MULTIPLE-INSTANCE LEARNING FROM DISTRIBUTIONS

by

GARY DORAN

Submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy

Department of Electrical Engineering and Computer Science

CASE WESTERN RESERVE UNIVERSITY

January, 2015

CASE WESTERN RESERVE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

We hereby approve the dissertation of

Gary Doran

candidate for the degree of **Doctor of Philosophy***.

Committee Chair

Dr. Soumya Ray

Committee Member

Dr. Harold Connamacher

Committee Member

Dr. Michael Lewicki

Committee Member

Dr. Stanislaw Szarek

Committee Member

Dr. Kiri Wagstaff

Date of Defense

November 24, 2014

*We also certify that written approval has been obtained

for any proprietary material contained therein.

Contents

List of Tables

List of Figures

Acknowledgments

First, I would like to acknowledge my advisor, Soumya Ray. Over the past five years, I have learned how important a role an advisor plays in a graduate student's life, and how fortunate I am to have an advisor like Professor Ray who pushes me to apply for internships and scholarships, to revise and resubmit rejected papers, and to continue investigating research questions when obstacles arise. Professor Ray's wide knowledge of machine learning and artificial intelligence and his excitement about the field have inspired me to continue learning about new research topics. Likewise, Professor Ray has been a source of wisdom on practical matters such as writing applications and paper reviews.

I was also fortunate to have had two great internship experiences during my graduate career. First, I spent the Summer of 2012 with the Machine Learning and Instrument Autonomy group at the Jet Propulsion Laboratory in Pasadena, California. There, I worked with Kiri Wagstaff, who serves on my dissertation committee and whose paper "Machine Learning that Matters" significantly influenced my thinking about the role of machine learning in its relation to practical applications (Wagstaff, 2012). I spent the Summer of 2013 with Bernhard Schölkopf's research group at the Max Planck Institute in Tübingen, Germany. There, I worked with Krikamol Muandet, Kun Zhang, and others who influenced my thinking on machine learning in general, but more specifically on how kernel methods might be applied to the problem of learning from distributions.

At Case Western Reserve University, my committee is comprised of Harold Connamacher, Mike Lewicki, and Stanislaw Szarek, each of whom gave me valuable feedback on my dissertation. I was lucky enough to have taken classes with Professor Lewicki, who taught me the importance of Bayesian reasoning in artificial intelligence, and Professor Szarek, whose course in real analysis provided me with many of the mathematical tools required for reasoning about the subjects I investigate in this dissertation. I would also like to thank other students at Case Western Reserve with whom I worked, particularly Andrew Latham, who contributed to the experimental code for the results in Chapter 6.

Finally, I would like to thank my parents, sister, grandparents, extended family, and friends for supporting me during my graduate school and long before then.

List of Acronyms

MI-SVM$_\mathbf{I}$ modified MI SVM algorithm.

***p*-concept** probabilistic concept.

3D-QSAR 3-dimensional Quantitative Structure–Activity Relationship.

APR axis-parallel rectangle.

AUC area under the ROC curve.

CBIR content-based image retrieval.

CC-NSK class-conditional NSK.

DNF disjunctive normal form.

EMD Earth-Mover's Distance.

ERM empirical risk minimization.

GMIL Generalized MIL.

IID independent and identically distributed.

KI-SVM key instance SVM.

KPCA kernel principal component analysis.

LDA latent Dirichlet allocation.

LP linear program.

MI multiple-instance.

MI-SVM MI SVM algorithm.

mi-SVM mixed-integer MI SVM algorithm.

MICA MI classification algorithm.

MIGraph MI Graph Algorithm.

miGraph MI Graph Algorithm, Version 2.

MIL multiple-instance learning.

MILES Multiple-Instance Learning via Embedded Instance Selection.

MMD Maximum Mean Discrepancy.

NMI noisy MI.

NSK normalized set kernel.

PAC probably approximately correct.

PDF probability density function.

QP quadratic program.

RBF radial basis function.

RKHS reproducing kernel Hilbert space.

ROC receiver operating characteristic.

sbMIL sparse balanced MIL.

SIL single-instance learning.

SIVAL spatially independent, variable area, and lighting.

sMIL sparse MIL.

SMILe Shuffled Multiple-Instance Learning.

SMILeSVM SMILe SVM algorithm.

SMM support measure machine.

SRM structural risk minimization.

stMIL sparse transductive MIL.

SVM support vector machine.

VC Vapnik–Chervonenkis.

YARDS Yet Another Radial Distance-based Similarity measure.

Glossary

bag a set of instances in the multiple-instance learning framework.

concept a function that maps examples to labels.

example an object, typically a feature vector, provided to a learning algorithm for training or classification.

feature a property of an example, usually encoded as one component of a real-valued vector corresponding to the example.

feature map a function mapping data from one "input" feature space to another feature space.

generative model a set of assumptions about the process by which examples are sampled from a distribution and how labels are applied to examples.

instance a single example; in the multiple-instance setting, an element contained within a bag.

kernel a real-valued function of two arguments from the input feature space representing the inner product of the images of the arguments under some feature map.

loss a measure of misclassification or error with respect to a dataset (cf. empirical risk).

margin the distance between the positive and negative classification boundaries of a support vector machine classifier.

pseudo-dimension the size of the largest set that some real-valued hypothesis class shatters.

risk a measure of expected misclassification or error of a function with respect to a distribution over instances (cf. expected loss).

testing the stage of learning in which the quality of a learned concept is evaluated using previously-unobserved labeled examples.

training the stage of learning in which labeled examples are provided to an algorithm to learn a concept.

Multiple-Instance Learning from Distributions

Abstract

by

GARY DORAN

I propose a new theoretical framework for analyzing the multiple-instance learning (MIL) setting. In MIL, training examples are provided to a learning algorithm in the form of labeled sets, or "bags," of instances. Applications of MIL include 3-D quantitative structure–activity relationship prediction for drug discovery and content-based image retrieval for web search. The goal of an algorithm is to learn a function that correctly labels new bags or a function that correctly labels new instances. I propose that bags should be treated as latent distributions from which samples are observed. I show that it is possible to learn accurate instance- and bag-labeling functions in this setting as well as functions that correctly rank bags or instances under weak assumptions. Additionally, my theoretical results suggest that it is possible to learn to rank efficiently using traditional, well-studied "supervised" learning approaches. These results also indicate that supervised approaches for learning from distributions can be used to directly learn bag-labeling functions efficiently. I perform an extensive empirical evaluation that supports the theoretical predictions entailed by the new framework. In addition to showing how supervised approaches can be applied to MIL, I prove new hardness results on using MI-specific algorithms to learn hyperplane labeling functions for instances. Finally, I propose a new resampling approach for MIL, analyze it under the new theoretical framework, and show that it can improve the performance of MI classifiers when training set sizes are small. In summary, the proposed theoretical framework leads to a better understanding of the relationship between the MI and standard supervised learning settings, and it provides new methods for learning from MI data that are more accurate, more efficient, and have better understood theoretical properties than existing MI-specific algorithms.

Chapter 1

Introduction

The field of machine learning has grown in the past several decades from a need to automatically extract useful information from ever-increasing quantities of data. What qualifies as "useful" information varies across domains in which machine learning is applied. As an example, one might be interested in automatically recognizing several distinct *classes* of objects within a set of objects, or being able to automatically determine the class of a new object given a previously-labeled set of objects. We can generate many concrete examples of such *classification* problems by appropriately defining which we mean by "object" and "class." To name just a few examples, we might be interested in determining what foods are "poisonous" or "not poisonous," which politicians are "conservative" or "liberal," or which tumors are "malignant" or "benign."

In its most general form, the classification problem above can be expressed simply mathematically: given a set of objects $\mathcal{X}$, a set of classes $\mathcal{Y}$, and some function $f : \mathcal{X} \to \mathcal{Y}$ that assigns a class label to each object, learn a good approximation for f. Ignoring for the moment interesting questions such as "What constitutes a *good* approximation for f?" or "Under what conditions can we expect to be capable of learning such a function?" we see that there is an even more fundamental question:

how can we represent objects in the real world such as foods, politicians, or tumors as a set $\mathcal{X}$ of mathematical objects?

In most of the work on classification, it is assumed that objects are represented mathematically as a set of vectors over $\mathbb{R}^k$, where each of the k components of the vector represents some *feature* of the object. In the examples above, a food might be represented as the quantities of a fixed set of k chemical compounds comprising it, a politician might be represented using their views on a particular set of k political issues, and a tumor might be represented using the outcomes of a fixed set of k diagnostic tests.

Although the "standard supervised" formalism described above is well understood, in practice, data does not always have such a straightforward representation as a single feature vector. In machine learning, there is a need to learn from structured objects that are more naturally represented using sets, tree, graphs, or even statements in first-order logic. As a simple example, consider the problem of drug activity prediction, which attempts to predict whether a molecule is "active" or "inactive," that is, whether it will bind to a target receptor protein or not. An illustration is provided in Figure 1.1, showing a molecule that binds to the receptor (Molecule A), and a molecule that does not bind to the receptor (Molecule B). The potential to bind to a protein largely depends on a molecule's structure, which can be represented as a feature vector (Cramer et al., 1988). On the other hand, flexible bonds in molecules mean that each molecule can exist in multiple shapes, called *conformations*, when dissolved in solution. Two conformations are shown for each of the molecules in Figure 1.1. If a molecule is observed to be active, that implies that *at least one* low-energy conformation will bind to the target. On the other hand, inactivity of a molecule means that *no* conformation binds. The binding conformation of Molecule A is indicated in the figure.

The multiple-instance learning (MIL) framework was motivated by the above

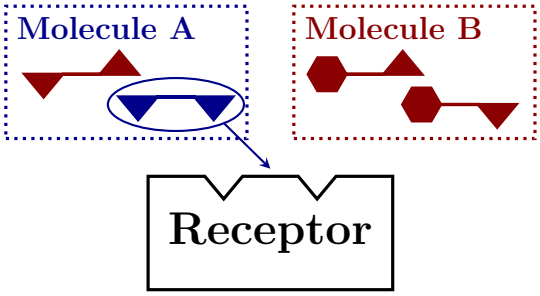


Figure 1.1: A cartoon example from the drug activity prediction domain. Molecule A binds to the receptor since at least one of its conformations has the appropriate structure, while Molecule B does not.

problem (Dietterich et al., 1997), and encodes this relationship between an observed label and a set of instances responsible for that label. In particular, a dataset is treated as a set of labeled *bags*, each of which contains one or more *instances*, which are feature vectors. If a bag is labeled positive, then at least one instance in the bag is positive. However, if a bag is negative, then every instance in the bag is negative. The learner for a multiple-instance (MI) classification problem must produce a classifier that can accurately label either new bags or instances.

MIL is a framework for learning from a very simple form of structured data. In MIL, bags are hierarchically structured objects containing subobjects (instances). Both types of objects have class labels, which are related to each other in such a way that it is possible to infer something about the labels at one level in the hierarchy given the labels at the other level. Therefore, MIL is applicable to many problems with data of this form, such as molecules containing conformations, images containing objects, or documents containing passages/words. From a theoretical perspective, the study of MIL provides a starting point for approaching the problem of learning from more complex structured data represented using trees or graphs, for example.

In addition to learning from structured data, there is also recent interest within machine learning in *learning from distributions*. In this line of research, the objects to be classified are not vectors, trees, or graphs, but *distributions* over a space of instances $\mathcal{X}$. In our work, we combine these research directions and explore a formulation of MIL as a problem of learning from distributions. In particular, although bags are traditionally represented as *sets* or *multisets* of instances in the MIL framework,

treating bags as distributions is more natural for many problem domains and provides new insights into the behavior of existing approaches for MI classification, the applicability of new state-of-the-art approaches to MI classification, and the relationship between MI and traditional supervised learning.

First, we introduce a new generative model for MIL that makes weaker assumptions and is more widely applicable to real-world MI problems than existing models. Second, we show that this generative model explains several surprising empirical observation made in prior work. In particular, some prior work shows that the classification performance of individual algorithms on the instance- and bag-labeling tasks is often uncorrelated (Tragante do Ó et al., 2011). We provide positive learnability results for instance and bag labeling, and our results suggest that these two labeling tasks should be addressed separately in practice. For learning bag labels, we show that the generative model can be used to justify the application of recent kernel-based approaches to learning from distributions and to analyzing existing bag-labeling algorithms under the bags-as-distributions model. Furthermore, we develop a new set of theoretical results showing that it is possible to learn to *rank* instances and bags using standard supervised approaches. Our work is consistent with previous empirical observations that supervised approaches can learn to rank in the MI setting (Ray and Craven, 2005), which were surprising given early work showing that supervised approaches are ineffective at learning from MI data (Dietterich et al., 1997). We perform our own extensive empirical evaluation to support the theoretical claims entailed by our model and to demonstrate the applicability of our results to practical scenarios.

Along with these positive results, we show new hardness results under the standard model in which bags are sets. In particular, we prove that it is generally not possible in the standard model for MI-specific approaches to have desirable "soundness," "completeness," and "convexity" properties that are present in standard supervised

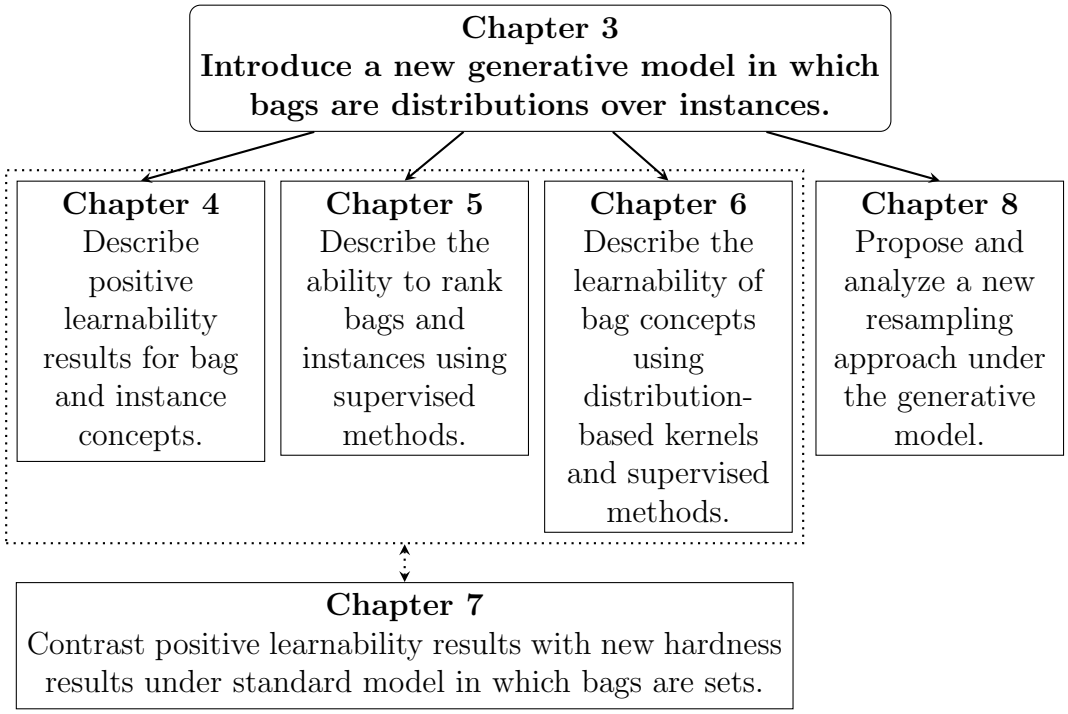


Figure 1.2: A summary of the contributions of this work.

methods. These results offer a contrast to the positive learnability results, which suggest that supervised approaches that possess these properties can learn from MI data under certain assumptions. Finally, we propose a new resampling approach for MIL, and the generative model is used to analyze this approach and describe how it can improve MI classification performance when training samples are small. The set of contributions is summarized in Figure 1.2.

Chapter 2

Background and Related Work

This chapter provides a brief overview of relevant concepts for the work in subsequent chapters. First, we formally describe supervised learning and multiple-instance learning, and the relationships between these learning frameworks. Then, we describe the support vector machine (SVM), a popular approach to classification in the supervised setting. Next, we describe approaches for representing distributions in a vector space such that supervised approaches such as SVMs can be applied to learning bag labels in the MI framework. Finally, we describe some concepts from learning theory that formalize what it means to "learn" concepts from data.

2.1 Learning Frameworks

As motivated in Chapter 1, many machine learning tasks can be framed as the problem of *learning to predict labels from examples*. However, this broad definition leaves unspecified details such as the structure of examples or how labels are applied to examples. Accordingly, there exist many possible learning frameworks that specify the nature of examples, labels, and the relationships between the two. These details are described by a *generative model*. Below, we describe two common learning frameworks and the relative granularity of representations they allow for labeled examples.

2.1.1 Supervised Learning

Suppose you are a newborn trying to learn to recognize various objects in the world. You observe the objects in Figure 2.1 and are told that some of these objects are spoons. Your goal is to learn the concept of a "spoon;" that is, learn to predict whether other objects you observe are spoons or non-spoons. Such a problem can be formalized in the framework of what is called *supervised learning*. The learning process is "supervised" by some oracle (perhaps a parent, in the example above) that provides labels for a set of examples that are used to train an algorithm.

More formally, in the supervised setting, there exists a given set of objects $\mathcal{X}$ and labels $\mathcal{Y}$. In classification, C is a finite set of class labels, often binary (either $\{0, 1\}$, $\{-1, 1\}$, or {"negative", "positive"}, respectively). For regression problems, C is the set $\mathbb{R}$ of real numbers. The set of objects is $\mathcal{X}$ is typically a real vector space $\mathbb{R}^k$, where each object (called an *instance* or an *example*) is represented as a k-dimensional vector. Each dimension of the example corresponds to a *feature* of the object. When instances are single feature vectors, we refer to this as the "standard supervised" learning setting.

In supervised learning, one assumes that there exists a function $f : \mathcal{X} \to \mathcal{Y}$, called a *target concept*, which assigns a label to each object. Here, we assume that a concept is a deterministic function, but in Section 2.4.3, we will discuss generalizations of this model in which concepts can be probabilistic. The concept f provides *ground truth* labels for each object. During training, a learning algorithm is provided with a set of labeled examples $\{(x_i, f(x_i)\}_{i=1}^n$. Since f is unknown to the algorithm, the goal of the learning process is to find some function g that approximates f as closely as possible. In Section 2.4, we discuss more precisely what it means to find a good approximation for a concept.

In the example of Figure 2.1, we assume that the various objects in the world can be represented with a vector of features. In general, determining an appropriate

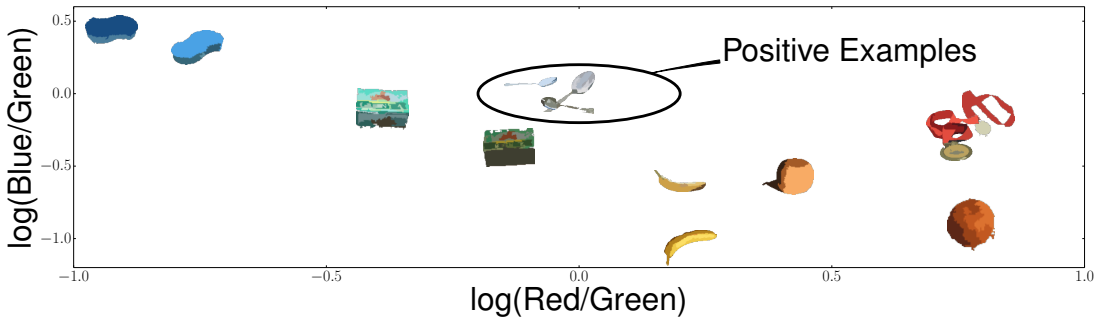


Figure 2.1: An example showing a supervised learning task. Here, the task is binary classification of objects from images as spoons and non-spoons. The positive examples are highlighted and the other objects are negative examples. Each objected is embedded in a two-dimensional feature space of color-based features.

set of features to describe objects is nontrivial but will not be discussed here. In Figure 2.1, each object is represented as a two-dimensional feature vector. The first feature represents the logarithm of the ratio between the average red pixel intensity and the average green pixel intensity. The second feature is the logarithm of the average blue and green pixel intensities. As shown, even these two simple features provides a nice separation between spoons (the positive examples) and other objects (the negative examples). Of course, if objects like forks were also included in the classification task, other shape-based features would be necessary to separate positive and negative examples.

2.1.2 Multiple-Instance Learning

For many applications, the most natural representation of an object is not an individual feature vector. The MIL framework was created as a generalization of supervised learning in which labeled objects have internal structure (Dietterich et al., 1997). For example, in the 3-dimensional Quantitative Structure–Activity Relationship (3D-QSAR) domain, molecules can be described as a set of conformations, each of which represented with its own feature vector and associated but unobserved "active" or "inactive" label. In the MIL setting, labeled objects are *multisets*, called bags, of

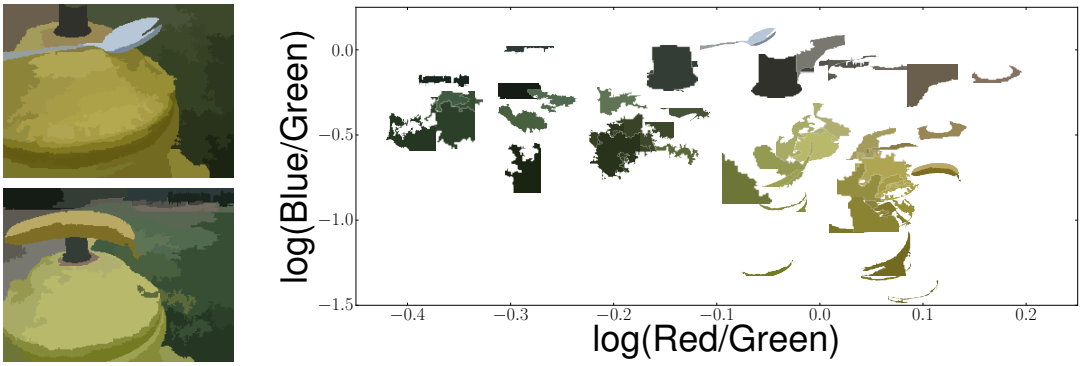


Figure 2.2: An example of positive and negative bags in the MI setting embedded in a feature space. The positive bag (top left) contains a positive example (the spoon), and the negative bag (bottom left) contains only negative examples.

individual feature vectors, called instances.

Expanding upon the example object classification task described in Section 2.1.1, suppose that we now have not individual labeled objects, but labeled images that contain sets of image patches roughly corresponding to objects or parts of objects.[1] Figure 2.2 shows a positive image (bag) at the top left containing a positive instance (the spoon) and a negative bag at the bottom left containing no spoon. In the MI setting, these bags are treated as sets of segments, which are embedded in a feature space as shown to the right of Figure 2.2. An MI learning algorithm knows that at least one of the segments in the positive image is a spoon, but it does not know which one. Thus, to learn instance-level labels, such an algorithm must not only learn to distinguish spoons from non-spoons, but also to determine *which* segments in the positive images correspond to spoons.

As in the object classification task, the *standard* MIL assumption for binary classification is that a bag is positive if and only if it contains at least one positive instance. Generalizations of the standard MI assumption are discussed in Section 2.1.3, but we use the standard assumption unless otherwise stated. More formally, let $\mathcal{X}$ be a set of instances and $\mathcal{X}^* = \bigcup_{n=1}^{\infty} \mathcal{X}^n$ denote the set of all possible bags (all finite subsets

[1]See Appendix A.1.2 for a more detailed description of how bags are generated from images in the content-based image retrieval (CBIR) domain.

of the instance space). Again, $\mathcal{Y}$ is a set of labels, which we assume to be $\{0, 1\}$ for binary classification. There exists a *bag concept* $F : \mathcal{X}^* \to \mathcal{Y}$ and an *instance concept* $f : \mathcal{X} \to \mathcal{Y}$ such that for any bag $B_i \in \mathcal{X}^*$, $F(B_i) = \max_{x_{ij} \in B_i} f(x_{ij})$. During training, an algorithm is provided with a set of labeled bags $\{B_i, F(B_i)\}_{i=1}^n$, where each bag is a set of instances $B_i = \{x_{ij}\}_{j=1}^{m_i}$. If $m_i = 1$ for every bag, then this special case is a standard supervised learning task as described in Section 2.1.1. The two possible learning tasks are to learn a function F' to approximate the bag concept F and to learn a function g to approximate the instance concept f. A key contribution of this work is describing when learning such concepts from MI data is possible.

So far, the only representational advantage presented for MIL has been treating objects as sets rather than as individual feature vectors. An alternative strategy such as concatenating the feature vectors in each bag to produce a single feature vector fails for a couple of reasons. First, since there is no fixed bag size, the feature vectors resulting from such an approach would not exist in the same vector space. Additionally, concatenation imposes an arbitrary order on instances, but there is no reason why the features of the i^{th} instance in one bag should be commensurate with only the features of the i^{th} instance in another bag. Other straightforward approaches such as averaging instances together provide only summary-level statistics about a bag. For example, two distinct bags with different labels could have the same average, so there would no longer be a deterministic function mapping bags to labels. Therefore, there *are* clear representational advantages in some scenarios when objects in the supervised setting are treated as sets rather than individual vectors.

There is another aspect of MIL that distinguishes it further from supervised learning. In particular, a key assumption in the MI setting is that instances, in addition to bags, have labels that are unobserved. In the object classification example in Figure 2.2, both entire images and individual segments can be labeled as containing the spoon. Accordingly, there are two labeling tasks in MIL: the bag-labeling task and

the instance-labeling task. The bag-labeling task can be viewed as a form of supervised learning, as discussed in Chapter 6. On the other hand, the instance-labeling task requires additional assumptions about the relationship between bag and instance labels. Of course, when bags all have size one, the instance- and bag-labeling tasks are identical and the learning task is equivalent to supervised learning. However, when bags contain more than one instance, the instance-labeling task is strictly more general than standard supervised learning, which does not permit ambiguous instance labels.

As discussed in Chapter 1, the MI setting was originally motivated by the drug activity prediction or 3D-QSAR problem (Dietterich et al., 1997). However, MIL has since been applied to many other problem domains. The example in Figure 2.2 is taken from the CBIR domain (Maron and Ratan, 1998). For text categorization, individual passages (e.g., sentences, paragraphs, message board posts) are represented using feature vectors of word frequencies, and the passages comprising a document correspond to a bag (Andrews et al., 2003; Settles et al., 2008). The MI setting has also been used for protein sequence classification (Wang et al., 2004; Tao et al., 2004), in which subsequences of amino acids are used to determine whether proteins belong to a particular "superfamily." As a final example, MIL has also been applied in the audio domain to recognize bird songs (Briggs et al., 2012). An audio recording is made using microphones placed in an experimental forest containing birds of several species. The spectrogram of the recording is then segmented to isolate individual bird songs. As in the CBIR domain, a recording contains the bird song of a particular if at least one segment in the recording corresponds to the species, so this becomes an MIL problem as well. Table 2.1 summarizes these problem domains, which are only a subset of those explored in prior work.

The seminal work on MIL describes an algorithm to learn axis-parallel rectangles (APRs) for the drug activity problem described in the introduction (Dietterich et al.,

Table 2.1: The application of MIL to various problem domains.

Domain	**Bags**	**Instances**
3D-QSAR	Molecules	Conformations
CBIR	Images	Segments
Text Categorization	Documents	Passages
Protein Sequence Classification	Proteins	Amino Acid Sequences
Birdsong Classification	Audio Recordings	Spectrogram Segments

Table 2.2: The extension of some supervised algorithms to the MI setting.

Algorithm	**MI Extension(s)**	**Description**
APRs	Dietterich et al. (1997)	Finds the smallest APR that covers some instance in each positive bag
Gaussian Models	Maron (1998) Zhang and Goldman (2001)	Models the positive instances as a Gaussian distribution; finds a Gaussian that assigns high likelihood to one instance in each positive bag with low likelihood to instances in negative bags
Decision Trees	Blockeel et al. (2005)	Modifies splitting of decision tree nodes to incorporate several MI-specific heuristics
Neural Networks	Ramon and De Raedt (2000) Zhou and Zhang (2002)	Neural outputs for each instance are combined with a "softmax" layer to produce a bag-level output
Logistic Regression	Xu and Frank (2004) Ray and Craven (2005)	Bag-level likelihood maximized using a combining function (e.g., softmax) over instance likelihoods
SVMs	Andrews et al. (2003) (see Section 2.2.3)	Encodes the MI assumption in SVM constraints

1997). The concept class of APRs is chosen specifically for the drug activity problem, but the paper suggested that other supervised learning algorithms might be extended to MI setting for solving problems from other domains. In fact, there is a large body of subsequent work on extending popular supervised algorithms to the MI setting. Most MI-specific algorithms are created by modifying supervised formulations to account for the relationship between bag and instance labels in the MI setting. Some example algorithms are listed in Table 2.2, and prior work has compared these approaches across a variety of datasets (Amores, 2013).

2.1.3 Generalizations of Multiple-Instance Classification

There have been several generalizations of the standard MI classification framework in prior work. One straightforward extension is MI regression, which generalizes the set of bag and instance labels $\mathcal{Y}$ to be $\mathbb{R}$, the set of real numbers (Amar et al., 2001; Ray and Page, 2001). As with MI classification, MI regression must specify how a bag label is derived from instance labels through what is called a *combining function*. Again motivated by the 3D-QSAR problem, the max combining function is often used so that $F(B_i) = \max_{x_{ij} \in B_i} f(x_{ij})$ as for classification. The intuition is that the observed activity of a molecule is the maximum activity over all of its likely conformations. However, in the MI regression setting, a richer class of combining functions can be used. For example, some prior work simply assumes that *some* relevant subset of instances in each bag are responsible for a bag's label (Wagstaff et al., 2008).

Another generalization of MI classification changes the combining function used to assign binary labels to bags. As described by Foulds and Frank (2010), consider a CBIR problem in which the task is to distinguish pictures of deserts, oceans, and beaches. Segments in these images are primarily either sand or water. In the standard MI setting, it is assumed that the presence of a single type of instance was sufficient to identify an image category. However, in this case, the presence of both sand *and* water is required to distinguish beaches from deserts (only sand) or oceans (only water). The existence of such concepts is permitted under the Generalized MIL (GMIL) framework (Scott et al., 2005). In GMIL, it is assumed that some types of instances are "attractive" and that others are "repulsive." For a bag to be positive, it must contain a certain number of attractive instance types and exclude some number of the repulsive instance types. The generalization allows for richer relationships between bag and instance labels.

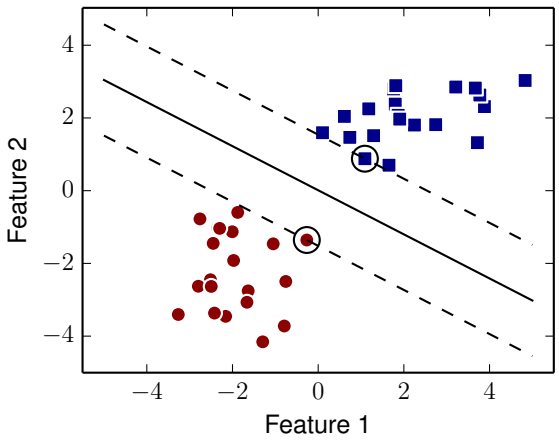


Figure 2.3: An SVM classifier that separates positive (blue) and negative (red) examples in a two-dimensional feature space. The classifying hyperplane is the solid line, and the boundary of the margin is indicated with dashed lines. The support vectors are circled.

2.2 Kernel Methods

So far, labeling functions in the standard supervised and MI learning settings have been discussed abstractly. For theoretical results discussed in Chapter 4 and Chapter 5, we continue to analyze these learning problems in an abstract setting. However, more detailed results can often be obtained by focusing on a particular *class* of labeling functions, or "hypotheses." For example, hyperplanes in the feature space (when $\mathcal{X}$ is an inner product space) is a popular class of hypotheses we discuss further in Chapter 6 and Chapter 7.

Kernel methods are a powerful and well-studied family of approaches that use "kernel functions" to learn nonlinear concepts for classification and regression tasks. Kernels are often used with some underlying classification technique, such as an SVM, and they provide a way to learn classifying hyperplanes on nonlinear features derived from an original training set. Below, we first describe the linear SVM. Then we describe how to introduce kernels to the SVM formulation to produce nonlinear concepts over data.

2.2.1 Support Vector Machines

The SVM (Vapnik and Kotz, 2006) is an approach to binary classification that attempts to "separate" data from the two classes. Figure 2.3 shows a dataset with two features, with the positive examples indicated with blue squares, and the negative examples with red circles. For this dataset, there are many lines that separate the data. The intuition behind the SVM is to choose the hyperplane that is as far away as possible from examples in each class. Equivalently, the SVM maximizes the margin, or the region between the positive and negative examples measured perpendicular to the hyperplane. The margin-maximization approach makes the SVM robust to error or noise in the data, and has been shown more formally to produce a classifier that generalizes well to new examples (Bartlett and Shawe-Taylor, 1999). The SVM gets its name from the examples, called "support vectors," that lie on the boundary of the margin. In Figure 2.3 the solid line is the classifying hyperplane, the boundary of the margin is indicated with dashed lines, and the support vectors are circled.

Formally, given a dataset of k-dimensional examples $X = \left\{x_i \in \mathbb{R}^k\right\}_{i=1}^{n}$ with corresponding binary labels $Y = \{y_i \in \{-1, +1\}\}_{i=1}^{m}$, a classifying hyperplane (w, b) assigns a label $f(x_i) = \text{sign}\left(\langle w, x_i\rangle + b\right)$ to examples. For training, the classifier must satisfy the constraints $f(x_i) \geq 1$ for positive examples and $f(x_i) \leq -1$ for negative examples so that examples are outside the margin. Equivalently, we can use the single constraint $y_i f(x_i) \geq 1$ for all examples. The size of the margin is inversely proportional to $\frac{1}{2}\left\|w\right\|^2$. Therefore, finding a hyperplane such as that in Figure 2.3 is equivalent to minimizing the following quadratic program (QP):

$$\begin{aligned} &\min_{w,b} \tfrac{1}{2}\left\|w\right\|^2, \\ &\text{s.t.} \quad y_i\left(\langle w, x_i\rangle + b\right) \geq 1. \end{aligned} \tag{2.1}$$

A more modern SVM formulation (Cortes and Vapnik, 1995) addresses two issues

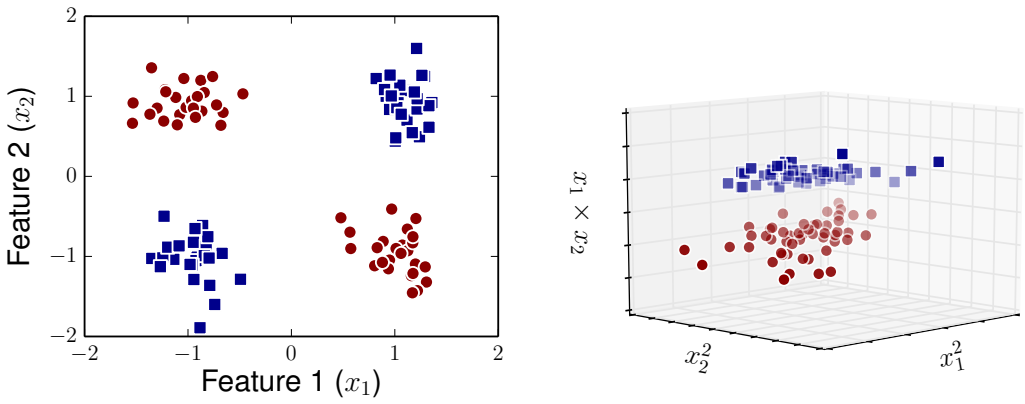


Figure 2.4: A linearly inseparable dataset (**left**) becomes linearly separable under the quadratic feature map (**right**).

with the "hard-margin" approach in Equation 2.1. First, not all datasets are as nicely separable with a hyperplane as the one shown in Figure 2.3. Additionally, we might be interested in misclassifying a few examples if it leads to a larger margin and hence provides better generalization to new examples. Therefore, the "soft-margin" approach introduces nonnegative *slack variables* $\xi_i \geq 0$ in the constraints: $y_i f(x_i) \geq 1 - \xi_i$. These slack variables allow misclassification of x_i by a magnitude of ξ_i to satisfy the constraints. Therefore, $\sum_i^n \xi_i$ can be interpreted as a measure of *loss*, or the cost of classification error of a particular classifier (w, b). The soft-margin formulation aims to minimize loss while maximizing the margin, using a constant parameter C to adjust the trade-off between these two goals:

$$\begin{aligned} &\min_{w,b,\boldsymbol{\xi}} \tfrac{1}{2}\|w\|^2 + C\sum_i \xi_i, \\ &\text{s.t.} \quad y_i\left(\langle w, x_i\rangle + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0. \end{aligned} \tag{2.2}$$

2.2.2 Kernels and Nonlinear Classifiers

As formulated in Equation 2.2, the SVM can only classify data using linear functions (hyperplanes) in the original feature space of the data. However, the concepts we would like to learn are generally nonlinear functions of the input data features. For

example, the left-hand side of Figure 2.4 shows a dataset that is not linearly separable. One solution to this problem is to use a *feature map*, which is a function mapping data from one "input" feature space to another feature space. From this point forward, we refer to the original space as the *input space* and reserve the term *feature space* for the resulting feature space in which classification is performed.

As in Figure 2.4, suppose $\mathcal{X} = \mathbb{R}^2$. Consider the feature map $\phi : \mathcal{X} \to \mathcal{H}$ defined by $\phi\big((x_1, x_2)\big) = (x_1^2, x_2^2, x_1 \times x_2)$; that is, ϕ maps a vector $(x_1, x_2) \in \mathcal{X}$ to a vector in $\mathcal{H} = \mathbb{R}^3$ consisting of all pairwise products of original features. The result of the mapping is shown on the right-hand side of Figure 2.4. In the new feature space $\mathcal{H}$, the data is now clearly linearly separable. By training an SVM classifier in the feature space $\mathcal{H}$ rather than the input space $\mathcal{X}$, we can avoid the problem of inseparability with respect to the original features.

However, there is an issue with this approach. In particular, suppose the input space dimensionality is k, then the dimensionality of $\mathcal{H}$ under the quadratic feature map is $O(k^2)$, which makes training an SVM classifier much less efficient. Furthermore, if a more complex feature map is needed to induce separability (for example, products of all triples of the original features), then the dimensionality of the feature space grows larger, making the feature mapping and SVM training computationally infeasible.

An elegant approach to incorporating the notion of a feature map while avoiding computational issues is to use a *kernel* (Schölkopf and Smola, 2002). Formally, a kernel k corresponding to a feature map ϕ is a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined by $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. Since the kernel is the inner product between two examples under a feature map, we require that the feature space $\mathcal{H}$ be an inner product space, or technically, a Hilbert space. We can define k with a particular feature map ϕ in mind. For example, the kernel $k(x_i, x_j) = \langle x_i, x_j \rangle^2$ corresponds to the quadratic

feature map in Figure 2.4.[2] However, as long as k is a symmetric function satisfying *Mercer's condition* ($\iint k(x,y)f(x)f(y)\,\mathrm{d}x\,\mathrm{d}y \geq 0$ for all square-integrable functions f), then k is a valid kernel corresponding to some implicit feature map ϕ (Schölkopf and Smola, 2002). Furthermore, assuming a constant time requirement to compute the kernel function, computation the kernel requires fixed $O(n^2)$ time regardless of the dimensionality of the feature space $\mathcal{H}$, which could be infinite.

The use of a kernel with the SVM or other technique is often called the *kernel trick*, and is justified by two observations. First, rewriting the soft-margin SVM formulation in Equation 2.2 in terms of the feature map, we get:

$$\begin{aligned} &\min_{w,b,\boldsymbol{\xi}} \tfrac{1}{2}\|w\|^2 + C\sum_i \xi_i, \\ &\text{s.t.} \quad y_i\left(\langle w, \phi(x_i)\rangle + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0. \end{aligned}$$

We see that the feature-mapped data $\phi(x_i)$ only ever appears in an inner product. Secondly, the representer theorem (Kimeldorf and Wahba, 1971; Yu et al., 2013) states that the hyperplane w that minimizes the objective function can be expressed as:

$$w = \sum_{i=1}^{n} \alpha_i \phi(x_i),$$

a linear combination of the dataset under the feature map (Schölkopf et al., 2001). Therefore, an equivalent optimization program is:

$$\begin{aligned} &\min_{w,b,\boldsymbol{\xi}} \tfrac{1}{2}\sum_{i,j} \alpha_i\alpha_j \langle \phi(x_i), \phi(x_j)\rangle + C\sum_i \xi_i, \\ &\text{s.t.} \quad y_i\left(\sum_j \alpha_j \langle \phi(x_j), \phi(x_i)\rangle + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0. \end{aligned}$$

By replacing the inner products with kernel functions, a *kernelized* version of the

[2]Strictly speaking, this k corresponds to the map $(x_1, x_2) \mapsto (x_1^2, x_2^2, \sqrt{2}x_1 \times x_2)$.

soft-margin SVM is obtained:

$$\begin{aligned} &\min_{w,b,\boldsymbol{\xi}} \tfrac{1}{2} \sum_{i,j} \alpha_i \alpha_j k(x_i, x_j) + C \sum_i \xi_i, \\ &\text{s.t.} \quad y_i \left(\sum_j \alpha_j k(x_j, x_i) + b \right) \geq 1 - \xi_i,\ \xi_i \geq 0. \end{aligned} \tag{2.3}$$

In practice, the Lagrangian dual of the SVM QP is optimized, which naturally results in a form that can be kernelized. Though a kernel function might seem to be little more than a convenient notation, the kernel trick that follows from the combination of observations above allows us to train classifiers in high-dimensional features spaces efficiently without explicitly performing feature maps of the original data.

2.2.3 Multiple-Instance SVMs

As with other supervised classifiers, SVMs have also been applied to the MI setting. First, as described in this section, one can modify the SVM optimization program (Equation 2.2) to accommodate data in the form of labeled bags. On the other hand, in addition to providing a means of learning nonlinear concepts over an input space, kernels also allow SVMs to be applied to input data that is not in a traditional vector representation. As long as a symmetric kernel function satisfies Mercer's condition, it can be defined with respect to any input space $\mathcal{X}$. Accordingly, kernels have been defined on sets, strings, tree, graphs, and other objects (Gärtner, 2008). Thus, as described in Section 2.3.1, one can define kernels on bags and use the standard SVM formulation.

As described in Section 2.1.2, there are two labeling functions of interest in the MI classification setting. The first is f, the instance-labeling function, and the other is F, the bag-labeling function. "Instance-based" approaches search over instance-labeling functions and modify the SVM formulation to minimize loss with respect to either the instance-labeling function itself, or the corresponding bag-labeling function. One of

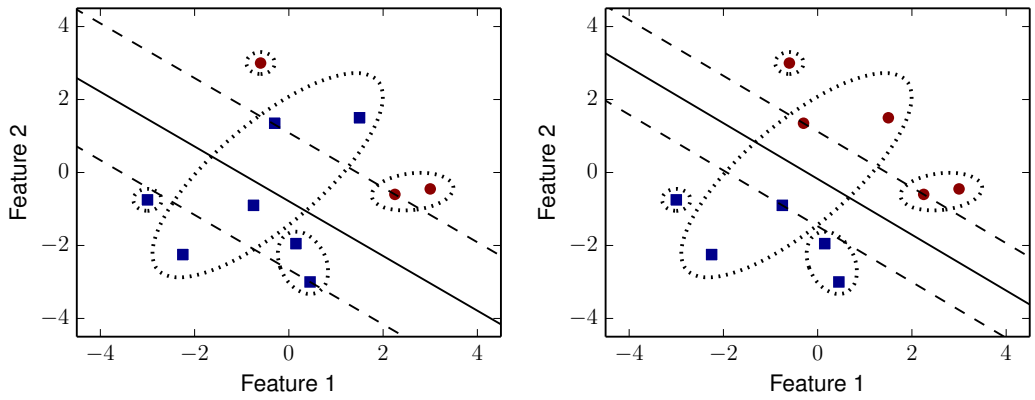


Figure 2.5: Example classifiers found by MI-SVM (**left**) and mi-SVM (**right**). Positive bag instances are represented with blue squares and negative bag instances with red circles. Each bag is indicated with dotted lines. The figure indicates the instance labels inferred during classification.

the first and most straightforward examples is the MI-SVM of Andrews et al. (2003):

$$\begin{aligned} &\min_{w,b,\boldsymbol{\xi}} \tfrac{1}{2}\|w\|^2 + C\sum_i \xi_i, \\ &\text{s.t.} \quad Y_i \left[\max_{x_{ij}\in B_i} \left(\langle w, \phi(x_{ij})\rangle + b\right)\right] \geq 1 - \xi_i,\ \xi_i \geq 0. \end{aligned} \tag{2.4}$$

In the MI-SVM formulation, the maximum in the constraint encodes the relationship $F(B_i) = \max_{x_{ij}\in B_i} f(x_{ij})$ between the bag- and instance-labeling functions, and loss is measured with respect to F given the bag labels Y_i.

Figure 2.5 (left) shows an example MI-SVM classifier on a small MI dataset. Bags are indicated with dotted lines enclosing a set of instances, with positive bag instances represented as blue squares and negative bag instances as red circles. As in the standard SVM formulation, MI-SVM attempts to maximize the hyperplane margin while classifying at least one instance in each positive bag positive, and every instance in each negative bag negative.

An alternative approach is to directly record the loss of f while allowing all possible instance labelings consistent with the given bag labeling. The mi-SVM formulation (Andrews et al., 2003) accomplishes this using a mixed-integer optimization program

Table 2.3: Some extensions of SVM-based approaches to the MI setting.

Algorithm	Reference	Description
	Andrews et al. (2003)	Modify SVM constraints or introduce additional variables to encode the MI assumption
MICA	Mangasarian and Wild (2008)	A convex combination of instances from each positive bag is used to represent the true positive instance
KI-SVM	Li et al. (2009)	Uses multiple kernel learning, in which a convex combination of various kernel functions is also learned during the optimization of the SVM
sMIL stMIL sbMIL	Bunescu and Mooney (2007)	Enforces a constraint on the average label assigned to an instance in each positive bag, with two versions of the algorithm enforcing additional constraints

that searches over instance labels:

$$\begin{aligned} &\min_{w,b,\boldsymbol{\xi},\mathbf{y}} \tfrac{1}{2}\|w\|^2 + C\sum_{i,j}\xi_{ij}, \\ &\text{s.t.} \quad y_{ij}\left(\langle w, \phi(x_{ij})\rangle + b\right) \geq 1 - \xi_{ij},\ \xi_{ij} \geq 0, \\ &\qquad y_{ij} \in \{-1, 1\},\ \begin{cases} \sum_j \frac{y_{ij}+1}{2} \geq 1 & \text{if } Y_i = 1 \\ y_{ij} = -1 & \text{if } Y_i = -1 \end{cases}. \end{aligned} \tag{2.5}$$

The second set of constraints enforce the consistency between Y_i and y_{ij}.

Figure 2.5 (right) shows an example classifier found by mi-SVM. The resulting hyperplane differs from that found with MI-SVM, since mi-SVM enforces that each instance has a label. The resulting instance labels inferred by mi-SVM are indicated in the figure.

In addition to MI-SVM and mi-SVM, numerous other extensions of the SVM to the MI setting have been proposed, as described in Table 2.3. Although such approaches attempt to learn bag-level labels using instance-level classifiers, better performance can often be achieved by directly learning bag-level classifiers. We make the case for this claim in Chapter 6. One reason why this is true is because even negative instances in positive bags can provide useful discriminative information to a

bag-level MI classifier. As an example, consider an image classification task in which instances are objects within an image, and bags are entire images containing sets of objects. The instance-labeling task might be to classify whether objects are cats, and the corresponding bag-labeling task is to classify whether an image contains a cat. Since images of cats might be more likely to contain objects such as toy mice or balls of yarn, having access to the entire set of objects in the image might afford an advantage to a classifier over one that looks only at each individual object to make a decision about the image. The intuition described above is formalized by the generative model described in Chapter 3. The set kernel approaches, described in Section 2.3.1, can be used to capture aggregate information about bags.

2.3 Kernel Embeddings of Sets and Distributions

When kernels are used with structured objects such as sets, the corresponding feature map is often described as *embedding* structured examples into a feature space in which a hyperplane-based model can be trained. Such embeddings provide an elegant means of representing structured data as feature vectors without requiring explicit enumeration of the features. The following sections describe two relevant embeddings of sets and distributions into feature spaces.

2.3.1 Set Kernels

The set kernel is a straightforward way of embedding an entire set into a kernel feature space by simply summing feature representations for each element of the set. Figure 2.6 shows an example of the set kernel feature map Φ applied to three sets. If a linear *instance kernel* k_{I} with feature map $\phi(x) = x$ is used as in the example to represent each element of a set $X_1 = \{x_i\}_{i=1}^{m}$, the set kernel feature map is defined

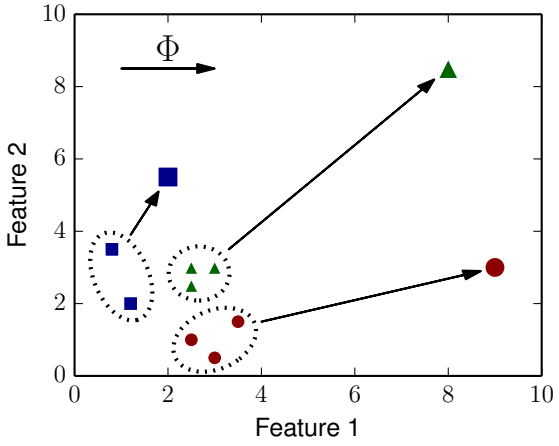


Figure 2.6: An example showing the set kernel feature representations (under the set kernel feature map Φ) for three bags with a linear instance kernel.

formally as:

$$\Phi(X_1) = \sum_{i=1}^{m} \phi(x_i).$$

Then, given another set, $X_2 = \{x_j\}_{j=1}^{n}$, the set kernel k_{Set} is defined in terms of the feature map as:

$$\begin{aligned} k_{\text{Set}}(X_1, X_2) &= \langle \Phi(X_1), \Phi(X_2) \rangle \\ &= \left\langle \sum_{i=1}^{m} \phi(x_i), \sum_{j=1}^{n} \phi(x_j) \right\rangle \\ &= \sum_{i=1}^{m} \sum_{j=1}^{n} \langle \phi(x_i), \phi(x_j) \rangle \\ &= \sum_{i=1}^{m} \sum_{j=1}^{n} k_{\text{I}}(x_i, x_j). \end{aligned} \tag{2.6}$$

Thus, the set kernel is simply the sum of pairwise instance kernel values between elements in the two sets (Gärtner et al., 2002).

Observe that for the set kernel in Equation 2.6, bags with different sizes will have different feature space representations. For example, the bag of blue squares in Figure 2.6 only contains two examples, and its representation is closer to the origin than the other bags, which contain three examples. However, for MIL, the bag size is not a relevant feature for learning to classify bags. In this case, we might wish to *normalize* the set kernel feature representation.

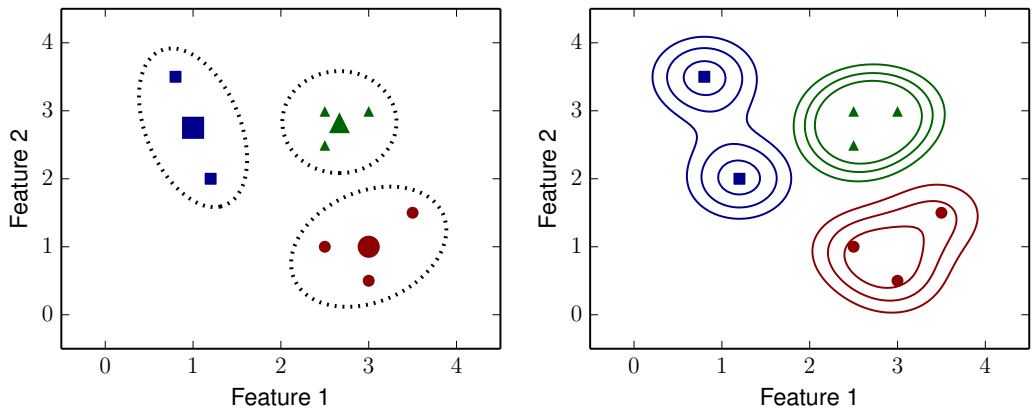


Figure 2.7: (**Left**) Three bags represented using an averaging-normalized set kernel with a linear instance kernel. The large points illustrate the feature space representations for each bag. (**Right**) The same three bags represented using the averaging-normalized set kernel with an RBF instance kernel. The feature space representations of the bags are functions, which are represented with contours.

One possible normalization is to average rather than sum examples within each set. Figure 2.7 (left) shows the same sets from Figure 2.6 under the averaging-normalized set kernel map:

$$\Phi(X_1) = \frac{1}{|X_1|} \sum_{i=1}^{m} \phi(x_i). \tag{2.7}$$

In general, the set kernel can also be normalized with a positive-valued normalization function f_{norm} applied to each bag. Using a derivation similar to that in Equation 2.6, the normalized set kernel (NSK) derived from the feature map is:

$$k_{\text{NSK}}(X_1, X_2) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} k_{\text{I}}(x_i, x_j)}{f_{\text{norm}}(X_1) f_{\text{norm}}(X_2)}. \tag{2.8}$$

The averaging-normalized set kernel is a special case of the kernel in Equation 2.8 with $f_{\text{norm}}(X) = |X|$. As described below, using averaging normalization produces a kernel with some interesting properties. Furthermore, empirical results show that the NSK with averaging normalization led to improve performance on the MI classification task compared with the unnormalized set kernel in Equation 2.6 (Gärtner et al., 2002).

In the examples of Figure 2.6 and Figure 2.7 (left), a simple linear instance kernel

is used to construct a kernel on sets. As a result, the feature space representations of these sets (either a sum or average of the instances) appear to "ignore" much of the structure of the sets. However, using a more powerful instance kernel, this need not be the case. For example, the radial basis function (RBF) kernel is defined as follows:

$$k(x_i, x_j) = e^{-\gamma \|x_i - x_j\|_2^2}, \tag{2.9}$$

where γ is a *bandwidth* parameter. The feature map of the RBF kernel maps an example to an unnormalized Gaussian function centered over the example with width controlled by γ. The NSK with an RBF instance kernel thus represents a set with an average of these Gaussian functions. The result is shown Figure 2.7 (right), where contours are used to show the functions corresponding to the feature space representations of the set. Section 7.2.2 describes the properties of the NSK in more detail. Section 2.3.2 describes the relationship between the NSK and a recent approach for defining kernels distributions.

2.3.2 Kernel Mean Embeddings

The averaging normalization applied to the set kernel as shown in Figure 2.6 was motivated by a desire to prevent bag size from contributing significantly to the feature space representation of a bag. However, the NSK can also be viewed as a way to embed *distributions* into a kernel feature space.

The example in Figure 2.8 gives the intuition for how the NSK embeds a distribution into a feature space. With an RBF instance kernel (Equation 2.9), the feature space representation of a single point can be viewed as an unnormalized Gaussian function centered over the point. This is represented as the point x_i being mapped from the one-dimensional instance space on the left-hand side of Figure 2.8 to $\phi(x_i)$ on the right-hand side.

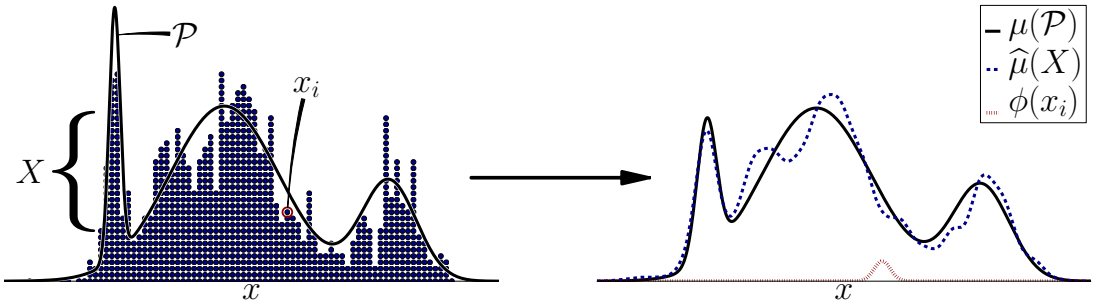


Figure 2.8: An example illustrating the embedding of a single example, a sample, and a distribution using the RBF base kernel. On the left is the actual PDF of a one-dimensional instance distribution with a sample plotted as a histogram. The right shows how the kernel embeds a single example x_i, the sample X, and the underlying distribution $\mathcal{P}$ into a space of functions.

Now, suppose we draw a sample X of size 1000 with each element independent and identically distributed (IID) according to the distribution $\mathcal{P}$ over the instance space. To visualize the sample in Figure 2.8, the elements are plotted in bins as a histogram. The NSK with averaging normalization (Equation 2.7) embeds this sample as the average of the individual feature space representations of each point. In the case of the RBF kernel, this is an average of Gaussian functions of the form shown as $\phi(x_i)$ in Figure 2.8. The resulting averaged function is the dashed blue line on the right-hand side of Figure 2.8. In this context, the feature space map of the NSK is called the empirical *kernel mean embedding* of the sample X, denoted $\widehat{\mu}(X)$ (Smola et al., 2007).

Now, suppose we draw ever-larger samples from $\mathcal{P}$ and compute their empirical kernel mean embeddings. We would expect that as sample size $n \to \infty$, the distance $\left\|\widehat{\mu}(X^{(n)}) - \mu(\mathcal{P})\right\|_{\mathcal{H}} \to 0$, where $\mu(\mathcal{P})$ is the kernel mean embedding computed with respect to the underlying distribution, $\mathcal{P}$:

$$\mu(\mathcal{P}) = \mathrm{E}_{x \sim \mathcal{P}} \left[\phi(x)\right].$$

Under some mild assumptions, the embedding of samples *do* converge in the feature space norm to the embedding of the underlying distribution Sriperumbudur et al.

(2010). Figure 2.8 shows that $\mu(\mathcal{P})$ and $\widehat{\mu}(X)$ are in fact close to each other after 1000 samples.

Thus, the averaging-normalized set kernel can be viewed as a way of embedding *distributions* into a feature space. When bags in the MI setting are viewed as samples from distributions, as in the generative model described in Chapter 3, the NSK is a natural way of representing bags in a feature space.

The kernel mean embedding has several advantages over other approaches for representing distributions; see Song et al. (2013) for a more detailed discussion. First, the mean embedding is nonparametric and does not require strong assumptions about the structure of the distribution being represented. As opposed to other finite or nonparametric mixture models that have been used to model distributions, the mean embedding does not require a computationally intensive optimization process. The mean embedding has some similarities to density estimation techniques. However, the mean embedding is much more effective at overcoming the "curse of dimensionality" encountered by density estimation approaches as the dimensionality of the input space increases (Gretton et al., 2006). In fact, the appropriate choice of kernel can achieve fast convergence of the empirical mean embedding $\widehat{\mu}$ to the underlying mean embedding μ with a rate that is independent of the input data dimensionality (Sriperumbudur et al., 2010, 2012). Finally, embedding distributions into a Hilbert space allows a variety of existing tools from linear algebra to be applied to perform operations on distributions. In particular, as described in Section 6.1, classification algorithms such as SVMs can be applied to learn from distributions.

2.3.3 Related Kernels

Although we are primarily interested in kernels that map distributions into a feature space, other kernels have been defined to act on entire bags. In particular, the "marginalized" set kernel (Kwok and Cheung, 2007) is a modification of the set kernel

designed to take instance labels into account. The box-counting kernel (Tao et al., 2008) counts the number of APRs (here, treated as boxes in some discretized version of the input space) that contain some instance from each bag. Because enumerating all boxes is computationally intractable, the authors present an approximation scheme. The mi-Graph and MI-Graph kernels construct a nearest-neighbor graph between instances in each bag, then define a kernel between these graphs (Zhou et al., 2009). However, no theoretical results about the representational power of these kernels are provided in prior work. In Section 6.3, we analyze these approaches in comparison to kernels that explicitly treat bags as distributions.

2.4 Learning Theory

As discussed in Section 2.1, the goal of supervised and other learning frameworks is to find a "good" function to approximate some target concept. Although there are many conceivable ways to address this question, Section 2.4.1 contains a discussion of the probably approximately correct (PAC) learning framework of Valiant (1984) and related concepts. PAC learning is a popular and well-studied formalization of what it means to "learn" from data and provides a theoretical framework for rigorously describing the ability of specific algorithms to learn concepts efficiently. Chapter 4 describes new learnability results for the MI setting using the PAC framework.

2.4.1 Probably Approximately Correct

To formalize what it means to learn from data, we must make further assumptions than were discussed in Section 2.1. In this section, we focus on the binary classification problem ($\mathcal{Y} = \{-1, 1\}$), but the ideas presented below can be stated in more general terms. First, we assume that the concept $f : \mathcal{X} \to \mathcal{Y}$ is an element of some fixed, known concept class $\mathcal{F}$. For example, $\mathcal{F}$ might be the set of all thresholded real-

valued (to produce a binary label) linear, quadratic, or bounded continuous functions on $\mathcal{X}$. Furthermore, we assume that there exists some distribution $D_{\mathcal{X}}$ over examples $\mathcal{X}$. When we observe a sample $X^{(n)} = \{(x_i, f(x_i)\}_{i=1}^n$, we assume that each x_i is drawn IID according to $D_{\mathcal{X}}$. When we observe new examples to label, these examples must also be drawn from $D_{\mathcal{X}}$; otherwise, our training examples might be completely unrelated to those we observe during the "testing" of the learned classifier.

Given these assumptions, we can describe the quality of a selected concept g with respect to the target concept $f \in \mathcal{F}$. Quality is measured using *risk* R, which measures the expected 0–1 loss, or discrepancy between f and g, with respect to the $D_{\mathcal{X}}$:

$$R_f(g) = \mathrm{E}_{x \sim D_{\mathcal{X}}} \left[\mathbb{1}[f(x) \neq g(x)] \right]. \tag{2.10}$$

Here, $\mathbb{1}[\cdot]$ denotes the indicator function, which is equal to 1 when the condition inside the brackets is satisfied and 0 otherwise. As described in Chapter 5, it is possible to substitute other measures such as area under the ROC curve (AUC) for 0–1 loss in the PAC framework. A concept g is *approximately correct* when $R_f(g) < \epsilon$ for some small $\epsilon > 0$. While we hope to learn from a finite sample $X^{(n)}$ of data, we are at the mercy of chance to provide a sample that is representative of the underlying distribution $D_{\mathcal{X}}$. Therefore, as the name suggests, the PAC framework requires that learning algorithms produce concepts that are only *probably* approximately correct. That is, given a concept g produced by training on input data $X^{(n)}$:

$$\mathrm{P}_{X^{(n)} \sim D_{\mathcal{X}}^n} \left[R_f(g) < \epsilon \right] > 1 - \delta,$$

for some small $\delta > 0$. Finally, PAC learning requires that the sample size n need not be "too large" to produce a good classifier. With the basic requirements for PAC learnability described above, the formal definition is as follows:

Definition 2.1 (PAC Learning)**.** *An algorithm $\mathcal{A}$ PAC-learns a concept class $\mathcal{F}$ over*

instances $\mathcal{X}$ if for any $f \in \mathcal{F}$, $D_{\mathcal{X}}$ over $\mathcal{X}$, and $\epsilon, \delta > 0$, given a sample $X^{(n)}$ drawn from $D_{\mathcal{X}}^n$ of size $n = O(\mathsf{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}))$, $\mathcal{A}$ produces a concept g such that $\mathrm{P}\left[R_f(g) < \epsilon\right] > 1 - \delta$.

Above, $\mathsf{poly}(\cdot)$ represents a polynomial in terms of its arguments. The notion of computational complexity can also be incorporated into PAC learnability:

Definition 2.2 (Efficiently PAC Learning)**.** *An algorithm $\mathcal{A}$ efficently PAC-learns a concept class $\mathcal{F}$ if it PAC learns $\mathcal{F}$ with a runtime that is $O(\mathsf{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}))$.*

Interestingly, algorithm-independent statements can be made about the PAC learnability of various concept classes using only properties of the concept class itself. Such results employ general approaches like empirical risk minimization (ERM), which entails selecting a function $g \in \mathcal{F}$ that minimizes the empirical risk, or loss, of g on the observed dataset. The next section describes properties of a concept class that make learning with approaches like ERM possible.

2.4.2 Capacity Measures

One useful property for characterizing the learnability of a concept class is a general measure of the size, or *capacity* of the class. As a simple example, consider the class of boolean functions of two variables. Since there are only $2^{2^2} = 16$ such functions, with high probability, a small training sample will be able to rule out irrelevant functions that lead to errors on the sample. More general and precise statements about learnability in the case of a finite concept class have been derived (Blumer et al., 1987). However, we are often interested in infinite concept classes such as hyperplanes. In this case, a more general notion of capacity is required.

One useful notion of capacity is the Vapnik–Chervonenkis (VC) dimension (Vapnik and Chervonenkis, 1971). The intuition behind the VC dimension is that even for infinite concept classes, there are only at most 2^n equivalence classes of functions for

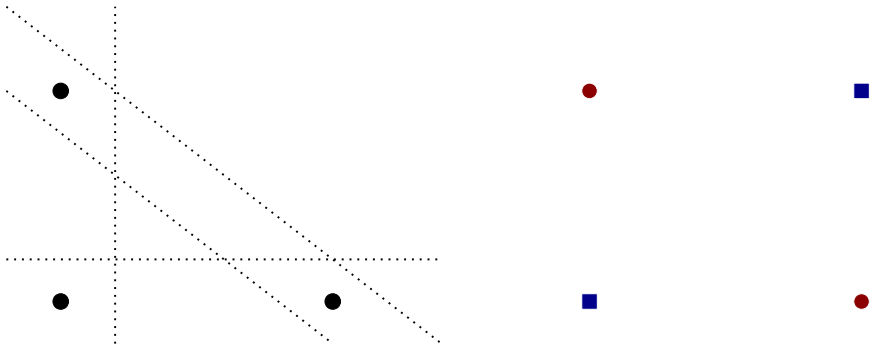


Figure 2.9: An example showing the shattering of points with two-dimensional hyperplanes. Any set of three points (**left**) can be shattered, but no hyperplane can assign the shown labeling of the given set of four points (**right**), or the appropriately-chosen labeling of any other set of four points.

any set of n examples. That is, each binary function in a concept class assigns one of 2^n possible labelings to the examples. A concept class $\mathcal{F}$ *shatters* a sets of size n there exist functions in $\mathcal{F}$ corresponding to each of the 2^n equivalence classes. That is, any of the 2^n labelings can be realized with some function in $\mathcal{F}$, for the set of n examples. The VC dimension of $\mathcal{F}$, denoted VC($\mathcal{F}$), is the size of the largest set that $\mathcal{F}$ can shatter *some* set of that size. If $\mathcal{F}$ shatters sets of any size, then VC($\mathcal{F}$) $= \infty$.

As an example, consider the concept class of two-dimensional hyperplanes. As illustrated in Figure 2.9, any set of three points can be shattered. The figure shows four hyperplanes, which with their "flipped" counterparts assign the eight possible labelings to the set of three points. However, the right-hand side of Figure 2.9 shows a labeling of a set of four points that cannot be assigned by a linear concept. In fact, for all sets of four points, there exists a labeling that cannot be assigned by a two-dimensional hyperplane. Therefore, the size of the largest set that can be shattered by two-dimensional hyperplanes (and hence the VC dimension of the concept class) is three. More generally, the VC dimension of k-dimensional hyperplanes is $k + 1$ (Vapnik and Chervonenkis, 1971).

The importance of VC dimension for learning theory is described in a result by

Blumer et al. (1989). In particular, a concept class is PAC learnable if and only if it has finite VC dimension. In particular, it is PAC learnable using:

$$n = O\left(\frac{1}{\epsilon}\left(\mathrm{VC}(\mathcal{F})\log\frac{1}{\epsilon} + \log\frac{1}{\delta}\right)\right)$$

examples (Blumer et al., 1989).

As a final note, in the SVM formulation of Equation 2.3, the concept class used for learning is potentially an infinite-dimensional Hilbert space with infinite VC dimension. However, by maximizing the margin of the selected hyperplane in addition to minimizing empirical risk, the SVM effectively restricts the capacity of the concept class it uses to learn a target concept. Simultaneously controlling the capacity of the hypothesis class while minimizing risk is called structural risk minimization (SRM). Similar learnability results can be shown for SRM approaches as well.

2.4.3 Probabilistic Concepts

As discussed in Section 2.1, there exist alternative formalizations of supervised learning in which a labeling concept $f : \mathcal{X} \to \mathcal{Y}$ need not be deterministic. In particular, the probabilistic concept (p-concept) model is a model for binary classification in which a p-concept $c : \mathcal{X} \to [0, 1]$ represents the probability that an instance $\mathcal{X}$ is observed with a positive label (Kearns and Schapire, 1994). In this setting, each example x is sampled according to $D_{\mathcal{X}}$, and the label of x is positive with probability $c(x)$ and negative with probability $1 - c(x)$.

In the case of p-concepts, risk of a hypothesis h is more naturally measured with respect to a *quadratic* loss function: $(h(x) - c(x))^2$. Risk is the expected loss, defined by:

$$R_c(h) = \mathrm{E}_{x\sim D_{\mathcal{X}}}\left[(h(x) - c(x))^2\right].$$

Using this measure of risk, PAC learnability of a p-concept class $\mathcal{C}$ can be defined for

the p-concept model analogously to the definition for a deterministic concept class $\mathcal{F}$ (Kearns and Schapire, 1994).

Just as VC dimension can be used to characterize the capacity of a deterministic concept class, pseudo-dimension is used to quantify the capacity of a p-concept class. The pseudo-dimension is similar to the VC dimension, but uses a different notion of "shattering." In particular, for a set of points with real-valued labels, $\{(x_i, y_i)\}_{i=1}^{n}$, a p-concept class $\mathcal{C}$ shatters the points if for any binary labeling of the points $\{b_i\}$, there exists some $c \in \mathcal{C}$ such that $c(x_i) \geq y_i$ if $b_i = 1$ and $c(x_i) < y_i$ if $b_i = 0$ (Haussler, 1992). The pseudo-dimension of $\mathcal{C}$, denoted $\mathrm{PD}(\mathcal{C})$, is the size of the largest set such that $\mathcal{C}$ shatters some set of that size.

As in the relationship between PAC learnability and VC dimension described in Section 2.4.2, it is possible to PAC-learn a p-concept class $\mathcal{C}$ when $\mathrm{PD}(\mathcal{C})$ is finite using:

$$n = O\left(\frac{1}{\epsilon^2}\left(\mathrm{PD}(\mathcal{C}) \log \frac{1}{\epsilon} + \log \frac{1}{\delta}\right)\right) \tag{2.11}$$

examples with binary labels assigned according to the p-concept (Kearns and Schapire, 1994). Chapter 5 describes how the PAC learnability of p-concepts can be used to show the ability of supervised classifier to correctly rank positive and negative instances in the MIL setting.

2.4.4 Area Under the Receiver Operating Characteristic Curve

For classifiers that output a confidence value $y_i \in [0, 1]$ that an example x_i is positive, 0.5 is typically used as the threshold for assigning a binary label to the example. That is, the maximum likelihood label given the classifier's confidence is 1 if the confidence is above 0.5 and 0 if the confidence is below 0.5. However, in practice, there might be a high cost associated with a false positive, so that the threshold should be set higher than 0.5. In other cases, there might be a high cost associated with a false negative, so

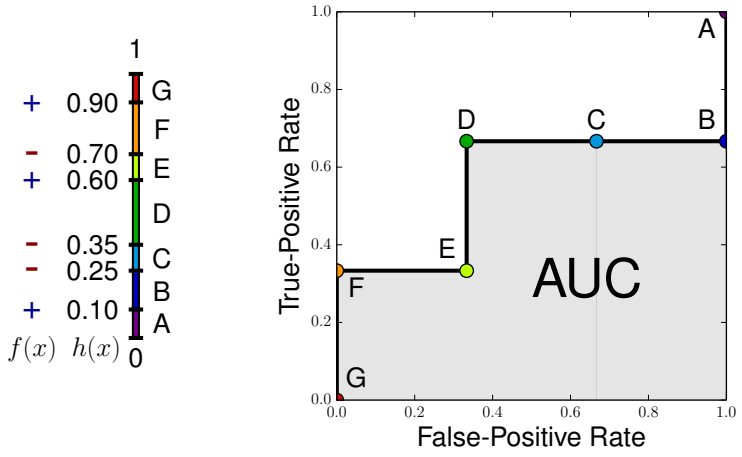


Figure 2.10: An example ROC curve. The left-hand side of the figure shows a set of examples with true confidence labels assigned by f and confidence values assigned by h. Equivalent sets of thresholds are indicated with letters, and the corresponding false-positive rates and true-positive rates are shown as the ROC curve on the right. The AUC is indicated by the shaded region.

that the threshold should be set lower than 0.5. The receiver operating characteristic (ROC) curve plots this trade-off between the false-positive and true-positive rates as the classifier threshold is varied.

An example ROC curve is shown in Figure 2.10. On the left-hand side of the figure are the class labels $f(x)$ and confidence values $h(x)$ assigned to a set of examples by the underlying labeling function f and classifier h. The threshold for binary classification can be set within the regions labeled "A" through "G." The resulting ROC curve is obtained by plotting the false-positive rate and true-positive rate corresponding to setting a threshold within each of these regions, as illustrated on the right-hand side of the figure.

For a perfect classifier, there is some threshold that correctly labels all examples, which achieves a 100% true-positive rate with a 0% false-positive rate. Accordingly, the ROC curve jumps immediately to 1 at a false-positive rate of 0. In general, the better a classifier, the more quickly the corresponding ROC curve increases to 1. Hence, the AUC is used to summarize the performance of a classifier. The AUC of

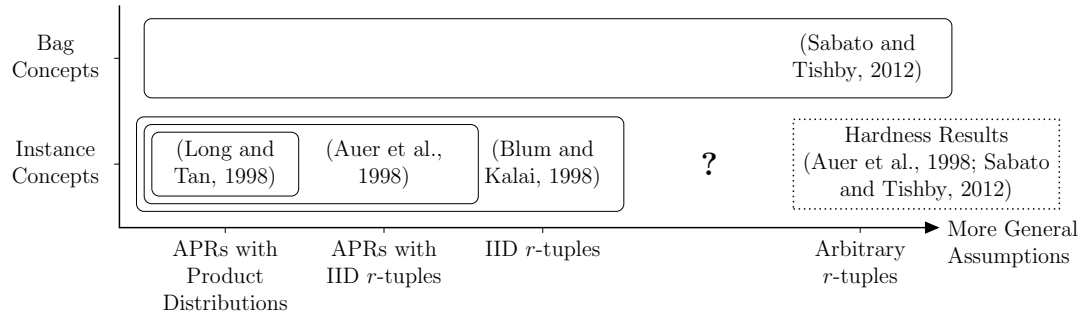


Figure 2.11: Prior learnability and hardness results in the MI setting.

the curve in Figure 2.10 is shown as a shaded region.

2.4.5 Multiple-Instance Learnability and Hardness Results

Figure 2.11 provides an overview of the prior learnability and hardness results in the MI setting. Recent results, described in more detail in Section 4.4, show that it is possible to learn bag concepts using ERM under relatively weak assumptions (Sabato and Tishby, 2012). However, the results do not provide any theoretical guarantees about learning instance concepts. The most recent learnability results for instance concepts, over a decade old, rely on the restrictive assumption that all instances across all bags are drawn IID from the same distribution over instances (Blum and Kalai, 1998). As discussed in prior work (Zhou et al., 2009), and in Section 3.3, this assumption is unrealistic for many practical MI problems. On the other hand, allowing for arbitrary distributions over tuples of instances leads to hardness results for instance concept learnability (Auer et al., 1998; Sabato and Tishby, 2012). However, these prior results leave open the question of whether instance concept learnability is possible for models that are less restrictive than the IID r-tuple model, but provide other weaker restrictions on the distribution over tuples. In Chapter 4, we answer this question in the affirmative. In particular, we show that under a generative model in which bags are viewed as distributions (Chapter 3), it is possible to generalize prior results on instance concept learnability (Blum and Kalai, 1998) while precluding hard

scenarios described in prior work (Auer et al., 1998; Sabato and Tishby, 2012). A detailed comparison with prior work is given in Section 4.4.

Chapter 3

Bags as Distributions

In this chapter, we describe a generative model for MI data in which bags are viewed as *distributions over instances* rather than as sets of instances.[1] We show that the proposed generative model actually encompasses previous, standard models of MI learning in which bags are sets or tuples. The choice of framing a problem within a particular theoretical model has significant practical consequences for designing or selecting an algorithm to solve the problem. This chapter provides a novel theoretical framework in which the MI classification problem can be analyzed. The model allows us to derive positive instance- and bag-concept learnability results for the MI setting as described in Chapter 4. Furthermore, as Chapter 5 shows, the generative model leads to a surprising yet testable hypothesis that standard supervised algorithms can learn from MI data. This hypothesis is evaluated experimentally, supporting the assumptions made by the model. In Chapter 6, existing bag-level hyperplane classifiers are analyzed under the generative model, and in Chapter 8, the model is used to analyze a new resampling method for MIL.

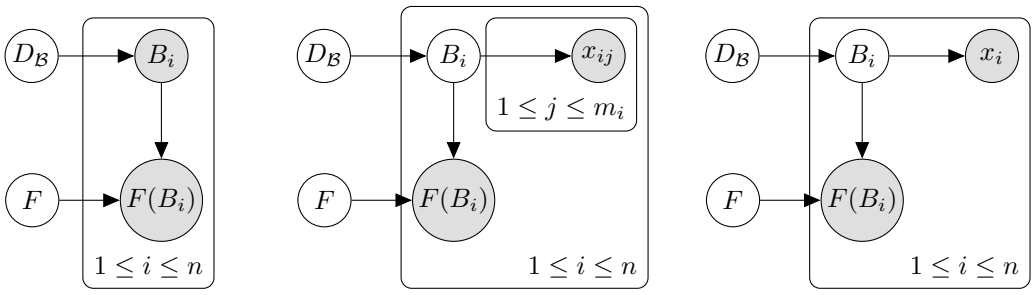
[1] A description of this new generative model also appears in Doran and Ray (2014).

3.1 The Generative Model

At the heart of this work is the claim that bags are best viewed as distributions rather than as finite sets of instances. Below, we formally define what we mean by this statement. But first, the example domain of drug activity prediction provides an intuitive justification for this claim.

As described in Chapter 1, in the drug activity prediction domain, the goal is to predict the ability of molecules to activate, or bind to, a receptor. To cast the problem as binary classification, we select some threshold so that each molecule's activity level either corresponds to an "active" or "inactive" label. In this case, we can think of each molecule (bag) as being drawn from a distribution $D_{\mathcal{B}}$ over molecules. Ignoring for the moment that each molecule has numerous conformations, this molecule either activates the receptor or not, so in nature the labeling function is defined at the level of bags. Prior models represent each molecule as a set or multiset of conformations, so they implicitly assume that each molecule exists in only a finite number of conformations. In reality, a molecule can transform continuously from conformation to conformation, producing an infinite set of conformations. In particular, each molecule exists in a state of dynamic equilibrium in which the amount of time it spends in each conformation is distributed according to Gibbs free energy such that low-energy conformations are preferred. Hence, the molecule (bag) corresponds to a *distribution over instances*. Constructing a bag from low-energy conformations, the common procedure for constructing bags in the drug activity domain, can be thought of as sampling instances from this distribution.

To formalize the intuition above, suppose we have an instance space $\mathcal{X}$. Typically, the space of bags is some subset of $\mathcal{X}^*$, the set of all finite subsets of $\mathcal{X}$. However, here, we let the space of bags be $\mathcal{B} = \mathscr{P}(\mathcal{X})$, the set of probability distributions on the input space. Hence, each bag $B \in \mathcal{B}$ is a probability distribution over instances, denoted $\mathrm{P}(x \mid B)$.



(a) Generative Process for Bags

(b) Generative Process for Bag Samples

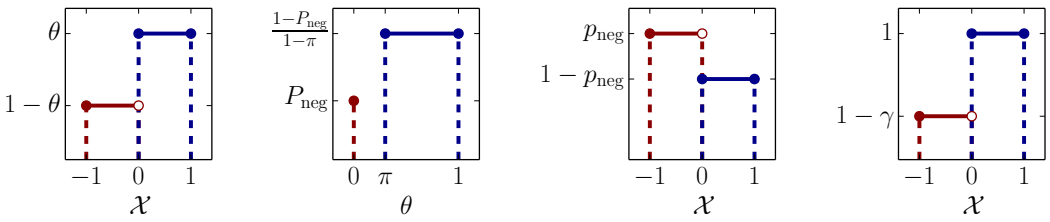
(c) Generative Process for Bag-labeled Instances

Figure 3.1: A comparison of the generative processes for bags, bag samples, and individual bag-labeled instances.

As a concrete example, suppose the instance space is the closed real-valued interval $\mathcal{X} = [-1, 1]$ and each bag B_θ is a distribution parametrized by a single real-valued parameter $\theta \in [0, 1]$. As illustrated in Figure 3.2(a), the bag distribution $\mathrm{P}(x \mid B_\theta)$ assigns $(1 - \theta)$ of the probability mass uniformly to the interval $[-1, 0)$, and θ of the mass uniformly to the interval $[0, 1]$. Each value of θ corresponds to a different bag, which is a different distribution over instances.

We propose that, at the level of bags, the MI generative process is similar to that for supervised learning. In particular, bags are sampled from some fixed distribution $D_{\mathcal{B}}$, which is a distribution over instance distributions ($D_{\mathcal{B}} \in \mathscr{P}(\mathscr{P}(\mathcal{X}))$). From this distribution $D_{\mathcal{B}}$, we sample some set of bags $\{B_i\}_{i=1}^n$, as illustrated by the plate model in Figure 3.1(a). Also, as in supervised learning, we assume that there exists some labeling function $F : \mathcal{B} \to \{0, 1\}$ that labels bags. Thus, a supervised dataset $\{(B_i, F(B_i))\}_{i=1}^n$ could be produced by sampling bags IID from $D_{\mathcal{B}}$ and applying the labeling function F.

Returning to the example of Figure 3.2, a distribution over bags is essentially a distribution over the bag parameter θ. Such a distribution is illustrated in Figure 3.2(b), and assigns P_{neg} of the mass to the set $\{0\}$ and the remaining $1 - P_{\text{neg}}$ portion of the mass uniformly to the interval $[\pi, 1]$. The probability of sampling a bag, $\mathrm{P}(B_\theta)$, corresponds to the probability of sampling the corresponding value of θ.



(a) $\mathrm{P}(x \mid B_\theta)$ (b) $D_{\mathcal{B}}$, $\mathrm{P}(B_\theta)$ (c) $D_{\mathcal{X}}$, $\mathrm{P}(x)$ (d) $c(x)$

Figure 3.2: An example generative process for MI data. Each bag distribution (a) is parametrized by θ, and the distribution over bags (b) corresponds to a distribution over values of θ. The resulting distribution over instance (c) is derived in Equation 3.3 and Equation 3.4. The probability of instances appearing in positive bags (d) is derived in Equation 3.6 and Equation 3.7.

Similarly, a bag-labeling function F can be defined in terms of θ as follows:

$$F(B_\theta) = \begin{cases} 0 & \text{if } \theta = 0 \\ 1 & \text{if } \theta > 0. \end{cases} \tag{3.1}$$

Thus, for this example, $P_{\text{neg}} = \mathrm{P}\left[F(B_\theta) = 0\right]$.

However, unlike the example in Figure 3.2 or the plate model in Figure 3.1(a), we do not typically observe bags directly in the MI setting. In the typical case, we only have access to samples $X_i = \{x_{ij}\}_{j=1}^{m_i}$, each drawn independently according to the distribution corresponding to each bag B_i so that $\left\{\left(\{x_{ij}\}_{j=1}^{m_i}, F(B_i)\right)\right\}_{i=1}^{n}$ is the observed MI dataset, as shown in Figure 3.1(b). Each bag can be a different size, but we will use $m_l \leq m_i \leq m_u$ to denote the lower and upper bounds on bag sizes, respectively.

Furthermore, in the MI setting, we assume that in addition to the bag-labeling function F, there also exists an *instance-labeling* function $f : \mathcal{X} \to \{0, 1\}$. A key component of the MI setting is not only the existence of both bag and instance labeling functions, but the relationship between the two as well. Traditionally, the MI assumption is stated with respect to particular sets of instances so that a bag label $F(B_i)$ is the logical `OR` (for boolean labels), or maximum (for numerical labels),

of its instances' labels: $F(B_i) = \max_j f(x_{ij})$. However, in the proposed generative model, bags are distributions with *a priori* labels regardless of the instances sampled from them. Therefore, our generative model requires a more nuanced description of the relationship between bag and instance labels.

In particular, we state the relationship between F and f at the level of the generative model. Accordingly, a bag is negative ($F(B) = 0$) if and only if probability of sampling a positive instance within the bag is zero: $\mathrm{P}_{x \sim B}\left[f(x) = 1\right] = 0$. In measure-theoretic terms, instances sampled within negative bags are almost surely negative, which implies that positive instances are almost surely sampled within positive bags. This condition corresponds to the standard MI assumption that negative bags contain only negative instances.

For the example in Figure 3.2, we define the instance-labeling function to be:

$$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ \\ 1 & \text{if } x \geq 0. \end{cases}$$

This definition is consistent with the bag-labeling function defined in Equation 3.1, since $F(B_\theta) = 0$ implies $\theta = 0$, which implies that the probability of sampling a positively labeled $x \in [0, 1]$ is zero as required.

In order to talk about the learnability of f, we must define some instance distribution with respect to which we will measure risk. An instance distribution naturally arises from our generative model if we first sample a bag B randomly from $D_{\mathcal{B}}$, then sample an instance x randomly from the distribution corresponding to B. The instance distribution $D_{\mathcal{X}}$ resulting from this two-level sampling procedure is effectively the distribution that marginalizes out the individual bag distributions. That is, given a probability distribution $\mathrm{P}_{\mathcal{B}}$ over bags corresponding to $D_{\mathcal{B}}$, we can define a distri-

bution $\mathrm{P}_{\mathcal{X}}$ corresponding to $D_{\mathcal{X}}$ as:

$$\mathrm{P}_{\mathcal{X}}(x) = \int_{\mathcal{B}} \mathrm{P}(x \mid B)\, \mathrm{dP}_{\mathcal{B}}(B). \tag{3.2}$$

Given that "x" is used to denote instances and "B" is used to denote bags, we subsequently drop subscripts from P when the sample space can be inferred from context. As we discuss in Section 4.4, the ability to marginalize out bag-specific distributions in our model plays a vital role in proving the learnability of instance- and bag-labeling functions. Given a bag distribution, the existence of such an instance distribution is guaranteed under relatively weak assumptions on the instance space $\mathcal{X}$ (Diestel and Uhl, 1977).

Returning to the example of Figure 3.2, if we marginalize out the bag distribution, we obtain the single instance distribution in Figure 3.2(c). Analytically, for any $x_- \in [-1, 0)$, we have:

$$\begin{aligned} \mathrm{P}(x_-) &= \int_0^1 \mathrm{P}(x_- \mid B_\theta)\, \mathrm{P}(B_\theta)\, \mathrm{d}\theta \\ &= 1 \cdot P_{\text{neg}} + \int_\pi^1 (1-\theta) \frac{1 - P_{\text{neg}}}{1 - \pi}\, \mathrm{d}\theta \\ &= P_{\text{neg}} + \tfrac{1}{2}(1 - P_{\text{neg}})(1 - \pi) \triangleq p_{\text{neg}}. \end{aligned} \tag{3.3}$$

Similarly, for $x_+ \in [0, 1]$:

$$\begin{aligned} \mathrm{P}(x_+) &= \int_0^1 \mathrm{P}(x_+ \mid B_\theta)\, \mathrm{P}(B_\theta)\, \mathrm{d}\theta \\ &= 0 \cdot P_{\text{neg}} + \int_\pi^1 \theta \frac{1 - P_{\text{neg}}}{1 - \pi}\, \mathrm{d}\theta \\ &= \tfrac{1}{2}(1 - P_{\text{neg}})(1 + \pi) = 1 - p_{\text{neg}}. \end{aligned} \tag{3.4}$$

As is the case in the standard MI framework, in our generative model, only bag labels are observed. Suppose we sample individual instances as illustrated in Fig-

ure 3.1(c) where we first sample a bag, record its label, and then sample an instance from the bag-specific distribution $\mathrm{P}(x \mid B)$ and assign the bag label to the instance. Then the resulting bag-labeled instances $\{(x_{i1}, F(B_i))\}_{i=1}^{n}$ are distributed according to $D_{\mathcal{X}}$, and will appear in positive bags some of the time and negative bags the remaining fraction of the time. Therefore, each instance will have some probability $c(x) \in [0, 1]$ of appearing with a positive label, which can be formally expressed as a probabilistic concept (p-concept):

$$c(x) \triangleq \mathrm{P}\left[F(B) = 1 \mid x\right]. \tag{3.5}$$

That is, the probability of observing a positive label for instance x is the conditional probability that the bag B in the two-level sampling procedure was positive, given that x was observed within B. This conditional probability can be derived from the joint distribution over instances and bag labels corresponding to the generative process in Figure 3.1(c).

It follows from the previously-stated relationship between F and f that for any positive instance x_+, $c(x_+) = 1$, since each positive instance is observed almost surely (with probability 1) within a positive bag. In order to distinguish positive and negative instances, we make the following weak assumption: there exists some $\gamma > 0$ such that for every negative instance x_-, $c(x_-) \leq 1 - \gamma$. Intuitively, this corresponds to the assumption that every negative instance is observed with some nonzero probability in a negative bag.

Since probability density functions exist for the example in Figure 3.2, we can analytically compute $c(x)$ given the following expression:

$$c(x) = \mathrm{P}\left[F(B) = 1 \mid x\right] = \frac{\int_{\mathcal{B}_+} \mathrm{P}(x \mid B)\, \mathrm{d}\, \mathrm{P}(B)}{\mathrm{P}(x)},$$

where $\mathcal{B}_+ = \{B : F(B) = 1\}$. As described, for positive instances $x_+ \in [0, 1]$, we

have:

$$c(x_+) = \frac{\int_\pi^1 \mathrm{P}(x_+ \mid B_\theta)\,\mathrm{P}(B_\theta)\,\mathrm{d}\theta}{\frac{1}{2}(1-P_{\text{neg}})(1+\pi)} = 1, \tag{3.6}$$

since positive instances always appear in positive bags. On the other hand, for negative instances,

$$\begin{aligned} c(x_-) &= \frac{\int_\pi^1 \mathrm{P}(x_- \mid B_\theta)\,\mathrm{P}(B_\theta)\,\mathrm{d}\theta}{P_{\text{neg}} + \frac{1}{2}(1-P_{\text{neg}})(1-\pi)} \\ &= \frac{\frac{1}{2}(1-P_{\text{neg}})(1-\pi)}{P_{\text{neg}} + \frac{1}{2}(1-P_{\text{neg}})(1-\pi)} \triangleq 1-\gamma. \end{aligned} \tag{3.7}$$

The resulting values of $c(x)$ are shown in Figure 3.2(d). Note that for this generative process, except for the trivial case in which $P_{\text{neg}} = 0$, $1-\gamma = c(x_-) < 1$, so $\gamma > 0$. Thus, the assumption that negative instances appear in negative bags is automatically satisfied for the example in Figure 3.2.

To see why negative instances must appear in negative bags in order to learn a concept, consider trying to learn the instance concept "spoon" in the content-based image retrieval (CBIR) domain, as described in Section 2.1.2. To learn this concept, you are given a set of images containing spoons, and a set of images not containing spoons. However, suppose that in every image containing a spoon, there is also a fork nearby. Furthermore, forks never appear alone in images without spoons. In this unfortunate scenario, you have no means of determining which of the fork or spoon is the positive instance given only image-level labels. However, if there is a guarantee that eventually you will see a negative image containing a fork but not a spoon, you will be able to learn that the fork is not the positive instance. We discuss learnability further in Chapter 4 and Chapter 6.

Finally, for learning bag-level concepts, we show in Section 4.2 that we require one additional assumption that there is some minimum fraction π of positive instances in each positive bag. That is, for every positive bag B_+, $\mathrm{P}\left[f(x) = 1 \mid B_+\right] \geq \pi$. By construction, the example in Figure 3.2 satisfies this assumption since there is zero probability of sampling a bag with $\theta \in (0, \pi)$ mass over positive bags.

Now, we can formally define MI-GEN, the set of generative processes for MI data consistent with the assumptions described above:

Definition 3.1 (MI-GEN)**.** *Given any* $\gamma \in (0, 1]$ *and* $\pi \in [0, 1]$, MI-GEN(γ, π) *is the set of all tuples* $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F)$*, each consisting of an instance distribution* $D_{\mathcal{X}}$ *(with corresponding* $\mathrm{P}(x)$*), bag distribution* $D_{\mathcal{B}}$ *(with corresponding* $\mathrm{P}(B)$*), instance-labeling function* f*, and bag-labeling function* F*, that satisfy the conditions:*

1. $\mathrm{P}(x) = \int_{\mathcal{B}} \mathrm{P}(x \mid B) \, \mathrm{d}\, \mathrm{P}(B)$
2. $\forall x : f(x) = 1 \implies \mathrm{P}\left[F(B) = 0 \mid x\right] = 0$
3. $\forall x : f(x) = 0 \implies \mathrm{P}\left[F(B) = 0 \mid x\right] \geq \gamma$
4. $\forall B : F(B) = 1 \implies \mathrm{P}\left[f(x) = 1 \mid B\right] \geq \pi.$

For simplicity, we will write MI-GEN(γ) for the case when $\pi = 0$, which corresponds to the weakest Condition 4. That is, for any fixed γ, MI-GEN$(\gamma) \supseteq$ MI-GEN(γ, π) for every $\pi \geq 0$.

Finally, note that for any $\gamma \in (0, 1]$, $\pi \in [0, 1]$, MI-GEN$(\gamma, \pi) \supseteq$ MI-GEN$(1, 1)$. That is, $\gamma = \pi = 1$ corresponds to the strongest constraints on the generative process. Even in this case, for *any* $D_{\mathcal{X}}$ and f, there exist $D_{\mathcal{B}}$ and F such that $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in$ MI-GEN$(1, 1)$. In particular, given a point mass δ_x centered on x, we can define $D_{\mathcal{B}}$ so that $\mathrm{P}_{\mathcal{B}}(\delta_x) = \mathrm{P}_{\mathcal{X}}(x)$ and F such that $F(\delta_x) = f(x)$. This choice of $(D_{\mathcal{B}}, F)$ corresponds to supervised learning expressed in our generative model. That is, sampling from our generative process in that case is indistinguishable from sampling directly from $D_{\mathcal{X}}$ with labels assigned according to f. Below, we discuss the relationship between our generative model and other proposed models for MI learning.

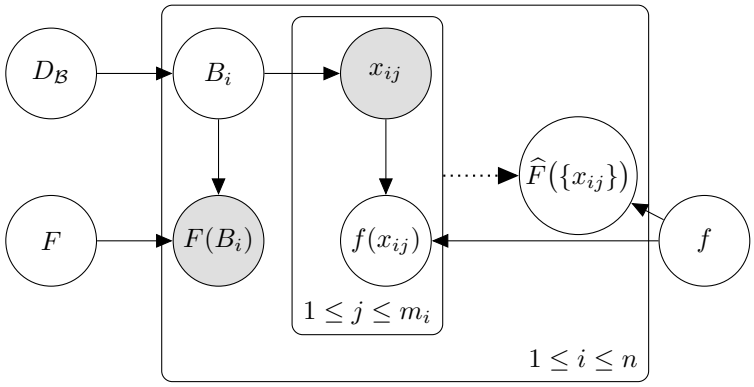


Figure 3.3: An illustration of the instance-, bag-, and empirical bag-labeling functions in MI-GEN.

3.2 The Empirical Bag-Labeling Function

In MI-GEN, the instance- and bag-labeling functions are defined independently at the level of the generative model, and must satisfy the relationships indicated in Definition 3.1. However, in the standard MI setting, bag labels are typically viewed as being derived from instance labels. That is, if a bag is a set, then it is positive if it contains at least one positive instance, and negative otherwise.

If we think of "bags" (in the sense of the standard generative model) of instances $\{x_{ij}\}_{j=1}^{m_i}$ as empirical samples drawn from the underlying bag distributions B_i in our model, as illustrated in Figure 3.1(b), then it is possible that samples from positive bags do not contain any positive instances. Hence, such "bags" would be negative in the sense of the standard model. To account for this discrepancy, we introduce the empirical bag-labeling function, $\widehat{F} : \mathcal{X}^* \to \{0, 1\}$:

$$\widehat{F}(X_i) = \max_j f(x_{ij}), \tag{3.8}$$

where $X_i = \{x_{ij}\}_{j=1}^{m_i}$ is any finite set of instances.

We can think of $\widehat{F}$ as the bag labels that would be assigned by an oracle that had perfect information about the instance-labeling function f, but only an empirical sample from each bag. An illustration of the empirical bag-labeling function is shown

in Figure 3.3. The labeling functions F and $\widehat{F}$ will always agree on negative bags, since only negative instances are observed in negative bags. However, there might be some discrepancy between F and $\widehat{F}$ on positive bags if only negative instances are sampled within a positive bag. We return to characterizing the discrepancy between F and $\widehat{F}$ in Section 4.2.

The discrepancy between F and $\widehat{F}$ is essentially bag-label noise that naturally results from our generative model. Previous generative models do not account for this potential source of label noise, despite its presence in some domains. For example, in the drug activity prediction domain, even if it is known that a molecule activates a receptor, a sample of conformations from this molecule might not contain the particular positive conformation that causes activation. Another observation is that some error naturally results due to the discrepancy between F and $\widehat{F}$ when one attempts to label bags given an instance-labeling function f. Even when f is known perfectly, there is still some error when using f to predict bag labels due to the particular sample of instances observed within each bag. Therefore, if bag-labeling is desired, better performance might be achieved by directly modeling F. In Section 6.4, we describe experimental results supporting this observation.

3.3 Relationship to Prior Models

The most general model in which instance learnability results have been previously shown is the "IID r-tuple" model (Blum and Kalai, 1998). The model, illustrated in Figure 3.4(a), assumes that each bag is generated by randomly sampling r instances in every bag from the same underlying instance distribution, $D_{\mathcal{X}}$. However, this is an unrealistic assumption for many domains. For example, consider the drug activity prediction setting. In this domain, that would mean that the conformations of every molecule are sampled independently from the same distribution, which is not true

as it requires that different molecules share the same conformations. Likewise, for CBIR, the IID assumption asserts that all segments in *all* images are sampled from the same distribution, when the distributions over objects/segments clearly change between images.

While our model assumes that each instance is sampled IID within every bag, it relaxes the standard IID assumption because instances can be drawn from different distributions within different bags. Our generative model allows for "structure," in the form of correlations between instances, to exist independently in each bag, which has been described as a more realistic assumption (Zhou et al., 2009).

To show that our model is more general than the IID r-tuple model, we now describe how to simulate this model within our model. First, we define each bag to be a probability distribution parameterized by an r-tuple of instances $B_{(x_1,\ldots,x_r)}$. This distribution will be a weighted sum of point masses over each of the r instances: $\mathrm{P}(x \mid B_{(x_1,\ldots,x_r)}) = \frac{1}{r}\sum_{i=1}^{r}\delta_{x_i}(x)$. Then, for any distribution $D_{\mathcal{X}}$ over instances (with $\mathrm{P}(x)$) and instance-labeling function f, we let the distribution over bags $D_{\mathcal{B}}$ be defined as $\mathrm{P}(B_{(x_1,\ldots,x_r)}) \triangleq \prod_{i=1}^{r}\mathrm{P}(x_i)$, which is the probability that the corresponding r-tuple would have been sampled from $D_{\mathcal{X}}^{r}$, and the bag-labeling function F to be $F(B_{(x_1,\ldots,x_r)}) = \max_{1\le i\le r} f(x_i)$. Let $p_{\text{neg}} = \mathrm{P}\left[f(x) = 0\right]$, then we claim that the $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F)$ described above is in MI-GEN $\left(p_{\text{neg}}^{r-1}, \frac{1}{r}\right)$.

First, we need to show that $D_{\mathcal{B}}$ as defined satisfies Condition 1 of Definition 3.1:

$$
\begin{aligned}
\mathrm{P}(x) &\stackrel{?}{=} \int_{\mathcal{B}} \mathrm{P}(x \mid B)\,\mathrm{d}\,\mathrm{P}(B) \\
&= \int_{\mathcal{B}} \frac{1}{r}\sum_{i=1}^{r}\delta_{x_i}(x)\,\mathrm{d}\,\mathrm{P}(B_{(x_1,\ldots,x_r)}) \\
&= \frac{1}{r}\sum_{i=1}^{r}\int_{\mathcal{X}}\cdots\int_{\mathcal{X}}\delta_{x_i}(x)\,\mathrm{d}\,\mathrm{P}(x_r)\cdots\mathrm{d}\,\mathrm{P}(x_1) \\
&= \frac{1}{r}\sum_{i=1}^{r}\left(\prod_{j\neq i}\int_{\mathcal{X}}\mathrm{d}\,\mathrm{P}(x_j)\right)\left(\int_{\mathcal{X}}\delta_{x_i}(x)\,\mathrm{d}\,\mathrm{P}(x_i)\right)
\end{aligned}
$$

$$= \frac{1}{r} \sum_{i=1}^{r} \left(1^{r-1}\right) \mathrm{P}(x) = \mathrm{P}(x).$$

So sampling instances under our two-step generative process is equivalent to sampling according to the original instance distribution.

Condition 2 of Definition 3.1 is trivially satisfied, since by the definition of F, positive instances never appear in negative bags. To show that Condition 3 holds, we must compute the probability that negative instances appear in a negative bag. Using the definition of conditional probability, this is:

$$\mathrm{P}\left[F(B) = 0 \mid x\right] = \frac{\int_{\mathcal{B}_-} \mathrm{P}(x \mid B) \, \mathrm{d}\, \mathrm{P}(B)}{\mathrm{P}(x)}.$$

Using the fact that in a negative bag $B_{(x_1,\ldots,x_r)}$, all instances must be negative, we can compute the numerator for a negative instance as:

$$\begin{aligned}
\int_{\mathcal{B}_-} \mathrm{P}(x \mid B) \, \mathrm{d}\, \mathrm{P}(B) &= \int_{\mathcal{B}_-} \frac{1}{r} \sum_{i=1}^{r} \delta_{x_i}(x) \, \mathrm{d}\, \mathrm{P}(B_{(x_1,\ldots,x_r)}) \\
&= \frac{1}{r} \sum_{i=1}^{r} \int_{\mathcal{X}_-} \cdots \int_{\mathcal{X}_-} \delta_{x_i}(x) \, \mathrm{d}\, \mathrm{P}(x_r) \cdots \mathrm{d}\, \mathrm{P}(x_1) \\
&= \frac{1}{r} \sum_{i=1}^{r} \left(\prod_{j \neq i} \int_{\mathcal{X}_-} \mathrm{d}\, \mathrm{P}(x_j) \right) \left(\int_{\mathcal{X}_-} \delta_{x_i}(x) \, \mathrm{d}\, \mathrm{P}(x_i) \right) \\
&= \frac{1}{r} \sum_{i=1}^{r} \left(p_{\mathrm{neg}}^{r-1}\right) \mathrm{P}(x) = p_{\mathrm{neg}}^{r-1} \, \mathrm{P}(x).
\end{aligned}$$

Thus, $\mathrm{P}\left[F(B) = 0 \mid x\right] = \frac{p_{\mathrm{neg}}^{r-1} \mathrm{P}(x)}{\mathrm{P}(x)} = p_{\mathrm{neg}}^{r-1}$. Since this probability is the same across all negative instances, this means that $\gamma = p_{\mathrm{neg}}^{r-1}$. This quantity is positive as long as $p_{\mathrm{neg}} > 0$. Otherwise, all instances are positive, so the $\gamma > 0$ assumption is vacuously satisfied.

Finally, to show that Condition 4 of Definition 3.1 is satisfied, we see that for a

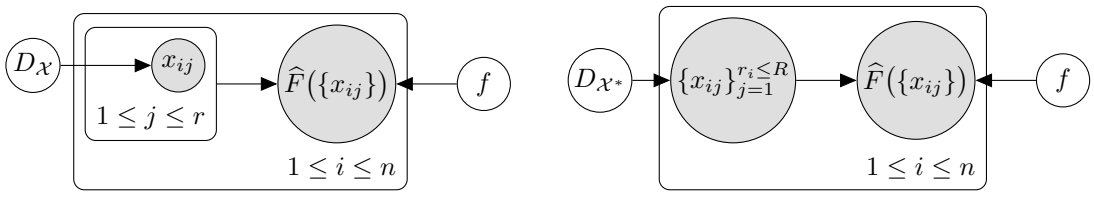
positive bag, B_i:

$$\begin{aligned} \mathrm{P}\left[f(x) = 1 \mid B\right] &= \int_{\mathcal{X}} f(x) \left(\tfrac{1}{r} \sum_{i=1}^{r} \delta_{x_i}(x) \right) \mathrm{d}x \\ &= \frac{1}{r} \sum_{i=1}^{r} f(x_i) \geq \frac{1}{r} = \pi, \end{aligned} \tag{3.9}$$

since at least one instance in the bag is such that $f(x) = 1$. Therefore, the IID r-tuple model is a special case of our model in which γ and π are positive, and determined by the fraction of negative instances and bag size r.

Another generative model, used to show the learnability of bag-level concepts (Sabato and Tishby, 2012), allows arbitrary distributions over r-tuples. The model further relaxes the r-tuple model by allowing bag sizes to vary from 1 to R, some maximum bag size. The model is illustrated in Figure 3.4(b), where $D_{\mathcal{X}^*}$ denotes the distribution over tuples of size at most R. However, this model is also restrictive for many problem domains like drug activity prediction, since it enforces that bag sizes are finite and bounded, whereas molecules can exist in infinitely many conformations.

We can also represent the generative model of Sabato and Tishby (2012) in a similar way as for the IID r-tuple model. We simplify the space of bags to be atomic distributions over $r \leq R$ tuples, and allow an arbitrary distribution $D_{\mathcal{B}}$ over bags rather than requiring that $\mathrm{P}(B_{(x_1,\ldots,x_r)}) = \prod_{i=1}^{r} \mathrm{P}(x_i)$. Now, $D_{\mathcal{X}}$ is not fixed, so we can define it in terms of Condition 1 of MI-GEN so that that condition is automatically satisfied. The bag-labeling function F is still defined in terms of the arbitrary instance-labeling function f, so Condition 2 is still trivially satisfied. Furthermore, by similar reasoning as in Equation 3.9, $\pi = \frac{1}{R}$ in this generative model, so Condition 4 is satisfied. However, the $\gamma > 0$ assumption (Condition 3) is no longer automatically satisfied by this generative process, since arbitrary distributions over tuples are allowed. Hence, while Sabato and Tishby (2012) analyze bag concept learnability with MI-GEN $\left(0, \frac{1}{R}\right)$, we require MI-GEN $\left(\gamma, \frac{1}{R}\right) \subset$ MI-GEN $\left(0, \frac{1}{R}\right)$ for instance concept



(a) The IID r-tuple model (b) The arbitrary R-tuple model

Figure 3.4: Previous generative models for MI data.

learnability. In Proposition 6.1, we discuss how the $\gamma > 0$ assumption is not required for directly learning bag concepts in our generative model.

Babenko et al. (2011) propose treating bags in the MI setting as *manifolds* in the instance space $\mathcal{X}$. While this allows describing a bag with an infinite number of instances, it assigns an equal "weight" to every instance. However, for a domain like drug activity prediction, a molecule is more likely to exist in certain conformations than in others. The varying weight of instances is naturally handled in our setting, but we do not treat bags as manifolds over instances, so our results may not apply to the generative process in which bags are manifolds.

The most similar generative model in prior work (Xu, 2003) also proposed to model bags as distributions, but only focused on analyzing empirical results for the bag-labeling task. Accordingly, the proposed probabilistic relationship between instance- and bag-labeling functions given in Definition 3.1 are novel. Furthermore, Xu (2003) focused on the case when each bag distribution has a parametric form, with a distribution over parameters used to define $D_{\mathcal{B}}$. In Section 6.3, we discuss bag classification algorithms that can be used under our generative model without requiring such strong assumptions.

3.4 Applicability to Problem Domains

At the beginning of this chapter, we motivated MI-GEN using the 3-dimensional Quantitative Structure–Activity Relationship (3D-QSAR) domain. In this section,

we elaborate on how bags can naturally be viewed as distributions in various other problem domains (see Table 2.1) and which labeling tasks illustrated in Figure 3.3 are of interest in each domain. Of course, as described in Section 3.3, standard generative models are special cases of MI-GEN, so previous applications of MIL for which it is most natural to think of bags as finite sets of instances can still be incorporated in this model.

Drug Activity Prediction

For the drug activity prediction or the 3D-QSAR problem, it is natural to think of each molecule as a distribution over conformations. While it is natural to view learning molecule-level activity F as the ultimate goal of 3D-QSAR, it is also important to learn the instance-labeling function f. Knowing whether an individual conformation binds to a receptor provides information about the structure of the receptor's binding site, which is practically difficult to measure directly. Hence, learning both instance- and bag-labeling functions are important in the 3D-QSAR domain.

Text Categorization

While it is popular to represent documents as a flat "bag of words" using a single feature vector comprised of word frequencies (Salton and McGill, 1983), prior work has acknowledged the benefits of representing document-specific structure. In particular, latent Dirichlet allocation (LDA) models each document as a mixture of distributions over words (Blei et al., 2003). Of course, LDA can also be applied to a coarser-grained representation in which documents are distributions over n-grams or paragraphs, which are like individual instances in the MI setting (Blei et al., 2003). Hence, treating documents as distributions is already a natural and popular representation for text. On the other hand, LDA treats each document distribution as taking a specific parametric form, whereas our results and analysis do not make any parametric

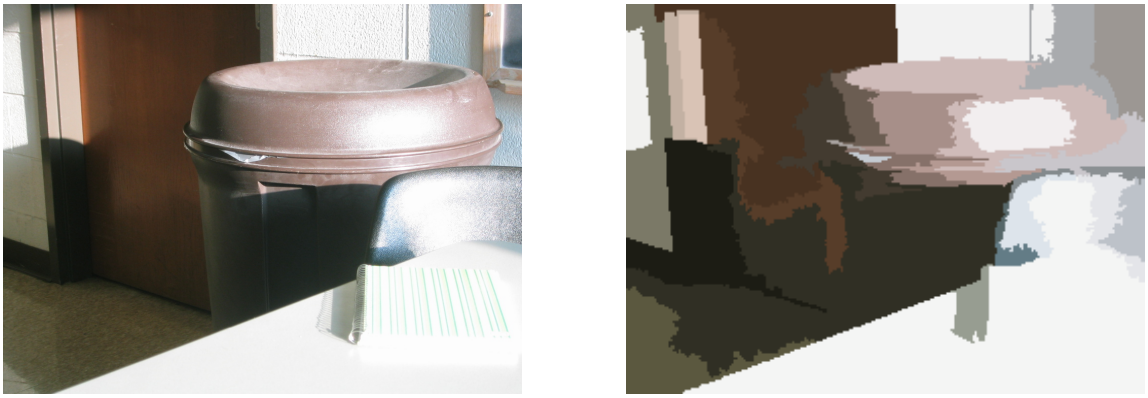


Figure 3.5: An example from the CBIR domain when a positive image does not contain a positive instance (the notebook) after segmentation.

assumptions about bag-level distributions.

As for 3D-QSAR, both document-level and instance-level categorization is important in the text categorization domain. For example, if certain types of documents like survey articles discuss various subjects, then it might be important to determine not just that the document as a whole discusses a particular subject, but also which specific passage or paragraph discusses the subject.

Content-Based Image Retrieval

Applying our generative framework to the CBIR task requires viewing images as distributions over objects such that the objects in each image are a sample from the corresponding distribution. However, it is not immediately clear what it means in this setting to have a positive bag from which only negative instances have been sampled. That is, whether an image contains an object is defined in terms of $\widehat{F}$, not some abstract image-labeling function F.

However, just as for 3D-QSAR, our generative process allows for noisy behavior during bag-generation. In 3D-QSAR, it is possible that a computational chemistry tool enumerating low-energy conformations will miss the binding conformation within a positive molecule. Likewise, extracting a set of objects from images is often per-

formed using segmentation achieved through local optimization (Andrews et al., 2003; Carson et al., 2002).[2] Therefore, it is possible that no single instance in a bag generated from a positive image will correspond to the positive instance. Figure 3.5 shows an example from the SIVAL dataset when, due to lighting conditions, the positive "notebook" instance in the image is grouped with the table during segmentation (Settles et al., 2008). This kind of "noise" is naturally captured by our generative model as the discrepancy between $\widehat{F}$ and F.

As for the other domains discussed, the bag-labeling function F is not the only latent variable of interest in CBIR. In additional to labeling new images, a CBIR system might be interested in determining the location of the object of interest within an image, which requires learning the instance-labeling function f.

3.5 Multiple-Instance Learning with Noisy Bags

The generative model described in Definition 3.1 allows for a form of bag-level label noise when $\pi = 0$. That is, when $\pi = 0$, it is possible for a positive bag to contain no positive instances, which corresponds to "noisy" bags with incorrect labels in the standard MI setting. In Section 4.1 and Section 5.1, we show that it is possible to learn instance-level concepts in the presence of such noise.

On the other hand, MI-GEN does not allow noisy negative bags that contain some positive instances. All positive instances must appear in positive bags. In Section 5.3, we show that it is possible to relax this assumption as well and retain instance concept learnability. We call this relaxation the noisy MI (NMI) setting.

Let η be the maximum fraction of the time that any positive instance appears in a negative bag. Intuitively, since negative instances appear at least γ fraction of the time in negative bags, if a positive instance also appeared $\eta \geq \gamma$ fraction of the time in negative bags, it would be indistinguishable from a negative instance. Thus, our

[2] See Appendix A.1.2 for a detailed description of how bags are generated from images.

generalized model assumes that $\eta < \gamma$ so that positive and negative instances can be distinguished. In Section 5.3, we prove this assertion more rigorously. Below, we formally specify the generalized generative model:

Definition 3.2 (NMI-GEN)**.** *Given any* $\gamma \in (0, 1]$ *and* $\eta \in [0, \gamma)$*,* NMI-GEN(γ, η) *is the set of all tuples* $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F)$ *that satisfy the conditions:*

1. $\mathrm{P}(x) = \int_{\mathcal{B}} \mathrm{P}(x \mid B) \, \mathrm{d}\, \mathrm{P}(B)$
2. $\forall x : f(x) = 1 \implies \mathrm{P}\left[F(B) = 0 \mid x\right] \leq \eta$
3. $\forall x : f(x) = 0 \implies \mathrm{P}\left[F(B) = 0 \mid x\right] \geq \gamma.$

The new generative model in Definition 3.2 relaxes Condition 2 in Definition 3.1. Hence, MI-GEN(γ) = NMI-GEN$(\gamma, 0)$, so that MI-GEN is a special case of NMI-GEN with $\eta = 0$. Note that since we do not discuss bag learnability under this model, Condition 4 from Definition 3.1 is no longer included in the definition of NMI-GEN ($\pi = 0$ is permitted).

The concepts allowed by NMI-GEN include some generalized MI concepts from the Generalized MIL (GMIL) setting (Scott et al., 2005), as described in Section 2.1.3. In particular, the GMIL framework of Scott et al. (2005) allows positive bags to be labeled according to whether they contain multiple types of instances from different subclasses of the class of positive instances. Section 2.1.3 describes an example GMIL problem of learning to classify images of beaches. Such images must contain instances of *both* sand and water to be positive, otherwise, they might be pictures of deserts or oceans.

If we consider the "sand" and "water" subclasses to contain the positive instances (the instances that are *necessary*, though perhaps not sufficient, for a bag to be positive), then a positive "sand" instance in the beach image classification task might appear in a negatively labeled desert image. The NMI model described in Definition 3.2 can account for this possibility.

Of course, not all GMIL concepts as defined by Scott et al. (2005) can be expressed within NMI-GEN. That is, whereas the $\gamma > 0$ assumption encapsulates most practical applications of MIL, the $\eta < \gamma$ assumption is a relatively stronger. In the beach example, requiring that $\eta < \gamma$ might significantly limit the relative proportion of beach to desert images in the dataset, so that sand instances do not appear too often in negative bags relative to positive bags. Therefore, the generative processes allowed by NMI-GEN represent only a limited, though still significant, subset of all GMIL problems. Chapter 6 discusses the ability of some approaches to learn bag-level GMIL concepts under weaker assumptions.

3.6 Summary

In this chapter, we described a new generative model for the MI setting in which bags are viewed as distributions over instances. The sets of instances observed in a training set are then viewed as samples from each underlying bag distribution. We then introduced several additional assumptions that we show entail instance and bag concept learnability in the following chapters. We discussed the relationship between the proposed model and those found in prior work. We also discussed how the model can be generalized to incorporate a form of "noise" on bag-level labels.

Chapter 4

Learning Accurate Concepts from MI Data

In this chapter, we describe new theoretical results that highlight the advantages of the generative model proposed in Chapter 3.[1] In particular, the new generative model allows new results about instance- and bag-concept learnability that previously only held under a much stronger set of assumptions. As we describe in Chapter 5, additional theoretical results imply the surprising but testable ability of standard supervised approaches to learn to rank instances and bags from MI data. Table 4.1 summarizes the theoretical contributions made in this and the following chapter, which demonstrates the learnability of the instance concept f, empirical bag-labeling function $\widehat{F}$, and bag-labeling function F with respect to both accuracy and ranking as measured by area under the ROC curve (AUC). The results in this chapter and the following chapter use a model of the instance labeling function f to derive models for the bag-labeling functions $\widehat{F}$ and F. Additional results in Chapter 6 discuss approaches for directly learning bag concepts. Table 4.2 provides an overview of the results on learning either instance or bag concepts using instance- or bag-labeling

[1] Results on instance concept learnability also appear in Doran and Ray (2014).

Table 4.1: A summary of the learnability results in Chapter 4 and Chapter 5.

		Accuracy	**AUC**
Instance	f	Theorem 4.1	Theorem 5.1 Theorem 5.4†
Bag	$\widehat{F}$	Theorem 4.2	Theorem 5.2
	F	Theorem 4.3	Theorem 5.3

†Results for NMI-GEN.

Table 4.2: A summary of results addressing different learning tasks and strategies.

	. . . using **instance-level** classifiers.	. . . using **bag-level** classifiers.
Learning **instance-level** concepts. . .	Section 4.1 Section 5.1	Section 7.2 (Hardness Result)
Learning **bag-level** concepts. . .	Section 4.2 Section 5.2	Chapter 6

functions.

As described in Section 2.4, defining the ability of an algorithm to learn a good approximation of a target concept requires some metric by which the quality of the approximation is to be measured. Traditionally, the quality of a classifier is measured in terms of expected 0–1 loss, as defined in Equation 2.10. This metric formalizes the intuitive notion of accuracy, or the fraction of all test data points correctly classified. We begin by investigating the ability of algorithms to learn accurate concepts from MI data in this sense. While there is only one learning task in the supervised setting, there are now both instance- and bag-concept learning tasks in the MI setting, which we explore separately in the following sections. Table 4.3 shows the notation used for the concepts in this chapter.

Table 4.3: Legend of the basic notation used in Chapter 4.

Symbol	Description/Definition
$\mathcal{X}$	Space of instances
$\mathcal{B}$	Space of bags (distributions over instances)
$\mathcal{X}^*$	Set of bag samples (sets of instances)
x_{ij}	Instance $x_{ij} \in \mathcal{X}$
B_i	Bag $B_i \in \mathcal{B}$
X_i	Bag sample $\{x_{ij}\}_{j=1}^{m_i} \in \mathcal{X}^*$, $x_{ij} \sim B_i$ $\quad (m_l \leq \lvert X_i \rvert \leq m_u)$
(m_l, m_u) m_i	(Minimum, Maximum) Bag Sample Size
f	Instance-Labeling Concept
F	Bag-Labeling Concept
$\widehat{F}$	Empirical Bag-Labeling Concept $\quad \widehat{F}(X_i) \triangleq \max_j f(x_{ij})$
g	Instance-Labeling Hypothesis
$\widehat{G}$	Empirical Bag-Labeling Hypothesis $\quad \widehat{G}(X_i) \triangleq \max_j g(x_{ij})$

4.1 Learning Accurate Instance Concepts

Section 2.4.1 describes the probably approximately correct (PAC) framework, which formalizes what it means to learn accurate concepts from supervised data. Since the generative process described in Chapter 3 differs from that for supervised learning, we must restate what it means to "PAC" learn an accurate instance concept under this model.

In the supervised setting, the learnability of some fixed concept class $\mathcal{F}$ is discussed without making any assumptions about the distribution over instances. The definition of MI-GEN in Definition 3.1 similarly allows any instance distribution, with which many bag distributions are consistent in the sense of Condition 1. To ensure that the target concept f is a member of the concept class $\mathcal{F}$, we must further restrict the set of models allowed by the generative process as follows:

Definition 4.1 (MI-GEN$_{\mathcal{F}}$)**.** *For any* $\gamma \in (0, 1]$ *and* $\pi \in [0, 1]$*:*

$$\text{MI-GEN}_{\mathcal{F}}(\gamma, \pi) \triangleq \big\{(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}(\gamma, \pi) : f \in \mathcal{F}\big\}.$$

Now, we can formally define PAC learnability for the MI setting:

Definition 4.2 (Instance MI PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ MI PAC-learns instance concept class $\mathcal{F}$ from MI data when for any tuple $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in$ MI-GEN$_{\mathcal{F}}(\gamma)$ with $\gamma > 0$, and $\epsilon_{\mathrm{I}}, \delta > 0$, $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{\gamma}, \frac{1}{\epsilon_{\mathrm{I}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an instance hypothesis g with risk $R_f(g) < \epsilon_{\mathrm{I}}$ with probability at least $1 - \delta$ over samples.*

Note that because our generative model allows us to discuss the marginalized instance distribution $D_{\mathcal{X}}$, the risk $R_f(g) = \mathrm{E}_{x \sim D_{\mathcal{X}}}\left[\mathbb{1}[f(x) \neq g(x)]\right]$ is measured with respect to this distribution exactly as in Equation 2.10. Now we show that instance concepts are MI PAC-learnable in the sense of Definition 4.2:

Theorem 4.1. *An instance concept class $\mathcal{F}$ with VC dimension* VC($\mathcal{F}$) *is Instance MI PAC-learnable using $O\left(\frac{1}{\epsilon_{\mathrm{I}}\gamma}\left(\mathrm{VC}(\mathcal{F}) \log \frac{1}{\epsilon_{\mathrm{I}}\gamma} + \log \frac{1}{\delta}\right)\right)$ examples.*

Proof. The proof proceeds by showing that learning instance concepts from MI data is equivalent to learning a standard instance concept in the presence of one-sided label noise; that is, noisy labels on the negative instances only. Then, recent results (Simon, 2012) imply the learnability of the underlying instance concept.

By Condition 1 in Definition 3.1, we can treat bag-labeled instances as being drawn from the underlying instance distribution $D_{\mathcal{X}}$. Instances are observed with some label noise with respect to true labels given by f. Since positive instances never appear in negative bags (by Condition 2 of Definition 3.1), noise on instances is one-sided. If every negative instance appears in negative bags at least some γ fraction of the time (by Condition 3), then the maximum one-sided noise rate is $\eta = 1 - \gamma$. Since $\gamma > 0$, $\eta < 1$, which is required for learnability. Under our generative assumptions, the noise is "semi-random" in that noise rate might vary across instances, but is bounded by

$\eta < 1$. Thus, learning an instance concept is equivalent to learning from data with one-sided label noise in this sense.

The result of Simon (2012) shows that in the presence of one-sided, semi-random noise, when a concept class $\mathcal{F}$ has Vapnik–Chervonenkis (VC) dimension $\mathrm{VC}(\mathcal{F})$, $\mathcal{F}$ is PAC-learnable from $O\left(\frac{1}{\epsilon_1(1-\eta)}\left(\mathrm{VC}(\mathcal{F})\log\frac{1}{\epsilon_1(1-\eta)}+\log\frac{1}{\delta}\right)\right)$ examples using a "minimum one-sided disagreement" strategy. This strategy entails choosing a classifier that minimizes the number of disagreements on positively-labeled examples while perfectly classifying all negatively-labeled examples. This strategy also works in the special case that all instances and bags are positive ($\eta = 0$, or $\gamma = 1$, since there are no negative instances). Substituting $1 - \gamma$ for η in the bound of Simon (2012) yields the bound in terms of γ. □

We note that Theorem 4.1 allows for "noisy" positive bags without positive instances ($\pi = 0$), since the additional bag-level noise is essentially absorbed into η, but not noisy negative bags that contain positive instances.

4.2 Learning Accurate Bag Concepts

As for instance concept learnability, we must formally define what we mean to learn accurate bag concepts in the MI setting. As described in Chapter 3, there are two bag-labeling functions we might be interested in learning. In our generative model, we assume that the MI relationship between bag and instance labels holds at the level of the generative process. That is, bags are directly assigned labels by a bag concept F. On the other hand, given a set of instances sampled from a bag, we might be interested in learning the more traditional bag-labeling concept in the MI setting, $\widehat{F}(X_i) = \max_j f(x_{ij})$, which determines whether an empirically observed bag sample $X_i \in \mathcal{X}^*$ contains any positive instance (Equation 3.8). We call this the empirical bag labeling function, and we can define the risk of a bag-labeling concept $\widehat{G}$ with respect

to the underlying empirical bag-labeling concept $\widehat{F}$ as follows:

$$\begin{aligned} R_{\widehat{F}}(\widehat{G}) &= \mathrm{E}\left[\mathbb{1}[\widehat{F}(X) \neq \widehat{G}(X)]\right] \\ &= \int_{\mathcal{B}} \int_{\mathcal{X}^*} \mathbb{1}[\widehat{F}(X) \neq \widehat{G}(X)] \,\mathrm{d}\,\mathrm{P}(X \mid B) \,\mathrm{d}\,\mathrm{P}(B), \end{aligned} \tag{4.1}$$

where $\mathrm{P}(X \mid B)$ is the probability of sampling the set of instances X from bag B. Since we assume that instances are sampled IID according to B, $\mathrm{P}(X \mid B) = \prod_{x \in X} \mathrm{P}(x \mid B)$. Given a formal definition of the risk of an empirical bag-labeling function, we can define learnability with respect to this function below:

Definition 4.3 (Empirical Bag MI PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ MI PAC-learns empirical bag-labeling functions derived from instance concept class $\mathcal{F}$ when for any $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}_{\mathcal{F}}(\gamma)$ with $\gamma > 0$, and $\epsilon_{\mathrm{B}}, \delta > 0$, $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{\gamma}, \frac{1}{\epsilon_{\mathrm{B}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an empirical bag-labeling function $\widehat{G}$ with risk $R_{\widehat{F}}(\widehat{G}) < \epsilon_{\mathrm{B}}$ with probability at least $1 - \delta$ over samples.*

To show empirical bag concept learnability under our generative model, we will show that by learning an accurate enough instance concept g, the resulting empirical bag-labeling concept given by $\widehat{G}(X_i) = \max_j g(x_{ij})$ will have low risk with respect to $\widehat{F}$. Thus, we start with a bound on $R_{\widehat{F}}(\widehat{G})$ in terms of $R_f(g)$.

Lemma 4.1. *Let $R_f(g)$ be the risk of an instance labeling concept g, and $R_{\widehat{F}}(\widehat{G})$ be the risk of the empirical bag-labeling function $\widehat{G}(X_i) = \max_j g(x_{ij})$. Then if bag sample sizes are bounded by m_u ($\forall i : |X_i| \leq m_u$), $R_{\widehat{F}}(\widehat{G}) \leq m_u R_f(g)$.*

Proof. First, observe that when all elements of an empirical bag X_i are labeled correctly by g, $\widehat{F}(X_i) = \widehat{G}(X_i)$, so when $\widehat{F}(X_i) \neq \widehat{G}(X_i)$, at least one instance in X_i is labeled incorrectly by g. In set notation, this implication is equivalent to the

statement:

$$\left\{X_i : \widehat{F}(X_i) \neq \widehat{G}(X_i)\right\} \subseteq \left\{X_i : \big(f(x_{i1}) \neq g(x_{i1})\big) \vee \ldots \vee \big(f(x_{im}) \neq g(x_{im})\big)\right\}.$$

Using indicator function ($\mathbb{1}[\cdot]$) notation, the statement above implies:

$$\begin{aligned}
\mathbb{1}\big[\widehat{F}(X_i) \neq \widehat{G}(X_i)\big] &\leq \mathbb{1}\big[\big(f(x_{i1}) \neq g(x_{i1})\big) \vee \ldots \vee \big(f(x_{im}) \neq g(x_{im})\big)\big] \\
&= \mathbb{1}\left[\bigvee\nolimits_{x_{ij} \in X_i} \big(f(x_{ij}) \neq g(x_{ij})\big)\right] \\
&\leq \sum_{x_{ij} \in X_i} \mathbb{1}\left[f(x_{ij}) \neq g(x_{ij})\right].
\end{aligned}$$

Using this inequality in the definition of risk for empirical bag-labeling functions (Equation 4.1) yields:

$$\begin{aligned}
R_{\widehat{F}}(\widehat{G}) &= \int_{\mathcal{B}} \int_{\mathcal{X}^*} \mathbb{1}\big[\widehat{F}(X_i) \neq \widehat{G}(X_i)\big] \,\mathrm{d}\,\mathrm{P}(X_i \mid B) \,\mathrm{d}\,\mathrm{P}(B) \\
&\leq \int_{\mathcal{B}} \int_{\mathcal{X}^*} \sum_{x_{ij} \in X_i} \mathbb{1}\left[f(x_{ij}) \neq g(x_{ij})\right] \,\mathrm{d}\,\mathrm{P}(X_i \mid B) \,\mathrm{d}\,\mathrm{P}(B).
\end{aligned}$$

By the independence of the instances $x_{ij} \in X_i$, and the bound m_u on bag sample sizes, we can rewrite the inner integral to conclude that:

$$\begin{aligned}
R_{\widehat{F}}(\widehat{G}) &\leq \int_{\mathcal{B}} m_u \int_{\mathcal{X}} \mathbb{1}\left[f(x) \neq g(x)\right] \,\mathrm{d}\,\mathrm{P}(x \mid B) \,\mathrm{d}\,\mathrm{P}(B) \\
&= m_u \int_{\mathcal{X}} \mathbb{1}\left[f(x) \neq g(x)\right] \,\mathrm{d}\,\mathrm{P}(x) \\
&= m_u R_f(g).
\end{aligned}$$

Exchanging the order of the integrals and marginalizing out the individual bag distributions to obtain an integral with respect to the instance distribution follows from Condition 1 in Definition 3.1. □

Given the bound demonstrated in Lemma 4.1, we can easily derive the following

result:

Theorem 4.2. *Empirical bag-labeling functions derived from instance concept class $\mathcal{F}$ with VC dimension $\mathrm{VC}(\mathcal{F})$ are PAC-learnable from MI data using*

$$O\left(\frac{m_u}{\epsilon_{\mathrm{B}}\gamma}\left(\mathrm{VC}(\mathcal{F})\log\frac{m_u}{\epsilon_{\mathrm{B}}\gamma}+\log\frac{1}{\delta}\right)\right)$$

examples.

Proof. The general strategy is to learn an approximation g for $f \in \mathcal{F}$ using minimum one-sided disagreement as mentioned in the proof of Theorem 4.1 and then to derive an empirical bag-labeling function $\widehat{G}$ from g.

For a desired bound ϵ_{B} on $R_{\widehat{F}}(\widehat{G})$, by using $\epsilon_{\mathrm{I}} = \frac{\epsilon_{\mathrm{B}}}{m_u}$ in Theorem 4.1, this ensures that the resulting instance classifier is such that $R_f(g) < \frac{\epsilon_{\mathrm{B}}}{m_u}$ with high probability. Combined with the result in Lemma 4.1, this implies that $R_{\widehat{F}}(\widehat{G}) \leq m_u R_f(g) < m_u\left(\frac{\epsilon_{\mathrm{B}}}{m_u}\right) = \epsilon_{\mathrm{B}}$, so $R_{\widehat{F}}(\widehat{G}) < \epsilon_{\mathrm{B}}$ as desired. Substituting $\epsilon_{\mathrm{I}} = \frac{\epsilon_{\mathrm{B}}}{m_u}$ into the bound in Theorem 4.1 gives the bound as stated in Theorem 4.2. □

Again, Theorem 4.2 allows for noisy positive bags without positive instances ($\pi = 0$). Furthermore, in the special case when every bag sample is a singleton $X = \{x\}$, then $m_u = 1$ and $\widehat{F}(\{x\}) = f(x)$. Thus, the instance concept learnability result in Theorem 4.1 is really just a special case of learning an empirical bag-labeling function with bags of size 1 as in Theorem 4.2.

Next, we turn our attention to learning the underlying bag-labeling function F. During both training *and* testing, we are only given access to a sample X_i from each bag B_i with which we can estimate $F(B_i)$. Therefore, we will again learn an empirical bag labeling function $\widehat{G}(X_i)$. However, now we will assess the quality of $\widehat{G}$

with respect to the underlying bag-labeling function F as follows:

$$\begin{aligned} R_F(\widehat{G}) &= \mathrm{E}\left[\mathbb{1}[F(B) \neq \widehat{G}(X)]\right] \\ &= \int_{\mathcal{B}} \int_{\mathcal{X}^*} \mathbb{1}[F(B) \neq \widehat{G}(X)] \,\mathrm{d}\,\mathrm{P}(X \mid B) \,\mathrm{d}\,\mathrm{P}(B). \end{aligned} \tag{4.2}$$

The definition of bag concept learnability then takes the same form as that in Definition 4.3 with the risk as given in Equation 4.2. As we will show in Lemma 4.2, we now also require the further assumption that π, minimum fraction of positive instances in positive bags, is nonzero.

Definition 4.4 (Bag MI PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ MI PAC-learns bag-labeling functions derived from instance concept class $\mathcal{F}$ when for any $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}_{\mathcal{F}}(\gamma, \pi)$ with $\gamma, \pi > 0$, and $\epsilon_{\mathrm{B}}, \delta > 0$, algorithm $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{\gamma}, \frac{1}{\pi}, \frac{1}{\epsilon_{\mathrm{B}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an empirical bag-labeling function $\widehat{G}$ with risk $R_F(\widehat{G}) < \epsilon_{\mathrm{B}}$ with probability at least $1 - \delta$ over samples.*

In order to show learnability of the bag-labeling concept F, we adopt a similar strategy as for Theorem 4.2 in which we first learn an instance-labeling concept g, then use g to derive an empirical bag-labeling concept $\widehat{G}$. Since Theorem 4.2 shows that we can a learn a concept $\widehat{G}$ that accurately models $\widehat{F}$, what remains to be shown is that $\widehat{F}$ is an accurate model of F under some additional conditions. First, we prove the following Lemma, that decomposes the risk $R_F(\widehat{G})$ into the discrepancy between $\widehat{G}$ and $\widehat{F}$, and the discrepancy between $\widehat{F}$ and F.

Lemma 4.2. *For any empirical bag-labeling concept $\widehat{G}$,*

$$R_F(\widehat{G}) \leq R_{\widehat{F}}(\widehat{G}) + R_F(\widehat{F}).$$

Proof. First, note that if $\widehat{G}(X) = \widehat{F}(X)$ and $\widehat{F}(X) = F(B)$, then $\widehat{G}(X) = F(B)$.

Thus, if $\widehat{G}(X) \neq F(B)$, then either $\widehat{G}(X) \neq \widehat{F}(X)$ or $\widehat{F}(X) \neq F(B)$. In set notation, this is equivalent to the statement:

$$\left\{(X,B) : \widehat{G}(X) \neq F(B)\right\} \subseteq \left\{(X,B) : \left(\widehat{G}(X) \neq \widehat{F}(X)\right) \vee \left(\widehat{F}(X) \neq F(B)\right)\right\}.$$

Using indicator function notation, the statement above implies:

$$\begin{aligned}\mathbb{1}\left[\widehat{G}(X) \neq F(B)\right] &\leq \mathbb{1}\left[\left(\widehat{G}(X) \neq \widehat{F}(X)\right) \vee \left(\widehat{F}(X) \neq F(B)\right)\right] \\ &\leq \mathbb{1}\left[\left(\widehat{G}(X) \neq \widehat{F}(X)\right)\right] + \mathbb{1}\left[\left(\widehat{F}(X) \neq F(B)\right)\right].\end{aligned}$$

Finally, substituting the expression above into the definitions of risk yields:

$$\begin{aligned}R_F(\widehat{G}) &= \int_{\mathcal{B}}\int_{\mathcal{X}^*} \mathbb{1}\left[\widehat{G}(X) \neq F(B)\right] \mathrm{d}\,\mathrm{P}(X \mid B)\,\mathrm{d}\,\mathrm{P}(B) \\ &\leq \int_{\mathcal{B}}\int_{\mathcal{X}^*} \mathbb{1}\left[\left(\widehat{G}(X) \neq \widehat{F}(X)\right)\right] \mathrm{d}\,\mathrm{P}(X \mid B)\,\mathrm{d}\,\mathrm{P}(B) \\ &\quad + \int_{\mathcal{B}}\int_{\mathcal{X}^*} \mathbb{1}\left[\left(\widehat{F}(X) \neq F(B)\right)\right] \mathrm{d}\,\mathrm{P}(X \mid B)\,\mathrm{d}\,\mathrm{P}(B) \\ &= R_{\widehat{F}}(\widehat{G}) + R_F(\widehat{F}).\end{aligned}$$

□

Now, we derive a bound on the discrepancy between the empirical bag-labeling function $\widehat{F}$ and the underlying bag-labeling function F. Since this discrepancy arises when we do not sample a positive instance within a positive bag, the bound depends on the minimum bag sample size and the minimum fraction π of positive instances in every positive bag.

Lemma 4.3. *Suppose bag samples are of size at least m_l ($\forall i : m_l \leq |X_i|$), then $R_F(\widehat{F}) \leq (1-\pi)^{m_l}$.*

Proof. Given the definition of $R_F(\widehat{F})$, we can decompose it as such:

$$\begin{aligned} R_F(\widehat{F}) &= \int_{\mathcal{B}} \int_{\mathcal{X}^*} \mathbb{1}[F(B_i) \neq \widehat{F}(X_i)] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i) \\ &= \int_{\mathcal{B}_+} \int_{\mathcal{X}^*} \mathbb{1}[F(B_i) \neq \widehat{F}(X_i)] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i) \\ &\quad + \int_{\mathcal{B}_-} \int_{\mathcal{X}^*} \mathbb{1}[F(B_i) \neq \widehat{F}(X_i)] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i). \end{aligned}$$

On the set of negative bags $\mathcal{B}_-$, F and $\widehat{F}$ always agree, since only negative instances are sampled within negative bags. Therefore, the second term of the decomposition can be eliminated and we are left with:

$$R_F(\widehat{F}) = \int_{\mathcal{B}_+} \int_{\mathcal{X}^*} \mathbb{1}[F(B_i) \neq \widehat{F}(X_i)] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i).$$

Now, we observe that for a positive bag B_i, the only way that F and $\widehat{F}$ can disagree is if every instance in X_i is negative. Using basic properties of indicator functions (namely, that $\mathbb{1}\left[\bigwedge_i E_i\right] = \prod_i \mathbb{1}[E_i]$), we can use this fact to rewrite the expression above as:

$$\begin{aligned} R_F(\widehat{F}) &= \int_{\mathcal{B}_+} \int_{\mathcal{X}^*} \mathbb{1}[F(B_i) \neq \widehat{F}(X_i)] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i) \\ &= \int_{\mathcal{B}_+} \int_{\mathcal{X}^*} \mathbb{1}\left[\bigwedge\nolimits_{x_{ij} \in X_i} \big(f(x_{ij}) = 0\big)\right] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i) \\ &= \int_{\mathcal{B}_+} \int_{\mathcal{X}^*} \prod_{x_{ij} \in X_i} \mathbb{1}\left[f(x_{ij}) = 0\right] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i). \end{aligned}$$

Since the instances $x_{ij} \in X_i$ are independent, we can rewrite the integral as:

$$\begin{aligned} R_F(\widehat{F}) &= \int_{\mathcal{B}_+} \int_{\mathcal{X}^*} \prod_{x_{ij} \in X_i} \mathbb{1}\left[f(x_{ij}) = 0\right] \,\mathrm{d}\,\mathrm{P}(X_i \mid B_i) \,\mathrm{d}\,\mathrm{P}(B_i) \\ &= \int_{\mathcal{B}_+} \prod_{x_{ij} \in X_i} \left(\int_{\mathcal{X}} \mathbb{1}\left[f(x_{ij}) = 0\right] \,\mathrm{d}\,\mathrm{P}(x_{ij} \mid B_i) \right) \mathrm{d}\,\mathrm{P}(B_i) \end{aligned}$$

$$
\begin{aligned}
&\leq \int_{\mathcal{B}_+} \prod_{x_{ij} \in X_i} (1-\pi)\ \mathrm{d}\,\mathrm{P}(B_i) \\
&\leq \int_{\mathcal{B}_+} (1-\pi)^{m_l}\,\mathrm{d}\,\mathrm{P}(B_i) \\
&= (1-\pi)^{m_l} \int_{\mathcal{B}_+} \mathrm{d}\,\mathrm{P}(B_i) \\
&\leq (1-\pi)^{m_l}.
\end{aligned}
$$

□

Finally, we can now show the following learnability result with respect to the underlying bag-labeling function. However, note that in Lemma 4.3, the error $R_F(\widehat{F})$ decreases with the minimum bag size m_l. Thus, in order to achieve low error with respect to F, we must ensure that bags in the test set are sufficiently large. Therefore, the following result is stated under the additional condition that the test bag sample sizes m_i satisfy some constraints. Note that these constraints arise naturally from the process that samples instances from bag distributions.

Theorem 4.3. *Bag-labeling functions derived from instance concept class $\mathcal{F}$ with VC dimension $\mathrm{VC}(\mathcal{F})$ are PAC-learnable from MI data using*

$$O\left(\frac{m_u}{\epsilon_{\mathrm{B}}\gamma}\left(\mathrm{VC}(\mathcal{F})\log\frac{m_u}{\epsilon_{\mathrm{B}}\gamma}+\log\frac{1}{\delta}\right)\right) \tag{4.3}$$

examples when test bag sample sizes are bounded by $m_l \leq m \leq m_u$ and m_l is large enough such that $m_l \geq \frac{1}{\pi}\log\frac{2}{\epsilon_{\mathrm{B}}}$. Furthermore, if the upper bound m_u on bag sizes is also such that $m_u = O\left(\frac{1}{\epsilon_{\mathrm{B}}\pi}\right)$, then bag-labeling functions are learnable using

$$O\left(\frac{1}{\epsilon_{\mathrm{B}}^2\gamma\pi}\left(\mathrm{VC}(\mathcal{F})\log\frac{1}{\epsilon_{\mathrm{B}}\gamma\pi}+\log\frac{1}{\delta}\right)\right) \tag{4.4}$$

examples.

Proof. Intuitively, we can learn an instance-labeling function g according to Theo-

rem 4.1 and then use the resulting empirical bag-labeling function $\widehat{G}$. By combining the previously stated results, we can bound $R_F(\widehat{G})$ as:

$$\begin{aligned} R_F(\widehat{G}) &\leq R_{\widehat{F}}(\widehat{G}) + R_F(\widehat{F}) && \text{(by Lemma 4.2)} \\ &\leq m_u R_f(g) + R_F(\widehat{F}) && \text{(by Lemma 4.1)} \\ &\leq m_u R_f(g) + (1-\pi)^{m_l}. && \text{(by Lemma 4.3)} \end{aligned}$$

In the case that $\pi = 1$, then the second term in the sum is zero. Otherwise, suppose the minimum bag size is such that:

$$m_l \geq \tfrac{1}{\pi} \log \tfrac{2}{\epsilon_{\mathrm{B}}} \geq \frac{\log \frac{\epsilon_{\mathrm{B}}}{2}}{\log(1-\pi)} = \log_{1-\pi} \tfrac{\epsilon_{\mathrm{B}}}{2},$$

where the second inequality follows from the fact that $\pi \leq -\log(1-\pi)$ for $\pi \in (0,1)$. Therefore, since $(1-\pi) < 1$, we have that:

$$(1-\pi)^{m_l} \leq (1-\pi)^{\log_{1-\pi} \frac{\epsilon_{\mathrm{B}}}{2}} = \tfrac{\epsilon_{\mathrm{B}}}{2}.$$

Furthermore, when learning the instance concept g, we can choose ϵ_{I} to be such that $\epsilon_{\mathrm{I}} = \frac{\epsilon_{\mathrm{B}}}{2m_u}$. Since g will be such that $R_f(g) < \epsilon_{\mathrm{I}}$ with probability $(1-\delta)$, with the same probability we have that:

$$\begin{aligned} R_F(\widehat{G}) &\leq m_u R_f(g) + (1-\pi)^{m_l} \\ &< m_u \left(\tfrac{\epsilon_{\mathrm{B}}}{2m_u} \right) + \tfrac{\epsilon_{\mathrm{B}}}{2} = \epsilon_{\mathrm{B}}. \end{aligned}$$

Substituting the expression for ϵ_{I} in terms of ϵ_{B} into the bound in Theorem 4.1 gives the sample complexity bound in Equation 4.3, which is the same bound as stated in Theorem 4.2.

In the case that there is also a conservative upper bound on sample size $m_u =$

$O\left(\frac{1}{\epsilon_{\mathrm{B}}\pi}\right)$, which is consistent with $m_l \geq \frac{1}{\pi}\log\frac{2}{\epsilon_{\mathrm{B}}}$, we can derive an expression for learnability in terms of π. Substituting this bound into that of Theorem 4.2 gives the second sample complexity bound as stated in Equation 4.4. □

4.3 Discussion

The results presented in Section 4.1 and Section 4.2 follow the same basic strategy. First, minimum one-sided disagreement is used to learn an accurate instance concept g in the presence of one-sided noise on bag-labeled instances. Then, for the bag-labeling task, instance labels are aggregated using an empirical bag-labeling function $\widehat{G}$ to approximate the empirical bag-labeling function $\widehat{F}$ or the underlying bag-labeling function F. The idea of combining instance labels to produce a bag-labeling function is used by many existing MI algorithms, as described in Chapter 7.

However, under the generative model that treats bags as distributions, the bag-labeling results derived in Section 4.2 are somewhat counterintuitive. On the one hand, if bags are distributions from which we observe samples, then the larger the samples, the more information an algorithm has about the underlying bag distribution. Intuitively, it seems that the better an algorithm can estimate the underlying bag distribution, which is the object of interest for classification, the better it can learn a concept to label new bags. On the other hand, the result in Theorem 4.2 suggests that it is harder to learn from larger bag sizes, since roughly $O(m_u \log m_u)$ more examples are required to learn an accurate concept.

Essentially, the source of this incoherence in reasoning is the use of an instance-labeling concept g to derive the bag-labeling concept $\widehat{G}$. In the process of combining instance labels to label a bag, small errors in the instance labeling function g compound quickly. For example, g must label *all* instances in a negative bag correctly for $\widehat{G}$ to label the bag correctly. As bag size increases, it becomes less likely that g

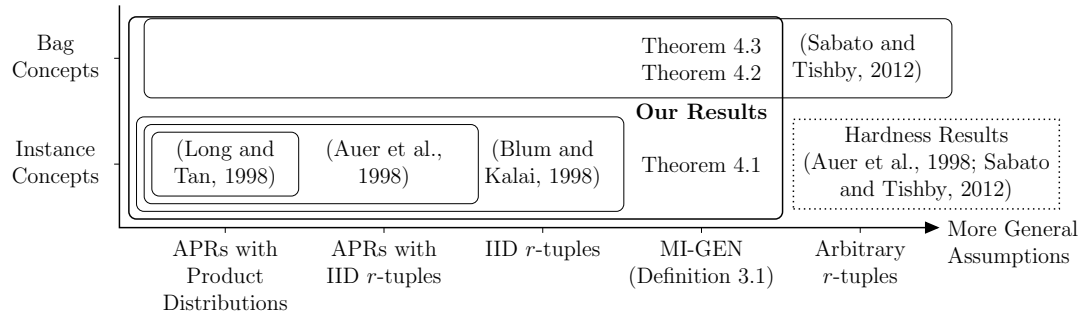


Figure 4.1: Relation to prior learnability results.

will agree with f across all instances.

Therefore, despite our positive results suggesting that learning an accurate instance-labeling function is sufficient to learn an accurate bag-labeling function, in practice, it might be advantageous to approach the instance- and bag-labeling tasks separately. That is, if one is interested in learning an accurate instance-labeling concept, then minimum one-sided disagreement can be applied. Otherwise, if one is interested in learning an accurate bag-concept, then a supervised algorithm that directly learns from labeled bag samples should be employed. We will explore this hypothesis in more detail in Chapter 6.

4.4 Relation to Prior Learnability Results

An overview of our results in the context of prior work is shown in Figure 4.1. Early work on instance learnability shows that APRs are learnable from MI data, but under the restrictive assumption that each bag contains r IID instances sampled from a product distribution (Long and Tan, 1998). Later work by Auer et al. (1998) extends these results to the case when the instance distribution is no longer a product distribution, but the instances are still sampled IID from a single distribution across bags. The most recent results on instance concept learnability in the MI setting are described by Blum and Kalai (1998). Like the proof of Theorem 4.1, Blum and Kalai (1998) also reduce the problem of learning instance concepts to learning from noisy

examples. However, the proof in Blum and Kalai (1998) requires that the label noise on negative examples be uniformly random. This condition is met under the strong assumption made in that work, that instances in all bags are drawn IID from the same distribution over instances. On the other hand, the result in Theorem 4.1 applies to our more general model in which the noise rate can vary across instances. Hence, our results rely heavily on the recent work of Simon (2012), which shows that it is possible to learn from instances corrupted with semi-random one-sided noise.

The bag learnability results in Section 4.2 show that an accurate bag concept can be learned by learning an accurate instance concept and deriving a bag concept by combining instance labels within a bag. Other recent work on bag concept learnability takes a different approach. The strategy of Sabato and Tishby (2012) is to directly learn empirical bag-labeling concepts using empirical risk minimization (ERM). That is, they suppose that an algorithm selects an instance-labeling function $g \in \mathcal{F}$ that minimizes $R_{\widehat{F}}(\widehat{G})$. As described in Section 2.4.2, general sample complexity bounds exist for empirical risk minimization (ERM) in terms of capacity measures such as VC dimension of a hypothesis class. Essentially, Sabato and Tishby (2012) proceed by proving that the capacity of the function class $\left\{\widehat{G} : \widehat{G}(X_i) = \max_j g(x_{ij}),\, g \in \mathcal{F}\right\}$ is bounded in terms of the capacity of $\mathcal{F}$. In fact, the results of Sabato and Tishby (2012) apply to more general cases in which the combining function used to derive a bag-labeling function from an instance-labeling function is other than the max function. However, by directly learning a bag-labeling function in this way, the results of Sabato and Tishby (2012) have nothing definite to say about the learnability of instance concepts.

As indicated in Figure 4.1, the results in Section 4.2 are not a strict generalization of those in Sabato and Tishby (2012), nor are those in Sabato and Tishby (2012) a generalization of those in Section 4.2. In particular, since MI-GEN treats bags as distributions, the results in Section 4.2 apply to cases not considered in Sabato

and Tishby (2012), in which bags are assumed to have finite size. On the other hand, while our generative model can encapsulate aspects of the generative model in Sabato and Tishby (2012) (see Section 3.3), arbitrary distributions over r-tuples are not permitted as in prior work.

Other recent work has discussed the difficulty, both theoretically and in practice, to relate the performance of the same classifier on the instance- and bag-labeling tasks (Tragante do Ó et al., 2011). In contrast, Lemma 4.1 illustrates a clear connection between the accuracy of an instance concept and that of the resulting empirical bag concept. Lemma 4.1 highlights an advantage of the relationship between bag and instance distributions in our generative model. In particular, Condition 1 of Definition 3.1 is employed to marginalize out the effect of individual bag distributions so that error on bags can be expressed directly in terms of the error on instances.

On the other hand, suppose we have an arbitrary empirical bag-labeling function $\widehat{G}$, not necessarily $\widehat{G}(X_i) = \max_j g(x_{ij})$, and we use it to produce instance labels by applying it to singleton bags, as in $g(x) = \widehat{G}(\{x\})$. Our model does not bound the error of the instance-labeling function f in terms of the error of $\widehat{G}$, unless it happens to be the case that $\widehat{G}(\{x\}) = g(x)$. Since many practical algorithms use a bag-level ERM approach as in the work of Sabato and Tishby (2012), the accuracy of the resulting instance-level classifiers is often not correlated in practice with bag-level accuracy (Tragante do Ó et al., 2011). Thus, one should directly learn the instance concept if a good instance-labeling function is desired.

4.5 Relation to Prior Hardness Results

The positive learnability results in Section 4.1 and Section 4.2 do not contradict existing hardness results about learning in the MI setting. Essentially, most hardness results are shown under the scenarios that lie on the far right of Figure 4.1. For

example, Sabato and Tishby (2012) observe that if only positive bags are generated, then learning the bag-labeling function is trivial, but no label information about instances is provided. In this case, learning instance labels is equivalent to learning in the *unsupervised learning* setting, for which no PAC-style guarantees can be made. However, the additional assumptions in MI-GEN preclude the case when only positive bags appear, since the negative instances would never appear in negative bags as required by Condition 3 in Definition 3.1.

Similarly, under the weak assumption in which arbitrary distributions over r-tuples are allowed, Auer et al. (1998) show that that efficiently PAC-learning MI instance concepts is impossible (unless $\mathsf{NP} = \mathsf{RP}$[2]). While the results on instance and bag learnability stemming from Theorem 4.1 show that a polynomial number of examples can be used to learn accurate concepts, they do not bound the computational complexity of learning from the examples. In particular, minimum one-sided disagreement is known to be NP-hard for certain concept classes and loss functions (Simon, 2012). Therefore, for some concept classes, instance and bag concepts are not efficiently PAC-learnable in the sense of Definition 2.2.

The apparent contradiction between our learnability results and the hardness results of Auer et al. (1998) is resolved by observing that MI-GEN precludes the scenario used to reduce learning disjunctive normal form (DNF) formulae to learning APRs from MI data. In the reduction used by Auer et al. (1998), each instance corresponds to a (variable assignment, clause) pair, and a bag is formed for each variable assignment by including a pair with that variable assignment for each clause. Bags are sampled uniformly over all variable assignments. Suppose a particular variable assignment v satisfies the first clause c_1, but not the second clause c_2. Then the instance (v, c_1) is positive, but (v, c_2) is negative. However, (v, c_2) only ever appears in bags

[2] RP is the class of decision problems for which a probabilistic Turing machine terminates in polynomial time, always returns `NO` when the answer is `NO`, and returns `YES` with probability at least $\frac{1}{2}$ when the answer is `YES`.

along with (v, c_1); that is, in positive bags. This violates the condition that $\gamma > 0$, or that negative instances appear with some probability in negative bags, so our results do not apply to this hard scenario.

Similarly, our generative model precludes scenarios used to show the hardness of learning hyperplane concepts for MI data (Kundakcioglu et al., 2010; Diochnos et al., 2012; Doran and Ray, 2013b). It is unknown whether there is an algorithm to efficiently learn hyperplanes that minimize one-sided disagreement. However, even ERM under 0–1 loss is NP-hard for the concept class of hyperplanes (Ben-David et al., 2003), which are widely used in practice for supervised learning. Thus, while previous results have characterized the hardness of MI learning as resulting from non-identical distributions across bags, our results suggest that the hardness arises from cases in which $\gamma = 0$, or when negative instances only occur in negative bags.

4.6 Must Instances be Dependent Samples?

As observed in prior work, most real-world examples of MIL have bags that contain non-IID instances (Zhou et al., 2009). Thus, our assumption that bag samples X_i are drawn IID according to their corresponding bag distributions B_i might seem unrealistic. However, note that our generative model *does* allow for dependencies between instances at the level of bag distributions, B_i. That is, although the samples X_i are drawn from bag distributions independently, we can use such independent samples to *approximate* the behavior of empirical bag-labeling functions on *non*independent samples.

As a concrete example, consider the arbitrary R-tuple model as illustrated in Figure 3.4(b). This model allows for arbitrary distributions over tuples of size at most R, which can be used to represent bags with relationships between instances. As described in Section 3.3, it is possible to represent this model within MI-GEN

where each bag is an atomic distribution over the instances in the tuple and the distribution over bags corresponds to the original distribution over tuples. Given this representation, $\pi = \frac{1}{R}$ in our model. Applying the result in Theorem 4.3, we see that if bag sizes in our model are of size $m = \left\lceil \frac{1}{\pi} \log \frac{2}{\epsilon_B} \right\rceil = \left\lceil R \log \frac{2}{\epsilon_B} \right\rceil$, then bag concepts are learnable with $O\left(\frac{R}{\epsilon_B \gamma} \log \frac{1}{\epsilon_B} \left(\mathrm{VC}(\mathcal{F}) \log \frac{R}{\epsilon_B \gamma} + \log \frac{1}{\delta}\right)\right)$ examples. If we had actually observed the R elements of the tuple, then the empirical bag-labeling function $\widehat{F}$ would be identical to the underlying bag-labeling function in our model, F. Then, according to Theorem 4.2, we could learn the underlying bag concept with $O\left(\frac{R}{\epsilon_B \gamma} \left(\mathrm{VC}(\mathcal{F}) \log \frac{R}{\epsilon_B \gamma} + \log \frac{1}{\delta}\right)\right)$ examples, given bags of size R. However, given independent samples, we now need a factor of size $O\left(\log \frac{1}{\epsilon_B}\right)$ larger bag samples and number of training examples in order to achieve the same accuracy. Thus, within a reasonable factor of size $O\left(\log \frac{1}{\epsilon_B}\right)$, we can achieve the same results using independent samples as if we had observed nonindependent samples.

4.7 Summary

In this chapter, we described new positive learnability results for learning instance or bag concepts from data generated by MI-GEN. We described how our generative model allows for learnability while excluding scenarios used to show hardness under other generative models. Nevertheless, our generative process extends prior results on instance concept learnability that are over a decade old (Blum and Kalai, 1998). We also showed that MI-GEN can incorporate the non-IID instance assumption within bag-specific distributions over instances, so assuming that samples from individual bags are drawn independently is not a restrictive assumption in our model.

Chapter 5

Learning to Rank from MI Data

Learnability results are often stated as in Chapter 4 with respect to the accuracy metric. However, other metrics often provide a more useful characterization of algorithm performance in practice. For example, for the 3D-QSAR problem, it is not necessary to accurately predict the activity of every molecule. Instead, a classifier can produce a ranked list indicating its confidence that each molecule is active. The set of active molecules with the highest predicted activity can then be investigated further by chemists.

In the 3D-QSAR example, a desirable property of a classifier is that it appropriately *ranks* bags or instances. That is, it assigns a higher real-valued confidence that a conformation is positive to actual positive conformations than to negative conformations. The AUC metric, described in Section 2.4.4, is commonly used to measure the ranking performance of a classifier. We show in this section that classifiers with high AUC are also learnable from MI data under our generative model. Unlike the prior work on learning accurate concepts from MI data as shown in Figure 4.1, there has been virtually no prior work on learning to rank in the MI setting. That is, although ranking algorithms have been developed for MIL (Bergeron et al., 2008), there is no formal analysis of the performance of such approaches. Furthermore, we show that

Table 5.1: Legend of the basic notation used in Chapter 5.

Symbol	Description/Definition
$\mathcal{X}$	Space of instances
$\mathcal{B}$	Space of bags (distributions over instances)
$\mathcal{X}^*$	Set of bag samples (sets of instances)
x_{ij}	Instance $x_{ij} \in \mathcal{X}$
B_i	Bag $B_i \in \mathcal{B}$
X_i	Bag sample $\{x_{ij}\}_{j=1}^{m_i} \in \mathcal{X}^*$, $x_{ij} \sim B_i \quad (m_l \leq \lvert X_i \rvert \leq m_u)$
$(m_l,\, m_u)\ m_i$	(Minimum, Maximum) Bag Sample Size
c	p-concept for bag-labeled instances $\quad c(x) \triangleq \mathrm{P}\left[F(B) = 1 \mid x\right]$
h	Instance-Labeling p-concept
$\widehat{H}$	Empirical Bag-Labeling p-concept $\quad \widehat{H}(X_i) \triangleq \max_j h(x_{ij})$
$p_{\mathrm{neg}},\, p$	$p_{\mathrm{neg}} \triangleq \mathrm{P}\left[f(x) = 0\right] \qquad p \triangleq \min\{p_{\mathrm{neg}}, 1 - p_{\mathrm{neg}}\}$
$\widehat{P}_{\mathrm{neg}},\, \widehat{P}$	$\widehat{P}_{\mathrm{neg}} \triangleq \mathrm{P}\left[\widehat{F}(X) = 0\right] \qquad \widehat{P} \triangleq \min\{\widehat{P}_{\mathrm{neg}}, 1 - \widehat{P}_{\mathrm{neg}}\}$
$P_{\mathrm{neg}},\, P$	$P_{\mathrm{neg}} \triangleq \mathrm{P}\left[F(B) = 0\right] \qquad P \triangleq \min\{P_{\mathrm{neg}}, 1 - P_{\mathrm{neg}}\}$

learning high-AUC concepts from MI data is easier than learning accurate concepts in the sense that it can be achieved using standard empirical risk minimization (ERM) approaches. This suggests that standard supervised algorithms can learn high-AUC concepts from MI data generated according to MI-GEN, a surprising hypothesis that we evaluate in the final section.[1]

5.1 Learning High-AUC Instance Concepts

Prior work has shown that the AUC is equivalent to the probability that a randomly selected positive example will be assigned a higher confidence than a randomly selected negative example (Hanley and McNeil, 1982). We can define a corresponding instance AUC error of a real-valued hypothesis h as $1 - \mathrm{AUC}$, or the probability that

[1] Results on instance concept learnability with respect to AUC appear in Doran and Ray (2014).

a negative instance is assigned a higher confidence than a positive instance:

$$\begin{aligned} R_f^{\mathrm{AUC}}(h) &= \int_{\mathcal{X}}\int_{\mathcal{X}} \mathbb{1}[h(x_-) > h(x_+)]\,\mathrm{d}\,\mathrm{P}(x_+ \mid f(x_+)=1)\,\mathrm{d}\,\mathrm{P}(x_- \mid f(x_-)=0) \\ &= \frac{\int_{\mathcal{X}_-}\int_{\mathcal{X}_+} \mathbb{1}[h(x_-) > h(x_+)]\,\mathrm{d}\,\mathrm{P}(x_+)\,\mathrm{d}\,\mathrm{P}(x_-)}{\mathrm{P}\left[f(x)=1\right]\mathrm{P}\left[f(x)=0\right]} \\ &= \frac{1}{(1-p_{\mathrm{neg}})p_{\mathrm{neg}}}\int_{\mathcal{X}_-}\int_{\mathcal{X}_+} \mathbb{1}[h(x_-) > h(x_+)]\,\mathrm{d}\,\mathrm{P}(x_+)\,\mathrm{d}\,\mathrm{P}(x_-). \end{aligned} \tag{5.1}$$

The first step follows from the definition of conditional probability, and we introduce $p_{\mathrm{neg}} = \mathrm{P}\left[f(x)=0\right]$ for notational convenience (see Table 5.1 for a list of notation used in this section). By definition, this quantity is zero in the cases when either all instance are positive or all instances are negative.

Given the formal definition of AUC, we can begin to describe how it is possible to learn high-AUC instance concepts from MI data. Since a classifier's confidence values are relevant for the AUC metric, we will consider the hypothesis class corresponding to a classifier to be a probabilistic concept (p-concept) class $\mathcal{C}$, as described in Section 2.4.3. For high-AUC instance learnability, we will show that it is sufficient to learn a p-concept $h \in \mathcal{C}$ that models the p-concept $c(x) = \mathrm{P}\left[F(B)=1 \mid x\right]$, the probability of observing instance x in a positive bag as defined in Equation 3.5.

To ensure that the target concept c is also a member of $\mathcal{C}$, we must formally restrict the set of bag labeling functions and distributions that are permitted by the generative model as follows:

Definition 5.1 (MI-GEN$_{\mathcal{C}}$)**.** *For any* $\gamma \in (0,1]$ *and* $\pi \in [0,1]$*:*

$$\text{MI-GEN}_{\mathcal{C}}(\gamma,\pi) \triangleq \Big\{(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}(\gamma,\pi) : \big(x \mapsto \mathrm{P}\left[F(B)=1 \mid x\right]\big) \in \mathcal{C}\Big\}.$$

Learnability of a p-concept with high AUC is then defined with respect to p-concept class $\mathcal{C}$:

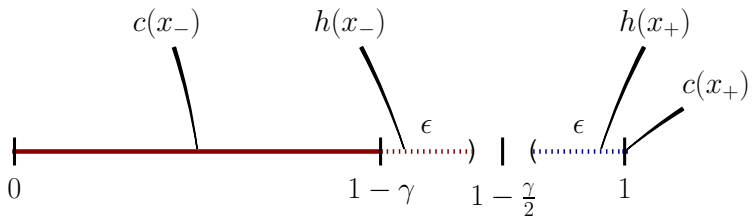


Figure 5.1: The intuition behind Theorem 5.1. A hypothesis h that closely approximates c will correctly rank instances with high probability.

Definition 5.2 (Instance MI AUC-PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ MI AUC-PAC-learns instance p-concept class $\mathcal{C}$ from MI data when for any $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in$ MI-GEN$_{\mathcal{C}}(\gamma)$ with $\gamma > 0$, and $\epsilon_{\mathrm{I}}, \delta > 0$, algorithm $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{\gamma}, \frac{1}{\epsilon_{\mathrm{I}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an instance p-concept hypothesis h with risk $R_f^{\mathrm{AUC}}(h) < \epsilon_{\mathrm{I}}$ with probability at least $1-\delta$ over samples.*

Whereas learning accurate instance concepts as in Definition 4.2 required the use of minimum one-sided disagreement, we show in Theorem 5.1 that it is possible to learn high-AUC concepts using ERM. In particular, the strategy used in the following theorem is to learn a p-concept h that models the concept c defined in Equation 3.5 using standard ERM. The intuition is that c already achieves perfect AUC; that is, $R_f^{\mathrm{AUC}}(c) = 0$. The reason is that for any negative instance x_- and positive instance x_+, $c(x_-) \leq 1-\gamma < 1 = c(x_+)$, see Figure 5.1 for an illustration. If we learn a p-concept h that closely approximates c to within some ϵ, then with high probability, h will also correctly rank instances. This reasoning is formalized in the theorem below:

Theorem 5.1. *An instance p-concept class $\mathcal{C}$ with pseudo-dimension* PD($\mathcal{C}$) *is Instance MI AUC-PAC-learnable using $O\left(\frac{1}{(\epsilon_{\mathrm{I}}\gamma p)^4}\left(\mathrm{PD}(\mathcal{C})\log\frac{1}{\epsilon_{\mathrm{I}}\gamma p} + \log\frac{1}{\delta}\right)\right)$ examples with standard ERM approaches, where $p = \min\{p_{neg}, 1-p_{neg}\}$.*

Proof. For any $c \in \mathcal{C}$, we can use ERM with respect to the quadratic loss function as described in Section 2.4.3 to learn a hypothesis h such that $\mathrm{E}\left[(h(x)-c(x))^2\right] < \epsilon$ with

probability $1 - \delta$ across samples. By Jensen's inequality, this bounds the expected absolute deviation between h and c:

$$\mathrm{E}\left[\,|h(x) - c(x)|\,\right] \leq \sqrt{\mathrm{E}\left[(h(x) - c(x))^2\right]} < \sqrt{\epsilon}.$$

Then, by Markov's inequality, this expression bounds the probability over examples that $|h(x) - c(x)|$ exceeds some constant t:

$$\mathrm{P}\left[\,|h(x) - c(x)| > t\right] \leq \frac{\mathrm{E}\left[\,|h(x) - c(x)|\,\right]}{t} < \frac{\sqrt{\epsilon}}{t}. \tag{5.2}$$

Therefore, with high probability, $|h(x) - c(x)|$ is small for small ϵ.

Now, we can proceed by following the intuition illustrated in Figure 5.1. In particular, we will show that the AUC risk is bounded when h and c agree on examples with high probability. First, suppose $|h(x) - c(x)| \leq \frac{\gamma}{2}$ for both of a pair (x_+, x_-) of positive and negative instances. Then for the negative instance, x_-, by Definition 3.1, Condition 3:

$$h(x_-) \leq c(x_-) + \tfrac{\gamma}{2} \leq (1 - \gamma) + \tfrac{\gamma}{2} = 1 - \tfrac{\gamma}{2}.$$

Similarly, for the positive instance, x_+, by Definition 3.1, Condition 2:

$$h(x_+) \geq c(x_+) - \tfrac{\gamma}{2} = 1 - \tfrac{\gamma}{2}.$$

Hence, we have that $h(x_-) \leq h(x_+)$.

By contraposition of the conclusion above, if $h(x_-) > h(x_+)$, then it is either the case that $|h(x_-) - c(x_-)| > \frac{\gamma}{2}$ or that $|h(x_+) - c(x_+)| > \frac{\gamma}{2}$. In set theoretic terms,

this means:

$$\big\{(x_+, x_-) : h(x_-) > h(x_+)\big\}$$
$$\subseteq \big\{(x_+, x_-) : |h(x_-) - c(x_-)| > \tfrac{\gamma}{2} \vee |h(x_+) - c(x_+)| > \tfrac{\gamma}{2}\big\}$$

In indicator function notation, this implies:

$$\begin{aligned}\mathbb{1}\big[h(x_-) > h(x_+)\big] &\leq \mathbb{1}\big[\,|h(x_-) - c(x_-)| > \tfrac{\gamma}{2} \vee |h(x_+) - c(x_+)| > \tfrac{\gamma}{2}\big] \\ &\leq \mathbb{1}\big[\,|h(x_-) - c(x_-)| > \tfrac{\gamma}{2}\big] + \mathbb{1}\big[\,|h(x_+) - c(x_+)| > \tfrac{\gamma}{2}\big].\end{aligned}$$

Substituting this expression into the definition of $R_f^{\mathrm{AUC}}(h)$ (Equation 5.1) yields:

$$\begin{aligned}R_f^{\mathrm{AUC}}(h) &= \frac{\int_{\mathcal{X}_-}\int_{\mathcal{X}_+} \mathbb{1}[h(x_-) > h(x_+)]\,\mathrm{d\,P}(x_+)\,\mathrm{d\,P}(x_-)}{(1-p_{\mathrm{neg}})p_{\mathrm{neg}}} \\ &\leq \frac{\int_{\mathcal{X}_-}\int_{\mathcal{X}_+} \mathbb{1}\big[\,\big|h(x_-) - c(x_-) > \tfrac{\gamma}{2}\big|\,\big]\,\mathrm{d\,P}(x_+)\,\mathrm{d\,P}(x_-)}{(1-p_{\mathrm{neg}})p_{\mathrm{neg}}} \\ &\quad + \frac{\int_{\mathcal{X}_-}\int_{\mathcal{X}_+} \mathbb{1}\big[\,|h(x_+) - c(x_+)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(x_+)\,\mathrm{d\,P}(x_-)}{(1-p_{\mathrm{neg}})p_{\mathrm{neg}}} \\ &= \frac{\int_{\mathcal{X}_-} \mathbb{1}\big[\,\big|h(x_-) - c(x_-) > \tfrac{\gamma}{2}\big|\,\big]\,\mathrm{d\,P}(x_-)}{p_{\mathrm{neg}}} \\ &\quad + \frac{\int_{\mathcal{X}_+} \mathbb{1}\big[\,|h(x_+) - c(x_+)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(x_+)}{1-p_{\mathrm{neg}}}.\end{aligned}$$

Then, using the definition $p = \min\{p_{\mathrm{neg}}, 1-p_{\mathrm{neg}}\}$, this becomes:

$$\begin{aligned}R_f^{\mathrm{AUC}}(h) &\leq \frac{\int_{\mathcal{X}_-} \mathbb{1}\big[\,\big|h(x_-) - c(x_-) > \tfrac{\gamma}{2}\big|\,\big]\,\mathrm{d\,P}(x_-)}{p} \\ &\quad + \frac{\int_{\mathcal{X}_+} \mathbb{1}\big[\,|h(x_+) - c(x_+)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(x_+)}{p} \\ &= \frac{\int_{\mathcal{X}} \mathbb{1}\big[\,\big|h(x) - c(x) > \tfrac{\gamma}{2}\big|\,\big]\,\mathrm{d\,P}(x)}{p} = \frac{\mathrm{P}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big]}{p}.\end{aligned}$$

Finally, using the inequality derived in Equation 5.2, we have:

$$R_f^{\text{AUC}}(h) \leq \frac{\text{P}\left[\,|h(x) - c(x)| > \frac{\gamma}{2}\right]}{p} < \frac{2\sqrt{\epsilon}}{\gamma p}.$$

Therefore, it is sufficient to choose $\epsilon = \frac{(\epsilon_{\text{I}}\gamma p)^2}{4}$ when learning h via ERM as so that $R_f^{\text{AUC}}(h) < \epsilon_{\text{I}}$.

Finally, the sample complexity bound results from substituting $\epsilon = \frac{(\epsilon_{\text{I}}\gamma p)^2}{4}$ into the existing bound in Equation 2.11 for learning p-concepts using ERM (Kearns and Schapire, 1994). □

Comparing Theorem 5.1 with Theorem 4.1 on learning accurate instance concepts, we see that neither results require that positive instances appear in positive bags ($\pi > 0$). In both cases, the addition label noise affects γ, but is tolerated by the underlying algorithm. The key difference between these results is that high-AUC concepts can be learned via standard ERM approaches, whereas accurate concept learning requires minimum one-sided disagreement. Additionally, the sample complexity bound in Theorem 5.1 contains an additional factor p that accounts for class imbalance. Intuitively, this factor appears because it is difficult to learn to effectively rank instances from different classes when one class appears very infrequently in the training set (p is small).

5.2 Learning High-AUC Bag Concepts

As for accuracy, we might be interested in learning either high-AUC instance *or* bag concepts from MI data. Following a similar strategy as employed in Section 4.2 for learning accurate bag concepts, here we will consider two measures of bag-level performance of a bag concept $\widehat{H}$ derived from an instance concept h. The same combining function as in Chapter 4, $\widehat{H}(X_i) = \max_j h(x_{ij})$, is commonly used to derive

real-valued bag-labeling functions in prior work (Ray and Craven, 2005). Following the analysis in Section 4.2, we will measure performance of $\widehat{H}$ with respect to both $\widehat{F}$, the empirical bag-labeling function, and later F, the underlying bag-labeling function.

For the empirical bag-labeling function, $\widehat{F}$, the intuitive definition of AUC is the probability that a bag-level hypothesis $\widehat{H}$ assigns a higher value to a bag sample given that it is labeled positive by $\widehat{F}$ (that is, containing at least one positive instance) than another bag sample labeled negative by $\widehat{F}$ (containing no positive instances). Formally, we can define the corresponding AUC-based risk as follows:

$$\begin{aligned} R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) &= \frac{\begin{array}{c}\int_{\mathcal{B}}\int_{\mathcal{B}}\int_{\mathcal{X}^*_-}\int_{\mathcal{X}^*_+} \mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big]\ldots \\ \ldots\,\mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_+)\,\mathrm{d\,P}(B_-)\end{array}}{\mathrm{P}\left[\widehat{F}(X)=1\right]\mathrm{P}\left[\widehat{F}(X)=0\right]} \\ &= \frac{\begin{array}{c}\int_{\mathcal{B}}\int_{\mathcal{B}}\int_{\mathcal{X}^*_-}\int_{\mathcal{X}^*_+} \mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big]\ldots \\ \ldots\,\mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_+)\,\mathrm{d\,P}(B_-)\end{array}}{(1-\widehat{P}_{\mathrm{neg}})\widehat{P}_{\mathrm{neg}}}. \end{aligned} \tag{5.3}$$

Above, $\mathcal{X}^*_-$ is the set of all negative bag samples, and $\mathcal{X}^*_+$ the set of all positive bag samples. The notation $\widehat{P}_{\mathrm{neg}} = \Pr\left[\widehat{F}(X)=0\right]$ is used for convenience. Now, we can define learnability with respect to this metric:

Definition 5.3 (Empirical Bag MI AUC-PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ MI AUC-PAC-learns empirical bag-labeling functions derived from p-concept class $\mathcal{C}$ when for any $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}_{\mathcal{C}}(\gamma)$ with $\gamma > 0$, and $\epsilon_{\mathrm{B}}, \delta > 0$, algorithm $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{\gamma}, \frac{1}{\epsilon_{\mathrm{B}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an empirical bag-labeling function $\widehat{H}$ with risk $R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) < \epsilon_{\mathrm{B}}$ with probability at least $1-\delta$ over samples.*

We will now show learnability of empirical bag-labeling functions by reducing the problem to learning an accurate model of the p-concept c. Hence, the approach of the proof follows that for learning accurate empirical bag-labeling functions. However,

because the relationship between instance- and bag-level AUC is more complex than for accuracy, there is no analog to Lemma 4.1. Instead, we proceed directly with the result:

Theorem 5.2. *Empirical bag-labeling functions derived from p-concept class $\mathcal{C}$ with pseudo-dimension* $\mathrm{PD}(\mathcal{C})$ *are AUC-PAC-learnable from MI data using*

$$O\left(\frac{m_u^4}{\left(\epsilon_{\mathrm{B}}\gamma\widehat{P}\right)^4}\left(\mathrm{PD}(\mathcal{C})\log\frac{m_u}{\left(\epsilon_{\mathrm{B}}\gamma\widehat{P}\right)}+\log\tfrac{1}{\delta}\right)\right)$$

examples with standard ERM approaches, where

$$\widehat{P}\triangleq\min\{\widehat{P}_{neg},1-\widehat{P}_{neg}\}\geq\min\{P_{neg},1-p_{neg}\},$$

and m_u is an upper bound on bag sample size.

Proof. As in Theorem 5.1, we will learn a p-concept h to model c accurately with high probability. Then, given bag samples X_+ with at least one positive instance and X_- with all negative instances, suppose that $|h(x)-c(x)|\leq\frac{\gamma}{2}$ for *all* instances across both samples. Then by the same argument as in Theorem 5.1 as illustrated in Figure 5.1, at least one instance in X_+ is assigned a label by h that is at least $1-\frac{\gamma}{2}$, and all instances in X_- are assigned a label by h of at most $1-\frac{\gamma}{2}$. Therefore, the maximum label assigned in X_+, $\widehat{H}(X_+)$, is greater than or equal to the maximum label in X_-, $\widehat{H}(X_-)$.

By contraposition, if $\widehat{H}(X_-)>\widehat{H}(X_+)$, then the label $h(x)$ of *some* instance x in either X_+ or X_- deviates by more than $\frac{\gamma}{2}$ from $c(x)$. That is:

$$\begin{aligned}&\left\{(X_+,X_-):\widehat{H}(X_-)>\widehat{H}(X_+)\right\}\\&\quad\subseteq\left\{(X_+,X_-):\left(\bigvee_{x\in X_+}|h(x)-c(x)|>\tfrac{\gamma}{2}\right)\vee\left(\bigvee_{x\in X_-}|h(x)-c(x)|>\tfrac{\gamma}{2}\right)\right\}\end{aligned}$$

Therefore, in indicator function notation:

$$\mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big] \leq \sum_{x \in X_+} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big] + \sum_{x \in X_-} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big].$$

Using the inequality above in the definition of $R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H})$ in Equation 5.3 gives:

$$\begin{aligned} R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) \leq & \frac{\int_{\mathcal{B}} \int_{\mathcal{B}} \int_{\mathcal{X}^*_-} \int_{\mathcal{X}^*_+} \sum_{x \in X_+} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big] \ldots}{\ldots \,\mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_+)\,\mathrm{d\,P}(B_-)} \\ & \qquad (1 - \widehat{P}_{\mathrm{neg}})\widehat{P}_{\mathrm{neg}} \\ & + \frac{\int_{\mathcal{B}} \int_{\mathcal{B}} \int_{\mathcal{X}^*_-} \int_{\mathcal{X}^*_+} \sum_{x \in X_-} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big] \ldots}{\ldots \,\mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_+)\,\mathrm{d\,P}(B_-)} \\ & \qquad (1 - \widehat{P}_{\mathrm{neg}})\widehat{P}_{\mathrm{neg}} \end{aligned}$$

Since the integrands above only depend on X_+ and X_-, we can rewrite the expression using the fact that

$$\begin{aligned} \int_{\mathcal{X}^*_-} \mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_-) &= \mathrm{P}\left[\widehat{F}(X) = 0\right] = \widehat{P}_{\mathrm{neg}} \\ \int_{\mathcal{X}^*_+} \mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(B_+) &= \mathrm{P}\left[\widehat{F}(X) = 1\right] = 1 - \widehat{P}_{\mathrm{neg}}. \end{aligned}$$

The result is:

$$\begin{aligned} R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) &\leq \frac{\int_{\mathcal{B}} \int_{\mathcal{X}^*_+} \sum_{x \in X_+} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(B_+)}{(1 - \widehat{P}_{\mathrm{neg}})} \\ &\quad + \frac{\int_{\mathcal{B}} \int_{\mathcal{X}^*_-} \sum_{x \in X_-} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_-)}{\widehat{P}_{\mathrm{neg}}} \\ &\leq \frac{\int_{\mathcal{B}} \int_{\mathcal{X}^*_+} \sum_{x \in X_+} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(X_+ \mid B_+)\,\mathrm{d\,P}(B_+)}{\widehat{P}} \\ &\quad + \frac{\int_{\mathcal{B}} \int_{\mathcal{X}^*_-} \sum_{x \in X_-} \mathbb{1}\big[\,|h(x) - c(x)| > \tfrac{\gamma}{2}\big]\,\mathrm{d\,P}(X_- \mid B_-)\,\mathrm{d\,P}(B_-)}{\widehat{P}} \end{aligned}$$

$$= \frac{\int_{\mathcal{B}} \int_{\mathcal{X}^*} \sum_{x \in X} \mathbb{1}\left[\, |h(x) - c(x)| > \frac{\gamma}{2} \right] \mathrm{d}\,\mathrm{P}(X \mid B)\, \mathrm{d}\,\mathrm{P}(B)}{\widehat{P}}.$$

By the independence of instances $x \in X$, the upper bound m_u on bag size, and Condition 1 in Definition 3.1, we can rewrite the expression above as:

$$\begin{aligned} R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) &\leq \frac{m_u}{\widehat{P}} \int_{\mathcal{B}} \int_{\mathcal{X}} \mathbb{1}\left[\, |h(x) - c(x)| > \tfrac{\gamma}{2} \right] \mathrm{d}\,\mathrm{P}(x \mid B)\, \mathrm{d}\,\mathrm{P}(B) \\ &= \frac{m_u}{\widehat{P}} \int_{\mathcal{X}} \mathbb{1}\left[\, |h(x) - c(x)| > \tfrac{\gamma}{2} \right] \mathrm{d}\,\mathrm{P}(x) \\ &= \frac{m_u}{\widehat{P}}\, \mathrm{P}\left[\, |h(x) - c(x)| > \tfrac{\gamma}{2} \right]. \end{aligned}$$

Then, by Markov's inequality in Equation 5.2,

$$R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) \leq \frac{m_u}{\widehat{P}}\, \mathrm{P}\left[\, |h(x) - c(x)| > \tfrac{\gamma}{2} \right] < \frac{2 m_u \sqrt{\epsilon}}{\gamma \widehat{P}}. \tag{5.4}$$

Therefore, choosing $\epsilon = \frac{\left(\epsilon_{\mathrm{B}} \gamma \widehat{P}\right)^2}{4 m_u^2}$, is sufficient to learn $\widehat{H}$ with $R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) < \epsilon_{\mathrm{B}}$. Substituting this ϵ into the bounds in Equation 2.11 gives the sample complexity of learning $\widehat{H}$ as stated in the theorem.

Finally, we show that $\widehat{P} \geq \min\{P_{\mathrm{neg}}, 1 - p_{\mathrm{neg}}\}$ as asserted in the theorem, which demonstrates that $\widehat{P}$ is independent of the bag size m (so there is no hidden dependence on bag size). First, observe that $\widehat{P}_{\mathrm{neg}} \geq P_{\mathrm{neg}}$. The reason is that whenever a negative *bag* is sampled, a sample of only negative instances is guaranteed to be sampled from the bag. Thus, the probability of a negative sample of instances is at least the probability of sampling a negative bag.

Additionally, $1 - \widehat{P}_{\mathrm{neg}} \geq 1 - p_{\mathrm{neg}}$. This is true because the probability of a sample containing a positive instance is at least the probability that the very first instance sampled is positive, which is $1 - p_{\mathrm{neg}}$.

Combining the observations above, we get:

$$\begin{aligned}\widehat{P} &\triangleq \min\{\widehat{P}_{\text{neg}}, 1 - \widehat{P}_{\text{neg}}\} \\ &\geq \min\{P_{\text{neg}}, 1 - p_{\text{neg}}\}.\end{aligned}$$

□

Note that Theorem 5.1 is a special case of Theorem 5.2 when $m_u = 1$. In this case $\widehat{P}_{\text{neg}} = p_{\text{neg}}$ when samples all have size 1, so $\widehat{P} = p$ and the sample complexity is the same.

Now, we can examine AUC-learnability with respect to the true bag-labeling function, F. To define AUC with respect to F, we measure the probability that a sample X_+ is labeled higher by $\widehat{H}$ than X_- is, given that X_+ is sampled from a positive bag and X_- is sampled from a negative bag. Formally, the AUC risk of $\widehat{H}$ with respect to F is:

$$\begin{aligned}R_F^{\text{AUC}}(\widehat{H}) &= \frac{\begin{array}{l}\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}^*}\int_{\mathcal{X}^*} \mathbb{1}\left[\widehat{H}(X_-) > \widehat{H}(X_+)\right]\ldots \\ \ldots\,\mathrm{d}\,\mathrm{P}(X_+ \mid B_+)\,\mathrm{d}\,\mathrm{P}(X_- \mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)\end{array}}{\mathrm{P}\left[F(B) = 1\right]\mathrm{P}\left[F(B) = 0\right]} \\ &= \frac{\begin{array}{l}\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}^*}\int_{\mathcal{X}^*} \mathbb{1}\left[\widehat{H}(X_-) > \widehat{H}(X_+)\right]\ldots \\ \ldots\,\mathrm{d}\,\mathrm{P}(X_+ \mid B_+)\,\mathrm{d}\,\mathrm{P}(X_- \mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)\end{array}}{(1 - P_{\text{neg}})P_{\text{neg}}}.\end{aligned} \tag{5.5}$$

The notation $P_{\text{neg}} = \mathrm{P}\left[F(B) = 0\right]$ is used to denote the probability of sampling a negative bag from the distribution over bags. We define the risk to be zero in the case that this probability is equal to either 0 or 1.

As for accuracy, the risk of an empirical bag-labeling function now depends on how representative a sample is of the underlying bag. Thus, in the definition of AUC-learnability with respect to F (Definition 5.4), we now again require an additional

assumption that positive instances appear some $\pi > 0$ fraction of the time in positive bags.

Definition 5.4 (Bag MI AUC-PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ MI AUC-PAC-learns bag-labeling functions derived from p-concept class $\mathcal{C}$ when for any $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}_{\mathcal{C}}(D_{\mathcal{X}}, f, \gamma, \pi)$ with $\gamma, \pi > 0$, and $\epsilon_{\text{B}}, \delta > 0$, algorithm $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{\gamma}, \frac{1}{\pi}, \frac{1}{\epsilon_{\text{B}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an empirical bag-labeling function $\widehat{H}$ with risk $R_F^{\text{AUC}}(\widehat{H}) < \epsilon_{\text{B}}$ with probability at least $1 - \delta$ over samples.*

Again, we will learn an instance p-concept h that models c, and then show that a sufficiently accurate p-concept can produce an empirical bag-labeling function $\widehat{H}$ that models F with high AUC. To do this, we will show that the AUC error of $\widehat{H}$ with respect to F, $R_F^{\text{AUC}}(\widehat{H})$, is bounded in terms of the AUC error of $\widehat{H}$ with respect to $\widehat{F}$, $R_{\widehat{F}}^{\text{AUC}}(\widehat{H})$.

Lemma 5.1. *Suppose bag samples are of size at least m_l ($\forall i : m_l \leq |X_i|$), then $R_F^{\text{AUC}}(\widehat{H}) \leq \frac{1}{P_{neg}} R_{\widehat{F}}^{\text{AUC}}(\widehat{H}) + (1-\pi)^{m_l}$.*

Proof. We can derive the inequality in Lemma 5.1 by transforming the definition of $R_F^{\text{AUC}}(\widehat{H})$ in Equation 5.5 to that of $R_{\widehat{F}}^{\text{AUC}}(\widehat{H})$ in Equation 5.3. Starting from $R_F^{\text{AUC}}(\widehat{H})$, we get:

$$
\begin{aligned}
R_F^{\text{AUC}}(\widehat{H}) &= \frac{\begin{array}{l}\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}^*}\int_{\mathcal{X}^*} \mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big]\ldots \\ \ldots\,\mathrm{d}\,\mathrm{P}(X_+ \mid B_+)\,\mathrm{d}\,\mathrm{P}(X_- \mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)\end{array}}{(1-P_{\text{neg}})P_{\text{neg}}} \\
&= \frac{\begin{array}{l}\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}^*}\int_{\mathcal{X}_+^*} \mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big]\ldots \\ \ldots\,\mathrm{d}\,\mathrm{P}(X_+ \mid B_+)\,\mathrm{d}\,\mathrm{P}(X_- \mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)\end{array}}{(1-P_{\text{neg}})P_{\text{neg}}} \qquad (A)
\end{aligned}
$$

$$\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}^*}\int_{\mathcal{X}_-^*}\mathbb{1}\big[\widehat{H}(X_-)>\widehat{H}(X_+)\big]\ldots$$
$$+\frac{\ldots\,\mathrm{d}\,\mathrm{P}(X_+\mid B_+)\,\mathrm{d}\,\mathrm{P}(X_-\mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)}{(1-P_{\text{neg}})P_{\text{neg}}}. \qquad (B)$$

Starting with (B), we see that since $\mathbb{1}\big[\widehat{H}(X_-)>\widehat{H}(X_+)\big]\leq 1$, we can rewrite this term as:

$$\begin{aligned}(\text{B}) &\leq \frac{\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}^*}\int_{\mathcal{X}_-^*}\mathrm{d}\,\mathrm{P}(X_+\mid B_+)\,\mathrm{d}\,\mathrm{P}(X_-\mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)}{(1-P_{\text{neg}})P_{\text{neg}}}\\ &= \frac{\int_{\mathcal{B}_+}\int_{\mathcal{X}_-^*}\mathrm{d}\,\mathrm{P}(X_+\mid B_+)\,\mathrm{d}\,\mathrm{P}(B_+)}{(1-P_{\text{neg}})}\\ &\leq \frac{\int_{\mathcal{B}_+}(1-\pi)^{m_l}\,\mathrm{d}\,\mathrm{P}(B_+)}{(1-P_{\text{neg}})} = (1-\pi)^{m_l}.\end{aligned}$$

The second step follows from the fact that $\int_{\mathcal{X}_-^*}\mathrm{d}\,\mathrm{P}(X_+\mid B_+)$ is the probability of sampling only negative instances within a positive bag of size at least m_l, which is at most $(1-\pi)^{m_l}$.

Continuing with term (A), we can rewrite this as:

$$\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}_-^*}\int_{\mathcal{X}_+^*}\mathbb{1}\big[\widehat{H}(X_-)>\widehat{H}(X_+)\big]\ldots$$
$$(\text{A}) = \frac{\ldots\,\mathrm{d}\,\mathrm{P}(X_+\mid B_+)\,\mathrm{d}\,\mathrm{P}(X_-\mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)}{(1-P_{\text{neg}})P_{\text{neg}}} \qquad (C)$$
$$\int_{\mathcal{B}_-}\int_{\mathcal{B}_+}\int_{\mathcal{X}_+^*}\int_{\mathcal{X}_+^*}\mathbb{1}\big[\widehat{H}(X_-)>\widehat{H}(X_+)\big]\ldots$$
$$+\frac{\ldots\,\mathrm{d}\,\mathrm{P}(X_+\mid B_+)\,\mathrm{d}\,\mathrm{P}(X_-\mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)}{(1-P_{\text{neg}})P_{\text{neg}}}. \qquad (D)$$

Now, we see that (D) = 0, since it involves an integral over bags with positive instances in negative bags, which occurs with probability zero by Condition 2 in Definition 3.1. For (C), since $0\leq\mathbb{1}\big[\widehat{H}(X_-)>\widehat{H}(X_+)\big]$, we can bound this term by taking the

outermost integrals with respect to the entire bag space:

$$\begin{aligned}(\mathrm{C}) &\leq \frac{\begin{array}{r}\int_{\mathcal{B}}\int_{\mathcal{B}}\int_{\mathcal{X}_-^*}\int_{\mathcal{X}_+^*}\mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big]\ldots \\ \ldots\,\mathrm{d}\,\mathrm{P}(X_+ \mid B_+)\,\mathrm{d}\,\mathrm{P}(X_- \mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)\end{array}}{(1-P_{\mathrm{neg}})P_{\mathrm{neg}}} \\ &\leq \left(\frac{(1-\widehat{P}_{\mathrm{neg}})\widehat{P}_{\mathrm{neg}}}{(1-P_{\mathrm{neg}})P_{\mathrm{neg}}}\right)\frac{\begin{array}{r}\int_{\mathcal{B}}\int_{\mathcal{B}}\int_{\mathcal{X}_-^*}\int_{\mathcal{X}_+^*}\mathbb{1}\big[\widehat{H}(X_-) > \widehat{H}(X_+)\big]\ldots \\ \ldots\,\mathrm{d}\,\mathrm{P}(X_+ \mid B_+)\,\mathrm{d}\,\mathrm{P}(X_- \mid B_-)\,\mathrm{d}\,\mathrm{P}(B_+)\,\mathrm{d}\,\mathrm{P}(B_-)\end{array}}{(1-\widehat{P}_{\mathrm{neg}})\widehat{P}_{\mathrm{neg}}}.\end{aligned}$$

Using the definition of $R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H})$ in Equation 5.3, we get that:

$$(\mathrm{C}) \leq \left(\tfrac{(1-\widehat{P}_{\mathrm{neg}})\widehat{P}_{\mathrm{neg}}}{(1-P_{\mathrm{neg}})P_{\mathrm{neg}}}\right) R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) \leq \tfrac{1}{P_{\mathrm{neg}}} R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}).$$

The second inequality results by observing that samples of only negative instances can be sampled within negative *or* positive bags, so $P_{\mathrm{neg}} \leq \widehat{P}_{\mathrm{neg}} \leq 1$ and $1 - P_{\mathrm{neg}} \geq 1 - \widehat{P}_{\mathrm{neg}}$.

Combining the terms above, we have that:

$$\begin{aligned}R^{\mathrm{AUC}}_{F}(\widehat{H}) &= (\mathrm{A}) + (\mathrm{B}) = \big((\mathrm{C}) + (\mathrm{D})\big) + (\mathrm{B}) \\ &\leq \tfrac{1}{P_{\mathrm{neg}}} R^{\mathrm{AUC}}_{\widehat{F}}(\widehat{H}) + (1-\pi)^{m_l}.\end{aligned}$$

□

Finally, given the bound in Lemma 5.1, we can derive a result on learning high-AUC bag concepts with respect to the underlying bag-labeling function F. As with the results in Theorem 4.3, we state the results conditioned on the fact that bag sizes m_i respect some constraints to account for the error that naturally results from insufficiently large samples of instances in positive bags.

Theorem 5.3. *Bag-labeling functions derived from p-concept class $\mathcal{C}$ with pseudo-*

dimension $\mathrm{PD}(\mathcal{C})$ *are AUC-PAC-learnable from MI data using*

$$O\left(\frac{m_u^4}{\left(\epsilon_{\mathrm{B}}\gamma\widehat{P}P_{neg}\right)^4}\left(\mathrm{PD}(\mathcal{C})\log\frac{m_u}{\left(\epsilon_{\mathrm{B}}\gamma\widehat{P}P_{neg}\right)}+\log\tfrac{1}{\delta}\right)\right). \tag{5.6}$$

examples using standard ERM approaches when bag sample sizes are bounded by $m_l \leq m \leq m_u$ *and* $m_l \geq \frac{1}{\pi}\log\frac{2}{\epsilon_{\mathrm{B}}}$*, where* $\widehat{P} = \min\{\widehat{P}_{neg}, 1-\widehat{P}_{neg}\}$*. Additionally, if the maximum bag size* m_u *is such that* $m_u = O\left(\frac{1}{\epsilon_{\mathrm{B}}\pi}\right)$*, then learnability is possible with*

$$O\left(\frac{1}{\left(\epsilon_{\mathrm{B}}^2\gamma\pi\widehat{P}P_{neg}\right)^4}\left(\mathrm{PD}(\mathcal{C})\log\frac{1}{\left(\epsilon_{\mathrm{B}}\gamma\pi\widehat{P}P_{neg}\right)}+\log\tfrac{1}{\delta}\right)\right). \tag{5.7}$$

examples.

Proof. As in Theorem 5.2, we will use ERM to learn a p-concept h to model c accurately with high probability. Then, we can bound $R_F^{\mathrm{AUC}}(\widehat{H})$ using:

$$\begin{aligned} R_F^{\mathrm{AUC}}(\widehat{H}) &\leq \frac{1}{P_{\mathrm{neg}}}R_{\widehat{F}}^{\mathrm{AUC}}(\widehat{H}) + (1-\pi)^{m_l} && \text{(by Lemma 5.1)} \\ &\leq \frac{2m_u\sqrt{\epsilon}}{\gamma\widehat{P}P_{\mathrm{neg}}} + (1-\pi)^{m_l}. && \text{(by Theorem 5.2, Equation 5.4)} \end{aligned}$$

In the case that $\pi = 1$, then the second term in the sum is zero. Otherwise, suppose the minimum bag size is such that:

$$m_l \geq \tfrac{1}{\pi}\log\tfrac{2}{\epsilon_{\mathrm{B}}} \geq \frac{\log\frac{\epsilon_{\mathrm{B}}}{2}}{\log(1-\pi)} = \log_{1-\pi}\tfrac{\epsilon_{\mathrm{B}}}{2},$$

where the second inequality follows from the fact that $\pi \leq -\log(1-\pi)$ for $\pi \in (0,1)$. Therefore, since $(1-\pi) < 1$, we have that:

$$(1-\pi)^{m_l} \leq (1-\pi)^{\log_{1-\pi}\frac{\epsilon_{\mathrm{B}}}{2}} = \tfrac{\epsilon_{\mathrm{B}}}{2}.$$

Furthermore, when learning the instance p-concept h, we can choose ϵ to be such

that $\epsilon = \left(\frac{\epsilon_{\mathrm{B}}\gamma\widehat{P}P_{\mathrm{neg}}}{4m_u}\right)^2$. By the bound on $R_F^{\mathrm{AUC}}(\widehat{H})$, with probability $(1-\delta)$, we have:

$$\begin{aligned} R_F^{\mathrm{AUC}}(\widehat{H}) &\leq \frac{2m_u\sqrt{\epsilon}}{\gamma\widehat{P}P_{\mathrm{neg}}} + (1-\pi)^{m_l} \\ &\leq \frac{\epsilon_{\mathrm{B}}}{2} + \frac{\epsilon_{\mathrm{B}}}{2} = \epsilon_{\mathrm{B}}. \end{aligned}$$

Substituting the expression for ϵ in terms of ϵ_{B} into the bound in Equation 2.11 for learning p-concepts using ERM (Kearns and Schapire, 1994) gives a sample complexity of:

$$O\left(\frac{m_u^4}{\left(\epsilon_{\mathrm{B}}\gamma\widehat{P}P_{\mathrm{neg}}\right)^4}\left(\mathrm{PD}(\mathcal{C})\log\frac{m_u}{\left(\epsilon_{\mathrm{B}}\gamma\widehat{P}P_{\mathrm{neg}}\right)} + \log\tfrac{1}{\delta}\right)\right)$$

as stated in Equation 5.6. If there is also a conservative upper bound on bag sample size $m_u = O\left(\frac{1}{\epsilon_{\mathrm{B}}\pi}\right)$, then by substituting this bound into that above gives the bound as stated in Equation 5.7. □

5.3 Learning High-AUC MI Concepts with Noise

Section 3.5 describes how we can generalize MI-GEN to account for positive instances occurring some of the time in negative bags. Such scenarios correspond to a subset of those allowed under the Generalized MIL (GMIL) setting (Scott et al., 2005). Unfortunately, if positive instances occur in negative bags, then the label noise present in bag-labeled instances becomes two-sided. In this case, we may no longer be able to use a minimum one-sided disagreement strategy to learn accurate concepts.

On the other hand, as we show in this section, it is still possible to learn high-AUC instance concepts if positive instances occur a small fraction of the time in negative bags. However, we only discuss the learnability of instance concepts, not bag concepts. The reason is that the max combining function can no longer be used

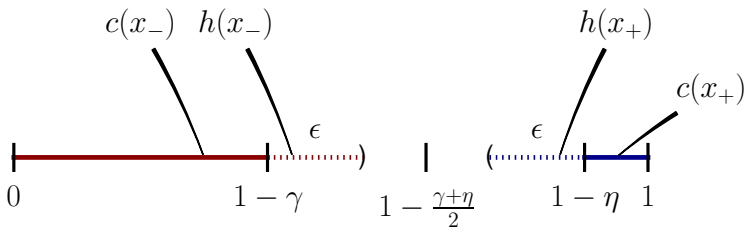


Figure 5.2: The intuition behind Theorem 5.4. A hypothesis h that closely approximates c will correctly rank instances with high probability when $\eta < \gamma$.

to derive bag labels from instance labels. In Chapter 6, we discuss a more general GMIL setting (see Definition 6.1), in which bag-level learners can be directly applied using kernel methods.

The intuition behind why we can learn high-AUC instance concepts in the presence of noisy bags is shown in Figure 5.2. As in Figure 5.1, we can learn a p-concept h that models c, the probability of an instance appearing in a positive bag. For a negative instance x_-, $c(x_-) \leq 1-\gamma$ as before by Condition 2 in Definition 3.2. However, now for a positive instance x_+, $c(x_+)$ is no longer 1, but is at least $1-\eta$ by Condition 1 in Definition 3.2. However, as Figure 5.2 shows, there is still a "margin" between the values assigned by c to positive and negative instances, so an accurate model h of c can still rank instances appropriately.

The theorem on instance AUC learnability under NMI-GEN is analogous to Theorem 5.1 with modified definitions of learnability. First, we define the set of bag labeling functions and distributions consistent with a p-concept class:

Definition 5.5 (NMI-GEN$_{\mathcal{C}}$)**.** *For any* $\gamma \in (0,1]$ *and* $\eta \in [0,\gamma)$*:*

$$\text{NMI-GEN}_{\mathcal{C}}(\gamma,\eta) \triangleq$$
$$\Big\{(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{NMI-GEN}(\gamma,\eta) : \big(x \mapsto \mathrm{P}\left[F(B)=1 \mid x\right]\big) \in \mathcal{C}\Big\}.$$

Then, we can define learnability with respect to this class of noisy MI concepts:

Definition 5.6 (Instance NMI AUC-PAC-learning)**.** *We say that an algorithm $\mathcal{A}$ NMI AUC-PAC-learns instance p-concept class $\mathcal{C}$ from MI data when for any tuple $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{NMI-GEN}_{\mathcal{C}}(\gamma, \eta)$ with $\gamma > 0$, $0 \leq \eta < \gamma$, and $\epsilon_{\text{I}}, \delta > 0$, $\mathcal{A}$ requires $O\big(\mathsf{poly}(\frac{1}{(\gamma-\eta)}, \frac{1}{\epsilon_{\text{I}}}, \frac{1}{\delta})\big)$ bag-labeled instances sampled independently from the MI generative process in Figure 3.1(c) to produce an instance hypothesis h with risk $R_f^{\text{AUC}}(h) < \epsilon_{\text{I}}$ with probability at least $1 - \delta$ over samples.*

Finally, we can state the theorem below:

Theorem 5.4. *An instance p-concept class $\mathcal{C}$ with pseudo-dimension $\text{PD}(\mathcal{C})$ is Instance NMI AUC-PAC-learnable using $O\left(\frac{1}{(\epsilon_{\text{I}}(\gamma-\eta)p)^4}\left(\text{PD}(\mathcal{C}) \log \frac{1}{\epsilon_{\text{I}}(\gamma-\eta)p} + \log \frac{1}{\delta}\right)\right)$ examples with standard ERM approaches, where $p = \min\{p_{neg}, 1 - p_{neg}\}$.*

Proof. For any $c \in \mathcal{C}$, we start as in Theorem 5.1 by using ERM to learn a hypothesis h such that $\text{E}\left[(h(x) - c(x))^2\right] < \epsilon$ with probability $1 - \delta$ across samples.

Then, we can follow the strategy illustrated in Figure 5.2. In particular, we will show that the AUC risk is bounded when h and c agree on examples with high probability. Suppose $|h(x) - c(x)| \leq \frac{\gamma-\eta}{2}$ for both of a pair (x_+, x_-) of positive and negative instances. Then for the negative instance, x_-, by Definition 3.2, Condition 3:

$$h(x_-) \leq c(x_-) + \tfrac{\gamma-\eta}{2} \leq (1 - \gamma) + \tfrac{\gamma-\eta}{2} = 1 - \tfrac{\gamma+\eta}{2}.$$

For the positive instance, x_+, by Definition 3.2, Condition 2:

$$h(x_+) \geq c(x_+) - \tfrac{\gamma-\eta}{2} = 1 - \eta - \tfrac{\gamma-\eta}{2} = 1 - \tfrac{\gamma+\eta}{2}.$$

Hence, we have that $h(x_-) \leq h(x_+)$.

Following this observation, we can proceed exactly as in Theorem 5.1 using $\frac{\gamma-\eta}{2}$

rather than $\frac{\gamma}{2}$ as the bound on $|h(x) - c(x)|$. Accordingly, we find that:

$$R_f^{\mathrm{AUC}}(h) \leq \frac{\mathrm{P}\left[\,|h(x) - c(x)| > \frac{\gamma-\eta}{2}\right]}{p}.$$

Using the Markov's inequality in Equation 5.2, we have:

$$R_f^{\mathrm{AUC}}(h) \leq \frac{\mathrm{P}\left[\,|h(x) - c(x)| > \frac{\gamma-\eta}{2}\right]}{p} < \frac{2\sqrt{\epsilon}}{(\gamma-\eta)p}.$$

Therefore, it is sufficient to choose $\epsilon = \frac{(\epsilon_{\mathrm{I}}(\gamma-\eta)p)^2}{4}$ when learning h via ERM so that $R_f^{\mathrm{AUC}}(h) < \epsilon_{\mathrm{I}}$. The sample complexity bound follows from Equation 2.11 as in Theorem 5.1. □

5.4 Discussion

The results on AUC learnability in the MI setting are surprising, because they imply the testable hypothesis that *standard supervised approaches* can be used to learn about instance and bag labels in the MI setting. The work that introduced the MI setting evaluated the performance, in terms of *accuracy*, of supervised approaches on MI problems and found them to perform poorly (Dietterich et al., 1997). Later empirical work found that supervised algorithms actually performed quite well on MI problems, in terms of *AUC* (Ray and Craven, 2005). This apparent discrepancy can be explained with the results in this chapter. Supervised approaches can perform well in terms of AUC on MI problems, but not, it seems, with respect to accuracy.

While the results in Chapter 4 do not formally show that supervised approaches cannot learn high-accuracy concepts, we conjecture that this is the case due to the one-sided noise inherent in learning to discriminate classes. As illustrated in Figure 5.1, the fact that negative instances appear some $\gamma > 0$ fraction of the time in negative bags means that learning to approximate c can be used to rank instances. However,

accurately labeling instances using an approximation of c requires choosing a *threshold* to discriminate between positive and negative instances. If the value of γ were known in advance, then such a threshold might be selected at $1 - \frac{\gamma}{2}$, for example. However, without knowledge of γ or other further assumptions about the generative process, proving that such a threshold might be selected accurately is a direction for future work.

5.5 Empirical Evaluation

Because the results in this chapter imply the surprising fact that standard supervised algorithms can be used to learn concepts with high-AUC, but not high accuracy, from MI data, we explicitly evaluate this hypothesis using real-world MI datasets. As always, there are some differences between theory and practice that might confound the experimental results. Below we first explain these two key differences and argue why they do not threaten the validity of our experimental results. Then, we discuss the remainder of our experimental methodology and results.

5.5.1 Single Instance Learning

In these experiments, we use single-instance learning (SIL) to apply supervised algorithms to MI data. The single-instance learning (SIL) procedure takes an MI dataset and applies to every instance the label of its bag. Hence, like in the generative model described in Chapter 3 used to show the theoretical results in this chapter, the SIL training set consists of bag-labeled instances. However, unlike in the generative model used in this chapter, SIL samples more than one instance per bag. As a result, SIL potentially introduces some "correlation" between instances in the training set. Figure 3.1 provies a comparison of the generative model of Chapter 3 (Figure 3.1(c)) and that of SIL (Figure 3.1(b)).

We could make SIL more closely resemble our generative model by randomly discarding all but one instance in every bag. However, this would dramatically reduce the size of most practical MI datasets and would needlessly "throw away" the information associated with the discarded instances. Instead, we use all instances in the dataset, and ignore the fact that they are potentially correlated, thereby assuming that every instance is sampled from an independent bag. The correlation could change the training distribution over instances, but this should *hurt* the performance of the supervised algorithm if it has any significant effect at all. Therefore, comparing SIL to MI-specific algorithms provides a comparison that is fair to the MI approaches.

5.5.2 Risk Minimization Approaches

The results on AUC learnability for MI data use results on learning via empirical risk minimization (ERM). ERM requires that some concept class $\mathcal{C}$ is fixed in advance, and a hypothesis $h \in \mathcal{C}$ that minimizes empirical risk (in terms of accuracy) is selected. In practice, however, $\mathcal{C}$ might not be known *a priori*. Thus, structural risk minimization (SRM) strategies are often used in practice, which simultaneously select a hypothesis h that minimizes empirical risk while controlling the capacity of the class from which h is selected. The standard support vector machine (SVM) is a structural risk minimization (SRM) approach, where the parameter C, selected via cross-validation, controls the trade-off between risk minimization and regularization. Although our theoretical result holds for ERM, we will use the SRM-based SVM for these experiments. The same SRM strategy is used across all of the baseline algorithms.

Similarly, the SVM outputs confidence values that range from $(-\infty, \infty)$ rather than from $[0, 1]$. Thus, the SVM technically does not learn a p-concept. However, prior work has shown how it is possible to fit a logistic regression model to an SVM's outputs to derive associated probabilities (Platt, 1999). However, since the resulting

rescaling of the data does not affect the relative rankings of the real-valued outputs produced by the SVM, the AUC of the classifier does not change. Accordingly, we report results using the raw confidence values produced by the SVM in these experiments.

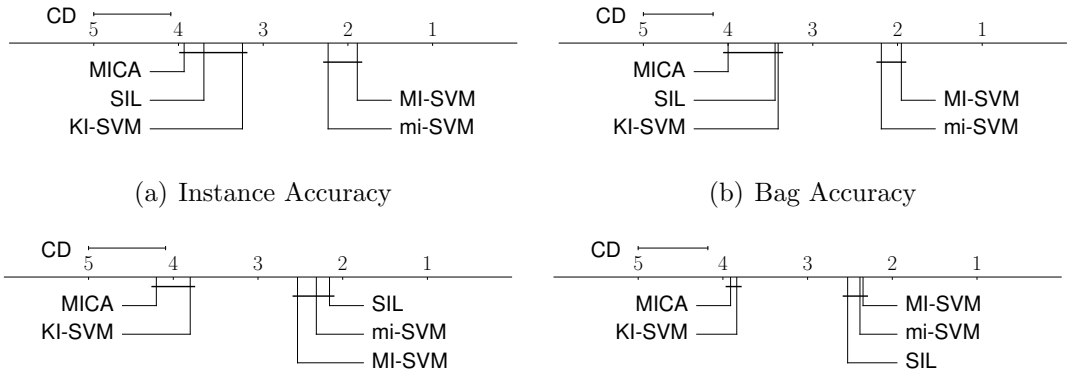
5.5.3 Methodology

To evaluate our hypothesis that a supervised SVM can perform well with respect to AUC for learning instance- and bag-labeling functions, we use a total of 55 real-world datasets across a variety of problem domains (see Table A.2). Of the 55 datasets, 45 of them have instance labels, which are only used to test the instance-level performance of classifiers, not for training.

The SIL approach combined with a standard supervised SVM is compared with four popular baseline MI SVM approaches: mi-SVM, MI-SVM (Andrews et al., 2003), MICA (Mangasarian and Wild, 2008), and the "instance" variant of KI-SVM (Li et al., 2009), which have been specifically designed to learn bag or instance labels from MI data. A brief description of these approaches can be found in Table 2.3.

We evaluate algorithms using 10-fold stratified cross-validation, with 5-fold inner-validation used to select parameters using random search (Bergstra and Bengio, 2012). Parameter selection is performed separately for each metric (accuracy and AUC) with respect to bag-level labels (since instance-level labels are unavailable at training time, even during cross-validation). We use the RBF kernel with all algorithms, with scale parameter $\gamma \in [10^{-6}, 10^{1}]$, and regularization–loss trade-off parameter $C \in [10^{-2}, 10^{5}]$. The L_2 norm is used for regularization in all algorithms.

To statistically compare the classifiers, we use the approach described by Demšar (2006). We use the nonparametric Friedman test to reject the null hypothesis that the algorithms perform equally at an $\alpha = 0.001$ significance level. Finally, we plot the average ranks using a *critical difference* diagram, which uses the Nemenyi test to



(a) Instance Accuracy

(b) Bag Accuracy

(c) Instance AUC

(d) Bag AUC

Figure 5.3: The average ranks (lower is better) of approaches on the instance- and bag-labeling tasks, evaluated using either accuracy or AUC. Statistically insignificant differences in performance are indicated with horizontal lines.

identify statistically equivalent groups of classifiers at an $\alpha = 0.05$ significance level.

5.5.4 Results and Discussion

The results are summarized using critical difference diagrams in Figure 5.3. For accuracy and AUC, the ranks of the approaches are averaged across the 45 instance-labeled datasets for the instance-level metrics and across the 55 datasets for the bag-level metrics. Lower ranks indicate better performance. A detailed description of the datasets and methodology can be found in Appendix A, with a full table of results in Appendix A.2.1.

As expected, SIL performs relatively poorly with respect to either instance- or bag-level accuracy. However, it is somewhat surprising that with respect to accuracy, SIL performs as well as some existing MI approaches. On the other hand, with respect to AUC, the relative performance of SIL increases significantly, and SIL performs as well the best MI approaches. For instance-level AUC, SIL is the highest-ranked approach. For bag-level AUC, SIL is not the best approach on average, but it is statistically equivalent to the top MI approaches. Since more samples are required to learn a bag-level concept using SIL, it could be that performance would improve even more

with a larger training sample. For the other MI algorithms, their relative rankings do not change significantly across either metric for either learning task.

These surprising results support the theoretical framework described in Chapter 3. In particular, the experimental results suggest that the assumptions made by our generative model hold in practice in many cases. For example, we claim that the assumption that negative instances appear in negative bags ($\gamma > 0$) is weak and reasonable for many MI domains. In content-based image retrieval (CBIR), negative background segments are likely to appear at least some of the time in images without the object of interest. The experimental results provide empirical support for this claim across the four domains on which we evaluate the classifiers. Of course, there might be domains for which this assumption does not hold. Determining whether any learnability results can be derived under weaker assumptions is an interesting question for future work.

There are also several ways that the theoretical and empirical results in this chapter can inform future work on MI learning. As mentioned earlier, the first work on MIL used accuracy as a performance measure, and found the SIL approach to be inaccurate in the MI setting for labeling bags (Dietterich et al., 1997). As a result, subsequent studies rarely used it as a baseline when evaluating new MI techniques. However, our results suggest that SIL should be used as a baseline when evaluating new MI approaches, especially if the intended application involves ranking bags or instances.

For researchers looking to learn high-AUC instance concepts from MI data, our results suggest that supervised approaches often suffice for this purpose in practice. Since supervised approaches are typically more computationally efficient than their MI counterparts, our theoretical and empirical justification for using supervised approaches with MI data provides a valuable practical benefit. The results in Figure 5.4 support this claim. The training time required by the algorithms for each dataset is

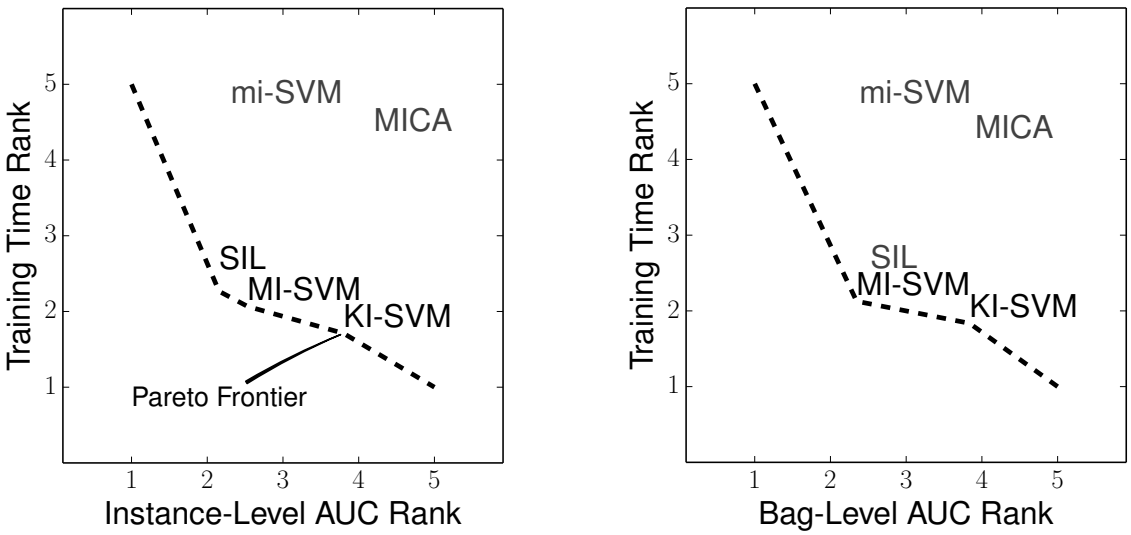


Figure 5.4: Comparison of supervised and MI-specific approaches in terms of running time and classification performance (AUC). Lower ranks correspond to better performance and faster training time. The Pareto frontier shows algorithms that are not dominated by any other algorithm along both dimensions.

ranked with 1 corresponding to the fastest algorithm, and these ranks are averaged across datasets. Then, the combined performance of each approach in terms of both AUC and training time is shown in Figure 5.4. The Pareto frontier, the set of algorithms for which there does not exist any other algorithm that has better performance along both dimensions, is indicated in the figure. SIL is at or near the Pareto frontier for both instance- and bag-labeling.

For learning high-AUC bag-labeling concepts, MI algorithms still have a slight (but statistically insignificant) advantage over SIL in terms of classifier performance and training time. However, as the next chapter shows, our generative model suggests that even better performance can be attained by applying standard supervised approaches directly to the bag-level learning task using kernels.

5.6 Summary

In this chapter, we argued that for many real-world applications of MIL, it is sufficient to *rank* instances or bags rather than assign accurate binary labels. Accordingly, we derived results about the ability to find high-AUC rankings of instances or bags from data generated by a process in MI-GEN. The surprising aspect of these results is that such rankings can be found via *standard supervised approaches*. We evaluated this surprising hypothesis empirically and found that supervised approaches *can* in fact learn to rank from MI data in practice. Thus, the empirical results support the assumptions made by MI-GEN.

Chapter 6

Learning Bag Hyperplanes from MI Data

In the previous chapters, we derived new learnability results for the MI setting that described how instance or bag concepts could be learned from MI data. For learning bag-level concepts we relied on first learning an instance-level concept f to derive an empirical bag-labeling function $\widehat{F}$ based on f in order to model the bag-labeling function F. In fact, this is a similar strategy required by prior models in which learnability is discussed (Blum and Kalai, 1998; Sabato and Tishby, 2012). On the other hand, the results in this chapter are based on the observation that for the bag-labeling task within MI-GEN, supervised approaches are sufficient to learn either accurate or high-AUC concepts from MI data.

The reason why supervised approaches are sufficient for bag-labeling in our generative model can be seen in Figure 3.1(a). In particular, since we assume that a bag-labeling function F exists that explicitly assigns labels to bags, F can be learned assuming there is some feature-vector representation for bags B_i. As it happens, there exist recent kernel-based approaches that can represent distributions, and thus bags, as single vectors in a feature space. We discuss these approaches below and

describe the relationship between these approaches and some alternatives that have been proposed in prior work. Since we are interested in a kernel-based representation for bags, we focus in this chapter on specifically learning hyperplanes from MI data rather than more general concepts as in the prior chapter.

6.1 Learning Hyperplanes from Distributions

In this section, we establish certain properties of kernels applied to distributions that make learning from distributions possible. In particular, we describe the "injectivity" and "universality" properties of distribution kernels. Injectivity ensures that distributions are represented uniquely in a feature space, and universality ensures that a kernel can be used to represent arbitrary continuous functions over distributions. Recent work has investigated these properties of distribution kernels (Muandet et al., 2012), which when used with a standard supervised support vector machine (SVM) are called support measure machines (SMMs), since examples are probability measures rather than vectors.

As described in Section 2.3.2, the normalized set kernel (NSK) with averaging normalization can be viewed as an instance of the kernel mean embedding, which maps distributions into a feature space. The NSK used with an SVM is one example of an SMM. We are interested in analyzing when such an SMM might be successfully applied to learning concepts in MI-GEN. First, in order to learn from distributions, we desire that the NSK faithfully represents the distribution it embeds into a feature space. In other words, for any two distinct distributions $\mathcal{P}$ and $\mathcal{Q}$, these distributions should be represented differently in the feature space so that $\|\mu(\mathcal{P}) - \mu(\mathcal{Q})\|_{\mathcal{H}} \neq 0$. In other words, kernel mean embedding (the feature map of the NSK) should be *injective*.

By definition, a kernel is *characteristic* whenever the resulting kernel mean em-

bedding is injective. Sriperumbudur et al. (2010) show that a kernel is characteristic whenever it is *universal*. A kernel is universal when the set of functions $\mathcal{F}$ it can represent (of the form $f(x) = \sum_{i=1}^{m} \alpha_i k(x, x_i) + b$) is a uniformly dense subset of the space $C(\mathcal{X})$ of bounded, continuous functions over the input space (Micchelli et al., 2006). In this context, the set of function representable by a kernel is typically called the reproducing kernel Hilbert space (RKHS). The popular radial basis function (RBF) kernel (Equation 2.9) is an example of a universal kernel.

Even though the NSK can provide a unique representation for distributions (and thus bags), is the class of functions over bags that an SVM learn using the NSK universal? That is, can an SVM using an NSK be used to learn arbitrarily good (in the uniform norm) approximations of continuous functions over *distributions*? These questions have been addressed by recent work on the SMM (Muandet et al., 2012).

In particular, the work by Muandet et al. (2012) describes the set of functions $\mathcal{F}$ on probability distributions that can be learned with the SMM. Let $\mathscr{P}(\mathcal{X})$ be the set of probability distributions on an input space $\mathcal{X}$ and $C(\mathcal{X})$ be the set of bounded, continuous functions on $\mathcal{X}$. Then the reproducing kernel Hilbert space (RKHS), the set of functions representable by the kernel mean embedding induced by some universal instance kernel, is dense in the set:

$$\mathcal{F} = \left\{ \mathcal{P} \mapsto \int_{\mathcal{X}} g \, \mathrm{d}\mathcal{P} : \mathcal{P} \in \mathscr{P}(\mathcal{X}),\, g \in C(\mathcal{X}) \right\}. \tag{6.1}$$

These are the functions we get by taking the expected value of a fixed but arbitrary continuous function with respect to probability distributions. However, the function class $\mathcal{F}$ is a subset of $C(\mathscr{P}(\mathcal{X}))$, the set of all bounded, continuous functions over the set of probability distributions (with respect to the weak topology on $\mathscr{P}(\mathcal{X})$). Thus, the mean embedding defined in terms of a universal kernel with respect to $C(\mathcal{X})$ is not itself universal with respect to $C(\mathscr{P}(\mathcal{X}))$.

However, as shown by Christmann and Steinwart (2010), it *is* possible to construct a universal kernel with respect to $C(\mathscr{P}(\mathcal{X}))$ using an additional level of embedding. That is, using the RBF kernel defined with respect to the mean embeddings of two distributions $\mathcal{P}$ and $\mathcal{Q}$:

$$k(\mathcal{P}, \mathcal{Q}) = e^{-\gamma \|\mu(\mathcal{P}) - \mu(\mathcal{Q})\|^2_{\mathcal{H}}}, \tag{6.2}$$

is universal with respect to $C(\mathscr{P}(\mathcal{X}))$ when μ is injective and $\mathcal{X}$ is compact. Note that the kernel in Equation 6.2 is equivalent in form to the RBF kernel given in Equation 2.9, but treats $\mathcal{H}$ rather than $\mathcal{X}$ as the input space. This iterated embedding is referred to as a *level-2* embedding (Muandet et al., 2012).

6.2 Learning Bag Hyperplanes from Distributions

Given the ability of the NSK and level-2 embedding to classify distributions, we can analyze the ability of the SMM with these kernels to learn bag concepts from MI-GEN and Generalized MIL (GMIL). When discussing kernel-based hyperplane classifiers, it is more natural to consider labels to be $\{-1, 1\}$ rather than $\{0, 1\}$ as in prior chapters. Thus, we adopt this notation for the remainder of this chapter. The first result shows that the NSK can learn bag concepts from MI-GEN given that instances are separable by a continuous function:

Proposition 6.1. *Let* $(D_{\mathcal{X}}, D_{\mathcal{B}}, f, F) \in \text{MI-GEN}(0, \pi)$ *for any* $\pi > 0$. *Further, suppose there exists* $g \in C(\mathcal{X})$, *a bounded, continuous function on* $\mathcal{X}$, *that separates instances with some margin. That is,* $f(x) = -1 \implies g(x) \leq -1$ *and* $f(x) = 1 \implies g(x) \geq 1$. *Then the NSK with a universal instance kernel can separate bags with respect to* F.

Proof. Since g is a bounded, continuous function, the sets $X_n = \{x : f(x) = -1\}$ and $X_p = \{x : f(x) = 1\}$ are disjoint, closed sets. Taking for granted that $\mathcal{X}$ is a

normal space,[1] by Urysohn's Lemma, there exists a continuous function $h : \mathcal{X} \to [0, 1]$ such that $f(x) = -1 \implies h(x) = 0$ and $f(x) = 1 \implies h(x) = 1$. Then (1) $G(B) = \frac{2}{\pi} \int_{\mathcal{X}} h \,\mathrm{d}\, \mathrm{P}(B) - 1$ separates bags and (2) G can be uniformly approximated by the NSK with a universal kernel.

Assertion (1) follows from the definition of h and MI-GEN. The function h acts like an indicator function for the set of positive instances, so $\int_{\mathcal{X}} h \,\mathrm{d}\, \mathrm{P}(B)$ quantifies the support of bag distribution B over the positive instances. According to MI-GEN, the support is 0 for negative bags, and at least π for positive bags. Thus, $G(B)$ will be at least $+1$ for positive bags, and -1 for negative bags, so G separates bags with respect to F.

Finally, assertion (2) follows from the results of Muandet et al. (2012) as described in Equation 6.1. □

It is worth noting that unlike instance concept learnability results, Proposition 6.1 holds whenever $\gamma \geq 0$, even when $\gamma = 0$. That is, if one is only interested in learning a bag concept, then the condition that negative instances appear in negative bags is no longer required. Furthermore, if the conditions of Proposition 6.1 are not met, but the bag-labeling function $F \in C(\mathscr{P}(\mathcal{X}))$, then the level-2 embedding kernel can be used to learn F.

The next proposition shows that GMIL concepts (Scott et al., 2005) can also be represented using an SMM with a universal level-2 embedding kernel. Recall that GMIL concepts (see Section 2.1.3) require that multiple types of instances be present or absent in a bag for it to be positive. For example, in classifying images of beaches, an image must contain both "sand" and "water" instances, otherwise it might be a picture of a desert or an ocean. First, we define a notion of GMIL compatible with a generative model in which bags are distributions.

[1] A topological space is normal if for any disjoint closed sets A and B in the space, there are disjoint open sets U and V containing A and B, respectively (Folland, 1999).

Definition 6.1 (Distribution-Based GMIL)**.** *Let $\{C_i\}_{i=1}^{a}$ be a set of "attractive" classes and $\{C_i\}_{i=1}^{r}$ be a set of "repulsive" classes, each a subset of $\mathcal{X}$. A bag B "hits" a class C_i if $\int_{C_i} \mathrm{d}\,\mathrm{P}(B) \geq c_i > 0$ for some class-specific threshold c_i. Likewise, a bag "misses" a class if $\int_{C_i} \mathrm{d}\,\mathrm{P}(B) = 0$. An instance of a GMIL problem $(D_{\mathcal{B}}, F)$ is such that every bag B in the support of $D_{\mathcal{B}}$ either hits or misses each class, and F assigns labels to each bag such that B is positive if and only if it "hits" some minimum number k of attractive classes and misses some minimum number s of repulsive classes.*

Given Definition 6.1, we can show the following:

Proposition 6.2. *Suppose there exists a bounded, continuous function g_i that separates every class C_i from the other classes with some margin. Then the universal level-2 kernel can arbitrarily approximate in the uniform norm a function that separates bags according to a GMIL concept F.*

Proof. Since a universal kernel can arbitrarily approximate continuous functions, it suffices to show that a GMIL concept F is separable with a continuous function.

By the same argument in the proof of Proposition 6.1, there is a function h_i for each C_i such that $x \in C_i \implies h_i(x) = 1$ and $x \notin C_i \implies h_i(x) = 0$. Thus, $H_i(B) = \max\left\{1, \frac{1}{c_i}\int_{\mathcal{X}} h \,\mathrm{d}\,\mathrm{P}(B)\right\}$ is a function that is 1 if B hits class C_i and 0 if B misses C_i. Furthermore, each H_i is a continuous function over bags since it is a composition of continuous functions.

Then, for a GMIL concept F as described in Definition 6.1, the following concept separates F:

$$G(B) = 2\min\left\{\sum_{i=1}^{a} H_i(B) - k, \sum_{i=1}^{r}(1 - H_i(B)) - s\right\} - 1.$$

The minimum is at least 1 if the number of hits and misses are both above their respective thresholds k and s. Thus, $G(B) \geq 1$ for positive bags according to F, and

$G(B) \leq -1$ for negative bags.

Finally, note that G is a composition of the continuous max function with continuous functions over bags, so it is in fact continuous and can be approximated by an element of the RKHS of the level-2 embedding kernel. □

The results above show that as distribution classifiers, the NSK and level-2 embeddings are powerful enough to represent general MI concepts. These results assume that perfect information about each bag B_i is known during training. However, in practice, only samples of instances are observed within each bag, as indicated in Figure 3.1(b). Nevertheless, as described in Section 2.3.2, empirical estimates of kernel mean embeddings converge quickly to the underlying embeddings as the sample size within each bag increases (Sriperumbudur et al., 2010).

No prior work has applied the level-2 embedding kernel to MI classification problems. However, other bag-level kernels have been defined for solving general MI classification problems. In the following section, we discuss some alternative approaches, and discuss their their relationship to the NSK and level-2 embedding kernels.

6.3 Bag Kernels as Distribution Kernels

In this section, we explore several other bag-level kernel embeddings that have been proposed for the MI setting. We describe that, in many cases, it is possible to view these kernels as naturally treating bags like distributions or samples from distributions.

EMD. The Earth-Mover's Distance (EMD), also known as the Wasserstein metric, is a popular distance metric commonly used within the content-based image retrieval (CBIR) domain (Rubner et al., 2000). The EMD is a proper distance metric between distributions, and its name come from an intuitive description of how it operates. If one views one distribution as a pile of dirt, and the other distribution as a hole in

the ground, then the EMD is a measure of the minimum amount of work, in terms of mass of dirt times Euclidean distance across the ground traveled, that it takes to fill the hole with the pile.

Given any distance metric, such as the EMD, a kernel can be constructed using the generalized RBF kernel (Schölkopf and Smola, 2002):

$$k_d(x_i, x_j) = e^{-\gamma d(x_i, x_j)^p}, \tag{6.3}$$

which is a generalization of Equation 2.9, for which $d(x_i, x_j) = \|x_i - x_j\|_2$ and $p = 2$. Thus, a kernel between bags B_i and B_j using the EMD with $p = 1$ is defined as follows:

$$k_{\text{EMD}}(B_i, B_j) = e^{-\gamma \,\text{EMD}(B_i, B_j)}. \tag{6.4}$$

The performance of the EMD kernel for MI classification has been explored in prior work (Amores, 2013).

The EMD kernel in Equation 6.4 and the level-2 embedding kernel in Equation 6.2 are actually members of the same family of RBF kernels given by Equation 6.3. In fact, the distance metric corresponding the level-2 kernel in Equation 6.2, $\|\mu(\mathcal{P}) - \mu(\mathcal{Q})\|_{\mathcal{H}}$, is known as the Maximum Mean Discrepancy (MMD), and has been studied in prior work (Gretton et al., 2007). Therefore, the empirical level-2 embedding kernel between bags can be rewritten in the form of Equation 6.3 as:

$$k_{\text{MMD}}(B_i, B_j) = e^{-\gamma \,\text{MMD}(B_i, B_j)^2}. \tag{6.5}$$

Given the similar form of Equation 6.4 and Equation 6.5, it is easy to see the connection between the level-2 embedding kernel and the EMD kernel approaches. Both are based on distance metrics defined on the space of distributions (or samples from distributions). In fact, both distance metrics induce the same weak topology on

the space of distributions over instances, $\mathscr{P}(\mathcal{X})$, when the instance space is separable (Sriperumbudur et al., 2010). As discussed above, the level-2 embedding kernel is universal in its ability to represent continuous functions over distributions. Although it intuitively seems that the EMD kernel should have similar representational abilities, it is still an open question as to whether the EMD kernel is similarly universal over the space of distributions.

Box-Counting Kernel. The box-counting kernel was specifically designed to represent GMIL concepts (Tao et al., 2004, 2008). The box-counting kernel is motivated by the assumption that "attractive" and "repulsive" classes of points as described in Definition 6.1 are contained within axis-parallel boxes in the feature space. Note that this is a stronger assumption than is made in Proposition 6.2, which allows these classes to be arbitrary closed sets. The box-counting kernel then constructs a Boolean feature corresponding to *every* axis-parallel box in a discretized version of the feature space. A box is represented with such a feature vector with a "1" for every feature with the corresponding box containing some point in the bag, and "0" for every feature whose corresponding box does not contain any points from the bag.

Because it is intractable to explicitly enumerate all such features, Tao et al. (2004) use a kernel by observing that the inner product between two Boolean feature vectors as described above will be equal to the number of boxes that contain points from both bags involved in the inner product. Hence, the kernel is called the "box-counting" kernel. However, the box-counting problem is #P-complete, so an approximation is used to make even the kernel computation tractable (Tao et al., 2004). The approximation scheme finds a value within a factor of ϵ of the true count with probability $1-\delta$, in $\mathsf{poly}(m_u, k, \frac{1}{\epsilon}, \frac{1}{\delta})$ time, where m is the bag size and k is the dimensionality of the input feature space (see Table 6.1 for a comparison of computation complexity).

MILES and YARDS. Another set of approaches for standard MI classification construct a representation for bags by using an RBF kernel as a similarly measure

between bags in a dataset and instances in a dataset. First, each instance in a dataset is represented using a feature vector of length $|X|$, with each feature an RBF kernel between that instance and one of the $x_i \in X$. This representation is similar to the "empirical" kernel representation used by the box-counting kernel. Then, some function combines the individual feature representations to construct a bag-level feature representation.

The Multiple-Instance Learning via Embedded Instance Selection (MILES) algorithm (Chen et al., 2006) combines instance representations using the max function, whereas the Yet Another Radial Distance-based Similarity measure (YARDS) approach (Foulds, 2008) uses the average instance representation to construct a bag-level feature vector. The original MILES paper used a linear SVM with L_1 regularization to learn a "sparse" classifier that used fewer of the $|X|$ features. However, we find that using the standard L_2 regularization with an RBF kernel outperforms the originally proposed formulation.

By averaging over instance feature representations, YARDS is using a mean embedding to represent each bag. However, YARDS uses an explicit feature embedding by enumerating the values of the kernel between each instance and the instances in X. Thus, YARDS can be viewed as a kernel mean embedding that uses an "empirical" version of the kernel feature map, which is equivalent to the implicit kernel feature map up to a linear rescaling of the features (Schölkopf and Smola, 2002).

Under the view of bag feature embedding of YARDS as described above, when the YARDS features are used with an RBF kernel, the RBF kernel performs a secondary embedding of bags analogous to the universal level-2 kernel. Thus, when YARDS is used with a standard SVM and an RBF kernel, it can learn the same concepts as the level-2 kernel used with the SMM. However, some practical differences between these approaches might be observed due to the linear rescaling of features, as we show in Section 6.4.

and . Several kernels have been designed for the MI setting that represent bags as graphs of related instances (Zhou et al., 2009). The MIGraph and miGraph approaches first construct graphs for each bag by connecting two instances in a bag with an edge if they are within a distance of τ of each other. As a heuristic, τ is chosen as the average distance between instances in a bag. The corresponding edge is weighted with a normalized reciprocal of the distance between the instances.

Like the NSK, the MIGraph kernel is a sum of pairwise kernel values between instances *and edges* across two bags B_i, B_j and their corresponding graphs $G(B_i)$, $G(B_j)$:

$$k_{\text{MIGraph}}(B_i, B_j) = \sum_{x_i \in B_i} \sum_{x_j \in B_j} k_{\text{I}}(x_i, x_j) + \sum_{e_i \in G(B_i)} \sum_{e_j \in G(B_j)} k_{\text{edge}}(e_i, e_j).$$

A kernel on edges is defined by representing each edge as a set of features. If an edge e connects instances x_i and x_j, then e is represented using four features that include the degree of x_i, the degree of x_j, the weight of the edge divided by the sum of the edge weights originating from x_i, and the weight of the edge divided by the sum of the edge weights originating from x_j. Then, a standard kernel such as the RBF kernel between these feature vectors can be used as the edge kernel.

The miGraph kernel is a computationally more efficient version of MIGraph that is equivalent to a weighted version of the NSK. In particular, k_{miGraph} can be written as:

$$k_{\text{miGraph}}(B_i, B_j) = \frac{1}{\sum_i w_i} \frac{1}{\sum_j w_j} \sum_{x_i \in B_i} \sum_{x_j \in B_j} w_i w_j k_{\text{I}}(x_i, x_j).$$

The weight w_i for each instance $x_i \in B$ is computed as follows:

$$w_i = \left(\sum_{x_j \in B} \mathbb{1} \left[(x_i, x_j) \in G(B) \right] \right)^{-1}.$$

That is, w_i is the reciprocal of the number of instances adjacent to x_i in the graph

$G(B)$ of bag B. $G(B)$ contains an edge for every pair of instances whose distance is less than some threshold τ and includes self-edges.

The computation of miGraph therefore gives more weight to instances that lie in less densely populated regions of the feature space within each bag. Under the view of bags as distributions, miGraph can be viewed as performing the mean embedding on a weighted sample, or a sample drawn from a modified version of the bag's underlying distribution. Intuitively, the modification of the bag's distribution increases the probability mass over regions with low mass and decreases the mass over regions with originally high mass. As a result, the modified bag distribution is more uniform over the support of the original distribution. Such a reweighting might provide an advantage when the support rather than distribution of instances within a bag is a more discriminating feature of bags. As observed in prior work (Zhou et al., 2009), when τ is very large and $G(B_i)$ is fully connected or τ is very small so that $G(B_i)$ contains only self-edges, k_{miGraph} is equivalent to the NSK because all instance weights are identical. In these limiting cases, the connection between miGraph and the NSK becomes apparent.

6.4 Empirical Evaluation

In this section, we compare the bag-level classification performance of select bag kernels discussed above. Furthermore, we discuss several practical details of the approaches, which show that distribution-based methods tend to also be more efficient than those derived using other heuristics.

We hypothesize that because bags are best understood as samples from distributions, kernels defined to act on distributions should outperform approaches that do not explicitly make this assumption. Furthermore, the theoretical results in Section 6.1 suggest that the level-2 embedding kernel provides a more powerful representation

than the NSK for learning from distributions (Muandet et al., 2012). The results presented below are the first application of the level-2 embedding to MI classification problems. Apart from the level-2 embedding, we expect other approaches that treat bags as distributions to perform well if the proposed model captures the underlying generative process of the data.

6.4.1 Methodology

Because bag-labeling approaches are generally more computationally efficient than instance-labeling approaches, we are able to use all available datasets from various problem domains. Only bag-level labels are used for training and evaluation of algorithms for these datasets.

We use 10-fold stratified cross-validation to evaluate algorithm performance in terms of accuracy and AUC, with 5-fold inner cross-validation and random parameter search (Bergstra and Bengio, 2012) used to select parameters. For the box-counting kernel, we use the default parameters from the original work (Tao et al., 2008): $\epsilon = 0.1$, $\delta = 0.01$, and the 50^{th} root with the normalized empirical kernel. The box-counting kernel also requires feature values to be integral, so we multiply original feature values by 10^2–10^3 and truncate to produce features with roughly the same numbers of significant digits as used in the original experiments. We search for the C parameter of the SVM within the range $[10^{-3}, 10^9]$. Of MIGraph and miGraph, we present results from the latter, which achieved better performance. We use the Euclidean distance to construct the neighborhood graph, with the average distance between all training instances used as the threshold τ. Of MILES and YARDS, we again select the latter approach as it had better performance. For the approaches other than the box-counting kernel, the range $[10^{-3}, 10^9]$ is used to select C. The EMD and NSK with averaging normalization only have a single RBF γ parameter selected from the range $[10^{-6}, 10^1]$. The miGraph kernel also has a γ parameter

(a) Accuracy (b) AUC

Figure 6.1: Ranks (lower is better) of the various bag kernel approaches on the bag-labeling task. The critical difference diagrams show the average rank of each technique across datasets, with techniques being statistically different at an $\alpha = 0.05$ significance level if the ranks differ by more than the critical difference (CD) indicated above the axis. Thick horizontal lines indicate statistically indistinguishable groups (i.e. a technique is statistically different from any technique to which it is *not* connected with a horizontal line).

selected from the same range, as well as a τ parameter that is chosen heuristically per bag as the average distance between instances in the bag. The YARDS and level-2 embedding approaches have an additional γ parameter to perform the embeddings, and this additional parameter is selected over the range $[10^{-6}, 10^{1}]$ for YARDS and $[10^{-4}, 10^{3}]$ for the level-2 kernel.

We use the approach described by Demšar (2006) to statistically compare the kernel approaches. We use the Friedman test to reject the null hypothesis that the algorithms perform equally at an $\alpha = 0.001$ significance level, and an $\alpha = 0.05$ significance level for the Nemenyi test and resulting critical difference diagram shown in Figure 6.1. A detailed description of the datasets and methodology can be found in Appendix A, with a full table of results in Appendix A.2.2.

6.4.2 Results and Discussion

With respect to either accuracy or AUC, the results in Figure 6.1 are generally consistent with the theoretical discussion presented in Section 6.3. The NSK and miGraph approaches produce very similar representations of the data, and also perform very similarly with no significant difference across datasets. Likewise, the level-2 embedding and EMD kernels do not significantly differ, which enforces the conjecture that

they have equivalent representational abilities. According to the theoretical discussion in Section 6.3, YARDS is an empirical version of the level-2 embedding, which is reflected by the statistically equivalent performance of these approaches.

Although the level-2 embedding offers greater representational power than the NSK, the performance of these approaches is also statistically equivalent. This likely reflects the fact that many of the problems studied respect the standard MI assumption, and do not require generalized MI concepts to separate bags. The box-counting kernel is also designed to represent generalized MI concepts, but it is an outlier that performs significantly worse that the other approaches. On the other hand, for the TRX protein dataset for which the box-counting kernel was specifically proposed, the box-counting kernel does outperform the NSK, which might be unable to represent the more general underlying MI concept. Nevertheless, the level-2 and EMD kernels perform at least as well as the box-counting kernel on this dataset. The performance of the box-counting kernel is especially low for the 20 high-dimensional Newsgroups datasets. As the number of potential boxes grows exponentially with the number of dimensions, the box-counting kernel appears to over-fit to the training data very quickly as the dimensionality of the instance space increases.

6.4.3 Practical Considerations

One significant advantage of directly learning bag-level concepts using bag kernels is computational efficiency. After the bag-level kernel matrix is computed, the subsequent SVM optimization procedure must solve a problem of size proportional to the number of bags rather than the number of instances. An empirical comparison of bag- and instance-level classifiers is discussed further in Chapter 7.

However, within the set of bag kernel classifiers, each approach requires differing amounts of time to construct the kernel matrix and train a classifier. The trade-off between training time and accuracy can be informative for selecting a bag kernel to

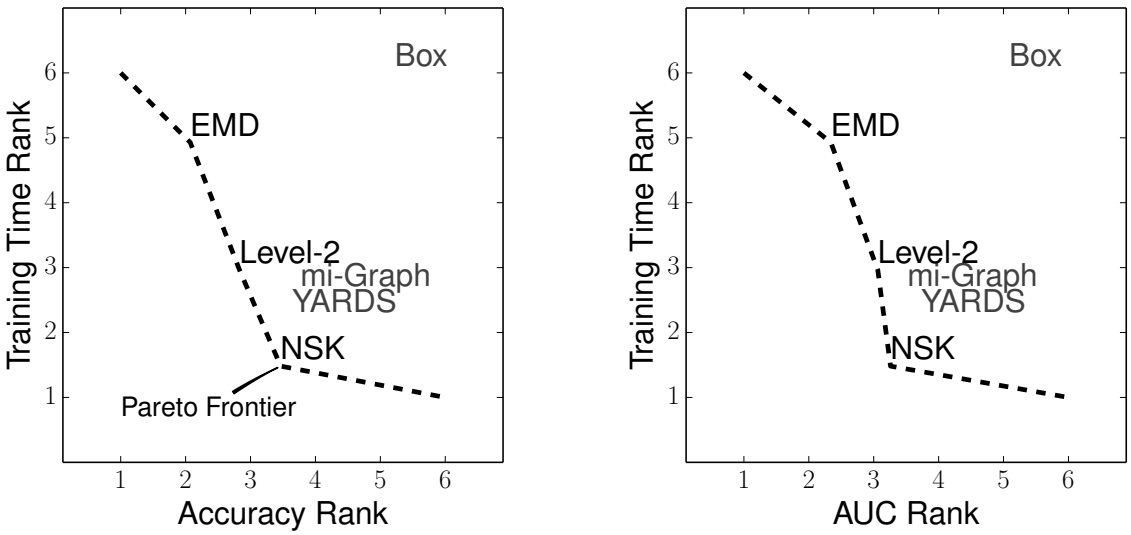


Figure 6.2: Comparison of bag-level kernel classifiers in terms of running time and classification performance (accuracy and AUC). Lower ranks correspond to better performance and faster training time. The Pareto frontier shows algorithms that are not dominated by any other algorithm along both dimensions.

apply in practice. Hence, Figure 6.2 shows a comparison of both the training time required by the bag-level kernel methods and their performance in terms of either accuracy or AUC. As for the other performance metrics, running time ranks are computed by ranking the algorithms on each dataset, with 1 corresponding to the fastest running time, and averaging the resulting ranks across datasets. Figure 6.2 shows the Pareto frontier, which is the set of approaches that are not outperformed by any other approach along both dimensions.

Below is a discussion of the observed running time and other practical considerations for the various bag kernels. A summary of the computation complexity for each approach is given in Table 6.1.

NSK and level-2. The NSK and level-2 kernel embeddings have the same computational complexity, despite the level-2 kernel constructing a more powerful representation of bags. Of course, in practice, the level-2 kernel requires a constant factor more mathematical operations than the NSK, so it takes more time in practice. As a result, the NSK is the most efficient of the approaches explored in the experiments,

but the level-2 embedding achieves better performance on average.

EMD. The computation of the EMD requires solving an optimization program. In the case of two samples X_1 and X_2, the EMD can be written formally as the solution to the following problem:

$$\begin{aligned} \text{EMD}(X_i, X_j) = \min_{0 \leq w_{ij}} & \sum_{x_i \in X_i} \sum_{x_j \in X_j} w_{ij} \left\| x_i - x_j \right\|_2, \\ \text{s.t.} \quad & \forall i : \sum_j w_{ij} = 1 \\ & \forall j : \sum_i w_{ij} = 1. \end{aligned} \tag{6.6}$$

In Equation 6.6, w_{ij} represents the amount of "dirt" that is moved from location x_i to location x_j, and the constraints enforce that dirt, or probability mass, is conserved. As such, the optimization in Equation 6.6 is an instance of the well-studied transportation problem, which has a solution computable in cubic time in terms of the set sizes (Edmonds and Karp, 1972). This makes the computation of the EMD relatively efficient, but less so than that of the NSK or level-2 embedding.

Box-Counting Kernel. The box-counting kernel suffers from several practical disadvantages. First, because its entries are computed using an approximation, the resulting matrix might not be a positive-definite kernel (Tao et al., 2004). A related issue is that the approximation presented in Tao et al. (2004) only bounds error within a factor of ϵ with probability $1 - \delta$. With probability δ, the approximation error is potentially unbounded, so the *expected* error of the kernel is also potentially unbounded. Therefore, it is not immediately clear that there exist PAC-style bounds on the ability of the box-counting kernel to learn GMIL concepts, which would require the error of the classifier, and hence the quality of the kernel approximation, to be bounded in expectation. Finally, note that the $1-\delta$ probability of an ϵ-approximation applies to each of the $O\left(|B|^2\right)$ kernel entry separately. Thus, the probability that *all*

entries in the kernel are ϵ-approximated is significantly lower than $1 - \delta$.

Furthermore, because there are so many possible boxes in the discretized space, the box counts along the diagonal of the kernel matrix ($k(B_i, B_i)$ is the number of boxes containing *any* points from B_i) tend to be extremely large relative to off-diagonal entries. Often, all entries in the matrix are in magnitude beyond what is representable using standard double-precision floating point numbers. To deal with numerical and diagonal-dominance issues, an "empirical" version of the kernel is used in which each row of the kernel matrix is viewed as a feature vector. The entries of this empirical kernel matrix are then transformed by taking the 50^{th} root to avoid numerical overflow and reduce the dominance of the diagonal features. Later versions of the kernel also deal with the diagonal-dominance issue by introducing normalization (Tao et al., 2008), in which each kernel entry is normalized by the number of boxes containing points from one bag or the other. However, the empirical kernel transformation is still required to obtain reasonable results. In our experiments, we use this empirical version of the box kernel and normalization heuristics to avoid numerical issues. Nevertheless, Figure 6.2 shows that the box-counting kernel is quite inefficient in comparison to other kernels.

MILES and YARDS. Due to the similarity between MILES and YARDS, we only evaluate YARDS in our experiments, since it achieves better performance than MILES does. Because YARDS explicitly enumerates $|X|$ features before computing the kernel, the computation complexity of the approach relative to that of the level-2 kernel increases ($|X| \gg |B_i|$). However, in practice, we observe that YARDS is slightly faster on average than the level-2 embedding, perhaps due to additional constant factors in the level-2 computational complexity. On the other hand the level-2 embedding achieves somewhat better classification performance on average than YARDS does.

and . A significant disadvantage of the MIGraph kernel is its computation

Table 6.1: Complexity of computing bag-level kernel entries, where m denotes bag size, $|X|$ is the number of instances in the dataset, ϵ is an approximation factor for the box-counting kernel, and $1 - \delta$ is the probability of ϵ-approximation.

Technique	**Complexity**
NSK	$O(m^2)$
Level-2 Embedding	$O(m^2)$
	$O(m^2)$
MILES/YARDS	$O(m\|X\|)$
EMD	$O(m^3)$
	$O(m^4)$
Box Counting	$O\left(m^2 \frac{1}{\epsilon^2} \log \frac{1}{\delta}\right)$

complexity. Since the number of edges is a bag graph grows roughly as the square of the bag size, computing all pairwise edge kernel values is quartic in terms of the bag size. For bags with on the order of 100 instances, as is typical in some applications, roughly 10^8 edge kernel values must be computed *per entry* of the MIGraph kernel. The authors of the work on MIGraph acknowledge this disadvantage in the approach, and therefore suggest using miGraph as a more efficient graph-based kernel (Zhou et al., 2009). The miGraph kernel is also more accurate than MIGraph. Accordingly, we only evaluate miGraph in our experiments.

The best-performing bag-level approaches studied in this chapter are the EMD and level-2 embedding kernels. Each of these approaches is also based on an RBF kernel derived from a distance metric between probability distributions. Thus, the results in Figure 6.1 support our hypothesis that bags should be viewed as distributions over instances. Furthermore, the results justify the novel application of the level-2 embedding to the MI classification setting. On the other hand, the box kernel, though offering similar representational abilities in theory, performs poorly in practice.

6.5 Summary

In this chapter, we described how distribution kernels can be used to directly learn bag-labeling functions with a standard SVM formulation. First, we provided theoretical guarantees about applying distribution kernels to bag-labeling concepts in MI-GEN. Then, we showed how some existing bag kernels can be interpreted as distribution kernels. With our experiments, we showed that distribution kernels generally outperform other bag kernels, even for GMIL tasks. Our results suggest that instance of learning bag concepts through a combination of instance labels, the bag concept can be more effectively learned in practice by directly applying standard supervised approaches.

Chapter 7

On the Difficulty of Learning Instance Hyperplanes from MI Data

We showed in Chapter 4 and Chapter 5 that we can learning bag concepts through instance concepts. Similarly, prior learnability results (Sabato and Tishby, 2012) show that it is possible to learn bag-level concepts using an instance-level hypothesis class. However, such approaches only work if an algorithm correctly measures the empirical risk, or training error. We show that there is a fundamental trade-off for algorithms that attempt to learn MI bag concepts using this strategy.[1] In particular, if we want to construct a convex optimization program to find instance-level hyperplanes that label bags, then such a program will inevitably fail to correctly measure the error on the training set. So in the case of instance hyperplanes, it is not possible to use efficient convex optimization procedures to find a good bag classifier in general.

[1] An analysis of the trade-offs required by MI support vector machine (SVM) classifiers appears in Doran and Ray (2013b).

7.1 Learning Instance Hyperplanes

In this section, we first describe three desirable properties of the standard supervised SVM approach. Then, we supplement known hardness results for learning hyperplanes from MI data by showing that no MI SVM formulation can have all three of these properties. While prior results describe the computational complexity of finding hyperplanes that consistently classify a dataset (Sabato and Tishby, 2012), we show that MI SVM formulations that perform bag-level structural risk minimization (SRM) using instance-level hyperplanes are inherently nonconvex, and therefore inefficient to solve in general, with respect to any loss function that appropriately measures risk. This suggests that bag-level hyperplanes should be learned directly using supervised approaches, as discussed in Chapter 6.

As described in Section 2.2.1, the standard supervised SVM selects a hyperplane of the form $\langle w, x \rangle + b$ to classify examples. As in Chapter 6, we adopt the standard convention in this chapter that labels of bags and instances are taken from $\{-1, 1\}$ rather than $\{0, 1\}$. The optimization program in Equation 2.2 has a space of *solutions* (the feasible region) corresponding to the space of *classifying hyperplanes*. The relationship that the SVM enforces between these two spaces has several intuitive and desirable properties.

First, consider the SVM formulation of Equation 2.2, duplicated below for convenience:

$$\min_{w,b,\boldsymbol{\xi}} \overbrace{\tfrac{1}{2}\|w\|^2}^{\text{Regularization}} + \overbrace{C\sum_i \xi_i}^{\text{Loss}}, \quad \text{s.t.} \quad y_i\left(\langle w, x_i \rangle + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0. \tag{7.1}$$

Recall that the first term in the objective function selects the hyperplane with the largest margin to perform *regularization* of the hypothesis space, which provably leads to better generalization on new examples. The second term in the objective function measures the *loss* of the solution, which is a measure of how much the solution

misclassifies the data in the training set.

A hyperplane is *consistent* with a training set when it correctly classifies each example in the dataset according to the rule $f(x_i) = \text{sign}\left(\langle w, x_i \rangle + b\right) = y_i$. Now, consider all solutions of Equation 2.2 for which the loss term is precisely zero. Given the constraints in the optimization program, all such solutions correspond to consistent hyperplanes. Thus, we say that the formulation is "sound." Conversely, for any consistent hyperplane, there exists a solution in the optimization program in which the slack variables are zero and the loss term is also zero. Accordingly, we say that the formulation is "complete." Therefore, there is a natural correspondence between consistent hyperplanes and zero-loss solutions of the SVM formulation. Finally, the SVM optimization program is *convex*, which means that it can be solved efficiently (with a number of iterations polynomial in the size of the training set).

These three properties, soundness, completeness, and convexity, have made the SVM a popular approach to classification in the supervised setting. In particular, soundness and completeness are required so that the loss of a solution is evaluated properly and the selected hyperplane generalizes to new examples. The desirability of these properties is also present in the MI setting. However, as demonstrated in Section 7.1.2, many popular MI SVM algorithms lack one or more of these properties. Furthermore, we prove that a trade-off between soundness, completeness, and convexity is *fundamental* in the MI setting for hyperplane classifiers. Below, we formally describe this result and empirically explore the practical effects of the trade-offs made by popular MI SVM algorithms. Finally, we contrast the hardness of directly learning MI hyperplanes with the positive learnability results derived under the new generative model.

7.1.1 Consistency, Soundness, and Completeness

As discussed intuitively above, we would like for MI SVMs to possess soundness and completeness properties with respect to the relationship between the set of consistent hyperplanes and the set of zero-loss solutions. To formalize these notions for the MI setting, first let $\mathcal{H}$ be a space of classifying hyperplanes (w, b) defined over the instance space $\mathcal{X}$. For an MI dataset (B, Y), a consistent hyperplane correctly labels all bags; that is, it labels at least one instance in each positive bag as positive, and all instances in negative bags as negative:

Definition 7.1 (Consistency)**.** *A classifying hyperplane* (w, b) *is* consistent *with* (B, Y) *if for each* $B_i \in B$*:*

$$\begin{cases} \exists x_{ij} \in B_i : \langle w, x_{ij} \rangle + b \geq 1 & \text{if } Y_i > 0 \\ \forall x_{ij} \in B_i : \langle w, x_{ij} \rangle + b \leq -1 & \text{if } Y_i < 0 \end{cases}$$

A consistent hyperplane separates bags, but not necessarily instances (e.g. an instance in a positive bag can lie within the margin as long as some other instance is classified as positive). Let $\mathcal{C} \subseteq \mathcal{H}$ denote the set of all consistent hyperplanes. Some algorithms are formulated with respect to a stronger notion of consistency in which each instance must lie outside the margin of the classifier:

Definition 7.2 (Strong Consistency)**.** *A classifying hyperplane is* strongly consistent *with* (B, Y) *if it is consistent and for each instance* x_{ij}*, either* $\langle w, x_{ij} \rangle + b \geq 1$ *or* $\langle w, x_{ij} \rangle + b \leq -1$*.*

The two definitions of consistency lead to equivalent hardness results. Below, "consistency" refers to the weaker form of unless otherwise noted.

In the standard supervised SVM shown in Equation 7.1, a combination of loss minimization and regularization known as SRM is used to select a classifying hyperplane. MI SVMs can also be written using the same abstract formulation. That is,

learning hyperplanes for MI data via SRM entails solving an optimization problem of the form:

$$\min_{s \in \mathcal{F}} \lambda \Omega(s) + \ell(s), \tag{7.2}$$

with a solution space $\mathcal{S}$, a feasible region $\mathcal{F} \subseteq \mathcal{S}$, a regularizer Ω, a nonnegative loss function ℓ, and a trade-off parameter λ.[2] An "MI optimization program" will refer to any program of the form in Equation 7.2 for which there exists a continuous function $\mu : \mathcal{S} \to \mathcal{H}$ that maps solutions in $\mathcal{S}$ to classifying hyperplanes in $\mathcal{H}$.

Most MI SVMs used in practice are encompassed by this general formulation. Typically, the variables (w, b) are included directly in the optimization program, in which case μ simply projects a solution $s \in \mathcal{S}$ onto the dimensions corresponding to (w, b). The definitions above can also be extended to kernelized versions of the SVM formulation (see Equation 2.3). When a kernel function $k(\cdot, \cdot)$ is used, the space of hyperplane classifiers $\mathcal{H}$ is the kernel feature space, and $s \in \mathcal{S}$ contains variables α_j such that a hyperplane $f \in \mathcal{H}$ is represented via $f(x_i) = \sum_j \alpha_j k(x_j, x_i) + b$.

We can now define two desired properties of a loss function ℓ used in an MI optimization program. For any MI dataset, if a hyperplane (w, b) is consistent, then there should exist a corresponding solution s ($\mu(s) = (w, b)$) with zero loss, since the hyperplane properly "separates" bags in the sense of Definition 7.1. Conversely, every zero-loss solution s should correspond (again, via μ) to a consistent hyperplane (w, b); otherwise, the hyperplane misclassifies bags without penalty. We call these two properties "completeness" and "soundness."

Definition 7.3 (Soundness)**.** *An MI optimization program is* sound *if for any dataset, all feasible, zero-loss solutions correspond to consistent hyperplanes; that is, if* $\mu(\mathcal{Z}) \subseteq \mathcal{C}$.

[2]For notational convenience, the trade-off parameter is a coefficient of the regularization term in Equation 7.2, whereas it is a coefficient of the loss term in Equation 7.1. The two formulations are equivalent with $\lambda = \frac{1}{C}$.

Definition 7.4 (Completeness)**.** *An MI optimization program is* complete *if for any dataset, there exists a feasible, zero-loss solution corresponding to every consistent hyperplane; that is, if* $\mathcal{C} \subseteq \mu(\mathcal{Z})$.

Therefore, sound and complete MI optimization programs have the property that $\mu(\mathcal{Z}) = \mathcal{C}$, or that the set of feasible, zero-loss solutions corresponds to the set of consistent hyperplanes. Note that these properties are quantified over all datasets, so although they focus on the datasets for which consistent hyperplanes exist, they apply to all MI optimization programs of the form in Equation 7.2. That is, these properties naturally apply to algorithms that handle nonseparable data with regularization. Though these properties in some sense "ignore" the behavior of such algorithms on nonseparable datasets, any algorithm that correctly measures empirical risk must *at least* do so correctly on separable datasets.

Finally, as for supervised learning, we desire the MI optimization program to be *convex* so that an optimal solution can be found efficiently. We define this property formally below:

Definition 7.5 (Convexity)**.** *An optimization program is* convex *if for any dataset, and any* $\lambda \geq 0$, $\mathcal{F}$ *is a convex set, and* $\lambda\Omega(s) + \ell(s)$ *is a convex function. With* $\lambda = 0$, *this implies that* $\ell(s)$ *is also a convex function.*

7.1.2 Properties of Instance Hyperplane Classifiers

In the past decade, there have been numerous extensions of SVMs to the MI setting, as described in Table 2.3. As described above, the ideal MI SVM approach is sound, complete, and convex. However, each of the algorithms listed in Table 2.3 lacks one or more of these three properties. Figure 7.1 summarizes the properties possessed by each algorithm, with a detailed analysis found in prior work (Doran and Ray, 2013b). Some examples are provided below to show how these various approaches

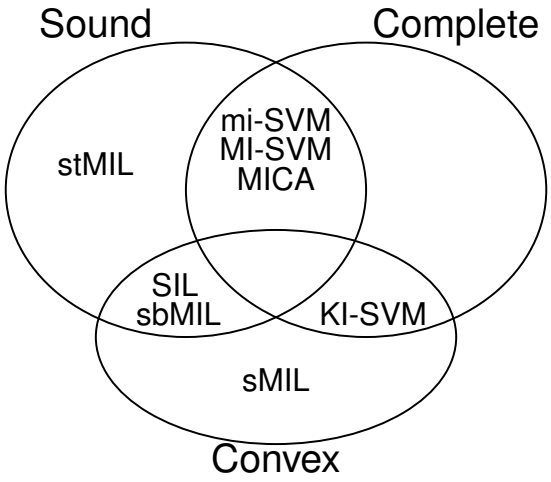


Figure 7.1: Soundness, completeness, and convexity of various algorithms.

lack soundness or completeness. Then, Section 7.1.3 shows that *no* MI optimization program can possess all three properties.

SIL (Sound, Convex). As a simple example, consider the single-instance learning (SIL) approach described in Section 5.5.1. SIL assigns each instance the label of its bag, creating a supervised learning problem but mislabeling negative instances in positive bags. SIL is sound, since each zero-loss solution labels at least one instance (in fact, all instances) in a positive bag positive, corresponding to a consistent MI hyperplane. However, because there are clearly consistent MI hyperplanes that do not require *all* instances in positive bags to be positively classified. SIL is not complete because it does not allow these solutions without loss. On the other hand, SIL is convex because it uses the standard convex SVM formulation (see Equation 2.2).

In light of the discussion in Section 5.4, the incompleteness of SIL suggests that it might not learn *accurate* bag concepts, which is confirmed by experimental observation in Figure 5.3. Section 7.1.4 continues the discussion about how the soundness and completeness properties are related to the learnability of MI concepts.

(Sound, Complete). The MI-SVM approach (see Equation 2.4) is similar to the standard SVM formulation, but uses a maximum over instance labels to compute loss via bag-level slack variables (Andrews et al., 2003). For any consistent hyperplane,

there exists a positive instance in each positive bag with a label at least 1, so the maximum label is at least 1 and the bag-level slack variable can be set to zero in some solution corresponding to that hyperplane. Similarly, for each negative bag, all instance labels are at most -1, so the maximum is at most -1 and the bag-level slack variables can set to zero for negative bags. Since there is a zero-loss solution for every consistent hyperplane, MI-SVM is complete.

On the other hand, suppose s is a zero-loss solution. Then the maximum label of instances in each positive bag is at least 1, so some instance in each positive bag is labeled positive. Furthermore, the maximum label of instances in each negative bag is -1, so all instances in negative bags must have negative labels. Thus, $\mu(s)$ must be consistent, so MI-SVM is sound. However, as discussed in Section 2.2.3, MI-SVM is not convex because of the maximum function in the constraints.

(Sound, Complete under strong consistency). The mi-SVM formulation (see Equation 2.5) uses standard SVM constraints while leaving the y_i variables unknown over $\{-1,\ +1\}$ for instances in positive bags (Andrews et al., 2003). Optimizing over binary labels makes the program nonconvex. An additional constraint $\sum_j \frac{y_{ij}+1}{2} \geq 1$ for positive bags ensures that at least one instance label in each positive bag is positive and guarantees soundness.

Some MI SVM approaches, including mi-SVM, make stronger assumptions about what it means for a hyperplane to be "consistent" with an MI dataset (see Definition 7.2). In particular, strong consistency also assumes that each instance has some $\{-1, +1\}$ label. Therefore, the set of strongly consistent hyperplanes $\mathcal{C}'$ is a subset of consistent hyperplanes $\mathcal{C}$. Soundness and completeness can also be defined with respect to $\mathcal{C}'$ rather than $\mathcal{C}$. This makes "strong soundness" a stronger condition than soundness, and "strong completeness" a *weaker* condition than completeness. Thus, a "complete" algorithm using the strong consistency assumption might not be complete in the sense of Definition 7.4. However, under the generative assumption that

each instance has a label, weakening the condition for completeness in this way does not affect the behavior of the algorithm with respect to SRM (the target classifying hyperplane assigns a $\{-1, +1\}$ label to each instance). Accordingly, such algorithms are considered complete with this caveat. Because a proper choice of each y_i allows a zero-loss solution for any (strongly) consistent hyperplane, mi-SVM is complete.

KI-SVM (Complete, Convex). The key instance SVM (KI-SVM) algorithm is an example of an MI optimization program that is not sound. KI-SVM uses multiple kernel learning, in which a convex combination of various kernel functions is also learned during the optimization of the SVM (Lanckriet et al., 2004). The set of kernels used corresponds to all possible ways of selecting a witness (or "key") instance from each positive bag. If there are n positive bags each with m instances, the optimization program searches over convex combinations of m^n kernels, each representing one possible consistent instance labeling. Therefore, even though this formulation is convex, the number of variables is exponential in the problem size. In practice, the cutting plane algorithm (Kelley, 1960) is used to avoid enumerating these variables. This approach is complete, since for any hyperplane corresponding to a consistent labeling, selecting (via the convex combination) the kernel corresponding to that labeling makes the solution corresponding to that hyperplane feasible with zero loss. On the other hand, there exist examples for which KI-SVM has zero-loss solutions with datasets for which there is *no* consistent hyperplane, as we show in prior work (Doran and Ray, 2013b).

sMIL (Convex). The sparse MIL (sMIL) algorithm uses weak constraints on the labels of instances in positive bags (Bunescu and Mooney, 2007). Assuming that in the worst case all but one instance in each positive bag is negative, the average instance label within positive bags is controlled by a "balancing constraint" $\frac{1}{|B_i|}\sum_{x_{ij}\in B_i}\big(\langle w, x_{ij}\rangle + b\big) \geq \frac{2-|B_i|}{|B_i|} - \xi_i$. The standard supervised constraint is used for negative instances. Because it uses weaker constraints on average instance labels,

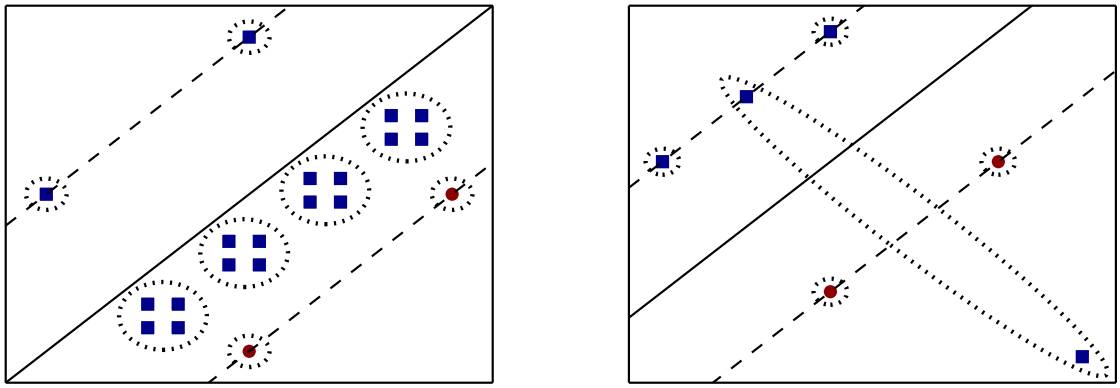


Figure 7.2: Synthetic datasets illustrating when soundness and/or completeness fail for sMIL and stMIL. (**Left**) An sMIL solution without loss allows a misclassification of an arbitrary number of bags whose averages lie close to the wrong size of the (inconsistent) classifier. (**Right**) A consistent MI separator with nonzero loss for sMIL and stMIL.

the sMIL approach is convex, but is neither sound nor complete. A counterexample to soundness is shown in Figure 7.2 (left). In the figure, all instances in positive bags are marked with blue squares, and the negative instances are marked with red circles. Because the misclassified bags contain four instances, they are allowed to be within $\frac{2-4}{4} = -\frac{1}{2}$ of the margin without any loss. Therefore, this solution is feasible and optimal without loss, but not consistent. In fact, an arbitrary number of positive bags can be placed within the margin as shown, leading to an arbitrarily poor classification of bags. A counterexample to the completeness of sMIL is shown in Figure 7.2 (right). While the solution is consistent, it is not feasible without loss because the average of the instances in the large positive bag lies below the separating line and therefore does not satisfy the balancing constraint.

stMIL (Sound). The sparse transductive MIL (stMIL) formulation includes the sMIL constraints, as well as $|\langle w, \phi(x_{ij})\rangle + b| \geq 1 - \xi_{ij}$ for every instance x_{ij} in a positive bag, which force instances within bags to be outside the margin (Bunescu and Mooney, 2007). The addition of these constraints makes the problem nonconvex. But like mi-SVM, these constraints impose a label on every instance, so stMIL is

sound by avoiding cases such as Figure 7.2 (left). The scenario in Figure 7.2 (right) is also a counterexample to the completeness of stMIL because the instances in the large bag satisfy the transductive constraint but violate the balancing constraint.

7.1.3 Fundamental Trade-Offs in Learning Instance Hyperplanes

None of the algorithms discussed above are sound, complete, *and* convex. In fact, this observation is no coincidence:

Theorem 7.1. *No MI optimization program is sound, complete, and convex.*

Proof. Suppose some MI optimization program is sound, complete, and convex. Then for any dataset, ℓ is a convex function, so $\{s \in \mathcal{S} : \ell(s) = 0\}$ is a convex set. Since the feasible region $\mathcal{F}$ is also convex, the set $\mathcal{Z} = \mathcal{F} \cap \{s \in \mathcal{S} : \ell(s) = 0\}$ is convex as well. Since $\mathcal{Z}$ is convex, it is a *path-connected* set. A set $V \in \mathcal{S}$ is path-connected when for any two points $v_1, v_2 \in V$, there exists a continuous parametric function $p : [0, 1] \to \mathcal{S}$ such that $p(0) = v_1$, $p(1) = v_2$, and $p([0, 1]) \subseteq V$. For a convex set, the lines connecting any two points in the set are such paths.

Furthermore, since the MI optimization program is sound and complete, the mapping between the solution and hyperplane spaces is such that $\mu(\mathcal{Z}) = \mathcal{C}$; that is, the set of consistent hyperplanes is the image of the $\mathcal{Z}$ under μ. Since μ is continuous and $\mathcal{Z}$ path-connected, this implies that $\mathcal{C}$ is path-connected. Intuitively, the image of a path-connect set under a continuous function is also path-connected because the composition of continuous functions is also continuous. Thus, any path in $\mathcal{Z}$ composed with μ produces a continuous path in $\mathcal{C}$.

However, consider the one-dimensional dataset with a positive bag $\{-2, 2\}$ and a negative bag $\{-1, 1\}$. It is possible that such a dataset was sampled from a process consistent with MI-GEN. A consistent linear "hyperplane" ($(w, b) \in \mathbb{R}^2$) must label

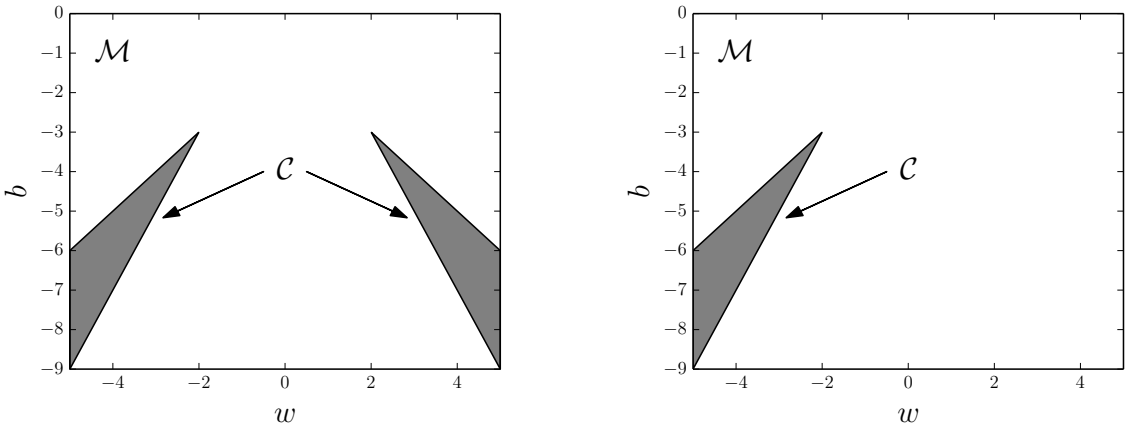


Figure 7.3: (**Left**) The shaded region shows the set $\mathcal{C}$ of consistent hyperplanes in the space of hyperplanes (w, b) for the example in the proof of Theorem 7.1. (**Right**) When a supervised labeling is applied to each instance, the set of consistent hyperplanes becomes a convex set.

either 2 or -2 "positive" and the other instances negative. The "support vectors" for these two scenarios are either -2 and -1, or 1 and 2. Therefore, the set of consistent hyperplanes is the union of the regions where $(1)w + b \leq -1$ and $(2)w + b \geq 1$, or where $(-1)w+b \leq -1$ and $(-2)w+b \geq 1$. This set is shown in Figure 7.3 (left), and is clearly not path-connected (no path connects the two disjoint regions). Thus, we have a contradiction with the implication that $\mathcal{C}$ is path-connected for every MI dataset, so there cannot be a sound, complete, and convex MI optimization program. □

Intuitively, the inability to satisfy all three properties is related to the disjoint nature of the set of consistent hyperplanes, which is in turn related to the combinatorial nature of the set of consistent instance labelings. In the standard supervised setting, this difficulty does not arise, since the set of consistent hyperplanes forms a convex set. For example, if we fix a labeling in the example of Theorem 7.1 so that -2 is positive and the other instances are negative, the set of consistent hyperplanes collapses to the convex set shown in Figure 7.3 (right).

Theorem 7.1 is in line with previous complexity results for MI classification via hyperplanes (Kundakcioglu et al., 2010; Diochnos et al., 2012). For clarity, we include

the theorem below, expressed in terms of our formalism:

Theorem 7.2. *Given an MI problem (B, Y), a set of bags with $|B_i| \leq k$, $k \geq 3$, the decision problem* MI−CONSIS *of determining whether there exists a hyperplane consistent with (B, Y) (i.e. is $\mathcal{C} = \varnothing$?) is* NP*-complete. It is also* NP*-complete to determine if $\mathcal{C}' = \varnothing$, where $\mathcal{C}'$ is the set of strongly consistent hyperplanes.*

The proof of Theorem 7.2 (Diochnos et al., 2012) reduces a 3−SAT instance to an instance of MI−CONSIS such that there is a strongly consistent hyperplane if the 3−SAT formula is satisfiable and no consistent (in the usual sense) hyperplane if the formula is not satisfiable. Thus, the proof works for either notion of consistency, though the distinction is not made in the original work.

If there were a sound, complete, and convex MI optimization program, the question $\mathcal{C} = \varnothing$ is equivalent to asking whether $\mathcal{Z} = \varnothing$, or whether there is a feasible, zero-loss solution to the MI optimization program. Similarly, if a set kernel approach is sound and complete, then $\mathcal{C}_{\mathrm{B}} = \varnothing \iff \mathcal{C}_{\mathrm{I}} = \varnothing$. Thus, if we could construct a sound and complete set kernel in polynomial time, we could use it in conjunction with a standard convex SVM formulation to search for a consistent bag classifier to decide whether the instances were separable. In either case, we could solve an NP-complete problem via a convex quadratic program (QP), which is generally regarded to be efficiently solvable, albeit in a non-Turing model of computation (Ben-Tal and Nemirovskiĭ, 2001).

7.1.4 Consequences for Learnability

Section 7.1 intuitively described why soundness and completeness were important properties for MI SVMs. The following provides a more detailed motivation for the importance of these properties and describes the relationship of the results above to recent learning theory results. This section concludes with a discussion of prior learnability results with respect to the new results provided in Chapter 4.

MI optimization programs of the form given in Equation 7.2 attempt to find a bag classifier by classifying individual instances with a hyperplane. Recent learning theory results justify this strategy for learning bag concepts. In particular, Sabato and Tishby (2012) show that the Vapnik–Chervonenkis (VC) dimension of the space of bag hypotheses is bounded in terms of the VC dimension of the underlying instance hypothesis space. The bound is general and holds for any instance hypothesis space, including spaces of hyperplanes. Therefore, given an appropriate instance hypothesis space (corresponding to a choice of kernel function), an MI SVM can learn to classify bags via the SRM strategy in Equation 7.2.

However, there are two caveats regarding the learnability results of Sabato and Tishby (2012). First, in order for an MI SVM to appropriately implement an SRM strategy, the loss function ℓ must provide an accurate assessment of empirical risk of a hypothesis. That is, the theoretical guarantees provided by Sabato and Tishby (2012) do not apply to algorithms whose loss functions do not reflect true empirical risk of a classifier on a training set. If an MI SVM is not sound, then empirical risk is not actually being minimized. On the other hand, if an MI SVM is not complete, then the approach unnecessarily restricts the hypothesis space, and the correct hypothesis might be incorrectly ruled out. Thus, these properties are fundamental (and usually trivially satisfied) in the supervised setting. However, as shown in Section 7.1.2, some existing MI SVM formulations *do not* satisfy both of these properties. Clearly, the process of constructing an SRM approach for MI classification is more subtle than that for supervised learning.

The second caveat is that although one perceived benefit of using an instance hypothesis class to construct a bag classifier is that the resulting instance hypothesis can be used for the instance-labeling task, there are currently no theoretical results guaranteeing that performing SRM at a bag level will produce a good hypothesis for instance-level labeling. In fact, it has been observed in practice that there is often

little correlation between the bag- and instance-level performance of MI classifiers (Tragante do Ó et al., 2011). In contrast, the theoretical results presented in Chapter 4 start with instance learnability results, derived by optimizing an instance-level risk function, and derive bag learnability results as a consequence. Furthermore, the hardness results above highlight the difficulty of constructing SRM approaches that can efficiently learn bag-level classifiers directly using instance-level hyperplanes. The new theoretical framework presented Chapter 4 provides an advantage to those designing new algorithms for MIL, which can leverage existing approaches for learning from individual labeled instances.

Below is a discussion of one final alternative approach to learning bag-level labels. While the methods discussed Section 7.1.2 using instance hypothesis classes to construct bag-level classifiers by combining instance labels, it is possible to directly learn bag labels by embedding bags in a feature space. The next section describes the theoretical properties of bag-level classifiers, followed by an empirical comparison of bag- and instance-level classifiers for MI classification.

7.2 Using Bag Kernels to Learn Instance Hyperplanes

As discussed in Section 2.3.1, the normalized set kernel (NSK) is a kernel defined on entire bags. Accordingly, the NSK maps entire bags into a feature space so that bag-level hyperplanes can be used for bag classification. In this section, we explore the properties of this approach with respect to the instance-labeling task. In particular, we are interested in whether bag kernels can be used to derive instance-level classifiers.

7.2.1 Bag-Level Soundness and Completeness

Above, Definition 7.1 defines consistency of an MI classifier with respect to labels assigned by an instance-labeling hyperplane. However, such a definition cannot be immediately extended to the NSK (Equation 2.8). In particular, since bags become individual points in a feature space, the set kernel corresponds to a space $\mathcal{H}_{\mathrm{B}}$ of hyperplanes in the space of *bags*, not instances. There is a separate classifying hyperplane space $\mathcal{H}_{\mathrm{I}}$ corresponding to the instance kernel k_{I}. As in Definition 7.1, there is a set $\mathcal{C}_{\mathrm{I}}$ of consistent solutions in the instance hyperplane space, and we say that k_{I} *separates* instances when $\mathcal{C}_{\mathrm{I}} \neq \varnothing$ (there is some consistent hyperplane that separates instances). Similarly, the set of consistent hyperplanes $\mathcal{C}_{\mathrm{B}}$ in the bag hyperplane space $\mathcal{H}_{\mathrm{B}}$ are the hyperplanes that, in a supervised learning sense, separate bags by assigning the appropriate label $F(B_i)$ to each bag B_i in the dataset. A set kernel k_{NSK} separates bags when $\mathcal{C}_{\mathrm{B}} \neq \varnothing$.

Since the set kernel approach uses the standard supervised SVM quadratic program with a modified kernel, its loss function is not problematic ($\mu(\mathcal{Z}) = \mathcal{C}_{\mathrm{B}}$); rather, the questions of soundness and completeness must now consider the relationship between consistent hyperplanes in the bag ($\mathcal{C}_{\mathrm{B}}$) and instance ($\mathcal{C}_{\mathrm{I}}$) hyperplane spaces, as done in prior work (Gärtner et al., 2002).

Definition 7.6 (Soundness for Set Kernels)**.** *A set kernel k_{NSK} is* sound *w.r.t. instance kernel k_{I} iff for any MI dataset, k_{NSK} separates bags only if k_{I} separates instances; i.e.,* $(\mathcal{C}_{\mathrm{B}} \neq \varnothing) \implies (\mathcal{C}_{\mathrm{I}} \neq \varnothing)$.

Definition 7.7 (Completeness for Set Kernels)**.** *A set kernel k_{NSK} is* complete *w.r.t. instance kernel k_{I} iff for any MI dataset, k_{NSK} separates bags if k_{I} separates instances; i.e.,* $(\mathcal{C}_{\mathrm{I}} \neq \varnothing) \implies (\mathcal{C}_{\mathrm{B}} \neq \varnothing)$.

For set kernels, soundness and completeness intuitively mean that it is possible to construct a set kernel from an instance kernel such that a zero-loss, consistent

hyperplane exists in the set kernel feature space if and only if one exists in the original instance kernel feature space. Note that these notions do not require a bijection between $\mathcal{C}_{\mathrm{B}}$ and $\mathcal{C}_{\mathrm{I}}$, because in general the feature maps corresponding to k_{I} and k_{NSK} can have an arbitrarily complex relationship depending on the specific set kernel. We show below that even this weak feature space correspondence is *not* maintained by the NSK: while it is always possible to construct a complete NSK, such kernels might not be sound in the sense of Definition 7.6.

7.2.2 Properties of Bag Hyperplane Classifiers

Prior work describes the application of set kernels, as defined Equation 2.8, to the MI classification problem (Gärtner et al., 2002). The central theoretical results of Gärtner et al. (2002) relate the ability of an instance kernel k_{I} to separate instances and the ability of the corresponding set kernel k_{Set} to separate bags. In order to ensure separability of bags by the set kernel given separability of instances given the instance kernel, the set kernel is generalized to the MI kernel, k_{MI}:

$$k_{\mathrm{MI}}(X_1, X_2) = \sum_{i=1}^{m} \sum_{j=1}^{n} k_{\mathrm{I}}^{p}(x_i, x_j). \tag{7.3}$$

Here, p is an integer power of the instance kernel. Gärtner et al. (2002) present a proof of the following result, which is obtained by appropriately selecting p:

Theorem 7.3 (Gärtner et al.)**.** *A bag-level concept*[3] *is separable by* k_{MI} *with nonzero margin if and only if the underlying instance concept is separable by the kernel* k_{I} *with nonzero margin.*

Theorem 7.3 of seems to prove that the unnormalized MI kernel is sound and complete with respect to Definition 7.6 and Definition 7.7. However, we show using counterexamples that soundness does not hold for the NSK. On the other hand, we

[3]The bag-level concept is called the *MI concept* by Gärtner et al. (2002).

show that completeness still holds for the unnormalized MI kernel, and we extend these results to the normalized version of the MI kernel:

$$k_{\text{NSK}}(X_1, X_2) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} k_{\text{I}}^{p}(x_i, x_j)}{f_{\text{norm}}(X_1) f_{\text{norm}}(X_2)}. \tag{7.4}$$

The section concludes with a discussion of the implications of the NSK's lack of soundness.

Lemma 4.2 in Gärtner et al. (2002) shows that if an underlying instance concept is separable by k_{I}, then there is some power $p > 0$ for which the *unnormalized* set kernel k_{MI} separates bags. Gärtner et al. (2002) use a different, equivalent notion of consistency in which instead of assigning ± 1 labels to instances, a hyperplane c_ϕ in the feature space of a kernel with feature map ϕ assigns a label $1 \leq \langle \phi(x_{ij}), c_\phi \rangle$ to all positive instances, and $0 \leq \langle \phi(x_{ij}), c_\phi \rangle \leq 1 - \epsilon$ to all negative instances. Here, $\epsilon > 0$ is some arbitrary margin. To better align our results with prior work, we adopt these conventions for the remainder of this section, without loss of generality. We start with our positive result: the completeness of the NSK.

Proposition 7.1. *A bag-level concept is separable with k_{NSK} (Equation 7.4), using sufficiently large p, if the underlying instance concept is separable with margin ϵ by k_{I}, the bag size is bounded by m, and there are constants L and U such that $0 < L \leq f_{norm}(B_i) \leq U$ for all bags B_i.*

Proof. Choose an integer power $p > 0$ satisfying $p > -\frac{\log(mU/L)}{\log(1-\epsilon)}$.

Let c_ϕ be the vector such that $g(x_{ij}) = \langle \phi(x_{ij}), c_\phi \rangle$ separates the instance concept in the instance kernel feature space. Then consider the function G on bags:

$$G(B_i) = \frac{U}{f_{\text{norm}}(B_i)} \sum_{x_{ij} \in B_i} \langle \phi(x_{ij}), c_\phi \rangle^p .$$

If B_i is a positive bag, then by the MI assumption, at least one instance $x_{ij} \in B_i$

satisfies $\langle \phi(x_{ij}), c_\phi \rangle \geq 1$, so:

$$G(B_i) \geq \frac{U(1^p)}{f_{\text{norm}}(B_i)} \geq \frac{U}{U} = 1.$$

On the other hand, if B_i is a negative bag, then all instances $x_{ij} \in B_i$ satisfy $\langle \phi(x_{ij}), c_\phi \rangle \leq 1 - \epsilon$, so:

$$G(B_i) \leq \frac{Um(1-\epsilon)^p}{f_{\text{norm}}(B_i)} \leq \frac{Um}{L}(1-\epsilon)^p < \frac{Um}{L}(1-\epsilon)^{-\frac{\log(mU/L)}{\log(1-\epsilon)}} = 1.$$

Therefore, this function separates bags.

To see that the function $G(B_i)$ can be written as a inner product in the feature space corresponding to k_{NSK}, first note that if $k(x,y) = \langle \phi(x), \phi(y) \rangle$, and we raise it to power p, this is also a positive definite kernel, which is equivalent to some $\langle \psi(\phi(x)), \psi(\phi(y)) \rangle$. Therefore, the NSK feature map is given by

$$\Phi(B_i) = \frac{\sum_{x_{ij} \in B_i} \psi(\phi(x_{ij}))}{f_{\text{norm}}(B_i)}$$

. Therefore, we can rewrite G as:

$$\begin{aligned} G(B_i) &= \frac{U}{f_{\text{norm}}(B_i)} \sum_{x_{ij} \in B_i} \langle \phi(x_{ij}), c_\phi \rangle^p \\ &= \frac{U}{f_{\text{norm}}(B_i)} \sum_{x_{ij} \in B_i} \langle \psi(\phi(x_{ij})), \psi(c_\phi) \rangle \\ &= \left\langle \frac{\sum_{x_{ij} \in B_i} \psi(\phi(x_{ij}))}{f_{\text{norm}}(B_i)}, U\psi(c_\phi) \right\rangle = \langle \Phi(B_i), C_\Phi \rangle . \end{aligned}$$

So G is a hyperplane C_Φ in the normalized set kernel feature space. □

Corollary 7.1. *An bag-level concept is separable by k_{NSK} (Equation 7.4) with averaging normalization, using $p > -\frac{2\log m}{\log(1-\epsilon)}$ if the underlying instance concept is separable with margin ϵ by k_{I} and the bag size is bounded by m. The required value of p is*

simply twice that required in the unnormalized case.

Proof. Given an upper bound m on bag size, the averaging normalization function $f_{\text{norm}}(B_i) = |B_i|$ is bounded by $1 \leq |B_i| \leq m$, so using $L = 1$ and $U = m$ in Proposition 7.1, $p > -\frac{2 \log m}{\log(1-\epsilon)}$ works to separate bags. □

As discussed in Section 2.3.2, the NSK with averaging normalization is equivalent to the kernel mean embedding. Thus, Corollary 7.1 shows that the kernel mean embedding is complete in the sense of Definition 7.7. Proposition 7.1 can be viewed as analogous to Proposition 6.1 in the standard MI generative process in which bags are finite sets of instances rather than distributions.

Another noteworthy normalization procedure for the set kernel is feature-space normalization, given by:

$$f_{\text{norm}}(X) = \|\Phi(X)\|_{\mathcal{H}} = \sqrt{k_{\text{NSK}}(X, X)} = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} k_{\text{I}}^p(x_i, x_j)}. \tag{7.5}$$

Essentially, the feature-space normalization ensures that the vector associated with a set X has norm 1 in the set kernel feature space. We can show the completeness of this normalized set kernel as well:

Corollary 7.2. *A bag-level concept is separable by k_{NSK} (Equation 7.4) with feature-space normalization, using $p > -\frac{3 \log m}{2 \log(1-\epsilon)}$ if the underlying instance concept is separable with margin ϵ by k_{I}, the bag size is bounded by m, and k_{I} is the radial basis function (RBF) kernel (Equation 2.9).*

Proof. The value of the RBF kernel is always positive, and bounded above by 1. Therefore, if we raise the kernel to any power, p, this only decreases its value. As a result, since bag sizes are bounded by m,

$$f_{\text{norm}}(X) = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} k_{\text{I}}^p(x_i, x_j)} \leq \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} 1} = m.$$

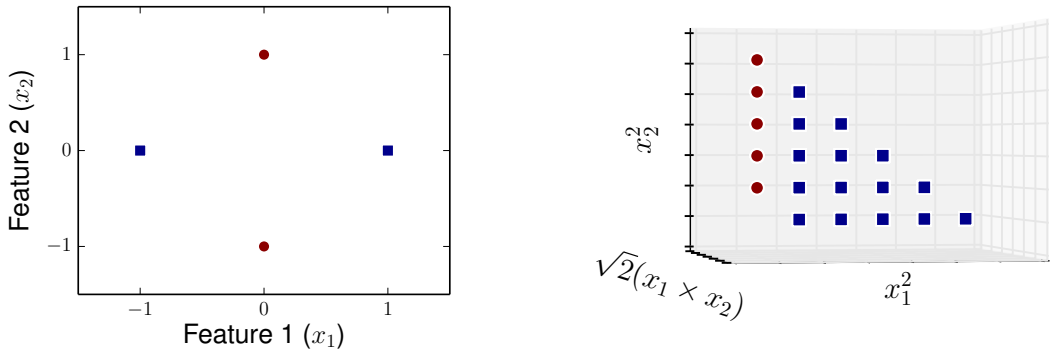


Figure 7.4: Even though instances are not separable with a linear kernel (**left**), all resulting bags are separable in the feature space of the MI kernel (**right**).

On the other hand, the RBF kernel has the property that for any x, $k_{\mathrm{I}}(x, x) = 1$. Hence, even if all other pairwise kernel values are zero, we have:

$$f_{\mathrm{norm}}(X) = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{m} k_{\mathrm{I}}^{p}(x_i, x_j)} \geq \sqrt{\sum_{i=1}^{m}\sum_{j=i} 1} = \sqrt{m}.$$

Therefore, we obtain the required value for p by using $L = \sqrt{m}$ and $U = m$ in Proposition 7.1. □

In contrast to the completeness shown in Proposition 7.1, a simple example can be used to show that the NSK is not sound in the sense of Definition 7.6. That is, the NSK can separate bags even when the corresponding instance kernel cannot separate the instances. Consider the instance space $\mathcal{X} = \{(1,0), (-1,0), (0,1), (0,-1)\}$, with respective labels $\{+1, +1, -1, -1\}$ corresponding to XOR, as illustrated in Figure 7.4 (left). With a linear instance kernel $k_{\mathrm{I}}(x, x') = \langle x, x' \rangle$, the instance concept is clearly not separable. However, with $p = 2$, the set kernel $k_{\mathrm{MI}}(X, X') = \sum_{x,x'} \langle x, x' \rangle^2$ *can* separate any MI dataset derived from these instances. To see why, consider the explicit feature map ϕ of the quadratic kernel $(x_1, x_2) \mapsto (x_1^2, \sqrt{2}x_1x_2, x_2^2)$, as described in Chapter 2. Then, the set kernel feature map is the sum of instance kernel feature

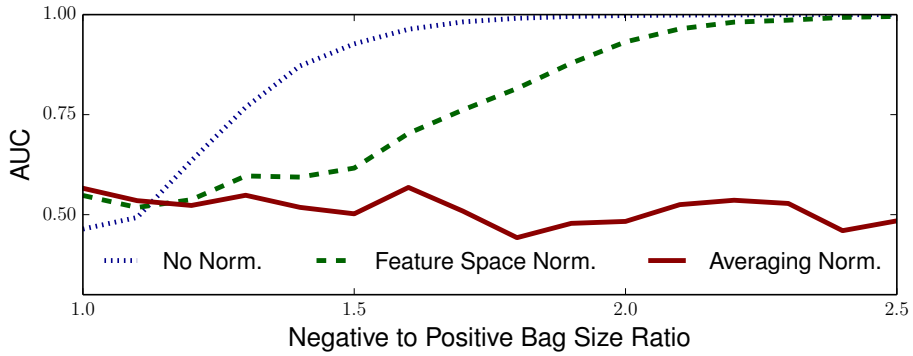


Figure 7.5: Certain types of NSK separate bags of different sizes with no underlying MI concept. The y-axis shows AUC.

maps: $\Phi(X) = \sum_{x \in X} \phi(x)$. The linear function $f(X) = \langle \Phi(X), (1, 0, 0) \rangle$ in the feature space of $k_{\text{MI}}(\cdot, \cdot)$ then separates any MI dataset, since the first component of the map $\Phi(X)$ is nonzero if and only if X contains either $(1, 0)$ or $(-1, 0)$. The feature maps corresponding to bags of size at most 5 are shown in the right-hand side of Figure 7.4. Gärtner et al. (2002) states that if f_{MI} separates an MI dataset, then $f_{\text{I}}(x) = f_{\text{MI}}(\{x\})$ can separate instances. However, this is not sound under our definition because $f_{\text{MI}} \notin \mathcal{H}_{\text{I}}$; i.e., there is no instance hyperplane in the original instance hyperplane space corresponding to the bag hyperplane applied to singleton sets.

Finally, another form of unsoundness arises for set kernels due to the effects of bag size. For example, consider an MI problem in which all bag instances are identical (say $x_{ij} = 1$), but positive bags have size 10 while negative bags have size 5. Then for an unnormalized linear kernel, the feature mapping of a positive bag will be 10, while the negative bag feature space value will be 5. Clearly, there is no underlying MI concept; yet, the set kernel is able to separate positive and negative bags in the feature space via the effects of bag size. We illustrate this further in Figure 7.5 using synthetic datasets. In these datasets, each instance has 25 features, which are drawn independently from the standard normal distribution $\mathcal{N}(0, 1)$. There are 50 positive bags, each with 10 instances, and 50 negative bags of sizes that vary across datasets.

Even though there is no underlying instance concept to learn, the set kernels with either no normalization or feature space normalization (Equation 7.5) can learn to distinguish between positive and negative bags as the discrepancy in sizes grows.

Therefore, the NSK is complete and convex, but not sound with respect to Definition 7.6. However, because the NSK is used in conjunction with the standard SVM formulation, it can find bag-level hyperplanes without this issue. Thus, the NSK has the ability to perform well with respect to the bag-labeling task, but its unsoundness suggests that it might not perform well with respect to the instance labeling task (if the bag-labeling function is restricted to singleton sets).

7.3 Empirical Evaluation

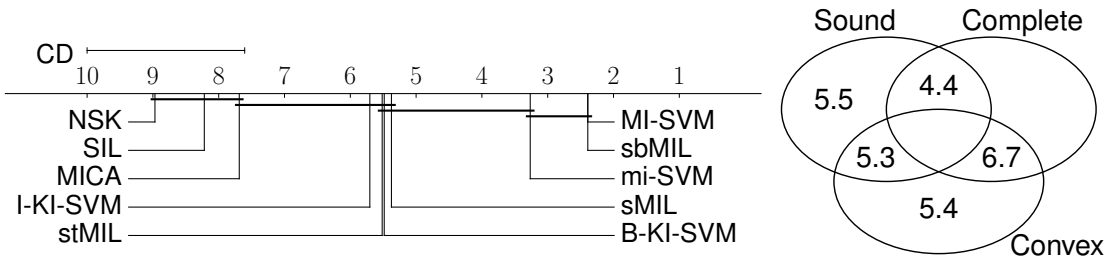
Below, we evaluate the performance of the instance- and bag-level classifiers described in the previous sections. In particular our theoretical analysis suggests several hypotheses that we will evaluate. As discussed in Section 7.1.4, for instance-level classifiers we hypothesize that soundness and completeness are important properties of classifiers, but that soundness is a more necessary property than completeness for generalization. Additionally, we hypothesize that bag-level kernels such as the NSK will not perform well at the instance-labeling task (Section 7.2). We examine these hypotheses below.

7.3.1 Methodology

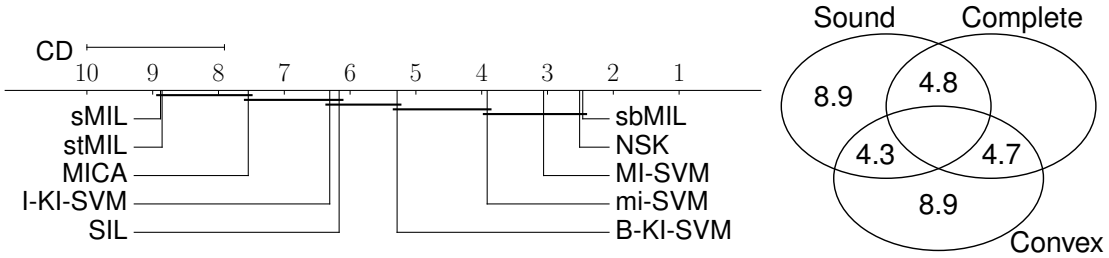
Due to the computational complexity of instance-based approaches, we use 42 MI datasets from various domains to evaluate performance (see Table A.2). Instance labels are available for 32 CBIR and Newsgroups datasets, and the remaining datasets come from CBIR, text categorization, and 3D-QSAR. As in previous experiments, instance labels are only used for evaluation, not training.

Since soundness and completeness are properties of algorithms that attempt to maximize accuracy, we evaluate instance- and bag-level accuracy as performance measures. We use 10-fold cross-validation to access the accuracy of the 12 algorithms described above, with random parameter search (Bergstra and Bengio, 2012) and 5-fold inner cross-validation to select parameters. We use the RBF kernel for all approaches, and it serves as the instance kernel in set kernel approaches. We implement the normalized set kernel with averaging normalization (Equation 2.7). The RBF kernel parameter γ is selected from $[10^{-6}, 10^{1}]$ and the regularization parameter C from $[10^{-2}, 10^{5}]$. For the set kernel, we fix $p = 1$, but with an RBF kernel, p can be absorbed into the constant γ. The sparse balanced MIL (sbMIL) approach uses a parameter η as an estimate of the fraction of true positive instances within positive bags, which we search for within the range $[0, 1]$. For an algorithm requiring $m \in \{1, 2, 3\}$ parameters, we evaluate 5^m random parameter combinations for the search. For techniques that rely on iteratively solving QPs, iteration continues at most 50 times or until the change in objective function value falls below 10^{-6}. MI classification algorithm (MICA) was originally formulated using L_1 regularization, but in our experiments we use the L_2 norm to provide a more direct comparison to other approaches. We only use bag labels when performing parameter cross-validation, even for the instance-labeling task (we only use instance labels to perform the final evaluation of instance predictions pooled across the ten outer folds).

To statistically compare the classifiers, we use the approach described by Demšar (2006), as in the experiments in Section 5.5. We use the Friedman test to reject the null hypothesis that the algorithms perform equally at an $\alpha = 0.001$ significance level. Finally, we plot the average ranks with a critical difference diagram, which uses the Nemenyi test to identify statistically equivalent groups of classifiers at an $\alpha = 0.05$ significance level. The resulting diagrams are shown in Figure 7.6. Also shown in Figure 7.6 are Venn diagrams with the average ranks of approaches with different



(a) Instance Accuracy



(b) Bag Accuracy

Figure 7.6: Ranks (lower is better) of the various MI SVM approaches on the instance and bag labeling tasks using accuracy for evaluation. The critical difference diagrams (**Left**) show the average rank of each technique across datasets, with techniques being statistically different at an $\alpha = 0.1$ significance level if the ranks differ by more than the critical difference (CD) indicated above the axis. Thick horizontal lines indicate statistically indistinguishable groups (i.e. a technique is statistically different from any technique to which it is *not* connected with a horizontal line). The Venn diagrams (**Right**) show the average ranks of techniques within each Sound/Complete/Convex categorization.

subsets of the soundness, completeness, and convexity properties. In addition to the approaches in Figure 7.1, the NSK approaches are categorized as complete and convex, as described in Section 7.2.2. A detailed description of the datasets and methodology can be found in Appendix A, with a full table of results in Appendix A.2.3.

7.3.2 Results and Discussion

Effect of Soundness and Completeness

In general, the results in Figure 7.6 are consistent with the hypothesis that soundness and completeness lead to more accurate classifiers than approaches that lack these

properties. One interesting exception is sbMIL, which is among the best approach for both instance- and bag-labeling despite being incomplete. The sbMIL algorithm lacks completeness because it assumes that there is some fraction η of positive instances within positive bags. Accordingly, it penalizes consistent hypotheses that do not positively label the appropriate number of instances in positive bags. However, although sbMIL essentially shrinks the space of feasible hypotheses, it does so in a reasonable way. Furthermore, by using cross-validation to select η, a good classifier consistent with the data can still be found. Hence, sbMIL is an example of how sacrificing completeness appropriately might benefit performance.

Effect of the Labeling Task

In general, there are similar patterns of performance across techniques with different combinations of properties for both the instance- and bag-labeling tasks. However, one important exception is the NSK. For the instance-labeling task, the NSK is among the worst approaches, which is consistent with our analysis of the NSK as being unsound with respect to instance labeling. On the other hand, for the bag-labeling task, the NSK is actually one of the best approaches. Again, this is consistent with the fact that the NSK is sound and complete with respect to bag-labeling, for which a standard supervised SVM is used. Our results show that even though the NSK can find good bag separators, simply applying the resulting separator to instances as singleton bags does not perform well at separating instances. Prior work empirically observed that there might be little correlation between the performance of algorithms on the two labeling tasks (Tragante do Ó et al., 2011). Our theoretical analysis now provides a basis for understanding why this is the case for the NSK.

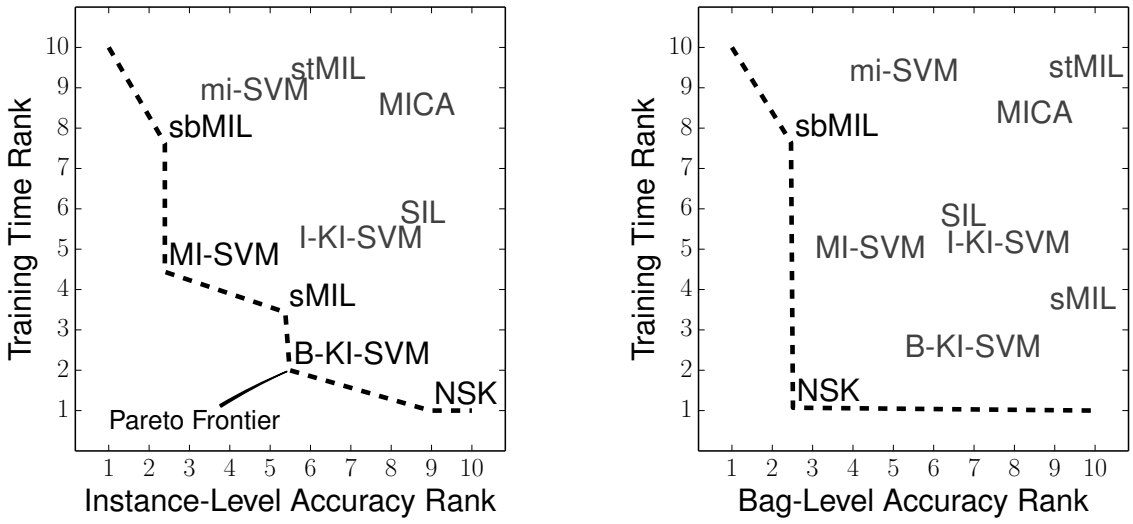


Figure 7.7: Comparison of MI SVM classifiers in terms of running time and classification performance (accuracy). Lower ranks correspond to better performance and faster training time. The Pareto frontier shows algorithms that are not dominated by any other algorithm along both dimensions.

Time and Space Requirements

The relative running times of the algorithms are summarized along with classification performance in terms of accuracy in Figure 7.7. The most noticeable trend is that instance kernel approaches are much more computationally expensive than set kernel approaches, since kernel sizes are $O(|X|^2)$ rather than $O(|B|^2)$. Larger kernels require more memory and lead to significantly increased training time due to the increased number of variables in the optimization program. Hence, in terms of bag-level classification performance, the NSK clearly dominates the other approaches except for sbMIL. However, sbMIL does not significantly outperform the NSK, and it requires a much longer training time due to the extra search for the balancing parameter η. For instance-level classification performance, the MI-SVM achieves the best accuracy with a relatively faster training time than sbMIL.

7.4 Summary

In this chapter, we first described a trade-off between soundness, completeness, and convexity properties of SVMs that attempt to learn bag classifiers using instance-level hyperplanes. We analyzed existing approaches in terms of these properties and proved that no approach can have all three properties simultaneously. Then, we performed an empirical comparison from which we can draw several conclusions (see Figure 7.6). First, although our results show that sound and complete approaches generally perform well on the instance-labeling task, the only theoretical guarantees for such approaches apply to their performance on the bag-labeling task (Sabato and Tishby, 2012), as described in Section 7.1.4. However, the results show that for bag-labeling, approaches like the NSK that directly label bags are more efficient and effective at performing this task. Therefore, our results suggest that the instance- and bag-labeling tasks should be treated separately. For instance labeling, the results in Chapter 4 provide theoretical guarantees and show that supervised approaches can perform well at instance labeling given large enough sample sizes. On the other hand, for bag labeling, it makes more sense to use bag-level classifiers rather than adapting an instance hypothesis space to label bags.

Chapter 8

Shuffled Multiple-Instance Learning

In the previous chapter, we saw how the two-level generative model can be used to analyze existing MIL classification algorithms. This chapter introduces a new resampling method for MIL, Shuffled Multiple-Instance Learning (SMILe), and uses the theoretical framework introduced Chapter 3 to analyze SMILe.[1] As shown below, resampling approaches such as SMILe can improve performance when the size of the training set is small. Thus, SMILe complements the asymptotic learnability results presented in Chapter 4 by providing a practically effective approach for learning from small samples. Furthermore, we show empirically that SMILe can improve the instance-level accuracy of MI classifiers, even with large training samples.

8.1 Ensemble and Resampling Methods

As described in previous chapters, there are numerous algorithms available to perform classification in the supervised for MI settings. Given prior knowledge about a problem, one might make an informed decision about which algorithm is best to use.

[1] The SMILe approach was described previously in Doran and Ray (2013a).

However, another approach is to simply create an *ensemble* of different classifiers and combine their outputs via voting, averaging, or some other scheme.

Alternatively, one might train multiple copies of the *same* classifier using different versions of the training dataset. One popular resampling approach is *boosting* (Freund and Schapire, 1995). Boosting trains a series of classifiers in which the weights of individual training examples are adjusted at each iteration to modify the training distribution. The weights of examples misclassified by the classifier in the previous iteration are increased so that the algorithm "focuses" more on classifying them correctly. At the end of the process, a weighted combination of the individual classifiers is used as the final classifier. Interestingly, the boosting procedure has the theoretical property that it can transform a "weak" classification algorithm (guaranteed to perform at least better than chance) into a "strong" classifier that can PAC learn the target concept.

An even simpler ensemble method is a resampling approach known as bootstrap aggregation, or "bagging" (Breiman, 1996). Bagging approximates drawing many samples from the underlying training distribution by resampling with replacement "bootstrap replicates" from the original training set. A classifier is trained on each bootstrap replicate, and the resulting classifiers are averaged together to produce a final ensemble classifier. Bagging can reduce the tendency of a classifier to *overfit* to the original training set.

The boosting and bagging approaches have been extended to the MI setting (Zhou and Zhang, 2003; Auer and Ortner, 2004; Antić and Ommer, 2013). However, these adaptations apply only to the bag-labeling task. In the boosting approach (Auer and Ortner, 2004), entire bags are reweighted to boost a weak bag classifier into a strong bag classifier. Similarly, the MI bagging approaches (Zhou and Zhang, 2003; Antić and Ommer, 2013) resample entire bags to improve the performance of bag-level classification.

Below, we describe a novel resampling approach, SMILe and its basic properties. Then, we analyze the properties of SMILe when combined with both instance- and bag-labeling algorithms using insights from the proposed generative model.

8.2 The SMILe Approach

In MI bagging approaches (Zhou and Zhang, 2003), entire bags are resampled to create bootstrap replicates of an MI dataset. Suppose instead that bags are split and the instances recombined to produce new bags. For example, consider a small MI dataset with three positive bags $\{x_1, x_2\}$, $\{x_3, x_4\}$, and $\{x_5, x_6\}$. According to the standard MI assumption, at least one instance in each bag is positive. We assume that this standard relationship between bag and instance labels in the training set holds. That is, bag samples are large enough such that $F(B_i) = \widehat{F}(B_i)$ for every bag in the training set. Without loss of generality, let x_1, x_3, and x_5 be the "true" positive instances within these positive bags. Now, suppose we (1) combine all instances from positive bags, and (2) randomly sample three instances without replacement from the combined set, then iterate these steps. We may get bags such as the following: $\{x_1, x_4, x_6\}$ and $\{x_2, x_4, x_5\}$.

It is no coincidence that each of the bags above contains at least one positive instance, and is therefore also a positive bag. The only way to sample a negative bag of size three is to sample the instances $\{x_2, x_4, x_6\}$ in some order, which occurs with probability only $\left(\frac{3}{6}\right)\left(\frac{2}{5}\right)\left(\frac{1}{4}\right) = 5\%$. Thus, each new bag produced by shuffling together and sampling instances from positive bags is positive with high probability. The shuffling process is uniformly random, and does not require knowing or estimating individual instance labels during the resampling process. Of course, we could just as easily sample shuffled negative bags from the set of all instances in negative bags. The shuffled negative bags are *always* negative, since by assumption, only negative

Algorithm 1 SMILe: Shuffled Multiple-Instance Learning

Require: MI dataset (B, Y), noise rate ϵ_s, base learner $\mathcal{A}$, number of shuffled positive bags $|S_p|$, number of shuffled negative bags $|S_n|$

$B_p \leftarrow \{B_i \mid Y_i = +1\}$
$X_p \leftarrow \bigcup_{B_i \in B_p} B_i$ ▷ "Positive Bag Instances"
$B_n \leftarrow \{B_i \mid Y_i = -1\}$
$X_n \leftarrow \bigcup_{B_i \in B_n} B_i$ ▷ "Negative Bag Instances"
$s \leftarrow \left\lceil \frac{|X_p|}{|B_p|} \log \frac{1}{\epsilon_s} \right\rceil$ ▷ Shuffled bag size based on Proposition 8.1
$S_p \leftarrow \text{SAMPLESHUFFLEDBAGS}(X_p, |S_p|, s)$
$S_n \leftarrow \text{SAMPLESHUFFLEDBAGS}(X_n, |S_n|, s)$
$B' \leftarrow B \cup S_p \cup S_n$
$Y' \leftarrow Y \cup (|S_p| \times [+1]) \cup (|S_n| \times [-1])$
$\mathcal{C} \leftarrow \mathcal{A}(B', Y')$ ▷ Train Classifier
return $\mathcal{C}$

function SAMPLESHUFFLEDBAGS(`pool`, `number`, `bag_size`)
    $S \leftarrow [\,]$
    for $i \leftarrow 1$ to `number` **do**
        $\texttt{pool}_i \leftarrow \texttt{pool}$ ▷ Same pool used for each shuffled bag
        $S_i \leftarrow \text{SAMPLEBAGWITHOUTREPLACEMENT}(\texttt{pool}_i, \texttt{bag\_size})$
    end for
    return S
end function

instances appear in negative bags. As we show later, for some algorithms, the addition of shuffled negative bags can be useful to mitigate class imbalance induced by the addition of shuffled positive bags.

Because resampling procedure described above shuffles instances across positive bags, we call the approach Shuffled Multiple-Instance Learning (SMILe). The pseudocode for SMILe is shown in Algorithm 1. The approach takes an MI dataset consisting of a list of bags B with associated binary labels Y ($Y_i \in \{-1, +1\}$). Each bag is a set of instances, or real-valued feature vectors: $B_i = \{x_{ij} \in \mathbb{R}^n\}$. The positive bags are denoted by $B_p = \{B_i \mid Y_i = +1\}$, and the set of positive bag instances by $X_p = \bigcup_{B_i \in B_p} B_i$. Since positive bags also contain negative instances, not all instances in X_p are positive. Similarly, X_n is the set of instances from negative bags, and all instances in X_n *are* negative. The notation used throughout the paper is summarized

Table 8.1: Legend of the basic notation used in Chapter 8.

Symbol	Definition	Symbol	Definition
B	Bags	S_p	Shuffled Positive Bags
Y	Bag Labels	S_n	Shuffled Negative Bags
B_p	Pos. Bags, $\{B_i \mid Y_i = +1\}$	s	Shuffled Bag Size
B_n	Neg. Bags, $\{B_i \mid Y_i = -1\}$	ϵ_s	Shuffled Pos. Bag Error
X	Instances, $\bigcup_{B_i \in B} B_i$		
X_p	Pos. Bag Insts., $\bigcup_{B_i \in B_p} B_i$		
X_n	Neg. Bag Insts., $\bigcup_{B_i \in B_n} B_i$		

in Table 8.1.

The basic SMILe algorithm works by generating sets S_p of additional positive bags and S_n of additional negative bags. Each shuffled bag contains s instances sampled without replacement from X_p or X_n, the set of instances pooled from all positive or negative bags, respectively. As we describe in Section 8.3, the shuffled bag size s can be chosen to ensure a bound on the noise incurred by positively labeling shuffled positive bags. The instances *are* replaced between the sampling of each bag in S_p or S_n. An MI classification algorithm $\mathcal{A}$ is given the augmented training set $B \cup S_p \cup S_n$ with associated labels, and the resulting classifier is used to predict new instance or bag labels.

As a more concrete example of the shuffling process for positive bags, consider an example from the content-based image retrieval (CBIR) domain. In this domain, images are represented as structureless "bags" of segments, which roughly correspond to object in the image. An image is positive if it contains at least one "positive" object of interest. Figure 8.1 shows two positive images in which a Coke can is the object of interest. Applying SMILe to these bags, we take the segments from the two images, shuffle them together, and sample two sets of instances to produce the shuffled bags on the right of Figure 8.1. As is apparent in the figure, the shuffled bags also contain Coke cans, and so they are also positive bags. By sampling enough instances in the shuffled bags, we ensure that they contain positive instances with high probability.

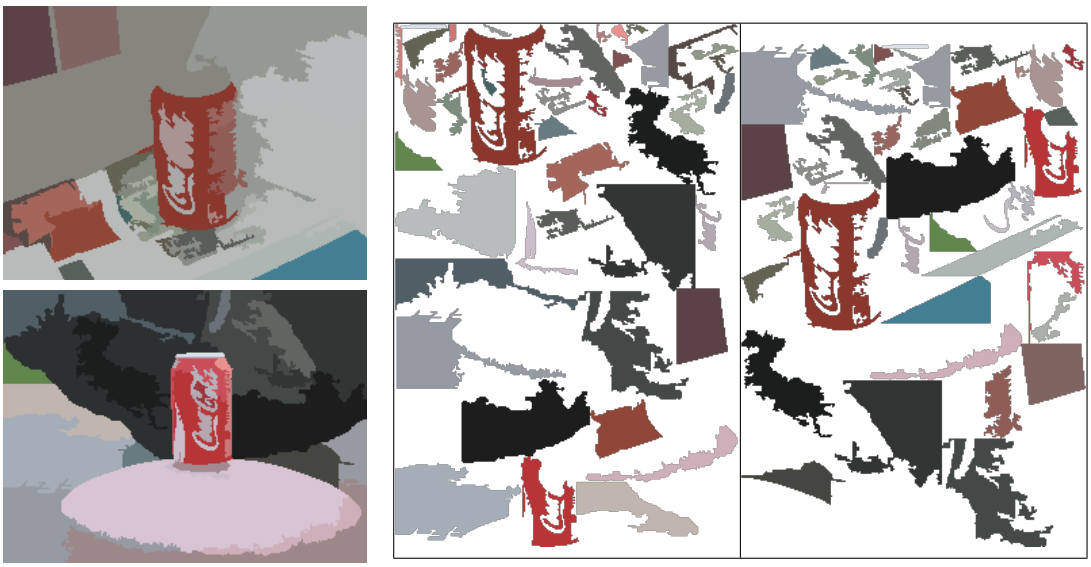


Figure 8.1: **Left:** Two positive images from the "Coke Can" class. **Right:** Two bags generated by sampling instances from the two images above.

SMILe adds additional examples that can improve a classifiers performance on the instance-labeling task. Intuitively, the justification for SMILe is that instances in shuffled bags are presented to an MI learning algorithm in different contexts, rather than remaining together in the same original bags as in other resampling approaches (Zhou and Zhang, 2003). For the instance-labeling task in the MI setting, the "context" that a bag provides corresponds to a logical constraint on the labels of the instances within the bag. In particular, if all instances have underlying Boolean labels, then the label of a bag is the logical disjunction of the labels of the instances within the bag. For MI learning algorithms that attempt to learn a concept f that assigns "true" or "false" labels to positive and negative instances, respectively, positive bags provide constraints on the set of concepts that are consistent with the possible instance labelings in the dataset.

Interpreting bags as constraints on the set of consistent instance labeling functions, we see that adding new bags can *add new constraints* on an instance-labeling concept. For example, consider the small dataset above with three positive bags of two instances each. From the first bag, $\{x_1, x_2\}$, we know that either x_1 or x_2 must be

a positive instance. After we add the shuffled bags, we infer that with high probability, x_1, x_4, or x_6 must be positive as well. If an algorithm is confident that x_4 and x_6 are negative, the added constraint means that x_1 is likely to be the positive instance in the shuffled bag and, thus, also in the original bag $\{x_1, x_2\}$. Generally speaking, these additional shuffled bags introduce new constraints that are deduced probabilistically from the initial training set. While rule-based algorithms might logically infer the additional constraints from the original data, we show below that statistical learning algorithms can benefit from having the additional constraints explicitly provided.

SMILe can also improve the performance of bag-labeling algorithms by providing entirely new examples to an algorithm that uses bag-level information for classification. In particular, some MI learning algorithms treat bags as graphs (Zhou et al., 2009) or map entire bags to feature spaces (Gärtner et al., 2002). For such algorithms, shuffled bags might possess properties not possessed by the original bags in the dataset. On the one hand, this means that SMILe can provide additional information to such algorithms using either positive *or* negative shuffled bags. For the instance-labeling case, shuffled negative bags do not provide any additional constraint information, since negative bags already imply instance-level constraints on the definitively negative instances they contain. On the other hand, as we show, because the shuffled bags are no longer necessarily sampled from the underlying training distribution over bags, SMILe may be less beneficial for the bag-labeling task than the instance-labeling task. However, we also show empirically that when training set sizes are small, such as in an active learning setting, the benefit of additional examples generated by SMILe outweighs the issues caused by shuffled bags affecting the training distribution over bags.

For either bag- or instance-labeling algorithms, by adjusting the size of shuffled bags, we can control the noise rate on the labels of additional positive bags (see Proposition 8.2 in the following section). The noise adjustment provides an additional

benefit of SMILe. By sampling large enough positive bags, it is possible to reduce existing positive bag label noise in the dataset. As we show below, the bag size only grows logarithmically with the inverse of the desired noise rate, so the shuffled bags required are not too large. The noise reduction improves generalization and reduces over-fitting, so SMILe also provides a benefit similar to standard bootstrap resampling.

As with other resampling approaches, SMILe can be used with any underlying MI algorithm. While other approaches like bagging are ensemble methods that require training multiple classifiers, SMILe can be implemented by simply adding shuffled bags to the existing dataset and training the base MI algorithm on the augmented sample. Thus, SMILe affords the advantages described above without requiring additional storage costs associated with ensemble methods. Furthermore, the interpretability of the base classifier is not affected as in ensemble methods. We also show below how for some specific MI base learners, SMILe can be incorporated implicitly into the algorithm without requiring that additional shuffled bags be explicitly added to the dataset. Thus, at least in some cases, the resampling of SMILe can be introduced to an algorithm without increasing the problem size.

8.3 Basic Properties of SMILe

In this section, we establish some basic properties of SMILe. First, we describe how the shuffled bag size s affects the noise rate on shuffled bag labels. Then, we describe how the addition of shuffled bags can decrease the bag label noise rate in an MI dataset.

We start by proving several results about the relationship between the shuffled bag size, s, and the bag label noise rate on the positive shuffled bags S_p (the notation used throughout this section is summarized in Table 8.1). We provide a worst-case analysis,

assuming only that there is at least one positive instance per positive bag. In the case that the fraction of positive instances within positive bags is known, a tighter bound can be similarly derived. When there is at least one positive instance per positive bag, there are at least $|B_p|$ true positive instances within X_p. The remaining $|X_p| - |B_p|$ instances within X_p are negative. Thus, in order to sample only negative instances in a shuffled bag, one of the $|X_p| - |B_p|$ instances must be sampled for each of the s instances within the bag. To simplify the analysis, we consider the probability of this occurring with replacement, which is an upper bound on the probability without replacement. With replacement, the probability of sampling a negative instance s times is at most $\left(\left(|X_p| - |B_p|\right) / |X_p|\right)^s$. Therefore, the probability of a sampling a negative shuffled bag decreases exponentially with the shuffled bag size. We formalize this using the following lemma (whose proof is straightforward):

Lemma 8.1. *For all $x \geq 0$, $\frac{x}{1+x} \leq \log(1+x)$.*

Proposition 8.1. *Suppose a set of shuffled bags is generated from an MI dataset without label noise on the original bags. Then if each shuffled bag is of size $s \geq \frac{|X_p|}{|B_p|} \log \frac{1}{\epsilon_s}$, each resulting bag will be positive with probability $1 - \epsilon_s$.*

Proof. To bound the true label noise on shuffled positive bags, it is sufficient to bound $\left(\left(|X_p| - |B_p|\right) / |X_p|\right)^s$ by the desired noise level, ϵ_s. This requirement implies that:

$$\begin{aligned}
\epsilon_s &\geq \left(\left(|X_p| - |B_p|\right) / |X_p|\right)^s \\
\log \epsilon_s &\geq s \log \left(\left(|X_p| - |B_p|\right)/|X_p|\right) \\
\log \tfrac{1}{\epsilon_s} &\leq s \log \left(|X_p|/\left(|X_p| - |B_p|\right)\right) \\
s &\geq \tfrac{1}{C} \log \tfrac{1}{\epsilon_s},
\end{aligned}$$

where $C = \log \left(|X_p|/\left(|X_p| - |B_p|\right)\right) = \log \left(1 + \left(|B_p|/\left(|X_p| - |B_p|\right)\right)\right)$. By Lemma 8.1,

$$C = \log \left(1 + \left(|B_p|/\left(|X_p| - |B_p|\right)\right)\right)$$

$$\geq \frac{|B_p|}{|X_p|-|B_p|} \Big/ \left(1 + \frac{|B_p|}{|X_p|-|B_p|}\right)$$
$$= \frac{|B_p|}{|X_p|-|B_p|} \Big/ \frac{|X_p|}{|X_p|-|B_p|} = \frac{|B_p|}{|X_p|}.$$

Thus, $s \geq \frac{|X_p|}{|B_p|} \log \frac{1}{\epsilon_s} \geq \frac{1}{C} \log \frac{1}{\epsilon_s}$ is a sufficient shuffled bag size. □

In many cases, all bags in an MI dataset have the same size. For example, the CBIR data shown in Figure 8.1 is generated such that bags contain roughly 30 instances. In this case, it is more straightforward to express the bound on the noise rate in terms of the bag size, b:

Corollary 8.1. *Suppose a set of shuffled bags is generated from an MI dataset without label noise on the original bags, which are each of size b. Then if each shuffled bag is of size $s \geq b \log \frac{1}{\epsilon_s}$, each resulting bag will be positive with probability $1 - \epsilon_s$.*

Proof. If a dataset contains $|B_p|$ positive bags, each of size b, then $|X_p| = b\,|B_p|$. Substituting this quantity into the coefficient of the bound in Proposition 8.1 gives: $s \geq \frac{|X_p|}{|B_p|} \log \frac{1}{\epsilon_s} = \frac{b|B_p|}{|B_p|} \log \frac{1}{\epsilon_s} = b \log \frac{1}{\epsilon_s}$. □

Thus, to achieve a worst-case noise rate of 10% on shuffled bags for the CBIR example with $b \approx 30$, it suffices to sample bags of size $s \geq 68$.

The bound given in Proposition 8.1 assumes a worst-case scenario in which bags are maximally "sparse," in that they contain only one positive instance. However, in some cases, the true sparsity of positive instances might be given as prior knowledge, or could be inferred from data Bunescu and Mooney (2007). In this case, we could express the bound of Proposition 8.1 in a relaxed form:

Corollary 8.2. *Suppose a set of shuffled bags is generated from an MI dataset without label noise on the original bags. Further, suppose it is known that η fraction of the instances X_p in positive bags are true positive instances. Then if each shuffled bag is of size $s \geq \frac{1}{\eta} \log \frac{1}{\epsilon_s}$, each resulting bag will be positive with probability $1 - \epsilon_s$.*

Proof. The proof proceeds as in Proposition 8.1, with the number of true positive instances equal to $\eta\,|X_p|$ rather than the worst case, $|B_p|$. □

An advantage of SMILe that applies to both instance- and bag-labeling algorithms is noise reduction. When there is existing positive bag label noise in an MI dataset, the addition of appropriately-sized shuffled bags can produce an overall reduction in noise.

Proposition 8.2. *Suppose an MI dataset B contains bags with a false-positive label noise rate of $\epsilon_+ > 0$. By choosing $s > \frac{|X_p|}{(1-\epsilon_+)|B_p|} \log \frac{1}{\epsilon_+}$, the expected false-positive noise rate on the resulting dataset $B_p \cup S_p$ will be $\widehat{\epsilon} < \epsilon_+$, where S_p is the set of positive shuffled bags added to the dataset.*

Proof. Given a false-positive noise rate of $\epsilon_+ > 0$, the worst case number of positive instances in positive bags is reduced from $|B_p|$ to $(1 - \epsilon_+)\,|B_p|$. Following as in the proof of Proposition 8.1, for an arbitrary desired ϵ_s error rate on the labels of shuffled bags, we can choose s, the size of a shuffled bag, to satisfy $s \geq \frac{|X_p|}{(1-\epsilon_+)|B_p|} \log \frac{1}{\epsilon_s}$. The resulting dataset has $|B_p|$ bags with noise ϵ_+, and $|S_p|$ bags with noise ϵ_s, for an expected false-positive noise rate of:

$$\widehat{\epsilon} = \frac{\epsilon_+\,|B_p| + \epsilon_s\,|S_p|}{|B_p| + |S_p|}.$$

If we choose ϵ_s small enough so that $\epsilon_s < \epsilon_+$, this is sufficient to have $\widehat{\epsilon} < \epsilon_+$ when shuffled bags are added. Therefore, it is sufficient that s is chosen such that $s > \frac{|X_p|}{(1-\epsilon_+)|B_p|} \log \frac{1}{\epsilon_+}$. □

Note that the advantage described in Proposition 8.2 does not hold for false-negative bag label noise. That is, when there are incorrectly labeled negative bags containing positive instances, it is possible that *most* of the negative shuffled bags are noisy. For example, if there is a single highly corrupted negative bag containing many

positive instances, then resampling instances from negative bags might be likely to include a corrupting positive instance. For this reason, if we expect negative bags to be noisy for some task, shuffling should not be applied to those bags ($|S_n|$ should be set to zero in the pseudocode of Algorithm 1) unless the base learner is tolerant to negative shuffled bag label noise. We describe such an algorithm in Section 8.5. Shuffling can still be applied to positive bags in any case.

As above, the following corollaries can be derived from Proposition 8.2 in the case when all bags have the same size b or there is a known fraction η of positive instances in positive bags.

Corollary 8.3. *Suppose an MI dataset B contains bags of size b with a false-positive label noise rate of $\epsilon_+ > 0$. By choosing $s > \frac{b}{(1-\epsilon_+)} \log \frac{1}{\epsilon_+}$, the expected false-positive noise rate on the resulting dataset $B_p \cup S_p$ will be $\widehat{\epsilon} < \epsilon_+$, where S_p is the set of positive shuffled bags added to the dataset.*

Proof. If a dataset contains $|B_p|$ positive bags, each of size b, then $|X_p| = b\,|B_p|$. Substituting this quantity into the coefficient of the bound in Proposition 8.2 gives:

$$s > \frac{|X_p|}{(1-\epsilon_+)|B_p|} \log \frac{1}{\epsilon_+} = \frac{b|B_p|}{(1-\epsilon_+)|B_p|} \log \frac{1}{\epsilon_+} = \frac{b}{(1-\epsilon_+)} \log \frac{1}{\epsilon_+}.$$

□

Corollary 8.4. *Suppose an MI dataset B contains bags with a false-positive label noise rate of $\epsilon_+ > 0$. Further, suppose it is known that η fraction of the instances X_p in true positive bags are true positive instances. By choosing $s > \frac{1}{(1-\epsilon_+)\eta} \log \frac{1}{\epsilon_+}$, the expected false-positive noise rate on the resulting dataset $B_p \cup S_p$ will be $\widehat{\epsilon} < \epsilon_+$, where S_p is the set of positive shuffled bags added to the dataset.*

Proof. Given random noise on positive bags, the number of instances in true positive bags is equal to $|X_p|\,(1-\epsilon_+)$. By assumption, the number of true positive instances is

then equal to $\eta |X_p| (1 - \epsilon_+)$. The proof then proceeds as in Proposition 8.2, with the number of true positive instances equal to $\eta |X_p| (1 - \epsilon_+)$ rather than $|B_p| (1 - \epsilon_+)$. □

Next, we review some related resampling approaches for MI learning and some possible alternatives to SMILe. Then, we discuss the effects of SMILe on the instance- and bag-level distributions in the training set, as well as the behavior of SMILe when combined with specific MI instance- and bag-level hyperplane classification algorithms.

8.4 Related Approaches

In contrast to the boosting and bagging approaches described in Section 8.1, SMILe is specific to the MI setting because it resamples instances from different bags to create new bags that are not in the original training set. By creating new examples, SMILe is most similar to an extension of prior work in the supervised setting that investigates how "virtual examples" can be added a training set by applying known class-preserving transformations such as rotations or translations to existing examples to improve accuracy on the test set (Schölkopf et al., 1997). In the case of SMILe, bag resampling is a transformation that approximately preserves class labels. Similar approaches transform examples by adding noise, which provides a form of regularization to a classifier, reducing the variance associated with the hypothesis class (Bishop, 1995). The motivation for SMILe is similar; by adding instance- and bag-level constraints in the form of new examples, SMILe also regularizes the hypothesis space and reduces variance.

Recent work in the supervised learning setting has explored how the virtual and noise-corrupted examples can be implicitly added to a training set by marginalizing out the distribution over virtual examples (Maaten et al., 2013). The resulting formulation essentially attempts to minimize loss over an infinite number of virtual

Table 8.2: Comparison of SMILe to other approaches that recombine instances from different bags. Only SMILe has the desired properties with small shuffled bags.

Resampling Approach	Avg. Shuffled Bag Size	Adds Constraints?	Reduces Bag Label Noise?
SMILe	$s = \frac{\lvert X_p \rvert}{\lvert B_p \rvert} \log \frac{1}{\epsilon_s}$	✓	✓
(1) Resample *all* instances	$\frac{\lvert X \rvert}{\lvert X_p \rvert} s$	✓	✓
(2) Combine pairs of positive bags	$2\frac{\lvert X_p \rvert}{\lvert B_p \rvert}$		✓
(3) Combine pairs of positive and negative bags	$2\frac{\lvert X \rvert}{\lvert B \rvert}$		

training examples. Experimental results show that implicitly providing an infinite number of virtual examples substantially increases generalization of the resulting classifiers. Furthermore, the procedure is computationally more efficient than training with explicitly added virtual examples. Below, we explore how ideas from this work might be applied to the new training examples (bags) produced by SMILe.

In formulating a novel approach such as SMILe for constructing new bags from existing training data, one might wonder whether there are alternative approaches worth considering. For example, there are three immediate alternatives to SMILe for generating new positive bags. Instead of sampling instances in positive shuffled bags only from X_p, instances can be resampled from the set of instances in *all* bags, X (1). Or, the instances from random pairs of positive bags can be combined to produce a new positive bag (2). Finally, a random positive bag can be combined with a random negative bag to produce a new positive bag (3). As summarized in Table 8.2, these alternative each have some disadvantage compared with SMILe. In the case of (1), resampling from all instances requires the shuffled bags to be larger by a factor of $(|X|/|X_p|)$ for a desired noise rate ϵ_s. Bags generated from approaches (2) and (3) do not introduce new constraints to a classifier. That is, if a classifier has zero loss on an initial set of bags, then it will also have zero loss on the bags of type (2) and (3) generate from the initial set. Finally, approach (3) has the additional disadvantage

that it does not reduce bag label noise for positive bags in the initial dataset. Thus, of these alternatives, only SMILe has the desired properties with a minimal shuffled bag size.

8.5 Instance-Level Classification with SMILe

As described above, an advantage of SMILe is that it introduces additional examples to an MI classification algorithm. As in prior work in the supervised setting (Bishop, 1995; Schölkopf et al., 1997), the addition of these new examples generated by SMILe should regularize hypotheses by reducing the set of consistent hypotheses. First, we analyze how SMILe generally affects the distribution over instances in a training set. Then, because it is difficult to analyze how SMILe regularizes arbitrary classifiers, we perform an analysis below with respect to the MI-SVM$_{\text{I}}$ algorithm (Han et al., 2010), a variant of the widely used MI-SVM approach (Andrews et al., 2003).

8.5.1 Effect on the Instance-Level Distribution

As described in Chapter 4, the learnability of instance concepts from MI data uses the notion of an instance-level distribution, defined in Chapter 3 that is obtained by marginalizing out individual bag distributions. Learnability requires that the instance-level distribution is identical for training and testing data. Since SMILe is intended to act as a wrapper around existing MI algorithms, it should leave the instance-level distribution invariant to ensure the learnability of instance-level concepts.

Figure 3.1 illustrates how the distribution over training instances in practice differs somewhat from that defined in Chapter 3. In particular, because multiple instances are sampled within each bag, instances might be more "correlated" than if only a single instance were sampled within each independently sampled bag. However, as

we observe in Section 5.5, this correlation seems to have little effect in practice. Therefore, we can think of the set of training instances as a good approximation of a sample from the underlying instance-level distribution.

Therefore, we desire that SMILe does not modify the distribution of instances in the training set. In fact, note that for any value of shuffled bag size, s, the addition of shuffled bags does not affect the distribution of in the dataset. Since instances within positive and negative shuffled bags are drawn independently and uniformly from the empirical class-conditional distributions, the distributions of instances in the shuffled bags also conform to the class-conditional distributions over instances. As long as the loss on the positive and negative bags in the dataset is balanced (e.g., using weights based on the class proportions in the original training sample) according to the original proportions of instances from each class, the overall instance distribution is maintained. Thus, for the sake of learning an instance-labeling concept, SMILe preserves the training distribution.

8.5.2 SMILe and MI-SVM$_\mathrm{I}$

We now analyze how SMILe affects the performance of a specific MI algorithm, MI-SVM$_\mathrm{I}$ (Han et al., 2010). In particular, we analyze how the loss function of MI-SVM$_\mathrm{I}$ is affected by the addition of shuffled bags, which provides insights into how parameters such as shuffled bag size affect the hyperplane classifier that is learned. Then, in an analysis similar to prior work in the supervised setting (Maaten et al., 2013), we describe a new algorithm, SMILeSVM, which directly optimizes the expected loss incurred by shuffled bags. We show empirically that SMILeSVM has practical advantages over explicitly adding shuffled bags and produces better performance on the test set.

The original MI-SVM formulation (Andrews et al., 2003) works by solving a non-convex quadratic program (QP), as shown in Equation 2.4. Prior work observed that

when using MI-SVM to label instances, using a max constraint to derive the label of negative bags tends to ignore labels assigned by the classifier f to many of the instances known to be negative (Han et al., 2010). The resulting classifier might therefore have a high false positive error rate. Thus, a variant of MI-SVM, which we denote MI-SVM$_\text{I}$, explicitly incorporates instance-level constraints on instances in positive bags (Han et al., 2010). With the appropriate coefficients to balance the loss on positive and negative examples, the MI-SVM$_\text{I}$ optimization program is:

$$\min_{f\in\mathcal{H},\,\boldsymbol{\Xi}} \tfrac{1}{2}\left\|f\right\|_{\mathcal{H}}^2 + \tfrac{C}{|B|}\left(\tfrac{|B_n|}{|X_n|}\sum_{ij}\xi_{ij}^- + \tfrac{|B_p|}{|B_p|}\sum_{i}\Xi_i^+\right),$$

$$\text{s.t.} \quad \begin{cases} Y_i\left(\max_{x_{ij}\in B_i} f(x_{ij})\right) \geq 1-\Xi_i^+ & \text{if } Y_i = +1 \\ \forall x_{ij}\in B_i : Y_i f(x_{ij}) \geq 1-\xi_{ij}^- & \text{if } Y_i = -1 \end{cases}$$

$$\Xi_i^+ \geq 0,\, \xi_{ij}^- \geq 0,$$

where ξ_{ij}^- are negative bag instance-level slack variables, Ξ_i^+ are positive bag-level slack variables, and a hyperplane f is chosen from the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ associated with some kernel.

MI-SVM$_\text{I}$ enforces the MI assumption that in each negative bag, all instances must be negative. On the other hand, every positive bag in the dataset corresponds to a constraint of the SVM QP, where the max in each constraint enforces the MI relationship between bag and instance labels for positive bags. As written, it is clear that the addition of positive shuffled bags will introduce further constraints into the optimization program. However, the effect of shuffled bags is more apparent if the max constraint is lifted into the objective function. In fact, each constraint is equivalent to simply choosing the *minimum* instance-level slack variable ξ_{ij}^+ that would be required if instance-level constraints were also introduced to instances in positive bags.

To see why, first observe that for the slack variables Ξ_i^+ of positive bags:

$$\begin{aligned}\max_{x_{ij}\in B_i} f(x_{ij}) &\geq 1 - \Xi_i^+ \\ \Xi_i^+ &\geq 1 - \max_{x_{ij}\in B_i} f(x_{ij}) \\ \Xi_i^+ &\geq \min_{x_{ij}\in B_i} \left(1 - f(x_{ij})\right).\end{aligned}$$

Since $\Xi_i^+ \geq 0$, and the objective function penalizes Ξ_i^+, the objective function is minimized when $\Xi_i^+ = \min_{x_{ij}\in B_i} \max\{0, 1 - f(x_{ij})\}$. For a supervised SVM, the slack variable for a positive *instance* is $\xi_{ij}^+ = \max\{0, 1 - f(x_{ij})\}$. Thus, in the MI-SVM$_\text{I}$ formulation, the slack variable for a positive bag is $\Xi_i^+ = \min_{x_{ij}\in B_i} \xi_{ij}^+$.

Given that we can rewrite positive bag constraints as described above, the MI-SVM$_\text{I}$ formulation can be written as:

$$\min_{f\in\mathcal{H},\,\boldsymbol{\xi}} \tfrac{1}{2}\left\|f\right\|_{\mathcal{H}}^2 + \tfrac{C}{|B|}\left(|B_n|\overbrace{\tfrac{1}{|X_n|}\sum_{x_{ij}\in X_n}\xi_{ij}^-}^{\text{Negative Bag Loss}} + |B_p|\overbrace{\tfrac{1}{|B_p|}\sum_{B_i\in B_p}\min_{x_{ij}\in B_i}\xi_{ij}^+}^{\text{Positive Bag Loss}}\right), \tag{8.1}$$

$$\text{s.t.}\quad Y_i f(x_{ij}) \geq 1 - \xi_{ij},\ \xi_{ij} \geq 0.$$

The reformulation above allows us to easily analyze how the addition of positive and negative shuffled bags affects the loss terms corresponding to positive and negative bags as decomposed in Equation 8.1. In particular, as written above, it becomes clear that $\frac{1}{|X_n|}\sum_{x_{ij}\in X_n}\xi_{ij}^-$ corresponds to the average or expected loss incurred by an instance in a negative bag, and $\frac{1}{|B_p|}\sum_{B_i\in B_p}\min_{x_{ij}\in B_i}\xi_{ij}^+$ is the expected value of the *minimum* slack within a positive bag.

Now, we can consider the consequences when these expected values are taken not with respect to the original training sample, but with respect to the generative process corresponding to SMILe in which bags are generated by sampling instances uniformly from X_n or X_p. Since there are only a finite number of possible shuffled bags of a

fixed size, all occurring with an equal probability, the expected loss is equivalent to the average loss over all possible shuffled bags. As with the analysis in Section 8.3, we make the simplifying assumption that sampling is performed *with* replacement. The assumption becomes more aligned with the process described in Algorithm 1 as the training set size increases. With this assumption, we use $\mathcal{S}_s^-$ and $\mathcal{S}_s^+$ to refer to the distribution over negative or positive shuffled bags of size s corresponding to the SMILe generative process.

For negative shuffled bags, the analysis is straightforward. If we sample a set S_n of negative shuffled bags and combine all negative instances into a set $X'_n = \bigcup_{S_i \in S_n} S_i$, then every instance in X'_n is sampled uniformly from X_n. Because we assume that instances are sampled independently within all shuffled bags, the size of the shuffled negative bags is irrelevant. With respect to a fixed classifier f, the expected loss incurred by f on the shuffled negative bags is:

$$\begin{aligned}
\mathrm{E}_{S_n \sim \mathcal{S}_s^{-|S_n|}} \left[\frac{1}{|X'_n|} \sum_{x_{ij} \in X'_n} \xi_{ij}^- \right] &= \frac{1}{|X'_n|} \sum_{x_{ij} \in X'_n} \mathrm{E}_{x_{kl} \sim \mathrm{Uniform}(X_n)} \left[\xi_{kl}^- \right] \\
&= \frac{1}{|X'_n|} |X'_n| \, \mathrm{E}_{x_{kl} \sim \mathrm{Uniform}(X_n)} \left[\xi_{kl}^- \right] \\
&= \frac{1}{|X_n|} \sum_{x_{ij} \in X_n} \xi_{ij}^-, \qquad (8.2)
\end{aligned}$$

which is identical to the loss incurred by f on the original negative bags in the dataset. Thus, negative shuffled bags do not provide any additional constraint information for negative instances, as described intuitively above.

SMILe has a different effect on the expected loss incurred by a classifier f on positive shuffled bags. For a set of positive shuffled bags, S_p, we can write the expected loss as:

$$\mathrm{E}_{S_p \sim \mathcal{S}_s^{+|S_p|}} \left[\frac{1}{|S_p|} \sum_{B_i \in S_p} \min_{x_{ij} \in B_i} \xi_{ij}^+ \right] = \frac{1}{|S_p|} \sum_{i=1}^{|S_p|} \mathrm{E}_{S^+ \sim \mathcal{S}_s^+} \left[\min_{x_{kl} \in S^+} \xi_{kl}^+ \right]$$

$$= \mathrm{E}_{S^+ \sim \mathcal{S}_s^+} \left[\min_{x_{kl} \in S^+} \xi_{kl}^+ \right]. \tag{8.3}$$

Thus, it suffices to analyze the expected loss incurred by f on a single positive shuffled bag S^+ of size s.

For notational convenience, let $\{\sigma_k\}_{k=1}^{|X_p|}$ be a classifier-specific re-indexing of the slack variables corresponding to all positive instances X_p in the original training set. For a fixed classifier f, the slack variables are ordered high-to-low so that for $1 \leq k < |X_p|$, $\sigma_k \geq \sigma_{k+1}$. Ties can be broken arbitrarily. Let $\mathrm{rank}_f(\xi_{ij}^+)$ be the high-to-low order of the slack variable ξ_{ij}^+ so that $\xi_{ij}^+ = \sigma_{\mathrm{rank}_f(\xi_{ij}^+)}$. The random variable $\min_{x_{ij} \in S^+} \xi_{ij}^+$ with $S^+ \sim \mathcal{S}_s^+$ takes one of the values of σ_k. The expected value is the sum of these σ_k weighted by the likelihood that the random variable takes the corresponding value:

$$\begin{aligned}
\mathrm{E}_{S^+ \sim \mathcal{S}_s^+} \left[\min_{x_{ij} \in S^+} \xi_{ij}^+ \right] &= \sum_{k=1}^{|X_p|} \mathrm{P}_{S^+ \sim \mathcal{S}_s^+} \left[\sigma_k = \min_{x_{ij} \in S^+} \xi_{ij}^+ \right] \sigma_k \\
&= \sum_{k=1}^{|X_p|} \mathrm{P}_{S^+ \sim \mathcal{S}_s^+} \left[k = \mathrm{rank}_f \left(\min_{x_{ij} \in S^+} \xi_{ij}^+ \right) \right] \sigma_k \\
&= \sum_{k=1}^{|X_p|} \mathrm{P}_{S^+ \sim \mathcal{S}_s^+} \left[k = \max_{x_{ij} \in S^+} \mathrm{rank}_f(\xi_{ij}^+) \right] \sigma_k.
\end{aligned}$$

The final step is justified by observing that since rank_f is a high-to-low (by magnitude) ranking of the slack variables, the smallest slack variable value has the highest value assigned by rank_f. In this form, we can rewrite the expression for the probability in the following way:

$$\mathrm{P}_{S^+ \sim \mathcal{S}_s^+} \left[k = \max_{x_{ij} \in S^+} \mathrm{rank}_f(\xi_{ij}^+) \right] = \mathrm{P}_{S^+ \sim \mathcal{S}_s^+} \left[\frac{k}{|X_p|} = \max_{x_{ij} \in S^+} \frac{\mathrm{rank}_f(\xi_{ij}^+)}{|X_p|} \right].$$

Since each $x_{ij} \in S^+$ is independently sampled (assuming sampling with replacement for simplicity), we can consider the behavior of the individual random variable

$\left(\text{rank}_f(\xi_{ij}^+)/|X_p|\right)$. Because x_{ij} is sampled uniformly from X_p, ξ_{ij}^+ is sampled uniformly from the set of all slacks on positive instances. Accordingly, the rank of the slack variable $\text{rank}_f(\xi_{ij}^+)$ is sampled according to a uniform distribution over $\{1, 2, \ldots, k\}$. This means that $\left(\text{rank}_f(\xi_{ij}^+)/|X_p|\right) \sim \text{Uniform}\left(\left\{\frac{1}{|X_p|}, \frac{2}{|X_p|}, \ldots, 1\right\}\right)$. We can make an approximation here by observing that as $|X_p| \to \infty$,

$$\text{Uniform}\left(\left\{\tfrac{1}{|X_p|}, \tfrac{2}{|X_p|}, \ldots, 1\right\}\right) \underset{\text{weakly}}{\Rightarrow} \text{Uniform}\left([0,1]\right).$$

So for large numbers of instances in positive bags, it follows that:

$$\begin{aligned}
\frac{\text{rank}_f(\xi_{ij}^+)}{|X_p|} &\underset{\text{approx.}}{\sim} \text{Uniform}\left([0,1]\right) \\
\max_{x_{ij} \in S^+} \frac{\text{rank}_f(\xi_{ij}^+)}{|X_p|} &\underset{\text{approx}}{\sim} \max_{x_{ij} \in S^+} \text{Uniform}\left([0,1]\right) \\
&\sim \text{Beta}(s, 1).
\end{aligned}$$

The last step uses the fact that the maximum of a sample of $|S^+| = s$ uniform random variables over $[0, 1]$ is distributed according to the beta distribution whose unnormalized likelihood function is $d(r) = r^{s-1}$ (Gentle, 2009). Figure 8.2 shows that even with relatively few instances in positive bags, the quality of the beta distribution approximation above falls within 1% of the true distribution in terms of assigning probability mass to slack variables.

Now, we can return to the original task of calculating the expected loss incurred by a classifier f on a shuffled bag. Recall that the expected loss is the sum of each positive instance slack value σ_k, weighted by the likelihood that the minimum slack across all instances in the shuffled positive bag is σ_k. Given the analysis above, we can approximate the likelihood that the random variable $\left(\text{rank}_f(\xi_{ij}^+)/|X_p|\right)$ is $k/|X_p|$ using the beta likelihood $d\left(k/|X_p|\right) = \left(k/|X_p|\right)^{s-1}$. Equivalently, we can approximate the likelihood that the random variable $\text{rank}_f(\xi_{ij}^+)$ is k by $d(k) = k^{s-1}$. A proper

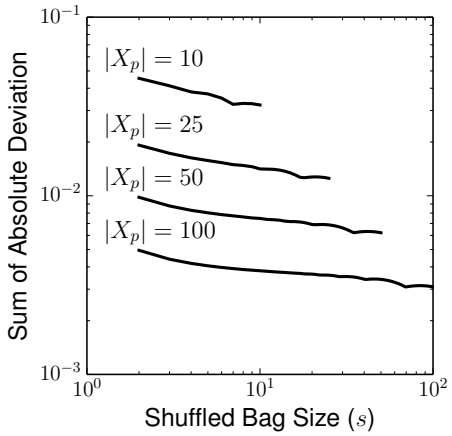


Figure 8.2: An illustration of the quality of the beta distribution approximation for different numbers of instances in positive bags and shuffled bag sizes. The approximation quality is measured as the sum of absolute differences between the true density function values over slack variables and those assigned by the beta distribution.

probability density function is obtained by normalizing d to yield $p(k) = \frac{1}{K}k^{s-1}$, where $K = \sum_{k=1}^{|X_p|} k^{s-1}$. The resulting expression for expected loss is then:

$$\begin{aligned}\mathrm{E}_{S^+\sim\mathcal{S}_s^+}\left[\min_{x_{ij}\in S^+}\xi_{ij}^+\right] &= \sum_{k=1}^{|X_p|}\mathrm{P}_{S^+\sim\mathcal{S}_s^+}\left[k = \max_{x_{ij}\in S^+}\mathrm{rank}_f(\xi_{ij}^+)\right]\sigma_k \\ &\approx \frac{1}{K}\sum_{k=1}^{|X_p|}k^{s-1}\sigma_k.\end{aligned}$$

As a final step in deriving the expected loss, we wish to rewrite the expression above so that it is stated in terms of the original slack variables ξ_{ij}^+ rather than σ_k. For this purpose, we use the rank_f notation, observing that by definition, $k = \mathrm{rank}_f(\sigma_k)$. Therefore,

$$\begin{aligned}\mathrm{E}_{S^+\sim\mathcal{S}_s^+}\left[\min_{x_{ij}\in S^+}\xi_{ij}^+\right] &\approx \frac{1}{K}\sum_{k=1}^{|X_p|}\mathrm{rank}_f(\sigma_k)^{s-1}\sigma_k \\ &= \frac{1}{K}\sum_{x_{ij}\in X_p}\mathrm{rank}_f(\xi_{ij}^+)^{s-1}\xi_{ij}^+.\end{aligned}$$

The last equality follows because the order of the variables σ_k are no longer important; the index k only serves to couple a slack variable with its rank. So given the bijection between $\{\sigma_k\}$ and $\{\xi_{ij}^+\}$, we can rewrite the expression using ξ_{ij}^+. Using the equivalence from Equation 8.3, the expected loss incurred by f on a set S_p of positive shuffled

bags is:

$$\mathrm{E}_{S_p \sim \mathcal{S}_s^{+|S_p|}} \left[\frac{1}{|S_p|} \sum_{B_i \in S_p} \min_{x_{ij} \in B_i} \xi_{ij}^+ \right] \approx \frac{1}{K} \sum_{x_{ij} \in X_p} \mathrm{rank}_f(\xi_{ij}^+)^{s-1} \xi_{ij}^+. \tag{8.4}$$

Given the expression above for the expected loss incurred by f on positive shuffled bags, we can analyze how the addition of these shuffled bags might affect the selected optimal classifier. The original MI-SVM$_\mathrm{I}$ formulation in Equation 8.1 only uses a single instance slack variable from each positive bag in the objective function to represent the loss of f incurred on each positive bag. Thus, for any fixed f, only a possibly small fraction $\frac{|B_p|}{|X_p|}$ of the positive instances are used to evaluate the classifier. On the other hand, the expected loss incurred by adding shuffled positive bags is an expression that involves a weighted combination of all positive instance slack variables.

The shuffled bag size s affects how the positive instance slack variables are weighted in the loss expression. When $s = 1$, the expected loss is:

$$\mathrm{E}_{S_p \sim \mathcal{S}_1^{+|S_p|}} \left[\frac{1}{|S_p|} \sum_{B_i \in S_p} \min_{x_{ij} \in B_i} \xi_{ij}^+ \right] \approx \frac{1}{K} \sum_{x_{ij} \in X_p} \mathrm{rank}_f(\xi_{ij}^+)^0 \xi_{ij}^+ = \frac{1}{|X_p|} \sum_{x_{ij} \in X_p} \xi_{ij}^+,$$

which is just the average loss of f on all positive bag instances (assumed to all have positive labels). Thus, the case $s = 1$ corresponds to the addition of "strong" instance-level constraints that include each instance equally in evaluating loss. In fact, the expected loss on positive bags with $s = 1$ is symmetric to the expected loss of negative shuffled bags given in Equation 8.2, which are both similar to the losses used single-instance learning (SIL), the supervised approach in which all instances in X_n and X_p are treated as negative and positive instances, respectively. Since not all instances in X_p are positive, it might seem that adding such positive shuffled "singleton" bags introduces too much label noise or completely destroys the +information provided by the structure within bags. However, as described in Chapter 4, such an approach

actually learn high-AUC concepts.

In the other extreme case, when $s \to \infty$, the majority of the weight is focused on the smallest positive instance slack variable ξ^+_{ij} for which $|X_p| = \text{rank}_f(\xi^+_{ij})$. If there are multiple instances with zero slack at any point during optimization, meaning that the instances are classified as positive by the current hyperplane, then the effect of the regularization will be to maximize the margin and allow some positive bag instances to fall within the margin. As a result, the slack values of these instances will increase. However, for all but one instance, the increase in slack comes without consequence. Therefore, at an optimal solution, only a single positive bag instance in the dataset is used to evaluate the quality of f, unless multiple instances happen to lie directly along the boundary of the margin. Thus, when s is large, the noise rate on the shuffled bags is small, but the shuffled bags provide little additional information to the classifier.

For other intermediate values of s, we can determine what portion of the mass in the weighted sum is assigned to the fraction t of the most-misclassified instances. These are the instances with the largest slack variables, which are $\{\sigma_1, \ldots, \sigma_{t|X_p|}\}$ in the re-indexed set that orders slack values from high to low. As above, when we assume a large sample, the weights on these instances' slacks are distributed according to Beta$(s, 1)$ with probability density function $p(r) = sr^{s-1}$, where r is the scaled rank of a slack variable ($r = k/|X_p|$ for σ_k). Therefore, the total mass on the fraction t of instances with the largest slack variables is $\int_0^t sr^{s-1}\mathrm{d}r = t^s$. If the classifier produces a correct labeling of instances, then at most $(|X_p| - |B_p|)/|X_p|$ fraction of the instances in X_p are actually negative and will have large slack values. Therefore, $((|X_p| - |B_p|)/|X_p|)^s$ fraction of the loss is wrongly incurred due to these instances. The amount of incorrectly attributed loss decreases with s, and corresponds precisely to the worst-case noise rate as given in Proposition 8.1.

In summary, deriving the expected loss on shuffled bags as in Equation 8.4 provides

insight into the effect of shuffled bags on the MI-SVM$_\text{I}$ optimization program. In particular, the effect of the shuffled bag size parameter s becomes more apparent. When $s = 1$, SMILe reduces to training a supervised classifier on the training set of bag-labeled instances. On the other hand, when $s \to \infty$, the classifier will tend to ignore all but one positive bag instance. For intermediate values, the classifier will focus on correctly classifying a subset of the instances in the positive bags. Of the remaining examples, in the worst case the classifier will place $((|X_p| - |B_p|)/|X_p|)^s$ fraction of the focus on the true negative instances in positive bags, which is consistent with the worst-case noise rate of Proposition 8.1.

8.5.3 SMILeSVM

In addition to allowing us to make observations about how shuffled bags affect the objective function of the MI-SVM$_\text{I}$ classifier, the analysis in Section 8.5.2 also suggests a way of modifying the MI-SVM$_\text{I}$ optimization program to avoid explicitly adding shuffled bags to the dataset. In particular, suppose we add a very large number of negative and positive shuffled bags to the training set in proportion to the original class ratios. In the limit of infinitely many shuffled bags, the loss function Equation 8.1 will be dominated by the expected loss on negative and positive shuffled bags. Thus, the behavior of MI-SVM$_\text{I}$ with many shuffled bags can be approximated by substituting the expression in Equation 8.2 for negative bag loss and Equation 8.4 for positive bag loss. The result is:

$$\min_{f \in \mathcal{H}, \boldsymbol{\xi}} \tfrac{1}{2} \|f\|_{\mathcal{H}}^2 + \tfrac{C}{|B|} \left(|B_n| \overbrace{\tfrac{1}{|X_n|} \sum_{x_{ij} \in X_n} \xi_{ij}^-}^{\text{Negative Bag Loss}} + |B_p| \overbrace{\frac{1}{K} \sum_{x_{ij} \in X_p} \text{rank}_f(\xi_{ij}^+)^{s-1} \xi_{ij}^+}^{\text{Positive Bag Loss}} \right),$$

$$\text{s.t.} \quad Y_i f(x_{ij}) \geq 1 - \xi_{ij},\ \xi_{ij} \geq 0.$$

Unfortunately, the formulation as written above is difficult to optimize, since

$\text{rank}_f(\xi_{ij}^+)$ is a discontinuous function of f and ξ_{ij}^+. Therefore, we must rewrite the objective function, particularly the expression in Equation 8.4, such that the optimization becomes tractable. To this end, we introduce some additional notation. Let $\boldsymbol{\xi}^+$ be a vector of the positive instance slack variables (in any order). Let $\boldsymbol{\rho}$ be an arbitrarily ordered vector of weights corresponding to the weights in the sum of Equation 8.4; that is, $\boldsymbol{\rho}_k = k^{s-1}$ for $1 \leq k \leq |X_p|$. In order to match up the slack variables with their appropriate weights, we can use a classifier-specific permutation matrix, given by:

$$\mathbf{P}_{ij}^{(f)} = \begin{cases} 1 & \text{if } \text{rank}_f(\boldsymbol{\xi}_j^+) = i \\ 0 & \text{otherwise.} \end{cases}$$

Then, the expected loss in Equation 8.4 can be written succinctly as:

$$\mathrm{E}_{S_p \sim \mathcal{S}_s^{+|S_p|}} \left[\tfrac{1}{|S_p|} \sum_{B_i \in S_p} \min_{x_{ij} \in B_i} \xi_{ij}^+ \right] \approx \frac{1}{K} \sum_{x_{ij} \in X_p} \text{rank}_f(\xi_{ij}^+)^{s-1} \xi_{ij}^+ = \tfrac{1}{K} \boldsymbol{\rho}^\intercal \mathbf{P}^{(f)} \boldsymbol{\xi}^+. \qquad (8.5)$$

Of course, the appropriate permutation matrix $\mathbf{P}^{(f)}$ above is still a discontinuous function of the classifier f. However, now we will show that we can relax the expression in Equation 8.5 to be a minimization problem that can be absorbed into the overall SVM objective function. Then, optimization can be performed by iteratively fixing a permutation matrix, optimizing the remainder of the objective function, then minimizing with respect to the permutation matrix. The strategy is thus to relax the objective function by replacing $\mathbf{P}^{(f)}$ with an arbitrary permutation matrix, $\mathbf{P} \in \mathcal{P}$. Then, we use the following result:

Lemma 8.2. *For any classifier f and associated positive instance slack variables $\boldsymbol{\xi}^+$*

as defined in Equation 8.1, the following equality holds:

$$\min_{\mathbf{P}\in\mathcal{P}} \boldsymbol{\rho}^\intercal \mathbf{P}\boldsymbol{\xi}^+ = \boldsymbol{\rho}^\intercal \mathbf{P}^{(f)}\boldsymbol{\xi}^+,$$

where $\mathcal{P}$ is the set of permutation matrices of size $|X_p|$.

Proof. Although the left-hand side of the equation above is a discrete optimization over permutation matrices, it can be relaxed to a convex linear program (LP). This fact follows from the Birkhoff–von Neumann theorem (Birkhoff, 1946; von Neumann, 1953), which states that the doubly-stochastic matrices (matrices whose rows and columns each sum to one) are the convex hull of permutation matrices. Thus, for a fixed $\boldsymbol{\rho}$ and $\boldsymbol{\xi}^+$, the left-hand side of the equation above has a global minimum that occurs at a vertex of the convex hull, which is a permutation matrix. Thus, we must simply show that $\mathbf{P}^{(f)}$ is a local minimizer of the LP; that is, that moving to any adjacent vertex of $\mathbf{P}^{(f)}$ increases the objective function on the left-hand side.

Any adjacent vertex of $\mathbf{P}^{(f)}$ can be obtained by swapping two elements in the permutation, say $(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_j$ and $(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_k$ with $j < k$, since $(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_j \geq (\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_k$ and $\boldsymbol{\rho}_j < \boldsymbol{\rho}_k$, we have that:

$$\begin{aligned}
\boldsymbol{\rho}_j \left((\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_j - (\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_k\right) &\leq \boldsymbol{\rho}_k \left((\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_j - (\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_k\right) \\
\boldsymbol{\rho}_j(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_j + \boldsymbol{\rho}_k(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_k &\leq \boldsymbol{\rho}_k(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_j + \boldsymbol{\rho}_j(\mathbf{P}^{(f)}\boldsymbol{\xi}^+)_k.
\end{aligned}$$

This means that $\mathbf{P}^{(f)}$ is a local minimizer of the expression, which means that it is a global minimum as stated in the Lemma. □

Lemma 8.2 allows us to rewrite the expression for the expected loss of shuffled bags in Equation 8.5 in a way that is independent of the current classifier f, but still

depends on the current slack variables $\boldsymbol{\xi}^+$:

$$\mathrm{E}_{S_p \sim \mathcal{S}_s^{+|S_p|}} \left[\frac{1}{|S_p|} \sum_{B_i \in S_p} \min_{x_{ij} \in B_i} \xi_{ij}^+ \right] \approx \frac{1}{\mathbf{1}^\intercal \boldsymbol{\rho}} \min_{\mathbf{P} \in \mathcal{P}} \boldsymbol{\rho}^\intercal \mathbf{P} \boldsymbol{\xi}^+. \tag{8.6}$$

The normalization constant K is rewritten above as $K = \sum_{k=1}^{|X_p|} k^{s-1} = \mathbf{1}^\intercal \boldsymbol{\rho}$, where $\mathbf{1}$ is a vector of ones. For negative shuffled bags, let $\boldsymbol{\xi}^-$ be a vector of the negative bag instance slack variables (in any order). Then we can express Equation 8.2 succinctly as:

$$\mathrm{E}_{S_n \sim \mathcal{S}_s^{-|S_n|}} \left[\frac{1}{|X_n'|} \sum_{x_{ij} \in X_n'} \xi_{ij}^- \right] = \frac{1}{|X_n|} \mathbf{1}^\intercal \boldsymbol{\xi}^-. \tag{8.7}$$

Substituting the expressions in Equation 8.5 for negative bag loss and Equation 8.6 for positive bag loss into Equation 8.1 yields the following formulation:

$$\begin{aligned} \min_{f \in \mathcal{H}, \boldsymbol{\xi}, \mathbf{P} \in \mathcal{P}} \; & \tfrac{1}{2} \|f\|_{\mathcal{H}}^2 + \frac{C}{|B|} \left(\frac{|B_n|}{|X_n|} \mathbf{1}^\intercal \boldsymbol{\xi}^- + \frac{|B_p|}{\mathbf{1}^\intercal \boldsymbol{\rho}} \boldsymbol{\rho}^\intercal \mathbf{P} \boldsymbol{\xi}^+ \right), \\ \text{s.t.} \quad & Y_i f(x_{ij}) \geq 1 - \xi_{ij}, \, \xi_{ij} \geq 0. \end{aligned} \tag{8.8}$$

We refer to the formulation in Equation 8.8 as SMILeSVM.

Since, as for MI-SVM$_\mathrm{I}$, the SMILeSVM optimization program is nonconvex, we use an iterative approach to solve it in practice. First, $\mathbf{P}$ is initialized to $\frac{1}{|X_p|} \mathbf{1}\mathbf{1}^\intercal$, the "average" permutation matrix that places an equal weight on each slack variable, and the other variables are used to minimize the resulting standard SVM formulation that learns from weighted instances. Then, with the other variables fixed, the objective function is optimized with respect to $\mathbf{P}$, which amounts to permuting the vector $\boldsymbol{\rho}$ so that the weights appear in the order corresponding to the values of $\boldsymbol{\xi}^+$. The alternating process repeats until $\mathbf{P}$ is unchanged, or a maximum number of iterations has been exceeded. The procedure is illustrated in Algorithm 2.

Algorithm 2

Require: MI dataset (B, Y), shuffled bag size s, regularization–loss trade-off parameter C, maximum iterations `maxiters`

$(X, y) \leftarrow \{(x_{ij}, Y_i) \mid x_{ij} \in B_i, (B_i, Y_i) \in (B, Y)\}$
$X_n \leftarrow \{x_i \mid y_i = -1\}$
$X_p \leftarrow \{x_i \mid y_i = +1\}$
$\mathbf{1}_p \leftarrow [1 \mid 1 \leq k \leq |X_p|]^\intercal$
$\mathbf{1}_n \leftarrow [1 \mid 1 \leq k \leq |X_n|]^\intercal$
$\boldsymbol{\rho} \leftarrow [k^{s-1} \mid 1 \leq k \leq |X_p|]^\intercal$
$C_n \leftarrow C\frac{|B_n|}{|B||X_n|}\mathbf{1}_n$
$C_p \leftarrow C\frac{|B_p|}{|B|\mathbf{1}_p^\intercal\boldsymbol{\rho}}\boldsymbol{\rho}$
$\mathbf{P}_{\text{last}} \leftarrow \frac{1}{|X_p|}\mathbf{1}_p\mathbf{1}_p^\intercal$
for 1 to `maxiters` **do**
    $f \leftarrow \textsc{WeightedSVM}([X_p; X_n], [+\mathbf{1}_p; -\mathbf{1}_n], [C_n; C_p\mathbf{P}_{\text{last}}])$
    $\mathbf{P}_{ij} \leftarrow \begin{cases} 1 & \text{if } \operatorname{rank}(1 - f(X_p[j])) = i \\ 0 & \text{otherwise} \end{cases}$
    if $\mathbf{P} = \mathbf{P}_{\text{last}}$ **then**
        break
    end if
    $\mathbf{P}_{\text{last}} \leftarrow \mathbf{P}$
end for
return f

The primary advantage of SMILeSVM is that it directly optimizes the expected loss with respect to shuffled bags. Accordingly, additional shuffled bags do not need to be explicitly added to the MI dataset, which would unnecessarily increase the number of variables in the MI-SVM$_\text{I}$ QP. In our experiments, we show that directly solving the SMILeSVM formulation leads to significantly improved performance over MI-SVM$_\text{I}$ or what is achievable by explicitly adding practically feasible numbers of shuffled bags to the original dataset.

8.6 Bag-Level Classification with SMILe

Now we discuss the effect of SMILe on bag-level classification. Unlike for the distribution over instances, the distribution over bags changes under SMILe. We first

discuss the effect of SMILe on the bag-level distribution, then discuss how this affects the kernel-based bag-level normalized set kernel (NSK) classifier.

8.6.1 Effect on Bag-Level Distribution

Although we saw that SMILe does not affect the distribution of instances in a training set, it clearly affects the distribution over *bags*. For example, consider a generative process in which two instances x_1 and x_2 are almost surely never sampled within the same bag, but each is sampled with some nonzero probability in some set of separate positive bags. Then, with nonzero probability, the instances will occur within the same shuffled bag. However, since this shuffled bag almost surely never appeared in the original distribution $D_{\mathcal{B}}$ over bags, the distribution over bags must have changed with the addition of shuffled bags.

In fact, assuming a large training sample, we can characterize the expected distribution over shuffled bags. Since instances are sampled from positive bags to generate positive shuffled bags, the distribution over these instances is given by $\mathrm{P}(x) = \int_{\mathcal{B}_+} \mathrm{P}(x \mid B)\,\mathrm{d}\,\mathrm{P}(B)$, which we denote $D_{\mathcal{X}_+}$. Similarly, the distribution over instances in negative shuffled bags is $\mathrm{P}(x) = \int_{\mathcal{B}_-} \mathrm{P}(x \mid B)\,\mathrm{d}\,\mathrm{P}(B)$, denoted $D_{\mathcal{X}_-}$. We call $D_{\mathcal{X}_+}$ and $D_{\mathcal{X}_+}$ the class-conditional instance distributions. Therefore, if bags are viewed as distributions, then all positive (or negative) shuffled bags are identical to $D_{\mathcal{X}_+}$ (or $D_{\mathcal{X}_-}$). The bag samples themselves differ due to sampling variance, which depends on bag size. Therefore, the MI generative process with SMILe collapses $D_{\mathcal{B}}$ to a distribution over two bags, given by $D_{\mathcal{X}_+}$ and $D_{\mathcal{X}_+}$. The probability mass assigned to each of these two bags is given by the fraction of shuffled bags of each class added to the dataset.

8.6.2 SMILe and the NSK

Given our understanding of how SMILe affects the distribution over bags, we can analyze the behavior of the NSK when SMILe is applied to a training set. Recall that the NSK, discussed in Section 2.3.1, averages instances under some kernel feature map (see Figure 2.7). We can use kernel principal component analysis (KPCA) to visualize how the original and shuffled bags are embedding into the feature space using the NSK with a radial basis function (RBF) kernel (Schölkopf et al., 1998). Figure 8.3 shows such a plot for the "Checkered Scarf vs. Data Mining Book" dataset from the CBIR domain. The RBF is used as an instance kernel, with the median distance between instances used as a heuristic for the RBF scale parameter. The shuffled bags are generated using a size that guarantees a worst-case noise rate of $\epsilon_s = 10\%$. Blue circles are positive bags, and red squares are negative bags. A Gaussian distribution has been fitted to each bag-level class-conditional distribution. When we add a sampled set of positive and negative shuffled bags (the hollow points in Figure 8.3) to the training set, we see that these bags share the same mean as the class-conditional distribution, but not the same variance. By fitting two more Gaussian distributions to the shuffled positive and negative bags, we clearly see that the distribution over shuffled bags differs from that over the original bags in the training set.

We can more formally analyze the behavior of the NSK demonstrated in Figure 8.3. Since the NSK treats positive and negative bags symmetrically, we will only analyze the behavior of positive shuffled bags; the analysis for negative shuffled bags is identical. Let $\mu(X_p)$ be the mean embedding of the set of positive instances. When the shuffled bag size s is sufficiently large, the mean embedding of a positive shuffled

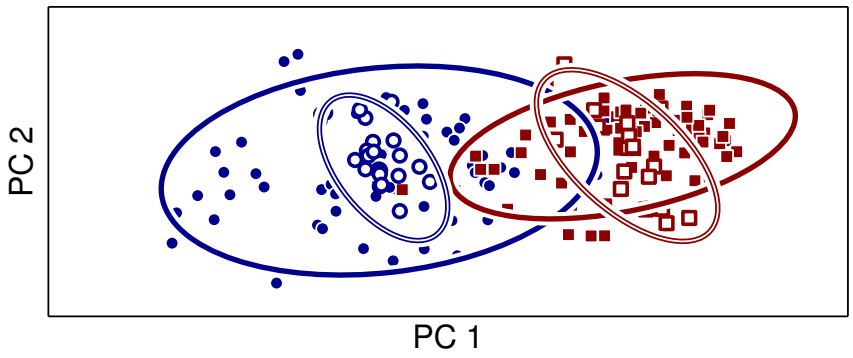


Figure 8.3: A KPCA plot of the NSK feature space embeddings of the positive bags (filled blue circles), negative bags (filled red squares), positive shuffled bags (hollow blue circles), and negative shuffled bags (hollow red squares) for the "Checkered Scarf vs. Data Mining Book" dataset. Fitted Gaussians are shown for both original and shuffled bags of each class.

bag $\mu(S^+)$ is close to $\mu(X_p)$. Furthermore, we can rewrite $\mu(X_p)$ as:

$$\begin{aligned}\mu(X_p) &= \frac{1}{|X_p|}\sum_{x_k \in X_p} \phi(x_k) \\ &= \frac{1}{|X_p|}\left(\cdots + |B_i|\,\frac{1}{|B_i|}\sum_{x_{ij}\in B_i^+}\phi(x_{ij}) + \cdots\right) \\ &= \frac{1}{\sum_{B_i \in B_p}|B_i|}\sum_{B_i \in B_p}|B_i|\,\mu(B_i).\end{aligned}$$

Thus, the kernel mean embedding of X_p is a weighted average of the mean embeddings of each bag. When all bags are the same size, as is often the case in practice, the expression simplifies to $\mu(X_p) = \frac{1}{|B_p|}\sum_{B_i \in B_p}\mu(B_i)$, a simple average of the bag embeddings.

Therefore, in the feature space of the instance kernel k_{I}, the shuffled bags generally lie close to the average of the bag embeddings, as observed in Figure 8.3. In practice, there is some variance due to sampling around these class-conditional embeddings, but as the shuffled bag size increases, the variance of the embeddings around $\mu(X_p)$ and $\mu(X_n)$ will decrease.

8.6.3 CC-NSK

Since we know the actual distributions from which shuffled bags are sampled, we can explicitly embed these distributions into the NSK feature space. In particular, we can use X_p and X_n as empirical samples from the underlying class-conditional instance distributions, $D_{\mathcal{X}_+}$ and $D_{\mathcal{X}_-}$. For simplicity, we consider the case when s is sufficiently large so that the variance of the embeddings around these class-conditional embeddings is small. It is an open question for future work to consider how the shuffled bag size s affects the variance of the embeddings. In the case of large bags, adding shuffled bags is simply like adding many copies of $\mu(X_p)$ and $\mu(X_n)$ to the dataset. When many such bags are added, the loss due to the original bags becomes negligible. Therefore, adding shuffled bags of each class proportional to $|B_p|$ and $|B_n|$ yields the following simple formulation:

$$\begin{aligned}
\min_{\substack{\alpha^+,\alpha^-,\\ b,\\ \xi^+,\xi^-}} \; & \tfrac{1}{2} \begin{bmatrix} \alpha^+ \\ \alpha^- \end{bmatrix}^\intercal \begin{bmatrix} k_{\text{NSK}}(X_p, X_p) & k_{\text{NSK}}(X_p, X_n) \\ k_{\text{NSK}}(X_n, X_p) & k_{\text{NSK}}(X_n, X_n) \end{bmatrix} \begin{bmatrix} \alpha^+ \\ \alpha^- \end{bmatrix} + \tfrac{C}{|B|} \begin{bmatrix} |B_p| \\ |B_n| \end{bmatrix}^\intercal \begin{bmatrix} \xi^+ \\ \xi^- \end{bmatrix} \\
\text{s.t.} \quad & \alpha^+ k_{\text{NSK}}(X_p, X_p) + \alpha^- k_{\text{NSK}}(X_n, X_p) + b \geq \quad 1 - \xi^+,\ \xi^+ \geq 0, \\
& \alpha^+ k_{\text{NSK}}(X_p, X_n) + \alpha^- k_{\text{NSK}}(X_n, X_n) + b \leq -1 + \xi^-,\ \xi^- \geq 0.
\end{aligned} \tag{8.9}$$

We call the approach in Equation 8.9 the class-conditional NSK (CC-NSK), since it uses the NSK to learn from the class-conditional distributions of instances from which X_p and X_n are samples. While the kernel computation for the NSK still takes $O(|X|^2)$ time, the resulting SVM in Equation 8.9 only has 5 variables regardless of the problem size, so it can be solved in constant time. Pseudocode for an implementation of the CC-NSK that uses standard weighted SVM with a precomputed kernel is shown in Algorithm 3.

Of course, since the embeddings of X_p and X_n are averages of the embeddings of all

Algorithm 3 CC-NSK

Require: MI dataset (B, Y), regularization–loss trade-off parameter C, instance kernel k_{I}

$(X, y) \leftarrow \{(x_{ij}, Y_i) \mid x_{ij} \in B_i, (B_i, Y_i) \in (B, Y)\}$
$X_n \leftarrow \{x_i \mid y_i = -1\}$
$X_p \leftarrow \{x_i \mid y_i = +1\}$
$C_n \leftarrow C\frac{|B_n|}{|B|}$
$C_p \leftarrow C\frac{|B_p|}{|B|}$
$\mathbf{K}_{11} \leftarrow \mathrm{NSK}(k_{\mathrm{I}}, X_p, X_p)$
$\mathbf{K}_{12} \leftarrow \mathrm{NSK}(k_{\mathrm{I}}, X_p, X_n)$
$\mathbf{K}_{21} \leftarrow \mathrm{NSK}(k_{\mathrm{I}}, X_n, X_p)$
$\mathbf{K}_{22} \leftarrow \mathrm{NSK}(k_{\mathrm{I}}, X_n, X_n)$
$f \leftarrow \textsc{WeightedSVM}(\mathbf{K}, [+1; -1], [C_n; C_p])$ ▷ Uses precomputed kernel
return f

function NSK(k_{I}, X_1, X_2)
    `sum` $\leftarrow 0$
    for $i \leftarrow 1$ to $|X_1|$ **do**
        for $j \leftarrow 1$ to $|X_2|$ **do**
            `sum` $\leftarrow$ `sum` $+ k_{\mathrm{I}}(X_1[i], X_2[j])$
        end for
    end for
    `sum` $\leftarrow$ `sum`$/(|X_1|\,|X_2|)$
    return `sum`
end function

bags in B_p and B_n, it may be that the CC-NSK will not generalize well in practice. On the other hand, for small sample sizes, averaging embeddings might help to regularize the hypothesis space. Therefore, the benefit of adding many shuffled bags with the NSK might vary depending on the application. We evaluate these hypotheses in the experiments below.

8.7 Empirical Evaluation

In this section, we describe several experiments to evaluate whether the addition of shuffled bags through SMILe improves the accuracy of MI classifiers. We explore both the standard MI classification setting and the active learning setting in which

training set sizes are initially small. For both sets of experiments, we evaluate the use of SMILe with the MI-SVM$_\mathrm{I}$ on the instance-labeling task and SMILe with the NSK for the bag-labeling task. Furthermore, we evaluate the performance of the new SMILeSVM and CC-NSK algorithms. The experimental results allow us to draw the following conclusions:

- SMILeSVM significantly improves instance-level classification performance measured across 32 datasets (see Figure 8.4).
- SMILeSVM significantly improves instance-level active learning performance measured across 32 datasets (see Figure 8.10).
- The CC-NSK or SMILe with the NSK do not improve bag-level classification performance, possibly due to the change induced in the bag-level distribution (see Figure 8.7, Figure 8.9).
- Both the CC-NSK and SMILe with the NSK significantly improve bag-level active learning performance measured across 40 datasets (see Figure 8.11).

A more detailed summary of these conclusions and recommendations for the application of SMILe in practice can be found in Section 8.8.

8.7.1 Instance-Labeling Task

We hypothesize that by adding additional instance-level constraints to an MI classifier, SMILe can improve the instance-labeling performance of such classifiers. As with our analysis in Section 8.5, we investigate the use of SMILe with the MI-SVM$_\mathrm{I}$ classification algorithm. Furthermore, we evaluate the newly proposed SMILeSVM algorithm, which encodes the addition of many shuffled bags without increasing the problem size. We compare the proposed approach to the baseline of MI-SVM$_\mathrm{I}$ without shuffled bags, as well as to another resampling approach for the MI setting, bag-level

bagging (Zhou and Zhang, 2003). We hypothesize that SMILe will outperform bag-level bagging, which does not introduce new instance-level constraints.

Methodology

In order to determine the performance of MI classifiers for the instance-labeling task, we must use MI datasets that have been annotated with instance labels (note that instance labels are not used during training). Accordingly, we use 12 instance-labeled spatially independent, variable area, and lighting (SIVAL) datasets from the CBIR domain and 20 Newsgroups datasets from the text categorization domain. A detailed description of the dataset properties can be found in Appendix A.1.

We train the base MI-SVM$_\mathrm{I}$ algorithm with 0, 25, and 50 shuffled positive bags generated as described in Algorithm 1. The shuffled bag size s is chosen so that the worst-case noise rate $\epsilon_s = 10\%$. Table 8.3 shows the shuffled bag sizes used for each dataset, as well as the true positive shuffled bag label noise rate that occurs with each choice of s. Due to datasets with several true positive instances per positive bag, the actual noise rate is much smaller in practice than the worst case estimate. The SMILeSVM approach given in Equation 8.8 is also evaluated using the same shuffled bag size parameter corresponding to $\epsilon_s = 10\%$. Additionally, we train the MI-SVM$_\mathrm{I}$ algorithm on the instance-labeled datasets using up to 50 bootstrap replicates obtained via bag-level bagging (Zhou and Zhang, 2003), in which $|B|$ bags are resampled with replacement from the original dataset to create each replicate.

We use the RBF kernel $k(x, x') = \exp(-\gamma \left\| x - + x' \right\|^2)$ with MI-SVM$_\mathrm{I}$ and SMILeSVM. The kernel parameter $\gamma \in 10^{[-6,\,1]}$ and SVM loss–regularization trade-off parameter $C \in 10^{[-3,\,5]}$ are selected using random parameter search Bergstra and Bengio (2012) with 5-fold inner cross-validation on the training set. We use an outer 10-fold stratified cross-validation to evaluate algorithm performance by pooling predictions across the folds. The maximum number of iterations for iterative optimization procedures

Table 8.3: Properties of the datasets used for the experiments. The rightmost columns compare the average bag size to the shuffled bag size s selected so that $\epsilon_s = 10\%$, and the actual label error rate resulting from that bag size.

Dataset	$\text{avg}_i \lvert B_i \rvert$	s	Actual ϵ_s
musk1	5	9	—
musk2	65	59	—
elephant	7	17	—
tiger	6	12	—
fox	7	14	—
field	9	20	—
flower	9	20	—
mountain	9	20	—
OHSUMED1	7	18	—
OHSUMED2	8	19	—
Apple vs. Coke Can	32	72	3×10^{-4}
Banana vs. Gold Medal	32	72	2×10^{-5}
Blue Scrunge vs. Ajax Orange	32	71	1×10^{-4}
Cardboard Box vs. Candle with Holder	32	72	0
Checkered Scarf vs. Data Mining Book	32	73	0
Dirty Work Gloves vs. Dirty Running Shoe	32	72	0
Fabric Softener Box vs. Glazed Wood Pot	32	72	0
Julie's Pot vs. Rap Book	32	72	0
Smiley Face Doll vs. Felt Flower Rug	32	72	0
Striped Notebook vs. Green Tea Box	32	72	0
WD-40 Can vs. Large Spoon	32	72	0
Wood Rolling Pin vs. Translucent Bowl	31	72	2×10^{-5}
alt.atheism	53	122	3×10^{-2}
comp.graphics	31	71	4×10^{-2}
comp.os.ms-windows.misc	50	116	4×10^{-2}
comp.sys.ibm.pc.hardware	47	109	3×10^{-2}
comp.sys.mac.hardware	43	100	3×10^{-2}
comp.windows.x	31	72	3×10^{-2}
misc.forsale	51	118	4×10^{-2}
rec.autos	34	80	4×10^{-2}
rec.motorcycles	48	110	3×10^{-2}
rec.sport.baseball	34	79	4×10^{-2}
rec.sport.hockey	20	46	4×10^{-2}
sci.crypt	42	96	4×10^{-2}
sci.electronics	32	74	5×10^{-2}
sci.med	30	70	4×10^{-2}
sci.space	37	86	3×10^{-2}
soc.religion.christian	49	112	3×10^{-2}
talk.politics.guns	35	80	4×10^{-2}
talk.politics.mideast	32	75	4×10^{-2}
talk.politics.misc	45	105	4×10^{-2}
talk.religion.misc	48	111	4×10^{-2}

is set to 50. Since SMILe only trains on a single random resampling, we perform 10 repetitions for each outer training fold to reduce variance in the performance estimate caused by the resampling procedure. A separate pool across the 10 folds is kept for each repetition, and the resulting pooled performance measures are averaged across repetitions.

Because there is a large imbalance between positive and negative instances in the datasets, we use balanced accuracy to evaluate predictions. Balanced accuracy is defined as the average of the true-positive rate and true-negative rate. We only use the true instance-level labels to evaluate performance, so we only compute instance-level balanced accuracy on the predictions pooled on outer folds. Therefore, we use bag-level balanced accuracy (which is essentially accuracy because classes are balanced in our datasets at the bag level) for parameter selection during the inner cross-validation.

We use the Wilcoxon signed-rank test to test the null hypothesis that the performance of SMILe or SMILeSVM differs significantly from MI-SVM$_\mathrm{I}$ across datasets. Independent tests are used to make pairwise comparisons between the approaches and the baseline with a significance level of $\alpha = 0.05$.

Results and Discussion

Figure 8.4 shows a summary of results comparing SMILe and SMILeSVM to the base MI-SVM$_\mathrm{I}$ classifier. Appendix A.2.4 lists a full table of numerical results. Figure 8.5 shows learning curves for bag-level bagging. To more easily compare across datasets, the percent improvement relative to the baseline of MI-SVM$_\mathrm{I}$ without bagging is shown. The average percent improvement across datasets is plotted as a dashed line.

In Figure 8.4, we see that SMILeSVM, which simulates adding a large sample of shuffled bags to the traing set, improves performance significantly across datasets with $\alpha = 0.05$ significance. Adding a finite number of shuffled bags occasionally improves performance, but not always. When 25 bags are added, there is no significant increase

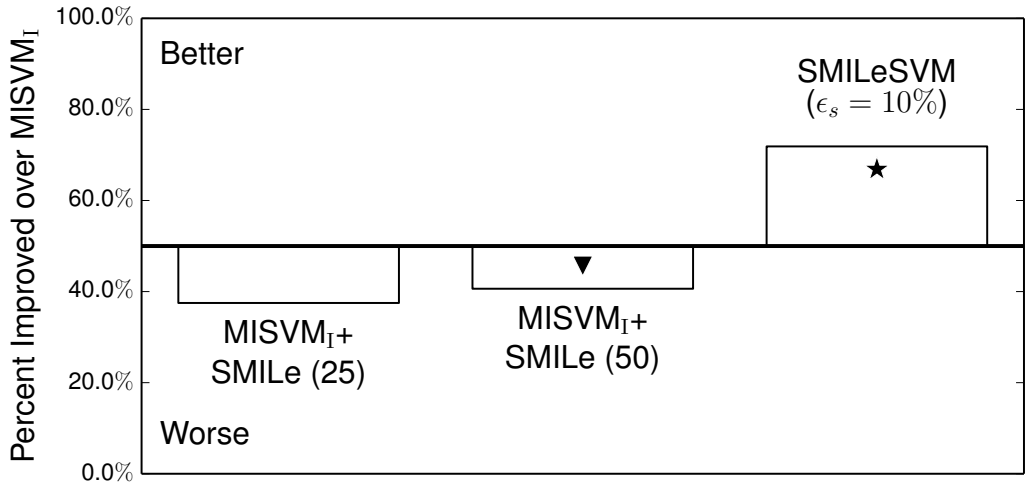


Figure 8.4: A summary of the percentage of the datasets for which SMILe or improves performance in terms of instance-level balanced accuracy over the baseline algorithm. Statistically significant differences are indicated with a star for increases in performance and a triangle for decreases in performance.

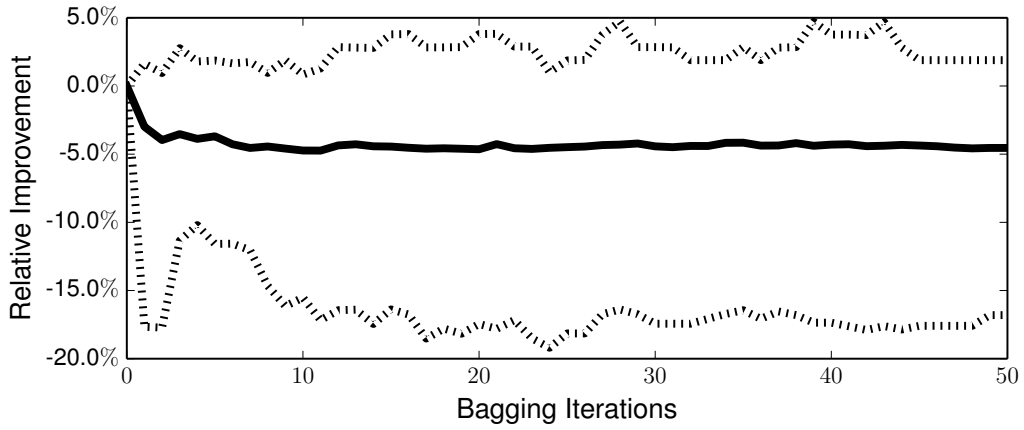


Figure 8.5: A plot of the percent improvement in performance relative to the baseline algorithm using bag-level bagging. The solid line shows how the average performance across datasets changes as various numbers of bootstrap replicates are used. The dashed lines show the minimum and maximum improvement.

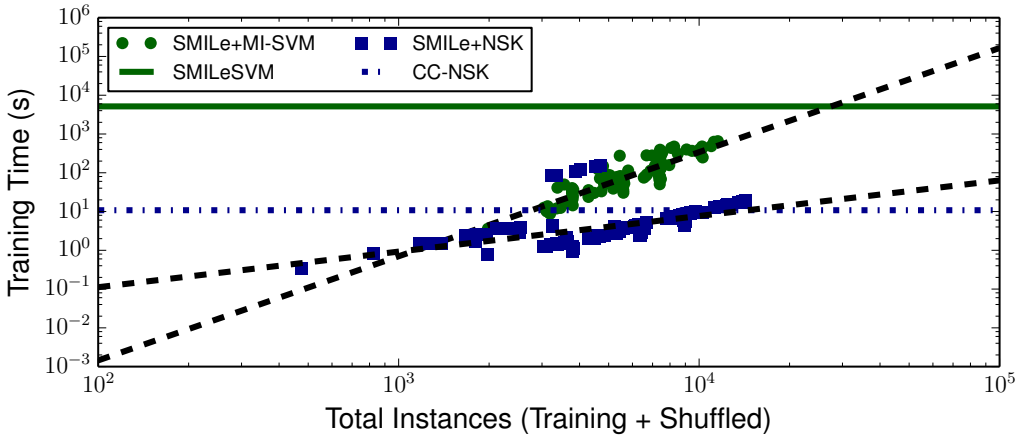


Figure 8.6: A comparison of the training times of the NSK (blue squares) and (green circles) with SMILe, as well as that of the CC-NSK (dashed blue line) and of (solid green line). Least-squares fits of the training times are plotted as black dashed lines.

or decrease in performance across the 32 datasets. However, when 50 shuffled bags are added to each dataset, the performance decreases significantly.

We conjecture that when many shuffled bags are added to the dataset, the complexity of the resulting QP may cause the decrease in performance. In particular, the addition of 50 shuffled bags more than doubles the number of instances and constraints for some datasets, which tends to make the nonconvex MI-SVM$_\mathrm{I}$ optimization procedure more challenging to solve numerically than that of SMILeSVM. Furthermore, the additional bags quadruples the kernel size and approaches the limit of what can be stored in memory.

The performance of SMILeSVM supports our hypothesis that the addition of shuffled bags can improve the performance of an MI classification algorithm. There are only 4 of 32 datasets for which SMILe or SMILeSVM does not improve the performance of MI-SVM$_\mathrm{I}$. However, due to the negative practical consequences of explicitly adding shuffled bags to the dataset, we recommending using SMILeSVM explicitly rather than adding shuffled bags with SMILe.

Figure 8.6 shows a comparison of the running times of SMILeSVM and MI-SVM$_\mathrm{I}$ with SMILe. The solid green line shows the average runtime of SMILeSVM. Because

of the additional iterative optimization required, SMILeSVM takes longer to run than MI-SVM$_\text{I}$ with shuffled bags (green circles). However, a best-fit line shows that explicitly adding shuffled bags will eventually require more training time than SMILeSVM, which implicitly uses an infinite number of shuffled bags. On the other hand, as discussed above, memory remains the bottleneck when explicitly adding shuffled bags, so MI-SVM$_\text{I}$ with SMILe will likely encounter practical issues before the dataset exceeds roughly 25,000 instances in size.

The results in Figure 8.5 show that bag-level bagging does not improve, and even harms, performance of instance-level MI classification. These results align with our hypothesis that resampling entire bags does not add instance-level constraint information to the classifier. While bag-level bagging might improve bag-labeling performance, recent work has described how good bag-level performance does not necessarily imply good instance-level performance (Tragante do Ó et al., 2011). Therefore, for the instance-labeling task, SMILe improves performance while resampling entire bags does not.

8.7.2 Bag-Labeling Task

We hypothesize that although SMILe alters the bag-level distribution of the training set, it can improve the performance of bag-level labeling by providing additional novel examples to an MI classification algorithm. As in Section 8.6, we focus our empirical analysis on the use of SMILe with the NSK. We also investigate the use of bag-level bagging with the NSK as an alternative resampling approach.

Methodology

As for the instance-labeling experiments, we use 12 SIVAL datasets and 20 Newsgroups datasets. However, because instance-level labels are no longer required for evaluation of bag-labeling approaches, we additionally include 3 animal and 3 scene

image datasets from the CBIR domain, 2 from the 3D-QSAR domain, and 2 from the text categorization domain (see Table A.2). The properties of these datasets are listed in Table 8.3, with additional details in Appendix A.1.

We train an SVM using the NSK with 0, 25, and 50 shuffled bags of each class. We choose the shuffled bag size so that the worst-case label noise on the shuffled positive bags is $\epsilon_s = 10\%$ (see Table 8.3 for the shuffled bag sizes used for each dataset). We also evaluate the CC-NSK approach given in Equation 8.9. The CC-NSK does not take any parameters beyond the kernel and SVM parameters. We also evaluate bag-level bagging with the NSK using all datasets. We use up to 50 bootstrap replicates as for the instance-labeling experiments. We use the RBF kernel for the instance kernel k_{I} used to construct the NSK. Parameter selection and cross-validation are performed as described above for instance-labeling experiments.

Because the datasets we use for bag-labeling experiments have roughly equal numbers of positive and negative bags, we use accuracy to evaluate performance. We use the Wilcoxon signed-rank test to test the null hypothesis that the performance of SMILe or CC-NSK differs significantly from the NSK across datasets. Independent tests are used to make pairwise comparisons between the approaches and the baseline with $\alpha = 0.05$ significance.

Results and Discussion

Figure 8.7 summarizes the performance of SMILe and the CC-NSK compared with the baseline NSK approach. A full table of results can be found in Appendix A.2.4. Figure 8.8 shows the results of bag-level bagging with the NSK base classifier. As in the instance-labeling experiments above, we plot relative percent improvement to make comparison easier across datasets. The dashed line shows the average percent improvement of bagging with increasing numbers of bootstrap replicates used along the x-axis.

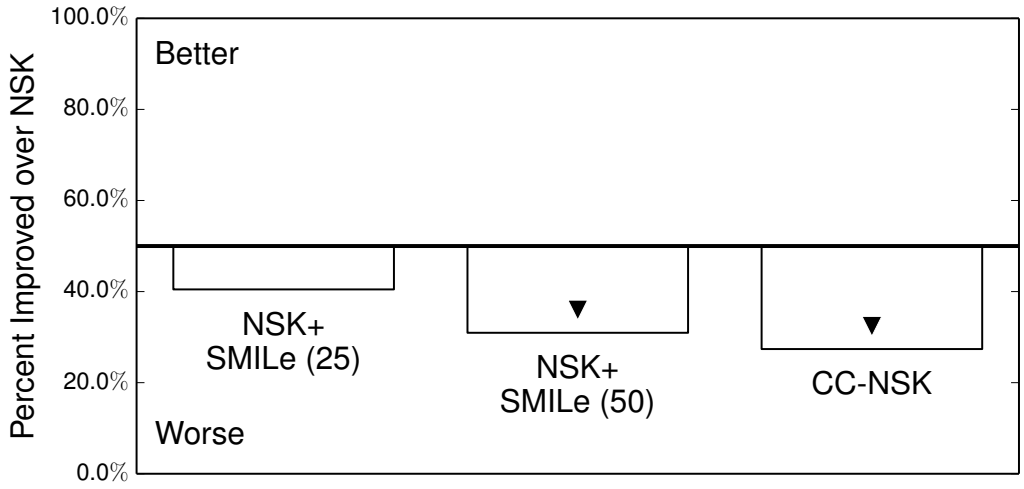


Figure 8.7: A summary of the percentage of the datasets for which SMILe or the CC-NSK improves performance in terms of bag-level accuracy over the baseline NSK approach. Statistically significant differences are indicated with a triangle for decreases in performance.

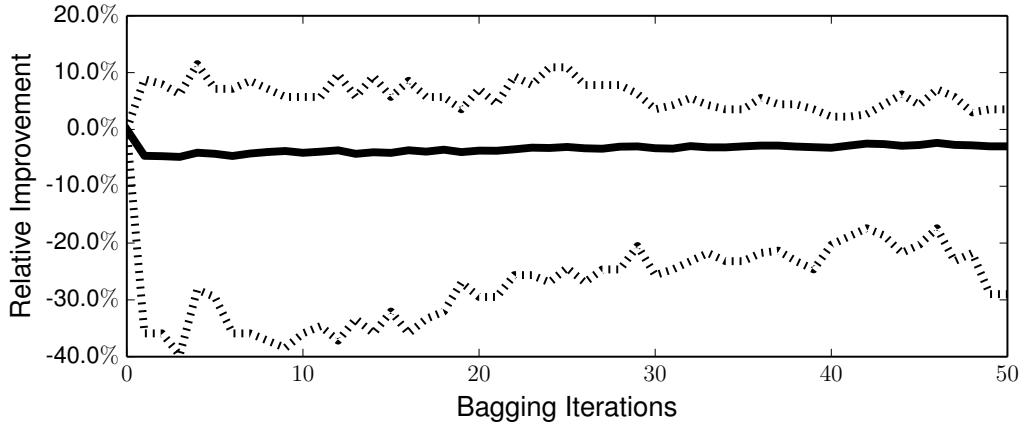


Figure 8.8: A plot of the percent improvement in performance relative to the baseline NSK approach using bag-level bagging. The solid line shows how the average performance across datasets changes as various numbers of bootstrap replicates are used. The dashed lines shows the minimum and maximum improvements.

First, we observe that if we add a small number of shuffled bags, the accuracy of SMILe relative to the baseline NSK occasionally improves, but for less than half of the datasets. As the number of shuffled bags increases towards the limiting case of the CC-NSK, the benefit of SMILe decreases and accuracy is statistically worse than the baseline after 50 shuffled bags are added.

The decrease in performance with large numbers of shuffled bags, or the limiting case simulated by the CC-NSK, is likely due to the difference between the train and test distributions over bags induced by SMILe, as discussed in Section 8.6. For bag-level classifiers like the NSK, the discrepancy between train and test distributions can lead to bad generalization, and might contribute to the poor performance observed in Figure 8.7. We conjecture that similar behavior would be observed for other MI algorithms that map entire bags into a feature space for classification. As noted in prior work, some bag-level classifiers can take advantage of the structure of instances within bags (Zhou et al., 2009), but SMILe disrupts this structure. We note that similar behavior is *not* observed for instance-labeling approaches, since SMILe does not alter the underlying distribution over *instances* as it does the distribution over bags—instances are sampled from the underlying instance distribution.

We observe that the performance of bag-level SMILe varies across datasets. For example, SMILe benefits performance for the "elephant" dataset, but significantly reduced the performance for the "Julie's Pot vs. Rap Book" dataset. We conjecture that such differences can be at least partially explained by the fact that SMILe introduces a discrepancy between the training and test distributions over bags, as described above. To test this conjecture, we explicitly measure the difference between the original training distribution and the distribution over shuffled bags induced by SMILe using a kernel-based two-sample test (Gretton et al., 2007, 2009). To apply the two-sample test to distributions over bags, we use the universal "level-2" embedding kernel (Muandet et al., 2012). That is, we first embed bags into an RKHS using

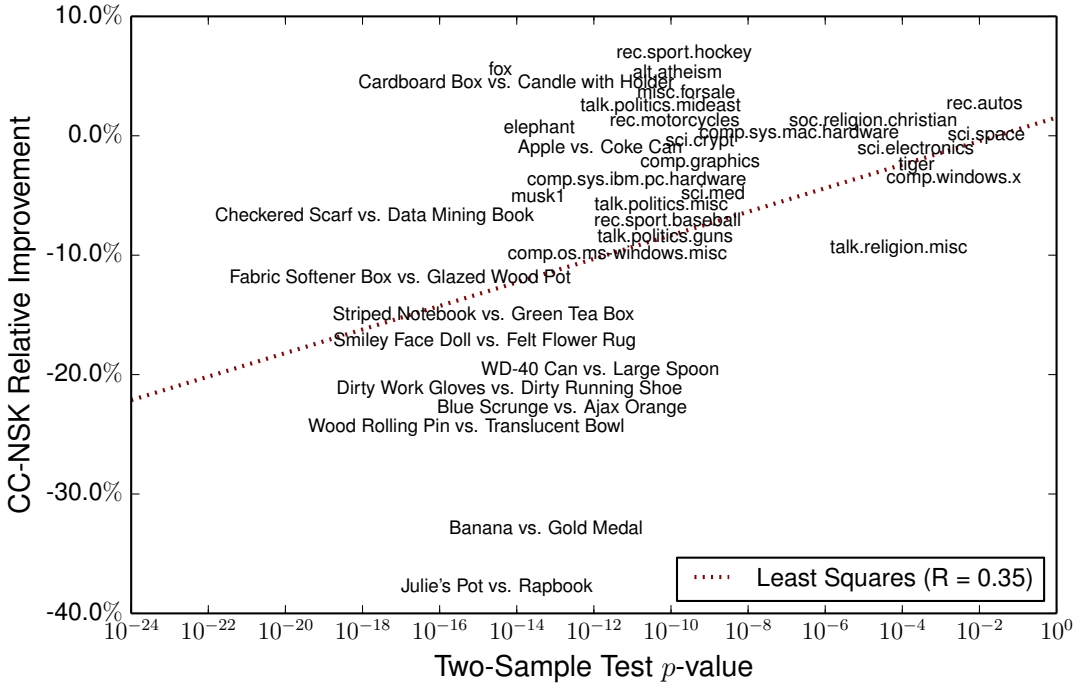


Figure 8.9: A plot demonstrating the relationship between the performance of the CC-NSK and the distance between the original training distribution and the distribution over shuffled bags, as measured by the kernel-based MMD test statistic p-value.

the RBF kernel with the γ selected via cross validation in the experiments described above, and then embed the resulting distributions over bags into an RKHS using a second RBF kernel with the bandwidth parameter selected using the median heuristic (Gretton et al., 2009). The two samples compared are the original set of bags and a set of shuffled bags containing the same number of positive and negative bags as the original training set. We use the biased Maximum Mean Discrepancy (MMD) test statistic, with a Gamma distribution used to approximate the null distribution (Gretton et al., 2009).

Figure 8.9 shows the relative performance of the CC-NSK (with respect to the baseline with no shuffled bags) as a function of the p-value of the MMD test statistic. Datasets with zero p-values are omitted. For datasets with large p-values, the two-sample test does not detect a large difference between the distributions over original and shuffled bags. In this case, we observe that there is often little degradation in

performance when using SMILe. On the other hand, as the p-value of the MMD test statistic decreases, meaning that there is a more apparent discrepancy between the original and shuffled bag distributions, SMILe tends to negatively affect the performance of the base classifier. The Pearson correlation coefficient between log p-values and relative performance is 0.35, which suggests a modest but statistically significant ($\alpha = 0.05$) relationship between the change in distribution as measured by the two-sample test and the performance of SMILe. Because the correlation is low, despite being significant, there are likely other factors that contribute to the poor performance of SMILe when used with the NSK for classification. For example, we conjecture that because the NSK with an RBF kernel essentially represents *linear* concepts over bags (Muandet et al., 2012), the variance reduction afforded by SMILe does not lead to an improvement in performance given the number of bags in the training sample. We show in Section 8.7.3 that with fewer training bags, SMILe can provide an advantage in most cases. On the other hand, the results in Figure 8.9 themselves suggests that SMILe can be conservatively applied to bag-level classifiers when there is not a significant difference between the training and shuffled bag distributions according to the kernel two-sample test. However, there are clearly some scenarios in which SMILe improves performance despite the discrepancy between distributions.

Additionally, we note that the results in Figure 8.7 appear to support our hypothesis that the CC-NSK approximates the limiting behavior of the NSK with many shuffled bags. To further quantify the relationship between the CC-NSK and the NSK with SMILe, we measure the correlation between the improvements over the baseline NSK of both approaches. After 50 shuffled bags, the Pearson correlation coefficient is 0.32, with is statistically significant at $\alpha = 0.05$. Therefore, the results suggest that, as hypothesized, the behavior of SMILe with the NSK is significantly related to the performance of the CC-NSK when large numbers of bags are used.

Though the CC-NSK solves a constant-sized optimization program, Figure 8.6

shows that the average training time of the CC-NSK (dashed blue line) is not significantly less than that of the NSK without shuffled bags (left-most blue circles). This is because both CC-NSK and the NSK require computing a kernel matrix with $O(|X|^2)$ entries. Although the NSK additionally requires solving a QP, which takes roughly $O(|B|^3)$ time, $|X| \gg |B|$ for many datasets, so the additional time is relatively negligible in practice. On the other hand, running time does increase with more explicitly added shuffled bags, so CC-NSK does have an advantage over SMILe in terms of running time.

Finally, we observe that bagging slightly decreases performance on average for this task. This is possibly due to low bag-level label noise in most of these datasets, which consist of manually annotated images (so that, for example, it is very rare that an image from the "Coke Can" class will be mislabeled as not containing a Coke can, etc.) and Newsgroups posts with ground-truth labels.

8.7.3 Active Learning Task

As argued in Section 8.2, SMILe can be viewed as adding "virtual examples" to a training set, which provides a form of regularization by reducing the variance in the underlying hypothesis class. Regularization can be particularly useful when the training set is small. Furthermore, a small training sample does not provide a reliable estimate of the underlying distribution over bags. For these reasons, the benefits of SMILe might outweigh the issues caused by the changes in distribution for bag-level labeling, as discussed above. Small datasets arise naturally in the active learning domain, in which an oracle (e.g., a human user of the system) is iteratively queried for labels to learn an accurate classifier using very few labeled examples. Active learning in the MI setting was introduced in prior work (Settles et al., 2008), where instance labels are queried to improve an instance classifier initially trained with few labeled bags. In these experiments, we instead focus on a setting in which a user is

queried for the labels of bags to train a classifier for predicting the labels of either instances or bags in a test dataset. As above, we evaluate active learning with bag- and instance-labeling approaches (NSK and MI-SVM$_{\mathrm{I}}$), as well as the novel CC-NSK and SMILeSVM techniques.

Methodology

For the active learning experiments, we use the same datasets as described in the experiments above. Because only small training samples are used during active learning, we do not perform cross-validation for parameters. Instead, we fix the RBF kernel parameter γ and the SVM regularization–loss trade-off parameter C to be the values found via cross-validation in the experiments above. For the case of SMILe with either the NSK or MI-SVM$_{\mathrm{I}}$, we use the parameter values found using cross-validation on the corresponding baselines without shuffled bags. Thus, if any bias is introduced in the SMILe+MI-SVM$_{\mathrm{I}}$ or SMILe+NSK experiments, it is in favor of the baseline with no shuffled bags.

We compute performance using predictions pooled across an outer 10-fold cross-validation. Within each training fold, we repeat the active learning procedure 20 times. The active learning procedure is repeated 20 times. The active learning procedure consists of sampling an initial number of labeled bags, either 5 or 10 from each class. The remaining bags form an unlabeled "pool." At each iteration, some fraction of shuffled bags are added to the training set to train the base classifier using SMILe. For MI-SVM$_{\mathrm{I}}$, either 0%, 50%, or 100% of the number of positive training bags are used, for a maximum of 40 shuffled bags of each class. For the NSK, the same fractions are used for shuffled positive *and* negative bags. No shuffled bags are added when the CC-NSK or SMILeSVM are used as the learners. We select the shuffled bag size corresponding to a worst-case noise rate of $\epsilon_s = 10\%$. Then we use the "simple margin" strategy (Tong and Koller, 2002) to query a batch of $k = 2$ labels of the bags

Algorithm 4 SMILe Active Learning

Require: Initial MI dataset (B_0, Y_0), unlabeled pool B_{pool}, bag-labeling oracle $\mathcal{O}$, noise rate ϵ_s, base learner $\mathcal{A}$, fraction of shuffled positive bags σ_p, fraction of shuffled negative bags σ_n, step size k, iterations n

$B_t \leftarrow B_0$
$Y_t \leftarrow Y_0$
$\mathcal{C}_0 \leftarrow \text{SMILe}(B_t, Y_t, \epsilon_s, \mathcal{A}, \sigma_p \left|B_t^+\right|, \sigma_n \left|B_t^-\right|)$ ▷ Initial classifier
for $i \leftarrow 1$ to $\lceil n/k \rceil$ **do**
    $Y_{\text{pool}} \leftarrow \mathcal{C}_{i-1}(B_{\text{pool}})$ ▷ Estimate labels of examples in pool
    for $j \leftarrow 1$ to k **do**
        $m \leftarrow \arg\min_i |Y_{\text{pool}}[i]|$ ▷ Find example closest to SVM hyperplane
        $Y_{\text{pool}}.\texttt{pop}(m)$ ▷ Remove label for next iteration
        $B_m \leftarrow B_{\text{pool}}.\texttt{pop}(m)$
        $B_t \leftarrow B_t \cup B_m$
        $Y_t \leftarrow Y_t \cup \mathcal{O}(B_m)$
    end for
    $\mathcal{C}_i \leftarrow \text{SMILe}(B_t, Y_t, \epsilon_s, \mathcal{A}, \sigma_p \left|B_t^+\right|, \sigma_n \left|B_t^-\right|)$
end for
return $\{\mathcal{C}_i\}$

that are closest to the SVM separator. After every batch of queries, the resulting classifier is retrained with the additional labeled bags and evaluated on the held-out test fold. The process described above continues until the labels of $n = 30$ bags have been queried. Algorithm 4 gives an overview of the procedure.

As in the experiments above, we use accuracy to evaluated bag-labeling performance and balanced accuracy to evaluate performance at the instance level. After each active learning query, classifier predictions are made on the 10 held-out test folds and pooled to compute accuracy. Then, these accuracy values are averaged across the 20 repetitions using different sets of initial bags to reduce variance due to sampling/resampling. After each batch of queries, we use a Wilcoxon signed-rank test to test the null hypothesis that the performance of the procedure with SMILe differs from the baseline of active learning without the addition of any shuffled bags.

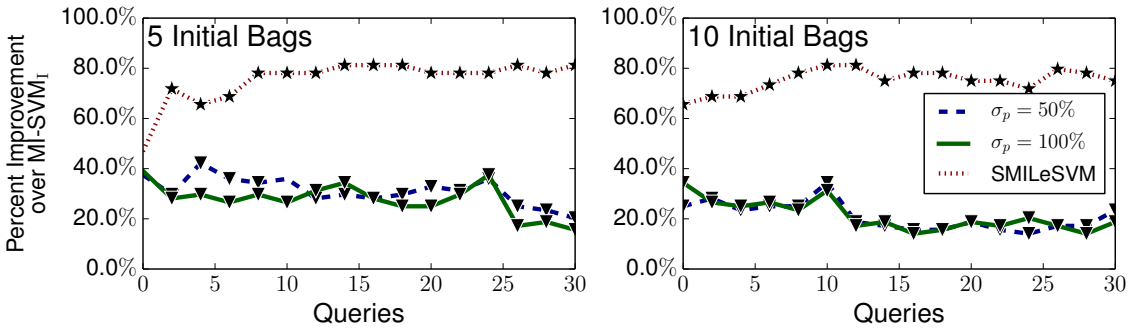


Figure 8.10: **(Instance-Level MI Active Learning)** Each plot shows an active learning experiment starting with 5 or 10 initial bags. For either 50% shuffled bags, 100% shuffled bags, or the , the y-axis represents the fraction of the times (across the 32 instance-labeled datasets) that SMILe outperforms the baseline of the with no shuffled bags. The results are plotted as the number of bag label queries increases along the x-axis. Stars and triangles indicate a significant increases and decreases in performance with SMILe relative to baseline balanced accuracy values using a 2-sided Wilcoxon signed-rank test at 0.05 significance.

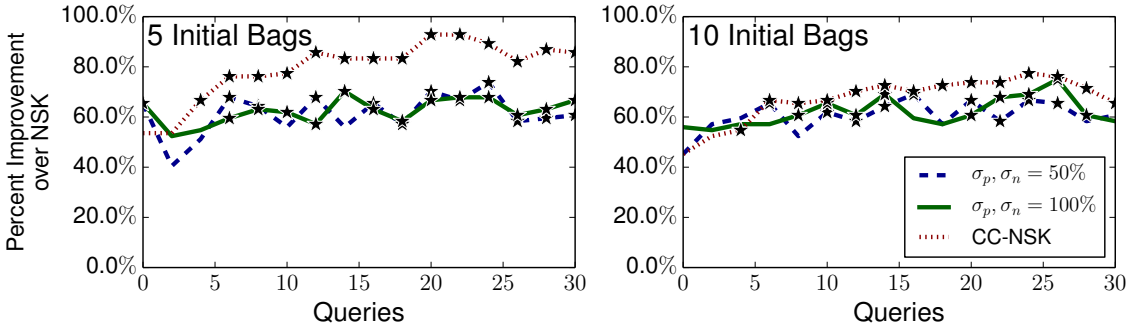


Figure 8.11: **(Bag-Level MI Active Learning)** Each plot shows an active learning experiment starting with 5 or 10 initial bags. For either 50% shuffled bags, 100% shuffled bags, or the CC-NSK, the y-axis represents the fraction of the times (across the 42 datasets) that SMILe outperforms the baseline of the NSK with no shuffled bags. The results are plotted as the number of bag label queries increases along the x-axis. Stars indicate a significant difference between SMILe and baseline accuracy values using a 2-sided Wilcoxon signed-rank test at 0.05 significance.

Results and Discussion

Figure 8.10 and Figure 8.11 show the active learning results. Figure 8.10 shows the fraction of the time that SMILe or SMILeSVM outperforms the baseline of the MI-SVM$_\text{I}$ with no shuffled bags with respect to instance-level balanced accuracy. Similarly, Figure 8.11 shows the fraction of the time that either SMILe or the CC-NSK outperforms the baseline of the NSK with no shuffled bags on bag-level accuracy. In both cases, significant differences for either better or worse performance according to the Wilcoxon signed-rank test with $\alpha = 0.05$ significance are indicated with stars or triangles, respectively.

The instance-level results in Figure 8.10 show a benefit of SMILe for improving the performance of the MI-SVM$_\text{I}$. As for the classification experiments, SMILeSVM clearly improves the performance of active learning relative to the baseline MI-SVM$_\text{I}$ approach. SMILe with a finite number of shuffled bags does not improve the performance of the baseline classifier, but actually leads to lower balanced accuracy. However, this result is not surprising given that the parameters have been fixed to those that produce the best performance of the *baseline* on the full training set.

The bag-level results in Figure 8.11 confirm our intuition that bag-level SMILe can be beneficial when datasets are small. In fact, even the CC-NSK, which performs poorly relative to the NSK on the full datasets, outperforms the baseline NSK approach in the active learning scenario. Thus, using small datasets, our results demonstrate that SMILe and the CC-NSK can prevent over-fitting by regularizing the hypothesis space. In the case of active learning, the benefit of the regularization afforded by SMILe can outweigh the issues caused by distorting the training distribution.

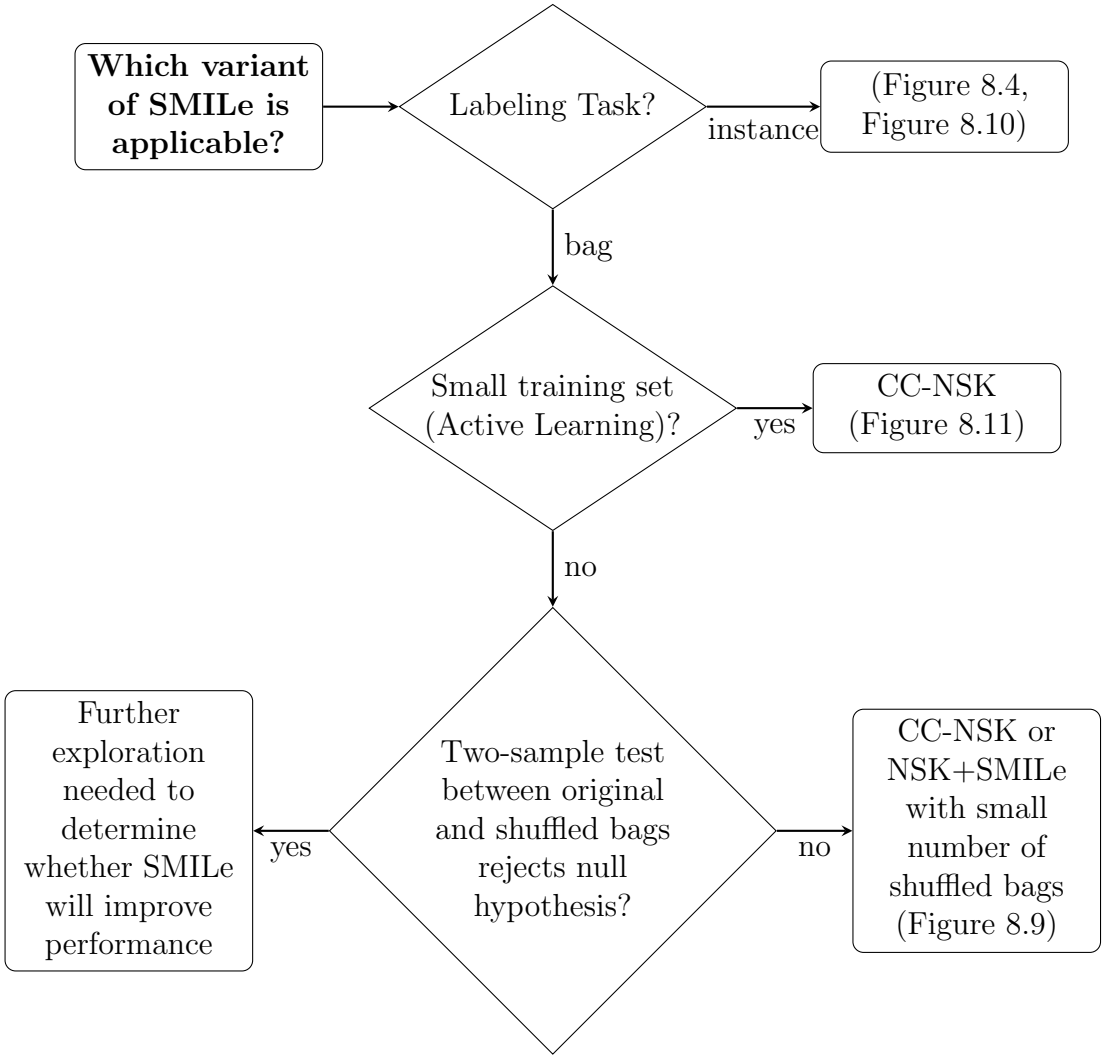


Figure 8.12: A flowchart summarizing recommendations for applying SMILe in practice, given the experimental results above.

8.8 Summary

In this chapter, we presented a new resampling approach for the MI setting in which new bags can be constructed to augment a small dataset. We used the new generative model to analyze the effect of the resampling process on instance- and bag-level distributions, and to develop new SVM-based classifiers. We then performed experiments, which inform when SMILe might be applied in practice. Given the experimental results, our recommendations for applying SMILe are presented as a flowchart in Figure 8.12. For the instance-labeling task, our results show SMILeSVM to be superior

in practice to explicitly adding shuffled bags (Figure 8.4). Furthermore, SMILeSVM improves performance significantly across datasets in both the standard classification and active learning settings (Figure 8.10). For bag-labeling, SMILe, and especially the CC-NSK, consistently improves performance when bag sizes are small in the active learning setting (Figure 8.11). For the standard classification task, SMILe can hurt the performance of bag-level classifiers (Figure 8.7). However, as long as the shuffled bags do not significantly affect the training distribution, it appears that SMILe will not hurt performance (Figure 8.9). Otherwise, SMILe could be applied to the bag-level classification task after checking the difference between the shuffled bag distribution and the original bag distribution.

Chapter 9

Conclusions

Although MIL is a well-established learning framework for which many algorithms exist, the theoretical basis for the setting has received relatively little attention. Previous models of how data is generated in the MI setting make assumptions that are not realistic (Blum and Kalai, 1998; Zhou et al., 2009). For example, the IID r-tuple model assumes that instances across *all* bags are drawn from the same distribution, when for the 3D-QSAR domain, this would imply that every molecule had an identical distribution over conformations. With these simplified generative models, a few positive learnability results are known (Blum and Kalai, 1998), but most results under relaxed models show that MIL is hard in the worst case (Auer et al., 1998).

At the same time, empirically, several counterintuitive observations have been observed in the literature. First, some work shows that if area under the ROC curve (AUC) is used as an evaluation metric, supervised learning algorithms appear to perform well on MIL tasks (Ray and Craven, 2005). Second, other work shows that unlike the generative assumption made in prior models that instance labels determine bag labels, in practice, there can be little correlation between the performance of MI algorithms on these two tasks (Tragante do Ó et al., 2011).

The contributions of this work, summarized below, include the introduction of a

new theoretical framework for the MI setting that can (1) make sense of counterintuitive empirical observations, (2) be used to demonstrate new positive learnability results, (3) analyze existing MI algorithms in a way that explains their behavior, and (4) develop and apply novel approaches to the MI learning problem.

9.1 Summary

My contribution is a two-level generative model that leverages the observation that bags can be viewed as distributions over instances. This generative model is described in detail in Chapter 3. I show that this generative model has explanatory and predictive power, both theoretically and empirically. It is primarily motivated by an attempt to be more realistic for real-world problem domains, such as 3D-QSAR, where molecules can be represented by a distribution over conformations. In fact, I show that most of the surprising empirical observations in literature can be explained under this new model. First, I show in Chapter 4 that instance and bag concepts are learnable under the model, providing the first positive instance learnability results since Blum and Kalai (1998), of which our results are a generalization. Further, we show in Chapter 5 that the new generative model allows us to show that supervised algorithms can learn to rank from MI data.

A summary of the theoretical results on instance and bag concept learnability can be found in Table 4.1. The results require some additional weak assumptions about the relationship between bag and instance labels, as summarized in Table 9.1. For instance learnability, we require that each negative instance appears some fraction of the time in negative bags ($\gamma > 0$). Otherwise, such instances would be indistinguishable from positive instances, which naturally appear only in positive bags. For bag learnability, we require that positive instances occur some fraction of the time in every positive bag ($\pi > 0$). If not, then there would be positive bags containing only

Table 9.1: A summary of the assumptions required for various results.

	... using **instance-level** classifiers.	... using **bag-level** classifiers.
Learning **instance-level** concepts...	Theorem 4.1 Theorem 5.1 $\gamma > 0$	—
Learning **bag-level** concepts...	Theorem 4.3 Theorem 5.3 $\gamma > 0$, $\pi > 0$	Proposition 6.1 $\pi > 0$

negative instances, which would be indistinguishable from negative bags that naturally only contain negative instances by the MI assumption. Essentially, the $\gamma > 0$ assumption is only essential for learning instance labels (see Chapter 4). For learning bag labels with a universal distribution kernel, only the $\pi > 0$ assumption is required (see Chapter 6). Both assumptions are only required simultaneously if one wishes to learn bag labels using an instance-level classifier whose predictions are combined via the max function (see Chapter 4). However, the results in Chapter 6 suggest that bag-level classifiers should be used for the bag-level learning task.

Finally, we introduce a new resampling approach, SMILe, which adds additional labeled bags to an MI training set. We show that SMILe can improve the performance of MI classifiers when the training set is initially small, as in the active learning setting. We use our generative model to analyze how the resampling procedure affects the generative process of the data.

We present empirical results that support the theoretical predictions made under our generative model. Our results also provide some indication of when certain approaches should be used in practice to learn from MI data. Figure 9.1 summarizes the lessons learned from the empirical results. The first relevant question is whether the training sample is large or small in terms of the number of bags. Since the theoretical results in Chapter 4 only show asymptotic learnability, specialized algorithms might

perform best when the sample size is small. In particular, if prior knowledge of the problem domain is available, then additional assumptions might be used to design a "generative" classifier that incorporates these assumptions. Otherwise, the results in Chapter 8 show that SMILe can improve the performance of both instance- and bag-level classifiers when the training set is small.

If the dataset is large, then generative classifiers could become inefficient, even if prior information is available. However, depending on the learning task, the use of more efficient supervised approaches is justified. For the bag-level classification task, Chapter 6 shows that using distribution-based kernels with a standard SVM implementation achieves state-of-the-art performance on the bag-labeling task with respect to either accuracy or AUC.

For the instance-level classification task, the choice of algorithm depends on the performance metric. If high AUC is desired, then a good classifier can be learned using single-instance learning (SIL), which labels every instance with its bag's label and trains a supervised classifier, such as an SVM. Theoretical and empirical support for this claim is provided in Chapter 5. If high accuracy is desired, then one can use a minimum one-sided disagreement strategy (see Chapter 4). However, although minimum one-sided disagreement is straightforward, it is not commonly used in practice. One alternative, as described in Chapter 8, is to use SMILeSVM, a new resampling-based SVM approach that we propose for the MI setting. We show empirically that SMILe improves instance-level balanced accuracy of the MI-SVM$_{\text{I}}$ classifier.

Thus, while empirical results show that specialized MI techniques such as SMILe perform well in certain scenarios, out-of-the-box supervised algorithms can achieve good performance for many MI learning tasks. As a consequence for future empirical studies, our results suggest that *newly proposed MI-specific approaches should be compared not just to existing MI baselines, but also to their supervised counterparts to clearly establish their value.*

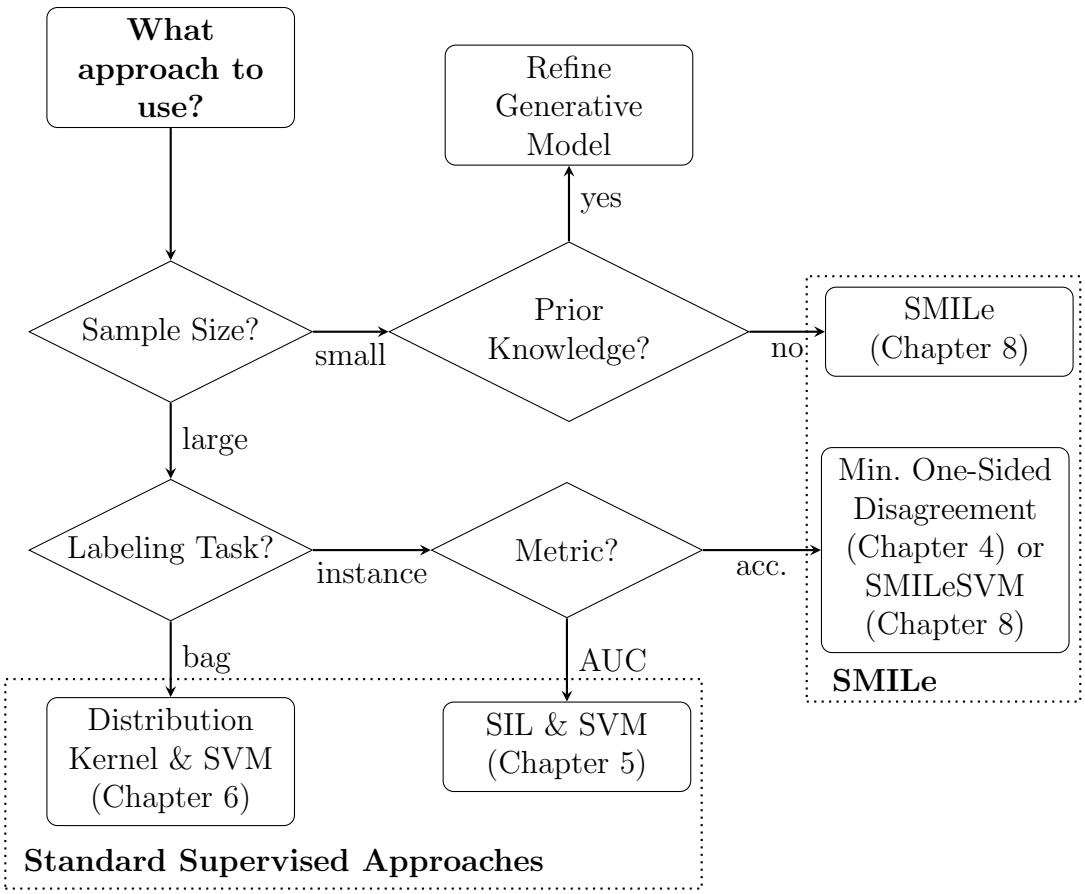


Figure 9.1: A summary of the recommended approaches for various MI learning scenarios, with the chapters containing theoretical or empirical justification.

Finally, our theoretical and empirical results also offer an explanation for why instance and bag labels are not correlated. Learning bag labels is computationally challenging using an instance-labeling hypothesis space (see Chapter 7), but our model offers a way to directly learn bag labels. In particular, we explore the application of distribution kernels to MI data. We show that previous bag-level kernel approaches can be explained as variations of the distribution kernel approaches (Chapter 6). Experimentally, many of these approaches are (a) efficient relative to instance-based approaches and (b) more accurate at predicting bag labels. As part of this work, we present a novel application of the level-2 kernel (Muandet et al., 2012) to MI problems, and we show that the performance of the kernel is promising. The empirical consequence of this work clearly demonstrates that the bag- and instance-level learn-

ing tasks should be treated separately, as treated by our generative process, and that different algorithms are appropriate at each level.

9.2 Future Work

Of course, we have only just begun to analyze the consequence of viewing bags in the MI setting as distributions. Below is a brief outline of some interesting research questions and directions that are suggested by our work.

One interesting technical question raised by the analysis in Section 6.3 concerns the representational power of the Earth-Mover's Distance (EMD) kernel. As described, the EMD and the Maximum Mean Discrepancy (MMD) are both metrics on the space of probability distributions, and both induce the same "weak" topology on that space. However, one key difference between these metrics is that the MMD is Hilbertian (it can be expressed in terms of an inner product), whereas the EMD is not. Currently, the proof of the universality of the MMD when used with the radial basis function (RBF) kernel relies on the fact that it is a Hilbertian metric. However, intuition suggests that the EMD used with an RBF kernel might be universal as well. Demonstrating the universality of this EMD kernel, or showing why it is not universal, is an interesting open question for future work, especially given the excellent performance of the EMD kernel for MI problems.

Another theoretical question of interest is whether specific algorithms are *necessary* to learn accurate concepts from MI data. In the results we present in Chapter 4, the minimum one-sided disagreement approach is required to learn accurate concepts from MI data. In contrast, the results in Chapter 5 demonstrate that standard supervised approaches can learn to *rank* from MI data. However, we have now shown that it is *not* possible to also use standard supervised approaches for learning accurate concepts. While we conjecture that learning accurate concepts with standard

supervised approaches is not generally possible in the MI setting, this remains an open question.

Section 2.1.3 discusses MI regression as a generalization of the standard MI classification setting. In MI regression, bag labels are real-valued, and formed through some more general combination of instance labels. Very recent work has established some theoretical guarantees for distribution-based kernel methods applied to the distribution-regression problem (Szabó et al., 2014). Such results can be used to show the learnability of bag-level concepts in the MI regression setting. Another open direction is to derive analogous instance-concept learnability results in this setting. Formulating such a result could be more challenging due to the many possible ways that real-valued bag labels could be derived from real-valued instance labels. It is unclear whether supervised approaches might be able to learn real-valued MI instance-level concepts.

The generative model proposed in this work make very few assumptions about the distribution over bags and the distributions corresponding to the bags themselves. Consequently, the learnability results derived in Chapter 4 only describe the asymptotic behavior of a learning algorithm when a large training sample is available. On the other hand, with more specific generative assumptions, it is possible to learn an accurate model from a much smaller training set. Classifiers that make assumptions about the data distribution and learn to model the joint distribution over examples and their labels are called "generative" classifiers. Classifiers that directly model the conditional distribution of labels given instances are called "discriminative." For example, naïve Bayes is a generative classifier and logistic regression is the corresponding discriminative classifier (Ng and Jordan, 2002). Naïve Bayes tends to perform better than logistic regression for small samples, but logistic regression eventually outperforms naïve Bayes as the samples grow (Ng and Jordan, 2002).

Most work on classification in the MI setting has been to develop discrimina-

tive classifiers, such as SVMs. For instance, MI logistic regression (Ray and Craven, 2005) was developed roughly 5 years prior to a version of naïve Bayes for the MI setting (Hernández and Inza, 2011). However, given the theoretical conclusions of our work, summarized in Figure 9.1, supervised approaches can actually outperform many MI-specific approaches with large training datasets, which is the realm in which discriminative approaches are advantageous. Therefore, our work suggests that future work should place more emphasis on developing *generative* multiple-instance algorithms which can perform well when the training sample is small.

For real-world domains such as computational biology, some examples are inherently high-dimensional. For example, millions of nucleotide pairs can be used to describe the genome of an individual. Therefore, dimensionality reduction plays an essential role in supervised learning for reducing the complexity of a model needed to classify examples. In the MI setting, dimensionality reduction might play an additional role. In particular, dimensionality reduction might be used to induce the property that negative instances appear in negative bags ($\gamma > 0$), a key assumption of the generative model proposed in Chapter 3. For example, consider the 3D-QSAR domain. Suppose that each molecular conformation was described precisely with a very large feature vector. Because every conformation is unique, given a precise enough description, no conformation will appear in two different bags. Thus, no negative instance in a positive bag will ever appear in a negative bag, violating the $\gamma > 0$ assumption. In this case, dimensionality reduction might be used to abstract away irrelevant properties of conformations such that it is possible to observe all negative instances in negative bags some fraction of the time.

As motivated in the introduction (Chapter 1), the MI representation is one of the most basic ways of representing structured data. However, there are many other forms of structured data with examples that are represented as tree, graphs, or relations stated in first-order logic. We have shown that by applying the idea of *learning from*

distributions to the MI setting, we are able to derive positive learnability results. Therefore, it might be possible to extend some of the techniques developed in our work to analyzing problems for other forms of structured data as well.

In relational learning, examples are expressed not as feature vectors, but as statements about the attributes of objects and the relationships between objects expressed in first-order logic. The expression of attributes and relations is accomplished using predicates. For example, given objects x_1 and x_2, attributes might be expressed as in $\texttt{attribute}_1(x_1)$, $\texttt{attribute}_2(x_1)$, and a relationship might be expressed as $\texttt{relationship}(x_1, x_2)$. Relationships might also be expressed using universal or existential quantification over a free variable. A relational learning dataset consists of a set of objects, attributes, and relations. The learning task might be to predict whether unseen objects have a particular attribute, which plays a role like a class label in supervised classification, given a set of known attributes and relations. Alternatively, in the relational learning setting, the task could be to predict the existence of a relationship between objects. Relational learning has many real-world applications, especially for learning from web and social network data. On social networks, relationships naturally exist between users, such as "friends" on Facebook, "connections" on LinkedIn, or "followers" on Twitter. Predicting when such relationships are likely to exist is important for recommending new friends, connections, or followers to users.

The supervised, MI, and relational learning settings possess strictly increasing representational power, as summarized in Figure 9.2. The results in Chapter 5 show that even though it might not be straightforward to learn accurate concepts from MI, it is possible to learn to rank bags and instances in the MI setting. Therefore, it might also be possible to learn to rank in the relational learning setting, under some weak assumptions, even when learning accurate concepts might be hard. Thus, our results for the MIL framework might provide a "bridge" between relatively easy supervised

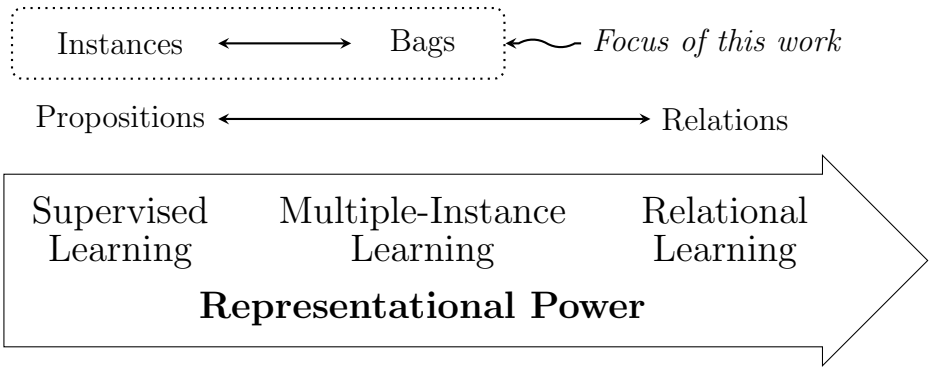


Figure 9.2: A summary of the relative representational power of the supervised, multiple-instance, and relational learning frameworks.

learning problems and generally challenging relational learning problems.

9.3 Conclusion

Although there are many future directions to explore, this work constitutes a significant contribution to MIL. The bags-as-distributions model described in Chapter 3 is straightforward, and it provides a more accurate description of the generative process for many real-world MI problems, as we support with empirical results. Furthermore, the generative model allows us to show *positive* results for learnability of instance- and bag-level concepts from MI data. For many practical scenarios, good models can be learned using standard supervised approaches, which are much more efficient, well-understood, and robust in practice than MI-specific approaches. The model can also be used to analyze MI-specific approaches such as SMILe, which can improve performance in scenarios for which supervised approaches perform poorly. Hence, we believe that our work contributes to an understanding of the MI setting in a way that will inform future research directions and impact the way that MIL is applied in practice.

Appendix A

Experiments and Results

The experiments used for this work were implemented in Python using NumPy (Ascher et al., 2001) and SciPy (Jones et al., 2001) for general matrix computations, the CVXOPT library (Dahl and Vandenberghe, 2009) for solving QPs, and scikit-learn (Pedregosa et al., 2011) for the SVM implementation used for the experiments in Section 6.4. We use the authors' original MATLAB code, found at `http://lamda.nju.edu.cn/code_KISVM.ashx`, for the KI-SVM approaches (Liu et al., 2012). Similarly, a Python wrapper around the original box-counting kernel implementation is used (Tao et al., 2008). Code for the remaining MI SVM implementations, the EMD, and other experiments is at `http://engr.case.edu/doran_gary/code.html`. This work made use of the High Performance Computing Resource in the Core Facility for Advanced Research Computing at Case Western Reserve University.

A.1 Datasets

The datasets used for the experiments above come from a variety of problem domains, listed in Table 2.1. For clarity, the 56 total datasets have been organized into 8 groups with similar properties. The properties of these groups are summarized in Table A.1. Since it is practically infeasible to perform *all* experiments using *all*

Table A.1: Dataset Groups

Domain	Group	Datasets	Features	Instance Labels?
3D-QSAR	Musk	2	166	
CBIR	Animal	3	230	
	Scene	3	15	
	SIVAL	12	30	✓
Text	Newsgroups	20	200	✓
	OHSUMED	2	6668–6842	
Audio	Birdsong	13	38	✓
Protein	TRX	1	8	

Table A.2: Dataset groups used in various experiments

Group	*Section 5.5*	*Section 7.3*	*Section 6.4*	*Section 8.7*
Musk	✓	✓	✓	✓
Animal	✓	✓	✓	✓
Scene	✓	✓	✓	✓
SIVAL	✓	✓	✓	✓
Newsgroups	✓	✓	✓	✓
OHSUMED	✓	✓	✓	✓
Birdsong	✓		✓	
TRX			✓	

datasets, some subsets of these groups are used in each experiment above. Table A.2 summarizes which dataset groups are used in different sets of experiments. More detailed properties of these datasets are described in the following sections.

Table A.3: 3D-QSAR Datasets

Group	Dataset	Bags	Instances
Musk	musk1	92	476
	musk2	102	6598

Table A.4: CBIR Datasets

Group	Index	Dataset	Bags	Instances
Animal		elephant	200	1391
		fox	200	1320
		tiger	200	1220
Scene		field	200	1800
		flower	200	1800
		mountain	200	1800
SIVAL	1	Apple vs. Coke Can	120	3789
	2	Banana vs. Gold Medal	120	3783
	3	Blue Scrunge vs. Ajax Orange	120	3780
	4	Cardboard Box vs. Candle with Holder	120	3791
	5	Checkered Scarf vs. Data Mining Book	120	3811
	6	Dirty Work Gloves vs. Dirty Running Shoe	120	3801
	7	Fabric Softener Box vs. Glazed Wood Pot	120	3796
	8	Julie's Pot vs. Rap Book	120	3793
	9	Smiley Face Doll vs. Felt Flower Rug	120	3799
	10	Striped Notebook vs. Green Tea Box	119	3776
	11	WD-40 Can vs. Large Spoon	120	3786
	12	Wood Rolling Pin vs. Translucent Bowl	120	3778

A.1.1 3D-QSAR Datasets

The two musk datasets taken from the UCI repository (Frank and Asuncion, 2010) are summarized in Table A.3. These datasets come from the original work on MIL (Dietterich et al., 1997), described in Chapter 1. The dataset consists of molecules which experts judge by smell to be either musky or not musky. A molecule smells musky if it activates a certain smell receptor. Each molecule (bag) is described as a set of low-energy conformations (instances). Each conformation is placed in a standard orientation, and described by features constructed by measuring the distance along a ray starting at a reference point to the surface of the conformation. The main difference between these two datasets are the number of molecules, and the number of conformations used to describe each molecule (there are more for the musk2 dataset).

A.1.2 CBIR Datasets

The CBIR datasets are listed in Table A.4. There are three groups of datasets comprised of animal, scene, and object images. The animal and scene images come from the Corel suite, and the others from the spatially independent, variable area, and lighting (SIVAL) dataset (Rahmani et al., 2005). All three groups of datasets are generated via different techniques, but each follow the same general template. First each image (bag) is segmented to form a set of instances. Then, each instance is described using a set of features derived from local colors or textures.

The scene datasets (Maron and Ratan, 1998) are segmented by moving a square "blob" across the image. The features of each blob are the mean RGB color values of pixels within the blob, as well as the differences in the mean color values in the four adjacent blobs above, below, and to the sides. As a result, $3 \times 5 = 15$ total features describe each instance.

The animal datasets (Andrews et al., 2003) are segmented using the *Blobworld* algorithm (Carson et al., 2002), which models the joint distribution over color, texture, and position features as a mixture of Gaussian distributions. The instances are then described using the features identified within each "blob."

Finally, the SIVAL datasets are constructed using the *ACCIO!* approach (Rahmani et al., 2005). After segmenting images, *ACCIO!* describes each segment using 3 color and 3 texture features for each segment and its 4 neighbors immediately above, below and to the sides, for a total of $6 \times 5 = 30$ features.

We use a version of the SIVAL dataset that has been manually annotated with instance-level labels (Settles et al., 2008). The original dataset contains 25 images classes, each corresponding to pictures of objects taken on various backgrounds. To create a set of smaller datasets, we randomly pair up images classes to create 12 one-vs.-one datasets, as shown in Table A.4. For brevity, we refer to these datasets in the results using the indices given in the second column.

Table A.5: Text Datasets

Group	Index	Dataset	Bags	Instances
Newsgroups	1	alt.atheism	100	5443
	2	comp.graphics	100	3094
	3	comp.os.ms-windows.misc	100	5175
	4	comp.sys.ibm.pc.hardware	100	4827
	5	comp.sys.mac.hardware	100	4473
	6	comp.windows.x	100	3110
	7	misc.forsale	100	5306
	8	rec.autos	100	3458
	9	rec.motorcycles	100	4730
	10	rec.sport.baseball	100	3358
	11	rec.sport.hockey	100	1982
	12	sci.crypt	100	4284
	13	sci.electronics	100	3192
	14	sci.med	100	3045
	15	sci.space	100	3655
	16	soc.religion.christian	100	4677
	17	talk.politics.guns	100	3558
	18	talk.politics.mideast	100	3376
	19	talk.politics.misc	100	4788
	20	talk.religion.misc	100	4606
OHSUMED	1	OHSUMED1	400	3224
	2	OHSUMED2	400	3344

A.1.3 Text Datasets

The text categorization datasets are listed in Table A.5. The semi-synthetic Newsgroups datasets (Settles et al., 2008) were generated by randomly sampling a number of posts from known online discussion topics. Hence, the labels of the instances are known for this dataset. Each post (instance) is represented using word frequency features describing the post.

The OHSUMED datasets are generated from the medical document classification task (Andrews et al., 2003). Each document (bag) is split into overlapping passages (instances) containing a maximum of 50 words each. As for the Newgroups datasets, these datasets are represented using word frequency features.

Table A.6: Audio Datasets

Group	Index	Dataset	Bags	Instances
Birdsong	1	Brown Creeper	548	10232
	2	Chestnut-backed Chickadee	548	10232
	3	Dark-eyed Junco	548	10232
	4	Hammond's Flycatcher	548	10232
	5	Hermit Thrush	548	10232
	6	Hermit Warbler	548	10232
	7	Olive-sided Flycatcher	548	10232
	8	Pacific-slope Flycatcher	548	10232
	9	Red-breasted Nuthatch	548	10232
	10	Swainson's Thrush	548	10232
	11	Varied Thrush	548	10232
	12	Western Tanager	548	10232
	13	Winter Wren	548	10232

A.1.4 Audio Datasets

The audio classification datasets in Table A.6 all come from the birdsong classification task (Briggs et al., 2012). The audio was recorded using unattended microphones in the H. J. Andrews Experimental Forest. A bag is 10 seconds of audio recording, and multiple labels of a bag correspond to the songs of the bird species present in the 10-second clip. A segmentation in time–frequency space is used to construct instances, each of which is described using the shape, frequency profile, and other statistics of the segment. The bird species corresponding to each segment has been manually annotated by experts.

Since there are multiple bird species present in each bag, the dataset is broken into 13 one-vs.-rest binary classification problems in which one species is the positive class, and the rest are considered the negative class.

A.1.5 Protein Dataset

There is one protein sequence classification dataset, TRX, summarized in Table A.7. The problem is to identify members of the Thioredoxin-fold "superfamily" of proteins

Table A.7: Protein Datasets

Group	Dataset	Bags	Instances
TRX	TRX	193	26611

that contain similar subsequences of amino acids (Wang et al., 2004). A sequence of amino acids corresponding to a protein (bag) is represented using properties of subsequences (instances) surrounding a central "motif" within the protein sequence. Since this dataset contains many instances, it is practically infeasible to use instance-based kernel classifiers. Thus, TRX is only used in experiments that compute bag-level kernels (see Table A.2).

A.2 Results

This section lists detailed numerical results for the experiments above. The results are organized by chapter.

A.2.1 Chapter 5 (Single-Instance Learning)

The full results for Section 5.5.4 are listed below. Table A.8 gives the instance-level accuracy results for Figure 5.3(a), Table A.9 gives the instance-level AUC results for Figure 5.3(c), Table A.10 shows the bag-level accuracy results for Figure 5.3(b), and Table A.11 shows the bag-level AUC results for Figure 5.3(d).

Table A.8: Instance-level accuracy results for Section 5.5.4. The best result is indicated in boldface.

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
SIVAL01	0.808	**0.964**	**0.964**	0.945	0.498
SIVAL02	0.509	**0.947**	0.933	0.935	0.926
SIVAL03	0.855	0.908	**0.953**	0.929	0.768
SIVAL04	0.734	**0.885**	0.828	0.881	0.884
SIVAL05	0.839	0.812	**0.896**	0.765	0.711
SIVAL06	0.811	0.872	0.865	0.872	**0.885**
SIVAL07	0.433	0.869	**0.880**	0.862	0.864

continued...

Table A.8: Instance-level accuracy results (continued).

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
SIVAL08	0.640	**0.899**	0.876	0.858	0.869
SIVAL09	0.685	**0.936**	0.919	0.919	0.917
SIVAL10	0.899	0.904	**0.955**	0.872	0.905
SIVAL11	0.789	0.894	**0.954**	0.867	0.411
SIVAL12	0.738	0.915	0.573	**0.938**	0.933
Newsgroups01	0.970	**0.992**	0.991	0.988	0.987
Newsgroups02	0.781	**0.985**	**0.985**	0.976	0.879
Newsgroups03	0.878	0.987	**0.991**	0.876	0.985
Newsgroups04	0.982	**0.990**	0.989	0.984	0.732
Newsgroups05	0.988	**0.991**	0.984	0.985	0.300
Newsgroups06	0.541	0.981	**0.982**	0.979	0.966
Newsgroups07	0.986	**0.990**	0.988	0.987	0.221
Newsgroups08	0.080	**0.990**	0.988	0.977	0.975
Newsgroups09	0.097	**0.992**	0.990	0.986	0.396
Newsgroups10	0.144	**0.989**	**0.989**	0.979	0.966
Newsgroups11	0.198	**0.985**	**0.985**	0.978	0.533
Newsgroups12	0.933	0.987	**0.988**	0.984	0.878
Newsgroups13	0.986	0.994	**0.996**	0.937	0.888
Newsgroups14	0.131	0.980	**0.986**	0.979	0.687
Newsgroups15	0.021	**0.989**	0.988	0.983	0.380
Newsgroups16	0.015	**0.990**	0.951	0.984	0.979
Newsgroups17	0.844	**0.986**	0.983	0.887	0.508
Newsgroups18	0.988	**0.993**	0.983	0.982	0.973
Newsgroups19	0.983	**0.989**	**0.989**	0.987	0.730
Newsgroups20	0.015	**0.987**	0.967	0.980	0.876
Birdsong01	0.931	**0.955**	**0.955**	0.946	0.941
Birdsong02	**0.977**	0.976	0.969	0.968	0.965
Birdsong03	**0.993**	**0.993**	0.992	0.992	0.992
Birdsong04	**0.971**	0.943	0.956	0.942	0.939
Birdsong05	**0.997**	**0.997**	**0.997**	**0.997**	**0.997**
Birdsong06	**0.988**	0.986	0.977	0.987	0.983
Birdsong07	**0.973**	0.967	0.965	0.970	**0.973**
Birdsong08	0.956	0.937	0.944	**0.962**	0.955
Birdsong09	0.961	0.956	**0.971**	0.950	0.960
Birdsong10	0.988	**0.989**	**0.989**	0.986	0.983
Birdsong11	0.994	0.993	**0.995**	0.993	0.986
Birdsong12	**0.993**	0.991	0.989	0.985	0.987
Birdsong13	0.900	0.928	**0.930**	0.926	0.921

Table A.9: Instance-level AUC results for Section 5.5.4. The best result is indicated in boldface.

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
SIVAL01	0.758	0.872	0.836	0.758	**0.898**
SIVAL02	**0.867**	0.841	0.782	0.761	0.815
SIVAL03	0.676	0.588	0.795	0.690	**0.934**
SIVAL04	0.647	0.651	**0.859**	0.595	0.836
SIVAL05	0.954	0.810	**0.961**	0.906	0.754
SIVAL06	0.619	0.489	0.603	0.516	**0.703**
SIVAL07	0.895	0.784	**0.903**	0.780	0.725
SIVAL08	**0.868**	0.852	0.768	0.556	0.759
SIVAL09	**0.829**	0.730	0.824	0.771	0.581
SIVAL10	0.882	0.788	**0.948**	0.686	0.721
SIVAL11	**0.965**	0.795	0.952	0.746	0.600
SIVAL12	0.566	0.541	0.515	**0.690**	0.623
Newsgroups01	**0.980**	0.953	0.968	0.834	0.584
Newsgroups02	**0.904**	0.899	0.864	0.850	0.572
Newsgroups03	**0.866**	0.783	0.782	0.686	0.576
Newsgroups04	**0.923**	0.883	0.885	0.846	0.612
Newsgroups05	**0.951**	0.922	0.906	0.796	0.537
Newsgroups06	0.946	**0.948**	0.895	0.824	0.587
Newsgroups07	**0.907**	0.853	0.835	0.827	0.604
Newsgroups08	0.753	0.881	**0.909**	0.808	0.551
Newsgroups09	0.711	0.962	**0.979**	0.869	0.560
Newsgroups10	0.660	**0.947**	0.908	0.746	0.565
Newsgroups11	0.728	0.971	**0.980**	0.968	0.702
Newsgroups12	0.958	**0.961**	0.942	0.767	0.536
Newsgroups13	**0.970**	0.939	0.911	0.920	0.608
Newsgroups14	0.823	**0.903**	0.884	0.902	0.614
Newsgroups15	0.736	0.949	**0.955**	0.930	0.553
Newsgroups16	0.454	0.938	0.906	**0.940**	0.600
Newsgroups17	**0.946**	0.913	0.921	0.854	0.565
Newsgroups18	**0.964**	0.914	0.922	0.797	0.605
Newsgroups19	**0.931**	0.914	0.826	0.803	0.558
Newsgroups20	0.573	0.884	**0.914**	0.912	0.556
Birdsong01	0.762	**0.925**	0.907	0.704	0.708
Birdsong02	**0.895**	0.884	0.849	0.574	0.748
Birdsong03	**0.782**	0.729	0.673	0.636	0.599
Birdsong04	**0.966**	0.927	0.932	0.905	0.858
Birdsong05	**0.686**	0.439	0.641	0.422	0.498
Birdsong06	0.627	**0.741**	0.581	0.540	0.719
Birdsong07	0.782	0.570	**0.857**	0.441	0.814
Birdsong08	**0.836**	0.615	0.796	0.552	0.774

continued...

Table A.9: Instance-level AUC results (continued).

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
Birdsong09	0.920	**0.940**	0.915	0.889	0.702
Birdsong10	0.858	0.859	**0.879**	0.763	0.757
Birdsong11	**0.989**	0.971	0.970	0.982	0.712
Birdsong12	**0.954**	0.907	0.918	0.490	0.745
Birdsong13	0.799	0.640	**0.806**	0.605	0.589

Table A.10: Bag-level accuracy results for Section 5.5.4. The best result is indicated in boldface.

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
musk1	0.848	0.772	0.848	**0.870**	0.511
musk2	0.804	**0.833**	0.647	0.755	0.696
elephant	0.740	0.815	0.750	**0.830**	0.680
fox	**0.645**	0.585	0.590	0.540	0.550
tiger	0.780	**0.785**	0.770	0.725	0.535
field	0.725	**0.805**	0.760	0.700	0.650
flower	0.775	**0.815**	0.800	0.810	0.780
mountain	0.830	0.855	**0.860**	0.790	0.615
SIVAL01	0.558	**0.875**	0.850	0.600	0.617
SIVAL02	0.542	**0.867**	0.758	0.708	0.558
SIVAL03	0.750	0.667	**0.817**	0.642	0.500
SIVAL04	0.583	0.658	0.525	0.583	**0.700**
SIVAL05	0.650	**0.917**	0.908	0.842	0.658
SIVAL06	0.650	0.600	0.733	0.650	**0.833**
SIVAL07	0.517	0.950	0.917	0.950	**0.967**
SIVAL08	0.542	**0.933**	0.775	0.450	0.725
SIVAL09	0.533	**0.900**	0.833	0.817	0.733
SIVAL10	0.655	0.807	0.798	0.714	**0.840**
SIVAL11	0.575	**0.975**	0.892	0.783	0.583
SIVAL12	0.583	0.533	0.500	0.683	**0.808**
Newsgroups01	0.540	**0.870**	0.840	0.580	0.500
Newsgroups02	0.720	**0.820**	0.760	0.470	0.490
Newsgroups03	0.650	0.640	**0.740**	0.470	0.480
Newsgroups04	0.670	0.750	**0.770**	0.480	0.480
Newsgroups05	0.770	**0.790**	0.740	0.690	0.480
Newsgroups06	0.630	0.780	**0.790**	0.700	0.480
Newsgroups07	0.690	0.750	**0.770**	0.540	0.500
Newsgroups08	0.500	**0.800**	0.790	0.570	0.540
Newsgroups09	0.500	**0.860**	0.850	0.600	0.470
Newsgroups10	0.500	0.830	**0.850**	0.620	0.560
Newsgroups11	0.500	**0.890**	**0.890**	0.720	0.500
Newsgroups12	0.570	**0.790**	0.720	0.520	0.460

continued...

Table A.10: Bag-level accuracy results (continued).

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
Newsgroups13	0.820	0.880	**0.940**	0.510	0.520
Newsgroups14	0.510	**0.830**	0.820	0.570	0.490
Newsgroups15	0.500	0.800	**0.860**	0.620	0.510
Newsgroups16	0.500	**0.780**	0.530	0.560	0.560
Newsgroups17	0.660	**0.720**	0.700	0.640	0.450
Newsgroups18	0.700	**0.860**	0.700	0.650	0.490
Newsgroups19	0.660	**0.690**	0.660	0.620	0.500
Newsgroups20	0.490	**0.720**	0.650	0.630	0.510
OHSUMED1	0.728	0.650	**0.762**	0.670	0.627
OHSUMED2	**0.637**	0.552	0.595	0.520	0.500
Birdsong01	0.726	0.912	**0.916**	0.823	0.704
Birdsong02	0.865	**0.880**	0.830	0.819	0.818
Birdsong03	0.976	**0.984**	0.964	0.956	0.964
Birdsong04	0.929	0.995	0.991	**0.998**	0.996
Birdsong05	**0.974**	**0.974**	**0.974**	0.973	**0.974**
Birdsong06	0.894	0.914	**0.947**	0.916	0.925
Birdsong07	0.841	**0.896**	0.891	0.849	0.841
Birdsong08	0.790	0.847	**0.885**	0.854	0.792
Birdsong09	0.945	0.903	**0.947**	0.858	0.940
Birdsong10	0.940	**0.956**	0.953	0.934	0.885
Birdsong11	**0.978**	0.960	**0.978**	0.974	0.858
Birdsong12	**0.978**	0.969	0.949	0.925	0.922
Birdsong13	0.945	0.923	0.949	**0.971**	0.801

Table A.11: Bag-level AUC results for Section 5.5.4. The best result is indicated in boldface.

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
musk1	0.922	0.845	**0.943**	0.836	0.849
musk2	0.897	**0.949**	0.661	0.665	0.913
elephant	**0.919**	0.912	0.916	0.676	0.871
fox	**0.662**	0.589	0.632	0.500	0.615
tiger	**0.859**	0.856	0.853	0.673	0.688
field	**0.923**	0.871	0.908	0.687	0.847
flower	0.907	0.873	**0.921**	0.810	0.759
mountain	0.916	0.915	**0.935**	0.759	0.830
SIVAL01	0.626	**0.954**	0.875	0.643	0.933
SIVAL02	0.826	**0.952**	0.731	0.747	0.863
SIVAL03	0.785	0.666	0.716	0.708	**0.906**
SIVAL04	0.657	0.683	0.831	0.697	**0.957**
SIVAL05	0.985	0.964	**1.000**	0.938	0.914
SIVAL06	0.648	0.756	0.753	0.542	**0.918**

continued...

Table A.11: Bag-level AUC results (continued).

Dataset	SIL	MI-SVM	mi-SVM	KI-SVM	MICA
SIVAL07	0.793	**0.993**	0.972	0.969	0.974
SIVAL08	0.874	**0.998**	0.812	0.488	0.865
SIVAL09	0.907	**0.979**	0.822	0.768	0.709
SIVAL10	0.819	0.772	**0.930**	0.643	0.785
SIVAL11	0.981	**1.000**	0.987	0.817	0.736
SIVAL12	0.601	0.621	0.516	0.692	**0.710**
Newsgroups01	0.928	**0.931**	0.870	0.746	0.535
Newsgroups02	0.873	0.794	**0.878**	0.826	0.538
Newsgroups03	0.755	0.715	**0.805**	0.640	0.517
Newsgroups04	0.765	**0.767**	0.727	0.631	0.511
Newsgroups05	0.776	**0.842**	0.760	0.800	0.538
Newsgroups06	0.814	**0.862**	0.837	0.741	0.521
Newsgroups07	0.802	0.798	0.789	**0.844**	0.576
Newsgroups08	0.674	0.810	**0.837**	0.759	0.532
Newsgroups09	0.728	**0.925**	0.918	0.784	0.552
Newsgroups10	0.752	**0.904**	0.900	0.696	0.550
Newsgroups11	0.709	**0.975**	0.957	0.816	0.669
Newsgroups12	0.844	0.805	**0.858**	0.695	0.530
Newsgroups13	**0.971**	0.911	0.914	0.930	0.602
Newsgroups14	0.648	0.825	0.861	**0.869**	0.600
Newsgroups15	0.742	0.886	0.913	**0.928**	0.555
Newsgroups16	0.564	**0.860**	0.538	0.838	0.543
Newsgroups17	0.630	**0.787**	0.751	0.674	0.533
Newsgroups18	0.864	**0.874**	0.797	0.771	0.558
Newsgroups19	0.754	**0.812**	0.745	0.640	0.548
Newsgroups20	0.575	**0.807**	0.757	0.770	0.520
OHSUMED1	**0.958**	0.914	0.954	0.779	0.816
OHSUMED2	0.740	0.638	**0.775**	0.520	0.715
Birdsong01	0.894	0.974	**0.976**	0.807	0.824
Birdsong02	**0.902**	0.895	0.895	0.570	0.827
Birdsong03	0.908	**0.916**	0.909	0.747	0.755
Birdsong04	**0.999**	0.998	0.989	0.992	0.945
Birdsong05	0.664	0.623	0.542	0.529	**0.699**
Birdsong06	0.926	0.915	0.964	0.728	**0.975**
Birdsong07	0.931	0.928	**0.934**	0.617	0.919
Birdsong08	0.913	0.901	**0.921**	0.723	0.908
Birdsong09	**0.984**	0.941	0.962	0.897	0.787
Birdsong10	0.947	**0.975**	0.956	0.819	0.862
Birdsong11	0.991	0.988	0.991	**0.995**	0.803
Birdsong12	**0.996**	0.961	0.993	0.609	0.737
Birdsong13	0.980	0.948	**0.985**	0.885	0.780

A.2.2 Chapter 6 (Bag-Level Hyperplane Classifiers)

For Section 6.4, which evaluates bag-level kernels, the bag-level accuracy results in Figure 6.1(a) are listed in Table A.12. Similarly, the bag-level AUC results are shown in Table A.13.

Table A.12: Bag-level accuracy results for Section 6.4. The best result is indicated in boldface.

Dataset	EMD	Level-2	NSK	mi-Graph	YARDS	Box
musk1	**0.924**	0.837	0.826	0.837	0.880	0.848
musk2	0.853	**0.882**	0.863	0.853	0.853	0.814
elephant	**0.900**	0.840	0.840	0.855	0.810	0.835
fox	**0.670**	0.605	0.640	0.580	0.570	0.645
tiger	**0.865**	0.840	0.835	0.810	0.795	0.735
field	**0.840**	0.820	0.825	0.825	0.815	0.810
flower	0.850	**0.865**	0.805	0.850	0.800	0.810
mountain	**0.880**	0.860	0.855	0.855	0.865	0.825
SIVAL01	0.883	**0.917**	0.875	0.842	0.850	0.792
SIVAL02	**0.958**	0.917	0.892	0.858	0.733	0.658
SIVAL03	**0.950**	**0.950**	0.942	0.925	0.858	0.783
SIVAL04	0.975	**0.992**	**0.992**	0.792	0.933	0.783
SIVAL05	**0.992**	0.983	**0.992**	0.942	0.983	0.942
SIVAL06	**0.950**	0.942	0.933	0.883	0.892	0.883
SIVAL07	0.942	0.883	0.875	**0.967**	0.900	0.917
SIVAL08	**0.933**	0.908	0.908	0.775	0.767	0.817
SIVAL09	0.942	**0.967**	0.925	0.892	0.908	0.892
SIVAL10	**0.992**	0.958	0.950	0.882	0.874	0.857
SIVAL11	**0.958**	0.950	**0.958**	0.933	0.892	0.783
SIVAL12	**0.925**	0.917	0.892	0.817	0.825	0.800
Newsgroups01	0.790	0.820	0.820	**0.840**	0.810	0.660
Newsgroups02	0.760	0.790	0.770	0.780	**0.800**	0.600
Newsgroups03	**0.730**	0.690	0.610	**0.730**	0.710	0.570
Newsgroups04	**0.820**	0.730	0.750	0.720	0.800	0.490
Newsgroups05	0.810	0.790	0.810	0.770	**0.820**	0.610
Newsgroups06	0.830	0.730	0.760	0.820	**0.840**	0.630
Newsgroups07	**0.770**	0.650	0.640	0.700	0.670	0.630
Newsgroups08	0.760	0.740	0.630	**0.790**	0.760	0.590
Newsgroups09	**0.880**	0.810	0.770	0.810	0.830	0.610
Newsgroups10	**0.840**	0.820	0.810	0.790	0.800	0.740
Newsgroups11	**0.880**	0.860	0.860	0.850	0.840	0.850
Newsgroups12	**0.740**	0.680	0.670	0.720	0.720	0.590
Newsgroups13	**0.920**	**0.920**	0.860	0.860	0.910	0.520

continued...

Table A.12: Bag-level accuracy results for bag-level kernels (continued).

Dataset	EMD	Level-2	NSK	mi-Graph	YARDS	Box
Newsgroups14	0.820	0.760	0.800	**0.840**	0.820	0.600
Newsgroups15	0.860	0.850	**0.880**	0.830	0.840	0.730
Newsgroups16	**0.770**	0.760	0.680	0.760	**0.770**	0.640
Newsgroups17	0.750	**0.780**	0.730	0.750	0.770	0.590
Newsgroups18	0.790	0.790	0.760	0.760	**0.820**	0.600
Newsgroups19	0.720	0.680	0.720	0.730	**0.750**	0.630
Newsgroups20	0.730	**0.770**	0.610	0.690	0.720	0.480
OHSUMED1	0.792	**0.892**	0.887	0.868	0.865	0.500
OHSUMED2	0.677	0.688	**0.695**	0.677	0.693	0.500
Birdsong01	**0.905**	0.892	0.894	0.891	0.889	0.867
Birdsong02	0.905	0.901	0.867	0.905	0.900	**0.909**
Birdsong03	0.964	0.969	0.965	0.960	0.964	**0.971**
Birdsong04	0.996	**0.998**	**0.998**	0.996	**0.998**	**0.998**
Birdsong05	0.974	**0.976**	0.974	0.974	0.974	0.974
Birdsong06	**0.949**	0.945	**0.949**	0.934	0.943	0.943
Birdsong07	**0.942**	0.938	0.934	0.931	0.920	0.907
Birdsong08	0.907	**0.909**	0.874	0.892	0.903	0.827
Birdsong09	**0.962**	0.947	0.953	0.931	0.954	0.949
Birdsong10	**0.974**	0.962	0.960	0.947	0.947	0.958
Birdsong11	**0.982**	0.974	0.971	0.978	0.978	0.969
Birdsong12	**0.980**	**0.980**	0.976	0.949	0.969	0.969
Birdsong13	0.965	**0.980**	0.969	0.973	0.956	0.969
TRX	0.891	0.865	0.850	**0.902**	0.865	0.865

Table A.13: Bag-level AUC results for Section 6.4. The best result is indicated in boldface.

Dataset	EMD	Level-2	NSK	mi-Graph	YARDS	Box
musk1	**0.968**	0.933	0.944	0.939	0.957	0.930
musk2	0.956	**0.961**	0.948	0.841	0.918	0.926
elephant	**0.965**	0.909	0.902	0.912	0.904	0.916
fox	0.697	0.680	0.704	0.669	0.626	**0.723**
tiger	**0.936**	0.908	0.889	0.862	0.860	0.869
field	0.921	0.917	**0.928**	0.921	0.917	0.906
flower	**0.936**	0.932	0.857	0.920	0.906	0.898
mountain	**0.909**	0.902	0.900	0.878	0.886	0.896
SIVAL01	**0.966**	0.956	0.951	0.918	0.904	0.884
SIVAL02	**0.993**	0.978	0.971	0.909	0.803	0.711
SIVAL03	**0.995**	0.988	0.988	0.987	0.930	0.894
SIVAL04	0.998	**0.999**	**0.999**	0.905	0.990	0.862
SIVAL05	**1.000**	0.999	**1.000**	0.999	0.999	0.988
SIVAL06	0.986	**0.992**	0.977	0.955	0.954	0.938

continued...

Table A.13: Bag-level AUC results for bag-level kernels (continued).

Dataset	EMD	Level-2	NSK	mi-Graph	YARDS	Box
SIVAL07	**0.993**	0.979	0.961	0.990	0.961	0.959
SIVAL08	**0.977**	0.951	0.973	0.783	0.851	0.897
SIVAL09	**0.980**	0.977	0.970	0.976	0.924	0.963
SIVAL10	**1.000**	0.997	0.997	0.980	0.957	0.966
SIVAL11	0.991	0.984	**0.996**	0.969	0.933	0.878
SIVAL12	**0.969**	0.965	**0.969**	0.898	0.879	0.851
Newsgroups01	0.859	0.877	**0.885**	0.872	0.861	0.712
Newsgroups02	0.867	0.874	0.868	**0.897**	0.879	0.556
Newsgroups03	**0.814**	0.760	0.630	0.651	0.772	0.569
Newsgroups04	0.830	**0.838**	0.809	0.759	0.833	0.606
Newsgroups05	0.853	**0.857**	0.821	0.843	0.835	0.578
Newsgroups06	**0.894**	0.856	0.817	0.849	0.882	0.679
Newsgroups07	0.758	0.680	0.606	**0.783**	0.780	0.638
Newsgroups08	0.784	0.698	0.701	0.798	**0.840**	0.665
Newsgroups09	**0.892**	0.707	0.723	0.849	0.866	0.674
Newsgroups10	0.846	**0.919**	0.831	0.883	0.852	0.790
Newsgroups11	0.831	0.916	0.883	**0.943**	0.916	0.874
Newsgroups12	0.809	0.674	0.675	**0.845**	0.795	0.642
Newsgroups13	0.934	0.762	**0.974**	0.838	0.925	0.428
Newsgroups14	0.775	0.743	0.728	**0.865**	0.856	0.650
Newsgroups15	0.901	0.850	**0.950**	0.856	0.919	0.766
Newsgroups16	0.826	0.820	0.738	0.822	**0.830**	0.706
Newsgroups17	0.764	0.776	**0.817**	0.765	0.793	0.667
Newsgroups18	0.846	0.824	0.855	**0.892**	0.831	0.663
Newsgroups19	0.718	0.772	0.724	0.735	**0.818**	0.603
Newsgroups20	0.804	**0.855**	0.624	0.816	0.790	0.463
OHSUMED1	0.885	0.955	**0.967**	0.944	0.919	0.500
OHSUMED2	0.727	0.789	**0.793**	0.761	0.760	0.500
Birdsong01	**0.962**	0.961	0.951	0.939	0.945	0.932
Birdsong02	**0.933**	0.909	0.865	0.883	0.919	0.927
Birdsong03	0.913	**0.977**	0.959	0.902	0.856	0.927
Birdsong04	**1.000**	0.984	**1.000**	**1.000**	**1.000**	0.987
Birdsong05	**0.955**	0.770	0.753	0.873	0.714	0.829
Birdsong06	**0.982**	0.976	0.979	**0.982**	0.973	0.973
Birdsong07	**0.984**	0.979	0.974	0.967	0.971	0.965
Birdsong08	**0.961**	0.953	0.921	0.945	0.952	0.898
Birdsong09	**0.984**	0.979	0.969	0.978	0.952	0.948
Birdsong10	0.911	0.982	0.963	0.973	0.980	**0.985**
Birdsong11	**0.998**	0.997	0.992	0.987	0.990	0.993
Birdsong12	0.969	0.983	**0.993**	0.964	0.986	0.986
Birdsong13	0.997	**0.998**	0.994	0.996	0.993	0.996

continued...

Table A.13: Bag-level AUC results for bag-level kernels (continued).

Dataset	EMD	Level-2	NSK	mi-Graph	YARDS	Box
TRX	**0.899**	0.887	0.760	0.872	0.833	0.873

A.2.3 Chapter 7 (Instance-Level Hyperplane Classifiers)

The results for Chapter 7 are shown below. In Section 7.3, which explores instance-level hyperplane classifiers, the instance-level accuracy results summarized in Figure 7.6(a) are shown in Table A.14 and Table A.15. Due to space limitations, the 12 algorithms analyzed are split across two tables. Similarly, the bag-level accuracy results summarized in Figure 7.6(b) are split across Table A.16 and Table A.17.

Table A.14: Instance-level accuracy for Section 7.3. The best result is indicated in boldface.

Dataset	SIL	MICA	sMIL	stMIL	sbMIL
SIVAL01	0.808	0.498	0.948	0.948	**0.971** ...
SIVAL02	0.509	0.926	0.935	0.935	0.939 **...**
SIVAL03	0.855	0.768	0.945	0.945	**0.976** ...
SIVAL04	0.734	0.884	0.877	0.877	**0.888** ...
SIVAL05	0.839	0.711	0.732	0.732	0.805 **...**
SIVAL06	0.811	0.885	0.877	0.877	**0.887** ...
SIVAL07	0.433	0.864	0.810	0.810	0.865 **...**
SIVAL08	0.640	0.869	0.863	0.863	**0.922** ...
SIVAL09	0.685	0.917	0.911	0.911	**0.942** ...
SIVAL10	0.899	0.905	0.866	0.866	0.917 **...**
SIVAL11	0.789	0.411	0.846	0.846	0.908 **...**
SIVAL12	0.738	0.933	0.936	0.936	0.923 **...**
Newsgroups01	0.970	0.987	0.987	0.987	0.990 **...**
Newsgroups02	0.781	0.879	0.984	0.979	**0.989** ...
Newsgroups03	0.878	0.985	0.987	0.987	0.988 **...**
Newsgroups04	0.982	0.732	0.986	0.986	0.988 **...**
Newsgroups05	0.988	0.300	0.984	0.984	0.989 **...**
Newsgroups06	0.541	0.966	0.977	0.977	**0.982 ...**
Newsgroups07	0.986	0.221	0.988	0.987	0.987 **...**
Newsgroups08	0.080	0.975	0.977	0.980	0.987 **...**
Newsgroups09	0.097	0.396	0.985	0.985	0.989 **...**
Newsgroups10	0.144	0.966	0.980	0.980	0.988 **...**

continued...

Table A.14: Instance-level accuracy for Section 7.3 (continued).

Dataset	SIL	MICA	sMIL	stMIL	sbMIL
Newsgroups11	0.198	0.533	0.966	0.966	**0.986** ...
Newsgroups12	0.933	0.878	0.984	0.984	**0.989** ...
Newsgroups13	0.986	0.888	0.981	0.981	**0.997** ...
Newsgroups14	0.131	0.687	0.982	0.978	0.984 **...**
Newsgroups15	0.021	0.380	0.980	0.980	0.987 **...**
Newsgroups16	0.015	0.979	0.985	0.988	0.989 **...**
Newsgroups17	0.844	0.508	0.981	0.981	0.980 **...**
Newsgroups18	0.988	0.973	0.980	0.980	0.990 **...**
Newsgroups19	0.983	0.730	0.986	0.986	**0.989 ...**
Newsgroups20	0.015	0.876	0.986	0.986	0.984 **...**

Table A.15: Instance-level accuracy for Section 7.3 (continued). The best result is indicated in boldface.

Dataset		MI-SVM	mi-SVM	I-KI-SVM	B-KI-SVM	NSK
SIVAL01	**...**	0.964	0.964	0.945	0.952	0.456
SIVAL02	...	**0.947**	0.933	0.935	0.890	0.540
SIVAL03	**...**	0.908	0.953	0.929	0.947	0.482
SIVAL04	**...**	0.885	0.828	0.881	0.883	0.598
SIVAL05	...	0.812	**0.896**	0.765	0.787	0.795
SIVAL06	**...**	0.872	0.865	0.872	0.886	0.489
SIVAL07	...	0.869	**0.880**	0.862	0.867	0.756
SIVAL08	**...**	0.899	0.876	0.858	0.859	0.596
SIVAL09	**...**	0.936	0.919	0.919	0.919	0.361
SIVAL10	...	0.904	**0.955**	0.872	0.875	0.547
SIVAL11	...	0.894	**0.954**	0.867	0.868	0.778
SIVAL12	...	0.915	0.573	**0.938**	0.931	0.517
Newsgroups01	...	**0.992**	0.991	0.988	0.986	0.864
Newsgroups02	**...**	0.985	0.985	0.976	0.879	0.918
Newsgroups03	...	0.987	**0.991**	0.876	0.981	0.902
Newsgroups04	...	**0.990**	0.989	0.984	0.983	0.875
Newsgroups05	...	**0.991**	0.984	0.985	0.980	0.910
Newsgroups06	**...**	0.981	**0.982**	0.979	0.977	0.856
Newsgroups07	...	**0.990**	0.988	0.987	0.978	0.817
Newsgroups08	...	**0.990**	0.988	0.977	0.977	0.904
Newsgroups09	...	**0.992**	0.990	0.986	0.988	0.876
Newsgroups10	...	**0.989**	**0.989**	0.979	0.979	0.892
Newsgroups11	**...**	0.985	0.985	0.978	0.984	0.896
Newsgroups12	**...**	0.987	0.988	0.984	0.986	0.858
Newsgroups13	**...**	0.994	0.996	0.937	0.977	0.913
Newsgroups14	...	0.980	**0.986**	0.979	0.983	0.832
Newsgroups15	...	**0.989**	0.988	0.983	0.984	0.850

continued...

Table A.15: Instance-level accuracy for Section 7.3 (continued).

Dataset		MI-SVM	mi-SVM	I-KI-SVM	B-KI-SVM	NSK
Newsgroups16	...	**0.990**	0.951	0.984	0.981	0.874
Newsgroups17	...	**0.986**	0.983	0.887	0.982	0.860
Newsgroups18	...	**0.993**	0.983	0.982	0.987	0.883
Newsgroups19	**...**	**0.989**	**0.989**	0.987	0.984	0.849
Newsgroups20	...	**0.987**	0.967	0.980	0.900	0.841

Table A.16: Bag-level accuracy for Section 7.3. The best result is indicated in boldface.

Dataset	SIL	MICA	sMIL	stMIL	sbMIL
musk1	0.848	0.511	0.750	0.728	0.859 **...**
musk2	0.804	0.696	0.608	0.618	**0.843** ...
elephant	0.740	0.680	0.520	0.590	0.810 **...**
fox	**0.645**	0.550	0.520	0.515	0.600 ...
tiger	0.780	0.535	0.590	0.615	0.800 **...**
field	0.725	0.650	0.500	0.500	0.810 **...**
flower	0.775	0.780	0.500	0.500	0.805 **...**
mountain	0.830	0.615	0.500	0.500	0.815 **...**
SIVAL01	0.558	0.617	0.500	0.500	**0.883** ...
SIVAL02	0.542	0.558	0.500	0.500	0.808 **...**
SIVAL03	0.750	0.500	0.500	0.500	**0.925** ...
SIVAL04	0.583	0.700	0.500	0.500	0.817 **...**
SIVAL05	0.650	0.658	0.500	0.500	**0.975** ...
SIVAL06	0.650	0.833	0.500	0.500	0.750 **...**
SIVAL07	0.517	**0.967**	0.500	0.500	0.958 **...**
SIVAL08	0.542	0.725	0.500	0.500	0.908 **...**
SIVAL09	0.533	0.733	0.500	0.500	**0.925** ...
SIVAL10	0.655	0.840	0.504	0.504	0.899 **...**
SIVAL11	0.575	0.583	0.500	0.500	0.967 **...**
SIVAL12	0.583	0.808	0.500	0.500	0.700 **...**
Newsgroups01	0.540	0.500	0.500	0.500	0.830 **...**
Newsgroups02	0.720	0.490	0.510	0.510	**0.820 ...**
Newsgroups03	0.650	0.480	0.500	0.500	0.720 **...**
Newsgroups04	0.670	0.480	0.510	0.510	0.680 **...**
Newsgroups05	0.770	0.480	0.500	0.500	0.770 **...**
Newsgroups06	0.630	0.480	0.510	0.510	0.760 **...**
Newsgroups07	0.690	0.500	0.500	0.500	0.730 **...**
Newsgroups08	0.500	0.540	0.500	0.500	**0.810** ...
Newsgroups09	0.500	0.470	0.500	0.500	0.820 **...**
Newsgroups10	0.500	0.560	0.500	0.500	0.840 **...**
Newsgroups11	0.500	0.500	0.500	0.500	0.840 **...**
Newsgroups12	0.570	0.460	0.500	0.500	0.770 **...**

continued...

Table A.16: Bag-level accuracy for Section 7.3 (continued).

Dataset	SIL	MICA	sMIL	stMIL	sbMIL
Newsgroups13	0.820	0.520	0.530	0.530	**0.950** ...
Newsgroups14	0.510	0.490	0.500	0.500	**0.830 ...**
Newsgroups15	0.500	0.510	0.500	0.500	0.800 **...**
Newsgroups16	0.500	0.560	0.500	0.500	0.720 **...**
Newsgroups17	0.660	0.450	0.500	0.500	0.730 **...**
Newsgroups18	0.700	0.490	0.500	0.500	0.800 **...**
Newsgroups19	0.660	0.500	0.500	0.500	**0.740** ...
Newsgroups20	0.490	0.510	0.510	0.510	0.710 **...**
OHSUMED1	0.728	0.627	0.500	0.500	**0.932** ...
OHSUMED2	0.637	0.500	0.500	0.500	0.618 **...**

Table A.17: Bag-level accuracy for Section 7.3. The best result is indicated in boldface.

Dataset		MI-SVM	mi-SVM	I-KI-SVM	B-KI-SVM	NSK
musk1	...	0.772	0.848	0.870	0.815	**0.913**
musk2	**...**	0.833	0.647	0.755	0.745	0.814
elephant	...	0.815	0.750	0.830	0.690	**0.845**
fox	**...**	0.585	0.590	0.540	0.555	0.615
tiger	...	0.785	0.770	0.725	0.760	**0.825**
field	...	0.805	0.760	0.700	0.745	**0.815**
flower	...	**0.815**	0.800	0.810	0.765	0.740
mountain	...	0.855	**0.860**	0.790	0.795	0.845
SIVAL01	**...**	0.875	0.850	0.600	0.758	0.842
SIVAL02	...	0.867	0.758	0.708	0.575	**0.908**
SIVAL03	**...**	0.667	0.817	0.642	0.783	0.825
SIVAL04	...	0.658	0.525	0.583	0.750	**0.883**
SIVAL05	**...**	0.917	0.908	0.842	0.942	0.958
SIVAL06	...	0.600	0.733	0.650	0.800	**0.908**
SIVAL07	**...**	0.950	0.917	0.950	**0.967**	0.908
SIVAL08	...	**0.933**	0.775	0.450	0.675	0.800
SIVAL09	**...**	0.900	0.833	0.817	0.875	0.867
SIVAL10	...	0.807	0.798	0.714	0.731	**0.966**
SIVAL11	...	**0.975**	0.892	0.783	0.742	0.933
SIVAL12	...	0.533	0.500	0.683	0.683	**0.933**
Newsgroups01	...	**0.870**	0.840	0.580	0.750	0.850
Newsgroups02	**...**	**0.820**	0.760	0.470	0.510	0.770
Newsgroups03	...	0.640	**0.740**	0.470	0.580	0.730
Newsgroups04	...	0.750	**0.770**	0.480	0.570	0.750
Newsgroups05	...	**0.790**	0.740	0.690	0.620	0.760
Newsgroups06	...	0.780	0.790	0.700	0.720	**0.800**
Newsgroups07	...	0.750	**0.770**	0.540	0.580	0.690

continued...

Table A.17: Bag-level accuracy for Section 7.3 (continued).

Dataset	MI-SVM	mi-SVM	I-KI-SVM	B-KI-SVM	NSK
Newsgroups08 ...	0.800	0.790	0.570	0.630	0.720
Newsgroups09 ...	**0.860**	0.850	0.600	0.720	0.830
Newsgroups10 ...	0.830	**0.850**	0.620	0.710	0.800
Newsgroups11 ...	**0.890**	**0.890**	0.720	0.820	0.830
Newsgroups12 ...	**0.790**	0.720	0.520	0.630	0.730
Newsgroups13 ...	0.880	0.940	0.510	0.580	0.920
Newsgroups14 ...	**0.830**	0.820	0.570	0.660	0.820
Newsgroups15 ...	0.800	0.860	0.620	0.660	**0.880**
Newsgroups16 ...	**0.780**	0.530	0.560	0.660	**0.780**
Newsgroups17 ...	0.720	0.700	0.640	0.590	**0.750**
Newsgroups18 ...	**0.860**	0.700	0.650	0.770	0.800
Newsgroups19 ...	0.690	0.660	0.620	0.630	0.710
Newsgroups20 ...	0.720	0.650	0.630	0.600	**0.730**
OHSUMED1 ...	0.650	0.762	0.670	0.642	0.885
OHSUMED2 ...	0.552	0.595	0.520	0.565	**0.682**

A.2.4 Chapter 8 (SMILe)

Table A.18 shows the results using SMILe with MI-SVM$_{\text{I}}$ by explicitly adding 25 or 50 shuffled positive bags, or directly using the SMILeSVM formulation. Similarly, Table A.19 shows the results using SMILe with the NSK by explicitly adding 25 or 50 shuffled positive bags, or directly using the CC-NSK formulation.

Table A.18: Comparison of performance (instance-level balanced accuracy) of to with SMILe using 25 and 50 shuffled bags and . In all cases $\epsilon_s = 10\%$. Boldface indicates when an approach outperforms , and the final row indicates significant differences ($\alpha = 0.05$) from across datasets.

Dataset	MI-SVM$_{\text{I}}$	MI-SVM$_{\text{I}}$ + SMILe (25)	MI-SVM$_{\text{I}}$ + SMILe (50)	SMILeSVM ($\epsilon_s = 10\%$)
SIVAL01	0.737	0.717	**0.752**	**0.874**
SIVAL02	0.658	0.637	0.640	**0.705**
SIVAL03	0.656	**0.770**	**0.762**	0.470
SIVAL04	0.545	**0.558**	**0.551**	**0.639**
SIVAL05	0.656	0.609	0.650	**0.801**
SIVAL06	0.511	0.506	0.494	**0.554**

continued...

Table A.18: Instance-level results for SMILe (continued).

Dataset	MI-SVM$_\mathrm{I}$	MI-SVM$_\mathrm{I}$ + SMILe (25)	MI-SVM$_\mathrm{I}$ + SMILe (50)	SMILeSVM ($\epsilon_s = 10\%$)
SIVAL07	0.650	0.650	**0.656**	0.641
SIVAL08	0.634	0.615	**0.644**	**0.778**
SIVAL09	0.649	0.597	**0.660**	**0.746**
SIVAL10	0.717	**0.757**	0.629	**0.815**
SIVAL11	0.633	0.594	**0.662**	**0.697**
SIVAL12	**0.546**	0.496	0.514	0.502
Newsgroups01	0.670	**0.733**	**0.691**	**0.784**
Newsgroups02	0.767	**0.835**	0.752	**0.844**
Newsgroups03	0.702	**0.704**	0.623	**0.763**
Newsgroups04	0.724	**0.756**	0.644	**0.767**
Newsgroups05	0.755	0.744	**0.759**	**0.756**
Newsgroups06	0.768	**0.792**	0.523	0.644
Newsgroups07	0.716	0.659	**0.720**	**0.749**
Newsgroups08	0.819	0.740	0.730	**0.845**
Newsgroups09	**0.818**	0.780	0.777	0.804
Newsgroups10	**0.813**	0.783	0.771	0.785
Newsgroups11	**0.884**	0.572	0.841	0.738
Newsgroups12	0.777	0.768	0.734	**0.790**
Newsgroups13	0.933	0.849	0.861	0.876
Newsgroups14	0.789	**0.803**	0.789	**0.831**
Newsgroups15	0.812	0.760	**0.813**	**0.814**
Newsgroups16	0.654	**0.773**	**0.689**	**0.751**
Newsgroups17	0.751	0.608	0.736	**0.828**
Newsgroups18	0.808	**0.817**	0.752	0.792
Newsgroups19	0.762	0.725	0.718	**0.774**
Newsgroups20	0.736	0.592	0.657	**0.757**
w.r.t. MI-SVM$_\mathrm{I}$:	—	—	Worse	Better

Table A.19: Comparison of performance (bag-level accuracy) of the NSK to the NSK with SMILe using 25 and 50 shuffled bags and to the CC-NSK. Bag size is chosen for SMILe so that $\epsilon_s = 10\%$. The final row indicates significant differences from NSK.

Dataset	NSK	NSK + SMILe (25)	NSK + SMILe (50)	CC-NSK
musk1	0.845	**0.874**	**0.876**	0.804
musk2	0.843	**0.862**	0.802	0.833
elephant	0.820	**0.833**	**0.841**	**0.825**
fox	0.575	0.570	**0.589**	**0.595**
tiger	0.810	**0.813**	**0.817**	0.790
field	0.825	0.814	0.789	0.770

continued...

Table A.19: Bag-level results for SMILe (continued).

Dataset	NSK	NSK + SMILe (25)	NSK + SMILe (50)	CC-NSK
flower	0.805	**0.823**	0.780	0.705
mountain	0.860	0.839	**0.866**	0.815
OHSUMED1	0.868	0.788	0.846	0.833
OHSUMED2	0.683	**0.696**	**0.691**	**0.723**
SIVAL01	0.842	**0.914**	**0.888**	0.833
SIVAL02	0.858	**0.910**	**0.883**	0.575
SIVAL03	0.917	**0.920**	0.861	0.717
SIVAL04	0.742	**0.901**	**0.968**	**0.767**
SIVAL05	**0.992**	0.975	0.985	0.925
SIVAL06	0.942	**0.954**	0.928	0.742
SIVAL07	**0.917**	0.908	0.859	0.817
SIVAL08	**0.925**	0.611	0.801	0.575
SIVAL09	0.925	**0.926**	0.865	0.775
SIVAL10	**0.950**	0.866	0.851	0.807
SIVAL11	**0.933**	0.888	0.896	0.750
SIVAL12	**0.892**	0.866	0.843	0.692
Newsgroups01	0.800	**0.838**	**0.804**	**0.830**
Newsgroups02	**0.790**	0.783	0.739	0.780
Newsgroups03	**0.760**	0.718	0.726	0.700
Newsgroups04	**0.800**	0.795	0.701	0.770
Newsgroups05	**0.800**	0.788	0.791	0.800
Newsgroups06	**0.780**	0.778	0.655	0.760
Newsgroups07	0.660	**0.715**	**0.731**	**0.680**
Newsgroups08	0.760	0.751	0.758	**0.780**
Newsgroups09	**0.850**	0.840	0.816	0.850
Newsgroups10	**0.800**	0.779	0.765	0.760
Newsgroups11	0.820	**0.846**	**0.833**	**0.860**
Newsgroups12	**0.770**	0.743	0.719	0.770
Newsgroups13	**0.900**	0.899	0.735	0.890
Newsgroups14	0.820	**0.826**	**0.831**	0.800
Newsgroups15	**0.850**	0.807	0.805	0.850
Newsgroups16	**0.770**	0.768	0.768	0.770
Newsgroups17	**0.780**	0.771	0.761	0.730
Newsgroups18	0.810	**0.811**	0.799	**0.830**
Newsgroups19	**0.690**	0.686	0.674	0.660
Newsgroups20	**0.740**	0.706	0.630	0.670
w.r.t. NSK:	—	—	Worse	Worse

Bibliography

Amar, R. A., Dooly, D. R., Goldman, S. A., and Zhang, Q. (2001). Multiple-instance learning of real-valued data. In *Proceedings of the International Conference on Machine Learning*, pages 3–10.

Amores, J. (2013). Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, 201:81–105.

Andrews, S., Tsochantaridis, I., and Hofmann, T. (2003). Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems*, pages 561–568.

Antić, B. and Ommer, B. (2013). Robust multiple-instance learning with superbags. In *Proceedings of the Asian Conference on Computer Vision*, pages 242–255.

Ascher, D., Dubois, P. F., Hinsen, K., Hugunin, J., and Oliphant, T. (2001). *Numerical Python*. Lawrence Livermore National Laboratory, Livermore, CA. `http://numpy.scipy.org/`.

Auer, P., Long, P. M., and Srinivasan, A. (1998). Approximating hyper-rectangles: learning and pseudorandom sets. *Journal of Computer and System Sciences*, 57(3):376–388.

Auer, P. and Ortner, R. (2004). A boosting approach to multiple instance learning.

In *Machine Learning Journal: European Conference on Machine Learning 2004*, volume 3201 of *Lecture Notes in Computer Science*, pages 63–74. Springer.

Babenko, B., Verma, N., Dollár, P., and Belongie, S. (2011). Multiple instance learning with manifold bags. In *Proceedings of the International Conference on Machine Learning*, pages 81–88.

Bartlett, P. and Shawe-Taylor, J. (1999). Generalization performance of support vector machines and other pattern classifiers. In *Advances in Kernel Methods*, pages 43–54. MIT Press.

Ben-David, S., Eiron, N., and Long, P. M. (2003). On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514.

Ben-Tal, A. and Nemirovskiĭ, A. (2001). *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. MPS-SIAM Series on Optimization. SIAM.

Bergeron, C., Zaretzki, J., Breneman, C., and Bennett, K. P. (2008). Multiple instance ranking. In *Proceedings of the International Conference on Machine Learning*, pages 48–55.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305.

Birkhoff, G. (1946). Three observations on linear algebra. *Revista Universidad Nacional de Tucumán, Serie A*, 5:147–151.

Bishop, C. M. (1995). Training with noise is equivalent to Tikhonov regularization. *Neural Computation*, 7(1):108–116.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Blockeel, H., Page, D., and Srinivasan, A. (2005). Multi-instance tree learning. In *Proceedings of the International Conference on Machine Learning*, pages 57–64.

Blum, A. and Kalai, A. (1998). A note on learning from multiple-instance examples. *Machine Learning Journal*, 30:23–29.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. (1987). Occam's razor. *Information Processing Letters*, 24:377–380.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the Vapnik–Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965.

Breiman, L. (1996). Bagging predictors. *Machine Learning Journal*, 24(2):123–140.

Briggs, F., Fern, X. Z., and Raich, R. (2012). Rank-loss support instance machines for MIML instance annotation. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 534–542.

Bunescu, R. and Mooney, R. (2007). Multiple instance learning from sparse positive bags. In *Proceedings of the International Conference on Machine Learning*, pages 105–112.

Carson, C., Belongie, S., Greenspan, H., and Malik, J. (2002). Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038.

Chen, Y., Bi, J., and Wang, J. Z. (2006). MILES: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1931–1947.

Christmann, A. and Steinwart, I. (2010). Universal kernels on non-standard input spaces. *Advances in Neural Information Processing Systems*, pages 406–414.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Cramer, R. D., Patterson, D. E., and Bunce, J. D. (1988). Comparative molecular field analysis (CoMFA). Effect on binding of steroids to carrier proteins. *Journal of the American Chemical Society*, 110(18):5959–5967.

Dahl, J. and Vandenberghe, L. (2009). CVXOPT: A Python package for convex optimization. `http://abel.ee.ucla.edu/cvxopt`.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

Diestel, J. and Uhl, J. J. (1977). *Vector Measures*. Mathematical surveys and monographs. American Mathematical Society.

Dietterich, T. G., Lathrop, R. H., and Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1–2):31–71.

Diochnos, D., Sloan, R., and Turán, G. (2012). On multiple-instance learning of halfspaces. *Information Processing Letters*.

Doran, G. and Ray, S. (2013a). SMILe: Shuffled multiple-instance learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 260–266.

Doran, G. and Ray, S. (2013b). A theoretical and empirical analysis of support vector machine methods for multiple-instance classification. *Machine Learning Journal*, pages 1–24.

Doran, G. and Ray, S. (2014). Learning instance concepts from multiple-instance data with bags as distributions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1802–1808.

Edmonds, J. and Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the Association for Computing Machinery*, 19(2):248–264.

Folland, G. (1999). *Real Analysis: Modern Techniques and Their Applications*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley.

Foulds, J. and Frank, E. (2010). A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, 25(01):1–25.

Foulds, J. R. (2008). *Learning instance weights in multi-instance learning*. PhD thesis, The University of Waikato.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository.

Freund, Y. and Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, pages 23–37. Springer.

Gärtner, T. (2008). *Kernels for structured data*, volume 72 of *Series in Machine Perception and Artificial Intelligence*. World Scientific.

Gärtner, T., Flach, P., Kowalczyk, A., and Smola, A. (2002). Multi-instance kernels. In *Proceedings of the International Conference on Machine Learning*, pages 179–186.

Gentle, J. (2009). *Computational Statistics*. Statistics and Computing. Springer.

Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. (2007). A kernel method for the two-sample-problem.

Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B., and Smola, A. J. (2006). A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems*, pages 513–520.

Gretton, A., Fukumizu, K., Harchaoui, Z., and Sriperumbudur, B. K. (2009). A fast, consistent kernel two-sample test. In *Advances in Neural Information Processing Systems*, pages 673–681.

Han, Y., Tao, Q., and Wang, J. (2010). Avoiding false positive in multi-instance learning. In *Advances in Neural Information Processing Systems*, pages 811–819.

Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36.

Haussler, D. (1992). Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and computation*, 100(1):78–150.

Hernández, J. and Inza, I. (2011). Learning naive bayes models for multiple-instance learning with label proportions. In *Advances in Artificial Intelligence*, pages 134–144. Springer.

Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python. `http://www.scipy.org/`.

Kearns, M. J. and Schapire, R. E. (1994). Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and System Sciences*, 48(3):464–497.

Kelley, Jr., J. E. (1960). The cutting-plane method for solving convex programs. *Journal of the Society for Industrial & Applied Mathematics*, 8(4):703–712.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95.

Kundakcioglu, O., Seref, O., and Pardalos, P. (2010). Multiple instance learning via margin maximization. *Applied Numerical Mathematics*, 60(4):358–369.

Kwok, J. T. and Cheung, P.-M. (2007). Marginalized multi-instance kernels. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 7, pages 901–906.

Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L., and Jordan, M. (2004). Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72.

Li, Y.-F., Kwok, J. T., Tsang, I. W., and Zhou, Z.-H. (2009). A convex method for locating regions of interest with multi-instance learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 15–30. Springer.

Liu, G., Wu, J., and Zhou, Z.-H. (2012). Key instance detection in multi-instance learning. In *Proceedings of the Asian Conference on Machine Learning*, pages 253–268.

Long, P. and Tan, L. (1998). PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance examples. *Machine Learning Journal*, 30(1):7–21.

Maaten, L., Chen, M., Tyree, S., and Weinberger, K. Q. (2013). Learning with marginalized corrupted features. In *Proceedings of the International Conference on Machine Learning*, pages 410–418.

Mangasarian, O. and Wild, E. (2008). Multiple instance classification via successive linear programming. *Journal of Optimization Theory and Applications*, 137:555–568.

Maron, O. (1998). *Learning from Ambiguity*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA.

Maron, O. and Ratan, A. L. (1998). Multiple-instance learning for natural scene classification. In *Proceedings of the International Conference on Machine Learning*, pages 341–349.

Micchelli, C. A., Xu, Y., and Zhang, H. (2006). Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667.

Muandet, K., Fukumizu, K., Dinuzzo, F., and Schölkopf, B. (2012). Learning from distributions via support measure machines. In *Advances in Neural Information Processing Systems*, pages 10–18.

Ng, A. Y. and Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. page 841.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and E., D. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*.

Rahmani, R., Goldman, S. A., Zhang, H., Krettek, J., and Fritts, J. E. (2005). Localized content based image retrieval. In *Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pages 227–236. ACM.

Ramon, J. and De Raedt, L. (2000). Multi instance neural networks. In *Proceedings of the ICML 2000 Workshop on Attribute-Value and Relational Learning*.

Ray, S. and Craven, M. (2005). Supervised versus multiple instance learning: an empirical comparison. In *Proceedings of the International Conference on Machine Learning*, pages 697–704.

Ray, S. and Page, D. (2001). Multiple instance regression. In *Proceedings of the International Conference on Machine Learning*, pages 425–432.

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

Sabato, S. and Tishby, N. (2012). Multi-instance learning with any hypothesis class. *Journal of Machine Learning Research*, 13:2999–3039.

Salton, G. and McGill, M. (1983). *Introduction to modern information retrieval*. McGraw-Hill Computer Science Series. McGraw-Hill.

Schölkopf, B., Herbrich, R., and Smola, A. J. (2001). A generalized representer theorem. In Helmbold, D. and Williamson, B., editors, *Computational Learning Theory*, volume 2111 of *Lecture Notes in Computer Science*, pages 416–426. Springer Berlin Heidelberg.

Schölkopf, B. and Smola, A. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press.

Schölkopf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319.

Schölkopf, S. P., Vapnik, V., and Smola, A. (1997). Improving the accuracy and speed of support vector machines. pages 375–381.

Scott, S., Zhang, J., and Brown, J. (2005). On generalized multiple-instance learning. *International Journal of Computational Intelligence and Applications*, 5(1):21–35.

Settles, B., Craven, M., and Ray, S. (2008). Multiple-instance active learning. In *Advances in Neural Information Processing Systems*, pages 1289–1296.

Simon, H. U. (2012). PAC-learning in the presence of one-sided classification noise. *Annals of Mathematics and Artificial Intelligence*, pages 1–18.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. (2007). A Hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31. Springer.

Song, L., Fukumizu, K., and Gretton, A. (2013). Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *Signal Processing Magazine, IEEE*, 30(4):98–111.

Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B., and Lanckriet, G. R. G. (2012). On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599.

Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. (2010). Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 99:1517–1561.

Szabó, Z., Gretton, A., Póczos, B., and Sriperumbudur, B. (2014). Two-stage sampled learning theory on distributions. `http://arxiv.org/abs/1402.1754`.

Tao, Q., Scott, S. D., Vinodchandran, N., Osugi, T. T., and Mueller, B. (2008). Kernels for generalized multiple-instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(12):2084–2098.

Tao, Q., Scott, S. D., Vinodchandran, N. V., and Osugi, T. T. (2004). SVM-based generalized multiple-instance learning via approximate box counting. In *Proceedings of the International Conference on Machine Learning*, pages 779–806.

Tong, S. and Koller, D. (2002). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66.

Tragante do Ó, V., Fierens, D., and Blockeel, H. (2011). Instance-level accuracy versus bag-level accuracy in multi-instance learning. In *Proceedings of the 23rd Benelux Conference on Artificial Intelligence*.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

Vapnik, V. and Kotz, S. (2006). *Estimation of Dependences Based on Empirical Data*. Information science and statistics. Springer.

Vapnik, V. N. and Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280.

von Neumann, J. (1953). A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games*, 2:5–12.

Wagstaff, K. L. (2012). Machine learning that matters. In *Proceedings of the International Conference on Machine Learning*.

Wagstaff, K. L., Lane, T., and Roper, A. (2008). Multiple-instance regression with structured data. In *Proceedings of the 4th International Workshop on Mining Complex Data*.

Wang, C., Scott, S. D., Zhang, J., Tao, Q., Fomenko, D. E., and Gladyshev, V. N. (2004). A study in modeling low-conservation protein superfamilies. Technical report, Department of Computer Science, University of Nebraska.

Xu, X. (2003). Statistical learning in multiple instance problems. Master's thesis, The University of Waikato.

Xu, X. and Frank, E. (2004). Logistic regression and boosting for labeled bags of instances. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 272–281.

Yu, Y., Cheng, H., Schuurmans, D., and Szepesvári, C. (2013). Characterizing the

representer theorem. In *Proceedings of the 30th International Conference on Machine Learning*, pages 570–578.

Zhang, Q. and Goldman, S. (2001). EM-DD: An improved multiple-instance learning technique. In *Advances in Neural Information Processing Systems*, pages 1073–1080.

Zhou, Z., Sun, Y., and Li, Y. (2009). Multi-instance learning by treating instances as non-IID samples. In *Proceedings of the International Conference on Machine Learning*, pages 1249–1256.

Zhou, Z. and Zhang, M. (2003). Ensembles of multi-instance learners. In *Machine Learning: ECML 2003*, volume 2837 of *Lecture Notes in Computer Science*, pages 492–502. Springer.

Zhou, Z.-H. and Zhang, M.-L. (2002). Neural networks for multi-instance learning. In *Proceedings of the International Conference on Intelligent Information Technology*.