A Machine Learning Approach to Genome Compression

Fermi Ma FERMIM@PRINCETON.EDU
Cyril Zhang CYRIL.ZHANG@PRINCETON.EDU
35 Olden Street, Princeton, NJ 08544

Abstract

A database of human genomes is sparse and redundant enough to admit a compact representation. However, to perform useful operations, this data typically needs to be decompressed. In this paper, we explore compressed genome representations on which we can operate directly. We hope that this will have applications in large-scale data mining of genomes.

1. Introduction

1.1. The 1000 Genomes data set

The 1000 Genomes Project is an open-source data set consisting of genome data from 2504 individuals spread over 26 worldwide ethnic groups (Consortium et al., 2012). The project aims to include all alleles that appear with greater than 1% frequency in each population. The data set does not come with any phenotype information other than population label and kin relations.

The full data set, in its final version, consists of 84,739,846 variants across 2504 people. In our preliminary studies, as a representative subset of autosomal genome data, we focus on 5000 contiguous sites on chromosome 22 for all 2504 individuals at which there is a single allelic variant (rather than multi-allelic sites).

1.2. Sparsity of allelic variation

Figure 1 shows the frequency of heterozygous and homozygous variant alleles (compared to the data set's reference genome) of an arbitrary representative sample of 5000 sites on chromosome 22. From this plot, we can see that over 40% of variants occur *only once* in the data set; in fact, around 90% of variants occur fewer than 10 times. This observation is central to representing genome data concisely: there exists a small subset of alleles that exhibits a high degree of variation.

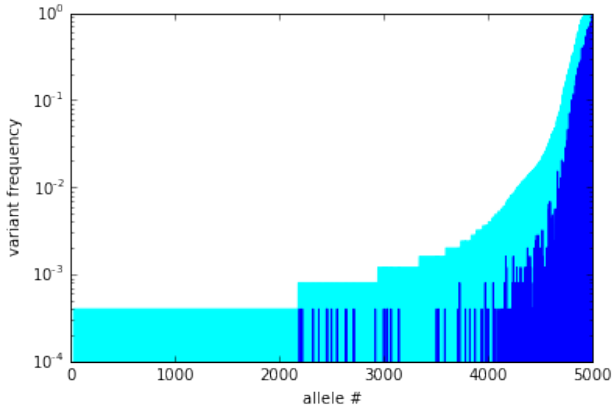

Figure 1. Frequencies for 5000 allelic variants (blue = homozygous, cyan = heterozygous).

The publishers of the 1000 Genotypes data set note that rarer alleles indicate more recent mutations, and thus are generally restricted to just one population. As a result, these rare alleles encode a disproportionately high amount of information about ethnicity. We will qualitatively verify this claim in this paper, and this will be an important fact in determining feature ranking in our compression scheme.

Another useful exploratory view of the data set is the variant bitmap, shown in Figure 2. This provides a visual verification of the sparsity structure of our data: a large region of isolated points (rare variants) dominates the grid, while the common variants from a thin, dense region of high frequency and variance. Additionally, it is interesting to note the emergence of a discrete band structure. The individuals (vertical axis entries) in Figure 2 are grouped by ethnicity, which strongly suggests that a linear model will successfully classify between these phenotypes. Each ethnic group evidently possesses a visually identifiable *fingerprint*.

It is not immediately clear, however, that the sparse region also encodes phenotypic information. In our methods, we recover features that are important in determining ethnicity

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

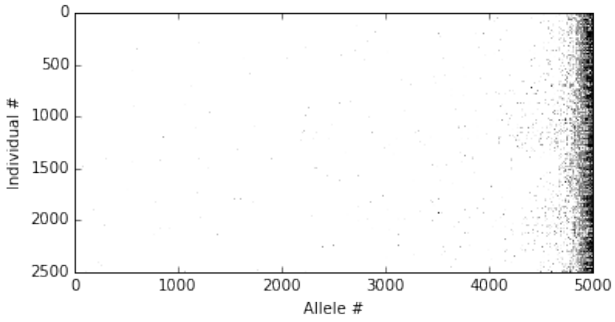

Figure 2. Variant bitmap for 5000 alleles across 2504 individuals (white = homozygous reference, black = homozygous variant, grey = heterozygous).

as the components that arise from sparse regression. We find that the maximally informative sparse genetic fingerprints do incorporate a mixture of high-variance and low-variance alleles.

From the perspective of compression, the sparse and dense regions can be represented using distinct schemes. As a first-cut measure, we can store the sparse components (variants which occur with frequency below some threshold α) as a variable-length list of nonzero coordinates, and the dense region as a bitmap. Both formats are very easy to parse in isolation, in comparison to the run-length encoding approach of Layer et al. (Layer et al., 2015) We reason that this simple compression scheme is sufficient to exploit the sparsity of the data; such strategies as run-length encoding, delta encoding, and Burrows-Wheeler transform pre-processing are designed to exploit further structure of the bitmaps as strings.

1.3. Ethnicity clustering

The ethnicity labels provided by the 1000 Genomes data set allows us to conveniently benchmark our methods; we can evaluate the viability of our compressed data representations by attempting to distinguish between clusters, which are well-separated in a simple linear model in the original vector space.

More specifically, we wish for our data to have the property that a short prefix of it (essentially truncating the coordinates of the genotype vectors corresponding to each individual) keeps clusters well-separated; as we read more input coordinates, our solution becomes more refined.

As mentioned, each genotype vector is labeled with one of 26 ethnic groups. The number of samples within each group ranges from 61 (ASW, African Americans in the southwestern US) to 113 (GWD, Gambians in the Western Division of the Gambia).

2. Related work

2.1. Data representation and compression

Human genotypes are typically stored as a table of variations relative to a fixed reference genome, stored elsewhere. This is a ubiquitous practice, and is regarded as part of the data reduction rather than compression, after sequence alignment. The result of this process is usually a table in **variant-centric format** (vcf), in which each row corresponds to a point of variation, and the columns give the identity of that allele across each individual in the sample set. In this paper, we will be concerned with further compressing the genome database, starting from the vcf representation.

Layer et al. give a heuristic method for genome compression (Layer et al., 2015). They take the transpose of the vcf table, and permute the columns (now corresponding to genotypes at each allele) in increasing order of allele count in the sample population. This greatly increases compressibility; they subsequently apply a simple run-length encoding scheme so that the data can be indexed, locally decompressed, and queried efficiently. The authors show that this gives a compression rate roughly on par with the LZ77 algorithm, reducing the space requirement of the 1.3TB raw vcf files to a much more manageable 14GB.

Christley et al. gave an alternative method for genome compression. Their method avoids storing the full position of each SNP along the chromosome, but instead the distance from the previous SNP, a much smaller number. They use Huffman coding as a final step, and achieve a 1000-fold compression on James Watson's genome (Christley et al., 2009). This work has inspired a number of other papers, such as one by Pavlichin et al., that slightly improve the compression ratio by taking advantage of known genome structure (Pavlichin et al., 2013).

Our work differs markedly from these approaches to genome compression. Current compression schemes are lossless and rely on a full reference genome in order to decompress the data. Furthermore, it seems difficult to reach any useful conclusions by looking only at the compressed files. In this paper, we focus on achieving a lossy compressed data representation that can be useful in its compressed form. Our data representation will allow us to better use the genotype vectors for linear models in machine learning.

2.2. Compression-based machine learning

One conceivable way to perform machine learning on compressed data is to use the symbols from the compression algorithm directly as features. This has been explored in natural language processing to some success. Sculley and Brodley show that the symbols from Lempel-Ziv and pre-

dictive prefix matching (PPM) family serve as effective feature vectors (in an implicit high-dimensional vector space) for a document classification problem. (Sculley & Brodley, 2006). Under this benchmark, they find that these vectors outperform the n-gram and binary bag-of-words models by a small margin, suggesting that the symbols encode the structure of the text.

In this paper, we examine an analogous method. However, we find that compression of genome data does not significantly preserve cluster structure. This is perhaps unsurprising: whereas the compressibility of natural language text derives from redundancy, the compressibility of genome data is largely due to the rarity of most alleles.

2.3. Machine learning in genomics

Libbrecht and Noble survey common techniques for applying machine learning to genomics (Libbrecht & Noble, 2015). Often, a supervised approach is used to train an algorithm to identify transcription splice sites, promoters, enhancers, or positioned nucleosomes. They also identifies a number of problems that arise when dealing with genomic data. Imbalanced class size is often an issue for supervised learning algorithms, since negative examples of certain structures can vastly outnumber positive examples.

The focus of this paper is different from previous machine learning work in genomics, as we are not trying to build a classifier. As such, we also face a very different set of problems than those mentioned in the Libbrecht and Noble paper (Libbrecht & Noble, 2015).

3. Methods and Assessment

3.1. PCA indicates high linear separability

One way to immediately verify the linear separability of ethnic group clusters is to run principal component analysis (PCA) on the genotype vectors (after z-scoring each coordinate). Even in two dimensions, some clusters are clearly well-separated. See Figure 3 for this striking result; although some pairs of ethnicities tend to be less separable under this very low-dimensional projection, this shows that the recoverability of ethnic clusters seems to be a good benchmark of lossy data compression.

3.2. Failure of compression-based features

We attempted to replicate the approach described by Sculley and Brodley, in which we take the domain of a compression algorithm's symbol table as the set of coordinates in an implicit feature space, with the appearance counts of the symbols as entries. Plotting the same clusters that were well-separated under 2-dimensional PCA on the raw data, we see in Figure 4 that the clusters are not recovered.

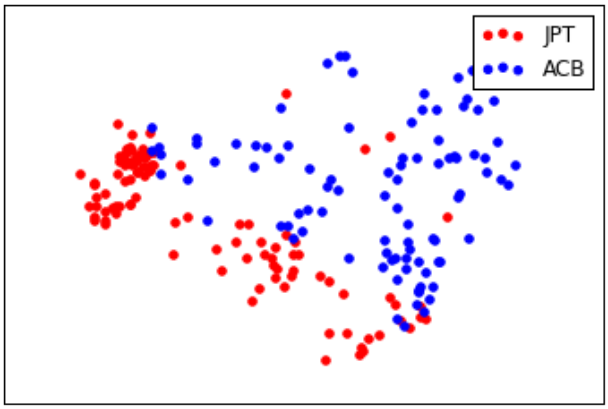

Figure 3. Plot of z-scored genotype vectors labeled JPT (Japanese in Tokyo) and ACB (African Caribbeans in Barbados), reduced to 2 dimensions via PCA.

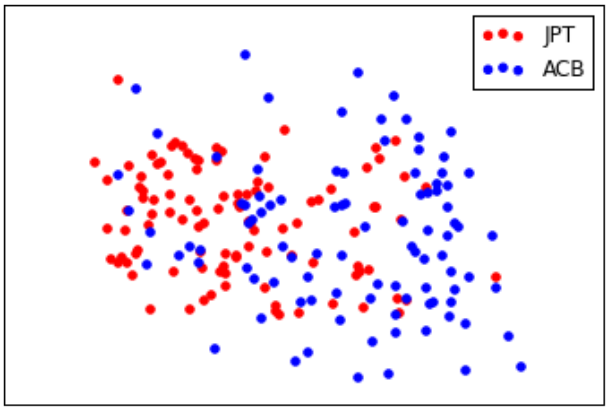

Figure 4. Plot of dimension-reduced vectors in JPT and ACB clusters, using LZW symbol table-derived feature space.

We found that although string compression algorithms do successfully produce a shorter representation of the genome data, the symbols obtained are not reliably informative of the structure of an individual's genome. Upon examining the symbol tables obtained by Lempel-Ziv compression algorithms, we find that they largely consist of long strings of zeros. The presence of nonzero entries at differing locations significantly perturbs the structure of the symbol table, rendering this approach infeasible.

We hypothesize that the compression-based feature may be salvageable for different problems in bioinformatics. Although it is difficult to justify interpreting a variant bitmap as a string, it is entirely natural to interpret a raw sequence of base pairs in the framework of string redundancy. As such, this approach may be fruitful in classifying between

snippets of DNA belonging to very different organisms based on the statistical properties of their base pair strings.

3.3. Finer-grained cluster separation via SVMs

To find hyperplanes that separate the clusters in our data, we can use the linear support vector machine (SVM) model. Since each allele corresponds to a feature, the dimensionality of the data set is extremely high; the features far outnumber the sample set. This will be true in any genomic data set in the foreseeable future, as it would be prohibitively expensive to perform whole exome sequencing on the order of 10^8 individuals. Even in our representative sample of 5000 alleles, careful attention must be taken as to prevent overfitting of models. Thus, we focus on the scope of high-dimensional data with low sample size.

Dimension reduction (via, say, PCA) is a common answer to these constraints, and is shown above to preserve separating hyperplanes. However, when we perform such a drastic reduction, we lose the informative value of specific coordinates.

Instead of finding a separating hyperplane, we strengthen the requirement significantly: we seek a separating hyperplane whose normal vector has support over a small subset of coordinates. This is the classic problem of sparse regression. We found that applying the well-known LASSO ℓ^1 regularization technique indeed produced sparse separators. For the JPT and ACB groups $(N = 104 + 96)$, we obtained a linear combination of 30 alleles (out of 5000) which almost completely separated the clusters; see Figure 5.

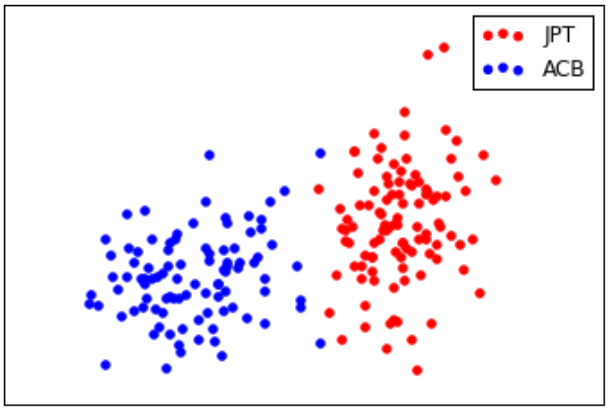

Figure 5. Plot of JPT and ACB clusters, with x-axis determined by component along sparse separating hyperplane normal. Y-axis is a random projection, for visualization purposes.

3.4. Cross-validation of sparse linear separators

We tested our sparse SVM for the possibility of overfitting. We tested the model with the upper bound on the ℓ^1 norm set to 0.3, which in this case returned a separating hyperplane with 42 nonzero components. For this experiment, we again focused on classifying the JPT and ACB populations.

We performed 10-fold cross validation, and found near-perfect separation on the unseen testing data for all 10 partitions. In Figure 6, we show the results of 2 of the 10 iterations of the 10-fold cross validation scheme.

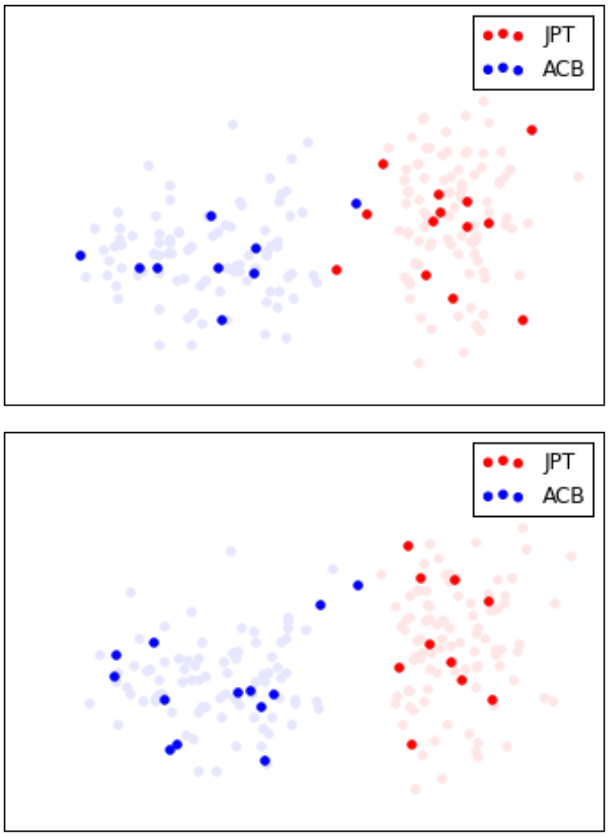

Figure 6. Plots of 2 of the 10 partitions in 10-fold cross validation between JPT and ACB clusters. The x-axis is determined by component along the separating hyperplane from a sparse SVM trained only from the training set. The y-axis is a random projection, for visualization purposes. Faded points are the training set (the remainder of the points) for each fold.

3.5. Compression method

We considered the following compression scheme:

1. Divide the data into *chunks* of 5000 alleles.

2. In each chunk, identify the most important alleles by finding (sparse normal) separating hyperplanes for each pair of ethnicity clusters; call these normals $\{n_j\}$.
3. Let $\hat{n}$ aggregate the importances of the alleles: $\hat{n}_i = \sum_j |n_j|$.
4. Within each chunk, separate the alleles $\{i\}$ into two groups: $A^+ = \{i|\hat{n}_i > 0\}$, and $A^0 = \{i|\hat{n}_i = 0\}$. Sort A^+ in decreasing order of $\hat{n}_i$.
5. Let F_1 be the interleaved rows of all the $\{A^+\}$ from each chunk. Store this as a dense matrix.
6. Let F_2 be the rows of all the $\{A^0\}$ with more than ϵn entries (we try $\epsilon = 0.1$), ordered by decreasing variant count. Store this as a dense matrix.
7. Let F_3 be the remainder of the data, the sparse rows of $\{A^0\}$, ordered by decreasing variant count. Store this as a sparse matrix.
8. Output F_1, then F_2, then F_3 as the reordered, compressed data set.

This orders the alleles by their importance when available, and heuristically by count otherwise. We tested our method on 20 contiguous chunks of chromosome 22, for a total of 100,000 alleles.

3.6. Assessment of compression efficiency

After processing the entire data set, one can find the size of our compact representation, and relate it to the total vcf size. This gives an empirical compression efficiency, which can be compared to others in practice.

On a segment of the data set, the compression ratio can be measured as

$$\frac{\text{size of dense data}}{\text{size of dense } F_1, F_2 + \text{size of sparse } F_3}.$$

3.7. Assessment of permutation quality

Our method can be evaluated against two other permutations of the data: random, and sorted by decreasing variance. We employ the following methodology:

1. Take the first $L = 10$ lines of the permutation. Train an ℓ^1-regularized SVM on the ethnicity classification test problem.
2. Record the classification accuracy.
3. Take the next L lines, and add them to the data set. Repeat with these new, higher-dimensional genotype vectors.

4. Preliminary results

4.1. Compression ratio

Our preliminary test on 100,000 alleles gives a compression ratio of 2.63, with chunk size 5000 and $\epsilon = 0.1$. Note that the reference uncompressed size is that of the dense matrix, which is a much more compact representation than the raw vcf files.

4.2. Data truncation experiment

Figure 7 shows typical results for the clustering problem on prefixes of permutations of the alleles. Our permutation stays reliably ahead of the others in terms of accuracy in low dimensions, and typically reaches a perfectly splitting hyperplane earlier than the others. Thus we conclude that our method succeeds in selecting important features for classification.

At this stage of the project, this is our most promising result; it indicates that our compression scheme works well, and can perhaps be improved with better tuning of parameters.

5. Discussion

5.1. Application in big-data genomics

We imagine that our results might be useful in a big-data setting, in which coordinate truncation can be modulated to trade off desired accuracy for running time. Our feature selection produces a rough ranking of alleles by importance; thus, in permuting the data set by this ranking, any statistical method can be tuned by taking the first D columns. In the future, as personal whole exome sequencing becomes increasingly affordable and accessible, available data will become richer. We propose the following workflow:

1. Select a phenotype P, and a desired classification error ϵ.
2. Take the first D rows (alleles) from the data set, and train a classifier for P.
3. If the classifier achieves error greater than ϵ, repeat with $2D, 4D, \ldots$ rows.

At the end of this process, assuming that the set of alleles correlated with the phenotype are not too localized, we have a truncation of coordinates suitable for our specific problem. We imagine that this may be useful when we wish to use the data set to classify (or regress) a large number of diverse phenotypes.

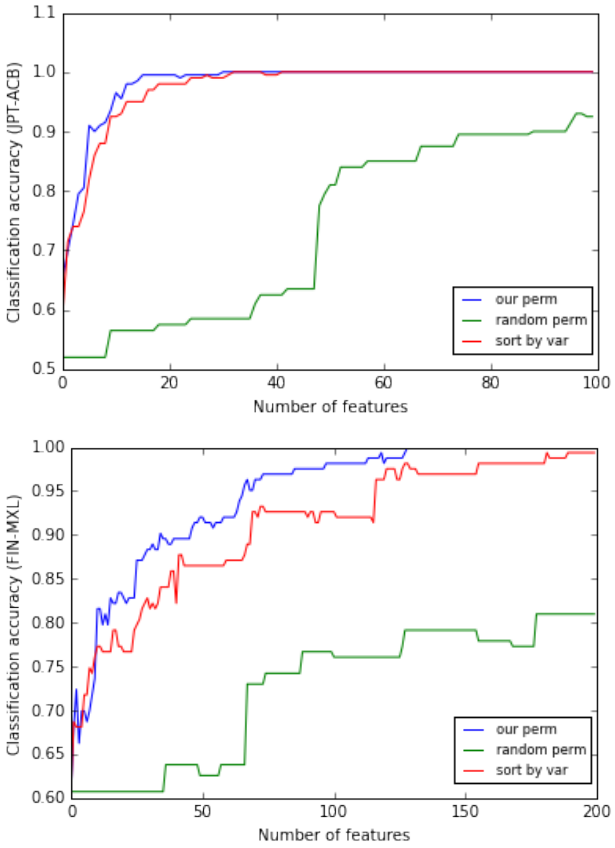

Figure 7. Plots of ℓ^1-regularized classification accuracy vs. number of features in the permutation associated with our method, alleles selected in decreasing order of variance, and a random permutation, for two typical pairs of clusters.

5.2. Future directions

In the following weeks, we plan to explore the following directions:

- Identify regions of interest using a parallel data set (e.g. OMIM genotype-phenotype database), and investigate their relationship with the important alleles determined by our feature selection scheme.
- Process the entire data set using supercomputer resources, once we commit to a method. Obtain a global compression result, which can be compared to competitors.
- Test our method on the other sets of labels for the genome vectors: kinship clusters.

References

Christley, Scott, Lu, Yiming, Li, Chen, and Xie, Xiaohui. Human genomes as email attachments. *Bioinformatics*, 25(2):274–275, 2009.

Consortium, 1000 Genomes Project et al. An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491(7422):56–65, 2012.

Layer, Ryan M, Kindlon, Neil, Karczewski, Konrad J, ExAC, Exome Aggregation Consortium, and Quinlan, Aaron R. Efficient genotype compression and analysis of large genetic variation datasets. 2015. doi: 10.1101/018259.

Libbrecht, Maxwell W and Noble, William Stafford. Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, 2015.

Pavlichin, Dmitri S, Weissman, Tsachy, and Yona, Golan. The human genome contracts again. *Bioinformatics*, 29 (17):2199–2202, 2013.

Sculley, D. and Brodley, C.E. Compression and machine learning: a new perspective on feature space vectors. pp. 332–341, March 2006. ISSN 1068-0314. doi: 10.1109/DCC.2006.13.