xxx

Multi-Instance Kernels

Thomas Gärtner THOMAS.GAERTNER@AIS.FRAUNHOFER.DE
Knowledge Discovery Team, Fraunhofer Institut Autonome Intelligente Systeme, Germany;
Department of Computer Science, University of Bristol, United Kingdom

Peter A. Flach PETER.FLACH@BRISTOL.AC.UK
Department of Computer Science, University of Bristol, Bristol, United Kingdom

Adam Kowalczyk ADAM@KERNEL-MACHINES.ORG
Telstra Research Laboratories, 770 Blackburn Road, Clayton, VIC 3168, Australia

Alex J. Smola ALEX.SMOLA@ANU.EDU.AU
RSISE, The Australian National University, Canberra, 0200 ACT, Australia

Abstract

Learning from structured data is becoming increasingly important. However, most prior work on kernel methods has focused on learning from attribute-value data. Only recently, research started investigating kernels for structured data. This paper considers kernels for multi-instance problems - a class of concepts on individuals represented by sets. The main result of this paper is a kernel on multi-instance data that can be shown to separate positive and negative sets under natural assumptions. This kernel compares favorably with state of the art multi-instance learning algorithms in an empirical study. Finally, we give some concluding remarks and propose future work that might further improve the results.

1. Introduction

Support vector machines (SVM) and other kernel methods (Boser et al., 1992; Schölkopf & Smola, 2002) have successfully been applied to various tasks in attribute-value learning. Most 'real-world' data, however, has no natural representation as a tuple of constants. Defining kernels on individuals that can not easily be described by a single feature vector means crossing the boundary between attribute-value and relational learning. It allows kernel methods to be applied more easily to complex representation spaces.

Multi-instance (MI) learning problems (Dietterich et al., 1997) occur whenever example objects, individuals, can not be described by a single characteristic feature vector, but by a bag of vectors. Any of these vectors could be responsible for the classification of the bag. The inherent difficulty of MI problems is to identify the characteristic element of each bag. Extending kernel methods to MI problems is one step towards crossing the boundary between attribute-value and relational learning.

The main result of this paper is a kernel on MI data that can be shown to separate positive and negative sets under natural assumptions. Empirical studies compare this kernel to other MI learning algorithms. On one hand, an SVM using this kernel is compared to state of the art MI algorithms. On the other hand, it is compared to other methods that can be used to apply SVMs to MI data. In both studies the kernel proposed in this paper compares favorably.

Section 2 introduces the MI setting under various aspects. Section 3 summarizes prior work on kernels for discrete spaces, in particular the work of Haussler (1999) and Gärtner (2000). Section 4 gives an account of the separability of MI problems in kernel feature space and devises a suitable kernel. Section 5 discusses an alternative way to apply attribute-value learning algorithms to MI problems. After that we empirically evaluate our kernel on the musk-dataset, and conclude with some final remarks.

2. The Multi-Instance Setting

MI problems have been introduced under this name by Dietterich et al. (1997). However, similar problems and algorithms have been considered earlier, for example in pattern recognition (Keeler et al., 1991). Within the last couple of years, several approaches have been made to upgrade attribute-value learning algorithms to tackle MI problems. Other approaches focused on new algorithms specifically designed for MI learning.

Formally, concepts are functions $\nu_I : \mathcal{X} \to \Omega$, where $\mathcal{X}$ is often referred to as the instance space or problem domain (examples are elements of this domain) and $\Omega = \{\top, \bot\}$ are the labels (in what follows Ω always means $\{\top, \bot\}$). There are $2^{|\mathcal{X}|}$ concepts on the instance space $\mathcal{X}$. A function $f : \mathcal{X} \to \mathbb{R}$ is said to separate the concept if $f(x) > 0 \Leftrightarrow \nu_I(x)$.

If examples are represented by subsets of some domain $\mathcal{X}$ concepts are functions $\nu_{\text{set}} : 2^{\mathcal{X}} \to \Omega$. There are $2^{2^{|\mathcal{X}|}}$ different concepts on sets (in other words: $\nu_{\text{set}} \equiv 1_{\mathcal{C}}, \mathcal{C} \subseteq \mathcal{X}$). Such concepts are sometimes referred to as multi-part concepts. MI concepts are a specific kind of these concepts.

Definition 2.1 *An MI concept is a function $\nu_{\text{MI}} : 2^{\mathcal{X}} \to \Omega$. It is defined as:*

$$\nu_{\text{MI}}(X) \Leftrightarrow \exists\, x \in X : \nu_I(x)$$

where ν_I is a concept over an instance space (referred to as the 'underlying concept'), and $X \subseteq \mathcal{X}$ is a set.

There are $2^{|\mathcal{X}|}$ different MI concepts. The difficulty in this task is not just to generalize beyond examples, but also identifying the characteristic element of each bag. Any learning algorithm that sees a positive bag (a bag with label $\top$) cannot infer much about the elements of the bag, except that one of its elements is positive in the underlying concept. With large bag sizes this information is of limited use.

A popular real-world example of an MI problem is the prediction of drug activity, described by Dietterich et al. (1997). A drug is active if it binds well to enzymes or cell-surface receptors. The binding strength is determined by the shape of the drug molecule. However, most molecules can change their shape by rotating some of their internal bonds. The possible shapes of a molecule, i.e., a combination of the angles of the rotatable bonds of the molecule, are known as conformations. A drug binds well to enzymes or cell-surface receptors if one of its conformations binds well. Thus the drug activity prediction problem is an MI problem.

Apart from approaches that ignore the MI setting during training, two major categories of approaches can be distinguished: Upgrading existing attribute-value learning algorithms, and designing a new learning algorithm specifically for MI problems. Algorithms that have specifically been designed for MI problems are: The axis-parallel rectangles (APR) algorithm and variants (Dietterich et al., 1997), an algorithm based on simple statistics of the bags (Auer, 1997), and algorithms based on the diverse density approach (Maron & Lozano-Pérez, 1998; Zhang & Goldman, 2002). Algorithms that have been upgraded until now are: the lazy learning algorithms Bayesian-kNN and Citation-kNN (Wang & Zucker, 2000), the neural network MI-NN (Ramon & De Raedt, 2000), the decision tree learner RELIC (Ruffo, 2001), and the rule learner (Naive)RipperMI (Chevaleyre & Zucker, 2001). Inductive logic programming algorithms have also been used, for instance, the first-order decision tree learner TILDE (Blockeel & De Raedt, 1998).

Of the aforementioned approaches, the APR algorithm is the 'classical' approach for MI learning. It assumes that the features of the conformations are independent and thus orthogonal. We then seek an axis-aligned (hyper-) rectangle covering at least one element of each positive bag and none of a negative bag.

3. Kernels on Discrete Spaces

In SVMs and other kernel methods the only information needed about instances is an inner product in some feature space - a positive definite *Mercer kernel* $k(.,.) = \langle \phi(.), \phi(.) \rangle$, where $\langle .,. \rangle$ denotes the inner product and ϕ is a feature transformation. Below we summarize prior work on kernels for discrete spaces that is most relevant in our context. For brevity, below *a kernel* always means a Mercer kernel.

Convolution Kernels The best known kernel for representation spaces that are not mere attribute value tuples, is the convolution kernel proposed by Haussler (1999). It is defined as:

$$k_{conv}(x,y) = \sum_{\vec{x} \in R^{-1}(x), \vec{y} \in R^{-1}(y)} \prod_{d=1}^{D} k_d(x_d, y_d)$$

where R is a relation between instances z and their parts $\vec{z}$, i.e., R^{-1} decomposes an instance into a set of D-tuples. z_d denotes the d-th component of $\vec{z}$, and k_d is a valid kernel function on this component.

The term 'convolution kernel' refers to the class of kernels given by the above definition. The advantage of convolution kernels is that they are very general and can be applied in many different problems. However, because of that generality they require a significant amount of work to adapt them to a specific problem, which makes choosing R a non-trivial task.

More specific kernels on discrete spaces can be found in Gärtner (2000). There, kernels for elementary symbols (k_δ - the *matching* kernel), sets and multi-sets of elementary symbols (k_{set} and k_{multiset}), and Boolean domains (k_Ω - and a polynomial variant) are discussed along with concept classes that can be separated by linear classifiers using one of these kernels. The specific kernels are described in more detail below.

3.1 Set Kernels

A kernel on sets can easily be derived form the definition of convolution kernels by letting R be the set-membership function, i.e., with $\vec{z} \in R^{-1}(Z) \Leftrightarrow \vec{z} \in Z, D = 1$ and for notational convenience $z_1 = z, k_1 = \kappa$ we obtain:

$$k_{\text{set}}(X, X') := \sum_{x \in X, x' \in X'} k_{\mathcal{X}}(x, x') \qquad (1)$$

where κ is a kernel on $\mathcal{X}$, and $X, X' \subseteq \mathcal{X}$. In particular, Haussler proved the following theorem:

Proposition 3.1 *A function k on sets defined by (1) is a kernel if and only if κ itself is a kernel.*

For instance, if $x, y \in \mathcal{X}$ are elementary symbols (constants - no internal structure) a natural choice for a kernel is the matching kernel, defined as:

$$k_\delta(x, y) = \delta_{x,y}$$

Since this kernel induces a Hilbert space on $\ell_2(\mathcal{X})$, assuming $\mathcal{X}$ is countable, all possible concepts $\nu(x) \Leftrightarrow x \in \mathcal{C}, \mathcal{C} \subseteq \mathcal{X}$ on this representation space can be separated using the matching kernel. Clearly for $\kappa = k_\delta$

$$k_{\text{set}}(X, Y) = \sum_{x \in X, y \in Y} k_\delta(x, y) = |X \cap Y|.$$

3.2 Cosets

Next we extend the summation operation over sets to more general structures on $\mathcal{X}$ by using the notion of cosets. Assume that there exists a set $\mathcal{Y}$ (the coset) associated with $\mathcal{X}$. Then we can represent sets and their generalizations as mappings $\mathcal{X} \to \mathcal{Y}$. The following examples explain the use of this definition:

Sets: here $\mathcal{Y} = \{0, 1\}$ and the map $X : x \to y$ indicates whether $x \in X$ (in which case $y = 1$) or not.

Multisets: setting $\mathcal{Y} = \mathbb{N} \cup \{0\}$ allows us to extend the notion of sets to multisets, i.e., sets X which may contain elements several times. Here $y = X(x)$ denotes the number of elements x contained in X, and, as before, $y = 0$ indicates that $x \notin X$.

Measures: we may extend the notion of cosets to general measures on $\mathcal{X}$ by letting $\mathcal{Y} = [0, \infty)$. Such a definition would also allow us to cater for situations where class membership is described in more vague terms, as common in Fuzzy sets (here $\mathcal{Y} = [0, 1]$).

Now that we extended our definition of sets, we need to define a kernel on it. In general, any kernel on the space of functions $X : \mathcal{X} \to \mathcal{Y}$ will satisfy the formal requirements. For practical purposes, however, we consider a class of kernels given by

$$k_{\text{set}}(X, X') = \sum_{x \in \mathcal{X}, x' \in \mathcal{X}} k_{\mathcal{X}}(x, x') k_{\mathcal{Y}}(X(x), X'(x')). \qquad (2)$$

A special case is the situation where $k_{\mathcal{Y}}(X(x), X'(x')) = X(x)X'(x')$. It is easy to check that (2) reduces to the kernel defined in (1) in the case of sets and in the case of multisets to a kernel where the summation ranges over all elements of the multiset.

3.3 Normalization

It is useful to note that (1) corresponds to an averaging process which is carried out *in feature space*, or in other words, given a kernel on all elements $x_{ij} \in X_i$ on all sets X_i, the kernel matrix on X_i is obtained by summing up corresponding rows and columns of a larger matrix.

Such a procedure indicates that if the cardinalities of X_i vary considerably, sets with a large cardinality will dominate the solution of the estimation problem (as can be observed in the experiments on the Musk dataset). This is not always desirable, which leads us to the issue of normalization. A natural definition in this regard is

$$k(X, X') := \frac{k_{\text{set}}(X, X')}{f_{\text{norm}}(X) f_{\text{norm}}(X')} \qquad (3)$$

where $f_{\text{norm}}(X)$ is a suitable normalization function which is nonnegative for any $k_{\text{set}}(X, X') \neq 0$. Below we see that various choices of $f_{\text{norm}}(X)$ will lead to feature-space normalization, sample normalization, variance rescaling, etc.

Feature-space normalization: We simply set

$$f_{\text{norm}}(X) := \sqrt{k_{\text{set}}(X, X)}. \qquad (4)$$

Thus we recover the normalization proposed by (Herbrich, 2002), which proves to be very useful in MI learning.

Averaging: In this case we need to compute the generalized cardinality of the set X, which can be achieved via

$$f_{\text{norm}}(X) := \sum_{x \in X} X(x). \qquad (5)$$

For simple sets this just means that we average the instances mapped into feature space. A minor modification is to use $\sqrt{f_{\text{norm}}(X)}$ instead, which will be more useful if most of the averages are close to 0.

4. Separating MI Problems

In this section we define a kernel function on sets of instances that separates MI problems under natural assumptions. Using this kernel function, SVMs and other kernel methods can easily be applied to MI problems. We begin by introducing the notion of separability.

Separability A concept $\nu_I : \mathcal{X} \to \Omega$ is called linearly separable in input space if

$$\forall\, x \in \mathcal{X} : \langle x, c \rangle \geq \theta \Leftrightarrow \nu_I(x)$$

holds for some constant c and threshold θ. Likewise, a concept is called linearly separable with respect to a feature map ϕ if

$$\forall\, x \in \mathcal{X} : \langle \phi(x), c_\phi \rangle \geq \theta \Leftrightarrow \nu_I(x)$$

holds for some $c_\phi \in \operatorname{span}\{\phi(x) | x \in \mathcal{X}\}$. In what follows we often use the term *separable* to refer to concepts that are linearly separable with respect to a feature transformation given implicitly by a kernel.

For our purposes we further need the notion of a concept that is separable with non-zero margin and some lower bound.

Definition 4.1 *We call a concept separable with margin $\epsilon > 0$ with respect to the feature map ϕ, if there exist c_ϕ, b such that for every $x \in \mathcal{X}$:*

$$\begin{array}{rclcl} \nu_I(x) & \Leftrightarrow & \langle \phi(x), c_\phi \rangle + b & \geq & 1 \\ \overline{\nu_I(x)} & \Leftrightarrow & 0 \leq \langle \phi(x), c_\phi \rangle + b & \leq & 1 - \epsilon. \end{array} \tag{6}$$

Note that we can, without loss of generality, assume that $b = 0$, since for given ϕ, c_ϕ, b the map $x \to (\phi(x), 1)$ together with (c_ϕ, b) will satisfy the above condition without a constant offset. Furthermore, any separable concept on a finite domain (or example set) is separable with non-zero margin and some lower bound on this domain (or example set).

We will now define a kernel k_{MI} that separates MI concepts. This kernel is a variant of the set kernel, as defined in (1). It will be useful in proving necessary and sufficient conditions for separability of MI problems.

$$k_{\text{MI}}(X, Y) = \sum_{x \in X, y \in Y} k_{\text{I}}^p(x, y) \tag{7}$$

where $p \in \mathbb{N}$ is a constant. Since products of kernels are kernels, also $k_I^p(x, y)$ is a kernel, and consequently also $k_{\text{MI}}(X, Y)$ is a kernel.

In order to show that MI concepts are separable if and only if the underlying concept is separable, we need the following two lemmas.

Lemma 4.2 *An MI concept ν_{MI} is separable with the kernel k_{MI}, as defined in (7), for sufficiently large p, if the underlying concept ν_{I} is separable with the kernel k_{I} and some ϵ, according to (6).*

Proof: Consider first the case that $\nu_I = 1_{\mathcal{X}}$, i.e., each instance satisfies the concept. Then it follows trivially from the definition of MI concepts that $\nu_{\text{MI}} = 1_{2^{\mathcal{X}}}$. For the remainder of the proof we can thus assume that at least one instance does not satisfy the concept.

Let now $p > 0$ if $\epsilon = 1$, and $p > -\frac{\log m}{\log(1-\epsilon)}$ otherwise. Here m is a bound on the cardinality of the bags X.

Since the concept is separable, there exists a c_ϕ satisfying (6). Now consider the function

$$f(X) = \sum_{x \in X} \langle \phi(x), c_\phi \rangle^p. \tag{8}$$

One can see that $f(x) < 1$ if and only if no $x \in X$ satisfies ν_I: in this case, we have

$$f(X) \leq m(1 - \epsilon)^p < m(1 - \epsilon)^{-\frac{\log m}{\log(1-\epsilon)}} = 1$$

The final step consists of showing that f can be written as a dot product in the feature space induced by k_{MI}. Clearly $f(\{x\})$ is a function in the space of dot products induced by $k(x, x')^p$. Next note that k_{MI} is the space of linear combinations of such functions on X, and clearly f, being a "monomial", is one of such linear combinations. $\square$

Example 4.1 *Let $\mathcal{X} = \Omega^4 = \{\top, \bot\}^4$, $x \in \mathcal{X}$, $x = (x_1, x_2, x_3, x_4)$, $\nu_I(x) \Leftrightarrow x_2 \wedge x_4$, and*

$$k_I(x, y) = k_\Omega(x, y) = \sum_{x_i = y_i = \top} 1$$

Then $m = 16$, as $\max_i |X_i| \leq |\mathcal{X}| = 2^4 = 16$, and $\epsilon = 1/2$. Here we have $p = 5 > \log 16/log 2$.

The following example illustrates that the *simple set*-kernel separates MI concepts on discrete sets.

Example 4.2 *Let $\mathcal{X} = \{a, b, c, d\}$, $\mathcal{C} = \{a, c\}$, $\nu_I(x) \Leftrightarrow x \in \mathcal{C}$, and $k_I(x, y) = k_\delta(x, y)$. Then $\epsilon = 1$, $m = 4$ as $\max_i |X_i| \leq |\mathcal{X}| = 4$), and $p = 1 > 0$. It follows that:*

$$k_{\text{MI}}\,(X, Y) = \sum_{x \in X, y \in Y} k_\delta(x, y) = |X \cap Y| = k_{\text{set}}(X, Y)$$

It follows directly from the lemma above and from the definition of convolution kernels (Haussler, 1999) that (for finite example sets) MI concepts can be separated with convolved Gaussian RBF kernels if the underlying concept can be separated with Gaussian RBF kernels, since in this case k_{I}^p is a Gaussian RBF kernel itself, albeit with width σ^2/p instead of σ^2. In this case the MI kernel does not require an additional parameter to be chosen. Also note that for RBF kernels (6) always holds with $b = 0$. We further need

Lemma 4.3 *If an MI concept ν_{MI} is separable then the underlying concept ν_I is separable.*

Proof: Say $f_{\mathrm{MI}}(Z)$ is a function that separates positive and negative bags ($f_{\mathrm{MI}}(Z) > \theta \Leftrightarrow \nu_{\mathrm{MI}}(Z)$). Then $f_I(z) = f_{\mathrm{MI}}(\{z\})$ is a function that separates the underlying concept (with $f_I(z) > \theta \Leftrightarrow \nu_I(z)$). $\square$

It is now easy to show:

Theorem 4.4 *An MI concept ν_{MI} is separable by k_{MI} with a non-zero margin if and only if the underlying concept ν_I is separable by the kernel k_{I} with a non-zero margin.*

Proof: In Lemma 4.2 the margin of the MI problem is $\epsilon_{\mathrm{MI}} = 1 - m(1-\epsilon)^p > 0$. Furthermore, assuming a margin ϵ on f_{MI} in Lemma 4.3, a margin of ϵ can be found on f_I. The lower bounds are maintained similarly. $\square$

Note that the MI kernel can also be used in distance-based algorithms such as k-nearest neighbors. The distance metric is then defined on the kernel in the standard manner $d(x,y) = \sqrt{\langle x,x\rangle - 2\langle x,y\rangle + \langle y,y\rangle}$.

Example 4.3 *Consider the kernel $k_{\mathrm{MI}}(X,Y) = |X \cap Y|$ for solving MI problems on discrete sets based on the matching kernel (see Example 4.2). In such a case, the corresponding distance metric can be derived easily as the symmetric difference $d(X,Y) = |X \triangle Y|$.*

5. Learning Ray Concepts

Having shown in the previous section that MI problems can be separated with our MI kernel, we will now describe a simple approach that can be used to apply any propositional learning algorithm to MI problems. The motivation is based on some observations in the drug activity prediction domain. The advantage of this approach is the efficiency - in real world drug activity prediction problems bags can be huge which renders the computation of k_{MI} too expensive. In the empirical evaluation of this method (see section 6) we will show that - in spite of the simplicity of this approach - an SVM using this kernel can outperform several other MI learning algorithms.

Statistics If we can make further assumptions on the properties of $\mathcal{X}$, such as being generated by a normal distribution, by a mixture thereof, or other properties that can be summarized in a compact fashion then computing statistics on $X, X' \in \mathcal{X}$ can be useful in defining kernels on sets.

Definition 5.1 *Denote by $s : X \to s(X)$ a map computing statistics on $X \subset \mathcal{X}$. Then we call*

$$k_{\mathrm{stat}}(X, X') := k(s(X), s(X')) \qquad (9)$$

the statistic kernel.

Here $s(X)$ is a collection of properties of the set, say the mean, median, maximum, minimum, etc. Typically, $s(X)$ will be a vector of real numbers.

A similar approach has been used in the context of inductive logic programming (Krogel & Wrobel, 2001), where relational aggregations are used to compute statistics over a database.

It is not uncommon in drug activity prediction to represent a molecule by a bag of descriptions of its different conformations. Each conformation is in turn described by a feature vector such that each component of the vector corresponds to one ray emanating from the origin and measuring the distance to the molecule surface (Dietterich et al., 1997). It is often believed that the concept space of drug activity prediction can be described by putting upper and lower bounds along each ray.

Consider MI rays, i.e., concepts on sets of real numbers such that $r_{MI-\theta}(X) \Leftrightarrow \exists\, x \in X : x \geq \theta$. These concepts are the complement (negation) of upper bounds. Motivated by the observation that $r_{MI-\theta}$ can be learned using only the maximal element of the set, we find the following statistics kernel particularly interesting for drug activity prediction:

Example 5.1 (Minimax Kernel) *Define s to be the vector of the coordinate-wise maxima and minima of X, i.e.,*

$$s(X) = (\min_{x\in X} x_1, \ldots, \min_{x\in X} x_m, \max_{x\in X} x_1, \ldots, \max_{x\in X} x_m).$$

In our experiments we used $s(X)$ combined with polynomial kernels, i.e., $k(X, X') = (\langle s(X), s(X')\rangle + 1)^p$.

The minimax kernel is related to the MI kernel as k_{MI} can be seen as a *soft* max function, while minimax corresponds to a *component-wise* min and max function.

6. Drug Activity Prediction

Often drug activity prediction problems are used to asses MI learning algorithms, most prominently the Musk data set (Dietterich et al., 1997). The problem consists of predicting the strength of synthetic musk molecules. The class labels have been found by human domain experts. Two overlapping data sets are available. Musk1 contains 47 molecules labeled as 'Musk' (if the molecule is known to smell musky) and 45 labeled as 'Non-Musk'. The 92 molecules are altogether described by 476 conformations. Musk2 contains 39 'Musk' molecules and 45 'Non-Musk' molecules, described by 6598 conformations altogether.

Several empirical results on these two data sets have been achieved and reported in literature. The results in Table 1 are in alphabetical order. They have either been obtained by multiple tenfold cross-validation runs (10CV) or by leave-one-out estimation (LOO). The best classification results from each section are marked in boldface. The table is organized as follows: The first section contains algorithms specifically designed for MI learning. The second one contains algorithms that are designed as general purpose learning algorithms, but have been adapted to learn MI problems. The third section contains algorithms that have been run on the musk data using the minimax feature space described above, and the forth section contains results achieved by ignoring the MI setting while learning but obeying it when testing. For the SVM, an RBF kernel was used with $\gamma = 10^{-6}$. Boosted NB (DT) refers to a boosted naive Bayes (decision tree) classifier.

The fifth section of the table contains results from using SVMs with a polynomial version of the minimax kernel. The sixth section contains results for SVMs using MI kernels. Due to the extremely small sample size (MUSK1 contains only 96 bags and Musk2 only 104) which is furthermore not divisible by 10, 10-fold cross-validation is a very noisy and unreliable process. To address this problem, we opted to compute an error estimate by averaging over 1000 trials of randomly leaving out 10 instances. The advantage of this approach is that in addition to the error estimates we also obtain confidence intervals, which allows us to compare our results with the ones obtained in the literature. Finally, for the sake of comparability, also the leave-one-out error was computed.

We chose a Gaussian RBF kernel for the dot product on the elements of the bag, using the rule of thumb that γ should be in the order of magnitude of $\frac{1}{2d^2} \approx 10^{-5}$ or lower, where d is the dimensionality

Table 1. Classification errors (in %) on Musk1 and Musk2

Algorithm	Musk1	Musk2	Eval.
EM-DD	**3.2**	**4.0**	10Cv
Gfs kde Apr	8.7	19.6	10Cv
Iterative Apr	7.6	10.8	10Cv
MaxDD	11.1	17.5	10Cv
Multinst	23.3	16.0	10Cv
Bayesian-kNN	9.8	17.6	LOO
Citation-kNN	**7.6**	13.7	LOO
Mi-nn	12.0	18.0	?
NaiveRipperMI	12.0	23.0	?
Relic	16.3	**12.7**	10Cv
RipperMI	12.0	23.0	?
Tilde	13.0	20.6	10Cv

Algorithm	Musk1		Musk2	
	LOO	10-out	LOO	10-out
Minimax Feature Space				
Decision Tree	19.6	18.5	19.6	17.6
Boosted DT	**14.1**	**12.0**	20.6	20.6
Boosted NB	**14.1**	**12.0**	**16.7**	**16.7**
Ignoring the MI Setting while Training				
Decision Tree		28.3		43.1
Boosted DT		18.5		28.4
Boosted NB		20.3		36.7
SVM		**13.0**		**18.6**

Kernel	Musk1		Musk2	
	LOO	10-out	LOO	10-out
Polynomial Minimax kernel				
$p = 5$	**7.6**	$\mathbf{8.4 \pm 0.7}$	13.7	13.7 ± 1.2
LOO		15.5 ± 2.2		17.5 ± 2.1
LOO, normalized		17.5 ± 2.1		18.5 ± 1.9
MI kernel ($k_I(x, x') = \exp(-\gamma\|x - x'\|^2)$)				
$\gamma = 10^{-5.5}$	13.0	13.6 ± 1.1	**7.8**	12.0 ± 1.0
LOO		15.0 ± 1.9		19.0 ± 2.2
LOO, normalized		19.0 ± 2.7		14.5 ± 2.4

of the data[1]. This led to an almost optimal choice of parameters (the optimum lay within an order of magnitude of the initial estimate). As for preprocessing, the data was rescaled to zero mean and unit variance on a per coordinate basis. In the ray-kernel case, we simply used a polynomial kernel on $s(X)$. In order to avoid adjusting too many parameters, we chose the ν-parameterization (Schölkopf et al., 2000), with ν set to 0.075. The latter corresponds to an error level comparable to the ones in the published literature.

Normalization proved to be critical: the un-normalized sets, and the sets normalized by $\sqrt{f_{\text{norm}}(X)}$ performed worst, whereas there was no significant difference in performance between the kernels which employed normalization in feature space and those with averaging in feature space. This may be due to the fact that the average length of the sum of features of equal length may

[1]Note that the parameter p of MI kernels is chosen implicitly when choosing γ; as for Gaussian RBF kernels, k_{I}^p is a Gaussian RBF kernel itself.

be more strongly concentrated, which makes both approaches qualitatively equivalent. We only report results on the kernel with normalization in feature space.

To assess the true performance of the estimation procedure, one must not, however, fix a set of parameters and only then use a cross-validation step to assess the error for the now *fixed* set of parameters. We therefore adjusted the parameters, such as kernel width and the value of ν and γ for each leave-10-out sample separately and computed the CV error for this procedure. 20 random leave-10-out samples were drawn. The corresponding results are reported in the fifth block of Table 1, denoted by LOO and LOO norm (the latter refers to kernels with normalization in feature space). It is rather obvious that the method degrades dramatically due to the small sample size effects (only 100 observations). Also, the choice of suitable normalization in feature space yields only diminishing returns, given the high variance of the model selection procedure. It is well known (Cherkassky & Mulier, 1998) that the choice of model selection rules has a significant influence on the performance of the overall estimator, quite often more than the choice of the estimator itself.

The fact that polynomial minimax kernels outperformed the MI kernel (Gaussian RBF) on Musk1 can be explained by the fact that Musk1 contains much fewer conformations per bag than Musk2, hence the min and max statistic $s(X)$ is a quite adequate description of each bag. A small bag size is a major shortcoming also of other (synthetically generated) datasets, making them very unrealistic - as to our understanding real-world drug activity prediction problems usually involve big bags.

While EM-DD (Zhang & Goldman, 2002) is superior on both datasets, among the algorithms that are based on general purpose learning methods, SVMs with MI kernels outperform all other methods. We conjecture that replacing the averaging process over the data in feature space by a weighted average taking an estimate of the class of each element in the bag into account would significantly improve generalization.

We consider now another drug activity prediction problem - predicting the mutagenicity of molecules. The original dataset (Srinivasan et al., 1996), described by a set of prolog predicates, has widely been used in the inductive logic programming community. The only time this dataset has been used with MI learning algorithms is described in (Chevaleyre & Zucker, 2001). While the basic dataset is described by two relations, the atoms of the molecule and the bonds between the atoms, other representations include global molecular descriptors. Two sets of instances are frequently used, the so called 'friendly' dataset containing 188 instances, and the 'unfriendly' dataset containing 42 instances. We consider only experiments without global molecular features and represent each molecule by set of bonds together with the two adjacent atoms. This setup is similar to the one described in (Chevaleyre & Zucker, 2001) where error rates between 18.0% (RipperMI) and 39.0% (Foil) are reported on the 'friendly' dataset. So far, no results on the 'unfriendly' dataset have been reported using MI algorithms. Using our MI kernel and the data description without any global molecular descriptors we are able to achieve error rates of 7.0% on the 'friendly' dataset and 25% on the 'unfriendly' dataset. As before, we converted the data to zero mean and unit variance. Symbolic values were converted to orthonormal vectors. Model selection (ν and γ) was carried out by LOO cross-validation, and the results are reported for an average over 20 random leave-10-out subsets.

7. Conclusions and Future Work

In this paper we demonstrated a successful approach to extend the applicability of SVMs to data other than mere attribute-value tuples. It has been mentioned in literature that MI problems capture most of the complexity of relational learning problems. Therefore, MI kernels are an important step in crossing the boundary between successful attribute-value learning and relational learning.

We defined a general kernel for MI problems, proved that it separates MI concepts under natural assumptions, and showed its performance on benchmark data. Favorable for our approaches are the high accuracy and the simplicity with which other kernel methods can now be extended to MI problems. For example, by simply plugging our kernel into SVM regression, the only very recently formulated problem of MI regression can be tackled (Amar et al., 2001; Ray & Page, 2001). Clustering and feature extraction tasks have to the best of our knowledge not yet been investigated for MI data. Support vector clustering and kernel principal component analysis algorithms can now easily be applied to MI data by using our kernel function. Other distance-based algorithms can be applied to MI problems by defining a distance on the kernel in the standard manner. These are promising topics for future research.

Finally, the idea of labeling the elements of the bags, as done in EM-DD indicates further space for improvement, e.g., by modifying the averaging process, that is currently carried out within the bags ($\sum_{x \in X} \phi(x)$) into a weighted average, where preference is given to elements which are close to elements of other bags with the same class label. The kernel function would be a simple extension of the coset kernel and our MI kernel.

Acknowledgments

Research supported in part by the Esprit V project (IST-1999-11495) *Data Mining and Decision Support for Business Competitiveness: Solomon Virtual Enterprise* and a grant of the ARC. The authors thank Tamás Horváth, Thorsten Joachims, John Lloyd, Bob Williamson, and Stefan Wrobel for valuable discussions and/or comments.

References

Amar, R. A., Dooly, D. R., Goldman, S. A., & Zhang, Q. (2001). Multiple-instance learning of real-valued data. *Proceedings of the 18th International Conference on Machine Learning* (pp. 425–432). Morgan Kaufmann.

Auer, P. (1997). On learning from multi-instance examples: Empirical evalutaion of a theoretical approach. *Proceedings of the 14th International Conference on Machine Learning* (pp. 21–29). Morgan Kaufmann.

Blockeel, H., & De Raedt, L. (1998). Top-down induction of first order logical decision trees. *Artificial Intelligence*, *101*, 285–297.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory* (pp. 144–152). Pittsburgh, PA: ACM Press.

Cherkassky, V., & Mulier, F. (1998). *Learning from data*. New York: John Wiley and Sons.

Chevaleyre, Y., & Zucker, J.-D. (2001). A framework for learning rules from multiple instance data. *Proceedings of the 12th European Conference on Machine Learning*. Springer-Verlag.

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

Gärtner, T. (2000). Kernel-based feature space transformation in inductive logic programming. Master's thesis, University of Bristol.

Haussler, D. (1999). *Convolution kernels on discrete structures* (Technical Report). Department of Computer Science, University of California at Santa Cruz.

Herbrich, R. (2002). *Learning kernel classifiers: Theory and algorithms*. MIT Press.

Keeler, J. D., Rumelhart, D. E., & Leow, W.-K. (1991). Integrated segmentation and recognition of hand-printed numerals. *Advances in Neural Information Processing Systems* (pp. 557–563). Morgan Kaufmann.

Krogel, M.-A., & Wrobel, S. (2001). Transformation-based learning using multirelational aggregation. *Proceedings of the 11th International Conference on Inductive Logic Programming*. Springer-Verlag.

Maron, O., & Lozano-Pérez, T. (1998). A framework for multiple-instance learning. *Advances in Neural Information Processing Systems*. The MIT Press.

Ramon, J., & De Raedt, L. (2000). Multi instance neural networks. *Attribute-Value and Relational Learning: Crossing the Boundaries. A Workshop at the Seventeenth International Conference on Machine Learning (ICML-2000).*

Ray, S., & Page, D. (2001). Multiple instance regression. *Proceedings of the 18th International Conference on Machine Learning* (pp. 425–432). Morgan Kaufmann.

Ruffo, G. (2001). *Learning single and multiple instance decision trees for computer securtity applications*. Doctoral dissertation, Universita di Torino.

Schölkopf, B., Smola, A., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, *12*, 1207–1245.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels*. The MIT Press.

Srinivasan, A., Muggleton, S., Sternberg, M., & King, R. (1996). Theories for mutagenicity: a study in first-order and feature-based induction. *Artificial Intelligence*, *85*, 277–299.

Wang, J., & Zucker, J.-D. (2000). Solving the multiple-instance problem: A lazy learning approach. *Proceedings of the 17th International Conference on Machine Learning* (pp. 1119–1125). Morgan Kaufmann.

Zhang, Q., & Goldman, S. (2002). EM-DD: An improved multiple-instance learning technique. *Advances in Neural Information Processing Systems*. The MIT Press.