Learning to Combine Distances for Complex Representations

Adam Woznica WOZNICA@CUI.UNIGE.CH
Alexandros Kalousis KALOUSIS@CUI.UNIGE.CH
Melanie Hilario HILARIO@CUI.UNIGE.CH
University of Geneva, Computer Science Department, Rue General Dufour 24, 1211 Geneva 4, Switzerland

Abstract

The k-Nearest Neighbors algorithm can be easily adapted to classify complex objects (e.g. sets, graphs) as long as a proper dissimilarity function is given over an input space. Both the representation of the learning instances and the dissimilarity employed on that representation should be determined on the basis of domain knowledge. However, even in the presence of domain knowledge, it can be far from obvious which complex representation should be used or which dissimilarity should be applied on the chosen representation. In this paper we present a framework that allows to combine different complex representations of a given learning problem and/or different dissimilarities defined on these representations. We build on ideas developed previously on metric learning for vectorial data. We demonstrate the utility of our method in domains in which the learning instances are represented as sets of vectors by learning how to combine different set distance measures.

1. Introduction

The k-Nearest Neighbor (kNN) algorithm (Aha, 1997) is an effective method to address classification problems. Similar to other distance-based algorithms kNN does not require a direct access to the training examples, instead it accesses the data only by a (dis-)similarity function. Although intuitively simple the kNN algorithm has proved its utility in many real-world applications (Aha, 1997). The common approach in most of the kNN classifiers is to represent the training instances as vectors in the $\mathbb{R}^n$ space where the Euclidean metric is used to measure the dissimilarities between examples. This approach has the advantages of simplicity and generality, however, it has two main limitations. First, most of today's machine learning applications hardly fit within the typical propositional representation and in such cases more general representations (e.g. sets, graphs) should be used (Woźnica et al., 2006; Ramon & Gärtner, 2003). Second, the Euclidean metric implies that the input space is isotropic, which is rarely valid in practical applications.

The proper representation of the learning examples and the actual dissimilarity over that representation are important constituents of kNN critically influencing its performance. Both of them should be directly determined by domain knowledge and application requirements, however, in practice we rarely have a solid description of the learning problem. As a result for a given problem there might exist several plausible dissimilarities which are defined on different representations and reflect different aspects of the data. As an example consider labeled graphs, a widely used representation in machine learning to represent complex objects, e.g. chemical compounds. Graphs, depending on the application requirements, can be represented among others as adjacency matrices, set of trees or sets of walks (Ramon & Gärtner, 2003). Moreover, for each of the above representations different dissimilarities can be applied. An obvious question is how to select from this set of predefined dissimilarities and representations the one that best fits the requirements of the problem at hand. A simple solution is to select the dissimilarity (usually from a predefined set of dissimilarities) by cross-validation. The main drawback of this approach is that only one dissimilarity per training set is selected which limits the expressiveness of the resulting method. Additionally this approach requires the use of extra data.

In this paper we show how to learn distance measures for the kNN classification by combining predefined distances and the corresponding representations. We exploit ideas developed previously for adapting a metric to a given task by learning it directly from the data. In this context several attempts have been recently made, either in supervised (Goldberger et al., 2005; Globerson & Roweis, 2006; Domeniconi & Gunopulos, 2002; Weinberger et al., 2006;

Appearing in *Proceedings of the 24^{th} International Conference on Machine Learning*, Corvallis, OR, 2007.

Yang et al., 2006; Schultz & Joachims, 2004) or in semi-supervised settings (Xing et al., 2003; Hillel et al., 2003). The distance measures are usually restricted to belong to a Mahalanobis metric family[1] parametrized by a positive semi-definite (PSD) matrix. All these methods were developed for vectorial data and are similar in the sense that the actual problem is cast as a mathematical optimization task. However, these algorithms differ wrt the actual objective function that is being optimized and hence they implicitly assume different distributions of the data. In this work we exploit three of the above methods, namely the ones presented in Xing et al. (2003), Goldberger et al. (2005) and Globerson and Roweis (2006), and adapt them so that they can be used in the context of complex structures. To the best of our knowledge it is the first attempt to learn distances over complex structures.

We will demonstrate our approach in applications in which learning examples are most naturally represented in the form of sets of vectors of possibly different cardinalities and the distances to be combined are different distances on sets. Nevertheless the framework is more general and can be applied on any arbitrary combination of distances and complex objects representation.

The paper is organized as follows. In Section 2 we describe the problem of choosing the complex representation and the dissimilarity function defined over that representation for a given application. In Section 3 we propose a framework for learning combinations of distance measures on complex objects where we exploit methods that were initially developed for learning metrics over vectorial data. Experimental results are reported in Sections 4 and 5, and Section 6 presents the related work. We conclude with Section 7 where we address open issues and future work.

2. Representations and Dissimilarities on Complex Objects

One of the main challenges in applications involving complex objects is that of the proper representation of the learning instances. The choice of the correct representation is crucial for the successful application of machine learning techniques since it renders the actual problem easier (if not trivial) to solve. Different languages for representing complex objects for the task of learning have been used over the years, mainly based on first order logic (Dzeroski & Lavrac, 2001). Within these languages the complex objects can be represented in different manners modeling for different semantics and aspects of the problem. For example, depending on the application, graphs can be represented as sets of trees, walks, etc. (Ramon & Gärtner, 2003) or adjacency matrices.

Strongly associated with the problem of selection of the appropriate representation, is that of selection of an appropriate dissimilarity on the selected representation. It is possible to have different dissimilarities for a given representation, where again each dissimilarity models different semantics. Using again the example of graphs: if these are represented as sets of objects then we can choose among different dissimilarity measures on sets, whereas if they are modeled as adjacency matrices then we should choose among different dissimilarity measures for matrices.

Ideally, both the representation of the learning instances and the dissimilarity employed on that representation, should be determined on the basis of domain knowledge. However, even in the presence of domain knowledge, it can be far from obvious which complex representation should be used or which dissimilarity should be applied in the chosen representation. In this paper we take the view that the establishment of the appropriate combination of representation and dissimilarity should be a part of the learning process. In what follows we will collectively identify the couple consisting of the representation and the dissimilarity employed on this representation as a *distance measure*.

2.1. Distances on Sets

We will demonstrate the utility of our approach on learning complex distance combinations on problems where the learning objects come in the form of sets and we need to combine different set distance measures. Thus in this section we will give a brief description of set distance measures. We should emphasize here that the use of sets does not mean we are limited only to applications for which sets is an adequate representation, in fact the ideas presented later for distance combination are applicable to any type of distance measures over complex objects.

The central idea in set distance measures used in this work is the definition of a mapping of elements of one set to elements of the other set such that the final distance is determined on the basis of specific pairs of elements from the two corresponding sets. Different types of mappings correspond to distances that have different semantics (Woźnica et al., 2006). It should be noted that the other approach for computing distances is performed by comparing summary statistics computed from the corresponding sets, e.g. (Tatti, 2007); distances of that type can also be incorporated within the framework presented here. Consider two nonempty and finite sets $A = \{a\} \subseteq \mathcal{X}$ and $B = \{b\} \subseteq \mathcal{X}$. Let $d(.,.)$ be a metric defined on $\mathcal{X}$. The set distance measure D defined on $2^{\mathcal{X}}$ as $D(A,B) = \sum_{(a,b)\in F} d(a,b)$, i.e. it is a sum of pairwise distances over specific pairs which are defined by $F \subseteq A \times B$. Most of the set distances are normalized by $D(A,B) := \frac{D(A,B)}{|F|}$

[1] The Mahalanobis metric parametrized by a positive semi-definite matrix A is defined as: $d_A(x,y) = (x-y)^T A(x-y)$

such that the final distance takes values between 0 and 1. Within this framework we can define the *Sum of Minimum Distances (*$\mathrm{D_{SMD}}$*)*, *Hausdorff (*$\mathrm{D_H}$*)*, *RIBL (*$\mathrm{D_{RIBL}}$*)*, *Surjections (*$\mathrm{D_S}$*)*, *Linkings (*$\mathrm{D_L}$*)*, *Fair Surjections (*$\mathrm{D_{FS}}$*)* and *Matchings (*$\mathrm{D_M}$*)* distances. Description of these distance measures together with the corresponding references to existing literature can be found in (Woźnica et al., 2006).

Nothing can be said about the general superiority of one distance measure over another. While this provides opportunities for exploiting prior knowledge, it can also be difficult in practice to find prior justification for the use of one set distance instead of another, thus providing an ideal context for distance combination.

3. Learning to Combine Distances

We view the problem of complex distance combination as an optimization problem where the two main constituents are the definition of a (differentiable) cost function that depends on the class labels and an optimization method. We only focus on learning a *global* distance measure as opposed to *local* methods which aim to determine a "stretched" neighborhood around each query instance such that class conditional probabilities are likely to be constant (Yang et al., 2006; Domeniconi & Gunopulos, 2002). Local methods form an interesting alternative and were shown to achieve better performance for data exhibiting "difficult" distributions, however, global methods have the advantage of providing insight into the underlying structure of the data which might be subsequently used e.g. for dimensionality reduction.

We begin with a labeled set of n complex objects $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where $x_i \in \mathcal{X}$ and $y_i \in \{1, 2, \ldots, c\}$. Each complex object x_i is given by the vector $[x_{i_1}, \ldots, x_{i_m}]$ where $x_{i_l} \in \mathcal{X}_l$ and $\mathcal{X}_l$, $l = 1, \ldots, m$, are different representation spaces of the complex objects. Thus $x_i \in \mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_m$. On each of the $\mathcal{X}_l$ a distance measure D_l is defined. We define $\vec{D}(x_i, x_j) = [D_1(x_{i_1}, x_{j_1}), \ldots, D_m(x_{i_m}, x_{j_m})]^T$. The quadratic combination of the above m distances is defined as:

$$D_A^2(x_i, x_j) = \vec{D}(x_i, x_j)^T A \, \vec{D}(x_i, x_j) \tag{1}$$

where A is a $m \times m$ positive semi-definite (PSD) matrix ($A \succeq 0$) to ensure that D_A is a valid pseudometric[2]. Equation 1 can be reparametrized as:

$$D_W^2(x_i, x_j) = \vec{D}(x_i, x_j)^T W^T W \vec{D}(x_i, x_j) \tag{2}$$

where $A = W^T W$ and W is a $m \times m$ marix. For any W we have $A = W^T W \succeq 0$. An optimization problem where the objective function is optimized wrt matrix W is not constrained by $W \succeq 0$ and thus is easier to solve. Note that the quadratic form of Equation 1 is similar to that of the Mahalanobis distance for vectorial data. The difference is that the Mahalanobis distance requires direct access to the training examples whereas D_A^2 (or D_W^2) accesses the data only by the corresponding distance functions. The above formulation can be applied to any complex data for which different distance measures are defined.

A common approach for learning a metric is to provide information in the form of equivalence relations as pairwise constraints on the data. In the classification framework there is a natural equivalence relation, namely whether two points are in the same class or not, i.e. $\mathcal{S} = \{(x_i, x_j) | y_i = y_j\}$ and $\mathcal{D} = \{(x_i, x_j) | y_i \neq y_j\}$. The general problem of metric learning in a supervised setting can be now stated as the following optimization problem:

$$\min_Z \mathcal{F}_Z(\mathcal{S}, \mathcal{D}, D_Z^2) \tag{3}$$

where $\mathcal{F}_Z$ is a differentable function, $Z = A$ or $Z = W$, and this optimization is subject to some constraints. For example, for the parametrization in Equation 1 the optimization from Equation 3 has to be constrained by $A \succeq 0$. Depending on the actual form of the function $\mathcal{F}_Z$ in Equation 3 and the optimization technique, different instantiations of the algorithm can be obtained.

In the next three subsections we explore three different instantiations of the above framework which differ with respect to the actual objective function which is being optimized. We based our study on methods originally developed for vectorial data, namely Xing's method (Xing et al., 2003), the Maximally Collapsing Metric Learning (MCML) method (Globerson & Roweis, 2006) and Neighborhood Component Analysis (NCA) method (Goldberger et al., 2005). We adapted these methods so that they can learn quadratic distance combinations of the form given in Equation 1 (for Xing's and MCML methods) and Equation 2 (for NCA). The methods differ with respect to the assumptions they make for the data distribution. We should emphasize here that the three distance learning methods we build upon were developed for vectorial data. In that context the methods were learning the appropriate weighted combination of attributes. In our context they will learn combinations of full blown distances on the complete representation of the learning objects.

3.1. Xing's Method

The first method proposed by Xing et al. (2003) was originally developed for semi-supervised clustering. Adapting from (Xing et al., 2003) we formulate the problem of dis-

[2] In fact D_A will be a pseudometric iff $A \succeq 0$ and all $D_l, l = 1, \ldots, m$ are (pseudo-)metrics. In general $A \succeq 0$ is necessary to ensure that D_A^2 is non-negative.

tance combination as the following optimization task:

$$\begin{aligned} \min_{A} \quad & \sum_{(x_i,x_j)\in\mathcal{S}} D_A^2(x_i,x_j) \\ \text{s.t.} \quad & \sum_{(x_i,x_j)\in\mathcal{D}} D_A(x_i,x_j) \geq 1 \ , A \succeq 0 \end{aligned} \tag{4}$$

It can be shown that the optimization problem from Equation 4 is convex and is equivalent (up to a multiplication of A by a positive constant) to minimizing:

$$\mathcal{F}_A = \sum_{(x_i,x_j)\in\mathcal{S}} D_A^2(x_i,x_j) - \log\left(\sum_{(x_i,x_j)\in\mathcal{D}} D_A(x_i,x_j)\right) \tag{5}$$

subject to $A \succeq 0$.

From Equation 5 it is clear that more emphasis is put on minimizing the pairwise distances between *all* examples in the same class. This implicitly assumes that instances from each class form a single compact connected set. In particular, for binary problems where the negative class contains *any* examples which do not have the property encoded by the positive class, the cost function for the Xing method will be severely penalized. A similar problem occurs for data exhibiting highly multi-modal distributions. The other problem with the method from Equation 4 is that the use of squared distance in the minimization term and the root of square distance for the constraint term is arbitrary and asymmetric.

3.2. MCML

The MCML algorithm is based on the simple geometric intuition that all points of the same class should be mapped onto a single location and far from points of the other classes (Globerson & Roweis, 2006). To learn the metric which would approximate this ideal geometrical setup a conditional distribution is introduced which for each example x_i selects another example x_j as its neighbor with some probability $p_A(j|i)$, and inherits its class label from the point it selects. The probability $p_A(j|i)$ is based on the softmax of the D_A distance:

$$p_A(j|i) = \frac{e^{-D_A^2(x_i,x_j)}}{\sum_{k\neq i} e^{-D_A^2(x_i,x_k)}} \ , p(i|i) = 0 \tag{6}$$

It can be shown (Globerson & Roweis, 2006) that any set of points which has the distribution $p_0(j|i) = 1$ if $(x_i, x_j) \in \mathcal{S}$ and $p_0(j|i) = 0$ if $(x_i, x_j) \in \mathcal{D}$ exhibits the desired ideal geometry. It is thus natural to seek a matrix A such that $p_A(j|i)$ is as close (e.g. in the sense of the KL divergence) to $p_0(j|i)$. This is equivalent to minimizing (subject to $A \succeq 0$):

$$\mathcal{F}_A = \sum_i KL[p_0(j|i)|p_A(j|i)] \tag{7}$$

MCML is not based on Gaussian assumptions and the sufficient statistics used in the method are n "spread" matrices centered at each point (Globerson & Roweis, 2006). The main difference between the MCML methods and Xing's is that the former puts more emphasis into the pairs of points which are in different classes. As a result the MCML is better suited for classification problems.

3.3. NCA

The NCA method from (Goldberger et al., 2005) attempts to directly optimize a continuous version of the leave-one-out error of the kNN algorithm on the training data. The main difference between NCA and the two previous methods is that optimization in NCA is done wrt matrix W of Equation 2. The actual cost function used is a differentiable function based on stochastic neighbor assignments in the weighted space which is based on $p_W(j|i)$ of Equation 6. In the following we denote the set of points that share the same class with x_i by $C_i = \{j|(x_i, x_j) \in \mathcal{S}\}$. Under this stochastic selection rule the probability $p_W(i)$ of classifying x_i correctly is given by $p_W(i) = \sum_{j\in C_i} p_W(j|i)$.

The objective function, as presented in (Goldberger et al., 2005), which is to be maximized is given by:

$$\mathcal{F}_W = -\sum_i \log(p_W(i)) \tag{8}$$

which expresses the probability of obtaining an error free classification on the training set (Goldberger et al., 2005).

The main advantage of the NCA method is that it is non-parametric, making no assumptions about the shape of the class conditional distributions or the corresponding boundaries. In this sense it is similar to the MCML method. As already mentioned the main problem with the NCA algorithm is that there is no guarantee that a gradient method will converge to the global optima.

3.4. Regularization

One possible problem with the optimization task of Equation 3 is that for full matrices Z the number of parameters to estimate is m^2. This could be problematic in cases where m is large wrt the number of instances in the training database and could lead to overfitting. One possible solution to overcome this problem is to add a soft constraint to the objective function, which results in the following regularized optimization task

$$\min_{Z} \left(\mathcal{F}_Z(\mathcal{S},\mathcal{D},D_Z^2) + \lambda R(Z)\right) \tag{9}$$

where $R(Z)$ is a regularization term and $\lambda > 0$ is a regularization parameter. For D_A defined in Equation 1 we set $R(A) = Tr(A)$, i.e. the trace of A, whereas for D_W given in Equation 2 we set $R(W) = \|W\|_{\mathcal{F}}^2$ where $\|W\|_{\mathcal{F}}$ is the Frobenious norm of W. It is easy to show that $Tr(A) = \|W\|_{\mathcal{F}}$.

An alternative solution that we will explore is to restrict matrix A (or W) to be diagonal resulting in a weighted combination of distances, i.e. $D_Z^2(x_i, x_j) = \sum_{l=1\ldots m} t_{ll} D_l^2(x_i, x_j)$ where t_{ll} are the diagonal elements of A or $W^T W$. This restriction can be seen as a simple form of regularization since it reduces the effective number of parameters from m^2 to m.

It should be noted that the regularization technique based on diagonal matrices, although faster then the one of Equation 9, is also less expressive since it does not account for interactions between different distance measures. When full matrices are used these interactions are weighted by off diagonal elements of matrix A or $W^T W$, i.e. $D_Z^2(x_i, x_j) = \sum_{l=1\ldots m, k=1\ldots m} t_{lk} D_l(x_i, x_j) D_k(x_i, x_j)$.

3.5. Solving Optimization Problems

As already mentioned in the Xing and MCML methods the objective functions are *convex* (Avriel, 2003) and thus the optimization problems are well defined in the sense that there exists a single global optimum. In order to ensure that $A \succeq 0$ we use the iterative projection approach as in Xing et al. (2003). We calculate the eigen-decomposition of $A = \sum_k \lambda_k u_k u_k^T$, where $\lambda_k, k = 1, \ldots, m$ are A's eigenvalues, u_k the corresponding eigenvectors, and set $A = \sum_k \max(\lambda_k, 0) u_k u_k^T$. In the NCA method the optimization is done wrt matrix W of Equation 2 which makes the optimization problem easier to solve, since it is unconstrained. However, the objective function is no longer convex and is thus susceptible to local minima.

To solve the optimization problems we exploit the conjugate gradient method (Avriel, 2003) where backtracking line search is used to optimize the step-size parameter. For full matrices this method has a complexity of $O(m^2)$ since it requires computing a gradient in the form of an $m \times m$ matrix. In the case of diagonal matrices the above complexity is reduced to $O(m)$.

4. Experiments

In the experiments we will evaluate the methods proposed in Section 3 on a number of relational benchmark datasets where training instances are represented as sets of vectors. The different complex distance combination methods are compared in the context of the kNN algorithm. The goal is to examine whether we can increase the predictive performance of kNN by combining them. We will denote kNN in which the combined distances are used by $\mathrm{kNN}_{\mathrm{DC}}$.

We compared two regularized instantiations of the Xing, MCML and NCA methods, over full and diagonal matrices which we will denote respectively $\mathrm{METHOD}_{\mathrm{full}}$ and $\mathrm{METHOD}_{\mathrm{diag}}$ where METHOD is Xing, MCML or NCA. We used two methods as a baseline for comparison. The first one, denoted as $\mathrm{kNN}_{\mathrm{Best}}$, is obtained by simply selecting the set distance which gives the best 10-fold CV performance on the full dataset and reporting that performance. It should be noted that this performance estimate is optimistically biased. The second, denoted as $\mathrm{kNN}_{\mathrm{CV}}$, is based on an inner cross-validation loop to select the appropriate set distance. More precisely on each training set an inner 10-fold stratified cross-validation is performed for each set distance in order to select the one with the highest accuracy.

We experiment on a number of relational problems: musk, mutagenesis, diterpenes and protein fingerprints. The musk dataset was described in Dietterich et al. (1997) and is a standard multi-instance benchmark dataset; we worked with both versions (1 and 2) of the dataset, containing respectively 92 and 102 instances. The Mutagenesis dataset was introduced in Srinivasan et al. (1994); we experiment with the 188-instance, "regression friendly" version of this dataset, and represent each molecule as a set of bonds together with the two adjacent atoms. In the diterpenes dataset (Dzeroski et al., 1996) the goal is to identify the type of diterpenoid compound skeletons given their ^{13}*C-NMR*-Spectrum; it contains 1503 instances. The last classification task was first described in (Hilario et al., 2004). Protein fingerprints are groups of conserved motifs (regions) drawn from multiple sequences alignment that can be used as diagnostic signatures to identify and characterize collections of protein sequences. Broadly speaking, fingerprints may be diagnostic for a gene family or superfamily (united by a common function), or a domain family (united by a common structural motif). In this work we model protein fingerprints as sets of their component motifs; the dataset contains 1842 instances.

In all the datasets the number of nearest neighbors was optimized in an inner 10-fold cross validation loop over $k = \{1, 3, 9\}$. Depending of the dataset, we used two different values of the regularization parameter λ: for datasets with a "small" number of instances wrt the number of distance measures, m, (musk and mutagenesis) we set $\lambda = 10$, i.e. heavy regularization, whereas for datasets with a "large" number of instances wrt m (diterpenes and protein fingerprints) we set $\lambda = 0.1$, i.e. soft regularization. We estimate accuracy using stratified ten-fold cross-validation and control for the statistical significance of observed differences using McNemar's test (sig. level=0.05). The results (with the significance test results) are presented in Table 1.

5. Results

In Table 1, we see that MCML and NCA have an advantage over the baseline methods considered. They are never significantly worse and sometimes they are significantly better than both $\mathrm{kNN}_{\mathrm{Best}}$ and $\mathrm{kNN}_{\mathrm{CV}}$. On the other hand Xing's

Table 1. Accuracy and significance test results of the $\mathrm{kNN_{DC}}$ algorithm for benchmark datasets (+ stands for a significant win of the first algorithm in the pair, - for a significant loss and = for no significant difference). The first sign in the parenthesis corresponds to the comparison of $\mathrm{kNN_{DC}}$ vs. kNN where the best distance is used (the $\mathrm{kNN_{Best}}$ column) and the second to the comparison of $\mathrm{kNN_{DC}}$ vs. kNN with cross-validation (the $\mathrm{kNN_{CV}}$ column). Additionally for $\mathrm{kNN_{Best}}$ the best set distance measure is given in parenthesis. $\mathrm{Xing_{full}}$, $\mathrm{MCML_{full}}$ and $\mathrm{NCA_{full}}$ denote methods where optimization is performed over a full matrix whereas in $\mathrm{Xing_{diag}}$, $\mathrm{MCML_{diag}}$ and $\mathrm{NCA_{diag}}$ the matrix is restricted to be diagonal.

	$\mathrm{Xing_{full}}$	$\mathrm{Xing_{diag}}$	$\mathrm{MCML_{full}}$	$\mathrm{MCML_{diag}}$	$\mathrm{NCA_{full}}$	$\mathrm{NCA_{diag}}$	$\mathrm{kNN_{Best}}$	$\mathrm{kNN_{CV}}$
MUSK1	80.4(=)(=)	75.0(=)(=)	84.8(=)(=)	84.8(=)(=)	85.9(=)(+)	89.6(=)(+)	84.8 ($\mathrm{D_L}$)	79.3
MUSK2	65.7(=)(=)	53.9(-)(-)	72.5(=)(=)	66.7(=)(=)	77.5(=)(=)	85.1(=)(+)	77.5 ($\mathrm{D_{FS}}$)	74.5
MUTA	80.8(=)(=)	83.5(=)(=)	80.3(=)(=)	89.1(+)(+)	78.7(=)(=)	76.1(=)(=)	84.0 ($\mathrm{D_{SMD}}$)	82.4
DITERP.	70.3(-)(-)	96.5(=)(=)	96.2(=)(=)	98.3(+)(+)	98.3(+)(+)	95.7(=)(=)	96.1 ($\mathrm{D_M}$)	96.1
FP	77.4(-)(-)	83.8(-)(-)	84.8(=)(=)	84.7(=)(=)	84.7(=)(=)	84.9(=)(=)	85.4 ($\mathrm{D_M}$)	85.5

method does not fair so well, being sometimes significantly worse than both baseline methods.

We also conducted experiments where the optimization is performed over full matrix instantiations without the regularization (these results are not reported in this paper). The main finding is that the regularized versions of MCML and NCA tend to have an advantage over the corresponding non-regularized instantiations, mainly for datasets with small number of examples (musk and mutagenesis). For large datasets the differences in performances are not statistically significant.

The poor performance of Xing's method might be a result of the fact that the objective function in this method is heavily penalized for data exhibiting a multi-modal distribution whereas MCML and NCA are non-parametric; they make no assumptions about the shape of the class conditional distributions or the boundaries between different classes. Another possible explanation for the poor performance of Xing's method, especially in musk1, musk2 and mutagenesis datasets, is that in these datasets the negative class contains *any* examples which do not have the property encoded by the positive class. As already mentioned, in such cases the cost function implemented by the Xing method is severely penalized. Finally note that the performances achieved by $\mathrm{MCML_{full}}$ and $\mathrm{NCA_{diag}}$ on the diterpenes dataset are the best reported so far in the literature.

The results of the optimization process, both for full and diagonal matrices, can be visualized to provide insight into the relative importance of the distance measures (or their combinations), as these are determined by each method. An example of such a visualization is given in Figure 1 for the musk2 dataset, where optimization is performed over diagonal matrices. The x-axis corresponds to a given distance measure for which the corresponding accuracy of kNN, estimated by 10-fold CV, is given in parenthesis. The y-axis represents the (normalized) weights returned by the three methods. The visualization results are in agreement with results from Table 1. In particular $\mathrm{NCA_{diag}}$ assigns high weights to distance measures which individually exhibit good performance ($\mathrm{D_{SMD}}$, $\mathrm{D_H}$, $\mathrm{D_S}$, $\mathrm{D_{FS}}$ and $\mathrm{D_M}$) and neglects the ones with low performance ($\mathrm{D_{RIBL}}$ and $\mathrm{D_M}$). On the other hand the poor performance of $\mathrm{Xing_{diag}}$ (53.9 %) is a result of assigning a very high weight to $\mathrm{D_{RIBL}}$. The full matrix obtained for the diterpenes dataset is displayed graphically in Figure 2. The highest weights are assigned to $\mathrm{D_H}$, $\mathrm{D_S}$, $\mathrm{D_{FS}}$ and $\mathrm{D_M}$ as well as the combinations of these distances.

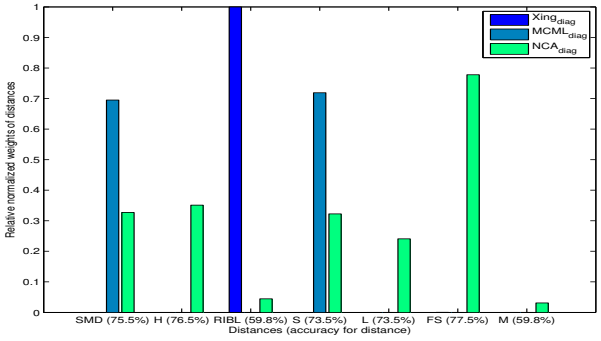

Figure 1. Relative importance of the different set distance measures, for the musk2 dataset, as these are computed by $\mathrm{Xing_{diag}}$, $\mathrm{MCML_{diag}}$ and $\mathrm{NCA_{diag}}$. Weights are normalized by the Frobenius norm of A. For each set distance measure we also give the corresponding performance of kNN in parenthesis.

Using the method instantiations based on diagonal matrices, it is easy to reduce the size of the problem by selecting only the l top-ranked distance measures (according to the assigned coefficients) which can be used to obtain the distance combination. We examined the performance of $\mathrm{kNN_{DC}}$ for l ranging from 1 to m (i.e. the total number of distance measures). The visualization for the musk1 dataset is given in Figure 3. It should be noted that the objective functions are optimized wrt all the distance functions considered. Indeed from Figure 3 it is clear that the best performance of $\mathrm{kNN_{DC}}$ is achieved when all the distance measures are used (i.e. $l = m$). However, the performance when we select the three top distance measures is very similar to the performance when all distances are

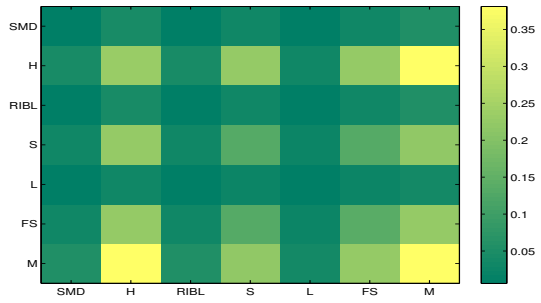

Figure 2. Relative importance of the different set distance measures, for the diterpenes dataset, as these are computed by $\mathrm{NCA}_{\mathrm{full}}$. Weights are normalized by the Frobenius norm of A.

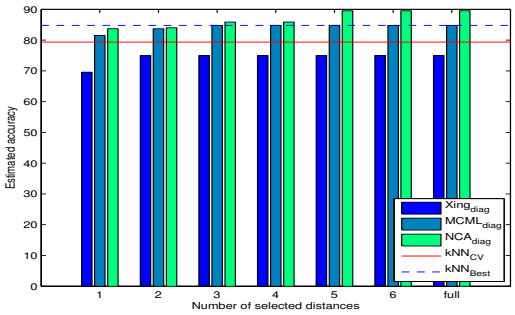

Figure 3. The estimated accuracy for $\mathrm{Xing}_{\mathrm{diag}}$, $\mathrm{MCML}_{\mathrm{diag}}$ and $\mathrm{NCA}_{\mathrm{diag}}$ for the musk1 dataset where $l = 1, \ldots, m$ top ranked distance measures (according to the assigned coefficients) are combined for computing the actual distance. Additionally the performance of $\mathrm{kNN}_{\mathrm{Best}}$ and $\mathrm{kNN}_{\mathrm{CV}}$ is given.

used.

There is an interesting synergy that arises from the nature of the set distances and the fact that we learn combinations of them. Remember that all the set distances we considered are defined on the basis of a given mapping, F, of elements of one set to elements of the other set. One can view the task of learning a set distance measure as learning the mapping F, i.e. which pairs of elements of the two sets should participate in the mapping and how important they are. Under this view learning a set distance measure would correspond to learning the weights ω_{ab} (associated with a pair of elements $(a, b) \in A \times B$), in the function $D(A, B) = \sum_{(a,b)\in A\times B} \omega_{ab} d(a, b)$ where $\omega_{ab} \in \{0, 1\}$ (or more general $\omega_{ab} \in [0, 1]$). It can be seen that the combination of distances from Equation 1 provides an intermediate solution to this problem. By restricting the matrix from Equation 1 to be diagonal, we obtain a set distance measure of the form $D(A, B) = \sum_{(a,b)\in F} w_{ab} d(a, b)$ where $F = \bigcup_{i=1}^{m} F_i$, F_i is the mapping corresponding to set distance measure D_i and w_{ab} is computed by adding the coefficients assigned to set distance measures in which (a, b) appears. The final mapping, F, is more expressive than any of its constituents, F_i, and cannot be obtained by considering any of the F_is individually.

6. Related Work

As already noted any (semi-)supervised metric learning method can be adapted to distance combination in complex domains, as long as in the objective function the access to data is only through a distance function. The RCA algorithm from (Hillel et al., 2003) constructs a Mahalanobis metric from a weighted sum of in-class covariance matrices; however, it only takes into account similar pairs of points and discards the dissimilar ones. As a result it is unlikely that this method will perform well on fully labeled data. The method proposed in (Schultz & Joachims, 2004) learns the metric from relative and qualitative examples of the form "A is closer to B than A is to C". It is not straightforward to extend this method to learn the metric in classification settings. Two algorithms from (Shalev-Shwartz et al., 2004) and (Weinberger et al., 2006) can be easily adapted to distance combinations. The cost functions in these algorithms are based on the notion of large margin which separates elements with different labels while keeping elements of the same class together. The main difference is that the latter focuses on the local neighborhood while the former seeks to minimize distance between *all* similarly labeled examples.

In addition to metric learning several attempts have been recently made to learn kernel operators directly from the data, e.g. (Lanckriet et al., 2004). This approach is more general than metric learning in the sense that any valid kernel k can be directly used to compute a pseudo metric in the feature space. The proposed methods differ in the objective functions as well as in the classes of kernels that they consider. In the case of set distances based on kernels there are two main problems. First, most of the kernels over sets proposed so far in the literature are based on averaging (i.e. they depend on *all* the elements in the two sets). This feature might be inappropriate for some applications (e.g. multiple-instance problems) where the classification depends on *specific* pairs of elements from the two sets (Woźnica et al., 2006). Second, with the exception of the work of Argyriou et al. (2005), all of the methods work only in a transductive setting, i.e. completing the labeling of a partially labeled dataset.

7. Conclusions and Future Work

In this paper we presented a framework that allows to combine different complex representations of a given learning

problem and/or different dissimilarities defined on these representations. We exploit ideas developed previously on metric learning for vectorial data. To the best of our knowledge this is the first time that distance combination (which amounts to distance learning) is applied to non-vectorial data.

In the future we will consider other metric learning algorithms described in Section 6. We will also examine the impact of other regularization techniques on classification performance. In particular, we will exploit more aggressive regularization similar to that used in LASSO (Tibshirani, 1996), which forces some of the elements of matrix A (W) shrunk to zero. We would like also to extend our framework to combine distances locally (Yang et al., 2006; Domeniconi & Gunopulos, 2002), which will result in an even more flexible method for distance combination on complex domains.

References

Aha, D. W. (1997). Editorial. *Artificial Intelligence Review*, *11*, 1–6. Special Issue on Lazy Learning.

Argyriou, A., Micchelli, C. A., & Pontil, M. (2005). Learning convex combinations of continuously parameterized basic kernels. *COLT 2005*. Bertinoro, Italy.

Avriel, M. (2003). *Nonlinear programming: Analysis and methods*. Dover Publications.

Dietterich, T. G., Lathrop, R. H., & Lozano-Perez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, *89*, 31–71.

Domeniconi, C., & Gunopulos, D. (2002). Adaptive nearest neighbor classification using support vector machines. In *Nips 14*. Cambridge, MA: MIT Press.

Dzeroski, S., & Lavrac, N. (2001). *Relational data mining*. Springer-Verlag New York, Inc.

Dzeroski, S., Schulze-Kremer, S., Heidtke, K. R., Siems, K., & Wettschereck, D. (1996). Applying ILP to diterpene structure elucidation from ^{13}c NMR spectra. *Inductive Logic Programming Workshop* (pp. 41–54).

Globerson, A., & Roweis, S. (2006). Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf and J. Platt (Eds.), *Nips 18*, 451–458. Cambridge, MA: MIT Press.

Goldberger, J., Roweis, S., Hinton, G., & Salakhutdinov, R. (2005). Neighbourhood component analysis. In *Nips*. Cambridge, MA: MIT Press.

Hilario, M., Mitchell, A., Kim, J. H., Bradley, P., & Attwood, T. (2004). Classifying protein fingerprints. *PKDD 2004*. Pisa, Italy: Springer-Verlag.

Hillel, A. B., Hertz, T., Shental, N., & Weinshall, D. (2003). Learning distance functions using equivalence relations. *ICML 2003*.

Lanckriet, G., Cristianini, N., Bartlett, P. L., Ghaoui, L. E., & Jordan, M. (2004). Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, *5*, 27–72.

Ramon, J., & Gärtner, T. (2003). Expressivity versus efficiency of graph kernels. *First International Workshop on Mining Graphs, Trees and Sequences (help with ECML/PKDD'03)*.

Schultz, M., & Joachims, T. (2004). Learning a distance metric from relative comparisons. In S. Thrun, L. Saul and B. Schölkopf (Eds.), *Advances in neural information processing systems 16*. Cambridge, MA: MIT Press.

Shalev-Shwartz, S., Singer, Y., & Ng, A. Y. (2004). Online and batch learning of pseudo-metrics. *ICML '04: Proceedings of the twenty-first international conference on Machine learning* (p. 94). New York, NY, USA: ACM Press.

Srinivasan, A., Muggleton, S., King, R., & Sternberg, M. (1994). Mutagenesis: ILP experiments in a non-determinate biological domain. *Proceedings of the 4th International Workshop on Inductive Logic Programming* (pp. 217–232).

Tatti, N. (2007). Distances between data sets based on summary statistics. *Journal of Machine Learning Research*, *8*, 131–154.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, *58*, 267–288.

Weinberger, K., Blitzer, J., & Saul, L. (2006). Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems 18*. Cambridge, MA: MIT Press.

Woźnica, A., Kalousis, A., & Hilario, M. (2006). Distances and (indefinite) kernels for sets of objects. *The IEEE International Conference on Data Mining (ICDM)*. Hong Kong.

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2003). Distance metric learning with application to clustering with side-information. In *Nips 15*, 505–512. Cambridge, MA: MIT Press.

Yang, L., Jin, R., Sukthankar, R., & Liu, Y. (2006). An efficient algorithm for local distance metric learning. *Proceedings of AAAI*.