A Likelihood-Free Inference Framework for Population Genetic Data using Exchangeable Neural Networks

Jeffrey Chan [1] Valerio Perrone [2] Jeffrey P. Spence [1] Paul A. Jenkins [2]
Sara Mathieson [3] Yun S. Song [1 4]

Abstract

Inference for population genetics models is hindered by computationally intractable likelihoods. While this issue is tackled by likelihood-free methods, these approaches typically rely on hand-crafted summary statistics of the data. In complex settings, designing and selecting suitable summary statistics is problematic and results are very sensitive to such choices. In this paper, we learn the first exchangeable feature representation for population genetic data to work directly with genotype data. This is achieved by means of a novel Bayesian likelihood-free inference framework, where a permutation-invariant convolutional neural network learns the inverse functional relationship from the data to the posterior. We leverage access to scientific simulators to learn such likelihood-free function mappings, and establish a general framework for inference in a variety of simulation-based tasks. We demonstrate the power of our method on the recombination hotspot testing problem, outperforming the state-of-the-art.

1. Introduction

Statistical inference in complex population genetics models is challenging, as the likelihood is often both analytically and computationally intractable. These models are usually based on the coalescent (Kingman, 1982), a stochastic process describing the distribution over genealogies of a random sample of chromosomes from a large population. Unfortunately, standard coalescent-based likelihoods require integrating over a large set of correlated high-dimensional combinatorial objects, rendering classical inferential techniques inapplicable. This limitation can be overcome by likelihood-free methods such as Approximate Bayesian Computation (ABC) (Beaumont et al., 2002) and deep learning (Sheehan & Song, 2016). These approaches leverage scientific simulators to draw samples from the generative model, and reduce population genetic data to a suite of summary statistics prior to performing inference. However, hand-engineered feature representations typically are not statistically sufficient for the parameter of interest, leading to loss in accuracy. In addition, these statistics are often based on the intuition of the user, need to be modified for each new task, and, in the case of ABC, are not amenable to hyperparameter optimization strategies since the quality of the approximation is unknown.

Deep learning offers the possibility to avoid the need for hand-designed summary statistics in population genetic inference and work directly with genotype data. The goal of this work is to develop a scalable general-purpose inference framework for raw genetic data, without the need for summary statistics. We achieve this by designing a neural network which exploits the exchangeability in the underlying data to learn feature representations that can approximate the posterior accurately.

As a concrete example, we focus on the problem of recombination hotspot testing. Recombination is a biological process of fundamental importance, in which the reciprocal exchange of DNA during cell division creates new combinations of genetic variants. Experiments have shown that some species exhibit *recombination hotspots*, that is, short segments of the genome with high intensity recombination rates (Petes, 2001). The task of recombination hotspot testing is to predict the location of recombination hotspots given genetic polymorphism data. Accurately localizing recombination hotspots would illuminate the biological mechanism that underlies recombination, and could help geneticists map the alleles causing genetic diseases (Hey, 2004). We demonstrate in experiments that we achieve state-of-the-art performance on the hotspot detection problem.

Our contributions focus on addressing major inferential challenges of complex population genetic inference. In Section 2 we review relevant lines of work in both the fields of machine learning and population genetics. In Section 3

[1] University of California, Berkeley [2] University of Warwick [3] Swarthmore College [4] Chan Zuckerberg Biohub. Correspondence to: Yun S. Song <yss@berkeley.edu>, Sara Mathieson <smathieson@cs.swarthmore.edu>.

we propose a scalable Bayesian likelihood-free inference framework for exchangeable data, which may be broadly applicable to many population genetic problems as well as more general simulator-based machine learning tasks. The application to population genetics is detailed in Section 4. In particular, we show how this allows for direct inference on the raw population genetic data, bypassing the need for *ad hoc* summary statistics. In Section 5 we run experiments to validate our method and demonstrate state-of-the-art performance in the hotspot detection problem.

2. Related Work

Likelihood-free methods like ABC have been widely employed in population genetics (Beaumont et al., 2002; Boitard et al., 2016; Wegmann et al., 2009; Sousa et al., 2009). In ABC the parameter of interest is simulated from its prior distribution, and data are subsequently simulated from the generative model and reduced to a pre-chosen set of summary statistics. These statistics are compared to the summary statistics of the real data, and the simulated parameter is weighted according to the similarity of the statistics to derive an empirical estimate of the posterior distribution. However, choosing summary statistics for ABC is challenging because there is a trade-off between loss of sufficiency and computational tractability. In addition, there is no direct way to evaluate the accuracy of the approximation.

Other likelihood-free approaches have emerged from the machine learning community and have been applied to population genetics, such as support vector machines (SVMs) (Schrider & Kern, 2015; Pavlidis et al., 2010), single-layer neural networks (Blum & François, 2010), and deep learning (Sheehan & Song, 2016). The connection between likelihood-free Bayesian inference and neural networks has also been studied previously by Jiang et al. (2015) and Papamakarios & Murray (2016). An attractive property of these methods is that, unlike ABC, they can be applied to multiple datasets without repeating the training process, which is commonly referred to as amortized inference. However, current practice in population genetics collapses the data to a set of summary statistics before passing it through the machine learning models. Therefore, the performance still rests on the ability to laboriously hand-engineer informative statistics, and must be repeated from scratch for each new problem setting.

The inferential accuracy and scalability of these methods can be improved by exploiting symmetries in the input data. Permutation-invariant models have been previously studied in machine learning for SVMs (Shivaswamy & Jebara, 2006) and, recently, gained a surge of interest in the deep learning literature. Recent work on designing architectures for exchangeable data include Ravanbakhsh et al. (2016), Guttenberg et al. (2016), and Zaheer et al. (2017), which exploit parameter sharing to encode invariances. To our knowledge, no prior work has been done on learning feature representations for exchangeable population genetic data.

We demonstrate these ideas on the problem of recombination hotspot testing. To this end, several methods have been developed (Fearnhead, 2006; Li et al., 2006; Wang & Rannala, 2009). However, none of these are scalable to the whole genome, with the exception of `LDhot` (Auton et al., 2014; Wall & Stevison, 2016), so we limit our comparison to this latter method. `LDhot` relies on a composite likelihood, which can be seen as an approximate likelihood for summaries of the data. It can be computed only for a restricted set of models (i.e., an unstructured population with piecewise constant population size), is unable to capture dependencies beyond those summaries, and scales at least cubically with the number of DNA sequences. The method we propose in this paper scales linearly in the number of sequences while using raw genetic data directly.

3. Methodology

In this section we propose a flexible framework to address the shortcomings of current likelihood-free methods. Although motivated by population genetics, we first lay out the ideas that generalize beyond this application. We describe the exchangeable representation in Section 3.1 and the training algorithm in Section 3.2, which are combined into a general likelihood-free inference framework in Section 3.3. The statistical properties of the method are studied in Section 3.4.

3.1. Feature Representation for Exchangeable Data

Population genetic datapoints $\mathbf{x}^{(i)}$ typically take the form of a binary matrix, where rows correspond to individuals and columns indicate the presence of a Single Nucleotide Polymorphism (SNP), namely a nucleotide variation at a given location of the DNA. For unstructured populations the order of individuals carries no information, hence the rows are exchangeable. More generally, given data $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)})$ where $\mathbf{x}^{(i)} \in \mathbb{R}^{n \times d}$ and $\mathbf{x}^{(i)} := (x_1^{(i)}, \ldots, x_n^{(i)}) \sim \mathbb{P}(\mathbf{x} \mid \theta^{(i)})$, we call $\mathbf{X}$ *exchangeably-structured* if, for every i,

$$\mathbb{P}\Big(x_1^{(i)}, \ldots, x_n^{(i)} \mid \theta^{(i)}\Big) = \mathbb{P}\Big(x_{\sigma(1)}^{(i)}, \ldots, x_{\sigma(n)}^{(i)} \mid \theta^{(i)}\Big),$$

for all permutations σ of the indices $\{1, \ldots, n\}$.

To obtain an exchangeable feature representation of genotype data, we proceed as follows. Let $\Phi : \mathbb{R}^d \to \mathbb{R}^{d_1}$ be a feature mapping. We apply a symmetric function $g : \mathbb{R}^{n \times d_1} \to \mathbb{R}^{d_2}$ to the feature mapped datapoint to obtain $g\big(\Phi(x_1^{(i)}), \ldots, \Phi(x_n^{(i)})\big)$, a feature representation of the exchangeably-structured data. This representation is very general and can be adapted to various machine learning settings. For example, Φ could be some *a priori* fixed feature mapping (e.g. a kernel or summary statistics) in which case g should be chosen such that the resulting feature representation remains informative. More commonly, the mapping Φ needs to be learned (such as in kernel logistic regression or a deep neural network), hence we choose some fixed g such that subgradients can be backpropagated through g to Φ. Some examples of such a function g include the element-wise sum, element-wise max, lexicographical sort, or higher-order moments. Throughout the paper, we choose to parameterize Φ with a neural network and choose g to be the element-wise max function, such that $g_j = \max\left(\Phi(x_1^{(i)})_j, \ldots, \Phi(x_n^{(i)})_j\right)$. A variant of this representation is proposed by Ravanbakhsh et al. (2016) and Zaheer et al. (2017).

This embedding of exchangeably-structured data into a vector space is suitable for many tasks such as regression or clustering. We focus on inference in which the objective is to learn the function $f : \mathbb{R}^{n\times d} \to \mathcal{P}_\Theta$, where Θ is the space of all parameters θ and $\mathcal{P}$ is the space of all probability distributions on Θ. Endowed with our exchangeable feature representation, a function $h : \mathbb{R}^{d_2} \to \mathcal{P}_\Theta$ can be composed with our symmetric mapping to get $f := (h \circ g)\left(\Phi(x_1^{(i)}), \ldots, \Phi(x_n^{(i)})\right)$. For simplicity, throughout the rest of the paper we focus on binary classification where $\theta \in \{0, 1\}$, so that $\mathcal{P}_\theta$ can be parameterized by $\mathbb{P}(\theta = 1 \mid \mathbf{x}^{(i)}, \phi)$, where ϕ are nuisance parameters and h is parameterized as a neural network such that both h and Φ can be learned via backpropagation with a cross entropy loss. Specifically, we will apply this construction to infer the presence of recombination hotspots, indicated by the parameter θ. The posterior $\mathbb{P}(\theta = 1 \mid \mathbf{x}^{(i)}, \phi)$ is estimated by a soft max application so that the output is defined on $[0, 1]$. This exchangeable representation has many advantages. While it could be argued that flexible machine learning models could learn the structured exchangeability of the data, encoding exchangeability explicitly allows for faster per-iteration computation and improved learning efficiency, since data augmentation for exchangeability scales as $O(n!)$. Enforcing exchangeability implicitly reduces the size of the input space from $\mathbb{R}^{n\times d}$ to the quotient space $\mathbb{R}^{n\times d}/S_n$, where S_n is the symmetric group on n elements. A factorial reduction in input size leads to much more tractable inference for large n. In addition, choices of g where d_2 is independent of n (e.g., element-wise operations with output dimension independent of n) allows for a representation which is robust to differing number of exchangeable variables between train and test time. This property is particularly desirable to construct feature representations of fixed dimension even with missing data.

3.2. Simulation-on-the-fly

In statistical decision theory, the Bayes risk for prior $\pi(\theta)$ is defined as $R_\pi^* = \inf_T \mathbb{E}_\mathbf{x}\mathbb{E}_{\theta\sim\pi}[l(\theta, T(\mathbf{x})]$, with l being the loss function and T an estimator. The excess risk over the Bayes risk resulting from an algorithm A with model class $\mathcal{F}$ can be decomposed as

$$R_\pi(\tilde{f}_A) - R_\pi^* = \underbrace{\left(R_\pi(\tilde{f}_A) - R_\pi(\hat{f})\right)}_{\text{optimization error}} + \underbrace{\left(R_\pi(\hat{f}) - \inf_{f\in\mathcal{F}} R_\pi(f)\right)}_{\text{estimation error}} + \underbrace{\left(\inf_{f\in\mathcal{F}} R_\pi(f) - R_\pi^*\right)}_{\text{approximation error}},$$

where $\tilde{f}_A$ and $\hat{f}$ are the function obtained via algorithm A and the empirical risk minimizer, respectively. The terms on the right hand side are referred to as the optimization, estimation, and approximation errors, respectively. Often the goal of statistical decision theory is to minimize the excess risk, motivating algorithmic choices to control the three sources of error. For example, with supervised learning, overfitting is a result of large estimation error. Typically, for a sufficiently expressive neural network optimized via stochastic optimization techniques, the excess risk is dominated by optimization and estimation errors.

When we have access to scientific simulators, the amount of training data available is limited only by the amount of computational time available for simulation, so we propose simulating each training datapoint afresh such that there is exactly one epoch over the training data. We refer to this as *simulation-on-the-fly*. A similar setting is commonly used in the reinforcement learning literature, a key to recent success of deep reinforcement learning in applications such as games (Silver et al., 2016; 2017), though it is rarely utilized in supervised learning since access to simulators is usually unavailable. In this setting, the algorithm is run for many iterations until the estimation error is sufficiently small, eliminating the pitfalls of overfitting. With fixed training data, additional iterations after the first epoch are not guaranteed to further minimize the estimation error. Furthermore, simulation-on-the-fly guarantees $R_\pi(\hat{f}) \approx \inf_{f\in\mathcal{F}} R_\pi(f)$, and given that $\inf_{f\in\mathcal{F}} R_\pi(f) \approx R_\pi^*$ by the Universal Approximation Theorem (Cybenko, 1989), we can conclude that $R_\pi(\hat{f}) \approx R_\pi^*$. The risk surface is much smoother than that for fixed training sets (shown empirically in Section 5). This reduces the number of poor local minima and, consequently, the optimization error.

An alternative viewpoint of the simulation-on-the-fly paradigm from the lens of stochastic optimization is to compare the gradients of the two training procedures when $\mathcal{A}$ is restricted to first-order stochastic approximation algorithms. In order to make explicit the optimization algorithm at hand, we parameterize the model class $\mathcal{F}$ by $w \in \mathcal{W}$ such that

$f_w \in \mathcal{F}$ if and only if $w \in \mathcal{W}$ where $\mathcal{W}$ is the space of all possible neural network weights for a fixed architecture. Denote the empirical risk with respect to the prior $\pi(\theta)$ as $\hat{R}_\pi$. In the simulation-on-the-fly regime, the t-th iteration approximates the gradient of the population risk $\nabla_{w_t} R_\pi(f_{w_t})$ at w_t by the unbiased random vector

$$g_{\text{sim}}(w_t) = \nabla_{w_t} R_\pi(f_{w_t}) + \xi_t, \qquad (1)$$

where ξ_t is a random vector such that $\mathbb{E}(\xi_t) = 0$ and $\mathbb{E}(\xi_t \mid g_{\text{sim}}(w_1), \ldots, g_{\text{sim}}(w_{t-1})) = 0$. On the other hand, for the fixed training set regime, the t-th iteration of the algorithm approximates the empirical risk gradient $\nabla_{w_t} \hat{R}_\pi(f_{w_t})$ at w_t by the unbiased random vector

$$g_{\text{fixed}}(w_t) = \nabla_{w_t} \hat{R}_\pi(f_{w_t}) + \xi_t, \qquad (2)$$

where once again $\mathbb{E}(\xi_t) = 0$ and $\mathbb{E}(\xi_t \mid g_{\text{fixed}}(w_1), \ldots, g_{\text{fixed}}(w_{t-1})) = 0$. A key point is that while $g_{\text{fixed}}(w_t)$ is unbiased with respect to $\nabla_{w_t} \hat{R}_\pi(f_{w_t})$, it is biased with respect to $\nabla_{w_t} R_\pi(f_{w_t})$. Using the formulation in (2), the fixed training data setting performs stochastic optimization on the empirical risk $\hat{R}_\pi$ and converges to the empirical Bayes risk $\hat{R}_\pi^*$ for decaying learning rate and suitably expressive $\mathcal{F}$. On the other hand, simulation-on-the-fly in (1) performs stochastic optimization directly on the population Bayes risk R_π, circumventing the bias incurred from using the empirical Bayes risk as a proxy for the population Bayes risk.

3.3. Likelihood-Free Inference Framework

With an exchangeable feature representation and an optimization procedure in hand, we can now combine these ingredients into an inference scheme. Let $\mathbf{x}$, θ, ϕ, and γ be the observed data, the latent parameter of interest, the nuisance parameters, and the prior hyperparameters, respectively. The latent parameter θ can be inferred by drawing samples from the prior distribution $\theta^{(i)}, \phi^{(i)} \sim \pi(\theta, \phi \mid \gamma)$ and from the density $\mathbf{x}^{(i)} \sim P(\mathbf{x} \mid \theta^{(i)}, \gamma, \phi^{(i)})$, while stochastic optimization under the simulation-on-the-fly paradigm fits $\tilde{f}_A(\mathbf{x}^{(i)})$ to $\theta^{(i)}$ in an online manner.

This Bayesian inference framework marginalizes over the uncertainty of the nuisance parameters. As neural networks have been empirically shown to interpolate well between examples, we recommend choosing a diffuse prior, which makes our trained model robust to model misspecification.

Another question about utilizing machine learning models for Bayesian inference is the calibration of the posteriors, since neural networks have been empirically shown to be overconfident in their predictions. Guo et al. (2017) showed that common deep learning practices cause neural networks to poorly represent aleatoric uncertainty, namely the uncertainty due to the noise inherent in the observations. These calibration issues are a byproduct of the fixed training set regime but do not apply to simulation-on-the-fly. The softmax probabilities are calibrated for a correctly specified model under simulation-on-the-fly, since for a sufficiently expressive neural network the minimizer approximates the true posterior. However, under large model misspecification, softmax probabilities should not directly be used as posteriors since they do not properly quantify epistemic uncertainty (uncertainty in the model) as they may overconfidently classify outliers dissimilar to the training set. For recombination hotspot testing, we found that the summary statistics from the 1000 Genomes dataset (1000 Genomes Project Consortium, 2015) were similar to the summary statistics of the simulated data, so for simplicity we use the softmax probabilities as the posterior.

3.4. Statistical Properties

Our deep learning method exhibits similar asymptotic properties to those of ABC, with additional guarantees for non-observed values of $\mathbf{x}$.

In the simulation-on-the-fly setting, convergence to a global minimum implies that a sufficiently large neural network architecture represents the true posterior within ϵ-error in the following sense: for any fixed error ϵ, there exist H_0 and N_0 such that the trained neural network produces a posterior which satisfies

$$\min_{\mathbf{w}} \mathbb{E}_{\mathbf{x}} \Big[KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \;\|\; \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w}) \Big) \Big] < \epsilon, \qquad (3)$$

for all $H > H_0$ and $N > N_0$, where H is the minimum number of hidden units across all neural network layers, $\mathbf{w}$ the weights parameterizing the network, and KL the Kullback–Leibler divergence between the population risk and the neural network. Then the following proposition holds.

Proposition 1. *For all $\mathbf{x}$, $H > H_0$, and $N > N_0$ and for any fixed error $\delta > 0$,*

$$KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \;\|\; \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w}^*) \Big) < \delta \qquad (4)$$

with probability at least $1 - \frac{\epsilon}{\delta}$, where $\mathbf{w}^*$ is the minimizer of (3).

We can get stronger guarantees in the discrete setting common to population genetic data.

Corollary 1. *Under the same conditions, if $\mathbf{x}$ is discrete and $\mathbb{P}(\mathbf{x}) > 0$ for all $\mathbf{x}$, the KL divergence appearing in (4) converges to 0 as $H, N \to \infty$ uniformly for all $\mathbf{x}$.*

The proofs are given in the Appendix. Note that it is computationally infeasible to train a neural network such that $H \to \infty$. Instead, we restrict the number of units to some fixed constant H inducing a model class over learnable functions $\mathcal{F}_H$. Our training procedure in the asymptotic regime for fixed H minimizes the objective function in (3) as $N \to \infty$. Similarly, under the finite sample regime, the training procedure directly minimizes the projected population risk for the restricted model class. An important property of neural networks with a finite number of hidden units is that this restricted model class is quite large and has been empirically shown to approximate many functions well, so finite H only introduces minimal error. Furthermore, deep learning has been empirically shown to converge in only a few thousand iterations for many real-world high-dimensional datasets (Zhang et al., 2016), hence N need not approach infinity to obtain a good approximation of the posterior. We later confirm this finding experimentally. Hyperparameter optimization in neural networks can be performed by comparing the relative loss of the neural network under a variety of optimization and architecture settings. On the other hand, ABC has no such theoretical or empirical results in the finite sample regime outside of toy examples in which the likelihood can be approximated, and it has been shown empirically that the iteration complexity scales exponentially in the dimension of the summary statistics due to the curse of dimensionality.

We now show that our neural network learns statistics which are asymptotically sufficient. While several variants of sufficiency in the Bayesian context have been defined in the literature (Kolmogoroff, 1942; Furmańczyki & Niemiro, 1998) we focus on the following.

Definition 1. A statistic $T(x)$ is called *prior-dependent Bayes sufficient* if for a parameter θ and fixed prior $\pi(\theta)$, the posterior satisfies, for all θ and x,

$$\mathbb{P}_\pi(\theta \mid x) = \mathbb{P}_\pi(\theta \mid T(x)).$$

Proposition 2. *Each layer of the neural network trained via the likelihood-free framework is prior-dependent Bayes sufficient with respect to $\pi(\theta)$ as $H \to \infty$ and assuming the optimization for each H converges to the global minimum.*

This is proved in the Appendix. The sufficiency of the exchangeable feature representation ensures that no inferential accuracy has been sacrificed while reducing the data to an exchangeable feature representation. Each layer of the neural network is sufficient, allowing this representation to be used for other tasks. While this notion of sufficiency does not cover a finite architecture, it allows us to compare against the asymptotic results of ABC. More details on the properties of ABC are given in the Appendix.

Another desirable property is having unbiased uncertainty estimates, namely posterior calibration. Fearnhead & Prangle (2012) note that ABC is asymptotically calibrated as its kernel bandwidth goes to 0, but not calibrated in general. Similarly, our deep learning procedure is calibrated as the number of hidden units $H \to \infty$. While neural networks are difficult to analyze in fixed architecture settings with nonconvex loss surfaces, we empirically find that our neural network is calibrated in Section 5.

4. Population Genetics Application

The framework we established overcomes many challenges posed by population genetic inference. In this setting, each observation $\mathbf{x}$ is encoded as follows. Let $\mathbf{x}_S$ be the binary $n \times d$ allele matrix with 0 and 1 as the major and minor alleles respectively, where n is the number of individuals and d is the number of SNPs. Let $\mathbf{x}_D$ be the $n \times d$ matrix storing the distances between neighboring SNPs, so each row of $\mathbf{x}_D$ is identical and the rightmost distance is set to 0. Define $\mathbf{x}$ as the $n \times d \times 2$ tensor obtained by stacking $\mathbf{x}_S$ and $\mathbf{x}_D$. To improve the conditioning of the optimization problem, the distances are normalized such that they are on the order of $[0, 1]$. As mentioned in Section 3.1, this is an instance of exchangeably-structure data.

The standard generative model for such data is the coalescent, a stochastic process describing the distribution over genealogies relating samples from a population of individuals. The coalescent with recombination (Griffiths, 1981; Hudson, 1983) extends this model to describe the joint distribution of genealogies along the chromosome. The recombination rate between two DNA locations tunes the correlation between their corresponding genealogies. Population genetic data derived from the coalescent obeys translation invariance along a sequence conditioned on local recombination and mutation rates also obeying translation invariance. In order to take full advantage of parameter sharing, our chosen architecture is given by a convolutional neural network with tied weights for each row preceding the exchangeable layer, which is in turn followed by a fully connected neural network. We choose g as the element-wise max, and the architecture is depicted in Figure 1.

4.1. Recombination Hotspot Testing

Recombination hotspots are short regions of the genome ($\approx$ 2 kb in humans) with high recombination rate relative to the background recombination rate. To apply our framework to the hotspot detection problem, we define the overall graphical model in Figure 2. Denote w as a small window (typically < 25 kb) of the genome such that X_w is the population genetic data in that window, and X_{-w} is the rest. Similarly, let ρ_w and ρ_{-w} be the recombination map in the window and outside of the window, respectively. Let q be the the relative proportion of the sample possessing each mutation, η the population size function θ the mutation rate, and h the indicator function for whether the window defines a hotspot. While ρ_w and ρ_{-w} have a weak dependence (dashed line) on X_{-w} and X_w respectively, this dependence decreases rapidly and is ignored for simplicity. Similarly, conditioned on q, η is only weakly dependent on X_w. The

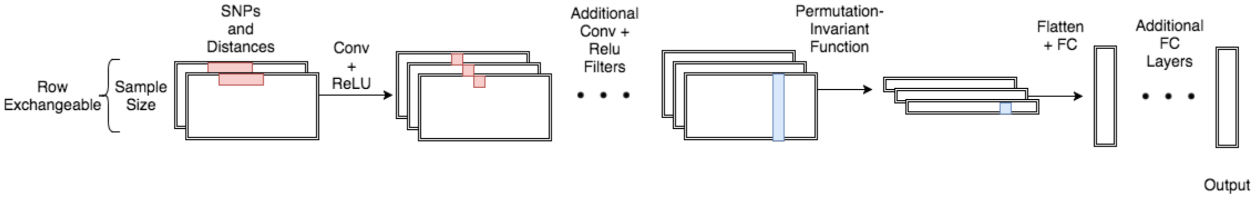

Figure 1. A cartoon schematic of the exchangeable architecture for population genetics.

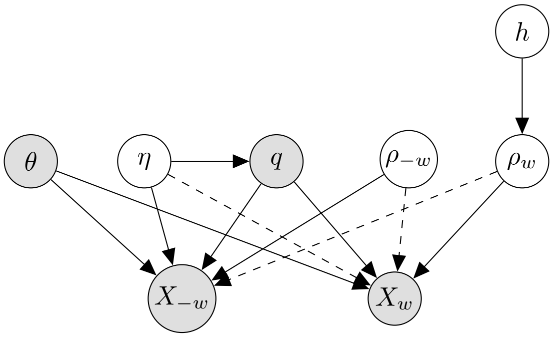

Figure 2. Graphical model of recombination hotspot inference: θ is the mutation rate, η the population size function, q the relative proportion of the sample possessing each mutation, ρ_{-w} the recombination rate function outside of the window, ρ_w the recombination rate function inside the window, h whether the window is a hotspot, X_{-w} the population genetic data outside of the window, and X_w the data inside the window. The dashed line signifies that, conditioned on q, η is weakly dependent on X_w for suitably small w, and ρ_{-w} and ρ_w are only weakly dependent on X_w and X_{-w}.

shaded nodes represent the observed variables.

We define our prior as follows. We sample the hotspot indicator variable $h \sim \text{Bernoulli}(0.5)$ and the local recombination maps $\rho_w \sim \hat{P}(\rho_w \mid h)$ from the released fine-scale recombination maps of HapMap (Gibbs et al., 2003). In addition, the demography is inferred via SMC++ (Terhorst et al., 2017) and fixed in an empirical Bayes style throughout training for simplicity. The human mutation rate is fixed to that experimentally found in Kong et al. (2012). Since SMC++ is robust to changes in any small fixed window, inferring $\hat{\eta}$ from X has minimal dependence on ρ_w.

To test for recombination hotspots, first simulate a batch of h and ρ_w from the prior, and X_w from msprime (Kelleher et al., 2016). Then, feed a batch of training examples into the network. Repeat until convergence or for a fixed number of iterations. At test time, slide along the genome to infer posteriors over h.

5. Experiments

In this section, we study the accuracy of our framework to test for recombination hotspots. As very few hotspots have been experimentally validated, we primarily evaluate our method on simulated data, with parameters set to match a human-like setting. The presence of ground truth allows us to benchmark our method and compare against LDhot. Unless otherwise specified, for all experiments we use the mutation rate, $\mu = 1.1 \times 10^{-8}$ per generation per nucleotide, convolution patch length of 5 SNPs, 32 and 64 convolution filters for the first two convolution layers, 128 hidden units for both fully connected layers, and 20-SNP length windows. The experiments comparing against LDhot used sample size $n = 64$ to construct lookup tables for LDhot quickly. All other experiments use $n = 198$, matching the size of the CEU population (i.e., Utah Residents with Northern and Western European ancestry) in the 1000 Genomes dataset. All simulations were performed using msprime (Kelleher et al., 2016). Gradient updates were performed using Adam (Kingma & Ba, 2014) with learning rate $1 \times 10^{-3} \times 0.9^{b/10000}$, b being the batch count.

5.1. Evaluation of Exchangeable Representation

We compare the behavior of an explicitly exchangeable architecture to a nonexchangeable architecture that takes 2D convolutions with varying patch heights. The accuracy under human-like population genetic parameters with varying 2D patch heights is shown in Figure 3. Since each training point is simulated on-the-fly, data augmentation is performed implicitly in the nonexchangeable version without having to explicitly permute the rows of each training point. As expected, directly encoding the permutation invariance leads to more efficient training and higher accuracy while also benefiting from a faster per-batch computation time. Furthermore, the slight accuracy decrease when increasing the patch height confirms the difficulty of learning permutation invariance as n grows. Another advantage of exchangeable architectures is the robustness to the number of individuals at test time. As shown in Figure 4, the accuracy remains robust during test time for sample sizes roughly 0.5–4× the train sample size.

5.2. Evaluation of Simulation-on-the-fly

Next, we analyze the effect of simulation-on-the-fly in comparison to the standard fixed training set. A fixed training set size of 10000 was used and run for 20000 training batches and a test set of size 5000. For a network using simulation-

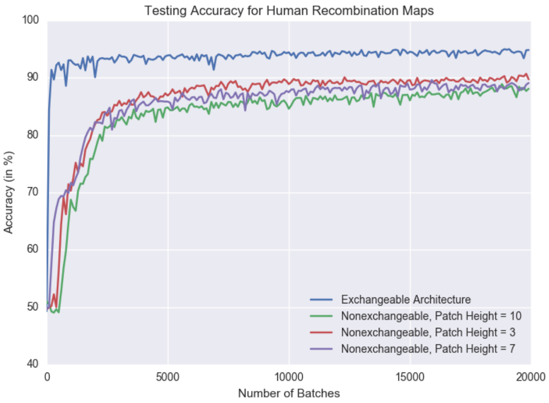

Figure 3. Accuracy comparison between exchangeable vs nonexchangeable architectures.

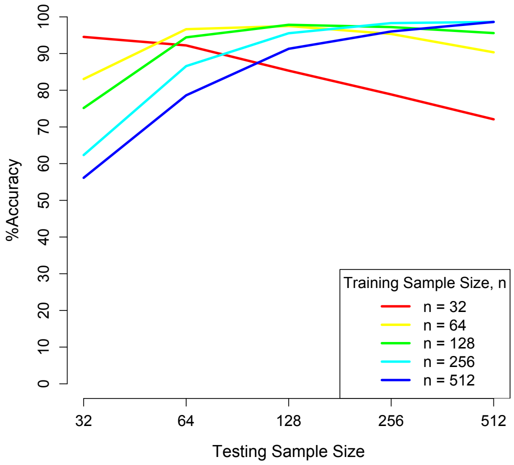

Figure 4. Performance of changing the number of individuals at test time for varying training sample sizes.

on-the-fly, 20000 training batches were run and evaluated on the same test set. The weights were initialized with a fixed random seed in both settings with 20 replicates. Figure 5 shows that the fixed training set setting has both a higher bias and higher variance than simulation-on-the-fly. The bias can be attributed to the estimation error of a fixed training set in which the empirical risk surface is not a good approximation of the population risk surface. The variance can be attributed to an increase in the number of poor quality local optima in the fixed training set case.

We next investigated posterior calibration. This gives us a measure for whether there is any bias in the uncertainty estimates output by the neural network. We evaluated the calibration of simulation-on-the-fly against using a fixed training set of 10000 datapoints. The calibration curves were generated by evaluating 25000 datapoints at test time and binning their posteriors, computing the fraction of true labels for each bin. A perfectly calibrated curve is the dashed black line shown in Figure 6. In accordance with the theory in Section 3.2, the simulation-on-the-fly is much better calibrated with an increasing number of training examples leading to a more well calibrated function. On the other hand, the fixed training procedure is poorly calibrated.

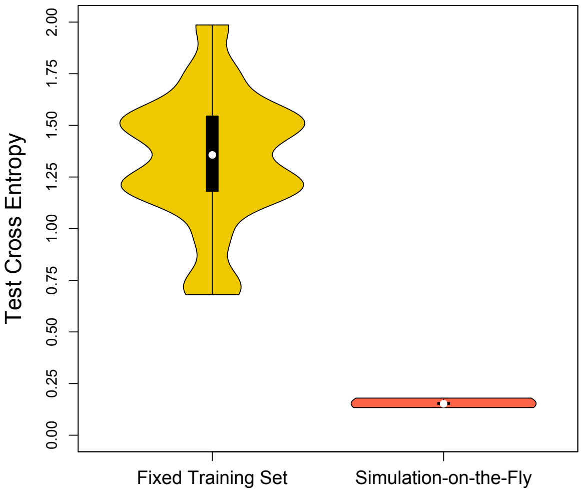

Figure 5. Comparison between the test cross entropy of a fixed training set of size 10000 and simulation-on-the-fly.

5.3. Comparison to `LDhot`

We compared our method against `LDhot` in two settings: (i) sampling empirical recombination rates from the HapMap recombination map for CEU and YRI (i.e., Yoruba in Ibadan, Nigera) (Gibbs et al., 2003) to set the background recombination rate, and then using this background to simulate a flat recombination map with 10 – 100× relative hotspot intensity, and (ii) sampling segments of the HapMap recombination map for CEU and YRI and classifying them as hotspot according to our definition, then simulating from the drawn variable map.

The ROC curves for both settings are shown in Figure 7. Under the bivariate empirical background prior regime where there is a flat background rate and flat hotspot, both methods performed quite well as shown on the top panel of Figure 7. We note that the slight performance decrease for YRI when using `LDhot` is likely due to hyperparameters that require tuning for each demography. This bivariate setting is the precise likelihood ratio test for which `LDhot` tests. However, as flat background rates and hotspots are not realistic, we sample windows from the HapMap recombination map and label them according to a more suitable hotspot definition

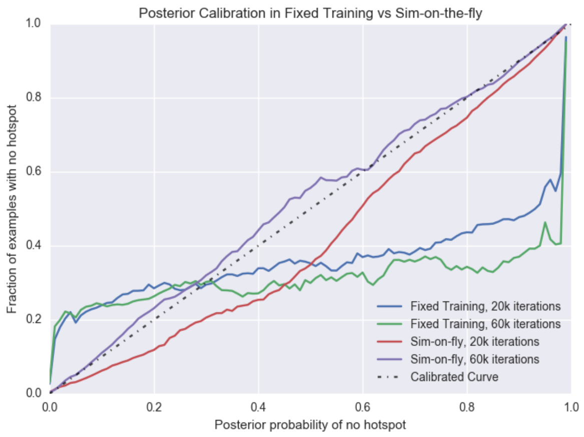

Figure 6. Posterior calibration. The black line is a perfectly calibrated curve. The red and purple lines are calibration curves for simulation-on-the-fly after 20000 and 60000 iterations, while the blue and green lines for a fixed training set of 10000 points, for 20000 and 60000 training iterations.

that ensures locality and rules out neglectable recombination spikes (the details are given in the Appendix). The bottom panel of Figure 7 uses the same hotspot definition in the training and test regimes, and is strongly favorable towards the deep learning method. Under a sensible definition of recombination hotspots and realistic recombination maps, our method still performs well while `LDhot` performs almost randomly. We believe that the true performance of `LDhot` is somewhere between the first and second settings, with performance dominated by the deep learning method. Importantly, this improvement is achieved without access to any problem-specific summary statistics.

Our approach reached 90% accuracy in fewer than 2000 iterations, taking approximately 0.5 hours on a 64 core machine with the computational bottleneck due to the `msprime` simulation (Kelleher et al., 2016). For `LDhot`, the two-locus lookup table for variable demography using the `LDpop` fast approximation (Kamm et al., 2016) took 9.5 hours on a 64 core machine (downsampling $n = 198$ from $N = 256$). The lookup table has a computational complexity of $O(N^3)$ while per-iteration training of the neural network scales as $O(n)$, allowing for much larger sample sizes.

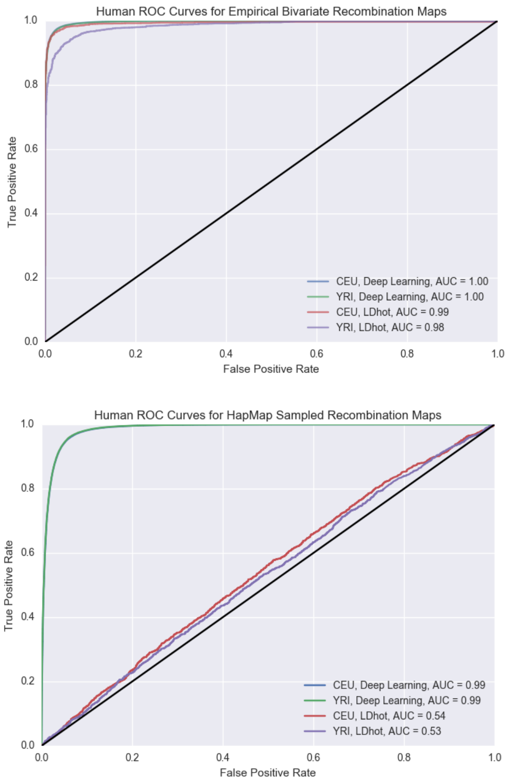

Figure 7. The black line represents a random classifier. (Top) ROC curve in the CEU and YRI setting for the deep learning and `LDhot` method. (Bottom) Windows of the HapMap recombination map drawn based on whether they matched up with our hotspot definition. The blue and green line coincide almost exactly.

6. Discussion

We developed the first likelihood-free inference method for population genetics that does not rely on handcrafted summary statistics. To achieve this, we designed a family of neural networks that learn an exchangeable representation of genotype data, which is in turn mapped to the posterior distribution over the parameter of interest. State-of-the-art accuracy was demonstrated on the challenging problem of recombination hotspot testing. Furthermore, we analyzed and developed general-purpose machine learning methods that can leverage scientific simulators to improve over preexisting likelihood-free inference schemes.

The theoretical and empirical results of simulation-on-the-fly illustrate the attractiveness of fields with model simulators as a testbed for new neural network methods. For instance, this approach allows the researcher to diagnose if regularization or convergence to poor local minima is affecting performance. We believe the simulator paradigm has a lot to offer to further understanding theoretical aspects of neural networks.

Quantifying uncertainty over a continuous parameter could be of interest in many other population genetic tasks, in which case softmax probabilities are inapplicable. Future work could adapt our method with ideas from the Bayesian neural networks literature to obtain posterior distributions over continuous parameters (Hernández-Lobato & Adams, 2015; Blundell et al., 2015; Gal & Ghahramani, 2016; Kingma et al., 2015).

ACKNOWLEDGMENTS

We thank Ben Graham for helpful discussions. This research is supported in part by an NSF Graduate Research Fellowship (JC); EPSRC grants EP/L016710/1 (VP) and EP/L018497/1 (PJ); an NIH grant R01-GM094402 (JC, JPS, SM, and YSS); and a Packard Fellowship for Science and Engineering (YSS). YSS is a Chan Zuckerberg Biohub investigator.

References

1000 Genomes Project Consortium. A global reference for human genetic variation. *Nature*, 526(7571):68–74, 2015.

Auton, A., Myers, S., and McVean, G. Identifying recombination hotspots using population genetic data. *arXiv: 1403.4264*, 2014.

Beaumont, M. A., Zhang, W., and Balding, D. J. Approximate Bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.

Blum, MGB and François, O. Non-linear regression models for Approximate Bayesian Computation. *Statistics and Computing*, 20(1):63–73, 2010.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. In *International Conference on Machine Learning*, volume 37, pp. 1613–1622, Lille, France, 2015.

Boitard, S., Rodríguez, W., Jay, F., Mona, S., and Austerlitz, F. Inferring population size history from large samples of genome-wide molecular data-an approximate bayesian computation approach. *PLoS genetics*, 12(3):e1005877, 2016.

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, 1989.

Fearnhead, P. SequenceLDhot: detecting recombination hotspots. *Bioinformatics*, 22:3061–3066, 2006.

Fearnhead, P. and Prangle, D. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(3):419–474, 2012.

Frazier, D. T., Martin, G. M., Robert, C. P., and Rousseau, J. Asymptotic properties of approximate bayesian computation. *arXiv:1607.06903*, 2016.

Furmańczyki, K. and Niemiro, W. Sufficiency in bayesian models. *Applicationes Mathematicae*, 25(1):113–120, 1998.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *International Conference on Machine Learning*, volume 48, pp. 1050–1059, New York, New York, USA, 2016.

Gibbs, R. A., Belmont, J. W., Hardenbol, P., Willis, T. D., Yu, F., Yang, H., Ch'ang, L.-Y., Huang, W., Liu, B., Shen, Y., et al. The international hapmap project. *Nature*, 426 (6968):789–796, 2003.

Griffiths, R. C. Neutral two-locus multiple allele models with recombination. *Theoretical Population Biology*, 19 (2):169–186, 1981.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. *arXiv:1706.04599*, 2017.

Guttenberg, N., Virgo, N., Witkowski, O., Aoki, H., and Kanai, R. Permutation-equivariant neural networks applied to dynamics prediction. *arXiv:1612.04530*, 2016.

Hernández-Lobato, J. M. and Adams, R. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pp. 1861–1869, 2015.

Hey, J. What's so hot about recombination hotspots? *PLoS Biol*, 2(6):e190, 2004.

Hudson, R. R. Properties of a neutral allele model with intragenic recombination. *Theoretical population biology*, 23(2):183–201, 1983.

Hudson, R. R. Two-locus sampling distributions and their application. *Genetics*, 159(4):1805–1817, 2001.

Jiang, B., Wu, T.-y., Zheng, C., and Wong, W.H. Learning summary statistic for approximate Bayesian computation via deep neural network. *arXiv:1510.02175*, 2015.

Kamm, J. A., Spence, J. P., Chan, J., and Song, Y. S. Two-locus likelihoods under variable population size and fine-scale recombination rate estimation. *Genetics*, 203(3): 1381–1399, 2016.

Kelleher, J., Etheridge, A. M., and McVean, G. Efficient coalescent simulation and genealogical analysis for large sample sizes. *PLoS computational biology*, 12(5): e1004842, 2016.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. In *Neural Information Processing Systems*, pp. 2575–2583, 2015.

Kingman, J. F. C. The coalescent. *Stochastic processes and their applications*, 13(3):235–248, 1982.

Kolmogoroff, A. Sur l'éstimation statistique des paramétres be la loi de gauss. *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya*, 6(1):3–32, 1942.

Kong, A., Frigge, M. L., Masson, G., Besenbacher, S., Sulem, P., Magnusson, G., Gudjonsson, S. A., Sigurdsson, A., Jonasdottir, A., Jonasdottir, A., et al. Rate of de novo mutations and the importance of father's age to disease risk. *Nature*, 488(7412):471–475, 2012.

Li, J., Zhang, M. Q., and Zhang, X. A new method for detecting human recombination hotspots and its applications to the hapmap encode data. *The American Journal of Human Genetics*, 79(4):628–639, 2006.

McVean, G. A. T., Myers, S. R., Hunt, S., Deloukas, P., Bentley, D. R., and Donnelly, P. The fine-scale structure of recombination rate variation in the human genome. *Science*, 304:581–584, 2004.

Papamakarios, G. and Murray, I. Fast ϵ-free inference of simulation models with Bayesian conditional density estimation. *arXiv:1605.06376*, 2016.

Pavlidis, P., Jensen, J. D., and Stephan, W. Searching for footprints of positive selection in whole-genome snp data from nonequilibrium populations. *Genetics*, 185(3):907–922, 2010.

Petes, T. D. Meiotic recombination hot spots and cold spots. *Nature Reviews Genetics*, 2(5):360–369, 2001.

Ravanbakhsh, S., Schneider, J., and Poczos, B. Deep learning with sets and point clouds. *arXiv:1611.04500*, 2016.

Schrider, D. R. and Kern, A. D. Inferring selective constraint from population genomic data suggests recent regulatory turnover in the human brain. *Genome biology and evolution*, 7(12):3511–3528, 2015.

Sheehan, S. and Song, Y. S. Deep learning for population genetic inference. *PLoS Computational Biology*, 12(3): e1004845, 2016.

Shivaswamy, P. K. and Jebara, T. Permutation invariant svms. In *International Conference on Machine Learning*, pp. 817–824, 2006.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv:1712.01815*, 2017.

Sousa, V. C., Fritz, M., Beaumont, M. A., and Chikhi, L. Approximate Bayesian computation without summary statistics: The case of admixture. *Genetics*, 181(4):1507–1519, 2009.

Terhorst, J., Kamm, J. A., and Song, Y. S. Robust and scalable inference of population history from hundreds of unphased whole genomes. *Nature genetics*, 49(2): 303–309, 2017.

Wall, J. D. and Stevison, L. S. Detecting recombination hotspots from patterns of linkage disequilibrium. *G3: Genes, Genomes, Genetics*, 2016.

Wang, Y. and Rannala, B. Population genomic inference of recombination rates and hotspots. *Proceedings of the National Academy of Sciences*, 106(15):6215–6219, 2009.

Wegmann, D., Leuenberger, C., and Excoffier, L. Efficient Approximate Bayesian Computation coupled with Markov chain Monte Carlo without likelihood. *Genetics*, 182(4):1207–1218, 2009.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R., and Smola, A. Deep sets. *Neural Information Processing Systems*, 2017.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *arXiv:1611.03530*, 2016.

Appendix

A. Statistical Properties of ABC

Understanding the statistical properties of ABC enables us to highlight the theoretical benefits of our approach. Variants of ABC are among the most widely-used likelihood-free inference techniques in the scientific literature. ABC simulates N draws of the parameter $\theta^{(i)} \sim \pi(\theta)$ and data $\mathbf{x}^{(i)} \sim \mathbb{P}(\mathbf{x} \mid \theta^{(i)})$ for $i = 1, \ldots, N$, then approximates the posterior conditioned on the observed summary statistics $s_{obs} = S(\mathbf{x}_{obs})$ by

$$\mathbb{P}_{ABC}(\theta \mid s_{obs}) \propto \sum_{i=1}^{N} \mathbb{P}(\theta^{(i)})\mathbb{P}(\mathbf{x}^{(i)} \mid \theta^{(i)}) \\ K\left(\frac{\|S(\mathbf{x}^{(i)}) - s_{obs}\|}{u}\right),$$

where $S : \mathcal{X} \to \mathcal{V}$ is a summary statistic of the data and $K : \mathcal{V} \to \mathbb{R}$ is a density kernel that integrates to 1 with bandwidth $u > 0$. Denote $\mathbf{a}$ as the N-dimensional vector corresponding to the kernel weight $K(\cdot)$ for each simulated parameter $\theta^{(i)}$. A common choice of K is the uniform kernel, though many variants exist. Intuitively, ABC can be interpreted as locally smoothing the empirical likelihood estimates of points near the observed data in the summary statistic space.

The ABC posterior asymptotically converges to the true posterior conditioned on the observed data $\mathbf{x}_{obs}$ for sample size n under suitable regularity conditions, so that

$$\lim_{u \to 0, N \to \infty} KL\Big(\mathbb{P}(\theta \mid \mathbf{x}_{obs}) \parallel \mathbb{P}_{ABC}^{(N)}(\theta \mid S(\mathbf{x}_{obs}))\Big) = 0,$$

for any choice of sufficient statistic S. See Frazier et al. (2016) for a formal treatment. However, in the finite-sample regime for fixed N, the hyperparameters of the ABC algorithm should be chosen such that

$$\inf_{u,S} \mathbb{E}\Big[KL\Big(\mathbb{P}(\theta \mid \mathbf{x}_{obs}) \parallel \mathbb{P}_{ABC}^{(N)}(\theta \mid S(\mathbf{x}_{obs}))\Big)\Big] \qquad (5)$$

is minimized. Based on this formulation, the computational and statistical tradeoffs based on the choice of u and S as a function of the computational budget N are made explicit. Unfortunately, hyperparameter optimization on u and S cannot be performed since the expected KL in (5) cannot be compared without access to the true posterior. Instead, practitioners often optimize a surrogate objective similar to

$$\inf_{u,S} J(u, S)\mathbb{1}\Big(\mathbb{E}\big(m(\mathbf{a})\big) > \tau\Big),$$

where τ is a sampling threshold and $m(\mathbf{a})$ is a user-defined function of the kernel weights, such as number of accepted samples or effective sample size. $J(u, S)$ is a user-defined positive function that is decreasing in u. $J(u, S)$ also satisfies $J(u, S_1) \leq J(u, S_2)$ for all S_1, S_2 where $\mathcal{V}_1 \subseteq \mathcal{V}_2$. Note that ABC practitioners are typically not explicit about their surrogate objective function when tuning ABC; however, for clarity, we specify the general surrogate objective above, and remark that many modifications do not affect the underlying tradeoffs stated below.

Intuitively, the surrogate objective function encourages practitioners to choose values of u and S such that u is close to 0, and S is close to sufficient while generating enough large kernel weights within the computational budget to obtain an empirical posterior. This procedure ignores the posterior completely and could result in arbitrarily poor approximations of the posterior. Poor approximations of the posterior can result for many reasons, including lack of information in S, large u, insufficient number of samples generated, insufficient computational budget, or incorrect choice of kernel K or norm $\|\cdot\|$ for the geometry of the posterior. Furthermore, this procedure must be re-run for each new dataset $\mathbf{x}_{obs}$ allowing for a smaller computational budget N when dealing with multiple datasets. There are no guarantees that the previous values of S and u remain good choices for a new dataset since the parameters depend on $\mathbf{x}_{obs}$.

B. Statistical Properties of Our Method: Proofs

Proof of Proposition 1 By the Universal Approximation Theorem and the interpretation of simulation-on-the-fly as minimizing the expected KL divergence between the population risk and the neural network, the training procedure minimizes the objective function for every $\epsilon > 0$, $H > H_0$, and $N > N_0$,

$$\min_{\mathbf{w}} \mathbb{E}_{\mathbf{x}}\Big[KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \parallel \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w})\Big)\Big] < \epsilon.$$

Let $\mathbf{w}^*$ be a minimizer of the above expectation. By Markov's inequality, we get for every $\mathbf{x}$ and $\delta > 0$ such that for all $H > H_0$ and $N > N_0$

$$KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \parallel \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w}^*)\Big) < \delta$$

with probability at least $1 - \frac{\epsilon}{\delta}$. □

Proof of Corollary 1 As above, we have

$$\min_{\mathbf{w}} \mathbb{E}_{\mathbf{x}}\Big[KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \parallel \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w})\Big)\Big] < \epsilon,$$

for all $\epsilon > 0$, $H > H_0$, and $N > N_0$. Furthermore, for all $\mathbf{x}$, the KL is bounded at the minimizer since $\mathbb{P}(\mathbf{x}) > 0$ for all $\mathbf{x}$ resulting in the following bound

$$KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \parallel \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w}^*)\Big) < \max_{\mathbf{x}} \frac{\epsilon}{\mathbb{P}(\mathbf{x})}$$

independent of $\mathbf{x}$. Thus, the training procedure results in a function mapping that uniformly converges to the posterior $\mathbb{P}(\theta \mid \mathbf{x})$. □

Proof of Proposition 2 For each H and N, the neural network is trained to find the $\mathbf{w}$ that minimizes

$$\mathbb{E}_{\mathbf{x}}\Big[KL\Big(\mathbb{P}(\theta \mid \mathbf{x}) \parallel \mathbb{P}_{DL}^{(N)}(\theta \mid \mathbf{x}; \mathbf{w})\Big)\Big].$$

As $H \to \infty$ and $N \to \infty$, this quantity converges to a global minimum. By the Universal Approximation Theorem this is achieved when the function learned by the neural network is the posterior $P(\theta \mid \mathbf{x})$. Thus, each layer of the neural network can be viewed as a statistic $T(\mathbf{x})$ of the input data $\mathbf{x}$. In other words each layer of the trained neural network is prior-dependent Bayes sufficient, $\mathbb{P}(\theta \mid \mathbf{x}) = \mathbb{P}(\theta \mid T(\mathbf{x}))$ for our chosen prior $\pi(\theta)$. □

C. Recombination Hotspot Details

Recombination hotspots are short regions of the genome with high recombination rate relative to the background. In order to develop accurate methodology, a precise mathematical definition of a hotspot needs to be specified in accordance with the signatures of biological interest. We use the following.

Definition 2 (Recombination Hotspot)**.** Let a window over the genome be subdivided into three subwindows $w = (w_l, w_h, w_r)$ with physical distances α_l, α_h, and α_r, respectively, where $w_l, w_h, w_r \in \mathcal{G}$ where $\mathcal{G}$ is the space over all possible subwindows of the genome. Let a mean recombination map $R : \mathcal{G} \to \mathbb{R}_+$ be a function that maps from a subwindow of the genome to the mean recombination rate per base pair in the subwindow. A recombination hotspot for a given mean recombination map R is a window w which satisfies the following properties:

1. Elevated local recombination rate: $R(w_h) > k \cdot \max\big(R(w_l), R(w_r)\big)$
2. Large absolute recombination rate: $R(w_h) > k\tilde{r}$

where $\tilde{r}$ is the median (at a per base pair level) genome-wide recombination rate, and k is the relative hotspot intensity.

The first property is necessary to enforce the locality of hotspots and rule out large regions of high recombination rate, which are typically not considered hotspots by biologists. The second property rules out regions of minuscule background recombination rate in which sharp relative spikes in recombination still remain too small to be biologically interesting. The median is chosen here to be robust to the right skew of the distribution of recombination rates. Typically, for the human genome we use $\alpha_l = \alpha_r = 13$ kb, $\alpha_h = 2$ kb, and $k = 10$ based on experimental findings.

The most widely-used technique for recombination hotspot testing is LDhot as described in (Auton et al., 2014). The method performs a generalized composite likelihood ratio test using the two-locus composite likelihood based on (Hudson, 2001) and (McVean et al., 2004). The composite two-locus likelihood approximates the joint likelihood of a window of SNPs w by a product of pairwise likelihoods

$$CL(\rho \mid \mathbf{x}) = \prod_{1 \le |i-j| \le z} L(\rho_{ij} \mid \mathbf{x}_{ij}),$$

where X_{ij} denotes the data restricted only to SNPs i and j, and ρ_{ij} denotes the recombination rate between those sites. Only SNPs within some distance, say $z = 50$, are considered.

Two-locus likelihoods are computed via an importance sampling scheme under a constant demography ($\eta = 1$) as in (McVean et al., 2004). The likelihood ratio test uses a null model of a constant recombination rate and an alternative model of a differing recombination rate in the center of the window under consideration:

$$\Lambda = -2\log\left(\frac{\sup_{\rho_{\text{hot}},\rho_{\text{bg}}} CL(\rho_{\text{hot}}, \rho_{\text{bg}} \mid X)}{\sup_{\rho_{\text{const}}} CL(\rho_{\text{const}} \mid X)}\right).$$

The two-locus likelihood can only be applied to single panmictic populations with constant demography, constant mutation rate, and without natural selection. Furthermore, the two-locus likelihood is an uncalibrated approximation of the true joint likelihood. In addition, the experiments in Wall & Stevison (2016) and Auton et al. (2014) do not demonstrate the efficacy of LDhot against a realistic variable background recombination rate as its null hypothesis leads to a comparison against a biologically unrealistic flat background rate. In order to fairly compare our likelihood-free approach against the composite likelihood-based method in realistic human settings, we extended the LDhot methodology to apply to a piecewise constant demography using two-locus likelihoods computed by the software LDpop (Kamm et al., 2016). Unlike the method described in Wall & Stevison (2016), our implementation of LDhot uses windows defined in terms of SNPs rather than physical distance in order to measure accuracy via ROC curves, since the likelihood ratio test is a function of number of SNPs. Note that computing the approximate two-locus likelihoods for a grid of recombination values is at least $O(n^3)$, which could be prohibitive for large sample sizes.

D. Additional Experiments

Regularization The simulation-on-the-fly paradigm obviates the need for modern regularization techniques such as dropout. This is due to the fact that there is no notion of overfitting since each training point is used only once and a large number of examples are drawn from the population distribution. As shown in Figure 8, dropout does not help improve the accuracy of our method and, in fact, leads to a minor decrease in performance. As expected, directly optimizing the population risk minimizer circumvents the problem of overfitting.

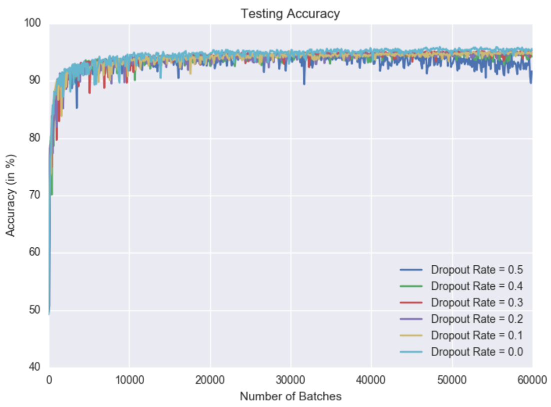

Figure 8. A comparison of different dropout rates. Dropout has a minimal (or slightly negative) effect on test accuracy under the simulation-on-the-fly regime.

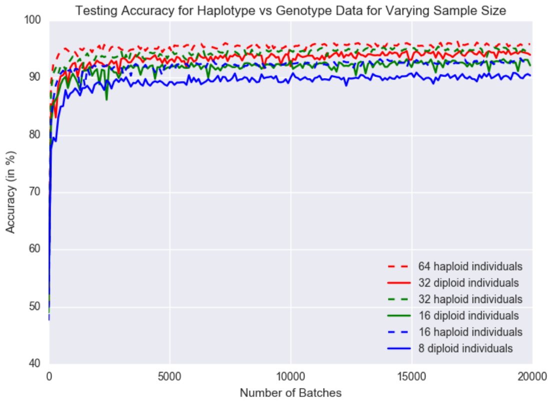

Figure 9. Accuracy comparison between haplotype and genotype data.

Phasing Deconvolving two haplotypes from genotype data is a challenging statistical problem, commonly referred to as phasing. Phasing without a high quality reference panel introduces significant bias into downstream inference. Our approach can flexibly perform inference directly on haplotype or genotype data, the latter being a challenge for model-based approaches. Inference directly on genotype data allows us to implicitly integrate over possible phasings, reducing the bias introduced by fixing the data to a single phasing. In the case of recombination hotspots, we have found only a minor decrease in accuracy for small sample sizes corresponding to the reduction in statistical signal when inference is performed on genotype data. We quantified the effect of having accurately phased data in comparison to genotype data. Specifically, inference was run by simulating haplotype data and randomly pairing them to construct genotype data such that the height of the genotype image is half that of the haplotype image. We ran the experiment for $n = 16, 32, 64$ as shown in Figure 9 and found that the our method is robust, remaining highly accurate for unphased data.

Missing Data Biological data typically contain significant amounts of missing data. The missingness results from a number of factors such as repetitive regions of the chromosome which are difficult to map, or low read coverage. Fortunately, haplotype data in population genetics is mostly missing completely at random; that is, the locations of missingness are independent of the data values. However, there is a strong correlation structure between the missingness of spatially close SNPs. To improve the robustness of our methods to missing data, we sample the missingness patterns from empirical data during training time.