DISSIMILARITY-BASED MULTIPLE INSTANCE LEARNING

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K. Ch. A. M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op 10 juni 2015 om 12.30 uur
door

Veronika Vladimirovna CHEPLYGINA

informatica ingenieur
geboren te Moskou, Sovjet-Unie

This dissertation has been approved by the promotors:

Prof. dr. ir. M. J. T. Reinders and
Prof. dr. M. Loog

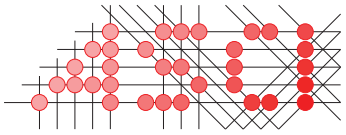
Composition of the doctoral committee::

Rector Magnificus,	
Prof. dr. ir. M. J. T. Reinders,	promotor
Prof. dr. M. Loog,	promotor

Independent members:

Prof. dr. R. Babuska,	3mE, TU Delft
Prof. dr. ir. H. L. W. Blockeel,	Katholieke Universiteit Leuven
	Universiteit Leiden
Prof. dr. L. I. Kuncheva,	Bangor University
Prof. dr. L. R. B. Schomaker,	Rijksuniversiteit Groningen
Prof. dr. M. Welling,	Universiteit van Amsterdam
	University of California Irvine
Prof. dr. ir. B. P. F. Lelieveldt,	TU Delft, substitute member

Dr. D. M. J. Tax has, as supervisor, contributed significantly to the preparation of this dissertation.



This work was carried out in graduate school ASCI.
ASCI dissertation series number 331.

Cover design by:	Hella Hekkelman
Printed by:	Proefschriftmaken.nl —— Uitgeverij BOXPress
Published by:	Uitgeverij BOXPress, s-Hertogenbosch

ISBN 978-94-6295-192-1

"IT IS WHAT YOU DON'T EXPECT...
THAT MOST NEEDS LOOKING FOR"

NEAL STEPHENSON, *Anathem*

Contents

1

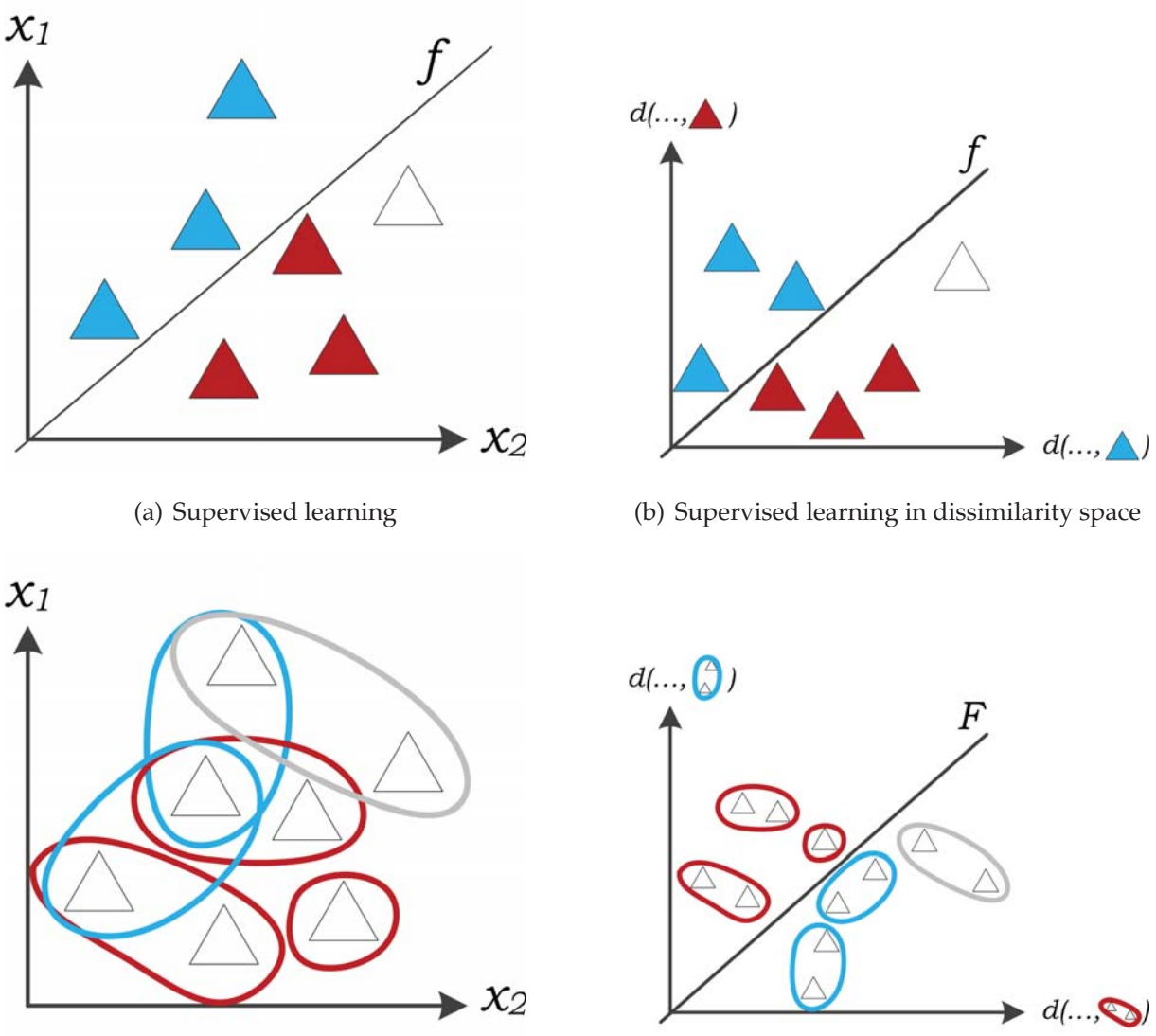
INTRODUCTION

Both humans and machines are able to learn from examples. In humans this happens naturally: repeated experiences lead to associations of input from the senses, to certain concepts or words ("mom"). "Mom" may look slightly different or say different things every day, but a child is able to generalize such variations of input and assign all of these to what "mom" means to the child.

In machine learning, the process of learning from examples is called supervised learning. Supervised learning consists of using labeled examples (such as faces, which are labeled with a person's name) to generalize input data, in order to be able to identify these concepts (faces) in previously unseen examples (other photographs). Other examples include spam filtering, computer aided diagnosis, drug discovery, and many others. In all of these problems, supervised learning requires (a) being able to describe each example by a finite number of measurements called *features*, and (b) being able to provide a *label* or category for each example. With these prerequisites, we can build a *classifier*, which, in turn, can label previously unseen examples, described by the same features as in (a). An example of a supervised classifier is shown in Fig. 1.1(a).

Unfortunately, standard supervised learning has many limitations, starting with defining discriminative features. For example, how does one represent a face? When describing a person, one could try to use high-level descriptions, such as "blue eyes", "round nose", or more low-level features such as the coordinates of interest points (such as eye and lip corners). However, during this process, information can be lost, i.e., representing faces of different people with the same feature vector, or noise can be introduced, i.e., representing faces of the same person with different feature vectors. It is challenging to define good features for a particular problem, because it is often unclear how a human performs the same recognition problem.

As an alternative to inventing features, the *dissimilarity representation*, where each example is described by (dis)similarities to a set of reference prototypes, has been introduced [133, 137]. Dissimilarities can be defined directly on examples, by asking an expert, or by measuring the amount of deformation that is needed to transform one ex-



(a) Supervised learning

(b) Supervised learning in dissimilarity space

(c) Multiple instance learning

(d) Multiple instance learning in dissimilarity space

Figure 1.1: Supervised and multiple instance learning. (a) Feature space in 2 dimensions with labeled instances (colored triangles) which are used to train a classifier f, which in turn can be used to classify unlabeled instances (white triangles). (b) Dissimilarity representation for a supervised problem, with two instances as prototypes. (c) Same feature space as in (a), but with labeled bags (colored blobs) of instances. Bag classifiers do not operate in this space. (d) Dissimilarity representation for a MIL problem, with two bags as prototypes. Here a bag classifier F can be trained.

Figure 1.2: Two labeled bags (groups) of unlabeled instances (faces). Both bags are labeled "Marco is in the picture". Given that information only about the left picture, you cannot say who Marco is, but given the extra information about the right picture, you should be able to say whether Marco is in a different picture you have never seen before

ample into another. Although this decreases the need for features, it becomes necessary to define a dissimilarity function and to choose a set of prototypes. Schematically, a dissimilarity representation can be depicted as in in Fig. 1.1(b).

A further limitation is that the classification task might not even fit into the standard supervised learning paradigm. In particular, in real life we can encounter situations where labels are not available for examples, but only for groups of examples. For example, with the information that Marco is in both pictures in Fig. 1.2, you are able to learn who Marco is, even though you are not provided with pictures of Marco alone.

In supervised learning terminology, the examples are called *instances* and the groups are called *bags*. The learning scenario (or rather, scenarios) where only labeled bags are available is called *multiple instance learning* or MIL. In MIL, the *standard assumption* is that *a bag is positive if and only if at least one of the instances is positive*. In the face example, this is true: if I tell you that Marco is in the picture, it means that at least one of the faces corresponds to Marco. A schematic example is shown in Fig. 1.1(c). Here only the bags are labeled, but using the standard assumption we could still infer the instance labels as shown in Fig. 1.1(a).

In these examples, the bags (photographs) are labeled, but it is assumed that the instances (faces) have labels, which are not given. This is also the case for many other classification tasks where labels are available only for bags, such as:

- Drug activity prediction. Each drug (molecule) is represented by its conformations (shapes the molecule can fold into). Each drug is a bag, each conformation is an instance. Labels (active or not active, i.e., binds to a particular molecule) are available on for on drug level.

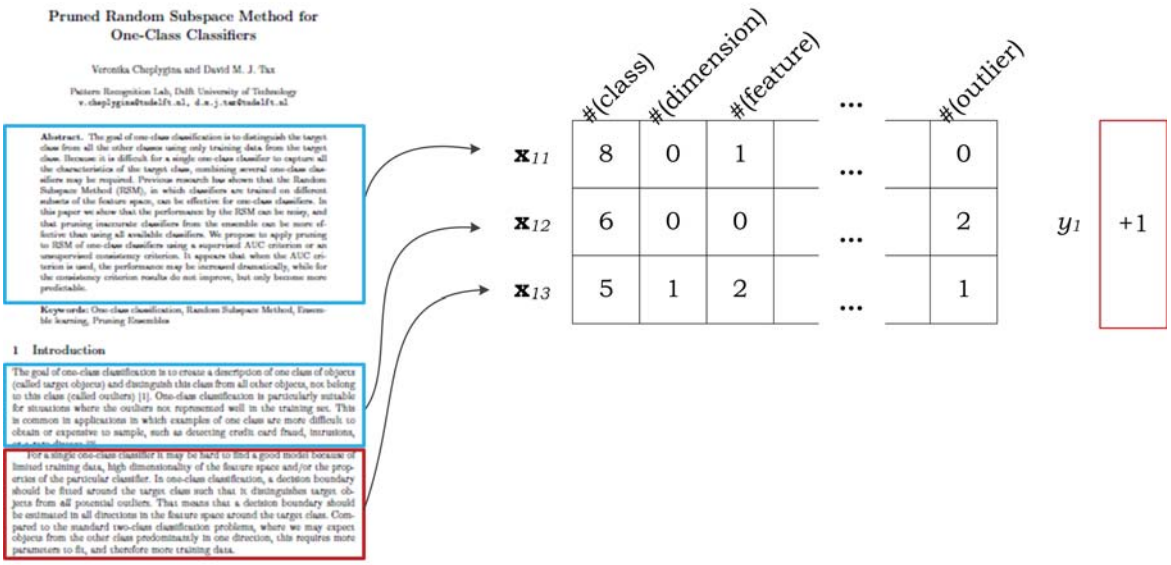


Figure 1.3: Representing a document in the MIL framework. The k-th paragraph of the i-th paper is described by a feature vector $\mathbf{x}_{ik}$ based on counts of different words (class, dimension, etc). The red paragraph is positive for a particular topic ("overfitting") while the blue ones are not. However, the classifier is only provided with the information that the whole paper is relevant for "overfitting", i.e., the bag is positive or $y_i = +1$.

- Document categorization. A document (such as a conference paper or a news item) is represented by several paragraphs, which are described by a histogram over dictionary words. Each document is a bag, and each paragraph is an instance. An example is shown in Fig. 1.3.
- Computer aided diagnosis. A medical image, such as a computed tomography (CT) scan, is represented by several image patches. The scan is labeled with a disease type, but not where in the image this disease is present.

One important difference in these examples is the relationship of the bag labels and the (hidden) instance labels. In drug activity prediction, a molecule is active as soon as at least one of its conformers is active, therefore the standard assumption holds. In document categorization, however, it is not as clear where a single paragraph on a particular topic is sufficient to assign that topic label to the whole document. For example, while this thesis contains a biography, it would be (hopefully) inappropriate to label the whole thesis as "biography".

Another important difference is the goal of the classifier. In document categorization, the end goal might be to label recently published articles, i.e., bags. In computer aided diagnosis, the end goal might be to provide a diagnosis for a new patient's CT scan. However, from a diagnostic point of view, it is also interesting to examine which patches

are responsible for this diagnosis, i.e., examine the instance labels.

The assumptions and goals of a problem largely determine the learning strategy of a classifier, and, in turn, affect factors such as computational complexity, and the type and interpretability of the output (bag and/or instance labels). For example, under the standard assumption, a reasonable learning strategy is to search for the concept: an instance that is present in all positive bags, but is not present in any negative bags. This process is computationally challenging, but is able to provide instance labels, which can be explained in terms of the instances' similarity to the discovered concept. Under the assumption that all instances contribute to the bag label, a strategy could be to compare bags on a higher level, for instance by averaging the instances. This approach is computationally very efficient because each bag is represented as single feature vector, but all interpretability of the provided bag labels is lost.

The dissimilarity representation offers an approach that is attractive both in terms of computation and interpretability. By representing each bag by dissimilarities to reference bags (Fig. 1.1(d)), the problem is transformed into a supervised learning problem. However, this representation allows the provided bag labels to be explained as a combination of dissimilarities to reference prototypes. As with any dissimilarity representation, there are several important questions in this process:

- How to define a dissimilarity function?
- How to choose a reference set of prototypes?
- What can this representation tell us about the problem?

The main goal of this work is to study these and related questions for the multiple instance learning problem. After a brief explanation of supervised learning, we will explain multiple instance learning and the dissimilarity representation in more detail. The remaining chapters of this thesis will focus on elaborating the link between the two techniques and on answering the questions above.

1.1 Supervised Learning

In a supervised learning problem, we have a dataset of objects $T = \{(\mathbf{x}_i, y_i) | i = 1, \ldots, N\}$ where the feature vectors $\mathbf{x}_i \in \mathbb{R}^m$ are the m measurements describing each object, and the labels $y_i \in \mathcal{Y}$ indicate the category that the object belongs to. $\mathcal{X} = \mathbb{R}^m$ is the input space and $\mathcal{Y}$ is the label space. Here we consider the binary case $\mathcal{Y} = \{-1, +1\}$ because problems with multiple classes can be decomposed into binary problems [68].

We are interested in finding a function $f : \mathcal{X} \to \mathcal{Y}$ which can provide labels for previously unseen feature vectors. Typically this is achieved by choosing a *function class* or a model, and then using T to estimate the model parameters. For example, in a

linear classifier (such as the classifiers shown in Fig. 1.1(a)) the label is modeled as $f(\mathbf{x}) = \text{sgn}(\mathbf{w}^\intercal \mathbf{x})$. The parameters, or *weights* $\mathbf{w}$ are estimated by minimizing a function which consists of a loss term $\mathcal{L}$ and a penalty term Ω:

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \mathcal{L}(y_i, \mathbf{w}^\intercal \mathbf{x_i}) + \lambda \Omega(\mathbf{w}) \tag{1.1}$$

where λ controls the trade-off of the two terms. Different choices for $\mathcal{L}$ result in different classifiers, for example the logistic loss $\mathcal{L}(y_i, \mathbf{w}^\intercal \mathbf{x_i}) = \log(1 + \exp(-y_i \mathbf{w}^\intercal \mathbf{x}))$ leads to logistic regression, and the hinge loss $\mathcal{L}(y_i, \mathbf{w}^\intercal \mathbf{x_i}) = \max\{0, 1 - y_i \mathbf{w}^\intercal \mathbf{x}\}$ leads to the support vector machine (SVM). The term Ω regularizes the classifier by controlling the complexity of the weight vector $\mathbf{w}$ and is frequently defined as a norm of $\mathbf{w}$.

A classifier with a very different strategy is the nearest neighbor (NN) classifier. This classifier assigns previously unseen objects the label of the closest objects in the training data T. Whereas in linear classifiers, T is used to find a weight vector $\mathbf{w}$ which can generalize (and T is no longer needed), in NN the classifier is fully defined by T. This results in a non-linear boundary, such as the example in Fig. 1.4. This example uses the Euclidean distance to define the closeness of the feature vectors, however, NN can even be used with distances defined on non-vectorial objects, such as graphs.

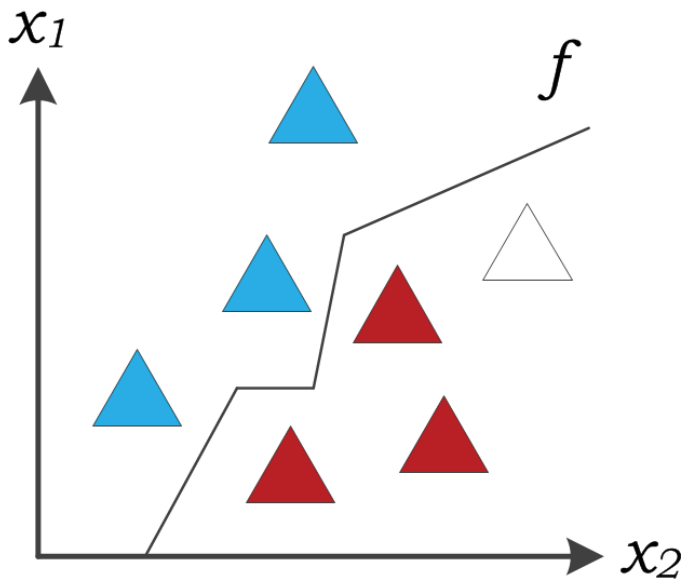


Figure 1.4: Nearest neighbor classifier.

Further explanations of these and other supervised classifiers can be found in [54, 68].

1.2 Dissimilarity Representation

Rather than representing an object in an absolute way by features, we can represent it in a relative way by dissimilarities to prototype objects in a *representation set* R. R is often taken to be a subset of T, although this is not strictly necessary, for example, prototypes from the test data may be used in a transductive learning setting [51]. For simplicity, here we let $R = T$.

Let d be a dissimilarity function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. For example, when $\mathcal{X} = \mathbb{R}^m$ we can simply use the Euclidean distance as d. We can then represent each object $\mathbf{x}_i$ as a vector of dissimilarities $\mathbf{d}_i = [d(\mathbf{x}_i, \mathbf{r}_1), \ldots, d(\mathbf{x}_i, \mathbf{r}_N)]$. As a result, each object is now represented in an N-dimensional space, the j-th feature corresponds to the dissimilarity to the j-th prototype. In this space, supervised classifiers can be trained as described in the previous section. A linear classifier in the dissimilarity space is a non-linear classifier in the original space.

The dissimilarity representation allows supervised learning on non-vectorial spaces $\mathcal{X}$, because all that is needed is a function d. Due to this, the dissimilarity representation has frequently been applied to structural pattern recognition problems [27, 58]. Although the dissimilarity function d may be a distance metric, it is not the case that only metric dissimilarities are suitable as choices for d [136]. In many cases, using d in the dissimilarity space outperforms using d together with a nearest neighbor classifier.

1.3 Multiple Instance Learning

In multiple instance learning (MIL), an object is represented by a bag $B_i = \{\mathbf{x}_{ik} | k = 1, ..., n_i\} \subset \mathbb{R}^m$ of n_i feature vectors or instances. The training set $T = \{(B_i, y_i) | i = 1, ...N\}$ now consists of positive ($y_i = +1$) and negative ($y_i = -1$) bags.

The standard assumption for MIL is that there exist instance labels z_{ik} which relate to the bag labels as follows: a bag is positive if and only if it contains at least one positive, or *concept* instance [49]. More general assumptions have also been proposed [65, 157, 188].

MIL methods can be broadly divided into two categories: instance-based and bag-based. This is a categorization we use in our work [40] as well as a slightly later published survey of MIL [3]. Instance-based methods use the constraints posed by the bag labels and the MIL assumptions to build an instance classifier, and combine instance classifications to classify bags [5, 114, 183, 195]. On the other hand, bag-based methods aim to classify bags directly by defining distances or kernels [71], or by transforming the data into a supervised problem [171, 194]. We explain the classifiers in more detail in the following sections.

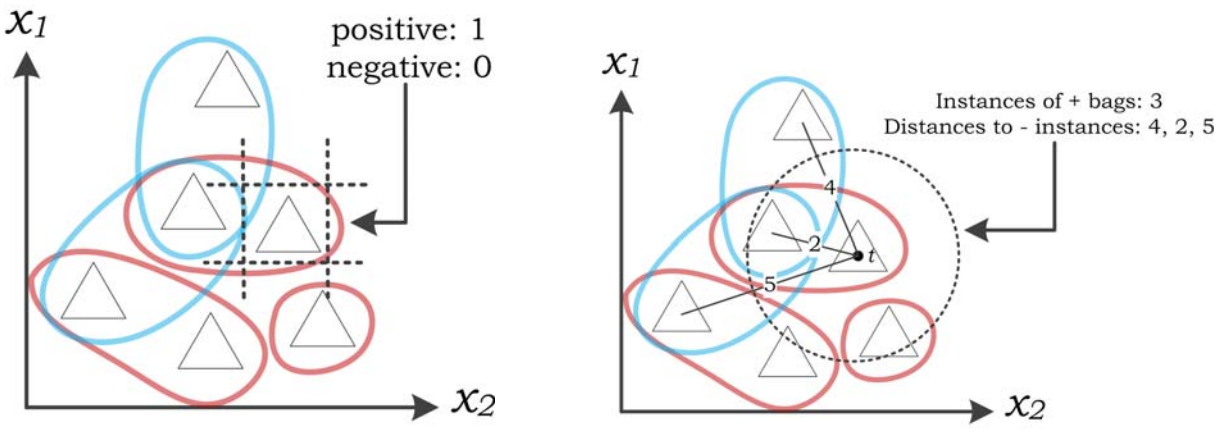
1.3.1 Instance-based Classifiers

Instance-based classifiers use assumptions about how the instance and bag labels are related to build an instance classifier f. Bags can then be classified by classifying that bag's instances, and combining the instance labels. For example, the noisy-or rule,

$$\frac{p(y=1|B_i)}{p(y=-1|B_i)} = \frac{1-\prod_{k=1}^{n_i}(1-p(z_{ik}=1|\mathbf{x}_{ik}))}{\prod_{k=1}^{n_i} p(z_{ik}=-1|\mathbf{x}_{ik})} \tag{1.2}$$

reflects the standard assumption that a bag is positive if and only if at least one of the instances is positive.

One group of instance-based classifiers works by *finding the concept*: a region in the feature space which contains at least one instance from each positive bag, but no instances from negative bags. The original class of MIL methods used an axis-parallel hyper-rectangle (APR) [49] as a model for the concept. A related technique is to express the "concept-ness" of a point t in the feature space as a density. For example, Diverse Density [114] $DD(t)$ is based on ratio between the number of positive bags which have instances near t, and the distance of the negative instances to t. The instance classifier f is then based on the presence of instances in the optimal APR, or on the distance from the target concept $t* = \arg\max_t DD(t)$. Illustrations of these methods are shown in Fig. 1.5.



(a) Axis-parallel hyper-rectangle model

(b) Diverse density model

Figure 1.5: Instance-based classifiers APR (left) and Diverse Density (right). Triangles are instances, red blobs are positive bags and blue blobs are negative bags. APR searches the instance space to find a region where only instances from positive bags, but not negative bags are contained. DD is a density based on the ratio of instances from positive bags close to a point t, and distances of negative instances to t.

Another group of instance-based classifiers works by *finding the instance labels*. These classifiers start with an initial hypothesis for instance labels, and thus the classifier f, and use the constraints posed by bag label assumptions to update the instance labels, and in turn, the classifier. These classifiers are typically supervised classifiers extended to work in the MIL setting, such as support vector machines (mi-SVM [5]), boosting (MILBoost [183]) or random forests (MIForest [107]).

Last but not least, a way to learn in MIL problems is to propagate the bag labels to the instances (thus assuming all instances in the bag have the same label), and use supervised learners on these propagated labels. We call this approach SimpleMIL. To obtain a bag label from predicted instance labels, the instance labels have to be combined. Here, the noisy OR rule or other combining methods can be used [108, 111]. For example the average rule,

$$\frac{p(y=1|B_i)}{p(y=-1|B_i)} = \frac{\sum_{k=1}^{n_i} p(z_{ik}=1|\mathbf{x}_{ik})}{\sum_{k=1}^{n_i} p(z_{ik}=-1|\mathbf{x}_{ik})} \tag{1.3}$$

assumes that all instances contribute to the bag label.

1.3.2 Bag-based Classifiers

Bag-based methods generally assume that bags of the same class are more similar to each other than bags from different classes, and thus build a bag classifier F directly. An early classifier adopting this approach is Citation-kNN [186], which uses the Hausdorff distance as a bag distance, and an adaptation of the nearest neighbor classifier, to classify previously unseen bags. In [71], a bag kernel is defined as a sum of the instance (linear or radial basis) kernels. The way a distance or kernel is defined affects which (implicit) assumptions are made about the problem. A drawback for real-world applications is that metricity or positive-definiteness requirements for distances or kernels exclude some domain-specific similarity functions.

Other bag-based methods have addressed MIL by representing each bag by a single feature vector. This can be done in an absolute manner, such as by summarizing instance statistics of each bag [71], or in a relative manner, by representing each bag by (dis)similarities to a set of prototypes $\mathcal{R} = \{R_1, \ldots, R_M\}$ in a so-called dissimilarity space [133]. Therefore, each bag is represented by a single feature vector $\mathbf{d}(B_i, \mathcal{R}) = [d(B_i, R_1), \ldots, d(B_i, R_M)]$, where d is a (dis)similarity measure. The prototypes can be chosen to be bags [42, 194] or instances [36, 67]. In this space, any supervised classifier can be used.

An advantage of bag-based methods is that the standard assumption might be too strict for certain types of MIL problems. By relying on similarity, more relaxed assumptions [157, 188], such as multiple concepts, can be captured.

Classifier	Type	Robustness	Complexity	Instance labels
APR [49]	Instance, concept	−−	−−	+
Diverse Density [114]	Instance, concept	−−	−	+
EM-DD [195]	Instance, concept	−	−−	+
miSVM [5]	Instance, labels	−/+	−/+	+
MILBoost [183]	Instance, labels	−/+	−/+	+
SimpleMIL	Instance, labels	+	+	+
Citation *k*-NN [186]	Bag, direct	+	−/+	−
Set kernel [71]	Bag, direct	+	+	−
mean-inst [71]	Bag, absolute	++	++	−
extremes [71]	Bag, absolute	++	++	−
MILES [36]	Bag, relative	+/++	−/+	+
Proposed (Chapter 3)	Bag, relative	++	+	−
Proposed (Chapter 4)	Bag, relative	++	−/+	+

Table 1.1: MIL classifiers. The type describes the strategy used: finding the concept or the instance labels for instance-based classifiers, and a direct, or absolute or relative representation for bag-based classifiers. Robustness indicates the ability to perform reasonably across different problems. Complexity expresses the computational complexity of training the classifier. Instance labels indicates whether the classifier is able to produce instance labels or not.

1.3.3 Summary

An overview of several instance-based and bag-based classifiers used in this thesis is shown in Table 1.1. Here we rank the classifiers based on properties that we feel are important when choosing a MIL classifier: robustness, computational complexity, and the ability to produce instance labels. While the computational complexity and instance labels are relatively straightforward, the robustness property requires additional explanation. With robustness, we try to express the classifier's ability to perform *reasonably* across different types of MIL problems. Although largely consistent with findings in a recent survey [3], these ranks reflect our own experiences with these classifiers. The goal is therefore to offer a sneak preview for the findings in the other chapters of this thesis.

1.4 Contributions

The main contributions of this thesis address a number of mismatches of problems and solutions in MIL. A problem here can be an abstract classification task or a particular

dataset. A solution can be an additional assumption, a learning strategy or a specific classifier. It seems needless to say that the solution must fit the problem. Therefore, the answers to questions such as "How to define a dissimilarity function?" or "How to choose a reference set of prototypes?" must invariably start with "It depends on your data".

Our first contribution is an overview of several extensions of supervised learning, including MIL. We show that the idea of representing training or test objects as sets of feature vectors has been used in several other learning scenarios and explain how these scenarios relate to MIL. Furthermore, we show that in machine learning literature, MIL in fact encompasses two different learning scenarios: MIL with the goal of labeling bags, and MIL with the goal of labeling instances. (Chapter 2)

Our second contribution is an understanding of a wide range of MIL problems: how the data is distributed, which assumptions might be more suitable and hence which classifiers might be more successful in each case. Through our investigations, we conclude that MIL (with the goal of labeling bags) also encompasses more types of problems than it was originally intended for. Furthermore, as we demonstrate with a real-life dataset of chronic obstructive pulmonary disease, our intuition about the fit of MIL to a particular application may not always be correct. (Chapters 3 and 6)

The third contribution of this thesis is a comparison of different bag dissimilarity measures. To define a bag dissimilarity measure, the first step is to define a model for the bag. The instances in a bag can be seen as a discrete point set, as a sampling from an underlying instance distribution, or as the nodes of a graph. Each model has several corresponding choices for the dissimilarity function: a measure of overlap between point sets, a distribution distance, or a graph distance. Next to affecting the computational complexity of computing the dissimilarities, these choices affect what information about the data is preserved, and therefore the classification performance. (Chapters 3 and 7)

The fourth contribution is a comparison of different choices of reference prototypes for the dissimilarity representation. A challenge in MIL is that prototypes can be bags or instances. There are several trade-offs here. Choosing prototypes as instances leads to a richer dissimilarity representation, but increases the dimensionality significantly. Despite the rich representation, the information about the origin of the instances is discarded, which may not be desirable in some problems. On the other hand, considering a bag as a single prototype can lead to a more robust representation, due to low dimensionality and/or the fact that the instances in a bag are considered jointly. Another trade-off here is that with instance prototypes, labeling instances is a possibility. We propose a classifier that combines the advantages of both methods. (Chapter 4)

The fifth contribution circles back to the two different goals in MIL: that of labeling bags, and that of labeling instances. Classifiers that optimize performance on bags, may not provide the best possible instance labels. Furthermore, these instance labels may be

unstable under different variations of the training data. This is dangerous, especially in cases where the instance labels carry biological or medical significance. We investigate the stability of instance labels provided by several MIL classifiers, and propose a stability measure which, in conjunction with a bag-level supervised measure, can be used to compare classifiers. (Chapter 5)

2

On Classification with Bags, Groups and Sets

Many classification problems may be difficult to formulate in the traditional supervised setting, where both training and test samples are individual feature vectors. It may be the case that samples are better described by sets of feature vectors, that labels are only available for sets rather than individual samples, or, if individual labels are available, that these are not independent. To better deal with such problems, several extensions of supervised learning have been proposed, where either training and/or test objects are sets of feature vectors. However, such extensions are often proposed independently, disregarding important similarities and differences with other existing classification problems. In this work, we provide an overview of such learning scenarios, propose a taxonomy to illustrate the relationships between them, and discuss directions for further research in these areas.

This chapter is accepted for publication as:
Veronika Cheplygina, David M. J. Tax, Marco Loog. On Classification with Bags, Groups and Sets. *Pattern Recognition Letters*

2.1 Introduction

Some pattern recognition problems are difficult to formulate as regular supervised classification problems where (feature vector, label) pairs are available to train a classifier that, in turn, can predict labels for previously unseen feature vectors. In such cases, the objects are, in fact, not individual feature vectors, but sets or *bags* of feature vectors or *instances*. Although terminology differs in different fields, we use the terms bags and instances in the rest of this paper.

The first reason for using a bag of instances is that a single feature vector is often too restrictive to describe an object. For example, in drug activity prediction, an application originally addressed in multiple instance learning [49], we are interested in classifying molecules as having the desired effect (active) or not. However, a molecule is not just a list of its elements: most molecules can fold into different shapes or conformations, which may have an influence on the activity of that molecule. Furthermore, the number of stable shapes is different per molecule. A more logical choice is therefore to represent a molecule as a set of its conformations.

The second reason is that labels on the level of feature vectors are difficult, costly and/or time-consuming to obtain, but labels on a coarser level may be obtained more easily. For computer aided diagnosis applications, it can be very expensive for a radiologist to label individual pixels or voxels in an image as healthy or diseased, while it is more feasible to label a full image, or large image regions. Such coarsely labeled scans or regions can then be used for predicting labels on bag level by labeling new patient scans or, on a finer grained instance level, by labeling individual pixels or voxels.

Another reason to consider the labeling of bags of instances, instead of single feature vectors, is that there may be structure in the labels of the instances. For example, in face verification, where a video of a person is available, considering all the video frames jointly can provide more confident predictions than labeling each of the frames individually and combining the decisions. Similarly, neighboring objects in images, videos, sounds, time series and so forth are typically very correlated, and thus should not be classified independently.

For all these reasons it may be advantageous to represent the objects by bags of instances. Different applications may ask for different representations in the training and the test phase. All possibilities shown in Fig. 2.1 occur: both training and test objects may be bags or instances. Traditional supervised learning is in the SI-SI scenario in the top left, where both training and test objects are instances. Predicting molecule activity is in the MI-MI scenario, where both training and test objects are bags. Image classification problems can be found in the MI-MI scenario (training on images, testing on images) as well as the MI-SI scenario (training on images, testing on pixels or patches). The face verification problem is best represented by the SI-MI scenario (training on a single face, testing on a set of faces).

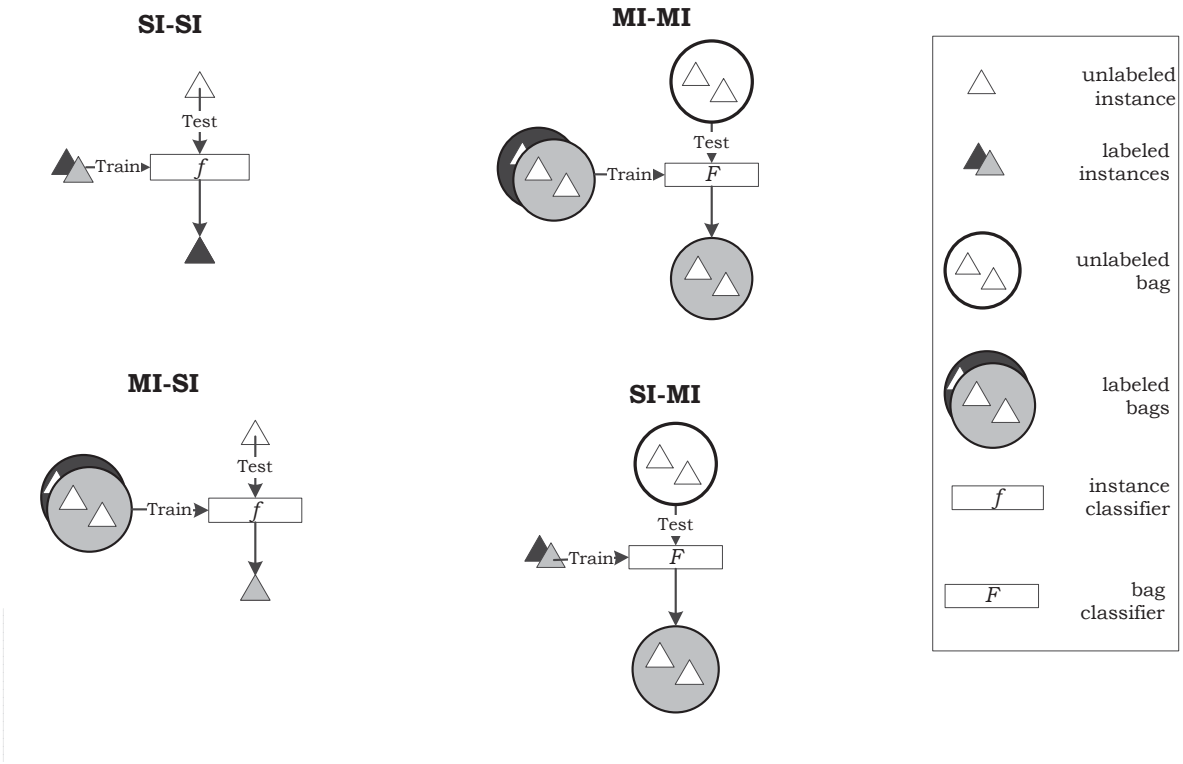


Figure 2.1: Supervised learning (SI-SI) and extensions. In the MI-MI scenario (Section 2.4), both training and test objects are bags. In the MI-SI scenario (Section 2.5), the training objects are bags and test objects are instances, while in the MI-SI scenario (Section 2.6), the training objects are instances and the test objects are bags.

The greater representational power, reduced need for labels and improved classification performance are attractive properties of learning domains where objects can be expressed as sets of feature vectors. This idea has been therefore applied in image recognition [36, 115], face verification [6, 126], tracking [7], computer aided diagnosis [156], molecule activity prediction [49] and document categorization [5], amongst others.

The success of a classifier in one (source) application may motivate other researchers to use the same method in a different (target) application. However, it is not necessarily the case that the assumptions of the source application still hold in the target application, which can lead to poor performances. On the other hand, it may also happen that the same type of problem occurs in two different applications, and that researchers in the respective fields approach the problem in different ways, without benefiting from each other's findings. We therefore believe that understanding the relationships between such learning scenarios is of importance to researchers in different fields.

With this work, our goal is to provide an overview of learning scenarios in which bags of instances play a role at any of the stages in the learning or classification process. We have gathered the papers in this overview by searching for papers that proposed novel learning scenarios, as well as by combining synonyms of the word "set" with words such as "classification" or "learning". This work is intended as a survey of learning problems, not of classifiers for a particular scenario, although we refer to existing surveys of this type whenever possible. Furthermore, we mainly focus on a single-label, binary classification scenario, as many problem formulations can be easily extended to a multi-label [174], multi-class setting.

This paper begins with an overview of notation and learning scenarios in Section 2.2. We explain the categories of learning scenarios in more detail in Sections 2.3 to 2.6. The paper concludes with a discussion in Section 2.7.

2.2 Notation and Overview

The basic terminology of bags and instances was already introduced in the previous section. This terminology is borrowed from the field of multiple instance learning (MIL). This is for two reasons: there are more papers on MIL than on other topics covered in this work, and the terms do not have other mathematical definitions that could be confusing.

Mathematically, an instance is represented by a single feature vector $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X} = \mathbb{R}^d$ is a d-dimensional space, while a bag is represented by a set of n_i feature vectors $B_i = \{\mathbf{x}_{ik}; k = 1...n_i\} \in 2^{\mathcal{X}}$. We denote the set of possible classes $\mathcal{C}$, and the set of possible labels $\mathcal{Y}$. In the case where each object has only one class label (and the focus of this overview), $\mathcal{Y} = \mathcal{C}$, in a multi-label scenario $\mathcal{Y} = 2^{\mathcal{C}}$.

Instance space	$\mathcal{X}$, typically $\mathbb{R}^d$
Instance	$\mathbf{x} \in \mathcal{X}$
Bag	$B \in 2^{\mathcal{X}}$
Discrete set of class labels	$\mathcal{C}$
Output space	$\mathcal{Y}$
Instance label	$y \in \mathcal{Y}$
Bag label	$Y \in \mathcal{Y}$
Instance classifier	$f : \mathcal{X} \to \mathcal{Y}$
Bag classifier	$F : 2^{\mathcal{X}} \to \mathcal{Y}$

Figure 2.2: Overview of important notation

When a test object is an instance, we are interested in finding an instance classifier $f : \mathcal{X} \to \mathcal{Y}$. When a test object is a bag, we are interested in finding a bag classifier $F : 2^{\mathcal{X}} \to \mathcal{Y}$

There are several aspects that differ in the learning scenarios covered in this paper. We choose to categorize the learning domains by the following characteristics:

- **Type of training data**. The type of data that is provided to train a classifier: labeled instances, or labeled bags. In the case a bag is provided, usually the labels for the individual instances are not available.

- **Type of test data**. The type of data that is classified by the trained classifier: instances (pixels in an image) or bags (entire images). Often this also determines how evaluation is done: on instance level or on bag level, but this is not always the case.

- **Assumptions on labels**. Different applications have different assumptions of how the labels of the instances and the labels of the bags are related: for example, an assumption could be that all pixels inside an image region have the same label. These assumptions play an important role in how the learning algorithms are developed.

These characteristics lead us to the categories in the leftmost column of Table 2.1. In the following sections, which are organized by the first two dimensions (types of training and test data), we will explain each category, the corresponding learning scenarios and assumptions, the equivalence of different terms in literature, or why the category is empty.

Table 2.1: Summary of learning scenarios. The columns show the section where we explain the scenario, the type of training and test data, the assumptions on how the instance and bag labels are related are weak or strong, and the main references where the learning scenario is applied.

Section	Train	Test	Assumptions	Main references
2.3. SI-SI	Inst	Inst	Weak	Supervised learning
	Inst	Inst	Strong	Batch classification[184]
				Collective classification[32, 117, 159]
2.4. MI-MI	Bags	Bags	Weak	Sets of feature vectors [86, 95, 96]
	Bags	Bags	Strong	Multiple instance learning [49, 114]
2.5. MI-SI	Bags	Inst	Weak	-
	Bags	Inst	Strong	Multiple instance learning [121, 180]
				Aggregate output learning [124]
				Learning with label proportions [141]
2.6. SI-MI	Inst	Bags	Weak	-
	Inst	Bags	Strong	Group-based classification [156]
				Set classification [126]
				Full-class set classification [100]

2.3 SI-SI: Train on Instances, Test on Instances

The first category of Table 2.1 contains traditional **supervised learning** where both training and test objects are assumed to be independently generated from some underlying class distributions. We assume that the reader is familiar with supervised learning and keep this section short. For a general introduction, please refer to [85]. With the assumption of independently drawn train and test instances, the best possible approach is to classify each feature vector individually.

However, in some situations data is not independently generated, and we can make more assumptions about the correlations in the test data, and use these assumptions to improve the performance. The classical, rather general way to model dependencies between observations is through Markov random fields [93] and the related, currently more popular conditional random fields [103]. There are, however, also approaches that act direct on the bag level and do not need an explicit probabilistic model in order to be applied. Examples can be found in **batch classification** [184] and **collective classification** [32, 117, 158]. Batch classification is concerned with classifying parts (segments) of medical images, and the assumption is that neighboring image segments are correlated, and thus often have the same label. Collective classification is often applied to documents, such as identifying requests in emails [32] or assigning categories to webpages [117]. The underlying assumption is that correlation exists between emails in the

same thread, or between webpages that link to each other, therefore, the labels of such neighboring instances are related.

2.4 MI-MI: Train on Bags, Test on Bags

When both the training objects and test objects are bags, but no additional assumptions about the labels are present, the goal is **classification of sets of feature vectors** [95]. This concept is also used for classifying images as bags of pixels [86] or prediction of binding of proteins [80] by defining kernels on bags directly. Various kernel functions on computer vision applications are also explored in [197], where a kernel on sets of feature vectors is called **ensemble similarity**. Another application in classifying websites [96], where each website is represented as a set of feature vectors, and a set distance together with a nearest neighbor classifier is used.

Another domain where both training and test objects are bags, but stronger assumptions are made is called **multiple instance learning** (MIL) [49, 114]. In MIL, the objects are referred to as *bags* of *instances*. Originally, it was assumed that $\mathcal{Y} = \{-1, +1\}$, and that the bag labels are determined by the (hidden) labels of their instances: a bag is positive if and only if there is at least one positive instance inside the bag; a bag is negative if and only if all of its instances are negative. Such reasoning has been applied to molecule or drug activity prediction [49, 66], image classification [36, 115], text categorization [5, 200], prediction of hard drive failures [123] and other settings. For example, in molecule activity prediction, a molecule is considered active if at least one of the conformations demonstrate the activity of interest.

There are two main approaches to achieve the goal of classifying bags. Due to the assumption on the relationship of the bag and instance labels, earlier methods focused on first finding an instance classifier f, and then applying a combining rule g to the instance outputs. To use the traditional assumption in MIL, g is defined by the noisy OR function, as follows:

$$F(B) = \begin{cases} +1 & \text{if } g(\{f(\mathbf{x}_k)\}_{k=1}^n) > 1 \\ -1 & \text{otherwise} \end{cases} \tag{2.1}$$

$$g(\{f(\mathbf{x}_k)\}_{k=1}^n) = \frac{1 - \prod_{k=1}^n 1 - f(\mathbf{x}_k)}{\prod_{k=1}^n 1 - f(\mathbf{x}_k)} \tag{2.2}$$

where $f(\mathbf{x}_k) = p(y_k = 1|\mathbf{x}_k)$.

More relaxed formulations of the traditional assumption have also been proposed [65, 188]. For instance, for a bag to be positive, it needs to have a specific fraction of positive instances. With such alternative assumptions, it is still possible to find f first and then

apply an appropriate g to determine the labels of the test bags. By assuming that all instances contribute to the bag label independently, for instance, g can be replaced by the product of the instance posterior probabilities. Other, generalized rules for combining posterior probabilities for instances can be found in [111].

Several MIL methods have moved away from the assumptions on the relationships of instance and bag labels [65], and learn using assumptions on bags as a whole, therefore taking a detour to the "classification of sets of feature vectors" domain. In other words, such methods aim at finding F directly rather than through a combination of f and g. A more general assumption is that bags of the same class are similar to one another, therefore such methods learn by defining distances [186] or kernels [71] between bags. Other approaches include converting bags to a single-instance representation using similarities [36], dissimilarities [171] or histograms in instance space [182], the so-called bag-of-words representation. These methods then borrow techniques from supervised learning to classify bags.

More extensive surveys of MIL assumptions and classifiers can be found in [3, 65, 198].

2.5 MI-SI: Train on Bags, Test on Instances

This section is concerned with the case where training data is only labeled on bag-level, while instance-level labels are desired in the test phase. Note that this is not possible if no assumptions are made about the label transfer between instances and bags. This is why the "train on bags, test on instances, no assumptions" category in Table 2.1 is empty (denoted by -). By making additional assumptions, however, something can be said about the instance-level labels of the test data.

2.5.1 Learning from Weakly Labeled Data

The standard assumption in **multiple instance learning** is one of the possibilities we can use to train the classifier using labeled bags, but provide instance-level labels for the test data. Although originally, the goal of MIL was to provide labels for bags, a side-effect of some algorithms is that instance labels are predicted as well. The fact that only bag labels are required to produce instance labels means that less labels are required than in the usual supervised setting. In several fields, where such weakly labeled data can be (more) easily obtained, the focus has shifted to classifying instances rather than bags.

The goals of classifying instances and classifying bags are not identical, and therefore, in many cases, the optimal bag classifier will not be the optimal instance classifier and vice versa. An important reason in MIL for this is the traditional assumption. If bag

classification is done by combining instance predictions, such as in (2.2), false negative instances are going to have less effect on the bag performance than false positive instances. Consider a positive bag where a positive instance is misclassified as negative: if the bag has any other positive instances, or a negative instance that has been falsely classified as positive, the bag label will still be correct. However, for a negative bag the label changes as soon as a single instance is misclassified. Similar observations have been made in [147] and in [173]. A more general reason why the optimal instance and bag classifiers do not necessarily correspond, is unequal bag sizes. Misclassifying a bag with a few instances will have less effect on the instance performance, than misclassifying a bag with many instances.

There are several examples where MIL is used for the purpose of predicting instance labels, rather than bag labels. For example, in image segmentation or annotation [121, 180] the goal is to label pixels as belonging to the background, or one of the objects portrayed in the image. This goal can be achieved with supervised learning, by providing fully annotated training images, where each pixel is labeled as background or foreground. However, providing such annotated images is costly – it is easier to approximately indicate where the foreground objects are present. MIL is therefore an interesting setting that can still offer image annotation, while only using coarsely labeled images as training input.

Weakly annotated data is also a benefit in tracking [7]. Instead of providing instances (patches) of the tracked object to the learner, bags of patches (with several inexact locations of the tracked object) can be used to improve performance. However, the goal of the tracking algorithm is to again label patches (instances), not bags. Other examples can be found in music information retrieval [113], where the goal is to predict tags (such as "rock", "pop") for songs, based on coarser-level tags for the albums or the artists. In [18], the goal is to classify fragments of bird songs, only by learning with bag-level labels for the whole recording.

Although these domains are not directly related to bags of instances, at this point it is important to mention that learning with such weakly annotated data has links to **semi-supervised learning** [35, 204] and **learning with only positive and unlabeled data** [61]. Both of these fields deal with weakly annotated data in a sense that some of it is annotated, and some of it is not. In multiple instance learning, all of the data (in the form of bags) is annotated, however, from the perspective of instances, these annotations are weak. More about the links between these fields can be found in [109, 201].

2.5.2 Learning with Other Label Spaces

Another setting where only training objects are sets of feature vectors is **learning about individuals from group statistics** [98], **aggregate output learning** [124] and **learning**

with label proportions [141], independent names for very related ideas. Here the bag labels are not just class labels, but proportions of class labels, $Y = \{y_i | i = 1, \ldots, |\mathcal{C}|, y_i \in \mathbb{R}, \sum_i y_i = 1\}$. For instance, a bag can be labeled as "75% positive, 25% negative".

In [98], the application is image annotation images are provided with labels ("tiger") along with a fraction of image patches that contain tigers. This is very similar to the image segmentation scenario described in Section 2.5, but the fraction of positives provides additional information to the classifier. Another possible application addresses privacy issues, such as when it is not desirable to provide the income (label) of a single person, but less problematic to provide the collective income (aggregated label) for a group of people.

The applications addressed in [141] are spam filtering and advertising. In spam filtering, proportions of spam/normal email are easier to estimate for a particular user than the exact labels of each email, however, the goal is to classify individual emails afterwards. In advertising, the proportions are related to customers that would buy a product only on discount, and customers that would buy a product in any circumstances. During an advertising campaign, estimating such proportions can help to predict which groups of customers should receive a discount coupon (and therefore buy the product).

This aggregated output / label proportions setting can be seen as multiple instance learning, where the fraction of positive instances (often called the witness rate) in the bags is already specified. An exact fraction is a stronger assumption than a non-zero fraction, therefore it should be easier to learn when the witness rate is given. For real-life MIL datasets, [98] assumes that a positive bag has a fraction of $\frac{1}{n_i}$ positive instances. Other MIL methods take advantage of this by estimating a witness rate first, and then using this estimate to build instance classifiers [73, 108].

2.6 SI-MI: Train on Instances, Test on Bags

This section is concerned with the scenario where instance-level labels are available for training, but bag-level instances are needed in the test phase. This may seem illogical, because it is already possible to build an instance classifier – why would bags be necessary?

If no assumptions are made about how the bags are generated, there is no added value in considering bags in the test phase, and the reason the category corresponding to SI-MI with few assumptions in Table 2.1 is empty. However, if additional information is available about the labels inside a bag, it may still be worthwhile to consider sets of feature vectors in the testing phase. Dependencies or constraints between the feature vectors inside a test bag can be exploited to improve the overall classification.

Consider the 1-dimensional binary classification problem in Fig. 2.3, and assume that

given objects from each class, we have found the Bayes optimal instance classifier. The black circles are the test set, and their true labels are -1. If we were to classify these instances independently, the error would be equal to $\frac{1}{3}$, because the leftmost object will be misclassified to the the positive class. However, with the added constraint that these instances belong to a group of objects from the same class, we could apply a combining rule on the instance outputs, classify the bag as negative, and propagate the label to all the individual instances, reducing the error to 0.

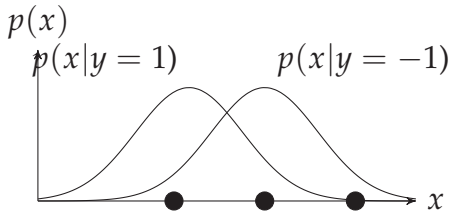


Figure 2.3: 1-dimensional binary classification problem. The shaded instances are from the $y = -1$ class. When classifying these instances jointly, the added information that they are all of the same class helps to reduce the classification error.

This situation occurs in **group-based classification** [20, 156] and **set classification** [126], independently proposed names for the setting where test objects are sets of feature vectors from the same class. Note that this setting can be easily transferred to the "train on bags, test on bags" category, because if the instances in one bag have the same label, it is straightforward to create bags from instances and vice versa.

In [156], a real-world application involves classifying groups of cells as healthy or anomalous, with the added information that all cells in a group share the same label. The classification of a test bag distance-based and is done by modifying the supervised versions of the nearest neighbor or the nearest mean classifiers. There are two broad approaches called the voting and the pooling scheme. In the voting scheme, each instance is labeled by a classifier f, such as the nearest neighbor, and the labels are combined with majority voting as g. In the pooling scheme, the distances are aggregated first, and only then converted to a label for the bag. The results show that the pooling scheme (i.e. a nearest neighbor classifier F applied on the bag distances) produces better results.

Another example from computer aided diagnosis is in [87], where classification of cells is applied on two levels: patches (image segments, or instances) and cell slides (full images, or bags). Although some patch-level labels are available and a patch classifier can be built, considering the slide-level labels is still beneficial for performance.

One of the applications in [126] is face classification. When multiple images of the same person are available (such as from different cameras, or from different frames in a video), the fact that the faces share the same label can help identification. The most straightforward approach involves combining predictions of each instance in a bag during the test phase. The best performing approach actually moves towards the

MI-MI scenario, because in both the training and test phase, instance subsets are generated. Kernels are defined on these subsets, and the test bag is classified by combining the predictions of its subsets.

In other literature on face classification, this problem is often referred to as **image-set classification** [92, 190], although here it is possible that the training objects are bags as well (i.e. there are multiple training faces available for each person). Such problems are therefore often also solved with set distances or kernels.

The added information that all instances in a set share the same label is just one of the examples of a setting where the testing objects are bags. A reversed setting is **full-class set classification** [100]. It has an additional constraint that each of the instances has a unique label, i.e. it is known beforehand which instance labels will be present in the bag. This is an appropriate setting for registration purposes, where it is known which objects will be present, but not which detected object is which. Here the output of the bag classifier is not a single class label, but a super-label $Y \in \mathcal{I}$, where $\mathcal{I}$ is the set of permutations of the all class labels. Because $|\mathcal{I}| < |2^{\mathcal{C}}|$, [100] shows that a classifier F that finds the instance labels jointly is guaranteed to perform better than concatenating the outputs of instance classifiers f.

Note that although instance labels are obtained, the labels we are interested in (the super-labels) are bag labels, and the performance is evaluated on bag level: either all instances were labeled correctly, or not. We illustrate this with the diagrams in Fig. 2.4.

2.7 Discussion

Many classification problems deal with objects that are represented as sets of feature vectors, or so-called bags of instances. This popularity is not surprising, as there are several motivating reasons for choosing such a representation at one or more stages of the classification process. Firstly, a set of feature vectors provides greater representational power than a single feature vector, and it might not be logical to express multiple entities (such as several face images of one person) as a single entity. Secondly, often labels may be available only on bag level, and too costly to obtain on instance level, therefore using the bag of instances representation as a form of weak supervision. Lastly, it can be advantageous to consider bags as a whole rather than as independent instances, because of relationships of the instances in a single bag.

This popularity is not without dangers: several different learning scenarios may be defined for the same problem, or several different problems may be incorrectly grouped under the same learning scenario. We proposed a taxonomy that illustrates the relationships of scenarios that deal with bags, groups or sets, and could help researchers relate novel problems to existing applications and research directions.

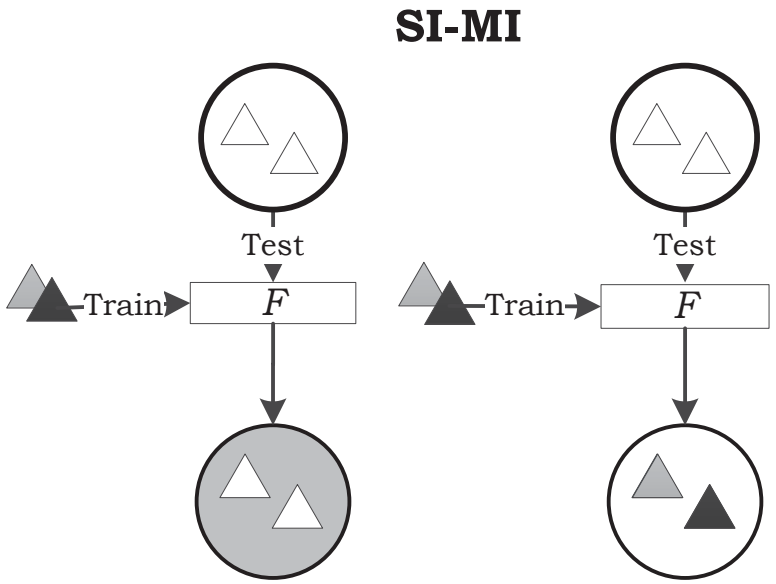


Figure 2.4: Variants of the SI-MI scenario. The training objects are instances and the test objects are bags, although the bag may be labeled by a set of instance labels (situation on the right). Note that in this case, the instance labels are decided jointly (as a bag super-label) by a bag classifier F, not by an instance classifier f.

While the proposed taxonomy allows for heterogeneity in training and test objects (i.e., where training objects are bags and test objects are instances and vice versa), it is limited because the training or test objects themselves are homogeneous. It would be interesting to investigate what happens in the case where in the training phase both labeled bags and labeled instances are available. As we already discussed in Section 2.5, the optimal bag classifier does not necessarily correspond with the optimal instance classifier. Therefore, deciding how to best use the available labels should depend on whether bags or instances are to be classified in the test phase. However, what if bags and instances can be expected in both the classification and test phases? A straightforward solution would be to train separate bag and instance classifiers, but when the bag and instance labels are related, an integrated classifier would perhaps be more suitable.

Another interesting observation is that the "hybrid" categories in the taxonomy (Section 2.6: SI-MI and Section 2.5: MI-SI) have attracted a lot of attention, and that the learning scenarios proposed here all need to rely on strong assumptions about the relationships of the instance and bag labels. One of the questions this raises is, what are the minimal assumptions needed to learn in such situations? Furthermore, the learning scenarios we reviewed do not exhaustively cover the types of constraints that could be present between the instance and bag labels. Learning scenarios that will be proposed in the future to fill some of these gaps, can now be easily placed in the context of the works described in this overview.

3

Multiple Instance Learning with Bag Dissimilarities

Multiple instance learning (MIL) is concerned with learning from sets (bags) of objects (instances), where the individual instance labels are ambiguous. In this setting, supervised learning cannot be applied directly. Often, specialized MIL methods learn by making additional assumptions about the relationship of the bag labels and instance labels. Such assumptions may fit a particular dataset, but do not generalize to the whole range of MIL problems. Other MIL methods shift the focus of assumptions from the labels to the overall (dis)similarity of bags, and therefore learn from bags directly. We propose to represent each bag by a vector of its dissimilarities to other bags in the training set, and treat these dissimilarities as a feature representation. We show several alternatives to define a dissimilarity between bags and discuss which definitions are more suitable for particular MIL problems. The experimental results show that the proposed approach is computationally inexpensive, yet very competitive with state-of-the-art algorithms on a wide range of MIL datasets.

This chapter is published as:
Veronika Cheplygina, David M. J. Tax, and Marco Loog. Multiple Instance Learning with Bag Dissimilarities. *Pattern Recognition* 48(1): 264–275, 2015.

3.1 Introduction

Many pattern recognition problems deal with complex objects that consist of parts: images displaying several objects, documents with different paragraphs, proteins with various amino acid subsequences. The success of supervised learning techniques forces such complex objects to be represented as a single feature vector. However, this may cause important differences in objects to be lost, degrading classification performance. Rather than representing such a complex object by a single feature vector, we can represent it by a set of feature vectors, as in multiple instance, or multi-instance learning (MIL) [49]. For example, an image can be represented as a bag of segments, where each segment is represented by its own feature vector. This is a more flexible representation that potentially can preserve more information than a single feature vector representation.

In MIL terminology, an object is called a *bag* and its feature vectors are called *instances*. MIL problems are often considered to be two-class problems, i.e., a bag can belong either to the positive or the negative class. During training, the bag labels are available, but the labels of the instances are unknown. Often assumptions are made about the instance labels and their relationship with the bag labels. The standard assumption is that positive bags contain at least one positive or *concept* instance, whereas negative bags contain only negative, *background* instances [49, 114]. An image labeled as "tiger" would therefore contain a tiger in at least one of its segments, whereas images with other labels would not depict any tigers. Many MIL approaches therefore focus on using the labeled bags to model the concept region in the instance space. To classify a previously unseen bag, the instances are labeled according to the best candidate model for the concept, and the bag label is then obtained from these instance labels.

It has been pointed out [36] that although for many problems the bag representation is useful, the assumptions on the bag and instance labels typically do not fit the application. For instance, for an image of the "desert" category, it would be wrong to say that "sand" is the concept, if images of the "beach" category are also present. Therefore, methods in which the standard assumption is relaxed, have emerged. In [185] an adaptive parameter is used to determine the fraction of concept instances in positive bags. Generalized MIL [157, 188] examines the idea that there could be an arbitrary number of concepts, where each concept has a rule of how many (just one, several or a fraction) positive instances are needed to satisfy each concept. A review of MIL assumptions can be found in [65].

This line of thought can be extended further to cases where it is difficult to define a concept or concepts, and where most, if not all, instances, contribute to the bag label. The implicit assumption is that bag labels can be related to distances between bags, or to distances between bags and instances. Such approaches have used bag distances [186], bag kernels [71], instance kernels [36] or dissimilarities between bags [40, 163, 171].

Bag-based approaches are attractive because because they transform the MIL dataset back to a standard feature vector representation such that regular supervised classifiers can be used. Unfortunately, some of the representational power of MIL can be lost when converting a bag to a single feature vector of (dis)similarities. It has indeed been pointed out that the definition of distance or similarity can influence how well the representation is suited for one or more concepts [65]. The question is how to do this in a way that still preserves information about the class differences. Furthermore, competing approaches offer a variety of definitions of (dis)similarity, and it is not always clear which definition should be preferred when a new type of MIL problem presents itself.

In this paper we propose a general MIL dissimilarity approach called MInD (**M**ultiple **In**stance **D**issimilarity). We discuss several ways in which dissimilarities between bags can be defined, show which assumptions these definitions are implicitly making, and hence which definitions are suitable for different types of MIL problems. We have collected several examples of such problems in a single repository online. Furthermore, we discuss why the dissimilarity space is an attractive approach for MIL in general. An important advantage is that there are no restrictions on the dissimilarity measure (such as metricity or positive-definiteness). This allows the use of expert-defined dissimilarities which often violate these mathematical restrictions. Similarly, there is no restriction on the classifier used in the dissimilarity space, which is attractive for potential end-users faced with MIL problems, and who already have experience with a certain supervised classifier. Lastly, with a suitable choice of dissimilarity and classifier, the approach is very computationally efficient, yet still provides interpretable state-of-the-art results on many datasets. For example, the average minimum distance between bags with a logistic classifier achieves very good performances, is easy to implement, and the classifier decisions can be explained in terms of dissimilarities to the prototypes.

After a review of MIL approaches in Section 3.2, we propose MInD in Section 3.3. In Section 3.4, we show some examples of MIL problems and demonstrate which dissimilarities are most suitable in each case. We then compare results to a range of MIL methods in Section 3.5, and discuss practical issues of dissimilarities and other bag-level methods in Section 3.6. A conclusion is given in Section 3.7.

3.2 Review of MIL Approaches

In multiple instance learning (MIL), an object is represented by a bag $B_i = \{\mathbf{x}_{ik} | k = 1, ..., n_i\} \subset \mathbb{R}^d$ of n_i feature vectors or instances. The training set $\mathcal{T} = \{(B_i, y_i) | i = 1, ...N\}$ consists of positive ($y_i = +1$) and negative ($y_i = -1$) bags. We will also denote such bags by B_i^+ and B_i^-. The standard assumption for MIL is that there are instance labels which relate to the bag labels as follows: a bag is positive if and only if it contains

http://www.miproblems.org

at least one positive, or *concept* instance [49].

Under this standard assumption, the strategy has been to model the concept: a region in the feature space which contains at least one instance from each positive bag, but no instances from negative bags. The original class of MIL methods used an axis-parallel hyper-rectangle (APR) [49] as a model for the concept, and several search strategies involving such APRs have been proposed.

Diverse Density [114] is another approach for finding the concept in instance space. For a given point t in this space, a measure $DD(t)$ is defined as the ratio between the number of positive bags which have instances near t, and the distance of the negative instances to t. The point of maximum Diverse Density should therefore correspond to the target concept. The maximization problem does not have a closed form solution and gradient ascent is used to find the maximum. The search may therefore converge to a local optimum, and several restarts are needed to find the best solution.

EM-DD [195] is an expectation-maximization algorithm that refines Diverse Density. The instance labels are modeled by hidden variables. After an initial guess for the concept t, the expectation step selects the most positive instance from each bag according to t. The maximization step then finds a new concept t' by maximizing DD on the selected, most positive instances. The steps are repeated until the algorithm converges.

Furthermore, several regular supervised classifiers have been extended to work in the MIL setting. One example is mi-SVM [5], an extension of support vector machines which attempts to find hidden labels of the instances under constraints posed by the bag labels. Another example is MILBoost [183], where the instances are reweighted in each of the boosting rounds. The bag labels are decided by applying a noisy OR [114] rule to the instance labels, which reflects the standard assumption.

It has been recognized that the standard assumption might be too strict for certain types of MIL problems. Therefore, relaxed assumptions have emerged [157, 188], where a fraction or a particular number of positive instances are needed to satisfy a concept, and where multiple concept regions are considered.

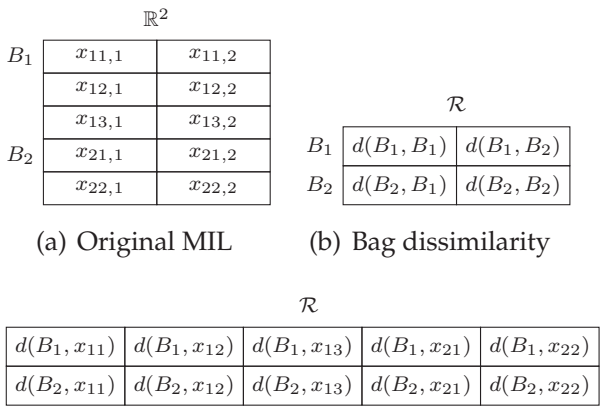
A similar notion is used in MILES [36], where multiple concepts, as well as so-called negative concepts (concepts that only negative bags have) are allowed. All of the instances in the training set are used as candidate concept targets, and each bag is represented by its similarities to these instances. A sparse 1-norm SVM is then used to simultaneously maximize the bag margin, and select the most discriminative similarities, i.e., instances that are identified as positive or negative concepts.

A step further are methods that do not make explicit assumptions about the instances or the concepts, but only assume that bags of the same class are somehow similar to each other, and then learn from distances or similarities between bags. Such methods include Citation-kNN [186], which is based on the Hausdorff distance between bags, bag kernels [71] and bag dissimilarities [171, 194]. In [71], a bag kernel is defined either

as a sum of the instance kernels, or as a standard (linear or RBF kernel) on a transformed, single instance representation of the bag. One example is the Minimax representation, where each bag is represented by the minimum and maximum feature values of its instances.

Last but not least, a way to learn in MIL problems is to propagate the bag labels to the instances, and use supervised learners on these propagated labels. We call this approach SimpleMIL. To obtain a bag label from predicted instance labels, the instance labels have to be combined. Here, the noisy OR rule or other combining methods can be used [108, 111]. It has been demonstrated that supervised methods can be quite effective in dealing with MIL problems [147].

All MIL methods can be more globally summarized by the representation that they use: the standard instance-vector-based representation, a bag dissimilairity representation and a bag-instance representation (see Fig. 3.1). The first representation is the standard representation of MIL, where each bag consists of several instances, and the dimensionality is equal to the dimensionality of the instance space. In this example, there are two bags which are represented in a 2D feature space. This is the representation used by EM-DD, mi-SVM, MILBoost and SimpleMIL. The representation on the top right is the bag representation, used by Citation-kNN, bag kernels and our bag dissimilarity approach. The representation in the bottom is the instance representation, used by MILES. In the latter two representations, regular supervised classifiers are again applicable. In these cases, less assumptions about the relationships of bag and instance labels are needed, but the definition of (dis)similarity creates implicit assumptions on which instances are important.



(a) Original MIL (b) Bag dissimilarity

(c) Instance dissimilarity

Figure 3.1: Representations of a MIL problem with 2 bags and 2 features. B_1 has 3 instances and B_2 has 2 instances. The dimensionality of the original representation depends on the number of features, while in the dissimilarity representation, the dimensionality depends on the number of bags or instances.

3.3 Bag Dissimilarity Representation

We can represent an object, and therefore also a MIL bag B_i, by its dissimilarities to prototype objects in a representation set $\mathcal{R}$ [133]. In our work, $\mathcal{R}$ is taken to be a subset of size M of the training set $\mathcal{T}$ of size N (typically $M \leq N$). Using these M prototypes, each bag is represented as $\mathbf{d}(B_i, \mathcal{T}) = [d(B_i, B_1), ...d(B_i, B_M)]$: a vector of M dissimilarities. Therefore, each bag is represented by a single feature vector and the MIL problem can be viewed as a regular supervised learning problem.

We propose a framework which encompasses different ways to define $d(B_i, B_j)$, the dissimilarity between two bags B_i and B_j. There are two main steps: the representation of a bag, and given this representation, the definition of the dissimilarity function. We distinguish the following approaches to treat the bags:

- As a point set, or subset of the feature space. In this case, d is defined through a set distance.
- As a distribution of instances. Here, d is defined through a distribution distance.
- As an attributed graph, where the instances are the nodes, and relationships between instances are the edges [200]. In this case, d is defined as a graph kernel or distance. However, because it is not straightforward to define the edges and determine the trade-off of nodes and edges in such a problem [105], we do not focus on this case here.

3.3.1 Bags as Point Sets

The first approach to define a dissimilarity of two bags is to consider each bag as a point set or a subset of a high-dimensional space. One possible distance that can be computed is based on the Hausdorff metric, under which two point sets B_i and B_j are close to each other when every point in B_i is close to a point in B_j. Closeness is defined through the distance d employed, which typically is Euclidean. The Hausdorff distance, derived from the metric, has been widely used in object matching in computer vision [53, 83]. The Hausdorff distance applied to bags uses the maximum mismatch between the instances of the respective bags, and is defined as $d_H(B_i, B_j) = \max(d_h(B_i, B_j), d_h(B_j, B_i)$ where d_h is defined as:

$$d_h(B_i, B_j) = \max_k \min_l d(\mathbf{x}_{ik}, \mathbf{x}_{jl}). \tag{3.1}$$

As d_h is not symmetric, the final Hausdorff distance d_H is symmetrized by taking the maximum of the directed distances. All of these steps ensure that the Hausdorff dis-

tance is metric, i.e., that it satisfies the identity, symmetry and triangle inequality properties.

It has been pointed out [53] that the maximum operation in the original definition might be too sensitive to outliers, and modified versions of the Hausdorff distance have been introduced [53, 196]. Two alternative examples of defining $d(B_i, B_j)$ are shown in (3.2) and (3.3). The underlying instance dissimilarity function we use here is the squared Euclidean distance.

$$d_{minmin}(B_i, B_j) = \min_k \min_l d(\mathbf{x}_{ik}, \mathbf{x}_{jl}) \text{ and} \tag{3.2}$$

$$d_{meanmin}(B_i, B_j) = \frac{1}{n_i} \sum_{k=1}^{n_i} \min_l d(\mathbf{x}_{ik}, \mathbf{x}_{jl}). \tag{3.3}$$

Fig. 3.2 shows the first step in computing such dissimilarities between two bags. The arrows in each diagram are the minimum distances between instances of two bags, which are asymmetric. For example, in the left diagram, the two instances in B_i have the same nearest instance in bag B_j, but this is not true for the diagram on the right. If the next step is to take the minimum of these instance distances, as in (3.2), the resulting dissimilarity will be symmetric. However, d_{minmin} is symmetric, it does not satisfy the identity property, that is, $d_{minmin}(B_i, B_j) = 0$ does not imply $B_i = B_j$ when an instance in B_i coincides with an instance in B_j, and the triangle inequality is not always satisfied. On the other hand, if we average the minimum distances as in (3.3), we will notice that $d_{meanmin}(B_i, B_j) \neq d_{meanmin}(B_j, B_i)$.

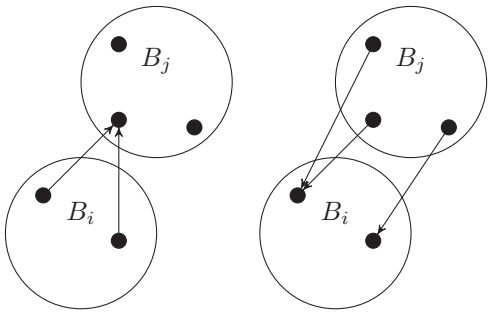


Figure 3.2: Minimum instance distances between two bags. The bag dissimilarity is defined as the minimum, maximum, average or other statistic of these distances. The directions of the arrows show that there are two, possibly asymmetric, dissimilarities: that of B_i to B_j, and that of B_j to B_i

These dissimilarities are therefore non-metric, which can be problematic when classifying bags based on the labels of their nearest neighbors. These non-metric properties are not a problem in the dissimilarity approach, because the dissimilarity matrix is viewed

as a collection of features and the relation between features is not constrained like is for distances. In fact, because we step away from the requirements of a distance, we can define even more general ways to obtain $d(B_i, B_j)$ from the pairwise instance dissimilarities.

First of all, there is no need to symmetrize the dissimilarities. In fact, it can be the case that one of the directions, i.e., measuring to the prototypes, or measuring from the prototypes, is more informative [39]. If both directions are informative, this information can be combined in more ways than by just enforcing symmetry. For instance, we can concatenate both the *to* and *from* dissimilarity matrices to obtain a $2N$-dimensional extended asymmetric dissimilarity space [138].

Furthermore, if the identity property is no longer a requirement, the first step in computing $d(B_i, B_j)$ does not need to be the minimum. For instance, we could measure the average of *all* instance distances between two bags, which is rather similar to the set kernel in [71]: $K(B_i, B_j) = \sum_{\mathbf{x}_{ik} \in B_i, \mathbf{x}_{jl} \in B_j} k(\mathbf{x}_{ik}, \mathbf{x}_{jl})$ where k is a polynomial, radial basis or other type of kernel on feature vectors. The dissimilarity version is:

$$d_{meanmean}(B_i, B_j) = \frac{1}{n_i n_j} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} d(\mathbf{x}_{ik}, \mathbf{x}_{jl}). \tag{3.4}$$

3.3.2 Bags as Instance Distributions

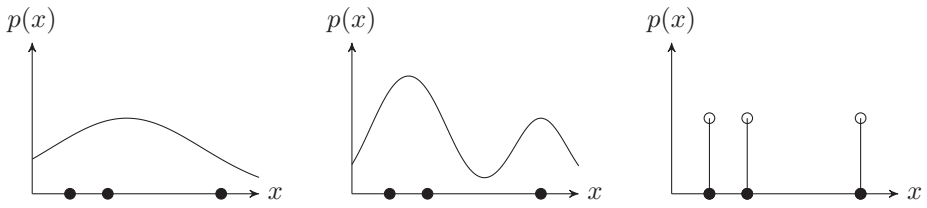
Alternatively, we can view each bag as a probability distribution in instance space, and define a bag dissimilarity as a distribution distance. Because the true distributions are not available, we have to approximate the instance distributions, and provide distances between the approximated distributions.

Fig. 3.3 shows a number of these approaches to approximate a 1D distribution. These approaches can be seen as a Parzen density with a very large, intermediate, and very small width parameter σ, resulting in a single Gaussian, an "intermediate" multi-modal density, and the empirical density consisting of Dirac deltas.

The first possibility is to approximate each bag B_i by a normal distribution with parameters (μ_i, Σ_i) and define the bag dissimilarity through the Mahalanobis distance:

$$d_{\text{Maha}}(B_i, B_j) = (\mu_i - \mu_j)^\intercal \left(\frac{1}{2}\Sigma_i + \frac{1}{2}\Sigma_j \right)^{-1} (\mu_i - \mu_j). \tag{3.5}$$

Approximating each bag by a normal distribution may, however, be too restrictive. In this case, a multivariate Gaussian (or a Parzen density with a smaller width parameter) can be used instead. The bag dissimilarity measure can then, for instance, be computed



(a) (b) (c)

Figure 3.3: Different ways to represent a 1-D bag with instances at $x = 0.5$, $x = 1$ and $x = 2.5$ as a distribution. From left to right: normal distribution, Parzen density, δ-peaks. The type of approximation influences the choice of distribution distance that can be applied.

as a divergence between the estimated distributions [23]. For instance, the Cauchy-Schwarz divergence, rewritten in our notation, can be estimated as:

$$d_{\mathrm{CS}}(B_i, B_j) = -\log\left(\frac{K_{\sigma_i+\sigma_j}(B_i, B_j)}{(K_{2\sigma_i}(B_i, B_i)K_{2\sigma_j}(B_j, B_j))^{\frac{1}{2}}}\right)$$

where

$$K_\sigma(B_i, B_j) = \sum_{\substack{\mathbf{x}_k \in B_i \\ \mathbf{x}_l \in B_j}} \frac{\exp\left(\frac{1}{-2\sigma^2}(\mathbf{x}_k - \mathbf{x}_l)^\mathsf{T}(\mathbf{x}_k - \mathbf{x}_l)\right)}{(2\pi\sigma^2)^{\frac{d}{2}}}. \tag{3.6}$$

We follow the advice of [23] to use a single parameter σ, i.e., $\sigma_i = \sigma_j$ when the distributions (bags) are from the same data source. There are similarities between (3.6) and (3.4). The use of the radial basis function in (3.6) reduces the influence of larger distances on the bag dissimilarity. In some sense, this also happens in (3.3), where the downscaling of the distances is relative to the instance, however, and not global as in (3.6).

Reducing the width parameter of the Parzen window even further, a distribution can be represented as Dirac deltas at each of the instances. One possible distance for this representation is the earth movers distance (EMD) [155]. EMD measures the minimum amount of work to transform one probability distribution B_i, or pile of earth, into another probability distribution B_j, or hole in the ground. We assume that in a bag with n_i instances, each instance has $1/n_i$ of the total probability mass, so the pile in fact consists of n_i smaller piles, and the hole consists of n_j smaller holes. The ground distances between piles and holes are defined by the (Euclidean) instance distances $d(\mathbf{x}_{ik}, \mathbf{x}_{jl})$. The distribution distance is then defined as follows:

$$d_{\text{EMD}}(B_i, B_j) = \sum_{\mathbf{x}_k \in B_i, \mathbf{x}_l \in B_j} f(\mathbf{x}_k, \mathbf{x}_l) d(\mathbf{x}_k, \mathbf{x}_l) \tag{3.7}$$

where $f(\mathbf{x}_k, \mathbf{x}_l)$ is the flow that minimizes the overall distance, and that is subject to constraints that ensure that the only available amounts of earth are transported into available holes, and that all of the earth is indeed transported: $f(\mathbf{x}_k, \mathbf{x}_l) \geq 0$, $\sum_{\mathbf{x}_k \in B_i} f(\mathbf{x}_k, \mathbf{x}_l) \leq 1/n_j$, $\sum_{\mathbf{x}_l \in B_j} f(\mathbf{x}_k, \mathbf{x}_l) \leq 1/n_i$ and $\sum_{\mathbf{x}_k \in B_i, \mathbf{x}_l \in B_j} f(\mathbf{x}_k, \mathbf{x}_l) = 1$.

3.3.3 Contrast with Related Approaches

There are several successful MIL methods in the literature that are related to the methods that we advocate. This section highlights the differences between our approach and those proposed by others.

Distances

Citation-kNN [186] uses the Hausdorff distance to define a distance between the bags. This distance matrix is used together with a nearest neighbor classifier, where to decide the label of a bag B_i, both the bags which are nearest neighbors of B_i, as well as bags for which B_i is their nearest neighbor, are used to decide the bag label. However, a nearest neighbor classifier does not use all the information that is contained in the dissimilarity matrix [131], and in our previous work we have demonstrated that for MIL, such matrices are more informative when used as features in the dissimilarity space [171].

Although our approach uses a dissimilarity matrix between bags, it is important to understand that although $d(B_i, B_j)$ can be seen as a bag *distance*, it is not necessarily our goal to arrive at a definition outputs low values for bags of the same class, and high values for bags of different classes. In this work, "distance" and "dissimilarity" are therefore not equivalent, rather, "dissimilarity" is a generalization of "distance". Given a certain prototype $R \in \mathcal{R}$, it is sufficient if a dissimilarity produces *discriminative* values for positive and negative bags. For a distance, only the situation on the left of Fig. 3.4 is acceptable, but for a dissimilarity approach, both situations are equally informative.

For methods such as nearest neighbor, metricity is therefore a desirable property. For the dissimilarity space, this is not the case as we will demonstrate in our Section 3.5.3

Kernels

In [71], a bag kernel is defined either as a sum of the instance kernels, $K(B_i, B_j) = \sum_{\mathbf{x}_{ik} \in B_i, \mathbf{x}_{jl} \in B_j} k(\mathbf{x}_{ik}, \mathbf{x}_{jl})$ where k is a polynomial, radial basis or other type of kernel on feature vectors, or as a standard (linear or RBF kernel) on instance statistics of each

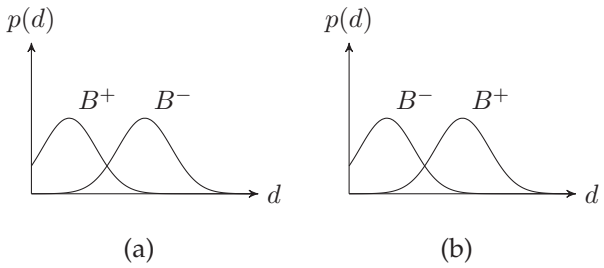


Figure 3.4: Distributions of $d(\cdot, R^+)$: distances or dissimilarities to a positive prototype bag. Only the situation on the left is suitable for a nearest neighbor approach, while the situation on the right is equally informative for classification in the dissimilarity space.

bag. One example is the Minimax representation, where each bag is represented by the minimum and maximum feature values of its instances.

Similarly to distance-based methods, kernel-based methods, such as [71, 200], also place restrictions on the definition of $K(B_i, B_j)$: a kernel matrix has to be positive semi-definite. In a dissimilarity approach, this is not the case. Any (dis)similarity function can be used, which increases the set of informative similarity functions [8]. This is particularly useful if expert advice is to be incorporated in the pattern recognition problem.

Furthermore, kernel-based methods expect a square matrix, therefore for N training bags, all N^2 values need to be available, which is not strictly necessary in a dissimilarity-based approach. In practice we only need dissimilarities to a subset of size M of the training bags. For instance, by choosing $M = log(N)$ and choosing the prototypes beforehand (randomly or by including expert advice), the cost of computing the matrix (and subsequently training the classifier) could greatly be reduced, while still producing good performances [134].

Instance similarity

MILES [36] uses a similarity function between bags and all instances in the training set to create a high-dimensional representation for bags. A sparse 1-norm SVM is then used to simultaneously maximize the bag margin, and select the most discriminative similarities, i.e., instances. Conceptually this approach is similar to ours, however there are crucial differences:

- Using bags, rather than instances, as prototypes, significantly reduces the dimensionality. Furthermore, considering instances in a bag jointly helps to capture interactions between instances, which are lost when considering each instance independently.
- Using a radial basis function as similarity assumes that the informative instances

are found in clusters, and that only small distances (inside a cluster) can be informative, as sufficiently large distances will be set to zero by the kernel. In cases where the cluster assumption is not correct, using the distances directly still leads to informative directions in the dissimilarity space.

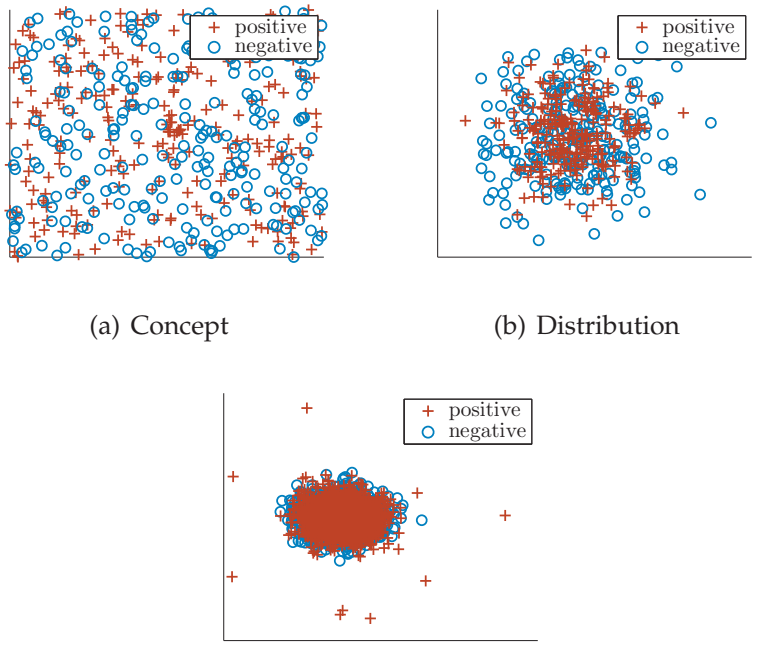
To illustrate these points more clearly, we introduce some artificial datasets in Fig. 3.5. The Concept dataset [114] is a traditional MIL dataset where positive bags have 1 concept instance and $(S-1)$ background instances, and negative bags have S background instances. Due to the dense concept in the middle, (dis)similarities to the concept instances are informative. MILES offers advantages in the Concept dataset, because only a few instances are informative, and each positive bag has such an informative instance.

The Distribution dataset also has bags with S instances each. Here, the bag as a whole is a more discriminative source of information than a particular instance, because the distributions overlap. Therefore, it is better to consider the instances in each bag jointly, as in our approach. On the other hand, MILES may try to select the negative instances that least overlap with the positive class as informative. However, because the sparsity (number of selected instances) cannot be controlled explicitly, and the high dimensionality, MILES might select too few instances to classify the bags correctly. Furthermore, the non-zero coefficients (the selected instances) may be unstable, leading to high variance in performance if the training set changes.

In the Multi-concept dataset, there are also several possible concepts outside the main concentration of instances, but only one of these concepts needs to be satisfied for the bag to be positive. If only a small number of bags is available, it may be the case that each of the concepts is satisfied only by one bag. This bag's concept instance, $\mathbf{x}_c$, is then at a relatively large distance to all other instances. The similarity to $\mathbf{x}_c$ will be set to zero for all bags, turning an informative instance into a non-informative feature. On the other hand, using the distance of $\mathbf{x}_c$ directly creates a feature that reflects the property "positive bags have an instance that is at a large distance from all bags", which is very informative for this data.

3.4 Multiple Instance Datasets

We have gathered a range of datasets, where the tasks vary from image classification to text categorization to predicting molecule activity – all applications where multiple instance learning has been thought to be beneficial. We believe that, due to the way the datasets are generated (feature extraction, sampling of instances, sampling of bags), not all of these datasets may behave in the same way as MIL has originally been defined. We therefore attempt to cover as many of such situations as possible in order to examine how different MIL methods perform in each case. A list of dataset properties is shown in Table 3.1, the datasets themselves (in MIL toolbox [169]) format can be found on our



(a) Concept (b) Distribution

(c) Multi-concept

Figure 3.5: Artificial datasets Concept (C), Distribution (D) and Multi-concept (M). The informative instances for these datasets are as follows: C - only the positive instances in the middle, D - all the instances, M - the outlying instances. In the plots, + and ∘ are instances of positive and negative bags respectively.

website http://www.miproblems.org.

The Musk datasets [49] are molecule activity prediction problems, where bags are molecules and instances are different conformations of these molecules. The shape of a conformation is responsible for its binding properties, the feature vectors therefore describe the surface properties of the conformations. The standard MIL assumption holds here: as soon as at least one of the conformers has a musky smell, the molecule is classified as having the "musky smell" property.

African and Beach are both from the Corel scene classification data [36]. It has been pointed out [36] that, for categories such as "Beach", there is not just a single concept: both "water" and "sand" probably need to be present. This means that the assumption that a single positive instance is sufficient to determine the bag label is not correct. For other categories, it is even difficult to imagine what the concepts might be. For instance, the class "Historical buildings" contains images of the ancient Greek and Roman structures, but also of the interiors (floors, staircases) of buildings from a much later period, that do not seem to share any of the same visual queues.

Table 3.1: MIL datasets, number of bags, dimensionality, number of instances and the average, minimum and maximum number of instances per bag. The datasets are available online at `http://www.miproblems.org`

Dataset	+bags	-bags	dim	inst	avg	min	max
Musk 1	47	45	166	476	5	2	40
Musk 2	39	63	166	6598	65	1	1044
Fox	100	100	230	1302	7	2	13
Tiger	100	100	230	1220	6	1	13
Elephant	100	100	230	1391	7	2	13
African	100	1900	9	7947	8	2	13
Beach	100	1900	9	7947	8	2	13
AjaxOrange	60	1440	30	47414	32	31	32
Web1	21	92	5863	2212	29	4	131
Web4	87	26	5863	2212	29	4	131
Alt.atheism	50	50	200	5443	54	22	76
Comp.graphics	49	51	200	3094	31	12	58
Brown Creeper	197	351	38	10232	19	2	43
Winter Wren	109	439	38	10232	19	2	43

The AjaxOrange datasets originates from the SIVAL [144] data. The original contains images of 25 different objects, such as a bottle of dish soap or an apple, and each object is shown in front of different backgrounds, with varying orientation and lighting conditions. Ideally, the whole object should be the concept. However, some objects, such as the bottle, are difficult to segment, causing several concepts (parts of that object) to emerge. Although this may seem similar to the "multiple concepts" situation as in scene classification, here orientation and lighting conditions may influence which of the object parts are actually seen in the image.

In the bird song datasets Brown Creeper and Winter Wren [19], a bag is an audio fragment consisting of bird songs of different species. Whenever a particular species is heard in the fragment, the bag is positive for that category. It could be expected that birds of the same species have similar songs, therefore there should be different concepts for different bird species. It is also possible that some species are heard together more often. Indeed, there are correlations up to 0.7 between the labels of some species. In this case, instances which are negative for one species, could still be helpful in classifying fragments as containing that species or not.

In Newsgroups [200], a bag is a collection of newsgroup posts or messages. At the first glance, it seems that this is a typical Concept-type dataset: a positive bag for the category "politics" contains 3% of posts about politics, whereas negative bags contain only posts about other topics. What is different here, is that posts about politics may have nothing in common and thus be very far apart in the feature space. In other words, there are several concepts, but it is sufficient to satisfy only one of these concepts. The

Multi-concept dataset may be a reasonable approximation for this dataset.

In Web Recommendation [199], a bag is a webpage, and instances are other webpages that the first webpage links to. The task is to predict whether to recommend a particular website (bag) to a user based on the linked content, or not. The websites in each of the datasets are the same, but the labels are different for each user that gave his or her preferences. This suggests that here, too, there might be multiple concepts (e.g. content about cooking, and content about sports), but that not all of these need to be satisfied in order for a user to like a webpage. In other words, a user would probably like webpages that link to either cooking, or sports, or both. The dataset has train and test sets defined, for our purposes we concatenated these datasets and use all the data in cross-validation.

3.5 Experiments

We start with a comparison of the proposed bag dissimilarities in Sections 3.5.1 and 3.5.2. In Section 3.5.3, a more in depth analysis of the characteristics of the dissimilarities of real world data is given. A subset of these bag dissimilarities is then compared to other MIL approaches in Section 3.5.4. The metric used for comparisons is area under the receiver-operating characteristic (AUC) [16]. This measure has been shown to be more discriminative than accuracy in classifier comparisons [82], and more suitable for MIL problems [170].

For intermediate experiments, we use a subset of the datasets from Table 3.1 because datasets from the same source (such as one-against-all datasets with a different positive class) are expected to show similar behaviour. For the final comparison, all the datasets are used.

In this section, the notation D is used to denote the full dissimilarity matrix, as opposed to d, which stands for a single dissimilarity value between two bags.

Each experiment is performed using cross-validation, where in each fold, the bags are split up into N_{tr} train and N_{te} test bags. The train bags are then used as prototypes to compute the $((N_{tr} + N_{te}) \times N_{tr})$ bag dissimilarity matrix D, i.e., no prototype selection is performed. The only dissimilarity measure where a parameter needs to be set is d_{CS}. Here we used a default value, square root of the dimensionality) for all the datasets. It is therefore possible that these results could be improved, however at the added cost of cross-validating over different values for σ.

In this dissimilarity space, any supervised classifier can be applied; in this paper we use the logistic and support vector (SVM) classifiers. On average, the support vector classifier results were superior to those of the logistic classifier, therefore in some of the further experiments, only SVM results are reported. The classifiers are used with default parameters: regularization parameter $C = 1$, and the SVM is used with a linear

kernel unless stated otherwise.

3.5.1 Point Set Dissimilarities

The first comparison is between the bag dissimilarities that are most closely related to the Hausdorff distance: the symmetrized versions of D_{minmin}, $D_{meanmin}$ and D_{maxmin}. The artificial datasets Concept, Distribution and Multi-concept from Fig. 3.5, denoted by C, D and M respectively, are very suitable to demonstrate the strengths and weaknesses of these bag dissimilarities. The success of a bag dissimilarity is determined by whether it allows the informative instances (those that cause the differences between positive and negative bags) to sufficiently influence the dissimilarity value. Because the overall mean dissimilarity in (3.4) is naturally symmetric, we also include it in this comparison.

Table 3.2 shows performances of the bag dissimilarities for the artificial (at two different sample sizes of 25 and 50 bags per class) and seven real-life MIL problems.

D_{minmin} dissimilarity is best suited for the Concept dataset, as only the minimum instance distances inside the concept are informative. $D_{meanmin}$ is expected to be best for the Distribution dataset, because most/all of the instances need to be considered to determine the bag label. D_{maxmin} dissimilarity is most suitable for the Multi-concept dataset, as the maximum operation selects the most informative, remote instances. Although adding more training data benefits most dissimilarities, one of the dissimilarities is doing better than the others. $D_{meanmean}$ performs well on the Distribution and Multi-concept problems it captures the same important characteristics as $D_{meanmin}$ and D_{maxmin} in these cases.

On the real-life datasets, the dissimilarities based on minimum instance distances have comparable performance. This suggests that the distributions of instances from positive and negative bags are different enough that any statistic of the instance distances is able to separate the classes well. Fig. 3.6 shows a 2D projection (obtained by multi-dimensional scaling) of the instances in the Musk 1 dataset. For example, the positive bags that have instances in the center cluster, are slightly closer to each other, than to other bags, under D_{minmin}, creating informative dissimilarities. However, other statistics of the instance distances also separate the classes quite well.

In AjaxOrange and alt.atheism datasets, D_{minmin} performs very badly. In these datasets, positive and negative bags are likely to contain exactly the same instances (background objects in AjaxOrange and very general, uninformative posts in alt.atheism). For example, in the right plot of Fig. 3.6, we see that the alt.atheism dataset has a cluster of instances in the middle (both from positive and negative bags) and outlying instances from positive bags, similar to the Multi-concept dataset. This is caused by the use of word frequency features: instances containing unusual words are far away from all the

Table 3.2: Point set, symmetrized dissimilarity, logistic and SVM classifiers. AUC and standard error ($\times 100$), 5×10-fold cross-validation. Bold = best (or not significantly worse) result per dataset.

Classifier	Data	D_{minmin}	$D_{meanmin}$	D_{maxmin}	$D_{meanmean}$
Logistic	C25	**61.4 (2.3)**	54.8 (2.2)	47.5 (2.2)	53.4 (2.1)
	C50	**98.6 (0.4)**	79.6 (1.9)	50.3 (3.3)	65.6 (3.0)
	D25	86.2 (1.8)	**97.8 (0.5)**	96.6 (0.6)	69.4 (2.4)
	D50	91.7 (1.1)	**100.0 (0.0)**	99.6 (0.2)	**100.0 (0.0)**
	M25	54.4 (2.2)	50.8 (1.9)	60.5 (2.3)	**71.4 (1.9)**
	M50	71.5 (2.4)	78.4 (2.1)	**84.3 (1.5)**	69.4 (2.4)
	Musk1	88.2 (1.8)	**90.2 (1.7)**	**91.8 (1.6)**	84.3 (1.8)
	Musk2	92.0 (1.2)	**92.6 (1.3)**	**93.3 (1.2)**	82.8 (1.7)
	African	**96.3 (0.4)**	94.4 (0.6)	94.2 (0.6)	91.1 (0.7)
	Ajax	68.4 (1.6)	**98.1 (0.5)**	**97.2 (0.7)**	87.8 (1.1)
	Alt.ath	49.2 (0.8)	**88.5 (1.7)**	83.7 (1.7)	85.2 (1.8)
	BrCr	89.6 (0.6)	**93.6 (0.4)**	91.1 (0.5)	82.3 (0.7)
	Web	69.7 (4.0)	**77.0 (3.2)**	66.8 (3.7)	69.9 (3.3)
LibSVM	C25	**57.7 (2.4)**	52.2 (2.6)	45.9 (2.1)	40.8 (1.5)
	C50	**98.6 (0.4)**	83.9 (2.0)	46.1 (2.4)	66.4 (3.0)
	D25	72.0 (2.4)	**78.9 (2.0)**	**82.7 (1.6)**	68.2 (2.5)
	D50	92.9 (1.1)	**100.0 (0.0)**	99.5 (0.2)	**100.0 (0.0)**
	M25	47.0 (2.2)	50.6 (2.3)	66.2 (2.3)	**71.6 (2.0)**
	M50	72.0 (2.4)	**78.9 (2.0)**	**82.7 (1.6)**	68.2 (2.5)
	Musk1	92.0 (1.2)	**93.4 (1.2)**	**93.4 (1.3)**	88.2 (1.5)
	Musk2	94.0 (1.3)	**95.4 (1.4)**	**95.3 (1.2)**	90.3 (1.5)
	African	**96.6 (0.3)**	**96.7 (0.4)**	95.5 (0.5)	90.1 (0.7)
	Ajax	71.1 (1.4)	**98.6 (0.4)**	**97.8 (0.5)**	84.0 (1.2)
	Alt.ath	50.0 (0.0)	**94.9 (1.0)**	91.4 (1.1)	94.2 (1.1)
	BrCr	87.8 (0.6)	**95.5 (0.3)**	92.6 (0.5)	54.4 (2.4)
	Web	53.2 (4.8)	**76.0 (2.7)**	43.3 (3.6)	**77.6 (3.3)**

others, and instances containing only regular words are very close to most others, i.e., in the middle cluster. Because the instance density in this cluster is very high, D_{minmin} always "selects" instances from this cluster. This results in many bag dissimilarities being equal to zero, creating large class overlap.

Although the performances on the real datasets are reasonable, $D_{meanmean}$ does not do as well as the other dissimilarities. This suggests that in real problems, the distributions of instances from positive and negative bags might be somewhat different, but that this is not as pronounced as in the artificial Distribution and Multi-concept cases. Therefore, the minimum distance matches for all the instances provide more reliable information about the bag class. An exception is alt.atheism, where the performance of $D_{meanmean}$ is quite high. This can also be explained with the plot in Fig. 3.6. Here we see outlying positive instances, all belonging to different positive bags, similar to the Multi-concept dataset. These instances are the most informative for the bag class, therefore dissimilarities involving these distances can perform well. This is the case for $D_{meanmin}$, D_{maxmin} and $D_{meanmean}$, which is reflected in the performances.

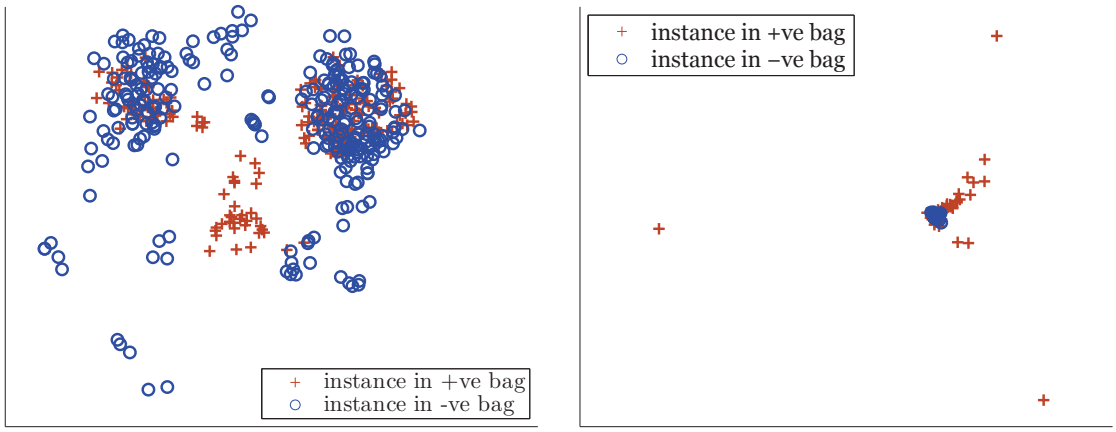


Figure 3.6: Multi-dimensional scaling projection of instances in the Musk 1 (left) and alt.atheism (right) datasets

Overall, $D_{meanmin}$ gives the best results. Although it is possible that only a few instances per bag are informative (as in the Concept artificial dataset), considering all instances (and their minimum distance neighbors) with equal weight seems to be already be sufficient in practice.

3.5.2 Distribution Dissimilarities

Table 3.3 shows the results of the distribution dissimilarities for the logistic and LibSVM

Table 3.3: Distribution dissimilarities. AUC and standard error ($\times 100$), 5×10-fold cross-validation. Bold = best (or not significantly worse) result per dataset.

Classifier	Data	D_{Maha}	D_{EMD}	D_{CS}
Logistic	Musk1	70.1 (2.3)	**88.7 (1.9)**	84.6 (2.0)
	Musk2	**91.5 (1.3)**	-	88.8 (1.5)
	African	59.9 (1.5)	**93.3 (0.6)**	85.9 (1.0)
	Ajax	86.9 (1.8)	**97.9 (0.4)**	95.5 (0.7)
	Alt.ath	49.9 (2.5)	**84.0 (1.9)**	59.2 (2.9)
	BrCr	63.3 (1.2)	**94.5 (0.4)**	89.4 (0.8)
	Web	-	69.4 (4.1)	**75.7 (3.4)**
LibSVM	Musk1	76.5 (2.9)	**89.8 (1.6)**	**88.2 (1.7)**
	Musk2	**96.0 (0.9)**	-	87.4 (1.8)
	African	64.8 (1.5)	**94.7 (0.4)**	**94.3 (0.5)**
	Ajax	87.3 (1.7)	**98.9 (0.3)**	98.1 (0.3)
	Alt.ath	47.0 (2.3)	**87.4 (1.7)**	41.9 (2.5)
	BrCr	59.7 (1.1)	**95.5 (0.3)**	93.9 (0.4)
	Web	-	**77.7 (2.7)**	69.5 (3.8)

classifiers. The result of Musk2 for D_{EMD} is missing because EMD does not scale well to large bags (some bags in Musk2 have 1000+ instances). The result of Web recommendation for D_{Maha} is missing because the dataset has almost 6000 dimensions and computing the dissimilarity therefore requires many inversions of 6000×6000 covariance matrices.

Overall, the results of D_{Maha} are not very good, except for Musk2. A possible reason could be that Musk2 is the only dataset with bags that are large enough for reliable estimation of an inverse covariance matrix. The results of EMD are quite good, however, for these datasets it does not offer clear advantages over the point set dissimilarities. Lastly, the results of d_{CS} are also reasonable, which is a surprise considering that the results are quite sensitive to the σ parameter.

It is interesting to compare the results of D_{CS} and $D_{meanmean}$ from the previous section. One of the main differences is that in D_{CS}, more emphasis is put on the smaller distances (due to the RBF kernel), while in $D_{meanmean}$ large distances influence the dissimilarity values significantly. This explains the large gap in the results of alt.atheism, where the outlier instances are very important. On the other hand, D_{CS} is much better than $D_{meanmean}$ on the Brown Creeper dataset, where a tight concept, and therefore small distances between concept instances, can be expected.

3.5.3 Properties of Dissimilarity Matrices

Table 3.4 shows several properties of dissimilarity matrices evaluated in our experiments. These properties demonstrate what we have emphasized in Section 3.3: it is not our goal to arrive at a definition of $d(B_i, B_j)$ that is consistent with one intuition about distance measures, and that dissimilarities that do not behave as distances, may still be informative for classification. The properties we examine [57, 137] are defined as follows:

- NEF stands for negative eigenfraction, and express how well the dissimilarity matrix can be embedded in a Euclidean space. For a Euclidean dissimilarity matrix, the eigenvalues λ_i of the corresponding Gram matrix are non-negative [137]. Negative λ_i therefore represent non-Euclidean behavior. NEF is defined as $\frac{\sum_{\lambda_i<0}|\lambda_i|}{\sum_{\lambda_j}\lambda_j|}$. For a Euclidean distance measure, the value is 0.

- NER stands for negative eigenratio, and is defined as $\frac{|\lambda_{min}|}{\lambda_{max}}$. For a Euclidean distance measure, the value is 0.

- NMF stands for non-metricity fraction. It is the percentage of triplets $\{D_{i,j}, D_{i,k}, D_{j,k}\}$ that disobey the triangle inequality. For a metric distance measure, the value is 0.

One interesting result is for D_{CS} with highly non-Euclidean / non-metric behaviour for Musk2 and Web data. We suspect that the cause of this behaviour is the highly varying bag size in these datasets. The computation of D_{CS} in (6) uses a sum (not an average) of similarity values. Therefore, it is possible that bags B_i, B_j and B_k, with instances in the exact same locations in the feature space, but where B_k is much better sampled than B_j, and B_j is much better sampled than B_i, are at different distances to each other. This difference would not picked up by any of the point set dissimilarities, because averaging is employed. It depends on the application whether this is a favorable property or not. For example, if the bag size is correlated with the bag label, a sum of instance distances could be more informative than an average. This is not the case for Musk 2 or Web, but it could be relevant for other data, for example in Brown Creeper, where the correlation between bag size and bag label is higher than 0.5.

Another unusual result is the NEF and NER obtained on the alt.atheism dataset for D_{mean}. The positive bags in this dataset contain very outlying instances, as explained in Section 3.5.1. The dissimilarities of bags with such outlier instances will be very large, which which causes significant problems to embed such objects in a Euclidean space. This causes large negative eigenvalues, and therefore large values for NEF and NER. Of course, this behaviour is also non-metric, however, NMF only measures the amount of dissatisfied triangle inequalities, disregarding the identity requirement of a metric.

Table 3.4: Properties of dissimilarity matrices: negative eigenfraction (in %), negative eigenratio and non-metricity fraction (in %).

Measure	Data	D_{minmin}	$D_{meanmin}$	D_{maxmin}	D_{mean}	D_{Maha}	D_{EMD}	D_{CS}
NEF	Musk1	31.2	25.5	22.6	21.0	48.9	27.4	27.1
	Musk2	31.1	25.3	21.7	20.5	28.4	-	96.3
	Afr	47.2	37.8	33.1	29.7	50.0	49.5	35.4
	Ajax	47.9	32.6	41.3	40.0	49.6	29.9	32.5
	Alt	49.5	21.0	22.0	93.1	36.9	37.4	9.2
	BrCr	45.5	30.7	33.6	51.1	49.9	37.3	27.3
	Web	18.6	4.8	3.7	25.9	-	10.3	100.0
NER	Musk1	0.4	0.2	0.2	0.2	1.0	0.2	0.2
	Musk2	0.4	0.2	0.2	0.1	0.3	-	28.1
	Afr	0.7	0.4	0.3	0.3	1.0	1.0	0.3
	Ajax	0.5	0.2	0.3	1.1	1.0	0.2	0.3
	Alt	1.0	0.5	0.6	24.1	0.8	0.7	0.8
	BrCr	0.5	0.4	0.3	0.8	1.0	0.6	0.2
	Web	1.0	1.1	0.9	1.1	-	0.5	0.0
NMF	Musk1	5.5	2.5	1.6	0.7	27.1	3.8	5.5
	Musk2	7.2	3.2	1.9	0.8	7.9	-	93.3
	Afr	20.1	8.1	8.8	2.3	27.8	11.4	12.4
	Ajax	17.5	0.1	0.5	0.0	18.9	0.3	4.2
	Alt	0.0	0.3	0.9	0.0	1.7	9.7	0.0
	BrCr	7.8	0.8	2.0	4.2	26.7	3.5	13.1
	Web	2.4	0.0	0.0	0.0	-	0.0	97.3

Overall, these results show that dissimilarity matrices do not need to be Euclidean or metric to still be informative and useful for dissimilarity-based classifiers, which is consistent with the conclusions in [136]. Of course, these measures do not take into account the actual labeling, so such properties alone cannot predict the performance of classifiers in the dissimilarity space. However, examining the dissimilarity matrices can aid in understanding more about the data and the different choices of dissimilarity measures.

3.5.4 Comparison to Other MIL Approaches

We have selected several methods, that are often being used in comparisons in recent papers, as algorithms for our own comparison. For a more detailed description of these methods, see Section 3.2. We consider $d_{meanmin}$ as the dissimilarity function used in MInD, and LibSVM [33] as the classifier used together with this representation. All classifiers are implemented in PRTools [56] and the MIL toolbox [169]. Default parameters are used for all cases, unless stated otherwise. The classifiers used are as follows:

- EM-DD [195], with 10 objects used at initialization.
- mi-SVM [5], with a linear kernel.
- MILBoost [183] with 100 rounds.
- MILES [36], with a radial basis kernel.
- Minimax MI-Kernel [71] + SVM with a linear kernel.
- MInD + SVM with a linear kernel.

Learning Curves

The presented classifiers vary significantly in the model assumptions and their complexity. Therefore, we expect that these methods are affected differently by the amount of training data provided, both in terms of classifier performance, as well as computational issues. We provide several learning curves (in the number of training bags) and show the performance and training/testing times of the classifiers.

The learning curves are generated as follows. For each of the 20 iterations, the dataset is split into 80% training and 20% test bags. Then, the training set is subsampled to contain 5, 10, 20 and 40 (or the maximum possible number of) bags per class. This ensures that the test set remains the same for increasing amounts of training data. When the training set is not large enough, the maximum possible number of bags is sampled instead.

The learning curves for six MIL datasets are shown in Fig. 3.7. In terms of AUC, it is clear that MInD and MILES perform well overall: their performance is always one of the best. There are two exceptions where the results of MInD and MILES are quite

different. In the alt.atheism dataset, MInD outperforms MILES significantly. We suspect that this is because this dataset is similar to the artificial Distribution data - all instances need to be considered to distinguish a positive bag from a negative bag. MInD does this naturally, while MILES tries to enforce sparsity by selecting a few important instances. On the other hand, MILES is superior on the AjaxOrange dataset. This is because MInD takes uninformative background distances into account, while MILES is able to select the instances belonging to the AjaxOrange bottle.

There is another interesting difference between the MInD and MILES approaches. In all cases except AjaxOrange, where MILES is always superior, the advantage of MInD is more apparent at lower sample sizes. MILES has a higher dimensionality than MInD, which is a disadvantage when only a few bags are available in the training set. This is why the performance of MILES decreases for alt.atheism. As the training set becomes larger, so does the dimensionality, making the problem of selecting informative instances more and more difficult.

The Minimax and SimpleMIL approaches also have reasonably good (but typically worse than MILES and MInD) and consistent performances. Both approaches do not require any extra parameters, except the classifier used on top of the representation. EM-DD is able to achieve good results, but only at larger training sizes, as estimating a high-dimensional density requires a lot of data. For MILBoost and mi-SVM, it is not directly clear when a good performance can be achieved. Interestingly, both classifiers try to recover the true instance labels, assuming that at least one instance per bag is positive. As explained in Section 3.4, this may not be the case for several of the MIL problems examined here. We suspect that this is the reason why methods that learn on bag level and thus make less assumptions, are more successful in our experiments.

In terms of time (Fig. 3.8), EM-DD is by far the slowest, followed by MILBoost and mi-SVM due to the optimization of the instance labels. They are followed by MInD and MILES, where creating the (dis)similarity representation is quadratic in the number of instances in the training data. SimpleMIL and Minimax are the fastest methods, because creating the representation is linear in the number of instances.

When both the performance and training time are taken into account, Minimax, MInD and MILES are the best choices. Although MInD and MILES have higher performances, they are significantly slower. Selecting bags or instances prior to creating the dissimilarity representation could, however, decrease their total training time. Prototype selection after the dissimilarity matrix is already available would increase the training time, but would decrease testing time as less dissimilarities would need to be computed in the test phase.

Another useful observation is that MILES and bag dissimilarities such as MInD are related because both representations are defined through minimum instance distances. Therefore, the quadratic cost of creating the representation can be shared among these methods, which allows a user to try several competing methods without adding signif-

icant computational effort. The same holds for trying several classifiers at a fraction of the cost. In the time curves, the time for computing the representation is included in the time that is displayed, so in practice, these methods would be faster.

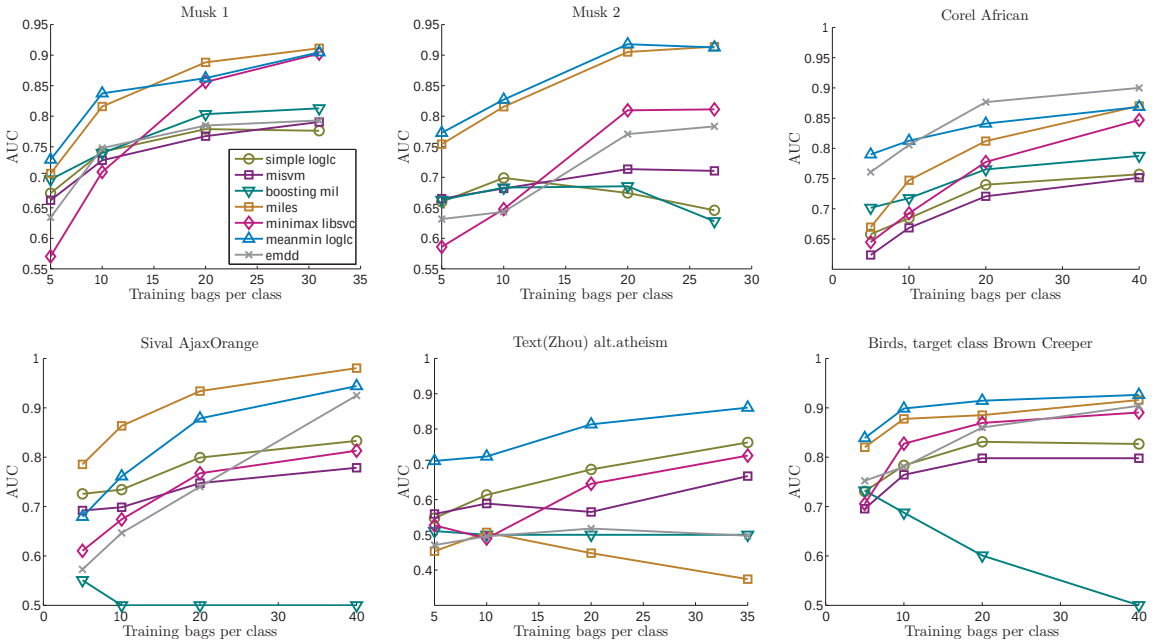


Figure 3.7: Learning curves (AUC performance) for Musk1, Musk2, African, AjaxOrange, alt.atheism and Brown Creeper datasets. The standard deviations are generally around 0.1, but are lower for MILES, Minimax and MInD at larger training sizes. Figure best viewed in color.

Comparison

As a final comparison, we present the results of 5 times 10-fold stratified cross-validation per dataset for all the available data. Unfortunately, some results cannot be reported: for EM-DD when one cross-validation fold lasts longer than five days; for MILES when there are too many instances in the training set; for MILBoost when features with the same value for all instances are present. Furthermore, for several computationally intensive methods it was infeasible to perform the experiment 50 times, therefore these performances are based on fewer runs.

For MILES, we use a radial basis kernel with $\sigma = 10$ as the instance similarity function. MiSVM, Minimax and MInD are all used with an SVM with a linear kernel and regularization parameter $C = 1$.

The results are shown in Table 3.5. Overall, the best results are given by MILES and MInD. Note that MILES is sensitive to the choice of width parameter σ, and for some

Table 3.5: Comparison of different MIL approaches, AUC and standard error ($\times 100$), 5×10-fold cross-validation. Significance is determined by the Friedman test, for which the critical difference is 2.0153. Classifiers in bold are best, or not significantly worse than best.

	Classifier					
Data	EM-DD	mi-SVM	MILBoost	MILES	Minimax	MInD
Musk1	87.4 (2.1)	81.3 (2.5)	74.3 (2.6)	92.8 (1.2)	89.1 (1.9)	93.4 (1.2)
Musk2	86.9 (2.1)	81.5 (2.1)	73.6 (2.3)	95.3 (0.8)	89.0 (1.5)	95.4 (1.4)
Fox	67.6 (3.2)	53.9 (1.6)	61.1 (1.9)	69.8 (1.7)	58.1 (1.3)	60.5 (1.9)
Tiger	75.4 (2.9)	83.3 (1.3)	84.1 (1.6)	87.2 (1.6)	81.4 (1.3)	85.1 (1.7)
Elephant	88.5 (2.1)	84.1 (1.4)	89.0 (1.4)	88.3 (1.3)	88.2 (1.0)	93.1 (0.8)
African	91.5 (1.0)	63.4 (1.2)	88.9 (0.9)	58.9 (1.7)	84.5 (1.5)	96.7 (0.4)
Beach	84.7 (1.3)	49.6 (1.6)	85.0 (1.1)	60.0 (1.9)	82.4 (0.9)	92.3 (0.6)
AjaxOrange	-	93.6 (1.1)	97.9 (0.5)	-	91.1 (0.9)	98.6 (0.4)
Alt.atheism	51.0 (5.2)	70.9 (2.6)	-	47.1 (2.4)	80.6 (1.8)	94.9 (1.0)
Comp.graphics	48.2 (3.2)	59.3 (2.8)	56.3 (2.6)	57.2 (2.6)	57.1 (2.7)	92.2 (1.4)
BrownCreeper	94.5 (0.9)	85.8 (0.7)	95.4 (0.4)	95.8 (0.3)	94.1 (0.4)	95.5 (0.3)
WinterWren	98.5 (0.3)	95.3 (0.4)	97.0 (1.5)	99.2 (0.2)	98.1 (0.2)	99.5 (0.1)
Web1	-	89.7	77.8 (5.7)	88.2 (4.7)	90.4	76.0 (2.7)
Web4	60.6 (1.1)	81.2	61.8 (4.9)	70.8 (1.6)	86.7	73.7 (3.2)
Friedman	4.1786	4.3571	3.9286	**3.1786**	**3.5714**	**1.7857**

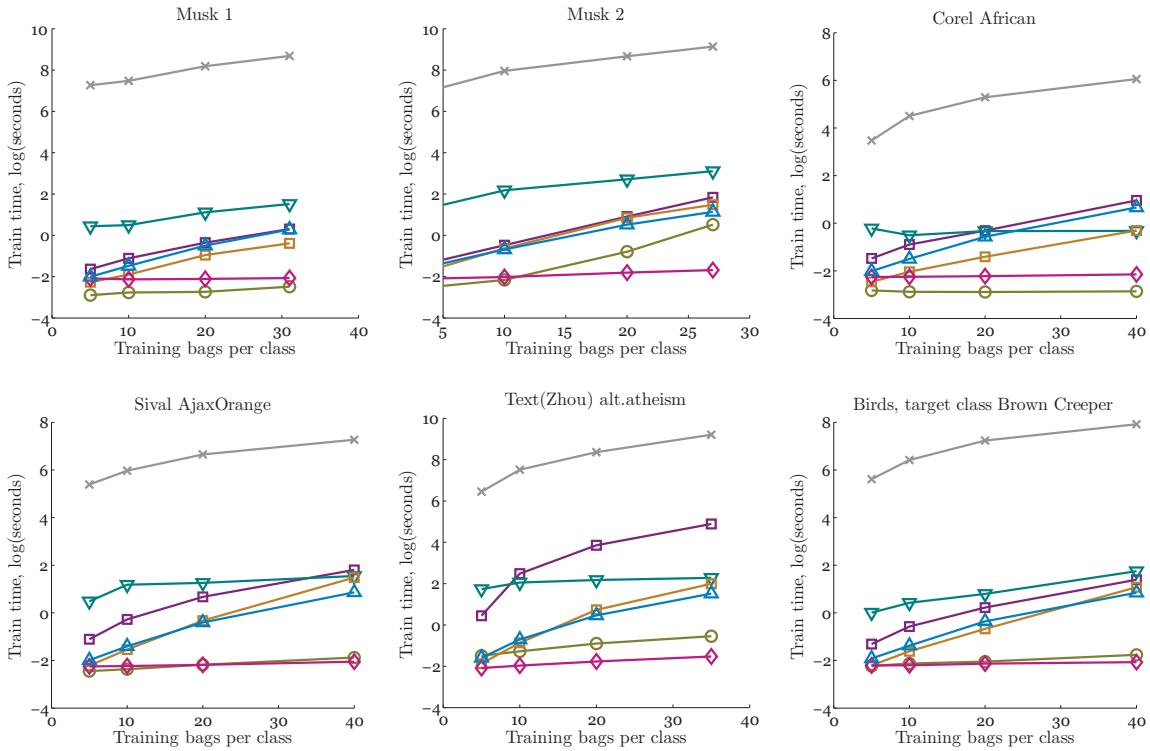


Figure 3.8: Training time curves for Musk1, Musk2, African, AjaxOrange, alt.atheism and Brown Creeper datasets. The standard deviations in time are all quite low except for EM-DD. Figure best viewed in color.

datasets, the default value results in poor performance. We also performed experiments with MILES with a linear kernel, and the performances were more stable across datasets, however, in general lower and significantly worse than MInD.

The last row shows the results of the Friedman test [47]. We have treated the missing results as random performance (AUC of 0.5) to obtain the classifier ranks. The ranks show that MInD is the best performing classifier, followed by MILES and Minimax, which are both not significantly different from MInD due to the critical difference of 2.0153 (14 datasets, 6 classifiers). MInD, however, is significantly better than the other four classifiers, which is not the case for MILES or Minimax.

The only methods that (i) always produce a reasonably good performance and (ii) always produce a result, are Minimax and MInD. Our advice for a different MIL problem would therefore be to try the Minimax, MInD and, if the dimensionality allows it, MILES. An additional benefit of these choices is that the costs of creating the MInD and MILES dissimilarity matrices can be shared, because both are based on instance distances.

3.6 Recommendations

In this section we provide several recommendations with respect to dealing with novel MIL problems, and in particular the bag dissimilarity approach.

3.6.1 Dissimilarity Measure

Based on our observations, $d_{meanmin}$ is a reasonable choice for many datasets. Although we can design datasets where $d_{meanmin}$ would fail, such as the Concept dataset in Fig. 3.5, we have not encountered such datasets in practice. However, our advice would be to inspect 2D or 3D projections of the instances (or a subset of the instances in case of very large datasets), to see whether there are similarities between the novel dataset and any existing toy or real problems. Based on such observations, one could reason whether a particular dissimilarity function would be more or less successful.

Another possibility is to have an expert involved in defining the dissimilarity, for instance, the underlying instance distance function could be replaced by an application-specific measure, such as an alignment measure for strings. This is likely to result in non-metric dissimilarities, which, as we have demonstrated in Section 3.5.3, is not a problem for the dissimilarity approach.

For a novel MIL problem, it is necessary to consider properties of the dataset (number of bags, bag size, and the dimensionality of the instances), both in terms of performance and computational complexity. For small bags, it is difficult to estimate the instance distribution well, especially if the feature space is high-dimensional. In this case, the point set measures can be preferable. For large bags, the instance distribution can be estimated more reliably. Furthermore, the point set dissimilarities may become too expensive to compute, so a distribution measure can be preferred. In a situation where both small and large bags are present, it is important to check whether the the bag size is informative for the problem (such as a bird species singing only when a lot of birds a singing), or an artefact (such as errors in the data generation procedure).

3.6.2 Classifier

A dissimilarity representation can, in principle, be used as an input for any supervised classifier. Of course, the number of objects (bags) and the number of features (prototypes) should be taken into consideration, as in any pattern recognition problem. For instance, it might not be advisable to use a very complex classifier when the number of bags is very low. In this paper, we only performed experiments with the logistic and support vector classifiers, but in other experiments, Parzen, dissimilarity-based nearest

neighbor, 1-norm SVM and others have also been successful [40, 171]. This offers flexibility to a (possibly non-expert) user, who might have a preference for a certain classifier. Using sparse classifiers (such as the 1-norm SVM) can also help interpretability: the classification result can then be explained as being (dis)similar to certain prototypes.

Another advantage of the wide choice of classifiers is that the original MIL setting (binary offline classification) can be easily transported to other learning settings. For instance, in a multi-class case with a large number of classes, it might be advantageous to use a classifier that is inherently multi-class (such as nearest neighbor) rather than combining many one-against-all binary classifiers.

3.7 Discussion and Conclusions

In this paper, we proposed a dissimilarity representation for multiple instance learning (MIL), where each bag is represented by its dissimilarities to the training bags. The problem is therefore converted to a supervised learning problem where any classifier can be used. There are many ways to define a dissimilarity between two bags, by viewing each bag as a point set, as a distribution in instance space, or as an attributed graph.

We gathered a wide range of artificial and real MIL problems and discussed which are the informative instances in each case. Through experiments, we have demonstrated that different dissimilarity definitions have different implicit assumptions about the informativeness of instances, therefore making some dissimilarities more suitable than others for the dataset in question. In practice, the dissimilarity based on averaging of the minimum instance distances between bags has shown good performance in all the real-life datasets we discussed. Our approach has shown very competitive performances to other MIL algorithms, while keeping the computational effort quite low.

Furthermore, we discussed the benefits of the proposed approach to a potential end-user. Because the dissimilarity approach does not impose restrictions on the dissimilarity matrix, expert advice can be incorporated in the dissimilarity definition. Non-metric dissimilarity measures may be more informative than their metric counterparts, and such properties can be dealt with naturally in a dissimilarity approach. Furthermore, the approach is flexible with respect to the classifier used, and can be easily extended to other learning settings.

One of the questions that is left is when to use the "point set" and the "instance distribution" approaches. Depending on the size of the bags, one of these may be more accurate than the other, however, computational issues may also become a factor. It would be interesting to investigate the exact trade-off of these two choices. Overall, we believe the proposed approach is a flexible, powerful and intuitive way to do MIL, and that combined, these qualities make it an attractive method for domains where data might be naturally grouped in bags, but MIL is not yet being used.

Addendum

In the previous work, we symmetrized the dissimilarities in order to concentrate on the main differences between D_{minmin} (which is already symmetric), $D_{meanmin}$ and D_{maxmin} dissimilarities. However, symmetric dissimilarities are not strictly necessary for a dissimilarity-based approach, and it is worth examining whether symmetrization is indeed a useful step.

Recall from Section 3.3.1 that the direction in which the dissimilarity is measured, defines which instances influence the bag dissimilarity. This begs the question of whether there are actually situations where one of the directions might be more informative. Consider the Multi-concept dataset and a negative prototype R^- from this dataset. If we measure the dissimilarity from this prototype to training bags B^+ and B^-, it could happen that $d(R^- \to B^+) = d(R^- \to B^-)$ if the dissimilarity ignores the informative instances in B^+. On the other hand, measuring from B^+ and B^- to the prototype ensures that the informative instances influence the dissimilarity value. For more detailed explanation and results, see [39].

In the experiments, we compare the directional dissimilarities to dissimilarities which are symmetrized, as well as the extended asymmetric dissimilarity space (EADS) [138], where the dissimilarity is not symmetrized but information from both directional dissimilarity matrices is included. The dissimilarities used in the comparison are as follows (note that the combining method is denoted by the superscript, unlike the dissimilarity definition, for which we use the subscript):

- D^{to}, dissimilarities measured from the objects to the prototypes.
- D^{from}, dissimilarities measured from the prototypes to the objects.
- $D^{avg} = D^{to} + D^{from}$, symmetric dissimilarity obtained by averaging the matrices.
- $D^{max} = \max(D^{to}, D^{from})$, symmetric dissimilarity obtained by performing an element-wise maximum operation.
- $D^{min} = \min(D^{to}, D^{from})$, symmetric dissimilarity obtained by performing an element-wise minimum operation.
- $D^{ext} = [D^{to} D^{from}]$, asymmetric dissimilarity obtained by concatenating the original matrices.

Parts of these results were published as:
Yenisel Plasencia-Calaña, Veronika Cheplygina, Robert P. W. Duin, Edel B. García-Reyes, Mauricio Orozco-Alzate, David M. J. Tax, and Marco Loog. "On the informativeness of asymmetric dissimilarities." In Similarity-Based Pattern Recognition, pp. 75-89. Springer Berlin Heidelberg, 2013.

Table 3.6 shows performances of these dissimilarity versions for $D_{meanmin}$ on a few real-life datasets for the logistic and LibSVM classifiers (the comparisons are performed per classifier). It is clear that it is not strictly necessary to have a symmetric dissimilarity to achieve good performance.

One of the interesting results is the difference between D^{to} and D^{from} on the alt.atheism dataset. The use of word frequency features causes some instances (those containing unusual words) to be far away from all the others, and other instances (those containing regular words) to be very close to most others. This situation is a bit similar to the artificial Multi-concept dataset: in the middle are the non-informative instances with regular words, and on the outside are the instances with unusual words. As a result, it may happen that a bag is often equidistant to all other bags. This may seem totally uninformative, but the catch is that the distance of bag B_i to all the others might be different than the distance of B_j to all the others. Considering the D^{from} and D^{to} dissimilarities, we then have the following situations: (i) a prototype is equidistant to all bags (D^{from}) or (ii) a bag is equidistant to all prototypes (D^{to}).

In the first situation, the dissimilarity is non-informative, because none of the prototypes will be able to discriminate between positive and negative bags. In the second situation, however, learning is still possible if positive bags, on average, have a different "equidistance distance" than negative bags. This is exactly the situation in alt.atheism, and the reason why D^{to} leads to higher performances. For LibSVM, the difference is especially large and the combining methods also do not succeed in outperforming this directed dissimilarity.

Learning Curves

In a different set of experiments, we examine learning curves for the different versions of the $d_{meanmin}$ dissimilarity. The datasets used are AjaxOrange (same as above) and Winter Wren (a dataset from the same source and with similar characteristics as the Brown Creeper dataset). The classifiers compared are the linear discriminant classifier (LDC) and the SVM. For LDC we use regularization parameters $R = 0.01$ and $S = 0.9$ to regularize the covariance matrices, for SVM we use a linear kernel and a regularization parameter $C = 100$. These parameters show reasonable performances on all the datasets under investigation (including non-MIL datasets not included in this thesis), and are, therefore, constant across all experiments and not optimized to fit a particular dataset.

We provide learning curves over 20 runs for each dissimilarity / classifier combination, for increasing training sizes from 5 to 30 objects per class. In each of the learning curves, the number of prototypes is fixed to either 5 or 20 per class in order to explore the behavior with a small and a large representation set size. This means that the dimensionality of the dissimilarity space is the same for D^{to}, D^{from} and the symmetrized versions, but twice as much for D^{ext}.

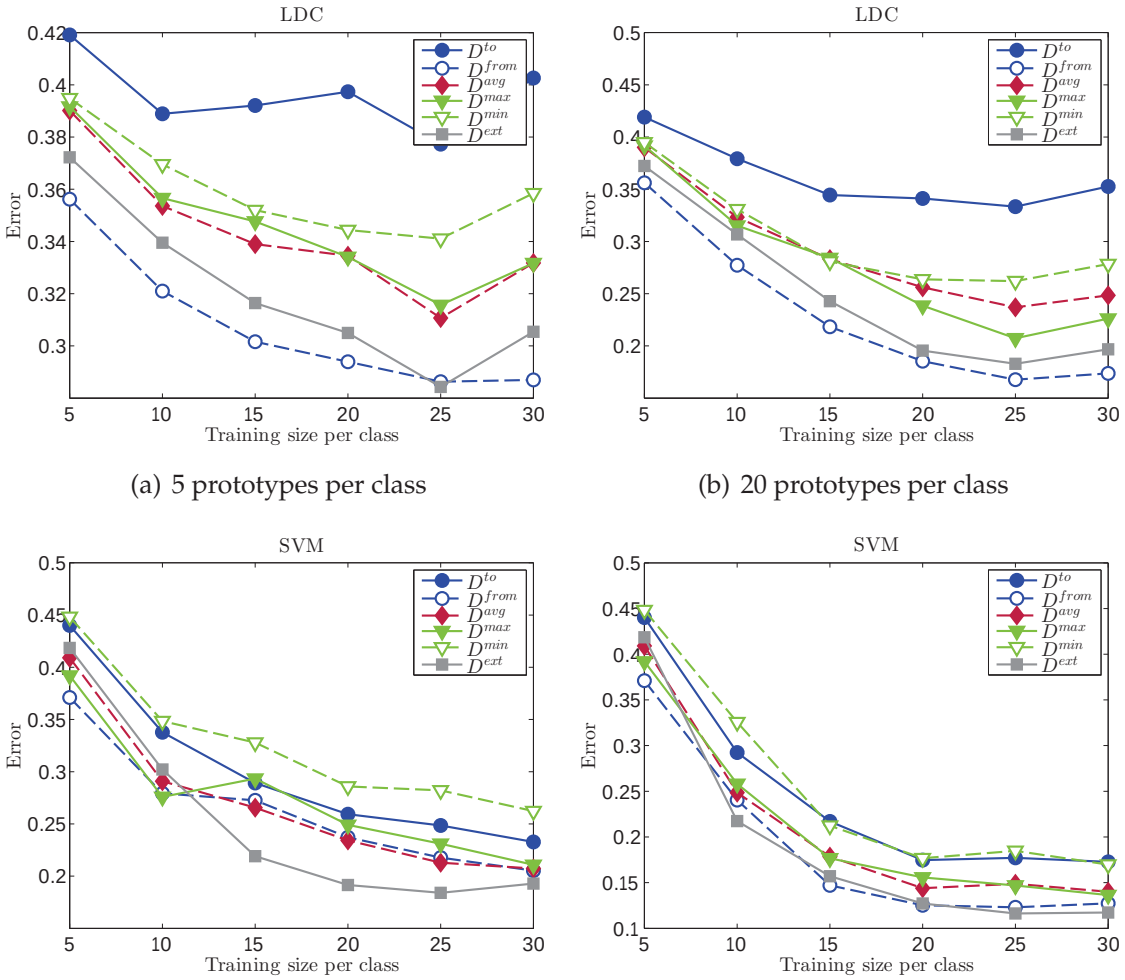
Table 3.6: Combining asymmetry information for the $D_{meanmin}$ dissimilarity. AUC and standard error (×100), 5 × 10-fold cross-validation. Bold = best (or not significantly worse) result per dataset.

	Data	D^{to}	D^{from}	D^{avg}	D^{max}	D^{min}	D^{ext}
Logistic	Musk1	**91.4 (1.5)**	**92.2 (1.4)**	90.2 (1.7)	89.6 (1.8)	90.0 (1.6)	**91.3 (1.6)**
	Musk2	**93.6 (1.3)**	91.8 (1.3)	92.6 (1.3)	**95.1 (1.0)**	89.4 (1.5)	92.9 (1.2)
	African	94.4 (0.5)	83.2 (0.9)	94.4 (0.6)	90.6 (0.9)	**95.2 (0.4)**	**95.5 (0.5)**
	Ajax	96.9 (0.7)	**98.4 (0.4)**	**98.1 (0.5)**	**98.1 (0.5)**	97.0 (0.7)	**98.3 (0.5)**
	Alt.ath	82.6 (1.9)	71.4 (2.6)	88.5 (1.7)	85.3 (1.5)	**91.8 (1.3)**	87.4 (1.8)
	BrCr	**93.8 (0.5)**	92.6 (0.5)	93.6 (0.4)	**94.1 (0.4)**	93.3 (0.4)	**94.3 (0.3)**
	Web	74.5 (3.1)	**80.7 (2.8)**	**77.0 (3.2)**	68.3 (4.4)	65.9 (3.9)	**76.8 (2.8)**
LibSVM	Musk1	**93.4 (1.2)**	**93.3 (1.2)**	**93.4 (1.2)**	**92.3 (1.5)**	**91.7 (1.5)**	**92.9 (1.3)**
	Musk2	**95.4 (1.4)**	93.4 (1.4)	**95.4 (1.4)**	**96.4 (0.9)**	92.2 (1.5)	**95.3 (1.2)**
	African	95.4 (0.5)	90.3 (0.8)	**96.7 (0.4)**	93.4 (0.7)	95.7 (0.4)	96.1 (0.5)
	Ajax	98.0 (0.7)	**98.9 (0.2)**	98.6 (0.4)	98.7 (0.4)	98.0 (0.6)	**98.9 (0.3)**
	Alt.ath	**94.5 (1.0)**	70.7 (2.5)	**94.9 (1.0)**	**94.7 (0.9)**	90.7 (1.4)	89.9 (1.6)
	BrCr	95.1 (0.4)	94.1 (0.4)	**95.5 (0.3)**	**95.6 (0.3)**	92.9 (0.4)	94.9 (0.4)
	Web	**75.0 (3.1)**	**77.8 (2.8)**	**76.0 (2.7)**	**75.2 (3.4)**	63.7 (4.0)	**78.2 (3.1)**

In AjaxOrange, it is an important observation that D^{from} is more informative than D^{to}, especially for the LDC classifier (see Fig. 3.9 (a) and (b)).. The dissimilarities are measured from the prototypes to the bags. The $d_{meanmin}$ dissimilarity in (3.3) therefore ensures that, for a positive prototype, the positive instances (the AjaxOrange bottle) influence the dissimilarity value by definition, as all instances of the prototype have to be matched to instances in the training bag. Measuring the dissimilarity to positive prototypes, on the other hand, may result in very similar values for positive and negative bags because of identical backgrounds, therefore creating class overlap.

Because D^{to} contains potentially harmful information, the combining methods do not succeed in combining this information from D^{to} and D^{from} in a way that is beneficial to the classifier. This is particularly evident for the LDC classifier (see Fig. 3.9 (a) and (b)), where only D^{ext} has similar (but still worse) performance than D_2. For the SVM classifier, EADS performs well only when a few prototypes are used, but as more prototypes (and more harmful information from D^{to}) are involved, there is almost no advantage over D_2 alone.

From the results reported in Fig. 3.10 for Winter Wren, we again see that D^{from} is more informative than D^{to}. However, what is different in this situation is that both directions contain useful information for classification, this is evident due to the success of the average, maximum and EADS combiners. The difference lies in the negative instances (fragments of other birds species, or background objects in the images) of positive bags. While in AjaxOrange, background objects are non-informative, the background in the



(a) 5 prototypes per class

(b) 20 prototypes per class

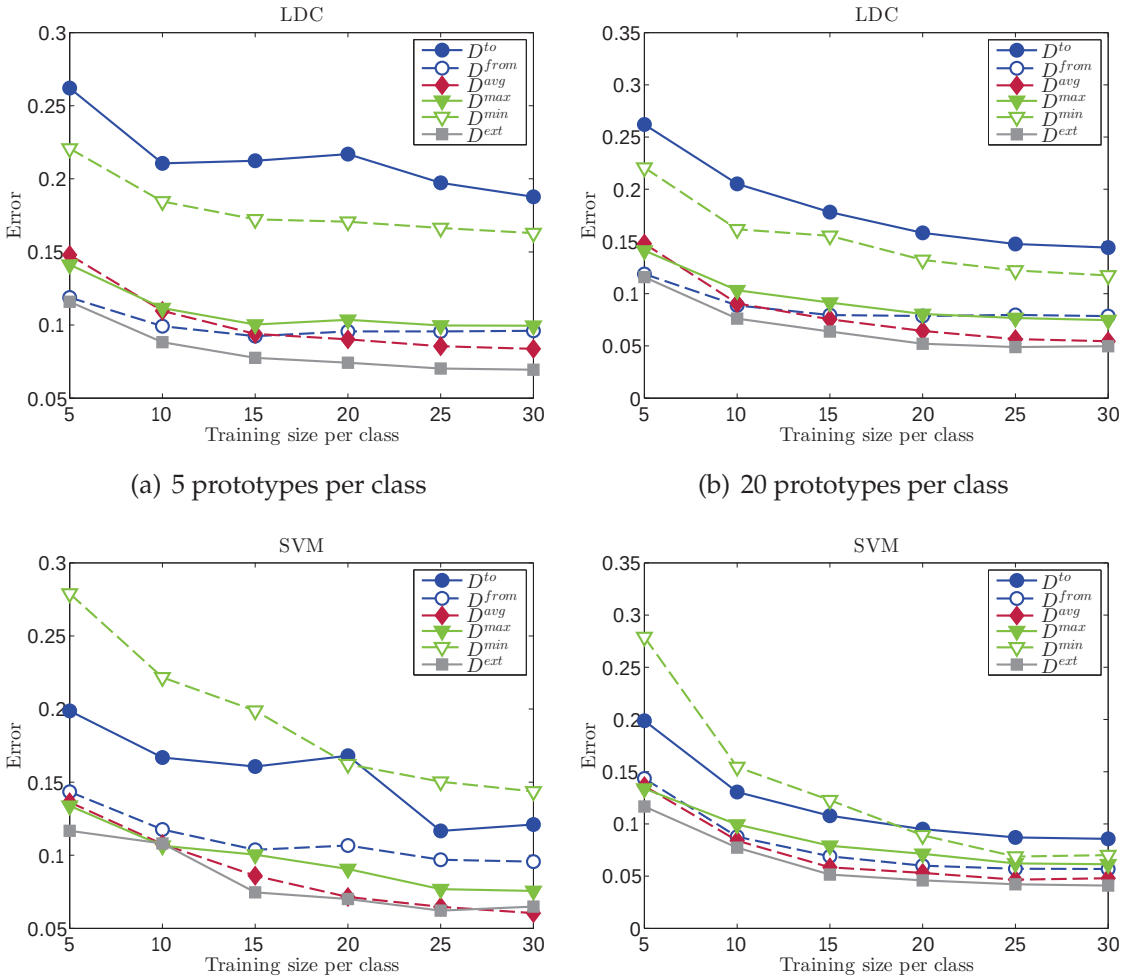
(c) 5 prototypes per class

(d) 20 prototypes per class

Figure 3.9: LDC and SVM classification results in dissimilarity spaces for the AjaxOrange dataset. The x-axis shows training set size per class, the y-axis shows the error averaged over 20 experiments.

audio fragments may be informative for the class of the sound. In particular, it is possible that some bird species are heard together more often: e.g. there is a correlation of 0.63 between the labels of Winter Wren and Pacific-slope Flycatcher. Therefore, also measuring dissimilarities to the prototypes produces dissimilarity values that are different for positive and negative bags.

The increased dimensionality of D^{ext} is one of the main problems of this approach, as in small sample size cases the increased dimensionality may lead to overfitting. In order to overcome this, prototype selection can be considered. We therefore demonstrate experiments using prototype selection for all the spaces compared, for the Winter Wren dataset. We use a fixed training set size of 200 objects, leading to spaces of dimensional-



(a) 5 prototypes per class

(b) 20 prototypes per class

(c) 5 prototypes per class

(d) 20 prototypes per class

Figure 3.10: LDC and SVM classification results in dissimilarity spaces for the Winter Wren dataset. The x-axis shows training set size per class, the y-axis shows the error averaged over 20 experiments.

ity 5, 10, 15, 20 and 25. Prototype selection is performed by forward selection optimizing the leave-one-out 1-NN classification error in the training set. Prototypes are selected for D^{ext} as it is usually done for a regular dissimilarity space. Prototypes using the two directed dissimilarities are available as candidates but the prototype selection method may discard one of the two or maybe both if they are not discriminative according to the selection criterion. D^{ext} is compared now with the other spaces on the basis of the same dimensionality. The results show that D^{avg} and D^{ext} are the best dissimilarity spaces for Winter Wren when prototype selection is performed.

One interesting issue of using prototype selection for D^{ext} is that not only the dimensionality is decreased, but also the accuracy of classifiers on D^{ext} may be improved,

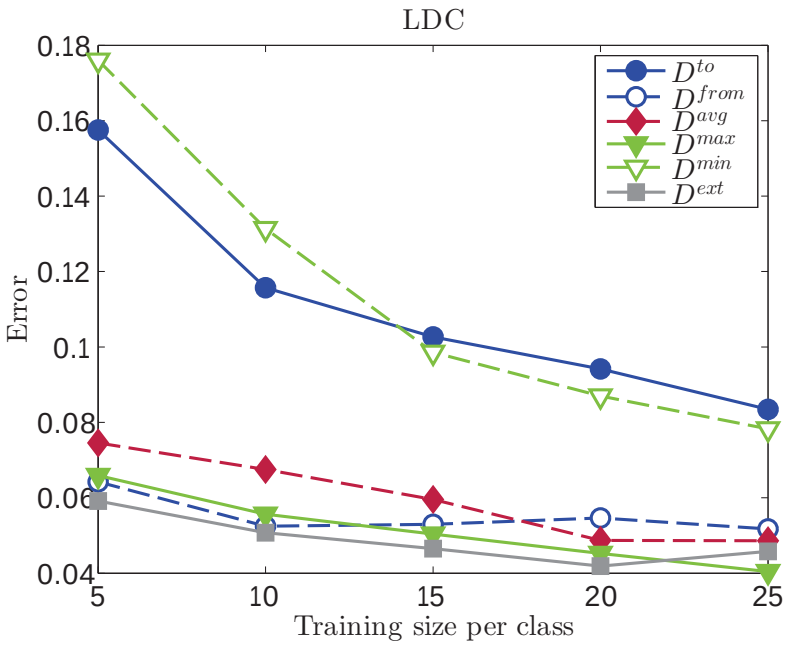


Figure 3.11: Classification results after prototype selection for Winter Wren dataset.

especially when one of the directed dissimilarities is very informative, while the other one is not at all. D^{ext} without prototype selection in these cases may be worse than the best directed dissimilarity (see Fig. 3.9 (a) and (b)). However, by using a suitable prototype selection method for D^{ext}, only the prototypes from the best directed dissimilarity should be kept, and noisy prototypes from the bad directed dissimilarity should be discarded. This should make the results of D^{ext} similar to those of the best directed distance.

4

Dissimilarity-based Ensembles for Multiple Instance Learning

In multiple instance learning, objects are sets (bags) of feature vectors (instances) rather than individual feature vectors. In this paper we address the problem of how these bags can best be represented. Two standard approaches are to use (dis)similarities between bags and prototype bags, or between bags and prototype instances. The first approach results in a relatively low-dimensional representation determined by the number of training bags, while the second approach results in a relatively high-dimensional representation, determined by the total number of instances in the training set. However, an advantage of the latter representation is that the informativeness of the prototype instances can be inferred. In this paper a third, intermediate approach is proposed, which links the two approaches and combines their strengths. Our classifier is inspired by a random subspace ensemble, and considers subspaces of the dissimilarity space, defined by subsets of instances, as prototypes. We provide insight into the structure of some popular multiple instance problems and show state-of-the-art performances on these datasets.

This chapter is accepted for publication as:
Veronika Cheplygina, David M. J. Tax, Marco Loog. Dissimilarity-based Ensembles for Multiple Instance Learning. *IEEE Transactions on Neural Networks and Learning Systems*

4.1 Introduction

Nowadays, many applications face the problem of using weakly labeled data for training a classifier. For example, in image classification, we may only have an overall label for an image (such as a "tiger"), not where the tiger is actually located in the image. Such problems are often formulated as multiple instance learning (MIL) [49] problems. MIL is an extension of supervised learning, and occurs in cases when class labels are associated with sets (*bags*) of feature vectors (*instances*) rather than with individual feature vectors. The bag labels provide weak information about the instance labels. For example, the label "tiger" could apply to only some of the image patches, because patches of sand, sky or other surroundings could be present as well. This is a natural representation for many real-world problems, therefore MIL has been successfully used to address molecule [49] or drug [66] activity prediction, image classification [4, 36], document categorization [200], computer aided diagnosis [69] and many other problems.

We can group methods that learn in this weakly supervised setting in two categories. The first category, which we call instance-based methods, relies on assumptions [65] about the labels to recover the instance labels, and classifies a bag by first classifying that bag's instances, and then fusing these outputs into a bag label. For example, the standard assumption is that a bag is positive if and only if at least one of its instances is positive or inside a concept [49]. The second category, which we call bag-based methods, often use the collective assumption: they assume all instances contribute to the bag label, and that bags with the same label are somehow similar to each other. Therefore, bags can be classified directly using distances [186] or kernels [71], or by converting bags to a single-instance representation and using supervised learners [36, 67, 171]. The bag-based methods have frequently demonstrated the best performances on a wide range of datasets.

One of the ways to represent structured objects, such as bags, in a feature space, is to describe them relative a set of reference objects or prototypes. This approach is called the dissimilarity representation [133] and is in contrast to traditional pattern recognition because the dimensions of this space are defined in a relative way: the dissimilarity to the j-th prototype can therefore be seen as the j-th feature in the transformed space. A successful approach we studied, uses *training bags* as prototypes [39, 171], while an alternative approach called MILES [36] uses all the *training instances* as prototypes. Both alternatives have demonstrated the best performances on a wide range of MIL problems, and, as we show in this paper, are in fact strongly related. However, both approaches are extremes with respect to the dimensionality and the information content of the resulting dissimilarity space. The bag representation reduces the dimensionality from the number of instances to the number of bags, but risks losing information contained in the individual instances. The instance representation preserves more information, but increases the dimensionality dramatically, possibly including many redundant features.

We propose a third alternative, which combines the advantages of using bags and instances as prototypes. We train classifiers on different subspaces of the instance dissimilarity space, where each subspace is formed by the instances of a particular bag, and combining the decisions of these classifiers in the test phase. This way, the information provided by information provided by different dissimilarities is preserved, but the dimensionality of each subspace is lower. Therefore the ensemble has the potential to be more robust than a single classifier, but still has the ability to provide interpretation for which instances are important in classification. Note that the bag, instance and subspace representations are analogous to different ways – averaging, concatenating and ensembles – of combining information from different sources [59, 94, 130]. The informativeness of these sources is central to this paper, and we focus on investigating which prototypes – bags, instances, or subspaces – are more informative in MIL problems.

We first introduced dissimilarity-based subspace ensembles in [41]. We now present an extended analysis of our approach, as well as significantly improved results on many MIL datasets. With respect to [41], this paper has several differences. In Section 4.2, we explain the preliminary tools used in our method, and relate our method to other dissimilarity-based approaches in MIL. In Section 4.3, we more formally define the proposed approach and explain the parameter choices that need to be made, that were treated as defaults in [41]. In Section 4.4, we provide understanding of the relationship between these parameters and the ensemble performance, which also explains our earlier results. We then demonstrate the effectiveness of the proposed approach with competitive results on several benchmark datasets. In addition to these results, we provide insight into the structure of some popular MIL problems, and why the dissimilarity space is a successful approach for such problems in general.

4.2 Dissimilarity-based Multiple Instance Learning

4.2.1 Data Representation

In multiple instance learning, an object is represented by a bag $B_i = \{\mathbf{x}_{ik} | k = 1, ..., n_i\} \subset \mathbb{R}^d$ of n_i feature vectors or instances. The training set $\mathcal{T} = \{(B_i, y_i) | i = 1, ...N\}$ consists of positive ($y_i = +1$) and negative ($y_i = -1$) bags, although multi-class extensions are also possible [202]. The standard assumption for MIL is that there are instance labels y_{ik} which relate to the bag labels as follows: a bag is positive if and only if it contains at least one positive, or *concept* instance: $y_i = \max_k y_{ik}$. In this case, it might be worthwhile to search for only these informative instances. Alternative formulations, where a fraction or even all instances are considered informative, have also been proposed [65].

We can represent an object, and therefore also a MIL bag B_i, by its dissimilarities to pro-

Datasets available online at http://www.miproblems.org

totype objects in a representation set $\mathcal{R}$ [133]. Often $\mathcal{R}$ is taken to be a subset of size M of the training set $\mathcal{T}$ of size N ($M \leq N$). If we apply this to MIL, each bag is represented as $\mathbf{d}(B_i, \mathcal{T}) = [d(B_i, B_1), ...d(B_i, B_M)]$: a vector of M dissimilarities. Therefore, each bag is represented by a single feature vector $\mathbf{d}$ and the MIL problem can be viewed as a standard supervised learning problem.

The bag dissimilarity $d(B_i, B_j)$ is defined as a function of the pairwise instance dissimilarities $[d(\mathbf{x}_{ik}, \mathbf{x}_{jl})]_{n_i \times n_j}$. There are many alternative definitions (see [40, 171]) but in this work we focus on the average minimum instance distance, which tends to perform well in practice. Suppose that we are only given one prototype B_j. With the proposed bag dissimilarity, the bag representation of B_i using prototype B_j is:

$$d^{bag}(B_i, B_j) = \frac{1}{n_i} \sum_{k=1}^{n_i} \min_l d(\mathbf{x}_{ik}, \mathbf{x}_{jl}) \tag{4.1}$$

Note that the dissimilarity between bag B_i and B_j is now reduced to a scalar, and $\mathbf{d}(B_i, \mathcal{T})$ becomes an M-dimensional vector.

A related method, MILES [36], considers a different definition of prototypes, by using the training instances rather than the training bags. The motivation is that, when we assume just a few concept instances per bag, it is better to consider just these informative instances rather than the bag as a whole. MILES is originally a similarity-based approach, where the similarity is defined as $s(B_i, \mathbf{x}) = \max_k \exp\left(-\frac{d(\mathbf{x}_{ik}, \mathbf{x})}{\sigma^2}\right)$ and σ is the parameter of the radial basis function kernel. However, by leaving out kernel and the need to choose σ, we get a dissimilarity-based counterpart. The instance representation of B_i using the instances of B_j is then defined as:

$$\mathbf{d}^{inst}(B_i, B_j) = [\min_l d(\mathbf{x}_{i1}, \mathbf{x}_{jl}), \min_l d(\mathbf{x}_{i2}, \mathbf{x}_{jl}), \cdots, \min_l d(\mathbf{x}_{in_i}, \mathbf{x}_{jl})] \tag{4.2}$$

Now the dissimilarity between B_i and B_j is summarized in a n_i-dimensional vector, resulting in a representation $\mathbf{d}(B_i, \mathcal{T})$ that has a dimensionality of $\sum_{k=1}^{M} n_k$.

From this point onwards, we will discuss the dissimilarity matrices D^{bag} and D^{inst}, which look as follows:

$$D^{bag} = \left[d^{bag}(B_i, B_j)\right]_{N \times M} \tag{4.3}$$

$$D^{inst} = \left[\mathbf{d}^{inst}(B_i, B_j)\right]_{N \times \sum_{k=1}^{M} n_k} \tag{4.4}$$

where $i = 1, \ldots, N$ and $j = 1, \ldots, M$.

D^{bag} and D^{inst} are two extremes with respect to the amount of information that is preserved. In cases where only a few instances per bag are informative, D^{bag} could suffer from averaging out these dissimilarities. D^{inst} would preserve these dissimilarities, but it could be difficult for the classifier to select only these relevant dissimilarities due to the high dimensionality of the representation. As an example, consider an image categorization problem, where an image is a bag, and an image region or patch is an instance. If many images in the training set contain regions that include the sky, the dissimilarities to the sky instances in D^{inst} will provide heavily correlated information about the bags. Therefore, D^{inst} could contain many redundant (but not necessarily uninformative) dissimilarities.

On the other hand, when most instances in a bag are informative, we would expect D^{bag} to perform well. D^{inst} would still have access to all the informative dissimilarities, however, selecting a few relevant dissimilarities, as in [67], where a single instance per bag is selected, might not be the best strategy if most instances are, in fact, relevant for the classification problem. The problem of being unable to specify how many dissimilarities are informative, still holds in this case.

4.2.2 Classifier and Informative Prototypes

In this work we consider linear classifiers $(\mathbf{w}, w_0)$ such that $f(\mathbf{d}) = \mathbf{w}^\intercal \mathbf{d} + w_0$ and $\mathbf{w}$ is an M-dimensional vector. The entries of $\mathbf{w}$ correspond to the weights assigned to each of the prototypes, either bags or instances. These weights are found by minimizing an objective function of the form:

$$\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \mathcal{T})) + \lambda \Omega(\mathbf{w}) \tag{4.5}$$

where $\mathcal{L}$ is the loss function evaluated on the training set, such as the logistic (for a logistic classifier) or hinge loss (for a support vector classifier or SVM). Ω is a regularization function of the weight vector and is often the l_2 or the l_1 norm. The l_2 norm typically results in most coefficients of $\mathbf{w}$ being non-zero, while the l_1 norm promotes sparsity, i.e., only some of the coefficients have non-zero values. λ is a parameter that trades off the loss with the constraints on the weight vector, and therefore influences the final values in $\mathbf{w}$.

A larger coefficient in $\mathbf{w}$ means the dissimilarity was found to be discriminative by the classifier, we therefore can examine the coefficients to discover which prototypes are more informative. However, the low sample size and high dimensionality/redundancy

of feature space can make it difficult to find the $\mathbf{w}$ that leads to the best performance on a held-out test set.

There are several alternatives for dealing with the redundancy of the prototypes, which are similar to the filter, wrapper and embedded methods in feature selection, which we describe in this section.

In the filter approach, (subsets of) prototypes are evaluated prior to training a classifier, therefore reducing the dimensionality of the training set. Such methods include clustering the prototypes and then selecting the cluster centers as prototypes. In [62] and [1], *clustering of instances* is performed. In [62], the authors first rely on the standard assumption to cluster and prune the instances of negative bags, creating negative prototypes. The positive prototypes are created by, for each bag, selecting the instance that is furthest from the negative prototypes. In [1], the prototypes are initialized as the k-means cluster centers of the instances. The prototypes are then updated by letting the prototypes take any value in the instance space $\mathbb{R}^d$. However, note that in both cases, clustering instances has the risk of excluding informative instances in sparsely populated parts of the feature space, because these would not have any neighbors to form a cluster with. On the other hand, [194] uses *clustering of bags*, by performing k-centers with a bag dissimilarity measure, such as (4.1), and selecting cluster centers as prototypes. Note that the dissimilarity in (4.1) is non-metric in which case clustering may lead to unpredictable results [84]. A more general disadvantage of filter approaches is that the informativeness of the filtered prototypes is not available.

In a wrapper approach, the classifier is used to determine which prototypes are more informative. The less informative prototypes are then removed, and the classifier is trained again on a smaller subset of prototypes. This procedure is repeated several times. This approach is used in MILIS [67] and is in fact an application of the popular SVM with recursive feature elimination (SVM-RFE) [74] on the D^{inst} representation. Again, a disadvantage is that in the final model, the informativeness of the removed prototypes is not available.

An example of an embedded approach is a sparse classifier that performs feature selection and classification simultaneously, such as the l_1-norm SVM [203] or Liknon classifier [12], used in MILES. This way, each prototype is associated with a weight, representing its informativeness. However, such sparse classifiers require cross-validation to set λ, which is ill advised in case the training set is small already. A common consequence of such problems is that a poor set of features might be chosen for the testing phase.

We would like to stress that it is not our purpose to focus on *selection* of prototypes, whether instances or bags. Next to the methods described above, there are many works on prototype selection in dissimilarity spaces [30, 134], that could be applied on either D^{bag} or D^{inst} representations. However, from the perspective of MIL, we are instead interested in the more general question of which prototypes – bags, instances or subspaces – are more informative, and which particular (for example positive or negative)

prototypes this might be.

4.3 Proposed Approach

4.3.1 Random Subspace Ensembles

We can see the bag and instance representations as two alternative ways of combining dissimilarities of different instances: by averaging or by concatenating. If we view these approaches as ways of combining different sources of information, a third alternative, ensembles, springs to mind.

The random subspace method (RSM) [79] is one way to create an ensemble that is particularly geared at small sample size, high-dimensional data. Each classifier is built on a lower-dimensional subspace of the original, high-dimensional feature space. This strategy addresses both aspects of a successful ensemble: accurate and diverse classifiers [22, 102]. Subsampling the feature space reduces the dimensionality for the individual base classifiers, therefore allowing for more accurate classifiers. Resampling of features introduces diversity [102], i.e. decorrelates the classifier decisions, which improves the performance of the overall ensemble.

More formally, the RSM ensemble consists of the following components:

- Number of subspaces L to be sampled
- Numbers of features $\{s_1 \dots s_L\}$ (or just s if $s_i = s_j \forall i, j$) to be selected for each subspace.
- Base classifier f, which is applied to each subspace. We denote the trained classifiers by $\{f_1, \dots, f_L\}$.
- Combining function g, which for a test feature vector $\mathbf{d}$, combines the outputs into a final classifier $F(\mathbf{d}) = g(f_1(\mathbf{d}), \dots f_L(\mathbf{d}))$.

RSM is interesting in high-dimensional problems with high feature redundancy [160]. For example, the expression levels of co-regulated (both influenced by another process) genes will provide correlated information about whether a subject has diabetes or not. Other genes may be irrelevant to diabetes, only adding noise. We typically do not have prior knowledge about the number of underlying processes that are responsible to diabetes, i.e., the amount of redundancy is unknown. This increases the number of possible relevant feature sets, and makes selecting only the relevant features more difficult. RSM decreases this risk, simplifying the feature selection problem for each individual classifier, and by still allowing access to all the (possibly relevant) features, thus letting the classifiers correct each other. Other examples where RSM is a successful approach in-

clude functional magnetic resonance imaging (fMRI) data [101], microarray data [11] and hyperspectral data [75].

The different prototypes in MIL may also provide redundant information, but we do not know in advance how many such redundant prototypes there might be. Furthermore, many MIL problems are small sample size problems in the number of bags, so additional classifier evaluations during training are undesirable. Therefore we believe that RSM can be an attractive method to address dissimilarity-based MIL, and to combine the strengths of the dissimilarity space with those of ensemble classifiers.

4.3.2 Choice of Subspaces

There are two alternatives for how the subspace classifiers can be defined:

- By choosing each prototype bag as a subspace, i.e. the subspace is formed by the dissimilarities to the instances of a prototype bag. This representation immediately follows from our intuition about bags and instances being analogous to averaging and concatenating different information sources. We denote this representation by D^{BS}, where BS stands for Bag Subspaces. The RSM parameters are straightforward here: $L = M$ and the subspace dimensionalities s_i correspond to the bag sizes n_i.
- By choosing each subspace randomly. We denote this representation by D^{RS}, where RS stands for Random Subspaces. D^{RS} offers more flexibility with regard to the RSM parameters. In [41], we used default parameters $L = M$ and $s = \frac{1}{N}\sum_i n_i$. However, alternative settings are possible as well, and we will demonstrate further on in this paper that other choices (which can be set by rules of thumb rather than cross-validation), can in fact improve the results significantly.

Equations 4.6 and 4.7 show the choices in matrix format:

$$D^{BS} = \{\left[d^{inst}(B_*, B_1)\right]_{N\times n_1}, \ldots, \left[d^{bag}(B_*, B_M)\right]_{N\times n_M}\} \tag{4.6}$$

$$D^{RS} = \{\left[d^{inst}(B_*, R_1)\right]_{N\times s}, \ldots, \left[d^{bag}(B_*, R_M)\right]_{N\times s}\} \tag{4.7}$$

where R_i are prototype bags, generated randomly from all available instances in the training set.

Note that these alternatives are both slightly different from RSM because the dissimilarity representation depends on the training set. In traditional RSM, all features are available to the classifier at any split of the training and test data, whereas with D^{BS} and D^{RS}, the features are defined through dissimilarities to the training set, which obviously changes with every training-test split. However, we still expect there to be a

Representation	Dimensionality	Classifiers
D^{bag}	M	1
D^{inst}	$\sum_i n_i$	1
D^{BS}	$\{n_1, \dots n_M\}$	M
D^{RS}	any	any

Table 4.1: Different ways for constructing dissimilarity representations. D^{bag} consists of dissimilarities to bags in the training set (one for each bag), whereas D^{inst} consists of dissimilarities to instances in the training set. In D^{BS}, a separate classifier is built on each prototype's instance dissimilarities. In D^{RS}, classifiers are built on random selections of all available instances.

relationship between how RSM parameters, and the choices in D^{BS} and D^{RS}, affect ensemble performance.

We provide a summary of the ensembles, as well as the single classifier dissimilarity representations in Table 4.1.

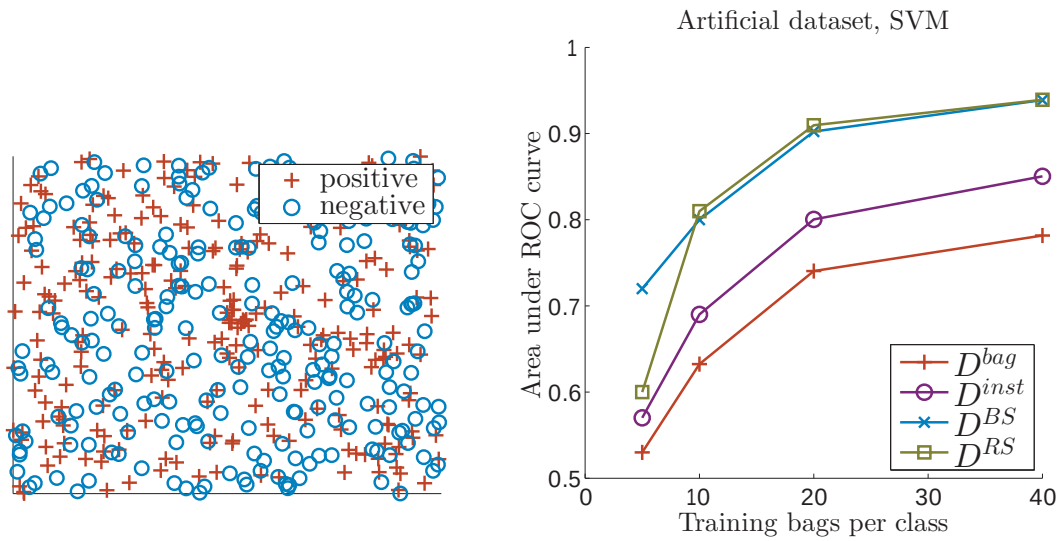
4.3.3 Illustrative Example

The basic intuition about the benefit of the proposed ensembles is illustrated by the artificial problem in the top of Fig. 4.1. This is the classical MIL problem from [114]. This dataset contains bags with 50 two-dimensional instances. The instances from the bags are uniformly distributed in a square, and the positive bags contain at least one feature vector from a concept region that is located in the middle of the square. Only the dissimilarities of the concept instances are informative. Averaging over the dissimilarities as in D^{bag} dilutes these informative features, and indeed, the learning curves in the bottom of Fig. 4.1 show that D^{bag} performs poorly here. D^{inst} has trouble selecting only the informative dissimilarities because many dissimilarities are uninformative, and because dissimilarities of the informative instances are correlated. The ensemble methods are more robust against these problems and achieve the best performances (Fig. 4.1, bottom).

4.4 Experiments

4.4.1 Data and Setup

We provide a list of datasets we used in Table 4.2. The Musk problems [49] are traditional benchmark MIL problems about molecule activity, Mutagenesis [166] is a drug activity prediction problem. Fox, Tiger and Elephant [4] are benchmark image datasets.



(a) (b)

Figure 4.1: Top: Artificial 2D MIL problem with informative instances in the center. Bottom: Learning curves for dissimilarity-based classifiers on this dataset. The amount of uninformative, and redundant instances deteriorates performances of D^{bag} and D^{inst}, but the ensemble methods D^{BS} and D^{RS} are more robust against these problems.

African and Beach are also image datasets originating from a multi-class scene recognition problem [36], but here formulated as one-against-all problems. The dataset alt.atheism originates from the Newsgroups data [200], and are concerned with text categorization. Brown Creeper and Winter Wren are both bird song [19] datasets, where the goal is to classify whether a particular bird species can be heard in an audio recording.

We preprocess the data by scaling each feature to zero mean and unit variance. The scaled data is used to compute the dissimilarity representations. The instance dissimilarity function is defined as the squared Euclidean distance: $d(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\intercal(\mathbf{x}_i - \mathbf{x}_j)$.

For the base classifier f, we consider linear classifiers as described in Section 4.2. We used several linear classifiers: logistic, 1-norm SVM and a linear SVM, where the primal formulation is optimized [34]. The trade-off parameter λ is set to 1 by default. The common characteristic of these classifiers is that we can inspect the weight vector $\mathbf{w}$ to determine which dissimilarities are deemed to be more important by the classifier. Although the individual performances of the classifiers differ, we observe similar trends (such as relative performances of two different ensembles) for these choices. We therefore only show results for the linear SVM.

For the combining function g, we average the posterior probabilities, which are obtained

after normalizing the outputs of each classifier to the [0, 1] range. We also considered majority voting, and using the product and the maximum of the posterior probabilities as combining schemes. With the product and maximum rules, the performances were lower, especially deteriorating towards larger ensemble sizes (thus being more sensitive to outlier, inaccurate base classifiers). With majority voting, the results were slightly lower than with averaging, but the standard deviations across different folds were larger. Therefore we chose averaging as a more robust combining strategy. Furthermore, averaging posterior probabilities performs well in practice in other problems as well [38, 172]. It could be argued that the fixed g can be replaced by a more flexible function, such as a linear classifier, trained on the outputs of the base classifiers. However, as discussed in [55], this would ideally require splitting the training data into two parts: one for training the base classifiers, and one for training the combiner. This is undesirable because we are already faced with a low object-to-feature (i.e., bag-to-instance) ratio. Indeed, our earlier experiments [41] with a nearest mean classifier as a combiner did not lead to improved results.

The metric used for comparisons is area under the receiver-operating characteristic (AUC) [16]. This measure has been shown to be more discriminative than accuracy in classifier comparisons [82], and more suitable for MIL problems [170].

Dataset	+bags	-bags	total	average
Musk 1	47	45	476	5
Musk 2	39	63	6598	65
Fox	100	100	1302	7
Tiger	100	100	1220	6
Elephant	100	100	1391	7
Mutagenesis 1	125	63	10486	56
African	100	1900	7947	8
Beach	100	1900	7947	8
Alt.atheism	50	50	5443	54
Brown Creeper	197	351	10232	19
Winter Wren	109	439	10232	19

Table 4.2: MIL datasets, number of bags, instances, the average number of instances per bag. The datasets are available online at `http://www.miproblems.org`

4.4.2 Subspace Experiments

We start by comparing the two alternatives of creating the subspaces, D^{BS} and D^{RS}. For simplicity, we base the parameters of D^{RS} on those for D^{BS}, as in [41]: M subspaces, each subspace with dimensionality $\frac{1}{N}\sum_i n_i$. We use a linear SVM as the classifier (C parameter is set to 1 by default) and perform 10-fold cross-validation. Fig. 4.2 shows the

distributions of the individual classifier performances and the ensemble performance for both representations.

The results show that overall,

1. the random subspaces are more informative than bag subspaces, and
2. the random subspace ensemble is better at improving upon the base classifiers.

Why do the random subspaces perform better than the bag subspaces? One difference might be the subspace size: the bag subspaces have variable dimensionalities, whereas the random subspaces are equally large. For D^{BS}, we plot the bag size against the performance of that subspace. A few of the results are shown in Fig. 4.3. In all datasets except alt.atheism, we find medium to high correlations between these quantities. Therefore, as prototypes, small bags are not very informative. This might seem counterintuitive in a MIL problem, because small bags are less ambiguous with respect to the instance labels under the standard assumption. The fact that large, ambiguous bags are better at explaining the class differences suggests that most instances are informative as prototypes.

The informativeness of most instances as prototypes is supported by the relationship between the bag label and the subspace performance in the plots. Although for a fixed bag size, positive bag subspaces perform better on average, negative bags can also be very good prototypes. This is also true for random bags, for which we do not have labels, but for which we can examine the percentage of instances, which were sampled from positive bags. We found no correlations between the positiveness of the random subspaces and their performance. This provides opportunities for using unlabeled data in a semi-supervised way: unlabeled bags can be used to extend the dissimilarity representation to improve performance, similar to the approach in [51].

The results with respect to the bag size suggest that it is advantageous to combine classifiers built on higher-dimensional subspaces. We therefore investigate the effects of subspace dimensionality in D^{RS}, where this parameter can be varied. We vary the subspace size for each classifier from 5 to 100 features, which for most datasets, would be larger than the default dimensionalities used previously, as shown in Table 4.2. We generate 100 subspaces of each dimensionality, and train a linear SVM on each subspace. The classifiers are then evaluated individually. Some examples of performance distributions at different subspace dimensionalities are shown in Fig. 4.4. For most datasets (except Newsgroups, as will be explained later), larger subspaces lead to more accurate classifiers. However, increasing the dimensionality too much, eventually using all the features, decreases the performance. This can be more clearly seen in the results of Musk2, where the total number of instances (and thus dimensionality) larger than in Musk1. The results of other datasets show similar trends depending on the bag/instance ratio, for example, the results of Mutagenesis are quite similar to Musk2.

Why is D^{RS} better than D^{BS} at improving upon the individual classifiers? A possible

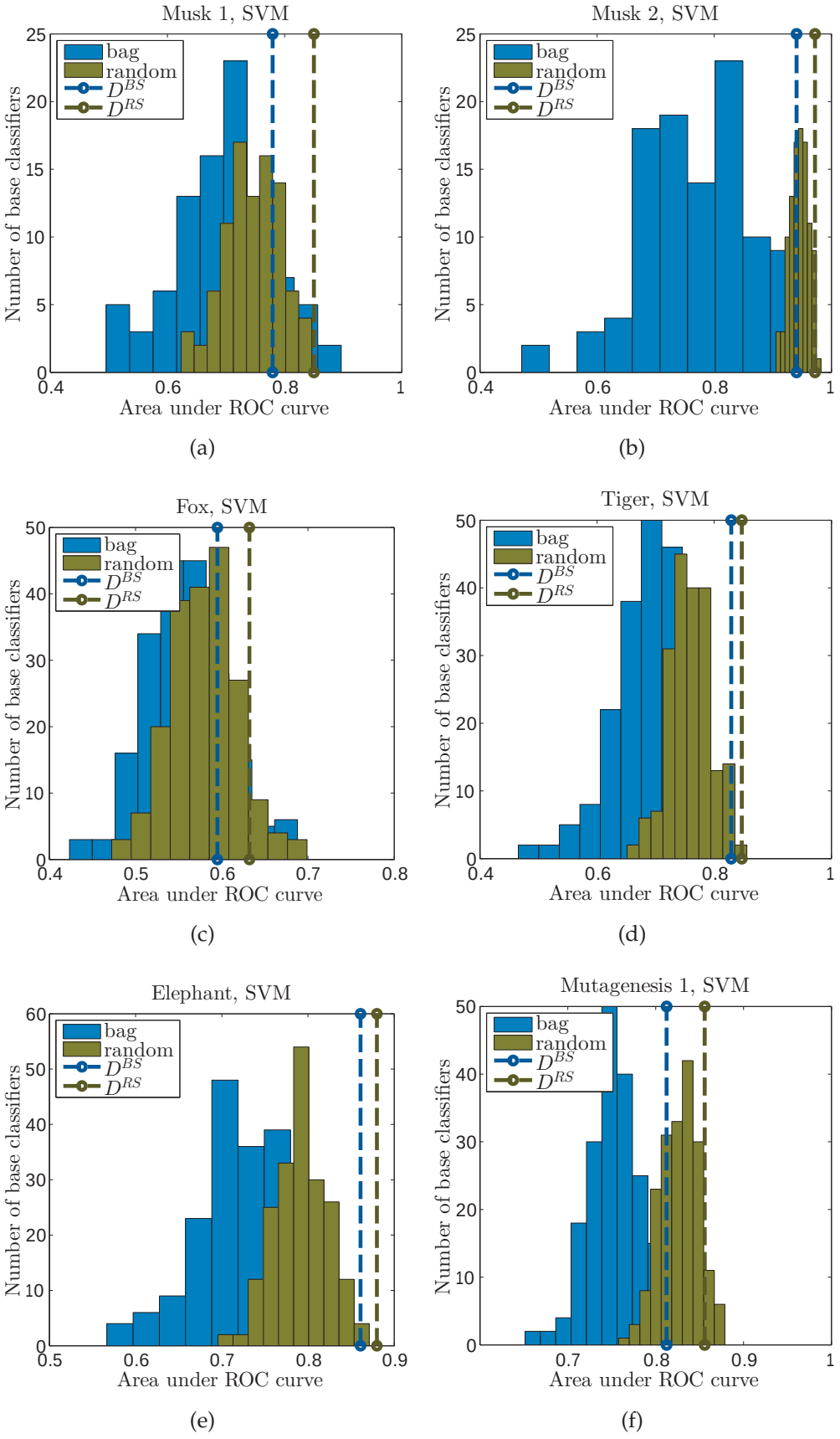
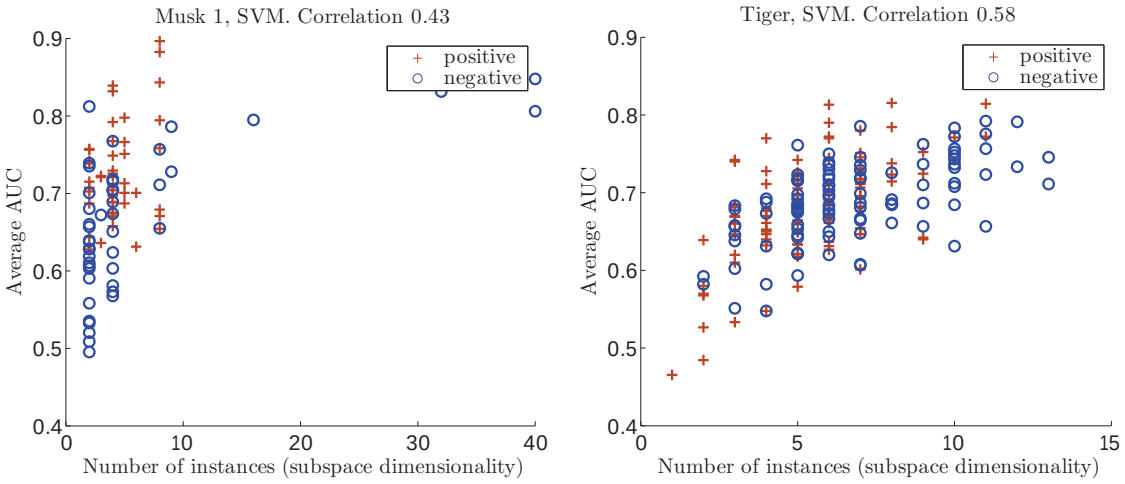
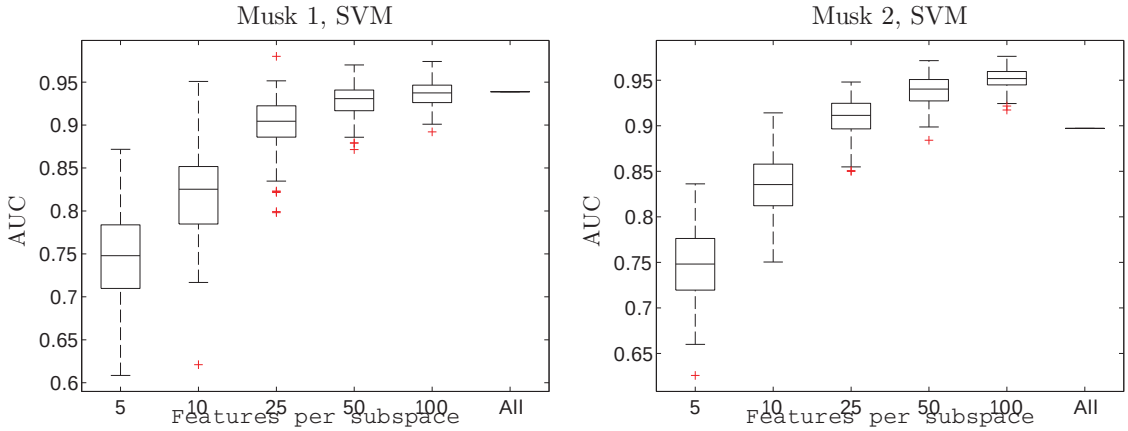


Figure 4.2: Distributions of AUC performances of individual bag subspace classifiers



(a) (b)

Figure 4.3: Relationship of bag size, bag label, and AUC performance of the dissimilarity subspace formed by that bag



(a) (b)

Figure 4.4: Distributions of AUC performances of individual subspace classifiers, for different dimensionalities of the subspaces. The degradation of performance using all features is larger in Musk2 because instance / bag (and thus feature / object) ratio is larger than in Musk1. Similar trends can be observed in other datasets.

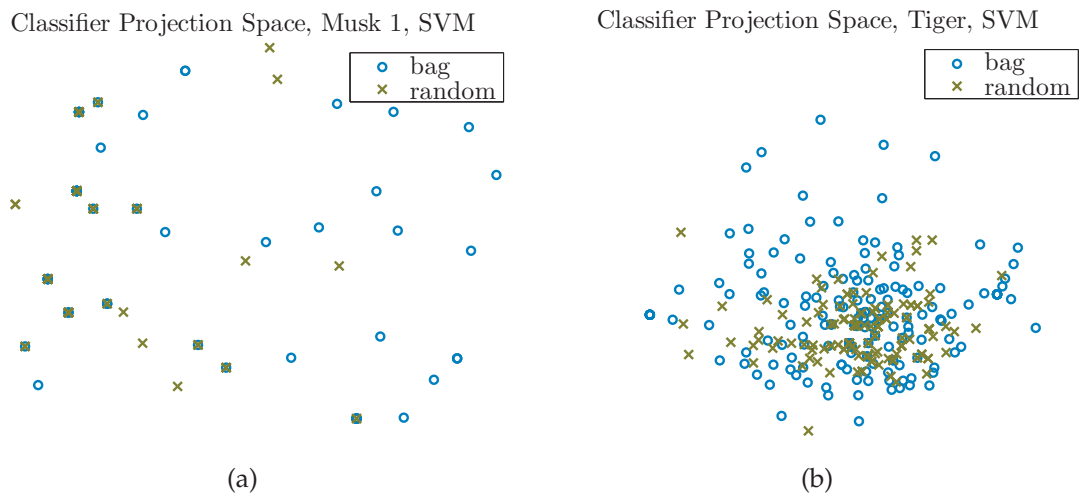
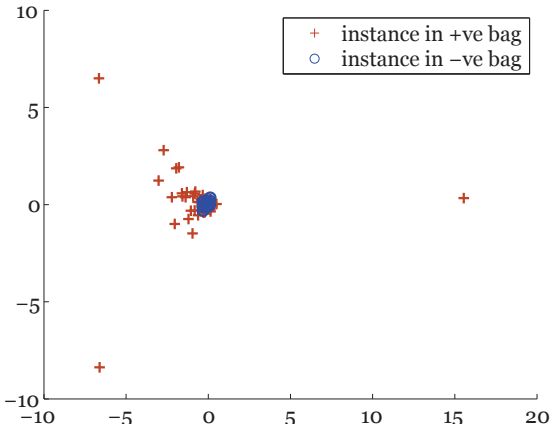


Figure 4.5: Classifier projection spaces. The plots show the relative disagreement of trained subspace classifiers on a test set. The higher the disagreement of two classifiers on the labels of the test set, the larger their distance in the plot.

explanation is that the classifiers created by D^{RS} are more diverse than the classifiers of D^{BS}. For each set of classifiers, we examine their $L \times L$ disagreement matrix C, where each entry $C_{i,j}$ corresponds to the number of test bags for which the i-th and j-th classifier provide different labels. $C_{i,j}$ can be seen as the distance of the two classifiers. We perform multi-dimensional scaling with each of these distance matrices and map the classifiers into a 2-dimensional classifier projection space [135].

The classifier projection spaces for two datasets are shown in Fig. 4.5. These results are surprising, because the classifiers in D^{BS} actually cover a larger part of the space, and are therefore more diverse. This diversity alone, however, is not able to improve the overall performance of the ensemble. A possible explanation is that here we are dealing with bad diversity [21]. For example, a classifier that is 100% wrong is very diverse with respect to a classifier that is 100% correct, but not beneficial when added to an ensemble. We showed in Fig. 4.3 that in D^{BS}, small bags often result in inaccurate classifiers, which are indeed responsible for higher diversity, but worsen the ensemble performance.

In the experiments, the Newsgroups data shows very different behavior from the other datasets. Most of the subspaces (both bag and random) have nearly random performance, and the performances are not correlated with the bag label or subspace dimensionality. A possible explanation is that in this dataset, many of the dissimilarities only contain noise, and the informativeness is distributed only across a few dissimilarities. RSM is particularly suitable for problems where the informativeness is spread out over many (redundant) features [160], which could explain the worse than random performance. Indeed, examining the individual informativeness (as measured by the nearest neighbor error) of each individual dissimilarity, it turns out that more than half of the



(a)

Figure 4.6: Multi-dimensional scaling of the instances in the alt.atheism dataset

dissimilarities in alt.atheism have worse than random performance, as opposed to only around 10% of dissimilarities for the other datasets.

We have noticed previously [39] that positive bags in the Newsgroups data consist of a dense cluster of instances and a few outliers, while negative bags consist only of the dense cluster. This distribution is caused by the bag of words representation of the data – while the outlier instances are in fact all positive for the alt.atheism topic, they do not consist of the same words, and are far from each other in the feature space. This situation is shown in Fig.4.6. The presence or absence of such outliers is very informative for the bag class. The definition of dissimilarity in (4.2), however, does not succeed in extracting this information: for any training bag, the instance closest to the prototype outlier instance, is in the dense cluster. In this case the minimum function in the dissimilarity is not suitable for the problem at hand, and much better performances can be obtained by, for instance, using bags and prototypes, and considering the asymmetry of D^{bag} [39].

4.4.3 Ensemble Experiments

With several interesting results for the individual performances of the subspace classifiers, we now investigate how the ensemble behaves when both the parameters of interest are varied. The ensembles are built in such a way that classifiers are only added (not replaced) as the ensemble size increases, i.e. an ensemble with 100 classifiers has the same classifiers as the ensemble with 50 classifiers, and 50 additional ones.

The results are shown in Fig. 4.7. Here different plots show the results for different subspace dimensionalities (5, 10, 25, 50 or 100 features), and the x-axis indicates the number of classifiers used in the ensemble. The black line shows the baseline perfor-

mance of D^{inst}. The performance metric is again the area under the ROC curve (AUC), and the results are averaged over 10-fold cross-validation.

Ensembles created with higher-dimensional subspaces tend to perform better. An unsurprising result is that the fixed dimensionality values in this experiment are not suitable for all datasets. For example, in Musk none of the ensembles outperform the single classifier. Clearly, the subspace dimensionality should depend on the dimensionality (and possibly redundancy of the dissimilarities) of the original problem.

Another interesting observation in Fig. 4.7 is that it is often sufficient to build the ensemble from a few base classifiers, and adding further classifiers probably would not improve performance significantly. This is in line with our earlier results for ensembles of one-class classifiers [38], even though both the data and the classifiers in question are very different. The recommendation in [101] is that when there is no prior knowledge about how redundant the features are, L should be chosen to be relatively small, while s should be relatively large.

Based on these observations, we settle on the following choices for D^{RS}: $L = 100$ and $s = \frac{1}{5}\sum_i^N n_i$. We emphasize that these choices are good rules of thumb and do not have to be set to these exact values, which is supported by our results from the previous section: the performance is quite robust to changes in L and s provided s is large enough.

The performances of the proposed ensemble against those of the single classifier representations D^{bag} and D^{inst} are shown in Table 4.3. Contrary to our earlier results in [41], D^{RS} is now a clear competitor for the single classifiers, and has especially surprising performances on the Musk 1, Fox and Mutagenesis datasets. The advantages of D^{RS} over the high-dimensional D^{inst} are more visible, however, there are no significant differences with D^{bag}. This suggests that many of the dissimilarities are, in fact, informative, and averaging the dissimilarities over each bag preserves sufficient information. Despite the similar performances of D^{bag} and D^{RS}, D^{RS} has an additional advantage: it is possible to recover the importance of different instances, as we will show in Section 4.4.5.

4.4.4 Comparison with Other MIL Classifiers

We compare our method to other popular MIL classifiers, which cover a range of instance-based and bag-based methods, and are often being compared to in recent papers.

EM-DD [195], mi-SVM [5], MILBoost [183] and MIForest [107] are instance-based methods. EM-DD is an expectation-maximization algorithm which uses diverse density (DD), which, for a given point t in the feature space space, measures the ratio between the number of positive bags which have instances near t, and the distance of the negative instances to t. The expectation step selects the most positive instance from each

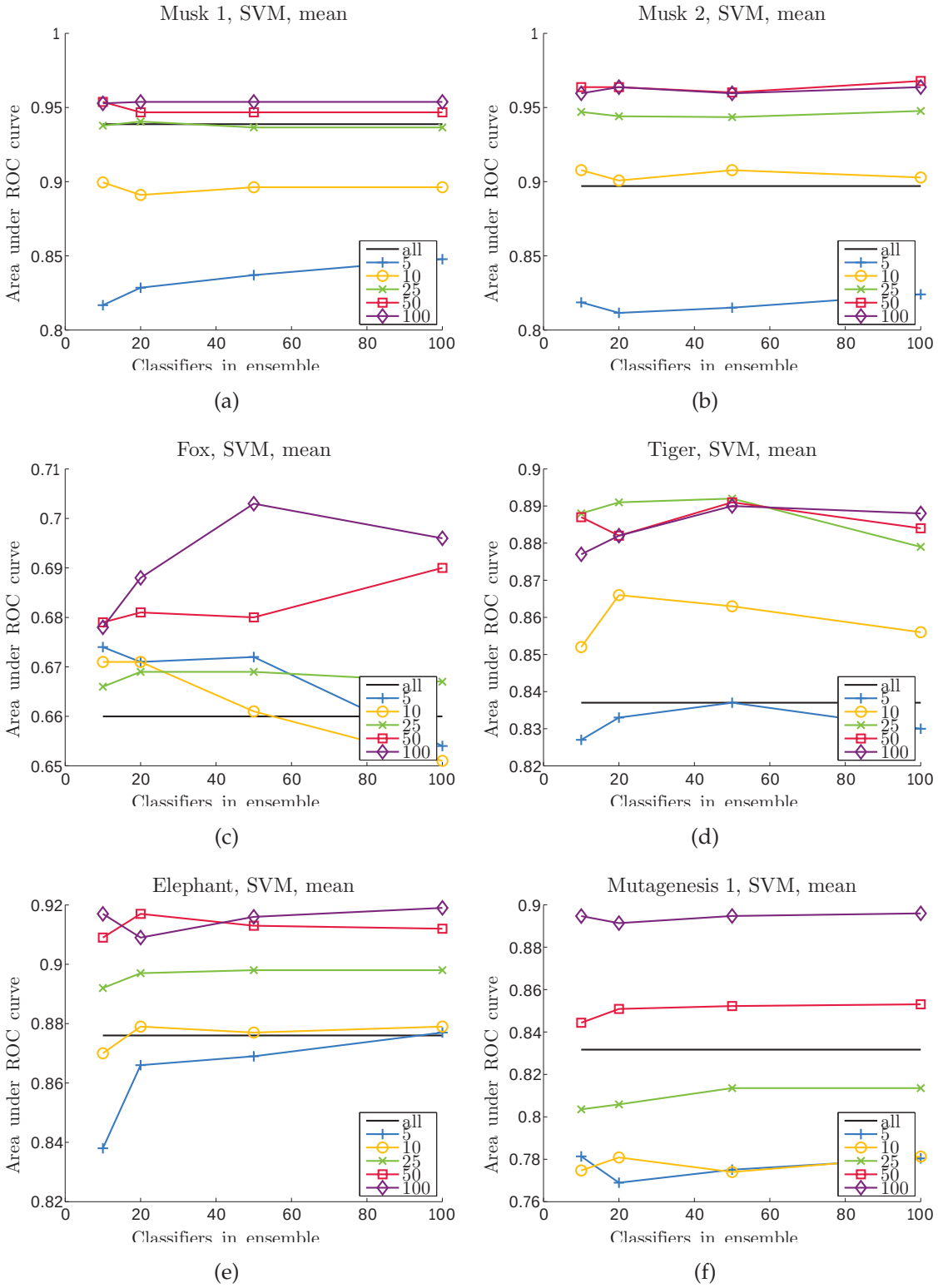


Figure 4.7: AUC performances of the instance representation (black line) and the ensemble classifiers. Different lines per plot indicate different dimensionalities of subspace classifiers.

	Representation		
Dataset	D^{bag}	D^{inst}	D^{RS}
Musk1	93.7 (3.5)	*93.9 (3.6)*	**95.4 (2.4)**
Musk2	**95.7 (1.4)**	89.7 (4.7)	*93.2 (3.2)*
Fox	*67.9 (2.5)*	66.0 (3.9)	**70.2 (1.8)**
Tiger	*86.8 (5.2)*	83.7 (3.9)	**87.8 (4.2)**
Elephant	*90.9 (2.5)*	87.6 (3.3)	**92.3 (2.7)**
Mutagenesis	*84.3 (2.9)*	83.2 (3.2)	**87.4 (3.5)**
Brown Creeper	**94.7 (1.0)**	93.9 (0.8)	*94.2 (0.8)*
Winter Wren	**99.5 (0.2)**	98.4 (0.5)	*99.0 (0.3)*
African	*92.5 (1.1)*	91.6 (1.4)	**92.8 (1.2)**
Beach	**88.3 (1.2)**	85.6 (1.2)	*87.9 (1.2)*
alt.atheism	**62.4 (8.3)**	46.4 (5.9)	*46.4 (5.2)*

Table 4.3: AUC ($\times$100) mean and standard error of 10-fold cross-validation of the single dissimilarity-based representations, and the proposed ensemble representation. **Bold** = best, *italics* = second best result per dataset

bag according to t, the maximization step then finds a new concept t' by maximizing DD on the selected, most positive instances. mi-SVM is an extension of support vector machines which attempts to find hidden labels of the instances under constraints posed by the bag labels. Likewise, MILBoost is an extension of boosting and MIForest is an extension of random forests, where the standard assumption is used to reweigh or relabel the instances in each iteration of the algorithm.

MILES [36] and the Minimax kernel are bag-based methods which convert bags to a single-instance representation. MILES is similar to the 1-norm SVM applied to the D^{inst}, except that in MILES, a Gaussian kernel is used for defining similarities, and an appropriate σ parameter is necessary. The Minimax kernel [71] is obtained by representing each bag by by the minimum and maximum feature values of its instances, this representation can then be used with a supervised classifier. All classifiers are implemented in PRTools [56] and the MIL toolbox [169] and default parameters are used unless stated otherwise.

Next to the standard MIL classifiers, we use D^{RS} with the guidelines from the previous section: the dimensionality of each subspace is 1/5th of the total dimensionality, and 100 classifiers are used in the ensemble. The base classifier is the linear SVM. For both MI-SVM and MILES, the radial basis kernel with width 10 was used.

The results are shown in Tables 4.4 and 4.5 . Some results could not be reported; in particular, for EM-DD when one cross-validation fold lasts longer than five days, and MILBoost when features with the same value for all instances are present, as in alt.atheism. Needless to say, there is no method that performs the best at all times. Some of the default parameter choices may be unsuitable for a particular dataset, or the assumptions

Table 4.4: AUC ×100, mean and standard error of 10-fold cross-validation of different MIL classifiers. **Bold** = best, *italics* = second best per dataset.

Dataset	Classifier EM-DD	MI-SVM	MILBoost	MILES	minimax +SVM	D^{RS} +SVM
Musk1	85.0 (5.1)	91.5 (3.7)	74.8 (6.7)	*93.2 (2.9)*	87.8 (5.0)	**95.4 (2.4)**
Musk2	88.1 (2.7)	*93.9 (2.8)*	76.4 (3.5)	**97.1 (1.6)**	91.3 (1.8)	93.2 (3.2)
Fox	67.6 (3.2)	*68.7 (2.6)*	61.3 (3.2)	66.8 (3.5)	55.8 (2.9)	**70.2 (1.8)**
Tiger	-	*87.2 (3.5)*	87.0 (3.0)	84.6 (4.5)	76.0 (4.1)	**87.8 (4.2)**
Elephant	88.5 (2.1)	*90.7 (2.3)*	88.8 (2.2)	88.4 (2.5)	88.4 (2.1)	**92.3 (2.7)**
Mutagenesis	67.4 (5.3)	60.3 (4.5)	**88.1 (3.1)**	72.1 (4.3)	63.7 (4.4)	*87.4 (3.5)*
Brown Creeper	94.5 (0.9)	92.8 (1.2)	*94.9 (0.9)*	**96.1 (0.6)**	94.2 (0.9)	94.2 (0.8)
Winter Wren	98.3 (0.5)	**99.2 (0.4)**	93.8 (5.6)	*99.1 (0.5)*	98.2 (0.3)	99.0 (0.3)
African	*91.2 (1.8)*	88.6 (1.7)	89.4 (1.7)	48.7 (2.3)	87.6 (1.4)	**92.8 (1.2)**
Beach	84.6 (2.0)	78.2 (2.5)	*85.2 (2.9)*	72.8 (5.0)	83.0 (2.3)	**87.9 (1.2)**
Alt.atheism	*52.0 (8.0)*	38.8 (5.2)	-	50.0 (5.5)	**80.0 (3.6)**	46.4 (5.2)

that the method is based on do not hold. However, overall the method we present is always able to provide competitive performance.

4.4.5 Instance Weights

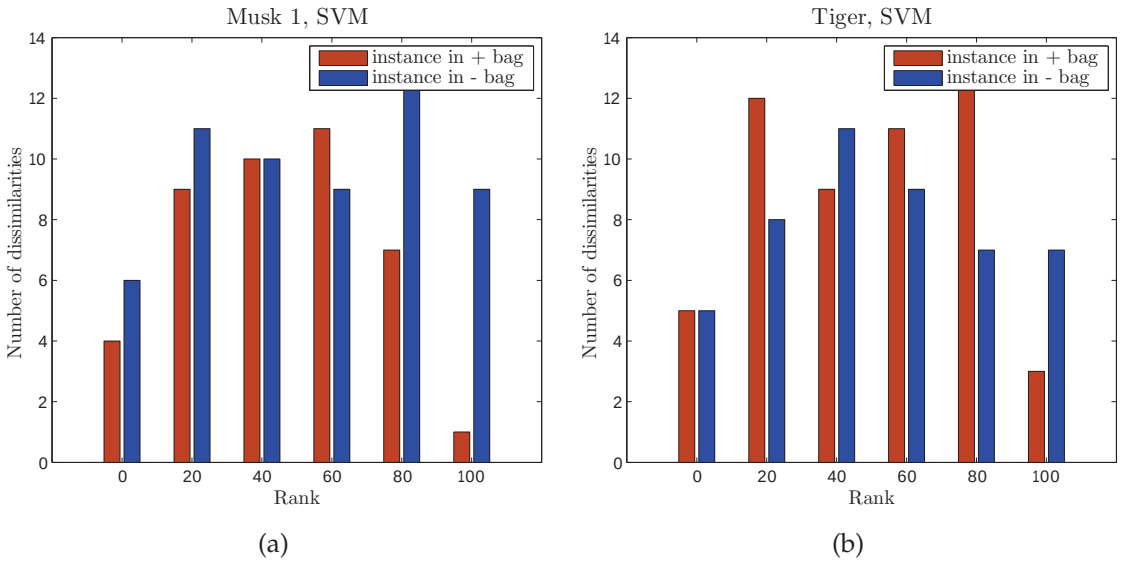
An advantage of linear classifiers is the interpretability of the result – from the weights $\mathbf{w}$ of the dissimilarities, we can derive which dissimilarities, and therefore which instances are more important (i.e., have a larger weight) to the classifier. This property can also be used in ensembles of linear classifiers, with a procedure described in [104]. For each dissimilarity, we calculate the average absolute value of its weight over all L subspaces in which the dissimilarity was selected. We then sort the dissimilarities by this average weight, and view the position of each dissimilarity in this list as its rank. The distributions of the dissimilarities with ranks 1 to 100 are shown in Fig.4.8 shows the distributions of top 100 dissimilarities. These most informative dissimilarities originate from both positive and negative bags, supporting the idea that not only concept, positive instances are important for these MIL problems.

4.5 Discussion

We proposed a dissimilarity-based ensemble as a novel classification method for MIL problems. When bags are represented by their dissimilarities to instances from the training set, such instances can provide redundant information about the problem. A ran-

Table 4.5: Accuracy $\times 100$, mean and standard error of 10-fold cross-validation of different MIL classifiers. **Bold** = best, *italics* = second best per dataset.

Dataset	Classifier EM-DD	MI-SVM	MILBoost	MILES	minimax +SVM	D^{RS} +SVM
Musk1	85.1 (4.1)	83.0 (4.6)	69.8 (5.6)	82.8 (4.7)	*86.3 (3.8)*	**89.3 (3.4)**
Musk2	81.5 (2.9)	76.3 (5.4)	66.0 (3.7)	**86.3 (3.4)**	82.3 (2.5)	*85.5 (4.7)*
Fox	62.0 (2.7)	*63.5 (2.2)*	63.0 (2.6)	62.5 (4.2)	58.0 (2.5)	**64.5 (2.2)**
Tiger	-	73.0 (2.9)	78.5 (2.8)	**81.0 (3.4)**	72.5 (3.9)	*81.0 (4.6)*
Elephant	*82.5 (1.5)*	76.0 (2.4)	79.5 (2.8)	79.0 (2.3)	82.5 (2.5)	**84.5 (2.8)**
Mutagenesis	61.1 (5.1)	66.5 (0.4)	*75.7 (2.7)*	71.1 (3.4)	68.9 (3.0)	**83.2 (3.4)**
Brown Cr	85.6 (1.5)	50.5 (1.5)	*88.7 (1.2)*	**89.8 (1.4)**	87.3 (1.6)	88.3 (1.0)
Winter Wr	94.5 (1.2)	81.0 (1.5)	89.0 (7.8)	**96.7 (0.8)**	93.8 (0.8)	*96.0 (0.8)*
African	82.7 (0.9)	94.7 (0.1)	*95.2 (0.2)*	95.0 (0.0)	95.0 (0.0)	**96.0 (0.5)**
Beach	73.4 (1.1)	95.0 (0.1)	**95.2 (0.2)**	*95.0 (0.0)*	94.9 (0.1)	94.8 (0.4)
Alt.atheism	49.0 (5.7)	48.0 (2.0)	-	*50.0 (4.5)*	**76.0 (4.0)**	44.0 (4.5)



(a) (b)

Figure 4.8: Top 100 ranked dissimilarities, where the rank is determined by average weight of dissimilarities across $L = 100$ subspace classifiers

dom subspace inspired ensemble, where classifiers are trained on different subspaces of the dissimilarity space, is a way of dealing with this redundancy. We show that our method achieves competitive performances with other MIL algorithms, and has intuitive parameters that do not need to be set by cross-validation to achieve these good results.

We investigated two choices for generating the subspaces: by using each training bag as a subspace, or by using a random selection of instances (with replacement) as a subspace. The random method achieved better results, especially when the dimensionality of the subspaces was increased. In fact, we found that the subspace dimensionality is the most important factor affecting the performance of the ensemble. On the other hand, the number of subspaces does not play a very important role and just a few classifiers are sufficient for good performance. These conclusions are in line with conclusions from other applications of the random subspace method, where the amount of redundancy of the features is unknown.

In general, the informativeness of a prototype is more related to the dimensionality of the subspace, then to the label of instances forming that subspaces. Negative bags, and unlabeled random sets of instances were often good prototypes, suggesting that most instances, and not only a few concept ones, are informative for these MIL problems. These results are more in line with the collective assumption for MIL, where all instances are considered to contribute to the bag label, rather than with the standard assumption, where only a few positive instances are considered important.

Based on the encouraging results concerning the effectiveness of random subspaces as prototypes, we also considered randomly sampling the instance space (rather than randomly selecting existing instances) to generate artificial prototype bags that are not in the training set. Although the results with artificial prototypes were slightly worse than with real prototypes, this does seem to provide opportunities for using unlabeled, or artificial bags in a semi-supervised way.

We would like to conclude by emphasizing that a dissimilarity-based representation combined with a linear classifier (or an ensemble thereof) is a powerful way of classifying MIL bags. A question that still remains is the use of structured norms in such linear classifiers, which would enable selection of groups of dissimilarities, therefore revealing more about the relationships of the instances.

5

Stability of Instance Labels in Multiple Instance Learning

We address the problem of *instance label stability* in multiple instance learning (MIL) classifiers. These classifiers are trained only on globally annotated images (bags), but often can provide fine-grained annotations for image pixels or patches (instances). This is interesting for computer aided diagnosis (CAD) and other medical image analysis tasks for which only a coarse labeling is provided. Unfortunately, the instance labels may be unstable. This means that a slight change in training data could potentially lead to abnormalities being detected in different parts of the image, which is undesirable from a CAD point of view. Despite MIL gaining popularity in the CAD literature, this issue has not yet been addressed. We investigate the stability of instance labels provided by several MIL classifiers on 5 different datasets, of which 3 are medical image datasets (breast histopathology, diabetic retinopathy and computed tomography lung images). We propose an unsupervised measure to evaluate instance stability, and demonstrate that a performance-stability trade-off can be made when comparing MIL classifiers.

A shorter version of this chapter is accepted for publication as:
Veronika Cheplygina, Lauge Sørensen, David M. J. Tax, Marleen de Bruijne and Marco Loog. Label Stability in Multiple Instance Learning. In *Medical Image Computing and Computer Assisted Interventions*, 2015.

5.1 Introduction

Obtaining ground-truth annotations for voxels, which can be used to train supervised classifiers for voxel-based image segmentation can be very costly and time-consuming. The same holds for annotations of image patches, which can be used for localization of patches with abnormalities. This hinders the application of supervised classifiers for these tasks. Fortunately, global labels for whole images, such as the overall condition of the patient or a visual rating by an expert, are available more readily. Multiple instance learning (MIL) is an extension of supervised learning which can train classifiers – for whole images, and/or for image parts – using only such weakly labeled data. For example, a classifier trained on images (which in the MIL setting are referred to as bags), where each bag is labeled as healthy or abnormal and is represented by several unlabeled image patches (instances in MIL), would be able to indicate where abnormalities are detected in novel images.

MIL is becoming more and more popular in CAD [13, 37, 89, 116, 118, 142, 167, 187, 192]. In many of these applications, it is desirable to obtain instance labels, and to inspect the instances which are deemed positive. For example, in [118], x-ray images of healthy subjects, and patients affected by tuberculosis are used to train a classifier which, for a test image, can provide local abnormality scores, which in turn can be visualized as a heatmap. In [193], histopathology images are used to train a classifier which, for a test image, can provide an overall cancer diagnosis, as well as a pixel-level segmentation.

A pitfall in using MIL classifiers to obtain instance labels is that the classifiers might be unstable in their decisions for individual instances, for example, if a different subset of the data is used for training. This is clearly undesirable in a diagnostic setting, because, abnormalities could be highlighted in different parts of the image. For example, in [116] a MIL classifier is used to identify which regions (instances) of the tibial trabecular bone (bag) are most related to cartilage loss (positive bag label). Each region is annotated with the percentage of classifier evaluations during cross-validation, in which the region was labeled positive. When the bone is divided into 8 regions or more, this number is at most 20%. This shows that the classifiers disagree on which instances to label as positive. We have not been able to identify other research where this phenomenon is observed or investigated, which emphasizes the importance of the present work.

In rare cases where instance-level annotations are available, such as in [89], instance-level classifiers can be evaluated using AUC. The results here show that the best bag classifier does not correspond to the best instance classifier, emphasizing that bag-level results are not reliable if instance-level classifications are needed. Another approach is to evaluate the instances qualitatively, by visually examining the parts of the image, found to be abnormal in a single run of the classifier. For example, in [118] tuberculosis abnormality scores, which can be used for visual examination, are produced by a classifier trained on a predefined training set. This raises the question whether the same

abnormalities would be found if the classifier would be trained on slightly different data.

We propose to evaluate the *stability* of instance-labeling MIL classifiers as an additional measure for classifier comparison. We evaluate two stability measures on three CAD datasets: computed tomography lung images with chronic obstructive pulmonary disease (COPD), histopathology images with breast cancer and diabetic retinopathy images. We demonstrate how stability varies in popular MIL classifiers, and show that choosing the classifier with the best bag-level performance may not lead to reliable instance labels.

5.2 Multiple Instance Learning

In multiple instance learning [49], a sample is a bag or set $B_i = \{\mathbf{x}_k^i | k = 1, ..., n_i\} \subset \mathbb{R}^d$ of n_i instances, each instance is thus a d-dimensional feature vector. We are given training samples $\{(B_i, y_i) | i = 1, ...N_{tr}\}$ where y_i are labels and $y_i \in \{0, 1\}$. The standard assumption is that there exist hidden instance labels $z_k^i \in \{0, 1\}$ which relate to the bag labels as follows: a bag is positive if and only if it contains at least one positive instance. More relaxed assumptions allowing positive instances in negative bags have also been explored. For a recent overview, see [3].

Originally, the goal in MIL is to train a bag classifier f_B which would be used to label previously unseen bags. Several MIL classifiers do this by inferring an instance classifier f_I, and combining the outputs of the bag's instances. If we consider posterior probabilities as outputs, we can define the noisy-or rule:

$$\frac{p(y = 1|B_i)}{p(y = 0|B_i)} = \frac{1 - \prod_{k=1}^{n_i}(1 - p(z_k^i = 1|\mathbf{x}_k^i))}{\prod_{k=1}^{n_i} p(z_k^i = 0|\mathbf{x}_k^i)}, \tag{5.1}$$

which reflects the standard assumption. The average rule, however, assumes that all instances in a bag contribute to its label:

$$\frac{p(y = 1|B_i)}{p(y = 0|B_i)} = \frac{\sum_{k=1}^{n_i} p(z_k^i = 1|\mathbf{x}_k^i)}{\sum_{k=1}^{n_i} p(z_k^i = 0|\mathbf{x}_k^i)}. \tag{5.2}$$

Examples of such classifiers, called instance-level classifiers, are miSVM [5] and mil-Boost [183], which are MIL adaptations of popular learning algorithms. For example, miSVM extends the SVM by not only searching for the hyperplane $\mathbf{w}$, but also for the instance labels:

$$\min_{\{z_i^k\}} \min_{\mathbf{w},\xi} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i,k}\xi_i^k \qquad \text{s.t.} \qquad (5.3)$$
$$\forall i,k : z_i^k(\langle \mathbf{w}, \mathbf{x}_i^k \rangle) \geq 1 - \xi_i^k, \quad \xi_i^k \geq 0, z_i^k \in \{-1,1\}, \max\{z_i^k\} = y_i.$$

Another intuitive classifier that can provide instance labels is simpleMIL, i.e. propagating the bag labels to the instances, performing supervised classification, and combining the instance outputs per bag.

Another group of classifiers, called bag-level classifiers, typically represent each bag as a single feature vector and use supervised classifiers for training f_B [36, 42]. Such classifiers are often robust, but the representation step can mean that instance labels cannot be obtained. A notable exception is MILES [36], which represents each bag by its similarities to a set of prototype instances (such as all instances from the training set), $\mathbf{s}_i = [s(B_i, \mathbf{x}_1^1), \ldots, s(B_i, \mathbf{x}_{n_1}^1), \ldots s(B_i, \mathbf{x}_{n_{N_{tr}}}^{N_{tr}})]$ where $s(B_i, \mathbf{x}) = \exp(-\min_k ||\mathbf{x} - \mathbf{x}_k^i||)$ or any other kernel. A sparse classifier then selects the most discriminative features, which correspond to discriminative instance prototypes. It is assumed that discriminative prototypes from positive bags are positive, instances can therefore be classified based on their similarity to these prototypes.

5.2.1 MIL in Computer Aided Diagnosis

In the last decade, MIL has been applied for detection and/or localization of various diseases. We summarize several examples of such studies in Table 5.1. We can see that supervised evaluation of instances is only performed in a few studies – those where (a part) of the data has been annotated at the instance level. Where this is not possible, some papers examine the instances qualitatively, for example, by displaying the most positive (i.e. most abnormal) instances [118, 187]. One paper, [116], uses an approach that can be seen as both qualitative and unsupervised, by aggregating the outputs of classifiers trained in different rounds of cross-validation, and displaying how often each test instance is labeled positive. However, this analysis is explorative rather than evaluative and no comparison across classifiers is performed. Lastly, a number of papers perform no instance-level evaluation at all, although instance labels would be interesting from a diagnostic point of view [37, 89]. As our proposed evaluation is unsupervised (i.e., instance annotations are not needed), the measure can easily be adopted in all the studies summarized here.

Table 5.1: Applications of MIL in CAD tasks. Abbreviations: chronic obstructive pulmonary disease (COPD), computed tomography (CT), radiograph (XR), electrocardiogram (ECG), accuracy (Acc), area under receiver operating characteristic curve (AUC), sensitivity (Se), specificity (Sp), precision-recall (PR), false positives (FP), region of clinical interest (RCI).

Data	Evaluation of bags	Evaluation of instances
Barett's cancer histology [89]	Acc, F1, AUC, PR	Acc, F1, AUC, PR
Diabetic retinopathy [89]	Acc, F1, AUC, PR	–
COPD in CT [37, 164]	AUC	–
Tuberculosis in XR [118]	Se-Sp	Qualitative
Osteoarthritis in MRI [116]	AUC	Qualitative / unsupervised
Diabetic retinopathy [142]	Acc, AUC	AUC, Qualitative
Myocardial infarction in ECG [167]	Acc, Se, Sp	–
Cancer histopathology [193]	Se-Sp	F1
Breast ultrasound[50]	Se, Sp, Acc	–
Colonography in CT [187]	AUC, Se-Sp	Qualitative
Pulmonary embolism in CT [60]	Se-FP, AUC, AUC-RCI	–
Colorectal cancer in CT [60]	Se-FP, AUC, AUC-RCI	–
Pulmonary angiography in CT [110]	Se-FP	–

5.3 Instance Stability

The general concept of stability is important in machine learning, and different aspects of it have been addressed in the literature. Leave-one-out stability [139] measures to what extent a decision boundary changes when a training sample $(\mathbf{x}_i, z_i)$ is removed from the training data. Feature selection stability [99] evaluates to what extent two outputs of feature selection procedures are similar. Clustering stability [10] similarly compares the outputs of two clustering procedures. The kappa statistic for interobserver agreement [181] can be used to measure the similarity of two sets of labels. As we will see shortly, most of these measures are not applicable for use in a MIL setting.

We are interested in evaluating the similarity of a labeling, or vector of outputs of two classifiers $\mathbf{z} = f_I(X)$ and $\mathbf{z}' = f'_I(X)$ where $X = [\mathbf{x}_1^1, \ldots, \mathbf{x}_{n_N}^N]^\intercal$. The stability measure should have the following properties:

Monotonicity The value should be monotonically increasing with the number of instances that the classifiers agree on.

Limits Disagreeing on all instances should result in stability of 0 and full agreement should result in stability of 1.

Dataset size The value must not depend on the number of instances, because the goal is to compare classifiers for a test set of fixed size.

Unsupervised The value must not depend on the hidden instance labels z_i.

The monotonicity property renders the kappa statistic unsuitable, for example $\kappa([0,1,1,1,0]^\intercal, [1,0,0,1,0]^\intercal) > \kappa([0,1,1,1,1]^\intercal, [1,0,1,1,1]^\intercal)$ which is counterintuitive based on the number of samples the classifiers agree on. The feature selection index considers selecting all features unstable, and similarly violates the monotonicity, because agreeing that all samples are of the same class would translate into stability of 0. Leave-one-out stability is a supervised measure, and therefore not applicable for instances in MIL data. Only the clustering stability index seems suitable, and is in fact a different formulation of the measures we propose in what follows.

Let $n_{00} = |\{i|z_i = 0 \wedge z'_i = 0\}|$, $n_{01} = |\{i|z_i = 0 \wedge z'_i = 1\}|$, $n_{10} = |\{i|z_i = 1 \wedge z'_i = 0\}|$ and $n_{11} = |\{i|z_i = 1 \wedge z'_i = 1\}|$. An intuitive measure that satisfies the properties above is the agreement or matching fraction:

$$S(\mathbf{z}, \mathbf{z}') = \frac{n_{00} + n_{11}}{n_{01} + n_{10} + n_{11} + n_{00}}. \tag{5.4}$$

A property of S is that, in a situation with many true negative instances, the agreement is inflated due to all the negative instances that the classifiers agree on. As a result, the classifier can still be unstable with respect to the instances it classifies as positive. Due

to the nature of CAD tasks, we might consider it more important for the classifiers to agree on the positive instances. In this case, in the definitions of the properties above, "instances" would be replaced by "positive instances". Therefore we also consider the agreement on positive labels only, which is related to the Jaccard distance:

$$S_+(\mathbf{z}, \mathbf{z}') = \frac{n_{11}}{n_{01} + n_{10} + n_{11}}. \tag{5.5}$$

5.3.1 Classifier Selection

If instance classification stability is a crucial issue, one can study our measure in combination with bag-level AUC (or any other accuracy measure) and select a MIL classifier with a good trade-off of AUC and instance stability. We can see each classifier as a possible solution, parametrized by these two values. Intermediate solutions between classifiers f_I and f'_I can in theory be obtained by designing a randomized classifier, which trains classifier f_I with probability p and classifier f'_I with probability $1 - p$.

In the AUC-stability plane, the Pareto frontier is the set of classifiers which are Pareto efficient, i.e. no improvement can be made in AUC without decreasing instance stability and vice versa. Optimal classifiers can therefore be selected from this Pareto frontier. While the classifier with the highest AUC is in this set, it is not necessarily the only desirable solution, if the instance labels are of importance.

5.4 Experiments

5.4.1 Datasets

The datasets used in our experiments are shown in Table 5.2. In Musk, which are the original problems for which MIL was proposed, a bag is a molecule and an instance is a conformation of that molecule. Molecules smell musky (+) or not (–), depending on their conformations. In the Breast dataset, a bag is a 896×768 tissue microarray analysis image from a patient with a malignant (+) or benign (–) tumor. An instance is a 7×7 patch described by histogram, LBP and SIFT features, as well as features extracted from the cells detected in each patch. In the Messidor dataset, an instance is a 135×135 patch from a 700×700 fundus image of a diabetes (+) or healthy (–) subject. Histogram, LBP and SIFT features are used. In the COPD dataset, a bag is a CT image of a lung of a subject with COPD (+) or a healthy subject (–). An instance is a region of interest (ROI) of $41 \times 41 \times 41$ voxels, with the center inside the segmentation of the lung field. Each

Table 5.2: List of datasets and corresponding numbers of bags, instances and features. Musk, Breast and Messidor datasets are public and can be downloaded from a repository of MIL datasets [42] (http://www.miproblems.org).

Name	Type	Bags	Inst	Inst/bag	Feat
Musk 1	benchmark MIL data [49]	47+, 45–	476	2 to 40	166
Musk 2	benchmark MIL data [49]	39+, 63–	6598	1 to 1024	166
Breast	cancer histopathology [90]	26+, 32–	2002	21 to 40	657
Messidor	diabetic retinopathy [45, 89]	654+, 546–	12352	8 to 12	687
		31+, 31– (tr)	6200 (tr)		
COPD	lung CT images [129, 164]	100+, 100– (va)	10000 (va)	50	287
		100+, 100– (te)	10000 (te)		

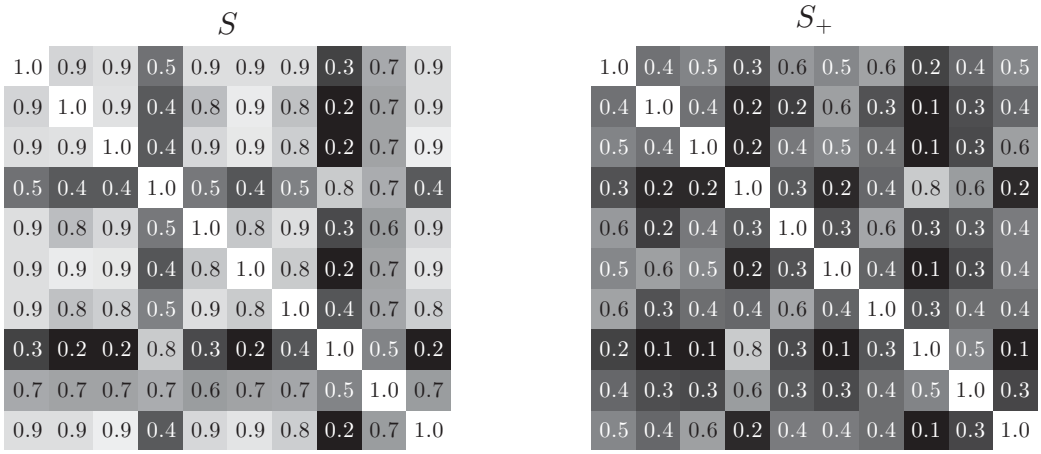


Figure 5.1: Pairwise stability for 10 MILES classifiers for agreement (left) and positive agreement (right) for the COPD dataset.

ROI is described by histograms of responses from a Gaussian filter bank. The feature size is reduced by a forward selection procedure, as described in [164].

5.4.2 Illustrative Example

We show how the instance labels change in the COPD validation dataset, for a MILES classifier, trained 10 times on random 80% of the training data. The pairwise stability measures are shown in Fig. 5.1. Using both measures there is considerable disagreement about the instance labels, which is suprising because of the large overlap of the training sets. The measures are quite correlated ($\rho = 0.76$), but S has higher values than S_+ because S is inflated by agreement on negative instances. The values of S_+ show that some of the classifiers are almost in complete disagreement about which instances to label as positive.

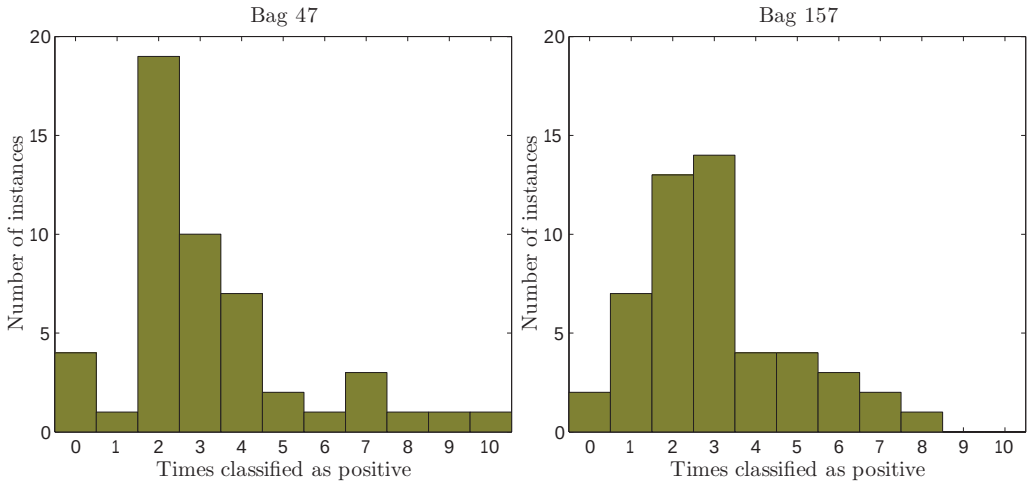


Figure 5.2: Distribution of agreement on the "positiveness" of 50 instances from 2 positive bags (COPD patients) for 10 MILES classifiers. The x-axis shows how often an instance is classified positive, the y-axis shows for how many instances this holds.

Fig. 5.2 shows how the instance classifications change in two true positive bags, i.e., CT images from COPD patients. These bags are always classified as positive, but the instance labels are unstable. A perfectly stable classifier would have a bimodal distribution, classifying instances as positive either 0 or 10 times. The situation is quite different here, as this only happens for very few instances. Fig. 5.3 shows a number of patches with stable and unstable labels from bag 157. Several patches containing emphysema have unstable classifications, and one emphysemous patch is even consistently classified as negative. This example shows that while the bag is always classified correctly, the instance labels may not be very reliable.

5.4.3 Classifier Evaluation

We evaluate a number of classifiers from the MIL toolbox [169], which we modified to output instance labels. Each classifier normalizes the training set to zero mean and unit variance, and applies the same normalization during the test phase. We use the following classifiers:

- simpleMIL with nearest mean (NM), 1-nearest neighbor (1-NN) and SVM
- miSVM with averaging combining rule
- miNM and mi1NN, versions of miSVM based on NM and 1-NN
- MILBoost with noisy-or combining rule and 100 reweighting rounds
- MILES

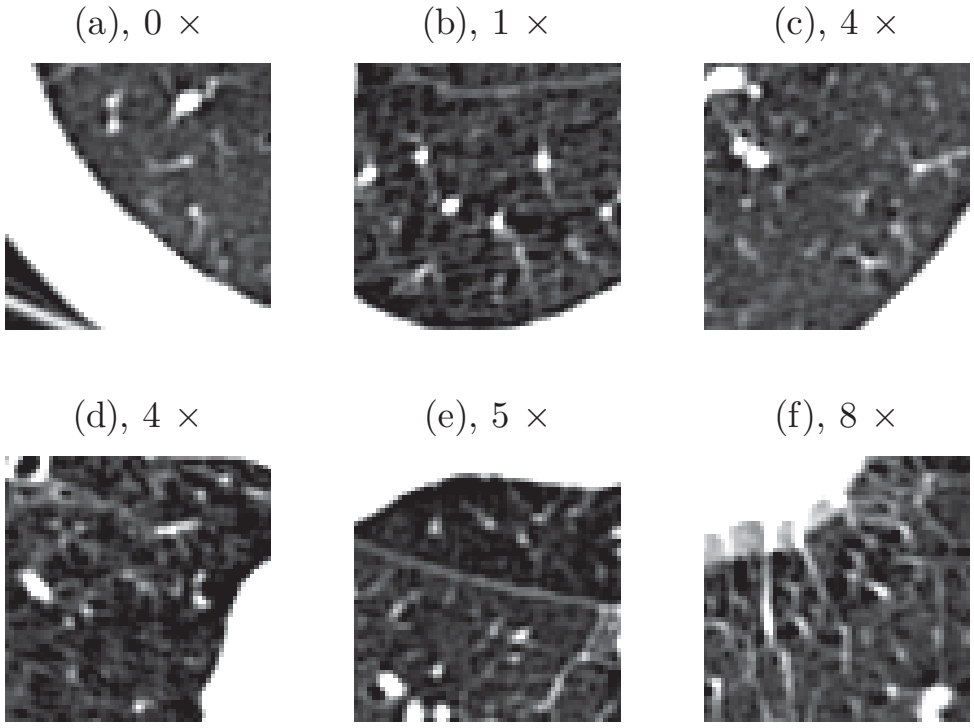


Figure 5.3: Visualization of the axial slice with most lung voxels below -910 hounsfield units in selected instances, or ROIs, from a COPD patient (bag 157). ROI size $61 \times 61 \times 61$ is used for better visualization. Intensity range [-1000 -400] is used. Patches (b), (d), (e) and (f) contain emphysema, which can be seen by the low intensity blobs or areas within the gray lung tissue, but only patch (f) is often classified as positive. Patches (a) and (c) are largely unaffected, but only (a) is consistently classified as negative.

We use a linear kernel and regularization parameter $C = 1$ for SVM, miSVM and MILES, and 100 reweighting rounds for MILBoost. These defaults led to computational problems for Messidor and hence we used $C = 1$ and 10 rounds. For each training/test split, we do the following 10 times: randomly sample 80% of the training bags (bag = subject), train the classifier, and evaluate on the test dataset. The train/test splits are done as follows: randomly 5 times for Musk and Breast, randomly 2 times for Messidor, and based on predefined sets for COPD, i.e. using the training set for training, and using both validation and test sets for testing.

5.5 Discussion

The average bag AUCs, average pairwise instance stabilities, and the corresponding Pareto frontiers are shown in Fig. 5.4. Note that the (1, 0.5) point can, both for S and S_+, be achieved by a classifier which labels all instances as positive. The main observation is that the most accurate classifier is often not the most stable one – there is a trade-off of bag AUC and instance stability. This is especially well-illustrated in the COPD validation and test datasets, where the classifiers range from accurate but unstable, to stable but inaccurate. In these datasets, the behavior across classifiers is similar. This shows that if we were to use the results on the validation set for classifier selection, we would obtain a classifier with similar performance and stability on the test set.

With regard to the classifiers, miSVM and its variants miNM and mi1NN seem to be relatively good choices. Due to its high bias and low variance, miNM is usually less accurate, but more stable. MILES, which is a popular classifier due to its good performance, can indeed be quite accurate, but at the same time unstable. The difference between the mi- classifiers and MILES is probably due to the fact that MILES trains a bag classifier f_B first, and infers f_I from f_B, while the mi- classifiers build f_I directly. MILBoost is both inaccurate and unstable. Its discrepancy in S and S_+ for COPD illustrates that the classifiers disagree on which instances to label as positive.

Note that the goal of these experiments is to demonstrate the trade-off between AUC and stability, not to maximize the AUC. The performances shown may be lower than in literature, because only 80% of the training data is used, and no parameter selection is performed. However, we did briefly examine the effect of different parameters. The combining rule noisy-or (5.1) for mi- classifiers has the same stability as the averaging rule (5.2), but a lower AUC. The regularization parameter C does not have a large effect on the AUC or stability of MILES, but does cause a trade-off for miSVM. The direction of the trade-off varies per dataset, i.e., more regularization does not necessarily lead to more stable classifiers. If sufficient training data is available, we recommend to follow the regular procedure of parameter selection, but to take stability into account when choosing a set of parameters.

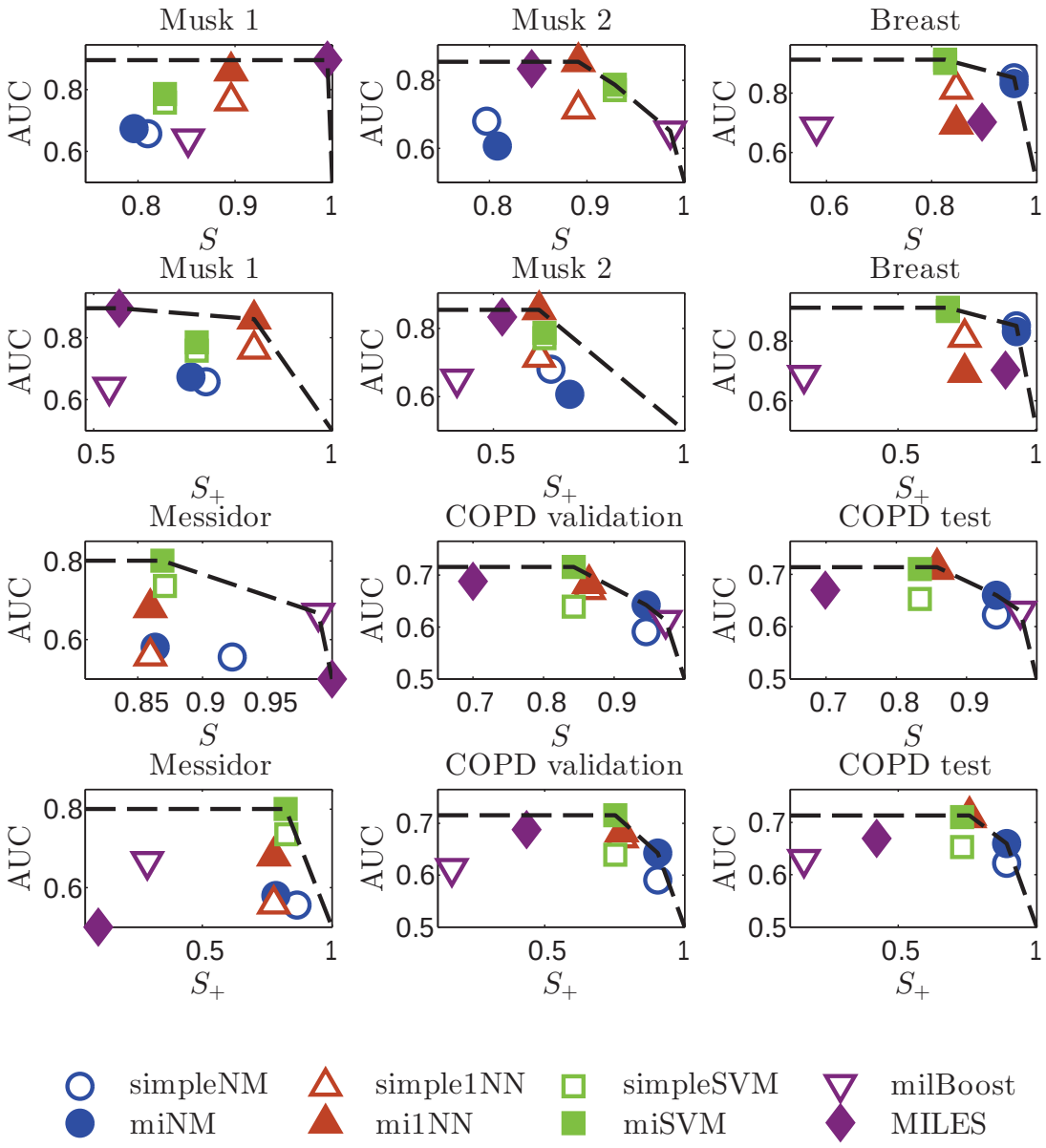


Figure 5.4: Bag AUC vs instance stability and the corresponding Pareto frontiers.

An open issue is how to take stability into account when designing a MIL-based CAD system. When using the training data, the classifier needs to be trained and tested several times, which could be impractical. It is worth examining how many classifiers, trained on which fraction of the data, are needed to get a good estimate of the stability. On the other hand, multiple trained classifiers present an opportunity because they can be used in a bagging ensemble [17], potentially surpassing the performance of a single classifier. Another interesting question is how stability is affected by the instance sampling. For example, if the instances are patches, as is the case for the 3 CAD datasets we used, the overlap of the patches and the patch size are both likely to affect the classifier decisions. As such, stability can be used not only in the classifier selection step, but also in the data preprocessing step. Finally, a remaining challenge is evaluating instance-level stability against instance-level performance, which calls for releasing fully annotated datasets to the public.

5.6 Conclusions

We addressed the issue of stability of instance labels provided by MIL classifiers. We examined two unsupervised measures of agreement: S based on all labels, and S_+ based on positive labels. Our observations suggest that S_+ might be more interesting from a CAD point of view, because this means that the classifiers agree on which parts of the image to label as abnormal. Our experiments demonstrate that the classifiers which perform well on bag-level, may not necessarily provide the most stable instance labels. Classifiers that provided good trade-offs of bag performance and instance stability are miSVM, and its variants based on the nearest mean and nearest neighbor classifiers. In general, we propose to use instance-level stability as an additional classifier evaluation measure when applying MIL classifiers in CAD. Open issues include examining the conditions needed for estimating stability, the effect of instance sampling on stability, as well as validating the measures on datasets, annotated at the instance level.

6

Classification of COPD with Multiple Instance Learning

Chronic obstructive pulmonary disease (COPD) is a lung disease where early detection benefits the survival rate. COPD can be quantified by classifying patches of computed tomography images, and combining patch labels into an overall diagnosis for the image. As labeled patches are often not available, image labels are propagated to the patches, incorrectly labeling healthy patches in COPD patients as being affected by the disease. We approach quantification of COPD from lung images as a multiple instance learning (MIL) problem, which is more suitable for such weakly labeled data. We investigate various MIL assumptions in the context of COPD and show that although a concept region with COPD-related disease patterns is present, considering the whole distribution of lung tissue patches improves the performance. The best method is based on averaging instances and obtains an AUC of 0.742, which is higher than the previously reported best of 0.713 on the same dataset. Using the full training set further increases performance to 0.776, which is significantly higher (DeLong test) than previous results.

This chapter is published as:
Veronika Cheplygina, Lauge Sørensen, David M. J. Tax, Jesper Holst Pedersen, Marco Loog, and Marleen de Bruijne. Classification of COPD with multiple instance learning. In *International Conference on Pattern Recognition*, pages 1508-1513, 2014.

6.1 Introduction

Chronic obstructive pulmonary disease (COPD) is a disease of the lungs that is caused, among others, by smoking and air pollution. COPD is characterized by chronic inflammation of the lung airways, and degradation of lung tissue, called emphysema, both of which result in airflow limitation [31, 143]. The disease progresses in several stages and can eventually lead to death, however, detecting the disease at an early stage can increase the survival rate [128].

Due to limitations of traditional spirometry and visual assessment of computed tomography (CT) scans, texture classification was proposed to quantify COPD [119, 127, 164, 165, 175]. One approach is to classify patches of lung tissue, or regions of interest (ROIs) in the image, and combine the classifications into an overall probability for COPD [127, 165]. However, these supervised approaches require manually annotated ROIs, which are difficult and costly to obtain.

An alternative is to use weakly labeled medical images, i.e., where only a global image label is provided, for training an image classifier. In the absence of labeled ROIs, the image label can be propagated to its ROIs, and an ROI classifier can be trained as usual [164]. We call this straightforward approach SimpleMIL. However, this disregards the fact that in scans of patients with COPD, only a subset of the ROIs may be affected, while signs of COPD may be already apparent in some regions for subjects not yet diagnosed with the disease. This increases the label noise for the ROI classifier.

A technique which can handle learning with such weakly labeled data is called multiple instance learning (MIL) [49, 114]. The goal is to build a classifier for a collection, or bag, of feature vectors, also referred to as instances. Often it is assumed that a bag is positive if and only if at least one of its instances is positive. A further assumption is that positive instances are found in a region of the feature space called the concept. For COPD, the concept could be a part of the feature space, containing ROIs that are typical for, for example, emphysema. In this scenario, as soon as a CT scan contains such an ROI, the whole image is diagnosed as COPD.

MIL methods can be broadly divided into two categories: instance-based and bag-based. Instance-based methods use the constraints posed by the bag labels and the MIL assumptions to build an instance classifier, and combine instance classifications to classify bags [5, 114, 183, 195]. On the other hand, bag-based methods aim to classify bags directly, often by defining kernels [71] or dissimilarities [171, 194] between bags.

Every MIL classifier makes explicit or implicit assumptions about the data. Instance-based classifiers typically rely on the assumption that there is a concept, and that positive bags contain instances from this concept. Therefore, only concept instances are important for determining the bag label. Bag-based classifiers assume that bags from the same class are similar, and the similarity definition further specifies this assumption.

In most definitions, all of the bag's instances are involved in defining the bag similarity, therefore the whole distribution of instances is important for the bag label. In [40] we have shown that many well-known MIL problems fall into these two categories (concept and distribution) and that this property determines how many MIL methods perform on the data.

Detection of COPD using lung texture has been tackled by classifying patches and combining their outputs, an approach we call SimpleMIL, in [164]. A more specialized MIL method, applied to this problem, is a dissimilarity-based approach in [162], and it shows promising results. Other dissimilarity or kernel-based approaches, which focus on the airways rather than lung texture, have also been successful for COPD classification [63, 161]. In this work we investigate a broader range of MIL methods for classification of the lung texture in COPD. We examine which assumptions, commonly used for the instance-based and bag-based methods, are more suitable for this problem, and demonstrate state-of-the-art results on a COPD dataset from the Danish Lung Cancer Screening Trial [129].

6.2 Multiple Instance Learning

In multiple instance learning (MIL), an object is represented by a bag $B_i = \{\mathbf{x}_{ik} | k = 1, ..., n_i\} \subset \mathbb{R}^d$ of n_i instances, where the k-th instance is described by a d-dimensional feature vector $\mathbf{x}_{ik}$. The training set $\mathcal{X}_{tr} = \{(B_i, y_i) | i = 1, ...N\}$ consists of positive ($y_i = +1$) and negative ($y_i = -1$) bags. One way to deal with this type of input is to propagate the bag labels to the instances, and building an instance classifier. A bag label is obtained by classifying that bag's instances, and combining the instance classifications, for example by fusing the posterior probabilities [111]. The noisy-or rule,

$$\frac{p(y=1|B_i)}{p(y=-1|B_i)} = \frac{1 - \prod_{k=1}^{n_i}(1 - p(z_{ik}=1|\mathbf{x}_{ik}))}{\prod_{k=1}^{n_i} p(z_{ik}=-1|\mathbf{x}_{ik})} \tag{6.1}$$

reflects the standard assumption that a bag is positive if and only if at least one of the instances is positive. On the other hand, the average rule,

$$\frac{p(y=1|B_i)}{p(y=-1|B_i)} = \frac{\sum_{k=1}^{n_i} p(z_{ik}=1|\mathbf{x}_{ik})}{\sum_{k=1}^{n_i} p(z_{ik}=-1|\mathbf{x}_{ik})} \tag{6.2}$$

assumes that all instances contribute to the bag label. This fusion rule has been used in [164], by classifying ROIs with a nearest neighbor classifier, and combining the outputs to classify the entire image. We refer to this strategy as SimpleMIL in the experiments.

The standard assumption for MIL is that there are hidden instance labels z_{ik} which relate to the bag labels as follows: a bag is positive if and only if it contains at least one

positive, or *concept* instance [49]. The strategy of earlier MIL approaches was to model the concept: a region in feature space which contains at least one instance from each positive bag, but no instances from negative bags. Diverse density [114] (DD) has been proposed to measure this property. For a given point t in the feature space, $DD(t)$ measures the ratio between the number of positive bags which have instances near t, and the sum of distances of the negative instances to t. The point where DD is maximized, t^* therefore corresponds to the target concept. Instances can be classified using their distance to t^*. However, the optimization problem suffers from local optima and, for the original DD algorithm, several restarts of the algorithm are needed. Therefore, an expectation-maximization version of this algorithm EM-DD [195] has been proposed. EM-DD has shown to perform well on a range of MIL problems, but is also very computationally intensive.

Several regular supervised classifiers have been extended to work in the MIL setting. One example is mi-SVM [5], an extension of support vector machines which attempts to find hidden labels of the instances under constraints, such as (6.1) or (6.2), posed by the bag labels. Another example is MILBoost [183], where the instances are reweighed in each of the boosting rounds. The instance weights indicate how informative the instances are in predicting the bag labels.

It has been recognized that the standard assumption might be too strict for certain types of MIL problems. Therefore, relaxed assumptions have emerged [188], where a fraction or a particular number of positive instances are needed to satisfy a concept, and where multiple concept regions are considered. In the case of COPD, this would correspond to the presence of a certain fraction of ROIs containing affected tissue, and/or different types of disease patterns. However, if the number of concepts and the fraction of positives per concept, are not given in advance, these extra parameters also need to be set using the training data, further increasing the risk of overtraining.

Therefore, methods which compare bags without explicitly relying on the standard, or relaxed assumptions, have been proposed. Such methods include Citation-kNN [186], and bag kernels [71]. Citation-kNN uses the Hausdorff distance between bags. For classification, both the k_R "referencing" nearest neighbors of a bag B, and the k_C "citing" neighbors (bags for which B is nearest neighbor) are taken into account. In [71], a bag kernel is defined either as a sum of the instance kernels, or as a standard (linear or radial basis) kernel on a summarized representation of the bag. This summary is created by, for each feature, averaging the bag's instances (which we refer to by mean-inst), or using both the minimum and the maximum instance values (which we refer to by extremes). In all cases, the way a kernel is defined affects which (implicit) assumptions are made about the problem. A drawback for real-world applications is that kernels must be positive semi-definitive, therefore excluding some domain-specific similarity functions.

Other bag-based methods have addressed MIL by representing each bag by (dis)similarities to a set of prototypes $\mathcal{R} = \{R_1, \ldots, R_M\}$ in a so-called dissimilarity space [133]. Therefore, each bag is represented by a single feature vector $\mathbf{d}(B_i, \mathcal{R}) =$

$[d(B_i, R_1), \ldots, d(B_i, R_M)]$, where d is a (dis)similarity measure. In this space, any supervised classifier can be used. In MILES [36], all training instances are used as prototypes, creating a very high-dimensional representation. A sparse classifier is used to select the most discriminative similarities, and therefore, instances. In a bag-of-words approach, prototypes are "words", or clusters of instances, and the dissimilarity measure between a bag and a word is the number of instances, belonging to that cluster.

Using bags as prototypes [40, 171] reduces the dimensionality, and therefore, the possibility of overtraining. In this paper, we use all training bags as prototypes, i.e., $\mathcal{R} = \mathcal{X}_{tr}$, but prototype selection can be further used to reduce $|\mathcal{R}|$. The advantage of such methods is that more can be gained from the training data than in nearest neighbor approaches, and that there are no restrictions on the bag similarity function [136]. In this paper, for example, we use two definitions of d that are not necessarily metric: the average minimum instance distance (6.3), and the earth mover's distance (EMD) [155], defined in (6.4). Herein, the instance dissimilarity d is the squared Euclidean distance.

$$d_{meanmin}(B_i, B_j) = \frac{1}{n_i} \sum_{k=1}^{n_i} \min_l d(\mathbf{x}_{ik}, \mathbf{x}_{jl}) \tag{6.3}$$

$$d_{\text{EMD}}(B_i, B_j) = \sum_{\mathbf{x}_k \in B_i, \mathbf{x}_l \in B_j} f(\mathbf{x}_k, \mathbf{x}_l) d(\mathbf{x}_k, \mathbf{x}_l) \tag{6.4}$$

where $f(\mathbf{x}_k, \mathbf{x}_l)$ is the flow that minimizes the overall distance, and that is subject to constraints that ensure that the only available amounts of "earth" (instances of B_i) are transported into available "holes" (instances of B_j), and that all of the instances are indeed transported: $f(\mathbf{x}_k, \mathbf{x}_l) \geq 0$, $\sum_{\mathbf{x}_k \in B_i} f(\mathbf{x}_k, \mathbf{x}_l) \leq 1/n_j$, $\sum_{\mathbf{x}_l \in B_j} f(\mathbf{x}_k, \mathbf{x}_l) \leq 1/n_i$ and $\sum_{\mathbf{x}_k \in B_i, \mathbf{x}_l \in B_j} f(\mathbf{x}_k, \mathbf{x}_l) = 1$.

6.3 Experiments

We use the dataset from [164], which describes how CT lung images from the Danish Lung Cancer Screening Trial [129] have been processed. Parts of such images highlighting healthy and emphysemous lung tissue, are shown in Fig. 6.1.

The dataset consists of three parts: training set $\mathcal{X}_{tr}$, validation set $\mathcal{X}_{val}$ and test set $\mathcal{X}_{te}$. Originally, each of these parts consists of 100 COPD (positive) and 100 healthy (negative) images. In previous work [164], a subset of the training data with 31 COPD and 31 healthy images was selected to improve the class separability in the training set. We therefore refer to the full training data as $\mathcal{X}_{tr}$ and to the subsampled training data as $\mathcal{X}_{sub}$.

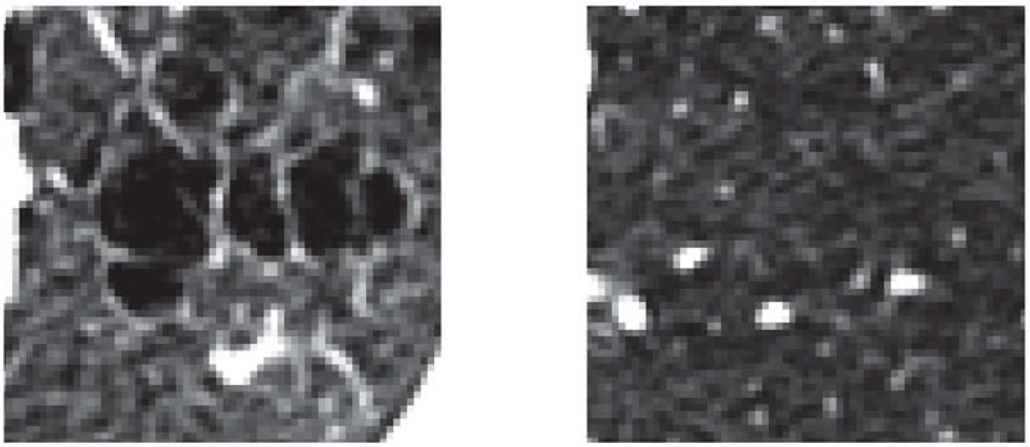


Figure 6.1: Examples of patches containing centrilobular emphysema (left), characterized by black holes within the lung tissue, and healthy tissue (right). Both images are approximately 1.5 times the size of the ROIs used for classification and the intensity values have been rescaled to facilitate viewing.

Each image is represented by 50 ROIs, sampled at random locations within the lungs. Each ROI is described by histograms of responses of 8 filters at 7 scales, which aim to capture the texture of the image. The filters used the following: Gaussian, gradient magnitude, Laplacian of Gaussian, first, second and third eigenvalue of the Hessian, Gaussian curvature and eigen magnitude. The scales range from 0.6 to 4.8 mm. The responses of each filter at each scale are stored in a histogram with 41 bins. This approach creates a 2296-dimensional feature vector for each ROI. In [164], the validation set was used to select the most appropriate filters and scales. Because these features are selected for a particular classifier only (SimpleMIL with nearest neighbor classifier), we use the full feature set here for all the classifiers.

The evaluated classifiers are available from the MIL toolbox [169] and PRTools [56]. We evaluate the following selection:

- SimpleMIL with a logistic (regularization parameter $C \in \{0.01, 0.1, 1, 10\}$) and nearest neighbor ($k \in \{25, 35, 45\}$) classifiers. We consider both noisy-or and average fusion rules.
- EM-DD with 10% of instances used for initialization
- miSVM with a polynomial kernel, where $p \in \{1, 2\}$ is the degree of the polynomial and $C \in \{0.01, 0.1, 1, 10\}$ is a regularization parameter. We consider both noisy-or and average fusion rules.
- MILBoost with 100 reweighting rounds

- Citation k-NN, $k_R \in \{1,5,10\}$, $k_C \in \{1,5,10\}$
- Averaging the instances (mean-inst), and minimum and maximum feature values for each bag (extremes) with an SVM, $p \in \{1,2\}$, $C = \{0.01, 0.1, 1, 10\}$.
- Bag-of-words (BoW) with {50, 100, 200} words and an SVM, $p \in \{1,2\}$, $C \in \{0.01, 0.1, 1, 10\}$
- MILES with a polynomial kernel, $p \in \{1,2\}$, $C \in \{0.01, 0.1, 1, 10\}$
- Bag dissimilarities (meanmin and emd) with k-NN, $k \in \{1,5,10\}$, and in the dissimilarity space with an SVM, $p \in 1,2$, $C = \{0.01, 0.1, 1, 10\}$

We perform evaluation in three ways. First, each classifier (with different parameter settings) is trained on $\mathcal{X}_{sub}$ and $\mathcal{X}_{tr}$, depending on the experiment. Each classifier is then evaluated on $\mathcal{X}_{val}$. The evaluation metric is the area under the receiver-operating characteristic curve, or AUC. We report the best of these performances on $\mathcal{X}_{val}$, and select the corresponding parameters. We then report the performance of this classifier with the best parameters on an independent test set $\mathcal{X}_{te}$. The difference in AUC on $\mathcal{X}_{val}$ and $\mathcal{X}_{te}$ is an indicator of overtraining, i.e., fitting the parameters too well to the validation set. Lastly, we randomly select half of the bags in $\mathcal{X}_{sub}$ or $\mathcal{X}_{tr}$, 10 times. For each subsample, we train a classifier, select parameters using $\mathcal{X}_{val}$, and evaluate on $\mathcal{X}_{te}$. The average and the standard deviations of the 10 performances are reported. This result gives an indication of a situation where less training data is available, and of the variance in performance due to a different sampling of the data.

The performances are shown in Table 6.1. For each training dataset, we compare the performances per column. The best performance and performances not significantly worse than best, are shown in bold. We test for significant differences using the DeLong test for ROC curves [46] for the performances on $\mathcal{X}_{val}$ and $\mathcal{X}_{te}$, and using a dependent t-test for the 10 cross-validation performances, both at a significance level of 0.05. A few results are not reported. For EM-DD, time requirements were too high for both datasets. For miSVM, the instance kernel matrix for $\mathcal{X}_{tr}$ was too large to fit in memory.

6.4 Discussion

6.4.1 Classifier Performance

Across the different training datasets, we can see similar trends in the classifier performances. It is clear that some classifiers suffer from the high dimensionality. For example, the bag of words approach, which uses a mixture of Gaussians to estimate words in feature space, is not able to do so in 2296 dimensions. For BoW, MILBoost, and EM-DD which could not handle the dimensionality computationally, we performed additional experiments with the 287-dimensional feature set that resulted from a feature selection

procedure used in [164]. The results on $\mathcal{X}_{te}$ are 0.657 (BoW), 0.641 (EM-DD) and 0.551 (MILBoost), suggesting that these classifiers benefit from feature selection. It may of course generally be interesting to study feature selection for the other classifiers as well.

The full training dataset $\mathcal{X}_{tr}$ has two main differences with respect to $\mathcal{X}_{sub}$: higher class overlap, and more bags, and therefore instances in total. Several classifiers, such as SimpleMIL logistic, mean-inst, extremes, and SVM in the dissimilarity space, show increases in performances due to the higher sample size. On the other hand, MILES suffers from the increased sample size, because the dimensionality of the dissimilarity representation is equal to the number of instances. This also explains why MILES performs better when the training set is subsampled to 50%.

The performances of many methods do not degrade very much when only 50% of the bags are used for training. This suggests that the subsampled dataset is still representative for the whole data distribution, and each class can be described well with only a few samples. Furthermore, most classifiers do not suffer a lot from overtraining, as the difference in performance on $\mathcal{X}_{val}$ and on $\mathcal{X}_{te}$ is quite small. Notable exceptions are the k-NN classifiers trained on the full training set $\mathcal{X}_{tr}$, where the parameter k is overfit to the validation set, causing lower performances on $\mathcal{X}_{te}$.

SimpleMIL performs quite well, especially when posterior probabilities of all instances are taken into account, as in the averaging fusion rule. Methods which assume a concept, such as EM-DD and miSVM, also perform reasonably, which suggests that there is a region in feature space with a high density of disease patches and low density of normal patches. However, the performances are lower than those of bag-based methods, suggesting that detecting the concept is not sufficient for the diagnosis of COPD. This is also supported by the fact that miSVM with the averaging rule outperforms miSVM with the noisy or rule, which shows that it is beneficial to take all instance classifications into account.

Methods with assumptions on bag level have the best performances, in particular, averaging all the instances in a bag is already able to separate the bag classes quite well. This suggests that negative instances in positive bags, and the negative instances in negative bags, do not originate from the same distribution. In other words, scans affected with COPD do not contain the same types of healthy patches, as healthy scans. The disease appears to be more diffuse, affecting a large part of the lung rather than small isolated regions.

For the bag-based methods, the mean-inst bag representation and the dissimilarity-based SVM perform particularly well. MILES suffers from the high dimensionality, but we expect that the performance would improve if instance selection techniques would be used. Another interesting observation is that the dissimilarity-based SVM significantly outperforms k-NN on the same dissimilarities. SVM is able to use the dissimilarities of the training set to create a more robust classifier, which is consistent with results in [162], although slightly different dissimilarities are used there. We expect that fur-

ther investigation into different bag dissimilarity measures could further improve these results.

Unfortunately, our results are not directly comparable to the dissimilarity-based approach of [162], because an earlier version of the dataset was used. The results in [164], however, are obtained by training on $\mathcal{X}_{sub}$ and the performances can be compared. There, the best approach obtains an AUC of 0.713. Our results show superior performances when training on $\mathcal{X}_{sub}$, with an AUC 0.742 for mean-inst and 0.746 for d_{EMD} in the dissimilarity space. However, these performances are not significantly better than the result in [164]. Using $\mathcal{X}_{tr}$ further improves the results, for an AUC of 0.758 for SimpleMIL with a logistic classifier, 0.776 for mean-inst, and 0.754 for $d_{meanmin}$ in the dissimilarity space. The best approach using $\mathcal{X}_{tr}$ in [164] obtains an AUC of 0.690, and our performances are significantly better according to the DeLong test.

Furthermore, we examined the output of the best-performing classifiers to see which images still get misclassified. Rather than only looking at the positive label for COPD, we now use the COPD stages [143], from mild (I) to moderate (III). The results show that most of the confusion is between the healthy scans, and stage I scans, which supports our intuition about where the class overlap is largest. Because the classifiers differ in some of their errors, it may be of interest to combine their decisions.

6.4.2 Concept Region

In order to better understand the classifier performances, we examine a 2D projection of the instances, obtained by t-distributed stochastic neighbor embedding (t-SNE) [177] (Fig. 6.2). We see two clusters of instances, a smaller cluster in the top left and a larger cluster. In the small cluster the density of instances from positive bags is clearly higher, which suggests that part of it could be a concept region. To investigate whether these patches display emphysema, we examined the intensity histograms of the Gaussian filters at the smallest scale. As emphysema results in darker patches, we would expect patches with emphysema to have intensity histograms skewed to the left. This is exactly what we find when averaging all the instances per cluster and plotting the two corresponding histograms in Fig. 6.3.

Visual inspection of patches from both the small cluster and from the lower right part of the big cluster in Fig. 6.2 confirmed the tendency we saw in the average Gaussian filter response histograms in Fig. 6.3. The patches in the small cluster were generally affected by emphysema whereas the patches in the lower part of the big cluster showed no or only faint signs of emphysema.

It is important to note that the dataset mainly contains mild to moderate COPD patients, and no patients with very severe emphysema. We expect that if this was the case, the concept or concepts would be more pronounced.

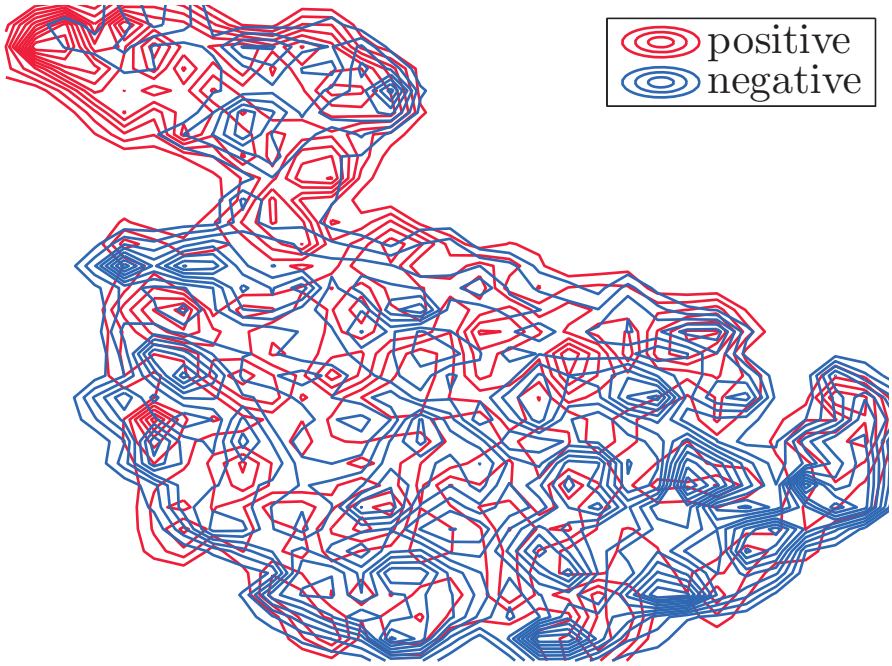


Figure 6.2: Density contours of t-SNE projection of instances of $\mathcal{X}_{sub}$

6.4.3 Interpretability

Next to the classifier performances, it is important to consider how these classifiers would be used in a medical setting. Despite slightly lower performances, instance-based methods are of interest because of their ability to provide instance labels for the ROIs. An expert could then inspect the instance labels in different regions of the lungs, allowing for better diagnosis or treatment planning. The instance labels, however, should be used with caution. Specialized MIL (i.e., except SimpleMIL) methods are trained to classify bags correctly, not instances, and the best bag classifier is not necessarily the best instance classifier [173]. Therefore, correct instance labels would be sacrificed for the greater good of correct bag labels.

Although bag-based methods perform better, their interpretability may be more difficult. For example, the average histograms (as in mean-inst) separate the classes very well, but this method can not provide information on how the affected tissue is distributed within the lungs, which could be important for determining the best treatment as well as for monitoring disease progression and therapy effect.

Dissimilarity-based methods provide more opportunities in terms of interpretability compared to mean-inst or extremes. For these methods, we can investigate which prototypes, i.e. CT images, patch clusters or individual patches, correspond to typical healthy or COPD cases. By using linear classifiers in the dissimilarity space, the diagnosis would

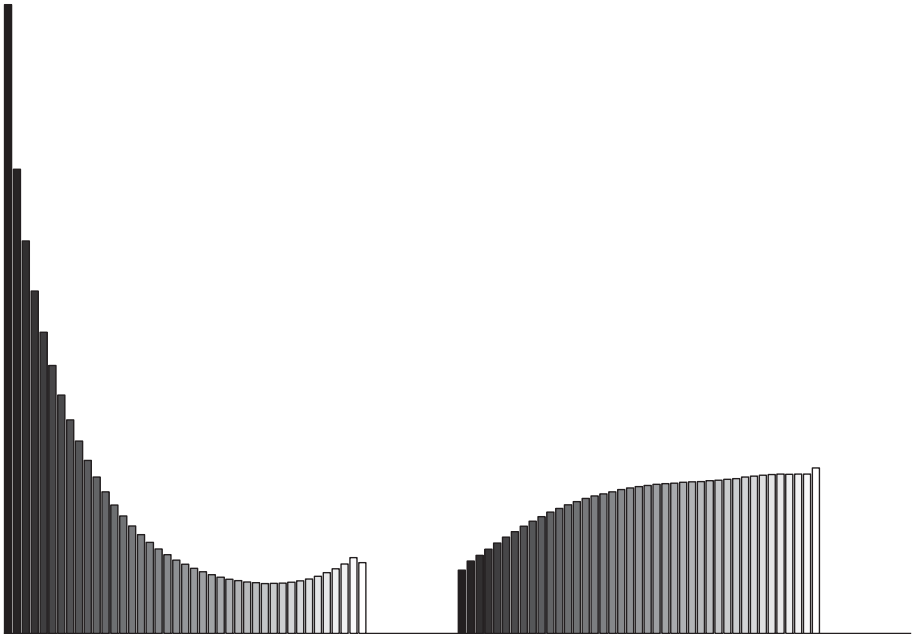


Figure 6.3: Histograms of Gaussian filter responses at the scale 0.6, for the averaged instances in the two clusters found in the t-SNE plot.

be explained in terms of a linear combination of dissimilarities to such prototypes.

6.5 Conclusions

We have studied the possibility of classifying COPD by means of various classical and more recent MIL approaches. The study revealed that MIL offers classification methods for this problem that are potentially better than the techniques previously proposed. The diversity of methods also enabled us to reason about the nature of COPD as a MIL problem. Although we found a concept region with patches showing typical disease patterns, considering the whole distribution of instances for bag classification improved the results. The best performing method is an SVM with a kernel based on the average instance per bag. This method obtains an AUC of 0.742 which is higher (but not significantly) than the previous best performance of 0.713 on the same dataset. By using the full training data we achieve a significantly higher AUC of 0.776.

Table 6.1: AUC performances ($\times 100$) of MIL classifiers, trained on $\mathcal{X}_{sub}$ (top) and $\mathcal{X}_{tr}$ (bottom). From left to right: best parameters on $\mathcal{X}_{val}$, same parameters on $\mathcal{X}_{te}$, mean(std) when subsampling $\mathcal{X}_{sub}$ or $\mathcal{X}_{tr}$ to 50% 10 times

	Trained on $\mathcal{X}_{sub}$		
Classifier	AUC $\mathcal{X}_{val}$	AUC $\mathcal{X}_{te}$	10x AUC $\mathcal{X}_{te}$
Simple logistic noisy	50.0	50.0	50.2 (0.7)
Simple logistic avg	**71.9**	**70.5**	67.9 (1.3)
Simple *k*-NN noisy	61.0	65.9	63.7 (2.3)
Simple *k*-NN avg	67.0	67.8	66.0 (1.5)
miSVM noisy	**69.7**	65.4	62.0 (3.1)
miSVM avg	**74.5**	**71.7**	69.4 (1.5)
MILBoost	55.8	61.4	59.3 (10.2)
Citation *k*-NN	65.2	61.5	63.5 (1.5)
mean-inst SVM	**74.0**	**74.2**	**72.3 (2.7)**
extremes SVM	**70.8**	68.6	68.3 (2.7)
BoW SVM	50.0	50.0	50.0 (0.0)
MILES	65.8	68.2	64.3 (4.2)
meanmin SVM	70.8	**71.3**	69.6 (2.1)
meanmin *k*-NN	65.0	**69.1**	65.7 (1.6)
emd SVM	**73.7**	**74.6**	69.3 (3.3)
emd *k*-NN	65.1	67.1	64.6 (1.8)

	Trained on $\mathcal{X}_{tr}$		
Classifier	AUC $\mathcal{X}_{val}$	AUC $\mathcal{X}_{te}$	10x AUC $\mathcal{X}_{te}$
Simple logistic noisy	60.9	60.7	50.0 (0.0)
Simple logistic avg	**73.5**	**75.8**	72.3 (3.1)
Simple *k*-NN noisy	64.3	68.2	66.9 (2.1)
Simple *k*-NN avg	66.8	69.7	68.5 (0.8)
MILBoost	54.6	54.3	62.3 (7.8)
Citation *k*-NN	65.9	56.9	60.4 (2.4)
mean-inst SVM	**77.2**	**77.6**	**76.5 (3.8)**
extremes SVM	**73.1**	65.2	67.2 (1.2)
BoW SVM	50.0	50.0	50.0 (0.0)
MILES	50.0	50.0	67.6 (2.5)
meanmin SVM	**74.0**	**75.4**	73.8 (2.6)
meanmin *k*-NN	59.0	53.5	53.5 (4.6)
emd SVM	**74.2**	72.9	**75.1 (2.7)**
emd *k*-NN	63.9	54.4	51.2 (4.3)

7

Bridging Structure and Feature Representations in Graph Matching

Structures and features are opposite approaches in building representations for object recognition. Bridging the two is an essential problem in pattern recognition as the two opposite types of information are fundamentally different. As dissimilarities can be computed for both the dissimilarity representation can be used to combine the two. Attributed graphs contain structural as well as feature based information. Neglecting the attributes yields a pure structural description. Isolating the features and neglecting the structure represents objects by a bag of features. In this paper we will show that weighted combinations of dissimilarities may perform better than these two extremes, indicating that these two types of information are essentially different and strengthen each other. In addition we present two more advanced integrations than weighted combining and show that these may improve the classification performances even further.

This chapter is published as:
Wan-Jui Lee, Veronika Cheplygina, David M. J. Tax, Marco Loog, and Robert P. W. Duin. Bridging structure and feature representations in graph matching. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(05), 2012.

7.1 Introduction

Most techniques from statistics, machine learning and pattern recognition that have been developed, such as dimension reduction, clustering, classification and regression tasks, have been applied to numerical data. These techniques have been applied with the assumption that data can be represented in a vector space where each feature forms a dimension of the space. However, for structural data, a data object includes not only the numerical feature values but also the inter-relationships between features. Structural information is essential in areas such as bioinformatics and social network analysis. Here, objects such as chemical compounds or network structures include not only numerical feature values, but also inter-relationships that can be modeled by graphs. Traditional machine learning and pattern recognition techniques for statistical data, however, cannot be applied directly to structural data because graph data cannot be embedded into a vector space in a straightforward manner.

How to utilize these two sources of information at the same time is a crucial issue. A naive way of thinking is to assume these sources to be independent, and apply statistical pattern recognition approaches to only features and structural pattern recognition approaches to only structures to obtain two (dis)similarity measures. These two results can then be combined with mathematical operations such as sum or product to derive a joint (dis)similarity measure.

However, structure might arise from inter-relationships of features. Therefore, it might be more informative to consider these two sources at the same time for comparing objects. More importantly, if two objects have very similar features, it is not possible to distinguish them, but if these objects have different structures, then the probability of separating them is much higher, and vice versa.

In structural pattern recognition [9, 25, 28, 43, 88], one of the most fundamental issues is the lack of techniques to embed the structural data objects into the same vector space. Currently, most problems can only be solved by deriving pairwise dissimilarities between such objects and deciding on the class of a new, unseen structural object based on the class of its nearest neighbor [44]. Most techniques from statistics, machine learning and pattern recognition [48, 54, 78, 85, 179] can analyze data sets that can typically be represented in a fixed-dimensional numerical vector space by means of so-called feature vectors. Unfortunately, this means that an essential part of these versatile and powerful techniques cannot be applied directly to the non-vectorial, structural data. In consequence, being able to turn structural data into a proper vectorial representation, the principal machine learning and pattern recognition tools could be utilized to the benefit of structural pattern recognition problems. Currently, in real-world applications, structural data is often simplified into numerical data by neglecting the structures and discarding inconsistent information. Methods such as graph edit distance or graph kernels use both structure and feature representations, but usually put more emphasis on

structures. One can also combine these two sources of information by assuming their independence and have an average sum of two individual distance measurements. To utilize both feature and structure representations in a flexible and smooth setting, we propose the modified graph edit distance and the modified shortest path graph kernel.

Graph edit distance [27, 148, 150, 151, 153, 154] is one of the most simple and intuitive approaches to compute the distances between graphs. The idea is to find the minimal cost for transforming one graph to the other. To define the cost for the graph edit distance, one has to set weights for nodes and edges. However, there is no adjusting between feature and structure representations. To have a better understanding of the utilization of both types of information, in this work we propose the modified graph edit distance which includes an extra parameter which controls the impact of these two different sources of information. As a result, the modified graph distance can compute dissimilarity matrices for using only pure structures, the combination of both structures and feature representations and pure feature representations.

Graph kernels [70, 72, 81] often express the similarity of two graphs by summing over the similarities between all subparts, such as walks, trees or paths, between these graphs. Similar graphs should have similar subparts and therefore a high similarity value, while different graphs should have less subparts that match. For instance, the shortest path kernel [14] sums the similarities between shortest paths between any two connected nodes in the graphs. This similarity is measured using both the nodes and edges of the paths involved. Here again, there is no way to tune how much impact these sources of information have on the similarity of the path. Here we propose a modified shortest path kernel where an extra parameter controls this impact.

The rest of the paper is organized as follows. In Section 7.2, an overview of multiple instance learning, graph edit distance and graph kernels is given. A small toy example is given in Section 7.3 to show the necessity of utilizing both structure and feature representations. The modified procedures of graph edit distance and shortest path kernel which include an extra parameter to control the impact of structure and feature information are described in Section 7.4. Simulation results are presented in Section 7.5. Finally, conclusions and discussions are given in Section 7.6.

7.2 Related Work

Graphs often have different numbers of nodes, and therefore are difficult to represent by a fixed-size vector. Hence, it is very challenging to handle even only the features in graphs with classical statistical pattern recognition techniques. By considering only the features on the nodes of a graph, each graph becomes a set of feature points, and the size of this set varies. Multiple Instance Learning (MIL) [5, 36, 49, 114, 168, 186] is one of the most well-known approaches for finding classifiers for bags of instances, where the

number of instances in each bag can be different. Therefore, it is suitable for computing distances between graphs if only features are considered.

There are also many techniques for computing distances between graphs, and they can be roughly divided into three groups: graph spectra, graph edit distance and graph kernels. The most well-known graph spectral are the sets of eigenvectors and eigenvalues derived by performing an eigendecomposition of the adjacency matrix or a derived representation of a graph. There are different variants of such spectra [29, 106, 140, 189], which are also much used in embedding and comparing graphs. All these spectral methods, however, suffer from an elementary shortcoming as they are only capable of embedding graphs with no features on the nodes. Graph edit distance and graph kernels on the other hand can use both structure and feature representations, and therefore we focus on developing modified procedures for these two approaches to include a mechanism for adjusting the impact between the feature and structure representations in this work.

In the following, we give brief introductions on MIL, graph edit distance and graph kernels.

7.2.1 Multiple Instance Learning

In Multiple Instance Learning it is assumed that an object is represented by a structureless collection of feature vectors, or, in MIL terminology, a bag of instances.[49] Objects are assumed to come from a positive or negative class, and typically it is assumed that objects from the positive class contain at least one instance from a so-called *concept*. The task of a classifier is then to identify if one of the instances belong to the concept, and label the object then to the positive class. Many MIL algorithms therefore contain an optimization strategy to search for the most informative instance per bag, and create a model of the concept. The original model proposed by [49] was an axis-parallel rectangle that was grown and shrunk to best cover the area of the concept. It is applied to a drug discovery problem where molecules have to be distinguished based on their shape into active and inactive molecules. It appears that this rectangular model fits well with the molecule shape classification, but it is less successful in other applications.

A probabilistic description of the MIL problem was given by [114]. The concept is modeled by a general probabilistic model (typically an axis-parallel Gaussian is used). Unfortunately, the optimization of the parameters requires a computationally expensive maximization of an likelihood that is adapted to include the constraint that at least one of the instances in a positive bag has a high concept probability. Newer methods often avoid the modeling of the concept by a density model, and try to separate concept instances from background instances using a discriminative approach [4, 183].

In time, more and more classification problems are identified as MIL problems, but the assumption of the presence of a single concept often does not hold. For many applica-

tions the overall distribution of instance is actually informative, and therefore training on all instances is feasible, or the application of general bag similarities [71].

7.2.2 Graph Definitions

A graph is a set of nodes connected by edges in its most general form. Consider the graph $G = (V, E, \mu, \nu)$ with

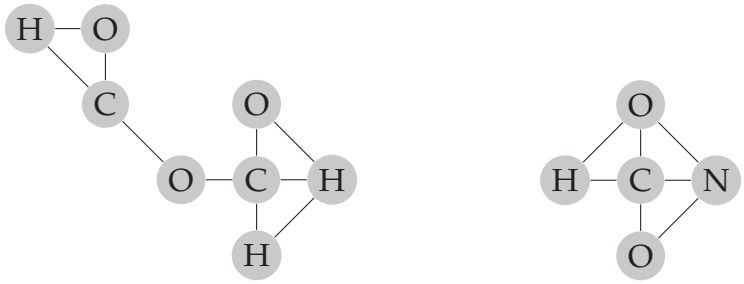
- the node set $V = \{v_1, v_2, \ldots, v_n\}$
- the edge set $E = \{e_1, e_2, \ldots, e_m\} \subset V \times V$
- the node labeling function $\mu : V \to L$
- the edge labeling function $\nu : E \to L$,

where n is the total number of nodes in the graph and is usually called the graph size, and m is the total number of edges in the graph. Edges, $e_1, e_2, \ldots, e_m$, are given by pairs of nodes $(v_i, v_j) \in E$, where v_i denotes the source node and v_j is the target node of a directed edge for directed graphs. By adding a reverse edge (v_j, v_i) for each edge, the undirected graphs can also be easily modeled. The node and edge labeling functions can assign either a set of integers $\in R^n$, or a set of symbolic labels $(\ell = a, b, c, \ldots)$. Furthermore, by assigning all the nodes the same label, one can obtain the unlabeled graphs. Similarly, by assigning all the edges the same label, one can derive unweighted graphs.

In a graph, a walk of length l is defined as a sequence of nodes $(v_1, v_2, \ldots, v_{l+1})$ where $(v_i, v_{i+1}) \in E, 1 \leq i \leq l$. A path is a walk $(v_1, v_2, \ldots, v_{l+1})$ such that $v_i \neq v_j \leftrightarrow i \neq j$.

7.2.3 Graph Edit Distance

A common way to define the dissimilarity of two graphs is to measure the minimal distortion that is needed for transforming one graph into the other. Graph edit distance is one of the most flexible ways for measuring dissimilarity between pairs of graphs. A standard set of edit distance operations to define distortion includes insertion, deletion and substitution of both nodes and edges. The substitution of two nodes v_i and v_j is denoted by $(v_i \to v_j)$, the deletion of node v_i is by $(v_i \to \phi)$, and the insertion of node v_j is by $(\phi \to v_j)$. Similar notations are also applied to edges. For each edit operation, a cost is given for measuring the strength of the corresponding operation and is to define whether an edit operation represents a strong modification of the graph. Given two graphs, the source graph G_1 and the target graph G_2 , the idea is to delete some nodes and edges from G_1, relabel some of the remaining nodes and edges and probably insert some nodes and edges, such that G_1 is completely transformed into G_2. The dissimilarity between the two is given by the cost of a sequence of edit operation



(a) (b)

Figure 7.1: Examples of two graphs

(so called edit path) that transforms G_1 into G_2. For a pair of graphs G_1 and G_2, there could be a number of different edit paths transforming G_1 into G_2, and the minimum cost edit path between two graphs is the edit distance of two graphs.

To find the optimal graph edit distance, the A^* searching tree [24, 77] is often used. However, the computational complexity for such algorithm is exponential in the number of nodes of the involved graphs. To cope better with the problem, in [148] a suboptimal method is proposed. The idea is to decompose graphs into sets of subgraphs consist of only one node and its connecting edges. The graph matching problem is then reduced to the problem of finding the optimal match between the sets of subgraphs by a bipartite matching procedure. This method aims for the suboptimal solutions and therefore generally returns an approximate graph edit distance.

The process of graph matching can be seen as an assignment problem by assigning the nodes of graph G_1 to the nodes of graph G_2, such that the overall edit costs are minimal. One of the most commonly used methods for solving the assignment problem is Munkres' algorithm [122]. Despite the fact that the nodes are assigned in a way that minimizes the cost, the edit path found by this method remains suboptimal. This is because the node assignments automatically lead to certain edge assignments, which may not be optimal. The costs of these edge assignments are added at the end of the algorithm, potentially resulting in a higher cost than the optimal edit path.

Fig. 7.1(a) and Fig. 7.1(b) are two example of graphs G_1 and G_2. In order to compute the distance between G_1 and G_2, these two graphs are decomposed node by node into local structures as those in the right and left side of Fig. 7.2. Each graph is represented by a set of nodes with their connecting edges. Then the problem becomes finding the best assignment between these two sets of nodes with their connecting edges.

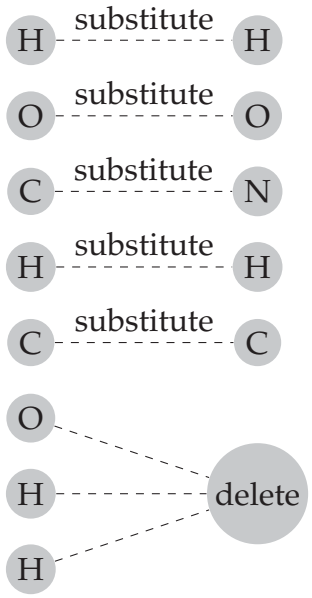


Figure 7.2: Example of the suboptimal graph edit distance approach.

7.2.4 Graph Kernels

A common approach to define kernels on graphs is to decompose the graphs into sets of substructures and count the number of matching substructures, i.e. the number of substructures in the intersection of the two sets. This is also called the intersection kernel [81]. Let X_1 and X_2 be respectively the sets of substructures of graphs G_1 and G_2. The intersection kernel is then defined as:

$$K_{\cap}(G_1, G_2) = |X_1 \cap X_2|. \tag{7.1}$$

This is equivalent to [81]:

$$K(G_1, G_2) = \sum_{x_i \in X_1, x_j \in X_2} k_{\delta}(x_i, x_j) \tag{7.2}$$

where $k_{\delta}(x_i, x_j) = 1$ if $x_i == x_j$ and 0 otherwise.

Different substructures have been used to define such kernels, such as walks [72, 91], subtrees [145] or cycles [81]. For instance, the random walk kernel counts the number of matching walks in two graphs. The match k_{walk} is originally defined to reflect an exact match between node labels. In [15], this definition is extended to:

$$k_{walk}((v_1, v_2, \ldots, v_l), (w_1, w_2, \ldots, w_l)) = \prod_{i=1,\ldots,l-1} k_{step}((v_i, v_{i+1}), (w_i, w_{i+1})) \tag{7.3}$$

where

$$k_{step}((v_i, v_j), (w_i, w_j)) = k_{node}(v_i, w_i) \cdot k_{node}(v_j, w_j) \cdot k_{edge}((v_i, v_j), (w_i, w_j)) \tag{7.4}$$

and where k_{node} and k_{edge} are kernels on node and edge labels (for instance, Radial Basis Function kernels).

For random walks, Eq. (7.2) can not be used directly because it is not possible to enumerate all the walks in a graph, as going back and forth between nodes ("tottering") is permitted. However, an approach to find the $K(G_1, G_2)$ without explicitly doing the calculation is proposed in [72]. Unfortunately, this approach requires the inversion of a $|V_1| \cdot |V_2| \times |V_1| \cdot |V_2|$ matrix, which leads to large time and memory requirements, especially for large graphs.

Next to being computationally expensive, the random walk and other similar kernels often have limited expressiveness, as pointed or walks which totter inflate the similarity value of two graphs, so that even two (globally) very different graphs can have a high similarity. Using paths instead of walks can help the tottering problem, however, the computation of this kernel is NP-hard.

Therefore in in [14] a kernel based on only shortest paths, which can be computed in polynomial time, is proposed. In this kernel, a graph G_1 is first transformed into a shortest path graph S_1, such that if there is a path of length l between two nodes in G_1, there is an edge with label l between these nodes in S_1. Because any walk $(v_1, v_2, \ldots, v_{l+1})$ in G_1 is reduced to a shortest path representation (v_1, v_{l+1}, l) in S_1, the kernel can be computed as the random walk kernel on walks of length 1:

$$K_{shortpath}(G_1, G_2) = \sum_{(v_i,v_j)\in E_1, (w_i,w_j)\in E_2} k_{path}((v_i, v_j), (w_i, w_j)) \tag{7.5}$$

where the kernel on two paths is defined as:

$$k_{path}((v_i, v_j), (w_i, w_j)) = k_{node}(v_i, w_i) \cdot k_{node}(v_j, w_j) \cdot k_{edge}((v_i, v_j), (w_i, w_j)) \tag{7.6}$$

7.3 A Toy Example

In this section, we use a toy example to illustrate the limitations of using only features or only structures.

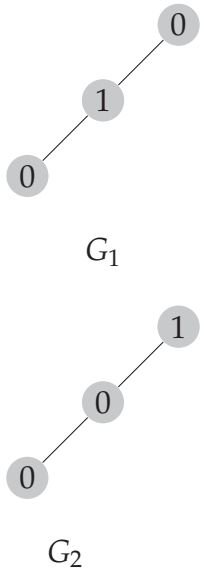


Figure 7.3: Toy example

Consider the example in Fig. 7.3. Looking only at the features, there are two bags of instances {0,1,0} and {0,0,1} which have a distance of 0 with many common MIL measurements. On the other hand, the structure of G_1 and G_2 is also identical. Simply averaging between distances on the feature representations and on the structure representations will not be able to highlight the difference between the graphs, therefore an approach that takes both types of information into account is necessary.

7.4 Modified Procedures

We modify the graph edit distance in Section 7.4.1 and the shortest path kernel in Section 7.4.2 to include an extra parameter for controlling the impact of structure and feature representations.

7.4.1 Modified Graph Edit Distance

Here we extend the idea proposed in [148] which is introduced in Section 7.2.3 by separating the cost matrix C into two parts: C_F for features and C_S for structure.

The cost matrix C becomes the weighted sum of these two matrices C_F and C_S in the form of $C = \alpha C_S + (1 - \alpha)C_F$, where α is an user-defined parameter. It is more general than the original graph edit distance method in the sense that the feature and structure

representations are scaled by the parameter α which decides how much feature and structure representations should be used in the distance measurement.

The cost matrices for structure C_S and feature C_F are by definition quadratic and are defined as

$$C_S = \left(\begin{array}{cccc|cccc} s_{1,1} & s_{1,2} & \cdots & s_{1,m} & s_{1,\phi} & \infty & \cdots & \infty \\ s_{2,1} & s_{2,2} & \cdots & s_{2,m} & \infty & s_{2,\phi} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \ddots & \infty \\ s_{n,1} & s_{n,2} & \cdots & s_{n,m} & \infty & \cdots & \infty & s_{n,\phi} \\ \hline s_{\phi,1} & \infty & \cdots & \infty & 0 & 0 & \cdots & 0 \\ \infty & s_{\phi,2} & \ddots & \vdots & 0 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \infty & \vdots & \ddots & \ddots & 0 \\ \infty & \infty & \cdots & s_{\phi,m} & 0 & \cdots & 0 & 0 \end{array}\right)_{n+m,n+m} \tag{7.7}$$

and

$$C_F = \left(\begin{array}{cccc|cccc} f_{1,1} & f_{1,2} & \cdots & f_{1,m} & f_{1,\phi} & \infty & \cdots & \infty \\ f_{2,1} & f_{2,2} & \cdots & f_{2,m} & \infty & f_{2,\phi} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \ddots & \infty \\ f_{n,1} & f_{n,2} & \cdots & f_{n,m} & \infty & \cdots & \infty & f_{n,\phi} \\ \hline f_{\phi,1} & \infty & \cdots & \infty & 0 & 0 & \cdots & 0 \\ \infty & f_{\phi,2} & \ddots & \vdots & 0 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \infty & \vdots & \ddots & \ddots & 0 \\ \infty & \infty & \cdots & f_{\phi,m} & 0 & \cdots & 0 & 0 \end{array}\right)_{n+m,n+m} \tag{7.8}$$

where $s_{i,j}$ and $f_{i,j}$ denote the feature and structure costs of a node substitution, $s_{i,\phi}$ and $f_{i,\phi}$ denote the feature and structure costs of a node deletion $(v_i \rightarrow \phi)$, and $s_{\phi,j}$ and $f_{\phi,j}$ denote the feature and structure costs of a node insertion $(\phi \rightarrow v_j)$. The costs for the insertions and deletions are found on the diagonals of the top right and left bottom quadrants of each matrix. A node can be inserted or deleted at most once, therefore the non-diagonal elements in these quadrants are set to ∞. The bottom right quadrant of the matrices is set to zero because substitutions of the form $(\phi \rightarrow \phi)$ should not cause any cost.

To keep the simplicity of the approach, $f_{i,j}$ is defined as the feature difference of node i and j, that is, the distance of two feature vectors from node i and j. If the graphs are with pure structure which means there are no feature vectors on the nodes, $f_{i,j}$ is set to 0. $f_{i,\phi}$ and $f_{\phi,j}$ are the distances of the feature vector to the origin.

Similarly, $s_{i,j}$ is the difference of numbers of edges between node i and j. If the graphs are with pure features which means there are no edges on the nodes, $s_{i,j}$ is set to 0. $s_{i,\phi}$ and $s_{\phi,j}$ are the number of edges of node i and node j, respectively.

By restricting the scaling parameter α between 0 and 1, the new cost matrix C can be computed as $\alpha C_S + (1-\alpha)C_F$. Based on the new cost matrix C, Munkres' algorithm can be executed to find the best assignment between the nodes of two graphs. Consequently, each node of graph G_1 is either uniquely assigned to a node of G_2, or to the deletion node ϕ. Vice versa, each node of graph G_2 is either uniquely assigned to a node of G_1, or to the insertion node ϕ. The ϕ nodes in G_1 and G_2 corresponding to rows $n+1, \cdots, n+m$ and columns $m+1, \cdots, m+n$ that are not used cancel each other out without any costs.

After finding the optimal node assignment between a pair of graphs, the implied edge costs need to be added to the cost computed with the Munkres' algorithm to derive the real graph edit cost.

7.4.2 Modified Graph Kernel

We use the shortest path kernel from [14] as the starting point. The kernel on two paths is defined as in Eq. (7.6). For simplicity of notation, let $k_{path} = k_{start} \cdot k_{end} \cdot k_{edge}$. These kernels may themselves also be a product of different kernels. In the original problem domain of protein prediction, multiple types of labels for nodes and edges are present. The kernel on nodes is then a product of kernels on the different labels of these nodes. We restrict ourselves to a case where nodes and edges only have a numeric label, and use a simple choice for both k_{node} and k_{edge}, namely the "bridge kernel". This kernel converts a distance between two vectors to a similarity value between 0 and a predefined threshold c, as follows:

$$k(x_i, x_j) = \max(0, c - \|x_i - x_j\|) \tag{7.9}$$

where $\|\|$ is the L_2 norm.

To be able to vary the amount of influence that the features and structure have on the shortest path kernel, we want to introduce a parameter $\alpha \in [0,1]$ in such a way that for $\alpha = 1$, k_{path} is only influenced by the node kernels, and for $\alpha = 0$ it is only influenced by the edge kernel. Because in the original formulation, k_{path} is a product and not a sum, we cannot apply α in a straightforward, weighted averaging way. A possible way to let α influence the product is:

$$k_{path} = (k_{start} \cdot k_{end})^{\alpha} \cdot (k_{edge})^{1-\alpha}. \tag{7.10}$$

Note that this approach does not guarantee that the kernel is positive definite, however, the information contained in such kernels may still be of interest [137].

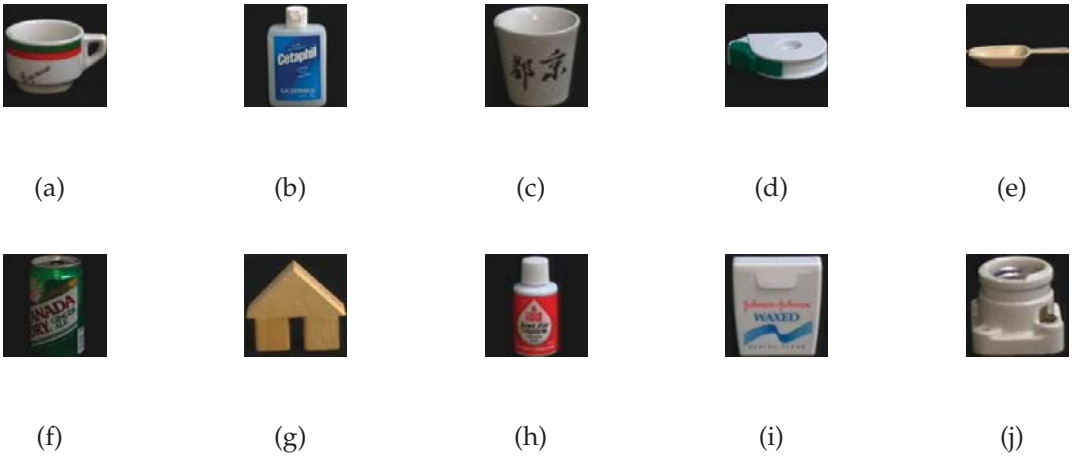


Figure 7.4: An example image of class (a) 10; (b) 13; (c) 18; (d) 40; (e) 44; (f) 49; (g) 51; (h) 61; (i) 65; and (j) 66, respectively.

7.5 Experiments

In this section, we compare the performance of the modified procedures with the weighted average of structure and feature representations using the 1-nearest neighbor classifier [44] in the original space (1nnc), the 1-nearest neighbor in the dissimilarity space (1nndc) and support vector machine (svc) [179] in the dissimilarity space [26, 132, 133, 149–151]. All the classifiers are built with PRTools [56].

Two real-world datasets, i.e., COIL-100 dataset [125] and Mutagenicity [152], are used in the experiments. The COIL-100 dataset consists of images of 100 different objects. Images of each objects are taken at intervals of 5 degrees, resulting in 72 images per object or 72 images per class. However, in our experiments, we do not use all available images. Only 10 classes (10, 13, 18, 40, 44, 49, 51, 61, 65, 66) are used. These 10 classes of images are selected because they have the lowest one-against-all performance, i.e. are relatively difficult to classify compared to the other classes. Fig. 7.4 shows an example image of each object from these 10 classes.

Three different datasets, i.e. coil-segment, coil-harris and coil-sift, of graphs are then generated from these 720 images. This is done as follows:

The coil-segment dataset is obtained by first converting the color images to grayscale, applying a mean-shift algorithm to get homogeneous graylevel segments, and finally removing image segments that were smaller than 50 pixels. A structure graph was obtained by connecting the segments in the graph that also have some neighboring

The original color features are very informative, such that the classification performance was already so high that the structure could not add any additional information.

pixels in the image domain. Overall, each image has around 3-10 segments.

The coil-harris dataset is directly obtained from the coil-del dataset in the IAM graph database [152]. Briefly, Harris corner detection [76] is used to extract corner features from images. Based on these corners, a Delaunay triangulation is applied. The triangulation is converted into a graph such that each line becomes an undirected, unlabeled edge, and each endpoint becomes a node described by its 2D coordinates.

The coil-sift dataset detects the sift keypoints [112] instead of Harris points. Sift points are corner points with 128 dimensional descriptors, which are used as the feature vectors for graph nodes. The coil-sift uses the same procedure for transforming images into graphs as the coil-harris.

The difference among the coil-segment, coil-harris and coil-sift datasets is that the coil-segment has much simpler structure compared to coil-harris and coil-sift because usually only few segments can be extracted from an image while many corner points or keypoints can be detected. The coil-sift dataset has more complex features (128 in total) while coil-harris and coil-segment only have 2 and 4 dimensions, respectively.

Another real-world dataset Mutagenicity [152] is also used in the experiments. Mutagenicity refers to a property of chemical compounds. The molecules are converted into graphs in a straightforward manner by representing atoms as nodes and the covalent bonds as edges. Nodes are labeled with the corresponding chemical symbol. The average number of nodes of a graph is 30.3 ± 20.1, and the average number of edges is 30.7 ± 16.8. The Mutagenicity dataset is divided into two classes, i.e., mutagen and nonmutagen. There are 4,337 objects in total (2,401 mutagen and 1,936 nonmutagen). However, we only take a subset of this dataset in the experiments by using only every 5th object, resulting in a subset of 867 objects.

All datasets are represented in the dissimilarity space. First, a subset of graphs is selected as the prototype set with R objects. Next, all the dissimilarities between all graphs and the prototype set are computed. Now each graph is represented by its R dissimilarities.

In the experiments, 20% of images are randomly taken as the training dataset, and the rest 80% are taken as the testing dataset. All the experiments are repeated 100 times and the average classification error is recorded. The standard deviations of the results are very small due to 100 repetitions and are therefore disregarded.

The results of the modified graph edit distance are presented in Figs. 7.5-7.8, the results of the modified shortest path kernel are presented in Figs. 7.9-7.12. In each figure, a comparison is made between the modified method (represented by solid lines with star symbols) and the weighted average of pure structure and pure feature information (represented by solid lines with circles). On the x-axis is the parameter α, when $\alpha = 0$, only structure information is used and when $\alpha = 1$, only feature information is used. Therefore, at $\alpha = 0$, the graph edit distance and shortest path kernel are applied on unlabeled

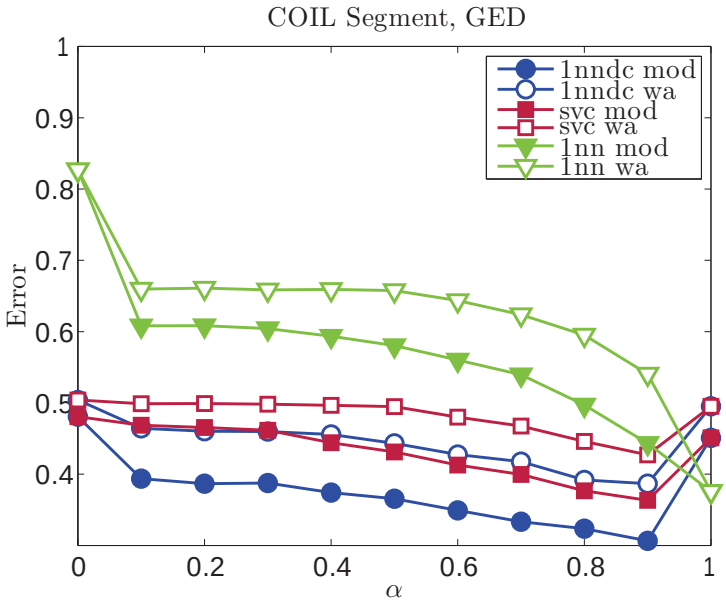


Figure 7.5: Results of coil-segment for modified graph edit distance (mod) and weighted average (wa).

graphs, producing a dissimilarity matrix D_0, whereas at $\alpha = 1$, the procedures are applied to two sets of feature vectors, resulting in dissimilarity matrix D_1. The weighted average matrix $D_W(\alpha)$ is then computed as follows:

$$D_W(\alpha) = \alpha D_1 + (1 - \alpha) D_0 \tag{7.11}$$

On the y-axis of the figures is the average classification error of three different classifiers: the nearest neighbor in the original space (denoted by 1nnc), the nearest neighbor in the dissimilarity space (denoted by 1nndc) and the support vector classifier (denoted by svc). Note that the graph edit distance procedures can only produce pairwise distances for graphs, and therefore only the nearest neighbor classifiers can be adopted in the original space. But in the dissimilarity space, the distances of one object to the other objects are taken as the feature values and therefore one can adopt any classical pattern recognition techniques.

7.5.1 Modified Graph Edit Distance

The coil-segment dataset is with simple structure and simple features, and therefore its performance of using only features or structures are both not good as in Fig. 7.5. However, if we combine these two sources of information, significant improvements can be observed both with weighted average and modified graph edit distance in the dissimilarity space, especially when $\alpha = 0.9$. This suggests that not only the combination of

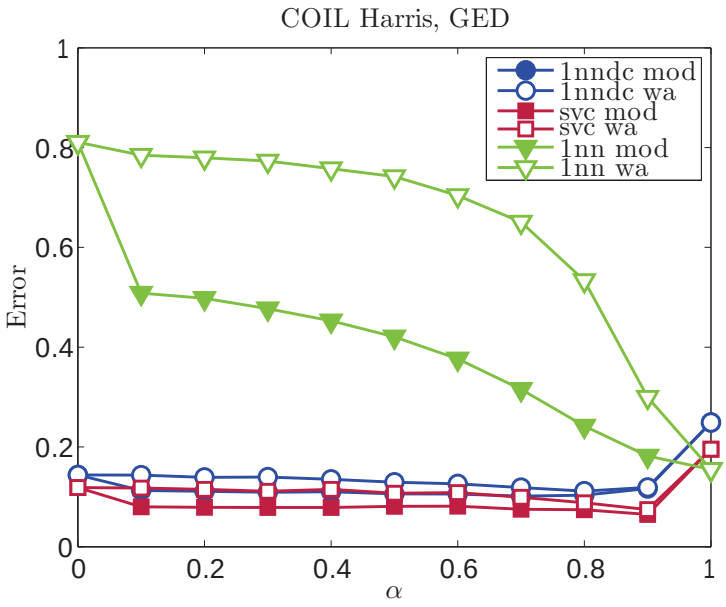


Figure 7.6: Results of coil-harris for modified graph edit distance (mod) and weighted average (wa)

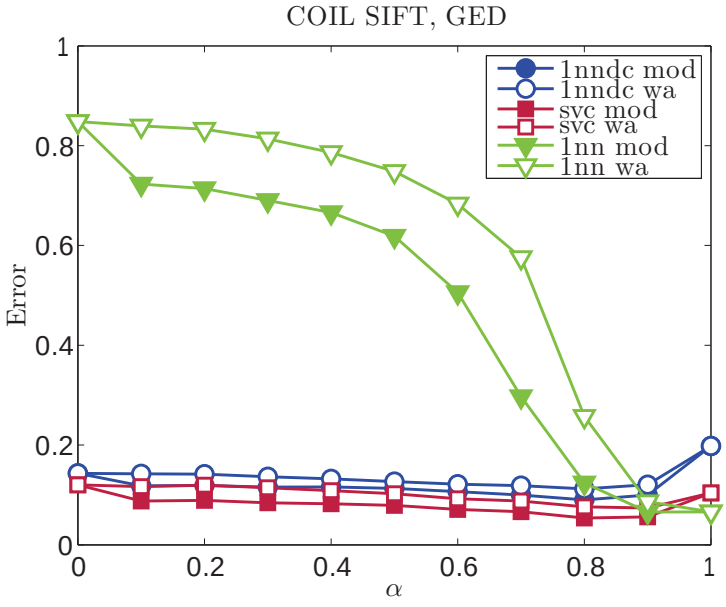


Figure 7.7: Results of coil-sift for modified graph edit distance (mod) and weighted average (wa).

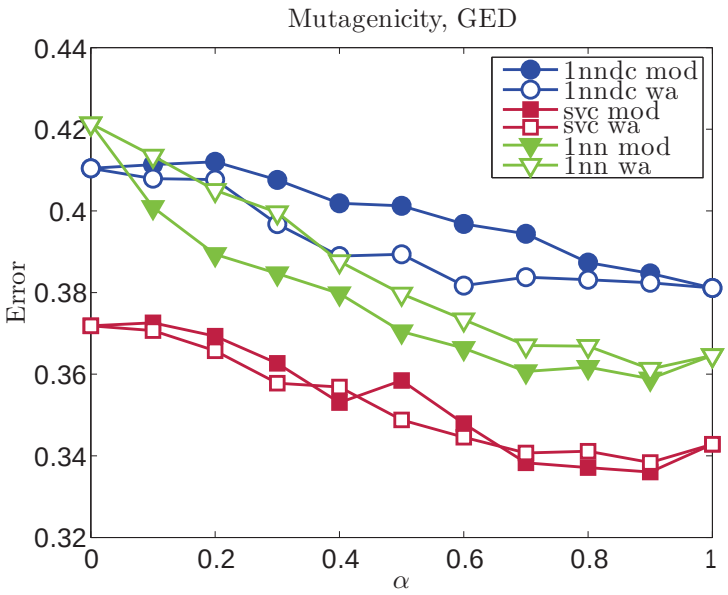


Figure 7.8: Results of Mutagenicity for modified graph edit distance (mod) and weighted average (wa).

feature and structure information can be helpful, but also changing the impacts of these two information could make a lot of difference, and it is necessary to balance these two information to obtain a better result. This phenomena can also be observed in Fig. 7.6 and Fig. 7.7, even though the structure information is doing very good already in these two datasets, and therefore the improvement of combining the two information is less significant as with the coil-segment dataset.

From Fig. 7.5, Fig. 7.6 and Fig. 7.7, we can also see that the modified graph edit distance is better than weighted average in all cases, and therefore the dependency between the feature and structure information should also be taken into account when computing the graph distances. Also, the classifiers in the dissimilarity spaces perform much better than classifiers in the original space, and svc is better than 1nnc. The dissimilarity space therefore not only allows different classifiers to be applied on structure data, but also in general gives much better results than the original space. One very intriguing results from these figures is the significant improvement of the dissimilarity space over the original space when $\alpha = 0$ in the case of considering pure structure. For the coil-harris and coil-sift datasets, the improvement is even around 70%. This suggests that there are intrinsic features within the graph structure and these features can only be utilized by the classifiers in the dissimilarity space, but not in the original space.

For the Mutagenicity dataset, the results are given in Fig. 7.8. The structures of graphs are naturally given by the molecules, and the structure is more sparse (the number of edges is much fewer) than the ones of the coil-100 dataset. One of the reasons why there is no significant improvement of modified graph edit distance over weighted average

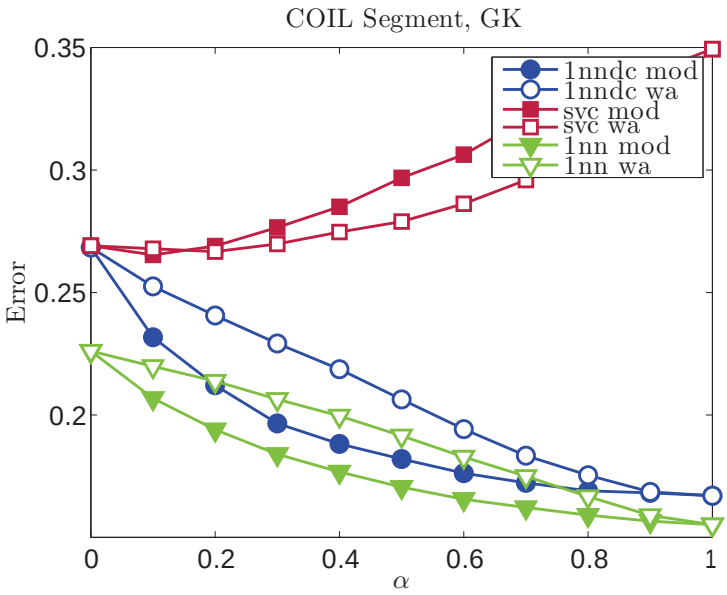


Figure 7.9: Results of coil-segment for modified shortest path kernel (mod) and weighted average (wa).

on the performance could be that the sparse structures contains less intrinsic features. The sparse structures of graphs probably also introduce less dependency between structures and features, and therefor modified graph edit distance is not always better than weighted average. Nevertheless, we can still observe that the extra parameter α helps for finding better results. Also, svc is much better than 1nnc classifiers in both original space and dissimilarity space.

7.5.2 Modified Graph Kernel

The results of the modified shortest path kernel compared to weighted averaging are presented in Figs. 7.9, 7.10, 7.11 and 7.12 for the coil-segment, coil-harris, coil-sift and mutagenicity datasets, respectively.

Before running the classifiers on the matrices produced by the modified shortest path kernel, we normalized the kernel matrix such that the diagonal is equal to 1 (by $K(x,y) = \sqrt{K(x,x) \cdot K(y,y)}$) and converted the kernel matrix to a dissimilarity matrix using a procedure in DisTools [57]. This was done in order to be able to compare 1nnc (which needs distances) to 1nndc and svc (which could also work with a similarity matrix directly).

For the coil-segment, coil-harris and coil-sift datasets, the best results are achieved when only feature information is used. This is somewhat surprising, because it is expected that combining both information sources would be beneficial. A possible explanation

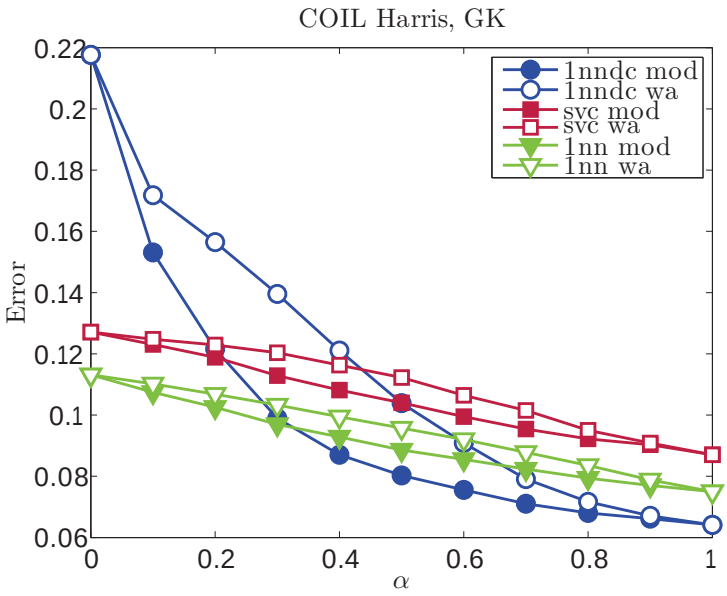


Figure 7.10: Results of coil-harris for modified shortest path kernel (mod) and weighted average (wa).

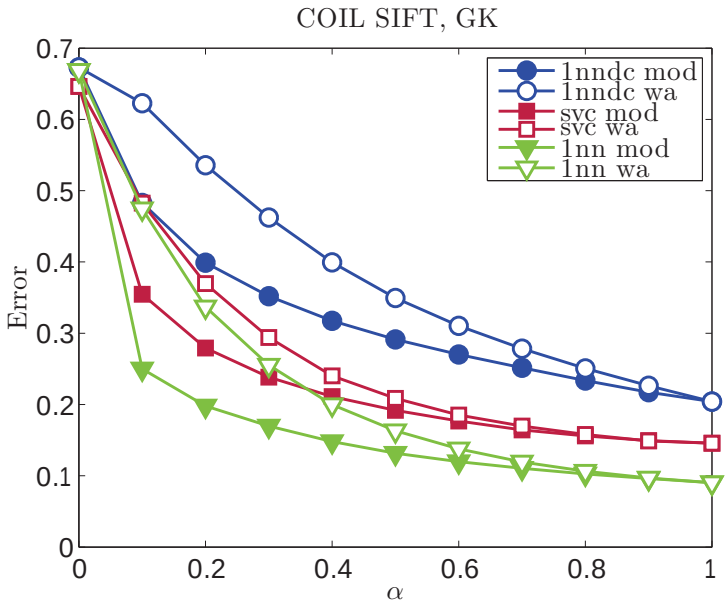


Figure 7.11: Results of coil-sift for modified shortest path kernel (mod) and weighted average (wa).

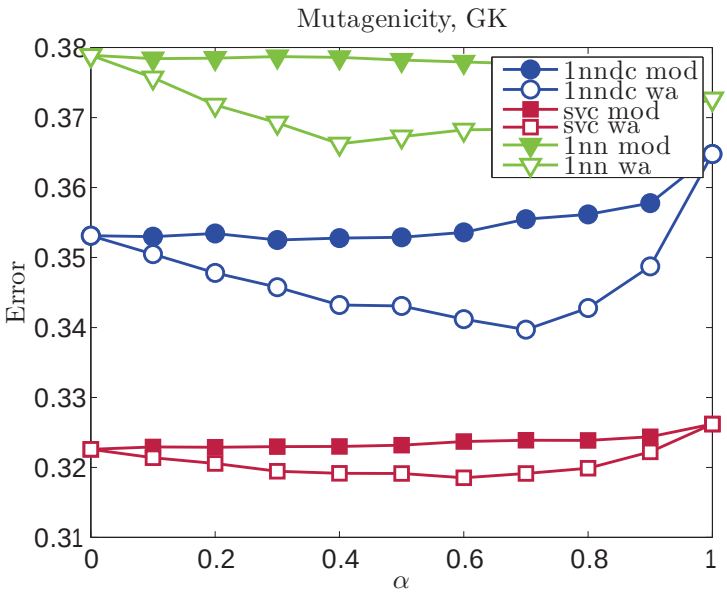


Figure 7.12: Results of Mutagenicity for modified shortest path kernel (mod) and weighted average (wa).

could lie in the thresholds used for the bridge kernel on node values. The main point of the threshold is to convert a distance to a similarity value greater or equal to 0, so in principle it should be sufficient to set the threshold to the maximum possible distance between nodes or edges, this way preserving as much information as is contained in the distances. However, in our preliminary experiments, we noticed that choosing a lower threshold, thus setting the larger distances to 0, often had a positive effect on performance. We did not investigate which thresholds are best for a particular dataset, but attempted to choose reasonable default values. For the coil-segment and coil-harris datasets, the original node distances were very large, but after normalizing the features to zero mean and unit variance, a threshold of 2 was reasonable. This threshold was also used for the coil-sift dataset. It might be the case that due to these thresholds, the performances at $\alpha = 1$ are very good, which masks the fact that it is beneficial to use structure information.

Similarly to the results of the edit distance, we see that in general, at different values of α the kernel performs better than the weighted average. This is only not the case for svc on the coil-segment dataset, where svc performs much worse than the nearest neighbor classifiers. Apart from this result, the three classifiers seem to have comparable performance. This is very different from the edit distance results, where the classifiers in the dissimilarity space are clearly outperforming 1nnc.

For the mutagenecity dataset, we see quite a different picture than in the coil datasets. As each of the classifiers performs similarly using only feature or only structure information, we see that combining the two is advantageous in this case. However, for

all three clasifiers the weighted averaging outperforms the shortest path kernel. This suggests that the way α is introduced in the modified kernel is not able to effectively combine the feature and structure similarities. This has an intuitive explanation. If we examine the inter-node distances in the dataset, we will notice that this distance is either 0 or 1, because the node features are discrete (presence or absence of chemical symbol). Therefore, applying α as we proposed does not change the node similarity of the path. As a result, the matrices produced by the kernel are very similar at different values of α, which is clearly reflected by the results. Weighted averaging does achieve different matrices, and therefore much more variable perfomances can be seen. A different usage of α in the modified kernel might therefore be advantageous.

7.6 Discussions and Conclusions

Combining different sources of information is expected to give better performance than single information sources. However, how to combine the information sources to achieve the most performance improvement is not always trivial. In this work, we propose the modified graph edit distance and the modified shortest path graph kernel to adopt an extra parameter for controlling the impact of feature and structure representations on computing graph distances. From the experimental results, we can see improvements when the impact of the feature and structure representations is properly scaled.

Another intriguing observation is that classifiers in the dissimilarity space can outperform the nearest neighbor in original space if the graph dataset has intrinsic dimensions within the structure. Even though the measurement of the intrinsic dimensions is still not clear, it is still worth further investigation on the performances of dissimilarity space with respect to the intrinsic dimensions of datasets. Another point which might be worth investigating is the way the parameter which affects feature and structure representations is applied to the shortest path kernel or graph kernels in general.

8

Discussion

We studied dissimilarity-based classification as an approach to multiple instance learning problems. We have examined several ways of defining dissimilarity functions on bags and choosing a set of reference prototypes. Through the experiments, we have seen that these choices need to differ according to the assumptions that are appropriate for different types of MIL problems. An assumption that works well in practice is that the instances in a bag have the same label. Dissimilarity-based classifiers are an effective, intuitive way to approach MIL problems. In the following, we discuss several future directions in which this work can be extended.

Type of Data. We have seen that MIL problems are heterogeneous, and that the success of a classifier depends on the way the data is distributed and therefore which assumptions are more suitable [40, 42]. Such insight is very helpful when approaching novel MIL problems. We believe that more steps can be taken in this direction.

From our experience, real-world problems often do not consist of just a single concept and are not normally distributed – the scenarios we investigated in [40]. More advanced scenarios (AND/OR combinations of instances from multiple concepts) are investigated in [2] in the context of comparing supervised and MIL classifiers. Using toy examples, it is shown that some learners are not able to perfectly classify (although the Bayes error is zero) datasets with more advanced concepts.

An open question is how to assess the similarity of a toy problem and a real-life problem. Next to results of different classifiers, we could compare low-dimensional embeddings of the data to see whether there are any clusters or outliers in the data. It would be also interesting to automate the toy problem generation process, with this yet to be defined "dataset similarity" measure as an optimization criterion. This would enable constructing synthetic datasets which have similar characteristics (such as high dimensionality and heterogeneity of classes) to real datasets, but for which the ground truth is available.

Another approach is taken in [178], where the goal is to learn a classifier which, based

on characteristics such as number of bags, or performance of simple classifiers, can decide which type of MIL classifier will be more appropriate for the problem. Datasets are taken from three groups: semi-artificial data generated from supervised problems, MIL image data and MIL text categorization data. Unfortunately, the experiments are done on each of these groups separately, which seems to defeat the purpose of meta learning. It would be interesting to investigate what these properties (other than prior knowledge about where the datasets originate) are, and include more MIL datasets in the analysis. We envision this approach to be similar to [97], where it is shown that selecting a classifier based on dataset properties as well as cross-validation performance leads to better results than selecting a classifier based on cross-validation performance alone.

One point that can be taken into consideration is how the bags and instances are generated. Is the relationship between bags and instances an "a part of" relationships (such as patches in an image or paragraphs of a document) or an "is a" relationship (such as conformations of a molecule, or multiple scales of the same image)? Are the instances sampled randomly (e.g. patches in an image) or is a more intelligent process involved (e.g. face detector)? Are the instances overlapping or not (such as paragraphs in a document, or overlapping parts of text)? All of these choices influence what assumptions can be made, and ultimately what works well in a particular problem.

Representation. This thesis has primarily considered bags as discrete sets of feature vectors, and briefly as distributions of instances. There are more opportunities to be explored in the latter representation. First of all, this could offer computational advantages for problems with a large number of bags and/or instances. Estimating a distribution per bag, and computing a dissimilarity between distribution parameters, such as Eq. (3.5), is significantly faster than computing a dissimilarity which uses all instances, such as Eq. (3.3), because the former scales linearly, and not quadratically, with the number of instances. An existing work that can directly be applied to generalized MIL problems is [120], where kernels are defined on probability distributions. The kernel values are approximated from the available data (for example, by assuming a Gaussian distribution and estimating the mean and variance), but in all cases, all the available data points have an equal influence on the kernel value. It is an open question whether the strict MIL assumption could also be used in this setting. Furthermore, it is unclear whether bag size, a measure that was shown to be informative in Chapter 4, can be taken into account in such formulations.

Considering bags as distributions could also offer ways to be more robust for situations when few bags are available for training. In [52], additional training bags are generated by shuffling instances that originate from different bags, however, this is done in an explicit manner. This has similarities to generating additional training data by adding noise to the features. Recently it has been shown that this corruption of features can be done implicitly for particular types of loss functions and noise distributions [176]. In other words, an infinite number of training objects are "generated" at just a fraction of

the cost of explicitly adding those objects to the training data. Perhaps approaches such as [52] can also benefit from such an implicit formulation.

Another research direction is representing bags as attributed graphs, as is done in Chapter 7, and in [200]. There is a fundamental difference between these representations. In [200], the graph structure is derived from the instances, i.e., instances that are close together in the feature space are connected by an edge. This essentially assigns larger weights to instances in dense areas of the feature space. It is therefore an assumption that instances in such areas are more informative, as they will have a larger effect on the similarity of bags, but no extra information is introduced into the problem. In Chapter 7, however, the graph structure is derived from a different source, such as the spatial adjacency of image segments, which introduces additional information. This is exactly the reason why the datasets in Chapter 7 are different from the MIL datasets used elsewhere in the thesis: for the MIL datasets, such additional information sources are not directly available.

Prototype Selection. In this thesis, we have primarily considered the scenario where all the training data is used as prototypes. However, prototype selection [30, 134] can be employed to reduce the dimensionality of the dissimilarity space, create a more robust classifier, and gain understanding of the informative prototypes.

One issue with prototype selection is that for MIL, we have encountered some highly non-metric, yet still informative dissimilarities. In particular, some of these do not only violate the triangle inequality, but also the identity and symmetry properties. As prototype selection often relies on the concept of distances, such non-metric dissimilarities could be problematic, and it is unclear whether prototype selection would still be worthwhile.

As an alternative to traditional prototype selection, generating prototypes could also be of interest. As we demonstrated in Chapter 4, artificially generated instances or bags can be informative prototypes. The question is whether it would be possible to avoid computing the full dissimilarity matrix, and instead infer where in the space informative prototypes would reside, or what kind of properties (such as label, bag size) these prototypes would have.

Classifier Evaluation. Although it has long been acknowledged there is no such thing as the best classifier [191], it is surprising how many papers still propose novel classifiers and claim their superiority due to slightly higher performance on a few datasets. We believe that next to absolute measures of performance [64] there should be more emphasis on other evaluating other metrics, relevant for applications where the proposed classifiers will be used.

An interesting possibility is the performance that can be achieved during a fixed amount

of computation time. Computation time is not an issue for many benchmark MIL datasets, but becomes more crucial in problems with large numbers of bags, instances and/or features. For example, the COPD dataset in Chapter 6 was already problematic for some (nave implementations) of classifiers. Another interesting, but more difficult to define performance metric, is the performance per fixed amount of time, spent by a novice user using the classifier. The goal would be to measure how intuitive a classifier is, for example, how difficult it is to set the classifier parameters for a novel problem.

For MIL classifiers in particular, it is important to realize that in many cases, the best bag classifier will not coincide with the best instance classifier. For applications where the prediction of instance labels is important, the bag performance is therefore not an objective evaluation measure. Next to more instance-labeled MIL datasets, alternative evaluation measures are also needed in this case. If both bag and instance performance are important, perhaps an interesting direction is to look at a curve for both quantities, in a similar fashion as is done in [146] for the trade-off of performance and feature selection stability.

Bibliography

[1] E. Akbas, B. Ghanem, and N. Ahuja. Mis-boost: Multiple instance selection boosting. *arXiv preprint arXiv:1109.2388*, 2011. Pages: 66

[2] E. Alpaydın, V. Cheplygina, M. Loog, and D. M. J. Tax. Single- vs. multiple-instance classification. *Pattern Recognition, in press*, 2014. Pages: 129

[3] J. Amores. Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, 201:81–105, 2013. Pages: 7, 10, 20, 85

[4] S. Andrews, T. Hofmann, and I. Tsochantaridis. Multiple instance learning with generalized support vector machines. In *National Conference on Artificial Intelligence*, pages 943–944, 2002. Pages: 62, 69, 112

[5] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems*, pages 561–568, 2002. Pages: 7, 9, 10, 16, 19, 30, 48, 77, 85, 98, 100, 111

[6] O. Arandjelovic and R. Cipolla. Face set classification using maximally probable mutual modes. In *International Conference on Pattern Recognition*, volume 1, pages 511–514. IEEE, 2006. Pages: 16

[7] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition*, pages 983–990. IEEE, 2009. Pages: 16, 21

[8] M.-F. Balcan, A. Blum, and N. Srebro. A theory of learning with similarity functions. *Machine Learning*, 72(1-2):89–112, 2008. Pages: 37

[9] M. Basu, H. Bunke, and A. D. Bimbo. Syntactic and structural pattern recognition. *Special Section of IEEE Transaction on Pattern Analysis and Machine Intelligence*, 7(27), 2005. Pages: 110

[10] A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific Symposium on Biocomputing*, pages 6–17, 2001. Pages: 88

[11] A. Bertoni, R. Folgieri, and G. Valentini. Bio-molecular cancer prediction with random subspace ensembles of support vector machines. *Neurocomputing*, 63:535–539, 2005. Pages: 68

[12] C. Bhattacharyya, L. Grate, A. Rizki, D. Radisky, F. Molina, M. I. Jordan, M. J. Bissell, and I. S. Mian. Simultaneous classification and relevant feature identification in high-dimensional spaces: application to molecular profiling data. *Signal Processing*, 83(4):729–743, 2003. Pages: 66

[13] J. Bi and J. Liang. Multiple instance learning of pulmonary embolism detection with geodesic distance along vascular structure. In *Computer Vision and Pattern Recognition*, pages 1–8, 2007. Pages: 84

[14] K. M. Borgwardt and H. P. Kriegel. Shortest-Path Kernels on Graphs. In *International Conference on Data Mining*. IEEE Computer Society, 2005. Pages: 111, 116, 119

[15] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. V. N. Vishwanathan, A. J. Smola, and H. P. Kriegel. Protein Function Prediction via Graph Kernels. *Bioinformatics*, 21:i47–i56, 2005. Pages: 115

[16] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997. Pages: 41, 71

[17] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996. Pages: 95

[18] F. Briggs, X. Z. Fern, and R. Raich. Rank-loss support instance machines for MIML instance annotation. In *International Conference on Knowledge Discovery and Data Mining*, pages 534–542. ACM, 2012. Pages: 21

[19] F. Briggs, B. Lakshminarayanan, L. Neal, X. Z. Fern, R. Raich, S. J. K. Hadley, A. S. Hadley, and M. G. Betts. Acoustic classification of multiple simultaneous bird species: A multi-instance multi-label approach. *The Journal of the Acoustical Society of America*, 131:4640, 2012. Pages: 40, 70

[20] S. D. Brossi and A. P. Bradley. A comparison of multiple instance and group based learning. In *International Conference on Digital Image Computing Techniques and Applications*, pages 1–8. IEEE, 2012. Pages: 23

[21] G. Brown and L. I. Kuncheva. Good and bad diversity in majority vote ensembles. In *Multiple Classifier Systems*, pages 124–133. Springer, 2010. Pages: 75

[22] G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005. Pages: 67

[23] M. Budka, B. Gabrys, and K. Musial. On accuracy of PDF divergence estimators and their applicability to representative data sampling. *Entropy*, 13(7):1229–1266, 2011. Pages: 35

[24] H. Bunke and G. Allermann. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1:245–253, 1983. Pages: 114

[25] H. Bunke and T. Caelli. Graph matching in pattern recognition and machine vision. *International Journal of Pattern Recognition and Artifitial Intelligence*, 18(3), 2004. Pages: 110

[26] H. Bunke and K. Riesen. Graph classification based on dissimilarity space embedding. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 996–1007. Springer, 2008. Pages: 120

[27] H. Bunke and K. Riesen. Recent advances in graph-based pattern recognition with applications in document analysis. *Pattern Recognition*, 44(5):1057–1067, 2011. Pages: 7, 111

[28] H. Bunke and A. Sanfeliu. *Syntactic and Structural Pattern Recognition: Theory and Applications*. Singapore: World Scientific, 1990. Pages: 110

[29] T. Caelli and S. Kosinov. An eigenspace projection clustering method for inexact graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:515–519, 2004. Pages: 112

[30] Y. P. Calaña, E. G. R. Reyes, M. Orozco-Alzate, and R. P. W. Duin. Prototype selection for dissimilarity representation by a genetic algorithm. In *International Conference on Pattern Recognition*, pages 177–180. IEEE, 2010. Pages: 66, 131

[31] P. M. A. Calverley. COPD: Early detection and intervention. *CHEST Journal*, 117 (5_suppl_2):365S–371S, 2000. Pages: 98

[32] V. R. Carvalho and W. W. Cohen. On the collective classification of email speech acts. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*,

pages 345–352. ACM, 2005. Pages: 18

[33] C. C. Chang and C. J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011. Pages: 48

[34] O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5): 1155–1178, 2007. Pages: 70

[35] O. Chapelle, B. Schölkopf, A. Zien, et al. *Semi-supervised learning*, volume 2. MIT press Cambridge, 2006. Pages: 21

[36] Y. Chen, J. Bi, and J. Wang. Miles: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1931–1947, 2006. Pages: 9, 10, 16, 19, 20, 28, 30, 37, 39, 48, 62, 64, 70, 79, 86, 101, 111

[37] V. Cheplygina, L. Sørensen, D. M. J. Tax, J. H. Pedersen, M. Loog, and M. de Bruijne. Classification of COPD with multiple instance learning. In *International Conference on Pattern Recognition*, pages 1508–1513, 2014. Pages: 84, 86, 87

[38] V. Cheplygina and D. M. J. Tax. Pruned random subspace method for one-class classifiers. In *Multiple Classifier Systems*, pages 96–105. Springer, 2011. Pages: 71, 77

[39] V. Cheplygina, D. M. J. Tax, and M. Loog. Class-dependent dissimilarity measures for multiple instance learning. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 602–610. Springer, 2012. Pages: 34, 55, 62, 76

[40] V. Cheplygina, D. M. J. Tax, and M. Loog. Does one rotten apple spoil the whole barrel? In *International Conference on Pattern Recognition*, pages 1156–1159, 2012. Pages: 7, 28, 54, 64, 99, 101, 129

[41] V. Cheplygina, D. M. J. Tax, and M. Loog. Combining instance information to classify bags. In *Multiple Classifier Systems*, pages 13–24. Springer, 2013. Pages: 63, 68, 71, 77

[42] V. Cheplygina, D. M. J. Tax, and M. Loog. Multiple instance learning with bag dissimilarities. *Pattern Recognition*, 48(1):264–275, 2015. Pages: 9, 86, 90, 129

[43] F. Chung. *Spectral Graph Theory*. CBMS Regional Conference Series in Mathematics, No. 92, USA: AMS, 1997. Pages: 110

[44] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967. Pages: 110, 120

[45] E. Decencière, X. Zhang, G. Cazuguel, B. Lay, B. Cochener, C. Trone, P. Gain, R. Ordonez, P. Massin, A. Erginay, B. Charton, and J.-C. Klein. Feedback on a publicly distributed image database: the Messidor database. *Image Analysis and Stereology*, pages 231–234, 2014. Pages: 90

[46] E. R. DeLong, D. M. DeLong, and D. L. Clarke-Pearson. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics*, pages 837–845, 1988. Pages: 103

[47] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006. Pages: 52

[48] P. A. Devijver and J. Kittler. *Pattern Recognition: a Statistical Approach*. London: Prentice-Hall, 1982. Pages: 110

[49] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997. Pages: 7, 8, 10, 14, 16, 18, 19, 28, 30, 39, 62, 69, 85, 90, 98, 100, 111, 112

[50] J. Ding, H. Cheng, J. Huang, J. Liu, and Y. Zhang. Breast ultrasound image classification based on multiple-instance learning. *Journal of Digital Imaging*, 25(5):620–627, 2012. Pages:

87

[51] V. C. Dinh, R. P. W. Duin, and M. Loog. A study on semi-supervised dissimilarity representation. In *International Conference on Pattern Recognition*, pages 2861–2864. IEEE, 2012. Pages: 7, 72

[52] G. Doran and S. Ray. Smile: Shuffled multiple-instance learning. In *AAAI*, 2013. Pages: 130, 131

[53] M. P. Dubuisson and A. K. Jain. A modified Hausdorff distance for object matching. In *International Conference on Pattern Recognition*, volume 1, pages 566–568, 1994. Pages: 32, 33

[54] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, 1999. Pages: 6, 110

[55] R. P. W. Duin. The combining classifier: to train or not to train? In *International Conference on Pattern Recognition*, volume 2, pages 765–770. IEEE, 2002. Pages: 71

[56] R. P. W. Duin, P. Juszczak, P. Paclík, E. Pękalska, D. de Ridder, D. M. J. Tax, and S. Verzakov. PRTools, A MATLAB toolbox for pattern recognition. online, http://www.prtools.org, 2013. Pages: 48, 79, 102, 120

[57] R. P. W. Duin and E. Pękalska. Dis-Tools, 2009. Pages: 46, 125

[58] R. P. W. Duin and E. Pękalska. The dissimilarity space: Bridging structural and statistical pattern recognition. *Pattern Recognition Letters*, 33(7):826–832, 2012. Pages: 7

[59] R. P. W. Duin and D. M. J. Tax. Experiments with classifier combining rules. In *Multiple Classifier Systems*, pages 16–29. Springer, 2000. Pages: 63

[60] M. M. Dundar, G. Fung, B. Krishnapuram, and R. B. Rao. Multiple-instance learning algorithms for computer-aided detection. *IEEE Transactions on Biomedical Engineering*, 55 (3):1015–1021, 2008. Pages: 87

[61] C. Elkan and K. Noto. Learning classifiers from only positive and unlabeled data. In *International Conference on Knowledge Discovery and Data Mining*, pages 213–220. ACM, 2008. Pages: 21

[62] A. Erdem and E. Erdem. Multiple-instance learning with instance selection via dominant sets. In *Similarity-Based Pattern Recognition*, pages 177–191. Springer, 2011. Pages: 66

[63] A. Feragen, J. Petersen, D. Grimm, A. Dirksen, J. H. Pedersen, K. Borgwardt, and M. de Bruijne. Geometric tree kernels: Classification of COPD from airway tree geometry. In *Information Processing in Medical Imaging*, pages 171–183, 2013. Pages: 99

[64] C. Ferri, J. Hernández-Orallo, and R. Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27–38, 2009. Pages: 131

[65] J. Foulds and E. Frank. A review of multi-instance learning assumptions. *Knowledge Engineering Review*, 25(1):1, 2010. Pages: 7, 19, 20, 28, 29, 62, 63

[66] G. Fu, X. Nan, H. Liu, R. Patel, P. Daga, Y. Chen, D. Wilkins, and R. Doerksen. Implementation of multiple-instance learning in drug activity prediction. *BMC Bioinformatics*, 13(Suppl 15):S3, 2012. Pages: 19, 62

[67] Z. Fu, A. Robles-Kelly, and J. Zhou. Milis: Multiple instance learning with instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (99):1–1, 2010. Pages: 9, 62, 65, 66

[68] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic press, 1990. Pages: 5, 6

[69] G. Fung, M. Dundar, B. Krishnapuram, and R. B. Rao. Multiple instance learning for computer aided diagnosis. In *Advances in Neural Information Processing Systems*, volume 19, page 425, 2007. Pages: 62

[70] T. Gärtner. Predictive Graph Mining with Kernel Methods. *Advanced methods for knowledge discovery from complex data*, pages 95–121, 2005. Pages: 111

[71] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *International Conference on Machine Learning*, pages 179–186, 2002. Pages: 7, 9, 10, 20, 28, 30, 34, 36, 37, 48, 62, 79, 98, 100, 113

[72] T. Gärtner, P. A. Flach, and S. Wrobel. On Graph Kernels: Hardness Results and Efficient Alternatives. In *Computational Learning Theory*. Springer Verlag, 2003. Pages: 111, 115, 116

[73] P. V. Gehler and O. Chapelle. Deterministic annealing for multiple-instance learning. In *International Conference on Artificial Intelligence and Statistics*, pages 123–130, 2007. Pages: 22

[74] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002. Pages: 66

[75] J. Ham, Y. Chen, M. M. Crawford, and J. Ghosh. Investigation of the random forest framework for classification of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):492–501, 2005. Pages: 68

[76] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988. Pages: 121

[77] P. E. Hart, N. J. Nilson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions of Systems, Science, and Cybernetics*, 4(2):100–107, 1968. Pages: 114

[78] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. New York: Springer-Verlag, 2001. Pages: 110

[79] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. Pages: 67

[80] B. Hoffmann, M. Zaslavskiy, J.-P. Vert, and V. Stoven. A new protein binding pocket similarity measure based on comparison of clouds of atoms in 3d: application to ligand prediction. *BMC Bioinformatics*, 11(1):99, 2010. Pages: 19

[81] T. Horváth, T. Gärtner, and S. Wrobel. Cyclic Pattern Kernels for Predictive Graph Mining. In *Knowledge Discovery and Data Mining*, pages 158–167. ACM, 2004. ISBN 1581138881. Pages: 111, 115

[82] J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3):299–310, 2005. Pages: 41, 71

[83] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9): 850–863, 1993. Pages: 32

[84] D. W. Jacobs, D. Weinshall, and Y. Gdalyahu. Classification with nonmetric distances: Image retrieval and class representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):583–600, 2000. Pages: 66

[85] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000. Pages: 18, 110

[86] T. Jebara. Images as bags of pixels. In *International Conference on Computer Vision*, pages 265–272, 2003. Pages: 18, 19

[87] H. Kalkan, M. Nap, R. P. W. Duin, and M. Loog. Automated colorectal cancer diagnosis for whole-slice histopathology. In *Medical Image Computing and Computer-Assisted Intervention*, pages 550–557. Springer, 2012. Pages: 23

[88] A. Kandel, H. Bunke, and M. Last. Applied graph theory in computer vision and pattern recognition. *Studies in Computational Intelligence*, 52, 2007. Pages: 110

[89] M. Kandemir and F. A. Hamprecht. Computer-aided diagnosis from weak supervision: A benchmarking study. *Computerized Medical Imaging and Graphics, in press*, 2015. Pages: 84, 86, 87, 90

[90] M. Kandemir, C. Zhang, and F. A. Hamprecht. Empowering multiple instance histopathology cancer diagnosis by cell graphs. In *Medical Imaging Computing and Computer Assisted Intervention*, 2014. Pages: 90

[91] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized Kernels Between Labeled Graphs. In *International Conference on Machine Learning*, volume 3, page 321, 2003. Pages: 115

[92] T.-K. Kim, J. Kittler, and R. Cipolla. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1005–1018, 2007. Pages: 24

[93] R. Kindermann, J. L. Snell, et al. *Markov random fields and their applications*, volume 1. American Mathematical Society Providence, RI, 1980. Pages: 18

[94] J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis & Applications*, 1(1):18–27, 1998. Pages: 63

[95] R. I. Kondor and T. Jebara. A kernel between sets of vectors. In *International Conference on Machine Learning*, volume 20, page 361, 2003. Pages: 18, 19

[96] H.-P. Kriegel and M. Schubert. Classification of websites as sets of feature vectors. In *Databases and Applications*, pages 127–132, 2004. Pages: 18, 19

[97] J. H. Krijthe, T. K. Ho, and M. Loog. Improving cross-validation based classifier selection using meta-learning. In *International Conference on Pattern Recognition*, pages 2873–2876. IEEE, 2012. Pages: 130

[98] H. Kuck and N. de Freitas. Learning about individuals from group statistics. In *Uncertainty in Artificial Intelligence*, pages 332–339, 2005. Pages: 21, 22

[99] L. I. Kuncheva. A stability index for feature selection. In *Artificial Intelligence and Applications*, pages 421–427, 2007. Pages: 88

[100] L. I. Kuncheva. Full-class set classification using the hungarian algorithm. *International Journal of Machine Learning and Cybernetics*, 1(1):53–61, 2010. Pages: 18, 24

[101] L. I. Kuncheva, J. J. Rodríguez, C. O. Plumpton, D. E. J. Linden, and S. J. Johnston. Random subspace ensembles for fmri classification. *IEEE Transactions on Medical Imaging*, 29(2):531–542, 2010. Pages: 68, 77

[102] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003. Pages: 67

[103] J. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, pages 282–289, 2001. Pages: 18

[104] C. Lai, M. J. T. Reinders, and L. Wessels. Random subspace method for multivariate feature selection. *Pattern Recognition Letters*, 27(10):1067–1076, 2006. Pages: 80

[105] W.-J. Lee, V. Cheplygina, D. M. J. Tax, M. Loog, and R. P. W. Duin. Bridging structure and feature representations in graph matching. *International Journal of Pattern Recognition and*

Artificial Intelligence, 26(05), 2012. Pages: 32

[106] W.-J. Lee and R. P. W. Duin. An inexact graph comparison approach in joint eigenspace. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 35–44, 2008. Pages: 112

[107] C. Leistner, A. Saffari, and H. Bischof. MIForests: multiple-instance learning with randomized trees. In *European Conference on Computer Vision*, pages 29–42. Springer, 2010. Pages: 9, 77

[108] Y. Li, D. M. J. Tax, R. P. W. Duin, and M. Loog. Multiple-instance learning as a classifier combining problem. *Pattern Recognition*, 2012. Pages: 9, 22, 31

[109] Y. Li, D. M. J. Tax, R. P. W. Duin, and M. Loog. The link between multiple-instance learning and learning from only positive and unlabelled examples. In *Multiple Classifier Systems*, volume 7872, pages 157–166. Springer Berlin Heidelberg, 2013. Pages: 21

[110] J. Liang and J. Bi. Computer aided detection of pulmonary embolism with tobogganing and mutiple instance classification in CT pulmonary angiography. In *Information Processing in Medical Imaging*, pages 630–641, 2007. Pages: 87

[111] M. Loog and B. van Ginneken. Static posterior probability fusion for signal detection: applications in the detection of interstitial diseases in chest radiographs. In *International Conference on Pattern Recognition*, volume 1, pages 644–647, 2004. Pages: 9, 20, 31, 99

[112] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. Pages: 121

[113] M. Mandel and D. P. W. Ellis. Multiple-instance learning for music information retrieval. In *International Society for Music Information Retrieval*, pages 577–582, 2008. Pages: 21

[114] O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems*, pages 570–576, 1998. Pages: 7, 8, 10, 18, 19, 28, 30, 38, 69, 98, 100, 111, 112

[115] O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *International Conference on Machine Learning*, volume 15, pages 341–349, 1998. Pages: 16, 19

[116] J. Marques. *Osteoarthritis imaging by quantification of tibial trabecular bone*. PhD thesis, Københavns Universitet, 2013. Pages: 84, 86, 87

[117] L. K. McDowell, K. M. Gupta, and D. W. Aha. Cautious inference in collective classification. In *National Conference on Artificial Intelligence*, volume 7, pages 596–601, 2007. Pages: 18

[118] J. Melendez, B. van Ginneken, P. Maduskar, R. H. H. M. Philipsen, K. Reither, M. Breuninger, I. M. O. Adetifa, R. Maane, H. Ayles, and C. I. Sanchez. A novel multiple-instance learning-based approach to computer-aided detection of tuberculosis on chest x-rays. *IEEE Transactions on Medical Imaging, in press*, 2014. Pages: 84, 86, 87

[119] C. S. Mendoza, G. R. Washko, J. C. Ross, A. A. Diaz, D. A. Lynch, J. D. Crapo, E. K. Silverman, B. Acha, C. Serrano, and R. S. J. Estepar. Emphysema quantification in a multi-scanner HRCT cohort using local intensity distributions. In *International Symposium on Biomedical Imaging*, pages 474–477, 2012. Pages: 98

[120] K. Muandet, K. Fukumizu, F. Dinuzzo, and B. Schölkopf. Learning from distributions via support measure machines. In *Advances in Neural Information Processing Systems*, pages 10–18, 2012. Pages: 130

[121] A. Müller and S. Behnke. Multi-instance methods for partially supervised image segmentation. *Partially Supervised Learning*, pages 110–119, 2012. Pages: 18, 21

[122] J. Munkres. Algorithm for the assignment and transportation problems. *Journal of the*

Society for Industrial and Applied Mathematics, 5:32–38, 1957. Pages: 114

[123] J. F. Murray, G. F. Hughes, and K. Kreutz-Delgado. Machine learning methods for predicting failures in hard drives: A multiple-instance application. *Journal of Machine Learning Research*, 6(1):783, 2006. Pages: 19

[124] D. R. Musicant, J. M. Christensen, and J. F. Olson. Supervised learning by training on aggregate outputs. In *International Conference on Data Mining*, pages 252–261. IEEE, 2007. Pages: 18, 21

[125] S. Nene, S. Nayar, and H. Murase. Columbia object image liberary: Coil-100. Technical report, Department of Computer Science, Columbia University, New York, 1996. Pages: 120

[126] X. Ning and G. Karypis. The set classification problem and solution methods. In *IEEE International Conference on Data Mining Workshops*, pages 720–729. IEEE, 2008. Pages: 16, 18, 23

[127] Y. S. Park, J. B. Seo, N. Kim, E. J. Chae, Y. M. Oh, S. Do Lee, Y. Lee, and S.-H. Kang. Texture-based quantification of pulmonary emphysema on high-resolution computed tomography: comparison with density-based quantification and correlation with pulmonary function test. *Investigative Radiology*, 43(6):395–402, 2008. Pages: 98

[128] R. A. Pauwels, A. S. Buist, P. M. A. Calverley, C. R. Jenkins, and S. S. Hurd. Global strategy for the diagnosis, management, and prevention of chronic obstructive pulmonary disease. *American Journal of Respiratory and Critical Care Medicine*, 163(5), 2012. Pages: 98

[129] J. H. Pedersen, H. Ashraf, A. Dirksen, K. Bach, H. Hansen, P. Toennesen, H. Thorsen, J. Brodersen, B. G. Skov, M. Døssing, et al. The Danish randomized lung cancer CT screening trial-overall design and results of the prevalence round. *Journal of Thoracic Oncology*, 4 (5):608–614, 2009. Pages: 90, 99, 101

[130] E. Pękalska and R. P. W. Duin. On combining dissimilarity representations. In *Multiple Classifier Systems*, pages 359–368. Springer, 2001. Pages: 63

[131] E. Pękalska and R. P. W. Duin. Dissimilarity representations allow for building good classifiers. *Pattern Recognition Letters*, 23(8):943–956, 2002. Pages: 36

[132] E. Pękalska and R. P. W. Duin. Dissimilarity representations allow for building good classifiers. *Pattern Recognition Letters*, 23(8):943–956, 2002. Pages: 120

[133] E. Pękalska and R. P. W. Duin. *The dissimilarity representation for pattern recognition: foundations and applications*, volume 64. World Scientific Pub Co Inc, 2005. Pages: 1, 9, 32, 62, 64, 100, 120

[134] E. Pękalska, R. P. W. Duin, and P. Paclík. Prototype selection for dissimilarity-based classifiers. *Pattern Recognition*, 39(2):189–208, 2006. Pages: 37, 66, 131

[135] E. Pękalska, R. P. W. Duin, and M. Skurichina. A discussion on the classifier projection space for classifier combining. In *Multiple Classifier Systems*, pages 137–148. Springer, 2002. Pages: 75

[136] E. Pękalska, A. Harol, R. P. W. Duin, B. Spillmann, and H. Bunke. Non-euclidean or non-metric measures can be informative. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 871–880. Springer, 2006. Pages: 7, 48, 101

[137] E. Pękalska, P. Paclík, and R. P. W. Duin. A generalized kernel approach to dissimilarity-based classification. *The Journal of Machine Learning Research*, 2:175–211, 2002. ISSN 1532-4435. Pages: 1, 46, 119

[138] Y. Plasencia Calaña, E. García Reyes, R. P. W. Duin, and M. Orozco-Alzate. On using

asymmetry information for classification in extended dissimilarity spaces. In *Iberoamerican Congress on Pattern Recognition*, volume 7441, pages 503–510, 2012. Pages: 34, 55

[139] T. Poggio, R. Rifkin, S. Mukherjee, and P. Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419–422, 2004. Pages: 88

[140] H. J. Qiu and E. R. Hancock. Clustering and embedding using commute times. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:1873–1890, 2007. Pages: 112

[141] N. Quadrianto, A. J. Smola, T. S. Caetano, and Q. V. Le. Estimating labels from label proportions. *Journal of Machine Learning Research*, 10:2349–2374, 2009. Pages: 18, 22

[142] G. Quellec, M. Lamard, M. D. Abràmoff, E. Decencière, B. Lay, A. Erginay, B. Cochener, and G. Cazuguel. A multiple-instance learning framework for diabetic retinopathy screening. *Medical Image Analysis*, 16(6):1228–1240, 2012. Pages: 84, 87

[143] K. F. Rabe, S. Hurd, A. Anzueto, P. J. Barnes, S. A. Buist, P. Calverley, Y. Fukuchi, C. Jenkins, R. Rodriguez-Roisin, C. van Weel, et al. Global strategy for the diagnosis, management, and prevention of chronic obstructive pulmonary disease: Gold executive summary. *American Journal of Respiratory and Critical Care Medicine*, 176(6):532–555, 2007. Pages: 98, 105

[144] R. Rahmani, S. A. Goldman, H. Zhang, J. Krettek, and J. E. Fritts. Localized content based image retrieval. In *International Workshop on Multimedia Information Retrieval*, pages 227–236. ACM, 2005. Pages: 40

[145] J. Ramon and T. Gärtner. Expressivity Versus Efficiency of Graph Kernels. In *International Workshop on Mining Graphs, Trees and Sequences*, pages 65–74. Citeseer, 2003. Pages: 115

[146] P. M. Rasmussen, L. K. Hansen, K. H. Madsen, N. W. Churchill, and S. C. Strother. Model sparsity and brain pattern interpretation of classification models in neuroimaging. *Pattern Recognition*, 45(6):2085–2100, 2012. Pages: 132

[147] S. Ray and M. Craven. Supervised versus multiple instance learning: An empirical comparison. In *International Conference on Machine Learning*, pages 697–704. ACM, 2005. Pages: 21, 31

[148] K. Riesen and H. Bunke. Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(7):950–959, 2009. Pages: 111, 114, 117

[149] K. Riesen and H. Bunke. Cluster ensembles based on vector space embeddings of graphs. In *Multiple Classifier Systems*, pages 211–221, 2009. Pages: 120

[150] K. Riesen and H. Bunke. Feature ranking algorithms for improving classification of vector space embedded graphs. In *Computer Analysis of Images and Patterns*, pages 377–384, 2009. Pages: 111

[151] K. Riesen and H. Bunke. Graph classification based on vector space embedding. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(6):1053–1081, 2009. Pages: 111, 120

[152] K. Riesen and H. Bunke. IAM Graph Database Repository for Graph Based Pattern Recognition and Machine Learning. *Structural, Syntactic, and Statistical Pattern Recognition*, pages 287–297, 2010. Pages: 120, 121

[153] K. Riesen, S. Fankhauser, H. Bunke, and P. J. Dickinson. Efficient suboptimal graph isomorphism. In *Graph-based Representations in Pattern Recognition*, pages 124–133, 2009. Pages: 111

[154] K. Riesen, M. Neuhaus, and H. Bunke. Bipartite graph matching for computing the edit

distance of graphs. In *Graph-based Representations in Pattern Recognition*, pages 1–12, 2007. Pages: 111

[155] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000. Pages: 35, 101

[156] N. A. Samsudin and A. P. Bradley. Nearest neighbour group-based classification. *Pattern Recognition*, 43(10):3458–3467, 2010. Pages: 16, 18, 23

[157] S. Scott, J. Zhang, and J. Brown. On generalized multiple-instance learning. *International Journal of Computational Intelligence and Applications*, 5(01):21–35, 2005. Pages: 7, 9, 28, 30

[158] P. Sen, G. Namata, M. Bilgic, and L. Getoor. Collective classification. In *Encyclopedia of Machine Learning*, pages 189–193. Springer, 2010. Pages: 18

[159] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93, 2008. Pages: 18

[160] M. Skurichina and R. P. W. Duin. Bagging and the random subspace method for redundant feature spaces. In *Multiple Classifier Systems*, pages 1–10. Springer, 2001. Pages: 67, 75

[161] L. Sørensen, P. Lo, A. Dirksen, J. Petersen, and M. De Bruijne. Dissimilarity-based classification of anatomical tree structures. In *Information Processing in Medical Imaging*, pages 475–485, 2011. Pages: 99

[162] L. Sørensen, M. Loog, P. Lo, H. Ashraf, A. Dirksen, R. P. W. Duin, and M. de Bruijne. Image dissimilarity-based quantification of lung disease from CT. In *Medical Image Computing and Computer-Assisted Intervention*, pages 37–44. 2010. Pages: 99, 104, 105

[163] L. Sørensen, M. Loog, D. M. J. Tax, W.-J. Lee, M. de Bruijne, and R. P. W. Duin. Dissimilarity-based multiple instance learning. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 129–138. Springer, 2010. Pages: 28

[164] L. Sørensen, M. Nielsen, P. Lo, H. Ashraf, J. H. Pedersen, and M. de Bruijne. Texture-based analysis of COPD: a data-driven approach. *IEEE Transactions on Medical Imaging*, 31 (1):70–78, 2012. Pages: 87, 90, 98, 99, 101, 102, 104, 105

[165] L. Sørensen, S. B. Shaker, and M. de Bruijne. Quantitative analysis of pulmonary emphysema using local binary patterns. *IEEE Transactions on Medical Imaging*, 29(2):559–569, 2010. Pages: 98

[166] A. Srinivasan, S. Muggleton, and R. D. King. Comparing the use of background knowledge by inductive logic programming systems. In *International Workshop on Inductive Logic Programming*, pages 199–230, 1995. Pages: 69

[167] L. Sun, Y. Lu, K. Yang, and S. Li. ECG analysis using multiple instance learning for myocardial infarction detection. *IEEE Transactions on Biomedical Engineering*, 59(12):3348–3356, 2012. Pages: 84, 87

[168] Q. Tao, S. D. Scott, N. V. Vinodchandran, T. T. Osugi, and B. Mueller. Kernels for generalized multiple-instance learning. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 30(12):2084–2098, 2008. Pages: 111

[169] D. M. J. Tax. MIL, a Matlab toolbox for multiple instance learning, May 2011. `http://prlab.tudelft.nl/david-tax/mil.html`. version 0.7.9. Pages: 38, 48, 79, 91, 102

[170] D. M. J. Tax and R. P. W. Duin. Learning curves for the analysis of multiple instance classifiers. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 724–733. Springer, 2008. Pages: 41, 71

[171] D. M. J. Tax, M. Loog, R. P. W. Duin, V. Cheplygina, and W.-J. Lee. Bag dissimilarities for multiple instance learning. In *Similarity-Based Pattern Recognition*, pages 222–234. Springer,

2011. Pages: 7, 20, 28, 30, 36, 54, 62, 64, 98, 101

[172] D. M. J. Tax, M. van Breukelen, R. P. W. Duin, and J. Kittler. Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition*, 33(9):1475–1485, 2000. Pages: 71

[173] V. Tragante do O, D. Fierens, and H. Blockeel. Instance-level accuracy versus bag-level accuracy in multi-instance learning. In *Benelux Conference on Artificial Intelligence*, 2011. Pages: 21, 106

[174] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007. Pages: 16

[175] R. Uppaluri, T. Mitsa, M. Sonka, E. A. Hoffman, and G. McLennan. Quantification of pulmonary emphysema from lung computed tomography images. *American Journal of Respiratory and Critical Care Medicine*, 156(1):248–254, 1997. Pages: 98

[176] L. van der Maaten, M. Chen, S. Tyree, and K. Q. Weinberger. Learning with marginalized corrupted features. In *International Conference on Machine Learning*, pages 410–418, 2013. Pages: 130

[177] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008. Pages: 105

[178] G. Vanwinckelen and H. Blockeel. A meta-learning system for multi-instance classification. In *ECML Workshop on Learning over Multiple Contexts*, pages 1–14, 2014. Pages: 129

[179] V. Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995. Pages: 110, 120

[180] A. Vezhnevets and J. M. Buhmann. Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning. In *Computer Vision and Pattern Recognition*, pages 3249–3256. IEEE, 2010. Pages: 18, 21

[181] A. J. Viera and J. M. Garrett. Understanding interobserver agreement: the kappa statistic. *Family Medicine*, 37(5):360–363, 2005. Pages: 88

[182] S. Vijayanarasimhan and K. Grauman. Keywords to visual categories: Multiple-instance learning forweakly supervised object categorization. In *Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. Pages: 20

[183] P. Viola, J. Platt, and C. Zhang. Multiple instance boosting for object detection. In *Advances in Neural Information Processing Systems*, volume 18, page 1417. Citeseer, 2006. Pages: 7, 9, 10, 30, 48, 77, 85, 98, 100, 112

[184] V. Vural, G. Fung, B. Krishnapuram, J. Dy, and B. Rao. Batch classification with applications in computer aided diagnosis. In *European Conference on Machine Learning*, pages 449–460. Springer, 2006. Pages: 18

[185] H. Y. Wang, Q. Yang, and H. Zha. Adaptive p-posterior mixture-model kernels for multiple instance learning. In *International Conference on Machine Learning*, pages 1136–1143, 2008. Pages: 28

[186] J. Wang. Solving the multiple-instance problem: A lazy learning approach. In *International Conference on Machine Learning*, 2000. Pages: 9, 10, 20, 28, 30, 36, 62, 100, 111

[187] S. Wang, M. T. McKenna, T. B. Nguyen, J. E. Burns, N. Petrick, B. Sahiner, and R. M. Summers. Seeing is believing: Video classification for computed tomographic colonography using multiple-instance learning. *IEEE Transactions on Medical Imaging*, 31(5):1141–1153, 2012. Pages: 84, 86, 87

[188] N. Weidmann, E. Frank, and B. Pfahringer. A two-level learning method for generalized multi-instance problems. In *European Conference on Machine Learning*, pages 468–479.

Springer, 2003. Pages: 7, 9, 19, 28, 30, 100

[189] R. C. Wilson, B. Luo, and E. R. Hancock. Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1112–1124, 2005. Pages: 112

[190] L. Wolf and A. Shashua. Learning over sets using kernel principal angles. *Journal of Machine Learning Research*, 4:913–931, 2003. Pages: 24

[191] D. H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996. Pages: 131

[192] D. Wu, J. Bi, and K. Boyer. A min-max framework of cascaded classifier with multiple instance learning for computer aided diagnosis. In *Computer Vision and Pattern Recognition*, pages 1359–1366, 2009. Pages: 84

[193] Y. Xu, J.-Y. Zhu, E. I. Chang, M. Lai, and Z. Tu. Weakly supervised histopathology cancer image segmentation and classification. *Medical Image Analysis*, 18(3):591–604, 2014. Pages: 84, 87

[194] M.-L. Zhang and Z.-H. Zhou. Multi-instance clustering with applications to multi-instance prediction. *Applied Intelligence*, 31(1):47–68, 2009. Pages: 7, 9, 30, 66, 98

[195] Q. Zhang, S. Goldman, et al. Em-dd: An improved multiple-instance learning technique. In *Advances in Neural Information Processing Systems*, volume 14, pages 1073–1080. Cambridge, MA: MIT Press, 2001. Pages: 7, 10, 30, 48, 77, 98, 100

[196] C. Zhao, W. Shi, and Y. Deng. A new Hausdorff distance for image matching. *Pattern Recognition Letters*, 26(5):581–586, 2005. Pages: 33

[197] S. K. Zhou and R. Chellappa. From sample similarity to ensemble similarity: Probabilistic distance measures in reproducing kernel hilbert space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):917–929, 2006. Pages: 19

[198] Z.-H. Zhou. Multi-instance learning: A survey. Technical report, Department of Computer Science and Technology, Nanjing University, 2004. Pages: 20

[199] Z. H. Zhou, K. Jiang, and M. Li. Multi-instance learning based web mining. *Applied Intelligence*, 22(2):135–147, 2005. Pages: 41

[200] Z. H. Zhou, Y. Y. Sun, and Y. F. Li. Multi-instance learning by treating instances as non-iid samples. In *International Conference on Machine Learning*, pages 1249–1256, 2009. Pages: 19, 32, 37, 40, 62, 70, 131

[201] Z.-H. Zhou and J.-M. Xu. On the relation between multi-instance learning and semi-supervised learning. In *International conference on Machine learning*, pages 1167–1174, 2007. Pages: 21

[202] Z.-H. Zhou and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In *Advances in Neural Information Processing Systems*, pages 1609–1616, 2006. Pages: 63

[203] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Advances in Neural Information Processing Systems*, pages 49–56, 2004. Pages: 66

[204] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005. Pages: 21

SUMMARY

Multiple instance learning (MIL) is an extension of supervised learning where the objects are represented by sets (*bags*) of feature vectors (*instances*) rather than individual feature vectors. For example, an image can be represented by a bag of instances, where each instance is a patch in that image. Only bag labels are given, however, the *standard assumption* is that that a bag is positive if and only if it contains a positive, or *concept* instance. In other words, only concept instances are informative for the bag label. The goal is to learn a bag classifier, although an instance classifier may also be desired. This scenario is suitable for applications where objects are heterogeneous and representing them as a single feature vector may lose important information, and/or in cases where only weakly labeled data is available.

Several approaches to MIL exist. Instance-based approaches rely on stronger assumptions about the relationship of the instance labels and the bag labels, and define a bag classifier through an instance classifier. Bag-based approaches learn a bag classifier directly, often by converting the problem into a supervised problem. These methods often disregard the standard assumption, and instead use the *collective assumption*, where *all* instances are informative. One way to convert the problem into a supervised one, is to describe each bag by a vector of its distances to a set of reference prototypes. In this so-called dissimilarity representation, supervised classifiers can be used. The goal of this thesis is to study the dissimilarity representation as a method for dealing with multiple instance learning problems. We address the questions of defining a dissimilarity function and choosing a reference set of prototypes, while considering the assumptions that these choices implicitly make about the problem.

SAMENVATTING

Multiple instance learning (MIL) is een uitbreiding op gesuperviseerd leren waar de voorbeelden niet individuele objecten zijn, maar groepen (zogenaamde bags) van objecten (zogenaamde instances). Zo kan een beeld worden beschreven door een bag, waarbij elke instance een stukje van het beeld is. Er worden alleen labels (zoals ja/nee) gegeven voor de bags, niet voor de instances. Vaak wordt er aangenomen dat een bag positief is voor een bepaald *concept* als, en alleen als, minstens n van de instances positief is, oftewel representatief is voor dat concept. Met andere woorden, zijn alleen deze concept instances verantwoordelijk voor de label van de bag. Het doel is om een bag classifier te leren die nieuwe bags van labels kan voorzien, howevel een instance classifier die nieuwe instances van labels voorziet, ook interessant zou kunnen zijn. Deze scenarios zijn relevant voor toepassingen met heterogene voorbeelden, waardoor elk voorbeeld als n featurevector beschrijven informatie verliest, en/of in toepassingen waar geen lokale annotaties beschikbaar zijn.

Er bestaan verschillende aanpakken voor MIL. Instance-gebaseerde aanpakken maken meer aannames over de relatie tussen de labels van de instances en van de bags. De bag classifier wordt dan gedefinieerd door de uitkomsten van instance classifiers te combineren. Bag-gebaseerde aanpakken leren direct een bag classifier, vaak door het omzetten van het probleem in een standaard gesuperviseerd probleem. Deze methoden negeren vaak de standaard aanname, en gebruiken in plaats daarvan een *collectieve aanname* waarbij alle instances, niet alleen de concept instances, informatief zijn. Eén van de manieren om een bag te beschrijven met één feature vector is door het te representeren door middel van de afstanden naar andere, referentie bags: het zogenaamde dissimilariteitsaanpak. Met deze representatie kunnen er weer gesuperviseerde classifiers gebruikt worden. Het doel van dit proefschrift is om het dissimilariteitsaanpak te bestuderen als mogelijkheid om MIL problemen op te lossen. Wij richten ons op de vragen van het definiëren van een dissimilariteitsfunctie en het kiezen van een referentieset van bags, met inachtneming van de impliciete aannames van deze keuzes.

Curriculum Vitae

Veronika Cheplygina was born in 1986 in Moscow, Soviet Union. In 1999, she moved to the Netherlands, where she attended the International School of The Hague. In 2004 she started her studies in Media & Knowledge Engineering at the Delft University of Technology. After receiving her BSc degree in 2007, she worked as a board member at W.I.S.V. 'Christiaan Huygens', the student society for mathematics and computer science students. Veronika became acquainted with pattern recognition during her MSc degree, which she successfully completed in 2010. Her thesis, "Random Subspace Method for One-Class Classifiers" about detecting outlier images in parcels was supervised by David Tax and partly carried out at Prime Vision.

In 2011 Veronika started her PhD project under the supervision of Marco Loog and David Tax. She investigated the dissimilarity representation as an approach to solve multiple instance learning problems in several applications, including computer aided diagnosis of chronic obstructive pulmonary disorder. As part of her PhD, she visited the Machine Learning & Computational Biology group at the Max Planck Institutes in Tübingen, Germany, and briefly, the Image Group at the University of Copenhagen, Denmark. Here she collaborated with Aasa Feragen on classification of autism spectrum disorder from brain networks.

Veronika is currently a postdoctoral researcher at the Biomedical Imaging Group Rotterdam at the Erasmus Medical Center in Rotterdam, the Netherlands.

List of Publications

Journals

1. Veronika Cheplygina, David M. J. Tax, and Marco Loog. Multiple instance learning with bag dissimilarities. *Pattern Recognition*, 48(1):264-275, 2015.
2. Ethem Alpaydın, Veronika Cheplygina, Marco Loog, and David M. J. Tax. Single- vs. multiple-instance classification. *Pattern Recognition, accepted*, 2015.
3. Veronika Cheplygina, David M. J. Tax, and Marco Loog. Dissimilarity-based ensembles for multiple instance learning. *IEEE Transactions on Neural Networks and Learning Systems, accepted*, 2015.
4. Veronika Cheplygina, David M. J. Tax, and Marco Loog. On classification with bags, groups and sets. *Pattern Recognition Letters, accepted*, 2015.
5. Wan-Jui Lee, Veronika Cheplygina, David M. J. Tax, Marco Loog, and Robert P. W. Duin. Bridging structure and feature representations in graph matching. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(05), 2012.

Conferences

1. Veronika Cheplygina, Lauge Sørensen, David M. J. Tax, Marleen de Bruijne and Marco Loog. Label stability in multiple instance learning. In *Medical Image Computing and Computer Assisted Intervention, accepted*, 2015.
2. Veronika Cheplygina, David M. J. Tax, Marco Loog, and Aasa Feragen. Network-guided group feature selection for classification of autism spectrum disorder. In *Machine Learning in Medical Imaging*, pages 190-197. Springer, 2014.
3. Veronika Cheplygina, Lauge Sørensen, David M. J. Tax, Jesper Holst Pedersen, Marco Loog, and Marleen de Bruijne. Classification of COPD with multiple instance learning. In *International Conference on Pattern Recognition*, pages 1508-1513, 2014.

4. Yenisel Plasencia Calaña, Veronika Cheplygina, Robert P. W. Duin, Edel García Reyes, Mauricio Orozco-Alzate, David M. J. Tax, and Marco Loog. On the informativeness of asymmetric dissimilarities. In *Similarity-Based Pattern Recognition*, pages 75-89. Springer, 2013.
5. Veronika Cheplygina, David M. J. Tax, and Marco Loog. Combining instance information to classify bags. In *Multiple Classifier Systems*, pages 13-24. Springer, 2013.
6. Veronika Cheplygina, David M. J. Tax, and Marco Loog. Class-dependent dissimilarity measures for multiple instance learning. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 602-610. Springer, 2012.
7. Veronika Cheplygina, David M. J. Tax, and Marco Loog. Does one rotten apple spoil the whole barrel? In *International Conference on Pattern Recognition*, pages 1156-1159, 2012.
8. David M. J. Tax, M. Loog, Robert P. W. Duin, Veronika Cheplygina, and Wan-Jui Lee. Bag dissimilarities for multiple instance learning. In *Similarity-Based Pattern Recognition*, pages 222-234. Springer, 2011.
9. Veronika Cheplygina and David M. J. Tax. Pruned random subspace method for one-class classifiers. In *Multiple Classifier Systems*, pages 96-105. Springer, 2011.