Data as Ensembles of Records: Representation and Comparison

Nicholas R. Howe NIHOWE@CS.CORNELL.EDU
Department of Computer Science, Cornell University, Ithaca, NY 14853

Abstract

Many collections of data do not come packaged in a form amenable to the ready application of machine learning techniques. Nevertheless, there has been only limited research on the problem of preparing raw data for learning, perhaps because widespread differences between domains make generalization difficult. This paper focuses on one common class of raw data, in which the entities of interest actually comprise collections of (smaller pieces of) homologous data. We present a technique for processing such collections into high-dimensional vectors, suitable for the application of many learning algorithms including clustering, nearest-neighbors, and boosting. We demonstrate the abilities of the method by using it to implement similarity metrics on two different domains: natural images and measurements from ocean buoys in the Pacific.

1. Introduction

A quick perusal of the UCI repository of machine learning data sets (Blake & Merz, 1999) reveals that the most frequently cited entries consist of data that are condensed into a convenient format easily digested by most machine learning algorithms. Typically such data consist of a set of instances, perhaps already divided into subsets for training and testing. Each individual instance is described by a set of features X, including a class feature that the learning algorithm must predict accurately.

Although the data sets in the UCI repository provide a convenient testbed for new ideas in machine learning, they do not fully represent the difficulty of solving problems encountered in the real world. Often the hardest part of applying learning methods to a previously unexamined task is the codification of the problem in a form that machine learning algorithms can handle. This step has already been performed on most of the repository data, and usually there is no access to the original form of the data set or documentation on how it was transformed. Thus there is need for research and discussion on the analysis of data in a more natural format.

Some types of raw data may not even be amenable to expression in the standard feature-value format, and therefore require special treatment. For example, some tasks involve learning properties of entities that are themselves made up of an arbitrary number of similar components. As a concrete example, consider investigating properties of credit accounts, where each account is represented as the set of transactions posted to the account. This paper focuses on such collective entities, or *ensembles*, in particular when the constituent components, or *records*, happen to have a concise featural description. Instead of devising features that summarize the ensemble as a whole, which can obscure useful information, we adopt a description that preserves important details of each individual record. The algorithm presented in this paper automatically generates a feature vector of uniform size from any ensemble, provided that the component records have a standard feature-vector description. This provides a novel way to look at data from many domains, including computer event logs, transactional data such as credit card histories, and other areas where ensemble data are involved. We present insight from implementations of the technique in two different domains: natural images and ocean measurements.

This work is far from the first to look at descriptions of data other than the canonical feature-value representation. Researchers in case-based reasoning often adopt complex or unusual case descriptions (Kolodner, 1993). For example, Branting (1991) looks at legal cases represented as graphs. Additionally, the field of reinforcement learning may be thought of as employing data in a nonstandard format (Kaelbling et al., 1996). While dealing with nonstandard data representations, these fields have not focused on the type of ensemble data examined here.

The remainder of this paper describes our treatment of ensemble data. Section 2 gives a description of the algorithms for processing and comparing data ensembles. Section 3 presents the two test domains and examines the performance of the system on them. Finally, Section 4 concludes with a discussion of possible directions for future research.

2. Handling Ensemble Data

We address domains in which the task requires learning properties of ensembles of records, where each ensemble may contain an arbitrary number of records. Furthermore, we assume that each record is described by a simple feature vector. To be precise, we will give a formal description of such an ensemble before describing how it is processed.

A record is an arbitrary set of m feature-value pairs, $r = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$, where $X = \{x_1, x_2, ..., x_m\}$ is a consistent set of features shared by all records in the data, and the y_j are values of those features. (In some domains, features may be missing from some records, and thus the features of r form a subset $X_r \subseteq X$.) An ensemble is simply a collection (possibly a weighted collection) of records, i.e., a set of ordered pairs $\{(r_1, w_1), (r_2, w_2), \ldots, (r_{n_e}, w_{n_e})\}$ where r_i is a record and $w_i \in \mathbb{R}_+$ is a positive real weight. As a concrete example, in a credit card domain each ensemble might represent one account. Its component records would be the charges posted to the account, each described by a feature set, such as $\{amount, charge\_date, payment\_date\}$. Some accounts would have fewer charges posted than others.

2.1 Data Preparation

Processing of ensemble data into a more manageable form takes place in two steps. First, we express the individual records in a discrete space $\mathcal{M}$, which is a discretization of the original feature space. Once this is done, a one-to-one function transforms the entire ensemble into a vector in a high-dimensional space $\mathcal{F}$. Vectors in this space may be thought of as joint histograms of the original record feature values. All subsequent processing takes place in $\mathcal{F}$, which is better suited to the application of standard machine learning techniques.

Records are mapped into space $\mathcal{M}$ by discretizing each feature x_j. Points in $\mathcal{M}$ are tuples of the discretized feature values. Thus, to map a record to a point $\mathbf{m}$ in $\mathcal{M}$ we simply determine the appropriate bin for each of its feature values. For the credit card example just described, a hypothetical record might map to a point like *($50-100,Jun99,Oct99)*. The discretization of feature value for the results reported in this paper has been done by hand, but automated techniques exist and might be applied (Fayyad & Irani, 1993).

Ensembles are represented as a set of ordered pairs, each consisting of a point in $\mathcal{M}$ and an associated positive weight. (Weights arise naturally in some domains, or can be set uniformly to one if not needed. If two or more records are described by the same $\mathbf{m}$, they are represented by a single ordered pair with weight equal to the sum of the individual weights.) We refer to this as the $\mathcal{M}$-representation $R_\mathcal{M}(e)$ of the ensemble e.

$$R_\mathcal{M}(e) = \{(\mathbf{m}_1, w_1), (\mathbf{m}_2, w_2), ..., (\mathbf{m}_{n_e}, w_{n_e})\} \quad (1)$$

where n_e is the number of records in the ensemble, and $w_i \in \mathbb{R}_+$ is a weight.

Space $\mathcal{F}$ has exactly one dimension corresponding to each point of space $\mathcal{M}$. Thus $\mathcal{F}$ is equivalent to $\mathbb{R}_+^{N_\mathcal{F}}$, where the dimensionality $N_\mathcal{F}$ is the product of the number of bins used for all the features X. For example, in the simple credit account domain described above, there might be ten bins for the *amount* feature, and 20 each for the two date features, giving $\mathcal{F}$ a total of $10 \cdot 20 \cdot 20 = 4000$ dimensions. Each of these dimensions represents a particular range of values for transaction amount, charge date, and payment date. We establish a one-to-one correspondance between points in $\mathcal{M}$ and the standard orthonormal basis vectors of $\mathcal{F}$. Thus there exists a bijective mapping f between $\mathcal{M}$-representations and vectors in $\mathcal{F}$.

$$f\left(R_\mathcal{M}(e)\right) = \sum_{i=1}^{n_e} w_i F(\mathbf{m}_i) \quad (2)$$

where $F(\mathbf{m})$ is the mapping from points in $\mathcal{M}$ to basis vectors of $\mathcal{F}$. This means that ensembles with unique $\mathcal{M}$-representations also have a unique representation in $\mathcal{F}$.

2.2 An Example

A concrete example may illustrate the creation of $\mathcal{F}$-representations. Suppose that an account in the credit-card domain has the following transactions posted (ignoring interest charges for simplicity):

Date	Action
June 16, 1999	Charge $75
September 2, 1999	Charge $20
September 28, 1999	Charge $35
October 13, 1999	Paid $130

An $\mathcal{M}$-representation for this account would be

{((*$50-100,Jun99,Oct99*),1),
((*$0-50,Sep99,Oct99*),2)}.

This corresponds to the $\mathcal{F}$-vector

$$e_{(\$50-100,Jun99,Oct99)} + 2e_{(\$0-50,Sep99,Oct99)}$$

where $e_{(amount,charge\_date,payment\_date)}$ is the basis vector in one of the dimensions of $\mathcal{F}$, as described by the subscript.

2.3 Vector Comparisons

In many applications, the significant information resides in the distribution of he records in $\mathcal{M}$ space rather than the actual number of records. If this is so, then the natural distance metric to use is the cosine metric, which measures the angular deviation between two vectors and ignores their length. The cosine metric has been extensively used for text retrieval (Salton, 1989). Thus in $\mathcal{F}$ space, the distance between two vectors $\mathbf{f}_1$ and $\mathbf{f}_2$ is

$$D_{\mathcal{F}}(\mathbf{f}_1, \mathbf{f}_2) = \cos^{-1}\left(\frac{\mathbf{f}_1 \cdot \mathbf{f}_2}{\sqrt{(\mathbf{f}_1 \cdot \mathbf{f}_1)(\mathbf{f}_2 \cdot \mathbf{f}_2)}}\right) \qquad (3)$$

While the plain cosine difference metric may work well in some situations, a number of considerations suggest the use of a slightly more complicated form than Equation 3. If continuous variables are discretized to form $\mathcal{M}$ space, then their relative ordering is lost. Even for discrete variables, some pairs of values indicate greater similarity than others. We would like to capture this information in the ensemble distance metric, and can do so by modifying Equation 3 to include a similarity matrix with cross terms:

$$D_{\mathcal{F}}(\mathbf{f}_1, \mathbf{f}_2) = \cos^{-1}\left[\frac{\mathbf{f}_1^T \mathbf{S} \mathbf{f}_2}{\sqrt{\left(\mathbf{f}_1^T \mathbf{S} \mathbf{f}_1\right)\left(\mathbf{f}_2^T \mathbf{S} \mathbf{f}_2\right)}}\right]. \qquad (4)$$

Here $\mathbf{S}$ is a matrix whose off-diagonal entries account for the varying similarity of different feature values. It should be a symmetric matrix so that distances are symmetric, and can be devised to have a Cholesky factorization $\mathbf{S} = \mathbf{T}^T\mathbf{T}$. Under these conditions, Equation 4 can also be interpreted as the simple cosine difference between two transformed vectors $\mathbf{T}\mathbf{f}_1$ and $\mathbf{T}\mathbf{f}_2$.

The choice of $\mathbf{S}$ will greatly affect distance measurements and therefore any results based upon them. We describe the system that we have used for generating suitable matrices, but numerous alternative possibilities also exist. Our approach has the advantage of allowing the user significant control over how much individual features contribute to the final distance, without an overly complex interface.

We form $\mathbf{T}$ (and hence $\mathbf{S}$) as the Kronecker product (or direct matrix product) of smaller matrices $\{\mathbf{T}_1, \mathbf{T}_2, ..., \mathbf{T}_m\}$, each corresponding to one of the features x_j. Each entry in the Kronecker product matrix is the product of exactly one entry from each of the Kronecker factor matrices. Intuitively, this represents the cross terms in $\mathbf{T}$, corresponding to the match between two points in $\mathcal{M}$ (or components in $\mathcal{F}$), as the product of the cross terms of the $\mathbf{T}_j$ matrices, each representing the match in one individual feature. Thus using the Kronecker product allows us to focus on one feature at a time, and also allows for significant computational efficiencies as explained in Section 2.4.

Each $\mathbf{T}_j$ is a square matrix of a size equal to the number of bins in the discretization of x_j. For features that were originally continuous, we use cross terms that decay exponentially with the distance between the bin centers:

$$\mathbf{T}_j(k, l) = (p_j)^{\Delta_j(y'_k, y'_l)}. \qquad (5)$$

Here $\Delta_j(y'_k, y'_l)$ is the distance between the centers of bins k and l, and $p_j \in [0, 1]$ is a parameter that controls the size of the cross terms. If $p_j = 0$, then only exact matches are allowed, while $p_j = 1$ means that any value of feature x_j matches all others equally well. Intermediate values of p_j result in better matches for closer values. Thus the setting of p_j is a knob by which the user can exert control over the system. Although neither of the example data sets we examine in Section 3 contain discrete features, $\mathbf{T}_j$ matrices for discrete features can be created via an analogous process, by using something like the value-difference metric (Stanfill & Waltz, 1986).

2.4 Practical Concerns

Scalability is a valid concern in the system so far outlined, but perhaps not as large a concern as might at first appear. Clearly, restraint must be used in discretizing features, since the dimensionality of $\mathcal{F}$ is the product of the number of bins in each feature. Nevertheless, the technique scales sufficiently for the analysis of interesting problems, as the experiments of Section 3 demonstrate. Here we explain a few key optimizations that make operations in $\mathcal{F}$ more feasible.

The slowest calculation to perform in a straightforward manner is multiplication by $\mathbf{S}$. Normally this would be an $N_{\mathcal{F}}^2$ operation, where $N_{\mathcal{F}}$ is the dimensionality of $\mathcal{F}$. However, by using $\mathbf{S}$ that is a Kronecker product, the matrix multiplication can be computed in time linear in $N_{\mathcal{F}}$ (Graham, 1981). (Essentially, the result is calculated through repeated multiplications by the smaller $\mathbf{S}_j$ matrices.) Furthermore, when comparing one ensemble represented by vector $\mathbf{f}_1$ to many others, the product $\mathbf{f}_1\mathbf{S}$ need only be computed once. For fixed $\mathbf{S}$ the denominator in Equation 4 can be precomputed; thus the calculation of multiple cosine differences can be quite fast. Under ideal circumstances, it amounts to one array lookup and two floating point operations per record.

3. Implementation and Evaluation

The algorithms described in the previous section define a similarity metric on arbitrary ensembles of records. As yet we have said nothing about how to evaluate the technique, nor what role it might play in a complete learning system. Fortunately, similarity lies at the heart of many learning systems. Nearest-neighbor algorithms, clustering, retrieval, and even sophisticated techniques like boosting all apply naturally in a suitable metric space, such as $\mathcal{F}$. To evaluate the usefulness of the $\mathcal{F}$-representation of ensembles, we must show that a proposed metric promotes effective learning. In particular, if entities with similar properties tend to be mutually similar under the cosine metric, then those properties can be successfully learned. This paper will look at some simple similarity-based tasks as an indication that more complex algorithms based on similarity may also use the cosine metric successfully. Thus the transformations described in this paper can provide the first stage of a potential learning system, packaging raw ensemble data for analysis by a standard learning algorithm.

Although it is easy to conceive of domains where data are structured as ensembles of records, actually acquiring such data is more difficult. In many cases, domains that fit the ensemble paradigm are described using summary information, with the raw data in ensemble form not available. For example, the UCI machine learning repository contains a credit screening domain, but information on individual accounts is condensed into sixteen features and there is no listing of transactions. Nevertheless, some candidate domains can be found. We describe the application of the techniques described above to two tasks: image retrieval in collections of natural images, and analysis of ocean climate measurements from the Pacific. The point of these evaluations is to show that machine learning can be done under the ensemble of records paradigm. Because the two domains differ significantly, each task reveals different aspects of the approach.

3.1 Natural Images

The ideas presented in this paper were first developed as part of an effort to improve on current algorithms for image retrieval. In the spirit of Impressionist art, images can be viewed as collections of many small patches with differing color, texture, and spatial properties. Visually similar images should comprise similar collections of patches, so a means of comparing patch collections would imply a means of comparing images. This observation motivates the current work.

To prepare an image for comparison, we first divide it into about 500 small patches using a simple local segmentation scheme (Felzenszwalb & Huttenlocher, 1998) and arbitrarily divide further any large regions generated. Next we record for each patch the values of color (in the Hue-Saturation-Value color space), texture (as measured by a local filter), and spatial location (normalized to the image dimensions). These are discretized to produce the $\mathcal{M}$-representation of the image. We use 28 color bins, 21 spatial bins, and three texture bins. Finally, after conversion to $\mathcal{F}$ space, we compare the image with others using Equation 4. Our $\mathbf{S}$ is created as described in Section 2.3, with the spread parameters ranging from 0.1 to 0.001. We adjusted the parameters by hand in a small pilot study, using a gradient descent approach.

We evaluate the ensemble-based algorithm on several tasks, in comparison with both a baseline approach (color histograms (Swain & Ballard, 1991)) and a state-of-the-art algorithm (color correlograms (Huang et al., 1997)) designed specifically for image retrieval. Our approach does consistently better than the baseline and competes well with the specialized retrieval algorithm.

The first test we apply uses artificially altered or degraded images as queries, with the task of retrieving the original from a collection of 19,000 images. (All images come from a commercially available collection published by Corel.) The query alterations come in three flavors: *Crop*, where 50% of the image area is trimmed around the borders, *Jumble*, where the image is broken into 16 rectangular tiles that are reshuffled at random, and *Low-Con*, where the image contrast is degraded. We test using a randomly selected sample of 1000 query images, and record the rank at which the original is retrieved. The two sets of numbers listed for the ensemble technique represent varying conditions: the first uses a default setting for $\mathbf{S}$, while the second uses an $\mathbf{S}$ that is chosen to be more appropriate to the specific task (i.e., ignoring features that are not useful.) The latter case simulates tasks in which domain knowledge is available *a priori* to guide the selection of $\mathbf{S}$.

The results, in Table 1, look more or less as one might expect. The histogram of the ranks is quite skewed and exhibits a long tail: most originals are recovered at a relatively low rank, but a few are not. To capture this behavior, two numbers are listed for each condition. The median rank represents the bulk of the cases, while the mean rank reveals the severity of the tail. Standard deviation (σ) is also given in the table, although due to the skewed distributions it shows the same trend as the mean.

The histogram method does worse than the others, except on the *Jumble* alteration, which of course does not affect the color histogram at all. The ensemble approach with default $\mathbf{S}$ does worse on this task because unlike the other techniques it incorporates spatial features explicitly representing the position of elements that are moved around in this test. However, when $\mathbf{S}$ is tuned to the task by smoothing out the spatial feature, it does much better. The tuned ensemble method does well on all three tasks.

We also look at two versions of a classification task where images come from one of several visually self-similar categories. Although the category definitions are somewhat arbitrary, this task provides some indication of whether using one image as a query will retrieve a related image. Because the top hits are generally most important for image retrieval, we focus on the top images retrieved rather than the full recall-precision curves. (The test is equivalent to one-nearest-neighbor classification under leave-one-out cross validation.) For this task we augment the patch descriptions with an additional feature measuring similarity to neighboring patches in the image. This feature records whether the patch matches the color and texture of all, some, a few, or none of its

Table 1. Rank of target for altered-image queries, out of 19,000 images. (Lower numbers are better.)

		Crop	*Jumble*	*Low-Con*
Histogram	median	18	(1)	86.5
	mean	126.6	(1)	350.3
	σ	310.0	(1)	795.5
Correlogram	median	1	1	5
	mean	12.4	2.0	83.6
	σ	53.7	6.4	288.7
Ensemble (Default)	median	1	26	1
	mean	38.9	205.2	18.2
	σ	181.4	529.6	155.6
Ensemble (Tuned)	median	1	1	1
	mean	17.0	1.2	22.6
	σ	103.9	1.8	242.6

neighbors. (The ease with which newly constructed descriptive features may be incorporated is a convenient feature of the ensemble approach.) The results are shown in Table 2. The ensemble approach performs better than the baseline overall, and competes favorably with the algorithm specialized for image retrieval on both of the two test sets.

3.2 Ocean Buoy Measurements

Besides the experiments with natural images, we also examine climate measurements taken from ocean buoys moored in the equatorial Pacific Ocean (Bay, 1999). This is a natural domain for application of the ensemble approach, since each buoy measurement forms a unit that must be aggregated with others to form a picture of the climate conditions at any one time. The buoy data set offers an interesting contrast with the natural image data: it derives directly from physical measurements rather than calculations, it contains examples of missing data, and it covers the study area in an irregular manner. Furthermore the goal is not retrieval or classification, but detection of patterns in the data. Thus although ensemble methods still form the core of our approach, the specifics will differ somewhat from the previous case.

The ocean buoy data span the period from March 1980 to June 1998, during which time there were four major El Nino events recorded (1982-83, 1986-87, 1991-92, and 1997-98). Buoys take measurements of wind speed and direction, humidity, and temperatures of the air and sea, along with the date and location where the measurement was taken. However, not all buoys are equipped for all types of measurements, and there are gaps in the data, particularly in 1980 and 1983. New buoys were deployed throughout the study period, particularly around 1985-89 and 1991-93.

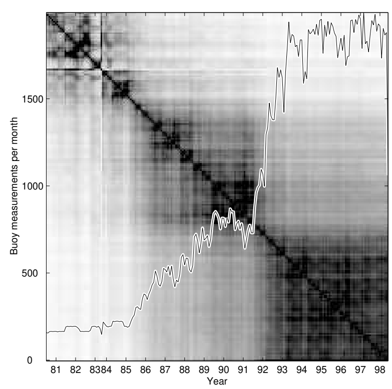
Table 2. Comparison of ensemble technique with other image retrieval methods on classification task. Numbers indicate the percentage of queries that retrieve a similarly classed image as the top rank. The tests employ 1100 and 1600 images, respectively.

Category	Hist.	Corr.	Ensemble
Airshows	57	59	**65**
Bald Eagles	55	**70**	**70**
Brown Bears	35	35	**43**
Mountains	76	**82**	78
Cheetahs, etc.	62	**76**	66
Deserts	47	**57**	52
Elephants	81	76	**85**
Fields	46	43	**54**
Night Scenes	68	70	**71**
Polar Bears	49	**66**	54
Sunsets	68	**75**	64
Tigers	97	**100**	99
Overall	63.4	**68.6**	68.3
Candy	59	**80**	69
Cars	57	63	**90**
Caves	34	**48**	42
Churches	33	37	**39**
Divers	71	**75**	61
Doors	39	52	**64**
Gardens	**72**	62	61
Glaciers	51	**74**	**74**
Hawks	60	**69**	57
MVs	33	42	**57**
Models	41	57	**66**
People	19	20	**25**
Ruins	40	48	**53**
Skiing	52	**65**	56
Stained Glass	74	**84**	76
Sunrises	52	60	**68**
Overall	49.2	58.5	**59.9**

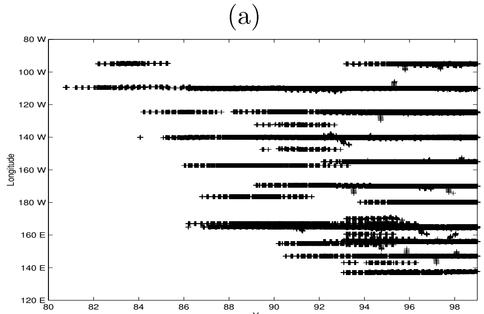
We discretize the buoy measurements as follows: longitude, five bins; zonal and meridional winds, seven bins each; humidity, eleven bins; air and sea temperatures, fifteen bins each. In addition, we include an extra bin for missing values of each feature except longitude. This gives a total of 983,040 dimensions in our final space $\mathcal{F}$. (Note that the latitude measurements hardly vary, so we do not consider this feature in the representation.)

To form ensembles we aggregate all buoy measurements made over a calendar month; in theory this should give a picture of the ocean climate during that month. (If the climate changes over a shorter period of time, we might want to aggregate over a shorter timespan, but a month seems suitable since the duration of an El Nino episode is about one year.) We also need to choose $\mathbf{S}$; we experimented with different settings, starting with $p_j = 0.1$ for each feature. Although the results are qualitatively similar for a wide range of **S** matrices, we get the clearest results with settings that focus on longitude and temperature. This is unsurprising since El Nino events are identified primarily as a rise in sea surface temperatures in the eastern Pacific (McPhaden et al., 1999).

After calculating the pairwise differences between months, the first trend that appears is a striking dependence on measurement date (Figure 1a). This turns out to result from the addition of buoys over time. One might expect the buoys to be added in a random fashion, but in fact the positions of the new buoys tend steadily westward (Figure 1b). This affects the measurements because the Western Pacific is warmer than the Eastern Pacific. Thus the character of the ocean described by the buoy measurements is significantly different in 1980 than in 1990, and the cosine metric reveals this fact.



(a)



(b)

Figure 1. Addition of buoys over time. In (a), the matrix of similarity between months (represented by background shading, with dark indicating greater similarity) shows close correlation with the number of buoy measurements taken each month (foreground curve). Months are most similar to those with a similar number of measurements. In (b), the longitude of measurements is plotted versus time, revealing a westward trend in later dates.

To account for the nonuniform deployment of buoys, we perform comparisons between months using only measurements from buoys that exist in both months. This procedure eliminates the gross time dependence exhibited in Figure 1a, although there are hints of weaker trends remaining over time. We may now look for other patterns in the data.

Since there are only four El Nino events during the span of the measurements, setting up a supervised learning task seems overzealous. However, if El Nino events are present in the data, they should present a distinctive pattern that is self-similar and unlike the data of other years. To test this hypothesis, we can plot the cosine scores of an El Nino period compared to all other observations. The result is somewhat noisy, so we smooth locally to get the curve shown in Figure 2. This plot shows clear dips in 1982-83, 1986-87, 1991-92, and 1997-98, all El Nino years. Given such a curve, it is straightforward to define a classifier simply by selecting an appropriate similarity threshold.

A closer look at the curve reveals a few more interesting observations. The 1991-92 El Nino was followed by two years of lingering El Nino effect, and this also appears in the curve. On the other hand, the 1982-83 El Nino was noted as the strongest of the century, and the size of the dip in the curve does not really reflect this. Missing data in the latter half of 1983 probably causes this effect by narrowing the minimum, which is then filled in by smoothing.

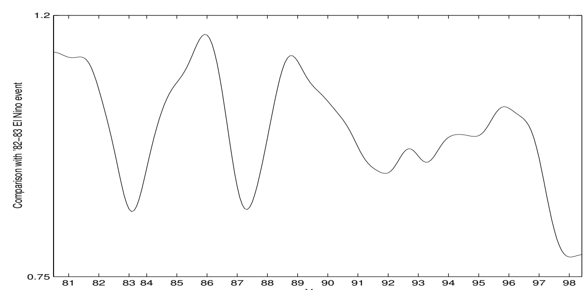


Figure 2. Comparison of month-by-month data with 1982-83 El Nino period (smoothed). Local minima reveal other El Nino events in 1986-87, 1991-92, and 1997-98.

As a comparison, we also plot a curve for the La Nina year 1995-96 in Figure 3. La Nina is the opposite of El Nino, so this curve should show minima in different spots. Indeed, the curve is high during El Nino years, but dips down during the La Nina year 1988-89. It also shows a fairly significant minimum around 1984-85, which was not classified as a La Nina episode. However, since this immediately follows the 1982-83 El Nino, there may well have been weak La Nina conditions that year that were not severe enough to warrant official classification. In any case, the plot shows that a classifier with an appropriate threshold could still detect both La Ninas without false positives.

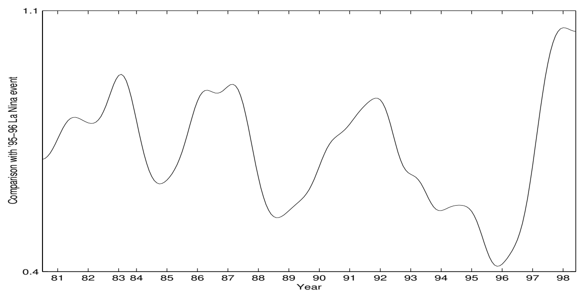


Figure 3. Comparison of month-by-month data with 1995-96 La Nina period (smoothed). Local minima reveal another La Nina in 1988-89 and a spurious signal in 1984-85.

Experience with the ocean data provides numerous insights regarding the ensemble method. The technique can clearly help in discerning patterns in physical raw data, but the application process may not be completely straightforward. Researchers need to remain alert to biases that may affect the records making up the ensembles being compared, as with the placement of new buoys. Also, since the El Nino pattern was originally identified through other means, it is not clear that the ensemble of records machinery provides any new insight in this case. The pattern exists in the data, and with enough perseverance can be detected though techniques using summary features. Nevertheless, as a test of the ensemble technique in a physically motivated domain the positive results are encouraging.

4. Discussion and Conclusion

The experiments described in this paper merely begin to examine what can be done with data organized as ensembles of records. Once ensembles are expressed in $\mathcal{F}$-space, virtually any machine learning technique can be adopted. For example, any two vectors $\mathbf{f}_1$ and $\mathbf{f}_2$ in $\mathcal{F}$ define a linear classifier based upon $\mathrm{sign}\,(\mathbf{f}_{new} \cdot (\mathbf{f}_1 - \mathbf{f}_2))$. Collections of such simple classifiers can be used to build decision trees, do boosting, and create other large-margin classifiers. We expect that other machine learning technology can be similarly applied.

The work described herein owes a debt to the field of text and information retrieval, which makes heavy use of the cosine metric (Salton, 1989). In one sense, our approach is an extension of that work, with an ensemble of records analogous to a bag of words. Thus advances in text retrieval may lead to insights applicable to ensembles of records. There is also some similarity between this work and multiple instance learning (Maron & Ratan, 1998). The latter uses collections that can contain many elements irrelevant to the target concept, whereas we assume that each record in an ensemble contributes to its identity.

In conclusion, analysis of raw data has often been overlooked in academic research but is crucial in real-world applications. This paper has presented a fresh approach to looking at one class of raw data. We describe algorithms for handling entities that can be organized in an *ensemble of records* paradigm. Beyond its success on specific data sets, this approach provides the machine learning community with a novel viewpoint for looking at raw data. Evidence of a need for more such viewpoints exists: although it is not hard to come up with domains structured as ensembles of records, it is difficult to find existing data sets organized in this manner. By and large, existing data sets are already processed into formats that fit existing paradigms. A principal contribution of this work, therefore, is the expansion of our viewpoint to include a new paradigm.

Acknowledgements

The author thanks those who read drafts of the paper and made comments, in particular David Skalak and the anonymous reviewers. This work was supported in part by DARPA grant DAAL01-97-K-0104 and NSF grant DGE-9454149.

References

Bay, S. D. (1999). UCI KDD archive. `http://kdd.ics.uci.edu`.

Blake, C. L., & Merz, C. J. (1999). UCI repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Branting, L. K. (1991). *Integrating rules and precedents for classification and explanation: Automating legal analysis.* Doctoral dissertation, University of Texas, Austin, TX.

Fayyad, U. M., & Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (pp. 1022–1027). San Mateo, CA: Morgan Kaufmann.

Felzenszwalb, P., & Huttenlocher, D. (1998). Image segmentation using local variation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 98–104). Los Alamitos, CA: IEEE Computer Society.

Graham, A. (1981). *Kronecker products and matrix calculus with applications.* Chichester, England: Ellis Horwood Ltd.

Huang, J., Ravi Kumar, S., Mitra, M., Zhu, W. J., & Zabih, R. (1997). Image indexing using color correlograms. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 762–768). Los Alamitos, CA: IEEE Computer Society.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 257–285.

Kolodner, J. (1993). *Case-based reasoning*. San Mateo, CA: Morgan Kaufmann.

Maron, O., & Ratan, A. L. (1998). Multiple-instance learning for natural scene classification. *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 341–349). San Mateo, CA: Morgan Kaufmann.

McPhaden, M. J., Kessler, W. S., & Soreide, N. N. (1999). El Nino story. `http://www.pmel.noaa.gov/toga-tao/el-nino-story.html`.

Salton, G. (1989). *Automatic text processing – the transformation, analysis, and retrieval of information by computer*. Reading, MA: Addison Wesley.

Stanfill, C., & Waltz, D. (1986). Toward memory-based reasoning. *Communications of the ACM*, *29*, 1213–1228.

Swain, M., & Ballard, D. (1991). Color indexing. *International Journal of Computer Vision*, *7*, 11–32.