Interpreting Generative Adversarial Networks to Infer Natural Selection from Genetic Data

Rebecca Riley[1], Iain Mathieson[2], and Sara Mathieson[1,†]

[1]Department of Computer Science, Haverford College, Haverford PA, USA.
[2]Department of Genetics, Perelman School of Medicine, University of Pennsylvania, Philadelphia PA, USA.
[†] Corresponding author: smathieson@haverford.edu

Abstract

Motivation: Understanding the landscape of natural selection in humans and other species has been a major focus for the use of machine learning methods in population genetics. Existing methods rely on computationally intensive simulated training data incorporating selection. Unlike efficient neutral coalescent simulations for demographic inference, realistic selection typically requires slow forward simulations. Large populations sizes (for example due to recent exponential growth in humans) make these simulations even more prohibitive. Because there are many possible modes of selection, a high dimensional parameter space must be explored, with no guarantee that the simulated models are close to the real processes. Since machine learning methods use the simulated data for training, mismatches between simulated training data and real test data are particularly problematic. In addition, it has been difficult to interpret the trained neural networks, leading to a lack of understanding about what features contribute to identifying selected variants.

Results: Here we develop a new approach to detect selection that does not require selection simulations during training. We use a Generative Adversarial Network (GAN) that has been trained to simulate neutral data that mirrors a real genomic dataset. The resulting GAN consists of a generator (demographic model) and a discriminator (convolutional neural network). For a given genomic region, the discriminator predicts whether it is "real" genomic data or "fake" in the sense that it could have been simulated by the generator. As the "real" training data includes regions that experienced selection and the generator cannot produce such regions, regions with a high probability of being real may have experienced selection. This enables us to apply the trained discriminator of the GAN to held-out test data and identify candidate selected regions. We show that this approach has high power to identify regions under selection in simulations, and that it reliably identifies selected regions

identified by state-of-the art population genetic methods in three human populations (YRI, CEU, and CHB). Finally, we show how to interpret the trained networks by clustering hidden units of the discriminator based on their correlation patterns with known summary statistics. In summary, our approach is a novel, efficient, and powerful way to use machine learning to detect natural selection.

Availability: Our software is available open-source at `https://github.com/mathiesonlab/disc-pg-gan`.

***K*eywords** Population genetics · Generative adversarial network · Natural selection · Interpretability

1 Introduction

In the last several years, numerous deep learning methods have been proposed as solutions to population genetic problems. In particular, convolutional neural networks (CNNs) have been effective and allowed the field to move away from summary statistics and towards analyzing haplotype matrices directly. The broad idea behind CNNs for genetic data is to treat these matrices (with individuals/haplotypes on the rows and sites/SNPs on the columns) as analogous to images, with convolutional filters picking up on correlations between nearby sites. Population genetic CNNs have now been developed for a variety of applications, including recombination hotspot identification [1], introgression, recombination rates, selection, and demography [2], natural selection [3, 4], adaptive introgression [5], balancing selection [6], and the task-agnostic software `dnadna` [7].

One drawback of these machine learning approaches is that they require simulated training data, as labeled examples (i.e. where the historical evolutionary "ground truth" is known) are rare in biology. If the simulated training data is not a good match for the real genetic data, inference results may not be accurate and robust. Default simulations can be a particularly poor match for understudied populations and non-model species, where less is known about their evolutionary parameter space. To address this problem, custom simulations can be created using Generative Adversarial Networks (GANs) [8], which have recently been developed for population genetic data. Our previous work [9] created a GAN (`pg-gan`) that fits an evolutionary model to any population – as training progresses, the generator produces synthetic data that is closer and closer to the real data, confusing the discriminator. The key innovation of `pg-gan` is that it learns an explicit evolutionary generative model, in contrast to other GANs which generate sequences that look like real data from random processes with no underlying model [10, 11].

GANs produce two networks, a generator which creates synthetic examples, and a discriminator which classifies examples as real or fake. After training, the focus is often on the generator, which can be used to produce novel results (i.e. faces, text, genetic data) from a vector of random noise. Although very powerful and well-trained discriminators can be used downstream to validate the examples created by other generators (e.g. [12]), there is typically less emphasis on using or analyzing the trained discriminator.

For `pg-gan` the generator is not a neural network but instead an evolutionary model. The fitted generator from `pg-gan` is therefore useful in its own right, as it is directly interpretable in terms of population genetic parameters (and is therefore directly comparable with traditional inference methods). Because the specification of such models is relatively standardized (population size changes, splits, migrations, admixtures, etc), we can easily use the resulting model in downstream applications or make it available to other researchers in a standard format (see [13]). For example, we could use the demography as a null model and additionally simulate natural selection or other deviations from neutrality. The resulting simulations could be used as the training data for a second machine learning method or as validation data to compare inference algorithms.

In this work we develop a new approach that uses the trained discriminators of `pg-gan` to identify non-neutral regions. The major advantage of this approach over existing approaches is that it does not require simulating selected regions during training. Despite recent advances (e.g. [14]), simulating realistic natural selection is very difficult, with many parameter choices that make mirroring real data challenging. These include the type of selection, time of onset of selection, selection strength, and frequency of the selected variant at the present. If the selected allele is lost, simulation needs to be redone for models with effective population size changes. For models with exponential growth or large population sizes, forward simulations become prohibitively computationally expensive. Our approach bypasses the need to simulate large numbers of selected regions by instead detecting regions of real data that do not conform to an inferred (neutral) demographic model. In many ways, this approach is analogous to traditional population genetic approaches to detecting selection by looking for regions that are outliers in terms of divergence, diversity or haplotype structure [15]. However instead of looking at one or a combination of summary statistics [16], our approach operates directly on the observations, and in principle is able to use all the information present in the data.

Understanding what neural networks actually learn has been a major goal of interpretability work for many years. Although great strides have been made in the image domain (e.g. [17–19], less is known about what CNNs for

population genetic data are learning (see [20] for recent progress). Our interpretability analysis links traditional summary statistics (which are known to be informative for selection and other evolutionary parameters) with the hidden nodes of the discriminator network. This gives us a window into the network's ability to compute existing summary statistics, importance and redundancy of different statistics, and differences between discriminators.

2 Methods

2.1 `pg-gan` discriminator

Our existing GAN application, `pg-gan`, consists of a CNN-based discriminator and an `msprime`-based generator that are trained in concert. During training, the generator selects parameters to use for the simulation software `msprime` [21, 22], creating "fake" regions. The discriminator learns to distinguish these neutral regions from the real training data, which contains both neutral and selected regions. With feedback from the discriminator, the generator learns which parameters will produce the most realistic simulated data, and the discriminator gets better at picking up on differences between the two types of data. `pg-gan` is implemented in `tensorflow` [23] and can be run on a GPU using `CUDA` [24]. As the discriminator is a CNN, we are able to train it using traditional gradient descent approaches (i.e. backpropagation). However, since we currently cannot pass gradients through the coalescent with recombination, we train the generator using a gradient-free approach (in this case simulated annealing, but other methods are possible). At the end of this optimization process, we obtain a generator that is good at creating realistic simulated data, and a discriminator that can identify real data accurately.

In this work we modify `pg-gan` to save the trained discriminator network. The input to this CNN is a genomic region represented by a matrix of shape $(n, S, 2)$ where n is the number of haplotypes, S is the fixed number of consecutive SNPs, and there are two separate channels: one for the haplotype data and one for the inter-SNP distances (duplicated down the rows). These two channels are analogous to the three RGB color channels for images. For one-population evolutionary models, this network has two convolutional layers (32 and 64 1×5 filters respectively), two fully connected layers (each with 128 units), and an output of a single scalar value (probability of the example being real). With 1×2 max-pooling after each convolutional layer and a permutation invariant function before the fully connected layers, this results in 76,577 trainable parameters, all of which are saved after training.

This saved network can then be used to predict outcomes for new genetic regions. Values closer to 1 indicate more similarity to the real training data, and values closer to 0 indicate more similarity to the simulated data. Since the real

data includes regions that have experienced selection, the intuition is that higher discriminator prediction probabilities correspond to non-neutral regions. To confirm this, below we outline several ways that we evaluate the resulting `pg-gan` discriminators. Most discriminators conform to this hypothesis, but as training a GAN is a minimax optimization problem, training problems are common, including degenerate states, local optima, and the discriminator and generator learning at different rates. Discriminators that do not have these issues and pass our evaluations are retained and averaged in an ensemble method to create the final predictions of selected and neutral regions.

2.2 Population data and demographic models

To ensure that we are training and testing on different datasets, we use three pairs of similar populations from the 1000 Genomes Project [25]: YRI and ESN (West African), CEU and GBR (Northern European), and CHB and CHS (East Asian). These pairs were selected since they are the most similar from a population structure perspective (see [25], Figure 2). Specifically, we train discriminators using YRI, CEU, and CHB, then test on ESN, GBR, and CHS respectively. Although the number of samples may differ between the populations, this is not a problem due to the permutation-invariant architecture of the discriminator. Population samples sizes and number of regions are given in Table 1.

Table 1: Populations analyzed along with the number of individuals in each population. The number of regions is determined using non-overlapping 36-SNP windows, where at least 50% of bases must be in the accessibility mask.

population	# individuals	# regions
YRI	99	468,686
ESN	99	464,098
CEU	99	294,755
GBR	91	278,729
CHB	99	278,818
CHS	105	271,902

For all populations we used an exponential growth model and fitted five parameters using `pg-gan`: the ancestral population size (N_1), the timing of a bottleneck (T_1), the bottleneck population size (N_2), the onset of exponential growth (T_2), and the growth rate (g). For all simulations we used a constant mutation rate of $\mu = 1.25 \times 10^{-8}$ per base per generation and recombination rates drawn from the HapMap recombination map [26]. We simulated $L = 50$kb regions and then extracted the middle $S = 36$ SNPs. To avoid assuming an ancestral state, for each SNP we encode the minor allele as 1 and the major allele as -1. We found that centering the data around zero worked best with typical neural network activation functions, and also allowed us to zero-pad a small number of regions with fewer than S SNPs. For real data we filter non-biallelic SNPs and ensure that at least 50% of the bases are inside callable regions (see [25]), so that the discriminator avoids making real vs. fake distinctions based on artifacts.

2.3 Validation on simulated data

Once the discriminators have been trained and saved, we can evaluate their performance on new genomic data for the purpose of identifying and filtering failed training runs. Since we do not use selection simulations during training, we are able to use them to evaluate the performance of the method. We use `SLiM` [14, 27], which can model a variety of types of natural selection. For the demography we use the same evolutionary model (exponential growth) as we used during `pg-gan` training. For European populations we use the parameters $N_1 = 22552$, $N_2 = 3313$, $g = 0.00535$, $T_1 = 3589$, $T_2 = 1050$, which were obtained from a run of `pg-gan` with CEU training data. During this training run, we capped the exponential growth parameter at 0.01, as allowing this parameter to be too large led to `SLiM` computations being too slow. See Figure 1 for a visualization of how well this simulated data matches CEU. Under this combination of parameters, we produced neutral regions, as well as regions with a single mutation undergoing positive selection with selection coefficients of of 0.01, 0.025, 0.05, and 0.1. For each selection coefficient we simulated 100 regions and recorded how many of them were classified as "fake".

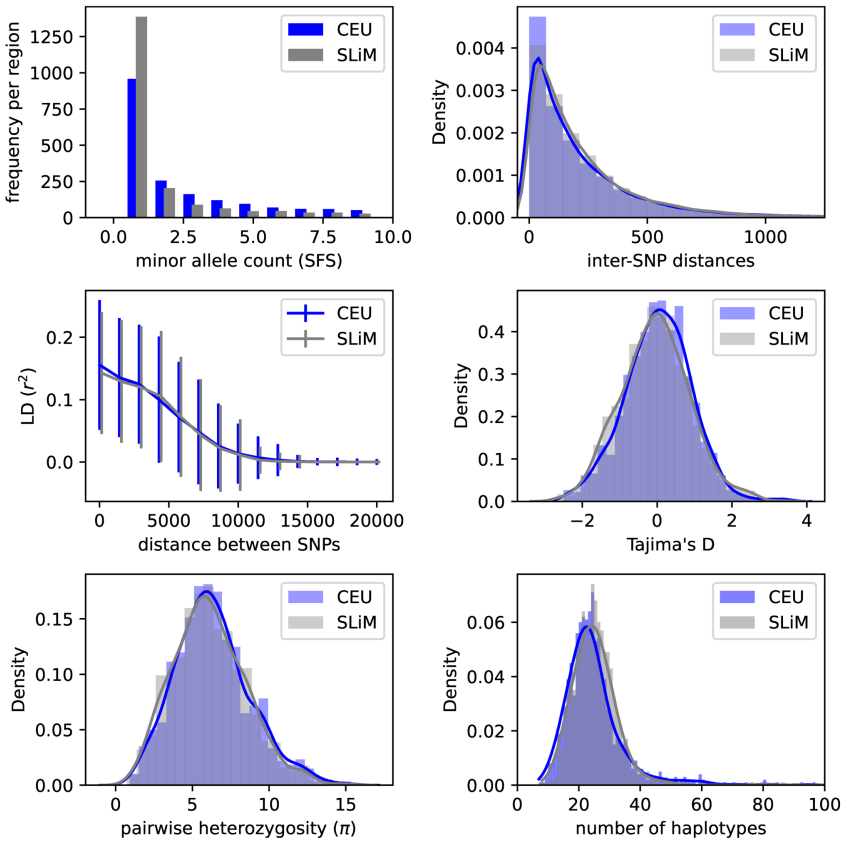


Figure 1: Summary statistic comparison of real CEU training data to data simulated under `SLiM` with parameters inferred by `pg-gan`. The number of regions used for comparison was 1000.

We used a similar procedure to obtain a neutral demographic model for YRI. Due to the effect of large population sizes on `SLiM` runtimes, we again capped g at 0.01 and additionally capped T_2 (the onset time of exponential growth) at 750 generations. The resulting parameters were $N_1 = 23231$, $N_2 = 29962$, $g = 0.00531$, $T_1 = 4870$, $T_2 = 581$ and a visualization of the match is shown in Figure S1.

Selection is introduced by adding the mutation to one individual in the population, 1000 generations before the present. If the mutation is lost in any generation, we restart the simulation from the previous generation until we have reached the present – the resulting tree sequence can then be stored for further processing. First, the tree sequence is recapitated using coalescent simulations for efficiency (using the `pyslim` module). This step ensures that we have a complete tree sequence with all necessary ancestral nodes. Next, we sample the population down to $n = 198$ haplotypes, to match the real data. This allows us to prune and simplify the tree sequence before adding neutral mutations with `msprime`. For each region we store arrays with S SNPs, as well as arrays with all the SNPs, as it allows us to compute Tajima's D.

We run the resulting selection simulations through each discriminator to obtain predictions. If we do not see a separation between strong selection and neutral/weak selection, there is likely a training problem with the discriminator and we do not use it in the ensemble predictor (see below).

2.4 Validation on known selected regions

Second, we examine each discriminator's performance on real genomic regions previously identified as targets of selection by Grossman *et al* [16] using a combination of different summary statistics. We use the 153 regions from their Table S1, including genes involved in metabolism, disease resistance, and skin pigmentation. We converted the start and end positions of each region to HG19 coordinates and sorted by population (CEU, YRI, and CHB/JPT). We also analyzed the HLA region separately. High levels of polymorphism in the HLA region have been linked to balancing selection on loci related to the immune system [28, 29]. Overall this resulted in a comparison between four types of data:

- Regions simulated using `msprime` under the demographic parameters inferred by `pg-gan` (corresponding to the current discriminator)
- Random regions from real data (mostly neutral)
- Regions under positive selection as identified in [16]
- Regions from the HLA, chromosome 6: 29640020-33120103

Successfully trained discriminators should group the predictions for simulated and real *neutral* regions together, with a distinction between this group and at least one of the selected types (known positive or balancing). If a discriminator groups all types of real data away from the simulations, this indicates a poor match between the inferred demography and the real data. In this case we would discard the discriminator from the ensemble predictor (see below).

To assess the discriminator's predictions genome-wide, we iterate through the entire genome (using non-overlapping 36-SNP windows) and make a prediction (logit score) for each region. Logits are converted to probabilities, then z-scores, then finally p-values. To visualize outlier regions, we create modified Manhattan plots using negative $\log_{10}$ p-values on the y-axis.

2.5 Ensemble method to detect outlier regions

Because individual trained discriminators are quite variable in their performance, we took the approach of averaging predictions for each region over multiple discriminators. However, some discriminators fail to train properly and would lead to incorrect predictions. We therefore use the heuristic described in the previous section to discard poorly trained discriminators before combining the predictions of the remaining discriminators. We describe this approach formally in Algorithm 1. In practice we find that around $\frac{2}{3}$ of discriminators train successfully.

Algorithm 1: Ensemble method for using `pg-gan` discriminators to detect non-neutral regions.

Θ – parameters for an evolutionary model
X – real training data (X_n neutral, X_s known selected)
Y – real test data (Y_n neutral, Y_s known selected)
Z – simulated data (Z_n neutral, Z_s under selection)

Input: K discriminators ($D_1, \cdots, D_K$) trained with X
Output: set of successful discriminators C, predictions for all regions in Y

for *each discriminator D_k, generator G_k, inferred parameters Θ_k* **do**
  Let $Z_n = G_k(\Theta_k)$ be neutral data simulated under Θ_k

  Let the following conditions define a successfully trained discriminator:
  1. D_k does not predict the same value for all regions
  2. $D_k(Z_s)$ is separated by selection strength
  3. $D_k(Y)$ is not all grouped away from $D_k(Z_n)$
  4. There are is at least one genome-wide significant result in $D_k(Y)$

  if *conditions 1-4 hold for D_k* **then**
    Add D_k to set of successfully trained discriminators C
  end
end

for *region y in Y* **do**
  for *discriminator D_i in C* **do**
    $p = D_i(y)$
  end
  Average all predictions - if result is significant at the 5×10^{-8} level, return as a putatively selected region
end

2.6 Interpretability analysis

Finally, to understand what the discriminator network is learning, we sought to investigate whether the network had learned how to compute summary statistics that are known to be informative for evolutionary parameters (demography, selection, etc). To this end, we computed the correlation between the node values of the last hidden layer and various summary statistics. The motivation for analyzing the last hidden layer is that deep into the network, information from the haplotypes should be greatly distilled and processed, with informative high-level "features" extracted. The remaining step in the network is a linear combination of these hidden node values to create a logit value, which can be converted into a probability. We computed pairwise correlations between 128 hidden values and the following 61 summary statistics:

- The first 9 entries of the folded site frequency spectrum (SFS), excluding non-segregating sites.
- The 35 inter-SNP distances between the 36 SNPs of each region. These distances are rescaled to fall between 0 and 1.
- 15 linkage disequilibrium (LD) statistics (see [9] for details of the computation).
- Pairwise heterozygosity (π).
- The number of unique haplotypes across the 36 SNPs.

Note that the inter-SNP distances are also an input to the network, so these statistics are included to see if the network is making use of this information. We calculated Pearson correlation coefficients and plotted them in a heatmap with summary statistics along rows and hidden values along the columns. To better visualize relationships between the hidden units, we clustered the columns according to their similarity using agglomerative clustering. This hierarchical clustering algorithm produces a tree, which we use to sort the columns. The final visualization groups hidden nodes that are performing similar computations – since we perform a dropout (with rate 0.5) after each fully connected layer, we expect some redundancy in node behavior. We experimented with reducing the number of hidden units in the last few layers, as well as adding three fully connected layers (instead of two). However, all these modifications to the discriminator CNN resulted in degenerate training results (e.g. predicting all regions as real) so we pursued only the original architecture.

3 Results

3.1 Summary of discriminator training

For each training population (YRI, CEU, and CHB) we ran `pg-gan` 15 times to obtain a set of discriminators. For each run we used a different seed, although even runs with the same seed can produce different results depending on the version of `tensorflow` and `CUDA`. In a few cases, training failed completely and the resulting discriminator predicted the same value for all regions (in other words, it ignores the input data). In a few other cases, the discriminator distinguished all types of real data (both those known to be under selection and neutral) and simulated data easily. This indicates that GAN training did not find demographic parameters able to create simulations matching the real data. Finally, there were a few cases of no genome-wide significant results, and one instance when simulated data under selection was grouped together (indicating little ability to distinguish strong selection from weak). After we discarded these less successful discriminators we were left with 30/45, summarized below for the different training populations.

- YRI: 9/15 discriminators successful (3 predicted all the same value, 1 had no genome-wide significant results, 1 distinguished real/simulated very easily, and 1 grouped all simulated selection strengths together)

- CEU: 11/15 discriminators successful (1 predicted all the same value, 2 had no genome-wide significant results, and 1 distinguished real/simulated very easily)

- CHB: 10/15 discriminators successful (2 predicted all the same value, 2 had no genome-wide significant results, 1 distinguished real/simulated very easily)

To identify genome-wide significant regions, we normalize output logit values to z-scores, then assume a standard normal distribution to convert to p-values. We use a conventional p-value threshold of 5×10^{-8} for genome-wide significance. If the difference in means between probabilities for real and simulated data was greater than 0.187, we classified the discriminator as distinguishing real/simulated very easily. For grouping all simulated selection strengths together, we used a threshold of -0.15 for the difference in means between neutral and $s = 0.10$ (indicating a reversal in the directionality of predictions). These thresholds were determined by inspection, but they could be recalibrated for different datasets. Below we discuss two example discriminators in detail, then explain how we used successfully trained discriminators in an ensemble method to predict natural selection.

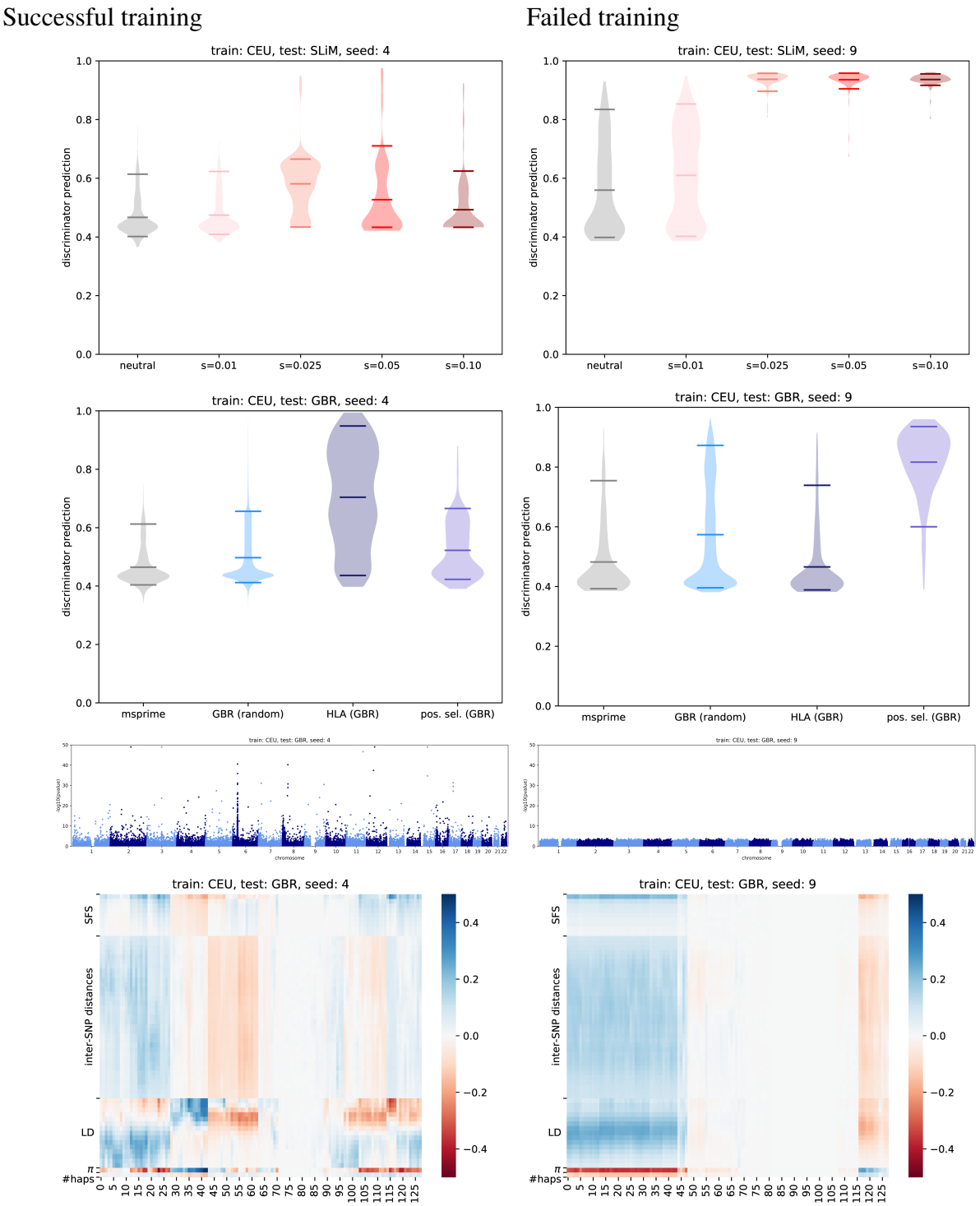


Figure 2: Two contrasting examples of CEU-trained discriminators with test data from GBR. Seed 4 on the left is an example of successful training and seed 9 on the right is an example of a discriminator that was discarded due to no genome-wide significant hits. Although seed 9 does identify simulated selected regions with high accuracy (first row), it does not identify known positively selected regions in real data (second and third rows), probably due to the fact that it is effectively computing only a simple summary statistic analogous to Tajima's D (fourth row). (first row) Performance of discriminators on simulated data under selection. (second row) Performance of discriminators on real genomic regions (both neutral and selected). The left discriminator is able to identify regions from the HLA (recognizing balancing selection) and the right discriminator is able to identify regions under positive selection. (third row) Genome-wide Manhattan plots of discriminator predictions (raw logit scores on the y-axis). On the left, many outlier regions are identified, whereas on the right no regions were genome-wide significant. (fourth row) Correlation heatmaps between discriminator hidden units (x-axis) and classical population genetics summary statistics (y-axis). The columns (hidden units) were clustered according to their similarity in terms of summary statistic correlation profiles.

3.2 Discriminator examples

3.2.1 Validation on simulated data

When running the data simulated with `SLiM` (of various selection strengths) through the trained discriminators, we typically see similar prediction patterns. Regions under strong selection are generally predicted as more "real" (higher discriminator output probability), and neutral regions or those under weak selection are predicted "fake". Although the network was not training with simulated selection data, the real data includes regions of selection, so this general pattern exposes the ability of most discriminators to see selection as realistic.

Most CEU-trained discriminators group neutral regions and those with selection strength $s = 0.01$ together, with probabilities around $0.4 - 0.5$. See Figure 2(first row) (seed 9), Figure S3(a) (seed 12) and Figure S4(a) for examples. Regions under stronger selection have much higher probabilities (around $0.8 - 1$). In Figure 2(first row) (seed 4) and Figure S3(a) (seed 13) we see the less common case of more confusion between different selection strengths, with high probabilities for a few regions under strong selection. YRI-trained discriminators behave similarly to those trained on CEU. Figure S2(a) shows two common examples of YRI-trained discriminators with very high output probabilities for regions under strong selection. For CHB-trained discriminators we often see even more separation between the distributions for different selection strengths. See Figure S5(a) for a few examples. Overall our simulated data results reveal that our initial hypothesis (that selected regions would have higher discriminator probabilities) holds for most discriminators.

3.2.2 Validation on known selected regions

To understand how the discriminator behaves when presented with real data from a test population, we produced predictions for regions known to be under selection (from [16]), regions from the HLA, and compared to predictions from the rest of the genome. We also included data simulated under the demography inferred by the `pg-gan` training run that produced the given discriminator. In theory this is the data which the discriminator thinks is most similar to real data.

Overall we find significant differences in the predictions for selected regions vs. neutral. For example, for the CEU-trained discriminator (seed 4) in Figure 2, the p-value for differences between mean probabilities is 2.575×10^{-5}. Of the 30 successfully trained discriminators, the p-value is significant (at the 0.05 confidence level) in all but one case. Although we do not use the HLA region to classify discriminators, we also note that frequently HLA predictions are also much higher than for neutral data. Simulated data (neutral) and neutral real data are almost always grouped

together, indicating a successful generator. In Figure 2(second row), we show the performance of two CEU-trained discriminators (seeds 4 and 9) on real data from GBR. Horizontal bars represent means and $0.05 - 0.95$ quantiles. Here we find differences in the discriminator's abilities to classify different types of regions. In the case of seed 4, there is a strong signal for HLA, and a weaker (but still significant) signal for regions under positive selection. Conversely, in the case of seed 9 the discriminator picks up on positive selection, but not HLA. We hypothesize that some discriminators are computing summary statistics related to balancing selection (which would apply more to the HLA) and some are computing summary statistics related to positive selection. See Figures S2(b), S3(b), S4(b), and S5(b) for further examples of higher probabilities for selected regions (for YRI-, CEU-, and CHB-trained discriminators).

For the same two CEU-trained discriminators in Figure 2, we performed a genome-wide scan of GBR regions (Figure 2, third row). In the case of seed 4 we see that the discriminator identifies outlier regions, including *LCT* (lactase) on chromosome 2 and *DUT* on chromosome 15. Overall we identify 510 putatively selected outlier regions at the 5×10^{-8} significance level. In the case of seed 9 the discriminator does not identify any regions as genome-wide significant, despite outputting relatively high probabilities for many known selected regions. This leads us to discard this discriminator, as it may be computing simple statistics of the regions that do not accurately capture the nuances of regions under various modes of selection. See Figures S2(c), S3(c), S4(c), and S5(c) for further Manhattan plot examples. Note the spike on chromosome 6 in the HLA region in Figure S4(c) (seed 0) and Figure S5(c) (seed 0).

3.2.3 Interpretability analysis

After performing correlations between the hidden units of each discriminator and traditional summary statistics (both calculated on the test data), we find some common patterns. Our hierarchical clustering approach shows extensive redundancy in the hidden units, which is to be expected given the dropout computations included in the last fully connected layers. We note that none of the correlation values are particularly high – the largest (in magnitude) are around ± 0.4. Frequently the highest correlations are with rare variants (first entries of the SFS), LD statistics, and π (pairwise heterozygosity). In Figure 2 (fourth row) we show two example heatmaps of correlation values between hidden values of the last layer and summary statistics. On the right (seed 4), the highest correlations are with the number of singletons and π. Although many hidden values are correlated with inter-SNP distances and LD statistics, we hypothesize that the network is computing something roughly equivalent to the difference between π and the number of singletons, which is close to Tajima's D (as the number of SNPs in each region is constrained to be the same, an exact computation of Tajima's D is not possible).

This may explain why the discriminator does not identify selected regions at the genome-wide significance level, as Tajima's D is not the most powerful predictor of selection. We also note that many nodes are not strongly correlated with any of these statistics – while this does not rule out their contribution, it is likely that fewer hidden units could express this particular network equally well.

On the left of Figure 2 (fourth row) (seed 4) there is more diversity in the hidden units, although we still see strong correlations with the number of singletons and π. LD also plays a bigger role, which could help this network identify balancing selection. See Figures S2(d), S3(d), S4(d), and S5(d) for heatmap examples in other populations.

To test the hypothesis that the seed 9 discriminator is mainly computing Tajima's D (Figure 2 (fourth row)), we computed the correlation between Tajima's D and the raw logit score. To compute Tajima's D, we first compute the median number of basepairs spanned by our $S = 36$ SNP regions (for GBR). This value is 7952 basepairs. For each region, we take the middle SNP (out of 36) and center a 7952 basepair region around it, then compute Tajima's D in this region. The results are shown in Figure 3, and the overall correlation is -0.381. This means that as Tajima's D decreases, logit scores (and therefore probabilities) generally increase. This makes sense as a negative Tajima's D is linked to selection.

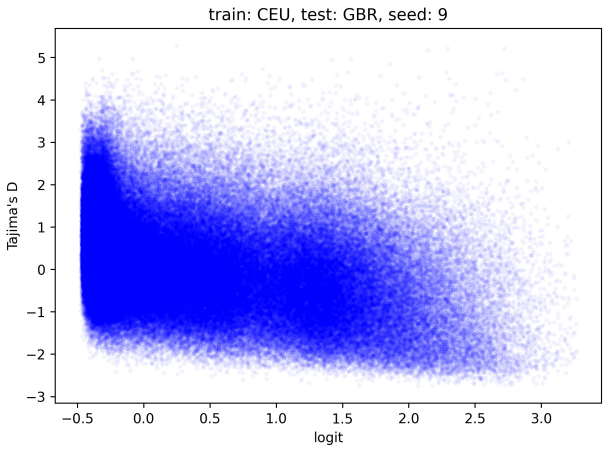


Figure 3: Relationship between raw logit output from the seed 9 CEU-trained discriminator and Tajima's D, for regions in the GBR genome. There is a negative correlation, which makes sense since a negative Tajima's D is linked to selection.

3.3 Ensemble results

For the successful discriminators of each training population, we average the probabilities for each test population to create ensemble results. We then convert the probabilities to logits, then z-scores, then p-values. For each p-value p

we plot $-\log_{10}(p)$ on the y-axis in Figure 4. We truncate the y-axis at 50 for visualization purposes, but many values are higher. Known selected regions are shown by gray bars. For these regions, the difference in mean p-values is highly significant (see p-values in Table 2). *HERC2* (related to pigmentation) is frequently identified in CEU-trained discriminators. We also find many additional genome-wide significant results, which are putative regions of natural selection. See Table 2 for the complete results.

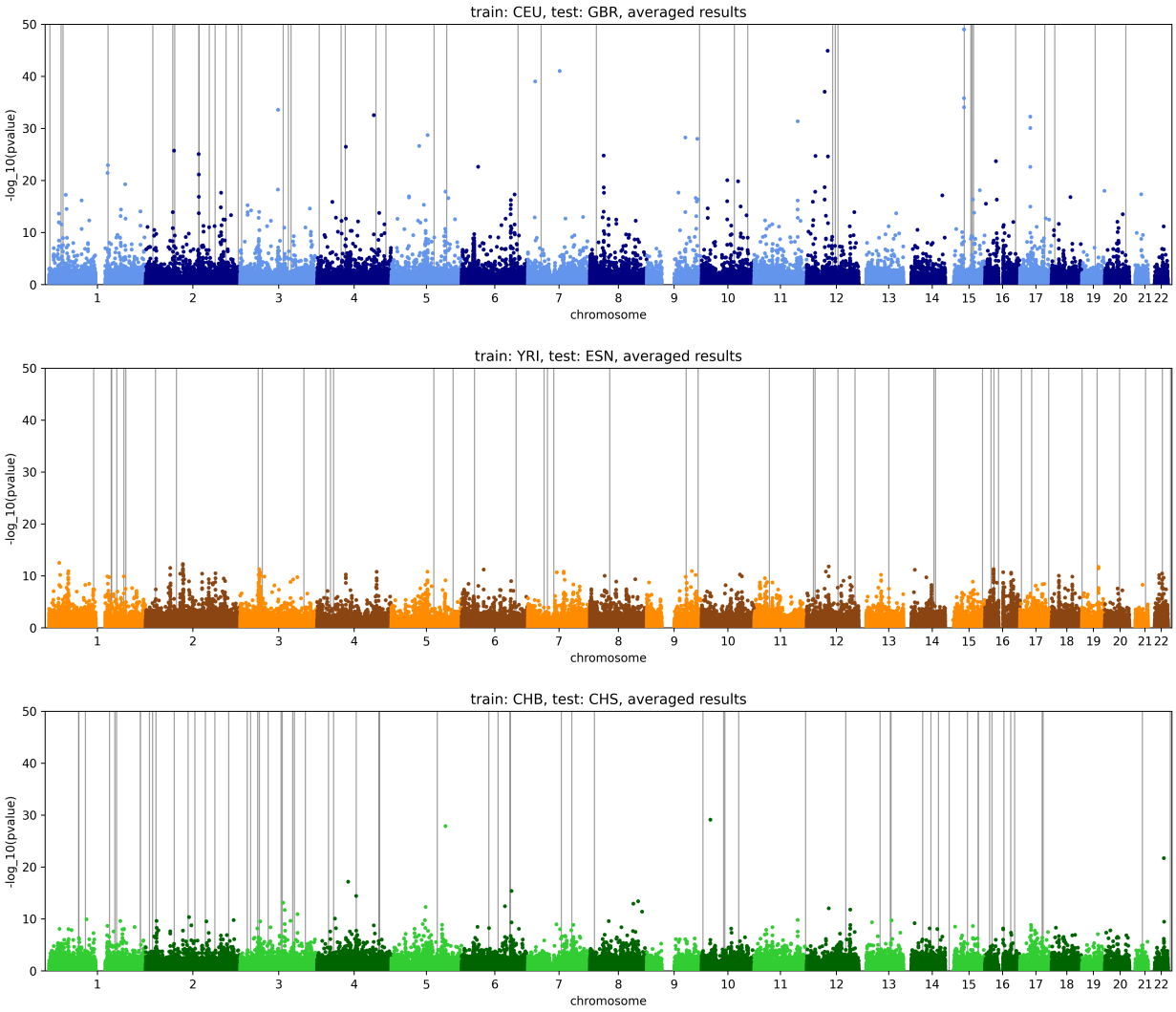


Figure 4: Ensemble results. Genome-wide selections scans for EUR/GBR (first row), AFR/ESN (second row), EAS/CHB (third row). In each case, the x-axis represents genomic position, the y-axis represents the $\log_{10}$ p-value, each point represents a 36-SNP window and grey vertical lines represent known selected regions from [16].

4 Discussion

In this work we have shown how a discriminator trained as part of a GAN algorithm can be used downstream to identify outlier regions in the human genome. These regions can be further analyzed as potential targets of natural selection.

Table 2: Ensemble results. For each population we report the mean predicted probabilities for regions that are selected vs. neutral (according to [16]), and the p-value for the difference in mean probabilities between these two sets of regions. We also report the fraction of all windows that are genome-wide significant in each analysis.

	EUR	AFR	EAS
# successful discriminators	11/15	9/15	10/15
mean neutral $D(Y_n)$	0.507	0.519	0.530
mean selection $D(Y_s)$	0.578	0.545	0.598
p-value for difference in means	1.70×10^{-37}	1.64×10^{-10}	6.82×10^{-57}
# significant regions	430	232	99
fraction significant	0.0015	0.0005	0.0004

Further, we show how the output of the discriminator can be interpreted in terms of known summary statistics. In particular, we find that rare variation, LD statistics, and pairwise heterozygosity are implicitly computed by hidden units in the discriminator network.

A major advantage of our approach is that we do not need to simulate thousands of selected regions across a high-dimensional space of selection parameters. By adding a few additional testing steps, we are able to quickly identify discriminators that have the ability to identify outlier regions. Training `pg-gan` is not instantaneous, but is much faster than simulating selection (usually on the order of 5-6 hours including simulating neutral training data).

A caveat of our approach is that the outlier regions identified by the discriminator cannot be directly compared to simulations under selection (as these were not used in training) and thus specific selection parameters. It is possible therefore that the regions we identify actually reflect different types of deviation from the generator model. For example, regions with very different mutation or recombination rates, or structural variation. However, we note that all approaches to detecting to selection are sensitive to these effects. Our results, like those of all scans should be treated as candidate regions requiring validation.

In order to apply our approach to other species, one important step is identifying a set of regions under positive selection (for testing). This could be bypassed with high-quality selection simulations, but these might be difficult to obtain depending on the species. Even for our work, the positive selection regions utilized from [16] are for YRI/CEU/CHB, but our testing data is from (similar but not identical) populations ESN/GBR/CHS. Future work could find more creative ways of incorporating a few selected regions (either real or simulated) into `pg-gan` training, in order to help the discriminator better identify particular regimes of selection.

Finally, our interpretability work is on a global scale, meaning that we analyze genome-wide patterns. Future work could examine why the discriminator makes predictions for particular regions, especially outliers with high probabilities.

Acknowledgements

SM is funded in part by NIH grant R15HG011528. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health or other funding sources.

References

[1] Jeffery Chan, Valerio Perrone, Jeffery Spence, Paul Jenkins, Sara Mathieson, and Yun Song. A likelihood-free inference framework for population genetic data using exchangeable neural networks. *Neural Information Processing Systems*, pages 8594–8605, February 2018.

[2] Lex Flagel, Yaniv Brandvain, and Daniel Schrieder. The unreasonable effectiveness of convolutional neural networks in population genetic inference. *Molecular Biology and Evolution*, 36(2):220–238, December 2018.

[3] Luis Torada, Lucrezia Lorenzon, Alice Beddis, Ulas Isildak, Linda Pattini, Sara Mathieson, and Matteo Fumagalli. Imagene: a convolutional neural network to quantify natural selection from genomic data. *BMC Bioinformatics*, 20(337):2560–2569, November 2019.

[4] Xinghu Qin, Charleston WK Chiang, and Oscar E Gaggiotti. Deciphering signatures of natural selection via deep learning. *Briefings in Bioinformatics*, 23(5):bbac354, 2022.

[5] Graham Gower, Pablo Iáñez Picazo, Matteo Fumagalli, and Fernando Racimo. Detecting adaptive introgression in human evolution using convolutional neural networks. *Elife*, 10:e64669, 2021.

[6] Ulas Isildak, Alessandro Stella, and Matteo Fumagalli. Distinguishing between recent balancing selection and incomplete sweep using deep neural networks. *Molecular Ecology Resources*, 21(8):2706–2718, 2021.

[7] Théophile Sanchez, Erik Madison Bray, Pierre Jobic, Jérémy Guez, Anne-Catherine Letournel, Guillaume Charpiat, Jean Cury, and Flora Jay. dnadna: a deep learning framework for population genetics inference. *Bioinformatics*, 2022.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[9] Zhanpeng Wang, Jiaping Wang, Michael Kourakos, Nhung Hoang, Hyong Hark Lee, Iain Mathieson, and Sara Mathieson. Automatic inference of demographic parameters using generative adversarial networks. *Molecular ecology resources*, 21(8):2689–2705, 2021.

[10] Burak Yelmen, Aurélien Decelle, Linda Ongaro, Davide Marnetto, Corentin Tallec, Francesco Montinaro, Cyril Furtlehner, Luca Pagani, and Flora Jay. Creating artificial human genomes using generative neural networks. *PLoS genetics*, 17(2):e1009303, 2021.

[11] William W Booker, Dylan D Ray, and Daniel R Schrider. This population doesn't exist: learning the distribution of evolutionary histories with generative adversarial networks. *bioRxiv*, 2022.

[12] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[13] Jeffrey R Adrion, Christopher B Cole, Noah Dukler, Jared G Galloway, Ariella L Gladstein, Graham Gower, Christopher C Kyriazis, Aaron P Ragsdale, Georgia Tsambos, Franz Baumdicker, et al. A community-maintained standard library of population genetic models. *Elife*, 9, 2020.

[14] Benjamin C Haller and Philipp W Messer. Slim 3: forward genetic simulations beyond the wright–fisher model. *Molecular biology and evolution*, 36(3):632–637, 2019.

[15] J. M. Akey. Constructing genomic maps of positive selection in humans: where do we go from here? *Genome Res*, 19(5):711–22, 2009. ISSN 1088-9051 (Print) 1088-9051 (Linking). doi: 10.1101/gr.086652.108. URL `https://www.ncbi.nlm.nih.gov/pubmed/19411596`.

[16] Sharon R Grossman, Kristian G Andersen, Ilya Shlyakhter, Shervin Tabrizi, Sarah Winnicki, Angela Yen, Daniel J Park, Dustin Griesemer, Elinor K Karlsson, Sunny H Wong, et al. Identifying recent adaptations in large-scale genomic data. *Cell*, 152(4):703–713, 2013.

[17] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[18] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[19] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital signal processing*, 73:1–15, 2018.

[20] J. Nait Saada, A. Hu, and P. F. Palamara. Inference of pairwise coalescence times and allele ages using deep neural networks. *NeurIPS Workshop of Learning Meaningful Representations of Life (LMRL)*, 2021.

[21] Jerome Kelleher, Alison M Etheridge, and Gilean McVean. Efficient coalescent simulation and genealogical analysis for large sample sizes. *PLoS computational biology*, 12(5):e1004842, 2016.

[22] Franz Baumdicker, Gertjan Bisschop, Daniel Goldstein, Graham Gower, Aaron P Ragsdale, Georgia Tsambos, Sha Zhu, Bjarki Eldon, E Castedo Ellerman, Jared G Galloway, et al. Efficient ancestry and mutation simulation with msprime 1.0. *Genetics*, 220(3):iyab229, 2022.

[23] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[24] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020. URL `https://developer.nvidia.com/cuda-toolkit`.

[25] 1000 Genomes Project Consortium et al. A global reference for human genetic variation. *Nature*, 526(7571):68, 2015.

[26] International HapMap Consortium et al. A second generation human haplotype map of over 3.1 million snps. *Nature*, 449(7164):851, 2007.

[27] Benjamin C Haller, Jared Galloway, Jerome Kelleher, Philipp W Messer, and Peter L Ralph. Tree-sequence recording in slim opens new horizons for forward-time simulation of whole genomes. *Molecular ecology resources*, 19(2):552–566, 2019.

[28] Diogo Meyer and Glenys Thomson. How selection shapes variation of the human major histocompatibility complex: a review. *Annals of human genetics*, 65(1):1–26, 2001.

[29] Alicia Sanchez-Mazas and Diogo Meyer. The relevance of hla sequencing in population genetics studies. *Journal of Immunology Research*, 2014, 2014.

Supplementary Material

Interpreting Generative Adversarial Architectures to Infer Natural Selection

Rebecca Riley[1], Iain Mathieson[2], and Sara Mathieson[1,†]

[1] Department of Computer Science, Haverford College, Haverford, PA
[2] Department of Genetics, University of Pennsylvania, Philadelphia, PA
[†] Corresponding author: Sara Mathieson, smathieson@haverford.edu

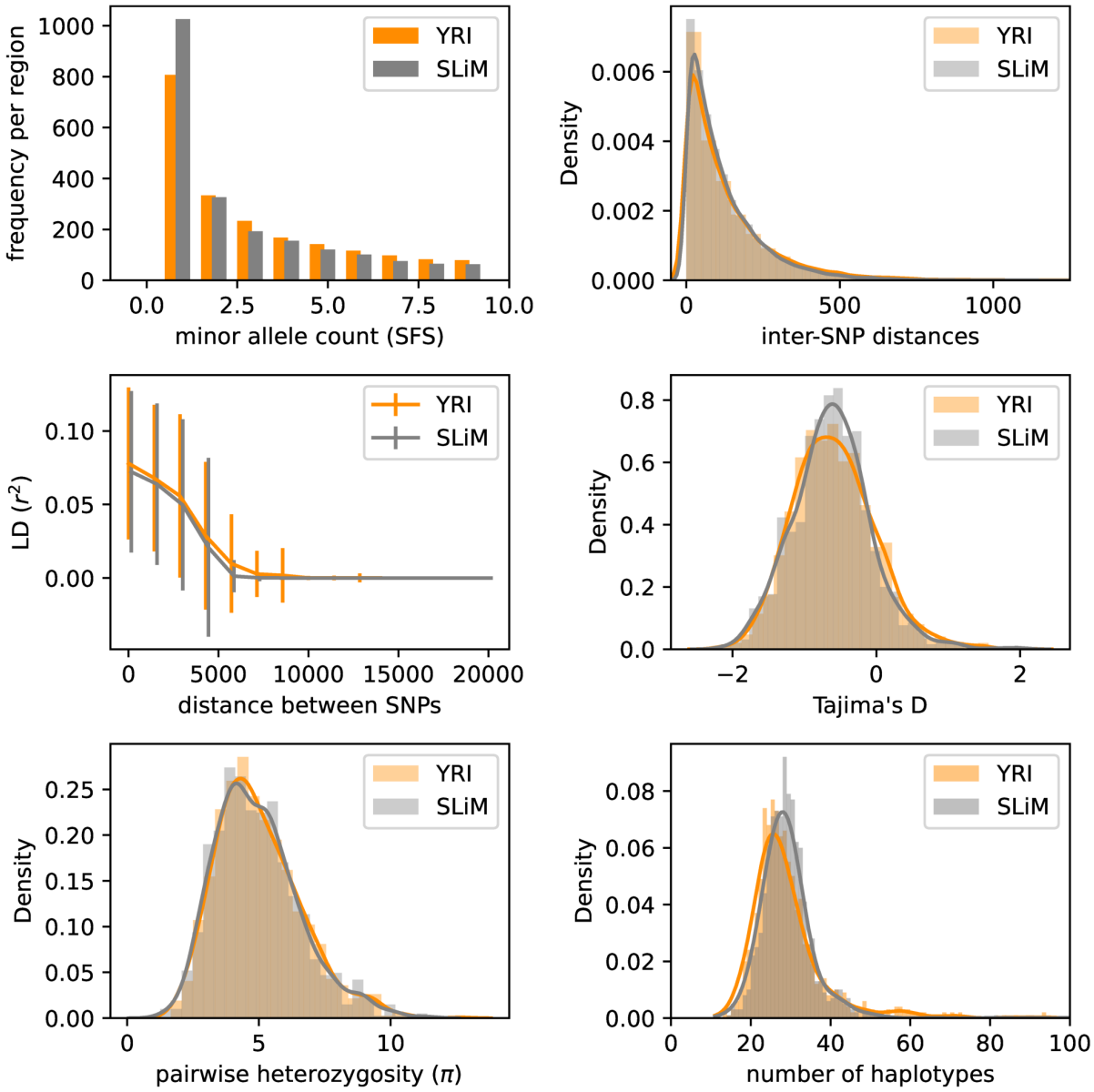
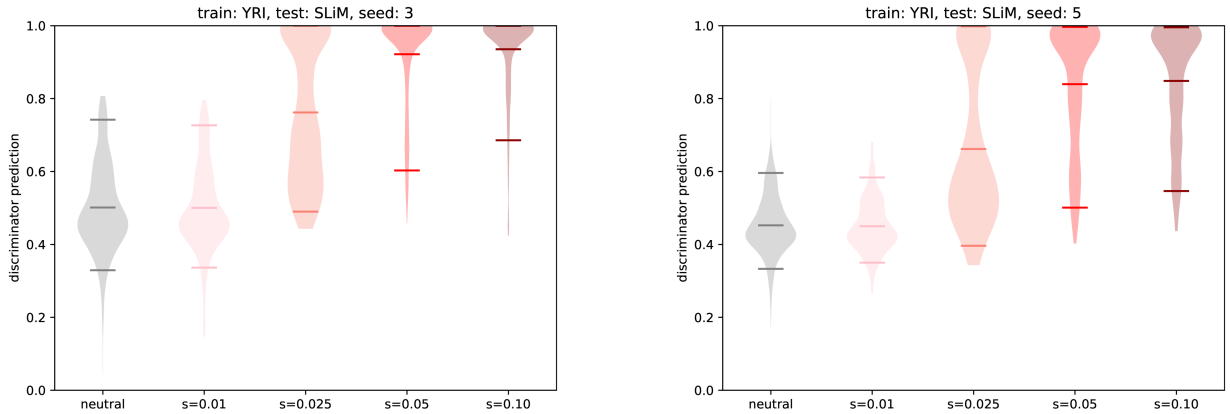
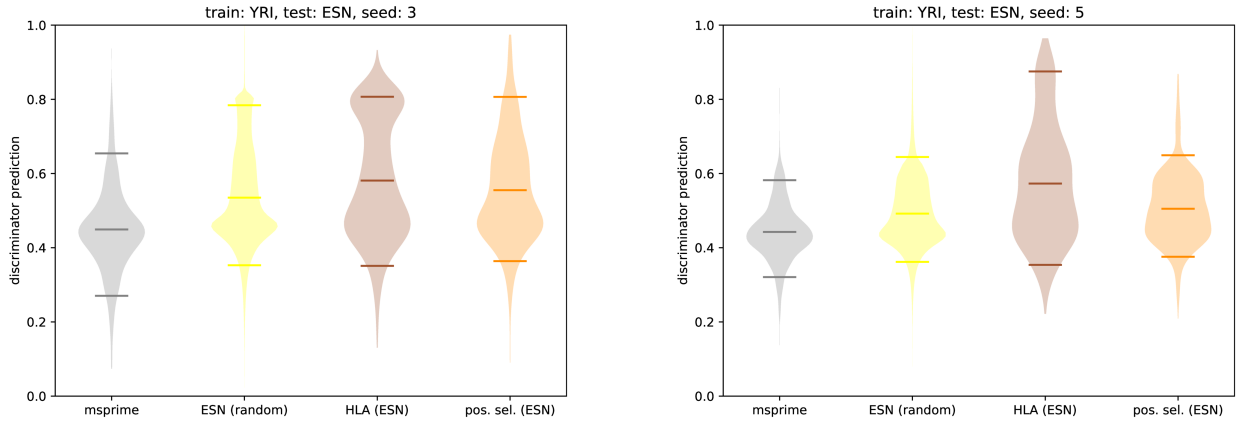


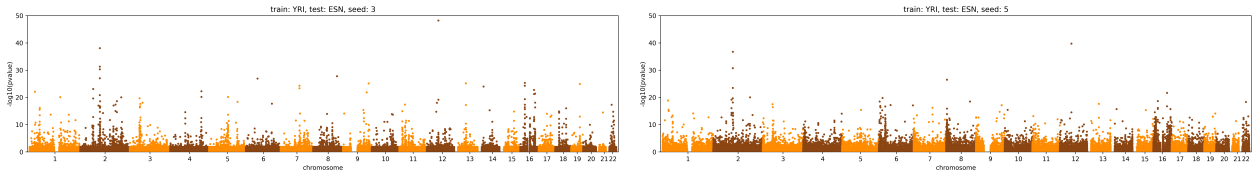
Figure S1: Summary statistic comparison of real YRI training data to neutral data simulated under `SLiM` with parameters inferred by `pg-gan`. The number of regions used for comparison was 1000.



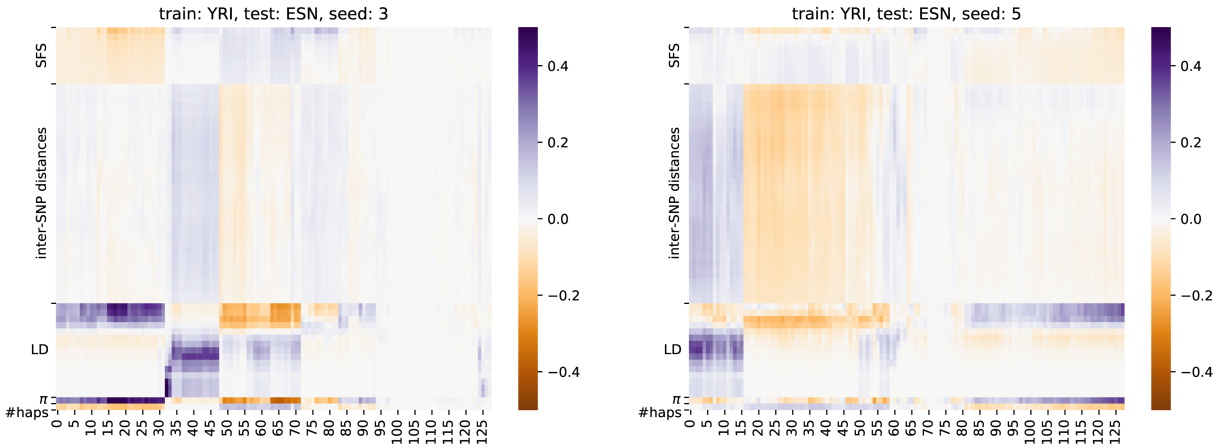
(a) Performance of discriminators on simulated data under selection.



(b) Performance of discriminators on real genomic regions (both neutral and selected).

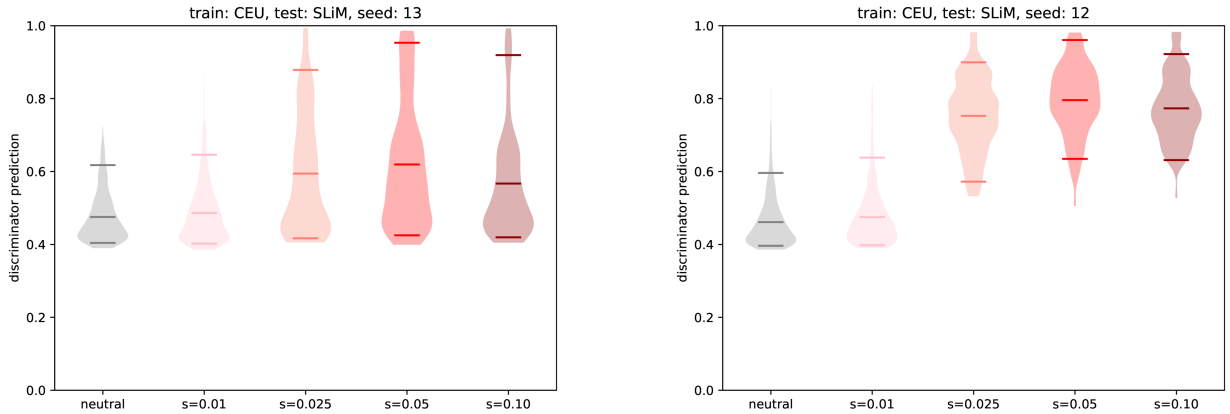


(c) Genome-wide Manhattan plots of discriminator predictions (raw logit scores).

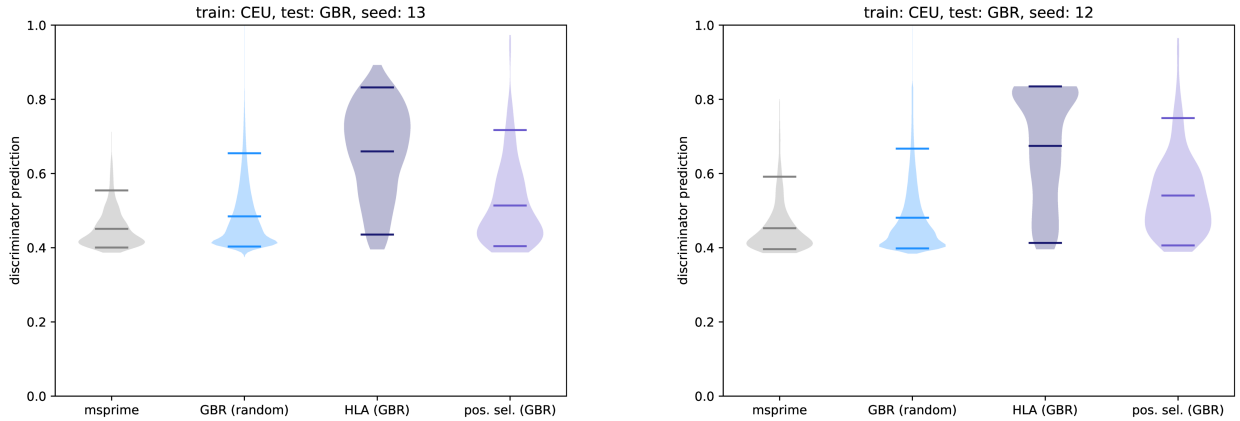


(d) Correlation heatmaps between discriminator hidden units (x-axis) and classic population genetics summary statistics (y-axis).

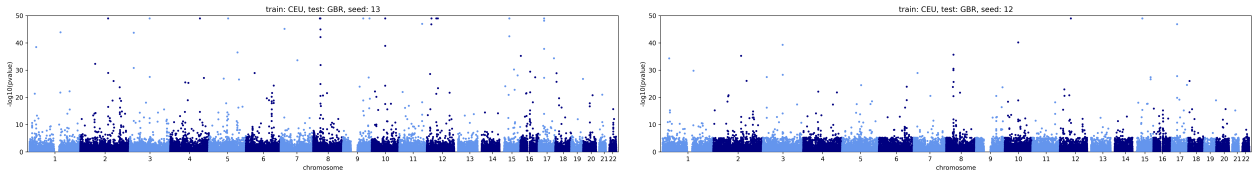
Figure S2: Two examples of YRI-trained discriminators with test data from ESN (seed 3 on the left and seed 5 on the right).



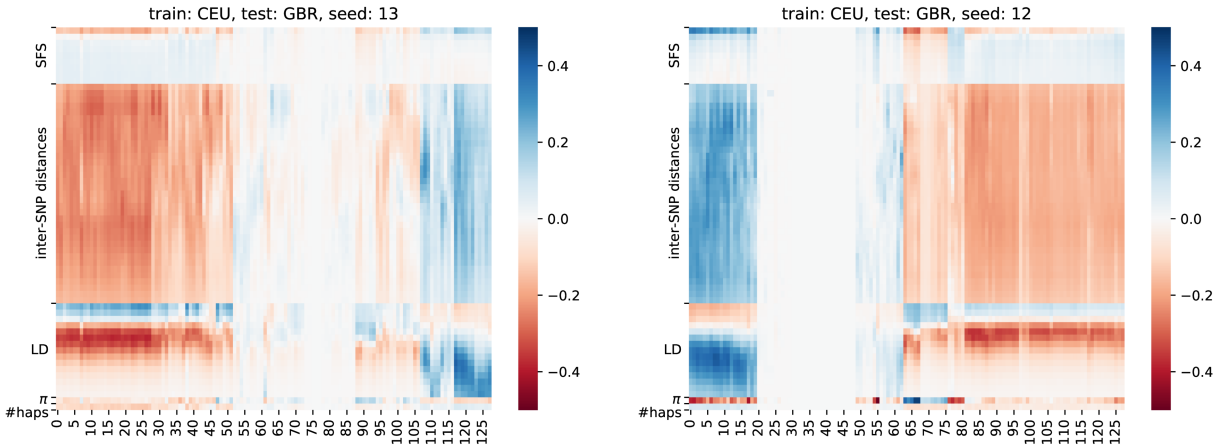
(a) Performance of discriminators on simulated data under selection.



(b) Performance of discriminators on real genomic regions (both neutral and selected).

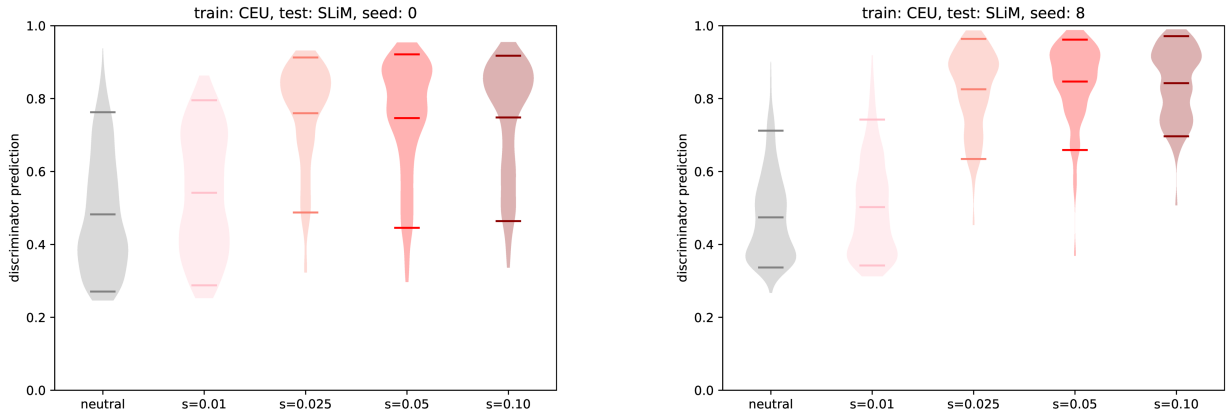


(c) Genome-wide Manhattan plots of discriminator predictions (raw logit scores).

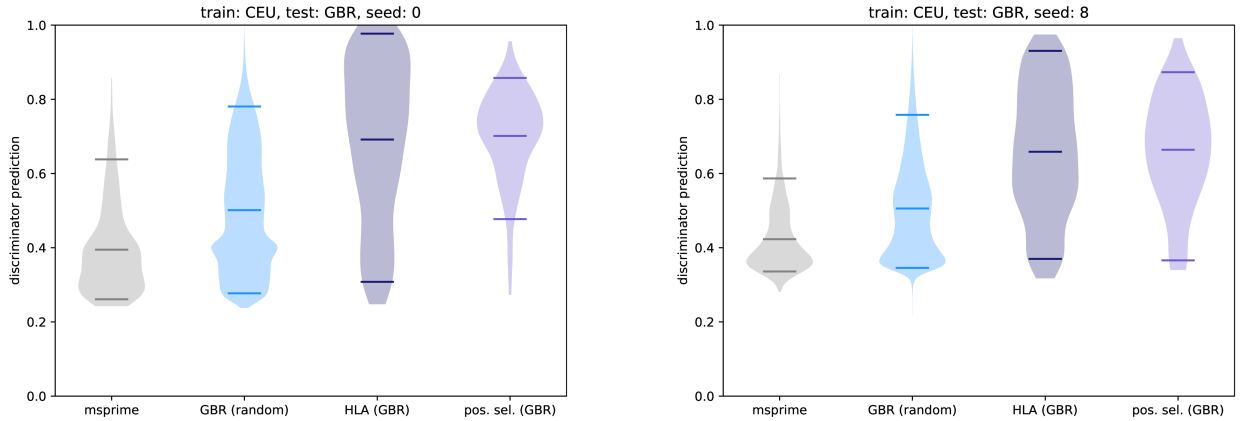


(d) Correlation heatmaps between discriminator hidden units (x-axis) and classic population genetics summary statistics (y-axis).

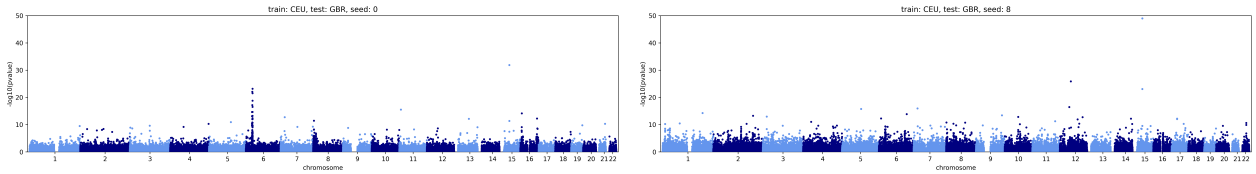
Figure S3: Two examples of CEU-trained discriminators with test data from GBR (seed 13 on the left and seed 12 on the right).



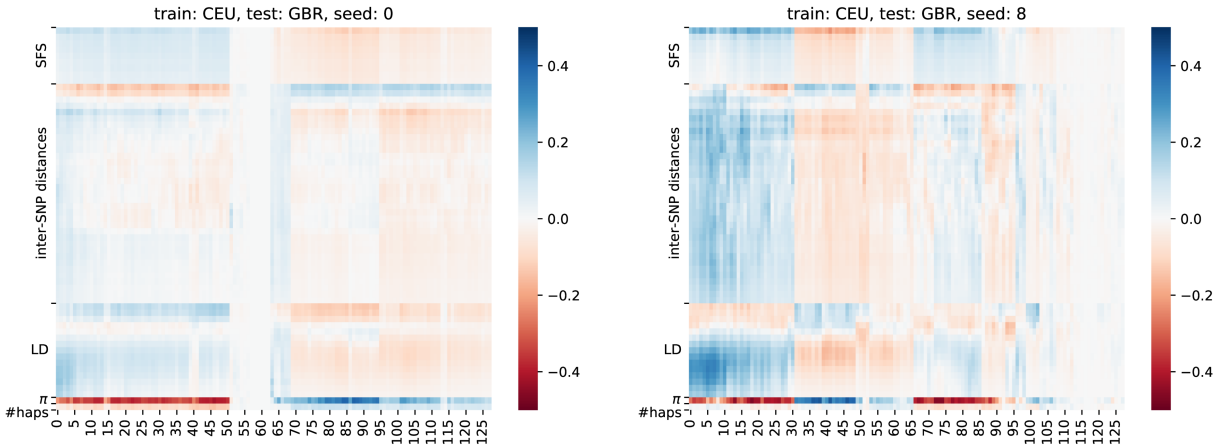
(a) Performance of discriminators on simulated data under selection.



(b) Performance of discriminators on real genomic regions (both neutral and selected).

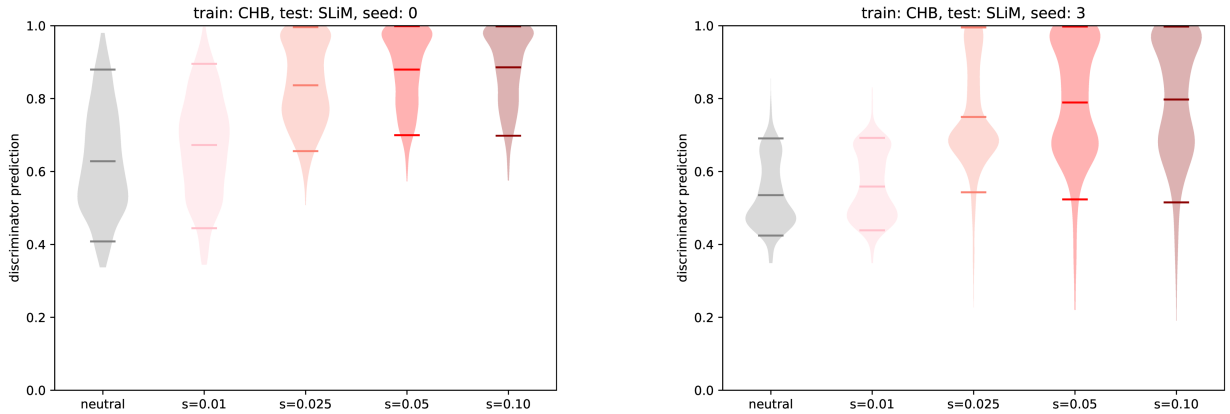


(c) Genome-wide Manhattan plots of discriminator predictions (raw logit scores).

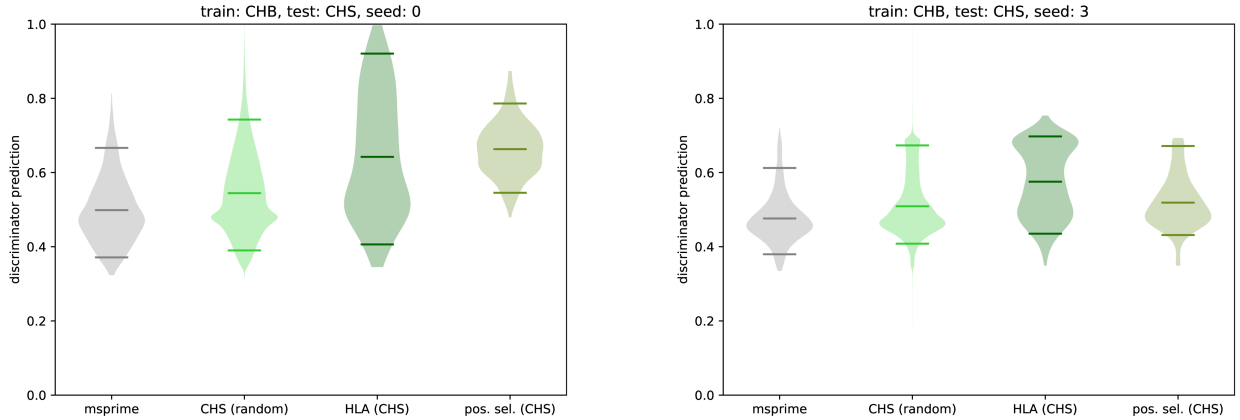


(d) Correlation heatmaps between discriminator hidden units (x-axis) and classic population genetics summary statistics (y-axis).

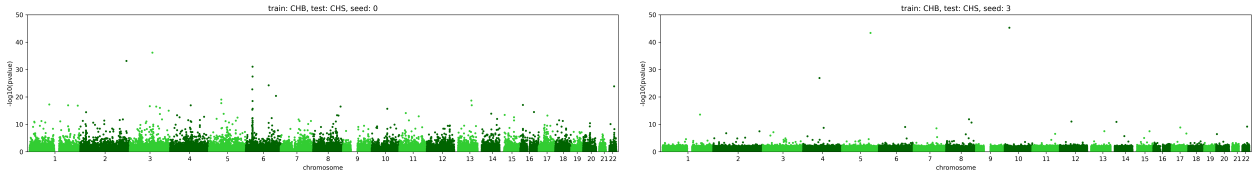
Figure S4: Two additional examples of CEU-trained discriminators with test data from GBR (seed 0 on the left and seed 8 on the right).



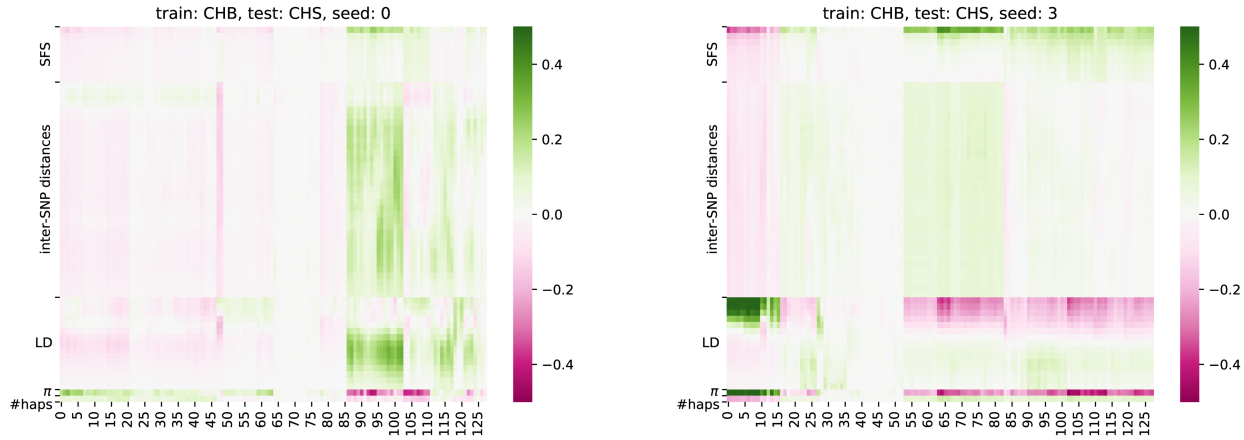
(a) Performance of discriminators on simulated data under selection. Note that in the case of CHB, default `pg-gan` parameters were used for the exponential growth demographic model.



(b) Performance of discriminators on real genomic regions (both neutral and selected).



(c) Genome-wide Manhattan plots of discriminator predictions (raw logit scores).



(d) Correlation heatmaps between discriminator hidden units (x-axis) and classic population genetics summary statistics (y-axis).

Figure S5: Two examples of CHB-trained discriminators with test data from CHS (seed 0 on the left and seed 3 on the right).