjusted hinge function.

In order to 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 (column 'SVM (test)' in Table 1),
2. to a boosting algorithm operating in the target domain with base classifiers being Gaussian SVMs. In that way, it was possible to check if this was the boosting algorithm that was responsible for the level of performance of `TransBoost`, with no use for transfer learning, or not. (For the results, see the remark below).
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 $h_{\mathcal{S}}(\text{regression}(\mathbf{x}_{\mathcal{T}}))$ (column 'SVR+SVM (test)' in Table 1).

The number of boosting steps for `TransBoost` was varied between 5, 10 and 20, with a slight increase of performance when augmenting the number of steps. In the experiments reported, the number of steps is $N = 20$.

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.

The *first group* corresponds to target time series of length $t_{\mathcal{T}} = 20$ which amounts to little signal. This is testimoned by the fact that both classifiers learned directly on these times series (column SVM (test)) and classifiers learned over time series completed by a SVR, reach levels of performance barely above random guessing, while `TransBoost` is remarkably efficient (with error rates below 0.1), at least when the noise level is below 0.200.

This is almost the same configuration for the *second group* with target time series of length $t_{\mathcal{T}} = 50$. Again `TransBoost` has very low testing error, while the two other approaches struggle. Of course, when the noise level is high (= 0.200) with a small slope (= 0.001), there is no signal to build upon and `TransBoost` does not fare better than other approaches.

slope, noise, $t_{\mathcal{T}}$	SVM (test)	$H_{\mathcal{T}}$ (train)	$H_{\mathcal{T}}$ (test)	SVR+SVM (test)
0.001, 0.001, **20**	0.50 ± 0.08	0.08 ± 0.03	**0.08** ± 0.02	0.49 ± 0.01
0.005, 0.001, **20**	0.49 ± 0.01	0.01 ± 0.01	**0.01** ± 0.01	0.45 ± 0.01
0.005, 0.002, **20**	0.49 ± 0.03	0.03 ± 0.02	**0.04** ± 0.02	0.43 ± 0.01
0.005, 0.020, **20**	0.48 ± 0.03	0.09 ± 0.01	**0.10** ± 0.01	0.47 ± 0.01
0.001, 0.200, **20**	0.50 ± 0.01	0.46 ± 0.02	0.51 ± 0.02	0.49 ± 0.01
0.010, 0.200, **20**	0.47 ± 0.03	0.34 ± 0.02	0.35 ± 0.02	0.35 ± 0.01
0.001, 0.001, **50**	0.50 ± 0.01	0.08 ± 0.03	**0.08** ± 0.02	0.41 ± 0.01
0.005, 0.001, **50**	0.28 ± 0.09	0.01 ± 0.01	**0.01** ± 0.01	0.28 ± 0.01
0.005, 0.002, **50**	0.30 ± 0.08	0.02 ± 0.01	**0.02** ± 0.01	0.28 ± 0.01
0.005, 0.020, **50**	0.30 ± 0.08	0.04 ± 0.01	**0.04** ± 0.01	0.31 ± 0.01
0.001, 0.200, **50**	0.50 ± 0.01	0.38 ± 0.03	0.44 ± 0.02	0.43 ± 0.01
0.010, 0.200, **50**	0.12 ± 0.04	0.10 ± 0.02	0.11 ± 0.02	0.15 ± 0.02
0.001, 0.001, **100**	0.47 ± 0.03	0.07 ± 0.02	**0.07** ± 0.02	0.23 ± 0.01
0.005, 0.001, **100**	0.07 ± 0.03	0.01 ± 0.01	**0.01** ± 0.01	0.07 ± 0.02
0.005, 0.002, **100**	0.10 ± 0.04	0.02 ± 0.01	**0.02** ± 0.01	0.07 ± 0.01
0.005, 0.020, **100**	0.09 ± 0.03	0.02 ± 0.01	**0.03** ± 0.01	0.07 ± 0.01
0.001, 0.200, **100**	0.46 ± 0.02	0.28 ± 0.02	0.31 ± 0.01	0.31 ± 0.01
0.010, 0.200, **100**	0.05 ± 0.02	0.04 ± 0.01	0.05 ± 0.01	0.05 ± 0.01

Table 1: Comparison of the error rate (lower is better) between: learning directly in the target domain, here using a Gaussian SVM (columns $h_{\mathcal{T}}$ (test)), using `TransBoost` (columns $H_{\mathcal{T}}$ (train) and $H_{\mathcal{T}}$ (test)) and, finally, mapping the time series with a SVR regression in order to project the target time series (of length $t_{\mathcal{T}}$) into the source domain (length $t_{\mathcal{T}}$) and using $h_{\mathcal{S}}$ a Gaussian SVM; called naïve transfer (column SVR (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.

Finally, in the *third group* with time series of length $t_{\mathcal{T}} = 100$, the signal in the target times series is enough for all approaches, but still with a significant advantage for `TransBoost`.

Inrerestingly, `TransBoost` does not exhibit overfitting as the columns '$H_{\mathcal{T}}$ (train)' and '$H_{\mathcal{T}}$ (test)' show.

Remark: 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.

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 9 and 10 that sum up a larger set of experimental 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 algorithms: either the hypothesis $h_{\mathcal{T}}$ learnt on the target training data alone (Figure 9), or the hypothesis $H'^{\mathcal{T}}$ learned on the target data projected on the source input space using a SVR regression (Figure 10). The remarquable 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 9 and 10 (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).

5.2.3. Additional Experiments

We show here, in Figures 11, 12 and 13 qualitative results obtained on the classical half-moon problem. It is apparent that `Transboost` brings satisfying results.

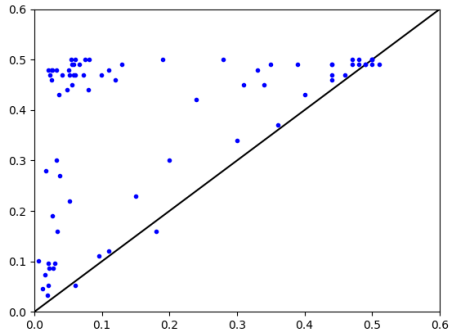


Figure 9: Comparison of error rates. y-axis: test error of the SVM classifier (without transfer). x-axis : test error of the TransBoost classifier with 10 boosting steps. The results of 75 experiments (each one repeated 100 times) are summed up in this graph.

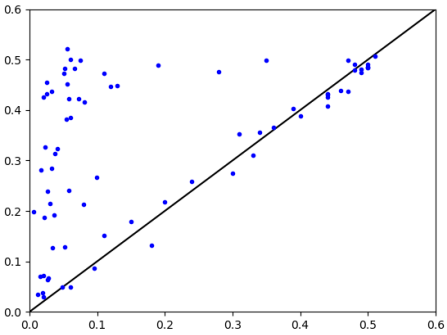
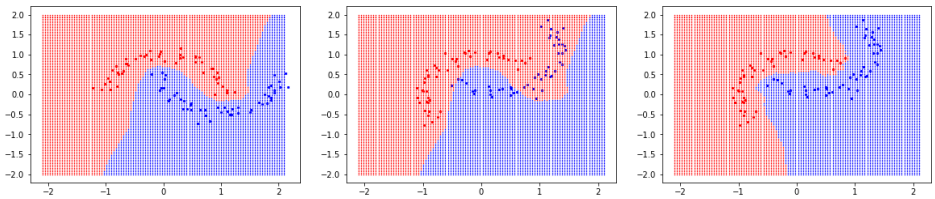


Figure 10: Comparison of error rates. y-axis: test error of the "nave" transfer method. x-axis : test error of the TransBoost classifier with 10 boosting steps. The results of 75 experiments (each one repeated 100 times) are summed up in this graph.

More generally, as long as projections from $\mathcal{X}_\mathcal{T}$ to $\mathcal{X}_\mathcal{S}$ can be defined, `TransBoost` can be applied. One very simple technique is to use the space of random matrices from $\mathcal{X}_\mathcal{T}$ to $\mathcal{X}_\mathcal{S}$ and retain the ones with the weak transfer property.



(a) kNN source model trained on the data source : it fits to the data source

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

(c) kNN source model trained on the data source transBoosted to the data target

Figure 11: The model trained on the source data (a) is not good on the target data (b), while the same source model using `TransBoost` fits the target data (c).

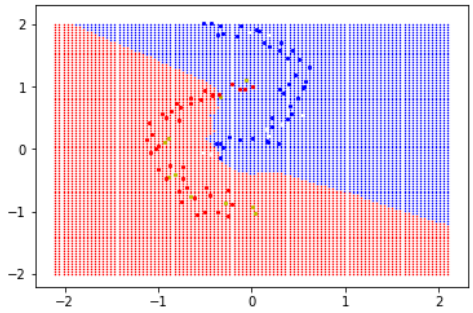


Figure 12: A KNN model trained on the few target data points (in yellow).

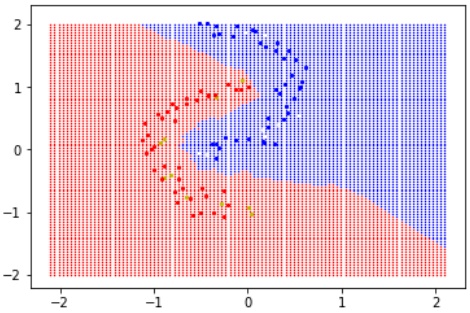


Figure 13: A KNN model transboosted on the few target data points.

5.3. Experiments with a random source hypothesis

The theoretical analysis described in Section 5.2 concluded that the performance of the source hypothesis on the source training data was irrelevant. But can this be really so?

slope, noise, $t_{\mathcal{T}}$	$h_{\mathcal{T}}$ (train)	$h_{\mathcal{T}}$ (test)	$H_{\mathcal{T}}$ (train)	$H_{\mathcal{T}}$ (test)
0.001, 0.200, 70	0.42 ± 0.03	0.48 ± 0.02	0.33 ± 0.02	**0.37** ± 0.02
0.001, 0.001, 70	0.44 ± 0.02	0.48 ± 0.02	0.06 ± 0.02	**0.06** ± 0.02
0.005, 0.020, 70	0.11 ± 0.04	0.12 ± 0.05	0.04 ± 0.02	**0.03** ± 0.01
0.005, 0.002, 70	0.10 ± 0.04	0.11 ± 0.05	0.01 ± 0.01	**0.01** ± 0.01
0.005, 0.001, 70	0.11 ± 0.04	0.13 ± 0.05	0.02 ± 0.01	**0.02** ± 0.02
0.010, 0.200, 70	0.06 ± 0.03	0.08 ± 0.03	0.08 ± 0.02	0.08 ± 0.02

Table 2: Learning without transfer directly on the target data (columns '$h_{\mathcal{T}}$ (train)' and '$h_{\mathcal{T}}$ (test)'), and with transfer using an apriori irrelevant source hypothesis (columns '$H_{\mathcal{T}}$ (train)' and '$H_{\mathcal{T}}$ (test)'). The difficulty of the task decreases from the top row to the bottom one.

In a *second set of experiments*, the source hypothesis was **generated randomly** according to the second protocol described above.

Table 2 shows a representative set of results for the case where the target domain is $\mathbb{R}^{70}$ while the source domain is $\mathbb{R}^{40}$. Again, `TransBoost` brings remarkable gains wrt. learning without transfer, except when the learning task is easy using directly the target data (e.g. slope = 0.01), or when the slope governing the target training data is so low (e.g. 0.001) that there is almost no remaining signal in the target training data. (Note that there is no error rate given for the source hypothesis since it was not learnt using a data set).

Of course, such a good performance is not attained with any source hypothesis $h_{\mathcal{S}}$. The source hypothesis plays indeed the role of a bias. Just as an expert may or may not provide a useful bias, the source hypothesis may or may not prove useful.

6. Conclusions

The common view held nowadays about transfer learning is highly influenced by the numerous works using deep neural networks. It posits that, in order to adapt a neural network from a learnt task to a new one, one should keep the first layers, those that extract the useful building blocks for representation, and adapts or fine-tunes the last layers [22, 23, 24][4].

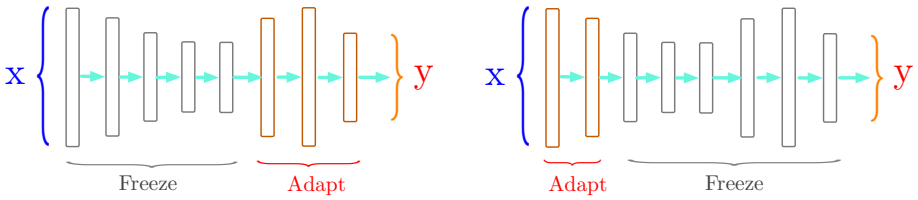


Figure 14: Transfer learning in neural networks. (Left) The current dominant perspective, where the first layers are freezed while the last ones are adapted. (Right) An alternative viewpoint, in the spirit of the approach described in this paper: adapt the first, perception, layers while keeping the last, decision, layers.

In this paper, we have presented an alternative approach. Keep the decision function, i.e. the last layers, and adapt the first ones to modify the representation so that that the same decision function can be applied to the new domain (see Figure 14). This second point of view is much more in line with the perspective of symbolic AI. In it, the same rules of logic that govern one domain are deemed to be applicable to a large spectrum of other domains. Once the relevant entities have been identified, the same rules can be applied with confidence. In fact, this view is also shared in studies about analogical reasoning, a kind of extreme transfer: with one example for the source domain and one

[4] [25] is a notable exception promoting to add layers before the first layers of the source neural network.

query about the target one. For instance, in the following analogical problem [26]:

```
  a b c    →  a b d
i j j k k k  →  ?
```

the preferred solution is the one that looks for successive elements in the input part: '`a b c`' or '`i j j k k k`', so that the general rule '`replace the last element by its successor`' can be applied. Then '`c`' would be replaced by its successor '`d`' in the source domain, and '`k k k`' identified as the last element of '`i j j k k k`' would be replaced by its successor '`l l l l`' in the target one.

Depending on which perspective is adopted, the notion of difference or distance between the source and target domains is seen differently. Whereas within the current deep learning view, transfer learning emphasizes finding a common representation of the source and target training sets thus insisting on devising measures of distance between distributions of examples, the alternative view put forward in this paper is that what matters is being able *to translate questions* in the target domain into questions that can be answered by the available source hypothesis. The consequence is that any source hypothesis can be relevant if one is able to identify suitable translations (here a weighted combination of weak translators) from target domain entries to source domains entries, thus enabling to use the source hypothesis.

Speaking in terms of the statistical theory of learning, this is similar to choosing of a good regularization term. Here, the source hypothesis forces the target hypothesis space to be of the form $h_S \circ \pi$ with $\pi : \mathcal{X}^{\mathcal{T}} \to \mathcal{X}^{\mathcal{S}}$. If the source hypothesis (regularizer) is ill-chosen, then the learning task is made difficult, if not impossible. In fact, negative transfer can occur when, in order to find a set of projections with the weak transfer property, one has to resort to a set with high capacity, resulting in over-fitting.

If this paper provocatively shows that an apparently irrelevant source hypothesis can be successfully used for transfer to a target domain, it remains that we would like to be able to know beforehand what source hypotheses might provide a useful bias. In inductive learning, one expects the bias, then translated into regularizers, to come from experts. For example, favoring sparse models because there are reasons to believe that what governs this phenomenon under study does not involve numerous factors. We hope this paper stimulates thoughts about how to identify good source domains and/or good source hypotheses for a given target problem.

[1] V. Vapnik, The nature of statistical learning theory, Springer, 1995.

[2] O. Chapelle, B. Scholkopf, A. Zien, Semi-supervised learning. 2006, Cambridge, Massachusettes: The MIT Press View Article 2.

[3] R. S. Sutton, A. Koop, D. Silver, On the role of tracking in stationary environments, in: Proceedings of the 24th international conference on Machine learning, 2007, pp. 871–878.

[4] L. G. Valiant, A theory of the learnable, Communications of the ACM 27 (11) (1984) 1134–1142.

[5] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, Analysis of representations for domain adaptation, Advances in neural information processing systems 19.

[6] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J. Wortman, Learning bounds for domain adaptation, Advances in neural information processing systems 20.

[7] I. Redko, A. Habrard, M. Sebban, Theoretical analysis of domain adaptation with optimal transport, in: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part II 10, Springer, 2017, pp. 737–753.

[8] N. Cesa-Bianchi, G. Lugosi, Prediction, learning, and games, Cambridge university press, 2006.

[9] H. Zhao, R. T. Des Combes, K. Zhang, G. Gordon, On learning invariant representations for domain adaptation, in: International conference on machine learning, PMLR, 2019, pp. 7523–7532.

[10] Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation, in: International conference on machine learning, PMLR, 2015, pp. 1180–1189.

[11] M. Arjovsky, L. Bottou, I. Gulrajani, D. Lopez-Paz, Invariant risk minimization, arXiv preprint arXiv:1907.02893.

[12] E. Rosenfeld, P. Ravikumar, A. Risteski, The risks of invariant risk minimization, arXiv preprint arXiv:2010.05761.

[13] J. Zhang, L. Bottou, Learning useful representations for shifting tasks and distributions, in: International Conference on Machine Learning, PMLR, 2023, pp. 40830–40850.

[14] V. Vapnik, A. Vashist, A new learning paradigm: Learning using privileged information, Neural networks 22 (5-6) (2009) 544–557.

[15] V. Vapnik, R. Izmailov, et al., Learning using privileged information: similarity control and knowledge transfer., J. Mach. Learn. Res. 16 (1) (2015) 2023–2049.

[16] A. Cornuéjols, P.-A. Murena, R. Olivier, Transfer learning by learning projections from target to source, in: Advances in Intelligent Data Analysis XVIII: 18th International Symposium on Intelligent Data Analysis, IDA 2020, Konstanz, Germany, April 27–29, 2020, Proceedings 18, Springer, 2020, pp. 119–131.

[17] R. E. Schapire, Y. Freund, Boosting: Foundations and Algorithms, MIT Press, 2012.
[18] M. Mohri, A. Rostamizadeh, A. Talwalkar, Foundations of machine learning, MIT press, 2018.
[19] Y. Achenchabe, A. Bondu, A. Cornuéjols, A. Dachraoui, Early classification of time series: Cost-based optimization criterion and algorithms, Machine Learning 110 (6) (2021) 1481–1504.
[20] A. Bondu, Y. Achenchabe, A. Bifet, F. Clérot, A. Cornuéjols, J. Gama, G. Hébrail, V. Lemaire, P.-F. Marteau, Open challenges for machine learning based early decision-making research, ACM SIGKDD Explorations Newsletter 24 (2) (2022) 12–31.
[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, the Journal of machine Learning research 12 (2011) 2825–2830.
[22] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1717–1724.
[23] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, Advances in neural information processing systems 27.
[24] H. Venkateswara, S. Chakraborty, S. Panchanathan, Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations, IEEE Signal Processing Magazine 34 (6) (2017) 117–129.
[25] S. Marullo, M. Tiezzi, M. Gori, S. Melacci, Friendly training: Neural networks can adapt data to make learning easier, in: 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–8.
[26] D. R. Hofstadter, Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thought., Basic books, 1995.