LEARNING CATEGORIES FROM FEW EXAMPLES WITH MULTI MODEL KNOWLEDGE TRANSFER

Tatiana Tommasi    Francesco Orabona
Barbara Caputo

Idiap-RR-16-2013

APRIL 2013
Learning Categories from Few Examples with Multi Model Knowledge Transfer

Tatiana Tommasi, Francesco Orabona and Barbara Caputo

Abstract—Learning a visual object category from few samples is a compelling and challenging problem. In several real-world applications collecting many annotated data is costly and not always possible. However a small training set does not allow to cover the high intraclass variability typical of visual objects. In this condition, machine learning methods provide very few guarantees. This paper presents a discriminative model adaptation algorithm able to proficiently learn a target object with few examples by relying on other previously learned source categories. The proposed method autonomously chooses from where and how much to transfer information by solving a convex optimization problem which ensures to have the minimal leave-one-out error on the available training set. We analyze several properties of the described approach and perform an extensive experimental comparison with other existing transfer solutions, consistently showing the value of our algorithm.

Index Terms—Knowledge Transfer, image categorization, discriminative learning

1 INTRODUCTION

As human beings, our learning ability develops progressively in time. At the age of six, we recognize around $10^4$ object categories [1] and we go on learning more while we grow up. All the information acquired through our five senses are encoded and stored in memory, with concepts and categories organized on the basis of their common properties. This intrinsically means that any new concept is not learned in isolation, but considering connections to what is already known, which makes the skill of building analogies one of the cores of human intelligence [2]. Even focusing only on visual tasks, we can give several examples of this cognitive ability. Have you ever seen a guava or an okapi? The guava is a fruit that externally looks like a lime, while its inner part is similar to an apple. An okapi is an animal that can be roughly described as a horse, with the legs of a zebra and the head of a giraffe (see Figure 1). Once we have seen a single image for each of the two target objects, we can easily memorize and recognize them by referring to the source objects mentioned in the provided description. In psychology this process is known as knowledge transfer: it encompasses phenomena ranging from simple (e.g. generalization of conditioned response between familiar and novel stimuli) to extremely complex (e.g. carrying over a solution from a problem in arithmetic to a novel class of problems) behaviors [3], and it makes learning further concepts extremely efficient. This capacity allows us to evaluate many kinds of recurrent patterns and regularities, giving the possibility to make inductive inferences on a new task even with only a small amount of data.

Fig. 1. The knowledge of the appearance of several objects can be used when learning something new. For example, using the source knowledge on fruits and animals while learning guava and okapi.

A large part of recent literature on visual object categorization focuses on reaching impressive results on large and difficult datasets [4], [5]. However, these works rarely refer to the effort done in collecting the data. In many real applications gathering fully annotated images can be extremely time consuming and might have a significant impact on the overall cost of the final system.

Standard learning techniques do not handle well the case of very small training sets. Differently from the described human cognition mechanism, all the learning approaches consider each task separately with respect to other possible source of relative information. Reproducing the knowledge transfer process in this scenario might consistently boost the learning performance. The basic intuition is that, if a system has already learned $j$ categories, learning the $(j+1)$-th should be easier even from one or few training samples [6].

A first practical implementation of the knowledge transfer...
idea was presented in [7] following a Bayesian approach. A
generic object model is estimated from some source categories
and it is then used as prior to estimate the target object
parameter distribution with a maximum-a-posteriori technique.
This work left some open questions discussed in its conclusive
section: (i) All the different known source categories are used
together to define a single prior; would a more sophisticated
multi-modal prior be beneficial in learning? (ii) Is there any
other productive point of view beside the generative Bayesian
one that allows to incorporate prior knowledge? (iii) Is it easier
to learn new target categories which are similar to some of
the source categories? Several other works in the computer
vision literature followed this first attempt [8], [9], [10], [11],
introducing different methods to increase the categorization
performance with respect to learning from scratch in case of
few available samples. However, due to the small differences
in the chosen settings, the proposed solutions were never
compared among each other.

In this work we focus on knowledge transfer across visual
object categories and our main contribution is a learning
algorithm that directly addresses the open problems in [7]. We
consider (i) the availability of several separate source models
and we introduce (ii) a discriminative approach based on
Least Square Support Vector Machines (LS-SVM, [12]). Any
new target class is learned through adaptation by imposing
closeness between the target classifier and a linear combination
of the source classifiers already learned on the \( j \) object sources.
The weight assigned to each source knowledge is defined
by solving a convex optimization problem which minimizes
an upper bound of the leave-one-out error on the training
set providing a principled solution for choosing from where
to transfer and how much to rely on each known source.
In practice, the proposed method (iii) autonomously tunes
the transfer process depending on the similarity between the
sources and the target tasks. We analyze in detail several
properties of the described approach and perform an extensive
experimental comparison with other existing transfer solutions,
consistently showing the value of our algorithm.

The rest of the paper is organized as follows. Section 2
provides a short introduction to the goals, challenges and
possible scenarios of knowledge transfer. Section 3 briefly
reviews the literature. A detailed description of the notation
and of the mathematical framework for our method follows
in Section 4. Section 5 contains the formal definition of
our knowledge transfer algorithm. Section 6 introduces an
extension to the case of heterogeneous sources. Finally in
Section 7 we present a thorough experimental evaluation of
our approach with a benchmark against several other state
of the art approaches. Section 8 concludes the paper with an
overall discussion and pointing out possible avenues for future
research.

2 Knowledge Transfer: Issues and Scenarios

The main assumption in theoretical models of learning, such
as the standard PAC (Probably Approximately Correct [13])
model, is that training instances are drawn according to the
same probability distribution as the unseen test examples.
This hypothesis permits the estimation of the generalization
error and the uniform convergence theory [14] provides basic
guarantees on the correctness of future decisions.

This ideal assumption is not always true in practical prob-
lems. It can happen that we have a lot of labeled data on a
source problem and the need to solve a different target
problem with few labeled samples, where source and target
present a distribution mismatch. In this case knowledge trans-
ferral (a.k.a learning [15]) may decrease the effort of
collecting new samples, while at the same time it may reduce
the lack of robustness issue (risk of overfitting) by leveraging
over the existing source knowledge to solve a target task. It is
possible to define three measures by which transfer improves
the effectiveness of learning; we list them below, referring to
Figure 2.

(1) Higher start: the initial performance achievable on the
target task is much better compared to that of an ignorant
agent [16]. This is true even using only the source transferred
knowledge, before any further learning on the target problem.

(2) Higher slope: this indicates a shorter amount of time
needed to fully learn the target task, given the transferred
knowledge, in comparison with learning from scratch [16].

(3) Higher asymptote: in the long run, the final performance
level achievable over the target task can be higher compared
to the final level without transfer [16].

How to get these advantages and up to which extent the
transfer process can be useful depends on the specific sce-
nario at hand (object categorization, recognition, segmentation,
reinforcement learning, etc.) and on the relation between
source and target tasks. Apart from the different levels of
semantic similarity, source and target might be represented
with the same or with different descriptors which give rise
respectively to a homogeneous or a heterogeneous transfer
process. Moreover, a transfer learning problem can scale with
respect to the number of annotated target samples and of
possible source sets (see Figure 3). Indeed, to fully define
any knowledge transfer method it is necessary to answer to
three main questions.

(1) What to transfer? It refers to which knowledge can be
transferred and to the form in which it is coded. In general
terms, some knowledge might be specific for a task while some
other knowledge might be common and sharable.

(2) How to transfer? This question is about the definition of

Fig. 2. Three ways in which transfer might improve
the learning performance when the number of target training
samples increases. Forcing the target learning process to
rely on unrelated sources produces the negative transfer
effect. (Figure reproduced and adapted from [16]).
3.2 How to Transfer

A large variety of methods have been studied to integrate in different ways the source and target information: boosting approaches [18], [9], KNN [25], Markov logic [26], graphical models [27]. Most of the work has however been done in the generative probabilistic setting. Given the data, the target model makes predictions by combining them with the prior source distribution to produce a posterior distribution. A strong prior significantly affects these results serving as a natural way for Bayesian learning methods to transfer source knowledge. Some discriminative (maximum margin) methods are presented in [21] by learning a distance metric, and in [11], [28] where a template learned previously for some object categories is used to regularize the training of a new target category for detection.

3.3 When to Transfer

Several knowledge transfer solutions have been proposed in the setting of useful source knowledge [10], [18]. However, in real learning scenarios, the information acquired in the past is not always relevant for a new target problem. Rosenstain et al. [29] empirically showed that if two tasks are dissimilar, brute force transfer hurts the performance producing the so called negative transfer (see Figure 2).

Ideally, a transfer method should be beneficial between appropriately related tasks while avoiding negative transfer when the tasks are not a good match. In practice, these goals are difficult to achieve simultaneously. Approaches that have safeguards to avoid negative transfer often produce a smaller effect from positive transfer due to their caution. Conversely, approaches that transfer aggressively and produce large positive-transfer effects often have no protection against negative transfer.

It is possible to identify two main strategies to decide when to transfer. One consists in rejecting bad information or at
least making sure that its impact is minimized so that the transfer performance is at least not worse than what obtained by learning only on the target task. This means that it is always necessary to choose how much to transfer, possibly disregarding the transferred knowledge completely. A different strategy can be applied when there are more than one source task: in this condition the problem becomes choosing the best source. Transfer methods without much protection against negative transfer may still be effective in this scenario, as long as the best source task is at least a decent match. Taylor et al. [29] proposed a transfer hierarchy, sorting the tasks by difficulty. Given a task ordering, it may be possible to locate the position of the target task in the hierarchy and select the most useful source set. In [30] the authors used conditional Kolmogorov complexity to measure relatedness between tasks and transfer the right amount of information.

Our work fits in this context. We propose a discriminative knowledge transfer method that relies on a set of models learned on the source categories (what to transfer) which are then used to regularize the target object model (how to transfer). The relatedness among the tasks is automatically evaluated (when to transfer) through a principled optimization problem without any need of hand tuned parameters, extra validation samples or a pre-defined ontology.

4 Mathematical Framework

We introduce here the formal notation that will be used in the paper and we present all the mathematical tools necessary to define our knowledge transfer approach. In the following we denote with small and capital bold letters respectively column vectors and matrices, e.g. \( \mathbf{a} = [a_1, a_2, \ldots, a_N]^T \in \mathbb{R}^N \) and \( \mathbf{A} \in \mathbb{R}^{M \times N} \) with \( A_{ji} \) corresponding to the \((j, i)\) element. When only one subscripted index is present, it represents the column index, e.g., \( \mathbf{A}_i \) is the \(i\)-th column of the matrix \( \mathbf{A} \). Moreover we indicate with \( \|\mathbf{a}\|_p := \left(\sum_{i=1}^N |a_i|^p\right)^{1/p} \) the \(p\)-norm of a vector \( \mathbf{a} \in \mathbb{R}^N \).

Let us assume \( \mathbf{x}_i \in \mathcal{X} \) to be an input vector to a learning system and \( y_i \in \mathcal{Y} \) its associated output. Given a set of data \( \mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N \) drawn from an unknown probability distribution \( \mathcal{P} \), we want to find a function \( f : \mathcal{X} \rightarrow \mathcal{Y} \) such that it determines the best corresponding \( y \) for any future sample \( \mathbf{x} \). We consider \( \mathcal{X} \subseteq \mathbb{R}^d \) and \( \mathcal{Y} = \{-1, 1\} \).

The described learning process can be formalized as an optimization problem which aims at finding \( f \) in the hypothesis space of functions \( \mathcal{H} \), which minimizes the structural risk [14]

\[
\Omega(f) + C \sum_{i=1}^N \ell(f(x_i), y_i) .
\]  

Here \( \Omega(f) \) is a regularizer, which encodes some notion of smoothness for \( f \), and guarantees good generalization performance avoiding overfitting. In the second term, \( \ell \) is some convex non-negative loss function which assesses the quality of the function \( f \) on the instance and label pair \( \{x_i, y_i\} \). In practice it expresses the price we pay by predicting \( f(x_i) \) in place of \( y_i \). The predictivity is a trade-off between the information provided by the training data and the complexity of the solution we are looking for, controlled by the parameter \( C > 0 \).

4.1 Adaptive Regularization

We set \( \mathcal{H} \) equal to space of all the linear models of the form

\[
f(x) = \mathbf{w}^\top \phi(x) + b .
\]

Here \( \phi(x) \) is a feature mapping that maps the samples into a high, possible infinite dimensional space, where the dot product is expressed with a functional form \( K(x, x') = \phi(x)^\top \phi(x') \) named kernel [31]. We also set the regularizer to be \( \Omega(f) = \frac{1}{2} \|\mathbf{w}\|^2 \), so that, regardless of the specific form of the loss function, the learning problem (1) becomes

\[
\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \ell(\mathbf{w}^\top \phi(x_i) + b, y_i) .
\]

In this classical scheme for inductive learning, the knowledge eventually gained on the data \( \mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N \) extracted from a distribution \( \mathcal{P} \), different with respect to the target one \( \mathcal{P} \), is not taken into consideration. However, if \( N \gg N \) with a small number of available samples \( N \sim 10 \) and if the two distributions \( \mathcal{P}, \mathcal{P} \) are somehow related, the auxiliary knowledge can be helpful in guiding the learning process.

Let us suppose that the optimal \( \hat{\mathbf{w}} \) has been already found by minimizing (3) for some source problem. When facing a new target task, we can always ask \( \mathbf{w} \) to be close to the known \( \hat{\mathbf{w}} \) by simply changing the regularization term [32] such that the learning problem results

\[
\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \hat{\mathbf{w}}\|^2 + C \sum_{i=1}^N \ell(\mathbf{w}^\top \phi(x_i) + b, y_i) .
\]

Thus, apart from minimizing the original term \( \|\mathbf{w}\|^2 \), the optimization problem aims now at obtaining a vector \( \mathbf{w} \) close to the source model \( \hat{\mathbf{w}} \) by maximizing the projection of the first on the second. To properly scale the importance of this projection in the optimization problem, it is possible to add a weighting factor \( \beta \) such that the regularizer becomes \( \|\mathbf{w} - \beta \hat{\mathbf{w}}\|^2 \).

5 Multi Model Knowledge Transfer

Consider the following situation. We want to learn the target object class okapi from few examples, having already a model for the source categories horse, zebra, melon and apple. On the basis of the visual similarity, we can guess that the final model for okapi will be close to that of horse and zebra. Thus in the learning process we would like to transfer information from these two categories. We would expect the model obtained in this way to produce better recognition results with respect to (i) just considering horse or zebra as reference, and (ii) relying over all the source knowledge in a flat way, as melon and apple might induce negative transfer. This kind of reasoning motivates us to design a knowledge transfer algorithm able to find autonomously the best subset of known models from where to transfer, and to weight properly the relevant information.
Any transfer method, based on the adaptive regularization described in the previous section, answers the question what to transfer in terms of model parameters, by passing the known \( \hat{w} \) to the new target problem. However, previous work did not pay too much attention on when and how much to transfer. The discussed weight factor \( \beta \) in the regularizer is usually set equal to 1 with the hypothesis that the known models are useful and related to the target problem [33]. In other cases \( \beta \) is treated as a learning parameter, and is chosen by cross validation assuming the availability of extra target training samples [11]. Both these choices present some issues: the first case does not consider the danger of negative transfer when only unrelated prior information is available, while in the second, the existence of extra data for cross validation is incoherent with the small sample scenario of transfer learning.

Here we study instead the case of multiple (\( J \)) available sources. We propose a learning method which relies over all of them and assigns to each a weight \( \beta_j \) for \( j = 1, \ldots, J \). These values are automatically tuned on the basis of the few available target training data. We name our algorithm Multi Model Knowledge Transfer (Multi-KT) and we present its basic components in the following subsections.

### 5.1 Adaptive Least-Square Support Vector Machine

The first step to define our transfer learning algorithm consists in combining linearly the source models to have \( \sum_{j=1}^J \beta_j \hat{w}_j \) and using this as reference instead of the single source in (4). Moreover, we choose the weighted square loss \( \ell(f(x_i), y_i) = \zeta_i (f(x_i) - y_i)^2 \) [34], where the parameter \( \zeta_i \) can be used to balance the contribution of positive and negative samples, taking into account that their proportion in the training set may be not representative of the operational class frequency.

The obtained optimization problem is:

\[
\begin{align*}
\min_{w,b} \quad & \frac{1}{2} \left\| w - \sum_{j=1}^J \beta_j \hat{w}_j \right\|^2 + C \sum_{i=1}^N \zeta_i \xi_i^2 \\
\text{subject to} \quad & y_i = w^T \phi(x_i) + b + \xi_i, \\
& \text{for } i = 1, \ldots, N. 
\end{align*}
\]

(5)

where we have introduced the slack variables \( \xi_i \) which measure the degree of misclassification on the data \( x_i \). Thus we obtain the soft version of Least Square Support Vector Machine (LS-SVM) [12], that use the adaptive regularizer introduced before. The corresponding Lagrangian \( \mathcal{L} \) is

\[
\mathcal{L} = \frac{1}{2} \| w - \sum_{j=1}^J \beta_j \hat{w}_j \|^2 + C \sum_{i=1}^N \zeta_i \xi_i^2 - \sum_{i=1}^N a_i \{ w^T \phi(x_i) + b + \xi_i - y_i \}.
\]

(6)

Here \( a \in \mathbb{R}^N \) is the vector of Lagrange multipliers and the optimality condition with respect to \( w \) is

\[
\frac{\partial \mathcal{L}}{\partial w} = 0 \implies w = \sum_{j=1}^J \beta_j \hat{w}_j + \sum_{i=1}^N a_i \phi(x_i) .
\]

(7)

Thus, the adapted model is given by the weighted sum of the pre-trained source models \( \hat{w}_j \) and a linear combination of the target samples. Note that when all the \( \beta_j \) are 0 we recover the original LS-SVM formulation without any adaptation to previous knowledge. Considering also the derivative of \( \mathcal{L} \) with respect to \( \xi_i \) and \( a_i \), we have respectively \( a_i = C \zeta_i \xi_i \) and \( w^T \phi(x_i) + b + \xi_i - y_i = 0 \). By combining them with (7) we find

\[
\sum_{k=1}^N a_k \phi(x_k)^T \phi(x_i) + b + \xi_i = y_i - \sum_{j=1}^J \beta_j \hat{w}_j^T \phi(x_i) .
\]

(8)

Denoting with \( K \) the kernel matrix, i.e. \( K_{ij} = K(x_j, x_i) = \phi(x_j)^T \phi(x_i) \), the obtained system of linear equations can be written more concisely in matrix form as

\[
\begin{bmatrix}
K + \frac{1}{2} Z & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
a \\
b
\end{bmatrix} = \begin{bmatrix}
y - \sum_{j=1}^J \beta_j \hat{y}_j \\
0
\end{bmatrix},
\]

(9)

where \( y \) and \( \hat{y}_j \) are the vectors containing respectively the label of each training sample and the prediction of the previous model \( j \), i.e. \( y = [y_1, \ldots, y_N]^T \) and \( \hat{y}_j = [\hat{w}_j^T \phi(x_1), \ldots, \hat{w}_j^T \phi(x_N)]^T \). Moreover, \( Z = \text{diag}(\zeta_1, \zeta_2, \ldots, \zeta_N) \) and to balance the contribution of differently labeled samples to the misfit term we set

\[
\zeta_i = \begin{cases} 
\frac{N}{\sum_{i} \zeta_i} & \text{if } y_i = 1 + 1 \\
\frac{N}{\sum_{i} \zeta_i} & \text{if } y_i = 1. 
\end{cases}
\]

(10)

Here \( N^+ \) and \( N^- \) represent the number of positive and negative examples respectively.

Finally, the model parameters can be calculated simply by matrix inversion:

\[
\begin{bmatrix}
a \\
b
\end{bmatrix} = P \begin{bmatrix}
y - \sum_{j=1}^J \beta_j \hat{y}_j \\
0
\end{bmatrix},
\]

(11)

where \( P = M^{-1} \) and \( M \) is the first matrix on the left in (9). We underline that the pre-trained models \( \hat{w}_j \) can be obtained by any training algorithm, as long as it can be expressed as a weighted sum of kernel functions; the framework is therefore very general.

### 5.2 When and How Much to Transfer

Finding the optimal value for the elements of the weight vector \( \beta \) corresponds to ranking the prior knowledge sources and decide from where and how much to transfer. We propose to choose \( \beta \) in order to minimize the leave-one-out error, which is an almost unbiased estimator of the generalization error [34]. While in general computing the leave-one-out error is a very expensive procedure, we show that for (5) it can be computed with a closed-formula, using quantities that are already computed for the training.

Let us denote by \( \hat{y}_i \), \( i = 1, \ldots, N \), the prediction on sample \( i \) when it is removed from the training set. LS-SVM in its original formulation makes it possible to write these leave-one-out predictions in closed form and with a negligible additional computational cost [34]. We show below that the same property extends to the modified problem in (5).

**Proposition 1:** Let \( [a^T, b]^T = P[y^T, 0]^T \) and \( [a'^T, b'^T]^T = P[y^T, 0]^T \) with \( a = a' - \sum_{j=1}^J \beta_j a_j^h \). If we indicate with \( A^h \) the matrix containing the vector \( a_j^h \) in
the j-th row, the prediction \( \tilde{y}_i \), obtained on sample i when it is removed from the training set, is equal to
\[
y_i = \frac{a'_i}{P_{ti}} + \frac{\beta'_j A''}{P_{ti}},
\]
where \( \beta \in \mathbb{R}^J \) is a vector containing all the values \( \beta_j \).

**Proof of Proposition 1:** We start from
\[
\begin{bmatrix}
a \\
b
\end{bmatrix} = \begin{bmatrix}
y - \sum_{j=1}^{J} \beta_j \tilde{y}_j
\end{bmatrix},
\]
and we decompose \( M \) into block representation isolating the first row and column as follows:
\[
M = \begin{bmatrix}
K + \frac{1}{J} Z & 1 \\ 
1 & 0
\end{bmatrix} = \begin{bmatrix}
m_{11} & m_{11}^\top \\
m_1 & M_{(-1)}
\end{bmatrix}.
\]
Let \( a_{(-i)} \) and \( b_{(-i)} \) represent the model parameters during the i-th iteration of the leave-one-out cross validation process. In the first iteration, where the first training sample is excluded we have
\[
\begin{bmatrix}
a_{(-1)} \\
b_{(-1)}
\end{bmatrix} = P_{(-1)} (y_{(-1)} - \sum_{j=1}^{J} \beta_j \tilde{y}_j_{(-1)}),
\]
where \( P_{(-1)} = M_{(-1)}^{-1} \), \( y_{(-1)} = [y_2, \ldots, y_N, 0]^\top \) and \( \tilde{y}_j_{(-1)} = [\tilde{w}_j^{(1)} \phi(x_2), \ldots, \tilde{w}_j^{(1)} \phi(x_N), 0]^\top \). The leave-one-out prediction for the first training sample is then given by
\[
\tilde{y}_1 = m_1^\top \begin{bmatrix}
a_{(-1)} \\
b_{(-1)}
\end{bmatrix} + \sum_{j=1}^{J} \beta_j \tilde{w}_j^{(1)} \phi(x_1)
\]
= \( m_1^\top P_{(-1)} (y_{(-1)} - \sum_{j=1}^{J} \beta_j \tilde{y}_j_{(-1)})) + \sum_{j=1}^{J} \beta_j \tilde{w}_j^{(1)} \phi(x_1) \).

Considering the last \( N \) equations in the system in (13), it is clear that \( m_1 M_{(-1)} [a, b]^\top = (y_{(-1)} - \sum_{j=1}^{J} \beta_j \tilde{y}_j_{(-1)}) \), and so
\[
\tilde{y}_1 = m_1^\top P_{(-1)} m_1 \begin{bmatrix}
a_{(-1)} \\
b_{(-1)}
\end{bmatrix} + \sum_{j=1}^{J} \beta_j \tilde{w}_j^{(1)} \phi(x_1)
\]
= \( m_1^\top P_{(-1)} m_1 \begin{bmatrix}
a_{(-1)} \\
b_{(-1)}
\end{bmatrix} + m_1^\top m_{11}^\top \tilde{y}_1 + \sum_{j=1}^{J} \beta_j \tilde{w}_j^{(1)} \phi(x_1) \).

In (13) the first equation of the system is \( y_1 = \sum_{j=1}^{J} \beta_j \tilde{w}_j^{(1)} \phi(x_1) = m_{11} a_1 + m_1^\top \begin{bmatrix}
a_2, \ldots, a_N, b
\end{bmatrix}^\top + \sum_{j=1}^{J} \beta_j \tilde{w}_j^{(1)} \phi(x_1) \).

By defining \([a'^\top, b'^\top]^\top = P [y_{(z)}^\top, 0]^\top \), \([a''^\top, b''^\top]^\top = P [\tilde{y}_{(z)}^\top, 0]^\top \), and \( a' = a - \sum_{j=1}^{J} \beta_j a''_j \), from the equation above we get
\[
\tilde{y}_i = y_i - \frac{a'_i}{P_{ti}} + \sum_{j=1}^{J} \beta_j A''_{ij} P_{ti} = y_i - \frac{a'_i}{P_{ti}} + \frac{\beta^\top A''}{P_{ti}},
\]
where \( \beta \in \mathbb{R}^J \) is a vector containing all the values \( \beta_j \) and \( A'' \) is the matrix containing the vector \( a''_j \) in the j-th row.

Notice that in (12) \( a \) depends linearly on \( \beta \), thus it is straightforward to obtain the learning model once all the \( \beta_j \) have been chosen. By multiplying the correct label \( y_i \) to (12) we get
\[
y_i \tilde{y}_i = 1 - y_i \left( \frac{a'_i}{P_{ti}} - \frac{\beta^\top A''}{P_{ti}} \right),
\]
thus the best values for \( \beta_j \) are those producing positive values for \( y_i \tilde{y}_i \), for each \( i \). However, focusing only on the sign of those quantities would result in a non-convex formulation with many local minima. We propose instead the following loss function:
\[
\ell(\tilde{y}_i, y_i) = \zeta_i \max \{0, 1 - y_i \tilde{y}_i\}
\]
= \( \zeta_i \max \{0, y_i \left( \frac{a'_i}{P_{ti}} - \frac{\beta^\top A''}{P_{ti}} \right) \} \).

This loss function is similar to the hinge loss. It is a convex upper bound to the leave-one-out classification loss and it favors solutions in which \( \tilde{y}_i \) has an absolute value equal or bigger than 1, and the same sign of \( y_i \). The weight \( \zeta_i \) is set again according to (10). Finally, the objective function is
\[
\min_\beta \sum_{i=1}^{N} \ell(\tilde{y}_i, y_i) \quad \text{subject to} \quad \|\beta\|_p \leq 1, \quad \beta_j \geq 0,
\]
where we added some constraint on the \( \beta \) vector as a form of regularization. They may be helpful to avoid overfitting problems when the number of known models \( J \) is large compared to the number of training samples \( N \). Depending on the value of \( p \), how the target learning model leverages over the source models changes:

**p = 2, L_2 norm constraint.** This is the well known Euclidean norm indicated by \( \| \cdot \|_2 \) or simply \( \| \cdot \| \). A regularization based on it generally induces numerical stability. The optimization process can be implemented by using a simple projected sub-gradient descent algorithm, where at each iteration \( \beta \) is projected onto the \( L_2 \)-sphere \( \|\beta\|_2 \leq 1 \) and then on the positive semi-plane. The pseudo-code is in Algorithm 1.

**p = 1, L_1 norm constraint.** This is simply the sum of the absolute values of the vector elements. This constraint induces a sparse solution, i.e. only some vector elements remain different from zero. Applied on prior knowledge regularization, the condition \( \|\beta\|_1 \leq 1 \) can be easily implemented, e.g. on the basis of the algorithm proposed in [35], and it gives rise to an automatic source selection technique.

**p = \infty, L_\infty norm constraint.** This norm is defined as
\[
\|x\|_\infty := \max\{|x_1|, \ldots, |x_d|\}.
\]

In practice, by using $\|\beta\|_{\infty} \leq 1$ as regularizer we are imposing that all the vector elements assume separately an absolute value not bigger than one. In this case the projection consists of a simple truncation.

The second condition in (16) limits the weights of the source knowledge models to be positive. In fact, in the object category detection problem, all the considered source and target sets have the background category as common negative class, thus it is reasonable to expect that the angle between $w$ and any $\hat{w}_j$ is always acute.

**Algorithm 1**: Projected Sub-gradient Descent Algorithm

**Input**: calculate $a'_i, a''_j$ and set $A'^T$ according to Proposition 1

**Initialize**: $\beta \leftarrow 0$ and $t \leftarrow 1$

repeat

\[
\tilde{y}_i \leftarrow y_i - \frac{a'_i}{\pi_i} + \sum_{j=1}^{J} \beta_j \frac{a''_j}{\pi'_j}, \quad \forall i = 1, \ldots, N
\]

\[
d_i \leftarrow 1\{\tilde{y}_i > 0\}, \quad \forall i = 1, \ldots, N
\]

\[
\beta_j \leftarrow \beta_j - \frac{1}{\sum_{i=1}^{N} d_i \tilde{y}_i \pi_i}, \quad \forall j = 1, \ldots, J
\]

if $\|\beta\|_2 > 1$ then

\[
\beta \leftarrow \frac{\beta}{\|\beta\|_2}
\]

end if

\[
\beta_j \leftarrow \max(\beta_j, 0), \quad \forall j = 1, \ldots, J
\]

$t \leftarrow t + 1$

until convergence

**Output**: $\beta$

5.3 Computational Complexity

From a computational point of view the runtime of the Multi-KT algorithm is $O(N^3 + JN^2)$, with $N$ the number of training samples, and $J$ the number of source models. The first term is related to the evaluation of the matrix $P$, which must anyway occur while training, while the second term is the computational complexity of (12), which results negligible, if compared to the complexity of training. Thus, we match the complexity of a plain SVM, which in the worst case is known to be $O(N^3)$ [36]. The computational complexity of each step of the projected sub-gradient descent to optimize (15) is $O(JN)$, and it results extremely fast (our MATLAB implementation takes just half a second with $N = 12$ and $J = 3$ on current hardware).

6 Heterogeneous Knowledge Transfer

The proposed Multi-KT transfer method is based on the idea of pushing the target model to close to a linear combination of prior known sources $\sum_{j=1}^{J} \beta_j \hat{w}_j$. However, to impose this closeness, all the vectors should live in a single space. This means that the kernel used in learning over all the sources and on the new target must be the same. This is quite a strict condition because it does not give the freedom to build the source knowledge over heterogeneous feature descriptors, and imposes a unique metric to evaluate the sample similarity.

We show here that this limit may be easily overcome by enlarging the space in which we seek the final learning function on the target task, by a multi-kernel approach. We call this variant MultiK-KT.

Let us assume to have $j = 1, \ldots, J$ mappings, each to a different space, where the image of a vector $x$ is $\phi_j(x)$. We can always compose all of them orthogonally (see Figure 4) obtaining the mapping to the final space by concatenation: $\phi'(x) = [\phi_1(x), \phi_2(x), \ldots, \phi_J(x)]^T$. The dot product in this new space is expressed by the kernel $K'$

\[
\phi'(x)^T \phi'(z) = \sum_{j=1}^{J} \phi_j(x)^T \phi_j(z)
\]

\[
= \sum_{j=1}^{J} K_j(x, z) = K'(x, z),
\]

where $K_j(x, z)$ is the kernel function in the $j$-th space.

Now let us consider the transfer learning problem with $j = 1, \ldots, J$ source object classes and suppose to solve the binary classification object-vs-background for each of them in a specific space, i.e. choosing different feature descriptors, different kernel functions, and/or different kernel parameters. The obtained model vectors are

\[
\tilde{w}_j = \sum_{i=1}^{N_j} \alpha_i^j \phi_j(x_i).
\]

These solutions can always be mapped in the composed new space using zero padding. In fact, $\phi_j(x) \rightarrow \phi'_j(x) = [0, \ldots, \phi_j(x), \ldots, 0]^T$, we have

\[
\tilde{w}_j \rightarrow \tilde{w}'_j = [0, \ldots, \tilde{w}_j, \ldots, 0]^T
\]

\[
= [0, \ldots, \sum_{i=1}^{N_j} \alpha_i^j \phi_j(x_i), \ldots, 0]^T.
\]

Hence, in the new space, a vector obtained as linear combination of all the known models results:

\[
\sum_{j=1}^{J} \beta_j \tilde{w}'_j = [\beta_1 \tilde{w}_1, \ldots, \beta_J \tilde{w}_J]^T
\]

\[
= [\beta_1 \sum_{i=1}^{N_1} \alpha_i^1 \phi_1(x_i), \ldots, \beta_J \sum_{i=1}^{N_J} \alpha_i^J \phi_J(x_i)]^T.
\]

By supposing that the target problem lives in the new composed space, we can apply our Multi-KT algorithm there. Hence the original optimization problem in (5) becomes

\[
\min_{w^0, b} \frac{1}{2} \left\| w' - \sum_{j=1}^{J} \beta_j \tilde{w}'_j \right\|^2 + C \sum_{i=1}^{N} \zeta_i (y_i - w'^T \phi'(x_i) - b)^2.
\]

(19)

The solving procedure is the same described in Section 5.1 and the optimal solution is:

\[
w' = \sum_{j=1}^{J} \beta_j \tilde{w}'_j + \sum_{i=1}^{N} a_i \phi'(x_i).
\]

When we use it for classification we get

\[
w'^T \phi'(z) = \sum_{j=1}^{J} \beta_j \tilde{w}'_j^T \phi'(z) + \sum_{i=1}^{N} a_i \phi'(x_i)^T \phi'(z)
\]

\[
= \sum_{j=1}^{J} \beta_j \tilde{w}'_j^T \phi_j(z) + \sum_{i=1}^{N} a_i \left( \sum_{j=1}^{J} \phi_j(x_i)^T \phi_j(z) \right),
\]

that is exactly the same that would be obtained from (7) using

\[
\text{Output:}
\]
Fig. 4. For Multi-KT the first the source and target models must live in the same space identified by the kernel \( K \). For MultiK-KT all the sources can be defined independently in their own space and the target solution lives in the space obtained by orthogonal combination. We show also a geometrical interpretation of the kernel combination.

\[
K'(x, z) \text{ as kernel. Even the original procedure to choose the best } \beta \text{ can be easily enlarged to the case of linearly combined orthogonal spaces. The vector } \hat{y}_j' \text{ containing the predictions of the } j\text{-th known model is:}
\]
\[
\hat{y}_j' = [\hat{w}_j^T \phi'(x_1), \ldots, \hat{w}_j^T \phi'(x_N)] = [\hat{w}_j^T \phi_j(x_1), \ldots, \hat{w}_j^T \phi_j(x_N)] = \hat{y}_j .
\]

This indicates that there is no real changing: Multi-K-KT is formally equivalent to the original Multi-KT with the kernel chosen as in (18). As a consequence the computational complexity of MultiK-KT is again \( O(N^3 + JN^2) \) (see Section 5.3).

7 Experiments

In this section we show empirically the effectiveness of our transfer algorithm\(^1\) on three datasets: Caltech-256 [37], Animals with Attributes (AwA) [24] and IRMA [38].

The Caltech-256 contains images of 256 object classes plus a clutter category that can be used as negative class in object-vs-background problems. Moreover, the objects are also organized in a hierarchical ontology that makes it easy to identify the related and unrelated categories. We downloaded\(^2\) the precomputed features of [39] and we selected four different image descriptors: PHOG Shape Descriptors [40], SIFT Appearance Descriptors [41], Region Covariance [42], and Local Binary Patterns [43]. They were all computed in a spatial pyramid [44] and we considered just the first level (i.e. information extracted from the whole image).

The AwA dataset contains 50 animal classes and it has been released with several pre-extracted feature representations for each image\(^3\). From the full set of categories we extracted the six sea mammals (killer whale, blue whale, humpback whale, seal, walrus and dolphin) and used them to define the background class. We used three of the precomputed descriptors for our experiments: color histogram, PHOG and SIFT.

The IRMA database is a collection of x-ray images presenting a large number of rich classes defined according to a four-axis hierarchical code [45]. We decided to work on the 2008 IRMA database version [38], just considering the third axis of the code: it describes the anatomy, namely which part of the body is depicted, independently to the used acquisition technique or direction. A total of 23 classes with more than 100 images were selected from various sub-levels of the third axis, 3 of them were used to define the background class. As features we used the global pixel-based and local SIFT-based descriptors following the experimental setup in [46].

We performed all the experiments in a leave-one-class-out approach, that is considering in turn each class as target and all the others as sources. The number of negative training samples is kept fixed while the number of positive training samples increases in subsequent steps till reaching the same amount of the negative set. The samples are extracted randomly 10 times for an equal number of experimental runs. Each prior knowledge model is built with classical LS-SVM. We use the Gaussian kernel both on the source and on the target for all the experiments \( K(x, x') = \exp(-\gamma \|x - x'|^2) \) (exceptions for the heterogeneous case are explicitly indicated). To integrate multiple (F) features we calculate one kernel for each of them and we use the average kernel \( K(x, x') = 1/F \sum_{f=1}^{F} K_f(x, x') \). All the transfer results are benchmarked against no transfer: this corresponds to learning from scratch with weighted-LS-SVM, i.e. solving the optimization problem in (5) with \( \beta = 0 \).

Regarding the parameters, a unique common value for \( \gamma \) was chosen for all the kernels by cross validation on the source sets. In particular, we trained a model for each class in the source set and we used it to classify on the remaining \( J - 1 \) source classes. Finally, we selected the \( \gamma \) value producing on average the best recognition rate. The value of \( C \) is instead determined as the one producing the best result when learning from scratch. There is no guarantee that the obtained \( C \) value is the best for the transfer approach; still in this way we compare against the best performance that can be obtained by learning only on the available training samples, without exploiting the source knowledge. We used this setup for all the experiments; specific differences are otherwise mentioned.

7.1 Setting the Constraints

To fully define the Multi-KT algorithm it is necessary to choose the \( p \) value in the constraint of (16). We evaluate empirically three cases with \( p = 1, 2, \infty \) and we compare the obtained results over three groups of data that differ in the level of relatedness among source and target knowledge. Specifically, we extracted 6 unrelated classes (harp, microwave, fire-truck, cowboy-hat, snake, bonsai), 6 related

\[\text{http://attributes.kyb.tuebingen.mpg.de/}\]
classes (all vehicles: bulldozer, fire-truck, motorbikes, school-bus, snowmobile, car-side) and 10 mixed classes (motorbikes, dog, cactus, helicopter, fighter, car-side, dolphin, zebra, horse, goose) from Caltech-256. We refer to a class as the combination of 80 object and 80 background images. For each class used as target, we extracted 20 training and 100 testing samples with half positive and half negative data.

The results in Figure 5 (top line) show the clear gain obtained by using Multi-KT with respect to no transfer. The advantage is maximum in case of related classes (the difference between Multi-KT \( L_2 \) and no transfer is 30\% in recognition rate for 1 positive sample), it is just a little bit smaller for mixed classes (34\%) and drops more in case of sources unrelated to the target task (29\%). However, regardless of the relatedness level, the choice of the constraint on prior knowledge weight \( \beta \) does not produce significantly different results\(^4\), apart for a slightly lower performance of the \( L_1 \) case with respect to the others. Hence, in the following we will always use the \( L_2 \) norm constraint.

### 7.2 Transfer Weights and Semantic Similarity

The Multi-KT algorithm defines automatically the relevance of each source model to the current target task. We analyze here the \( \beta \) vector obtained as a byproduct of the transfer process, to verify if its elements have a correspondence with the real visual and semantic relation among the tasks.

We start from the results obtained in the previous section with the \( L_2 \) norm constraint and we consider the intermediate training step with 5 positive samples. We average the \( \beta \) vectors obtained over the 10 runs defining a matrix of weights with one row for each class used as target. By simple algebra we can transform it to a fully symmetric matrix containing measures of class dissimilarities evaluated as \((1 - \beta_{ij})\) and apply multidimensional scaling on it [48]. To have an immediate visualization we considered only two dimensions and we obtain plots where each point represents a class, and the distance among the points is directly proportional to the input dissimilarities.

Figure 5 (bottom line) shows the obtained results. It can be seen that in the case of unrelated classes the corresponding points tend to be far from each other. On the other hand, among the related classes extracted from the general category motorized-ground-vehicles, the four wheels vehicles (fire-truck, school-bus and car-side) form a cluster, leaving aside motorbikes (two wheels), snowmobile (skis) and bulldozer (tracks). Finally, among the mixed classes, helicopter and fighter-jet appear close to each other and to dolphin. Probably this is due to the shape appearance of these object classes and to the common uniformity of the sky and water background. Moreover, all the four legged animals (zebra, horse and dog) appear on the right side of the plot, while the vehicles (car-side and motorbikes) are in the left bottom corner.

Globally all the results indicate that the \( \beta \) vectors actually contain meaningful values in terms of semantic relation between the object classes.

\(^4\) We used the sign test [47] to evaluate the statistical significance of the results for all the experiments.

### 7.3 Comparison and Evaluation

Here we evaluate our Multi-KT algorithm in comparison with several state of the art transfer learning approaches. We briefly review them before discussing the experimental results.

**Single Source.** Most of the existing knowledge transfer methods suppose the availability of a single source knowledge. Among the approaches listed below, the first two are based on transferring model parameters as our Multi-KT, while the last one is an instance transfer approach and exploits directly the source samples.

- **Adaptive SVM (A-SVM).** This method has been originally presented in [33] and is based on substituting the usual regularizer of the SVM formulation with the adaptive version

\[
\min_w \| w - \beta \hat{w} \|^2 + C \sum_{i=1}^{N} \ell_H (w^\top \phi(x_i), y_i).
\]

- **Projective Model Transfer SVM (PMT-SVM).** Maximizing the projection of \( w \) onto \( \hat{w} \) corresponds also to minimizing the projection of \( w \) onto the prior knowledge separating hyperplane (orthogonal to \( \hat{w} \)). Following this idea the objective function of PMT-SVM is

\[
\min_w \| w \|^2 + \beta \| R w \|^2 + C \sum_{i=1}^{N} \ell_H (w^\top \phi(x_i), y_i)
\]

subject to \( w^\top \theta \geq 0 \),

where \( R \) is the projection matrix and \( \| R w \|^2 = \| w \|^2 \sin^2 \theta \), where \( \theta \) is the angle between \( w \) and \( \hat{w} \).

- **TrAdaBoost: boosting for Transfer Learning.** A simple instance transfer approach can be defined by extending the AdaBoost learning framework. Specifically TrAdaBoost [18] considers a mix of source and target data in training and is based on a mechanism which decreases the weights of the source samples in order to weaken their impact. Finally, the source instances with large training weights help the learning algorithm to train better classifiers.

**Experiments.** We benchmark here our Multi-KT algorithm against the described A-SVM, PMT-SVM and TrAdaBoost. Since these baseline methods were defined in the hypothesis of a single available source set, we considered two cases: a pair of unrelated and a pair of related classes. Both the pairs were extracted from Caltech-256 and each of the classes is considered in turn as target while the other represents the source task.

We used the MATLAB code of PMT-SVM provided by its authors, together with their implementation of A-SVM \(^5\) slightly modifying them to introduce the weights \( \zeta_i \) for \( i = 1, \ldots, N \) in the corresponding loss function, so to have a fair comparison with our Multi-KT. The original formulation considered the linear kernel, thus we chose \( K(x, z) = x^\top z \) for all the experiments together with the SIFT feature descriptors. In [11] the \( \beta \) value is defined by cross validation on extra validation target samples. Here we decided to simply tune it on the test set, showing the best result that could be obtained. The same approach was adopted to choose the number of boosting iterations for TrAdaBoost.

\(^5\) http://www.robots.ox.ac.uk/~vgg/software/tabularasa/
we consider one single source knowledge \( j \) at the time, the corresponding loss \( \ell_j(\tilde{y}_i, y_i) \) will depend on the difference 

\[
\ell_j(\tilde{y}_i, y_i) = \log(1 + \exp(-\alpha_i))
\]

where at each iteration the weak classifier is extracted from the target task and consists in selecting them by assigning to each a proper weight. To our knowledge, only our Multi-KT method is based on the third selective technique.

**MultiSourceTrAdaBoost: boosting by transferring samples.** An extension to the TrAdaBoost approach in the case of multiple available sources has been presented in [9]. The method MultiSourceTrAdaBoost considers one source set at the time, combining it with the target set and defining a candidate weak classifier. The final classifier is then chosen as the one producing the smallest training target classification error by automatically selecting the corresponding best source.

**TaskTrAdaBoost: boosting by transferring models.** This is a parameter transfer approach consisting of two steps. Phase I deploys traditional AdaBoost separately on each source task to get a collection of candidate weak classifiers. Only the most discriminative are stored by asking that the weight assigned by the boosting process to each classifier is greater than a certain threshold \( \tau \); this guarantees to avoid overfitting. Phase II is again an AdaBoost loop over the target training data where at each iteration the weak classifier is extracted from the set produced in the previous phase. The choice is done on the basis of the minimal classification error produced on the target training set.

**Single KT.** Our Multi-KT algorithm chooses the best set of weights for all the prior knowledge models at once on the basis of the loss function defined in (15). An alternative approach can be defined adopting a logistic loss function [49]:

\[
\ell(\tilde{y}_i, y_i) = \eta \frac{1}{1 + \exp(-10(\tilde{y}_i - y_i))}
\]

If we consider one single source knowledge \( j \) at the time, the corresponding loss \( \ell_j(\tilde{y}_i, y_i) \) will depend on the difference
Recall
unrelated
fire truck school bus
0.2 0.4 0.6 0.8 1
Recall
related

![Graphs showing recognition rate as a function of the number of positive training samples.](image)

**Fig. 6.** Left and middle columns: recognition rate as a function of the number of positive training samples. In each experiment we consider in turn one of the classes as target and the others as source, on ten random training sets. The final results are obtained as average over all the runs. Top Right: the histogram bars represent the recall produced by the model of the source class (indicated on the x-axis) when used to classify on the other class, considered as target. Bottom Right: average norm of the difference between two $\beta$ vectors obtained for a pair of subsequent training steps.

\[(\hat{y}_i - y_i) = \left( \frac{A_{\hat{y}}}{\lambda_i} - \beta_j \frac{A_{y}}{\lambda_j} \right) \text{ for } i = 1, \ldots, N. \]

Although this formulation results in a non convex objective function with respect to $\beta_j$, it is always possible to evaluate (22) for a finite set $S$ of weights. We can store for each source the value $\min_S \{ \sum_j \epsilon_j (\hat{y}_j, y_j) \}$, and then compare all the results to identify the best prior knowledge model and the best weight value to assign. We call this variant of our method Single-KT.

**Average Prior Knowledge.** As already mentioned in the introduction, the first knowledge transfer approach able to perform one-shot learning on computer vision problems was presented in [7]. This approach does not make any assumption on the reliability of the prior knowledge, which is always considered as an average over all the known classes. The algorithm structure is strictly related to the part-based model descriptors and neither the code nor the feature used for the experiments in [7] have ever been publicly released. However, following the proposed main idea, any transfer learning method that originally considers the existence of a single source task can be extended to the case of multiple sources by relying on the average of all the prior known models.

**Experiments.** Here we show a benchmark evaluation of our Multi-KT algorithm against its Single-KT version, MultiSourceTrAdaBoost and TaskTrAdaBoost. Following the basic idea of [7] we also use A-SVM as baseline supposing to consider the average of all the prior models as source knowledge, thus $\hat{w} = \frac{1}{2} \sum_{j=1}^N \hat{w}_j$ and $\beta = 1$.

We used the same experimental setting of the previous section considering the linear kernel, SIFT features and two randomly extracted sets of 10 and 20 classes from Caltech-256. In particular, the second set is obtained by adding an extra random group of 10 classes to the first one. For the boosting approaches all the parameters (number of boosting iterations and the threshold in Phase I of TaskTrAdaBoost) where tuned on the test set and only the best results are presented. From Figure 6 it is clear that in both the experiments our Multi-KT approach clearly outperforms Single-KT and the two boosting methods ($p \leq 0.01$), besides producing better results than learning from scratch ($p \leq 0.01$). Moreover, for very few samples, properly weighting each prior knowledge source with Multi-KT is better ($p \leq 0.05$) than averaging over all the known models as done by A-SVM: the two approaches are equivalent only after five positive training samples with 10 classes and respectively three positive training samples for 20 classes.

For any method that chooses only one source model in transferring, every time there is a change in the selected source, the behavior of the algorithm might change. This indicates low stability. Recent work has shown that the more stable is an algorithm, the better is its generalization ability [50]. The plot on the right in Figure 6 (bottom right) shows the comparison of Multi-KT with its Single-KT version in terms of stability. The best $\beta_j$ value chosen by Single-KT can be considered as an element of the full $\beta$ vector where all the remaining elements are zero. For each pair of subsequent steps in time, corresponding to a new added positive training sample, we calculate the difference between the obtained $\beta$ both for Multi-KT and Single-KT. From the average norm of these differences...
it is evident that choosing a combination of the prior known models for transfer learning is more stable than relying on just a single source (lower average variation in the vector $\beta$).

7.4 Heterogeneous Knowledge

In this section we consider an heterogeneous experimental setting where each source knowledge lives in its own feature space and we compare the performance of MultiK-KT with that of Multi-KT applied on a restricted homogeneous condition. We show that the space enlarging trick at the basis of MultiK-KT, not only allows to overcome the problem raised by the existence of a variability in the sources, but also, by exploiting this higher level of freedom, produces better results than Multi-KT in the corresponding single space case.

We ran experiments on the same subset of data used in the section 7.1. Here we considered SIFT as unique descriptor together with the generalized Gaussian kernel: $K(x, z) = \exp(-\gamma d_{\rho,\delta}(x, z))$, where $d_{\rho,\delta}(x, z) = \sum_i |x_i^p - z_i^p|^\delta$. Each source knowledge is defined by using the best set $\{\gamma, \rho, \delta\}$ obtained by cross validation on the corresponding object category, while we learn on the target class considering the sum over the source kernels. We name no transfer multiK the baseline corresponding to learning from scratch in this combined space. Figure 7 presents the obtained results in comparison with the case of using a single standard Gaussian kernel, with fixed $\gamma$ for sources and target tasks (no transfer and Multi-KT curves in the plot): MultiK-KT always performs significantly better than Multi-KT ($p \leq 0.002$).

Among the baseline methods that we considered in the homogeneous experiments, the only one that allows also the use of heterogeneous sources is TaskTrAdaBoost. We compare it with MultiK-KT over the random set of ten classes already used in the previous section. For each source we suppose to have already learned an SVM model with SIFT descriptors and Gaussian kernel where the $\gamma$ parameter is set to the mean of the pairwise distances among the samples. This means that each source model lives in its own specific feature space. TaskTrAdaBoost in each boosting iteration simply chooses one of the source models, while MultiK-KT learns the target task in the composed space defined by all the sources and obtained on the basis of the sum kernel. Figure 8 shows that multiK-KT outperforms TaskTrAdaBoost ($p \leq 0.01$) besides obtaining better results than learning from scratch.

7.5 Increasing Number of Sources

For any open-ended learning agent the number of known object categories is expected to grow in time. An increasing number of sources may give rise to a scalability problem in transfer learning due to the necessity of checking each of them and evaluate the reliability for the new task. Specifically, for $10^2$ source sets the boosting methods described in Section 7.3 become extremely expensive in computational terms (indeed the paper which presented them considered a maximum of 5 sources [9]).

We performed experiments with 100 and 256 object classes from Caltech-256 dataset, reporting the result of Multi-KT, no transfer and A-SVM with average prior knowledge in Figure 9. In both cases, properly choosing the weights to assign to each source pays off with respect to average over all the sources for very few training samples: Multi-KT outperforms A-SVM ($p \leq 0.05$) for less than three positive samples. With enough training samples and a rich prior knowledge set, the best choice is to not neglect any source information.

We can expect that with a growing prior knowledge set, also the probability to find a useful source for the target task increases. To verify this behavior we focus on the Multi-KT results obtained with a single positive image. The one-shot performance obtained in the previous experiments for 2 unrelated classes, 2 related classes, random sets of 10, 20, 100 classes plus the full final set of 256 objects are summarized in Figure 9 (right). Although some small oscillation due to the specific group of classes considered, it is clear that by increasing the number of available sources of one order of magnitude the one-shot recognition rate obtained with Multi-KT grows. After an evident gain obtained by passing from $10^0$ to $10^4$ classes, the difference becomes less evident from $10^4$ to $10^2$ classes.

7.6 Increasing Number of Samples

Transfer learning has its maximum effectiveness in the small sample scenario in comparison to learning from scratch. However, it is also interesting to evaluate the performance of a knowledge transfer approach when the number of available
Fig. 7. Performance of the MultiK-KT method in comparison with the single kernel Multi-KT formulation. The curves identified by no transfer and no transfer multiK corresponds respectively to learning from scratch by using only the Gaussian kernel or the combination of generalized Gaussian kernels. The results correspond to average recognition rate over the categories, considering each class out experiment repeated ten times.

Fig. 10. Top line: recognition rate as a function of the number of positive training samples. Each experiment is defined by considering in turn one of the classes as target and the others as sources. The final results are obtained as average over ten runs. Bottom line: Maximum value over the elements of the $\beta$ vector averaged over the classes and the splits.

training instances increases, thus checking its asymptotic behavior (see Figure 2).

We repeated the experiments on the full Caltech-256 dataset considering \{1, 5, 10, 30, 50\} positive training samples with a fixed set of 50 negative training samples. We also run analogous experiments on the AwA and IRMA dataset, considering respectively 60 (60) and 70 (70) positive (negative) training samples. For all the datasets the test set contains 60 (30 positive and 30 negative) instances.

All the results are reported in Figure 10 (top line). Although it is clear the gain of Multi-KT with respect to learning from scratch for limited available data, in general this advantage disappears when the number of positive training samples reaches 50. Figure 10 (bottom line) indicates that the weights associated to prior knowledge progressively decrease. The absence of the asymptotic advantage was to be expected for Multi-KT and can be justified in theoretical terms: when the number of training samples increases, the adaptive regularization loses its relevance and the problem reduces to learning from scratch.

8 CONCLUSION

A learning system able to exploit prior knowledge when learning something new should rely only on the available target information for choosing from where and how much to transfer. To be autonomous it should not need an external teacher providing either information on which is the best source to use, or extra target training samples. In this paper we presented our Multi-KT algorithm, a LS-SVM based transfer learning approach with a principled technique to rely on source models and avoid negative transfer. The results of extensive experiments demonstrated the effectiveness of Multi-KT for object categorization problems with respect to other existing transfer learning methods. Moreover the weight assigned to the source knowledge set proved to be meaningful in terms of the semantic relation among the considered classes. We also extended our algorithm to the heterogeneous setting.

Recently the computer vision literature has seen an increasing interest towards high scale (10^5) object problems [5]. Most of the proposed transfer learning algorithms in this setting has been developed for object detection [51] and segmentation [52], while how to scale up the classification problem is still an open issue. Introducing a structure on the source knowledge while learning something new might be a promising strategy to use Multi-KT in this condition. Moreover the associated scalability problem due to the increasing number of training examples can be overcome by casting Multi-KT in an online learning framework [53]. All this clearly indicates possible directions for future research.

REFERENCES


