A New Framework for Dissimilarity and Similarity Learning

Adam Woźnica and Alexandros Kalousis

University of Geneva, Computer Science Department
7 Route de Drize, Battelle batiment A
1227 Carouge, Switzerland
{Adam.Woznica,Alexandros.Kalousis}@unige.ch

Abstract. In this work we propose a novel framework for learning a (dis)similarity function. We cast the learning problem as a binary classification task or a regression task in which the new learning instances are the pairwise absolute differences of the original instances. Under the classification approach the class label we assign to a specific pairwise difference indicates whether the two original instances associated with the difference are members of the same class or not. Under the regression approach we assign positive target values to the pairwise differences of instances from different classes and negative target values to the differences of instances of the same class. The computation of the (dis)similarity of two examples amounts to the computation of prediction scores for classification, or the prediction of a continuous value for regression. The proposed framework is very general as we are free to use any learning algorithm. Moreover, our formulation generally leads to a (dis-)similarity which, depending on the learning algorithm, can be efficient and simple to learn. Experiments performed on a number of classification problems demonstrate the effectiveness of the proposed approach.

1 Introduction

The k-Nearest Neighbour (kNN) algorithm is an effective method to address classification problems that has proved its utility in many real-world applications [1]. Most common kNN classifiers use the Euclidean metric to measure the dissimilarities between examples. This approach has the advantages of simplicity and generality, however, its main limitation is that the Euclidean metric implies that the input space is isotropic which is rarely valid in practical applications [2].

Since the Euclidean metric is not appropriate for many real-world learning problems different researchers have recently proposed methods for learning the parameters of a parametrized distance measure directly from the data, either in a fully supervised setting [2–7] or using side information [8–10]. The distances in the above methods are usually restricted to belong to a Mahalanobis metric family parametrized by a positive semi-definite (PSD) matrix $\boldsymbol{A}$.[1] The goal in metric learning is to discover an "optimal" matrix $\boldsymbol{A}$ that achieves a higher kNN predictive performance than the Euclidean metric.

[1] The Mahalanobis metric is defined as: $d_{\boldsymbol{A}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{A}(\boldsymbol{x}_i - \boldsymbol{x}_j)$.

M.J. Zaki et al. (Eds.): PAKDD 2010, Part II, LNAI 6119, pp. 386–397, 2010.

In most of the above metric learning methods the input information is given in the form of equivalence relations. A widely used equivalence relation in the classification setting indicates whether two instances, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, belong to the same class or not. It is important to realize that the existing techniques do not require a direct access to the instances, instead they access them through their pairwise distances, or equivalently through their pairwise difference vectors $|\boldsymbol{x}|_{ij}$. The elements of the latter are the absolute differences of the attributes of the two instances. More formally, for $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathrm{I\!R}^p$, $|\boldsymbol{x}|_{ij}$ is defined as:

$$|\boldsymbol{x}|_{ij}^T = (|x_{i_1} - x_{j_1}|, \ldots, |x_{i_p} - x_{j_p}|)^T \tag{1}$$

where x_{i_l} denotes the l-th attribute of $\boldsymbol{x}_i$. Metric learning algorithm assign weights to the attributes vectors of the form of (1), or to their pairwise products in the case of quadratic metrics. The weighted differences are then aggregated to compute the final Mahalanobis metric. In general, the metric learning methods assign the weights so that under the new metric pairs of instances of the same class will be close together (i.e. their Mahalanobis metric will have a value close to 0), and pairs of instances of different classes will be far apart (i.e. their Mahalanobis metric will have larger values).

Based on the above observations we go one step further and use the pairwise difference vectors of the form given by equation 1 as our learning instances. More precisely, our approach is based on casting the dissimilarity learning problem as a binary classification or a regression task defined over the space of the absolute difference vectors. When we treat the problem as a classification task we assign negative labels to these difference vectors that correspond to pairs of instances of the same class and positive labels to the difference vectors that correspond to pairs of instances from different classes. In the regression scenario we assign negative and positive target values, respectively. The construction of the learning problem in this new space is justified by the fact that for instances of the original space that belong to the same class, some attributes of $|\boldsymbol{x}|_{ij}$ should have small values, while for instances of different classes these attributes should have large values. Moreover, by exploring classification or regression algorithms that produce models based on weighted combinations of the input attributes (e.g. Support Vector Machines or Logistic Regression), we expect that the non-discriminatory attributes will be assigned low weights, and hence instances in the new space with positive and negative classes (or positive and negative numbers) will be moved respectively far from and towards the origin of the new space. In our framework the computation of a dissimilarity measure between two examples amounts to computing a prediction score in classification or a continuous value in regression; the higher the predicted score or the predicted value of the target variable, the more dissimilar are the corresponding two input instances.

The paper is organized as follows. In Sect. 2 we present the existing metric learning techniques under a common framework. This will directly motivate the main contribution of this work presented in Sect. 3, where we propose a new framework for learning (dis-)similarity measures. Experimental results are reported in Sect. 4. Finally, we conclude with Sect. 5 where we discuss major open issues and future work.

2 Metric Learning

In this section we will present the metric learning problem in a common framework that will help us to motivate the main contribution of this work in Sect. 3. We begin with a labeled set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\} = (\mathcal{X}, \mathcal{Y})$ where $\boldsymbol{x}_i \in \mathbb{R}^p$ and $y_i \in \{1, 2, \ldots, c\}$.

Based on the notation from (1), we will represent in a compact way both Euclidean and Mahalanobis metrics between two instances $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^p$. More precisely, the squared Euclidean metric can be defined as $d^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = |\boldsymbol{x}|_{ij}^T |\boldsymbol{x}|_{ij}$ while the squared Mahalanobis metric, parameterized by a positive semi-definite matrix $\boldsymbol{A} \in \mathbb{R}^{p \times p}$ ($\boldsymbol{A} \succeq 0$), can be represented as:

$$d^2_{\boldsymbol{A}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = |\boldsymbol{x}|_{ij}^T \boldsymbol{A}\ |\boldsymbol{x}|_{ij}. \tag{2}$$

We note that it is sometimes useful to reparametrize the Mahalanobis metric as:

$$d^2_{\boldsymbol{W}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = |\boldsymbol{x}|_{ij}^T \boldsymbol{W}^T \boldsymbol{W}\ |\boldsymbol{x}|_{ij} \tag{3}$$

where $\boldsymbol{A} = \boldsymbol{W}^T \boldsymbol{W}$ and $\boldsymbol{W} \in \mathbb{R}^{p \times p}$. For any $\boldsymbol{W}$ we have $\boldsymbol{A} = \boldsymbol{W}^T \boldsymbol{W} \succeq 0$. In what follows, to emphasize that all the above metrics between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ depend only on $|\boldsymbol{x}|_{ij}$, we will denote $d^2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $d^2_{\boldsymbol{L}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ ($\boldsymbol{L} = \boldsymbol{A}$ or $\boldsymbol{L} = \boldsymbol{W}$) as $d^2(|\boldsymbol{x}|_{ij})$ and $d^2_{\boldsymbol{L}}(|\boldsymbol{x}|_{ij})$, respectively.

A common approach in the existing metric learning methods is to provide information in the form of equivalence relations as pairwise constraints on the input instances. In the classification framework there is a natural equivalence relation, namely whether two vectors share the same class label or not, i.e. $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : c_{ij} = 0\}$ and $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : c_{ij} = 1\}$ where $c_{ij} \in \{0, 1\}$ indicates whether or not the labels y_i and y_j match:

$$c_{ij} = \begin{cases} 0 & \text{if } y_i = y_j \\ 1 & \text{otherwise.} \end{cases} \tag{4}$$

It should be stressed that the existing distance learning methods do not require a direct access to the pairs of instances in either of the above sets $\mathcal{S}$ and $\mathcal{D}$; instead, they access the data through the distance functions $d_{\boldsymbol{A}}$ or $d_{\boldsymbol{W}}$ and hence only though pairwise distance vectors $|\boldsymbol{x}|_{ij}$ of the form given in (1). Consequently, to emphasize the above fact, we will consider only the following versions of the equivalence relations $\mathcal{S}' = \{|\boldsymbol{x}|_{ij} : c_{ij} = 0\}$ and $\mathcal{D}' = \{|\boldsymbol{x}|_{ij} : c_{ij} = 1\}$.

The general problem of metric learning in a supervised setting can be now stated as the following optimization problem:

$$\min_{\boldsymbol{L}}.\ \mathcal{F}_{\boldsymbol{L}}(\mathcal{S}', \mathcal{D}') + \lambda \Omega(\boldsymbol{L}) \tag{5}$$

where $\mathcal{F}_{\boldsymbol{L}}$ is a (possibly non-differentiable) cost function, $\boldsymbol{L} = \boldsymbol{A}$ or $\boldsymbol{L} = \boldsymbol{W}$ and $\Omega(\cdot)$ is a regularization term[2] whose importance is controlled by the λ regularization parameter. Additionally, the above optimization problem is possibly subject to a number

[2] We set $\Omega(\boldsymbol{A}) = Tr(\boldsymbol{A})$ and $\Omega(\boldsymbol{W}) = \|\boldsymbol{W}\|_F^2$, where $Tr(\cdot)$ and $\|\cdot\|_F$ denote the matrix trace and the Frobenious norm, respectively.

of constraints. For example, for the parametrization from (2) the optimization given in (5) has to be constrained by $\boldsymbol{A} \succeq 0$. Depending on the actual form of the function $\mathcal{F}_{\boldsymbol{L}}$ different instantiations of the algorithm can be obtained.

One possible problem with the optimization problem of (5) is that for full matrices $\boldsymbol{L}$ the number of parameters to estimate is p^2. For large p (i.e. in the order of few thousands), this would render the optimization task non tractable as there will be too many parameters to optimize over. We explore a solution to this problem that is based on restricting matrices $\boldsymbol{L}$ to be diagonal, resulting in a weighted combination of features (this restriction can be seen as a simple form of regularization since it reduces the effective number of parameters from p^2 to p). It should be noted that the approach based on diagonal matrices, although faster then the one based on full matrices, is also less expressive since it does not account for interactions between different attributes. On the other hand, it allows for the application of metric techniques also on high-dimensional datasets.[3]

In the rest of this section we will present 3 different instantiations of the above framework which differ with respect to the objective function $\mathcal{F}_{\boldsymbol{L}}$ and hence the assumptions they make for the data distribution. More precisely, in this work we will focus on the following state-of-the-art metric learning algorithms: Large Margin Nearest Neighbor (LMNN) [11], Maximally Collapsing Metric Learning (MCML) [4] and Neighborhood Component Analysis (NCA) [3].

LMNN. The cost function $\mathcal{F}_{\boldsymbol{A}}$ of LMNN [2] is constructed in such a way that it penalizes both large distances between each sample and its similarly labeled nearest neighbors, and small distances between differently labeled instances. Equivalently, the criterion of LMNN seeks for a metric in which each sample has a large margin between nearest neighbors of same class and samples in different classes. We use $\eta_{ij} \in \{0, 1\}$ to denote the neighbourhood relation between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ where $\eta_{ij} = 1$ indicates the sample $\boldsymbol{x}_i$ is one of the neighbors of sample $\boldsymbol{x}_j$, and $\eta_{ij} = 0$ otherwise. The LMNN method can be now formulated as the following optimization problem $\min._{\boldsymbol{A}} \mathcal{F}_{\boldsymbol{A}} = \sum_{ij} \eta_{ij} d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{ij}) + \mu \sum_{il} \eta_{il}(1 - c_{il})\xi_{ijl}(\boldsymbol{A})$ where $\xi_{ijl}(\boldsymbol{A}) = \max\{0, 1 + d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{ij}) - d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{il})\}$, μ is a constant (we fix $\mu = 1$) and $\boldsymbol{A} \succeq 0$.

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 [4]. To learn the metric which would approximate this ideal geometrical setup a conditional distribution is introduced which for each example $\boldsymbol{x}_i$ selects another example $\boldsymbol{x}_j$ as its neighbor with some probability $p_{\boldsymbol{A}}(j|i)$, and $\boldsymbol{x}_i$ inherits its class label from $\boldsymbol{x}_j$. The probability $p_{\boldsymbol{A}}(j|i)$ is based on the softmin of the $d^2_{\boldsymbol{A}}$ distance measure, i.e. $p_{\boldsymbol{A}}(j|i) = \frac{\exp(-d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{ij}))}{\sum_{k \neq i} \exp(-d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{ik}))}$, $p_{\boldsymbol{A}}(i|i) = 0$. It can be shown [4] that any set of points which has the distribution $p_0(j|i) \propto 1$ if $|\boldsymbol{x}|_{ij} \in \mathcal{S}'$ and $p_0(j|i) = 0$ if $|\boldsymbol{x}|_{ij} \in \mathcal{D}'$ exhibits the desired ideal geometry. It is thus natural to seek a matrix $\boldsymbol{A}$ such that $p_{\boldsymbol{A}}(\cdot|i)$ is as close (in the sense of the Kullback-Leibler divergence) to $p_0(\cdot|i)$. This, after a number of transformations [4], is equivalent to minimizing $\mathcal{F}_{\boldsymbol{A}} = -\sum_{ij}(1 - c_{ij}) \ln(\frac{\exp(-d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{ij}))}{\sum_{k \neq i} \exp(-d^2_{\boldsymbol{A}}(|\boldsymbol{x}|_{ik}))})$ subject to $\boldsymbol{A} \succeq 0$.

[3] An alternative approach is to reduce the dimensionality of the input data to p' ($p' \ll p$); we will also consider this solution in Sect. 4.

NCA. The NCA method 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 with respect to matrix $\boldsymbol{W}$ of (3). Its cost function $\mathcal{F}_{\boldsymbol{W}}$ is based on stochastic neighbor assignments in the weighted feature space, which is based on $p_{\boldsymbol{A}}(j|i)$ defined above where $\boldsymbol{A}$ is replaced with $\boldsymbol{W}$. Under this stochastic selection rule the probability $p_{\boldsymbol{W}}(i)$ of correctly classifying $\boldsymbol{x}_i$ is given by $\sum_j (1-c_{ij})\frac{\exp(-d^2_{\boldsymbol{W}}(|\boldsymbol{x}|_{ij}))}{\sum_{k\neq i}\exp(-d^2_{\boldsymbol{W}}(|\boldsymbol{x}|_{ik}))}$. In this work the actual function to minimize is $\mathcal{F}_{\boldsymbol{W}} = -\sum_i \ln\{\sum_j (1-c_{ij})\frac{\exp(-d^2_{\boldsymbol{W}}(|\boldsymbol{x}|_{ij}))}{\sum_{k\neq i}\exp(-d^2_{\boldsymbol{W}}(|\boldsymbol{x}|_{ik}))}\}$ which expresses the probability of obtaining an error free classification on the training set [3].

3 Scoring Based (Dis-)Similarity Learning

In this section we present the main contribution of this work and define a new framework for (dis-)similarity learning over vectorial data. As already mentioned in Sect. 2, most of the existing metric learning techniques do not require a direct access to the training data; instead, the only ”interface” to the data is through the equivalence sets $\mathcal{S}'$ and $\mathcal{D}'$, the elements of which are the pairwise difference vectors as these were defined in equation (1). Motivated by this observation, we will cast the problem of dissimilarity learning as a problem of learning a binary classification scoring (or regression) function from the training data constructed from $\mathcal{S}'$ and $\mathcal{D}'$. In classification the new labels will be negative for elements of $\mathcal{S}'$ and positive for elements of $\mathcal{D}'$. In regression pairs of instances of different and pairs of instances of the same class will be assigned positive and negative numbers, respectively. More formally, the new training data is given as:

$$\mathcal{I} = \{\bigcup_{i>j}^{n} (|\boldsymbol{x}|_{ij}, c_{ij})\} \cup (\mathbf{0}^T, 0) \tag{6}$$

where c_{ij} is defined in (4) and $\boldsymbol{O}$ denotes a vector of zeros; $(\mathbf{0}^T, 0)$ is included in the new training data as it models for the fact that duplicate instances share the same class. It should be noted that the size of the training dataset from (6) scales as $O(n^2)$ (it contains exactly $\frac{(n-1)(n-2)}{2}+1$ examples). In these settings, the learning phase amounts to training a learning algorithm over the training data $\mathcal{I}$, whereas the computation of the (dis-)similarity between two examples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ boils down to a computing prediction score (for classification) or predicted depended variable (for regression) for an instance $|\boldsymbol{x}|_{ij}$. In the remaining part of this work we will sometimes denote the scoring (or regression) function for $|\boldsymbol{x}|_{ij}$ as $score(\boldsymbol{x}_i, \boldsymbol{x}_j) = score(|\boldsymbol{x}|_{ij})$. The procedure of classifying an instance $\boldsymbol{x}_{new} \in \mathcal{X}$ by the kNN algorithm, that is based on computing $score(|\boldsymbol{x}|_{ij})$ to select nearest neighbours, is presented in Algorithm 1.

It is important to realize that for the definition of c_{ij} from (4), we can interpret the scoring function $score(\boldsymbol{x}_i, \boldsymbol{x}_j)$ as a *dissimilarity measure* since it assigns *lower* values to elements of $\mathcal{S}'$ and *higher* values for elements of $\mathcal{D}'$. However, by redefining c_{ij} from (4) so that it assigns negative labels for elements of $\mathcal{D}'$ and positive labels for elements of $\mathcal{S}'$, the corresponding scoring functions can be interpreted as a *similarity measure*. In this study we will only focus on learning dissimilarity measures.

Algorithm 1. kNN classification using scoring function.

```
$\boldsymbol{kNN\_classify}(\boldsymbol{x}_{new}, \mathcal{X}, \mathcal{Y}, \boldsymbol{score}(\cdot), k))$
// $\boldsymbol{x}_{new}$: an instance to classify
// $\mathcal{X}, \mathcal{Y}$: input data
// $\boldsymbol{score}(\cdot)$: a scoring function learned on (6)
// $k$: number of nearest neighbours
$S \leftarrow \emptyset$
for $i \in 1, \ldots, n$ do
    $|\boldsymbol{x}|^{T}_{new,i} \leftarrow (|x_{new_1} - x_{i_1}|, \ldots, |x_{new_p} - x_{i_p}|)^T$
    $S \leftarrow S \cup \boldsymbol{score}(|\boldsymbol{x}|_{new,i})$
end for
$S \leftarrow \boldsymbol{sort}(S, ascend)$
$N(\boldsymbol{x}_{new}) \leftarrow k$ nearest neighbors of $\boldsymbol{x}_{new}$ according to $S$
$N_c(\boldsymbol{x}_{new}) \leftarrow$ elements of $N(\boldsymbol{x}_{new})$ of class $c$
return $\mathrm{argmax}_{c \in C} \sum_{\boldsymbol{x} \in N_c(\boldsymbol{x}_{new})} \delta(class(\boldsymbol{x}), c)$
```

The proposed framework has several advantages over existing metric learning methods. First, it is very general as we are free to use almost any classification (or regression) algorithm as long as its decision is based on a classification score (the predictions of a regression algorithm could be interpreted right away as dissimilarities). The only restriction we put on the learner is that it should scale well with respect to the number of input instances; this is due to the fact that the number of instances in the new training set $\mathcal{I}$ scales as $O(n^2)$, and hence any algorithm applied in the new space, whose computational complexity is higher than say log-linear would be prohibitive but for toy learning problems. Second, unlike most of the existing techniques, in general no semi-definite programming or eigenvalue computations are required, and hence depending on the employed learner the resulting dissimilarity can be efficiently learned. Finally, our formulation generally leads to a dissimilarity that can be more expressive, and at the same time simpler to learn, than the standard Mahalanobis metrics.

We also mention that there are two main drawbacks of our approach. First, in general the learned dissimilarity measures are not valid metrics.[4] However, several authors have reported state-of-the-art classification performance of kNN over a variety of learning problems where the underlying distance measures were not valid metrics, see e.g. [12, 13]. In particular, most of the examined non-metric distance measures do not satisfy the triangle inequality; the latter guarantees that if a point $\boldsymbol{x}_i$ is close to $\boldsymbol{x}_j$ and close to $\boldsymbol{x}_l$ then $\boldsymbol{x}_j$ will be also close to $\boldsymbol{x}_k$. Moreover, as we will see in the experimental part of this study, the kNN algorithm, where the nearest neighbors are selected using $score(|\boldsymbol{x}|_{ij})$, generally outperforms kNN with adaptive metrics that are learned using different state-of-the-art metric learning techniques. This might suggest that the metric conditions of a dissimilarity function are not necessary for kNN to achieve good predictive performance. However, the dissimilarity measures that are not metrics might be

[4] Technically, a function $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathrm{I\!R}$ is a metric if it is: (i) non-negative ($d(x, y) \geq 0$), (ii) reflexive ($d(x, x) = 0$), (iii) strict ($d(x, y) = 0 \Rightarrow x = y$), (iv) symmetric ($d(x, y) = d(y, x)$) and (v) satisfies triangle inequality ($d(x, z) \leq d(x, y) + d(y, z)$), $\forall_{x,y,z \in \mathcal{X}}$.

not adequate for some applications.[5] We will discuss both the forms and characteristics of our dissimilarities in the remainder of this section.

The second, and potentially more severe, problem is that the learning instances from (6) are not independent. This renders the application of the learning algorithms in the new space questionable, as the basic assumption that training instances should be independently and identically distributed (i.i.d) does not hold. However, the good experimental performance of our framework reported in Sect. 4 suggests that this difficulty is lifted by the above mentioned flexibility of learned dissimilarities. In other words, the advantage of using very flexible dissimilarities might overcome the problem of non-independently distributed data.

In the remainder of this section we will describe 3 different instantiations of our framework that differ with respect to the employed learning algorithm. As already mentioned, the two requirements we put on the learning algorithms applied in the new space are that they should (i) output a function indicating how similar are two instances and (ii) scale well with respect to the number of instances in $\mathcal{I}$. To perform a fair comparison with the existing metric learning techniques we will only focus on algorithms that produce linear models whose parameters are directly related with the parameters of the Mahalanobis metric. Based on the above considerations, in this study we focused on two classification (Linear Support Vector Machines and Logistic Regression) and one regression algorithm (Ridge Regression) which fulfil the above requirements.

Support Vectors Machines. In Linear Support Vector Machines (L-SVM) the learning phase amounts to solving the following unconstrained quadratic optimization problem [15]:[6] $\min._{\boldsymbol{w}\in\mathbb{R}^p} \sum_{ij} \max\{0, 1 - c_{ij}\langle \boldsymbol{w}, |\boldsymbol{x}|_{ij}\rangle\} + \lambda\|\boldsymbol{w}\|^2$, where λ is a user-defined regularization parameter and $i = 1, \ldots, n; j = i, \ldots, n$. The dissimilarity between $\boldsymbol{x}_i, \boldsymbol{x}_j$ is given as: $score(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{w}^*, |\boldsymbol{x}|_{ij}\rangle$, where $\boldsymbol{w}^*$ is a solution of the optimization problem of SVM. It is easy to verify that the above function is not a metric as it can take negative values and does not satisfy the triangle inequality, however, it is reflexive, strict (assuming a non-degenerate $\boldsymbol{w}^*$) and symmetric. For solving the optimization problem of L-SVM we exploit the algorithm recently proposed in [15] that scales linearly with respect to the number of instances in $\mathcal{I}$, i.e. its computational complexity is $O(n^2)$. It is worth noting that the method from [5] also exploits the notion of SVM to learn a metric. The main difference from our framework is that this method is a *local* one that aims to determine a stretched neighbourhood around each query instance such that class conditional probabilities are likely to be constant. Moreover, [5] exploits the softmax function to obtain a valid metric. We plan to perform a detailed comparison of the two approaches in the future.

Logistic Regression. Logistic Regression (LR) [1] is a well known binary classification method where the classification decisions are based on a scoring function $score(|\boldsymbol{x}|_{ij})$ that can be interpreted as a probability p_{ij} that an instance $|\boldsymbol{x}|_{ij}$ belongs to the positive class. More formally, $score(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is modelled as: $score(\boldsymbol{x}_i,\boldsymbol{x}_j) \equiv p_{ij} = \frac{\exp(\langle \boldsymbol{w}, |\boldsymbol{x}|_{ij}\rangle)}{1+\exp(\langle \boldsymbol{w}, |\boldsymbol{x}|_{ij}\rangle)}$,

[5] We also note that the metric conditions are crucial if one employs efficient nearest neighbour search strategies that are based on storing training examples in hierarchical data structures [14].

[6] As we will exploit the learned weights only to compute a scoring function we do not include in L-SVM (and in Logistic Regression) the bias term b.

where $\boldsymbol{w} \in \mathbb{R}^p$ are parameters of the logistic regression model; $score(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is non-negative and symmetric, but does not satisfy the triangle inequality and is neither strict nor reflexive. We will exploit the regularized version of LR where the optimal solution $\boldsymbol{w}^*$ is obtained by solving the following optimization problem: $\min_{\cdot \boldsymbol{w} \in \mathbb{R}^p} \; -\sum_{ij}(1 - c_{ij}) \ln(\frac{\exp(\langle \boldsymbol{w}, |\boldsymbol{x}|_{ij} \rangle)}{1+\exp(\langle \boldsymbol{w}, |\boldsymbol{x}|_{ij} \rangle)}) + \lambda \|\boldsymbol{w}\|^2$.

Ridge Regression. We also experimented with one regression method, namely the Ridge Regression (RR) algorithm [1] that solves the following optimization problem: $\min_{\cdot \boldsymbol{w} \in \mathbb{R}^p} \; \sum_{ij}(\langle \boldsymbol{w}, |\boldsymbol{x}|_{ij} \rangle - c_{ij})^2 + \lambda \|\boldsymbol{w}\|^2$. In this context, the dissimilarity function has an identical form (and properties) as in the case of L-SVM $score(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{w}^*, |\boldsymbol{x}|_{ij} \rangle$, where $\boldsymbol{w}^* \in \mathbb{R}^p$ is a solution of the optimization problem.

4 Experiments

We evaluated the performance of the proposed approach on a number of real-world classification problems. The goal is to examine whether the three instantiations of our dissimilarity learning framework from Sect. 3 (i.e. the L-SVM, LR and RL linear algorithms) achieve better predictive performance than a number of existing metric learning algorithms. The quality of the different dissimilarity measures will be only compared in the context of kNN (we will not use the underlying algorithms such as logistic regression directly for classification).

We compared the above 3 methods with the LMNN, MCML and NCA state-of-the-art metric learning algorithms. We experimented with 2 instantiations of the above metric learning techniques, over full and diagonal matrices denoted respectively as $\mathrm{METHOD}_{\mathrm{full}}$ and $\mathrm{METHOD}_{\mathrm{diag}}$, where METHOD is LMNN, MCML or NCA. For comparison reasons we also provide the performance of the standard kNN algorithm with the Euclidean metric. We experimented with different values of k ($k = 1, 3, 10$); the relative performance of the different methods did not vary with k, we report results only for $k = 1$. In all the above methods we set the λ parameter to 1. In all the experiments we estimate accuracy using 10-fold cross-validation and control for the statistical significance of observed differences using McNemar's test [16] (sig. level of 0.05).

We experimented with 13 datasets. First, we used 4 standard datasets from the UCI repository (Liver, Wdbc, Wine, BalanceScale); these datasets are standard benchmarks used in the context of distance learning. Then, we have chosen to experiment with high-dimensional data from two different application domains, namely genomics and proteomics (description of these datasets can be found in [17]). The genomics datasets correspond to DNA-microarray experiments. We worked with three different datasets: colon cancer (Colon), central nervous system (Central) and Leukemia (Leuk). All proteomics datasets come from the domain of mass spectrometry. We worked with 4 different datasets: ovarian cancer (Ovarian), prostate cancer (Prostate), an early stroke diagnosis (Stroke), and MaleFemale (MaleF). In Ovarian, Prostate and Stroke we experimented with 2 versions of each of the above proteomics datasets where we used different parameters in the prepossessing step for feature extraction (in MaleF we had access only to one version of this dataset). All features correspond to intensities of mass values and are continuous. All the above genomics and proteomics datasets, in addition

Table 1. Accuracy and significance test results of kNN (with and without distance learning). The numbers in parentheses indicate the number of significance points that the algorithm scores in a given dataset; the algorithms with the highest score for each dataset are highlighted. n and p denote respectively the number of instances and the dimensionality of the data.

Dataset	n	p	$\text{LMNN}_{\text{diag}}$	$\text{MCML}_{\text{diag}}$	NCA_{diag}	L-SVM	LR	RR	kNN
Liver	345	6	58.6 (3.5)	**62.4 (4.0)**	54.4 (2.0)	57.5 (3.5)	58.1 (3.5)	47.7 (0.5)	**62.4 (4.0)**
Wdbc	569	30	**94.8 (3.5)**	**94.8 (3.5)**	**95.0 (3.5)**	**93.4 (3.5)**	84.5 (1.0)	71.6 (2.5)	95.5 (3.5)
Wine	178	13	**96.7 (4.5)**	94.9 (4.0)	91.7 (3.0)	**97.4 (4.5)**	73.3 (1.0)	55.2 (0.0)	94.9 (4.0)
Balance	625	4	77.7 (3.0)	77.4 (3.0)	75.5 (2.5)	76.0 (2.5)	76.2 (2.5)	76.1 (3.0)	**79.4 (4.5)**
Colon	62	2000	82.1 (3.0)	80.4 (3.0)	77.5 (2.5)	85.4 (3.5)	85.4 (3.5)	**87.1 (4.0)**	73.8 (1.5)
Central	60	7129	60.0 (3.0)	56.7 (3.0)	48.3 (2.0)	**66.7 (3.5)**	**70.0 (3.5)**	61.7 (3.0)	56.7 (3.0)
Leuk	72	7129	**97.1 (4.0)**	**97.1 (4.0)**	87.5 (1.0)	95.7 (3.5)	**97.1 (4.0)**	**97.1 (4.0)**	84.9 (0.5)
MaleF	134	1524	77.6 (4.0)	68.8 (2.5)	55.7 (0.5)	**84.2 (4.5)**	**83.4 (4.5)**	**81.9 (4.5)**	55.5 (0.5)
Ovarian1	253	385	**96.8 (4.5)**	93.0 (1.0)	91.5 (1.0)	**97.6 (4.5)**	**98.0 (4.5)**	**96.8 (4.5)**	92.2 (1.0)
Ovarian2	253	10361	95.7 (4.5)	86.6 (1.0)	88.0 (1.0)	95.7 (4.5)	95.0 (3.5)	**97.6 (5.0)**	90.6 (1.5)
Stroke1	208	172	**68.6 (4.0)**	63.9 (3.0)	58.1 (2.5)	60.4 (2.5)	63.9 (3.0)	60.7 (3.0)	64.1 (3.0)
Stroke2	208	2810	**69.0 (4.0)**	60.1 (1.5)	63.5 (3.0)	**70.7 (4.0)**	68.7 (3.5)	**70.1 (4.0)**	55.5 (1.0)
Prostate1	322	390	85.8 (5.0)	78.3 (2.0)	70.8 (0.5)	**86.1 (5.5)**	82.6 (3.5)	78.6 (2.0)	82.4 (2.5)
Prostate2	322	12600	90.3 (3.5)	83.3 (1.0)	85.0 (2.5)	92.3 (4.0)	92.0 (4.0)	**94.2 (4.5)**	82.2 (1.5)
Average rank			3.86	2.61	1.96	3.86	3.29	3.14	2.29

to large number of features, are also characterized by a small number of observations, making these datasets a difficult learning scenario. In all the above datasets the numeric attributes were normalized so that they takes values between 0 and 1. In Table 1 we provide the number of instances and attributes in the examined datasets.

To better understand the relative performances of the examined algorithms we established a ranking schema of these algorithms based on the results of the pairwise comparisons. More precisely, if an algorithm is significantly better than another it is credited with 1 point; if there is no significant difference between two algorithms then they are credited with 0.5 points; if an algorithm is significantly worse than another it is credited with 0 point. Thus, in the case m algorithms are examined, an algorithm that is significantly better than all the others for a given dataset is assigned a score of $m - 1$.

Experiments on these datasets have 2 goals. First, we study the relative performance of our methods with the existing metric learning algorithms. In these experiments we use the diagonal versions of the existing metric learning techniques as it allows for a fair comparison with the proposed framework; similar to the metric learning techniques based on diagonal matrices, L-SVM, LR and RR do not account for interactions between different attributes. Second, we compare the predictive performance of our method with the metric learning methods based on full matrices. For the latter methods applied on the high-dimensional datasets (i.e. all the genomics and proteomics classification problems) we exploited the PCA method to reduce the data dimensionality; this procedure was widely used in the context of metric learning [2]. More precisely, the training instances are projected into a lower dimensional subspace accounting for at least 99 $\%$ of the data's total variance.

Results and Analysis. In Table 1 we present the results (with the score ranks) of the comparison between $\mathrm{LMNN}_{\mathrm{diag}}$, $\mathrm{MCML}_{\mathrm{diag}}$, $\mathrm{NCA}_{\mathrm{diag}}$, L-SVM, LR, RR and the standard kNN (recall that the maximum score for an algorithm in a given dataset is 6). From these results we can make several observations. First, with the exception of the Liver and Balance datasets, there is at least one adaptive method that outperforms the standard kNN algorithm (in Liver and Balance, kNN is placed in the first position according to our ranks). Second, the developed dissimilarity methods based on L-SVM, LR and RR generally outperform both $\mathrm{MCML}_{\mathrm{diag}}$ and $\mathrm{NCA}_{\mathrm{diag}}$; the advantage of our methods is most visible in the genomics and proteomics high dimensional datasets. Finally, the methods that most often win are $\mathrm{LMNN}_{\mathrm{diag}}$, L-SVM, LR and RR.

To quantify the relative performances of the different algorithms we computed for each method its average rank over all the examined datasets. These ranks are presented in the last row of Table 1. We observe that the best performance of 3.86 points is obtained for the margin based methods (L-SVM and $\mathrm{LMNN}_{\mathrm{diag}}$), which have a similar computational complexity both in theory (they scale as $O(n^2)$) and in practice (they had similar running times). This result is remarkable since L-SVM, which is based on a simple idea, performs equally well as the more elaborate metric learning algorithm that has been reported to consistently outperform other metric learning techniques over a number of of non-trivial learning problems [2]. Finally, we mention that the surprisingly poor performance of $\mathrm{NCA}_{\mathrm{diag}}$ might be explained by the fact that its cost function is not convex and hence it is sensitive to the initializations of $\boldsymbol{W}$.

In the second set of experiments we compare the performance of the metric learning methods with the full matrix to that of metric learning with diagonal matrix. Here we want to examine whether methods that account for feature interaction outperform the methods proposed in Sect. 3. As already mentioned, we first reduced the dimensionality of all the high-dimensional datasets by mapping the instances to lower dimensional subspaces defined by the PCA method; depending on the datasets, the dimensionalities of the subspaces, that account for at least 99 $\%$ of the data's total variance, were between 50 and 178. The results are presented in Table 2. From these results we can see that with the exception of $\mathrm{NCA}_{\mathrm{full}}$ in Liver and Prostate1, all the metric learning with full matrix have similar or worse performances than their corresponding versions with diagonal matrices (the first signs in parenthesis are mostly ”=” or ”-”). This could suggest that even though the data dimensionality is significantly reduced, the metric learning algorithms based on full matrices might still be prone to overfitting (the other explanation might be simply that by removing features that have low variance we also remove important discriminatory information).

We have also compared the performances of full matrix metric learning methods with that of L-SVM, LR and RR; the significance tests results corresponding to this comparison are given by the 2nd, 3th and 4th signs in parenthesis in Table 2. From these results we can see that the relative performances between metric learning methods that are based on full matrices and the methods from Sect. 3 depend on the actual dataset. Indeed, in the first 3 UCI datasets (Liver, Wdbc and Wine) there is an advantage of full matrix metric learning algorithm; in Balance, Central and Stroke1 there is almost no difference in performances; in Prostate1 no conclusions can be drawn; and in all the remaining datasets the L-SVM, LR and RR methods generally outperform

Table 2. Accuracy and significance test results of the algorithms based on full matrices. The 4 signs in parenthesis correspond to a comparison between $\mathrm{METHOD}_{\mathrm{full}}$ and $\mathrm{METHOD}_{\mathrm{diag}}$, L-SVM, LR and RR, respectively (the "+" sign stands for a significant win of a the first algorithm in a pair, "-" for a significant loss, and "=" for no significant difference).

Dataset	$\mathrm{LMNN}_{\mathrm{full}}$	$\mathrm{MCML}_{\mathrm{full}}$	$\mathrm{NCA}_{\mathrm{full}}$	Dataset	$\mathrm{LMNN}_{\mathrm{full}}$	$\mathrm{MCML}_{\mathrm{full}}$	$\mathrm{NCA}_{\mathrm{full}}$
Liver	55.4 (====)	61.5 (===+)	62.4 (+==+)	MaleF	67.4 (- - - -)	60.3 (= - - -)	57.1 (=- - -)
Wdbc	94.8 (==+=)	94.8 (==+=)	95.5 (==+=)	Ovarian1	93.4 (- - - =)	87.8 (- - - -)	91.8 (=- - -)
Wine	95.5 (==++)	96.8 (==++)	94.9 (==++)	Ovarian2	97.2 (====)	74.7 (- - - -)	90.5 (=- =-)
Balance	77.6 (====)	77.4 (====)	76.1 (====)	Stroke1	66.1 (====)	57.6 (====)	65.1 (====)
Colon	66.7 (- - - -)	65.0 (- - - -)	72.1 (=- - -)	Stroke2	58.7 (- - - -)	55.5 (=- - -)	56.4 (=- - -)
Central	60.0 (====)	58.3 (====)	55.0 (==- =)	Prostate1	83.6 (==+=)	78.3 (=- - =)	83.0 (+=+=)
Leuk	83.2 (- - - -)	83.2 (- - - -)	84.9 (=- - -)	Prostate2	84.2 (=- - -)	72.5 (- - - -)	82.2 (=- - -)

the metric learning techniques based on full matrices. These results suggest that in the majority of the high dimensional datasets, the feature interactions are not important, and hence the methods that do not account for feature interactions have in general better performances. Alternatively, it might suggest that stronger regularization is needed. Moreover, it is interesting to note that the cases for which the full matrix metric learning methods are good are mostly the UCI datasets that correspond to rather not difficult classification problems. This hints that there might be a bias of method development towards methods that perform well on UCI datasets; however, one can argue that they are really representative of the real world.

5 Conclusions and Future Work

In this paper we prosed a novel framework for learning a (dis-)similarity function over vectorial data, where the learning problem is cast as a binary classification or regression task defined over the space of pairwise differences of the input instances. Our approach generally leads to adaptive (dis-)similarities that are not valid metrics; however, we argue that by learning (dis-)similarities that do not fulfil metric conditions (and are not of the Mahalanobis type) we might have more freedom in adapting these (dis-)similarities for a given problem. Our claim is supported by experimental evidence that, in terms of predictive performance, shows that our framework compares favourably with alternative state-of-the-art distance learning techniques which are trained to learn both full and diagonal Mahalanobis metrics.

In the future we want to exploit other learning techniques applied over the new data representation (e.g. decision trees). We also plan to investigate a Non-Linear version of Support Vectors Machines (NL-SVM) to learn the higher-order interaction between features, similar to that modelled in the Mahalanobis metric. This can be achieved by exploiting the polynomial kernel of degree 2 that induces a feature space that is indexed by all the monomials of degree 2. Since we will not be able to rely on efficient implementation of L-SVM, the main challenge here is to make NL-SVM scalable and practical. Moreover, we plan to test more carefully the pros and cons of the L-SVM and

LMNN methods that in our experiments had similar performance and outperformed other algorithms. Finally, we would like to compare our framework with the approach from [5] as the latter also exploits the notion of SVM to locally learn a metric.

Acknowledgments. This work was partially funded by the European Commission through EU projects DebugIT (FP7-217139) and e-LICO (FP7-231519). The support of the Swiss NSF (Grant 200021-122283/1) is also gratefully acknowledged.

References

1. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)
2. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. 10, 207–244 (2009)
3. Goldberger, J., Roweis, S., Hinton, G., Salakhutdinov, R.: Neighbourhood component analysis. In: NIPS. MIT Press, Cambridge (2005)
4. Globerson, A., Roweis, S.: Metric learning by collapsing classes. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) NIPS, vol. 18, pp. 451–458. MIT Press, Cambridge (2006)
5. Domeniconi, C., Gunopulos, D.: Adaptive nearest neighbor classification using support vector machines. In: NIPS, vol. 14. MIT Press, Cambridge (2002)
6. Davis, J., Kulis, B., Jain, P., Sra, S., Dhillon, I.: Information-theoretic metric learning. In: Proc. 24th International Conference on Machine Learning, ICML (2007)
7. Hastie, T., Tibshirani, R.: Discriminant adaptive nearest neighbor classification and regression. In: NIPS, vol. 8 (1996)
8. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning with application to clustering with side-information. In: NIPS, vol. 15, pp. 505–512. MIT Press, Cambridge (2003)
9. Hertz, T., Bar-Hillel, A., Weinshall, D.: Boosting margin based distance functions for clustering. In: ICML'04, p. 50. ACM Press, New York (2004)
10. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Advances in Neural Information Processing Systems, vol. 16. MIT Press, Cambridge (2004)
11. Weinberger, K.Q., Saul, L.K.: Fast solvers and efficient implementations for distance metric learning. In: International Conference on Machine Learning, ICML (2008)
12. Woźnica, A., Kalousis, A., Hilario, M.: Distances and (indefinite) kernels for sets of objects. In: The IEEE International Conference on Data Mining (ICDM), Hong Kong (2006)
13. Horvath, T., Wrobel, S., Bohnebeck, U.: Relational instance-based learning with lists and terms. Machine Learning 43(1/2), 53–80 (2001)
14. Liu, T., Moore, A.W., Gray, A.: New algorithms for efficient high-dimensional nonparametric classification. J. Mach. Learn. Res. 7, 1135–1158 (2006)
15. Franc, V., Sonnenburg, S.: Optimized cutting plane algorithm for support vector machines. In: ICML '08: Proceedings of the 25th international conference on Machine learning (2008)
16. McNemar, Q.: Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika 12, 153–157 (1947)
17. Kalousis, A., Prados, J., Hilario, M.: Stability of feature selection algorithms: a study on high-dimensional spaces. Knowledge and Information Systems 12, 95–116 (2006)