Iterative Self-labeling Domain Adaptation for Linear Structured Image Classification*†

Amaury Habrard, Jean-Philippe Peyrache and Marc Sebban
University of Saint-Etienne
Laboratoire Hubert Curien
UMR CNRS 5516
42000 Saint-Etienne Cedex 2 - France
{amaury.habrard,jean.philippe.peyrache,marc.sebban}@univ-st-etienne.fr

A strong assumption to derive generalization guarantees in the standard PAC framework is that training (or source) data and test (or target) data are drawn according to the same distribution. Because of the presence of possibly outdated data in the training set, or the use of biased collections, this assumption is often violated in real-world applications leading to different source and target distributions. To go around this problem, a new research area known as *Domain Adaptation* (DA) has recently been introduced giving rise to many adaptation algorithms and theoretical results in the form of generalization bounds. This paper deals with self-labeling DA whose goal is to iteratively incorporate semi-labeled target data in the learning set to progressively adapt the classifier from the source to the target domain. The contribution of this work is three-fold: First, we provide the minimum and necessary theoretical conditions for a self-labeling DA algorithm to perform an actual domain adaptation. Second, following these theoretical recommendations, we design a new iterative DA algorithm, called GESIDA, able to deal with structured data. This algorithm makes use of the new theory of learning with (ϵ,γ,τ)-good similarity functions introduced by Balcan et al., which does not require the use of a valid kernel to learn well and allows us to induce sparse models. Finally, we apply our algorithm on a structured image classification task and show that self-labeling domain adaptation is a new original way to deal with scaling and rotation problems.

Keywords: Domain Adaptation, Edit Distance, Sparse Learning

1. Introduction

Supervised learning algorithms aim at inferring from labeled *source* examples a classifier that offers good generalization guarantees on independent unlabeled *target* data. In spam detection, for example, the goal is to learn from a collection of mails the way to discriminate between spams and hams. Once learned, the induced model can then be plugged into mailers to classify new mails. It is worth noting that to be able to derive theoretical guarantees (*e.g.*, as in the PAC framework [1]), most

*This article is an extended version of our paper presented at the 23rd IEEE International Conference on Tools with Articial Intelligence (ICTAI 2011).

†Electronic version of an article published as International Journal on Artificial Intelligence Tools Vol. 22, No. 5, 30 pages, DOI: 10.1142/S0218213013600051, © copyright World Scientific Publishing Company, www.worldscinet.com/ijait

machine learning methods have to assume that the *target* examples on which the classifier is deployed are drawn from the same (unknown) distribution as the *source* data this classifier has been trained from. Unfortunately, this assumption does not hold in many real-life applications, such as in natural language processing [2,3] or in computer vision [4]. Such situations may occur when source data are outdated or not representative enough of the underlying distribution (*e.g.* because of a bias in the collection). Coming back to spam detection, let us suppose we learn a model from a collection of labeled (spam/ham) mails randomly drawn from some users' mailboxes. If this classifier is applied on a new user mailbox, we may not be able to ensure theoretical guarantees on its consistency. Indeed, because of possibly different mailing lists subscribed by the new user, or a different notion of what is a spam (which may depend on user' job), the underlying distribution of the new target mails is likely to be different from that of the training source corpus.

In this context, recent works have been devoted to go around this kind of problems, leading to a new research area: *domain adaptation* (DA). Roughly speaking, DA aims at making use during the learning process of information coming from both source and target domains to allow an automatic adaptation. DA issues have been addressed in two different ways in the literature: (i) the supervised case where one has a large set of labeled source examples and a small set of labeled target points; (ii) the unsupervised case for which, in addition to labeled source data, we have a large amount of target examples, all unlabeled. In this paper, we mainly focus on this (clearly more difficult) second case.

In the past few years, DA has been widely studied in the literature from a statistical learning theory point of view. The objective is to determine under what theoretical conditions a classifier learned on the source domain can be adapted for application on the target domain. The pioneer work presented in [5] provides a generalization bound on the target error (which generalizes classic PAC bounds) which basically depends on the source error and a measure of divergence (called $\mathcal{H}\Delta\mathcal{H}$-*divergence*) between the source and target distributions. An extension of this work introducing a *discrepancy distance* is proposed in [6]. This new divergence measure is based on the Rademacher complexity allowing the use of arbitrary loss functions leading to generalization bounds for, *e.g.*, Support Vector Machine (SVM) or regression methods. In [7], the authors derive a generalization bound on the target error using the notion of robustness.

Taking their inspiration from these theoretical recommendations, many DA algorithms have been introduced in the literature. When the training source data are statistically not representative of the target domain, some of these methods aim at reducing this *sample selection bias* by resorting to data resampling schemes (*e.g.* see [8,9,10,11,12]). Some others select the subset of features shared by both domains, or learn new latent features to represent source and target data in a new low-dimensional space, like in [13,14,15,16]. On the other hand, *self-labeling iterative* DA algorithms use step by step a part of the (unlabeled) target data, label them with an hypothesis built from the current source examples, and learn a new classi-

fier with this self-labeled information [17,18].

In this paper, we focus on this last category of DA methods and present three main contributions:

The first one takes the form of a **theoretical study** on the minimum and necessary conditions for an iterative DA algorithm to perform an actual domain adaptation. This part gives us useful information about the strategy to efficiently select the target points in a self-labeling approach.

Following these theoretical requirements, we provide an **algorithmic contribution** via a new iterative DA algorithm able to handle **structured data**. It is worth noting that dealing with strings, trees or graphs in DA has not received a lot of attention in the literature. Indeed, extending state of the art DA algorithms to structured data is rarely straightforward. This can be explained by the fact that it is definitely more difficult to (i) define divergence measures between structured data distributions, (ii) design and compute similarities between strings, trees of graphs, or (iii) find low-dimensional spaces for such data. In this paper, we introduce a new self-labeling DA algorithm which deals with structured data and is inspired from the efficient iterative DASVM approach [17]. DASVM iteratively replaces source data by *semi-labeled* target examples to progressively modify an SVM classifier. A naive approach to deal with structured data would consist in simply using in DASVM a *valid* edit kernel (based on the well known *edit distance* [19]), *i.e.* an edit similarity function which fulfills the conditions of positive semi-definiteness (PSD) and symmetry. However, it has been proven in [20] that most of the similarity functions based on the *edit distance* are not PSD. To go around this problem, a standard approach consists in plugging the edit distance e_d into a Gaussian-like kernel, such that $K(x, x') = e^{-t.e_d(x,x')}$ and tuning the meta-parameter t by cross-validation which would likely ensure the resulting kernel to be valid. Despite the fact that a slight modification of t can lead to dramatic changes in performance, the use of SVMs in such an iterative DA process leads to a costly learning algorithm. To overcome these problems, we suggest in this paper to relax the kernel constraint by making use of the new theory introduced by Balcan et al. in [21,22] about learning with non-PSD similarity functions. The authors prove that if a similarity function is (ϵ, γ, τ)-*good*, then it can be used to build a linear separator [a] that has margin γ and error arbitrarily close to ϵ. Interestingly, this separator can be trained using a linear program and is supposedly sparse by making use of the L_1-norm. In this paper, we exploit this framework to design a new theoretically founded iterative DA algorithm.

[a]This separator is learned in an explicit space where similarity scores to so-called *reasonable points* are used as features. τ represents the minimal proportion of reasonable points (see Section 2.2 or [22] for more details).

Our last contribution consists in applying our new DA algorithm in the context of **image classification**. While a usual way in this research area resorts to numerical representations of so-called bag-of-visual-words [23], such feature vectors do not allow the integration of additional structural information or topological relationships between the objects of the images. An efficient alternative consists in representing images in the form of strings, trees, or graphs. The pioneer work has been proposed by Freeman [24] where binary objects are encoded in the form of sequences of symbols built from an alphabet of eight directions. In [25], the authors propose to map each contour of the objects into a string whose components are pairs of symbols (the angle between the contour point and its neighbors and the normalized distance from the center of mass). On the other hand, in [26,27], in order to code the topological relationship between iconic objects, a 2D string is used to represent the relative location of the visual words in the original 2D-space to build a graph of similarities. In this last contribution, we apply our new DA algorithm on a handwritten character recognition task, focusing on scaling and rotation problems, where images are represented in the form of strings or trees.

The rest of this paper is organized as follows: we review in Section 2 some related work in DA and briefly introduce the theory of (ϵ, γ, τ)-good similarities. Section 3 presents a theoretical analysis on some necessary conditions to ensure a good domain adaptation in an iterative unsupervised DA process. Then, we introduce in Section 4 our novel DA algorithm which is based on good edit similarities and tends to satisfy the theoretical requirements stated in the previous section. We present in Section 5 the experimental results obtained on a handwritten character recognition task, focusing on scaling and rotation problems. To show an experimental evidence of the efficiency of our algorithm, we use not only a string but also a tree-structured representation of the images. We conclude this paper in Section 6 opening the door to new promising lines of research.

2. Related Work

As defined by Mansour in [28], the underlying problem of domain adaptation (DA) can be specified as follows: "*DA is a fundamental learning problem where one wishes to use labeled data from one or several source domains to learn a hypothesis performing well on a different,* ***yet related****, domain for which no labeled data is available*".

Throughout this paper, we will consider the following binary problem: we have a training set $L_S = \{(x_i^S, y_i^S)\}_{i=1}^N$ of N labeled examples generated from a *source* distribution (or domain) D_S over $X \times \{-1, 1\}$, where X is the instance space. We want to learn a classifier $h \in \mathcal{H} : X \to \{-1, 1\}$ from L_S whose error rate ϵ_T is as low as possible over a *target* distribution D_T. To adapt h from D_S to D_T, along with L_S we have a set $L_T = \{x_j\}_{j=1}^T$ of T unlabeled examples drawn from the marginal

distribution of D_T. The (unknown) generalization source error of h is defined as follows:

$$\epsilon_S(h) = \mathbb{E}_{(x,y)\sim D_S}[h(x) \neq y],$$

while the (unknown) generalization target error of h is

$$\epsilon_T(h) = \mathbb{E}_{(x,y)\sim D_T}[h(x) \neq y],$$

where [.] is an indicator function.

2.1. *Related Work in Domain Adaptation*

In the past few years, DA has been widely studied from a statistical learning theory point of view. Typically, these theoretical results take the form of upper bounds on the generalization target error of h, which has been learned from the source data. To assess the adaptation difficulty of a given problem at hand, Ben-David et al. introduce in [29] the $\mathcal{H}\Delta\mathcal{H}$-divergence, noted $d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T)$, which is a measure between the source and the target distributions w.r.t. $\mathcal{H}$. To estimate this divergence, the authors suggest to label each source example of L_S with -1 and each unlabeled example of L_T as 1 and train a classifier to discriminate between source and target data. The $\mathcal{H}\Delta\mathcal{H}$-divergence is immediately computable from the error of the classifier and a term of complexity which depends on the Vapnik-Chervonenkis dimension (VC-dim) of the hypothesis space. Intuitively, if the error is low, we are thus able to easily separate source and target data that means that the two distributions are quite different. On the other hand, the higher the error, the less different D_S and D_T. The authors provide a generalization bound for domain adaptation on $\epsilon_T(h)$ which generalizes the standard bound on $\epsilon_S(h)$ by taking into account the $\mathcal{H}\Delta\mathcal{H}$-divergence between the source and domain distributions. More formally, we get:

$$\epsilon_T(h) \leq \epsilon_S(h) + d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) + \lambda, \tag{1}$$

where λ is the error of the ideal joint hypothesis on the source and target domains (which is supposed to be a negligible term if the adaptation is possible). Equation 1 expresses the idea that to adapt well, the DA algorithm has to learn an hypothesis h which works well on the source data while reducing the divergence between D_S and D_T.
Based on this work, Mansour et al. introduce a new divergence measure in [6], called *discrepancy distance*. Its empirical estimate is based on the Rademacher complexity (rather than the VC-dim) allowing the use of arbitrary loss functions leading to generalization bounds useful for more general learning problems, such as in classification with SVMs or in regression.

Following the underlying ideas of Equation 1, many DA algorithms have been proposed in the literature in different settings. We present in the following some of

these approaches (for more details, the interested reader may refer to the following surveys [30,31,32]).

Instance Weighting When the training examples, drawn in a biased manner, are not representative enough of D_T, we have to face a problem of *sample selection bias*. *Covariate shift* describes the sample selection bias scenario where the prior distributions $P_{D_S}(x)$ and $P_{D_T}(x)$ differ but the conditional probabilities are the same, *i.e.* $P_{D_S}(y|x) = P_{D_T}(y|x)$. Assuming that a certain mass of the source data can be used for learning the target examples, the domain adaptation can be done by resorting to a reweighting scheme of the data. Some solutions have been proposed (*e.g.*, see [8,9]) that use the ratio of the test and training input densities. Roughly speaking, the idea is to increase the importance of source points located into a region where there is a high density of target points. A learning process is then launched on this reweighted training set, which is supposed to be much closer from the target distribution than the unweighted one. In [10], the sample selection bias is corrected by resorting to an entropy maximization, while in [12] the source data are reweighted while minimizing the Kullback-Leibler-divergence between the source and target distributions. Finally, in [11], a classifier is found as the solution of an optimization problem integrating the covariate shift problem.

New feature representations Rather than reweighting the source data, another way to handle DA problems consists in changing the feature representation of the data to better describe shared characteristics of the two domains D_S and D_T. We distinguish two different strategies: the first one assumes that some features are generalizable, and thus they can be shared by both domains, while some others are domain-specific. In this context, a feature selection algorithm which penalizes or removes some features can be used to find the shared low-dimensional space. In [15], the authors suggest to train a model maximizing the likelihood over the training sample on a subset of the original features, minimizing the distance between the two distributions. The second strategy aims at learning new latent features. For example, Florian et al. proposed in [33] an algorithm that builds a model on the source domain and uses the predicted labels as new features for both source and target data. In [13], the authors introduce the notion of *Structural Correspondence Learning* (SCL). They identify the correspondence between features of the two domains by modeling their correlations. In this case, the principle is to project the examples from both domains into the same low-dimensional space, obtained by a mapping of the original features as done for example with a Principal Component Analysis (PCA). In [16], the authors use SCL in a Multi-View way. They first choose m bridge features, as does SCL and then learn for each domain, a correspondence between these bridge features and the other features. Finally, a Multi-View Principal Component Analysis is applied to learn a low-dimensional space. In [34], a mapping is learned from the original n-dimensional space to a new $3n$-dimensional space,

where the first n features are shared by both domains, and the last $n+n$ features are specific to the source and target domains respectively. Note that in this work, the author makes use of a small but labeled set of target data.

Iterative Self labeling In order to take advantage of the available unlabeled target data during the learning process, *iterative self-labeling* methods use and label some of them with an hypothesis built from source examples. Such target data are usually called semi-labeled examples. Then, a new classifier is trained taking into account these semi-labeled data, thus considering information from both domains. The main difficulty is to define the way of choosing the considered target points (and the appropriate proportion). Intuitively, a good strategy would retain the semi-labeled target data for which we have the greater confidence in their classification. In [18], Perez et al. introduce a two-step algorithm: first, a statistical model is learned from the source domain. Then, using an extension of the EM algorithm, they estimate how these source parameters change in the target set. The assumption is that large changes are less likely to occur than small ones. Finally, they use this re-estimated model to learn a classifier for the target domain. Other works have been proposed, based on co-training or self-training methods [35].

Our new iterative algorithm is inspired from DASVM, introduced in [17], and which was shown to be very competitive. For this reason, let us go into detail of this method whose idea is to iteratively incorporate target samples from L_T into the learning set to progressively modify an SVM classifier h. DASVM changes $2k$ original source data by $2k$ newly labeled examples from L_T. More precisely, on each side of the separator, the k examples falling into the margin band that are the closest to the margin bounds are selected. These $2k$ semi-labeled examples take the place of the $2k$ original source examples the farthest from the hyperplane. Then, a new classifier is learned from the updated training set L_S. This process is repeated iteratively. The bound minimization problem solved at each iteration is the following:

$$\forall l = 1, \ldots, \mu^{(i)}, (x_l^S, y_l^S) \in L_S^{(i)}, \forall u = 1, \ldots, \eta^{(i)}, (x_u^T, \hat{y}_u^{T(i-1)}) \in L_S^{(i)}, \text{ as:}$$

$$\begin{cases} \min\limits_{w,b,\xi^S,\xi^T} \{\frac{1}{2}||w^{(i)}||^2 + C^{(i)} \sum\limits_l \xi_l^S + \sum\limits_u C_u^\star \xi_u^T\} \\ y_l^S \cdot (w^{(i)} \cdot x_l^S + b^{(i)}) \geq 1 - \xi_l^S \\ \hat{y}_u^{(i-1)} \cdot (w^{(i)} \cdot x_u^T + b^{(i)}) \geq 1 - \xi_u^T \\ \xi_l^S, \xi_u^T \geq 0, \end{cases}$$

where $h^{(i)}(x) = w^{(i)} \cdot x + b^{(i)}$, $\mu^{(i)}$ is the number of remaining original source examples, $\eta^{(i)}$ the number of semi-labeled examples already inserted into $L_S^{(i)}$ (such that $|L_S^{(i)}| = \mu^{(i)} + \eta^{(i)}$). Note that $C_u^\star$ and $C^{(i)}$ are regularization parameters for semi-labeled and original source domain examples respectively. They aim to control the number of misclassified samples of $L_S^{(i)}$. Increasing their values boils down to

increasing the penalty associated with errors. In addition, if a semi-labeled example inserted in $L_S^{(i)}$ gets, at the ith iteration, a predicted label different from the one previously assigned, the sample is put back into L_T. This allows one to increase the weight of the examples keeping iteratively the same label. In the same time, the weight of the original samples from L_S decreases with the iterations, in order to give more importance to the target samples.

Note that although this method has shown to be efficient, it needs an expensive computation because of the quadratic complexity of the bound minimization problem which requires to be done several times. Moreover, as our objective in this paper is to directly exploit string and tree-structured data, using DASVM would lead to two major problems: First, the algorithmic constraints of DASVM are compounded by the (more expensive) calculation of similarities between strings (typically in $\mathcal{O}(n \times m)$) or trees (typically in $\mathcal{O}(n^2 \times m^2)$ where n and m are the sizes of the data; second, adapting SVMs to string or tree similarity functions is not straightforward. Indeed, to be a valid kernel, the similarity function involved in the learning process must be positive semi-definite (PSD) and symmetric. However, it has been proven in [20] that the edit distance, which is the most commonly used similarity measure between strings or trees, is not PSD.

To go around this problem, we suggest in the following to make use of the recent theory of learning with (ϵ, γ, τ)-*good similarity functions*, introduced in [21,22], which does not need to have PSD functions to learn well with generalization guarantees. Interestingly, this theory provides a much sparser alternative to SVM theory. Let us briefly review Balcan et al's framework in the next section.

2.2. *Learning with Good Similarity Functions*

The theory of Balcan et al. [21,22] on learning with *good similarity functions* is a strict generalization of kernel methods. Indeed, a classical kernel is a good similarity function, but the theory also holds for similarity functions that are neither PSD nor symmetric that allows us to address problems that can not be solved with classical kernels. The intuitive idea is that any similarity function is *good* if it has a given power of discrimination. More formally, Balcan et al. propose the following definition.

Definition 2.1. A similarity function K is an (ϵ, γ, τ)-good similarity function for a learning problem P if there exists a (random) indicator function $R(x)$ defining a (probabilistic) set of "reasonable points" such that the following conditions hold:

(i) A $1 - \epsilon$ probability mass of examples (x, ℓ) satisfy

$$\mathbb{E}_{(x',\ell')}\ P[\ell\ell' K(x, x')|R(x')] \geq \gamma$$

(ii) $Pr_{x'}[R(x')] \geq \tau$.

The first condition can be interpreted as a large proportion of examples must be more similar to random reasonable examples of the same class than to random reasonable examples of the opposite class. Moreover, from the second condition, at least a τ proportion of the examples should be reasonable. Balcan et al. have proved that using this kind of similarities allows one to learn a good linear classification in a projection space corresponding to similarities to a set of reasonable points R. This is formalized by the following PAC-like guarantee.

Theorem 2.1. Let K be an (ϵ, γ, τ)-good similarity function for a learning problem D. Let $L = \{x'_1, x'_2, \ldots, x'_d\}$ be a (potentially unlabeled) sample of $d = \frac{2}{\tau}\left(\log(2/\delta) + 8\frac{\log(2/\delta)}{\gamma^2}\right)$ landmarks drawn from D. Consider the mapping $\phi^S : X \to \mathbb{R}^d$ defined as follows: $\phi^S_i(x) = K(x, x'_i)$, $i \in \{1, \ldots, d\}$. Then, with probability at least $1 - \delta$ over the random sample L, the induced distribution $\phi^S(D)$ in $\mathbb{R}^d$ has a linear separator of error at most $\epsilon + \delta$ relative to L_1 margin at least $\gamma/2$.

Thus, given an (ϵ, γ, τ)-good similarity, we can learn with high probability a linear separator in the explicit ϕ-space. Balcan et al. have proposed a learning formulation, using d_u unlabeled examples and d_l labeled examples, to efficiently learn this separator $\alpha \in \mathbb{R}^{d_u}$ corresponding to solve the following problem.

$$\min_{\alpha} \sum_{i=1}^{d_l} \left[1 - \sum_{j=1}^{d_u} \alpha_j \ell_i K(x_i, x'_j)\right]_+ \quad \text{s.t.} \sum_{j=1}^{d_u} |\alpha_j| \leq 1/\gamma, \tag{2}$$

where $[1 - z]_+ = \max(0, 1 - z)$ is the hinge-loss.

Since, in this paper, we deal with structured data, we need to define a similarity adapted to this particular type of data. Among all the existing similarity measures, the edit distance [19] - also called the Levenshtein distance - is one of the most popular distance for strings.

Definition 2.2. The Levenshtein distance $e_L(x, x')$ between two strings x and x' (of length m and n respectively) is the minimum number of edit operations to transform x into x'. The allowable operations are insertion, deletion and substitution of a single symbol.

e_L can be computed in $O(m \times n)$ using dynamic programming techniques. When using this distance in binary SVM-based classification problems, a common choice is to plug e_L in a Gaussian-like kernel such as $K(x, x') = \exp^{-t \times e_L(x,x')}$ where $t > 0$ is a parameter. However, it has been shown that this kind of similarity is not a valid kernel in general [20].

In our case, following the framework of Balcan et al., we propose to directly use the similarity $K(x, x') = -e_L(x, x')$ as a good similarity function, renormalized in $[-1, 1]$. We will show in the experimental section that this similarity actually has some goodness properties that fit in the Balcan's framework. Thus, we propose to use it to learn linear classifiers in an explicit projection space (rather than in an implicit

one as in SVM-based approaches) made up of similarities to reasonable examples. Since this set is a priori unknown, we use the whole learning set L_S as landmark points. This allows us to learn a linear classifier of the form $h(\cdot) = \sum_{x_i \in L_S} \alpha_i K(\cdot, x_i)$ to perform an iterative adaption approach following the principle of DASVM. Before presenting our new iterative DA algorithm, we present in the following section a theoretical study where we derive some necessary conditions for an iterative DA algorithm to perform an actual domain adaptation.

3. Theoretical Analysis

An iterative DA algorithm successively incorporates at each time i in the training set $L_S^{(i)}$ some semi-labeled target data drawn from the target set L_T in order to adapt the current classifier to the target concept. Since we do not have access to the true label of such target examples, the choice of these points and the design of the DA algorithm need a special care to prevent from divergence phenomena due to possibly successive mislabellings. In this section, we study some minimum and necessary conditions for such iterative DA algorithms to perform an actual domain adaptation. This theoretical study is done in the baseline case of a random selection of the semi-labeled target examples. Therefore, the goal of a relevant iterative DA algorithm will consist in satisfying at least this minimum requirement. Inspired from boosting, we first introduce the notion of weak DA assumption. Afterwards, we provide the condition for an hypothesis learned from some semi-labeled data to be a weak DA hypothesis (Theorem 3.1). Then, we prove via Theorem 3.2 the necessary conditions for an iterative DA algorithm which successively infers a set of weak DA hypotheses to perform an actual adaptation. Finally, we provide some insights about the meaning of the DA conditions.

Let us start by recalling the concept of *weak learner* presented in [36].

Definition 3.1. A classifier $h^{(i)}$ learned at time i from the current training set $L_S^{(i)}$ is a weak learner if its empirical error $\hat{\epsilon}_S^{(i)}(h^{(i)})$ over $L_S^{(i)}$ is smaller than 0.5, *i.e.*

$$\hat{\epsilon}_S^{(i)}(h^{(i)}) = \mathbb{E}_{x_l^S \in L_S^{(i)}}[h^{(i)}(x_l^S) \neq y_l^S] = \frac{1}{2} - \gamma_S^{(i)},$$

where $\gamma_S^{(i)} > 0$ measures how much better than random are $h^{(i)}$ predictions.

Extending this definition to DA, we define the notion of *weak DA learner* w.r.t. $2k$ semi-labeled target data.

Definition 3.2. A semi-labeled target point inserted in $L_S^{(i)}$ at step i in place of a source point is an example randomly drawn from L_T (without replacement) labeled by the hypothesis $h^{(i-1)}$ learned from the previous training set $L_S^{(i-1)}$.

Definition 3.3. A classifier $h^{(i)}$ learned at step i from the current training set $L_S^{(i)}$ is a weak DA learner with respect to a set $SL_j = \{x_1^T \dots x_{2k}^T\}$ of $2k$ semi-labeled

target data inserted at step j if its empirical error $\hat{\epsilon}^{(j)}_{S_{2k}}(h^{(i)})$ over these $2k$ points - according to their true (unknown) label - is smaller than 0.5, *i.e.*

$$\hat{\epsilon}^{(j)}_{S_{2k}}(h^{(i)}) = \mathbb{E}_{x^T_l \in SL_j}[h^{(i)}(x^T_l) \neq y^T_l] < \frac{1}{2}.$$

We now give a first necessary condition on the error of a weak DA learner.

Theorem 3.1. Let $h^{(i)}$ be a weak hypothesis learned at step i from $L^{(i)}_S$ and $\tilde{\epsilon}^{(i)}_S(h^{(i)}) = \frac{1}{2} - \gamma^{(i)}_S$ its corresponding empirical error[b]. Let $\hat{\epsilon}^{(i)}_T(h^{(i)}) = \frac{1}{2} - \gamma^{(i)}_T$ be the (unknown) empirical error of $h^{(i)}$ over the target sample L_T. $h^{(i)}$ is a weak DA learner w.r.t. a set $SL_j = \{x^T_1 \dots x^T_{2k}\}$ of $2k$ semi-labeled target data inserted at step j ($j \leq i$) if $\gamma^{(j-1)}_T > 0$.

Proof. A semi-labeled point inserted at time j is mislabeled by $h^{(i)}$ either if it has a wrong semi-label (given by $h^{(j-1)}$ with probability $(\frac{1}{2} - \gamma^{(j-1)}_T)$) and has been correctly classified by $h^{(i)}$ (with probability $(\frac{1}{2} + \gamma^{(i)}_S)$) or if it has a correct semi-label (given by $h^{(j-1)}$ with probability $(\frac{1}{2} + \gamma^{(j-1)}_T)$) and has been misclassified by $h^{(i)}$ (with probability $(\frac{1}{2} - \gamma^{(i)}_S)$). Following Definition 3.3, $h^{(i)}$ is a weak DA learner w.r.t. a set $SL_j = \{x^T_1 \dots x^T_{2k}\}$ of $2k$ semi-labeled target data inserted at step j if

$$\begin{aligned}
&\hat{\epsilon}^{(j)}_{S_{2k}}(h^{(i)}) = \mathbb{E}_{x^T_l \in SL_j}[h^{(i)}(x^T_l) \neq y^T_l] < \frac{1}{2} \\
\Leftrightarrow\ &(\frac{1}{2} + \gamma^{(i)}_S)(\frac{1}{2} - \gamma^{(j-1)}_T) + (\frac{1}{2} - \gamma^{(i)}_S)(\frac{1}{2} + \gamma^{(j-1)}_T) < \frac{1}{2} \\
&\Leftrightarrow \gamma^{(i)}_S \gamma^{(j-1)}_T > 0 \\
&\Leftrightarrow \gamma^{(j-1)}_T > 0 \text{ since } \gamma^{(i)}_S > 0 \text{ by the weak asumption.}
\end{aligned}$$

□

Theorem 3.1 is simple to interpret. It means that $h^{(i)}$ will be able to correctly classify (w.r.t. their unknown true label) more than k semi-labeled target examples among $2k$ if at least half of them have been correctly semi-labeled.

In the following, we prove the necessary conditions for an iterative DA algorithm which infers weak DA hypotheses to perform an actual adaptation.

Theorem 3.2. Let $L^{(0)}_S$ be the original learning set made of N labeled source data and L_T a set of $T \geq N$ unlabeled target examples. Let $\mathcal{A}$ be an iterative DA algorithm which randomly changes, at each step i, $2k$ original source labeled points from $L^{(i)}_S$ by $2k$ semi-labeled target examples randomly drawn from L_T without replacement and infers at each step a weak DA hypothesis (according to Definition 3.3).

[b]In this self-labeling DA context, we use the tilde symbol rather than the usual hat symbol because for the examples of $L^{(i)}_S$ coming from L_T, we have only access to their semi-label which can be wrong as expressed in the proof.

Let $h^{(\frac{N}{2k})}$ be the weak DA hypothesis learned by $\mathcal{A}$ after $\frac{N}{2k}$ such iterations needed to change $L_S^{(0)}$ into a new learning set made of only target examples. Algorithm $\mathcal{A}$ performs an actual domain adaptation with $h^{(\frac{N}{2k})}$ if

$$\gamma_S^{(i)} \geq \gamma_T^{(i)}, \forall i = 1 \dots \frac{N}{2k}, \tag{3}$$

$$\gamma_S^{max} > \sqrt{\frac{\gamma_T^{(0)}}{2}}, \tag{4}$$

where $\gamma_S^{max} = max(\gamma_S^{(0)}, \dots, \gamma_S^{(n)})$.

Proof. $h^{(\frac{N}{2k})}$ performs an actual domain adaptation iff $\hat{\epsilon}_S^{(\frac{N}{2k})}(h^{(\frac{N}{2k})}) < \hat{\epsilon}_T^{(0)}(h^{(0)})$. Roughly speaking, this condition simply means that the hypothesis $h^{(\frac{N}{2k})}$ learned only from semi-labeled target examples must outperform $h^{(0)}$ which have been learned only from source labeled data. Let $\hat{\epsilon}_{S_{2k}}^{(j)}(h^{(i)})$ be the error made by $h^{(i)}$ over the $2k$ semi-labeled examples inserted in the learning set at step j.

$$\hat{\epsilon}_{S_{2k}}^{(j)}(h^{(i)}) = (\frac{1}{2} + \gamma_S^{(i)})(\frac{1}{2} - \gamma_T^{(j-1)}) + (\frac{1}{2} - \gamma_S^{(i)})(\frac{1}{2} + \gamma_T^{(j-1)}) = \frac{1}{2} - 2\gamma_S^{(i)}\gamma_T^{(j-1)}.$$

We deduce that:

$$\hat{\epsilon}_S^{(\frac{N}{2k})}(h^{(\frac{N}{2k})}) < \hat{\epsilon}_T^{(0)}(h^{(0)})$$

$$\Leftrightarrow \frac{1}{N}\sum_{j=1}^{\frac{N}{2k}} 2k(\frac{1}{2} - 2\gamma_S^{(\frac{N}{2k})}\gamma_T^{(j-1)}) < \frac{1}{2} - \gamma_T^{(0)}$$

$$\Leftrightarrow \frac{4k}{N}\gamma_S^{(\frac{N}{2k})}\sum_{j=1}^{\frac{N}{2k}} \gamma_T^{(j-1)} > \gamma_T^{(0)}$$

$$\Leftrightarrow \frac{4k}{N}\gamma_S^{(\frac{N}{2k})}\sum_{j=1}^{\frac{N}{2k}} \gamma_S^{(j-1)} > \gamma_T^{(0)}, \text{ because of Condition (3).} \tag{5}$$

$$\Leftrightarrow \frac{4k}{N}\frac{N}{2k}(\gamma_S^{max})^2 > \gamma_T^{(0)}, \text{ where } \gamma_S^{max} = max(\gamma_S^{(0)}, \dots, \gamma_S^{(n)}) \tag{6}$$

$$\Leftrightarrow \gamma_S^{max} > \sqrt{\frac{\gamma_T^{(0)}}{2}}.$$

□

Before analyzing this theorem, let us interpret via Theorem 3.3 the meaning of Condition (3).

Theorem 3.3. If Condition (3) of Theorem 3.2 is verified, this means that $\forall i$

$$\tilde{\epsilon}_S^{(i+1)}(h^{(i)}) > \tilde{\epsilon}_s^{(i)}(h^{(i)}).$$

Proof.

$$\begin{aligned}
\gamma_S^{(i)} &\geq \gamma_T^{(i)} \\
\Leftrightarrow \frac{2k}{N}\gamma_S^{(i)} &\geq \frac{2k}{N}\gamma_T^{(i)} \\
\Leftrightarrow (\frac{1}{2} - \gamma_S^{(i)}) + \frac{2k}{N}\gamma_S^{(i)} &\geq (\frac{1}{2} - \gamma_S^{(i)}) + \frac{2k}{N}\gamma_T^{(i)} \\
\Leftrightarrow (\frac{1}{2} - \gamma_S^{(i)}) + \frac{2k}{N}\gamma_S^{(i)} - \frac{2k}{N}\gamma_T^{(i)} &\geq \frac{1}{2} - \gamma_S^{(i)} \\
\Leftrightarrow (1 - \frac{2k}{N})(\frac{1}{2} - \gamma_S^{(i)}) + \frac{2k}{N}(\frac{1}{2} - \gamma_T^{(i)}) &\geq \frac{1}{2} - \gamma_S^{(i)} \\
\Leftrightarrow \tilde{\epsilon}_S^{(i+1)}(h^{(i)}) &\geq \tilde{\epsilon}_s^{(i)}(h^{(i)}),
\end{aligned}$$

where $\tilde{\epsilon}_S^{(i+1)}(h^{(i)})$ is the empirical error (according to the - possibly wrong - labels at the disposal of the learner) of the old classifier $h^{(i)}$ over the new training set $L_S^{(i+1)}$ obtained once $2k$ new target semi-labeled data have been inserted. □

As in boosting, one can interpret Theorem (3.3) as the classifier built from $L_S^{(i+1)}$ must learn something new about the target distribution D_T. Note that in Adaboost, the constraint is rather $\hat{\epsilon}_S^{(i+1)}(h^{(i)}) = 0.5$, which is sufficient, because Adaboost performs a linear combination of all the learned weak classifiers. Since iterative DA algorithms only keep the last induced hypothesis, the condition $\hat{\epsilon}_S^{(i+1)}(h^{(i)}) = 0.5$ is not sufficient. To sum up, by constraining the old classifier $h^{(i)}$ to work well on the new training set $L_S^{(i+1)}$, Theorem (3.3) means the DA algorithm has to infer a new hypothesis able to learn something new about D_T.

However, Condition (3) is not sufficient to perform an actual adaptation. In Theorem 3.2, Condition (4) expresses the idea that, in addition to Condition (3), at least one hypothesis has to significantly outperform the behavior of the classifier $h^{(0)}$ (learned only from source data) over L_T. Figure 1 illustrates these two conditions.

Discussion Before presenting our new iterative DA algorithm, let us end this section by a general discussion.

The main and intuitive information provided by the previous theoretical results is that a DA algorithm $\mathcal{A}$ adapts well if at each step i the induced classifier $h^{(i)}$ satisfies the following conditions:

- $h^{(i)}$ **must work well on** L_T: The hypothesis $h^{(i)}$ learned by $\mathcal{A}$ from the *source* domain has to perform reasonably well on the *target* domain. This intuitive idea is formally expressed by the DA weak assumption of Definition 3.3 and Theorem 3.1 ($\gamma_T^{(i)} > 0$).
- $h^{(i)}$ **must work well on** L_S: As the quality of the target data inserted in the learning set mainly depends on the semi-labels assigned by $h^{(i)}$, one has

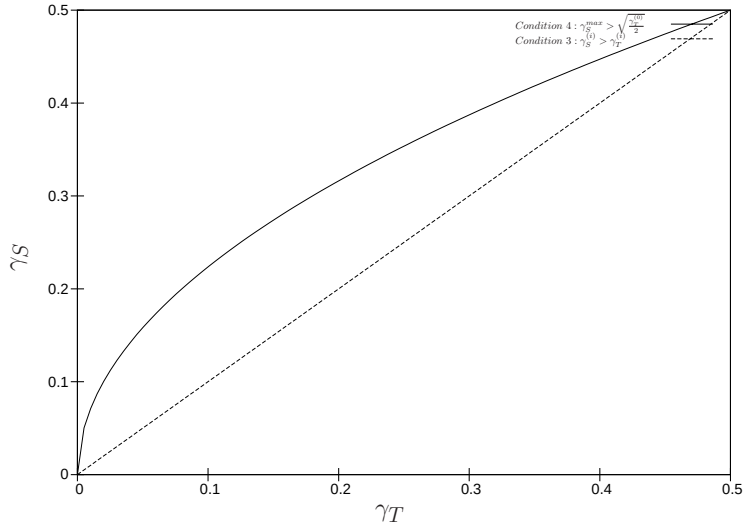

Fig. 1. Conditions for a DA algorithm to adapt well. The dashed line means that, according to Condition 3 of Theorem (3.2), γ_S must be at least greater than γ_T to learn something new at each iteration. The solid line expresses the additional Condition 4, that is, γ_S^{max} must also be at least greater than $\sqrt{\frac{\gamma_T^{(0)}}{2}}$ (i.e. the area above the curve).

to be confident in $h^{(i)}$. Therefore, $h^{(i)}$ has to perform sufficiently well on the data it has been learned from. This idea is formally expressed by Condition 3 of Theorem (3.2) ($\gamma_S^{(i)} > \gamma_T^{(i)}$).

- $\mathcal{A}$ **works better than a non adaptation process**: Indeed, the final classifier $h^{(\frac{N}{2k})}$ learned only from semi-labeled target data has to work better on L_T than a classifier learned only from source data. This idea is described by Condition 4 of Theorem (3.2) ($\gamma_S^{max} > \sqrt{\frac{\gamma_T^{(0)}}{2}}$) and illustrated by Figure 1.

A second remark about this theoretical study is that in the proof of Theorem (3.2), we made use of γ_S^{max} to bound every $\gamma_S^{(j)}$ to go from Equation 5 to Equation 6. It is worth noting that we could have used another strategy that would have led to a slightly different result. Coming back to Equation 5, we get:

$$\frac{4k}{N}\gamma_S^{(\frac{N}{2k})}\sum_{j=1}^{\frac{N}{2k}}\gamma_S^{(j-1)} > \gamma_T^{(0)}, \text{ because of Condition (3).}$$

$$\Leftrightarrow 2\gamma_S^{(\frac{N}{2k})}\frac{2k}{N}\sum_{j=1}^{\frac{N}{2k}}\gamma_S^{(j-1)} > \gamma_T^{(0)}$$

$$\Leftrightarrow 2\gamma_S^{(\frac{N}{2k})}\bar{\gamma}_S > \gamma_T^{(0)} \text{ where } \bar{\gamma}_S = \tfrac{2k}{N}\textstyle\sum_{j=1}^{\frac{N}{2k}}\gamma_S^{(j-1)}$$

$$\Leftrightarrow \bar{\gamma}_S > \gamma_T^{(0)}, \text{ because } \forall j, \gamma_S^{(j)} < \tfrac{1}{2}. \quad (7)$$

Equation 7 provides a different condition from that of described in Condition 4. By using the average of the γ values rather than the maximum value, we obtain with Equation 7 a less constrained condition to adapt well. Indeed, while we used $\frac{N}{2k}$ upper bounds to change Equation 5 into Equation 6, we resorted to only upper approximation to lead to Equation 7. Since the theoretical results presented in this section constitute the minimal requirement to adapt well, we preferred to keep the most constrained condition.

Finally, note that throughout our analysis, we used the quantity γ_T to provide theoretical guarantees of an iterative DA algorithm. We can notice that in practise, $\gamma_T^{(i)}$ is unknown because L_T is originally composed of unlabeled examples. Obviously, we claim that this does not challenge the interest of the previous theorems which actually give many information about the strategy to adapt well and to efficiently select the target points. For example, recall that DASVM selects on each side of the linear separator, the k examples falling into the margin band that are the closest to the margin bounds. In the light of our theoretical study, this strategy tends to satisfy the required conditions to adapt well. Indeed, on one hand, such selected points are those with the highest probability to modify the next SVM (because there are likely to be associated with nonzero Lagrange multipliers) and thus to **learn something new about the target domain**. On the other hand, by choosing those which are the closest to the margin bounds, *i.e.* the farthest from the hyperplane, one increases the probability for the semi-labeled data to be correctly classified. Therefore, DASVM also tends to **satisfy the weak DA assumption**.

For these reasons, we took inspiration from DASVM to design a new iterative DA algorithm, called GESIDA (for Good Edit Similarity-based Iterative Domain Adaptation), able to deal with structured data and aiming at satisfying the theoretical requirements to adapt well.

4. GESIDA

4.1. *Introduction*

Like DASVM, GESIDA takes as input a labeled training set L_S drawn from the source domain and an unlabeled set L_T from the target domain, both of them made up of structured data. It iteratively moves target instances from L_T to L_S to progressively modify a current classifier to get closer to the target problem. To relax the PSD constraint imposed by kernels, we suggest to replace the SVM-based procedure used in DASVM by a *good* similarity function-based process w.r.t. the framework introduced by Balcan et al in [22]. This choice allows us to directly make use of edit similarities even though they are not valid kernels, and learn a classifier by solving a simple linear program.

Let us denote by $L_S^{(i)}$ and $L_T^{(i)}$ respectively, the labeled sample set and the pool of unlabeled target data considered at each iteration i, such that $L_S^{(0)} = L_S$ and $L_T^{(0)} = L_T$. At time i, we learn a classifier $h^{(i)}$ from $L_S^{(i)}$ by solving the following linear problem:

$$\min_{\alpha} \sum_{i=1}^{|L_S^{(i)}|} \left[1 - \sum_{j=1}^{|L_S^{(i)}|} \alpha_j \ell_i K(x_i, x'_j) \right]_+ \text{ s.t. } \sum_{j=1}^{|L_S^{(i)}|} |\alpha_j| \leq 1/\gamma, \tag{8}$$

where γ is the desired margin, and $K(x, x') = -e_L(x, x')$ is simply the negative edit distance, renormalized in $[-1, 1]$. Then, we associate a candidate class $y^{T(i)} = sign(h^{(i)}(x^T))$ to every unlabeled target examples $x^T \in L_T$. At each iteration i, $2k$ such *semi-labeled* examples of the current pool $L_T^{(i)}$ are selected and moved into the training set $L_S^{(i+1)}$, in place of $2k$ original source examples. In the following section, we describe the strategy we use in GESIDA to select the target and source points involved at each iteration.

4.2. *Data Selection*

We proved in Section 3 that, to adapt well, more than half of the target data have to be correctly semi-labeled at each iteration i. As $\gamma_T^{(i)}$ is unknown, we use the margin to assess the confidence in the semi-labeled data. Indeed, as stated in [17], the higher the distance from the separation hyperplane, the higher the probability for an unlabeled data to be correctly classified (*i.e.* likely impliying that $\gamma_T^{(i)} > 0$). On the other hand, selecting only target data with the highest margin would not affect sufficiently the current classifier leading to the non respect of the condition $\tilde{\epsilon}_S^{(i+1)}(h^{(i)}) > \tilde{\epsilon}_s^{(i)}(h^{(i)})$ needed to learn something new. To find a good compromise between these two constraints, we propose a data selection in two steps:

- During the first one, we mainly aim at building a classifier "stable" enough on the target data. For this reason, we select the k semi-labeled examples of each class with the largest margin, considering that they are likely to be the closest from the source data, while being representative of their class in D_T. As done in DASVM[17], the $2k$ selected semi-labeled points replace $2k$ original source examples of $L_S^{(i)}$, those that are the farthest from the separator.
- During the second step, we aim at focusing on more difficult points for which the current model is not sufficiently *good*, that is the semi-labeled data whose margin is smaller than γ. We first insert the target points which have a margin slightly smaller than γ and progressively we concentrate our effort on the data which are closer and closer to the hyperplane.

More formally, let I and k be two meta-parameters of our algorithm and $T^{(i)} = T_{+1}^{(i)} \cup T_{-1}^{(i)}$ be the set of selected examples defined according to the following rules:

- First step: from $i = 0$ to $i = I$:

Data: a sample L_S of N source labeled examples, a sample L_T of $T > N$ unlabeled target instances, parameters k and I
Result: A classifier h
Initialization: $L_S^{(0)} = L_S$; $T_S^{(0)} = T_S$;
for $i = 1 \rightarrow \frac{N}{2k}$ **do**
  Learn $h^{(i)}$ from $L_S^{(i)}$ by solving Problem 2
  Compute the margin of the examples in L_S and L_T
  Update $L_S^{(i)}$ and $L_T^{(i)}$ to deal with semi-labeled outliers in $L_S^{(i)}$
  Build $T^{(i)}$ and $S^{(i)}$ according to parameters k and I
  $L_S^{(i+1)} \leftarrow T^{(i)} \cup (L_S^{(i)} \setminus S^{(i)})$
  $L_T^{i+1} \leftarrow L_T^{(i)} \setminus T^{(i)}$
end
Return the final classifier learned from $L_S^{(\frac{N}{2k})}$ by solving Problem 2;

Algorithm 1: Pseudo-code of GESIDA.

(a) $T_{+1}^{(i)}$ contains the k examples of $L_T^{(i)}$ with the highest margins according to $h^{(i)}$.
(b) $T_{-1}^{(i)}$ contains the k examples of $L_T^{(i)}$ with the lowest margins according to $h^{(i)}$.

- Second step, for $i > I$:
  (a) $T_{+1}^{(i)}$ contains the k examples of $L_T^{(i)}$ with the highest margins, according to $h^{(i)}$, immediately below $\gamma - (\frac{k}{|L_T|+1} \times (i - I))$.
  (b) $T_{-1}^{(i)}$ contains the k examples of $L_T^{(i)}$ with the lowest margins, according to $h^{(i)}$, immediately above $-(\gamma - (\frac{k}{|L_T|+1} \times (i - I)))$.

4.3. *Outlier Detection and Algorithm*

A classical risk in iterative learning algorithms is the insertion of outliers in the training set, especially during the first iterations. In DASVM, the authors move back any semi-labeled example from the current training set $L_S^{(i)}$ to $L_T^{(i+1)}$ (without the associated label) if its label changes between two iterations. We propose to replace this strong condition by another one inspired from the theory of boosting. In the BrownBoost algorithm, Freund [37] suggests to de-emphasize outliers when it seems clear that they are too hard to classify correctly after a given number of iterations. We apply this principle in our algorithm, by considering a minimum margin to reach by a semi-labeled example after a given number of iterations. Note that for each removed example, we replace it in the current training set by another semi-labeled example selected from $L_T^{(i+1)}$ in order to keep the same number of instances in the learning set. Our iterative process stops at iteration i when no more source example of the original L_S are present in the current learning sample $L_S^{(i)}$. The pseudo-code of GESIDA is presented in Algorithm 1.

5. Experimental Results

In this section, we provide an experimental evaluation of our DA algorithm for structured data both on string and tree representations. We propose to deal with original DA tasks by tackling the problems of scaling and rotation usually encountered in image classification. We use the well-known NIST Special Database 3 of the National Institute of Standards and Technology, containing a set of handwritten digits illustrated by bitmap images. On this task, it is possible to extract a suitable structured representation encoding each instance with Freeman codes [24].

We carry out five types of experiments. First, we begin with a preliminary study whose objective is to check the (ϵ, γ, τ)-goodness of the edit similarity used in our algorithm. Then, we provide a series of experiments with string-structured data followed by another series using a tree-based representation. Afterwards, we present an analysis of the reasonable points - corresponding to the discriminative patterns the classifier is based on - found by the algorithm during the iterations. We terminate this section by an experimental validation of the theoretical study of Section 3. Before detailing these experiments, we begin with the process used for encoding the images into structured data.

5.1. *Building Structured Representations from Handwritten Digits*

The NIST database contains digits encoded in 128×128 bitmap images. Each digit (from 0 to 9) is provided with 1000 different instances.

For obtaining strings, we represent each digit as a sequence of Freeman codes [24]. In order to find the encoding, the idea consists in following the contour of the digit from its top left pixel until returning to this pixel. The string corresponding to the digit is the sequence of Freeman codes representing the successive directions of the contour (see Figure 2 for an illustration).

To get tree structured data, we follow the principle proposed in [38]. To encode a given digit, the idea is first to build a root labeled by a fictive symbol `"-1"`. The next step consists in extracting the string representation based on Freeman codes as presented just before. This string is then parsed from left to right, each digit defining a new child of the root. If the same Freeman code is found during the parsing, each repetition becomes a child of the current node. An example of a tree-based representation is provided in Figure 3.

For string-structured data, we use the edit similarity function $K(x, x') = -e_L(x, x')$. For trees, we replace e_L by a tree edit distance. In the experiments, we use the Selkow tree edit distance algorithm [39] based on three edit operations: substitution of a node label, insertion of an entire subtree and deletion of an entire subtree (see [40] for a survey on tree edit distances).

5.2. *Goodness of Edit Similarities*

In this preliminary study, we check if the edit similarity K has good properties in terms of (ϵ, γ, τ)-goodness with respect to Definition 2.1 of the framework of Balcan

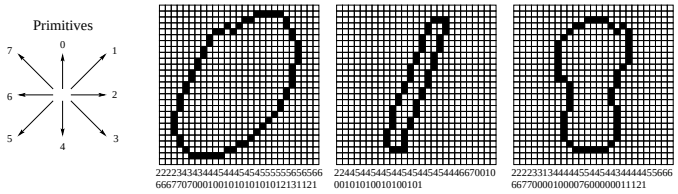

Fig. 2. Picture of three digits $(0, 1, 8)$ with their respective string representations.

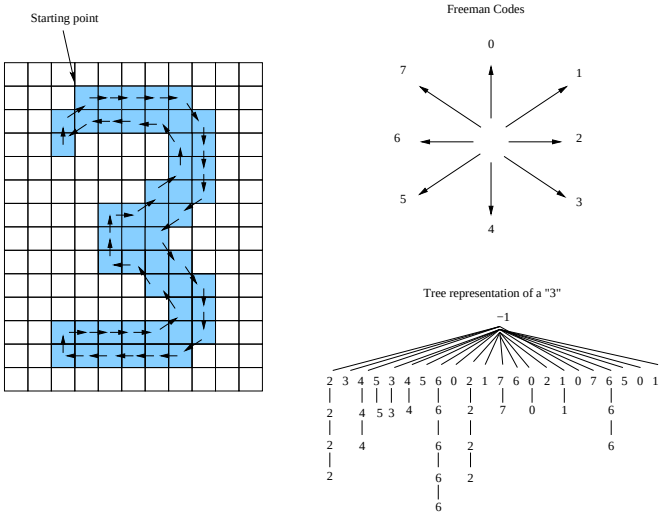

Fig. 3. Picture of a handwritten three and its tree structured representation.

et al. We estimate the goodness of this edit similarity on the 45 possible binary supervised classification problems (one for each pair of digits, without any domain adaptation issue), using 500 instances per class. We fix τ as equal to one, meaning that we consider all points as reasonable, and we estimate ϵ as a function of γ. For sake of simplicity, we only discuss the results obtained on two subproblems: "0 vs. 1" and "0 vs. 8", using a string encoding. The curves are shown in Figure 4. The interpretation is that a margin γ leads to an ϵ proportion of examples violating the margin. Figure 4 shows that even without selecting a relevant subset of reasonable points, there is a minority of examples violating the condition on the margin for a value of γ smaller than 0.3. A similar trend has been noticed on the other binary problems, and for tree-structured data as well (although more margin violations are encountered). From this experiment, we can conclude that the edit similarity offers reasonable goodness properties, and thus good generalization guarantees, according to the framework of Balcan et al.

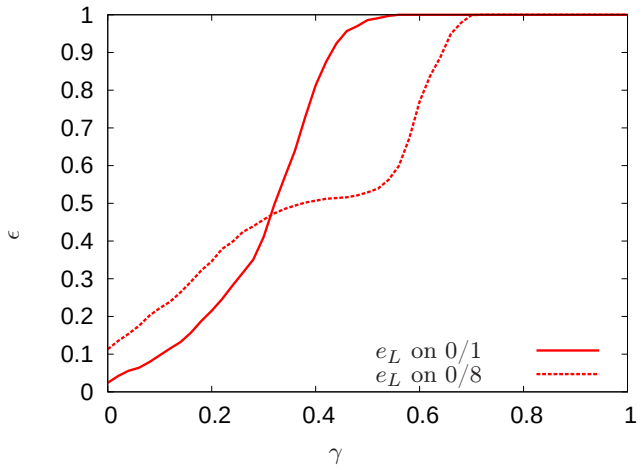

Fig. 4. Estimation of ϵ as a function of γ on two handwritten digits classification tasks.

5.3. *Experiments on String-structured Data*

We now present the experiments carried out for evaluating the ability of GESIDA to deal with two important invariance problems encountered in image processing corresponding to scaling and rotation operations. We study these problems for every possible binary task of the NIST database. Our objective is to check if DA techniques can provide a nice solution for these issues, usually tackled by defining invariant features such as SIFT [41]. We compare four approaches: (i) DASVM using $K(x, x') = \exp^{-t \times e_L(x,x')}$ (we used a simple LIBSVM based implementation to perform these experiments), (ii) GESIDA using the edit similarity $K(x, x') = -e_L(x, x')$, (iii) an approach following our principle but using a random selection of the examples to insert in and remove from $L_S^{(i)}$ at each iteration, and (iv) a baseline method without any adaptation, *i.e.* learning the model only from the source data. The meta-parameters of these methods are tuned by cross-validation on an independent set of examples. All the experiments are achieved according to a five-fold cross-validation procedure.

5.3.1. *Scaling Problems*

By encoding the contour of each bitmap image with Freeman codes, we obtain sequences of symbols whose length represents in some way the size of the digit. To deal with the scaling problems, we build for each binary problem a source training set L_S with the 100 smallest examples of each class, while the target set L_T - on which we aim at adapting the classifier - contains 100 digits among the 200 largest instances. The remaining 100 examples are used as a test set.

In the column entitled *Scaling* of Table 1, we report the averaged results obtained over all the binary problems along with the standard deviation. We can see that our

Table 1. Average results on the 45 scaling and rotation binary classification problems using a string-structured representation.

Total average	*Scaling*	*Rotation*
Accuracy by DASVM (in %)	83.3 ± 3.2	57.1 ± 11.7
Final number of support vectors	120 ± 7.8	113 ± 9.3
Accuracy by GESIDA (in %)	$\mathbf{94.7 \pm 2.1}$	$\mathbf{59.2 \pm 8.1}$
Final number of reasonable points	$\mathbf{11 \pm 2.4}$	$\mathbf{17 \pm 2.7}$
Accuracy by DA-random selection (in %)	52.14 ± 0.6	56.63 ± 7.8
Accuracy without adaptation (in %)	50.21 ± 1.7	55.48 ± 7.7

DA edit similarity approach is able to adapt well and is significantly better than DASVM (using a Student-paired t-test). Moreover, taking advantage of the L_1-norm, GESIDA induces much sparser models with an average number of reasonable points ten times lower than the average number of support vectors required with DASVM. We can also note that the accuracy reached by a random selection process is much lower than the one obtained with our selection strategy. The last line of Table 1 shows that a model learned from the source and directly applied on the target domain without any adaptation is not better than random guessing. This provides an experimental evidence that this scaling problem deserves an adaptation.

In Figure 5, we compare GESIDA with DASVM on each of the 45 scaling binary problems, represented by circles. We can see that, 28 circles out of 45 are above the line $y = x$, meaning that GESIDA outperforms DASVM on 28 scaling binary problems.

5.3.2. *Rotation Problems*

In this series of experiments, for every binary problem, we randomly choose 100 instances of each class to constitute the training set L_S, and 200 other instances of each class on which we apply a 90 degrees rotation. This setup allows us to define a rotation problem as a new DA task. As we did before, we keep 100 of these examples in the target sample L_T, the remaining ones being used to evaluate the methods.

The averaged results on the 45 rotation subproblems are reported in the column entitled *Rotation* of Table 1. Once again, we can see that GESIDA outputs much sparser models and is significantly better than DASVM on average. Despite this interesting behavior, we can note that rotation problems seem to be more difficult to overcome than scaling ones. This is confirmed by the rather small gains obtained by DA methods in comparison with an approach without adaptation or based on a random selection.

Figure 5 provides a comparison with DASVM for the 45 rotation binary problems, represented by triangles. GESIDA outperforms DASVM 27 times out of 45.

5.4. *Experiments on Tree-structured Data*

In this section, we report the experimental results obtained with tree-structured data. We restrict our study to scaling problems since the results obtained with

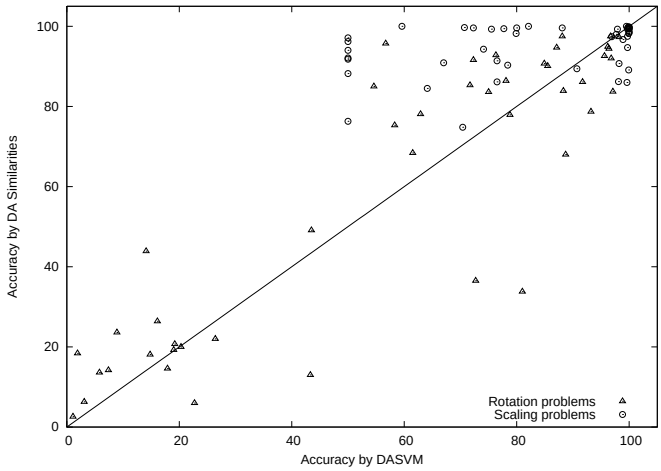

Fig. 5. Comparison between GESIDA and DASVM on the 45 binary tasks, both for scaling and rotation problems.

rotations problems follow the same trend as those presented for strings. We use the same experimental setup as introduced in the previous section, except that we use an edit similarity based on the Selkow tree edit distance.

The averaged results on the 45 subproblems are reported in Table 2. This experiment shows that our approach is able to provide better results than a random selection and than an approach without adaptation, which confirms that our method is able to perform an adaptation. GESIDA is on average, slightly better than DASVM, but the overall accuracies are significantly lower than the results obtained with strings. This limited performance illustrates the fact that the tree-structured representation used here is a priori not fully relevant for dealing with scaling or rotation problems in image classification[c]. More interestingly, we notice that the models obtained are still very sparse, leading again to cheap computational costs.

In Figure 6, we compare the results obtained by GESIDA and DASVM on each of the 45 binary scaling problems. We can see that, 28 circles out of 45 are above the line $y = x$, meaning that GESIDA is better than DASVM using a win/loss test.

5.5. *Study of Reasonable Points*

One interesting feature of GESIDA is its ability to produce very sparse models which has been confirmed in all the experiments. One may then wonder what these examples selected as reasonable points look like. Intuitively, these points should be

[c]For example, if one considers the two sequences of Freeman codes 22222222 and 22122122, their edit distance is 2 while their distance according to the tree-representation with Selkow's algorithm is 12. A more complex structuration seems thus needed in the tree-representation.

Table 2. Average results on the 45 scaling binary classification problems using tree-structured representations.

Total average	*Scaling*
Accuracy by DASVM (in %)	66.30 ± 15.3
Final number of support vectors	95 ± 17.6
Accuracy by GESIDA (in %)	$\mathbf{67.46 \pm 14.9}$
Final number of reasonable points	$\mathbf{16 \pm 6.9}$
Accuracy by DA-random selection (in %)	53.21 ± 0.4
Accuracy without adaptation (in %)	50.04 ± 0.25

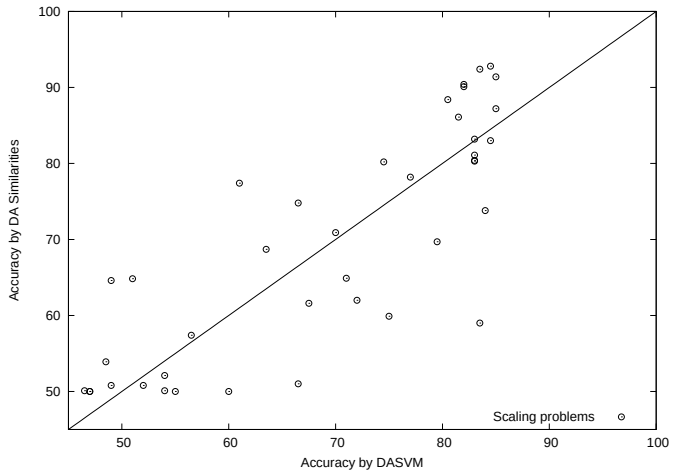

Fig. 6. Comparison between GESIDA and DASVM on the 45 binary tasks corresponding to scaling problems using tree-structured data.

some kind of discriminative instances a classifier is based on. In this section, our objective is to study them according to the different iterations of the DA process.

To analyse these points, we consider a task aiming to separate even and odd digits, from a string-structured representation. We propose to deal with a rotation problem as a DA task because it highlights immediately the origin of each example among the source and target data. We then use the following setup. First, we build a learning set containing 100 digits labeled "even", and 100 labeled "odd" (20 instances of each digit). We construct a target set with the same principle except that we apply a 90 degrees rotation on each of the 100 examples. Then, we apply GESIDA with a high margin value to ensure very sparse models.

In Figure 7, we show the set of reasonable points selected at the beginning of the process, when we learn the first classifier from the source data only. This small set captures a sufficient level of "diversity" of some examples to be able to discriminate. The different 0's, 3's and 6's correspond to several variations of one digit. We can note that some digits are not represented, like 8's, 1's and 9's. This can be justified by the fact that the contour of 0's and 6's can cover 8's. Similarly, the lower part

of 3's and 5's can represent the lower part of a 9, and 7's - which are not similar to any even digit - can stand for 1's.

Now, we consider reasonable points obtained for a model at the middle of the DA process. At this stage, the learning sample contains both source and target examples and thus we expect to have both types of examples as reasonable points. The points selected by the model are shown in Figure 8. First, as expected, both types of points are present, but the number of points is larger than the first set presented before. This is explained by the fact that at this stage, the model must find discriminative patterns for both source and target tasks which implies to consider more examples. The digits are a bit different from the first set and do not cover the same diversity. One possible exaplanation is that some examples may be mislabeled, reducing the number of different patterns the model must consider.

The reasonable points selected for the last model learned during the DA process are represented in Figure 9. As expected, this set contains only target digits. It is also smaller than the one obtained at the middle of the process which can be justified by the fact that the algorithm needs only to rely on one type of examples. This set is also a bit different from the first one: the 7 being replaced by a 1, the 8 and 6 by a 0. The diversity is captured here by instances of 4's and 2's, but there is a lack of representativeness for odd numbers. This last point can explain why we can not obtain very good results on this task.

Finally, in Figure 10, we show the proportion of reasonable points coming from the original source and from target data according to the different iterations. Target points become the most represented only from the last third of the iterations showing that the source examples keep a high importance even after the middle of the process.

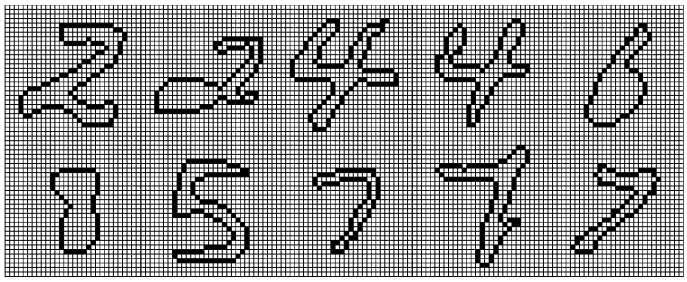

Fig. 7. Set of reasonable points selected by the first classifier learned by GESIDA for a task aiming to separate even and odd digits

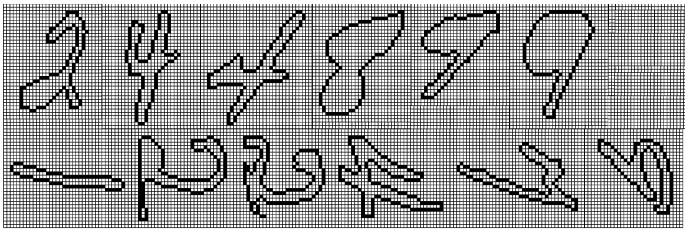

Fig. 8. Set of reasonable points obtained at the middle of the DA procedure dealing with a rotation problem for a task aiming to separate even and odd digits.

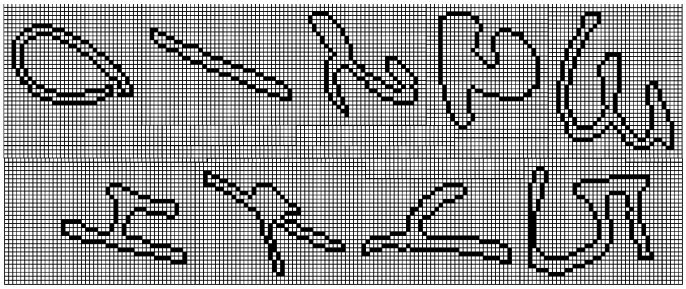

Fig. 9. Set of reasonable points obtained at the end of a DA procedure dealing with a rotation problem for a task aiming to separate even and odd digits.

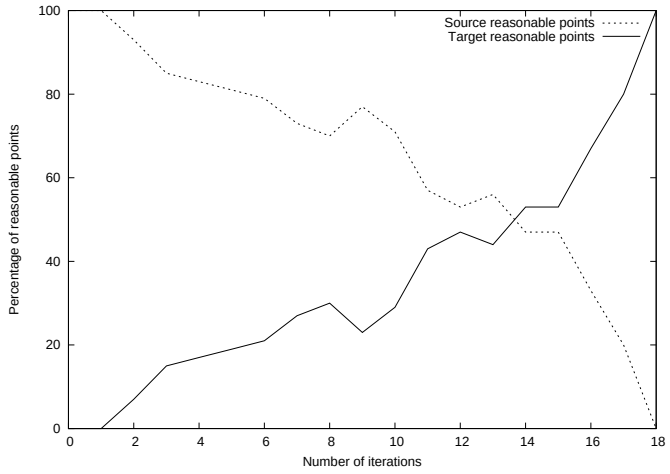

Fig. 10. Proportion of reasonable points coming from the source and target data according to the iterations of a DA procedure dealing with a rotation problem for a task aiming to separate even and odd digits.

5.6. *Experimental Evaluation of the Random Selection Approach*

In this last series of experiments, we aim at verifying that the theoretical conditions presented in Section 3 to adapt well, in the context of a random selection over the

Table 3.
Experiments on random selection on two classification problems.

Iteration	P_1			P_2		
	$\gamma_S^{(i)}$	$\gamma_T^{(i-1)}$	1 - $\hat{\epsilon}_T^{(i)}$	$\gamma_S^{(i)}$	$\gamma_T^{(i-1)}$	1 - $\hat{\epsilon}_T^{(i)}$
1	0.5	0	0.585	0.50	-0.1	0.32
2	0.475	0.085	0.75	0.50	-0.18	0.285
3	0.48	0.25	0.73	0.50	-0.215	0.285
4	0.49	0.23	0.795	0.50	-0.215	0.24
5	0.49	0.295	0.875	0.50	-0.26	0.18
6	0.49	0.375	0.94	0.50	-0.32	0.205
7	0.49	0.44	0.94	0.50	-0.295	0.19
8	0.49	0.44	0.94	0.50	-0.31	0.12
9	0.49	0.44	0.94	0.50	-0.38	0.145
10	0.495	0.44	0.985	0.50	-0.355	0.115
11	0.5	0.485	0.99	0.495	-0.385	0.115

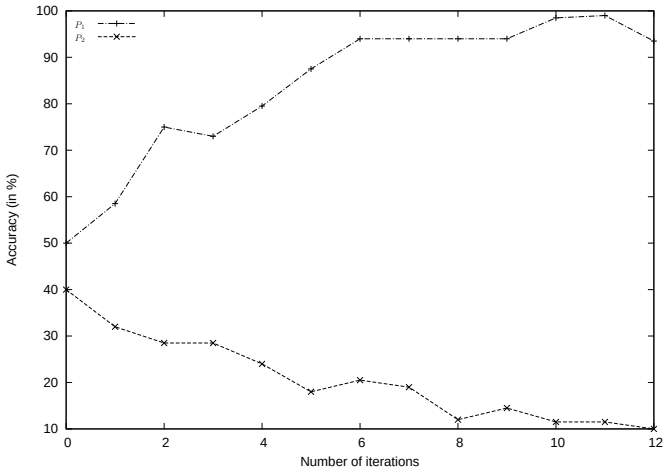

Fig. 11. Evolution of the generalization target error on two problems P_1 and P_2.

examples, are empirically confirmed. In this context, we assume to have access to the true label of the target examples in order to be able to compute $\gamma_T^{(i)}$, usually unknown. We consider two DA problems: P_1 corresponding to a scaling binary classification task between 0's and 1's, and P_2 corresponding to a rotation binary classification task between 5's and 7's. These two problems are selected such that GESIDA works well on P_1 and diverges on P_2. We report in Table 3 the results for the values $\gamma_S^{(i)}$, $\gamma_T^{(i)}$ and 1 - $\hat{\epsilon}_T^{(i)}$.

First, we can remark that in the two cases, the conditions presented in Theorem 3.2 are fulfilled during all the iterations, *i.e.* $\forall i, \gamma_S^{(i)} > \gamma_T^{(i)}$ and $\gamma_S^{max} > \sqrt{\frac{\gamma_T^{(0)}}{2}}$. However, the hypotheses inferred at each iteration in problem P_2 are not weak DA learners in the sense of Definition 3.3. This explains the divergence phenomenon on this problem. On the other hand, in problem P_1, all the hypotheses are weak DA

learners, justifying that our algorithm converges. This behavior associated with the two necessary conditions of Theorem 3.2 ensures a good adaptation. The evolution of the generalization target errors during the iterations for P_1 and P_2 are reported in Figure 11. This figure illustrates that the divergence is immediate from the very first iterations, while the accuracy increases regularly when a good adaptation is possible. This experiment highlights then that the theoretical study of Section 3 gives necessary conditions to adapt well.

6. Conclusion

In this paper, we have proposed a novel iterative DA approach - GESIDA - specifically designed to deal with structured data such as strings or trees. In opposition to methods projecting such data in a numerical space, we directly handle them by using an edit-based similarity having some (ϵ, γ, τ)-good capabilities to be used within the framework of Balcan et al. We took advantage of this framework to propose a domain adaptation algorithm using edit similarities. Our method iteratively incorporates target examples in the learning set by selecting them according to their current margin.

We have also provided a theoretical analysis that addresses the general case of iterative domain adaption methods. We have given minimum necessary conditions to adapt. These conditions rely notably on the necessity of having weak DA learners and a minimal relationship between the initial weak DA learner and the best weak learner found during the algorithm.

We have evaluated GESIDA by considering original DA tasks dealing with problems of robustness to scaling and rotation operations in image classification. Our method has shown good adaptation capabilities, especially on scaling problems with string-structured data and is better than DASVM on average. An important feature of the algorithm is that it outputs very sparse models. This point is crucial from a large scale application perspective since the computation of an edit similarity is generally costly.

A first theoretical perspective concerns our theoretical study. The results have been established in the baseline case of a random selection of the target examples. Therefore, these results are independent from the strategy used in the algorithm for selecting the examples. Thus, an analysis able to take into account this strategy could give tighter results, more informative on the behavior of the algorithm. Another direction consists in trying to derive some generalization bounds showing that our approach can iteratively converge to the expected joint error over the two domains. From an application standpoint, a possible direction is to study the influence of other structured representations for representing images. Studying the ability of DA methods to address more general invariance problems in image classification tasks is also an interesting future work. Finally, we also aim at applying this kind of iterative DA procedure for regression problems or temporal data classification.

References

1. L. G. Valiant, "A theory of the learnable," *Commun. ACM*, vol. 27, no. 11, pp. 1134–1142, 1984.
2. B. Roark and M. Bacchiani, "Supervised and unsupervised pcfg adaptation to novel domains," in *Proceedings of HLT-NAACL*, 2003, pp. 126–133.
3. R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling." *Computer Speech and Language*, vol. 10, pp. 187–228, 1996.
4. A. M. Martinez, "Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class." *IEEE Trans. Pattern Anal. Mach.Intell.*, vol. 24, pp. 748–763, 2002.
5. S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proceedings of NIPS 2006*, 2006, pp. 137–144.
6. Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proceedings of COLT 2009*, 2009.
7. Y. Mansour and M. Schain, "Robust domain adaptation," in *ISAIM*, 2012.
8. J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting sample selection bias by unlabeled data," in *Proceedings of NIPS'2006*, 2006, pp. 601–608.
9. M. Sugiyama, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Proceedings of NIPS'2007*, 2008.
10. M. Dudík, R. E. Schapire, and S. J. Phillips, "Correcting sample selection bias in maximum entropy density estimation," in *NIPS*, 2005.
11. S. Bickel, M. Brückner, and T. Scheffer, "Discriminative learning for differing training and test distributions," in *Proceedings of the 24th international conference on Machine learning*, ser. ICML '07. New York, NY, USA: ACM, 2007, pp. 81–88.
12. Y. Tsuboi, H. Kashima, S. Hido, S. Bickel, and M. Sugiyama, "Direct density ratio estimation for large-scale covariate shift adaptation," *JIP*, vol. 17, pp. 138–155, 2009.
13. J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. 2006 Conf. Empirical Methods in Natural Language Processings*, 2006, pp. 120–128.
14. S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *AAAI*, 2008, pp. 677–682.
15. S. Satpal and S. Sarawagi, "Domain adaptation of conditional probability models via feature subsetting," in *PKDD*, 2007, pp. 224–235.
16. Y. Ji, J. Chen, G. Niu, L. Shang, and X. Dai, "Transfer learning via multi-view principal component analysis," *J. Comput. Sci. Technol.*, vol. 26, no. 1, pp. 81–98, 2011.
17. L. Bruzzone and M. Marconcini, "Domain adaptation problems: A dasvm classification technique and a circular validation strategy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 770–787, 2010.
18. Ó. Pérez and M. A. Sánchez-Montañés, "A new learning strategy for classification problems with different training and test distributions," in *IWANN*, 2007, pp. 178–185.
19. R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *J. ACM*, vol. 21, no. 1, pp. 168–173, 1974.
20. C. Cortes, P. Haffner, and M. Mohri, "Rational kernels: Theory and algorithms," *Journal of Machine Learning Research*, vol. 5, pp. 1035–1062, 2004.
21. M.-F. Balcan and A. Blum, "On a Theory of Learning with Similarity Functions," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2006, pp. 73–80.
22. M.-F. Balcan, A. Blum, and N. Srebro, "Improved guarantees for learning via similarity

functions," in *Proceedings of COLT*, 2008, pp. 287–298.
23. G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, no. 11, pp. 613–620, Nov. 1975.
24. H. Freeman, "Computer processing of line-drawing images," *ACM Comput. Surv.*, vol. 6, no. 1, pp. 57–97, 1974.
25. M. R. Daliri and V. Torre, "Robust symbolic representation for shape recognition and retrieval," *Pattern Recogn.*, vol. 41, no. 5, pp. 1799–1815, May 2008.
26. P. Punitha and D. S. Guru, "Symbolic image indexing and retrieval by spatial similarity: An approach based on b-tree," *Pattern Recogn.*, vol. 41, no. 6, pp. 2068–2085, 2008.
27. S.-M. Hsieh and C.-C. Hsu, "Retrieval of images by spatial and object similarities," *Inf. Process. Manage.*, vol. 44, no. 3, pp. 1214–1233, 2008.
28. Y. Mansour, "Learning and domain adaptation," in *Proceedings of the 12th International Conference on Discovery Science*, 2009, pp. 32–34.
29. S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, pp. 151–175, May 2010.
30. A. Margolis, "A literature review of domain adaptation with unlabeled data," *Tec. Report*, pp. 1–42, 2011.
31. S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345–1359, 2010.
32. J. Jiang, "A Literature Survey on Domain Adaptation of Statistical Classifiers," Available from: http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/da_survey.html, Mar. 2008.
33. R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos, "A statistical model for multilingual entity detection and tracking," in *Proc. 2004 Conf. North Am. Chapter ofthe Assoc. for Computational Linguistics and Human Language Technology*, 2004, pp. 1–8.
34. H. Daumé, "Frustratingly easy domain adaptation," in *Proc. 45th Ann. Meeting of the Assoc for Computational Linguistics*, 2007, pp. 256–263.
35. A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, ser. COLT' 98. ACM, 1998, pp. 92–100.
36. Y. Freund and R. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of ICML'96*, 1996, pp. 148–156.
37. Y. Freund, "An adaptive version of the boost by majority algorithm," in *In Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, 2000, pp. 102–113.
38. M. Bernard, L. Boyer, A. Habrard, and M. Sebban, "Learning probabilistic models of tree edit distance," *Pattern Recognition*, vol. 41, no. 8, pp. 2611–2629, 2008.
39. S. Selkow, "The tree-to-tree editing problem," *Information Processing Letters*, vol. 6, pp. 184–186, 1977.
40. P. Bille, "A survey on tree edit distance and related problem," *Theoretical Computer Science*, vol. 337, no. 1-3, pp. 217–239, 2005.
41. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.