

# A Probabilistic Framework for Memory-Based Reasoning

Simon Kasif<sup>1</sup>  
Steven Salzberg<sup>1</sup>  
David Waltz<sup>2</sup>  
John Rachlin<sup>1</sup>  
David Aha<sup>3</sup>

kasif@cs.jhu.edu  
salzberg@cs.jhu.edu  
waltz@research.nj.nec.com  
rachlin@cs.jhu.edu  
aha@aic.nrl.navy.mil

## Abstract

In this paper, we propose a probabilistic framework for Memory-Based Reasoning (MBR). The framework allows us to clarify the technical merits and limitations of several recently published MBR methods and to design new variants. The proposed computational framework consists of three components: a specification language to define an adaptive notion of relevant context for a query; mechanisms for retrieving this context; and local learning procedures that are used to induce the desired action from this context. Based on the framework we derive several analytical and empirical results that shed light on MBR algorithms. We introduce the notion of an MBR transform, and discuss its utility for learning algorithms. We also provide several perspectives on memory-based reasoning from a multi-disciplinary point of view.

## 1 Introduction

Reasoning can be broadly defined as the task of deciding what action to perform in a particular state or in response to a given query. Actions can range from admitting a patient into a hospital, moving to the right position to catch a baseball, prefetching a page in a computer system, or performing a financial transaction. If the domain is finite and the number of states is small, reasoning can be solved simply by retrieval from a table that enumerates all states and their associated actions. However, in most domains this is not feasible. As a result, reasoning takes many forms. One common approach to reasoning from data is to first learn a model of the data (a knowledge representation) such as a set of rules or a probability distribution, and later perform reasoning by deduction or probabilistic inference. Another approach is to use a set of stored experiences as the basis for answering queries about newly encountered states. Direct reasoning from stored memories relies on retrieving a relevant context upon encountering a new event and choosing the appropriate action by interpolating from a set of similar instances contained in the context. With the substantial growth in recent years of scientific and commercial databases available, the potential uses for this latter form of reasoning are rapidly expanding. This paradigm of performing inferences from data is often broadly referred to as *memory-based reasoning* (MBR).

Memory-based reasoning has been used successfully in a number of domains such as classification of news articles [MLW92], census data [CMSW92], software agents [MK93], computational biology [YL93, ZMW92, CS93], robotics [MAS95], diagnosis [BH95], computer vision [Ede94], and many other pattern recognition and machine learning applications. Given this popularity, it is important to define a unified framework for this work which will standardize the components of some MBR systems, unify programming notation, and provide the basis for theoretical studies. This paper makes a small step in this direction. In this paper we outline a probabilistic framework for a class of MBR methods. This formalism is motivated by the method originally proposed by Stanfill and

---

<sup>1</sup>Computer Science Dept., Johns Hopkins U., Baltimore, MD

<sup>2</sup>NEC Research Institute, Princeton, NJ 08540

<sup>3</sup>Navy Center for Applied Research in AI, Naval Research Laboratory, Washington DC 20375

Waltz [SW86]. The framework clarifies the technical merits and limitations of the original approach and the papers that followed it. With the large number of papers that have been published using this form of MBR, it is particularly important to shed light on this paradigm.

## 2 The MBR Framework

Our definition of an MBR framework has several components:

1. A stored database of examples is used as the implicit storehouse on which to base all reasoning.
2. The user specifies a probabilistic model that will be fit to data. This specification includes a set of random variables associated with different features in the data. The user also specifies a set of probabilistic assumptions such as priors and independence assumptions. Such knowledge can be expressed using Bayes networks (see Pearl [Pea88]), which provide graphical support for specifying complex probability distributions. The difficulty of this step is obviously related to the complexity of the model that user wants to impose on the data. In most cases, this can be done by using a library of models or using domain knowledge.
3. The MBR system can generate a complete probabilistic model by fitting a model consistent with the properties specified by the probabilistic network. The induced model may contain new hidden variables.
4. The probabilistic model is used to define an adaptive notion of relevance that is based on both the data and the desired task. This is achieved by the use of an adaptive distance (similarity) metric on the domain. The induced distance is adaptive in the sense it is learned from data and gets modified when new data items are added.
5. In some cases the system induces an explicit transformation (see below) on the data (which we call the MBR transform). Thus, each data instance is transformed to a new instance that may or may not include new attributes (e.g., hidden variables). A data structure that supports efficient retrieval of relevant instances can now be mechanically derived in the transformed space.
6. When a query is posed, the MBR system retrieves the relevant context for the query; i.e., those instances from the database that are most similar. A local model is then constructed over the set of relevant instances. Relevancy is implicitly defined by the learned distance metric, which in turn is induced from the probabilistic model. MBR performs **local learning** on this relatively small set of relevant instances, thereby producing a local model which is used to answer the specific query. Local modeling can be performed using regression, local decision trees, neural networks or other learning methods.

The first and the last components are common to all MBR systems that we are familiar with. Steps 2-5 (i.e. the use of graphical probabilistic models to define relevant context for answering queries) are unique to our proposed framework and generalize the original proposal by [SW86]. The framework introduced above is aimed at the goal of developing a set of automated procedures that use both standard (“off the shelf”) and novel components to build flexible MBR systems. We can use a variety of existing software systems to specify probabilistic graphical models [SDLC93]. Such packages often provide effective procedures to learn the parameters of the model. The system then needs to code the transformation on the space induced by the model. In many cases this

transformation is straightforward (see below). However, in general formalizing and automating this transformation (from model to retrieving relevant context) is still an open research question. Finally, given a query the system can use local learning procedures (such as local decision trees or regression) to predict the appropriate action in the relevant context. This component can be easily automated.

The MBR paradigm is more general than the above probabilistic framework. However, this overview is broad enough to capture many existing algorithms (e.g., [SW86, CS93, MK93, ZMW92, MAS95]). In addition, this framework clarifies and analyzes the advantages of the technique originally proposed in [SW86], and suggests new variants that do not exist in the literature.

### 3 Summary of Related Research

Inducing distance metrics and judging relevance using complex probabilistic models and data is a very basic research topic. In a text-to-speech translation system, how similar is one data instance (e.g., DEAR) to another (e.g., NEAR)? Or more interestingly, is DEAR closer to NEAR than to BEAR? If our training set consists of protein sequences, we face the similar problem of determining functional similarity based on sequence similarity; i.e., does the protein’s structure change if one replaces amino acid G by amino acid H? (Indeed biologists have produced mutation matrices that capture this notion [JTT92]).

The key technical question is how to learn a “good” distance metric from data (and optionally partial models provided by domain experts). This issue raises a large number of questions which have been addressed in statistical research on relevance and similarity, and that resulted in methods such as multidimensional scaling, singular value decomposition, principal components analysis, and factor analysis [KM78, She80] and vector quantization. Since the notions of relevance and similarity are somewhat ill-defined without a specific task in mind, there is substantial literature on axiomatic definitions of relevance and similarity [Tve77].

The issues addressed in this paper (e.g. probabilistic models of relevance or similarity) are quite broad and have been addressed in psychology, statistics, information retrieval, AI, pattern recognition, computer vision and natural language processing. Thus, it is impossible to enumerate all possible references to potentially relevant papers. Instead, we mention only the work most relevant to our particular computational framework.

Memory-based reasoning has its roots in work that dates back to near the beginning of this century, although of course computational methods arose much more recently. The earliest algorithms that might properly be classed as MBR methods date back to work in the 1960s on the application of local regression to a set of nearest neighbors, a technique known as kernel regression [Nad64, Roy66]. A good historical collection of nearest-neighbor algorithms is Dasarathy [Das91], which contains references going back to the 1950s. The use of local models for function estimation and smoothing in an MBR framework is described in Atkeson et al. [AMS95], who also include a review of the literature. See also [AKA91] for a variety of results on nearest-neighbor learning algorithms.

Kernel density estimation and Partzen windows are broadly defined areas of non-parametric statistics that rely on memories to perform classification, to learn functions, and to estimate probability distributions [DH73]. These techniques are widely used in statistics and pattern recognition.

The use of the term memory-based reasoning in a broad context was introduced by Stanfill and Waltz [SW86], who also introduced the Value Difference Metric (VDM) to define similarity when

using symbolic-valued features. The VDM is an adaptive distance metric that adjusts itself to a database of examples, and can then be used for retrieval (see Section 4).

Tree-based methods for partitioning data into regions (e.g., [Omo89, Omo87]) such as k-d trees or decision trees [Qui93] also can be used to define a relevant local neighborhood. Thus, instead of seeing a decision tree as a classification device in the MBR context, a decision tree defines a static partitioning of space into regions. In other words, the distance between data instances that are grouped in the same region (same leaf of the tree) is defined (implicitly) to be zero. Once a query is given, we can retrieve all the points in the relevant region of the tree – i.e all points which are distance zero to the query point – and perform local interpolation. (Note that in supervised classification, the regions in pruned decision trees are not necessarily labelled by the same class label.) Since decision trees can be viewed as a compact way to express probability distributions, this method will be considered as a particular variant of our framework.

In natural language processing we find some similar notions. For example, one can define the “semantics” of a verb as a probability distribution over the set of nouns that follow it [PTL93]. Thus, each verb is mapped into a point in a high-dimensional real-valued space. Then the similarity of two verbs can be computed by computing the distance between the two probability distributions (such as the Kullback-Leibler divergence). There is a rich theory that studies such distances, which is often referred to as information geometry.

In computer vision, some forms of memory-based reasoning have been a popular theme in applications such as character recognition and face recognition. For example, there is the notion of a chorus of prototypes where an object is defined again by a vector of distances to other objects [Ede94]. That is, each object is mapped into a probability distribution and the distance between two objects is determined by computing a standard distance (geodesic) in a space of probability distributions. A different but obviously related notion is the idea of radial basis functions defined over clusters of instances. However, relatively little work is available on using memory-based reasoning to perform complex visual tasks such as object recognition in a cluttered environment or 3-D navigation.

In information retrieval, we find a recent use of Bayes networks to specify the notion of relevance of a document to a topic or a query [TC91]. In this domain, the notion of retrieving a relevant context or a document is related to our work. Recent ground breaking work in economics introduced the notion of case-based decision theory [GS94], where the utility of an action in a particular state is computed by using a kernel density estimation technique over a set of stored memories. Some very recent work in statistics generalizes traditional nearest neighbor learning with adaptive neighborhood techniques. There the issue is finding the correct (relevant) neighborhood to a given query in a classification task [HT94, Fri94]. Finally, in a collection of papers, Vapnik and his group introduce a theoretical notion of local learning which corresponds to learning a local function at a particular point in response to a query [BV92, VB93].

## 4 Value Difference Metrics

To introduce the MBR framework, we will first define and give an example of an adaptive distance function. We begin by considering a traditional classification problem. Assume we are given  $N$  examples where each example  $X = (x_1, \dots, x_d)$  is a point in some  $d$ -dimensional space. Each dimension corresponds to some natural feature in the domain, and each point  $X$  has a class label  $C$ . The standard classification problem is to predict the class of a new, previously unseen point.

We can perform classification using a standard nearest-neighbor method such as k-NN, but

we will use an adaptive distance between instances called the Modified Value Difference Metric (MVDM). MVDM defines the value difference between two values  $v_1$  and  $v_2$  of a given feature to be:

$$\delta(v_1, v_2) = \sum_{i=1}^k \left| \frac{v_{1_i}}{V_1} - \frac{v_{2_i}}{V_2} \right| \quad (1)$$

where  $k$  is the number of classes,  $v_{1_i}$  is the number of times  $v_1$  occurred for instances of class  $C_i$  and  $V_1$  is the number of times  $v_1$  occurred for all classes. Thus the ratio  $v_{1_i}/V_1$  is the empirically observed probability of class  $C_i$  given that the feature has value  $v_1$ ,  $P(C_i|v_1)$ .

Intuitively, MVDM defines the distance between two examples as the sum of the value differences across all features. It suggests that the distance between two examples should be related to the effect each feature has on the action taken (in this case classification). Value-difference metrics (VDM) were introduced by Stanfill and Waltz [SW86]. MVDM is a modified variant of VDM, introduced by Cost and Salzberg [CS93] and incorporated in the MBR system PEBLS.

Note that the VDM and MVDM operate on databases of examples with discrete attributes, (*symbolic* attributes or discretized real-valued features), i.e., each database entry contains a set of discrete values and possible other information such as a class label. Cost and Salzberg [CS93] demonstrated that PEBLS using MVDM performs well relative to several other learning algorithms on a number of practical problems (see also [ZMW92]).

## 5 The MBR Transform

In this section we present a probabilistic framework for MBR. We show a case study where the definition of a distance metric such as MVDM follows naturally from a simple probabilistic model. We start with a simple example of a binary classification problem. Given an instance  $(x_1, \dots, x_n)$  we want to classify it as being a member of class  $C_1$  or  $C_2$ . In this case, the MVDM computation can be restated as performing a simple transform on the space of the attributes. Essentially, it converts each instance  $(x_1, \dots, x_n)$  to an instance  $(P(C_1|x_1), \dots, P(C_1|x_n))$ , and stores the new instance in the database. Given a new instance  $(y_1, \dots, y_n)$ , we convert it to  $(P(C_1|y_1), \dots, P(C_1|y_n))$  and then simply seek the nearest neighbor in this transformed space. The reader can easily verify that this transformation faithfully models the original MVDM computation. This follows from the observation that the distance between  $x_1$  and  $y_1$  for any two instances  $(x_1, \dots, x_n)$  and  $(y_1, \dots, y_n)$  is

$$|P(C_1|x_1) - P(C_1|y_1)| + |P(C_2|x_1) - P(C_2|y_1)| = 2(|P(C_1|x_1) - P(C_1|y_1)|)$$

In general, we are given a database of tuples  $\{X_j\}$ . Each tuple  $X_j = (x_1, \dots, x_d, C)$ , where the  $x_i$ 's are random variables that correspond to input features, and  $C$  corresponds to an action (e.g. classification, value of a function). We transform each event  $x_i = a_i$  to a discrete probability distribution  $(P(C = C_1|x_i = a_i), \dots, P(C = C_k|x_i = a_i))$ . In some cases, we transform  $x_i = a_i$  to  $(\log(P(C = C_1|x_i = a_i)), \dots, \log(P(C = C_k|x_i = a_i)))$  (see the following sections).

Thus, in general we transform a  $d$ -dimensional input space to a  $kd$ -dimensional space where  $k$  is the number of classes. As seen above,  $k - 1$  dimensions are sufficient.

To obtain a geometric intuition for the utility of the MBR transform consider as an example, a simple binary classification problem in two dimensions where the classes are linearly separable. We map each point  $(x_1, x_2)$  into the two dimensional space  $(P(C = C_1|x_1), P(C = C_1|x_2))$ . To compute

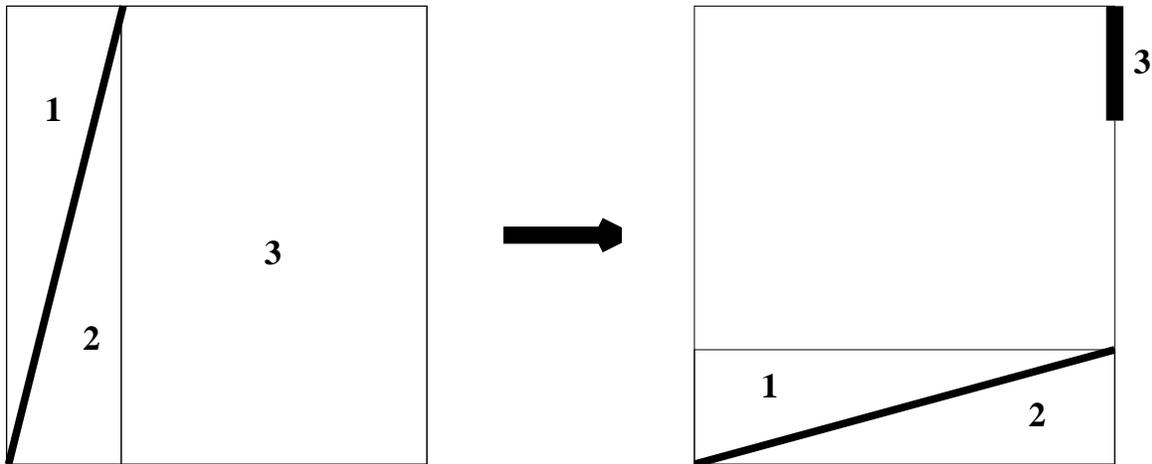


Figure 1: On the left is shown a simple partitioning induced by the function  $y = 4x$  on a finely discretized unit square. The area above the line is class  $C_1$ , and below the line is class  $C_2$ . Thus, points in region 1 are in class  $C_1$ , and points in regions 2 and 3 are in class  $C_2$ . On the right is the transformed space. All points in the left figure are mapped to the corresponding regions in the right figure. Note that class  $C_2$  region is mapped to two distinct regions on the right. The majority of the space (region 3), comprising the rectangle from the point  $(0.25, 0)$  to  $(1, 1)$  in the left-hand graph, is mapped to a small line segment from  $(1, 0.75)$  to  $(1, 1)$  in the right-hand graph.

the probabilities, we use a finely discretized real-valued instance space. Real-valued attributes are used in this example primarily to provide a geometric intuition for the MBR transform that is easy to visualize. It is likely that probabilistic MBR might be most useful in symbolic domains. The MBR transform for this case is illustrated in Figure 1. In particular, note that a large portion of the space (region 3) is mapped into a one-dimensional region. This is a significant dimension reduction which obviously is helpful in reducing the sample complexity of training and therefore facilitates faster learning (i.e., faster convergence rate of learning procedures).

### 5.1 A probabilistic framework for the MBR transform

The transformations above are clearly induced by a very simple probabilistic model that assumes independence between the attributes. We want the user to have a general, flexible facility to specify such transformations. In our framework, the MBR transformation is induced by a simple probabilistic model that can be specified graphically using a two-layer causal tree. In particular, to specify the transformation done by MVDM, we use a model that assumes independence of the joint probability distribution on all the variables (including the class). In other words:

$$P(x_1, \dots, x_d, C) = \prod_j P(x_j|C)P(C)$$

The derivation of the transformation is given in Section 6.2. MBR then uses this model as the basis for the transformation. The model is illustrated in Figure 2. The variables in the figure will normally be defined by a domain expert; however, they could be augmented by automatically constructed features or combinations of features. The values  $p(C|x_i)$  correspond to the  $\lambda$ -messages

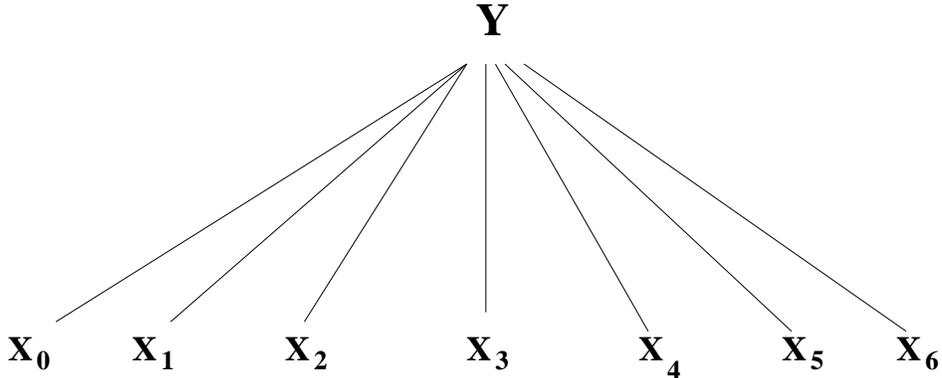


Figure 2: The class variable  $Y$  at the root of the causal network is dependent on all of the variables  $X_i$  represented by the leaf nodes. Each leaf node  $X_i$  is class conditionally independent of any other variable  $X_j$  given the class  $Y$ . Each edge of the tree is therefore associated with a matrix of conditional probabilities of the variable  $X_i$  given the class  $Y$ .

in probabilistic networks, and thus can be obtained directly from the network [Pea88]. As mentioned in the introduction, this paper is not addressing the automatic derivation of the transformation from the model. We primarily want to motivate the practical usefulness of probabilistic MBR methods, analyze their performance and suggest the possibility automating the transform. The transformation of values into probabilities defined here will be referred to as the **MBR transform**. The MBR transform has several desirable theoretical properties, as discussed in later sections.

## 6 Towards Probabilistic Reasoning Using MBR

The approach outline above generalizes to more complex graphical probabilistic models such as in [SDLC93] which may include hidden variables and therefore perform constructive induction. Since this initial paper on the topic of probabilistic MBR focuses on the analysis of VDM type approaches (in the probabilistic framework), we don't provide a detailed analysis of this generalization which would be discussed in a follow up paper. In general, probabilistic graphical models provide support for the following capabilities:

- A framework for specifying the probabilistic dependencies that we want to capture in the data. This is typically coded by the structure of the probabilistic network. This structure indicates which parameters (conditional probabilities) must be learned from data. The structure is specified by the modeler, and may include hidden variables. There are also a number of approaches that learn the structure from data.
- A computational framework for combining these conditional probabilities using algorithms that take advantage of the graphical properties of the model to simplify (speed-up) computation.

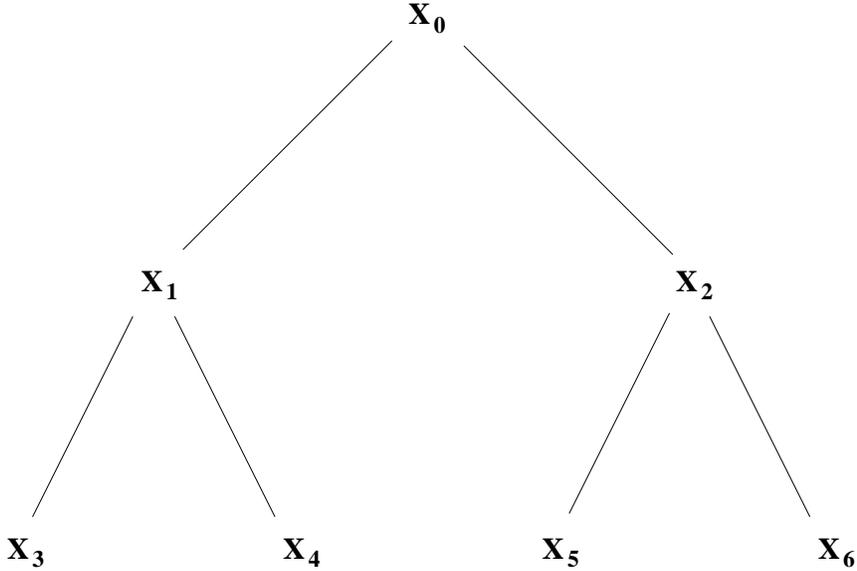


Figure 3: Tree Model

However, there are two problems that we might encounter in practical learning tasks.

- The structure of the prespecified model may be incorrect (this is most problematic aspect of the approach).
- The computation required to fuse the probabilities to answer a particular query in a complex model may be computationally unfeasible.

Our approach, advocates using probabilistic models only for recording the probabilistic dependencies among variables as specified by the model. However, we then perform prediction where we use the information computed by the network as a set of constructed features (probabilistic features). Thus, we may use other learning methods (in this case MBR) to combine/fuse the probabilistic information previously recorded by the network. This allows us in some cases to specify a simplified model of the data that facilitates an efficient inference of probabilistic quantities, and let the MBR procedures to “correct” the simplified assumptions made. Note that matching of the query to objects in the database now takes place in a space of probability distributions.

We illustrate the intuition behind the more general framework than MVDM using the example of a model in the form of a causal tree defined on seven binary random variables (see Figure 3). This probabilistic model implies that the joint probability distribution on seven variables has a simple form. Specifically, it can be factored as a product:

$$p(x_0, x_1, x_2, x_3, x_4, x_5, x_6) = p(x_1|x_0)p(x_2|x_0)p(x_3|x_1)p(x_4|x_1)p(x_5|x_2)p(x_6|x_2)p(x_0)$$

The reader is referred to [Pea88] for a detailed information on the computational and statistical advantages of this factorization. Let us now assume we have a large database of tuples of the form  $(x_0, x_1, x_2, x_3, x_4, x_5, x_6)$ . For instance, a tuple  $(x_1, x_2, x_3, x_4, x_5, x_6)$  may define the attributes associated with a given biological cell, and  $x_0$  could be a random variable that corresponds to a medical classification (e.g, malignant).

Assume we are now given a query in the form of a new tuple  $(x_3, x_4, x_5, x_6)$  and are asked to predict the variable  $x_0$ , a standard classification task. In the framework of probabilistic networks

this problem can be handled directly by assuming the new tuple is the evidence provided to the network which we call  $E$ . Given the evidence  $E$  we can easily compute the probability distribution of  $x_0$  given  $E$ , using the efficient inference algorithm for trees outlined in [Pea88]. The algorithm will perform optimally (minimizing Bayes risk) if the model is correct. However, in practice the model may not be correct. Thus the model will not provide adequate performance on this prediction task even if infinite amount of data is available. In statistical terms, we lack consistency (i.e., convergence to optimal performance as the amount of data is increased).

Note that traditional MBR methods cannot handle this scenario directly since two of the attributes have missing values, and standard MBR inference is typically made based only on existing values. Alternatively, using the probabilistic MBR transform approach, we can transform the query  $(x_3, x_4, x_5, x_6)$  to a tuple  $\lambda(x_0), \lambda(x_1), \lambda(x_2), \lambda(x_3), \lambda(x_4), \lambda(x_5), \lambda(x_6)$  where the vector  $\lambda$  is defined as  $\lambda(x_i) = p(E|x_i)$  (see [Pea88] for additional information). Note that attributes  $x_1$  and  $x_2$  are in fact getting “filled”, although no specific direct evidence for these variables has been provided. Instead, we assume these attributes assume the value  $p(E|x_1)$  and  $p(E|x_2)$  respectively. We can now perform the same transformation on every tuple of the database and match the transformed query to the transformed database.

Note that if the tree was in fact the correct model we do not cause any damage since  $x_0$  really depends on the conditional probability distributions of  $x_1$  and  $x_2$  which are as new attributes included in the transformed databases. We also note that terms such as  $p(E|x_2)$  can be factored into simpler expressions because of the independence assumptions made in the structure of the tree network [Pea88]. We now can match the transformed query to each item in the database as specified above for MVDM. It is obvious that if the network is the correct model for the data it is sufficient that the MBR method learns the mapping  $p(x_0|E) > 0.5$  which is relatively easy to learn with relatively few examples given the transformed database. However, if the network is incorrect we still maintained our ability to learn a correct probability distribution.

As mentioned above, if the model is indeed correct we would not benefit from using this form of MBR. However, it is usually impossible to specify the correct model for a complex domain. Our framework suggests a methodology to incorporate “user provided advice” into memory-based reasoning. In other words, the modeler or domain expert initially might specify the structure of a particular probabilistic model in the form of a set of rules. E.g, consider a rule where a conjunction of binary events  $A_1, \dots, A_n$  implies the binary event  $C$ . This rule might not be valid in practice. What we typically do in probabilistic frameworks is create a network with binary random variables  $A_1, \dots, A_n$  and  $C$  in the hope of recording the actual probabilistic dependency of  $C$  on  $A_1, \dots, A_n$ . However, this process is often prohibitive since the conditional probability table has an exponential (in  $n$ ) number of entries. There are many solutions within the probabilistic networks framework; we primarily suggest another methodology. Using MVDM type approaches we assume a causal network structure (as in the figure above), and then “correct” the invalid independence assumptions made using MBR.

We will show in subsequent sections that the MBR approach expands both the expressiveness and the learning capacity of the specified model. That is, we can learn to perform more accurately on a more general class of problems than the original probabilistic model would allow us to do. In particular, we report on an experimental and analytical comparison between a particular MBR system called PEBLS [CS93] and a widely-used probabilistic method known as the naive Bayesian classifier, which is specified by the simple probabilistic network in Figure 2.

We would like to emphasize that we are not advocating specific learned metrics and their induced MBR transforms. Here we considered one such transform (i.e., MVDM) to illustrate the advantage of the technique. One could just as easily use tools such as mutual information or logarithms of

probabilities (see next section) to re-express the values, and in some cases other metrics might have theoretically better properties. In most cases the usefulness of a particular approach needs to be determined empirically.

As an example of another transform, consider a set of points in two-dimensional discrete space that form  $K$  distinct clusters. One natural transformation is to convert each instance to a pair, where the first component is the mean vector of the cluster containing the instance, and the second value is the distance in standard deviations from the mean. The model implicit in this transformation is a probabilistic model of a set of points normally distributed around a small number of centroids, which can be easily specified by a Bayes network. Our framework captures this and other natural distributions such as statistical mixtures [TSM85], probabilistic hierarchies and complex probabilistic models [Pea88], and models that include hidden variables [Lau95].)

The model itself will vary depending on the goals of the system; in fact, the same database can be used with different models to produce different MBR systems that solve different problems. In each case the model will induce a different transformation. Thus we have an adaptive notion of relevancy. A feature may be relevant in the context of one action and quite irrelevant in another. Our approach makes it possible to define this dynamically on the same database.

## 6.1 MBR transform for static databases

In previous papers on MBR the transformation is typically implicit. Thus, the model is used to induce a particular distance, which is subsequently used to retrieve relevant instances. Retrieval is done by linear sequential search or an “embarrassingly simple” parallel implementation of such search. When the MBR transform is relatively static (e.g., a fixed classification problem on a fixed database), it makes sense to actually apply the transformation to each of the instances and store the transformed space rather than the original database. This explicit application of the MBR transform may lead to substantial improvements in computational efficiency when using MBR for retrieving relevant instances. The advantage of performing the transformation is that if we store the transformed instances (instead of the original ones), we can use computational geometry techniques to perform fast approximate retrieval of nearest neighbors in real-valued spaces, in time which is logarithmic in the number of instances. There are numerous data structures that have been developed to search the database of instances efficiently when a query is presented, mostly based on the  $k$ - $d$  tree framework [Ben75]. These techniques allow one to find relevant instances in logarithmic expected time (logarithmic in the size of the transformed database). Note that it may be difficult to define an efficient data structure on the original space since the attributes may be symbolic. Since the transform is highly sensitive to data, in the case of highly dynamic databases we have to devise incremental procedures to maintain an accurate partitioning into regions. This is algorithmically challenging, and in the case of dynamic databases it usually makes sense to keep the data in the original form and use a efficient linear time distance computations to decide on a relevant context.

## 6.2 Theoretical motivation for the MBR transform

In this section we make two simple technical observations that motivate the MBR transform in the context of MVDM-like metrics defined above. For simplicity, we assume a classification problem where the examples are labelled by two classes  $A$  and  $B$ , and each example has two features, where the features are class conditionally independent. The analysis trivially extends to  $d$ -dimensional

vectors. Thus, for any object  $X = (x_1, x_2)$  we have

$$p(A|X) = p(x_1|A)p(x_2|A)p(A)/p(X)$$

Consequently, the discrimination function (the decision boundary between the two classes) is defined by:

$$\log(p(A|X)/p(B|X)) = \log p(x_1|A) - \log p(x_1|B) + \log p(x_2|A) - \log p(x_2|B) + \text{constant}.$$

In other words, the discrimination function is linear in the log probabilities. That is, points above and below the linear discriminant are in class  $A$  and  $B$  respectively. This is a well know fact in pattern recognition, however, it has interesting implications for MBR transforms. In particular, if we transform each tuple in the training set  $(x_1, x_2)$  to a 4-tuple

$$(\log(x_1|A), \log(x_1|B), \log(x_2|A), \log(x_2|B))$$

, we now have to learn a simple linear classifier. It is known that the standard  $K$ -nearest neighbor algorithm achieves a very rapid convergence rate on linearly separable concept classes. This analysis can be generalized to more complex probabilistic networks, that do not make such strong independence assumptions. The intuition is that any probability distribution expressed by a Bayes network can be expressed as a product of terms involving conditional probabilities. Therefore, the discrimination function becomes a linear function of the logs of these terms, which makes it easy to learn by a nearest neighbor methods. This will be discussed in a follow-up paper.

Additional motivation is provided by the following analysis. Consider two instances  $X$  and  $Y$ , each is a two dimensional binary vector. We observe that certain MBR transforms (similar to MVDM) have a desirable feature, namely instances that are close to each other tend to be classified similarly.

Assuming a uniform distribution on  $X$  and  $Y$  and independence we get:

$$p(A|X)/p(A|Y) = p(x_1|A)p(x_2|A)/p(y_1|A)p(y_2|A)$$

Thus,

$$\begin{aligned} |\log(p(A|X)/p(A|Y))| &= |\log p(x_1|A) + \log p(x_2|A) - \log p(y_1|A) - \log p(y_2|A)| \\ |\log p(A|X)/p(A|Y)| &= |(\log p(x_1|A) - \log p(y_1|A)) + (\log p(x_2|A) - \log p(y_2|A))| \end{aligned}$$

Finally we obtain:

$$|\log p(A|X)/p(A|Y)| \leq |\log p(x_1|A) - \log p(y_1|A)| + |\log p(x_2|A) - \log p(y_2|A)|$$

Therefore, as the right-hand side of the equation gets small, so does the left-hand side. However, the right-hand side corresponds to the distance between instances after the MBR transform. Therefore, as the distance between  $X$  and  $Y$  gets small, the probability of a given class for these two points will become similar. Thus, transforming  $x_i$  to  $\log p(x_i|A)$  behaves “smoothly” in that examples that are near to each other will tend to be classified the same.

## 7 Naive Bayes Classifier

Note that the Bayes network above corresponds exactly to a simple (naive) Bayesian classifier that has been used in many studies in the machine learning literature. This classifier, for each class  $C_i$  and feature value  $x_j$ , estimates  $P(x_j|C_i)$  from the training data. A new point is classified into  $C_i$  if  $P(C_i) \prod_j P(x_j|C_i)$  is maximal. This classifier has been evaluated in recent machine learning papers (e.g., [CN89]) and many variations on the Bayesian approach have been considered in a wide range of domains. The classic work on Bayesian classifiers goes back many years, [DH73, Aok65, Hug68, Cha71, CH69, AW68]. Some recent results on naive Bayes classifiers in the machine learning community can be found in [PS90, LI92].

It is important to note that PEBLS computes the same statistics as the naive Bayes classifier. Thus, during training the running time of both methods is the same. However, while Bayes summarizes these statistics in simple rules, PEBLS uses them as part of an MBR classifier. In later sections we show that the use of MBR in conjunction with the probabilistic model of the naive Bayes classifier model expands the representational capabilities of the system and provides better accuracy in empirical studies.

## 8 Empirical Studies on Naturally Occurring Data

In this section, we describe a number of experiments designed to better understand the relative performance characteristics of a naive Bayes classifier specified by a two-level causal net and PEBLS. We shall see that since the model is inaccurate, PEBLS can outperform it on these simple classification tasks, although PEBLS uses a somewhat inaccurate transformation. In the section that follows we shall see that on some tasks an incorrectly specified model can be “fatal” for an MBR method. For completeness, we provide comparisons to nearest-neighbor (NN) using the overlap metric (which counts the number of feature value mismatches between two examples). To get an initial sense of relative performance, we selected eight datasets from the University of California at Irvine’s repository of machine learning databases [Mur95]. For the experiments below, all methods treated the feature values as symbols, even if the values were numbers. In other words, if a feature had values 1, 2, and 3, we would treat those exactly the same as if they were A, B, and C. Continuous valued-features were discretized by dividing them into 10 equal intervals. Solving this type of problem is analogous to predicting the behavior of an electronic circuit based on the color of the resistors and the size of the capacitors: the system has to learn how the symbolic values correspond to other phenomena.

In the case of the letter recognition and mushroom databases, which are both relatively large, we trained the algorithms on 90% of the examples, and tested on the remaining 10%. This process was repeated on ten randomly chosen training sets. The values contained in the table below represent the average of these ten trials. All other tests were conducted using the leave-one-out strategy (i.e., each instance is tested after first training on all other instances in the dataset).

Table 1. Results on eight UCI databases. The highest accuracy for each database appears in boldface.

	Bayes	PEBLs	NN
TicTacToe	69.4	<b>94.4</b>	81.7
Letter	74.6	<b>95.7</b>	88.5
Soybean	81.4	<b>93.2</b>	91.2
Zoology	89.1	95.0	<b>96.0</b>
Iris	90.7	<b>95.3</b>	92.0
Promoter	<b>91.5</b>	90.6	80.5
WI-Breast	<b>96.7</b>	<b>96.7</b>	95.6
Mushroom	99.7	<b>100.0</b>	<b>100.0</b>

While the performance on PEBLS on these benchmarks is quite good, the nature of the concept class is not perfectly understood for these data sets (and, in fact, some of these datasets are known to be quite easy to classify [Hol93]). Therefore, one cannot make any strong conclusions from these results. We primarily wanted to demonstrate that MVDM-based systems can be effective for some standard benchmarks. In [SC92, ZMW92] MVDM is used to deliver a relatively high accuracy on protein secondary structure prediction. That is, it matches or exceeds the accuracy of carefully tuned neural networks on a relatively sparse training data which is rather surprising. We now turn to artificial data to better gauge the specific conditions under which one method outperforms another.

## 9 Tests on Artificial Data

In this section we describe tests that compare the performance of Bayes, PEBLS, and NN when applied to artificial datasets. This section illuminates some clear strengths and weaknesses of simple MBR methods. In particular we devise artificial concept classes where the “advice” given to a memory-based reasoning is so incorrect that it cannot recover and perform a correct classification even if all the instances in the domain are given! Artificially defined datasets are of interest because they allow one to control the nature and distribution of examples, and are thus more amenable to formal analysis. In Section 9.1 we consider a number of basic functional distributions for 2-dimensional data on a finite grid. We show relative learning curves for PEBLS and Bayes. Experiments on additional artificial data (generated by a Markov process) are described in Rachlin et al. [RKSA94], which also reports on artificially generated data in higher dimensional spaces.

### 9.1 Two-dimensional feature space

Our first set of tests considered 10,000 points on an evenly spaced  $100 \times 100$  grid on the unit square. For simplicity, the examples have just two class labels, A and B. Some of the class distributions that we examined are shown in Figure 4. Note that our classification methods treat grid coordinates as arbitrary symbols in a feature space, and thus have no knowledge of the underlying Euclidean space.

In each experiment, we trained the algorithms on some fixed percentage of the points, and then tested on *all* grid points. Thus the tests measured generalization performance on the entire population. All training points were randomly selected from the uniform distribution. Figure 5 shows the relative performances of Bayes, PEBLS, and NN on these class distributions. In all cases, PEBLS did as well as or better than Bayes. It is interesting to note that in the case when the two classes are separated by the function  $y = x^2$ , Bayes does not converge to 100% accuracy even when trained on all possible examples. (See Section 10 for more details and an explanation of this phenomenon.)

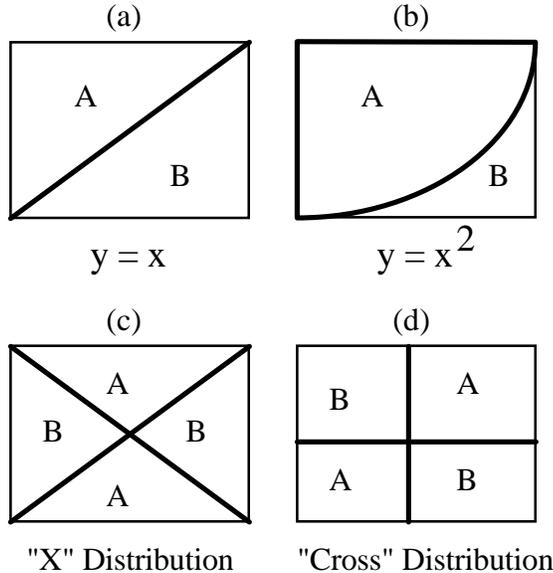


Figure 4: Class distributions with examples uniformly distributed on a 2D lattice

PEBLs and Bayes show some unusual behavior when applied to the “cross” distribution. While Bayes’ accuracy remains at 50% regardless of the training set size, PEBLS at first begins to improve, but then eventually drops off, and when all instances have been seen, the performance returns to the 50% level. This occurs because for a specific value of  $x$  (or  $y$ ), the number of points in Class A and Class B is about the same. As a result, the  $x$  and  $y$  values considered independently provide no hint as to the relative probability of each class. Thus, Bayes can only guess the correct class. However, PEBLS can take advantage of the fact that for any point, it will assign a distance of zero to itself. When the sample size is small, other points will be assigned non-zero distances due to imperfect sampling. Thus, as the training set size begins to increase, so does PEBLS’ accuracy. However, the MVDM similarity function employed by PEBLS will eventually assign a zero distance between all points in the space, once every point in this particular class distribution is included in the training set. When this occurs, PEBLS will also have to guess the correct classification. This explains why PEBLS’ accuracy begins to degrade as the training set size increases beyond a certain point. Figure 5 shows that NN using the overlap metric normally produces poor results, although its accuracy is proportional to the training set size, and thus increases linearly.

## 10 Analytical Results

In this section we present some theoretical analyses of the performance of Bayes vs. PEBLS. These results apply to learning algorithms using MVDM. We focus here on results that provide substantial intuition for the relative strengths and weaknesses of PEBLS (an MVDM-based system) versus naive Bayes. Again, this comparison is interesting since MVDM is based on making the same independence assumptions as naive Bayes, and the computational requirements during learning are identical.

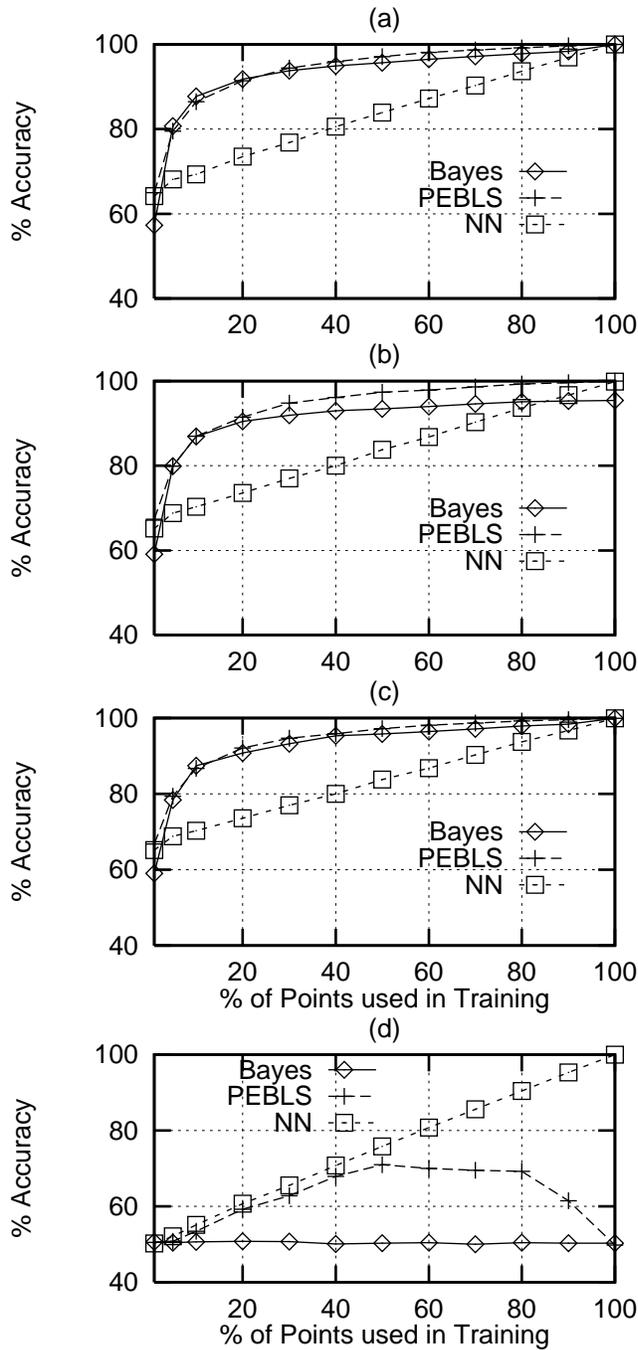


Figure 5: Comparative accuracy of Bayes, PEBS (MVDM), and NN (Overlap) on (a)  $y = x$ , (b)  $y = x^2$ , (c) "x"-distribution, and (d) "cross"-distribution.

## 10.1 Bayes classifier and linearly separable concepts

In section 9.1, we noted that under certain conditions, Bayes did not converge to 100% accuracy even when the classifier was trained on the entire space of examples. On the one hand, this is not very surprising because we are making an independence assumption that is known to be false. However, the precise reasons may not be clear, especially since PEBLS was able to obtain 100% despite making the same assumption. In this section, we analyze the ability of Bayes to learn concept classes in  $Z^2$  when the discriminant function is of the form  $y = ax$ . It is easy to show that if the separator is of the form  $y = x$  on the rectangle  $[0, 0] \times [1, 1]$  (i.e., the separator is a diagonal from the origin to the point  $[1, 1]$ ), then Bayes does converge to 100% accuracy, which is somewhat counter-intuitive at first glance, because the features are dependent on each other. However, with a little analysis, one can prove that for many discriminant functions on the unit square (including  $y = ax$  when  $a \neq 1$ ) Bayes does not converge to 100% accuracy. To illustrate this we consider a *linearly separable* concept class where the discriminating function is  $y = ax$ . We will show that for this function, Bayes will not converge to 100% accuracy when  $a \neq 1$ .

Assume that all points in the unit square above  $y = ax$  are in class A, and all points below are class B. Without loss of generality, assume that  $a \geq 1$ . Note that for a point to be classified by Bayes as Class A, we must have the condition that:

$$P(A|x, y) > P(B|x, y) \tag{2}$$

Making some common independence assumptions, we can write:

$$P(A|x, y) = \frac{P(A)P(x|A)P(y|A)}{P(x, y)} \tag{3}$$

Using Bayes' rule for  $P(x|A)$  and  $P(y|A)$ , we can rewrite equation 3 as:

$$P(A|x, y) = \frac{P(A|x)P(x)P(A|y)P(y)}{P(A)P(x, y)}$$

Performing the same calculation for  $P(B|x, y)$  and applying Eq. 2 yields a new expression for when Bayes will classify a point as A:

$$\frac{P(A|x)P(A|y)}{P(A)} > \frac{P(B|x)P(B|y)}{P(B)} \tag{4}$$

For the class separator  $y = ax$  in the range  $(0, 0) \dots (1, 1)$  we have:

$$P(A) = \frac{1}{2a}; P(B) = 1 - \frac{1}{2a}$$

$$P(A | x) = 1 - ax; P(A | y) = y/a$$

$$P(B | x) = ax; P(B | y) = 1 - y/a$$

Substituting into equation 4 tells us that an example will be classified as A when

$$y > \frac{a^2x}{2a + 2ax - 2a^2x - 1} \tag{5}$$

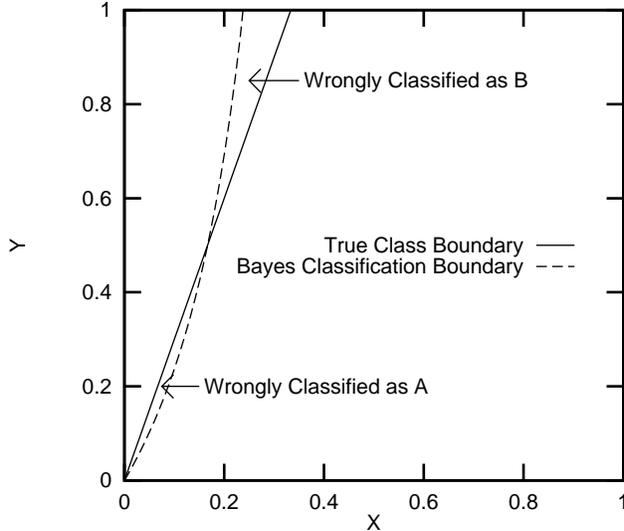


Figure 6: The failure of Bayes for the separator  $y = 3x$ .

For  $a = 1$ , this gives just  $y > x$ ; i.e., the Bayesian classifier will converge to 100% accuracy (which is consistent with the experimental results in Section 9.1). However, if  $y = 3x$  (for example), the condition for being classified as  $A$  is

$$y > \frac{9x}{5 - 12x}$$

This function is plotted in Figure 6 along with the original function  $y = 3x$ . It shows clearly where the Bayesian classifier will make mistakes.

Piecewise integration over the difference between this function and the function  $y = 3x$  allows us to predict a convergence for Bayes of 97.1% in the limit (when all examples in the lattice have been seen). This value has been verified experimentally. A similar analysis for the function  $y = x^2$  predicts an error rate of 4.6%, which is exactly what was observed earlier in Section 9.1.

## 11 MBR and Irrelevant or Noisy Attributes

In this section, we provide a few observations on the utility of MBR in the context of irrelevant or noisy attributes (features). The reader can observe from the discussion in previous sections that MBR performs a process similar (but not identical) to attribute weighting, which may be very useful for symbolic as well as real-valued attributes. In particular it may be helpful for coping with irrelevant attributes. Consider for instance the case where the class  $B$  in a 2-dimensional  $N \times N$  grid is defined by points below the separating plane  $y = N/2$ . In this case the  $x$  attribute is clearly irrelevant. If the examples are sampled from the uniform distribution, then after seeing  $kN$  examples (for some small constant  $k$ ) each  $X$  and  $Y$  coordinate in the grid will have been observed approximately  $k$  times (with high probability). It is clear that  $P(B|x)$  converges rapidly to  $1/2$  (for all  $x$ ).  $P(B|y) = 1$  for all examples below the line and  $P(B|y) = 0$  otherwise. Given a new point  $(x_0, y_0)$  in class  $B$ , which has not been seen before, the MBR transform maps it to:

$$(P(B|x_0), P(B|y_0)) \rightarrow (p(B|x_0), 1)$$

In other words, the transformation-function converges to a function that converts all instances above the line to  $(1/2, 1)$ , and instances below line to  $(1/2, 0)$ . In effect, the value of  $x$  will be ignored, exactly as it should be.

Stronger results than this can be derived. For example, we can show good performance, in the PAC-learnability sense [Val84], after seeing a small number of examples. Let  $e$  be the probability of error of the classifier. Then it is not difficult to show that after seeing  $C/\epsilon(\ln 1/\delta)$  examples (where  $C$  is some constant),  $P(e > \epsilon) < 1 - \delta$ . This argument trivially holds for the case of one relevant attribute and  $D - 1$  irrelevant attributes on an  $N^D$  grid. The number of examples necessary to achieve PAC-learnability in this case is also  $C/(\epsilon \ln 1/\delta)$ , where the constant  $C$  depends linearly on  $D$ . In contrast, the standard NN algorithm will require  $N^D$  examples. In general, the convergence rate is likely to be better for nearest neighbor voting procedures such as k-NN.

A similar argument can be made for attributes that include uniformly distributed Gaussian noise. This informal discussion suggests that the MBR transform may be helpful for coping with irrelevant or noisy attributes in symbolic as well as discretized real-valued spaces. We note, that in general, the worst case convergence of nearest-neighbor procedures including MBR in  $D$  dimensions is exponentially slow. However, as pointed out earlier, when the domain model is accurate the learning rate improves dramatically.

## 11.1 MVDM learning functions

In Section 9.1, we saw concept classes that both the naive Bayes classifier and PEBLS had difficulty learning. However, given the poor performance of the naive Bayes classifier for  $y = ax$ , we would like to see if PEBLS has similar difficulty learning this family of concept classes. This is the question of statistical consistency, namely at the limit (or in finite cases when we see all the examples) we want our system to produce only correct classifications. As we have seen before, it is not obvious that MVDM-based systems achieve 100% classification accuracy even when it sees all the examples in the domain; in fact it often cannot. Somewhat surprisingly, we can show that PEBLS can actually learn any concept on  $Z^D$  provided the classes are separated by a function. We recall again that our algorithms treat all inputs as arbitrary symbols in a feature space, and thus have no knowledge of the underlying Euclidean space. Thus, the standard NN convergence proofs are not immediately applicable to this case.

The next theorem addresses this issue and provides a condition that guarantees the consistency of MVDM at the limit (on finite domains). Note that our theorems are not learnability theorems, but rather address representational issue. We note, that by making some standard smoothness assumptions on the discriminant function, one can extend the consistency result to prove convergence, that is, learnability at the limit; this type of analysis is beyond the scope of this paper.

**Theorem 1** *Let  $G$  be a finite  $N \times N$  grid. Let  $A$  be a concept that is a subset of  $G$ , and assume  $A$  is separated from  $B$  by a function  $f$ .  $f$  is a function from  $1 \dots N$  to  $1 \dots N$ . Let  $A$  be the region above the function  $f$ ; i.e.,  $(x, y)$  is in  $A$  iff  $f(x) \geq y$  and  $(x, y)$  is in  $B$  iff  $f(x) < y$ . Then the MVDM classifier will achieve 100% accuracy on  $A$  and  $B$  at the limit.*

Proof: The proof is by contradiction. Assume  $(x_1, y_1)$  is a point in  $A$  that PEBLS classifies incorrectly. Since the distance from  $(x_1, y_1)$  to itself is zero, this implies there is a point  $(x_2, y_2)$  in  $B$  whose distance to  $(x_1, y_1)$  is also zero. From the definition of MVDM, this means that

$$|P(A|x_1) - P(A|x_2)| = 0$$

$$|P(A|y_1) - P(A|y_2)| = 0$$

Since  $P(A|x_i) = f(x_i)$ , we have  $f(x_2) = f(x_1)$ . That is,

$$y_2 > f(x_2) = f(x_1) \geq y_1$$

Therefore,  $y_2 > y_1$ . Let  $A_1$  be the set of all  $x$  such that  $f(x) \geq y_1$ . Similarly, let  $A_2$  be the set of all  $x$  such that  $f(x) \geq y_2$ . If  $x$  is in  $A_2$ ,  $f(x) \geq y_2$ . Then since  $y_2 > y_1$ ,  $f(x) > y_1$  and therefore  $x$  is in  $A_1$ . Thus  $A_2$  is a subset of  $A_1$ . However,  $P(A|y_1)$  is the size of  $A_1$  and  $P(A|y_2)$  is the size of  $A_2$ ; therefore, since  $P(A|x_1) = P(A|x_2)$ ,  $A_1$  and  $A_2$  are the same size. So we have two sets in  $Z^2$ , one of which is contained in the other, and the sizes of which are equal; thus the sets must be equal. However,  $x_1$  is a member of  $A_1$ , but it cannot be a member of  $A_2$  because  $y_2 > f(x_1) \geq y_1$ .  $\square$

This **consistency** theorem basically shows that at the limit, MVDM-based systems can learn any concept in  $Z^2$  separated by a function. A more general theorem easily follows for concept classes on multidimensional grids. However, in multiple dimensions it is difficult to derive an equally general conditions on achieving consistency and the technical details are involved. For instance, if the decision boundary is given by a function of  $d - 1$  dimensions in a  $d$ -dimensional space, we do not necessarily get 100% accuracy with MVDM. (Simply extend the cross-concept discussed above to three dimensions). However, it is easy to see that we can learn any linearly separable concepts using MVDM.

The goal above was merely to establish conditions that will guarantee that MVDM classifiers achieve consistency, i.e., can learn accurately a much more general family of classification tasks than the original model used to induce the transformation would allow by itself. Clearly, a classifier rarely sees every possible example. Nevertheless, this is a useful observation in view of the limitations of a simple Bayes classifier, which cannot even learn linearly separable concepts when all the examples are given.

A natural question to ask is what is the convergence rate of MVDM in the limit for various other classes of concepts. We already pointed out that MVDM speeds up the finite sample convergence rate when there are many irrelevant attributes. However, one can show that the worst case behavior of MVDM is similar to standard 1-NN algorithms, which can be exponentially slow even when the examples are sampled using a uniform distribution.

## 11.2 Bayes learning implies MVDM learning

We next decided to investigate whether there are any concepts for which the Bayesian classifier obtains 100% accuracy while PEBLS does not. Again, to simplify the discussion, let's consider the situation at the limit; that is, when all possible examples of the concept have been seen.

It turns out that, for *any* finite domain, PEBLS will *always* achieve 100% accuracy if Bayes is known achieve to 100% accuracy. The proof is straightforward and is given below.

**Theorem 2** *Given a finite domain for which the Bayesian classification rule achieves 100% accuracy at the limit, the PEBLS program using MVDM will also achieve 100% accuracy at the limit.*

Proof: In the limit, a classifier will see every possible example in a domain. Since we know that Bayes classifies every point correctly, we know that for any point  $p_1 = (x_1, y_1)$  in class  $A$ ,

$$\frac{P(A|x_1)P(A|y_1)}{P(A)} > \frac{P(B|x_1)P(B|y_1)}{P(B)}$$

This must be true in order for Bayes to classify the point correctly as  $A$ .

Now consider how PEBLS will classify the point. Since it has seen the point before, it will always find the point itself as its own nearest neighbor, since the distance from a point to itself is always 0 using MVDM. Thus it will classify it correctly unless there is another point  $p_2 = (x_2, y_2)$  that is labelled  $B$  that has distance 0 from  $p_1$ . As explained earlier, the distance between  $p_1$  and  $p_2$  as computed by the MVDM is the following:

$$|P(A|x_1) - P(A|x_2)| + |P(B|x_1) - P(B|x_2)| + \\ |P(A|y_1) - P(A|y_2)| + |P(B|y_1) - P(B|y_2)|$$

Assuming that such a point exists, then this distance is zero. Therefore all four terms must be zero, and thus  $P(A|x_1) = P(A|x_2)$ ,  $P(B|x_1) = P(B|x_2)$ , etc. Now, since  $p_2$  is in class  $B$ , then

$$\frac{P(A|x_2)P(A|y_2)}{P(A)} < \frac{P(B|x_2)P(B|y_2)}{P(B)}$$

because the Bayesian classifier classifies it correctly. But since  $P(A|x_1) = P(A|x_2)$  etc., we can substitute  $x_1$  and  $y_1$  for  $x_2$  and  $y_2$  in this equation, which gives us a contradiction. Therefore no point in class  $B$  will have distance 0 from any point in class  $A$ .  $\square$

## 12 Conclusion

We have described a framework for memory-based reasoning. This framework advocates the synthesis of memory-based reasoning algorithms based on probabilistic models (specified by the modeller or domain expert). The models are used to induce a notion of relevance on the domain. Given a query about a desired action in some state, we retrieve the relevant neighborhood (as induced by the model) and perform the appropriate action by interpolating among the instances in that neighborhood. The definition of relevant neighborhood is adaptive and will change as the database changes. It will also vary depending on the desired action.

Our analysis considered a popular class of MBR procedures (MVDM) induced by a very simple Bayes network (the naive Bayes classifier). We showed that MBR has a greater representational capacity than the probabilistic model used to induce the transformation and therefore can be used for more general tasks. In particular, despite the computational similarity during training, MBR and the Bayesian classifier display several rather striking differences in their classification accuracy when applied to symbolic domains. We demonstrated that the PEBLS system, which uses the naive Bayes model to induce the MBR transform, outperforms naive Bayes under a wide range of conditions, including cases in which the instances are both dense and sparse with respect to the feature space, and in some simple domains. Previous work demonstrated the effectiveness of this class of MBR procedures in complex scientific domains ([ZMW92, CS93]).

In the limit, when instances completely fill the space, we show that if Bayes can learn the distribution perfectly, then the system PEBLS that uses MVDM will always learn perfectly. Indeed, while Bayes cannot perform perfectly for some simple class definitions, PEBLS can achieve perfect accuracy on a wider range of classes. We also established some advantages of MVDM classifiers for both symbolic and real-valued attributes. We showed that MVDM may be helpful for coping with irrelevant and noisy attributes. Thus, the convergence rate of MVDM's accuracy is likely to be better than standard nearest neighbor when the number of irrelevant attributes is large. In some important cases, the convergence rate is dependent only on the number of relevant attributes.

Much is left to be done. We are currently in the process of building a general system based on this framework. The system will provide a programming notation for specifying general probabilistic models, probabilistic model learning, automatic MBR transforms based on probabilistic models, retrieval of relevant contexts, and finally prediction by local interpolation. Our framework opens the door for a number of new variants of MBR which have not been explored in the literature. For instance, few current schemes use both adaptive neighborhoods and local learning. This will be supported naturally in this general system.

While we did not elaborate on this in depth, our approach suggests a generic way to incorporate “knowledge-based advice” into memory-based reasoning procedures. The advice in this case corresponds to a set of rules that can induce the structure of a probabilistic network. Then learning procedures can estimate the conditional probabilities that must be learned from data. The learned network is then used to induce an MBR transform on the data, mapping the data into a probability distribution space. That is, we use the probabilistic network to compute a set of probabilistic “features” which are included in the subsequent MBR inference. This perspective suggests several interesting possibilities. For instance, it suggests the possibility of providing approximate models, emphasizing the ability to compute probabilistic features efficiently, and then relying to MBR inference to “correct” for the simplified independence assumptions made by the modeler.

Our framework suggests some interesting cognitive explorations as well, since memory-based reasoning has many desirable properties as a plausible form of cognitive activity. In this paper we focused on the practical aspects of this proposal, namely taking advantage of probabilistic model building algorithms, and suggesting a method to improve on their representational capabilities and accuracy.

## Acknowledgements

Part of the development of the MBR framework was done while the first author visited NEC Research Institute and Princeton University. The preliminary experiments with PEBLS were conducted by John Rachlin at Johns Hopkins University and supported in part by National Science foundation under Grants No. IRI-9116843, IRI-9223591 and IRI-9220960.

## References

- [AKA91] D. Aha, D. Kibler, and M. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1), 1991.
- [AMS95] Christopher Atkeson, Andrew Moore, and Stefan Schaal. Locally weighted learning. submitted for publication, April 1995.
- [Aok65] M. Aoki. On some convergence questions in bayesian optimization problems. *IEEE Trans. on Automatic Control*, 10:180–182, 1965.
- [AW68] R. Albrecht and W. Werner. Error analysis of a statistical decision method. *IEEE Transactions on Information Theory*, 10:34–38, 1968.
- [Ben75] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

- [BH95] J. S. Breese and D. Heckerman. Decision theoretic case-based reasoning. In *Proc. AI and Statistics*, pages 56–63, 1995.
- [BV92] L. Bottou and V. Vapnik. Local learning algorithms. *Neural Computation*, 4:888–900, 1992.
- [CH69] B. Chandrasekan and T. Harley. Comments on the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 15:421–423, 1969.
- [Cha71] B. Chandrasekan. Independence of measurements and the mean recognition accuracy. *IEEE Transactions on Information Theory*, 17:452–456, 1971.
- [CMSW92] R. Creecy, B. Masand, S. Smith, and D. L. Waltz. Trading MIPS and memory for knowledge engineering. *Communications of the ACM*, 35(8):48–64, 1992.
- [CN89] P. E. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–284, 1989.
- [CS93] S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57–78, 1993.
- [Das91] Belur V. Dasarathy, editor. *Nearest neighbor (NN) norms: NN pattern classification techniques*. IEEE Computer Society Press, Los Alamitos, CA, 1991.
- [DH73] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
- [Ede94] S. Edelman. Representation, similarity and the chorus of prototypes. *Minds and Machines*, 1994.
- [Fri94] J. H. Friedman. Flexible metric nearest neighbor classification. Technical report, Stanford University, Statistics Dept., November 1994.
- [GS94] Z. Gilboa and D. Schmeidler. Case-based decision theory. Technical report, Northwestern University, 1994.
- [Hol93] R. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–90, 1993.
- [HT94] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. Technical report, Stanford University, Statistics Dept., December 1994.
- [Hug68] G. Hughes. On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14:55–63, 1968.
- [JTT92] D. Jones, W. Taylor, and J. Thornton. The rapid generation of mutation data matrices from protein sequences. *CABIOS*, 8(3):275–282, 1992.
- [KM78] J. B. Kruskal and Wish M. *Multidimensional Scaling*. Sage Publications, 1978.
- [Lau95] S. L. Lauritzen. The em algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, pages 191–201, 1995.
- [LI92] P. Langley and W. Iba. An analysis of Bayesian classifiers. In *Proc. Tenth Natl. Conf. on Artificial Intelligence*, pages 223–228, Menlo Park, CA, 1992. AAAI Press.

- [MAS95] A. Moore, C. Atkeson, and S. Schaal. Memory-based learning for control. *Artificial Intelligence Review (to appear)*, 1995.
- [MK93] P. Maes and R. Kozierok. Learning interface agents. In *Proc. of the Eleventh National Conf. on Artificial Intelligence*, pages 459–465, Washington, D.C., 1993. MIT Press.
- [MLW92] B. Masand, G. Linoff, and D. L. Waltz. Classifying news stories using memory based reasoning. In *Proceedings of the SIGIR*, pages 59–65, 1992.
- [Mur95] P.M. Murphy. UCI repository of machine learning databases – a machine-readable data repository. Maintained at the Department of Information and Computer Science, University of California, Irvine. Anonymous FTP from ics.uci.edu in the directory pub/machine-learning-databases, 1995.
- [Nad64] E.A. Nadaraya. On estimating regression. *Theory of Probability and Its Applications*, 9:141–142, 1964.
- [Omo87] S. Omohundro. Efficient algorithms with neural network behavior. *Complex Systems*, pages 273–347, 1987.
- [Omo89] S. Omohundro. Five balltree construction algorithms. Technical Report 89-063, International Computer Science Institute, Berkeley, California, 1989.
- [Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- [PS90] M. Pazzani and W. Sarrett. Average case analysis of conjunctive learning algorithms. In *Proc. of the Workshop on Machine Learning*, pages 339–347, 1990.
- [PTL93] F. Pereira, N. Tishby, and L. Lee. Distributional clustering of English words. In *Proc. of the 30-th Meeting of the Assoc. for Computational Linguistics*, pages 183–190, 1993.
- [Qui93] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.
- [RKSA94] J. Rachlin, S. Kasif, S. Salzberg, and D. Aha. Towards a better understanding of memory-based and bayesian classifiers. In *Proc. of the Eleventh Internatl. Conf. on Machine Learning*, pages 242–250, New Brunswick, NJ, July 1994.
- [Roy66] R.M. Royall. *A Class of Nonparametric Estimators of a Smooth Regression Function*. PhD thesis, Department of Statistics, Stanford University, 1966.
- [SC92] S. Salzberg and S. Cost. Predicting protein secondary structure with a nearest-neighbor algorithm. *Journal of Molecular Biology*, 227:371–374, 1992.
- [SDLC93] D.J. Spiegelhalter, A.P. Dawid, S.L. Lauritzen, and R.G. Cowell. Bayesian analysis in expert systems. *Statistical Science*, 8(3):219–283, 1993.
- [She80] R. N. Sheppard. Multidimensional scaling, tree fitting and clustering. *Science*, pages 1317–1323, 1980.
- [SW86] C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, 1986.

- [TC91] H. Turtle and W. Bruce Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 1991.
- [TSM85] D. M. Titterington, A. Smith, and U. E. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley, New York, 1985.
- [Tve77] A. Tversky. Features of similarity. *Psychological Review*, 84:324–357, 1977.
- [Val84] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.
- [VB93] V. Vapnik and L. Bottou. Local learning algorithms for pattern recognition and dependency estimation. *Neural Computation*, 5:893–909, 1993.
- [YL93] T.-M. Yi and E. Lander. Protein secondary structure prediction using nearest-neighbor methods. *Journal of Molecular Biology*, 232:1117–1129, 1993.
- [ZMW92] Xiru Zhang, Jill P. Mesirov, and David L. Waltz. A hybrid system for protein secondary structure prediction. *Journal of Molecular Biology*, 225:1049–1063, 1992.