On The Role of Pattern Matching In Information Theory

Aaron D. Wyner
Bell Laboratories, Lucent Technologies

Jacob Ziv *
Department of Electrical Engineering
Technion, Haifa, Israel

Abraham J. Wyner †
Department of Statistics
UC Berkeley, California

In Memory of Aaron. D. Wyner (1939-1997)

Abstract

In this paper, the role of pattern matching information theory is motivated and discussed. We describe the relationship between a pattern's recurrence time and its probability under the data generating stochastic source. We motivate how this relationship has led to great advances in universal data-compression. We then describe non-asymptotic uniform bounds on the performance of data compression algorithms in cases where the size of the training data that is available to the encoder is not large enough so as to yield the asymptotic compression: the Shannon entropy.

We then discuss applications of pattern matching and universal compression to universal prediction, classification and to entropy estimation.

---

*This research is partially supported by the Bi-national US-Israel Science fund
†The work of A.J. Wyner was supported by NSF grant DMS-9508933.

I. Introduction

The self-information of a random event or a random message is a term coined by C.E.Shannon who defined it to be "minus the logarithm of the probability of the random event". The Shannon "entropy" of the stochastic source that generated the event is the expectation of the self-information.

Shannon discovered that the entropy of a stochastic source has a clear and important physical meaning: on average, it is the smallest number of bits that it takes to faithfully represent or communicate events generated from the stochastic source.

Suppose, for example, we are interested in finding efficient representations of incoming random messages or random events. In a broad sense, we consider three possible circumstances:

- The source distribution is completely known.
- The source distribution is unknown, but it belongs to a parameterized family of probability distributions.
- The source distribution is known to be stationary and ergodic, but no other information is available.

If the source distribution is completely known then there is a wide variety of efficient and practical solutions. Shannon himself showed how to find a code which assigns to every random message a codeword whose length is nearly the self-information (log-likelihood) of the message [1]. Consider then the situation where the source's underlying probability law is *not* completely known, which is indeed the case when dealing with practical information sources. The obvious way to proceed is by the "plug-in" approach: This involves estimation of the source distribution, which is then used in the coding algorithm in place of the unknown distribution. If, for example, the source distribution is not specified completely but is known to be a member of a parametric family then the unknown parameters are readily estimated

[1] We call this the Shannon code. See [1] for a description.

from the message itself or from training data. The actual representation can be accomplished by finding a Shannon code that uses codewords whose lengths are nearly the self-information of messages with respect to the estimated measure instead of the true measure. With enough data the estimate will be sufficiently close to the truth and the representation will be nearly optimal. On the other hand, as we shall see, conventional methods for estimating the source probability measure are not always optimal and are rarely practical in universal settings where no prior information is available. Consequently, we propose the following general question:

Can we find an appropriate and universal way to estimate the probability measure that governs the generation of messages by the source?

This is a question of wide ranging interest to scientists, mathematicians and engineers (for perhaps different reasons). We will attempt to answer this question from the point of view of information theory.

A natural (frequentist) understanding of the probability of an event begins with a long realization from a stochastic source, which we will assume to be stationary and ergodic. The number of occurrences of a random event when divided by the length of the realization, is nearly the probability of the event. Thus the time between events, called the recurrence time, is on average inversely proportional to its probability of occurrence. For example, suppose we observed a monkey typing at a typewriter. The number of occurrences of the pattern "CLAUDESHANNON" in the monkey manuscript is expected to be the probability of the pattern multiplied by the number of letters in the manuscript. Therefore, the time measured in letters, that it will take the monkey to type the pattern "CLAUDESHANNON" is simply the inverse of the probability of the pattern. Since the probability of the pattern is easily seen to be 2^{-13} the average recurrence time is 2^{13}. This is accurately expressed by Kac's Lemma which states that the expected time until the recurrence of a fixed pattern in a stationary ergodic sequence, is the inverse of the pattern's probability. We can rewrite the quantity 26^{13} as $2^{13\log 26}$ which in turn is equal to $2^{\ell H}$, where ℓ is the length of the pattern and H is defined to be $\log_2 26$ [2]. Thus, for this pattern, the expected logarithm

[2]All logarithms will be base 2.

of the recurrence time divided by the length of the pattern is not more than H. For some source distributions it is possible to find the distribution of the recurrence time for any fixed pattern using probabilistic and analytical techniques [32][33].

In the above discussion, we discussed the recurrence time for any fixed pattern. The information theorist, on the other hand, is interested in random messages and the recurrence time of random patterns.

Let us introduce some notation: The random variables X_i are assumed to take values in a finite alphabet $\mathbf{A}$ with $|\mathbf{A}| = A \leq \infty$. For any positive integer ℓ we write

$$X_1^\ell = X_1, X_2, \ldots, X_\ell.$$

For stationary sources, define the ℓ-th order per letter entropy:

$$H(X_1^\ell) = -\frac{1}{\ell} E \log P(X_1^\ell).$$

The entropy rate is defined to be

$$H = \lim_{\ell \to \infty} H(X_1^\ell)\,. \tag{1}$$

We define N_ℓ to be the time of the first recurrence of X_1^ℓ in the stochastic source. That is N_ℓ is the smallest integer N so that a copy of X_1^ℓ equals $X_{N+1}^{N+\ell}$. The asymptotic equipartition theorem (AEP) implies that for ℓ large enough the random pattern is with high probability *typical* which means that minus the log of the probability of X_1^ℓ divided by ℓ is nearly its expected value H. Thus, for almost every pattern (the typical ones) Kac's theorem implies that the log of the recurrence time divided by ℓ is nearly H. This is stated formally in the following theorem:

A Recurrence Time Theorem [8, 9, 10]: Let N_ℓ be the first recurrence of pattern X_1^ℓ in a stationary, ergodic, finite alphabet source. Then

$$\lim_{\ell \to \infty} \frac{\log N_\ell}{\ell} = H \text{ with probability 1.} \tag{2}$$

In light of this result [3], it should not be at all surprising that matching a pattern onto it's first recurrence, moving backward into the suffix of the string, turns out to be an important device for generating an efficient estimate for the probability of the pattern.

What *is* surprising is that the recurrence time may be the *only* available tool for estimating probabilities while other more "intuitive" estimates are useless. There is of course a significant practical problem: For a given sequence of letters X_1^n and a fixed ℓ, it may be that $N_\ell > n$. To avoid this uncertainty, we turn the problem inside out and consider a different kind of pattern matching.

Define L_n to be the "longest match" of any prefix of the incoming sequence X_1^∞ into the sequence X^0_{-n+1} of the n most recent observations. Mathematically, the longest match behaves like the recurrence time since

$$\{N_\ell > n\} = \{L_n < \ell\}.$$

The following result is the match length equivalent of the recurrence time theorem:

A Match Length Theorem [8]: Let L_n be the longest match of the incoming sequence X_1^∞ into the past n observations, X^0_{-n+1}. Then

$$\lim_{n\to\infty} \frac{L_n}{\log n} = \frac{1}{H} \text{ in probability.} \tag{3}$$

Let us motivate this result by establishing directly the relationship between the longest match and the probability of the pattern. Taking our lead from renewal theory, we introduce the stopping time $T = T_n$ equal to the smallest k such that $-\log P(X_1^k|X^0_{-n+1}) \geq \log n$. For most sources it follows (informally) that $-\log P(X_1^T|X^0_{-n+1}) \approx \log n$. Now consider a pattern of length $\ell = T + \delta$, where δ is any positive integer. ¿From the linearity of expectations it is easy to see that if $S_n(X_1^\ell)$ is the number of occurrences of X_1^ℓ in X^0_{-n+1} then

$$ES_n(X_1^\ell) = nEP(X_1^\ell) \approx EP(X_{T+1}^{T+\delta}|X_1^T). \tag{4}$$

[3]**Historical Note:** Convergence in probability and half of an almost sure extension appeared first in [8]. A complete proof of almost sure convergence first appeared in [9]. A short proof can be found in [10]

For δ large the right hand side of (4) is small which implies that strings longer than T are not expected to appear even once in the past n observations; this in turn implies that $L_n < \ell$. Now let $\ell = T - \delta$. Then

$$ES_n(X_1^\ell) = nEP(X_1^\ell) \approx EP(X_{T-\delta+1}^T | X_1^{T-\delta})^{-1}. \tag{5}$$

If δ is large then the right hand side of (5) is *large* which implies that we expect many occurrences of X_1^ℓ in the past n observations; this in turn implies that $L_n > \ell$.

Taken together, we have shown that the longest match is likely to be sandwiched between $T - \delta$ and $T + \delta$. To prove this precisely we would need to show (1) that if the expected number of pattern occurrences is large than the probability of at least one occurrence is close to one; and (2) that the maximum conditional probability of any pattern goes to zero sufficiently fast. For sources with vanishing memory these conditions are satisfied, and the random variable $\Delta = |L_n - T|$ is not too large [14].

In summary, we have established the connection between the match length and the probability of a pattern: the longest match is approximately the first prefix of X_1^∞ whose probability is less than $\frac{1}{n}$.

We shall see, in section II, that that this approach is efficient and enormously practical. Consequently, pattern matching has blossomed into an important tool for information theorists, especially when no knowledge of the underlying probability measure is available.

In this paper we describe the role of pattern matching in information theory. In section II, we develop a general way to use recurrence times and patterns to estimate a probability measure, specifically in order to construct, improve and analyze universal coding algorithms. We show how pattern-matching based data compression algorithms can achieve optimal compression rates. We then show how some string-matching algorithms for universal data-compression are not only asymptotically optimal when the length of the training set tends to infinity, but are also optimal for intermediate amounts of training data. In sections III and IV we consider applications of pattern matching and recurrence times to problems of classification, prediction, and entropy estimation.

II. Universal Data Compression

As mentioned earlier, C.E. Shannon was the first one to point out that for a given source, the entropy is the lowest average number of bits per input letter that a noiseless (i.e. error-free) encoder can achieve. Indeed, for a given source code, let $L(X_1^\ell)$ denote the length function of X_1^ℓ defined to be the number of bits that represent X_1^ℓ. It is well known (see, for example [1]) that

$$E\,\frac{L(X_1^\ell)}{\ell} \geq H(X_1^\ell)\,. \tag{6}$$

Let $L(X_1^\ell)$ be the length function associated with the application of the Shannon coding algorithm, which can easily be applied when the source probabilities are known. This length function satisfies:

$$-\log\, P(X_1^\ell) \leq L(X_1^\ell) \leq -\log\, P(X_1^\ell) + 1 \quad \text{hence} \quad H(X_1^\ell) \leq \frac{1}{\ell}\, EL(X_1^\ell) \leq H(X_1^\ell) + \frac{1}{\ell}\,.$$

Thus, with ℓ going to infinity, it follows that H is the lowest ACHIEVABLE average number of bits per source letter for any given stationary source.

Sometimes, no a-priori information about the underlying statistics of the source is available. To formulate a representation of our random event in such a circumstance we can utilize various different universal data compression algorithms that take the following forms:

(A) Universal data-compression algorithms that operate on the input sequence that is to be compressed. These algorithms may be adaptable to empirical statistics generated by the sequence itself.

(B) Universal data-compression algorithms, that utilize a finite "training-sequence" which was either emitted by the same source, or some other finite binary vector that conveys some description of the statistics of the source.

(C) Universal data compression algorithms that utilize a training set, but are also adaptable to statistics generated from the sequence itself.

Let us begin our exploration of universal data compression algorithms of type (A). Our goal is to compress a given sequence of n letters using those letters and no other information. Let

$$X_1^n = X_1, X_2, \ldots X_n \tag{7}$$

and let

$$\tilde{Q}(x_1^\ell) \triangleq \frac{1}{n-\ell+1} \sum_{i=1}^{n-\ell+1} 1\{X_i^{i+\ell-1} = x_1^\ell\} \tag{8}$$

The quantity defined in equation (5) is called the ℓ^{th}-order empirical probability measure. We also define $\tilde{H}(X_1^\ell)$ to be the entropy rate of the empirical probability $\tilde{Q}$.

Now, by the concavity of the entropy function and Jensen's inequality it follows that:

$$E\tilde{H}(X_1^\ell) = \frac{1}{\ell} E\{-\sum_{\alpha^\ell} \tilde{Q}(X_1^\ell) \log \tilde{Q}(X_1^\ell)\} \leq H(X_1^\ell)\,. \tag{9}$$

By [2, 3], for any ℓ (smaller than n) one can, for example, empirically evaluate $\tilde{Q}(X_1^\ell)$ for each ℓ vector and then apply the appropriate Shannon coding algorithm on consecutive ℓ-blocks. (The description of $\{\tilde{Q}(X_1^\ell)\}$ takes about $A^\ell \log n$ bits). The length function may therefore be upper-bounded by:

$$L(X_1^n) = A^\ell \log n + \sum_{i=0}^{\frac{n}{\ell}-1} L\left(X_{i\ell+1}^{i+\ell}\right) \tag{10}$$

where it is assumed that ℓ divides n and that the length function $L\left(X_{i\ell+1}^{i+\ell}\right)$ is that produced by the Shannon Coding algorithm. It therefore follows that if we let $\ell \leq \log n$, then

$$-\log \tilde{Q}(X_1^\ell) \leq L(X_1^\ell) \leq -\log \tilde{Q}(X_1^\ell) + 1 + o(n).$$

If we further assume that $\ell << \log n$ then the per letter cost of describing $\tilde{Q}$ tends to zero. Thus, taking expectation in 10, it follows that

$$\frac{1}{n}\, EL(X_1^n) \leq H(X_1^\ell) + \frac{1}{\ell} + \frac{o(n)}{n}\,. \tag{11}$$

Thus the expected compression of the sequence X_1^n is very nearly $H(X_1^\ell)$.

This will be good if $H(X_1^\ell)$ is close to H. If ℓ is such that $H(X_1^\ell)$ is not close to H we may try to close the gap by increasing ℓ. But increasing ℓ will sharply increase in the length of the description of the empirical distribution! Good sense dictates that we try to find the ℓ that achieves the shortest overall representation. It was a similar approach that led J. Rissanen to suggest the MDL (i.e. Minimum Description Length, see [12]) as an alternative to Shannon's self-information for an individual sequence.

At first glance this approach appears to solve the compression problem completely; but there are drawbacks. For any ℓ, the compression $H(X_1^\ell)$ is achieved by introducing a *coding delay* of n letters. This means that no decoding is possible until the entire n-block has been encoded. Furthermore, if there also exists a training set, then this approach will not necessarily yield the best compression.

Finally, if the source belongs to a parametric family, more efficient, coding schemes (as well as achievable lower bounds) do exist. (see [3]).

We now introduce an alternative approach to data compression which is optimal in a very general sense, since:

- no knowledge of the source is required.
- the coding delay is not long.
- it is simple and easy to implement.

The approach uses pattern matching.

Universal data compression with a training sequence

Rather than to generate the empirical statistics from the incoming data to be compressed, one can use past data which was emitted by the source or some other description of the source in order to generate an appropriate encoder for the incoming data (this is case (B) above). With that an mind we propose that the encoder be fed with two inputs:

(a) The incoming data X_1^ℓ.

(b) A *training sequence* that consists of N_0 letters, emitted by the very same source. For example, the training sequence may consist of the most recent N_0 letters, $X_{-N_0+1}^0$, prior to the incoming sequence X_1^ℓ. If the training set is shifted with incoming data then the training set is called a *sliding window*. If the training sequence is not shifted then the training set is called a *fixed database*.

Our first attempt at data compression in this setting is the most intuitive approach: the plug-in method. Given the training sequence of length N_0, we choose an integer ℓ and then compute the relative frequency $\tilde{Q}(X_1^\ell)$ of all ℓ-vectors. Assuming that the empirical distribution is the true probability law we then encode incoming ℓ-blocks using a Shannon code, or better still, the appropriate Huffman code. The expected compression ratio in this case will be nearly $-E\log\tilde{Q}(X_1^\ell)$. The primary concern is whether it is possible to make the expected compression ratio close to $H(X_1^\ell)$. This may not be the case since $-\frac{1}{\ell}\,E\log\tilde{Q}(X_1^\ell)$ is *not* lower-bounded by $H(X_1^\ell)$. In fact, if X_1^ℓ is generated independently of the training sequence $X_{-N_0+1}^0$ then it follows that:

$$-\frac{1}{\ell}\,E\log\tilde{Q}(X_1^\ell) \geq H(X_1^\ell) \tag{12}$$

by the concavity of the logarithmic function.

Thus, in general, the highly intuitive plug-in approach, based on frequency counting, does not always work if the length N_0 of the training sequence is not large, even though the distance between $E\log\tilde{Q}(X_1^\ell)/\ell$ and $H(X_1^\ell)$ goes to zero with N_0 very quickly for most sources. We therefore have to seek other encoding methods for intermediate values of N_0. To this end, we resume our investigation of the connection between a sequence's probability and its recurrence time. Our first task is to define the first recurrence of a pattern looking backwards into the suffix of a training sequence:

Definition: Let $N_\ell(X_{-N_0+1}^\ell)$ be the *smallest* integer $N \geq 1$ such that: $X_1^\ell = X_{-N+1}^{-N+\ell}$ provided that $N \leq N_0$. If no such N can be found, we let $N_\ell\left(X_{-N_0+1}^\ell\right) \triangleq N_0$.

We will often wish to evaluate the expected recurrence time conditional on the opening sequence. To this end, let $E_{X_1^\ell}(\cdot)$ denote the conditional expectation $E(\cdot|X_1^\ell)$. The following simple lemma forms the basis of an enormously powerful tool:

Kac's Lemma [4, 6][19]: For all stationary ergodic sources the expected recurrence time into a training sequence of length N_0 can be bounded by:

$$E_{X_1^\ell}\left\{N_\ell\left(X^{\ell}_{-N_0+1}\right)\right\} \leq \frac{1}{P(X_1^\ell)} \tag{13}$$

Equality is achieved for $N_0 \to \infty$. It then follows by convexity that:

$$\frac{1}{\ell} E \log N_\ell(X^{\ell}_{-N_0+1}) \leq H(X_1^\ell)\,. \tag{14}$$

Here is a possible coding algorithm (following [5]), which is a simplified variant of the LZ algorithm [11]: encode each block of length ℓ into a binary sequence. The first bit of this sequence will be a "yes-no" flag to indicate if $N_\ell(X^{\ell}_{-N_0+1}) < N_0$. If "yes", then a copy of the sequence X_1^ℓ occurs in $X^{0}_{-N_0+1}$. In that case, we append the binary encoding of the pointer $N_\ell(X^{\ell}_{-N_0+1})$ to the location of its most recent occurrence. If there is no such occurrence (the flag is "no"), then we append the binary encoding of the ordinal number of the vector X_1^ℓ in A^ℓ (which requires $\ell \log A$ bits). We define the length function $L(X_1^\ell|X^{0}_{-N_0+1})$ to be the total number of bits in the binary sequence. It is roughly equal to:

1) $L(X_1^\ell|X^{0}_{-N_0+1}) \approx \log N_\ell(X^{\ell}_{-N_0+1}) + O(\log\log N_0)$ if $N_\ell(X^{\ell}_{-N_0+1}) < N_0$

2) $L(X_1^\ell|X^{0}_{-N_0+1}) \approx \ell \log A$ (no compression) otherwise .

Recurrence Time Coding Theorem: Let δ be some arbitrary small positive number. For any $R > 0$ and any stationary ergodic source (assume that $A = 2$), we define the set $T_R = \{x_1^\ell :\ P(x_1^\ell) < 2^{-R\ell}\}$. Let $B_\ell = \min[R : \Pr\{T_R\} \leq \delta\,]$. For N_0 sufficiently large and any ℓ such that $B_\ell \le \frac{\log N_0}{\ell} - \delta$:

$$\frac{1}{\ell} EL\left(X_1^\ell | X_{-N_0+1}^0\right) \le H(X_1^\ell) + O\left(\frac{\log \ell}{\ell}\right) + \delta \tag{15}$$

Proof: Consider any N_0 and ℓ for which $B_\ell \le \frac{\log N_0}{\ell} - \delta$. If $X_1^\ell \notin T_{B_\ell}$ then the encoding takes at most $\log N_\ell + O(\log\log N_0)$ bits if $N_\ell \le N_0$. Otherwise, the encoding takes at most ℓ bits. Thus

$$EL(X_1^\ell) \le E \log N_\ell + O(\log\log N_0) + \ell \Pr\{X_1^\ell \notin T_{B_\ell}, N_\ell > N_0\} + \ell \Pr\{X_1^\ell \in T_{B_\ell}\}$$

For any sequence $X_1^\ell \notin T_{B_\ell}$ the Markov inequality implies that

$$\Pr\{N_\ell > N_0 | X_1^\ell\} \le \frac{E_{X_1^\ell} N_\ell}{N_0}.$$

Applying Kac's Lemma to $X_1^\ell \notin T_{B_\ell}$ with the smallest probability implies

$$\Pr\{X_1^\ell \notin T_{B_\ell}, N_\ell > N_0\} \le \max_{X_1^\ell \notin T_{B_\ell}} \frac{1}{\Pr\{X_1^\ell\} N_0} \le \frac{2^{B_\ell \ell}}{N_0} \le 2^{-\delta \ell}.$$

Thus,

$$\frac{1}{\ell} EL\left(X_1^\ell | X_{-N_0+1}^0\right) \le H(X_1^\ell) + O(\frac{\log\log N_0}{\ell}) + \delta + 2^{-\delta\ell}. \tag{16}$$

Now (16) holds for all ℓ and N_0 with $B_\ell \le \frac{\log N_0}{\ell} - \delta$. If N_0 is sufficiently large it follows from the AEP, for any δ, that $B_\ell \le \frac{\log N_0}{\ell} - \delta$ for $\ell = \frac{\log N_0}{H+2\delta}$; (15) follows.

Discussion: We measure performance in terms of the compression ratio. The recurrence time coding encodes each ℓ-block using $L(X_1^\ell)$ bits. Thus the per block compression ratio is $\frac{L(X_1^\ell)}{\ell}$. We would like to measure the average per-block compression ratio. Since the algorithm encodes fixed length blocks into variable length strings the average compression ratio must be $\frac{EL(X_1^\ell)}{\ell}$. We point out that if the encoding mapped variable length blocks into variable length binary strings, then the block length ℓ would be random. In this case, the expected compression can be defined as *either* $\frac{EL(X_1^\ell)}{E\ell}$ or $E\frac{L(X_1^\ell)}{\ell}$. The distinction is real since the definitions result in possibly different compression ratios, corresponding to sound operational motivations, albeit distinct (see [14] and [29] for a discussion).

The practical result of the coding theorem is that the recurrence time provides a basic tool for construction of a workable universal algorithm: in the sense that as N_0 tends to infinity the compression ratio will tend to $H(X_1^\ell)$. For most sources, the plug-in method may satisfy the same result.

There is a complication with this algorithm: For any given N_0 the algorithm is effective only for those ℓ with $B_\ell \leq \frac{\log N_0}{\ell} - \delta$. It is therefore essential to know the values of B_ℓ in order to design the appropriate algorithm. The plug-in approach has a similar problem: if the block length is too short you waste data; but if it is too long, the method fails outright.

This problem is solved by replacing this universal Fixed-to-Variable (F-V) scheme by a sliding-window version of the universal LZ-77 algorithm ([6]) which is Variable-to-Variable (V-V); it encodes blocks of variable length into variable length codes. This algorithm does not require the user to choose a block length. The fixed length $\ell-$blocks are replaced by variable length blocks defined using the "longest match" idea. More formally, we define

$$L_{N_0} = \max\{k : X_1^k = X_{-i}^{k-i-1} \text{ for some } 0 \leq i \leq N_0\}.$$

The block length L_{N_0} is the longest prefix of the incoming data that matches a contiguous substring in the training set. In this context the sequence $X_1^{L_{N_0}}$ is called a *phrase*, and L_{N_0} is the *phrase length*. As before, the encoding of each phrase is the binary representation of the location of the match, plus additional bits to encode L_{N_0} as a binary string. That is if $L(X_1^\ell) = L(X_1^\ell | X^0_{-N_0+1})$ is the length of the binary encoding of X_1^ℓ with $\ell = L_{N_0}$, then

$$L(X_1^\ell) \approx \log N_0 + O(\log\log N_0).$$

In another version of the LZ-algorithm, LZ-78, the training sequence itself is parsed into *unique* phrases. This eliminates the need to encode the phrase lengths, although the incoming data is parsed into phrases that are shorter than in LZ-77. The coding in either version is optimal as N_0 tends to infinity, with an encoding delay that is also variable (i.e. the encoding cannot proceed until at least $L_{N_0} + 1$ letters are observed), but is on average $O(\log N_0)$.

Perhaps the most significant advantage of the LZ algorithm over the recurrence time algorithm that is described above is that there are no choices for the encoder since the encoding delay is entirely data driven. The fixed length blocks are replaced by variable length blocks which are "just right" automatically: Successive variable length phrases are all approximately equiprobable with common probability $\frac{1}{N_0}$. Furthermore, the approach is also very practical since no explicit estimate of the probability needs to be computed. There is a small price: The phrase length need to be encoded, although a clever encoding (see [16]) can make even these extra bits negligible.

A Match Length Coding Theorem: Let L_{N_0} be the longest match of the incoming sequence X_1^∞ into the past N_0 observations $X_{-N_0+1}^0$. For N_0 sufficiently large:

$$\text{the expected compression ratio} = \frac{EL(X_1^\ell)}{EL_{N_0}} = H + \frac{H \log\log N_0}{\log N_0} + o\left(\frac{\log\log N_0}{\log N_0}\right). \tag{17}$$

Proof: For N_0 sufficiently large it follows from (3) that each phrase is approximately $\frac{\log N_0}{H}$ letters long. The encoding of each phrase requires $\log N_0$ bits to encode the location of the match in the training sequence and an additional $\log L_{N_0} + o(\log\log L_{N_0})$ bits to encode the phrase length. If we form the compression ratio as indicated, we have the result.

We remark that we have not formally proven the convergence of the LZ-77 algorithm (any variant). This would require a convergence theorem that holds jointly for all phrases. This is harder to prove (although intuitively true) since consecutive phrases are not independent even if the source itself is memoryless. See [6][14][24][30][34] for complete proofs and useful results.

In summary, we have seen how the recurrence time is closely related to Shannon's self-information and Shannon's entropy. We now know how to construct practical universal coding algorithms without a-priori information about the source probability law. We also have a new interpretation of the Shannon self-information: "the logarithm of the recurrence time".

Indeed, the Lempel-Ziv algorithm is optimal in the limit as the length of the training sequence tends to infinity. What is not at all clear, however, is if, in cases where the memory which is constrained to some "reasonable" finite value N_0 and a delay that is $O(\log N_0)$, one can't achieve better compression.

In general, a sliding-window data compression algorithm with a training-sequence of N_0 letters, encodes substrings (phrases) of $...X_{-2}, X_{-1}, X_0, X_1...X_i....$ into binary strings. Let $\{s = i\}$ denote the event that a phrase has ended at X_{i-1} and thus X_i is the first letter of the next phrase. Conditional on $\{s = i\}$ let the training-data be a sequence $Y_{-N_0+i}^{i-1}$ (of length N_0 letters). In most applications the sequence $Y_{-N_0+i}^{i-1}$ is $X_{-N_0+i}^{i-1}$ ("sliding window" case). On the other hand, by introducing $Y_{-N_0+i}^{i-1}$ we may consider other cases. The training set for fixed database algorithms is always a fixed vector $Y^0_{-N_0+1}$, that may or may not be the first N_0 observations of X. We will however insist that the distribution of $Y_{-N_0+i}^{i-1}$ be the same as $X_{-N_0+i}^{i-1}$.

A *code* consists of a collection of words: $C_i = C(Y_{-N_0+i}^{i-1}, s = i) = \{X_1^j; 1 \leq j \leq \tau\}$ that satisfy the property that no word is a prefix of any other word, and any sequence $X_1^k; ; k \geq \tau$ has a word in C_i as it prefix. Here τ is the maximum allowable delay (i.e. the maximal length of a code word in any of the code-books C_s). Since we have assumed stationarity we may restrict our attention to the case $\{s = 1\}$. Each code-word X_1^j in C_1 is mapped into a distinct binary vector of length $L(X_1^j|Y^0_{-N_0+1}; s = 1)$ ("length function"). where $\sum_{X_1^j \in C_1} 2^{-L(X_1^j|Y^0_{-N_0+1}; s=1)} \leq 1$.

We now introduce the random variable K defined to be the largest integer k such that X^1_{-k} is a substring of $Y^0_{-N_0+1}$. If no such k is found, K is defined to be zero. Thus K is the length of the longest match moving backwards into the past N_0 observations. We point out that the random variable K has the same distribution as the LZ-77 phrase lengths. Consider the "constrained conditional entropy" defined to be

$$H(X_1|X^0_{-K}) = -E[\log P(X_1|X^0_{-K})].$$

It follows from (3) that $K = O(\log N_0)$ which implies that $H(X_1|X^0_{-K})$ converges to H as N_0 tends to infinity.

Our main results, presented below, connect the optimal performance of universal compression algorithms (as measured by either definition of the expected compression ratio) to the constrained conditional entropy.

Claims:

(a) Let $C_v(N_0)$ be the expected compression ratio for any universal coding algorithm, with a training-sequence of length N_0 and a variable length (V-V) delay $\tau = O(\log N_0)$. There exists a *fixed* block-length universal algorithm (F-V) with a training-sequence of length about N_0, a block-length ℓ, and an expected compression ratio C_f that satisfies:

$$|C_f(N_0) - C_v(N_0)| < O(\frac{\log\log N_0}{\ell}).$$

(b) At least for some ergodic sources, the expected compression ratio $C(N_0)$ that may be achieved by *any* sliding-window universal coding algorithm with a window of length N_0 and a delay of no more than $\tau = O(\log N_0)$, satisfies the following lower bound:

$$C(N_0) > H(X_1|X^0_{-K-\ell+1}) - O(\frac{\log\log N_0}{\ell}) > H.$$

for $O(\log\log N_0) \leq \ell < O(\log N_0)$

(c) Consider the LZ family of universal data-compression algorithms. These are all compression algorithms that are "dictionary-type" algorithms in the sense that they encode incoming strings by referring to entries in a "dictionary" of phrases from a training sequence of length N_0.

If the training-data $Y^0_{-N_0+1}$ is independent of the incoming data and the source is stationary and ergodic, then the expected per letter compression ratio $C(N_0)$ satisfies the following lower bound:

$$C(N_0) > H(X_1|X^0_{-K-\ell+1}) - O(\frac{\log\log N_0}{\ell}),$$

for any $\ell \leq \log N_0$. If we assume further that the source satisfies an appropriate "vanishing memory" condition (for example, see [17]) then the above lower bound holds for the LZ sliding window algorithm [6] for all such "vanishing memory" sources.

(d) The HZ sliding-window context algorithm [7] is essentially "optimal" in the sense of Claim (b) above and achieves an expected compression ratio $C_{HZ}(N_0)$ upper-bounded by

$$C_{HZ}(N_0) < H(X_1|X^0_{-K+\ell-1}) + O(\frac{\log\log N_0}{\ell}).$$

for $O(\log\log N_0) \leq \ell < O(\log N_0)$.

This holds for *any* ergodic source.

We leave the proof of Claims (a) and (c) to the Appendix. The proofs of Claim (b) and Claim (d) follow from claim (a) and [7].

Discussion: Claims (a)-(d) are best understood against the background of what is known already about the LZ algorithms. As indicated earlier, there are two standard implementations of the algorithm: the LZ-77 and the LZ-78. Brushing aside minor differences in implementation, it is known that the LZ-77 algorithm (with a training sequence of length n) achieves a compression ratio equal to $H + \frac{H \log\log n}{\log n}$ when applied to sources with vanishing memory (see [14]). The LZ-78 is slightly better (at least asymptotically) since it is known (see [18]) that it achieves a compression ratio equal to $H + O(\frac{1}{\log n})$ when applied to memoryless sources. It is also known to be no worse than $H + O(\frac{1}{\log n})$ (see [20]) for Markov sources. In [16] it was demonstrated that the LZ-77 algorithm can achieve the efficiency of the LZ-78 algorithm but only if modified. As informative as these results may be, they are nevertheless all asymptotic in character. They indicate that *eventually* the compression ratio will be within a specified distance from the entropy. In contrast, claims (a)-(d) establish a non-asymptotic standard of optimal efficiency. Let us examine each claim in turn:

In (a) we learn that all universal coding algorithms that parse incoming data into variable length phrases can be adapted to parse using fixed phrases of length ℓ. We prove the claim by construction leaving the proof for the appendix. Of course, it should be pointed out that this conversion involves a penalty, but it is only $O(\frac{\log\log N_0}{\ell})$. Claim (a) serves mainly as a tool for proving claims (b)-(d). It is interesting in its own right, however. We point out that it follows from claim (a) that both definitions of the expected compression ratio (as defined earlier) yield the same value.

Claim (b) establishes a lower bound on the achievable compression for at least some stationary ergodic sources. Furthermore, in contrast to the lower bounds of [14], [18] and [20] this lower bound is non-asymptotic in character. Since for any size training set the compression will be near the constrained conditional entropy to within terms that are $O(\frac{\log\log N_0}{\log N_0})$. We know that the constrained conditional entropy converges to the true entropy eventually, but possibly very slowly. Thus we get a performance bound even for training sequences of moderate size. Since the lower bound is above the entropy, the difference between the constrained conditional entropy and the actual entropy is a measure of the difference between what is realizable with a finite training set and that which is theoretically achievable with an infinite training set (which is equivalent to a perfect knowledge of the source statistics).

Claim (c) establishes the lower bound for a specific class of widely used algorithms. It should be pointed out that if more is known about the source, for example, if the source is known to be a Markov source, one can get better lower bounds than that of (c) ([14] [18] [29]).

Finally, claim (d) establishes that the HZ context algorithm is optimal in the sense of claim (b).

A last point: the constrained conditional entropy is a natural alternative to the classical Shannon conditional entropy, specifically when universal coding is on the agenda.

III. Universal prediction and classification with memory constraints

Consider the following situation: A device called a "classifier" observes a probability law P_ℓ on ℓ-vectors $z \in \mathbf{A}^\ell$. Its task is to observe data X_1^n, from a second probability law Q_ℓ and decide whether $P_\ell = Q_\ell$ or else P_ℓ and Q_ℓ are sufficiently different according to some appropriate criterion. Specifically, the classifier must produce a function $f_c(X_1^n, P_\ell)$ which with high probability equals 0 when $P_\ell = Q_\ell$ and 1 when $D_\ell(P_\ell||Q_\ell) \geq \Delta$, where

$$D_\ell(P_\ell||Q_\ell) = \sum_{z \in \mathbf{A}^\ell} P_\ell(z) \log \frac{P_\ell(z)}{Q_\ell(z)}$$

and Δ is a fixed parameter. The divergence $D_\ell(P_\ell||Q_\ell)$ is a positive measure of "differentness" which equals 0 only if $P_\ell = Q_\ell$. We will require nothing of the classifier if the divergence is greater than 0 but less than Δ (i.e. close enough).

Suppose that the classifier f_c has sufficient memory resources to store the statistics of the entire probability law P_ℓ. We now introduce the pattern matching technique to provide us with a suitable estimate of Q_ℓ which we then "plug-in" to the divergence formula. To this end, for any pattern $z \in \mathbf{A}^\ell$ let $\hat{N}(z, X_1^n)$ be the smallest integer such $N \in [1, N - \ell + 1]$ such that a copy of z is equal to $X_N^{N+\ell+1}$. If z never occurs in X_1^n then let $\hat{N}(z, X_1^n) = n + 1$. For n sufficiently large, $\hat{N}(z, X_1^n)$ is the *waiting time* until pattern z occurs in string X_1^n. For most z (those without repetitive substructure) the waiting time is nearly the recurrence time which implies that the probability of z can be estimated using

$$\hat{Q}_\ell(z) = \frac{1}{\hat{N}_\ell(z, X_1^n)}.$$

In [17] it is shown that for a finite memory source the classification task can be completed successfully provided n is at least $2^{\ell H + o(\ell)}$, where H is the entropy of Q_ℓ. More formally, for sufficiently large ℓ and $n = 2^{(H+\epsilon)\ell}$ it can be shown that $\hat{N}(z, X_1^n) \leq n$ with high probability and that

$$\Pr\left\{\frac{1}{\ell}|\log \hat{N}(z, X_1^n) - \log \frac{1}{Q_\ell(z)}| < \epsilon\right\} \approx 1. \tag{18}$$

It therefore follows (informally) from (18) that

$$\hat{D}_\ell(P_\ell||Q_\ell) = \sum_{z\in\mathbf{A}^\ell} P_\ell(z)\log\frac{P_\ell(z)}{\hat{Q}_\ell(z)} \approx D(P_\ell||Q_\ell).$$

The classifier then sets $f_c(X_1^n, P_\ell) = 1$ or to 0 accordingly as $\hat{D}$ exceeds a threshold (which depends on Δ). It turns out that this technique works, but only with a slight modification. Complete details are given in [17].

The case where two unknown Markov processes, each represented solely by a sequence which is a realization of the source, is discussed in [37]. There, an efficient, asymptotically optimal estimate of the divergence between the two sources is introduced. This estimator is based on pattern-matching parsing of one sequence relative to the other.

Now consider a different situation. Suppose the training data is a prefix of the incoming ℓ letters, and Q_ℓ is some empirical measure obtained from observations X^0_{-n+1} generated from probability law P. This is the natural setup for predicting X_1^ℓ given X^0_{-n+1}.

It reasonable to suppose that our best efforts at predicting an incoming ℓ-vector X_1^ℓ is limited by our ability to empirically estimate $P(X_1^\ell|X^0_{-n+1})$. Assume, for example, that the closeness between the empirical measure and the true measure is expressed by the requirement that the divergence between the true probability $P(X_1^\ell|X^0_{-n+1})$ and its empirical estimate $Q(X_1^\ell|X^0_{-n+1})$ be small. Specifically we say that Q and P are within ε if

$$D_{X^0_{-n+1}}(P \parallel Q) = \frac{1}{\ell}\, E\log\frac{P(X_1^\ell|X^0_{-n+1})}{Q(X_1^\ell|X^0_{-n+1})} \le \varepsilon.$$

Intuitively, one may accept the idea that efficient universal compression algorithms efficiently squeeze out of the past history all the essential available statistics about the true probability law that governs the source. Hence, they should lead to empirical estimate Q which is "close" to P.

The next result shows that no empirical estimate Q can be too good for all stationary ergodic sources, unless the training data is long enough so as to yield efficient universal data compression (i.e achieving a compression ratio close to the entropy of the source).

Converse Claim: At least for some stationary ergodic sources

$$D_{X^0_{-n+1}}(P \parallel Q) \geq H(X_1|X^0_{-K(X^1_{-n})})$$

$$-H(X_1|X^0_{-n+1}) - 0\left(\frac{\log\log n}{\log n}\right)$$

This follows from the fact that $-\log Q(X_1^\ell|X^0_{-n+1})$ is a proper length-function. We can then use this length function as the basis for a Shannon code that will achieve an expected compression equal to $H(X_1|X^0_{-n+1})+D_{X^0_{-n+1}}(P \parallel Q)$. From claim (b) of the preceding section (replacing N_0 by n) this expected compression must be lower bounded by $H(X_1|X^0_{-K(X^1_{-n})})-0\left(\frac{\log\log n}{\log n}\right)$. This proves the claim.

Indeed, for large enough n, $H(X_1|X^0_{-n+1}) \approx H \approx H(X_1|X^0_{-K})$, for some K (the "memory" of the source). Hence, unless n is large enough so as to make, with high probability, $K(X^1_{-n}) \geq K$, the universal prediction error (as measured by $D_{X^0_{-n+1}}(P \parallel Q)$) can not vanish.

On the other hand we can also use the HZ data compression scheme to construct an empirical measure that works well for values of n which are just "right", namely, for which $H(X_1|X^0_{-n+1}) \approx H \approx H(X_1|X^0_{-K})$.

Claim: Let

$$Q(X_1^\ell|X^0_{-n+1}) = \frac{2^{-L_{HZ}(X_1^\ell|X^0_{-n+1})}}{\sum 2^{-L_{HZ}(X_1^\ell|X^0_{-n+1})}}$$

where $L_{HZ}(X_1^\ell|X^0_{-n+1})$ is the length function of the HZ universal encoder. Then:

$$D_{X^0_{-n+1}}(P \parallel Q) \leq H(X_1|X^0_{-K(X^1_{-n})+\ell}) - H(X_1|X^0_{-n+1}) + 0\left(\frac{\log\log n}{\ell}\right)$$

for $O(\log\log n) \leq \ell \leq O(\log n)$.

Thus the empirical measure generated from the HZ length functions is close to the true measure P once n satisfies $H(X_1|X^0_{-K(X^1_{-n}}) \approx H(X_1|X^0_{-n+1})$.

IV. On the role of pattern matching in entropy estimation

We have seen already how a pattern matching based approach to estimating a probability distribution has led to universal data compression algorithms and universal classifiers and predictors. In this section we demonstrate, both theoretically and with an e xample, how the entropy of a stochastic source can be estimated using pattern matching.

Shannon discovered that the entropy of a stochastic process has physical meanings: It measures a source's predictability as well as its uncertainty. It is also a computable measure of complexity. It even has a gambling interpretation [1]. The estimation process begins with a sequence of observations from a stochastic source. Since the entropy is a function of the probability law estimation can always be accomplished by forming the empirical probability measure and calculating the actual entropy of the estimated probability distribution. As pointed out earlier, this "plug-in" approach is not always accurate: to be successful it usually requires model assumptions and large amounts of data. This estimate of the entropy is only as good as the estimate of the probability measure.

We have seen seen that compression can be accomplished using pattern matching in situations where a straightforward Shannon code is either impossible to construct or not likely to work. Thus, one should expect that entropy estimation could also be accomplished by means of pattern matching in situations where the probability law cannot be accurately determined. This is indeed the case, as demonstrated below.

Let us return to the discussion of the relationship of the recurrence time of a random sequence looking backward into the past and the sequence's probability. We have seen that

$$\lim_{\ell \to \infty} \frac{\log N_\ell}{\ell} = -E[\log P(X_1^\ell)] = H, \text{ with probability 1.}$$

The recurrence time theorem offers a reliable way to approximate the entropy which is widely applicable since it holds for all stationary, ergodic sources. On the other hand it is quite impractical since the convergence is slow.

Stronger results are possible if P is assumed to satisfy an appropriate vanishing memory condition. For example, given any $t > 0$, it follows [31] that

$$\Pr\{N_\ell P(X_1^\ell) > t\} \approx \exp(-t). \tag{19}$$

The result is surprisingly general: It holds for d-dimensional random fields with memory restrictions and for ℓ also random, but (almost) independent of the past. An example of such a random length is the stopping time $T_i(n) = \max\{k : -\log(P(X_i^{i+k}) > \log n\}$. Following the discussion in the introduction we have (for vanishing memory sources) that $|L_i(n) - T_i(n)| = O(1)$, where $L_i(n)$ is the longest match of sequence $X_i, X_{i+1}, \ldots$ into the past n observations: X_{i-n}^{i-1}. The entropy reappears in the theory, since [14]

$$\lim_{n\to\infty} \frac{ET_i(n)}{\log n} = \frac{1}{H}.$$

Thus it is true that for sources with an appropriate vanishing memory condition:

$$\frac{EL_i(n)}{\log n} = \frac{1}{H} + \frac{O(1)}{\log n}.$$

Similar results hold even for processes whose memory vanishes quite slowly [15].

We construct an entropy estimation algorithm based on the mean convergence of $L_i(n)$ to $\frac{1}{H}$. Consider the following: Let k and n be chosen arbitrarily. Given observations X_{-n+1}^k from P with entropy H, let $L_i(n)$ be the match length function as defined above. Define

$$\hat{H}(n,k) = \frac{\log n}{\sum_{i=1}^k L_i(n)}.$$

Since the match lengths are calculated into a sliding window of length n, we label this the "sliding window" entropy estimator (abbr. SWE). In many respects, the estimate is basically an achievable compression ratio; that is, $\hat{H}$ measures the "compression" stripped of excess overhead which can be substantial [11]. Thus, the advantages of pattern matching based coding also apply to pattern matching based entropy estimation. Specifically, pattern matching based entropy estimation is useful when one or more of the following is likely to be true:

- The model (or model class) is unspecified.
- The effect of model misspecification is large.
- The data has more than trivial dependencies.
- The number of observations is small. (equivalently, the source statistics change over time, even if the entropy does not)

A single real example should make some of these issues more concrete. Since entropy is closely identified with information and complexity, there is consequently great interest in estimating the entropy of genetic sequences. The genetic code is billions of bases in length (a base is one of four letters: A,G,T or C), with a distinct time arrow and finite memory. Yet DNA is not stationary. The code is divided into distinct regions of known and unknown function. In this experiment we consider 25460 bases [36] that comprise the coding regions (exons) of section DS02740 of the Drosophila Melanogaster (the fruit fly) [4]. We choose to work only with the exons because the exon entropy is known to be closer to the maximum of 2. We point out that it is difficult to determine or even define stationarity in this setting. It is hoped that the sequence of concatenated exons will be more stationary than a contiguous stretch of DNA. We report that the marginal frequency of each base remains fairly constant over the entire sequence.

We denote our sequence by X_1^N, with $N = 25460$, and we compute the sliding window entropy estimate for varying parameters. As a standard of comparison, we compute plug-in estimates of the entropy using different order Markov models. That is, for varying k we compute the empirical probabilities $\hat{P}_\ell(\cdot)$ for k-vectors $x \in \{A, G, C, T\}^k$. Then we let

$$\hat{H}(k) = \frac{1}{k} \sum_{x \in \{A,C,G,T\}^k} -\hat{P}_k(x) \log \hat{P}_k(x).$$

To investigate the robustness of this procedure we take a plug-in approach. We do not know the true empirical distribution $P_k(\cdot)$, for any k. We can however, assume that $\hat{P}_k(\cdot)$ is the

[4]The entire 83527 base pair sequence is located in Genbank, accession number L49408.

true distribution on k-tuples. With this assumption in place we can simulate from $\hat{P}_k(\cdot)$ to generate replicates of the original sequence, each of length 25460.

In our experiment we generate 200 replicates for varying k: these we label $X^*_{i,k}$ for $i = 1, \ldots, 200$. We can compute, for any j the average of the j^{th} order entropy estimates over all 200 replicates. These we label $\hat{H}^*_k(j)$. By comparing $\hat{H}^*_k(j)$ to $\hat{H}(k)$ we can estimate bias and measure the effect of model misspecification. To this end, we report the quantity $E(k,j) = \frac{\hat{H}^*_k(j) - \hat{H}(k)}{2 - \hat{H}(k)}$, which corresponds to the relative error in redundancy incurred by specifying a j^{th} order model when the true model is k^{th} order.

model-order	$\hat{H}(k)$	$\hat{H}^*_k(1)$ $[E(k,1)]$	$\hat{H}^*_k(2)$ $[E(k,2)]$	$\hat{H}^*_k(4)$ $[E(k,4)]$
$k = 1$	1.98	1.98 [0]	1.98 [.1]	1.95 [2.1]
$k = 2$	1.98	1.99 [.15]	1.97 [.15]	1.93 [1.7]
$k = 4$	1.93	1.98 [.64]	1.98 [.92]	1.91 [.285]

Table 1: Markov Model Entropy Estimates

The entropy estimates vary from 1.98 (first order Markov) to 1.93 (fourth order Markov). This is a three fold increase in the redundancy . Observe that the relative error is small if model is specified correctly ($E(k,k)$ ranges from a minimum of 0 when $k = 1$ to a maximum of .285 for $k = 4$) Thus the plug-in approach is not too bad (especially for small k) if the model is accurately specified. On the other hand, the effect of model misspecification can be very large (as measured by $E(k,j)$ for $j \neq k$). The worst errors result from specification of a large model when in fact a small one is true. Significant errors are also observed when a small model is assumed when a larger one is in fact true.

In contrast to the uncertainty of the plug-in approach, the estimates based on pattern matching are universal and thus no model selection is required. Since the expected difference between $L_i(n)$ and $\frac{1}{H}$ tends to zero like $O(\frac{1}{\log n})$ there is a considerable bias problem associated with the SWE for even reasonably large values of n. This problem is fixable. It is possible to estimate a model for the sequence and correct for this bias again using the bootstrap (see [35]) [5]. As before, we would generate replicates of X_1^N using a parametric approximation

[5]Bias correction using the bootstrap is not always possible. An accepted practice is to test the consistency

for the unknown "source" that generated the DNA sequence. The entropy of each replicate can then be computed using the SWE for varying window sizes. These values would then be subtracted from the known entropy of the replicate sequence, and then averaged over all replicates to estimate the bias of the SWE. Correcting for bias has the effect of restoring the natural entropy scaling (with a maximum of 2). We present in Table 2 the SWE estimates of the entropy, both bias corrected and uncorrected, computed for varying choices in n (the window size). ¿From table 2 we notice that the size of the bias adjustments diminish as n

Window Size	Mean Match Length	$\hat{H}$	Bias Corrected $\hat{H}$
64	3.12	1.92	1.86
128	3.61	1.94	1.89
512	4.60	1.96	1.92
1024	5.07	1.97	1.94

Table 2: Sliding Window Entropy Estimates

increases. This follows from the theory which predicts a bias proportional to $\frac{1}{\log n}$. Observe also that the uncorrected entropy estimates increase in n, which is surprising since entropy estimates usually decrease as the window size increases (this is analogous to improvements in code performance as block length and delay increase). This is evidence that the substantial drop in entropy is due to local features in the genetic code. We confirm this by computing the quantities L_i equal to largest k such that a copy of the sequence x_i^{i+k-1} is contained *anywhere* in the sequence. The main difference between L_i and $L_i(n)$ is the latter only looks for matches into the past n observations but the former searches through the entire sequence, front and back. The resulting estimate was proposes by Grassberger [21]

$$\hat{H}_G = \frac{\log N}{\sum_{i=1}^{N} L_i},$$

where N is the total number of observations. In our example $N = 25460$. For stationary sequences the Grassberger estimate behaves much like the SWE but only for large n (near N). In our example, the Grassberger estimate is 1.98. Since this estimate is so much higher

of the bias correction procedure with known models. This has been established for the sliding window estimate of entropy.

than the estimates obtained with smaller windows we speculate that the statistics of higher order patterns are not consistent with an assumption of stationarity over the entire sequence (despite the approximate constancy of the marginal frequencies).

The lowest estimate (and thus the best) is 1.86 obtained from the SWE with a short window (likely necessary to account for slowly changing statistics) and then adjusted for bias. In real terms, this implies that sequence contains more than a tenth of a bit of redundancy per symbol. For the curious, a great deal more theory and application of this method can be found in [15] (as applied to the English language) and [13] (as applied to DNA).

V. Concluding Remarks

We have tried to motivate and explain applications of pattern matching to a variety of problems in information theory. Although it has been more than twenty years since the publication of [11] and ten years since [8] we are still surprised at how easily and thoroughly pattern matching is able to uncover information about a probability measure. In our paper, we have chose to restrict our discussion of pattern matching to fundamental concerns and basic applications. Regrettably, we have omitted discussion of a great variety of substantial and important works. Indeed, the literature on the subject continues to grow in a variety of directions.

One area of great activity concerns the extension of pattern matching ideas to approximate string matching and "lossy" data compression. This work has led to a variety of noisy data compression algorithms based on LZ that are also characterized by a small computational complexity. There are a number of publications that discuss the role pattern matching in lossy data compression [22, 18, 25, 24, 26, 27, 28, 37]. It seems, however, that low computational complexity is achievable only at the expense of yielding a non-optimal distortion.

VI. Appendix

Kac's lemma states that the average distance between occurrences of a fixed pattern is equal to the inverse of the probability of the pattern. Consider now the fixed pattern x^{ℓ}_{-k+1}. In any long realization the proportion of times the pattern x_1^{ℓ} occurs after x^0_{-k+1} will be nearly $P(X_1^{\ell} = x_1^{\ell} | x^0_{-k+1})$. Equivalently, we would expect $\frac{1}{P(X_1^{\ell}=x_1^{\ell}|x^0_{-k+1})}$ occurrences of x^0_{-k+1} for every occurrence of x_1^{ℓ}.

Below we state the conditional version of Kac's Lemma. In [7] this lemma is used to analyze the HZ context algorithm. On its own, it yields an efficient data compression scheme, although not necessarily as efficient as the "optimal" HZ algorithm.

Modified Kac's Lemma [7]: Let $N_{\ell+k}$ be the time of the first recurrence of the pattern X^{ℓ}_{-k+1} moving backward into the training sequence $X^{0}_{-N_0+1}$. If there is no recurrence then let $N_{k+\ell} = N_0$.

(a)

$$E_{X^{\ell}_{-k+1}} \left\{ \sum_{i=1}^{N_{\ell+k}} 1\{X^{-i}_{-k+1-i} = X^{0}_{-k+1}\} \right\} \leq \frac{1}{P(X_1^{\ell}|X^{0}_{-k+1})} \tag{20}$$

(Equality is achieved for $N_0 \to \infty$). We could then average over X^{ℓ}_{-k+1} to prove:

(b)

$$\frac{1}{\ell} E \log \sum_{i=1}^{N_{\ell+k}} 1\{X^{-i}_{-k+1-i} = X^{0}_{-k+1}\} \leq H(X_1^{\ell}|X^{0}_{-k+1}) \,.$$

The HZ universal coding scheme: We shall now describe the HZ universal coding scheme which, by adaptively changing k, fully utilizes the training sequence in an appropriate way.

Consider blocks of length ℓ'. Let $H_0 \ll \log A$ and δ be some arbitrary positive numbers. Define $\ell = \frac{\ell'}{\delta}$ and $N_0 = \ell\, 2^{H_0 \ell}$. Furthermore, let

$$\hat{\imath} = \max_i \left\{ i \;:\; N_{\ell'+i}(X^{\ell'}_{-N_0+1}) < N_0 - i \right\} \tag{21}$$

(i.e. $X^{\ell'}_{-\hat{\imath}+1}$ is the *longest* $X^{\ell'}_{-i+1}$ that re-occurs in $X^{\ell'-1}_{-N_0+1}$).

Let

$$K(X^{\ell'}_{-N_0+1}) = \begin{cases} \min\{\hat{\imath}\,;\; \ell - \ell'\} - 1\,; & \hat{\imath} \geq 1 \\ 0; & \text{otherwise}\,. \end{cases} \tag{22}$$

The block $X_1^{\ell'}$ is encoded into a binary string which consists of the binary expansion of $K(X^{\ell'}_{-N_0+1})$ (about $\log \ell$ bits), followed by the binary expansion of the pointer to the first occurrence of $X_1^{\ell'}$ in $X^{\ell'-1}_{-N_0+1}$ among all ℓ' vectors with a prefix that is equal to $X^{\ell'-l}_{K(X^{\ell'}_{-N_0+1})}$ (This takes about $\log N_{\ell'+\hat{\imath}}(X^{\ell'}_{-N_0+1}) + \log\log N_0$ bits).

Proof of Claim (a):

Assume that one is given a coding procedure that parses a long block x_1^n into variable length phrases, using sliding-window or fixed data-base training sequence, then apply the appropriate V-V code to each phrase. The goal of claim (a) is to show that almost the same performance can be obtained by parsing into fixed-length ℓ-phrases and using a F-V code on each ℓ-phrase, where the particular code used is allowed to depend on the past of the phrase.

Lemma 1: Let C be a complete and proper set of variable length words and let $L(w)$ be the length function for the word w . For each j, each $1 \le k \le j$ and each x_1^K, there are prefix codes on $\mathbf{A^j}$ and on $\mathbf{A^j_{k+1}}$ with respective length functions $L_j(X_1^k)$ and $L_j(X_{k+1}^j|X_1^k)$ such that,

$$L_j(X_1^k) + L_j(X_{k+1}^j|X_1^k) \le L(X_1^j) + 2, X_1^j \in \mathbf{A^j}$$

Proof: Extend $L(w)$ to ALL words by defining L(w)=∞ if w is not in C. Fix j and define the distribution

$$Q_j(w) = \frac{2^{-L(w)}}{\sum_{z \in \mathbf{A^j}} 2^{-L(z)}}, w \in \mathbf{A^j}.$$

For each $1 \le k \le j$, let $Q_{j,k}(.)$ be the projection of Q_j onto $\mathbf{A^k}$. Also let, for each X_1^k, let $Q_j(.|X_1^k)$ be the *conditional* distribution defined by the following two formulas,

$$Q_{j,k}(X_1^k) = \sum_{X_{k+1}^j} Q_j(X_1^j), X_1^j \in \mathbf{A^k},$$

$$Q_j(X_{k+1}^j|X_1^k) = \frac{Q_j(X_1^j)}{\sum_{Z_1^j : Z_1^k = X_1^k} Q_j(Z_1^k)}, X_{k+1}^j \in \mathbf{A^j_{k+1}}.$$

Let $L_j(X_1^k)$ be the length function for the Shannon code defined by $Q_{j,k}(.)$ and let $L_j(X_{k+1}^j|X_1^k)$ be the the length-function of the Shannon code defined by $Q_j(.|X_1^k)$. The factorization

$$Q_j(X_1^j) = Q_{j,k}(X_1^k) Q_j(X_{k+i}^j|X_1^k)$$

combined with the fact that, being a length function, $L(w)$ satisfies the Kraft inequality and therefore $-\log Q_j(w) \leq L(w)$, completes the proof of Lemma 1 above.

V-V to F-V Theorem: Suppose X_1^n is coded by a V-V code with a training sequence of length N_0 and delay $\tau = O(\log N_0)$ into a binary sequence with length $L(X_1^n)$. Given $\ell \leq \tau = O(\log N_0)$, there is a F-V code with block-length ℓ and training sequence of length N_0 that, given the suffix $X^0_{-\ell+1}$, encodes X_1^n into a binary sequence of length $L^{'}(X_1^n)$ such that

$$L^{'}(X_1^n) \leq L(X_1^n) + \frac{n}{\ell} O(\log\log N_0)$$

Proof: Change notation so that X_1^ℓ is the next ℓ -block to be encoded and, for the V-V original parsing of the n-sequence, let $s(1) \leq 1 \leq e(1)$ be the left and right endpoints of the parsed phrase that includes X_1 and let $s(2) \leq \ell \leq e(2)$ be the left and right endpoints of the parsed phrase that includes X_ℓ. Assume that the encoder and decoder both know how the n-sequence was parsed by the V-V code, starting from any position $s(1)$.

Assume for the moment that the positions $s(1), e(1), s(2)$ and $e(2)$ are known to both the encoder. The encoder first transmits these values to the decoder. This requires $4 \log \tau$ bits. The encoder next transmits the block $X_1^{e(1)}$ using the Shannon code defined by the conditional distribution $Q_{e(1)-s(1)+1}(\cdot | X^0_{s(1)}))$, as defined in Lemma 1. The block $X_{e(1)+1}^{s(2)-1}$ is then transmitted using the V-V code, and finally, the block $X_{s(2)}^\ell$ is transmitted using the Shannon code defined by the projection of the distribution $Q_{e(2)-s(2)+1}(\cdot)$ onto its first $\ell - s(2) + 1$ coordinates, as defined in Lemma 1. The decoder, knowing the values of $s(1), e(1), s(2)$ and $e(2)$, as well as the V-V code and hence the codes of Lemma, can correctly decode.

However, the encoder need not know the values $s(1), e(1), s(2)$ and $e(2)$. The encoder can try all possible values $s(1), e(1), s(2)$ and $e(2)$ and determines the values that produce the shortest code, transmits these values and uses the corresponding code. This can only improve code performance. This, together with Lemma 1 complete the proof of the V-V to

F-V Theorem and claim (a).

Proof of Claim (c): We begin with the independent fixed database. Thus the training sequence is the vector of observations $Y^0_{-N_0+1}$ and the incoming data is X. We remind the reader that Y has the same distribution as X. We begin the proof by conditioning on the random variable $\{S(1) = -t\}$ where $t < \tau$. Let ℓ be the length of the fixed block-length algorithm whose expected compression is nearly the expected compression of the original variable length algorithm, by claim (a). It follows that

$$\begin{aligned}\frac{1}{\ell}EL(X_1^\ell|Y^0_{-N_0+1}, X^0_{S(1)}; S(1)) \;&\geq\; \frac{1}{\ell}H(X_1^\ell|X^0_{S(1)}, Y^0_{-N_0+1}) - \frac{1}{\ell}H(S(1)|X^0_{S(1)}, Y^0_{-N_0+1}) \\ &\geq \frac{1}{\ell}H(X_1^\ell|X^0_{S(1)}) - \frac{\log\tau}{\ell}.\end{aligned}$$

For the sake of clarity, the unormalized form of the entropy function was used here.

The last inequality follows specifically from the independence of the database $Y^0_{-N_0+1}$ and the incoming data X. Now, $-S(1) < K$ by definition. Hence

$$\frac{1}{\ell}EL(X_1^\ell|Y^0_{-N_0+1}, X^0_{S(1)}) \geq \frac{1}{\ell}H(X_1^\ell|X^0_{-K}) - \frac{\log\tau}{\ell}.$$

This completes the proof of (c) for the independent fixed data base.

Under an appropriate "vanishing memory" condition [17], one can view the LZ sliding window case [6] as a slightly shorter fixed database case; that is, if $X^{-k}_{-N_0+1}$ is "almost" independent of X_1^ℓ. The construction of claim (a) and the proof of claim (c) (for the independent fixed database) can be then be adapted provided that it can be shown that match lengths into the shortened data base are expected to be nearly the same as the original match lengths: i.e. if $EL(X_1^\ell|X^0_{-N_0+1}) \approx EL(X_1^\ell|X^0_{S(1)}, X^{S(1)-K-1}_{-N_0+1})$.

Acknowledgment: The authors wish to thank Neri Merhav, Paul Shields, W. Szpankowsky, Jack Wolf and Frans Willems for valuable remarks. The V-V to F-V Theorem in the Appendix was greatly revised and simplified by Paul Shields.

References

[1] Thomas M. Cover & Joy A. Thomas: "Elements of Information Theory", John Wiley & Sons Inc. 1991.

[2] B.M. Fitingof: "The Compression of Discrete Information", *Prob.Inform Trans. Vol.,* **3** pp. 28-36, 1967.

[3] F.M.J Willems, Y.Shtarkov and T.J. Tjalkens: "The Contex-Tree Weighting Method: Basic Properties", *IEEE Trans. on Information Theory* **41** , pp. 653-664 , May 1995

[4] M. Kac, "On the notion of recurrence in discrete stochastic processes", *Bull. Amer. Math. Soc.*, **53**, pp. 1002-1010, Oct. 1947.

[5] F.J. Willems: "Universal Compression and Repetition Times", *IEEE Trans. on Information Theory* **35**, No. 1, January 1989, pp. 54-58.

[6] A.D. Wyner and J. Ziv: "The Sliding-Window Lempel-Ziv Algorithm is Asymptotically Optimal", (Invited paper), *Proceedings of the IEEE,* **82**, June 1994, pp. 872- 877.

[7] Y. Hershkovits and J. Ziv: "On Sliding-Window Universal Data Compression with Limited Memory", *IEEE Trans. on Information Theory*, vol 44, pp. 66-78, January 1998

[8] A.D. Wyner and J. Ziv: "Some Asymptotic Properties of the Entropy of a Stationary Ergodic Data Source with Applications to Data Compression", *IEEE Trans. on Information Theory*, Vol. IT–35, pp. 1250–1258, November 1989.

[9] D. Ornstein and B. Weiss, "Entropy and Data Compression Schemes". *IEEE Trans. on Information Theory*, Vol. 39, January 1993. pp 78-83.

[10] A.D. Wyner, "1994 Shannon Lecture: Typical Sequences an All That: Entropy, Pattern Matching and Data Compression". *IEEE Information Theory Newsletter*, June, 1995.

[11] J. Ziv and A. Lempel: "A Universal Algorithm for Sequential Date- Compression". *IEEE Trans. on Inf. Theory*, Vol. IT–23, No. 3, May 1977, pp. 337–343.

[12] J.Rissanen; "Universal Coding, Information, Prediction, and Estimation" *IEEE Trans. on Inf. Theory*, Vol. IT-30, No. 4, pp. 629-636.

[13] M. Farach, M. Noordewier, S. Savari, L. Shepp, A. Wyner, and J. Ziv. "On the Entropy of DNA: Algorithms and Measurements based on Memory and Rapid Convergence" . *Symposium on Discrete Algorithms* (SODA), 1995.

[14] A.J.Wyner. "The Redundancy and Distribution of the Phrase Lengths for the Fixed-Database Version of the Lempel- Ziv Algorithm". *IEEE Transactions of Information Theory*, September 1997.

[15] I. Kontoyiannis, P.H. Algoet, Yu. M. Suhov and A.J. Wyner. Non-Parametric Entropy Estimates for Stationary Processes and Random Fields with applications to English text". *IEEE Transactions Information Theory.* May, 1998. 1319-1327.

[16] Wyner, A. D. and Wyner, A.J. "Improved Redundancy of a Version of the Lempel-Ziv algorithm" *IEEE Trans. on Information Theory*, Volume 41, pp. 723-731, May, 1995.

[17] Wyner, A. D. and Ziv, J."Classification with Finite Memory", *IEEE Trans. on Information Theory* Vol. 42, No. 2, March 1996, pp. 337-347.

[18] Louchard, G. and Szpankowski, W. "On the Average Redundancy Rate of the Lempel-Ziv Code". *IEEE Transactions on Information Theory*, Vol. 43, No. 1, 1-7 1997.

[19] Shields, P.C. "The Ergodic Theory of Discrete Sample paths" American Mathematical Society 1996.

[20] Savari, A. "Redundancy of the Lempel-Ziv Incremental Parsing Rule". *IEEE Transactions on Information Theory*, Vol. 43, No. 1, 1997.

[21] Grassberger, P. "Estimating the information content of symbol sequences and efficient codes". *IEEE Transactions on Information Theory* vol. 35, pp 669-675, May, 1989.

[22] P. C. Shields, "Approximate-match waiting times for the substitution/deletion metric," preprint, submitted to ISIT-98.

[23] T. Luczak and W. Szpankowski, "A lossy data compression based on string matching: preliminary analysis and suboptimal algorithms," preprint 1997.

[24] E.H. Yang and J. C. Kieffer, "On the performance of data compression algorithms based upon string matching," IEEE Trans. on IT, vol. 44, no. 1, pp. 47–65, January 1998.

[25] Y. Steinberg and M. Gutman, "An algorithm for source coding based upon string matching," *IEEE Trans. on Information Theory* , vol. 39, no. 3, pp. 877-886, May 1993.

[26] I. Kontoyiannis, "A practical lossy version of the Lempel-Ziv algorithm that is asymptotically optimal - part I: memoryless sources," preprint 1998.

[27] H. Morita and K. Kobayashi, "An extension of LZW coding algorithm to source coding subject to a fidelity criterion," In 4th Joint Swedish-Soviet Int. Workshop on Inform. Theory, pp. 105-109, Gotland, Sweden 1989.

[28] E.-h. Yang and J. C. Kieffer, "Simple universal lossy data compression schemes derived from the Lempel-Ziv algorithm," *IEEE Transactions of Information Theory*, vol. 42, no. 1, pp. 239-245, January 1996.

[29] E.H. Yang and J.C. Kieffer, "On the Redundancy of the Lempel-Ziv Algorithm for ψ-mixing sources", *IEEE Transactions of Information Theory*, **43**, 1101-1111, 1997.

[30] P. Jaquet and W. Szpankowski, "Autocorrelation on Words and its Applications. Analysis of Suffix Trees by String-Ruler Approach". *J. Combinatorial Theory, Ser. A*, 66, pp. 237-269, 1994.

[31] A.J. Wyner. "More on Recurrence and Waiting Times". Technical Report, #486, Department of Statistics, University of California, Berkeley. September, 1996. To Appear in the *Annals of Applied Probability.*

[32] L.J. Guibas and A.M. Odlysko, " String overlaps, pattern matching, and non-transitive games" *Comb, Theory and App.*, Vol. 30, pp. 183-208, 1981.

[33] Shuo-Yen Robert Li, "A martingale approach to the study of occurrence of sequence patterns in repeated experiments", *Annals of Probability*, Vol 8, 1171-1176, 1980

[34] E. Plotnick, M.J. Weinberger, and J. Ziv, "Upper bounds on the probability of sequences emitted by finite-state sources and on the Redundancy of the Lempel-Ziv Algorithm", *IEEE Transactions of Information Theory*, Vol 38, No. 1 pp 66-72, January 1992

[35] B. Efron and R. Tibshirani. "An Introduction to the Bootstrap. Chapman and Hall, London. 1993.

[36] Rubin, Gerry. *Berkeley Drosophila Genome Project.* May, 1997. Private Communication.

[37] N.Merhav and J.Ziv, "A Measure of Relative Entropy between Individual Sequences with Application to Universal Classification", *IEEE Trans. on Information Theory* Vol 39, pp. 1270–1279, July 1993.