Data Minability Evaluation by Compression – An Experimental Study

Dan A. Simovici, Dan Pletea, and Saaid Baraty
University of Massachusetts Boston,
Boston, USA,
{dsim,dpletea,sbaraty}@ cs.umb.edu

***Abstract*—The effectiveness of compression algorithms is increasing as the data subjected to compression contains patterns which occur with a certain regularity. This basic idea is used to detect the existence of regularities in various types of data ranging from market basket data to undirected graphs. The results are quite independent of the particular algorithms used for compression and offer an indication of the potential of discovering patterns in data before the actual mining process takes place.**

Keywords-data mining; lossless compression; LZW; market basket data; patterns; Kronecker product.

I. Introduction

Our goal is to show that compression can be used as a tool to evaluate the potential of a data set of producing interesting results in a data mining process. The basic idea that data that contains patterns that occur with a certain regularity will be compressed more efficiently compared to data that has no such characteristics. Thus, a pre-processing phase of the mining process should allow to decide whether a data set is worth mining, or compare the interestingness of applying mining algorithms to several data sets.

Since compression is generally inexpensive (and certainly less expensive than mining algorithms), and compression methods are well-studied and understood, pre-mining using compression will help data mining analysts to focus their efforts on mining resources that can provide a highest payout without an exorbitant cost.

Compression has received lots of attention in the data mining literature. As observed by Mannila [14], data compression can be regarded as one of the fundamental approaches to data mining [14], since the goal of the data mining is to “compress data by finding some structure in it”.

The role of compression developing parameter-free data mining algorithms in anomaly detection, classification and clustering was examined in [8]. The size $C(x)$ of a compressed file x is as an approximation of Kolmogorov complexity [4] and allows the definition of a pseudo-distance between two files x and y as

$$d(x,y) = \frac{C(xy)}{C(x)+C(y)},$$

where xy is the file obtained by concatenating x and y. Note that this is not the common definition of a pseudo-distance (see, for example [21]); instead, it is simply a numerical evaluation of the similarity of the files x and y; its minimal value is obviously equal to 0.5.

Further advances in this direction were developed in [9], [10] and [23]. A Kolmogorov complexity-based dissimilarity was successfully used to texture matching problems in [3] which have a broad spectrum of applications in areas like bioinformatics, natural languages. and music. Compression algorithms are used in the actual mining process to handle data mining explorations that return huge sets of results by extracting those results that actually are representative of the data set (see, for example [19], [22]).

Our goal in this paper is to show that compression can be used for assessing the interestingness of applying an actual data mining process. In other words, to evaluate the minability of a data set using compression. We justify experimentally this idea by evaluating data sets that have different characteristics and sources.

In general, data mining is task-oriented and the mining process entails seeking specific patterns. Thus, our assessment of minability will not necessarily help identify patterns of interest; instead, it will signal that such patterns may exist and will invite to further exploration.

There are two broad classes of compression algorithms: lossy compression, that reduces significantly data but does not allow the full inverse transformation, from compressed data to the original data, and lossless compression, that achieves data reduction and can be completely reversed. We illustrate the use of lossless compression in pre-mining data by focusing on several distinct data mining processes: files with frequent patterns, frequent item sets in market basket data, and exploring similarity of graphs.

The LZW (Lempel-Ziv-Welch) algorithm was introduced in 1984 by T. Welch in [24] and is among the most popular compression techniques. The algorithm does not need to check all the data before starting the compression and the performance is based on the number of the repetitions and the lengths of the strings and the ratio of 0s/1s or true/false at the bit level. There are several versions of the LZW algorithm. Popular programs (such as Winzip or the zip function of MATLAB) use variations of the LZW compression. These algorithms work both at the bit level and at the character level.

An important role in evaluating concentrations of values in various probability distributions is played by the notion of entropy, Namely, if $\mathbf{p} = (p_1, \ldots, p_n)$ is a probability distribution with $p_i \leqslant 0$ and $\sum_{i=1}^{n} p_i = 1$, the entropy of this distribution is

$$\mathcal{H}(\mathbf{p}) = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}.$$

It is well-known (see [6], [13]) that the maximum value of the entropy is obtained when

$$p_1 = p_2 = \cdots = p_n = \frac{1}{n}$$

and this value is $\log_2 n$. The minium value is 0 and this occurs when there exists p_i with $p_i = 1$ and $p_j = 0$ for $j \neq i$, $1 \leqslant j \leqslant n$. The entropy helps us to evaluate the diversity of the values assumed by a random variable: the more concentrated these values are the lower the entropy.

After examining compressibility of binary strings in Section II we explore several experimental settings that provide strong empirical evidence of the correlation between compression ratio and the existence of hidden patterns in data. In Section III we discuss the compressibility of sequences of symbols produced by various generative mechanisms. Section IV is dedicated to the compressibility of adjacency matrix for graphs relative to the entropy of distribution of subgraphs. Finally, in Section V, we examine the compressibility of files that contain market basket data sets. This paper is an extension of our contribution [1].

II. Patterns in Strings and Compression

An *alphabet* is a finite and non-empty set whose elements are referred to as *symbols*. Let A^* be the set of sequences on the alphabet A. We refer to these sequences as *words* or *strings*. The length of a string w is denoted by $|w|$. The null string on A is denoted by λ and we define A^+ as $A^+ = A^* - \{\lambda\}$. The subsets of A^* are referred to as *languages* over A.

If $w \in A^*$ can be written as $w = utv$, where $u, v \in A^*$ and $t \in A^+$, we say that the pair (t, m) is an occurrence of t in w, where m is the length of u.

The occurrences (x, m) and (y, p) are overlapping if $p < m + |x|$ and $m < p + |y|$. If this is the case, $m < p$ and $p + |y| > m + |x|$ then there is a proper suffix of x that equals a proper prefix of y. If x is a word such that the sets of its proper prefixes and its proper suffixes are disjoint, there are no overlapping occurrences of x in any word.

The number of occurrences of a string x in a string w is denoted by $n_x(w)$. Clearly, we have $\sum\{n_a(w) \mid a \in A\} = |w|$ for any symbol $a \in A$. The *prevalence* of x in w is the number $f_x(w) = \frac{n_x(w)\cdot|x|}{|w|}$ which gives the ratio of the characters contained in the occurrences of t relative to the total number of characters in the string.

The result of applying a compression algorithm C to a string $w \in A^*$ is denoted by $C(w)$ and the *compression ratio* is the number

$$\mathsf{CR}_C(w) = \frac{|C(w)|}{|w|}.$$

We shall use the binary alphabet $B = \{0, 1\}$ and the LZW algorithm, the compression algorithm of the package `java.util.zip`, or the `zip` function of MATLAB.

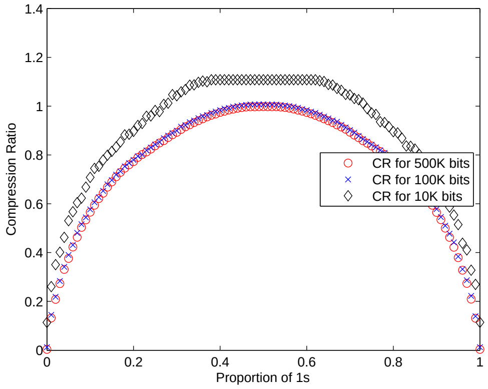


Figure 1. Baseline CR_{jZIP} Behavior

We generated random strings of bits (0s and 1s) and computed the compression ratio for strings with a variety of symbol distributions. A string w that contains only 0s (or only 1s) achieves a very good compression ratio of $\mathsf{CR}_{jZIP}(w) = 0.012$ for 100K bits and $\mathsf{CR}_{jZIP} = 0.003$ for 500K bits, where $jZIP$ denotes the compression algorithm from the package `java.util.zip`. Figure 1 shows, as expected, that the worst compression ratio is achieved when 0s and 1s occur with equal frequencies.

For strings of small length (less than 10^4 bits) the compression ratio may exceed 1 because of the overhead introduced by the algorithm. However, when the size of the random string exceeds 10^6 bits this phenomenon disappears and the compression ratio depends only on the prevalence of the bits and is relatively independent on the size of the file. Thus, in Figure 1, the curves that correspond to files of size $100K$ bits and $500K$ bits overlap. We refer to the compression ratio of a random string w that contains $n_0(w)$ zeros and $n_1(w)$ ones as the *baseline compression ratio* for this distribution of bits.

We created a series of binary strings $\varphi_{t,m}$ which have a minimum guaranteed number m of occurrences of patterns $t \in \{0, 1\}^k$, where $0 \leqslant m \leqslant 100$. The compression baselines for files containing the patterns 01, 001,0010, and 00010 are shown in Table I.

TABLE I. Baseline Compression Ratio for Files Containing a Minimum Guaranteed Number of Patterns

Pattern	Proportion of 1s	Baseline
01	50%	1.007
001	33%	0.934
0010	25%	0.844
00010	20%	0.779

Specifically, we created 101 files $\varphi_{001,m}$ for the pattern 001, each containing 100K bits and we generated similar series for $t \in \{01, 0010, 00010\}$. In the case of the 001 pattern the baseline is established at 0.934, and after the prevalence exceeds 20% the compression ratio drops dramatically. Results of the experiment for 001 are shown in Table II. In Figure 2 we show that similar results hold for all patterns mentioned

TABLE II. PATTERN '001' PREVALENCE VERSUS THE COMPRESSION RATIO CR_{jZIP}

Prevalence of '001' pattern	CR_{jZIP}
0%	0.93
10%	0.97
20%	0.96
30%	0.92
40%	0.86
50%	0.80
60%	0.72
70%	0.62
80%	0.48
90%	0.31
95%	0.19
100%	0.01

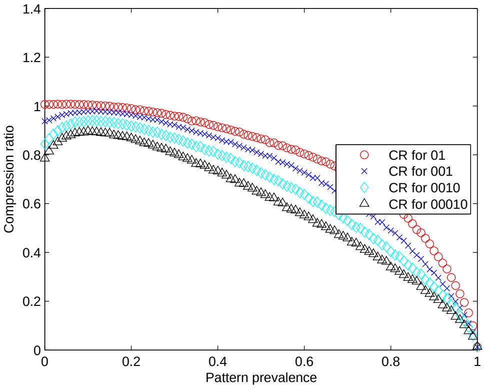


Figure 2. Dependency of Compression Ratio on Pattern Prevalence

above.

III. COMPRESSIBILITY OF LANGUAGES AND SEQUENCES

Sequences or sets of sequences of symbols are often subjected to data mining processes and identifying those sequences that contain interesting patterns before the actual mining process may be computationally significant.

We begin by examining the well-known sequence called the Thue-Morse sequence [2] that has many applications ranging from crystal physics [16], counter synchronization [25], metrology [7], [11], and chess playing [15], as well as in game theory, fractals and turtle graphics, chaotic dynamical systems, etc.

This sequence contains patterns but not repetitions.

Definition 3.1: Let $n \in \mathbb{N}$ be a natural number. The *Thue-Morse sequence* $\mathbf{s}_n = s_0 s_1 \cdots s_n$ is a word over the alphabet $\{0, 1\}$ defined as:

$$s_i = \begin{cases} 1 & \text{if } i \text{ has an odd number of 1s} \\ & \text{in its binary representation} \\ 0 & \text{otherwise,} \end{cases}$$

for $0 \leqslant i \leqslant n$. □

TABLE III. THE COMPRESSION RATIO $\mathsf{CR}_{jZIP}(\mathbf{s}_{2^k})$ FOR THUE-MORSE SEQUENCES

k	$\|seq_{2^k}\|$	$\mathsf{CR}_{jZIP}(seq_{2^k})$
5	32	34
8	256	4.625
10	1024	1.226
12	4096	0.328
14	16384	0.0932
15	32768	0.0542
16	65536	0.0322
17	131072	0.0208
18	262144	0.0151
19	524288	0.012
20	1048576	0.010
21	2097152	0.010
22	4194304	0.009

For example, we have

$$\mathbf{s}_{16} = (0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0).$$

It is clear that if $m, n \in \mathbb{N}$ and $m \leqslant n$, $\mathbf{s}_m$ is a prefix of $\mathbf{s}_n$. Thus, the successive Thue-Morse sequences define an infinite sequence.

An equivalent method for defining the Thue-Morse sequence is by starting with 0 and concatenating the complement of the sequence obtained so far. This procedure yields 0, then 01, 0110, 01101001, and so on. It is known (see [18], for example) that the Thue-Morse sequence is a cube-free sequence, that is, the sequence does not contain substrings of the form www.

We generated the Thue-Morse sequences and stored this sequence of 0s and 1s at the bit level. By using the zip compression utility from the `java.util.zip` package the compression ratios shown in Table III were obtained.

For small values of k, the sequence is incompressible due to the overhead produced by the compression process. As Table III and Figure 3 show, for k big enough ($2^k \geqslant 2000$) the sequence becomes compressible and the compression ratio reaches a low value (of less than 1%) for Thue-Morse sequences longer than $4,000,000$ characters. Since the Thue-Morse sequence $\mathbf{s}_{2^k}$ has equal number of 0s and 1s for any value of k and its compression ratio is well below the baseline compression ratio established for sequences of bits in Section II, we can conclude that even in the absence of repetitions, compression can be used for the detections of patterns.

In a series of experiments involving generative grammars we examined the compressibility of language fragments generated by these grammars. A generative grammar, or in short, a *grammar* is defined as a 4-tuple $G = (A_N, A_T, S, P)$, where A_N and A_T are non-empty, finite and disjoint sets referred to as the non-terminal and the terminal alphabet, respectively, $S \in A_N$ is the *initial symbol of the grammar* G, and P is a finite set of pairs of the form (α, β), where $\alpha \in (A_N \cup A_T)^+$ and $\beta \in (A_N \cup A_T)^*$. A pair $(\alpha, \beta) \in P$ is a production of the grammar G. Productions are used for rewriting words over $A_N \cup A_T$. Namely, if $\gamma, \delta \in (A_N \cup A_T)^*$, $\gamma = \gamma_1 \alpha \gamma_2$, and $\delta = \gamma_1 \beta \gamma_2$ for some production $(\alpha, \beta) \in P$, we write $\gamma \underset{G}{\Rightarrow} \delta$. The reflexive and transitive closure of the binary relation $\underset{G}{\Rightarrow}$ is denoted by "$\overset{*}{\underset{G}{\Rightarrow}}$". The language generated by G is the set

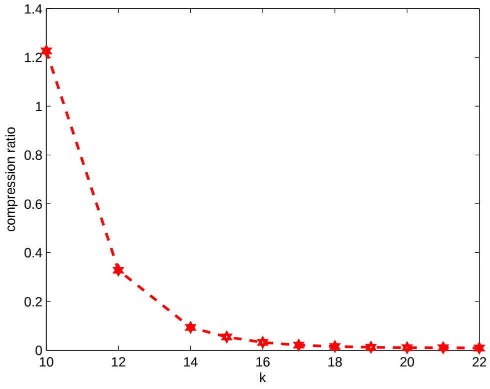


Figure 3. Compression Ratio Behavior of Thue-Morse Sequence

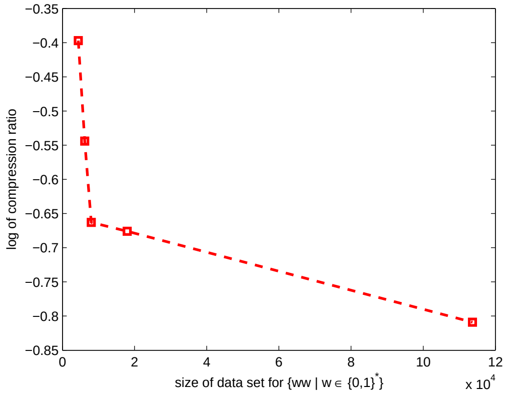


Figure 4. Compression Ratio Behavior of the language L_1

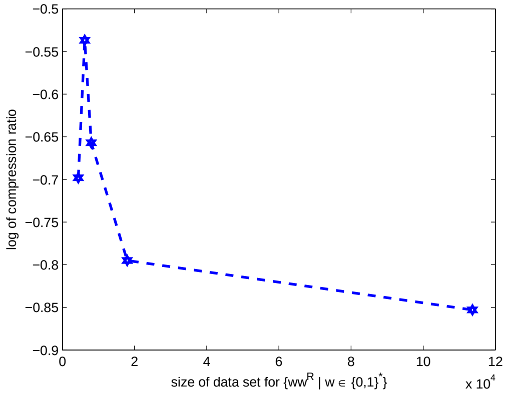


Figure 5. Compression Ratio Behavior of the language L_2

$L(G) = \{x \in A_T^* \mid S \overset{*}{\underset{G}{\Rightarrow}} \}$.

Grammars are used as generative devices that produce languages over their terminal alphabet. Chomsky's hierarchy (see [17] or [20]) defines four classes of grammars based on the complexity of their productions. In turn, these classes of grammars, define a strict hierarchy of classes of languages $\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$, where $\mathcal{L}_3$ is the class of regular languages, $\mathcal{L}_2$ is the class of context-free languages, $\mathcal{L}_1$ is the class of context-sensitive languages, and $\mathcal{L}_0$ is the class of recursively enumerable languages. It is worth noting that the classes $\mathcal{L}_3$ and $\mathcal{L}_2$ collapse on languages over one-symbol alphabet. In other words, if L is a language over an one-symbol alphabet, then $L \in \mathcal{L}_2$ implies $L \in \mathcal{L}_3$.

We evaluate the compressibility of a language L over an alphabet A by considering the increasing sequence of finite languages $\mathcal{S}(L) = (L_0, L_1, \ldots, L_n, \ldots)$, where L_n consists of the first n words of L in lexicographic order, computing the compression ratios $CR_{jZIP}(L_n)$, and examining the dependency of this ratio on n.

We examinine comparatively the compressibility of the languages $L_1 = \{ww \mid w \in \{0,1\}^*\}$ (a context-sensitive language) versus the compressibility of a similar language $L_2 = \{ww^R \mid w \in \{0,1\}^*\}$ (a context-free language) which has a simpler structure. Here, the word w^R is the *reversal* of the word w and is defined as $\lambda^R = \lambda$ and $(a_{i_1} \cdots a_{i_n})^R = a_{i_n} \cdots a_{i_1}$.

The results shown in Figures 4 and 5 show that L_2, the less complex language has a better (lower) compression ratio, and therefore, higher compressibility.

Similar results are obtained when comparing the compressibility of the context-sensitive languages L_{exp} and L_{prime} over the one-symbol alphabet $\{a\}$ defined by

$$\begin{array}{rcl} L_{exp} & = & \{a^{2^n} \mid n \in \mathbb{N}\}, \\ L_{prime} & = & \{a^p \mid p \text{ is a prime number}\}. \end{array}$$

The reference [17] (see Chapter 1, section 2) contains specific grammars developed for both languages. Namely, the grammar for L_{exp} has 6 productions, while the second grammar that generates L_{prime} has 42 productions. As expected, experiments summarized in Figure 6 show that the L_{exp} is more compressible than L_{prime} which has a rather complex generating process.

These results suggest that the compressibility of languages is related to the complexity of the generative process that produce them. This will be the object of further investigations.

IV. Random Insertion and Compression

For a matrix $M \in \{0,1\}^{u \times v}$ denote by $n_i(M)$ the number of entries of M that equal i, where $i \in \{0,1\}$. Clearly, we have $n_0(M) + n_1(M) = uv$.

For a random variable $\mathcal{V}$ which ranges over the set of matrices $\{0,1\}^{u \times v}$ let $\nu_i(\mathcal{V})$ be the random variable whose values equal the number of entries of $\mathcal{V}$ that equal i, where $i \in \{0,1\}$.

Let $A \in \{0,1\}^{p \times q}$ be a $0/1$ matrix and let

$$\mathcal{B} : \begin{pmatrix} B_1 & B_2 & \cdots & B_k \\ p_1 & p_2 & \cdots & p_k \end{pmatrix},$$

be a matrix-valued random variable where $B_j \in \mathbb{R}^{r \times s}$, $p_j \geqslant 0$ for $1 \leqslant j \leqslant k$, and $\sum_{j=1}^{k} p_j = 1$.

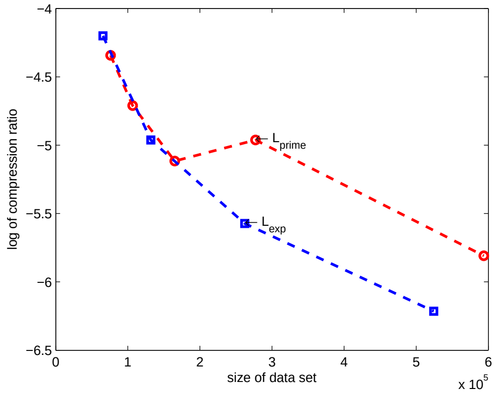


Figure 6. Compression Ratios of Languages L_{exp} and L_{prime}

Definition 4.1: The random variable $A \leftarrow \mathcal{B}$ obtained by the *insertion* of $\mathcal{B}$ into A is given by

$$A \otimes \mathcal{B} = \begin{pmatrix} a_{11}\mathcal{B} & \dots & a_{1n}\mathcal{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathcal{B} & \dots & a_{mn}\mathcal{B} \end{pmatrix} \in \mathbb{R}^{mr\times ns}$$

In other words, the entries of $A \leftarrow \mathcal{B}$ are obtained by substituting the block $a_{ij}B_\ell$ with the probability p_ℓ for a_{ij} in A.

Note that this operation is a probabilistic generalization of Kronecker's product for if

$$\mathcal{B} : \begin{pmatrix} B_1 \\ 1 \end{pmatrix},$$

then $A \leftarrow B$ has as its unique value the Kronecker product $A \otimes B$.

The expected number of 1s in the insertion $A \leftarrow \mathcal{B}$ is

$$E[\nu_1(A \leftarrow \mathcal{B})] = n_1(A) \sum_{j=1}^{k} n_1(B_j)p_j$$

When $n_1(B_1) = \cdots = n_1(B_k) = n$, we have $E[\nu_1(A \leftarrow \mathcal{B})] = n_1(A)n$.

In the experiment that involves insertion, we used a matrix-valued random variable such that $n_1(B_1) = \cdots = n_1(B_k) = n$. Thus, the variability of the values of $A \leftarrow \mathcal{B}$ is caused by the variability of the matrices $B_1, \dots, B_k$ which can be evaluated using the entropy of the distribution of $\mathcal{B}$,

$$\mathcal{H}(\mathcal{B}) = -\sum_{j=1}^{k} p_j \log_2 p_j.$$

We expect to obtain a strong positive correlation between the entropy of $\mathcal{B}$ and the degree of compression achieved on the file that represents the matrix $A \leftarrow \mathcal{B}$, and the experiments support this expectation.

In a first series of experiments, we worked with a matrix $A \in \{0,1\}^{106\times 106}$ and with a matrix-valued random variable

$$\mathcal{B} : \begin{pmatrix} B_1 & B_2 & B_3 \\ p_1 & p_2 & p_3 \end{pmatrix},$$

where $B_j \in \{0,1\}^{3\times 3}$, and $n_1(B_1) = n_1(B_2) = n_1(B_3) = 4$.

Several probability distributions were considered, as shown in Table IV. Values of $A \leftarrow \mathcal{B}$ had $106^2 * 3^2 = 101124$ entries.

In Table IV, we had 39% 1s and the baseline compression rate for a binary file with this ratio of 1s is 0.9775. We also computed the correlation between the CR_{jZIP} and the Shannon entropy of the probability distribution and obtained the value 0.98 for the insertion of a matrix-valued random variable having three values.

In Table V, we did the same experiment but with 4 different matrices of format 4 × 4. An even stronger correlation (0.99) was observed between CR_{jZIP} and the Shannon entropy of the probability distribution.

TABLE IV. INSERTION OF A THREE-VALUED RANDOM VARIABLE, ENTROPY AND COMPRESSION RATIOS

Probability distribution			Compression Ratio	Entropy
p_1	p_2	p_3		
0	1	0	0.33	0
1	0	0	0.33	0
0	0	1	0.33	0
0.9	0.1	0	0.51	0.46
0.8	0	0.2	0.61	0.72
0	0.3	0.7	0.7	0.88
0.2	0.2	0.6	0.77	1.37
0.6	0.2	0.2	0.74	1.37
0.15	0.35	0.5	0.78	1.44
0.49	0.25	0.26	0.77	1.5
0.33	0.33	0.34	0.79	1.58

TABLE V. INSERTION OF A FOUR-VALUED RANDOM VARIABLE, ENTROPY AND COMPRESSION RATIOS

Probability distribution				Compression Ratio	Entropy
p_1	p_2	p_3	p_4		
0	1	0	0	0.23	0
0.4	0	0.2	0.4	0.53	1.52
0.45	0.12	0.22	0.21	0.61	1.83
0.3	0.1	0.2	0.4	0.65	1.84
0.2	0.2	0.2	0.4	0.69	1.92
0.25	0.25	0.25	0.25	0.69	2

The relationship between the compression ratio CR_{jZIP} and the Shannon entropy of the probability distribution of the inserted random variable is shown in Figure 7 for both experiments.

This experiment reconfirms that data that contains patterns can be better compressed than randomly generated files and that the compressibility is less pronounced when the diversity of these patterns increases.

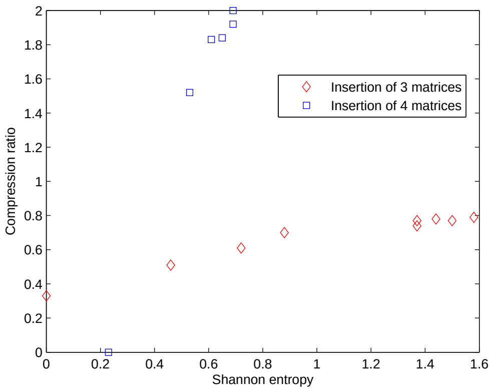


Figure 7. Evolution of CR_{jZIP} and Shannon Entropy for Insertions

Next, we examine the compressibility of binary square matrices and its relationship with the distribution of principal submatrices. An $m \times m$ *principal submatrix* of a matrix $A \in \mathbb{R}^{n\times n}$ is the matrix $A[I]$ defined by a non-empty m-element subset I of the set $\{1,\ldots,n\}$ and is obtained by selecting entries of A of the form a_{ij}, where $i,j \in I$. We mention that the principal submatrices of the adjacency matrix of a graph correspond to the adjacency matrices of the subgraphs of that graph. The patterns in a graph are captured in the form of frequent isomorphic subgraphs.

A binary square matrix is compressed by first vectorizing the matrix and then compressing the resulting binary sequence. There is a strong correlation between the compression ratio of the adjacency matrix of a graph and the frequencies of the occurrences of isomorphic subgraphs of it. Specifically, the lower the compression ratio is, the higher are the frequencies of isomorphic subgraphs and hence the worthier is the graph for being mined.

Let $\mathcal{G}_n$ be an undirected graph having $\{v_1,\ldots,v_n\}$ as its set of nodes. The adjacency matrix of $\mathcal{G}_n$, $\mathbf{A}_{\mathcal{G}_n} \in \{0,1\}^{n\times n}$ is defined as

$$(\mathbf{A}_{\mathcal{G}_n})_{ij} = \begin{cases} 1 & \text{if there is an edge between } v_i \text{ and } v_j \text{ in } \mathcal{G}_n \\ 0 & \text{otherwise.} \end{cases}$$

We denote with $\mathsf{CR}_C(\mathbf{A}_{\mathcal{G}_n})$ the compression ratio of the adjacency matrix of graph $\mathcal{G}_n$ obtained by applying the compression algorithm C.

Let $S = \{i_1,\ldots,i_k\}$ be a subset of $\{1,\ldots,n\}$. The principal submatrix $\mathbf{A}_{\mathcal{G}_n}[S]$ is the adjacency matrix of the subgraph of $\mathcal{G}_n$ which consists of the nodes with indices in S along with those edges that connect these nodes. We denote by $\mathcal{P}_n(k)$ the collection of all subsets of $\{1,2,\ldots,n\}$ of size k where $2 \leqslant k \leqslant n$. We have $|\mathcal{P}_n(k)| = \binom{n}{k}$.

Let $(\mathbf{A}_1^k,\ldots,\mathbf{A}_{\ell_k}^k)$ be an enumeration of possible adjacency matrices of graphs with k nodes where $\ell_k = 2^{\frac{k(k-1)}{2}}$. We define the finite probability distribution

$$P(\mathcal{G}_n,k) = \left(\frac{\mathsf{n}_1^k(\mathcal{G}_n)}{|\mathcal{P}_n(k)|},\ldots,\frac{\mathsf{n}_{\ell_k}^k(\mathcal{G}_n)}{|\mathcal{P}_n(k)|}\right),$$

where $\mathsf{n}_i^k(\mathcal{G}_n)$ for $1 \leqslant i \leqslant \ell_k$ is the number of subgraphs of $\mathcal{G}_n$ with adjacency matrix $\mathbf{A}_i^k$. The Shannon entropy of this probability distribution is:

$$\mathcal{H}_P(\mathcal{G}_n,k) = -\sum_{i=1}^{\ell_k} \frac{\mathsf{n}_i^k(\mathcal{G}_n)}{|\mathcal{P}_n(k)|}\log_2\frac{\mathsf{n}_i^k(\mathcal{G}_n)}{|\mathcal{P}_n(k)|}.$$

If $\mathcal{H}_P(\mathcal{G}_n,k)$ is low, there are to be fewer and larger sets of isomorphic subgraphs of $\mathcal{G}_n$ of size k. In other words, small values of $\mathcal{H}_P(\mathcal{G}_n,k)$ for various values of k suggest that the graph $\mathcal{G}_n$ contains repeated patterns and is susceptible to produce interesting results. Note that although two isomorphic subgraphs do not necessarily have the same adjacency matrix, the number $\mathcal{H}_P(\mathcal{G}_n,k)$ is a good indicator of the diversity of isomorphic subgraphs and hence of the frequency subgraph patterns.

We evaluated the correlation between $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ and $\mathcal{H}_P(\mathcal{G}_n,k)$ for different values of k.

As expected, the compression ratio of the adjacency matrix and the distribution entropy of graphs are roughly the same for isomorphic graphs, so both numbers are characteristic for an isomorphism type. If ϕ is a permutation of the vertices of $\mathcal{G}_n$, the adjacency matrix of the graph $\mathcal{G}_n^\phi$ obtained by applying the permutation is defined by $\mathbf{A}_{\mathcal{G}_n^\phi}$ is given by

$$\mathbf{A}_{\mathcal{G}_n^\phi} = P_\phi \mathbf{A}_{\mathcal{G}_n} P_\phi^{-1}.$$

We compute the adjacency matrix $\mathbf{A}_{\mathcal{G}_n^\phi}$, the entropy $\mathcal{H}_P(\mathcal{G}_n^\phi,k)$, and the compression ratio $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n^\phi})$ for several values of k and permutations.

Graphs with $n = 60$ nodes and various number of edges ranging from 5 to 1765 were randomly generated. For each generated graph, we randomly produced twenty permutations of its set of nodes and computed $\mathcal{H}_P(\mathcal{G}_n^\phi,k)$ and $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n^\phi})$.

Finally, for each graph we calculated the ratio of standard deviation over average for the computed compression ratios, followed by the same computation for distribution entropies.

The results of this experiment are shown in Figures 8 and 9 against the number of edges. As it can be seen, the deviation over mean of the compression ratios for $n = 60$ does not exceed the number 0.05. Also, the deviation over average of the distribution entropies for various values of k do not exceed 0.006. In particular, the deviation of the distribution entropy for the graphs of 100 to 1500 edges falls below 0.001, which allows us to conclude that the deviations of both compression ratio and distribution entropy with respect to isomorphisms are negligible.

For each $k \in \{3,4,5\}$, we generated randomly 560 graphs having 60 vertices and sets of edges whose size were varying from 10 to 1760. Then, the numbers $\mathcal{H}_P(\mathcal{G}_n,k)$ and $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ were computed. Figure 10 captures the results of the experiment. Each plot contains two curves. The first curve represents the changes in average $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ for forty randomly generated graphs of equal number of edges. The second curve represents the variation of the average $\mathcal{H}_P(\mathcal{G}_n,k)$ for the same forty graphs. The trends of these two curves are very similar for different values of k.

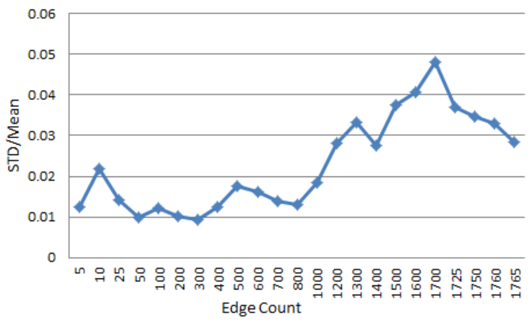


Figure 8. Standard deviation vs. average of the $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ for a number of different permutations of nodes for the same graph. The horizontal axis is labelled with the number of edges of the graph.

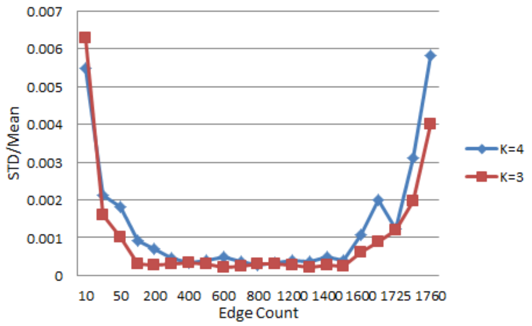


Figure 9. Standard deviation vs. average of the $\mathcal{H}_P(\mathcal{G}_n, k)$ of a number of different permutations of nodes for the same graph. The horizontal axis is labelled with the number of edges of the graph. Each curve corresponds to one value of k.

Table VI contains the correlation between $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ and $\mathcal{H}_P(\mathcal{G}_n, k)$ calculated for the 560 randomly generated graphs for each value of k.

TABLE VI. CORRELATIONS BETWEEN $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ AND $\mathcal{H}_P(\mathcal{G}_n, k)$

k	Correlation
3	0.92073175
4	0.920952812
5	0.919256573

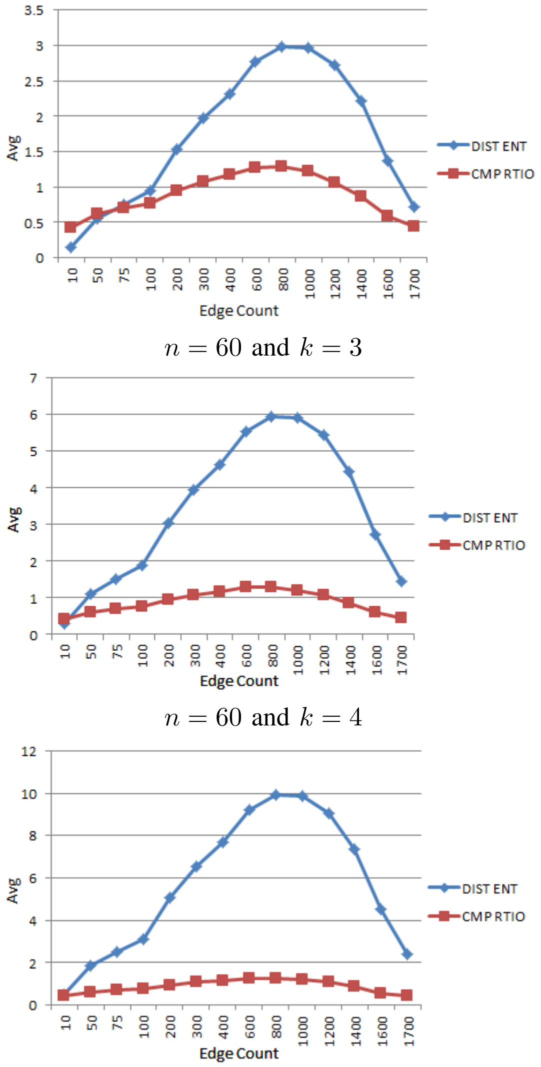
V. FREQUENT ITEMS SETS AND COMPRESSION RATIO

A market basket data set consists of a multiset $\mathcal{T}$ of *transactions*. Each transaction T is a subset of a set of items $I = \{i_1, \ldots, i_N\}$. The multiplicity of a transaction T in the multiset $\mathcal{T}$ is denoted by $m(T)$.

A transaction is described by its characteristic N-tuple $t = (t_1, \ldots, t_N)$, where

$$t_k = \begin{cases} 1 & \text{if } i_k \in T. \\ 0 & \text{otherwise,} \end{cases}$$

for $1 \leqslant k \leqslant N$. The length of a transaction T is $|T| = \sum_{k=1}^{N} t_k$, while the average size of transactions is $\frac{\sum\{|T| \mid T \text{ in } \mathcal{T}\}}{|\mathcal{T}|}$.



$n = 60$ and $k = 3$

$n = 60$ and $k = 4$

$n = 60$ and $k = 5$

Figure 10. Plots of average $\mathsf{CR}_{jZIP}(\mathbf{A}_{\mathcal{G}_n})$ (CMP RTIO) and average $\mathcal{H}_P(\mathcal{G}_n, k)$ (DIST ENT) for randomly generated graphs $\mathcal{G}_n$ of equal number of edges with respect to the number of edges.

The support of a set of items K of the data set $\mathcal{T}$ is the number

$$\mathsf{supp}(K) = \frac{|\{T \in \mathcal{T} \mid K \subseteq T\}|}{|\mathcal{T}|}.$$

The set of items K is s-frequent if $\mathsf{supp}(K) > s$.

The study of market basket data sets is concerned with the identification of association rules. A pair of item sets (X, Y) is an association rule, denoted by $X \rightarrow Y$. Its support, $\mathsf{supp}(X \rightarrow Y)$ equals $\mathsf{supp}(X)$ and its confidence $\mathsf{conf}(X \rightarrow Y)$ is defined as

$$\mathsf{conf}(X \rightarrow Y) = \frac{\mathsf{supp}(X \cup Y)}{\mathsf{supp}(X)}.$$

Using the artificial transaction ARMiner generator described in [5], we created a basket data set.

Transactions are represented by sequences of bits $(t_1, \cdots, t_N)$. The multiset $\mathcal{T}$ of M transactions was represented as a binary string of length MN obtained by concatenating the strings that represent transactions.

We generated files with 1000 transactions, with 100 items available in the basket, adding up to 100K bits.

For data sets having the same number of items and transactions, the efficiency of the compression increases when the number of patterns is lower (causing more repetitions). In an experiment with an average size of a frequent item set equal to 10, the average size of a transaction equal to 15, and the number of frequent item sets varying in the set $\{5, 10, 20, 30, 50, 75, 100, 200, 500, 1000\}$, the compression ratio had a significant variation ranging between 0.20 and 0.75, as shown in Table VII. The correlation between the number of patterns and the compression ratio was 0.544. Although the frequency of 1s and baseline compression ratio were roughly constant (at 0.75), the number of patterns and compression ratio were correlated.

TABLE VII. NUMBER OF ASSOCIATION RULES AT 0.05 SUPPORT LEVEL AND 0.9 CONFIDENCE

Number of Patterns	Frequency of 1s	Baseline compression	Compr. ratio	Number of rules
5	16%	0.75	0.20	9,128,841
10	17%	0.73	0.34	4,539,650
20	17%	0.73	0.52	2,233,049
30	17%	0.76	0.58	106,378
50	19%	0.75	0.65	2,910,071
75	18%	0.75	0.67	289,987
100	18%	0.75	0.67	378,455
200	18%	0.75	0.70	163
500	18%	0.75	0.735	51
1000	18%	0.75	0.75	3

Further, there was a strong negative correlation (-0.92) between the compression ratio and the number of association rules indicating that market basket data sets that satisfy many association rules are very compressible.

VI. CONCLUDING REMARKS

Compression ratio of a file can be computed fast and easy, and in many cases offers a cheap way of predicting the existence of embedded patterns in data. Thus, it becomes possible to obtain an approximative estimation of the usefulness of an in-depth exploration of a data set using more sophisticated and expensive algorithms.

The presence of patterns in strings leads to a high degree of compression (that is, to low compression ratios). Thus, a low compression ratio for a file indicates that the mining process may produce interesting results. Compressibility however, does not guarantee that a sequence contains repetitions. Strings that are free of repetitions but contain patterns can display a high degree of compressibility as shown by the well-known Thue-Morse binary string.

The use of compression as a measure of minability is illustrated on a variety of paradigms: graph data, market basket data, etc. Compression has been applied in bioinformatics as a tool for reducing the size of immense data sets that are generated in the genomic studies. Furthermore, specialized algorithms were developed that mine data in compressed form [12].

Our current work shows that identifying compressible areas of human DNA by comparing the compressibility of certain genomic regions is a useful tool for detecting areas where the gene replication mechanisms are disturbed (a phenomenon that occurs in certain genetically based diseases).

AKNOWLEDGEMENTS

This work was supported by a Science and Technology grant from the President of the University of Massachusetts.

The authors appreciate the remarks of the anonymous reviewers who contributed to the improvement of this paper.

REFERENCES

[1] D. Simovici, D. Pletea, and S. Baraty. Evaluating data minability through compression - an experimental study. In *Proceedings of DATA ANALYTICS*, pages 97–102. Think Mind Digital Library, 2012.

[2] J.-P. Allouche and J. Shallit. The ubiquitous Prouhet-Thue-Morse sequence. In *Sequences and their Applications*, pages 1–16. Springer London, 1999.

[3] B. J. L. Campana and E. J. Keogh. A compression based distance measure for texture. In *SDM*, pages 850–861, 2010.

[4] R. Cilibrasi and P. M. B. Vitànyi. Clustering by compression. *IEEE Transactions on Information Theory*, 51:1523–1545, 2005.

[5] L. Cristofor. The ARMiner project, 2000, Univ. of Massachusetts Boston. http://www.cs.umb.edu/ ∼laur/ARMiner.

[6] I. Csiszár and J. Körner. Information Theory - Coding Theorems for Discrete Memoryless Systems. (second edition). Cambridge University Press, Cambridge, UK, 2011.

[7] L. Jin, K. L. Parthasarathy, T. Kuyel, R. L. Geiger, and D. Chen. High-performance adc linearity test using low-precision signals in non-stationary environments. In *Proceedings 2005 IEEE International Test Conference*, pages 1182–1191, 2005.

[8] E. Keogh, S. Lonardi, and C. A. Ratanamahatana. Towards parameter-free data mining. In *Proc. 10th ACM SIGKDD Intnl Conf. Knowledge Discovery and Data Mining*, pages 206–215. ACM Press, 2004.

[9] E. Keogh, S. Lonardi, C. A. Ratanamahatana, L. Wei, S. Lee, and J. Handley. Compression-based data mining of sequential data. *Data Mining and Knowledge Discovery*, 14:99–129, 2007.

[10] E. J. Keogh, L. Keogh, and J. Handley. Compression-based data mining. In *Encyclopedia of Data Warehousing and Mining*, pages 278–285. 2009.

[11] T. Kuyel D. Chen L. Jin, K. Parthasarathy and R. Geiger. Accurate testing of analog-to-digital converters using low linearity signals with stimulus error identification and removal. *IEEE Transactions on Instrumentation and Measurement*, 54:1188–1199, 2005.

[12] P. R. Loh, M. Baym, and B. Berger. Compressive genomics. *Nature Biotechnology*, 30:627–630, 2012.

[13] D. J. C. McKay. Information Theory, Inference, and Learning Algorithms. Cambridge University Press, Cambridge, UK, 2004.

[14] H. Mannila. Theoretical frameworks for data mining. *SIGKDD Exploration*, 1:30–32, 2000.

[15] M. Morse and G. A. Hedlund. Unending chess, symbolic dynamics, and a problem in semigroups. *Duke Mathematical Journal*, 11:1–7, 1944.

[16] L. Youran R. Ricklund, S. Mattias. The Thue-Morse aperiodic crystal, a link between the Fibonacci quasicrystal and the periodic crystal. *International Journal of Modern Physics B*, 1:121–132, 1987.

[17] A. Salomaa. *Formal Languages*. Academic Press, New York, 1973.

[18] A. Salomaa. *Jewels of Formal Language Theory*. Computer Science Press, Rockville, Maryland, 1981.

[19] A. Siebes, J. Vreeken, and M. van Leeuwen. Items sets that compress. In *Proceedings of SIAM International Conference on Data Mining*, pages 393–404. SIAM, 2006.

[20] D. A. Simovici and R. L. Tenney. *Formal Language Theory with Applications*. World Scientific, Singapore, 1999.

[21] D. A. Simovici and C. Djeraba. *Mathematical Tools for Data Mining*. (second edition) Springer, London, 2014.

[22] J. Vreeken, M. van Leeuwen, and A. Siebes. KRIMP: mining items that compress. *Data Mining and Knowledge Discovery*, 23:169–214, 2011.

[23] L. Wei, J. Handley, N. Martin, T. Sun, and E. J. Keogh. Clustering workflow requirements using compression dissimilarity measure. In *ICDM Workshops*, pages 50–54, 2006.

[24] T. Welch. A technique for high performance data compression. *IEEE Computer*, 17:8–19, 1984.

[25] R. Yarlagadda and J. Hershey. Counter synchronization using the Thue-Morse sequence and psk. *IEEE Transactions on Communications*, 32:947–977, 1984.