Maximum A Posteriori Classification of DNA Structure from Sequence Information

David M. Loewenstern
Department of Computer Science, Rutgers University, Piscataway, NJ 08855
and Bell Laboratories, Whippany, NJ 07981
loewenst@paul.rutgers.edu.

Helen M. Berman
Department of Chemistry and Waksman Institute,
Rutgers University, Piscataway, NJ 08855
berman@adenine.rutgers.edu.

Haym Hirsh
Department of Computer Science, Rutgers University, Piscataway, NJ 08855
hirsh@cs.rutgers.edu.

We introduce an algorithm, LLLAMA, which combines simple pattern recognizers into a general method for estimating the entropy of a sequence. Each pattern recognizer exploits a partial match between subsequences to build a model of the sequence. Since the primary features of interest in biological sequence domains are subsequences with small variations in exact composition, LLLAMA is particularly suited to such domains. We describe two methods, LLLAMA-length and LLLAMA-alone, which use this entropy estimate to perform maximum *a posteriori* classification. We apply these methods to several problems in three-dimensional structure classification of short DNA sequences. The results include a surprisingly low 3.6% error rate in predicting helical conformation of oligonucleotides. We compare our results to those obtained using more traditional methods for automated generation of classifiers.

1 Introduction

Although it is often convenient to think of DNA as a sequence of characters drawn from an alphabet $\{A, C, G, T\}$, it is of course a chemically active molecule with a complex three-dimensional structure. It would be of biological interest to be able to predict three-dimensional structural characteristics of a sequence of DNA without deriving it using x-ray crystallography or NMR spectroscopy.

This paper examines several methods for predicting structural characteristics of a test sequence of DNA given only the sequence of nucleotides, and no other information about the sequence. In particular, conformational geometry, crystallographic unit cell, and space group information for the test sequence is not made available, and is in fact predicted by the methods.

There are three structural characteristics, or tasks, of interest. The first task is to predict the helical conformational class, *i.e.*, whether the DNA sequence forms an A-, B-, or Z-helix. The second task is to predict the crystal type, which is to say the crystallographic unit cell and space group. The third task is prediction of packing motif: a group of crystal types belong to the same motif if the molecular interactions within the crystal are similar.[2] We would like to solve these tasks for short DNA sequences (fewer than 13 nucleotides). For our purposes, it is sufficient to label the sequence with exactly one helical conformational class, one crystal type, and one packing motif. Furthermore, we wish to use general machine learning techniques that can be easily applied to a range of DNA classification tasks. To this end, we extracted a training corpus from the Nucleic Acid Database (NDB)[3] composed of all nucleotide sequences with exactly one known helical conformational class, crystal type, and packing motif. The corpus (unlike the NDB) contains only symbols representing the sequence itself, and does not contain three-dimensional coordinate information. To train the classifier for helical class, the corpus is labeled with helical classes from the NDB; similarly, the corpus is labeled with crystal types to train for crystal type classification and packing motifs to train for packing motif classification. Rather than engineering a specific classifier for each task by hand, we explored machine learning methods that extract classification information from the labeled training corpus alone, without using other biological information.

To rephrase the problem as a machine learning task, for each of the three tasks, we construct from the NDB a corpus of sequences drawn from one of two fixed alphabets. Each sequence in the corpus is labeled with a class that has also been extracted from the NDB, and ultimately was associated with each sequence on the basis of prior and independent analysis of X-ray crystallographic results. The corpora are used to train and test, and so compare, several different classification methods.

To address this task, we describe an entropy estimation algorithm, LLLAMA,[a] that is well-suited to entropy estimation of biological sequences, because it exploits inexact repeats of subsequences to make its nucleotide predictions. We then make use of this algorithm in two ways. One way classifies a test sequence by predicting for the test sequence the training class most likely to have generated the test sequence according to the LLLAMA-model of the class. The second way classifies by predicting the training class most likely to have generated the test sequence according to a LLLAMA-model constructed from a reduced training corpus of sequences of the same length as the test sequence. We demonstrate that these entropy-estimation methods perform better in gen-

[a] LLLAMA Looks Like A Meaningful Acronym, with apologies to Ogden Nash.[6]

eral on the given tasks than any other methods tested in this paper. We also demonstrate that all of the methods perform surprisingly well on the three tasks, and so give hope that at least small-scale DNA structure prediction is computationally tractable, and notably, primarily from the sequence alone.

The key result of our work is that several types of methods perform adequately on all three tasks. The best method, LLLAMA-length, has 96.4% accuracy at predicting helical conformational class, 82.1% accuracy at predicting crystal type, and 89.1% accuracy at predicting packing motif given only a nucleotide sequence. These results are made more impressive considering the very small size of the training corpus (138 sequences, 6–12 nucleotides each). It is expected that as more sequences are entered into the NDB, it will become possible to train more accurate classifiers.

Section 2 describes the data in more detail. Section 3.1 provides a brief introduction to our alternative benchmark methods. Section 3.2 then describes the LLLAMA-alone and LLLAMA-length methods, and Section 3.3 the underlying LLLAMA algorithm. Finally, this paper presents experimental results comparing the classification methods in Section 4 and draws conclusions in Section 5.

2 Data

2.1 DNA sequences

The data used for these analyses were those contained in the Nucleic Acid Database (NDB). The NDB contains all three-dimensional structures determined using x-ray crystallography. The data are organized in a relational database which is queried using the program NDBQuery.[9] Constraints were applied so that the resulting reports contained sequences sorted according to conformation type, crystal type, and packing type. The conformation types for DNA helices are the two right-handed forms, A-DNA and B-DNA, and the left-handed form Z-DNA. For this study, structures containing modified residues were included; structures with mismatches were rejected. In the NDB, the structures are classified according to their crystal type and packing motif. Crystal types are defined according to the unit cell dimensions and space groups. Structures are considered to be isomorphous if they have the same crystal type. At the time of these analyses, there were 11 crystal types among the A-DNA structures. B-DNA structures had 16 crystal types, and Z-DNA structures had 5 crystal types. Packing motifs are defined according to the way in which molecules interact in the crystal. For B-DNA there were three motifs, for Z-DNA there were two and for A-DNA there was one.[2]

2.2 Data Representation

For testing purposes, the data for each of the three tasks were considered separately. Each sequence in each data set was labeled by the structure class of the sequence. For each of the three data sets, training and test sets were constructed repeatedly using a leaving-one-out method.[8] The training sets were then used to train each of the various classification methods, which were compared on their average accuracy on the test sets as discussed in Section 4.[b]

The data representation used in this paper includes only the sequence of nucleotides on one strand of the oligonucleotide. A 13-character alphabet was used: $a, c, g, t, u, I, A, C, G, T, U, \wedge, \$$, where I=inosine and A, C, G, T, U are chemically modified a, c, g, t, u respectively, and where $\wedge$ denotes the 3' end of the sequence and \$ the 5' end. Other than the delimiters $\wedge$ and \$, no information outside the sequences themselves was encoded into the data. The size of this data set is 138 sequences, each of length 6 to 12 nucleotides.

3 Methods

3.1 Baseline Classification Methods

To provide a standard of comparison for our method, we rated the classification performance of three other methods on the same data.

The first of these methods was C4.5.[7] For this method, the data were encoded as a twelve-position feature vector with feature 1 corresponding to the 3'-most nucleotide in the sequence, feature 2 its 5' neighbor, and so forth. If the sequence was shorter than twelve nucleotides, the last several features were given the placeholder value "_".

C4.5 works by considering each feature (nucleotide) separately. Each feature is examined to find the feature that best splits the training data into separate classes. C4.5 then repeats this procedure for each subset formed by splitting using the feature, until each subset contains sequences that are nearly all from the same class. A test sequence is classified by using its features to determine which final subset it would have joined if it had been part of the training set, and classifying it with the dominant class of that subset. It should be noted that the representation making each feature correspond to a single nucleotide is sensitive to the alignment of the sequences.

The second method was a k-nearest-neighbor algorithm,[8] using the Levenshtein edit distance function to estimate the distance between sequences.

[b] See Loewenstern *et al.*[4] for a more complete discussion of data representations and data sets. The complete data sets are available at `http://www.cs.rutgers.edu/ loewenst/psb/index.html`.

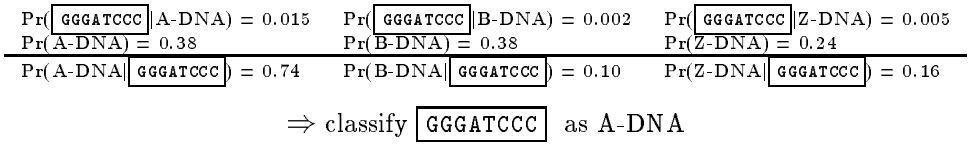

Figure 1: Sample data as handled by LLLAMA-alone.

k-nearest-neighbor classifies a test sequence by giving it the class of its nearest match in the training set.

The third method was a simple baseline method that classifies each test sequence as the most frequently occuring class among all sequences of the same length in the training corpus. If there is no sequence of the same length in the training corpus, the most frequently occuring class of all sequences in the training corpus is used. This method is labeled "MFC" in the tables.

3.2 Classification Method

Our overall goal is to classify unidentified DNA fragments according to class. We have three tasks: to classify by DNA conformation (3 classes: A-, B-, or Z-DNA), by crystal type (20 classes), or by packing motif (11 classes).

Our classification method is Maximum A Posteriori (MAP). We assume that there is an underlying probabilistic model for each class, and each DNA sequence in the class can be understood as being generated stochastically from the model. Therefore, there is a probability $\Pr(s|m)$ that a given sequence s could have been generated by the model m. We will discuss estimating $\Pr(s|m)$ in Section 3.3.

We make use of our models in two different classification methods. In the first, called LLLAMA-alone, we build a model m_C for each class C, and we estimate $\Pr(m_C)$ simply by counting the number of sequences in class C and dividing by the total number of sequences: that is, we assume $\Pr(m_C) = \Pr(C)$ and in general $\Pr(\cdot|m_C) = \Pr(\cdot|C)$. In Figure 1, as an example, we build models m_{A-DNA}, m_{B-DNA}, and m_{Z-DNA} using LLLAMA and the corresponding class of data in the training set. We are interested in classifying GGGATCCC, which is to say that we wish to choose the class C in {A-DNA, B-DNA, Z-DNA} which maximizes $\Pr(C|\texttt{GGGATCCC})$. By Bayes' rule:

$$\Pr(C|\texttt{GGGATCCC}) = \frac{\Pr(\texttt{GGGATCCC}|C)\Pr(C)}{\Pr(\texttt{GGGATCCC})} \quad (1)$$

Since $\Pr(\texttt{GGGATCCC})$ is the same for all classes C, we can set it to normalize $\Pr(C|\texttt{GGGATCCC})$; we use m_C to estimate $\Pr(\texttt{GGGATCCC}|m_C)$, which we assume

is the same as $\Pr(\texttt{GGGATCCC}|C)$. Finally, we classify sequence GGGATCCC as belonging to the class C whose model m_C maximizes $\Pr(\texttt{GGGATCCC}|m_C)\Pr(m_C)$, which in this case is A-DNA.

In the second classification method, called LLLAMA-length, we build different models $m_{C,L}$ for each class C and sequence length L. We can then estimate $\Pr(m_{C,L})$ by counting the number of sequences of length L in class C and dividing by the number of sequences of length L: that is, $\Pr(m_{C,L}) = \Pr(C|L)$.

3.3 Model generation: LLLAMA

Motivation

As described above, to maximize the *a posteriori* probability $\Pr(s|m)\Pr(m)$ for a test sequence of nucleotides $s = (s_1, s_2, ...s_n)$ and model m, we must estimate $\Pr(s|m)$. A reasonable way to do this is to view the sequence as a time series, and estimate each of the nucleotides incrementally, scanning from 3' to 5': $\Pr(s|m) = \Pr(s_1|m)\Pr(s_2|s_1, m)...\Pr(s_n|s_1, s_2, ..., s_{n-1}, m)$. For example, for GGGATCCC, we would estimate $\Pr(\texttt{GGGATCCC}|m)$ as $\Pr(\texttt{G}|m)\Pr(\texttt{G}|\texttt{G}, m)\Pr(\texttt{G}|\texttt{GG}, m)\Pr(\texttt{A}|\texttt{GGG}, m)\Pr(\texttt{T}|\texttt{GGGA}, m)...\Pr(\texttt{C}|\texttt{GGGATCC}, m)$. Each of the probability estimates for each nucleotide depends on a *context* of "previous" nucleotides. Each of the nucleotide probability estimates can be learned from a training set of sequences in the same class.

The problem is that it is entirely possible that a particular context in the test sequence has never been seen in the training set. In that case, we may either relax our matching criterion, thereby permitting near matches to our context when estimating a nucleotide probability, or we may use a shorter context for matching. We then have the problem of choosing which of several possible estimates to make.

We resolve this issue in LLLAMA by measuring the degree to which we relax the matching requirements by counting the number of mismatches, or Hamming distance, and expressing the overall prediction for $\Pr(s|m)$ as a weighted sum over different context sizes and Hamming distances. Our objective, therefore, is to combine matches from many different context sizes and many different match distances, placing greater weight with matches that are more likely to have greater predictive accuracy.

Description

In this section we describe our model in formal terms.[c] One may view a predictor for a given combination of match distance and context size as corresponding

[c] See Loewenstern and Yianilos[5] for a more complete treatment.

to a predictive expert. The prediction of the 2-mismatch, window size 7 expert is formed by examining all past matches to our 7-nucleotide trailing context window with exactly 2 mismatches, and capturing the distribution of the following character by maintaining a simple table of counters. The simplest way to combine these experts is by a fixed set of weights that sum to one.

But suppose that while trying to predict a particular character position with context size $w = 7$, our past experience includes no perfect matches (*i.e.*, no 0-mismatches, or matches of Hamming distance 0), and no 1-mismatches, or matches of Hamming distance 1. In this case, it makes no sense to give any weight to the opinions of the experts for Hamming distances 0 or 1 – in fact their opinion is not even well-defined in this case. So only the 6 experts corresponding to Hamming distance $2-7$ are relevant. In what follows, we will refer to this value as *first Hamming*. Finally, since we don't know *a priori* how window size will influence the prediction, our model is formed at the uppermost level by a mixture of models, each considering a fixed window size from some fixed prior set.

We denote by b the discrete random variable representing the character of the test sequence to be predicted. By w we denote the positive integer random variable corresponding to the length of our trailing context window; it ranges from 1 to the length of the longest sequence in the training set, L. Next, f denotes the first Hamming distance to be considered. It may assume values $0, \ldots, min(w, h_{max})$, where h_{max} is an external parameter not set by the LLLAMA algorithm, which may assume values $0, \ldots, L$. By h we denote the Hamming distance associated with each expert, so h lies in the range $0, \ldots, min(w, h_{max})$. Finally, we use *past* to represent our modeling past, *i.e.* the training set. Therefore, to estimate $\Pr(s_n|s_{n-1}, s_{n-2}, ..., s_1, m)$, we will estimate $\Pr(b|past)$ for $b = s_n$.

Given a fixed window size $w = k$, and distance $h = i$, there is a natural prediction $\Pr(b|h = i, w = k, past)$ formed by locating all distance i matches in any training sequence to the trailing context of length k, and then using the distribution of characters that follow them. This is a single *expert* as described above. This prediction is independent of f so $\Pr(b|h = i, f = j, w = k, past) = \Pr(b|h = i, w = k, past)$ for all legal values of j. Our prediction $\Pr(b|past)$ arises from the joint probability $\Pr(b, h, f, w|past)$ by summing over the hidden variables h, f, w as follows:

$$\Pr(b|past) = \sum_{i,j,k} \Pr(b|h = i, f = j, w = k, past) \cdot \Pr(h = i, f = j, w = k, past) \quad (2)$$

The final term is then expressed as a product of conditionals:

$$\begin{aligned}\Pr(h = i, f = j, w = k, past) \quad = \quad & \Pr(h = i | f = j, w = k, past) \cdot \\ & \Pr(f = j | w = k, past) \cdot \\ & \Pr(w = k | past) \cdot \Pr(past) \end{aligned} \tag{3}$$

In this expression $\Pr(past) = 1$ and $\Pr(f = j | w = k, past) = 1$ for j equal to the distance of the closest match to our trailing context window of length k, in the *past*. At all other values $f = 0$. That is, f is a Boolean selector function $f(j, k, past)$. We assume w is independent of the past in our model, and so $\Pr(w = k)$ consists of a fixed vector of L mixing coefficients that select a window size. Finally, in our model, h is also independent of the past, and so $\Pr(h = i | f = j, w = k)$ consists of a fixed vector of $k - j + 1$ mixing coefficients that select a Hamming distance given the earlier choice of a window size k, and observation that the nearest past match is at distance j. We then have:

$$\begin{aligned}\Pr(b | past) \quad = \textstyle\sum_{i,j,k} \quad & \Pr(b | h = i, w = k, past) \cdot \Pr(h = i | f = j, w = k) \cdot \\ & f(j, k, past) \cdot \Pr(w = k) \end{aligned} \tag{4}$$

The first term in the summation is recognized as a single expert, the second selects an expert based on f and w, the third deterministically selects a single f value, which receives probability 1, and the final term selects a window size.

The learning task before us is to estimate the parameters $\Pr(h = i | f = j, w = k)$ and $\Pr(w = k)$ by examining the training set T. Our algorithm is an application of the Baum-Welch algorithm for Hidden Markov Models.[1] This method iteratively updates the above parameters, with the guaranteed result that the probability of the training set $\prod_{b \in T} \Pr(b | past_b)$ increases or remains the same with each iteration. The number of iterations used by LLLAMA is an external parameter, n_{iter}.

4 Experimental Results

The performance of all of the methods is shown graphically in Figure 2. LLLAMA-length can be trained to perform each of the three tasks well, with accuracy far greater than chance. Specifically, the best accuracy achieved on the helical conformational classification task was 96.4%; on the crystal type task, 82.1%; and on the packing motif task, 89.1%. To provide a point of reference, we compare our model directly against the methods described in Section 3.1. Each method and representation is shown in order of increasing error rate, as calculated by the leaving-one-out method.[8]

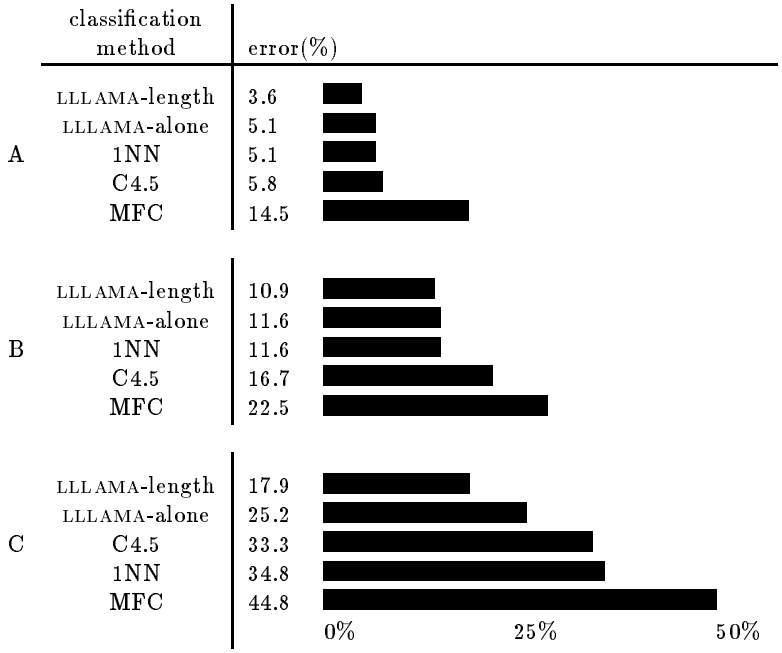

Figure 2: Error rates on the full data set, (A) helical conformation task, (B) packing motif task, (C) crystal type task. LLLAMA-length and LLLAMA-alone are new methods proposed in this paper. C4.5 is a standard machine learning algorithm. 1NN is 1-nearest-neighbor, also a standard machine learning algorithm. MFC is most-frequent-class, the error rate associated with choosing the most frequent class of all sequences in the training set with the same length as the test sequence.

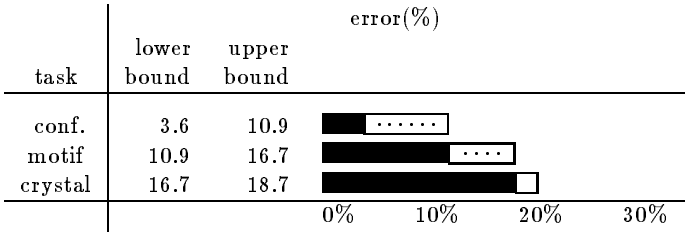

Figure 3: LLLAMA-length classification error (lower and upper bounds) on all testing tasks.

There were several possible ways to address the issue of tuning LLLAMA's external parameters, h_{max} and n_{iter}. The ideal method would have been to divide the data set into separate training, parameter tuning, and test sets, but this was not feasible with such a small data set. Nor did the data set size support a cross-validation suite in which the training partition was subdivided into training and parameter tuning sections. We decided instead to estimate lower and upper bounds on error on unseen data by tuning the parameters on the test task itself and on a related task respectively. That is, to estimate an upper bound for the error on a given "test" task, we chose values for h_{max} and n_{iter} that minimized the error of the same method on either of the other two tasks. To estimate a lower bound, we chose values that minimized error on the test task itself. With sufficient data, we believe this lower bound would converge to the actual error. Figure 3 reports both the lower and upper bounds for LLLAMA-length for all tasks.

For k-nearest-neighbor, we chose the default value $k = 1$ for the number of neighbors to compare before making a classification decision. In fact, classification accuracy for k-nearest-neighbor using edit distance fell monotonically with increasing values of k in the range tested ($\{1, 3, 5, 7\}$) for all tasks. For similar reasons, C4.5 is reported only for unpruned trees using the default parameters.

LLLAMA-length outperforms all other methods for all tasks for at least two of the three possible choices of tuning sets. In fact, it turns out that tuning the parameters on the motif task is a very good way to find good parameters for the conformation task and *vice versa*. The k-nearest-neighbor method performs comparably to LLLAMA-alone in most cases.

5 Discussion and Future Work

There are several conclusions which the work presented in this paper appear to support. The first conclusion, perhaps most interesting from a biological perspective, is that DNA helical conformation, packing motif, and crystal type can be predicted from sequence information alone, at least for short sequences. No additional biological information was encoded into the representation we used. Our results our especially promising using LLLAMA-length, but even standard machine learning methods such as C4.5 and k-nearest-neighbor perform credibly well. The 96.4% classification accuracy on the helical conformation prediction task is especially noteworthy. These results also argue that oligonucleotide length is generally a useful feature for classification but not sufficient for classification by itself, as seen with the MFC method.

The entropy-estimation/MAP methodology used by LLLAMA-length and

LLLAMA-alone provide broad applicability that would be difficult to capture using other methods. If new crystal type or packing motif classes are added to the NDB, their models may be learned and added to LLLAMA-length or LLLAMA-alone without requiring retraining of the models for the other classes.

In addition, the LLLAMA models may be used directly, without retraining, for more than classification. For example, the LLLAMA model trained on the A-DNA corpus may be used to find the degree of local A-helical conformational propensity of each nucleotide of a long sequence such as a gene. LLLAMA has been applied to large biological sequence problems as well, such as comparing coding and non-coding regions in an entire chromosome.[5] Since there is nothing in the LLLAMA algorithm or the LLLAMA-alone classification method that requires it to be applied to nucleotides, they may be expected to be applicable in many situations in which modeling or classification of sequences is desired, such as protein secondary structure prediction.

One surprising ramification of our work concerns the identification of the principal factors determining the helical conformation of oligonucleotides. Common belief and intuition is that the length of an oligonucleotide and the specific environmental conditions of its crystallization play an important role in determining its conformation. However, our results indicate that the nucleotide sequence itself is a very significant factor in determining a compound's conformation. From sequence alone (without explicitly taking into account either oligonucleotide length or environmental conditions) LLLAMA-alone predicts helical conformation with 94.9% accuracy. Even the standard classification techniques C4.5 and k-nearest-neighbor, which were used here as baselines, also predict helical conformation well without explicitly considering length or environmental conditions. In contrast, MFC, which does take length into account but does not consider nucleotide sequence or environmental conditions, performs more poorly than the other methods presented. Similar observations can be made concerning the importance of sequence in determining packing motifs and crystal types. Although they are less well-understood and thus the factors that impact upon them still not fully clear, our results indicate that nucleotide sequence is at least as important as environmental conditions and length for predicting these two finer-grained structural characteristics as well.

From a computer science perspective, it is noteworthy that LLLAMA displayed significant advantages over more standard methods. In all cases, the best classification method was LLLAMA-length. In a domain in which data is relatively plentiful, building a set of LLLAMA-based classifiers with different parameters and choosing the one that performs best on a separate tuning set may be expected to classify new data even better than the other methods presented here.

In future work, it will also be worthwhile to examine alternatives to the length-partitioning used in LLLAMA-length. This method worked well on our data, but it does discard a great deal of training data. The methodology used in LLLAMA-length and MFC was applied to C4.5, for instance, but the results were substantially worse across the board than for C4.5 alone. A better method might be to incorporate length partitioning as another partition within the LLLAMA model, much as first Hamming distance is now handled. This would allow the method to examine mixtures of length-partitioned and length-ignoring models.

Acknowledgments

The NDB is funded by the NSF. We thank Steve Norton for his comments.

References

1. L. E. BAUM AND J. E. EAGON, Bull. AMS, 73 (1967), pp. 360–363.
2. H. M. BERMAN, A. GELBIN, AND J. WESTBROOK, Prog. Biophys. Mol. Biol., 66 (1996), pp. 255–288.
3. H. M. BERMAN, *et al.*, Biophys. J., 63 (1992), pp. 751–759.
4. D. M. LOEWENSTERN, H. M. BERMAN, AND H. HIRSH, Tech. Rep. DCS-TR 331, Department of Computer Science, Rutgers University, 1997.
5. D. M. LOEWENSTERN AND P. N. YIANILOS, in Proceedings of the Data Compression Conference, March 1997.
6. O. NASH, *The Lama*, in Selected Poetry of Ogden Nash, Little, Brown, 1995, p. 310.
7. J. R. QUINLAN, Machine Learning, 1 (1986), pp. 81–106.
8. S. WEISS AND C. KULIKOWSKI, *Computer Systems that Learn*, Morgan Kaufmann, Palo Alto, CA, 1991.
9. J. WESTBROOK, T. DEMENY, AND S.-H. HSIEH, Tech. Rep. NDB99, NDB, Rutgers University, 1996.