SAND2019-11996C

REORDERING GENOMIC SEQUENCES FOR ENHANCED CLASSIFICATION VIA COMPRESSION ANALYTICS

Christina Ting, Renee Gooding, Richard Field, Jr., and Jacob Caswell

Sandia National Laboratories
Albuquerque, NM 87123

ABSTRACT

The full implications of sharing genomic information are still largely unknown. Understanding what attributes can be inferred from available information is therefore a critical part of genomic privacy and security. We show that compression analytics are successful at classifying, or inferring, unknown attributes of genomic sequences without the need for a predefined feature set and with very little training data. Compression analytics perform best when predictable elements within a sequence are local; however, long range dependencies are ubiquitous in the human genome. We therefore consider a variety of schemes to reorder genomic sequences so as to localize predictable elements and improve the performance of compression analytics. Compression analytics on both native and reordered sequences are shown to outperform more traditional, feature-based machine learning approaches.

Index Terms— Machine learning; Genomic privacy; Genomic Security, Genomic inference; Compression; Louvain

1. INTRODUCTION AND PREVIOUS WORK

Personalized genomics is transforming modern medical therapies [1]. Therefore, there is a strong motivation for the collection, sharing, and analysis of genomic data. At the same time, as our understanding of genomic data increases, so too does the risk of revealing interdependent information that may be considered sensitive due to known privacy concerns or—perhaps more significantly—because the full implications of sharing that information are still largely unknown. Understanding what attributes can be inferred from available information is therefore a critical part of genomic privacy and security.

Several recent surveys provide good overviews on the issues of genomic privacy and security [2, 3]. In terms of privacy, the most commonly discussed risks include re-identification, attribute inference, and revelation of familial relationships. Although there exist methods in place to protect against each of these privacy breaches, including data access controls, data anonymization, and cryptographic techniques, both real and demonstrated examples of successful attacks have revealed weaknesses in existing privacy protection methods. For example, widely used data anonymization techniques that explicitly remove identifying meta-data still allow for an adversary to make *linking* attacks using available background information from, for example, genealogical databases [4, 5, 6].

A second type of attack is based on inferences that can be made directly from the rich background of publicly-available genomic information [7, 8, 9]. For example, pairwise correlations, or *linkage disequilibrium* (LD), between genomic variants can be used to infer unpublished genomic data from published genomic data. These previous demonstrations of genomic inference attacks have primarily exploited first and second order statistics of genomic variants. More recently, demonstrations using high-order statistics have proven much more powerful [10, 11]. In these studies, the probability of a genomic variant occurring in the genome is conditioned on a contiguous set of the preceding k variants observed. This model is called a k-th order Markov chain; although generally successful, there are serious limitations when applied to genomic data. In particular, it is well known that long-range dependencies far outside (both before and after) the preceding k variants exist, which brings into question the effectiveness of such a model.

Machine learning (ML) algorithms provide an alternative approach to genomic inference [12, 13]. Standard ML algorithms seek to identify—from a *predetermined* feature set—the features most useful for defining a decision boundary for a given prediction problem. While ML algorithms enable the efficient use of high-dimensional feature sets without explicit knowledge of the relationships among the features, the best features for a given prediction problem are generally not known *a priori*. For genomic data, a standard feature set is the normalized count of all subsequences of length k, that is, the k-mer distribution, which again assumes that contiguous sequences of k variants contain the predictive elements.

Herein, we develop an alternative, *compression-based* approach that capitalizes on the ability of compression algo-

This work is supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND2019-xxxx.

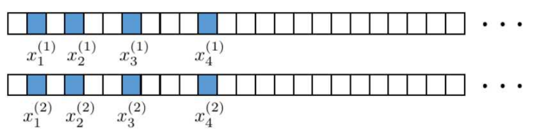

Fig. 1: Schematic of an individual's genome with the SNVs highlighted.

rithms to operate on raw data and discover the predictable elements. This approach does not require a predetermined feature set and has been shown to outperform genomic analyses based on k-mer distributions [14]. However, compression algorithms perform best when predictable elements within a sequence are *local* (we define what we mean by local in Section 3.3). In many datasets, predictable elements are indeed local and compression algorithms have shown the ability to identify subtle features in complex datasets that are useful for, among other prediction tasks, authorship and deception detection [15]. However, in genomic sequences, where dependencies can be long-ranged and non-sequential, this constraint is not necessarily satisfied. Therefore, we also propose multiple reordering schemes to localize predictable elements, thereby improving the performance of compression analytics on genomic sequences.

Major contributions of our work include the development of a general compression-based approach that can be applied without expert-derived feature engineering, and that automatically identifies the most predictable elements in data, particularly non-sequential data, for a given prediction task. Furthermore, unlike other approaches that also try to discover the features [16], compression algorithms are successful even with very little training data. Section 2 provides background on the necessary genomics foundations for understanding this work. Sections 3–6 describe the method, data, results, and conclusions.

2. GENOMICS FOUNDATIONS

DNA is a molecular sequence composed of base units referred to as nucleotides, and each nucleotide contains one of four nucleobases: cytosine (`C`), guanine (`G`), adenine (`A`), or thymine (`T`). We can visualize an individual's *genome* as a pair (one from each parent) of DNA sequences of length $N \approx 3.3$ billion nucleotides. Most of the genome is identical across the human population, but there are still millions of positions where two or more variants, termed *alleles*, occur. These positions are referred to as Single Nucleotide Variants (SNVs) and carry privacy-sensitive information [10, 17]. Understanding the information that can be inferred from SNVs is therefore a critical part of genomic privacy and security.

The genomic information for an individual i can be described by the corresponding sequence of SNVs, denoted by

$$s_i^{(\alpha)} = x_{i,1}^{(\alpha)} x_{i,2}^{(\alpha)} \cdots x_{i,M}^{(\alpha)}, \tag{1}$$

where $\alpha = 1, 2$ represents a copy from each parent, and $M \ll N$ denotes the number of SNVs; see Fig. 1 for a schematic. Most SNVs are bi-allelic, where the allele occurring most (least) often is referred to as the *major* (*minor*) allele; we limit our analysis to bi-allelic SNVs. The sequence of SNVs can thus be represented as a sequence of binary random variables, which have some useful properties that we discuss next.

Let $X_k \in \{0, 1\}$ be a random variable that represents the major or minor allele at SNV position k. The probabilities $\Pr(X_k = 0)$ and $\Pr(X_k = 1)$ then denote the major and minor *allele frequencies* of SNV k. Because each SNV is assumed bi-allelic, the joint probability function between two SNVs k and l takes only four values that can be expressed in the following manner [17]:

$$\begin{aligned}
\Pr(X_k = 0, X_l = 0) &= (1 - p_k)\,(1 - p_l) + \ell_{k,l} \\
\Pr(X_k = 0, X_l = 1) &= (1 - p_k)\,p_l - \ell_{k,l} \\
\Pr(X_k = 1, X_l = 0) &= p_k\,(1 - p_l) - \ell_{k,l} \\
\Pr(X_k = 1, X_l = 1) &= p_k\,p_l + \ell_{k,l}
\end{aligned}$$

where, to simplify notation, $1 - p_k$ and p_k denote the major and minor allele frequencies of SNV k. The term $\ell_{k,l}$ that is present in each line above is referred to as the *linkage disequilibrium* (LD) between SNVs k and l, and quantifies the degree of statistical dependence between them, where $\ell_{k,l} = 0$ if and only if X_k and X_l are independent. We note that the LD depends on the allele frequency; a normalized version independent of allele frequency is usually defined (assuming $p_k, p_l \neq 0$):

$$\widehat{\ell}_{k,l} = \ell_{k,l} / \sqrt{(1 - p_k)\,p_k\,(1 - p_l)\,p_l}. \tag{2}$$

Each normalized LD satisfies $-1 \leq \widehat{\ell}_{k,l} \leq 1$. In Section 3.3, we will apply the allele frequencies and normalized LD values to define various reordering schemes for improved compression analytics.

3. METHODS

In their standard use cases, compression algorithms operate on individual sequences, which means that the underlying statistics have to be derived not only from the sequence itself, but from the components of the sequence leading up to the current token that is being compressed. For compression analytics, each sequence is compressed using the statistics of training data, which forms a *model*. The size (in bits) of the compressed sequence with respect to a given model is used to determine how well the model describes, or *predicts*, the data, where fewer bits means better prediction.

Arithmetic coding with prediction by partial matching (PPM) is a compression algorithm well-suited for analytics [18, 19]. The key component here is that the better the prediction, that is, the larger the probability, for some token occurring in a sequence s, the better the compression for s. There are a variety of ways to compute the probabilities of some token occurring. PPM is a method for computing the probabilities that takes into account the previous n tokens, referred to as a context of order n. Taking into account the context can result in larger probabilities and better compression. Importantly, PPM adaptively chooses the context order n, up to a user–defined maximum (we use $n = 10$), and therefore allows an analysis of the raw data at multiple scales. In the following section, we formalize the application of compression for analytics.

3.1. Compression for classification

Recall that the genomic information for an individual i is described by a pair of SNV sequences defined by Eq. (1). To encode this information into a single sequence, we first let each $x_{i,k}^{(1)}$, $x_{i,k}^{(2)}$ be one of the four nucleobases `A`, `T`, `G`, or `C`. Each of the possible 10 *combinations* for the pair $x_{i,k}^{(1)} x_{i,k}^{(2)}$ is then encoded by a single token $\chi_{i,k}$, where each $\chi_{i,k}$ is one of `AA`, `AC`/ `CA`, . . ., `TT`. Thus, we can represent a sequence of SNVs for individual i as

$$s_i = \chi_{i,1}\chi_{i,2}\cdots\chi_{i,m}, \tag{3}$$

where we use $m \ll M$ to denote that, in general, we don't have any individual's full genomic information. We note that the binary random variables discussed in Section 2 could be used as an alternative encoding.

Now suppose we have labeled SNV sequences for q individuals, each from exactly one of $r \leq q$ distinct populations. We can assemble the SNV data into a training set, denoted by $\mathcal{S} = \{s_1, s_2, \ldots, s_q\}$, where each s_i has corresponding population label y_i. Next let $\mathcal{S}_j \subseteq \mathcal{S}$ be the subset of SNV sequences for individuals that are a member of population j, We can *train* a PPM model μ_j on subset $\mathcal{S}_j$; refer to Fig. 2. The PPM model constitutes a set of conditional probability tables, one for each observed context in $\mathcal{S}_j$ of order n less than the max order. A table for a given context contains all the predictions for that context [18]. By repeating this training process for each population, we obtain a collection of PPM models.

Next suppose we have an unlabeled SNV sequence s_i for an individual from an unknown population. Let $c(s_i; \mu_j)$ denote the score corresponding to the compressed size of sequence s_i using PPM model μ_j. This process, which is repeated for each of the r PPM models, corresponds to the testing process and is depicted by the right hand side of Fig. 2.

Finally, we *predict* the population label of individual i as the population whose PPM model compresses s_i in the fewest

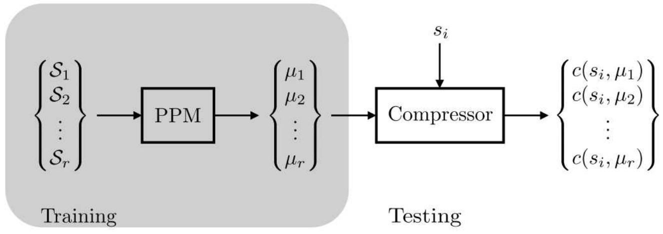

Fig. 2: Diagram of the classification by compression process.

number of bits, that is,

$$Y_i = \underset{j=1,\ldots,r}{\arg\min}\, c(s_i; \mu_j), \tag{4}$$

where we use Y_i to represent both the population index and the population. The intuition behind this scoring function is that the PPM model that has been trained on sequences most similar to s_i will, in general, contain the best prediction (larger conditional probabilities at higher context orders) for the next token in s_i, leading to better compression in fewer bits.

3.2. Slice compression (SC) for classification

We next present a modified version of standard compression based on the slice compression (SC) algorithm [20]. In the normal use of compression analytics, the compression algorithm is applied to the entire sequence s at once, producing a global score in the form of the number of bits required to describe s in its compressed state. In contrast, the objective of SC is to identify local behavior within a sequence by considering the compression of subsequences, or *slices*, of s.

Similar to Section 3.1, we train a set of PPM models μ_j for each subset $\mathcal{S}_j$; refer again to Fig. 2. However, instead of considering $c(s_i; \mu_j)$ for individual i, we define a sequence of SC scores as

$$z_i(k; \mu_j) := \frac{1}{w}\, c(\sigma_i(k; w); \mu_j), \tag{5}$$

where $\sigma_i(k; w)$ is a slice of s_i beginning at index k and with width w. By following the evolution of the SC scores across the sequence, we can track the local behavior of the sequence up to the resolution of the slice width, where we use $w = 100$ as the window width. The SC testing process can similarly be depicted by Fig. 2 if we replace s_i and $c(s_i; \mu_j)$ with $\sigma_i(k; w)$ and $z_i(k; \mu_j)$, respectively.

Let $\bar{z}_i(\mu_j)$ denote the average of $z_i(k; \mu_j)$ over its length; we can then apply Eq. (4) with $c(s_i; \mu_j)$ replaced by $\bar{z}_i(\mu_j)$ to classify the population of individual i using the SC scores with respect to the different population models.

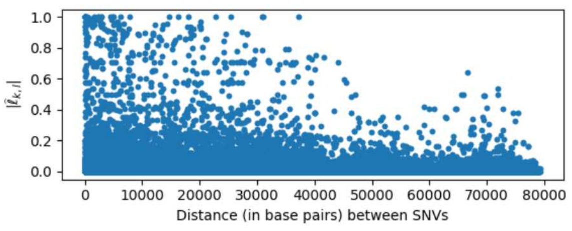

Fig. 3: Absolute value of the normalized LD as a function of the distance between SNVs.

3.3. Reordering schemes

As mentioned in Section 1, compression algorithms rely on predictable elements within a sequence to be *local*. More specifically, they require that predictable elements are completely contained within subsequences of length less than the maximum context order of the PPM algorithm. However, genomic sequences can exhibit dependencies over lengths larger than any reasonable context order. For example, Fig. 3 demonstrates that two SNVs separated by thousands of basepairs can be highly interdependent ($|\widehat{\ell}_{k,l}|$ near one). Therefore, we present multiple reordering schemes for localizing predictable elements within each sequence, thereby improving the performance of compression analytics.

We define a reordering function $\alpha(k) = \hat{k}$, where k denotes the index of a SNV in its native order, and $\hat{k}$ denotes the index of the same SNV within the reordered sequence. The reordering functions α we consider will utilize the allele frequencies and normalized LD; see Section 2. For the latter, we define an $m \times m$ adjacency matrix $\mathcal{L} = \{\widehat{\ell}_{k,l}\}$, where $\widehat{\ell}_{k,l}$ is the normalized LD defined by Eq. (2). In order to limit the reordering function to significant pairwise dependencies, we set all normalized LD values whose absolute value is less than some tolerance ℓ^* to zero. The corresponding matrix is denoted by $\mathcal{L}^*$, which can be viewed as an adjacency matrix for a weighted graph.

The reordering functions α we consider are as follows:

1. Reorder by allele frequency. Recall that p_k denotes the minor allele frequency of SNV k. For this case, α is defined such that $p_{\hat{k}} \leq p_{\hat{k}+1}$, i.e., the reordered sequence has the minor allele frequency in ascending order.
2. Reorder by community assignment of $\mathcal{L}^*$. Louvain is an approach for identifying communities in large scale networks based on a greedy optimization of the graph modularity [21]. Let L_k denote the community assigned to SNV k by Louvain. For this case, α is defined such that $L_{\hat{k}} \leq L_{\hat{k}+1}$, i.e., the reordered sequence is sorted by community assignment in ascending order.
3. Reorder to minimize the matrix bandwidth of $\mathcal{L}^*$. The bandwidth of $\mathcal{L}^*$ is defined as the number b such that $\widehat{\ell}_{k,l} = 0$ if $|k - l| > b$. The Reverse Cuthill-McKee (RCM) algorithm [22] rearranges the nodes of a graph to reduce the bandwidth of $\mathcal{L}^*$. In this case, $\alpha(k)$ is the row/column of SNV k in $\mathcal{L}^*$ after application of RCM.

We use symbol $\hat{s}$ to denote SNV sequence s in its reordered configuration. Note that the reordering is non-unique, e.g., two or more SNVs may have the same minor allele frequency or be assigned to the same community.

To incorporate reordering into compression analytics, the approach discussed in Section 3.1 is modified as follows. During the training phase, we apply one of the proposed reordering schemes to obtain $\hat{\mu}_j$, the PPM model trained on $\hat{\mathcal{S}}_j$, the set of reordered sequences belonging to population j. In general, each population will have a different reordering scheme. Now suppose we have an unlabeled SNV sequence s_i. The population label for s_i is predicted using Eq. (4) with $c(s_i; \mu_j)$ replaced by $c(\hat{s}_i; \hat{\mu}_j)$. Here each $\hat{s}_i$ is reordered according to the same reordering used to obtain the corresponding $\hat{\mu}_j$. An analogous modification is used to incorporate the reordering scheme into the slice compression (SC) technique discussed in Section 3.2.

4. DATA

The data used in this study are obtained from the 1000 Genomes Project [23]. The stated goal of the 1000 Genomes Project is to identify "genetic variants with frequencies of at least 1% in the populations studied". The final phase (phase 3) of this project provides data for 2,504 individuals, together with associated metadata including 26 population groups, 5 superpopulations, and some familial relationships, if available. We focus our analysis on $r = 3$ population groups: (1) Chinese Dai in Xishuangbanna Prefecture (CDX); (2) Finnish in Finland (FIN); and (3) Mexican ancestry in Los Angeles, California (MXL). There are 93, 99, and 64 individuals from each of the CDX, FIN, and MXL population groups, respectively. As mentioned previously, we filter our data such that only bi-allellic SNVs are considered. For training purposes, we further require that the minor allele occur at least once in our dataset. Finally we note that there is nothing particular about the 3 populations considered, only that each population belongs to a different superpopulation.

5. RESULTS

We seek to understand the impact of reordering on the locality of dependent SNVs by visualizing the linkage disequilibrium (LD) matrices (Section 5.1) and quantifying the model compressibility (Section 5.2) of both the native and reordered SNV sequences. Then, in Section 5.3, we show that reordering to localize pairwise dependent SNVs improves the ability of compression analytics to predict attributes that may be associated with privacy concerns, where here we use the population attribute provided by our dataset.

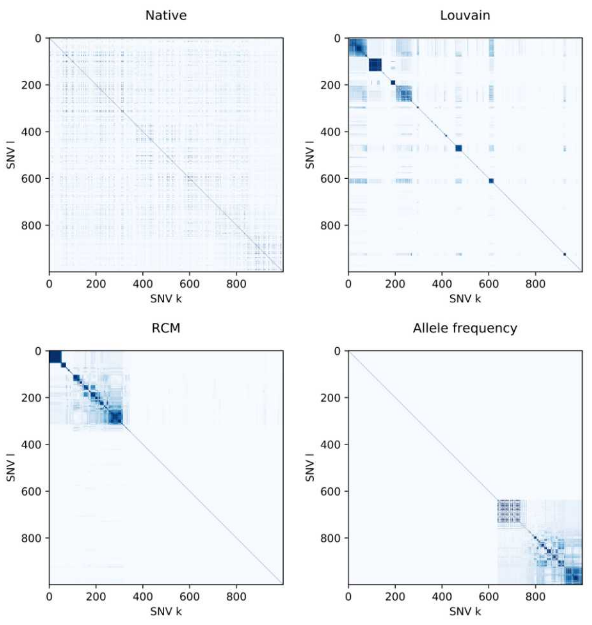

Fig. 4: Visualization of the LD matrix for $m = 1000$ SNVs in the native and reordered sequences of the FIN population. Reordering localizes dependent SNVs.

Rather than apply our algorithm to the full set of SNVs in the genome, we non-specifically choose a subsequence of $m = 1000$ SNVs to show that, given a limited amount of non-specific information on a limited training set (we only have data for the individuals in the 3 population groups described in Section 4), compression can discover elements that are useful for predicting attributes. Furthermore, we choose 100 different subsequences of $m = 1000$ SNVs to show that this result is general and does not depend on which subsequence is chosen. We emphasize that no *a priori* information is used in selecting the subsequences nor in deriving a feature set.

5.1. Visualizing reordering

Figure 4 shows the structure of the LD matrix for a subsequence of $m = 1000$ SNVs belonging to the FIN population. We begin with the LD matrix for the native sequence of SNVs. Non-zero values of $\widehat{\ell}_{k,l}$ are seen distributed throughout $\mathcal{L}$ and in particular far from the diagonal; these correspond to the long-ranged dependencies that motivated the reordering schemes introduced in Section 3.3. If instead dependencies between SNVs were local, the LD matrix would constitute a narrow band of high $\widehat{\ell}_{k,l}$ values along the diagonal, representing local communities of highly dependent SNVs.

Next, we consider the structure of the corresponding LD matrices for the SNV sequences reordered according to Louvain, RCM, and allele frequency. For Louvain and RCM, the reordering is computed from the thresholded adjacency matrix with $\ell^* = 0.5$; reordering by allele frequency is independent of the LD matrix. It can be seen that all reordering algorithms produce small, dense communities of highly dependent SNVs, even those algorithms that don't explicitly operate on optimizing graph community structure. A reduction in dimension and increase in density of the communities represent a localization of dependent elements, that is, we have reduced the distance between two SNVs k and l with high $\widehat{\ell}_{k,l}$. The effects of localizing dependent elements within a sequence on the compressibility and performance of the corresponding model are discussed next, where for illustration purposes we focus on reordering using RCM with threshold $\ell^* = 0.5$. Results using the other reordering schemes are qualitatively similar and are summarized in Table 1.

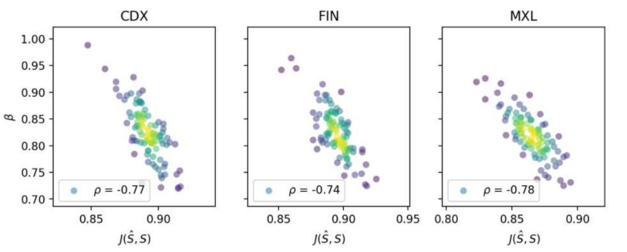

Fig. 5: Relationship between the model compressibility ratio β_j and the extent of reordering, as quantified by the distance $J(\mathcal{S}_j, \hat{\mathcal{S}}_j)$. Reordering by RCM increases model compressibility. Colors correspond to point densities.

5.2. Model compressibility

Recall from Section 3.1 that $\mathcal{S}_j \subseteq \mathcal{S}$ denotes the set of SNV sequences for individuals that are a member of population j. If $\hat{\mathcal{S}}_j$ denotes the reordered sequences of the same individuals, then

$$\beta_j = c(\hat{\mathcal{S}}_j; \hat{\mu}_j)/c(\mathcal{S}_j; \mu_j) \tag{6}$$

less than one indicates an increase in model compressibility with reordering. There are multiple metrics for quantifying the extent of reordering. The Lempel Ziv Jaccard Distance (LZJD) [24] is a convenient information-theoretic metric for quantifying the distance between the sequences before and after reordering.

Figure 5 shows the model compressibility ratio β_j as a function of the LZJD distance $J(\mathcal{S}_j, \hat{\mathcal{S}}_j)$ for populations $j =$ CDX, FIN, and MXL. Each marker is the result using a subsequence of $m = 1000$ SNVs from the individuals in our dataset, and we consider 100 such subsequences. For all populations, the greater the reordering, as quantified by larger distances $J(\mathcal{S}_j, \hat{\mathcal{S}}_j)$, the more compressible the reordered model, as quantified by lower β_j. The Pearson correlation coefficient, which measures the strength and direction of the linear relationship between the two metrics, is also provided in the

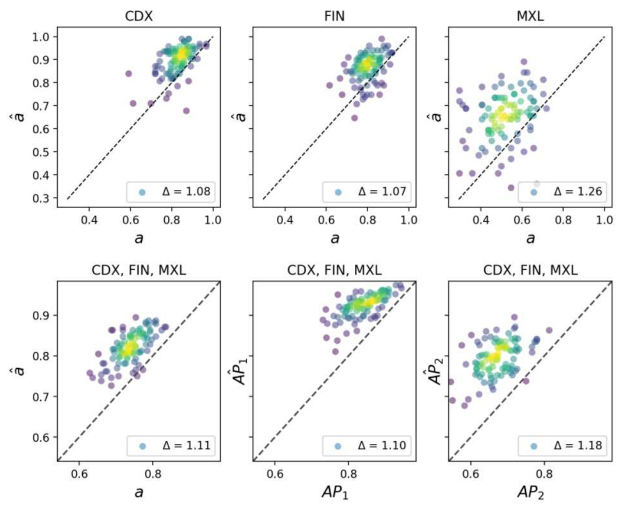

Fig. 6: Performance of compression analytics before and after reordering by RCM. Top: population accuracy; bottom (L-R): total accuracy, average precision using confidence score, and average precision using minimum score. Reordering improves performance in all cases. Colors correspond to point densities.

figure legends. Next, we show that increasing the local structure and compressibility of the sequences forming the PPM models improves the performance of compression analytics.

5.3. Classification

To quantify classifier performance, we estimate the probability that sequence s_i with true label y_i is correctly classified, that is, $\Pr(Y_i = y_i)$, where Y_i denotes the predicted label. This probability for each population j can be approximated by

$$a_j \approx \frac{1}{n_j} \sum_{i=1}^{n_j} \mathbf{1}(Y_i = y_i \mid y_i = j), \qquad (7)$$

where $\mathbf{1}(A)$ is the indicator function, equal to 1 or 0 if event A is true or false, and n_j is the number of test sequences belonging to population j. In standard ML terminology, Eq. (7) is equivalent to the class accuracy of our classifier. Similarly, $\hat{a}_j$ denotes the analogous class accuracy after reordering.

We begin with results using the standard compression approach described in Section 3.1. Figure 6 (top) illustrates class accuracy before and after incorporating the reordering schemes. Results are obtained using a 3-fold cross validation on our dataset, where we train on the larger split. Each marker corresponds to results for a sequence of $m = 1000$ SNVs, and we study the same 100 subsequences presented in Fig. 5. Regardless of the sequence of SNVs chosen to train a model, it can be seen that after reordering, the class accuracy increases for all populations, where it can be seen that most points are above the diagonal, where $\hat{a}_j > a_j$. We can quantify the improvement Δ as the mean ratio of the performance after reordering to the performance before reordering. Reordering showed the greatest improvement for population j = MXL, where $\Delta = 1.26$; less significant improvements are noted for populations CDX and FIN, where $\Delta = 1.08$ and 1.07, respectively. However, MXL had the lowest initial accuracies using the native sequences, where there is perhaps the greatest potential for improvement.

We also quantify performance by calculating the accuracy and the average precision (AP) [25] across all populations; see Fig 6 (bottom). To calculate the AP, we assign each sequence s_i a confidence score $\gamma_i^{(1)}$ that compares how well each population model μ_j compresses s_i, and assigns a higher score to sequences where the difference between the minimum score and the average of the remaining scores is large. The confidence score is defined as follows:

$$\gamma_i^{(1)} = \left(\frac{1}{r-1} \sum_{j=1; j \neq Y_i}^{r} c(s_i; \mu_j) \right) - c(s_i; \mu_{Y_i}), \qquad (8)$$

where $c(s_i; \mu_{Y_i})$ is the minimum compression score; see Eq. (4). We also consider assigning each sequence a score based on the minimum compression score itself, that is, $\gamma_i^{(2)} = c(s_i; \mu_{Y_i})$, where we note that lower scores are preferred when computing the average precision. The average precision using $\gamma^{(1)}$ and $\gamma^{(2)}$ are denoted AP_1 and AP_2, and we can obtain analogous scores that account for the reordering. It can be seen that all three metrics indicate an increase in performance with reordering.

Next, we apply SC for classification; see Section 3.2. Results are once again obtained using a 3-fold cross validation on our dataset, where we train on the larger split. Figure 7 illustrates some statistics of the SC scores belonging to the FIN population for a subsequence of $m = 1000$ SNVs before (top) and after (bottom) reordering by RCM. Each color corresponds to SC scores using a different population model for the compression, where the mean of each is plotted (dark line) along with the interquartile ranges (shaded regions).

Figure 7 (top) shows the result before reordering. The SC scores with respect to the three population models are moderately compressible, do not display any notable structure, and are not that distinguishable from each other. Figure 7 (bottom) shows the result after reordering. Note that the reordering schemes yield different reorderings for the different population models. Therefore it is not possible to compare the SC scores at a given $\hat{k}$; however, it is still possible to compare the SC scores over all $\hat{k}$, which we do next. The SC scores with respect to the three population models are separated into distinguishable regions of high and low compressibility (compared to before reordering). Furthermore the correct model (FIN) has one small incompressible region, whereas the incorrect models (CDX and MXL) exhibit multiple and significant incompressible regions. Because lower scores correspond to better compression, for correct prediction of individual i we

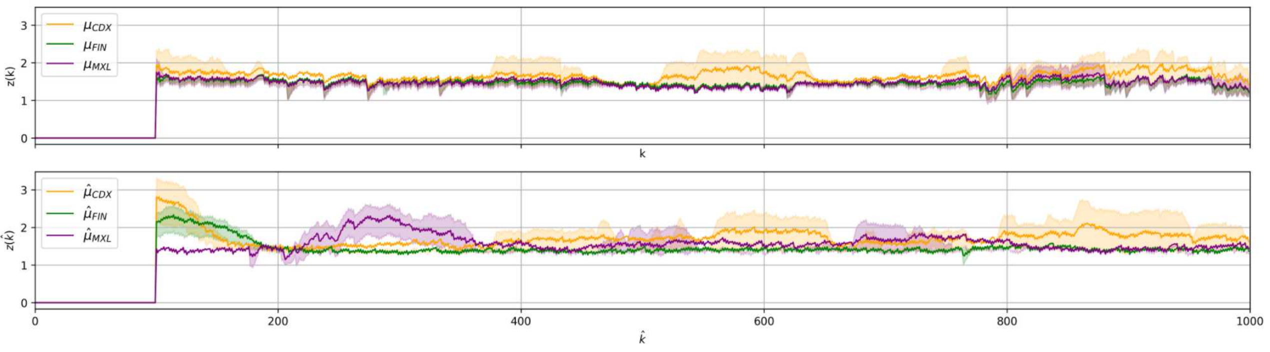

Fig. 7: SC scores as a function of SNV position for a subsequence of $m = 1000$ SNVs belonging to the FIN population before (top) and after (bottom) reordering by RCM. Reordering increases distinguishability of the different population models.

require $\bar{z}_i(\mu_{\text{FIN}}) \leq \bar{z}_i(\mu_{\text{CDX}})$ and $\bar{z}_i(\mu_{\text{FIN}}) \leq \bar{z}_i(\mu_{\text{MXL}})$; see Section 3.2. For this particular subsequence, reordering increases class accuracy from $a_{\text{FIN}} = 0.45$ to $\hat{a}_{\text{FIN}} = 0.72$. We note that the window size used in SC provides a local representation of the data; alternative scoring functions exist that take into account this local structure. We defer a more detailed investigation of the local structure of the SC score to future work.

Table 1 summarizes the total accuracy of the different classifiers and reordering schemes, averaged across the 100 subsequences. For both the standard compression and SC approaches, reordering by RCM shows the greatest improvement, followed by Louvain and allele frequency. Although reordering by allele frequency yields the least improvement, it has the benefit of not requiring the LD matrix computation, which can be expensive for large sequences. Finally, we compare compression analytics to the Random Forest (RF) classifier, a traditional feature-based ML algorithm [26, 27]. As mentioned in Section 1, feature-based ML algorithms require a *predefined* feature set; here we explore multiple feature sets, including the SNV sequence, where each SNV in the sequence is a feature, and the k-mer histograms with $k = 1, 10$. Compression outperforms feature-based ML on all feature sets considered. We note that reordering should modify the k-mer histogram for $k = 10$; we find that it does not yield an increase in the performance of RF.

Table 1: Summary of the total accuracy and accuracy improvement (Δ), if relevant. All reordering schemes considered improve accuracy of both standard compression and SC, with RCM yielding the most significant improvements.

Method	Accuracy	Δ
RF (Native)	0.67	–
RF (k-mer histogram, $k = 1$)	0.55	–
RF (k-mer histogram, $k = 10$)	0.68	–
Compression (Native)	0.74	–
Compression (Louvain)	0.81	1.10
Compression (RCM)	**0.82**	**1.11**
Compression (Allele Frequency)	0.79	1.06
SC (Native)	0.70	–
SC (Louvain)	0.75	1.08
SC (RCM)	**0.77**	**1.10**
SC (Allele Frequency)	0.73	1.04

6. CONCLUSIONS

Understanding what attributes can be inferred from available genomic information is a critical part of genomic privacy and security. We have demonstrated that compression analytics are successful at discovering relevant features in genomic data without the need for a predefined feature set and with very little training data. Furthermore, we have modified compression analytics to handle long range dependencies that are ubiquitous in the human genome. Our approach outperformed a more traditional feature-based machine learning method using Random Forests.

Finally, we note that an interesting analogy exists between genomic sequences, which can be thought of as code, and other forms of code, such as computer code. Both genomic and computer code can contain references that introduce long-ranged dependencies and non-sequential functional interpretations. Developing methods for identifying predictable elements in genomic sequences is therefore useful beyond the application considered here.

7. REFERENCES

[1] G. S. Ginsburg and H. F. Willard, "Genomic and personalized medicine: Foundations and applications," *Translational Research*, vol. 154, no. 6, pp. 277–287, 2009.

[2] M. Naveed, E. Ayday, et al., "Privacy in the genomic era," *ACM Computing Surveys*, vol. 48, no. 1, pp. 6:1–

6:44, Aug. 2015.

[3] Y. Erlich and A. Narayanan, “Routes for breaching and protecting genetic privacy,” *Nature Reviews Genetics*, vol. 15, pp. 409–421, 05 2014.

[4] J. Gitschier, “Inferential genotyping of Y chromosomes in Latter-Day Saints founders and comparison to Utah samples in the HapMap project,” *American Journal of Human Genetics*, vol. 84, no. 2, pp. 251–258, 02 2009.

[5] M. Gymrek, A. L. McGuire, D. Golan, E. Halperin, and Y. Erlich, “Identifying personal genomes by surname inference,” *Science*, vol. 339, no. 6117, pp. 321–324, 2013.

[6] A. Harmanci and M. Gerstein, “Quantification of private information leakage from phenotype-genotype data: Linking attacks,” *Nature Methods*, vol. 13, pp. 251–256, 02 2016.

[7] N. Homer, S. Szelinger, et al., “Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays,” *PLOS Genetics*, vol. 4, no. 8, pp. 1–9, 08 2008.

[8] M. T. Goodrich, “The mastermind attack on genomic data,” in *2009 30th IEEE Symposium on Security and Privacy*, May 2009, pp. 204–218.

[9] R. Wang, Y. F. Li, et al., “Learning your identity and disease from research papers: Information leaks in genome wide association study,” in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, New York, NY, USA, 2009, CCS ’09, pp. 534–544, ACM.

[10] S. Samani, Z. Huang, et al., “Quantifying genomic privacy via inference attack with high-order SNV correlations,” in *2015 IEEE Security and Privacy Workshops*, 2015, pp. 32–40.

[11] N. von Thenen, E. Ayday, and A. E. Cicek, “Re-identification of individuals in genomic data-sharing beacons via allele inference,” *Bioinformatics*, vol. 35, no. 3, pp. 365–371, 07 2018.

[12] M. W. Libbrecht and W. S. Noble, “Machine learning applications in genetics and genomics,” *Nature reviews. Genetics*, vol. 16, no. 6, pp. 321–332, 06 2015.

[13] D. R. Schrider and A. D. Kern, “Supervised machine learning for population genetics: A new paradigm,” *Trends in Genetics*, vol. 34, no. 4, pp. 301–312, 2018.

[14] M. Li, X. Chen, et al., “The similarity metric,” *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3250–3264, 2004.

[15] C. L. Ting, A. N. Fisher, and T. L. Bauer, “Compression-based algorithms for deception detection,” in *Proceedings of the 9th International Conference on Social Informatics*, 2017, pp. 257–276.

[16] Daniel Quang, Yifei Chen, and Xiaohui Xie, “DANN: a deep learning approach for annotating the pathogenicity of genetic variants,” *Bioinformatics*, vol. 31, no. 5, pp. 761–763, 10 2014.

[17] M. Humbert, E. Ayday, J.-P. Hubaux, and A. Telenti, “Quantifying interdependent risks in genomic privacy,” *ACM Transactions on Privacy and Security*, vol. 20, no. 1, pp. 3:1–3:31, 2017.

[18] J. G. Cleary and I. H. Whitten, “Data compression using adaptive coding and partial string matching,” *IEEE Transactions on Communications*, vol. 32, pp. 396–402, 1984.

[19] I. H. Witten, R. M. Neal, and J. G. Cleary, “Arithmetic coding for data compression,” *Communications of the ACM*, vol. 30, pp. 520–540, 1987.

[20] C. Ting, R. Field, A. Fisher, and T. Bauer, “Compression analytics for classification and anomaly detection within network communication,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1366–1376, May 2019.

[21] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, pp. P10008, oct 2008.

[22] E. Cuthill and J. McKee, “Reducing the bandwidth of sparse symmetric matrices,” in *Proceedings of the 1969 24th National Conference*, New York, NY, USA, 1969, ACM ’69, pp. 157–172, ACM.

[23] The 1000 Genomes Consortium, “A global reference for human genetic vatiation,” *Nature*, vol. 526, pp. 68–74, 2015.

[24] E. Raff and C. Nicholas, “Lempel-Ziv Jaccard distance, an effective alternative to ssdeep and sdhash,” *Digital Investigation*, vol. 24, pp. 34 – 49, 2018.

[25] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA, 2008.

[26] Leo Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[27] F. Pedregosa, G. Varoquaux, et al., “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.